PQC-FHE Integration Platform

Technical Report v2.3.4

Post-Quantum Cryptography + Fully Homomorphic Encryption

Version	2.3.4
Release Date	2025-12-30
PQC Standards	FIPS 203 (ML-KEM), FIPS 204 (ML-DSA)
FHE Scheme	CKKS (DESILO Implementation)
License	MIT

Table of Contents

1. Executive Summary

The PQC-FHE Integration Platform provides a production-ready framework combining Post-Quantum Cryptography (PQC) with Fully Homomorphic Encryption (FHE) for enterprise security applications. This platform addresses the emerging threat of quantum computers to current cryptographic systems while enabling privacy-preserving computation on sensitive data.

Key Features

* NIST-standardized PQC algorithms (FIPS 203, FIPS 204)
* CKKS-based homomorphic encryption via DESILO FHE
* Real-time data integration from verified public sources
* REST API with comprehensive Swagger documentation
* Interactive Web UI with live demonstrations
* GPU acceleration support (CUDA 12.x/13.x)

v2.3.4 Updates (2025-12-30)

Version 2.3.4 introduces enhanced live data fetching capabilities with robust fallback mechanisms, fixes for numpy array handling in FHE demo endpoints, and improved multi-platform installation documentation.

2. Market Context and Business Value

The Quantum Threat

Quantum computers pose an existential threat to current public-key cryptography. Shor's algorithm can break RSA-2048 in polynomial time, and Grover's algorithm reduces symmetric key security by half. NIST estimates cryptographically-relevant quantum computers may emerge within 10-15 years, necessitating immediate migration to quantum-resistant alternatives.

Market Opportunity

Segment	2024	2034	CAGR
PQC Solutions	$302M	$30B	58%
FHE Market	$200M	$2.5B	28%
Quantum Security	$1.2B	$15B	29%

Source: Markets and Markets, Gartner (2024)

3. System Architecture

Component Overview

Layer	Component	Technology
Presentation	Web UI	React + Tailwind CSS
API	REST Server	FastAPI + Swagger UI
Cryptography	PQC Manager	liboqs-python (ML-KEM, ML-DSA)
Cryptography	FHE Engine	DESILO FHE (CKKS)
Data	Live Fetcher	VitalDB, yfinance, Ethereum RPC
Infrastructure	GPU Accel.	CUDA 12.x/13.x + cuQuantum

Data Flow

1. Client sends request to REST API (FastAPI server on port 8000)
2. API validates input and routes to appropriate handler
3. For PQC operations: liboqs-python performs key generation/signing
4. For FHE operations: DESILO engine encrypts/computes/decrypts
5. Live data fetcher retrieves external data with automatic fallback
6. Response returned to client with full audit trail

4. Post-Quantum Cryptography Implementation

NIST Standards Compliance

This platform implements NIST's finalized post-quantum cryptography standards: FIPS 203 (ML-KEM) for key encapsulation and FIPS 204 (ML-DSA) for digital signatures. These standards were published on August 13, 2024 and represent the foundation of quantum-resistant cryptography.

Key Encapsulation Mechanisms (FIPS 203)

Algorithm	Level	Public Key	Ciphertext	Use Case
ML-KEM-512	1	800 B	768 B	IoT/Embedded
ML-KEM-768	3	1,184 B	1,088 B	General Purpose
ML-KEM-1024	5	1,568 B	1,568 B	High Security

Digital Signature Algorithms (FIPS 204)

Algorithm	Level	Public Key	Signature	Speed
ML-DSA-44	2	1,312 B	2,420 B	Fastest
ML-DSA-65	3	1,952 B	3,309 B	Balanced
ML-DSA-87	5	2,592 B	4,627 B	Maximum

5. Fully Homomorphic Encryption Implementation

CKKS Scheme Overview

The platform uses the CKKS (Cheon-Kim-Kim-Song) homomorphic encryption scheme, which supports approximate arithmetic on encrypted real numbers. CKKS is ideal for machine learning and statistical analysis on encrypted data [4].

Key Properties:

* Supports addition and multiplication on encrypted floating-point numbers
* Slot packing allows SIMD operations on thousands of values simultaneously
* Rescaling maintains precision across multiplicative depth
* Bootstrapping enables unlimited computation depth

DESILO FHE Configuration

Parameter	Value	Description
poly_degree	16,384	Polynomial ring dimension (N)
coeff_mod_bit_sizes	[60,40,40,40,60]	Coefficient modulus chain
scale	2^40	Encoding scale for precision
max_mult_depth	4	Maximum multiplicative depth
slot_count	8,192	Number of plaintext slots

6. Live Data Sources

Version 2.3.4 provides robust live data fetching with automatic fallback to embedded sample data. This ensures demonstrations work reliably while showcasing real-world data integration capabilities.

Healthcare: VitalDB [6]

Property	Value
Dataset	VitalDB Open Dataset
Method	vitaldb Python library
Data Type	Surgical patient vital signs (BP, HR, SpO2)
Sample Size	6,388 surgical cases
DOI	10.1038/s41597-022-01411-5
License	CC BY-NC-SA 4.0

Finance: Yahoo Finance [8]

Property	Value
Method	yfinance Python library
Data Type	Real-time stock prices, market cap
Symbols	AAPL, MSFT, GOOGL, AMZN, NVDA, META, TSLA, JPM
License	Yahoo Finance Terms of Service

IoT: UCI Machine Learning Repository [7]

Property	Value
Dataset	Individual Household Electric Power Consumption
Data Type	Smart meter power readings
DOI	10.24432/C52G6F
License	CC BY 4.0

Blockchain: Ethereum RPC [9]

Priority	Endpoint	Provider
1	rpc.ankr.com/eth	Ankr (Primary)
2	ethereum-rpc.publicnode.com	PublicNode
3	cloudflare-eth.com	Cloudflare
4	eth.drpc.org	DRPC
5	1rpc.io/eth	1RPC

Note: All endpoints are free and require no API key.

7. Enterprise Use Cases

Healthcare: HIPAA-Compliant Analytics

Hospitals can analyze patient vital signs without exposing Protected Health Information (PHI). FHE enables computation on encrypted blood pressure readings to identify hypertension trends across populations while maintaining full HIPAA compliance. Clinical interpretation follows AHA Guidelines [12].

Finance: Confidential Portfolio Analysis

Investment firms can perform growth projections on encrypted portfolio values. Client holdings remain confidential even during third-party analysis, enabling secure outsourcing of financial computations.

IoT: Secure Smart Grid Analytics

Utility companies can aggregate encrypted smart meter readings for demand forecasting without accessing individual household consumption patterns, preserving consumer privacy while enabling grid optimization.

Blockchain: Quantum-Resistant Transactions

Cryptocurrency platforms can migrate from ECDSA to ML-DSA signatures, protecting transaction integrity against future quantum attacks. This platform demonstrates the migration path with side-by-side comparison of signature sizes.

8. API Reference

Endpoint Summary

Endpoint	Method	Description
`/health`	GET	Health check
`/pqc/algorithms`	GET	List PQC algorithms
`/pqc/kem/keypair`	POST	Generate KEM keypair
`/pqc/kem/encapsulate`	POST	Encapsulate secret
`/pqc/kem/decapsulate`	POST	Decapsulate secret
`/pqc/sig/keypair`	POST	Generate SIG keypair
`/pqc/sig/sign`	POST	Sign message
`/pqc/sig/verify`	POST	Verify signature
`/fhe/encrypt`	POST	Encrypt data
`/fhe/decrypt`	POST	Decrypt ciphertext
`/fhe/add`	POST	Add ciphertexts
`/fhe/multiply`	POST	Multiply by scalar
`/enterprise/healthcare`	GET	Healthcare data
`/enterprise/finance`	GET	Finance data
`/enterprise/iot`	GET	IoT data
`/enterprise/blockchain`	GET	Blockchain data

Interactive documentation available at http://localhost:8000/docs (Swagger UI)

9. Installation Guide (Multi-Platform)

System Requirements

Component	Minimum	Recommended
Python	3.9+	3.11+
RAM	8 GB	32 GB
Storage	1 GB	5 GB
GPU (optional)	CUDA 11.x	CUDA 12.x/13.x

Build Dependencies by Platform

Debian/Ubuntu:

```
sudo apt update
sudo apt install -y cmake gcc g++ libssl-dev python3-dev git
```

Fedora/RHEL/CentOS:

```
sudo dnf install -y cmake gcc gcc-c++ openssl-devel python3-devel git
```

Arch Linux:

```
sudo pacman -S cmake gcc openssl python git
```

macOS (Homebrew):

```
brew install cmake openssl@3 git
```

Installing liboqs-python [10]

liboqs-python is NOT available via pip. It must be built from source. The recommended method auto-downloads and builds liboqs at runtime:

Option A: Automatic Build (Recommended)

```
git clone --depth=1 https://github.com/open-quantum-safe/liboqs-python
cd liboqs-python
pip install .
cd ..
```

Option B: Manual Build

```
# Build liboqs C library
git clone --depth=1 https://github.com/open-quantum-safe/liboqs
cmake -S liboqs -B liboqs/build -DBUILD_SHARED_LIBS=ON
cmake --build liboqs/build --parallel 8
sudo cmake --build liboqs/build --target install

# Set library path (Linux)
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib

# Install Python wrapper
git clone --depth=1 https://github.com/open-quantum-safe/liboqs-python
cd liboqs-python && pip install . && cd ..
```

Installing DESILO FHE [5]

```
# CPU mode
pip install desilofhe

# GPU mode (choose based on CUDA version)
pip install desilofhe-cu130  # CUDA 13.0
pip install desilofhe-cu124  # CUDA 12.4
pip install desilofhe-cu121  # CUDA 12.1
```

Complete Quick Start

```
# 1. Extract package
unzip pqc_fhe_portfolio_v2.3.4_final.zip
cd pqc_fhe_v2.3.0
pip install -r requirements.txt

# 2. Install liboqs-python
git clone --depth=1 https://github.com/open-quantum-safe/liboqs-python
cd liboqs-python && pip install . && cd ..

# 3. Install DESILO FHE
pip install desilofhe

# 4. Optional: Live data libraries
pip install yfinance vitaldb

# 5. Start server
python -m uvicorn api.server:app --host 0.0.0.0 --port 8000

# 6. Access Web UI
# Open http://localhost:8000/ui
```

10. Security Analysis

NIST Security Levels

NIST defines five security levels based on classical and quantum attack complexity. This platform supports Levels 1, 2, 3, and 5:

Level	Classical Equivalent	Algorithms
1	AES-128	ML-KEM-512
2	SHA-256	ML-DSA-44
3	AES-192	ML-KEM-768, ML-DSA-65
5	AES-256	ML-KEM-1024, ML-DSA-87

FHE Security

CKKS security is based on the Ring Learning With Errors (RLWE) problem, which is believed to be hard for both classical and quantum computers. The configured parameters provide at least 128-bit security against known attacks.

11. Future Roadmap

Version	Timeline	Features
v2.4.0	Q1 2025	SLH-DSA (SPHINCS+) hash-based signatures
v2.5.0	Q2 2025	Hybrid classical/PQC mode
v2.6.0	Q3 2025	Multi-party computation (MPC) integration
v3.0.0	Q4 2025	FIPS validation and enterprise hardening

Appendix A: Algorithm Parameters

ML-KEM Parameters (FIPS 203)

Parameter	ML-KEM-512	ML-KEM-768	ML-KEM-1024
n (dimension)	256	256	256
k	2	3	4
eta_1	3	2	2
eta_2	2	2	2
d_u	10	10	11
d_v	4	4	5

ML-DSA Parameters (FIPS 204)

Parameter	ML-DSA-44	ML-DSA-65	ML-DSA-87
q (modulus)	8,380,417	8,380,417	8,380,417
k	4	6	8
l	4	5	7
eta	2	4	2
tau	39	49	60

Appendix B: Performance Benchmarks

PQC Operations (Intel i7-12700H)

Operation	ML-KEM-768	ML-DSA-65
Key Generation	0.03 ms	0.08 ms
Encap/Sign	0.04 ms	0.18 ms
Decap/Verify	0.04 ms	0.06 ms

FHE Operations

Operation	CPU	GPU (RTX 4090)
Key Generation	2.5 s	0.8 s
Encrypt (8192 slots)	15 ms	3 ms
Add	0.5 ms	0.1 ms
Multiply (scalar)	2 ms	0.3 ms
Multiply (ct*ct)	50 ms	8 ms
Decrypt	10 ms	2 ms

References

[1] NIST. FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard. National Institute of Standards and Technology, August 2024. https://csrc.nist.gov/pubs/fips/203/final

[2] NIST. FIPS 204: Module-Lattice-Based Digital Signature Standard. National Institute of Standards and Technology, August 2024. https://csrc.nist.gov/pubs/fips/204/final

[3] NIST. FIPS 205: Stateless Hash-Based Digital Signature Standard. National Institute of Standards and Technology, August 2024. https://csrc.nist.gov/pubs/fips/205/final

[4] Cheon JH, Kim A, Kim M, Song Y. Homomorphic Encryption for Arithmetic of Approximate Numbers. ASIACRYPT 2017. DOI: 10.1007/978-3-319-70694-8_15

[5] DESILO. DESILO FHE Library Documentation. https://fhe.desilo.dev/latest/

[6] Lee HC, Park Y, Yoon SB, et al. VitalDB, a high-fidelity multi-parameter vital signs database in surgical patients. Scientific Data 9, 279 (2022). DOI: 10.1038/s41597-022-01411-5

[7] Hebrail G, Berard A. Individual Household Electric Power Consumption Data Set. UCI Machine Learning Repository. DOI: 10.24432/C52G6F

[8] Aroussi R. yfinance: Download market data from Yahoo! Finance API. https://github.com/ranaroussi/yfinance. License: Apache 2.0

[9] Ethereum Foundation. Ethereum JSON-RPC API. https://ethereum.org/developers/docs/apis/json-rpc/

[10] Open Quantum Safe Project. liboqs-python: Python 3 wrapper for liboqs. https://github.com/open-quantum-safe/liboqs-python. License: MIT

[11] Ramirez S, et al. FastAPI: Modern, fast web framework for building APIs. https://fastapi.tiangolo.com/. License: MIT

[12] American Heart Association. Understanding Blood Pressure Readings. https://www.heart.org/en/health-topics/high-blood-pressure/understanding-blood-pressure-readings

---

Generated: 2025-12-30

PQC-FHE Integration Platform v2.3.4