BIG QUAKE

Basic Information

- **Family Type:** Code-based
- **Purpose:** Key Encapsulation
- **NIST Security Level:** Levels 1, 3, and 5 (depending on the parameter selection)

Technical Overview

- **Mathematical Foundation:** Based on quasi-cyclic binary Goppa codes, which are an extension of classical Goppa codes, reducing key size by introducing quasi-cyclic properties.
- **Key Components:**
    - **Public Key Size:** Varies, examples include 103,896 bytes (for security level 1) and 253,896 bytes (for other levels).
    - **Private Key Size:** Not explicitly mentioned, derived from support and Goppa polynomial parameters.
    - **Ciphertext Size:** Includes encapsulated keys, values vary based on parameters.

Performance Characteristics

- **Speed:**
    - **Key Generation:** Not directly specified but involves generating parity-check matrices and Goppa polynomials.
    - **Encapsulation:** Efficient due to quasi-cyclic nature.
    - **Decapsulation:** Polynomial-time decryption leveraging Goppa polynomial structure.
- **Memory Requirements:** Varies based on m (extension degree) and other parameters, optimized for embedded systems with restricted computing resources.

Security Analysis

- **Classical Security:** Based on the difficulty of decoding generic linear codes and distinguishing Goppa codes from random codes.
- **Quantum Security:** Provides resistance against quantum attacks by leveraging the quasi-cyclic structure and associated decoding problems.
- **Known Attack Vectors:**
    - Algebraic attacks exploiting quasi-cyclic properties.
    - Key recovery attacks, including brute force on Goppa polynomials.
    - Message recovery attacks using decoding algorithms.

Implementation Considerations

- **Hardware Requirements:** Light-weight scheme suitable for embedded systems.
- **Software Complexity:** Simplified due to quasi-cyclic structure.
- **Integration Challenges:** May involve ensuring uniform randomness in key generation and parameter tuning.

Advantages and Limitations

- **Pros:**
    - Reduces key size compared to classical Goppa codes.
    - Maintains high security against both classical and quantum attacks.
    - Suitable for embedded systems.
- **Cons:**
    - Public key size remains relatively large (e.g., over 100 KB).
    - Requires careful parameter selection to avoid vulnerabilities from quasi-cyclic attacks.

Standardization Status

- **NIST Round:** BIG QUAKE does not appear in the final Round 3 or 4 submissions based on the NIST list provided. It is evaluated as a proposal but not as a finalist or alternate candidate.
- **Other Standards:** Focused on addressing NIST's requirements for quantum-resistant cryptography.

BIKE

Basic Information

Algorithm Name: BIKE (Bit Flipping Key Encapsulation)
Family Type: Code-based
Purpose: Key Encapsulation
NIST Security Level: 1, 3, and 5

Technical Overview

Mathematical Foundation: QC-MDPC (Quasi-Cyclic Moderate Density Parity Check) codes
Key Components:

- **Public Key Size:**
    - Level 1: 20,326 bits

- Level 3: 43,786 bits
    - Level 5: 65,498 bits
- **Private Key Size:**
    - Level 1: 2,130 bits
    - Level 3: 2,296 bits
    - Level 5: 4,384 bits
- **Ciphertext Size:**
    - Level 1: 20,326 bits
    - Level 3: 43,786 bits
    - Level 5: 65,498 bits

Performance Characteristics

Speed:

- **Key Generation:**
    - Level 1: 730,025 cycles
    - Level 3: 1,709,921 cycles
    - Level 5: 2,986,647 cycles
- **Encapsulation:**
    - Level 1: 689,193 cycles
    - Level 3: 1,850,425 cycles
    - Level 5: 3,023,816 cycles
- **Decapsulation:**
    - Level 1: 2,901,203 cycles
    - Level 3: 7,666,855 cycles
    - Level 5: 17,483,906 cycles

Memory Requirements:
Dependent on representation but generally:

- **Private Key:** w·log2(r)w \cdot \log_2(r)w·log2(r)
- **Public Key and Ciphertext:** nnn (equal to their bit size above)

Security Analysis

Classical Security: Equivalent to NIST levels (AES-128, AES-192, AES-256)
Quantum Security: Comparable reductions for QC-MDPC codes
Known Attack Vectors:

1. Information Set Decoding (ISD)
2. Exploiting the quasi-cyclic structure (provides r-speedup)

Implementation Considerations

Hardware Requirements: Supports x86 platforms with AVX2 and AVX512 optimizations
Software Complexity: Requires cryptographic libraries for modular operations
Integration Challenges: Batch key generation for efficiency requires secure state management

---

Advantages and Limitations

Pros:

- Compact ciphertext size compared to other code-based schemes
- Relatively simple decoding using bit flipping algorithms

Cons:

- High decapsulation latency
- Dependence on quasi-cyclic structure may introduce vulnerabilities

---

Standardization Status

NIST Round:

- Advanced to Round 4
  Other Standards: No other concurrent standards mentioned

CFPKM

Basic Information

Family Type: Multivariate Polynomial-based
Purpose: Key Encapsulation
NIST Security Level: Not explicitly mentioned in the document

---

Technical Overview

Mathematical Foundation: Solving a system of noisy non-linear polynomials, also known as the PoSSo with Noise problem

Key Components:

- **Public Key Size:** 696 bytes (CFPKM128)
- **Private Key Size:** 128 bytes (CFPKM128)
- **Ciphertext Size:** 729 bytes

Performance Characteristics

Speed:

- **Key Generation:** ~72 ms
- **Encapsulation:** ~108 ms
- **Decapsulation:** ~143 ms

Memory Requirements: Not specifically stated

Security Analysis

Classical Security: 128 bits (CFPKM128)
Quantum Security: Not explicitly provided
Known Attack Vectors:

1. Exhaustive Search
2. Arora-Ge Grobner Basis
3. Hybrid Attack combining error analysis and Grobner Basis

Implementation Considerations

Hardware Requirements: Tested on a Linux platform with 31.3 GiB RAM and Intel i7-6600U CPU
Software Complexity: Relatively high due to reliance on solving noisy polynomial systems and Grobner Basis techniques
Integration Challenges: Complexity in ensuring parameter selection avoids vulnerabilities from known attacks

---

Advantages and Limitations

Pros:

1. Smaller key and communication sizes compared to lattice-based schemes
2. Flexible and secure against many known classical and quantum attacks

Cons:

1. Reliance on relatively new and less analyzed PoSSo with Noise problem
2. Increased computational complexity due to Grobner Basis requirements

---

Standardization Status

Other Standards: No other concurrent standards mentioned

Classic McEliece

Basic Information

- **Family Type:** Code-based cryptography
- **Purpose:** Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** IND-CCA2 (Categories 5)

Technical Overview

- **Mathematical Foundation:** Binary Goppa Codes
- **Key Components:**
    - **Public Key Size:** mceliece8192128 uses 1,357,824 bytes
    - **Private Key Size:** mceliece8192128 uses 14,080 bytes
    - **Ciphertext Size:** mceliece8192128 uses 240 bytes

Performance Characteristics

- **Speed:**
    - **Key Generation:** 4 billion cycles (approx. 2 seconds)
    - **Encapsulation:** ~300,000 cycles
    - **Decapsulation:** ~450,000 cycles
- **Memory Requirements:** Large RAM usage due to key size

Security Analysis

- **Classical Security:** Over 256-bit security against information-set decoding
- **Quantum Security:** Resilient to quantum attacks using Grover's algorithm
- **Known Attack Vectors:** Information-set decoding remains the primary attack vector

Implementation Considerations

- **Hardware Requirements:** FPGA implementations optimize performance significantly
- **Software Complexity:** High due to the size of public/private keys

- **Integration Challenges:** Large public key size poses challenges in practical implementations

Advantages and Limitations

- **Pros:**
  - Proven stability and security track record over decades
  - Resilient against both classical and quantum attacks
- **Cons:**
  - Very large public key size
  - Relatively slow key generation process

Standardization Status

- **NIST Round:** Advanced to Round 4
- **Other Standards:** No other concurrent standards mentioned

Compact-LWE

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Public Key Encryption
- **NIST Security Level:** Comparable to AES-192 (194 bits of classical security)

Technical Overview

- **Mathematical Foundation:** Learning with Errors (LWE) problem, modified with additional secret values and errors (Compact-LWE).
- **Key Components:**
  - **Public Key Size:** ~2064 bytes
  - **Private Key Size:** ~232 bytes
  - **Ciphertext Size:** ~36 bytes for a 4-byte plaintext block

---

Performance Characteristics

- **Speed:**
  - **Key Generation:** ~1.55 seconds for 10,000 key pairs
  - **Encryption:** ~1.29 seconds for 32-byte plaintext (10,000 encryptions)

- **Decryption:** ~0.18 seconds for 32-byte plaintext (10,000 decryptions)

- **Memory Requirements:** Designed for lightweight applications, with an implementation for Contiki OS on wireless sensor nodes.

Security Analysis

- **Classical Security:** 194 bits
- **Quantum Security:** Likely comparable but not explicitly defined in the document
- **Known Attack Vectors:**
  - Resistant to lattice-based attacks such as CVP and SIS.
  - Errors introduced in Compact-LWE samples are too large for traditional lattice-based attacks to succeed.

Implementation Considerations

- **Hardware Requirements:** Minimal; tested on low-power devices like Tmote Sky sensor nodes.
- **Software Complexity:** Straightforward implementation; uses simple mathematical operations.
- **Integration Challenges:** Public key size (~2 KB) is relatively large compared to RSA or ECC, which may limit its utility in some contexts.

Advantages and Limitations

- **Pros:**
  - Resistant to standard LWE attacks due to its construction.
  - Lightweight design suitable for constrained environments.
  - Deterministic correctness with no decryption failures.
- **Cons:**
  - Larger public key size compared to classical encryption schemes.
  - Relatively less adoption and standardization compared to other lattice-based schemes.

Standardization Status

- **NIST Round:** Did not advance beyond Round 1.
- **Other Standards:** Not mentioned in the document.

CRYSTALS-DILITHIUM

Basic Information

Family Type: Lattice-based
Purpose: Digital Signatures
NIST Security Level: 1-5 (depending on parameters)

Technical Overview

Mathematical Foundation: Hardness of finding short vectors in lattices
Key Components:

- **Public Key Size:** 896 - 1760 bytes (varies by security level)
- **Private Key Size:** 112 - 3856 bytes
- **Signature Size:** 1487 - 3366 bytes

Performance Characteristics

Speed:

- **Key Generation:** ~170,000-512,000 cycles (Haswell)
- **Signing:** ~765,000-1,817,000 cycles (Haswell)
- **Verification:** ~196,000-548,000 cycles (Haswell)

Memory Requirements: Efficient for various devices, including AVX2 optimizations

Security Analysis

Classical Security: 68-176 bits
Quantum Security: 62-160 bits
Known Attack Vectors: Lattice reduction techniques (e.g., BKZ)

Implementation Considerations

Hardware Requirements: Efficient with AVX2 optimizations for modern CPUs
Software Complexity: Modular and efficient, leveraging SHAKE-128/256
Integration Challenges: None significant due to standardized components

Advantages and Limitations

Pros:

- Compact public key and signature size
- Simple to implement securely
- Deterministic signing to minimize attacks

Cons:

- Performance may vary on non-optimized hardware
- Requires careful parameter selection to maintain security

Standardization Status

NIST Round: Advanced to Round 3 Finalist
Other Standards: No other concurrent standards mentioned

CRYSTALS-KYBER

Basic Information

- **Family Type**: Lattice-based (Module Learning with Errors - MLWE).
- **Purpose**: Key Encapsulation Mechanism (KEM).
- **NIST Security Level**: Supports Levels 1, 3, and 5.

Technical Overview

- **Mathematical Foundation**: Hardness of solving MLWE problems in module lattices.
- **Key Components**:
  - **Public Key Size**: 736 bytes (Kyber512), 1088 bytes (Kyber768), 1440 bytes (Kyber1024).
  - **Private Key Size**: 1632 bytes (Kyber512), 2400 bytes (Kyber768), 3168 bytes (Kyber1024).
  - **Ciphertext Size**: 800 bytes (Kyber512), 1152 bytes (Kyber768), 1504 bytes (Kyber1024).

Performance Characteristics

- **Speed** (for Kyber512, reference implementation on Intel Core i7-4770K):
  - Key Generation: 141,872 cycles.
  - Encapsulation: 205,468 cycles.
  - Decapsulation: 246,040 cycles.

Memory Requirements

- Efficient in memory usage; no heap allocations required.

Security Analysis

- **Classical Security**: Based on MLWE problem; core-SVP hardness of 112 bits (Kyber512).
- **Quantum Security**: Core-SVP hardness of 102 bits (Kyber512).
- **Known Attack Vectors**: Decryption failures, side-channel attacks, multi-target attacks.

Implementation Considerations

- **Hardware Requirements**: Supports optimizations via AVX2 for better performance.

- **Software Complexity**: Moderate; requires careful implementation to avoid timing and side-channel attacks.
- **Integration Challenges**: Requires efficient pseudorandom number generation and compression techniques.

Advantages and Limitations

Pros:

1. Strong security based on MLWE.
2. Small ciphertext and key sizes compared to other lattice-based schemes.

Cons:

1. Requires high cycle counts for operations on constrained devices.
2. Vulnerable to side-channel attacks if not implemented with constant-time operations.

Standardization Status

- **NIST Round**: Advanced to Round 3 and selected as one of the final standards.
- **Other Standards**: Adopted widely in post-quantum cryptography discussions.

DAGS

Basic Information

- **Family Type:** Code-based
- **Purpose:** Key Encapsulation
- **NIST Security Level:** 1, 3, 5 (depending on parameter sets)

Technical Overview

- **Mathematical Foundation:** Syndrome Decoding Problem (SDP) and Quasi-Dyadic Generalized Srivastava (GS) codes
- **Key Components:**
  - **Public Key Size:**
    - DAGS-1: 6760 bytes
    - DAGS-3: 8448 bytes
    - DAGS-5: 11616 bytes
  - **Private Key Size:**
    - DAGS-1: 432640 bytes
    - DAGS-3: 1284096 bytes
    - DAGS-5: 2230272 bytes

    - **Ciphertext Size:**
        - DAGS-1: 552 bytes
        - DAGS-3: 944 bytes
        - DAGS-5: 1616 bytes

Performance Characteristics

- **Speed (cycles):**
    - **Key Generation:**
        - DAGS-1: ~49 billion
        - DAGS-3: ~107 billion
        - DAGS-5: ~137 billion
    - **Encapsulation:**
        - DAGS-1: ~20 million
        - DAGS-3: ~26 million
        - DAGS-5: ~49 million
    - **Decapsulation:**
        - DAGS-1: ~23 million
        - DAGS-3: ~25 million
        - DAGS-5: ~261 million
- **Memory Requirements:** Significant, especially for private key storage.

Security Analysis

- **Classical Security:** ≥128 bits for all levels
- **Quantum Security:** Same as classical due to reliance on structured coding problems
- **Known Attack Vectors:**
    - Information Set Decoding (ISD)
    - Faugère-Otmani-Perret-Tillich (FOPT) structural attack

Implementation Considerations

- **Hardware Requirements:** Significant computational and memory resources
- **Software Complexity:** Relatively high due to intricate matrix operations
- **Integration Challenges:** Large key sizes may limit deployment on constrained devices

Advantages and Limitations

- **Pros:**

    - IND-CCA security in both classical and quantum random oracle models
    - Efficient encapsulation and decapsulation operations
    - Compact public key compared to other code-based schemes
- **Cons:**
    - Large private key size
    - Vulnerable to FOPT attacks if parameters are not carefully chosen

Standardization Status

- **NIST Round:** Did not advance beyond Round 1
- **Other Standards:** None mentioned

Ding Key Exchange

Basic Information

Family Type: Lattice-based
Purpose: Key Encapsulation
NIST Security Level: Not specified

Technical Overview

Mathematical Foundation: Ring Learning with Errors (RLWE)
Key Components:

- **Public Key Size:** Not explicitly mentioned
- **Private Key Size:** Not explicitly mentioned
- **Ciphertext Size:** Not explicitly mentioned

Performance Characteristics

Speed:

- **Key Generation:** Not explicitly mentioned
- **Encryption/Signing:** Not explicitly mentioned
- **Decryption/Verification:** Not explicitly mentioned

Memory Requirements: Not explicitly mentioned

Security Analysis

Classical Security: Not explicitly mentioned
Quantum Security: Based on the RLWE problem, resistant to known quantum attacks
Known Attack Vectors:

- BKZ lattice reduction
- Sieving algorithms

Implementation Considerations

Hardware Requirements: Not specified
Software Complexity: Not specified
Integration Challenges: Not explicitly detailed

Advantages and Limitations

Pros:

- Efficient rounding and reconciliation methods reduce communication costs
- Based on RLWE, providing strong security guarantees

Cons:

- Requires further analysis on RLWE hardness and parameter choices
- Communication costs, while reduced, remain larger compared to some alternatives

---

Standardization Status

NIST Round: Did not advance beyond Round 1
Other Standards: No other concurrent standards mentioned

DME: Double Matrix Exponentiation Cryptosystem

Family Type: Multivariate-based
Purpose: Public-key encryption, digital signatures, and key encapsulation
NIST Security Level: Not specified for standard levels but claims to achieve 128-bit and 256-bit classical security

---

Technical Overview

Mathematical Foundation: Multivariate polynomial systems based on double matrix exponentiation over finite fields

Key Components:

- **Public Key Size:** 1152 bytes (128-bit security) or 2304 bytes (256-bit security)

- **Private Key Size:** 144 bytes (128-bit security) or 288 bytes (256-bit security)
- **Ciphertext Size:** 18 bytes (128-bit security) or 36 bytes (256-bit security)

Performance Characteristics

Speed:

- **Key Generation:** Parameters suggest practical performance with modern processors
- **Encryption/Signing:** Supports encryption for arbitrary message sizes within the field
- **Decryption/Verification:** Fast decryption using explicit inversion maps

Memory Requirements:

- Efficient for hardware and software implementations with modest resource requirements

---

Security Analysis

Classical Security: Achieves 128-bit and 256-bit security against classical attacks
Quantum Security: Claims equivalent security to AES-256 for 256-bit parameters but lacks thorough structural attack analysis
Known Attack Vectors:

- Gröbner basis attacks
- Algebraic cryptanalysis
- Structural attacks (yet to be fully explored)

Implementation Considerations

Hardware Requirements: Supports efficient implementation on modern CPUs with typical resources
Software Complexity: Medium complexity due to structured polynomial transformations
Integration Challenges: New and untested; may face challenges in widespread adoption

---

Advantages and Limitations

Pros:

- Flexible parameterization for different security levels
- Compact ciphertexts and private keys
- Resistant to timing side-channel attacks

Cons:

- Limited analysis of resistance to new structural attacks
- Some failure probability in digital signatures requiring message padding

---

Standardization Status

NIST Round: Did not advance beyond Round 1
Other Standards: No concurrent standards

DRS

Basic Information

Family Type: Lattice-based
Purpose: Digital Signatures
NIST Security Level: Not explicitly stated; parameters suggest up to Level 5 security.

Technical Overview

Mathematical Foundation:

- Based on the Guaranteed Distance Decoding (GDD) problem in diagonal dominant lattices.
- Relies on the unique Shortest Vector Problem (uSVP) and Bounded Distance Decoding (BDD).

Key Components:

- **Public Key Size:** Proportional to the lattice dimension nnn and the bound on coefficients of the matrix PPP.
- **Private Key Size:** Defined by lattice dimension nnn, noise levels, and a random seed.
- **Signature Size:** Proportional to the lattice dimension and the norm bounds on reduced vectors.

Performance Characteristics

Speed:

- Key Generation: Involves lattice basis transformations, dependent on matrix size and dimensions.
- Signing: Reduction of message vectors until they satisfy lattice conditions.
- Verification: Relies on equality checks and modular arithmetic.

Memory Requirements:

- Public key matrix storage, private seed storage, and intermediate matrix operations.

Security Analysis

Classical Security: Estimated for different lattice dimensions, up to 256 bits.
Quantum Security: Claims resilience against known quantum attacks due to reliance on hard lattice problems.
Known Attack Vectors:

- BDD-based attacks using uSVP reduction.
- Security depends on parameters such as lattice dimension nnn, reduction matrix sparsity, and noise levels.

Implementation Considerations

Hardware Requirements: Moderate computational and memory resources for large matrix operations.
Software Complexity: Involves generating diagonal dominant lattices, vector reductions, and signature verification.
Integration Challenges: Ensuring efficient handling of large matrix operations and avoiding overflow errors in modular arithmetic.

Advantages and Limitations

Pros:

- Strong theoretical foundation in lattice problems.
- Provides digital signatures with tunable security parameters.

Cons:

- High computational overhead for key generation and verification.
- Relatively large key and signature sizes compared to other schemes.

Standardization Status

NIST Round: Did not advance to Round 3 or Round 4.
Other Standards: Not currently adopted in other cryptographic standards.

DualModeMS

Basic Information

- **Family Type**: Multivariate-based
- **Purpose**: Digital Signatures
- **NIST Security Level**: Levels 1, 3, and 5 (parameters provided for 128, 192, and 256-bit security levels)

Technical Overview

- **Mathematical Foundation**: Multivariate quadratic equations over finite fields (HFEv-schemes)
- **Key Components**:
  - **Public Key Size**: Varies by security level:
    - Level 1: 528 bytes
    - Level 3: 1,560 bytes
    - Level 5: 2,112 bytes
  - **Private Key Size**: Approximately 18 MB
  - **Signature Size**:
    - Level 1: 32 KB
    - Level 3: 79 KB
    - Level 5: 149 KB

Performance Characteristics

- **Speed**:
  - **Key Generation**:
    - Optimized: ~552 seconds
    - Non-optimized: ~797 seconds
  - **Signing**: 2.05 seconds
  - **Verification**: 2.84 ms
- **Memory Requirements**: Public key and signature sizes significantly affect memory, especially with higher security levels.

Security Analysis

- **Classical Security**:
  - Resistant to exhaustive search and Grover's algorithm-based quantum attacks.
  - Security depends on the difficulty of solving quadratic systems of equations over finite fields.
- **Quantum Security**:
  - Specific parameter choices address quantum adversaries, although Gröbner basis attacks and BooleanSolve have implications for efficiency.
- **Known Attack Vectors**:
  - Gröbner basis attacks

- Approximation algorithms
- Key-recovery attacks

Implementation Considerations

- **Hardware Requirements**: Requires substantial resources for large-scale key generation and signing.
- **Software Complexity**: Moderate, leveraging existing multivariate signature scheme principles.
- **Integration Challenges**: Large signature sizes may challenge low-bandwidth environments.

Advantages and Limitations

- **Pros**:
  - Small public key size relative to similar multivariate schemes.
  - Comprehensive security analyses, including provable reductions.
- **Cons**:
  - Large signature size.
  - Computationally expensive key generation.

Standardization Status

- **NIST Round**: Did not advance to Round 3 or Round 4.

EMBLEM

Basic Information

- **Family Type**: Code-based
- **Purpose**: Key Encapsulation Mechanism (KEM)
- **NIST Security Level**: Level 3

Technical Overview

- **Mathematical Foundation**: Decoding random linear codes over finite fields
- **Key Components**:
  - **Public Key Size**: 147,456 bytes
  - **Private Key Size**: 8,192 bytes
  - **Ciphertext Size**: 256 bytes

Performance Characteristics

- **Speed**:

- **Key Generation**: ~1,000 operations/second
- **Encryption/Encapsulation**: ~1,500 operations/second
- **Decryption/Decapsulation**: ~1,200 operations/second

- **Memory Requirements**: Moderate memory requirements (RAM usage dependent on implementation).

Security Analysis

- **Classical Security**: 128 bits
- **Quantum Security**: 64 bits
- **Known Attack Vectors**:
  - Algebraic decoding attacks
  - Syndrome decoding attacks

Implementation Considerations

- **Hardware Requirements**: Suitable for standard hardware with moderate computational power.
- **Software Complexity**: Relatively straightforward to implement.
- **Integration Challenges**: Large public key size can present storage and transmission challenges.

Advantages and Limitations

- **Pros**:
  - High level of security against classical attacks.
  - Efficient encryption and decapsulation.
- **Cons**:
  - Very large public key size.
  - Moderate resistance against quantum attacks.

Standardization Status

- **NIST Round**: Did not advance beyond Round 1.
- **Other Standards**: Not mentioned.

R.EMBLEM

Basic Information

- **Family Type**: Code-based
- **Purpose**: Key Encapsulation Mechanism (KEM)

- **NIST Security Level**: Level 3

Technical Overview

- **Mathematical Foundation**: Based on random quasi-cyclic linear codes.
- **Key Components**:
    - **Public Key Size**: 73,728 bytes
    - **Private Key Size**: 4,096 bytes
    - **Ciphertext Size**: 128 bytes

Performance Characteristics

- **Speed**:
    - **Key Generation**: ~2,000 operations/second
    - **Encryption/Encapsulation**: ~2,500 operations/second
    - **Decryption/Decapsulation**: ~2,000 operations/second
- **Memory Requirements**: Lower memory requirements compared to EMBLEM.

Security Analysis

- **Classical Security**: 128 bits
- **Quantum Security**: 64 bits
- **Known Attack Vectors**:
    - Information-set decoding attacks
    - Structural attacks on quasi-cyclic codes

Implementation Considerations

- **Hardware Requirements**: Suitable for devices with constrained resources.
- **Software Complexity**: Simple to implement with available libraries.
- **Integration Challenges**: Smaller public key size compared to EMBLEM makes it more practical.

Advantages and Limitations

- **Pros**:
    - Reduced key size compared to EMBLEM.
    - Faster encryption and decapsulation.
- **Cons**:
    - Lower security margin against advanced quantum attacks.
    - Dependent on the strength of the quasi-cyclic structure.

Standardization Status

- **NIST Round**: Did not advance beyond Round 1.
- **Other Standards**: Not mentioned.

FALCON

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Digital Signatures
- **NIST Security Level:** 1, 2, 3, 5

Technical Overview

- **Mathematical Foundation:** Based on the Gentry-Peikert-Vaikuntanathan (GPV) framework and NTRU lattices
- **Key Components:**
    - **Public Key Size:** Varies based on security level, around 897–1793 bytes
    - **Private Key Size:** 512–1024 bytes
    - **Signature Size:** Compact, 666–1330 bytes depending on security level

Performance Characteristics

- **Speed:**
    - **Key Generation:** Efficient, due to reliance on Fast Fourier Transform (FFT)
    - **Encryption/Signing:** Supports over 1000 signatures per second on standard hardware
    - **Decryption/Verification:** Extremely fast due to modular design and FFT
- **Memory Requirements:** Optimized for compactness, making it suitable for embedded devices

Security Analysis

- **Classical Security:** Proven secure under lattice hardness assumptions
- **Quantum Security:** Secure under the quantum random oracle model
- **Known Attack Vectors:**
    - Lattice reduction
    - Key recovery attacks
    - Overstretched NTRU attacks

Implementation Considerations

- **Hardware Requirements:** Requires FFT operations, but efficient for constrained devices
- **Software Complexity:** Moderate; key generation and FFT operations demand precise implementation
- **Integration Challenges:** Limited by reliance on floating-point arithmetic for some operations

Advantages and Limitations

- **Pros:**
    - Highly compact signatures and keys
    - Proven security in both classical and quantum models
    - Very fast signing and verification
    - Modular and adaptable design for other lattice types
- **Cons:**
    - Relies on floating-point arithmetic, limiting some hardware compatibility
    - Implementation requires advanced understanding of lattice-based mathematics

Standardization Status

- **NIST Round:** Advanced to Round 3 as a Digital Signature Finalist
- **Other Standards:** None currently mentioned.

FrodoKEM

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Key Encapsulation
- **NIST Security Level:** FrodoKEM-640 targets Level 1, FrodoKEM-976 targets Level 3.

Technical Overview

- **Mathematical Foundation:** Learning With Errors (LWE) problem
- **Key Components:**
    - **Public Key Size:** FrodoKEM-640: ~9.7 KB; FrodoKEM-976: ~15.4 KB
    - **Private Key Size:** FrodoKEM-640: ~9.7 KB; FrodoKEM-976: ~15.4 KB
    - **Ciphertext Size:** FrodoKEM-640: ~9.5 KB; FrodoKEM-976: ~15.2 KB

Performance Characteristics

- **Speed:**
    - **Key Generation:** Not specified in document

    - **Encapsulation:** Not specified in document
    - **Decapsulation:** Not specified in document
- **Memory Requirements:** High due to large matrix operations.

Security Analysis

- **Classical Security:** Matches AES-128 for FrodoKEM-640 and AES-192 for FrodoKEM-976.
- **Quantum Security:** Assumes hardness of LWE against quantum attacks.
- **Known Attack Vectors:** Based on lattice reduction and attacks on LWE problems.

Implementation Considerations

- **Hardware Requirements:** Supports x64 Intel with optional AES acceleration; ARM implementation available.
- **Software Complexity:** Relatively simple compared to ring-LWE-based constructions.
- **Integration Challenges:** None specified.

Advantages and Limitations

- **Pros:**
    - Conservative and highly secure parameterization.
    - Easy to implement due to algebraically unstructured lattices.
- **Cons:**
    - Larger key and ciphertext sizes compared to structured-lattice approaches.
    - Higher computational cost.

Standardization Status

- **NIST Round:** Advanced to Round 2 but not a finalist.
- **Other Standards:** None specified.

GeMSS

Basic Information

- **Family Type:** Multivariate-based
- **Purpose:** Digital Signatures
- **NIST Security Level:** 1, 3, 5 (depending on parameter set)

Technical Overview

- **Mathematical Foundation:** Hidden Field Equations with vinegar and minus modifiers (HFEv-)
- **Key Components:**

  - **Public Key Size:**
    - 352.18 KB (GeMSS128)
    - 1237.96 KB (GeMSS192)
    - 3040.69 KB (GeMSS256)
  - **Private Key Size:**
    - 14.208 KB (GeMSS128)
    - 39.440 KB (GeMSS192)
    - 82.056 KB (GeMSS256)
  - **Signature Size:**
    - 48 bytes (GeMSS128)
    - 88 bytes (GeMSS192)
    - 104 bytes (GeMSS256)

Performance Characteristics

- **Speed:**
  - **Key Generation:**
    - GeMSS128: 42 ms (optimized)
    - GeMSS192: 166 ms (optimized)
    - GeMSS256: 424 ms (optimized)
  - **Signing:**
    - GeMSS128: 260 ms (optimized)
    - GeMSS192: 694 ms (optimized)
    - GeMSS256: 1.09 s (optimized)
  - **Verification:**
    - GeMSS128: 41 μs (optimized)
    - GeMSS192: 117 μs (optimized)
    - GeMSS256: 336 μs (optimized)
- **Memory Requirements:** Depends on parameter set; large public key sizes require significant storage.

Security Analysis

- **Classical Security:** Matches specified security levels (128, 192, 256 bits)
- **Quantum Security:** Designed to withstand attacks leveraging Grover's and BooleanSolve algorithms

- **Known Attack Vectors:** Grobner bases attacks, algebraic structure exploitation

Implementation Considerations

- **Hardware Requirements:** Supports optimizations with AVX2 and specialized polynomial multiplication
- **Software Complexity:** Requires handling of large matrix operations and efficient polynomial arithmetic
- **Integration Challenges:** Large public key size could pose challenges in constrained environments

Advantages and Limitations

- **Pros:**
  - Very small signature size
  - Fast verification
  - Well-studied mathematical foundation
- **Cons:**
  - Very large public key size
  - High computational cost for key generation and signing

Standardization Status

- **NIST Round:** Reached Round 3 as a Digital Signature Algorithm
- **Other Standards:** Not mentioned in concurrent standards

Giophantus

Basic Information

- **Family Type**: Algebraic Surface Cryptosystem (ASC)
- **Purpose**: Public-key encryption
- **NIST Security Level**: Not specified in the document.

Technical Overview

- **Mathematical Foundation**: Solving indeterminate equations over quotient rings RqR_qRq.
- **Key Components**:
  - **Public Key Size**: Polynomial equations of degree dXd_XdX (actual size not specified).
  - **Private Key Size**: Polynomials of degree n−1n-1n−1 over RℓR_\ellRℓ (size not explicitly stated).

- **Ciphertext Size**: Dependent on parameters n,ℓ,q,dX,drn, \ell, q, d_X, d_rn,ℓ,q,dX,dr.

Performance Characteristics

- **Speed**:
  - **Key Generation**: Details not provided.
  - **Encryption**: Dependent on bivariate polynomial generation.
  - **Decryption**: Relies on solving equations with small solutions over RqR_qRq.
- **Memory Requirements**: Not specified in the document.

Security Analysis

- **Classical Security**: Based on the hardness of solving non-linear indeterminate equations.
- **Quantum Security**: Introduces a new computational assumption analogous to Learning With Errors (LWE), termed Indeterminate Equation Learning with Errors (IE-LWE).
- **Known Attack Vectors**:
  - Linear Algebra Attack
  - Lattice-Based Attacks (including subfield lattice attack)

Implementation Considerations

- **Hardware Requirements**: Computational operations on polynomial rings over FqF_qFq.
- **Software Complexity**: Requires efficient polynomial arithmetic in RqR_qRq.
- **Integration Challenges**: Designing parameters to balance efficiency and security.

Advantages and Limitations

- **Pros**:
  - Hardness of indeterminate equations ensures robustness.
  - Proven IND-CPA security with Fujisaki–Okamoto conversion.
- **Cons**:
  - No general solution for parameter recommendations.
  - Vulnerability to lattice-related attacks if parameters are not carefully chosen.

Standardization Status

- **NIST Round**: Did not advance to Round 3 or Round 4.
- **Other Standards**: Not mentioned

Gravity-SPHINCS

Basic Information

Family Type: Hash-based
Purpose: Digital Signatures
NIST Security Level: Not explicitly mentioned in the provided documents

Technical Overview

Mathematical Foundation: Hash-based signature scheme using Merkle trees and Winternitz one-time signatures (WOTS).
Key Components:

- **Public Key Size:** Not explicitly stated
- **Private Key Size:** Not explicitly stated
- **Signature Size:** Typically 20–30 KiB

Performance Characteristics

Speed:

- Not explicitly detailed, but optimizations like batch signing are intended to enhance performance.

Memory Requirements: Reduced memory due to optimizations like secret key caching.

Security Analysis

Classical Security: Relies on collision resistance of hash functions.
Quantum Security: Secure under the assumption that hash functions remain resistant to quantum attacks.
Known Attack Vectors:

- Collision resistance of hash functions is critical to the scheme's security.

Implementation Considerations

- **Hardware Requirements:** Optimized for AES-NI instructions, supports pipelining and multithreading.
- **Software Complexity:** Includes multiple optimizations for practical efficiency.
- **Integration Challenges:** The scheme’s signature size can be a limitation in certain applications.

Advantages and Limitations

Pros:

- High assurance of security based on well-understood hash function properties.
- Flexibility in performance and size trade-offs.
- Stateless design simplifies implementation.

Cons:

- Large signature size compared to other schemes.
- Complex construction with multiple layers and optimizations.

Standardization Status

NIST Round: Did not advance beyond Round 1.
Other Standards: None mentioned.

Guess Again

Basic Information

Family Type: Unconditionally Secure Scheme (with Decryption Errors)
Purpose: Public-Key Encryption
NIST Security Level: Not specified

Technical Overview

Mathematical Foundation: Random walks and probabilistic interval guessing.
Key Components:

- **Public Key Size:** ~18,000 bits
- **Private Key Size:** ~16,000 bits
- **Ciphertext Size:** ~18,000 bits for a single bit encryption

Performance Characteristics

Speed:

- **Key Generation:** Computationally intensive (involves multiple random walks).
- **Encryption:** ~0.016 seconds per bit (without parallelization).
- **Decryption:** Same order as encryption.

Memory Requirements: ~32 MB (including offline pre-computation phase).

Security Analysis

Classical Security: Security against computationally unbounded adversaries, with controlled decryption errors.
Quantum Security: Claims security against quantum adversaries as well.
Known Attack Vectors: No specific computational attacks; security relies on information-theoretical principles.

Implementation Considerations

Hardware Requirements: Basic computational resources; efficiency improves with pre-computation.
Software Complexity: High due to the reliance on random walk computations and probabilistic interval selection.
Integration Challenges: Limited to specific scenarios due to high ciphertext expansion factor.

Advantages and Limitations

Pros:

1. Security does not rely on computational assumptions.
2. Resistant to quantum adversaries.

Cons:

1. Extremely high ciphertext expansion factor.
2. Practical use is limited to applications requiring very high security for small data.

Standardization Status

NIST Round: Did not advance to Round 3 or 4.
Other Standards: Not mentioned in the documentation.

Gui

Basic Information

- **Family Type:** Multivariate Cryptography, BigField schemes
- **Purpose:** Digital Signatures
- **NIST Security Level:** Not explicitly mentioned; suggested for categories I-VI based on proposed parameter sets.

Technical Overview

- **Mathematical Foundation:** HFEv- signature scheme (modification of HFEv, focusing on optimized key/signature sizes and security).
- **Key Components:**
  - **Public Key Size:**
    - Gui-184: 416.3 KB
    - Gui-312: 1,955.1 KB
    - Gui-448: 5,789.2 KB
  - **Private Key Size:**
    - Gui-184: 19.1 KB
    - Gui-312: 59.3 KB
    - Gui-448: 155.9 KB
  - **Signature Size:**
    - Gui-184: 360 bits
    - Gui-312: 504 bits
    - Gui-448: 664 bits

Performance Characteristics

- **Speed (Gui-184):**
  - **Key Generation:** 213 ms
  - **Signature Generation:** 10.4 ms
  - **Signature Verification:** 0.051 ms
- **Memory Requirements:**
  - Gui-184: 3.3 MB (verification) to 3.5 MB (key generation).

Security Analysis

- **Classical Security:** Varied by instance, ranging from 143 bits to 274 bits based on parameter sets.
- **Quantum Security:** Resistant to Grover's algorithm and quantum brute-force attacks (as claimed by the document).
- **Known Attack Vectors:**
  - Brute force attacks
  - Direct attacks
  - Rank attacks (Kipnis-Shamir type)
  - Distinguishing attacks

Implementation Considerations

- **Hardware Requirements:** Implementation optimized for processors supporting PCLMULQDQ instruction sets.
- **Software Complexity:** Involves extensive use of finite field arithmetic and Cantor-Zassenhaus algorithms.
- **Integration Challenges:** High memory requirements for higher security parameter sets.

Advantages and Limitations

- **Pros:**
  - Balanced security-efficiency trade-off.
  - Scalable across various NIST security levels.
  - Side-channel resistant implementation.
- **Cons:**
  - Large key sizes, especially for higher security levels.
  - Computationally expensive key generation process.

Standardization Status

- **NIST Round:** Did not advance to Round 3 or Round 4.
- **Other Standards:** Not mentioned.

HILA5

Basic Information

- **Family Type**: Lattice-based
- **Purpose**: Key Encapsulation Mechanism (KEM) and Public Key Encryption
- **NIST Security Level**: Claims to meet Category 5 (comparable to AES-256 level)

Technical Overview

- **Mathematical Foundation**: Ring Learning With Errors (Ring-LWE)
- **Key Components**:
  - **Public Key Size**: 1824 bytes
  - **Private Key Size**: 1824 bytes
  - **Ciphertext Size**: 2012 bytes

Performance Characteristics

- **Speed**:
  - **Key Generation**: 68.7 μs
  - **Encapsulation**: 89.9 μs
  - **Decapsulation**: 175.4 μs
- **Memory Requirements**: Comparable to other lattice-based cryptosystems

Security Analysis

- **Classical Security**: Comparable to AES-256
- **Quantum Security**: Estimated to meet the highest NIST Category 5 requirements
- **Known Attack Vectors**: Standard lattice-based attacks

Implementation Considerations

- **Hardware Requirements**: Efficiently implementable on both CPUs and FPGAs
- **Software Complexity**: Moderate, based on the Ring-LWE construction
- **Integration Challenges**: Reconciliation method introduces minor implementation complexities

Advantages and Limitations

- **Pros**:

    - Very low decryption failure rate
    - Efficient performance for both hardware and software implementations
- **Cons**:
    - Slightly larger ciphertext size compared to some other candidates
    - Requires careful tuning of reconciliation parameters

Standardization Status

- **NIST Round**: Did not advance to Round 3
- **Other Standards**: None noted

HiMQ-3

Basic Information

Family Type: Multivariate Quadratic Equations (MQ) **Purpose:** Digital Signatures **NIST Security Level:** Not explicitly mentioned in the document; requires confirmation for the specific implementation.

Technical Overview

Mathematical Foundation: The algorithm is based on the hardness of the Multivariate Quadratic Problem (MQ-Problem) and integrates the Isomorphism of Polynomials (IP) and MinRank problems. It employs a three-layer structure for the generation of multivariate quadratic equations.

Key Components:

- **Public Key Size:** Not explicitly detailed in this section of the document.
- **Private Key Size:** Consists of the central map, linear affine mappings, and associated secret parameters.
- **Signature/Ciphertext Size:** Signature depends on the output of the multivariate quadratic system, specific sizes need further verification.

Performance Characteristics

Speed:

- **Key Generation:** Optimized for reduced public and private key sizes through efficient parameterization.
- **Signing:** Designed for high-speed signing operations, leveraging efficient solvers for central maps.
- **Verification:** Relies on inverting polynomial systems, ensuring verification under defined constraints.

Memory Requirements: Minimal memory overheads for key storage, but dependent on implementation.

Security Analysis

Classical Security: Relies on the infeasibility of solving MQ-Problems with current algorithms.

Quantum Security: Offers resistance to quantum attacks by leveraging the intrinsic difficulty of the MQ and related problems, achieving an approximate 112-bit security level against quantum algorithms like Grover's.

Known Attack Vectors:

- **Direct Attacks:** Complexity estimates indicate resilience with lower bounds provided for security levels.
- **Key Recovery Attacks:** Resilient to attacks exploiting equivalent or good keys.
- **MinRank Attack:** Complexity tied to extracting low-rank matrices from the quadratic system.
- **HighRank Attack:** Defense against extraction of low-usage variables in polynomial terms.
- **Kipnis-Shamir Attack:** Resilient under parameter constraints.

Implementation Considerations

Hardware Requirements: Moderate; parameters ensure lightweight operations on standard hardware.

Software Complexity: Moderate to high, given the need for polynomial system handling and secure parameter generation.

Integration Challenges: Compatibility with existing cryptographic frameworks and standards needs validation.

Advantages and Limitations

Pros:

1. High-speed signing performance.
2. Reduced key sizes compared to alternatives.
3. Strong theoretical foundation in multivariate quadratic systems.

Cons:

1. Complexity in parameter tuning.
2. Higher computational demands for certain attack defenses.

Standardization Status

NIST Round: Did not advance beyond Round 1. However, exhibits foundational contributions to MQ-based schemes.

Other Standards: None explicitly mentioned.

HQC

Hamming Quasi-Cyclic (HQC)

Basic Information

- **Family Type**: Code-based Cryptography
- **Purpose**: Key Encapsulation Mechanism (KEM)
- **NIST Security Level**: 1, 3, and 5

Technical Overview

- **Mathematical Foundation**: Syndrome Decoding Problem for Quasi-Cyclic Codes
- **Key Components**:
    - **Public Key Size**: Varies with parameters (e.g., ~5,558 bytes for security level 1)
    - **Private Key Size**: Varies (e.g., ~252 bytes for security level 1)
    - **Ciphertext Size**: Varies (e.g., ~5,622 bytes for security level 1)

Performance Characteristics

- **Speed**:
    - **Key Generation**: ~0.17 ms for security level 1
    - **Encapsulation**: ~0.36 ms for security level 1
    - **Decapsulation**: ~0.57 ms for security level 1
- **Memory Requirements**: Not explicitly provided

Security Analysis

- **Classical Security**: Matches security levels 1, 3, and 5
- **Quantum Security**: Matches security levels 1, 3, and 5
- **Known Attack Vectors**: Based on decoding errors in Quasi-Cyclic codes

Implementation Considerations

- **Hardware Requirements**: No specialized hardware needed
- **Software Complexity**: Moderate, includes BCH code operations
- **Integration Challenges**: Larger key sizes compared to other candidates

Advantages and Limitations

- **Pros**:
    - IND-CCA2 security achievable
    - Code-based cryptography with strong theoretical foundation
- **Cons**:
    - Large key sizes
    - Higher computation time compared to lattice-based schemes

Standardization Status

- **NIST Round**: Advanced to Round 4
- **Other Standards**: Not specified

KINDI

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Key Encapsulation Mechanism (KEM) and Public-Key Encryption (PKE)
- **NIST Security Level:** Levels 2 to 5 (varies by parameter set)

Technical Overview

- **Mathematical Foundation:** Module Learning With Errors (MLWE)
- **Key Components:**
  - **Public Key Size:** Varies from 1184 bytes to 2368 bytes
  - **Private Key Size:** Varies from 1472 bytes to 2752 bytes
  - **Ciphertext Size:** Varies from 1792 bytes to 3328 bytes

Performance Characteristics

- **Speed (in CPU cycles):**
  - Key Generation: ~203,096 to ~429,952 (Reference Implementation)
  - Encryption/Encapsulation: ~247,793 to ~562,640 (Reference Implementation)
  - Decryption/Decapsulation: ~312,211 to ~698,041 (Reference Implementation)
- **Memory Requirements:**
  - **Ciphertext:** 1792 to 3328 bytes
  - **Public Key:** 1184 to 2368 bytes
  - **Secret Key:** 1472 to 2752 bytes

Security Analysis

- **Classical Security:** 181 to 365 bits (depending on parameter set)
- **Quantum Security:** 164 to 330 bits (depending on parameter set)
- **Known Attack Vectors:** Primal attacks, dual attacks, and lattice sieving-based attacks

Implementation Considerations

- **Hardware Requirements:** Tested on Intel Core i5-6200U with 8GB RAM

- **Software Complexity:** Relies on SHAKE functions and FFT optimizations for efficiency
- **Integration Challenges:** Minimal; suitable for TLS and constrained environments

Advantages and Limitations

- **Pros:**
    - Efficient encryption and decryption with low failure rates
    - Supports flexible parameter sets for various security levels
    - Compact keys and ciphertexts
- **Cons:**
    - Slightly higher decryption times for higher security levels
    - Requires careful parameter tuning to avoid decryption failures

Standardization Status

- **NIST Round:** Did not advance beyond Round 1

LAC

Basic Information

Family Type: Lattice-based Cryptosystems
Purpose: Key Encapsulation Mechanism (KEM), Digital Signatures
NIST Security Level: 1, 3, 5

Technical Overview

Mathematical Foundation: Polynomial Learning with Errors (poly-LWE) problem over the ring Rq=Zq[x]/(xn+1)R_q = Z_q[x]/(x^n + 1)Rq=Zq[x]/(xn+1)
Key Components:

- Public Key Size: 544 bytes (LAC128), 1056 bytes (LAC192, LAC256)
- Private Key Size: 1056 bytes (LAC128), 2080 bytes (LAC192, LAC256)
- Ciphertext Size: 1024 bytes (LAC128), 1536 bytes (LAC192), 2048 bytes (LAC256)

Performance Characteristics

Speed:

- **Key Generation:** 12.56 μs (LAC128, optimized version), 39.62 μs (AVX2-based version)
- **Encryption:** 17.21 μs (LAC128, optimized version)
- **Decryption:** 8.79 μs (LAC128, optimized version)

Memory Requirements: Minimal due to efficient AVX2-based implementation.

Security Analysis

Classical Security: Up to 256 bits
Quantum Security: Up to 290 bits
Known Attack Vectors: Based on solving the poly-LWE problem using lattice reduction techniques like BKZ.

Implementation Considerations

Hardware Requirements: Intel x64 processors with AVX2 instructions for optimized performance.
Software Complexity: Moderate, relies on AVX2 and efficient BCH coding.
Integration Challenges: Requires specialized AVX2-compatible processors for best performance.

Advantages and Limitations

Pros:

- High efficiency with small key and ciphertext sizes.
- Resistance to known quantum and classical attacks.

Cons:

- Limited to specific dimensions (e.g., n=512,1024n = 512, 1024n=512,1024).
- Requires AVX2 for optimal performance.

Standardization Status

NIST Round: Advanced to Round 3.
Other Standards: None.

LAKE

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Level 1, 3, and 5 (different parameter sets for varying security levels)

Technical Overview

- **Mathematical Foundation:** Learning With Errors (LWE) problem
- **Key Components:**
    - **Public Key Size:** 966 bytes (Level 1), 1506 bytes (Level 3), 1982 bytes (Level 5)
    - **Private Key Size:** 1218 bytes (Level 1), 1866 bytes (Level 3), 2472 bytes (Level 5)

- **Ciphertext Size:** 1087 bytes (Level 1), 1696 bytes (Level 3), 2270 bytes (Level 5)

Performance Characteristics

- **Speed:**
  - **Key Generation:** 50,000+ ops/sec (Level 1)
  - **Encapsulation:** ~40,000 ops/sec (Level 1)
  - **Decapsulation:** ~35,000 ops/sec (Level 1)
- **Memory Requirements:** Minimal RAM footprint for implementation on constrained devices

---

Security Analysis

- **Classical Security:** Matches claimed security levels against classical attacks
- **Quantum Security:** Designed to resist quantum attacks leveraging Grover's and Shor's algorithms
- **Known Attack Vectors:** Analysis against brute force, side-channel, and lattice reduction attacks

---

Implementation Considerations

- **Hardware Requirements:** Designed for efficient implementation on low-power devices
- **Software Complexity:** Moderate complexity due to lattice operations
- **Integration Challenges:** Requires careful parameter tuning for high-performance environments

---

Advantages and Limitations

- **Pros:**
  - Compact key and ciphertext sizes for lattice-based schemes
  - High-speed operations, suitable for real-time applications
- **Cons:**
  - Potential overhead in parameter generation
  - Requires optimization for memory-constrained environments

---

Standardization Status

- **NIST Round:** Did not advance beyond Round 2

LEDAkem

Basic Information

- **Family Type:** Code-based
- **Purpose:** Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Not explicitly mentioned in the document, but parameters are chosen to meet standard security levels for post-quantum cryptography.

Technical Overview

- **Mathematical Foundation:** Based on Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) codes and Niederreiter cryptosystem.
- **Key Components:**
  - **Public Key Size:** $(n_0 - 1) * p$ bits
  - **Private Key Size:** $n_0(d_v + m)d\log_2(p)e$ bits (dependent on parameters)
  - **Ciphertext Size:** Size of the syndrome vector, p bits.

Performance Characteristics

- **Speed:**
  - **Key Generation:** Efficient due to QC-LDPC structure
  - **Encryption/Signing:** Syndrome computation involves sparse matrix multiplications
  - **Decryption/Verification:** Efficient iterative decoding algorithm (Q-decoder)
- **Memory Requirements:** Optimized due to sparse matrix representation; public key size proportional to p.

Security Analysis

- **Classical Security:** Resistant to classical ISD attacks and decoding attacks based on sparse matrix representations.
- **Quantum Security:** Parameters account for potential speed-ups from Grover's algorithm in ISD.
- **Known Attack Vectors:** Reaction attacks (mitigated by ephemeral keys), decoding attacks, and low-weight codeword finding attacks.

Implementation Considerations

- **Hardware Requirements:** Efficient matrix operations can be implemented on low-end hardware.

- **Software Complexity:** Requires careful implementation of sparse matrix operations and decoding algorithms.
- **Integration Challenges:** Ensuring parameter choices meet desired security levels against both classical and quantum attacks.

Advantages and Limitations

- **Pros:**
    - Compact public keys due to QC-LDPC structure
    - Efficient decoding and encryption algorithms
    - Resistant to several known attack vectors
- **Cons:**
    - Decryption Failure Rate (DFR) needs to be carefully managed
    - Parameter selection impacts security and efficiency trade-offs

Standardization Status

- **NIST Round:** Did not advance beyond Round 1.
- **Other Standards:** None mentioned.
-

LEDApkc :

Basic Information

Family Type: Code-based
Purpose: Public-key cryptosystem for encryption
NIST Security Level: Multiple security levels proposed, depending on parameters.

Technical Overview

Mathematical Foundation: Based on low-density parity-check (LDPC) codes and McEliece cryptosystem principles.
Key Components:

- **Public Key Size:** Depends on parameters; typically large for code-based cryptosystems.
- **Private Key Size:** Smaller compared to public key due to compact representations.
- **Ciphertext Size:** Parameter-dependent; efficiency improvements noted with LDPC codes.

Performance Characteristics

Speed:

- **Key Generation:** Parameterized for efficiency; relies on LDPC principles.
- **Encryption:** Exploits LDPC decoding efficiencies.

- **Decryption:** Includes tailored decoding for LDPC structures, offering reduced computational load.

Memory Requirements: Focuses on minimizing storage through compact LDPC representation.

Security Analysis

Classical Security: Depends on the hardness of decoding linear codes.
Quantum Security: Parameter-dependent; LDPC codes are resilient under standard assumptions.
Known Attack Vectors: Primarily statistical and structural attacks on LDPC codes. Mitigated through parameter tuning.

Implementation Considerations

Hardware Requirements: Modular arithmetic with large matrices.
Software Complexity: Moderate to high due to decoding algorithms and parameterization.
Integration Challenges: Compatibility with existing systems due to large key sizes.

Advantages and Limitations

Pros:

- Compact private keys.
- Exploits LDPC efficiency for performance.

Cons:

- Large public key size.
- Vulnerable to specific QC-LDPC-related attacks if improperly parameterized.

Standardization Status

NIST Round: Did not advance beyond Round 1.
Other Standards: Not mentioned.

Lepton

Basic Information

Family Type: Learning Parity with Noise (LPN)-based

Purpose: Key Encapsulation Mechanism (KEM)

NIST Security Level: Not specified in the document

Technical Overview

Mathematical Foundation: Learning Parity with Noise (LPN) and Compact Learning Parity with Noise (CLPN)

Key Components:

- **Public Key Size:** Not explicitly mentioned
- **Private Key Size:** Not explicitly mentioned
- **Ciphertext Size:** Not explicitly mentioned

Performance Characteristics

Speed:

- **Key Generation:** Implementation details suggest ANSI C is used for reference, targeting Intel x64 processors.
- **Encryption/Signing:** Concrete cycle counts were not detailed but mention optimization potential.
- **Decryption/Verification:** Similarly, no precise cycle count provided.

Memory Requirements: Moderate, as the implementation avoids trading space for speed except for a small pre-computed file (~32.4 KB).

Security Analysis

Classical Security: Estimated classical security varies based on parameters, e.g., 103-bit to 299-bit.

Quantum Security: Estimated quantum security ranges from 51-bit to 149-bit.

Known Attack Vectors:

1. Brute-force attacks on LPN and CLPN problems
2. Best known attacks target low-noise LPN solvers

Implementation Considerations

Hardware Requirements: Compatible with Intel Core-i7 processors and 4 GB RAM (reference implementation).

Software Complexity: Moderate, with potential for further optimization.

Integration Challenges: Not explicitly mentioned.

Advantages and Limitations

Pros:

1. Based on well-studied LPN problem
2. Efficient for low-power devices

Cons:

1. Large key sizes compared to alternatives
2. Efficiency challenges for specific parameter settings

Standardization Status

NIST Round: Did not advance beyond Round 1

Other Standards: No other concurrent standards mentioned

LIMA

Basic Information

- **Family Type**: Lattice-based
- **Purpose**: Key Encapsulation Mechanism (KEM)
- **NIST Security Level**: Configurable from Levels 1 to 5 (depending on parameter sets)

Technical Overview

- **Mathematical Foundation**: Learning With Errors (LWE) problem
- **Key Components**:
  - **Public Key Size**: Ranges from 1,024 bytes to 2,048 bytes (depending on security level)
  - **Private Key Size**: Ranges from 1,024 bytes to 2,048 bytes
  - **Ciphertext Size**: Ranges from 1,024 bytes to 2,048 bytes

Performance Characteristics

- **Speed**:
  - **Key Generation**: Fast (specific benchmarks depend on the implementation platform)
  - **Encryption**: Moderate
  - **Decryption**: Moderate
- **Memory Requirements**: Requires substantial memory due to lattice operations and parameter sizes.

Security Analysis

- **Classical Security**: Equivalent to standard LWE-based cryptographic systems
- **Quantum Security**: Resistant to known quantum attacks (based on LWE hardness)

- **Known Attack Vectors**: Vulnerable to side-channel attacks if implemented improperly; relies on secure noise sampling.

Implementation Considerations

- **Hardware Requirements**: Efficient for hardware implementation; can leverage vectorized instructions for performance.
- **Software Complexity**: Medium to high due to lattice arithmetic.
- **Integration Challenges**: Requires careful parameter selection for balancing security and performance.

Advantages and Limitations

- **Pros**:
  - Configurable security levels for different applications.
  - Strong theoretical foundation in lattice-based cryptography.
- **Cons**:
  - Larger key and ciphertext sizes compared to some other KEMs.
  - Relatively slower compared to non-lattice-based schemes.

Standardization Status

- **NIST Round**: Participated in Round 1 but did not advance to Round 2.
- **Other Standards**: Not part of any concurrent standards.

Lizard

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Level 1, Level 3, and Level 5 (configurable)

Technical Overview

- **Mathematical Foundation:** Learning With Errors (LWE) problem with modulus switching for efficiency.
- **Key Components:**
  - **Public Key Size:** 1,376 bytes (Level 1), 3,072 bytes (Level 3), 4,928 bytes (Level 5)
  - **Private Key Size:** 2,048 bytes (Level 1), 4,608 bytes (Level 3), 7,424 bytes (Level 5)

    - **Ciphertext Size:** 1,760 bytes (Level 1), 3,904 bytes (Level 3), 6,272 bytes (Level 5)

Performance Characteristics

- **Speed:**
    - **Key Generation:** 2,500 operations/second
    - **Encapsulation:** 1,400 operations/second
    - **Decapsulation:** 1,500 operations/second
- **Memory Requirements:**
    - Moderate RAM usage, suitable for constrained environments.

Security Analysis

- **Classical Security:** At least 128 bits for Level 1, 192 bits for Level 3, and 256 bits for Level 5.
- **Quantum Security:** Slightly lower than classical due to Grover's algorithm but still meets NIST's required levels.
- **Known Attack Vectors:**
    - Attacks leveraging algebraic properties of LWE.
    - Side-channel attacks mitigated with countermeasures.

Implementation Considerations

- **Hardware Requirements:**
    - Minimal hardware requirements, efficient on embedded devices.
- **Software Complexity:**
    - Straightforward implementation; modulus switching adds some complexity.
- **Integration Challenges:**
    - None reported; suitable for existing protocols like TLS and VPNs.

Advantages and Limitations

- **Pros:**
    - Compact key and ciphertext sizes compared to similar lattice-based schemes.
    - Configurable security levels.
    - Efficient in both software and hardware implementations.
- **Cons:**
    - Requires careful parameter selection to avoid leakage.
    - Performance slightly slower than some competing lattice-based schemes.

Standardization Status

- **NIST Round:** Did not advance beyond Round 1.
- **Other Standards:** Not standardized elsewhere.

LOCKER

Basic Information

Family Type: Rank-based cryptography
Purpose: Key Encapsulation Mechanism (KEM)
NIST Security Level: Levels 1, 3, and 5

Technical Overview

Mathematical Foundation: Rank Syndrome Decoding (RSD) Problem
Key Components:

- **Public Key Size:** Varies (e.g., 5,893 to 12,367 bits based on security levels)
- **Private Key Size:** Not explicitly mentioned in the provided material.
- **Ciphertext Size:** Varies (e.g., 6,405 to 12,879 bits based on security levels)

Performance Characteristics

Speed:

- **Key Generation:** Approximately 1.09 ms to 10.4 ms across security levels
- **Encapsulation:** Approximately 0.22 ms to 1.49 ms across security levels
- **Decapsulation:** Approximately 1.04 ms to 6.6 ms across security levels

Memory Requirements: Not specified.

Security Analysis

Classical Security: Well-defined against the RSD and Ideal-LRPC problems.
Quantum Security: Estimated to be robust against known quantum attacks.
Known Attack Vectors: Includes combinatorial attacks and algebraic attacks.

Implementation Considerations

Hardware Requirements: Benchmarks based on Intel Core i7.
Software Complexity: Moderate, with constant-time decoding for enhanced security.
Integration Challenges: None explicitly mentioned.

Advantages and Limitations

Pros:

- Efficient in terms of both key size and computational cost.
- Strong theoretical foundation in rank-based problems.

Cons:

- Limited historical use of rank-based cryptography.

Standardization Status

NIST Round: Did not advance beyond Round 1.
Other Standards: None noted.

LOTUS

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Public Key Encryption (PKE) and Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Levels 1, 3, and 5 (128, 192, and 256-bit security)

Technical Overview

- **Mathematical Foundation:** Learning with Errors (LWE) assumption
- **Key Components:**
  - **Public Key Size:**
    - LOTUS-128: 658.95 KB
    - LOTUS-192: 1025.0 KB
    - LOTUS-256: 1471.0 KB
  - **Private Key Size:**
    - LOTUS-128: 700.42 KB
    - LOTUS-192: 1101.0 KB
    - LOTUS-256: 1590.8 KB
  - **Ciphertext Size:**
    - LOTUS-128: 1.144 KB
    - LOTUS-192: 1.456 KB

- LOTUS-256: 1.768 KB

Performance Characteristics

- **Speed (Reference Implementation on Intel Core i7-7700K):**
    - **Key Generation:**
        - LOTUS-128: 6,385.842 μs
        - LOTUS-192: 11,109.302 μs
        - LOTUS-256: 17,197.583 μs
    - **Encryption/Encapsulation:**
        - LOTUS-128: 75.299 μs
        - LOTUS-192: 105.636 μs
        - LOTUS-256: 149.075 μs
    - **Decryption/Decapsulation:**
        - LOTUS-128: 91.091 μs
        - LOTUS-192: 139.862 μs
        - LOTUS-256: 210.126 μs
- **Memory Requirements:** Not explicitly mentioned

Security Analysis

- **Classical Security:** Equivalent to AES-128, AES-192, and AES-256 for respective levels
- **Quantum Security:** Quantum-secure when hash functions are modeled as random oracles
- **Known Attack Vectors:** Limited to solving the LWE problem with specified parameters

Implementation Considerations

- **Hardware Requirements:** Can be optimized for AVX2 instruction sets
- **Software Complexity:** Moderate
- **Integration Challenges:** Key sizes are large, which might impact practicality

Advantages and Limitations

- **Pros:**
    - Strong theoretical foundation based on LWE
    - Proven IND-CCA2 security under the random oracle model
- **Cons:**
    - Large key sizes

- Cryptographic proofs rely on the random oracle model

Standardization Status

- **NIST Round:** Did not advance to Round 3 or 4
- **Other Standards:** None mentioned

LUOV

Basic Information

- **Family Type**: Multivariate-based
- **Purpose**: Digital Signatures
- **NIST Security Level**: Levels 2, 4, and 5 (depending on parameter set)

Technical Overview

- **Mathematical Foundation**: Based on the hardness of solving systems of multivariate quadratic equations over finite fields.
- **Key Components**:
  - **Public Key Size**: Varies (15.5 KB to 98.6 KB for specific parameters)
  - **Private Key Size**: 32 bytes
  - **Signature Size**: Varies (319 bytes to 4.7 KB for specific parameters)

Performance Characteristics

- **Speed**:
  - **Key Generation**: ~21M to 146M cycles (depending on security level)
  - **Signing**: ~5.87M to 216M cycles
  - **Verification**: ~4.93M to 124M cycles
- **Memory Requirements**: Minimal RAM usage, with specific memory dominated by augmented matrix storage (e.g., 4032 bytes for specific parameters).

Security Analysis

- **Classical Security**: At least 2^160 to 2^299 operations (depending on security level)
- **Quantum Security**: At least 2^146 to 2^257 operations (depending on security level)
- **Known Attack Vectors**:
  - Direct Attack
  - UOV Attack
  - Reconciliation Attack

Implementation Considerations

- **Hardware Requirements**: Efficiently implementable using simple arithmetic over finite fields.
- **Software Complexity**: Low, primarily involving SHA-3 operations and basic field arithmetic.
- **Integration Challenges**: Relatively large public key size compared to other schemes.

Advantages and Limitations

- **Pros**:
  - Small signature sizes
  - Deterministic and stateless signature generation
  - High security margins against known attacks
  - Simple arithmetic operations
  - Flexibility in parameter selection for trade-offs between key and signature sizes
- **Cons**:
  - Large public key sizes
  - Only supports digital signatures (no encryption or KEM)

Standardization Status

- **NIST Round**: Did not advance beyond Round 1.

McNie

Family Type: Code-based Cryptography
Purpose: Public-key Encryption
NIST Security Level: Levels 1-5 (128-bit, 192-bit, 256-bit)

Technical Overview

- **Mathematical Foundation:** Low Rank Parity Check (LRPC) Codes
- **Key Components:**
  - Public Key Size: Varies by parameter set; e.g., 2775 bytes for 4-quasi-cyclic LRPC at 128-bit security.
  - Private Key Size: Larger due to LRPC-based design; details vary.
  - Ciphertext Size: Proportional to message length; scales with parameter choices.

Performance Characteristics

- **Speed:**
  - Key Generation: Ranges from ~45 ms to ~288 ms depending on parameters.
  - Encryption: ~0.5 ms to ~2.94 ms based on parameter sets.

    - Decryption: ~1.17 ms to ~4.35 ms based on parameters.
- **Memory Requirements:** Efficient for chosen parameters but larger than lattice-based approaches due to code-based design.

Security Analysis

- **Classical Security:** Meets target levels based on parameter choices (128, 192, and 256 bits).
- **Quantum Security:** High resistance due to reliance on rank-metric decoding and structural attack resistance.
- **Known Attack Vectors:** Structural attacks and direct message recovery; parameters chosen to resist these.

Implementation Considerations

- **Hardware Requirements:** Tested on Intel Core i7-4790 3.60GHz (8GB RAM).
- **Software Complexity:** Moderate; employs quasi-cyclic LRPC encoding and decoding.
- **Integration Challenges:** Larger keys may impact deployment in constrained environments.

Advantages and Limitations

- **Pros:**
    - Highly secure against structural and ISD attacks.
    - Flexible design supports different security levels (128, 192, 256 bits).
    - Smaller key sizes compared to other code-based schemes.
- **Cons:**
    - LRPC decoding is probabilistic, leading to potential decryption failures.
    - Requires careful parameter optimization to balance key size and failure probability.

Standardization Status

- **NIST Round:** Advanced to Round 1 but did not progress further.

Mersenne-756839

Basic Information

Family Type: Lattice-based
Purpose: Key Encapsulation Mechanism (KEM)
NIST Security Level: Not mentioned

Technical Overview

Mathematical Foundation: Based on arithmetic modulo Mersenne numbers (i.e., numbers of the form p=2n−1p = 2^n - 1p=2n−1, where nnn is a prime). The scheme uses Hamming weights for cryptographic security.

Key Components:

- **Public Key Size**: Not explicitly stated, but includes values RRR and TTT modulo P=2n−1P = 2^n - 1P=2n−1.
- **Private Key Size**: 256 bits.
- **Ciphertext Size**: Determined by modulo operations, dependent on PPP.

Performance Characteristics

Speed:

- **Key Generation**: Includes generating random values modulo PPP and computing T=fR+gT = fR + gT=fR+g modulo PPP.
- **Encryption/Signing**: Computation involves modular arithmetic and error-correcting encoding.
- **Decryption/Verification**: Relies on decoding noisy data using error-correcting codes.

Memory Requirements: Usage depends on the size of n=756,839n = 756,839n=756,839 for modular arithmetic and error-correcting codes.

Security Analysis

Classical Security: Relies on the hardness of the Mersenne Low Hamming Combination problem.
Quantum Security: Claims resistance to Grover's algorithm and other quantum attacks, assuming h=256h = 256h=256.
Known Attack Vectors:

- Weak key attacks.
- Grover's algorithm for quadratic speedup.

Implementation Considerations

Hardware Requirements: Efficient for large number modular arithmetic, requiring optimized libraries.
Software Complexity: Utilizes pseudo-random number generators and error-correcting codes.
Integration Challenges: Ensuring efficient modular operations and minimizing decryption errors.

Advantages and Limitations

Pros:

- Simple arithmetic structure based on Mersenne primes.
- Quantum resistance via large Hamming weights.

Cons:

- Vulnerable to chosen-ciphertext attacks without additional wrappers.
- Error rates may require complex error-correcting code optimizations.

Standardization Status

NIST Round: Did not advance beyond Round 1.
Other Standards: None mentioned.

MQDSS

Basic Information

- **Family Type:** Multivariate Quadratic (MQ) Functions
- **Purpose:** Digital Signatures
- **NIST Security Level:** Not explicitly stated but evaluated for resistance against quantum and classical attacks.

Technical Overview

- **Mathematical Foundation:** Based on the Multivariate Quadratic (MQ) problem, which is considered NP-complete and believed to be quantum-resistant.
- **Key Components:**
    - **Public Key Size:** Depends on the parameter set; ranges provided in specifications.
    - **Private Key Size:** Depends on the parameter set; ranges provided in specifications.
    - **Signature Size:** Highly compact due to multivariate construction.

Performance Characteristics

- **Speed:**
    - **Key Generation:** Efficient due to reliance on matrix operations over finite fields.
    - **Signing:** Compact and fast signature generation due to underlying MQ structures.
    - **Verification:** Designed to balance efficiency with security requirements.
- **Memory Requirements:** Optimized for constrained environments, leveraging the lightweight nature of MQ operations.

Security Analysis

- **Classical Security:** Based on the hardness of solving the MQ problem.
- **Quantum Security:** Resistant to Grover's algorithm due to the exponential search space and inefficiencies in algebraic simplification for MQ systems.
- **Known Attack Vectors:**
  - Algebraic attacks like XL and F4/F5.
  - Exhaustive search and quantum-enhanced hybrid methods.

Implementation Considerations

- **Hardware Requirements:** Optimized for software implementation; low hardware complexity for embedded systems.
- **Software Complexity:** Moderate; involves matrix manipulations and Groebner basis computations.
- **Integration Challenges:** Requires thorough analysis of parameter selection to balance size, speed, and security.

Advantages and Limitations

- **Pros:**
  - Compact public and private keys.
  - Efficient and secure digital signature scheme.
- **Cons:**
  - Computational overhead for verification.
  - Limited scalability with certain parameter choices.

Standardization Status

- **NIST Round:** Did not advance beyond earlier evaluation rounds.
- **Other Standards:** Not currently adopted in other standardization efforts.

NewHope

Family Type: Lattice-based

Purpose: Key Encapsulation Mechanism (KEM)

NIST Security Level:

- NewHope512: Level 1
- NewHope1024: Level 5

Technical Overview

Mathematical Foundation: Based on the Ring-Learning With Errors (Ring-LWE) problem.

Key Components:

- **Public Key Size:**
  - NewHope512: 928 bytes
  - NewHope1024: 1824 bytes
- **Private Key Size:**
  - NewHope512: 869 bytes
  - NewHope1024: 1792 bytes
- **Ciphertext Size:**
  - NewHope512: 1088 bytes
  - NewHope1024: 2176 bytes

Performance Characteristics

Speed:

- Key Generation: Optimized for efficiency on x86 and ARM platforms.
- Encapsulation: Fast operations leveraging polynomial arithmetic.
- Decapsulation: Utilizes efficient decoding mechanisms.

Memory Requirements:

- Moderate for lattice-based schemes; specific optimizations for constrained devices.

Security Analysis

Classical Security:

- Provides high security against classical cryptanalysis.
  Quantum Security:
- Ring-LWE hardness is conjectured to be quantum-secure under standard assumptions.
  Known Attack Vectors:
- Focus on lattice reduction and hybrid lattice-based attacks.

Implementation Considerations

Hardware Requirements:

- Compatible with general-purpose processors.
  Software Complexity:
- Moderate; requires understanding of polynomial operations and FFT.
  Integration Challenges:
- Supports hybrid implementations for backward compatibility.

Advantages and Limitations

Pros:

- Strong security proofs based on RLWE.
- Smaller key sizes compared to standard LWE.

Cons:

- Larger ciphertext sizes than traditional cryptosystems.
- Relies on parameter optimization for specific hardware platforms.

Standardization Status

NIST Round:

- Advanced to Round 3.

Other Standards:

- Not concurrently standardized elsewhere.

NTRUEncrypt

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Public Key Encryption (PKE) and Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Levels 1–5 (depending on parameter set)

Technical Overview

- **Mathematical Foundation:** Hardness of lattice-based problems such as Shortest Vector Problem (SVP) and Learning With Errors (LWE) for certain variants.
- **Key Components:**
  - **Public Key Size:** Between 611 and 1023 bytes (depending on parameter set)
  - **Private Key Size:** Between 701 and 8194 bytes
  - **Ciphertext Size:** Between 611 and 4097 bytes

Performance Characteristics

- **Speed:**
    - **Key Generation:** ~440 μs for NTRU-443; ~43.5 ms for NTRU-1024
    - **Encryption:** ~82 μs for NTRU-443; ~67 ms for NTRU-1024
    - **Decryption:** ~109 μs for NTRU-443; ~115 ms for NTRU-1024
- **Memory Requirements:** Moderate, scalable with parameter sets.

Security Analysis

- **Classical Security:** Ranges from 128 to 256 bits (depending on parameter set)
- **Quantum Security:** Ranges from 84 to 198 bits
- **Known Attack Vectors:** Hybrid lattice-reduction and meet-in-the-middle attacks, sieving-based attacks, and Grover's algorithm for key search.

Implementation Considerations

- **Hardware Requirements:** Moderate computational power required, optimized for both software and embedded systems.
- **Software Complexity:** Moderate; includes several optimizations for efficiency.
- **Integration Challenges:** Requires careful parameter selection to balance security and performance.

Advantages and Limitations

- **Pros:**
    - Highly scrutinized with over two decades of cryptanalysis.
    - Compact public key and ciphertext sizes, making it suitable for constrained environments like handshake protocols.
- **Cons:**
    - Lacks provable security guarantees against worst-case lattice problems.
    - Computational performance at higher security levels can be intensive.

Standardization Status

- **NIST Round:** Advanced to Round 3
- **Other Standards:** Standardized in IEEE 1363 (2008) and ANSI X9.98 (2010).

pqNTRUSign

Basic Information

- **Family Type**: Lattice-based
- **Purpose**: Digital Signatures

- **NIST Security Level**: Multiple parameter sets, ranging from 128-bit classical security to 149-bit quantum security.

Technical Overview

- **Mathematical Foundation**: Modular lattice signature based on the NTRU lattice; integrates rejection sampling to prevent leakage of private keys.
- **Key Components**:
    - **Public Key Size**: 2065 bytes (Gaussian-1024 and Uniform-1024)
    - **Private Key Size**: 2604 bytes (Gaussian-1024 and Uniform-1024)
    - **Signature Size**: 11264 bits (Gaussian-1024); varies based on parameterization.

Performance Characteristics

- **Speed**:
    - **Key Generation**: ~48 ms
    - **Signing**: 72 ms (Uniform sampling); 120 ms (Gaussian sampling)
    - **Verification**: ~0.97 ms
- **Memory Requirements**: Standard lattice-based implementation; includes Gaussian sampler APIs.

Security Analysis

- **Classical Security**: 128–269 bits (based on parameter set)
- **Quantum Security**: 149 bits
- **Known Attack Vectors**:
    - Public key attacks using hybrid lattice reduction and meet-in-the-middle attacks.
    - Forgery attacks involving approximate closest vector problems.

Implementation Considerations

- **Hardware Requirements**: Performance gains possible via AVX2 optimizations and GPU acceleration (not included in this submission).
- **Software Complexity**: Includes support for Gaussian and uniform samplers.
- **Integration Challenges**: Suitable for classical and quantum environments; allows for parameter agility.

Advantages and Limitations

- **Pros**:
    - Small signature size due to modular lattice structure.
    - NTRU trapdoor ensures efficiency and robust cryptanalysis history.

  - Sampler agility enables balance between security and performance.
- **Cons**:
  - Signing speed could improve with efficient Gaussian samplers or Number Theoretic Transform (NTT) optimization.
  - Gaussian-based rejection sampling requires fine-tuned parameters.

Standardization Status

- **NIST Round**: Did not advance to Round 3 or 4.
- **Other Standards**: None mentioned.

NTRU-HRSS-KEM

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Designed to meet at least AES-128 security level.

Technical Overview

- **Mathematical Foundation:** Ring-based lattice construction.
- **Key Components:**
  - **Public Key Size:** 1138 bytes
  - **Private Key Size:** 1418 bytes
  - **Ciphertext Size:** 1278 bytes

Performance Characteristics

- **Speed (Reference C Implementation):**
  - **Key Generation:** 18,151,998 cycles
  - **Encapsulation:** 1,208,946 cycles
  - **Decapsulation:** 3,578,538 cycles
- **Speed (Optimized AVX2 Implementation):**
  - **Key Generation:** 294,874 cycles
  - **Encapsulation:** 38,456 cycles
  - **Decapsulation:** 68,458 cycles
- **Memory Requirements:**
  - Reference: ~11 KiB stack

    - Optimized: ~43 KiB stack

Security Analysis

- **Classical Security:** Estimated to match or exceed AES-128 security.
- **Quantum Security:** Designed to provide equivalent post-quantum security against Grover-based attacks.
- **Known Attack Vectors:** Lattice reduction attacks, including primal and hybrid attacks.

Implementation Considerations

- **Hardware Requirements:** Optimized for platforms with AVX2 support.
- **Software Complexity:** Moderate, relies on lattice arithmetic and modular reduction.
- **Integration Challenges:** High computational and memory requirements for high security.

Advantages and Limitations

- **Pros:**
    - Deterministic decryption with zero failure probability.
    - Direct KEM construction avoids padding mechanisms.
    - All secret keys are invertible.
- **Cons:**
    - Larger key and ciphertext sizes compared to some competitors.
    - Requires large modulus, increasing communication cost.

Standardization Status

- **NIST Round:** Advanced to Round 3.
- **Other Standards:** None mentioned.

NTRU Prime

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Public-key Encryption and Key Encapsulation
- **NIST Security Level:** Level 1-5 (varies by parameter set)

Technical Overview

- **Mathematical Foundation:** Integer Polynomial Ring-based Public Key Cryptosystem with a focus on avoiding ring-learning with errors (Ring-LWE) structures.
- **Key Components:**

    - **Public Key Size:** 1218 bytes (example parameter set)
    - **Private Key Size:** 1412 bytes (example parameter set)
    - **Ciphertext Size:** 1087 bytes (example parameter set)

Performance Characteristics

- **Speed:**
    - **Key Generation:** Fast and efficient; specific benchmarks are parameter-dependent.
    - **Encryption:** Moderate computational complexity.
    - **Decryption:** Efficient with low latency.
- **Memory Requirements:** Designed for constrained environments; precise RAM usage depends on implementation.

Security Analysis

- **Classical Security:** Meets 128-bit classical security.
- **Quantum Security:** Resistant to attacks leveraging quantum computing, due to the avoidance of structures vulnerable to known quantum attacks.
- **Known Attack Vectors:** Focuses on mitigating weaknesses in Ring-LWE that affect other lattice-based schemes.

Implementation Considerations

- **Hardware Requirements:** Suitable for both software and hardware implementation with low computational overhead.
- **Software Complexity:** Moderate complexity in ensuring security properties against potential cryptanalysis.
- **Integration Challenges:** Requires attention to parameter selection and compatibility with existing cryptographic protocols.

Advantages and Limitations

- **Pros:**
    - Strong resistance to both classical and quantum attacks.
    - Avoids reliance on potentially vulnerable mathematical structures (e.g., Ring-LWE).
- **Cons:**
    - Larger key and ciphertext sizes compared to some other algorithms.
    - Implementation requires careful tuning for specific applications.

Standardization Status

- **NIST Round:** Advanced to Round 3 as an Alternate Candidate.

- **Other Standards:** Not currently adopted in other cryptographic standards

NTS-KEM

Basic Information

Family Type: Code-based
Purpose: Key Encapsulation Mechanism (KEM)
NIST Security Level: Categories 1, 3, and 5

Technical Overview

Mathematical Foundation: Decoding random linear codes using binary Goppa codes
Key Components:

- **Public Key Size:**
    - Level 1: 319,488 bytes
    - Level 3: 929,760 bytes
    - Level 5: 1,419,704 bytes
- **Private Key Size:**
    - Level 1: 9,216 bytes
    - Level 3: 17,524 bytes
    - Level 5: 19,890 bytes
- **Ciphertext Size:**
    - Level 1: 1,024 bits
    - Level 3: 1,296 bits
    - Level 5: 2,024 bits

Performance Characteristics

Speed:

- Key Generation: Efficient for Goppa codes
- Encapsulation: Fast due to compact ciphertexts
- Decapsulation: Optimized using decoding algorithms for Goppa codes

Memory Requirements:
Dependent on the public and private key sizes, particularly for high-security categories.

Security Analysis

Classical Security:

- Level 1: 128-bit
- Level 3: 192-bit
- Level 5: 256-bit

Quantum Security:

- Level 1: 64-bit
- Level 3: 96-bit
- Level 5: 128-bit

Known Attack Vectors: Decoding random linear codes and distinguishing permuted Goppa codes.

Implementation Considerations

Hardware Requirements: Handles large public keys but is practical for long-term key usage.
Software Complexity: Standard decoding algorithms (e.g., Berlekamp-Massey).
Integration Challenges: Large public key size may pose challenges for constrained environments.

Advantages and Limitations

Pros:

- Long-term security against quantum attacks.
- Compact ciphertexts compared to other code-based schemes.

Cons:

- Large public key sizes.
- High memory requirements for certain implementations.

Standardization Status

NIST Round: Advanced to Round 3
Other Standards: None mentioned

Odd Manhattan

Basic Information

- **Family Type**: Lattice-based
- **Purpose**: Key Encapsulation and Encryption
- **NIST Security Level**: Unspecified in the document

Technical Overview

- **Mathematical Foundation**: Based on the Gap Shortest Vector Problem (GapSVP) and Bounded Distance Decoding (BDD) problems on lattices
- **Key Components**:
  - **Public Key Size**: Dependent on lattice dimension ddd and determinant ppp (e.g., (D−1)P(D-1)P(D−1)P bytes, where P=⌈N/8⌉ P = \lceil N / 8 \rceilP=⌈N/8⌉ for determinant p=2N−C p = 2^N - Cp=2N−C)
  - **Private Key Size**: DPDPDP bytes
  - **Ciphertext Size**: λP\lambda PλP bytes (e.g., 180,224 bytes for 21282^{128}2128-bit security)

Performance Characteristics

- **Speed**:
  - **Key Generation**: Variable (e.g., 45,696 microseconds for 21282^{128}2128-bit security)
  - **Encryption**: Variable (e.g., 16,317 microseconds for 21282^{128}2128-bit security)
  - **Decryption**: Variable (e.g., 17,701 microseconds for 21282^{128}2128-bit security)
- **Memory Requirements**: Not explicitly provided, but dependent on lattice dimension ddd and parameters

Security Analysis

- **Classical Security**: Not explicitly quantified but related to solving GapSVP and BDD problems
- **Quantum Security**: Not explicitly quantified
- **Known Attack Vectors**:
  - Lattice reduction attacks (e.g., using LLL or BKZ)
  - Hermite factor-based analysis

Implementation Considerations

- **Hardware Requirements**: Efficient modular reduction and precomputation for large dimensions
- **Software Complexity**: Requires implementation of lattice-based operations, modular arithmetic, and precomputation for key and ciphertext handling
- **Integration Challenges**: High computational cost for larger dimensions

Advantages and Limitations

- **Pros**:

- Strong theoretical foundation in lattice problems
- Supports CPA and CCA encryption

- **Cons**:
  - High computational cost for encryption and decryption
  - Large key and ciphertext sizes

Standardization Status

- **NIST Round**: Not specified in the document
- **Other Standards**: No concurrent standardization efforts mentioned

Odd Manhattan

Basic Information

- **Family Type:** Code-based
- **Purpose:** Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Categories 1, 3, and 5

Technical Overview

- **Mathematical Foundation:** Error-correcting codes in rank metric.
- **Key Components:**
  - **Public Key Size:** Instance-dependent (as per Table 2 of the document).
  - **Private Key Size:** Instance-dependent (as per Table 2).
  - **Ciphertext Size:** Instance-dependent (as per Table 2).

Performance Characteristics

- **Speed:**
  - **Key Generation:** <specific measurements from Table 4>.
  - **Encryption:** <specific measurements>.
  - **Decryption:** <specific measurements>.
- **Memory Requirements:** Not explicitly listed; platform uses minimal overhead according to known answer tests.

Security Analysis

- **Classical Security:** 128-bit (Category 1) to 256-bit (Category 5).
- **Quantum Security:** Same assumptions as classical against generic attacks.
- **Known Attack Vectors:** Rank metric algebraic and combinatorial attacks.

Implementation Considerations

- **Hardware Requirements:** Standard hardware with GCC 7.2 support.
- **Software Complexity:** Relatively optimized reference implementations, no further vectorized optimizations.
- **Integration Challenges:** Low-level implementation complexity.

Advantages and Limitations

- **Pros:**
  - Strong theoretical basis using rank metric problems.
  - Compact key sizes compared to lattice-based schemes.
- **Cons:**
  - Limited practical implementation details for non-reference platforms.
  - Susceptibility to quantum optimization of specific attacks.

Standardization Status

- **NIST Round:** Did not advance past Round 1.
- **Other Standards:** No.

Post-Quantum RSA-Encryption

Basic Information

- **Family Type:** Number-theoretic (RSA-based)
- **Purpose:** Key Encapsulation Mechanism (KEM), Public-Key Encryption (PKE), Digital Signatures
- **NIST Security Level:** Targeted Category 2 for large key sizes

Technical Overview

- **Mathematical Foundation:** Integer factorization problem
- **Key Components:**
  - **Public Key Size:** Variable based on parameter sets; for pqrsa15 ~ 512 bytes
  - **Private Key Size:** Larger due to multiple primes and auxiliary values
  - **Ciphertext Size:** Same as the public key size; additional space for signatures

Performance Characteristics

- **Speed:**
  - **Key Generation:** Ranges from ~3.5 billion cycles (pqrsa15) to much higher for larger key sizes
  - **Encryption/Signing:** Around 17 million cycles for encapsulation

    - **Decryption/Verification:** Decapsulation ~122 million cycles for pqrsa15
- **Memory Requirements:** Public key and secret key sizes grow with parameters; scalability is a challenge

Security Analysis

- **Classical Security:** Relies on RSA key sizes providing pre-quantum security (e.g., >2100 bits)
- **Quantum Security:** Designed to resist Shor's algorithm by using extremely large RSA key sizes
- **Known Attack Vectors:** Factorization (quantum and classical), small-factor vulnerabilities

Implementation Considerations

- **Hardware Requirements:** Significant computational resources for large parameters
- **Software Complexity:** Moderate; implementation closely follows traditional RSA structures
- **Integration Challenges:** Large key and ciphertext sizes pose challenges for existing systems

Advantages and Limitations

- **Pros:**
    - Combines encryption and signing in a single framework
    - High pre-quantum security levels
    - Leverages extensive RSA knowledge and infrastructure
- **Cons:**
    - Computationally expensive
    - Large key sizes and ciphertext sizes may limit practicality

Standardization Status

- **NIST Round:** Did not advance beyond Round 1
- **Other Standards:** None mentioned

Post-Quantum RSA-Signature

Basic Information

- **Family Type:** Integer Factorization-Based
- **Purpose:** Digital Signatures
- **NIST Security Level:** Level 1–5 (varies based on modulus size)

Technical Overview

- **Mathematical Foundation:** Based on the hardness of integer factorization
- **Key Components:**
    - **Public Key Size:** Variable, depends on modulus size
    - **Private Key Size:** Larger than public key, includes primes for modulus factorization
    - **Signature Size:** Depends on modulus and hash function used, typically smaller than public key

Performance Characteristics

- **Speed:**
    - **Key Generation:** Relatively slow due to prime generation and modulus computation
    - **Signing:** Moderate, involves modular exponentiation
    - **Verification:** Faster than signing, uses modular arithmetic
- **Memory Requirements:** Moderate, scales with modulus size

Security Analysis

- **Classical Security:** Comparable to RSA, requires factoring large integers
- **Quantum Security:** Vulnerable to Shor's algorithm, but designed with increased modulus sizes for post-quantum resistance
- **Known Attack Vectors:** Side-channel attacks, factoring attacks

Implementation Considerations

- **Hardware Requirements:** General-purpose processors, additional hardware for efficiency
- **Software Complexity:** Moderate, requires optimized modular arithmetic
- **Integration Challenges:** Transition from classical RSA to post-quantum RSA may require protocol adjustments

Advantages and Limitations

- **Pros:**
    - Compatible with existing RSA infrastructures
    - Smaller public keys and signatures compared to lattice-based schemes
- **Cons:**
    - Computationally intensive
    - Requires larger key sizes for quantum resistance

Standardization Status

- **NIST Round:** Did not advance beyond Round 1
- **Other Standards:** None mentioned

pqsigRM

Basic Information

Family Type: Code-based
Purpose: Digital Signature
NIST Security Level: Varies by parameter set (Levels 1, 3, and 5)

Technical Overview

Mathematical Foundation: Based on punctured Reed-Muller (RM) codes with random insertion, improving the CFS signature scheme.
Key Components:

- **Public Key Size:** Varies (e.g., 336,804 bytes for RM(4,12))
- **Private Key Size:** Varies (e.g., 1,382,118 bytes for RM(4,12))
- **Signature Size:** Depends on parameters (e.g., 260 bytes for RM(4,12))

Performance Characteristics

Speed:

- **Key Generation:** ~9.6M cycles (for RM(4,12))
- **Signing:** ~15M cycles (for RM(4,12))
- **Verification:** ~81k cycles (for RM(4,12))

Memory Requirements: Depends on parameters; large due to the size of RM codes.

Security Analysis

Classical Security: Strong against known classical attacks.
Quantum Security: Expected to resist quantum attacks due to reliance on RM codes.
Known Attack Vectors: Information set decoding, Minder-Shokrollahi, and Chizhov-Borodin attacks, mitigated by puncturing techniques.

Implementation Considerations

Hardware Requirements: Requires substantial memory and processing power due to large key and signature sizes.
Software Complexity: Moderate to high, with probabilistic signing and verification algorithms.
Integration Challenges: Key and signature size may pose challenges for real-world applications.

Advantages and Limitations

Pros:

1. Strong security (EUF-CMA secure).

2. Controllable signing time and security level via parameters.

Cons:

1. Large key and signature sizes.
2. Longer signing time compared to some alternatives.

Standardization Status

NIST Round: Not advanced beyond Round 1
Other Standards: None mentioned in the document.

QC-MDPC KEM

Basic Information

- **Family Type**: Code-Based
- **Purpose**: Key Encapsulation Mechanism (KEM)
- **NIST Security Level**: Levels vary depending on parameter selection, but designed for IND-CPA and IND-CCA2 security.

Technical Overview

- **Mathematical Foundation**: Based on Quasi-Cyclic Moderate Density Parity-Check (QC-MDPC) McEliece encryption scheme.
- **Key Components**:
  - **Public Key Size**: 4097 bytes
  - **Private Key Size**: 548 bytes (using optimized sparse representation)
  - **Ciphertext Size**: 8226 bytes

Performance Characteristics

- **Speed**:
  - **Key Generation**: ~131 million cycles
  - **Encryption/Encapsulation**: ~20 million cycles
  - **Decryption/Decapsulation**: ~230 million cycles
- **Memory Requirements**: Public key size of 4097 bytes, private key requires 548 bytes.

Security Analysis

- **Classical Security**: Built on a strong foundation with well-studied code-based cryptography principles.
- **Quantum Security**: Resistant to quantum attacks due to reliance on QC-MDPC structure.
- **Known Attack Vectors**:
    - GJS Attack (mitigated by ephemeral key usage)
    - Decoding failure attacks

Implementation Considerations

- **Hardware Requirements**: Efficiently implementable on modern hardware, optimized for QC-MDPC.
- **Software Complexity**: Moderate, relies on QC-LDPC decoding algorithms.
- **Integration Challenges**: None specified.

Advantages and Limitations

- **Pros**:
    - Relatively compact public and private keys.
    - Strong security foundation with well-researched cryptographic assumptions.
- **Cons**:
    - Slower compared to lattice-based schemes for specific applications.
    - Increased ciphertext size may not be suitable for all use cases.

Standardization Status

- **NIST Round**: Did not advance beyond Round 2.
- **Other Standards**: Not specified.

qTESLA

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Digital Signatures
- **NIST Security Levels:** qTESLA-128 (Level 1), qTESLA-192 (Level 3), qTESLA-256 (Level 5)

Technical Overview

- **Mathematical Foundation:** Decisional Ring Learning with Errors (R-LWE)

- **Key Components:**
  - **Public Key Size:**
    - qTESLA-128: 2976 bytes
    - qTESLA-192: 6176 bytes
    - qTESLA-256: 6432 bytes
  - **Private Key Size:**
    - qTESLA-128: 1856 bytes
    - qTESLA-192: 4160 bytes
    - qTESLA-256: 4128 bytes
  - **Signature Size:**
    - qTESLA-128: 2720 bytes
    - qTESLA-192: 5664 bytes
    - qTESLA-256: 5920 bytes

Performance Characteristics

- **Speed:**
  - **Key Generation:**
    - qTESLA-128: 3402K cycles
    - qTESLA-192: 5875K cycles
    - qTESLA-256: 12,433K cycles
  - **Signing:**
    - qTESLA-128: 2495K cycles
    - qTESLA-192: 9686K cycles
    - qTESLA-256: 26,063K cycles
  - **Verification:**
    - qTESLA-128: 520K cycles
    - qTESLA-192: 1065K cycles
    - qTESLA-256: 1310K cycles
- **Memory Requirements:** Moderate, suitable for embedded systems.

Security Analysis

- **Classical Security:** Matches NIST post-quantum security categories.
- **Quantum Security:** Provable tight reduction in the quantum random oracle model.

- **Known Attack Vectors:** Lattice-based attacks (e.g., BKZ, dual lattice attacks).

Implementation Considerations

- **Hardware Requirements:** Does not strictly require hardware optimization.
- **Software Complexity:** Moderate, optimized for practical implementation.

Advantages and Limitations

- **Pros:**
    - Compact signatures compared to other post-quantum schemes.
    - Provably secure under lattice-based assumptions.
    - Resistant to side-channel attacks.
- **Cons:**
    - Larger key and signature sizes compared to classical schemes like RSA or ECDSA.

Standardization Status

- **NIST Round:** Advanced to Round 3

RaCoSS

Basic Information

- **Family Type:** Code-based
- **Purpose:** Digital Signatures
- **NIST Security Level:** Category 1 (177 bits)

Technical Overview

- **Mathematical Foundation:** Null Syndrome Decoding Problem (NSDP)
- **Key Components:**
    - **Public Key Size:** 99.6 KB
    - **Private Key Size:** 703 KB
    - **Signature Size:** 0.297 KB (optimized compression)

Performance Characteristics

- **Speed:**
    - **Key Generation:** 243 ms (optimized)
    - **Signing:** 7.07 ms (optimized)
    - **Verification:** 6.87 ms (optimized)
- **Memory Requirements:** Efficient implementation on standard hardware, requiring minimal additional memory for optimized operations.

Security Analysis

- **Classical Security:** 177 bits
- **Quantum Security:** Not explicitly quantified but assumed sufficient due to reliance on NSDP.
- **Known Attack Vectors:** Vulnerable to Information Set Decoding (ISD) algorithm.

Implementation Considerations

- **Hardware Requirements:** Runs efficiently on Intel Core i7-class processors with basic hardware specifications.
- **Software Complexity:** Moderate, with ANSI C implementations provided for reference and optimized versions.
- **Integration Challenges:** Compression techniques required for key and signature size optimization.

Advantages and Limitations

Pros:

1. Provides strong existential unforgeability under chosen message attack (SEUF-CMA).
2. Compact signature size compared to most code-based alternatives.
3. Demonstrates fast signing and verification times in optimized implementations.
4. Compatible with parallel processing for further performance improvement.

Cons:

1. Key size is relatively large compared to other signature schemes.
2. Security depends on the hardness of NSDP, which has multiple known solutions.

Standardization Status

- **NIST Round:** Did not advance beyond initial submission.
- **Other Standards:** No concurrent standardization efforts mentioned.

Rainbow

Basic Information

- **Family Type**: Multivariate Cryptography, SingleField schemes
- **Purpose**: Digital Signatures
- **NIST Security Level**: I-VI (depending on parameter set)

Technical Overview

- **Mathematical Foundation**: Hash-and-Sign scheme with multivariate quadratic equations (MQ)
- **Key Components**:

- **Public Key Size**: 148.5 kB (smallest) to 1,683.3 kB (largest)
- **Private Key Size**: 97.9 kB (smallest) to 1,244.4 kB (largest)
- **Signature Size**: 512 bits (smallest) to 1,632 bits (largest)

Performance Characteristics

- **Speed**:
  - **Key Generation**: Varies by parameter set (328 ms to 13,655 ms depending on security level)
  - **Encryption/Signing**: Signature generation time from 23 μs to 1.76 ms
  - **Decryption/Verification**: Verification time from 8 μs to 3.40 ms
- **Memory Requirements**: 3 MB to 10 MB RAM for most operations

Security Analysis

- **Classical Security**: Up to 274-bit security depending on parameter set
- **Quantum Security**: Ranges due to vulnerabilities to Grover's algorithm
- **Known Attack Vectors**: Direct algebraic attacks, MinRank attack, HighRank attack, UOV attack, Rainbow-Band-Separation attack

Implementation Considerations

- **Hardware Requirements**: Efficient on low-cost devices with modest computational capabilities
- **Software Complexity**: Easy to implement due to simplicity of design
- **Integration Challenges**: Large key sizes may pose challenges in constrained environments

Advantages and Limitations

- **Pros**:
  - Extremely fast signature generation
  - Relatively small signature sizes compared to other post-quantum schemes
  - Simple and efficient design
  - Resistant to many classical and quantum attack vectors
- **Cons**:
  - Extremely large public and private key sizes
  - Security assumptions rely on heuristic resistance to known attacks

Standardization Status

- **NIST Round**: Advanced to Round 3 for Digital Signatures

Ramstake

Basic Information

- **Family Type**: Lattice-based
- **Purpose**: Key Encapsulation Mechanism (KEM)
- **NIST Security Level**: Level 1, Level 3, Level 5 (depending on parameter set)

Technical Overview

- **Mathematical Foundation**: Based on the hardness of the Ring-Learning with Errors (R-LWE) problem.
- **Key Components**:
    - **Public Key Size**: 1,024 bytes
    - **Private Key Size**: 1,024 bytes
    - **Ciphertext Size**: 1,024 bytes

Performance Characteristics

- **Speed**:
    - **Key Generation**: ~10,000 operations/second
    - **Encryption**: ~8,000 operations/second
    - **Decryption**: ~8,000 operations/second
- **Memory Requirements**: ~16 KB RAM for implementation

Security Analysis

- **Classical Security**: Provides 128-bit, 192-bit, and 256-bit security levels depending on the parameter set.
- **Quantum Security**: Equivalent to classical security levels, leveraging the difficulty of R-LWE in both classical and quantum scenarios.
- **Known Attack Vectors**:
    - Lattice reduction attacks
    - Side-channel vulnerabilities in specific implementations

---

Implementation Considerations

- **Hardware Requirements**: Suitable for resource-constrained devices due to low computational complexity.
- **Software Complexity**: Moderate; requires efficient polynomial arithmetic.
- **Integration Challenges**: Minimal, but care must be taken to handle side-channel resistance in hardware implementations.

Advantages and Limitations

- **Pros**:
    1. Efficient for both key generation and encryption/decryption.
    2. Relatively compact key sizes for lattice-based cryptography.
- **Cons**:
    1. Ciphertext size is relatively large compared to other post-quantum algorithms.
    2. Requires careful parameter selection to balance security and performance.

Standardization Status

- **NIST Round**: Did not advance beyond Round 1.

RLCE-KEM

Basic Information

- **Family Type:** Code-based
- **Purpose:** Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Equivalent to AES-128 and AES-192

Technical Overview

- **Mathematical Foundation:** Decoding random linear codes (NP-hard problem)
- **Key Components:**
    - **Public Key Size:**
        - AES-128 Security: 110 KB
        - AES-192 Security: 280 KB
    - **Private Key Size:** Not explicitly mentioned
    - **Ciphertext Size:**
        - AES-128 Security: 785 bytes
        - AES-192 Security: 1238 bytes

Performance Characteristics

- **Speed:**
  - Optimized for Reed-Solomon hardware decoders and vector instructions
  - Encryption and decryption are efficient due to reliance on well-established Reed-Solomon codes
- **Memory Requirements:** Supports low-power and constrained environments like 8-bit processors and satellite applications

Security Analysis

- **Classical Security:** Relies on the NP-hardness of decoding random linear codes
- **Quantum Security:** Believed to resist quantum attacks; detailed security analysis in "Appendix A" of the RLCE documentation
- **Known Attack Vectors:** The algorithm does not rely on structured codes, which reduces vulnerabilities associated with structural assumptions.

Implementation Considerations

- **Hardware Requirements:**
  - Can leverage Reed-Solomon hardware decoders
  - Suitable for constrained environments like smartcards
- **Software Complexity:** Efficient due to reliance on existing Reed-Solomon code implementations
- **Integration Challenges:** Large public key sizes may pose challenges in resource-constrained environments

Advantages and Limitations

- **Pros:**
  1. Smaller public keys compared to Goppa code-based McEliece schemes
  2. Based on widely deployed Reed-Solomon codes with industry experience
  3. Highly efficient encryption and decryption processes
  4. Does not rely on specific structure assumptions for security
  5. Compatible with low-power and constrained environments
- **Cons:**
  1. Large public key sizes (e.g., 110 KB for AES-128 security)
  2. Ciphertexts are relatively large compared to some other schemes

Standardization Status

- **NIST Round:** Not explicitly mentioned as advancing beyond Round 1 in provided content.

RQC

Basic Information

Family Type: Code-based
Purpose: Key Encapsulation Mechanism (KEM)
NIST Security Level: Categories 1, 3, and 5

Technical Overview

Mathematical Foundation: Syndrome Decoding Problem on Rank Codes
Key Components:

- **Public Key Size:** 1,491 bytes (Category 1), 2,741 bytes (Category 3), 3,510 bytes (Category 5)
- **Private Key Size:** 1,491 bytes (Category 1), 2,741 bytes (Category 3), 3,510 bytes (Category 5)
- **Ciphertext Size:** 1,555 bytes (Category 1), 2,805 bytes (Category 3), 3,574 bytes (Category 5)

Performance Characteristics

Speed:

- **Key Generation:** 0.23 ms (Category 1), 0.52 ms (Category 3), 0.83 ms (Category 5)
- **Encryption:** 0.58 ms (Category 1), 1.65 ms (Category 3), 1.90 ms (Category 5)
- **Decryption:** 1.56 ms (Category 1), 4.25 ms (Category 3), 5.29 ms (Category 5)

Memory Requirements: Benchmarks performed on a 32 GB memory machine

Security Analysis

Classical Security: Up to 256 bits (Category 5)
Quantum Security: Reduces quantum attacks based on rank metric problems
Known Attack Vectors:

- Combinatorial Attacks
- Algebraic Attacks using Groebner basis

Implementation Considerations

Hardware Requirements: Efficient with no specific hardware dependencies
Software Complexity: Moderate; uses coding theory techniques
Integration Challenges: None reported

Advantages and Limitations

Pros:

1. Tight security reduction to well-known rank metric problems
2. No decryption failure

3. Compact key sizes compared to many code-based systems

Cons:

1. Relatively slower encryption and decryption compared to lattice-based systems
2. Requires a strong understanding of coding theory for implementation

Standardization Status

NIST Round: Advanced to Round 2
Other Standards: None reported

Round2

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Key Encapsulation Mechanism (KEM) and Public Key Encryption (PKE)
- **NIST Security Level:** Configurable to Levels 1, 3, and 5

Technical Overview

- **Mathematical Foundation:** General Learning with Rounding (GLWR) Problem
- **Key Components:**
  - **Public Key Size:** Depends on the parameter set; typically a few kilobytes
  - **Private Key Size:** Similar to public key size
  - **Ciphertext Size:** Configurable based on security level and parameters

Performance Characteristics

- **Speed:**
  - **Key Generation:** Optimized for both RLWR and LWR settings
  - **Encryption/Signing:** Fast polynomial operations with configurable performance
  - **Decryption/Verification:** Efficient decryption with support for various configurations
- **Memory Requirements:** Lightweight, designed for constrained environments

Security Analysis

- **Classical Security:** Hardness based on GLWR
- **Quantum Security:** Proven reductions from LWE and RLWE under certain constraints
- **Known Attack Vectors:** Lattice-based attacks (e.g., primal, dual, hybrid attacks)

Implementation Considerations

- **Hardware Requirements:** Efficiently implementable on modern CPUs
- **Software Complexity:** Moderate, with support for common cryptographic libraries

- **Integration Challenges:** None reported; designed to be deployable in real-world systems

Advantages and Limitations

- **Pros:**
  - Unified design supporting both RLWR and LWR
  - Adaptable parameters for optimized security and performance
  - Lightweight and efficient for constrained devices
- **Cons:**
  - Performance sensitive to parameter choices
  - Security assumptions depend heavily on the hardness of underlying lattice problems

Standardization Status

- **NIST Round:** Advanced to Round 2 (did not reach Round 3 or 4)
- **Other Standards:** Not mentioned

SABER

Basic Information

Family Type: Lattice-based
Purpose: Key Encapsulation Mechanism (KEM)
NIST Security Level: Levels 1, 3, and 5

Technical Overview

Mathematical Foundation: Module Learning With Rounding (Mod-LWR) problem

Key Components:

- **Public Key Size**: 672 bytes (LightSaber), 992 bytes (Saber), 1312 bytes (FireSaber)
- **Private Key Size**: 1568 bytes (LightSaber), 2304 bytes (Saber), 3040 bytes (FireSaber)
- **Ciphertext Size**: 736 bytes (LightSaber), 1088 bytes (Saber), 1472 bytes (FireSaber)

Performance Characteristics

Speed:

- **Key Generation**: ~105,881 cycles (LightSaber), ~216,597 cycles (Saber), ~360,539 cycles (FireSaber)
- **Encapsulation**: ~155,131 cycles (LightSaber), ~267,841 cycles (Saber), ~400,817 cycles (FireSaber)
- **Decapsulation**: ~179,415 cycles (LightSaber), ~318,785 cycles (Saber), ~472,366 cycles (FireSaber)

Memory Requirements:

- Compact implementation avoids modular reduction, using lightweight polynomial arithmetic.

Security Analysis

Classical Security:

- LightSaber: 115 bits
- Saber: 180 bits
- FireSaber: 245 bits

Quantum Security:

- Slightly reduced due to Grover's algorithm but remains aligned with target levels.

Known Attack Vectors:

- Primal and dual lattice reduction attacks using BKZ algorithm.

Implementation Considerations

Hardware Requirements:

- Designed for efficiency on constrained devices.
- Polynomial arithmetic implemented using Karatsuba and Toom-Cook multiplication.

Software Complexity:

- Relatively simple due to absence of modular reduction.

Integration Challenges:

- None significant; designed for ease of use across diverse platforms.

Advantages and Limitations

Pros:

- Low randomness and bandwidth requirements.
- Scalable security levels using modular structure.

Cons:

- Limited to encryption and KEM, no signature scheme provided.

Standardization Status

NIST Round: Advanced to Round 3

SIKE

Basic Information

Family Type: Supersingular Isogeny-based Cryptography
Purpose: Key Encapsulation Mechanism (KEM)
NIST Security Level: Levels 1, 3, and 5 (depending on parameter set)

Technical Overview

Mathematical Foundation: Supersingular isogenies over elliptic curves
Key Components:

- **Public Key Size**: 330–564 bytes (depending on parameter set)
- **Private Key Size**: 350–610 bytes (depending on parameter set)
- **Ciphertext Size**: 346–596 bytes (depending on parameter set)

Performance Characteristics

Speed:

- **Key Generation**: Computationally expensive due to elliptic curve isogeny calculations
- **Encapsulation**: Moderate speed
- **Decapsulation**: Moderate speed

Memory Requirements: Optimized for compactness; RAM usage depends on the implementation platform and optimization.

Security Analysis

Classical Security: Comparable to standard cryptographic schemes (112–256 bits of classical security depending on parameters).
Quantum Security: Designed to resist attacks by quantum computers, achieving ~128 bits of security at Level 1.
Known Attack Vectors: Vulnerable to side-channel and active fault injection attacks without countermeasures.

Implementation Considerations

Hardware Requirements: Efficient for embedded systems; supports lightweight implementations.
Software Complexity: Medium complexity due to the mathematical operations involved.
Integration Challenges: Requires a deep understanding of elliptic curve arithmetic and isogeny-based operations.

Advantages and Limitations

Pros:

- Extremely compact key and ciphertext sizes.
- Proven security under isogeny problems.

Cons:

- High computational cost compared to lattice-based cryptography.
- Vulnerable to side-channel attacks without specific protections.

Standardization Status

NIST Round: Advanced to Round 4.
Other Standards: No concurrent standards.

SPHINCS+

Basic Information
Family Type: Hash-based
Purpose: Digital Signatures
NIST Security Level: Supports Levels 1, 3, and 5

Technical Overview

Mathematical Foundation: Hash-based cryptographic signatures using hypertree and few-time signatures (FTS) with WOTS+ and FORS components.

Key Components:

- **Public Key Size:** 32 bytes
- **Private Key Size:** 64 bytes
- **Signature Size:** Typically larger due to tree authentication; approximately 10s of kilobytes, depending on parameter sets.

Performance Characteristics

Speed:

- **Key Generation:** Medium-speed due to hash tree construction.
- **Signing:** Slow compared to lattice-based signatures; involves tree hashing.
- **Verification:** Medium-speed; requires path traversal in Merkle trees.

Memory Requirements: Medium to high; depends on the tree height and the hash size.

Security Analysis

Classical Security: 256 bits for Level 5
Quantum Security: ~128 bits for Level 5
Known Attack Vectors: Classical cryptanalysis focuses on hash collision and pre-image resistance; quantum attacks leverage Grover's algorithm.

Implementation Considerations

Hardware Requirements: Suitable for environments supporting hash functions like SHA256 or SHAKE256.
Software Complexity: Moderate to high due to tree management and hashing strategies.
Integration Challenges: Large signature sizes may hinder applications requiring bandwidth efficiency.

Advantages and Limitations

Pros:

- Stateless design simplifies implementation.
- High security confidence due to reliance on hash functions.

Cons:

- Large signature sizes.
- Slower signing times compared to lattice-based schemes.

Standardization Status

NIST Round: Advanced to Round 3.
Other Standards: None mentioned.

ThreeBears

Basic Information

- **Family Type:** Lattice-based
- **Purpose:** Key Encapsulation Mechanism (KEM)
- **NIST Security Level:** Ranges from Class II to Class V, depending on parameters.

Technical Overview

- **Mathematical Foundation:** Integer Module Learning With Errors (I-MLWE)
- **Key Components:**
  - **Public Key Size:** Approximately 804 to 1584 bytes, depending on variant.
  - **Private Key Size:** 40 bytes.
  - **Ciphertext Size:** Approximately 917 to 1697 bytes, depending on variant.

Performance Characteristics

- **Speed:**
  - Key Generation: Fast; uses fast matrix operations.
  - Encryption/Signing: Efficient multiplication routines.
  - Decryption/Verification: Comparable to lattice-based alternatives.
- **Memory Requirements:** Minimal, suited for constrained environments.

Security Analysis

- **Classical Security:** Up to 355 bits of effort for the highest parameter sets.
- **Quantum Security:** Up to 322 bits of effort for the highest parameter sets.
- **Known Attack Vectors:**
    - Lattice attacks.
    - Brute force for key or ciphertext guessing.
    - Hybrid lattice-reduction and meet-in-the-middle attacks.

Implementation Considerations

- **Hardware Requirements:** Compatible with existing big-integer libraries; optimal for platforms with fast integer arithmetic.
- **Software Complexity:** Straightforward implementation with support for error correction.
- **Integration Challenges:** None significant; supports modern cryptographic primitives.

Advantages and Limitations

Pros:

1. Simple design and efficient computation.
2. Compact key sizes relative to classical systems.
3. Suited for lightweight devices and embedded systems.

Cons:

1. Relatively novel; I-MLWE problem is less studied than other lattice approaches.
2. Lower noise may expose vulnerabilities to specific attacks.
3. Limited to encryption and KEM; no signature scheme included.

Standardization Status

- **NIST Round:** Did not advance beyond Round 2.

Titanium

Basic Information
Family Type: Lattice-based
Purpose: Public-key encryption and Key Encapsulation Mechanism (KEM)
NIST Security Level: Category 1–5 (depending on parameter sets)

Technical Overview
Mathematical Foundation: Middle-Product Learning with Errors (MP-LWE) problem
Key Components:

- **Public Key Size:** 14,720 bytes (Titanium-CPA, Std128 parameter set)
- **Private Key Size:** 32 bytes
- **Ciphertext Size:** 3,520 bytes

Performance Characteristics
Speed:

- **Key Generation:** ~1,981,835 cycles
- **Encryption:** ~1,508,258 cycles
- **Decryption:** ~261,583 cycles

Memory Requirements: Moderate (optimized for AVX2 instructions and fast NTT computations)

Security Analysis
Classical Security: Up to 149 bits (conservative estimates for Std128 parameter set)
Quantum Security: Up to 136 bits
Known Attack Vectors:

- Lattice reduction algorithms (e.g., BKZ)
- Potential weaknesses in polynomial families

Implementation Considerations
Hardware Requirements: Efficient implementations on modern CPUs using AVX2 instructions
Software Complexity: Moderate, with optimizations for fast NTT computations
Integration Challenges: Ciphertext compression and constant-time implementation ensure robustness against side-channel attacks

---

Advantages and Limitations
Pros:

- Tight security reduction from MP-LWE problem
- Efficient parameter sets optimized for security and performance
- Flexibility in choosing dimensions without constraints like power-of-2 cyclotomics

Cons:

- Larger ciphertexts compared to other lattice-based schemes like Kyber
- Higher computational cost for decryption than RLWE-based schemes

Standardization Status
NIST Round: Advanced to Round 2
Other Standards: No other concurrent standards mentioned

WalnutDSA

Basic Information

- **Family Type**: Group-Theoretic Cryptography
- **Purpose**: Digital Signatures
- **NIST Security Level**: Not specified explicitly; designed for constrained devices.

Technical Overview

- **Mathematical Foundation**: Based on the difficulty of reversing E-Multiplication, a group-theoretic one-way function derived from braid groups.
- **Key Components**:
    - **Public Key Size**: Not explicitly mentioned; depends on the braid group and field parameters.
    - **Private Key Size**: Not explicitly mentioned; consists of two braid elements.
    - **Signature Size**: Compact; exact size depends on implementation.

Performance Characteristics

- **Speed**:
    - **Key Generation**: Efficient for constrained devices.
    - **Signing**: Extremely fast, suitable for low-power environments.
    - **Verification**: Optimized for constrained devices, outperforming ECDSA.
- **Memory Requirements**: Designed to operate with minimal memory, making it ideal for devices with limited resources.

Security Analysis

- **Classical Security**: Resistant to classical attacks based on group-theoretic hardness assumptions.

- **Quantum Security**: Claims quantum resistance through the non-abelian group problem.
- **Known Attack Vectors**:
    - Potential vulnerabilities in certain parameter configurations, as discussed in cryptographic analyses.

Implementation Considerations

- **Hardware Requirements**: Minimal; designed for constrained devices like 8-bit or 16-bit microcontrollers.
- **Software Complexity**: Moderate; requires careful implementation of braid group operations.
- **Integration Challenges**: None noted; suitable for devices where traditional algorithms fail.

Advantages and Limitations

Pros:

1. Optimized for constrained devices.
2. Claims quantum resistance.

Cons:

1. Relatively new; less scrutiny compared to established algorithms.
2. Some theoretical vulnerabilities under specific conditions.

Standardization Status

- **NIST Round**: Did not progress to Round 3.
- **Other Standards**: None mentioned.

Comparison

BIG QUAKE	Code-based	Quasi-Cyclic	104 KB -	104 KB - 254 KB	Efficient for embedded systems; key gen	Strong	Optimized for embedded	Large public key size

		Goppa codes	254 KB		involves Goppa polynomials		systems; reduced key size from classical	
BIKE	Code-based	QC-MDPC decoding	2.5 KB - 8 KB	2.5 KB - 8 KB	Efficient encapsulation; moderate decapsulation cycle counts	Strong	Compact ciphertext and key sizes, efficient bit-flipping decoding	Decapsulation latency higher than encapsulation
CFPKM	Multivariate	PoSSo with Noise problem	696 bytes	729 bytes	Moderate computational time; efficient memory usage	Moderate	Smaller keys and communication sizes compared to lattice-based	Complexity in solving noisy polynomial systems
Classic McEliece	Code-based	Binary Goppa codes	261 KB - 1.3 MB	128 - 240 bytes	Large space requirements for keys; efficient ciphertext processing	Strong	Well-studied, small ciphertexts, high resistance	Very large public key size; complex key management
Compact LWE	Lattice-based	Learning with Errors problem	~2 KB	~36 bytes	Lightweight design; deterministic correctness	Moderate	Suitable for constrained environments	Public key size relatively large for lightweight applications
CRYSTALS-DILITHIUM	Lattice-based	Module-LWE, MSIS	~1.3 - 2.6 KB	~2.4 - 4.5 KB	Deterministic signing; scalable security levels	Strong	Compact keys, fast verification	Larger signature sizes
CRYSTALS-KYBER	Lattice-based	Module-LWE	~1.5 KB	~1 KB	Efficient on hardware/software; scalable security levels	Strong	High efficiency, compact keys, IND-	Moderate ciphertext size

							CCA2 security	
DAGS	Code-based	Syndrome Decoding Problem	6 KB - 11 KB	552 - 1,616 bytes	Efficient decapsulation; large private key size	Strong	IND-CCA security, efficient decapsulation	Very large private key sizes
Ding Key Exchange	Lattice-based	Ring-LWE	Not specified	Not specified	Efficient rounding/reconciliation methods	Moderate	Reduces communication costs	Communication costs still larger compared to some alternatives
DME	Multivariate	Double matrix exponentiation	~1 KB	~18 bytes	Efficient for hardware/software implementations	Moderate	Compact ciphertext and private key sizes	Limited analysis of resistance to structural attacks
DRS	Lattice-based	Guaranteed Distance Decoding	Variable	Variable	Efficient signing, verification with tunable parameters	Strong	Digital signatures with flexible parameters	Computational overhead for large matrix operations
DualModeMS	Multivariate	HFEv schemes	~18 MB	32 KB - 149 KB	Small public key size; fast signing	Moderate	Comprehensive security analysis	Large signature sizes, computationally expensive key generation
Edon-K	Hash-based	Hash functions	Not specified	Not specified	Secure hash function design	Strong	Compact and efficient hashing	Limited applicability to certain cryptographic systems

EMBLEM and R.EMBLEM	Code-based	Random linear codes	74 KB - 147 KB	128 - 256 bytes	Efficient encryption/decapsulation; practical for constrained devices	Strong	Compact ciphertext sizes; optimized for resource-constrained systems	Moderate quantum resistance
FALCON	Lattice-based	NTRU lattices	~1 KB	~700 - 1,280 bytes	Compact keys and signatures; fast verification	Strong	Highly compact signatures, versatile	Complex implementation, floating-point arithmetic
FrodoKEM	Lattice-based	Standard LWE	9 KB - 15 KB	9 KB - 15 KB	High memory usage; conservative parameters	Strong	Easy implementation, no reliance on algebraic structures	Larger keys and ciphertext sizes compared to Ring-LWE schemes
GeMSS	Multivariate	HFE polynomials	Large (varies)	Very short	Fast verification; flexible parameters	Moderate	Extremely short signatures	Large public key sizes
Giophantus	Algebraic Surface	Indeterminate equation problems	Not specified	Not specified	Efficient for encryption using bivariate polynomials	Moderate	Robustness from solving indeterminate equations	Vulnerability to lattice attacks if parameters are not carefully chosen
Gravity-SPHINCS	Hash-based	Hash-based signature schemes	Not specified	~20 - 30 KB	Stateless design; optimized for security	Strong	High assurance based on hash properties	Large signature sizes
Guess Again	Probabilistic	Random walks, interval guessing	~18,000 bits	~18,000 bits	Secure against computationally unbounded adversaries	Strong	Security not reliant on computational	Extremely high ciphertext

							assumptions	expansion factor
Gui	Multivariate	HFEv signature scheme	416 KB - 5.8 MB	360 - 664 bits	High-speed signing, fast verification	Strong	Balanced security-efficiency trade-off	Large public key size, computationally expensive key generation
HILA5	Lattice-based	Ring-LWE	1.8 KB	2 KB	Low decryption failure rate, efficient performance	Strong	Suitable for hardware/software platforms	Larger ciphertext size compared to alternatives
HiMQ-3	Multivariate	MQ problem with layers	Variable	Variable	High-speed signing; lightweight implementation	Moderate	Reduced key sizes	Complex parameter tuning
HK17	Hash-based	Sponge hash functions	Not specified	Not specified	Efficient hashing; suitable for compact systems	Moderate	Stateless design ensures simplicity	Limited analysis available
HQC	Code-based	Quasi-Cyclic Syndrome Decoding	5 KB - 7 KB	5.6 KB - 14 KB	Small public key size compared to alternatives	Strong	Compact key size for code-based scheme	Higher decryption failure rate
KCL	Lattice-based	Compact Learning with Errors	Not specified	Not specified	Lightweight encryption, efficient reconciliation methods	Moderate	Minimal computational overhead	Requires detailed parameter analysis for robustness
KINDI	Lattice-based	Module-LWE	1.2 KB - 2.4 KB	1.8 KB - 3.3 KB	Scalable for security levels, low failure rates	Strong	Compact keys and ciphertexts	Higher decryption times for advanced

								security levels
LAC	Lattice-based	Polynomial-LWE	0.5 KB - 1 KB	1 KB - 2 KB	High efficiency with small key sizes	Strong	Resistant to classical and quantum attacks	Requires AVX2 for optimal performance
LAKE	Lattice-based	LWE	966 B - 1.9 KB	1 KB - 2.3 KB	Compact key and ciphertext sizes	Strong	Optimized for constrained environments	Performance overhead for parameter generation
LEDAkem	Code-based	QC-LDPC codes	Variable	Variable	Efficient decoding; optimized for embedded systems	Strong	Compact public keys	Requires careful parameter selection to manage decryption failure
LEDApkc	Code-based	LDPC codes	Variable	Variable	Efficient for encryption/decryption	Moderate	Compact private keys	Large public key size
Lepton	LPN-based	Compact Learning Parity with Noise	Variable	Variable	Efficient for low-power devices	Moderate	Lightweight design for constrained environments	Key sizes larger than alternatives
LIMA	Lattice-based	LWE	1 KB - 2 KB	1 KB - 2 KB	Moderate encryption/decryption speeds	Strong	Configurable security levels	Larger key and ciphertext sizes compared to some schemes
Lizard	Lattice-based	LWE with modulus switching	1.4 KB - 4.9 KB	1.7 KB - 6.3 KB	Efficient for software/hardware implementations	Strong	Configurable security levels, compact parameters	Performance slower compared to similar lattice-

								based systems
LOCKER	Rank-based	Rank Syndrome Decoding	5.9 KB - 12 KB	6.4 KB - 13 KB	Efficient for constrained environments	Strong	Strong theoretical foundation in rank-based problems	Limited historical usage of rank-based cryptography
LOTUS	Lattice-based	LWE	~660 KB - 1.5 MB	1.1 KB - 1.8 KB	Efficient for encryption and decapsulation	Strong	Proven security under LWE assumptions	Larger key sizes
LUOV	Multivariate	Solving quadratic equations	~16 KB - 99 KB	~300 B - 5 KB	Small signature sizes	Moderate	Stateless, deterministic signing	Larger public key sizes
McNie	Code-based	Low Rank Parity Check Codes	~3 KB - 9 KB	Proportional	Flexible design supports various security levels	Strong	Highly secure against structural and ISD attacks	Probabilistic decoding introduces decryption failures
Mersenne-756839	Lattice-based	Mersenne Low Hamming Problem	Variable	Variable	Simple arithmetic structure using Mersenne primes	Moderate	Quantum resistance via large Hamming weights	Vulnerability to chosen-ciphertext attacks without protection
MQDSS	Multivariate	MQ problem	Variable	Variable	Compact public/private keys; efficient signature schemes	Strong	Secure against classical and quantum attacks	Limited scalability with certain parameter choices
NewHope	Lattice-based	Ring-LWE	928 B - 1.8 KB	1 KB - 2 KB	High efficiency on x86/ARM platforms	Strong	Smaller key sizes compared to standard	Relies on parameter optimization

							LWE schemes	
NTRUEncrypt	Lattice-based	NTRU lattice problems	~600 B - 1 KB	~600 B - 4 KB	Efficient for constrained environments	Strong	Compact public key and ciphertext sizes	Computational performance at higher security levels
pqNTRUSign	Lattice-based	Modular NTRU lattice problems	2 KB	~11 KB	Efficient signing; robust against attacks	Strong	Compact signature sizes	Rejection sampling requires fine-tuned parameters
NTRU-HRSS-KEM	Lattice-based	Ring-based lattice	~1 KB	~1 KB	Deterministic decryption; optimized for low failure probability	Strong	Efficient for hardware/software platforms	Larger modulus increases communication cost
NTRU Prime	Lattice-based	Integer Polynomial Ring	~1 KB	~1 KB	Efficient encryption and decryption	Strong	Avoids vulnerable mathematical structures	Larger key sizes compared to some lattice-based schemes
NTS-KEM	Code-based	Binary Goppa codes	~300 KB - 1.4 MB	1 KB - 2 KB	Fast encapsulation with compact ciphertexts	Strong	Long-term security against quantum attacks	Large public key size
Odd Manhattan	Lattice-based	Gap Shortest Vector Problem	Variable	Variable	Strong theoretical basis	Moderate	Supports CPA and CCA encryption	High computational cost for larger dimensions
Ouroboros-R	Blockchain-based	Proof-of-Stake	Variable	Not applicable	Highly scalable for blockchain networks	Strong	Energy-efficient,	Applicability limited to blockchai

		consensus					secure consensus	n-related use cases
Picnic	Symmetric-key	Symmetric primitives, MPC	Variable	Variable	Resistant to classical and quantum attacks	Strong	Modular design, flexible parameterization	Large signature sizes
Post-quantum RSA-Encryption	Number-theoretic	Integer factorization problem	Variable	Variable	High pre-quantum security levels	Moderate	Compatible with existing RSA infrastructure	Requires extremely large key sizes for quantum resistance
Post-quantum RSA-Signature	Number-theoretic	Integer factorization problem	Variable	Variable	Moderate signing and verification times	Moderate	Smaller public keys compared to lattice schemes	Computationally intensive
pqsigRM	Code-based	Reed-Muller codes	~336 KB	~260 B	Efficient verification; fast signing	Strong	Compact signature sizes	Large public key sizes
QC-MDPC KEM	Code-based	QC-MDPC McEliece scheme	~5 KB	~8 KB	Efficient decapsulation, moderate encryption speeds	Strong	Compact keys and ciphertext sizes	Increased ciphertext size
qTESLA	Lattice-based	Decisional Ring-LWE	2 KB - 6 KB	2.7 KB - 5.9 KB	Compact signatures; efficient signing and verification	Strong	Resistant to side-channel attacks	Larger key sizes compared to classical schemes
RaCoSS	Code-based	Null Syndrome Decoding Problem	~700 KB	~300 B	Fast signing and verification	Strong	Compact signature sizes	Relatively large key sizes
Rainbow	Multivariate	Multivariate	100 KB -	512 - 1,632 bits	Extremely fast signing; compact signatures	Moderate	Resistant to many classical and	Extremely large public and

		quadratic problem	1.7 MB				quantum attack vectors	private key sizes
Ramstake	Lattice-based	Ring-LWE	~1 KB	~1 KB	Efficient for constrained devices	Strong	Relatively compact key sizes	Larger ciphertext sizes compared to alternatives
RankSign	Rank-based	Rank syndrome decoding	Variable	Variable	High security from rank-based assumptions	Strong	Efficient implementations for digital signatures	Limited adoption
RLCE-KEM	Code-based	Reed-Solomon codes	~100 KB - 300 KB	~800 B - 1.2 KB	Fast encryption/decryption; widely deployable	Strong	Smaller public keys than Goppa-based schemes	Larger ciphertexts compared to some schemes
Round2	Lattice-based	General Learning with Rounding	Variable	Variable	Lightweight; designed for constrained environments	Strong	Unified design for various lattice problems	Performance sensitive to parameter choices
RQC	Code-based	Syndrome Decoding Problem	1.5 KB - 3.5 KB	1.5 KB - 3.5 KB	Compact keys; efficient encryption and decryption	Strong	No decryption failure	Relatively slower compared to lattice-based systems
RVB	Blockchain-based	Randomized Voting Blockchain	Variable	Not applicable	Efficient and transparent voting mechanisms	Strong	Suitable for secure voting platforms	Applicability limited to voting systems
SABER	Lattice-based	Module-LWR	~1 KB	~1 KB	Low randomness and bandwidth requirements	Strong	Scalable security levels using	Limited to encryption and KEM, no

							modular structure	signature scheme provided
SIKE	Isogeny-based	Supersingular isogeny problem	~500 B	~500 B	Extremely compact key and ciphertext sizes	Moderate	Suitable for lightweight devices	High computational cost
SPHINCS+	Hash-based	Hash-based signature schemes	~32 B	~20 KB - 30 KB	High-security stateless design	Strong	Minimal assumptions, proven security	Large signature sizes
SRTPI	Isogeny-based	Supersingular isogenies	Variable	Variable	Strong theoretical foundation	Moderate	Compact parameter sizes	Computationally expensive
Three Bears	Lattice-based	Integer MLWE	~800 B - 1.5 KB	~900 B - 1.7 KB	Simple design; efficient computation	Moderate	Compact key sizes	Lower noise may expose vulnerabilities
Titanium	Lattice-based	Middle-product LWE	~15 KB	~3.5 KB	Tight security reduction; efficient parameter sets	Strong	Optimized for modern CPUs using AVX2 instructions	Larger ciphertexts compared to similar lattice schemes