CRYPTREC
耐量子計算機暗号の研究動向調査報告書

2025 年 3 月

CRYPTREC
暗号技術調査ワーキンググループ（耐量子計算機暗号）

目次

第 1 章

はじめに

暗号は情報を保護するための基礎的な手段である。基本的な暗号の分類として共通鍵暗号と公開鍵暗号があり，さらに公開鍵暗号の下位分類として通信相手の認証などを目的とした署名方式，情報の守秘を目的とした公開鍵暗号方式[*1]，秘密鍵の共有を目的とした鍵共有が存在する[*2] 。これらを含めた基本的な暗号方式を部品（プリミティブ）とした高機能暗号 [5] が数多く提案されている。2025 年現在，署名目的で DSA,ECDSA 等，情報の守秘目的で RSA-OAEP 等，秘密鍵共有の目的では DH,ECDH 等[*3] が国際的な標準暗号方式 [96] として用いられており，日本においても電子政府推奨暗号 [142] とされている。

これらの暗号方式の安全性と深く関わる計算問題として，素因数分解問題や楕円曲線上の離散対数問題があり，古典コンピュータ[*4]では効率的に解くことが困難であると信じられている。このことから，RSA 暗号や ECDSA 署名はある程度の大きさの鍵長を用いることで安全性が保てると考えられている [141]。一方で，Shor の量子アルゴリズム [122, 124] はこれらの計算問題を効率的に解くため，量子コンピュータの高性能化が情報セキュリティに影響を及ぼすとされている。以上の背景のもと，古典コンピュータ上での効率的な実装が可能であり，かつ古典・量子双方のコンピュータを用いた攻撃に対しても安全性を確保できる暗号方式が必要とされている。

本報告書で扱う耐量子計算機暗号の範囲 量子コンピュータによる攻撃への耐性を耐量子計算機性と呼び，耐量子計算機性を持つ暗号技術を耐量子計算機暗号（Post-Quantum Cryptography: PQC）と呼称する。しかしながら，耐量子計算機性の定式化はそれぞれの暗号技術の定式化を踏まえて行われており，一義的な意味で用いられる単語ではないことに注意が必要である。

例えば，公開鍵暗号方式は古典アルゴリズムの 3 つ組として定式化され，それらの古典安全性モデルの議論は IND-CCA 安全性をデファクトスタンダードとして収束している。それを踏まえ耐量子計算機性を持つ公開鍵暗号方式は，同じ定式化を持ちかつ耐量子計算機性の安全性モデルを満たす方式と捉えることができる。ただし 2025 年現在 IND-CCA 安全性の量子版は様々に提案されており，例えば Boneh と Zhandry による IND-qCCA2[24]，Chevalier らによる qIND-qCCA2[37] など，攻撃者が量子計算機をどのように用いるのかという点で定式化に細かい違いがある。
また，第 2 章冒頭にあるように，暗号技術のレイヤーを離れそれらを利用する暗号システムに関する耐量子計算機性を考える事も可能である。

PQC の考え方が出現した文脈は RSA，ECDSA の代替となる公開鍵暗号の開発であり [29]，PQC 候補とされるほとんどの暗号技術は古典コンピュータでの実装を前提として提案されている。この，古典コンピュータによる実装可能

[*1] 本報告書の中では公開鍵暗号を Public-Key Cryptography の意味で用い，その下位分類としての Public-Key Encryption を公開鍵暗号方式と表記する。
[*2] 基本的な暗号方式の定義と性質に関しては，例えば教科書 [149, 1.3 節] などを参照。
[*3] これらの方式には多くの解説記事があるが，DH，DSA に関しては例えば [180] を，ECDH,ECDSA に関しては [147] がある。
[*4] 理論的には決定的チューリングマシンを物理的に実装した計算機で，狭義においては CMOS 半導体を用いた論理回路による計算機を指す。現在普及しているコンピュータとほぼ同義である。

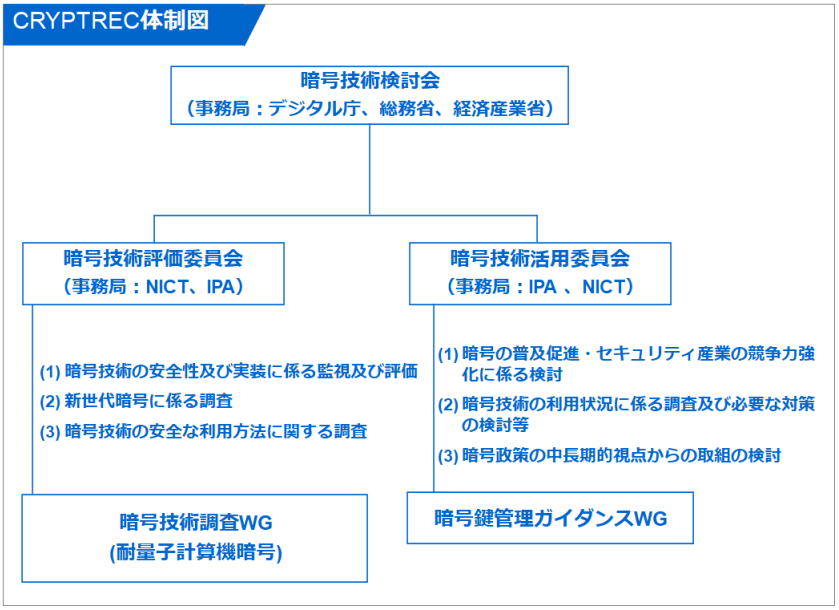


図 1.1: 2024 年度 CRYPTREC 体制図

性は情報を量子状態に乗せて伝達することで安全性を保証する量子暗号・量子鍵共有と PQC を区別する点とされることもある（例えば [85, p. 3] を参照）。

以上の状況と本報告書の中で扱う暗号技術の種類（1.4 節も参照）を踏まえ，報告書内では特に断りのない場合，耐量子計算機暗号（PQC）の言葉を，古典アルゴリズムの組み合わせにより定式化され，かつ耐量子計算機性を持つことを技術的に判断できる暗号方式とする。例えば米国の国立標準技術研究所（NIST）における PQC 標準化プロジェクトでの選定基準 [101, 4.A 節] では，公開鍵暗号方式，鍵共有（KEM），署名方式に対して IND-CCA2，EUF-CMA 等の古典の安全性ゲームから古典または量子による多項式時間帰着を行った先の計算問題の量子計算量がある値よりも大きいという基準が用いられ，暗号方式の提案者はその根拠を提案書に述べている。

本報告書の背景および調査内容 近年の世界的な量子コンピュータの開発と商用マシンの普及と並行して，PQC に関する研究及びその標準化に向けた活動も世界各国の組織で進んでおり（1.3 節を参照），国内でも PQC の研究動向を把握する必要性がある。2020 年度第 2 回暗号技術検討会において，2021 年度から暗号技術評価委員会の活動計画として 2 年をかけて PQC の研究動向を調査し，ガイドラインを作成することが決定された。暗号技術評価委員会は暗号技術調査ワーキンググループ（耐量子計算機暗号）を設置し，ワーキンググループにおいて 2022 年 9 月 30 日までの調査結果をガイドライン [4] と調査報告書 [3] としてまとめ，出版した。その後，2022 年度第 2 回暗号技術検討会において，さらに 2 年間の研究動向調査活動を継続し新たなガイドラインと調査報告書を作成することが決定され，暗号技術評価委員会は暗号技術調査ワーキンググループ（耐量子計算機暗号）を設置した（図 1.1）。

本ワーキンググループでは PQC の代表的な候補である 5 種類の分類（格子に基づく暗号技術，符号に基づく暗号技術，多変数多項式に基づく暗号技術，同種写像に基づく暗号技術，ハッシュ関数に基づく署名技術）について調査し，原則 2024 年 9 月 30 日までの調査結果をガイドラインと調査報告書にまとめた。本調査報告書の中で「現在」と表記する場合，特に断りがなければ 2024 年 9 月 30 日時点での情報を指すものとする。

ガイドラインは暗号初学者を対象としており，調査報告書は暗号についての知見のある技術者や専門家を対象としている。第 1 章ではガイドラインと調査報告書の概要，PQC を必要とする背景，研究及び標準化に関する動向，調査対象とした PQC の種類についてまとめている。第 2 章では PQC の活用方法と移行に関する内容，特に守秘・鍵共有・署名のための PQC の利用などについて記載している。第 3 章以降では暗号技術に携わる研究者及び技術者を読者と

して想定し，PQC の代表的な候補である 5 種類の分類をまとめた。ただし，これらの章ではガイドラインの記載内容は調査報告書の簡略版となっており，ガイドラインでは専門的な内容を省略し，暗号初学者が代表的な PQC 方式を把握するために最小限の内容のみを記載した。

共通鍵暗号と暗号学的ハッシュ関数の耐量子計算機性　本報告書では詳しく述べていないが，共通鍵暗号や暗号学的ハッシュ関数に対しても古典的な定式化を踏まえた上で量子的な攻撃に対する安全性モデルが提案されている。代表的な量子攻撃モデルとして，復号オラクルに対して重ね合わせクエリを許さない Q1 モデルと許される Q2 モデル [73] がある。

Grover の量子検索アルゴリズム [61] による共通鍵暗号方式の安全性の低下 [26] や暗号学的ハッシュ関数の衝突発見の高速化 [27] が知られている。アルゴリズムの最適性 [139] から，攻撃の量子計算量は鍵長の指数関数であり，ほぼ全ての共通鍵暗号，暗号学的ハッシュ関数は耐量子計算機性を持つものと認識されている。ただし，[74, 16] のように共通鍵暗号の暗号利用モードによっては安全性が著しく低下する例は知られている。パラメータに関しても鍵長を数倍に伸ばすだけで量子攻撃計算量を古典攻撃計算量と同等に増やすことが可能である．公開鍵暗号のように全く異なるアルゴリズムへの移行が必要とされないため，影響は限定的と考えられている [148, 49]。共通鍵暗号方式の安全性への影響を調査した報告書として CRYPTREC による [148]，日本銀行金融研究所による [162] 等に詳しい記述がある。

1.1　暗号の安全性に影響のある量子コンピュータの開発状況

量子コンピュータは重ね合わせ，エンタングルメント等の量子的な物理現象を用いて計算を行うコンピュータの総称である（[152, 第 2 章]，[150] 等を参照）。本節では，暗号の安全性に影響を及ぼすと考えられる量子コンピュータの開発状況についてまとめる。

1.1.1　量子コンピュータの分類

量子コンピュータの開発は世界中で進められており，その形も多様であるが計算モデル，物理的実装，性能により分類できる。

計算モデルによる分類　量子計算の基本的な計算操作と物理的操作の対応関係を表すモデルにより，量子回路型計算，測定型量子計算，断熱型量子計算，アナログ量子シミュレーション，トポロジカル量子計算，ホロノミック量子計算等に分類できる[*5]。

量子回路型計算，測定型量子計算　特に量子回路型計算，測定型量子計算では多くの種類の物理実装が存在する。超伝導量子ビット，冷却原子（中性原子），イオントラップ，シリコン量子，光量子，カラーセンター量子コンピュータ等の開発が進められている。Shor のアルゴリズム等，暗号に大きな影響のあるアルゴリズムは量子回路を用いて記述されていることから，これらのコンピュータの大規模化が現代暗号に大きな影響を与えると考えられる。1.2.2 節に述べるように，多くの素因数分解実験が量子回路型計算のフレームワークで行われている。

断熱型量子計算と量子アニーリング　断熱型量子計算は基底状態が簡単に用意できる初期ハミルトニアンから，組み合わせ最適化問題の解が基底状態に対応するようなハミルトニアンへとゆっくりと変化させることで解を得る計算フレームワーク [14, Def. 1] である。断熱型量子計算の下位分類の中で特に量子アニーリング（Quantum Annealing: QA）はクラウドサービスを通じた商用コンピュータが提供されていることから注目を集めている。

量子アニーリングは元々，Apolloni ら [19] によりシミュレーテッドアニーリング（Simulated Annealing: SA）に類似したアルゴリズムを量子的に構成したことから名付けられたものであるが，現在では量子断熱計算のモデル [14,

[*5] 分類に関しては [150, 134, 51] および [70, Sect 1.6] を参照。

Def. 1] における条件を開放系とし，有限時間に設定し，ハミルトニアンをイジングモデルに制限した計算フレームワークを指すものと見なされている [169, § 3]。イジングモデルを用いた素因数分解実験も数多く行われている。（1.2.2 節を参照）

ハミルトニアンの形に制限のない量子断熱計算の計算能力は量子回路型計算と多項式時間等価であり，計算量クラス BQP に属する [9]。この論文内では量子回路とハミルトニアンを互いに多項式サイズの差で変換する手法もで与えられてはいるものの，2025 年現在でその変換を暗号方式への攻撃に応用した例は確認されていない。

3-XORSAT 問題のように古典多項式時間で計算可能であるにも関わらず，自然な形でインスタンスの変換を行ったイジングモデルによる量子アニーリングでは効率的に解くことが困難であることが示唆されている計算問題の存在も知られており [71]，暗号に関係する問題が同様の性質を持つかどうかが未解決問題となっている。

量子ゲート型と量子アニーリング型　量子回路型計算を超伝導量子ビット，イオントラップにより実現したコンピュータ，断熱型量子計算の中でも量子アニーリングを超伝導磁束量子ビットにより実現したコンピュータは物理的なハードウェアの進化とプログラミング環境の進化により商用利用が進んでいる。これらは量子ハードウェアを専門としない技術者でもクラウドを通じて容易に利用可能であることから注目を集めていることを踏まえ，量子回路型コンピュータと量子アニーリング型コンピュータを指してそれぞれ量子ゲート型と量子アニーリング型という名称で分類し対比することもある [152, 第 2 章, p. 11]。

アナログ量子シミュレータ　近年，中性原子や光格子を用いた様々な実装が急速に進展している計算フレームワークである。人工的な量子系を用いて別の量子系をシミュレーションするコンピュータの総称であり [173]，古典コンピュータを用いて量子回路や量子アニーリングの出力をシミュレーションする技術とは異なる。

規模と性能による分類　物理的な実現方法・計算モデルによる分類以外に，規模と性能による分類も提案されている。NISQ は Noisy Intermediate-Scale Quantum の略で，2018 年に Preskill[69] により提案された概念である。NISQ デバイスは搭載される物理量子 bit が数十から数百程度で，実行時のノイズが大きい量子デバイスを指す。量子誤り訂正や大規模な計算を行うには不十分な性能とされる。2025 年現在，全ての量子コンピュータは NISQ デバイスであると考えられる。

FTQC は Fault-Tolerant Quantum Computation の略で Shor[123] により提案された概念である。ノイズやデコヒーレンスのある量子デバイスでもその影響を量子誤り訂正等を用いて低減することで，大規模かつ長時間の計算が可能となる理論を指す。そのような計算を実現するデバイスは FTQC デバイスと呼ばれ，必然的に多くの論理量子 bit による非常に低いエラーレートでの量子計算を可能とする。実用的な暗号方式に用いられる大きさの素因数分解問題，離散対数問題の計算を行うためにはこの規模のコンピュータが必要と考えられている。

実際には FTQC デバイスが実現される前でもある程度の性能の量子エラー訂正を用いることで有用な計算が可能となると考えられており，NISQ と FTQC の中間的な性能のデバイスを指す様々な概念が提案されている。汎用的なものでは early-FTQC[175] という，数万物理量子 bit 程度の規模を持つ量子デバイスの概念があるが，特に暗号に関係する概念として CRQC（Cryptographically Relevant Quantum Computer）があり，古典コンピュータでは解くことが困難な暗号学的問題を解くことのできる量子コンピュータとして定義されている [8]。

1.1.2　ハードウェアの進展とロードマップ

前節の量子コンピュータの分類を踏まえ現在の量子コンピュータの開発状況と各組織のロードマップを概観する。量子コンピュータの開発は世界中で進められており，網羅的な記述を行うことは本報告書の目的ではない。近年開発された量子コンピュータの中で特筆すべき性能を持つもの，暗号に関係する応用に用いられたもの，日本国内で開発されたものに絞り紹介する。

以下では，物理量子 bit は搭載されている物理的な量子ビットの数を表し，論理量子 bit は量子誤り訂正などを行っ

た後の論理レベルでの量子 bit 数を表すものとする。

量子回路型コンピュータの開発は米国の民間企業を中心に 2010 年代以降急速に発展しており，特に超伝導量子ビットによる実装と中性原子による実装が 1000 物理量子 bit を超えるプロセッサを実現している。しかしながら，調査の範囲で確認された量子コンピュータは総じて NISQ デバイスに留まっており，CRQC レベルのものは確認されていない。一方で，数年前までは誤り訂正処理を行うことで逆にノイズが蓄積しエラーレートが悪化する状態であったものが，2023 年には誤り訂正後のエラーレートが下回るという結果が報告されており [125, 95, 140, 38]，FTQC に向け安定な論理量子ビットの構築が進んでいる。また，近年では多くのコンピュータが実験室レベルではなく，商用として開発されクラウドサービスを通じて公開されている [64, 15, 130] ことも特徴である。

世界的に FTQC の開発を目指して研究が進められており，大きな枠組みでは例えば以下の目標が掲げられている。2020 年 1 月に決定された日本のムーンショット目標 6 では，様々な実装による量子コンピュータの開発を行い，2050 年までの FTQC 実現を目指している [154]。欧州の European Quantum Flagship が 2024 年 2 月に公表したロードマップでは，2020 年代後半に様々な実装での 1000 物理量子 bit，2030 年までに 99% 以上の忠実度[*6]をもつ 1000 論理量子 bit デバイスの実現を目標として掲げている [51]。

なお，量子コンピュータの性能を十分に引き出す強力なアルゴリズムを実現するためには量子 bit 数の増加のみではなく，ゲート操作の忠実度の向上，コヒーレント時間の向上などの課題を克服し，量子誤り訂正，量子ランダムアクセスメモリ等の 2025 年現在では完全には実用化されていない技術を用いる必要がある。それらの開発スピードの予測困難性が，量子コンピュータが暗号に与える影響の将来予測を困難なものとしている。

以下，各実装方式ごとの開発状況とロードマップを概観する。

超伝導量子ビット 超伝導量子ビットによる量子コンピュータは集積可能性と設計自由度が高いこと [156]，十分な品質の量子ビットが比較的安定に実現できること [174] から，大規模化に向けた開発が他の方式よりも先行した。特に 2010 年代から IBM のクラウドサービスによる一般公開があり，代表的な量子コンピュータ方式として認識されている。近年でも 2019 年 10 月に Google が 53 物理量子 bit の量子プロセッサ Sycamore を開発し量子超越性を宣言した事 [52]，2022 年 12 月に中国のチームが 110 物理量子 bit の量子プロセッサ [137] を用いて素因数分解アルゴリズムの実験を行った事 [138] 等多くの話題がある。

量子 bit 数の多いプロセッサでは, IBM が 2023 年 12 月に開発した 1121 物理量子 bit のプロセッサ IBM Condor[55]，中国科学院の量子情報・量子技術創新研究院が 2024 年 4 月に開発した 504 物理量子 bit のプロセッサ骁鸿（Xiaohong）[78] 等が代表的である。IBM が 2023 年 12 月に発表したロードマップ [65] によれば，2029 年までに実行可能ゲート数を 1 億に増やし，2033 年には実行可能ゲート数 10 億，論理量子 bit 数を 1000 まで上げるとしている。

量子誤り訂正に関する話題では 2022 年に訂正後のエラーレートが訂正前よりも下がるブレークイーブンポイントを達成したとの報告が米国，中国それぞれの研究チームから行われている [125, 95]。その後，2024 年 8 月には Google のチームが表面符号を用いた量子メモリの実装実験 [38] を行い，誤り訂正後のエラーレートが訂正前よりも下がり，構成された論理量子 bit の寿命が物理量子 bit の寿命よりも 2 倍程度長いと報告している。

日本国内では 2023 年 3 月に富士通と理化学研究所を中心としたチームが 64 物理量子 bit の量子コンピュータ叡を開発，現在までに 3 台がリリースされクラウドを通じて利用されている [166, 177]。理研の次の目標は 144 物理量子 bit デバイスの実現であるとしている [167]。富士通では 2024 年 5 月にロードマップ [155] を公開し，2025 年中に 256 物理量子 bit を実現し，2026 年度以降に 1000 物理量子 bit を達成するとしている。

また，ムーンショット目標 6 の中のプロジェクト「スケーラブルな高集積量子誤り訂正システムの開発」においても 2030 年までに超伝導量子ビットを用いた 100 万物理量子 bit の FTQC を，2025 年に 100 物理量子 bit を用いた 1 論理量子 bit のプロトタイプを作成することをマイルストーンとしている [151, 2:30]。

[*6] ここでは量子ゲート操作の忠実度（gate fidelity）を指す。厳密な定義は決まっていないものの，大まかに量子デバイスの出力が理想的な計算結果とどの程度一致しているかを測る指標である。

イオントラップ イオントラップ方式の利点はコヒーレント時間の長さと量子操作時のエラー率の低さである [146]。同じ量子 bit 数を持つ超電導量子ビットのコンピュータと比べると量子体積[*7] などの面で優位性がある [111]。2020 年 10 月に IonQ が 32 物理量子 bit[36]，2024 年 6 月には Quantinuum が 56 物理量子 bit[112] のデバイスを発表している。Quantinuum と Microsoft により，量子誤り訂正後のエラーレートが訂正前よりも 800 倍程度低減されたという実験報告が行われている [140]。

中性原子 中性原子（冷却原子）を用いた量子コンピュータはコヒーレント時間が長く，スケーラビリティが超伝導量子ビットやイオントラップよりも良いとされる。また，動的光ピンセット技術により量子ビット間の全結合が可能であるともされている [179]。開発は Atom computing による 1180 物理量子 bit の量子プロセッサ [40] を筆頭にここ数年で急速に進展している。また，Harvard 大学らのチームが 280 物理量子 bit を用いて量子誤り訂正を行った 48 論理量子 bit の構築とベンチマーク [23] を行い，訂正後のゲート操作の忠実度が上がることを確認している。日本の自然科学研究機構分子科学研究所ではほぼ理論限界に近い 6.5 ns でのゲート操作速度を可能とする 400 物理量子 bit のシステムが構成されており [176, 8:45]，2030 年の事業化を目指している [160]。

シリコン量子ビット この方式はシリコン中の電子スピンを用いるため，既存の半導体製造技術を応用可能であると見込まれている [164]。また，動作温度が 10K 程度と超伝導量子ビットど比べて高いため冷却器の小型化が可能である点なども利点として挙げられている。

Intel は 2010 年代から研究を進め，2023 年 6 月に 12 物理量子 bit のプロセッサ Tunnel Falls を発表した [67]。また，日立製作所 [157, 161] と理化学研究所 [165] が実用化に向けて研究を進めている。日本のムーンショット目標 6 では，2040 年までに 10 万〜100 万物理量子 bit による誤り訂正の実証を行い，2050 年までに 100 万以上の物理量子 bit からなる FTQC デバイスの実現を目指すとしている [163, p. 2]。

光量子 光子を用いることから室温や大気中での動作が可能で，装置の小型化が見込まれている [145]。2022 年 6 月にカナダの Xanadu 社が 216 量子 bit を搭載した Borealis を発表した [76]。2024 年 11 月には理研などが計算プラットフォームを開発し [159]，2030 年までに光ファイバ型の連続量光量子コンピュータの実現を目指すとしている [145]。

量子アニーリングマシン D-Wave が 2010 年代から超伝導磁束量子ビット型アニーリングマシンを商用に発表している。2020 年 9 月にリリースされた D-Wave Advantage は約 5000 物理量子 bit を搭載していた [82]。2024 年に発表された D-Wave Advantage 2 では量子ビット同士の結合数とノイズが改良されており，2024 年 6 月時点でのプロトタイプは約 1200 物理量子 bit を搭載 [45]，将来的に 7000 物理量子 bit 規模のアニーリングマシンを提供する予定であるとしている [44]。

日本国内では産業技術総合研究所が組み合わせ最適化問題に特化した超伝導磁束量子ビット型の 6 物理量子 bit 量子アニーリングマシンを開発 [168]。また，NEC と東北大学が中心となり，超伝導パラメトロンを用いた 8 物理量子 bit の量子アニーリングマシンが開発されている [153]。

1.2 耐量子計算機暗号（PQC） の必要性について

本節では，量子コンピュータによる現代暗号への影響と PQC の必要性についてまとめる。2024 年 9 月現在，以下に述べる素因数分解問題，離散対数問題を解く実験の他，検証用に構成した小規模な共通鍵暗号の鍵復元実験 [107]，耐量子計算機暗号の計算問題を解く実験 [114] が報告されている。調査の範囲では既存の量子コンピュータの性能が古典コンピュータの暗号解読性能を超えたという報告，および実社会で用いられている大きさのパラメータを持つ暗号方式

[*7] NISQ デバイスの性能を計る指標で，2018 年に IBM の研究者により提案された [43]。ベンチマーク用量子回路をデバイスで実行した測定結果と理想的な実行結果を比較し，量子ノイズの影響により結果が理想から大きくずれないような最大の量子回路サイズ（量子ビット数 × 回路深さ）を性能指標とする。

が解かれたという報告は無く，現代暗号に対する量子コンピュータの直接的な脅威は現時点では生じていないと考えられる。

一方で，各機関が発表しているロードマップが予定通りに達成されると仮定すると，今後数十年で RSA，ECDSA をはじめとする素因数分解問題や離散対数問題の計算困難性に基づいた暗号の解読を可能とする規模の量子計算を実行可能な量子コンピュータが開発される。暗号方式の提案から社会的な普及までは RSA 暗号・楕円曲線暗号で 20 年ほどの期間が必要とされたことから，PQC の場合でも同程度の期間が必要と想定されるため，長期間の移行スケジュールを策定し，準備を行う必要がある。

1.2.1 量子コンピュータの影響による現代暗号の危殆化予測

Shor による素因数分解問題と離散対数問題の量子多項式時間アルゴリズム [124] が発表されて以降，数千 bits の RSA 暗号を危殆化させる量子コンピュータの規模の見積もり [56, 58, 172, 57]，実現時期の予測 [30, 18, 79, 90, 91] に関する研究が進められている。2025 年現在，直近の数年間で実用的な RSA 暗号，ECDSA 署名を攻撃可能な古典または量子コンピュータが開発される可能性は極めて低いと考えられる。一方で，各国の標準機関は長期的には新たな暗号方式に移行する必要があると考えプロジェクトを進めている。例えば米国 NIST は 2024 年 11 月に公表した NIST IR 8547 （initial public draft）[88, Sec. 4.1] において 2048，3072bits RSA と，224，256bits ECDSA，EdDSA を 2035 年までに段階的に廃止，利用禁止にするとしている。

暗号解読のモデルと数値化方法 Shor のアルゴリズムが量子回路を用いて表現されていること，量子回路型計算を行う量子コンピュータの開発が他の方式よりも先行していることから，現代暗号の解読量子計算量の見積り [56, 58, 172, 57]，および量子コンピュータ性能の将来予測を通じた共通鍵暗号，耐量子計算機暗号のパラメータ設定は量子回路モデルを用いて数値化 [101, p. 18] がされている。量子計算量理論の観点からは 1.1.1 節冒頭に挙げた量子計算モデルがいずれも多項式倍の差を除いて等価であることが示されており，量子回路型計算で困難であると予想される計算問題が断熱量子型計算（量子アニーリング）のような他の既知のモデルを用いた量子計算により危殆化する可能性は小さいと考えられる。

なお，2048bits の合成数を公開鍵に用いた RSA 暗号（以下，RSA-2048 と表記）は古典で 112-bit 安全性を持つとされており [141]，暗号に影響のある量子コンピュータの開発が仮に実現しなかった場合でも，古典コンピュータの性能の伸びにより長期的には危殆化すると考えられている。このことから，将来的な鍵長の変更もしくは新たな暗号方式への移行は量子コンピュータの大規模化とは独立した課題として準備を進める必要があることは長年議論されてきた [141, 21, 120] ことを指摘しておく。

危殆化時期の予測 RSA-2048 に対する量子コンピュータの影響とその危殆化時期に関して，様々な予測が存在する。定量的な予測に基づいたものでは 2039 年以降 [30]，2050 年前後 [18] と少なくとも 20 年程度は実現に時間がかかるとされている。

セキュリティ・量子分野の専門家の予測では，Mariantoni が PQCrypto2014 の招待講演 [79] で調査に 5 年，開発に 10 年程度で 15 年後（2029 年）としたもの，Mosca が Workshop on Cybersecurity in a Post-Quantum World（2015 年開催）で 2026 年から 2031 年 [90] と予測したものが有名である。近年では国際会議 RSA Conference 2023 内で開かれた暗号専門家によるパネルディスカッションの中で，Shamir が RSA，DH，ECDH に影響を及ぼす量子コンピュータがあと 30 年から 40 年で開発される可能性があると発言している [129]。

個人ではなく，多くの専門家へのアンケートを集計した結果が 2019 年から毎年 Global Risk Institute により Quantum Threat Timeline として発行されている。2023 年に行われたアンケートを基にした予測レポート [91] では 24 時間で RSA-2048 を解読可能な量子コンピュータが 15 年以内に出現する可能性が 33% から 54% 程度であると分析している。この調査結果を引用するレポートは多く，例えば金融におけるサイバーリスクを取り扱う国際的なコン

ソーシアム Financial Services Information Sharing and Analysis Center の 2023 年報告書 [60] では危殆化時期をあと 10 年から 30 年，米国の金融サービス標準を決定する ASC X9 の 2022 年度版報告書 [59] では CRQC の登場時期について “There is no consensus on this issue” としながらも，危殆化時期をあと 5 年から 30 年としている。

日本国内の専門家へのアンケート調査では，2019 年に行われた文部科学省科学技術・学術政策研究所（NISTEP）による科学技術予測調査 [170, (II-4)p. 48, 52] がある。この中ではある程度コヒーレンス時間の長い数百物理量子 bit 規模の量子回路コンピュータの登場を 2033 年頃と予測しているため，現代暗号に対して脅威となる量子コンピュータが出現するのはそれ以降と解釈できる。ムーンショット目標 6 では 2050 年頃までに FTQC を実現するとしている [154] ことから，予測が実現されるのであれば現代暗号の量子コンピュータによる危殆化もその付近で起こると考えられる。

1.2.2 量子コンピュータによる素因数分解・離散対数問題計算の現状

将来的に RSA，ECDSA が危殆化すると考える専門家が多数存在する一方で，量子コンピュータ実機を用いた素因数分解問題及び離散対数計算の実験は小規模なものに留まっている。本節では，量子コンピュータを用いた Shor のアルゴリズムに関する実験，その他の素因数分解アルゴリズムに関する実験，および関連する理論的な成果についてまとめる。

なお，量子回路型計算および量子アニーリングの古典計算機によるシミュレーションを用いた素因数分解の実験報告が多く存在する [136, 172] が，本報告書では省略し量子的な現象を用いた計算機による報告のみを取り上げる。

Shor のアルゴリズムの実機実験 量子回路型コンピュータ実機を用いた実験は，CRYPTREC 外部調査報告書「Shor のアルゴリズム実装動向調査」[158] に挙げられているもの及びその後の [128, 126, 135] を含めて 15，21，35 の素因数分解実験および離散対数問題 $2^z \equiv 1 \pmod 3$ の離散対数の計算実験を行ったもののみしか知られていない。

[132, 133] をはじめとする Shor のアルゴリズムを用いた初期の報告は $N = 15$ の素因数分解回路の量子フーリエ変換部分を除いた部分回路を実装する予備実験的なもの，位数や N の情報を用いて過度な簡略化を行ったものが多かった。2019 年には指数計算部分が簡略化されているものの，量子フーリエ変換部分と組み合わせた回路に対して IBM Quantum による $N = 15, 21, 35$ の実験報告 [83] が行われている。また，離散対数問題の実装実験報告 [18] が出版されるなど，実際に問題のインスタンスサイズには表れない量子回路規模の拡大は着実に続いていると考えられる。

Shor のアルゴリズムに関する理論の進展 1.1.2 節に紹介した量子コンピュータの性能進化がターゲットとなる数の目に見える伸びに繋がらない理由が量子コンピュータ実機の性能と Shor のアルゴリズムの性質双方の観点から検証され，明らかになりつつある。

Ichikawa らによる量子コンピュータ実機実験に関するサーベイ論文 [66] によると，2016 年から 2022 年の間に出版された 748 件の実験報告で用いられた実際の量子 bit 数の中央値が 5 から 6 に増えたのみでありほぼ横ばいである。これは 2022 年に 433 物理量子 bit を搭載した IBM Quantum Osprey [39] が発表されていたこととは対照的である。量子ノイズ，デコヒーレンス等の影響により，デバイスに搭載されている物理量子 bit 数と，実際に安定して動作し測定可能な物理量子 bit 数の間には大きな差がある。

また，2024 年には Cai により Shor のアルゴリズムが量子ノイズに弱い事の理論的な証明 [31] が与えられている。この現象は経験的には知られていたが [18, Sec. VI]，より大きな規模で Shor のアルゴリズムを実行するためには量子ビット数の増加だけではなく，量子ノイズの影響を下げる必要があることが理論的に示された形となる。

以上をまとめると，より大きな数の素因数分解を Shor のアルゴリズムを用いて行うためには十分にノイズが小さく，安定して動作する量子ビットを搭載した量子コンピュータが必要であると考えられる。

また，Shor のアルゴリズムを用いて現在より大きな数の素因数分解を行うためには，これまでの実験のように入力インスタンスに合わせ簡略化した量子回路ではなく，汎用の剰余加算・乗算回路による構成を行う必要がある。しかしながら，$N = 15$ の素因数分解を行うために 4 ビットの汎用剰余加算・乗算回路を用いて Shor のアルゴリズムを構成

すると，ゲート数約 13,000，深さ約 10,500 となるという評価 [172] がある。これは, 現在の量子コンピュータでは実行不可能な回路規模であり，簡略化されていない量子回路の実行は現在のところ困難であると考えられる。

一方，2023 年 8 月に Regev[115] により Shor のアルゴリズムよりも量子 bit 数を増やす代わりに量子ゲート数の少ないアルゴリズムが提案された。多くのフォロー論文 [113, 48] が発表されているものの，量子コンピュータ実機を用いた実験は確認されていない。

別の理論的な可能性として，量子アニーリングと一般的な断熱量子計算の中間にあるストカスティック断熱量子計算[*8]による Shor のアルゴリズムのシミュレーションがある。ストカスティック計算はハミルトニアンを基底に関して非正定値の非対角実行列に制限した断熱型量子計算のクラス [14, Sect. VI] であり，計算量理論ではクラス StoqAQC と名付けられている。このクラスの能力を持つコンピュータは，Shor のアルゴリズムを効率的にシミュレート可能であることが知られており [53, 54]，ストカスティック断熱量子計算を実行可能な量子コンピュータが出現するかどうかが安全性に関わると考えられる。

量子アニーリングによる実験 Shor のアルゴリズム以外の素因数分解の計算手法のうち代表的なものとして，2 進数乗算の筆算形式で式展開したものを，組み合わせ最適化問題（Quadratic Unconstrained Binary Optimization: QUBO）として定式化したものがある。QUBO とイジングモデルは自明な変換が知られていることから，量子アニーリングを中心とした断熱量子計算を用いた実験が多数報告されている。

2000 年代後半の初期の実験 [106] ではハミルトニアンに合わせて有機化合物を合成し，最適化問題の変数に対応する原子のスピンを核磁気共鳴（Nuclear Magnetic Resonance: NMR）を用いた分析により結果を取り出すという手法で計算を行っていたためスケーリングが困難であったが，D-Wave の量子アニーリングマシンがオンライン上で比較的手軽に利用可能になって以降は実験報告が相次いでいる [171, 136]。素因数分解のターゲットとなる数は着実に大型化しており，現時点での最大は 2023 年に D-wave Advantage 4.1 を用いた 23 ビットの 8219999=32749×251 [47] であり，最適化問題の表現方法，変数の省略など多くの技術を使い Pegasus トポロジーで接続された量子ビットの性能を引き出している。

その他の素因数分解手法 Shor のアルゴリズム，量子アニーリング以外の手法でも様々な素因数分解の実験が行われている。アニーリングと同様の QUBO を Quantum Approximate Optimization Algorithm （QAOA）を用いて解く実験 [110]（143，291311 を分解），Variational Quantum Eigensolver （VQE）を用いて解く実験 [127]（251 を分解），Digitized adiabatic quantum computation を用いて解く実験 [62]（2479 を分解）の報告がある。これらの実験はいずれも IBM Quantum を用いて行われた。

量子回路型コンピュータ上で QAOA を用いた素因数分解問題へのアプローチとして，Schnorr アルゴリズム [119] の部分的な量子化の研究が存在する。Schnorr アルゴリズムは数体篩法の関係探索を係数制限付きの近似最近ベクトル問題に変換して行うが，[138] ではこれをさらに最適化問題に落とし込み，QAOA を 10 量子 bit 回路上で実行することで 48bits の数の素因数分解実験結果を報告している。

また，中性原子による実装の一種として，リュードベリ原子によるアナログ量子シミュレータを用いてグラフの最大独立集合問題を解くための枠組みが整理されており，これを用いた素因数分解の実験も行われている。[105] では 6,15,35 の素因数分解のインスタンスを SAT を経由して最大独立集合問題に変換して実験を行っている。

1.3 PQC の研究及び標準化等に関する動向

現在 PQC として扱われている暗号のほとんどは 1994 年に Shor のアルゴリズムが発表される以前から効率性および理論的側面から研究が行われており [81, 75, 80]，2000 年代以降に耐量子計算機性が注目されたものである。な

[*8] ストカスティック（stoquastic）は stochastic と quantum の合成語で，2006 年に Bravyi らの論文 [28] で導入された単語される。

お，Post-Quantum Cryptography の用語自体は 2004 年の論文 Post-Quantum Signatures[29] が初出であり，p.1 に Bernstein の造語であることが明記されている。

現在，PQC に関する研究成果は暗号の国際会議で主に発表されている。特に Crypto，Eurocrypt，Asiacrypt 等の暗号全般を扱う会議で取り扱われることも多いが，その他 PQC を専門に扱う国際会議として PQCrypto が 2006 年から開催され，2024 年までに 15 回が開催されている。

以下，各国における標準化の動向を述べる。米国の国立標準技術研究所（NIST）は PQC の標準化活動を初期から大規模に行っており世界への影響力が大きいため，まず米国の状況について述べてその後に各国の状況について述べる。

1.3.1 米国 NIST における標準化の動向

2015 年 8 月国家安全保障局（NSA）が PQC への移行計画 [7] を発表したことを受け，標準化活動が NIST により開始された。2016 年 2 月には福岡で開催された国際会議 PQCrypto 2016 において NIST の Moody により NIST PQC 標準化プロジェクトに関する講演 [86] が行われ，選定基準に関する意見募集を経て 12 月に Call for Proposals [101] が正式公開された。

2017 年 11 月 30 日の公募締め切りまでに世界中から耐量子計算機暗号の候補 82 方式が提案され，公募条件を満たした 69 方式が標準化プロジェクト第 1 ラウンド候補として公開されたが，5 方式は公開後に取り下げられている。締め切り直後からメーリングリスト pqc-forum [109] 上では世界中の暗号研究者，暗号方式の提案者らを交えて理論的な脆弱性から実装リファレンスコードの軽微なバグの指摘に至るまで多彩な議論が行われた。このときの議論は [12, 108] 等にまとめられている。

2019 年 1 月 30 日には，第 2 ラウンドへ進む 26 方式が発表され，その後 2020 年 7 月 22 日には，第 3 ラウンドへ進む Finalists の 7 方式と Alternate Candidates の 8 方式が発表された [89]。両者の違いは Finalists が第 3 ラウンドの終了時に標準化方式となるかどうかが判断されるもの，Alternate Candidates が標準化方式の候補ではあるものの第 3 ラウンドの終了時点では判断が行われない可能性が高いものとされていた。しかし実際には Finalists であった Classic McEliece が第 4 ラウンド候補として判断を保留された一方で，Alternate Candidates であった SPHINCS$^+$ がそのまま標準化方式として選ばれている。

2022 年 7 月 5 日に NIST から標準化方式として公開鍵暗号 1 方式と署名 3 方式が発表された [13]。これら 4 方式のうち，格子に基づく公開鍵暗号方式 CRYSTALS-Kyber は FIPS 203（ML-KEM）[98] として，格子に基づく署名方式 CRYSTALS-Dilithium は FIPS 204 （ML-DSA）[97] として，ハッシュ関数に基づく署名方式 SPHINCS$^+$ は FIPS 205 （SLH-DSA）[100] として 2024 年 8 月にそれぞれ標準化されている。また，格子に基づく署名方式 FALCON についてもアルゴリズムの微修正を経た後に FIPS 206 （FN-DSA）として標準化される予定である [25]。

標準化の 4 方式が決定されると同時に，第 3 ラウンド候補の中から第 4 ラウンドへと進む公開鍵暗号方式の 4 方式が発表され，さらに追加の電子署名方式が再公募されることが周知された [118]。第 4 ラウンドに進んだ 4 方式のうち，BIKE，Classic McEliece，HQC の 3 方式が符号に基づく公開鍵暗号方式，SIKE が同種写像に基づく公開鍵暗号方式であった。その後，2022 年 8 月に SIKE に対する古典多項式時間による鍵復元攻撃が発表され [32]，致命的であることが確認されたことから提案チームにより候補から取り下げられた。2024 年 4 月に開催された第 5 回 PQC Standardization Conference における Moody の講演によると，NIST は 2024 年末までに残った第 4 ラウンド候補の中から数件を標準化に選ぶとしている [85, p. 12] [*9] 。

署名方式の追加公募 第 4 ラウンドの発表と並行して，NIST は 2022 年 9 月から正式に追加の NIST PQC 標準化プロジェクト追加署名（Additional Digital Signature Schemes）の募集を開始した。締切の 2023 年 6 月 1 日までに 50 方式の応募があり，翌 7 月に公募条件を満たした 40 方式が発表された。公募の事前情報として，2022 年 7 月に

[*9] 2025 年 3 月に HQC が標準化方式として選ばれたことが発表されている [121]。

pqc-forum に投稿された文書 [84] では NIST が署名長と検証時間の小さい方式を求めているとし，一例として多変数多項式に基づく署名方式の一種である UOV 方式が挙げられている。また，Module 格子のような構造化格子に基づく署名方式は既に CRYSTALS-Dilithium と FALCON が標準化に決まっていることから，構造化格子に基づく手法以外が望ましいとしており，後半の内容は募集要項にも明記された [99, p. 2]。結果として格子に基づく署名は 7 方式，UOV 型の多変数多項式に基づく署名では 7 方式の応募があった。

2022 年の署名方式公募後，NIST は選考の第 1 ラウンド候補を分類ごとに発表したが，その中に 2016 年の標準化には存在しなかったカテゴリ MPC-in-the-Head が新たに登場している。これはマルチパーティ計算から構成したゼロ知識証明プロトコルに Fiat-Shamir 変換を適用することで署名方式を得る構成フレームワークであり，格子，符号のように安全性の根拠となる計算問題の種類を表すものではない。MPC-in-the-Head に分類されているそれぞれの方式の安全性は実際には符号問題，多変数方程式問題，共通鍵暗号方式の平文復元問題の困難性などに帰着されている。

2024 年 10 月には第 2 ラウンドの候補となる 14 方式 [117] が発表された。格子に基づく署名方式は格子同型性判定問題を安全性の根拠とした HAWK の 1 方式，多変数多項式に基づく UOV 型署名が 4 方式，MPC-in-the-Head 型の構成を行った署名が 5 方式であった。選定に関わるレポートは [11] で公開されている。

PQC への移行 2015 年に Mosca の提案した暗号の危殆化に関わる不等式 [90] では，　X（情報を保護する期間）+Y（システム移行期間）と Z（CRQC 開発までの期間）の大小関係によりシステム移行の準備期間を設定する必要があるとしている。一方で，暗号化データを保存し，将来的にコンピュータの性能が上がってから解読するハーベスト攻撃（2.2.2 節も参照）を想定すると，CRQC 開発までの年数によらず，現在の暗号利用にはリスクがあるとも考えられている（例えば [88, Sec. 1] を参照。）以上の背景のもと，2022 年 5 月公表された国家安全保障覚書 NSM-10[92] では 2035 年を目処に暗号システムで使用する暗号を PQC に移行することを目標としている。同様に，2022 年 9 月に発表された商用国家安全保障アルゴリズムのリスト 2.0[6] では 2035 年までにシステムに耐量子計算機性をもたせることを目標としたタイムラインを掲載している。

現在使われている暗号から PQC への移行を推進するため，NIST 内の NCCoE（National Cybersecurity Center of Excellence）を中心にコンソーシアムが設立された [50]。組織における暗号のユースケース，相互運用性やリスク評価を含めた移行計画の策定に関する包括的な技術文書が NIST SP 1800-38A から 38C として発行される予定であり，現在は Initial Preliminary Draft[94] が公開されている。

安全性レベル NIST PQC 標準化プロジェクトにおいて，暗号方式の安全性はレベル 1 から 5 で定義されており，提案者は応募時にパラメータと達成される安全性レベルを示す必要があった。レベル 1, 3, 5 はそれぞれ AES128, AES192, AES256 などの 128, 192, 256bits の秘密鍵を持つブロック暗号の鍵復元の困難性と同等かそれ以上の計算量であり，レベル 2 と 4 はそれぞれ SHA256/SHA3-256 と SHA384/SHA3-384 などの 256bits と 384bits の暗号学的ハッシュ関数の衝突探索の困難性と同等かそれ以上の古典もしくは量子計算量とされている。レベル 1 から 5 の具体的な計算量は表 1.1 で与えられる [99]。古典コンピュータによる攻撃者に対しては古典論理回路のゲート数が，量子コンピュータを利用可能な攻撃者に対しては量子回路のゲート数と最大深さの積が与えられている。計算量評価において，公開鍵暗号方式では，IND-CCA2 安全性を考える際には 2^{64} 個以下の選択暗号文を復号オラクルに古典的にクエリできるとし，署名方式では，EUF-CMA 安全性を考える際には 2^{64} 個以下のメッセージを署名オラクルに古典的にクエリできるとしている。

また，レベル 1,3,5 の量子回路計算量で $2^{157}, 2^{221}, 2^{285}$ とされている部分は 2016 年の Call for proposals[101, 4.A.5 節]　では $2^{170}, 2^{233}, 2^{298}$ であった。つまり，2016 年の PQC 候補でレベル 1,3,5 とされているものは 2022 年の定義でもレベル 1,3,5 の基準を満たすことになる。この更新は AES を解読する量子回路の改良により，量子計算量が改善されたことによる。

NIST 標準化に伴い，FIPS 203,204,205 の各文書ではセキュリティカテゴリの定義に関して NIST SP800-57 Part 1[21] を参照しているが，2024 年 9 月時点で最新の Rev. 5 において対応する定義が存在しないことが指摘されており，

NIST の担当者から次期リビジョンでの修正が予告されている [87]。

表 1.1: 2022 年に公表された NIST PQC 標準化プロジェクト追加署名の Call for proposals [101] における安全性レベルと計算量の対応表。各レベルは古典，量子のどちらか一方の基準を満たすものとして定義されている。

レベル	量子回路の（最大深さ）×（ゲート数）	古典論理ゲート数
レベル 1	2^{157}	2^{143}
レベル 2	–	2^{146}
レベル 3	2^{221}	2^{207}
レベル 4	–	2^{210}
レベル 5	2^{285}	2^{272}

1.3.2 米国以外での動向

米国以外でも世界各国の機関が調査活動を行い，調査レポートの出版 [20, 93]，各国における PQC 標準方式または推奨暗号方式の選定 [6, 34, 17, 120, 33, 35, 102, 131] を進めている。国際的な機関では ISO/IEC[68]，IETF[63] 等で移行，標準化の議論が進められている。

各国の政府機関から PQC の標準暗号リスト，推奨暗号リストが公表されている。代表的なものを表 1.2 にまとめる。多くの国が NIST PQC 標準化プロジェクトに提案された暗号方式を採用しているが，FrodoKEM のように NIST PQC 標準化プロジェクトの第 3 ラウンド以降の選考に漏れた方式，Classic McEliece のように第 4 ラウンド選考中の状態で選ばれた例も存在する。また，多くの機関が NIST 標準方式の単独利用ではなく古典的安全性がよく知られている RSA や ECDSA とのハイブリッドを推奨していること，レベル 3 以上のパラメータ利用を推奨していることも特徴的である。国家による標準暗号以外でも Streamlined NTRU Prime[22] のように OpenSSH の実装 [104] を通じて実用化されている方式も存在する。

韓国では量子耐性暗号研究団の主催する KpqC プロジェクト [178] が耐量子計算機性を持つ公開鍵暗号方式と署名方式の公募を 2022 年に開始している。2022 年 10 月の締切までに公開鍵暗号方式・鍵共有が 8 方式，署名方式が 9 方式応募された。2022 年 11 月に第 1 ラウンドを開始，2023 年 12 月に第 2 ラウンドの選考が開始され，2025 年 1 月に共通鍵暗号に基づく MPC-in-the-Head パラダイムの署名方式 AIMer，および格子に基づく公開鍵暗号方式 NTRU+，署名方式 HAETAE，鍵交換方式 SMAUG-T の 4 方式が最終方式として選ばれたことがアナウンスされた。

中国では中国暗号学会（CACR）が中心となり PQC の公募を行っている [143]。2018 年 6 月の募集要項に従い 2019 年 2 月の締切までに公開鍵暗号 38 方式と共通鍵暗号 22 方式が応募されている。2019 年 9 月の第 2 ラウンドの時点で公開鍵暗号 14 方式と共通鍵暗号 10 方式に絞られ，最終的には 2020 年 1 月に一等，二等，三等としてランク付けが発表された [144]。一等として公開鍵暗号方式 LAC.PKE，Aigis-enc，署名方式 Aigis-sig，共通鍵暗号方式 uBlock，Ballet が挙げられている。

日本では CRYPTREC の暗号技術調査ワーキンググループにおいて 2014 年度に PQC の代表的な候補である格子に基づく暗号技術について調査を行い，報告書「格子問題等の困難性に関する調査」を公開している [1]。さらに 2017 年度から 2018 年度にかけて，PQC の代表的な候補である 4 種類の分類（格子に基づく暗号技術，符号に基づく暗号技術，多変数多項式に基づく暗号技術，同種写像に基づく暗号技術）について調査し，報告書にまとめた [2]。また，2021 年度から 2022 年度にかけて，ハッシュ関数に基づく署名技術を加えた 5 種類の技術について調査を行い，報告書 [3] およびガイドライン [4] としてまとめている。

表 1.2: 世界各国の標準暗号，推奨暗号リストの状況。表中の勧告，推奨，許容等はそれぞれのレポートからの翻訳であるため，厳密に同じ意味ではない。許容されているバージョン，安全性レベルなど，詳細は引用先のレポートを参照のこと。

方式の名称	NIST PQC（米）	CNSA 2.0（米）[6]	NCSC（英）[34]	ANSSI（仏）[17]	BSI（独）[120]	NCSC（蘭）[33, 35]	NÚKIB（チェコ）[102]	TRAFICOM（フィンランド）[131]
ML-KEM（CRYSTALS-Kyber）	標準化（FIPS 203 [98]）	勧告 [a]	推奨 [c]	許容 [d]	推奨 [ef]	推奨	推奨 [h]	暗号要件 [k]
FrodoKEM	Round 3	–	–	許容 [d]	推奨 [e]	許容	許容 [i]	–
Classic McEliece	Round 4	–	–	–	推奨 [e]	許容	許容 [i]	–
ML-DSA（CRYSTALS-Dilithium）	標準化（FIPS 204 [97]）	勧告 [a]	推奨 [c]	許容 [d]	推奨 [ef]	推奨/許容 [g]	推奨 [h]	暗号要件 [k]
FN-DSA（FALCON）	標準化中	–	–	許容 [d]	–	推奨/許容 [g]	推奨 [i]	–
XMSS/LMS	標準化（NIST SP 800-208）	勧告 [b]	推奨	許容 [d]	推奨	推奨/許容 [g]	推奨 [j]	–
SLH-DSA SPHINCS$^+$	標準化（FIPS 205 [100]）	–	推奨 [c]	許容 [d]	推奨 [ef]	推奨/許容 [g]	推奨 [i]	暗号要件 [k]

注釈一覧	
a	汎用的な耐量子アルゴリズムとしてレベル 5 パラメータの使用を勧告
b	ソフトウェア・ファームウェアに対する署名のための使用を勧告
c	標準化の最終文書を元に堅牢な実装がされたものの利用を推奨
d	メインストリームの PQC として適切だが，可能な限りパラメータを大きく取ること
e	古典的な安全性が確保された方式とのハイブリッドのみを推奨
f	NIST 標準化の安全性レベル 3,5 を推奨パラメータとする意向
g	緊急性のシナリオによって推奨と許容の方式が異なる
h	単独利用は安全性レベル 5 のみ，他は古典の推奨暗号とのハイブリッド利用とする
i	古典の推奨暗号とのハイブリッド利用とする
j	ファームウェア・ソフトウェアの保護目的での単独利用を推奨
k	古典の推奨暗号とのハイブリッド利用を推奨

1.4 本調査で対象とした PQC の種類

本調査報告書では PQC の調査を格子に基づく暗号技術，符号に基づく暗号技術，多変数多項式に基づく暗号技術，同種写像に基づく暗号技術，ハッシュ関数に基づく署名技術の 5 分類で行った。この分類は，安全性の根拠となる数学的な計算問題の種類に基づいて行われている。

例えば，教科書的な RSA 暗号では 2 つの異なる大きな素数 p, q と指数 d を秘密鍵，積 $N = pq$ と指数 e を公開鍵としている。鍵復元の困難性と素因数分解問題の困難性は多項式時間等価であるため [116, 41]，RSA 暗号の鍵復元の安全性は素因数分解問題の困難性に基づくものと考えることができ，素因数分解に基づく暗号に分類できる。同様に，楕円曲線暗号の場合も例えば楕円曲線上の ElGamal 暗号のように安全性が楕円曲線上の離散対数問題の困難性に基づくため，離散対数問題に基づく暗号に分類できる。

本ガイドライン・報告書で扱う代表的な 5 種類の PQC （格子に基づく暗号技術，符号に基づく暗号技術，多変数多項式に基づく暗号技術, 同種写像に基づく暗号技術，ハッシュ関数に基づく署名技術） も RSA 暗号等と同様に，暗号の安全性がそれぞれ格子問題の困難性，符号復号問題の困難性，多変数代数方程式の求解困難性，同種写像上の計算問

題の困難性，ハッシュ関数の衝突発見困難性に基づいている。そして，これらの問題を量子コンピュータを利用して効率よく解くアルゴリズムはまだ発見されていないことから，上に示した暗号技術は PQC であると期待されている。暗号方式と数学的な計算問題の具体的な関係は各章の第 1 節に記載されている。

これらの 5 種類を選んだ理由は主に研究期間の長さ，研究コミュニティの大きさによる。より細かい歴史的な背景は各章の第 4 節に記載されている。

格子に基づく暗号技術は 1997 年の Ajtai と Dwork による論文 [10] から 25 年以上の歴史を持ち，解読技術である格子アルゴリズムに関しても 50 年の歴史を持つ [46, 72, 77]。符号に基づく暗号技術は McEliece による 1978 年の論文 [81] から 40 年以上の歴史を持ち，解読技術は通信における符号の復号技術であり符号理論として 70 年以上研究が行われている。多変数多項式に基づく暗号技術は Ong と Schnorr による 1983 年の論文 [103] を源流[*10]とし，1988 年の Matsumoto-Imai 暗号 [80] を経て 40 年以上の研究が続けられている。同種写像の計算問題に基づく暗号技術も Couveignes による 1997 年の提案 [42] から 25 年以上研究が続けられている。ハッシュ関数に基づく署名方式は Lamport による 1979 年の論文 [75] から 40 年以上の研究が行われている。

1.5 耐量子計算機暗号調査報告書執筆者リスト

主査	國廣 昇	筑波大学
委員	青木 和麻呂	文教大学
委員	伊藤 忠彦	セコム株式会社
委員	下山 武司	国立情報学研究所
委員	高木 剛	東京大学
委員	高島 克幸	早稲田大学
委員	成定 真太郎	KDDI 総合研究所
委員	廣瀬 勝一	福井大学
委員	安田 貴徳	岡山理科大学
委員	安田 雅哉	立教大学
事務局	篠原 直行	情報通信研究機構
事務局	五十部 孝典	情報通信研究機構
事務局	伊藤 竜馬	情報通信研究機構
事務局	大久保 美也子	情報通信研究機構
事務局	大東 俊博	情報通信研究機構
事務局	小川 一人	情報通信研究機構
事務局	金森 祥子	情報通信研究機構
事務局	黒川 貴司	情報通信研究機構
事務局	高安 敦	情報通信研究機構
事務局	横山 和弘	情報通信研究機構
事務局	吉田 真紀	情報通信研究機構
事務局	青野 良範	情報通信研究機構

[*10] ただし Ong と Schnorr による方式の安全性は素因数分解問題に基づくため耐量子計算機性を持たないことに注意。

第 1 章の参照文献

[1] CRYPTREC 暗号技術調査 WG (暗号解析評価). 格子問題等の困難性に関する調査. CRYPTREC EX-2404-2014, https://www.cryptrec.go.jp/exreport/cryptrec-ex-2404-2014.pdf. 2015-03.

[2] CRYPTREC 暗号技術調査 WG (暗号解析評価). 耐量子計算機暗号の研究動向調査報告書. CRYPTREC TR-2001-2018, https://www.cryptrec.go.jp/report/cryptrec-tr-2001-2018.pdf. 2019-04.

[3] CRYPTREC 暗号技術調査 WG (耐量子計算機暗号). CRYPTREC 耐量子計算機暗号の研究動向調査報告書. CRYPTREC TR-2001-2022, https://www.cryptrec.go.jp/report/cryptrec-tr-2001-2022.pdf. 2023-03.

[4] CRYPTREC 暗号技術調査 WG (耐量子計算機暗号). CRYPTREC 暗号技術ガイドライン (耐量子計算機暗号). CRYPTREC GL-2004-2022, https://www.cryptrec.go.jp/report/cryptrec-gl-2004-2022.pdf. 2023-03.

[5] CRYPTREC 暗号技術調査 WG (高機能暗号). CRYPTREC 暗号技術ガイドライン (高機能暗号). CRYPTREC GL-2005-2022, https://www.cryptrec.go.jp/report/cryptrec-gl-2005-2022.pdf. 2023-03.

[6] National Security Agency. Announcing the Commercial National Security Algorithm Suite 2.0. https://media.defense.gov/2022/Sep/07/2003071834/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS_.PDF. 2022-09. (2024-12-06 閲覧).

[7] National Security Agency. Cryptography Today. https://web.archive.org/web/20150815072948/https://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml. 2015-08. (2024-12-05 Internet Archive 版を確認).

[8] National Security Agency. Frequently Asked Questions, Quantum Computing and Post-Quantum Cryptography. https://media.defense.gov/2021/Aug/04/2002821837/-1/-1/1/Quantum_FAQs_20210804.PDF. 2021-08. (2024-12-01 閲覧).

[9] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, O. Regev. Adiabatic Quantum Computation is Equivalent to Standard Quantum Computation. SIAM J. Comput. Vol. 37, Num. 1 (2007), pp. 166–194.

[10] M. Ajtai, Cynthia Dwork. A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence. STOC. ACM, 1997, pp. 284–293.

[11] G. Alagic et al. Status Report on the First Round of the Additional Digital Signature Schemes for the NIST Post-Quantum Cryptography Standardization Process. NIST IR 8528, https://nvlpubs.nist.gov/nistpubs/ir/2024/NIST.IR.8528.pdf. 2024-10.

[12] G. Alagic et al. Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process. NIST IR 8204, https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8240.pdf. 2019-01.

[13] G. Alagic et al. Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process. NIST IR 8413, https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8413-upd1.pdf. 2022-07.

[14] T. Albash, D. A. Lidar. Adiabatic quantum computation. Rev. Mod. Phys. Vol. 90, Iss. 1 (2018), p. 015002.

[15] Amazon Braket 量子コンピュータ. https://aws.amazon.com/jp/braket/quantum-computers/.

[16] M. V. Anand, E. E. Targhi, G. N. Tabia, D. Unruh. Post-Quantum Security of the CBC, CFB, OFB, CTR, and XTS Modes of Operation. PQCrypto. Vol. 9606. Lecture Notes in Computer Science. Springer, 2016, pp. 44–63.

[17] ANSSI. ANSSI views on the Post-Quantum Cryptography transition (2023 follow up). https://cyber.gouv.fr/sites/default/files/document/follow_up_position_paper_on_post_quantum_cryptography.pdf. 2023-12. (2024-12-06 閲覧).

[18] Y. Aono, S. Liu, T. Tanaka, S. Uno, R. Van Meter, N. Shinohara, R. Nojima. The Present and Future of Discrete Logarithm Problems on Noisy Quantum Computers. IEEE Transactions on Quantum Engineering. Vol. 3 (2022), pp. 1–21.

[19] B. Apolloni, N. Cesa-Bianchi, D. De Falco. A numerical implementation of "quantum annealing". Proceedings of the Ascona-Locarno conference. 1988, pp. 97–111.

[20] GSM Association. Post Quantum Cryptography – Guidelines for Telecom Use Cases. https://www.gsma.com/newsroom/wp-content/uploads/PQ.03-Post-Quantum-Cryptography-Guidelines-for-Telecom-Use-v1.0.pdf. 2024-02. (2024-12-06 閲覧).

[21] E. Barker. Recommendation for Key Management: Part 1 – General. NIST SP 800-57 Part 1 Rev. 5, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf. 2020-05.

[22] D. J. Bernstein et al. NTRU Prime. https://ntruprime.cr.yp.to/. (2024-12-06 閲覧).

[23] D. Bluvstein et al. Logical quantum processor based on reconfigurable atom arrays. Nature. Vol. 626, Num. 7997 (2023), pp. 58–65.

[24] D. Boneh, M. Zhandry. Secure Signatures and Chosen Ciphertext Security in a Quantum Computing World. CRYPTO (2). Vol. 8043. Lecture Notes in Computer Science. Springer, 2013, pp. 361–379.

[25] C. Boutin. NIST Releases First 3 Finalized Post-Quantum Encryption Standards. https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards. 2023-08. (2024-12-06 閲覧).

[26] G. Brassard. Searching a Quantum Phone Book. Science. Vol. 275, Num. 5300 (1997), pp. 627–628.

[27] G. Brassard, P. Høyer and A. Tapp. Quantum Cryptanalysis of Hash and Claw-Free Functions. LATIN. Vol. 1380. Lecture Notes in Computer Science. Springer, 1998, pp. 163–169.

[28] S. Bravyi, D. P. DiVincenzo, R. Oliveira, B. M. Terhal. The complexity of stoquastic local Hamiltonian problems. Quantum Inf. Comput. Vol. 8, Num. 5 (2008), pp. 361–385.

[29] J. Buchmann et al. Post-Quantum Signatures. (2004). https://eprint.iacr.org/2004/297.

[30] J. Sevilla and C. J. Riedel. Forecasting timelines of quantum computing. (2020). arXiv: 2009.05045.

[31] J.-Y. Cai. Shor's algorithm does not factor large integers in the presence of noise. Science China Information Sciences. Vol. 67, Num. 7 (2024).

[32] W. Castryck, T. Decru. An Efficient Key Recovery Attack on SIDH. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 423–447.

[33] National Cyber Security Centre. Guidelines for quantum-safe transport-layer encryption. https://www.ncsc.nl/documenten/publicaties/2022/juli/guidelines-for-quantum-safe-transport-layer-encryption/guidelines-for-quantum-safe-transport-layer-encryption. 2022-07. (2024-12-06 閲覧).

[34] National Cyber Security Centre. Next steps in preparing for post-quantum cryptography. https://www.ncsc.gov.uk/pdfs/whitepaper/next-steps-preparing-for-post-quantum-cryptography.pdf. 2024-08. (2024-12-06 閲覧).

[35] National Cyber Security Centre. The PQC Migration Handbook, Guidelines for migrating to post-quantum cryptography (Version 2). https://publications.tno.nl/publication/34641918/oicFLj/attema-2023-pqc.pdf. 2023-12. (2024-12-06 閲覧).

[36] P. Chapman. Introducing the World's Most Powerful Quantum Computer. https://ionq.com/posts/october-01-2020-introducing-most-powerful-quantum-computer. 2020-10. (2024-12-01 閲覧).

[37] C. Chevalier, E. Ebrahimi, QH Vu. On Security Notions for Encryption in a Quantum World. INDOCRYPT. Vol. 13774. Lecture Notes in Computer Science. Springer, 2022, pp. 592–613.

[38] Google Quantum AI and Collaborators. Quantum error correction below the surface code threshold. Nature. Vol. 616, Num. 7955 (2024).

[39] H. Collins, C. Nay. IBM Unveils 400 Qubit-Plus Quantum Processor and Next-Generation IBM Quantum System Two. https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two. 2022-11. (2024-12-06 閲覧).

[40] Atom Computing. Quantum startup Atom Computing first to exceed 1,000 qubits. https://atom-computing.com/quantum-startup-atom-computing-first-to-exceed-1000-qubits/. 2023-10. (2024-12-01 閲覧).

[41] J.-S. Coron, A. May. Deterministic Polynomial-Time Equivalence of Computing the RSA Secret Key and Factoring. J. Cryptol. Vol. 20, Num. 1 (2007), pp. 39–50.

[42] J.-M. Couveignes. Hard Homogeneous Spaces. Cryptology ePrint Archive, Paper 2006/291. 2006. https://eprint.iacr.org/2006/291.

[43] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, J. M. Gambetta. Validating quantum computers using randomized model circuits. Phys. Rev. A. Vol. 100, Iss. 3 (2019), p. 032328.

[44] D-Wave. Ahead of the Game: D-Wave Delivers Prototype of Next-Generation Advantage2 Annealing Quantum Computer. https://www.dwavesys.com/company/newsroom/press-release/ahead-of-the-game-d-wave-delivers-prototype-of-next-generation-advantage2-annealing-quantum-computer/. 2022-06. (2024-12-01 閲覧).

[45] D-Wave. The Most Connected and Powerful Quantum Computer Built for Business. https://www.dwavesys.com/solutions-and-products/systems/. (2024-12-01 閲覧).

[46] U. Dieter. How to calculate shortest vectors in a lattice. Mathematics of Computation. Vol. 29 (1975), pp. 827–833.

[47] J. Ding, G. Spallitta, R. Sebastiani. Experimenting with D-Wave quantum annealers on prime factorization problems. Frontiers Comput. Sci. Vol. 6 (2024).

[48] M. Ekerå, J. Gärtner. Extending Regev's Factoring Algorithm to Compute Discrete Logarithms. PQCrypto (2). Vol. 14772. Lecture Notes in Computer Science. Springer, 2024, pp. 211–242.

[49] ETSI. Quantum-Safe Cryptography (QSC); Limits to quantum computing applied to symmetric key sizes. https://www.etsi.org/deliver/etsi_gr/QSC/001_099/006/01.01.01_60/gr_QSC006v010101p.pdf. 2017-02. (2024-12-01 閲覧).

[50] National Cybersecurity Center of Excellence. Migration to Post-Quantum Cryptography. https://www.nccoe.nist.gov/crypto-agility-considerations-migrating-post-quantum-cryptographic-algorithms.

[51] European Quantum Flagship. Strategic Research and Industry Agenda. https://qt.eu/media/pdf/Strategic-Reseach-and-Industry-Agenda-2030.pdf. 2024-02.

[52] A. Frank et al. Quantum supremacy using a programmable superconducting processor. Nature. Vol. 574, Num. 7779 (2019), pp. 505–510.

[53] K. Fujii. Quantum speedup in stoquastic adiabatic quantum computation. (2018). arXiv: 1803.09954.

[54] K. Fujii. Quantum speedup in stoquastic adiabatic quantum computation. 2019-01. QIP 2019 Poster session https://jila.colorado.edu/qip2019/qip2019_posters_monday.pdf.

[55] J. Gambetta. The hardware and software for the era of quantum utility is here. https://jila.colorado.edu/qip2019/qip2019_posters_monday.pdf. 2023-12. (2024-12-01 閲覧).

[56] C. Gidney, M. Ekerå. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. Quantum. Vol. 5 (2021), p. 433.

[57] É. Gouzien, D. Ruiz, F.-M. Le Régent, J. Guillaud, N. Sangouard. Performance Analysis of a Repetition Cat Code Architecture: Computing 256-bit Elliptic Curve Logarithm in 9 Hours with 126 133 Cat Qubits. Phys. Rev. Lett. Vol. 131, Iss. 4 (2023), p. 040602.

[58] É. Gouzien, N. Sangouard. Factoring 2048-bit RSA Integers in 177 Days with 13 436 Qubits and a Multimode Memory. Phys. Rev. Lett. Vol. 127, Iss. 14 (2021), p. 140503.

[59] ASC X9 Quantum Computing Risk Study Group. Quantum Computing Risks to the Financial Services Industry. https://x9.org/download-qc-ir/. 2022-11. (2024-12-05 閲覧).

[60] FS-ISAC's post-quantum cryptography working group. Preparing for a post-quantum world by managing cryptographic risk. https://www.fsisac.com/hubfs/Knowledge/PQC/Preparing/ForAPostQuantumWorldByManagingCryptographicRisk.pdf?hsLang=en. 2023-03. (2024-12-05 閲覧).

[61] L. K. Grover. A fast quantum mechanical algorithm for database search. STOC. ACM, 1996, pp. 212–219.

[62] N. N. Hegade, K. Paul, F. Albarrán-Arriagada, X. Chen, E. Solano. Digitized adiabatic quantum factorization. Phys. Rev. A. Vol. 104, Iss. 5 (2021), p. L050403.

[63] P. E. Hoffman, S. Celi. Post-Quantum Use In Protocols (pquip). https://datatracker.ietf.org/wg/pquip/about/. (2024-12-06 閲覧).

[64] IBM Quantum Platform. https://quantum.ibm.com/.

[65] IBM、次世代量子プロセッサーおよび IBM Quantum System Two を発表するとともに、 実用的な量子コンピューティングの時代の前進に向けロードマップを拡張. https://jp.newsroom.ibm.com/2023-12-05-IBM-Debuts-Next-Generation-Quantum-Processor-IBM-Quantum-System-Two,-Extends-Roadmap-to-Advance-Era-of-Quantum-Utility. 2023-12. (2024-12-01 閲覧).

[66] T. Ichikawa et al. Current numbers of qubits and their uses. Nature Reviews Physics. Vol. 6, Num. 6 (2024), pp. 345–347.

[67] Intel's New Chip to Advance Silicon Spin Qubit Research for Quantum Computing. https://www.intel.com/content/www/us/en/newsroom/news/quantum-computing-chip-to-advance-research.html. 2023-06. (2024-12-01 閲覧).

[68] ISO. PQCRYPTO Post-quantum cryptography for long-term security. https://www.iso.org/organization/5984715.html. (2024-12-06 閲覧).

[69] P. John. Quantum Computing in the NISQ era and beyond. Quantum. Vol. 2 (2018), p. 79.

[70] S. P. Jordan. Quantum Computation Beyond the Circuit Model. 2008. arXiv: 0809.2307.

[71] T. Jörg, F. Krzakala, G. Semerjian, F. Zamponi. First-Order Transitions and the Performance of Quantum Algorithms in Random Optimization Problems. Phys. Rev. Lett. Vol. 104, Iss. 20 (2010), p. 207206.

[72] R. Kannan. Improved Algorithms for Integer Programming and Related Lattice Problems. STOC. ACM, 1983, pp. 193–206.

[73] M. Kaplan, G. Leurent, A. Leverrier, M. Naya-Plasencia. Quantum Differential and Linear Cryptanalysis. IACR Trans. Symmetric Cryptol. Vol. 2016, Num. 1 (2016), pp. 71–94.

[74] H. Kuwakado, M. Morii. Security on the quantum-type Even-Mansour cipher. ISITA. IEEE, 2012, pp. 312–316.

[75] L. Lamport. Constructing digital signatures from a one-way function. SRI International Technical Report, CSL-98. 1979-10.

[76] J. Lavoie, Z. Vernon. Beating classical computers with Borealis. https://www.xanadu.ai/blog/beating-classical-computers-with-Borealis. 2022-06. (2024-12-01 閲覧).

[77] A. K. Lenstra, H. W. Lenstra, L. Lovász. Factoring polynomials with rational coefficients. Mathematische Annalen. Vol. 261, Num. 4 (1982), pp. 515–534.

[78] Linwen. Quantum Leap: China's 504-Qubit Chip Narrows the US Gap. https://thechinaacademy.org/quantum-leap-chinas-504-qubit-chip-narrows-the-us-gap/. 2024-04. (2024-12-01 閲覧).

[79] M. Mariantoni. Building a superconducting quantum computer (Invited Talk). PQCrypto 2014. 2024-10. (2024-12-01 閲覧) 暗号危殆化の予測については動画 https://www.youtube.com/watch?v=wWHAs--HA1c の 49:30 で述べられている.

[80] T. Matsumoto, H. Imai. Public Quadratic Polynominal-Tuples for Efficient Signature-Verification and Message-Encryption. EUROCRYPT. Vol. 330. Lecture Notes in Computer Science. Springer, 1988, pp. 419–453.

[81] R. J. McEliece. A Public-Key Cryptosystem Based On Algebraic Coding Theory. Deep Space Network Progress Report. Vol. 44 (1978), pp. 114–116.

[82] C. McGeoch, P. Farre. D-Wave Advantage システム: 概要. https://dwavejapan.com/app/uploads/2020/12/14-1049A-A_J-The_D-Wave_Advantage_System_An_Overview_0-.pdf. 2020-12. (2024-12-01 閲覧).

[83] A. Mirko, Z. H. Saleem, K. Muir. Experimental study of Shor's factoring algorithm using the IBM Q Experience. Physical Review A. Vol. 100, Iss. 1 (2019), p. 012305.

[84] D. Moody. Announcement: The End of the 3rd Round – the First PQC Algorithms to be Standardized. https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/G0DoD7lkGPk/m/f3Hl0sh3AgAJ. 2022-07. (2024-12-06 閲覧).

[85] D. Moody. Are we there yet? An Update on the NIST PQC Standardization Project. https://csrc.nist.gov/csrc/media/Presentations/2024/update-on-the-nist-pqc-standardization-project/images-media/moody-are-we-there-yet-pqc-pqc2024.pdf. 2024-04. (2024-12-01 閲覧).

[86] D. Moody. Post-Quantum Cryptography: NIST's Plan for the Future. https://pqcrypto2016.jp/data/pqc2016_nist_announcement.pdf. 2016-02. (2024-12-06 閲覧).

[87] D. Moody. security category reference. https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/OmLRb2rQyN4/m/_7y82chdAQAJ. 2022-09. (2024-12-06 閲覧).

[88] D. Moody, R. Perlner, A. Regenscheid, A. Robinson, D. Cooper. Transition to Post-Quantum Cryptography Standards. NIST IR 8547 (initial public draft), https://nvlpubs.nist.gov/nistpubs/ir/2024/NIST.IR.8547.ipd.pdf. 2024-11. (2025-02-17 閲覧).

[89] D. Moody et al. Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process. NIST IR 8309, https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8309.pdf. 2020-07.

[90] M. Mosca. Cybersecurity in a quantum world: will we be ready? Workshop on Cybersecurity in a Post-Quantum World. Session 8. 2015-04. (2024-02-29 閲覧).

[91] M. Mosca, M. Piani. 2023 Quantum Threat Timeline Report. https://globalriskinstitute.org/publication/2023-quantum-threat-timeline-report/. 2023-12. (2024-12-02 閲覧).

[92] National Security Memorandum on Promoting United States Leadership in Quantum Computing While Mitigating Risks to Vulnerable Cryptographic Systems. https://www.whitehouse.gov/briefing-room/statements-releases/2022/05/04/national-security-memorandum-on-promoting-united-states-leadership-in-quantum-computing-while-mitigating-risks-to-vulnerable-cryptographic-systems/. 2022-05. (2025-01-11 閲覧).

[93] NCSA. Guidelines for Post – Quantum Readiness. https://www.navy.mi.th/storage/frontend/article/23852/file/th/Quantum%20Readiness.pdf. 2023-12. (2024-12-06 閲覧).

[94] W. Newhouse et al. Migration to Post-Quantum Cryptography: Preparation for Considering the Implementation and Adoption of Quantum Safe Cryptography. NIST SP 1800-38 (initial preliminary draft), https://csrc.nist.gov/pubs/sp/1800/38/iprd-(1). 2023-12. (2025-02-17 閲覧).

[95] Z. Ni et al. Beating the break-even point with a discrete-variable-encoded logical qubit. Nature. Vol. 616, Num. 7955 (2023), pp. 56–60.

[96] NIST. Digital Signature Standard (DSS). NIST FIPS 186-5, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf. 2023-02.

[97] NIST. Module-Lattice-Based Digital Signature Standard. NIST FIPS 204, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.204.pdf. 2024-08.

[98] NIST. Module-Lattice-Based Key-Encapsulation Mechanism Standard. NIST FIPS 203, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.pdf. 2024-08.

[99] NIST. Standardization of additional digital signature schemes, call for proposals. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/call-for-proposals-dig-sig-sept-2022.pdf. 2022-10. (2024-03-05 閲覧).

[100] NIST. Stateless Hash-Based Digital Signature Standard. NIST FIPS 205, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.205.pdf. 2024-08.

[101] NIST. Submission requirements and evaluation criteria for the post-quantum cryptography standardization process. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf. 2016-12. (2024-03-05 閲覧).

[102] NÚKIB. Minimum requirements for cryptographic algorithms – Cryptographic security recommendations. https://nukib.gov.cz/download/publications_en/Minimum_Requirements_for_Cryptographic_Algorithms_final.pdf. 2023-11. (2024-12-06 閲覧).

[103] H. Ong, C. P. Schnorr. Signatures through Approximate Representation by Quadratic Forms. CRYPTO. Plenum Press, New York, 1983, pp. 117–131.

[104] OpenSSH 9.0 was released. https://www.openssh.com/txt/release-9.0. 2022-04. (2024-12-06 閲覧) Streamlined NTRU Prime と X25519 を組み合わせたハイブリッド鍵交換は 8.5 で試験的に実装され，9.0 からはデフォルトで利用される仕様となっている.

[105] J. Park et al. Rydberg-atom experiment for the integer factorization problem. Physical Review Research. Vol. 6, Iss. 2 (2024), p. 023241.

[106] X. Peng, Z. Liao, N. Xu, G. Qin, X. Zhou, D. Suter, J. Du. Quantum Adiabatic Algorithm for Factorization and Its Experimental Implementation. Physical Review Letters. Vol. 101, Iss. 22 (2008), p. 220405.

[107] L. Phab, S. Louise, R. Sirdey. First Attempts at Cryptanalyzing a (Toy) Block Cipher by Means of Quantum Optimization Approaches. J. Comput. Sci. Vol. 69 (2023), p. 102004.

[108] Post-Quantum Cryptography Lounge. https://www.safecrypto.eu/pqclounge/.

[109] pqc-forum. https://groups.google.com/a/list.nist.gov/g/pqc-forum. 2016/08/01 開始.

[110] L. Qiu, M. Alam, A. Ash-Saki, S. Ghosh. Resiliency analysis and improvement of variational quantum factoring in superconducting qubit. ISLPED. ACM, 2020, pp. 229–234.

[111] Quantinuum H-Series quantum computer accelerates through 3 more performance records for quantum volume. https://www.quantinuum.com/blog/quantinuum-h-series-quantum-computer-accelerates-through-3-more-performance-records-for-quantum-volume. 2023-06. (2024-12-01 閲覧).

[112] Quantinuum's H-Series hits 56 physical qubits that are all-to-all connected, and departs the era of classical simulation. https://www.quantinuum.com/blog/quantinuums-h-series-hits-56-physical-qubits-that-are-all-to-all-connected-and-departs-the-era-of-classical-simulation. 2024-06. (2024-12-01 閲覧).

[113] S. Ragavan, V. Vaikuntanathan. Space-Efficient and Noise-Robust Quantum Factoring. CRYPTO (6). Vol. 14925. Lecture Notes in Computer Science. Springer, 2024, pp. 107–140.

[114] S. Ramos-Calderer, C. Bravo-Prieto, R. Lin, E. Bellini, M. Manzano, N. Aaraj, J. I. Latorre. Solving systems of Boolean multivariate equations with quantum annealing. Phys. Rev. Res. Vol. 4, Iss. 1 (2022), p. 013096.

[115] O. Regev. An Efficient Quantum Factoring Algorithm. arXiv: 2308.06572.

[116] R. L. Rivest, A. Shamir, L. M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Commun. ACM. Vol. 21, Num. 2 (1978), pp. 120–126.

[117] Round 2 Additional Signatures. https://csrc.nist.gov/projects/pqc-dig-sig/round-2-additional-signatures. 2024-10. (2024-12-06 閲覧).

[118] Round 4 Submissions. https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions. 2022-07. (2024-12-06 閲覧).

[119] C. P. Schnorr. Fast Factoring Integers by SVP Algorithms, corrected. Cryptology ePrint Archive, Paper 2021/933. 2021. https://eprint.iacr.org/2021/933.

[120] Federal office for information security. Cryptographic mechanisms: recommendations and key lengths version: 2024-1. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.html. 2024-02. (2024-12-05 閲覧).

[121] Selected Algorithms. 2025-03. https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms. (2025-03-30 閲覧).

[122] P. W. Shor. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. FOCS. IEEE Computer Society, 1994, pp. 124–134.

[123] P. W. Shor. Fault-Tolerant Quantum Computation. FOCS. IEEE Computer Society, 1996, pp. 56–65.

[124] P. W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput. Vol. 26, Num. 5 (1997), pp. 1484–1509.

[125] V. V. Sivak et al. Real-time quantum error correction beyond break-even. Nature. Vol. 616, Num. 7955 (2023), pp. 50–55.

[126] U. Skosana, M. Tame. Demonstration of Shor's factoring algorithm for $N = 21$ on IBM quantum processors. Scientific Reports. Vol. 11, Num. 16599 (2021).

[127] M. Sobhani, Y. Chai, T. Hartung, K. Jansen. Variational Quantum Eigensolver Approach to Prime Factorization on IBM's Noisy Intermediate Scale Quantum Computer. arXiv: 2410.01935.

[128] E. G. Johansen and T. Simula. Prime number factorization using a spinor Bose-Einstein condensate-inspired topological quantum computer. Quantum Inf. Process. Vol. 21, Num. 1 (2022), p. 31.

[129] The Cryptographers' Panel. https://www.rsaconference.com/library/presentation/usa/2023/the%20cryptographers%20panel. 2023-04. RSA Conference 2023 (2024-12-05 閲覧).

[130] The Leap quantum cloud service. https://www.dwavesys.com/solutions-and-products/cloud-platform/.

[131] TRAFICOM. Kryptografiset vahvuusvaatimukset luottamuksellisuuden suojaamiseen – kansalliset turvallisuusluokat. https://www.kyberturvallisuuskeskus.fi/sites/default/files/media/file/Kryptografiset_vahvuusvaatimukset_-_kansalliset_turvallisuusluokat_0.pdf. 2024-09. (2024-12-06 閲覧).

[132] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, R. Cleve, I. L. Chuang. Experimental Realization of an Order-Finding Algorithm with an NMR Quantum Computer. Phys. Rev. Lett. Vol. 85, Iss. 25 (2000), pp. 5452–5455.

[133] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, I. L. Chuang. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance. Nature. Vol. 414, Num. 6866 (2001), pp. 883–887.

[134] D.-S. Wang. A comparative study of universal quantum computing models: Toward a physical unification. Quantum Eng. Vol. 3, Num. 4 (2021).

[135] W. Wang, Z. You, S. Wang, Z. Tang, H. Ian. Computing Shor's algorithmic steps with classical light beams. Scientific Reports. Vol. 12, Num. 21157 (2022).

[136] D. Willsch, P. Hanussek, G. Hoever, M. Willsch, F. Jin, H. De Raedt, K. Michielsen. The State of Factoring on Quantum Computers. 2024. arXiv: 2410.14397.

[137] S. Xu et al. Digital Simulation of Projective Non-Abelian Anyons with 68 Superconducting Qubits. Chinese Physics Letters. Vol. 40, Num. 6 (2023), p. 060301.

[138] B. Yan et al. Factoring integers with sublinear resources on a superconducting quantum processor. arXiv: 2212.12372.

[139] C. Zalka. Grover's quantum searching algorithm is optimal. Phys. Rev. A. Vol. 60, Iss. 4 (1999), pp. 2746–2751.

[140] J. Zander. Advancing science: Microsoft and Quantinuum demonstrate the most reliable logical qubits on record with an error rate 800x better than physical qubits. https://blogs.microsoft.com/blog/2024/04/03/advancing-science-microsoft-and-quantinuum-demonstrate-the-most-reliable-logical-qubits-on-record-with-an-error-rate-800x-better-than-physical-qubits/. 2024-04. (2024-12-01 閲覧).

[141] デジタル庁, 総務省, 経済産業省. 暗号強度要件（アルゴリズム及び鍵長選択）に関する設定基準. CRYPTREC LS-0003-2022r1, https://www.cryptrec.go.jp/list/cryptrec-ls-0003-2022r1.pdf. 2022-03.

[142] デジタル庁, 総務省, 経済産業省. 電子政府における調達のために参照すべき暗号のリスト（CRYPTREC 暗号リスト）. CRYPTREC LS-0001-2022r1, https://www.cryptrec.go.jp/list/cryptrec-ls-0001-2022r1.pdf. 2024-05.

[143] 中国密码学会. 全国密码算法设计竞赛通知. https://sfjs.cacrnet.org.cn/site/content/309.html. 2018-06. (2025-01-11 閲覧).

[144] 中国密码学会. 关于全国密码算法设计竞赛算法评选结果的公示. https://sfjs.cacrnet.org.cn/site/content/854.html. 2020-01. (2025-01-11 閲覧).

[145] 橋本 俊和, 梅木 毅伺, 柏﨑 貴大, 井上 飛鳥. 連続量光量子コンピュータに向けた光技術. https://www.rd.ntt/research/JN202304_21560.html. 2023-04. (2024-12-01 閲覧).

[146] 豊田 健二. 究極のコンピュータへ「もう一つの道」 イオンで可視化する量子情報. https://resou.osaka-u.ac.jp/ja/story/2021/specialite_002_6. 2021. (2024-12-01 閲覧).

[147] 宮地 充子. 楕円曲線の理論的及び実用的可能性. IEICE FUNDAMENTALS REVIEW. Vol. 14, Num. 4 (2021), pp. 329–336.

[148] 細山田 光倫. 量子コンピュータが共通鍵暗号の安全性に及ぼす影響の調査及び評価. CRYPTREC EX-2901-2019, https://www.cryptrec.go.jp/exreport/cryptrec-ex-2901-2019.pdf. 2020-01.

[149] 縫田 光司. 耐量子計算機暗号. 森北出版, 2020.

[150] 伊藤 公平. 量子計算. 2010-02. https://www.ieice-hbkb.org/files/ad_base/view_pdf.html?p=/files/S2/S2gun_05hen_03.pdf. 電子情報通信学会 知識ベース 知識の森 S2 群（ナノ・量子・バイオ）5 編（量子通信と量子計算）3 章.

[151] 小林 和淑. ムーンショット目標 6 公開シンポジウム 2023 〜誤り耐性型汎用量子コンピュータの実現を目指して〜. https://www.youtube.com/watch?v=ebkT0LyKIKk. 2023-03.

[152] 国立国会図書館調査及び立法考査局. 量子情報技術：科学技術に関する調査プロジェクト報告書. 2022-03. https://www.ndl.go.jp/jp/diet/publication/document/2022/index.html.

[153] 国立大学法人東北大学, 日本電気株式会社. 新開発の 8 量子ビット量子アニーリングマシンを利用して東北大学と NEC が将来のコンピュータシステムに関する共同研究を開始. https://jpn.nec.com/press/202306/20230628_01.html. 2023-06. (2024-12-01 閲覧).

[154] 国立研究開発法人科学技術振興機構. 目標 6 2050 年までに、経済・産業・安全保障を飛躍的に発展させる誤り耐性型汎用量子コンピュータを実現. https://www.jst.go.jp/moonshot/program/goal6/index.html. (2024-12-01 閲覧).

[155] 富士通. 量子コンピュータの誤り耐性量子計算を解説！エラー訂正とエラー緩和の最新トレンドを紐解く. https://activate.fujitsu/ja/key-technologies-article/ta-fault-tolerant-quantum-computation-20240515. 2024-05. (2024-12-01 閲覧).

[156] 向井 寛人, 朝永 顕成, 蔡 兆申. 超伝導量子コンピュータの基礎と最先端. 低温工学. Vol. 53, Num. 5 (2018), pp. 278–286.

[157] 鈴木 教洋. 日立の量子コンピュータ研究開発戦略. https://www8.cao.go.jp/cstp/ryoshigijutsu/jitsuyo_wg/3kai/siryo2-2.pdf. 2022-12. (2024-12-01 閲覧).

[158] 高安 敦. Shor のアルゴリズム実装動向調査. CRYPTREC EX-2005-2020, https://www.cryptrec.go.jp/exreport/cryptrec-ex-3005-2020.pdf. 2021-06.

[159] 新方式の量子コンピュータを実現 ー世界に先駆けて汎用型光量子計算プラットフォームが始動ー. https://group.ntt/jp/newsrelease/2024/11/08/241108a.html. 2024-11. (2024-12-01 閲覧).

[160] 日本経済新聞. 量子計算機で新会社　富士通・日立など 10 社 産学で商用化. https://ohmori.ims.ac.jp/news/2024/02/27/2607/. 2024-02. (2024-12-01 閲覧).

[161] 日立、量子コンピュータの実用化に向けて 量子ビットの寿命を 100 倍以上長く安定化させる操作技術を開発. https://www.hitachi.co.jp/New/cnews/month/2024/06/0617.html. 2024-06. (2024-12-01 閲覧).

[162] 清藤 武暢, 四方 順司. 量子コンピュータが共通鍵暗号の安全性に与える影響. 金融研究. Vol. 38, Num. 1 (2019), pp. 45–72. https://cir.nii.ac.jp/crid/1523106604811659392.

[163] 樽茶 清悟. 拡張性のあるシリコン 量子コンピュータ技術の開発. https://www.jst.go.jp/moonshot/sympo/20230328/pdf/01_20230328_tarucha.pdf. 2023-03. (2024-12-01 閲覧).

[164] 理化学研究所. シリコン量子ビットの高温動作に成功 ー大型冷却装置が不要に、センサーなど幅広い量子ビット応用へー. https://www.riken.jp/press/2019/20190124_3/. 2019-01. (2024-12-01 閲覧).

[165] 理化学研究所. シリコン量子ビットの高精度読み出しを実現 ー半導体系の誤り耐性量子コンピュータの実現に前進ー. https://www.riken.jp/press/2024/20240213_2/index.html. 2024-02. (2024-12-01 閲覧).

[166] 理化学研究所. 量子コンピュータを利用できる「量子計算クラウドサービス」開始 ー国産超伝導量子コンピュータ初号機の公開ー. https://www.riken.jp/pr/news/2023/20230324_1/. 2023-03. (2024-12-01 閲覧).

[167] 理化学研究所. 量子コンピュータ開発に挑む若手研究者たち. https://www.riken.jp/pr/closeup/2023/20230904_1/index.html. 2023-09. (2024-12-01 閲覧).

[168] 産業技術総合研究所. 独自のアーキテクチャを用いた超伝導量子アニーリングマシンを実現. https://www.aist.go.jp/aist_j/new_research/2021/nr20210706/nr20210706.html. 2021-07. (2024-12-01 閲覧).

[169] 大関 真之. 量子アニーリングが拓く機械学習と計算技術の新時代 (量子システム推定の数理). 数理解析研究所講究録. Vol. 2059 (2017), pp. 13–23. https://cir.nii.ac.jp/crid/1050564288163922560.

[170] 文部科学省 科学技術・学術政策研究所科学技術予測センター. 第 11 回科学技術予測調査 デルファイ調査. https://nistep.repo.nii.ac.jp/?action=repository_uri&item_id=6692&file_id=13&file_no=3. 2020-06. (2024-12-05 閲覧).

[171] 山口 純平, 伊豆 哲也. イジング計算を用いた暗号解析について. オペレーションズ・リサーチ：経営の科学. Vol. 67, Num. 6 (2022), pp. 290–296. https://cir.nii.ac.jp/crid/1520011030559130112.

[172] 山口 純平, 伊豆 哲也, 國廣 昇. 素因数分解問題に対する Shor アルゴリズムの実装と量子計算機シミュレータを用いた実験. 暗号と情報セキュリティシンポジウム (SCIS 2023). 2023-01, 4A2–3.

[173] 大塩 耕平. アナログ量子シミュレータの開発動向と応用. https://www.mizuho-rt.co.jp/publication/others/pdf/mhrt003_01.pdf. 2024-03. (2024-12-06 閲覧).

[174] 塩見 英久. マイクロ波技術者から学ぶ超伝導量子コンピュータ入門. MWE2023 マイクロウェーブ ワークショップ プログラム FR6A 基礎講座. (2023). https://apmc-mwe.org/mwe2024/pdf/tut23/FR6A-1.pdf.

[175] 徳永 裕己. 誤り耐性量子コンピュータの早期実現に向けた取り組み. NTT 技術ジャーナル. Vol. 35, Num. 9 (2023), pp. 26–29.

[176] 大森 賢治. 大規模・高コヒーレンスな動的原子アレー型・誤り耐性量子コンピュータ. https://www.youtube.com/watch?v=OIVQ5ZmdCEo. 2024-03. (2024-12-01 閲覧).

[177] 大阪大学 量子情報・量子生命研究センター. 【プレスリリース】大阪大学に設置した超伝導量子コンピュータ国産 3 号機の クラウドサービスを開始. https://qiqb.osaka-u.ac.jp/20231220pr/. 2023-12. (2024-12-01 閲覧).

[178] 量子耐性暗号研究団. KpqC. https://kpqc.or.kr/. (2024-12-06 閲覧).

[179] 富田 隆文. 冷却原子型量子コンピュータの急速な発展とその展望について. https://www-conf.kek.jp/joint-colloquium/slides/Tomita.pdf. 2024-03. (2024-12-01 閲覧).

[180] 満保 雅浩. 公開鍵暗号. 映像情報メディア学会誌. Vol. 69, Num. 9 (2015), pp. 714–720.

第 2 章

PQC の活用方法

将来，一定以上の能力を持つ量子コンピュータが登場した場合には，既存の公開鍵暗号が解読される（破られる）という脅威が指摘されている [1, 20]。本章では，現在，標準的に用いられている公開鍵暗号の解読が可能となる水準の量子コンピュータを Cryptographically Relevant Quantum Computer（CRQC）と記載し，CRQC を用いた攻撃に対しても安全な性質を「耐量子計算機性」と記載する。また，耐量子計算機暗号（Post-Quantum Cryptography: PQC）とは，耐量子計算機性を持つ暗号アルゴリズムを意味し，本稿の対象である公開鍵暗号アルゴリズム以外にも，共通鍵暗号やハッシュ関数も含まれるものとする [1]。加えて，「耐量子計算機性を持つ情報システム」とは，CRQC を用いた攻撃に対しても安全な情報システムを示すものとする。

ある情報システムが，既存の公開鍵暗号を利用していた場合，その情報システムは，将来における CRQC を用いた攻撃の脅威に晒されることになる。そのような脅威への対応方法としては，情報システム内の（耐量子計算機性を持たない）既存の公開鍵暗号方式部分を，耐量子計算機性を持つ公開鍵暗号方式に置き換えることで，その情報システムに耐量子計算機性を持たせることが考えられる。

なお，耐量子計算機性を持たせるためには異なるアプローチも考えられる。例えば，今まで公開鍵暗号を利用していた情報システムを，公開鍵暗号を利用しない仕組みに置き換えるアプローチである。具体的には，信頼できる特使等の別の情報共有手段を利用し，通信相手と共通鍵の事前共有を行う方法である。しかし，このアプローチでは，情報システムの「スケーラビリティ」[*1]が損なわれることが予想され，場合によっては実現不可能なコストが発生する。

現在，普及している情報システムの中には，公開鍵暗号を利用することにより，そのサービスのスケーラビリティを維持しているものも多い。インターネットはその代表例であり，通信相手を認証する用途等で公開鍵暗号を利用することにより，大規模な通信ネットワークの構築及び維持を実現している [4, 11, 14, 18]。このような大規模情報システムにおいて，仮に，耐量子計算機性を持たせるために公開鍵暗号の利用を取りやめた場合，スケーラビリティが損なわれ，その結果，維持・運用コストが大きく上昇してシステムの維持も困難となる。このため，公開鍵暗号を利用した情報システムの現在及び将来においてスケーラビリティ上の懸念が発生しないという見通しがない限り，耐量子計算機性の実現のためのアプローチとしては，耐量子計算機性を持つ公開鍵暗号を利用することが望ましい。

以下では，より具体的に，耐量子計算機性を持たせるためのアプローチについて紹介する。公開鍵暗号によって暗号化（守秘・鍵共有）を行う情報システムに対して，耐量子計算機性を持たせるアプローチには，表 2.1 に示す手法及びその組み合わせが存在するが，一般に下段のアプローチになるほどスケーラビリティが低下する。ここで，最もスケーラビリティが期待できるデータ削除や匿名化といった手法は，そのデータが削除や匿名化が可能であるか否かを検討した後に実施する必要があり，運用上のスケーラビリティは高いものの，導入前の検討のために時間を必要とし，情報シ

[*1] スケーラビリティとは，要求される処理量等の変化に応じてそのシステムの対処能力を柔軟に増減させることができる能力である。
https://www.gartner.com/en/information-technology/glossary/scalability
本章では，情報システムの規模（ステークホルダ数，利用者数，処理量等）が増減した場合でも，その情報システムが消費するリソース（計算量，通信量，人の手間等）が極端に増加しない，又は，減少させることができる能力の意味で利用する。

表 2.1: 公開鍵暗号による暗号化（守秘・鍵共有）を行う情報システムに対して耐量子計算機性を持たせるためのアプローチ

	アプローチ	概要
1.	削除・匿名化	情報システムが，漏洩しても問題ない情報以外は保管しない/扱わないようにする。又は，保管する/扱う情報を加工することによって，漏洩しても問題ないように変形する。この方式は，スケーラビリティが最も高いが，可用性が大きく低下することが考えられ，選択できないことも多い。
2.	耐量子計算機性を持つ公開鍵暗号の採用	最も一般的な解決策であり，スケーラビリティを確保できる。現代暗号の利点を維持するアプローチである。
3.	公開鍵暗号を用いない鍵共有手段の導入	公開鍵暗号を利用している情報システムを，公開鍵暗号を利用しない仕組み（例えば，物理的に通信相手全員に IC カードを配布することで，共通鍵の事前共有を行うなど）に置き換えることで，耐量子計算機性を持たせる。暗号技術の観点からは，公開鍵暗号が登場する以前の思想で再設計することになる。スケーラビリティが低く，不特定多数が利用するシステムでは採用が困難と考えられる。また，通信当事者の捕捉が容易となることも考えられ，匿名性の確保やプライバシ保護に関する再設計も併せて必要になる可能性がある。
4.	物理アクセス制御	1〜3 のアプローチが採用できない場合にも採用可能である。暗号技術の観点からは，暗号技術が発展する以前の思想で再設計することになる。実装コスト及び運用コストが非常に高くなることが予想される。

ステムの可用性が低下するおそれもある。また，法令やポリシー等で削除・匿名化が許容されていない場合には，実施できないおそれもある。

これらの事情より，耐量子計算機性を持たせるための最も汎用的かつ根本的な対応は，既存の公開鍵暗号方式を耐量子計算機性を持つ公開鍵暗号方式に置き換えることであると考えられる。

ただし，情報システムで利用されている公開鍵暗号方式を，耐量子計算機性を持つ公開鍵暗号方式に置き換えることは容易ではない。それは単に実装を切り替えただけでは完了せず，公開鍵暗号がどのように利用されているのかについて認識した上で，運用やデータ管理に係る様々な処理も併せて実施することが要求される（以降，暗号方式の置き換えに加えて，これらの処理を行うことを「暗号移行」と呼ぶ)。そこで本章では，公開鍵暗号のいくつかの利用形態を念頭に，耐量子計算機性を持つ公開鍵暗号方式への暗号移行について紹介する。まず，現行の公開鍵暗号の利用形態を紹介した上で，各利用形態における CRQC による脅威及びその対策について，システム運用やデータ管理処理の観点を踏まえて概説する。また，脅威を評価する上で重要となる，保護対象となるデータの保護期間について記載した上で，利用形態や保護対象を踏まえた対応についても概説する。

2.1 公開鍵暗号の利用形態

既存の公開鍵暗号方式を，耐量子計算機性を持つ方式へと暗号移行するに際しては，その公開鍵暗号方式の利用形態ごとに，暗号移行のプロセスが大きく異なることが予想される。そこで，本節で公開鍵暗号の利用形態について概説した上で，次節以降で各利用形態における暗号移行のプロセスについて述べる。公開鍵暗号にはいくつかの利用形態が存在するが，本章では「電子政府における調達のために参照すべき暗号のリスト」[24] （以下「CRYPTREC 暗号リスト」と呼ぶ。）に合わせて，公開鍵暗号を署名・守秘・鍵共有に分類し，以降その分類に沿って概説する。また，本節で

は，署名用途／守秘用途／鍵共有用途の耐量子計算機性を持つ公開鍵暗号方式を，それぞれ署名用途／守秘用途／鍵共有用途の PQC と表記する。

2.1.1 署名用途での公開鍵暗号の利用

本節では，署名を付与する行為を「デジタル署名処理」と呼び，付与される署名データを「デジタル署名」と呼ぶ。デジタル署名が付与されたコンテンツを改竄すると，その改竄を検知することができる。このため，署名用途の公開鍵暗号を用い，コンテンツにデジタル署名を付与することで，コンテンツの改竄によりもたらされる被害を防止することができる。コンテンツは，人が読む文章（ドキュメントデータ），動画等の情報であることもあれば，暗号鍵の鍵情報[*2]であることもある。また，デジタル署名処理に用いられる秘密鍵が，対応する公開鍵を含む電子証明書によって所定の人物／組織／装置等と紐づいている場合では，コンテンツの生成人物／組織／装置を確認（認証）することもできる。このように署名用途の公開鍵暗号は，コンテンツの改竄防止，署名者の認証，データ元の認証等に利用される。

具体的な署名用途の公開鍵暗号の利用例としては，TLS 通信 [18] におけるクライアント認証（利用者の認証）やサーバ認証（サービス提供者の認証），OS のコードサインの確認（バイナリデータが改竄されていないことの確認）等に広く利用されている。また，公開鍵の配布手段の一種である公開鍵暗号基盤（PKI）の構成においても，公開鍵暗号は広く利用されており [4]，コンテンツに対して署名が付与された時刻を確認可能なタイムスタンプ署名方式 [23] 等も存在する。CRYPTREC 暗号リストには，DSA，ECDSA，EdDSA，RSA-PSS，及び RSASSA-PKCS1-v1_5 が署名用途の公開鍵暗号として記載されている。

2.1.2 守秘用途での公開鍵暗号の利用

守秘用途の公開鍵暗号によって暗号化された暗号文は，対応する秘密鍵なしに復号することは困難となる。このため，守秘用途の公開鍵暗号は，意図した相手だけにデータを提示するために利用することができる。暗号化処理による保護は，ドキュメントデータ，動画等の情報に対して行われることもあれば，暗号鍵の鍵情報[*3]に対して行われることもある。保護が鍵情報に対して行われるユースケースとしては，鍵情報を通信当事者間で共有する場合や，暗号鍵所有者がその鍵情報をバックアップする場合等が該当する。

守秘用途及び鍵共有用途の公開鍵暗号の一般的な実装形態として，公開鍵暗号方式により別の暗号鍵を保護し，その暗号鍵を利用した共通鍵暗号方式によりコンテンツの秘匿性や完全性を保護するというアプローチが存在する。このアプローチでは，共通鍵暗号方式の暗号鍵（以下，単に共通鍵と呼ぶ）は送信者により作成され，配送される。したがって，ある時点で共通鍵が漏洩した場合には，過去にその秘密鍵を持つ利用者に対して配送された共通鍵が漏洩するおそれがある。また，受信者は共通鍵の生成に関わることがないため，送信者が別の通信相手と共通鍵を使い回していても察知することができない。このため，昨今の TLS 通信等における共通鍵の共有においては，守秘用途の公開鍵暗号でなく，次節で概説する鍵共有用途での公開鍵暗号を一時的な鍵と組み合わせて利用することが望ましいと考えられている [5, 19]。なお，「TLS 暗号設定ガイドライン」[5] においても，鍵交換（鍵共有・守秘）においては，Perfect Forward Security（PFS）[*4]の特性を持つ DHE（又は ECDHE）を選択することがセキュリティ上望ましいと記載されている。CRYPTREC 暗号リストには，RSA-OAEP 及び RSAES-PKCS1-v1_5[*5]が守秘用途の公開鍵暗号として記載されている。また，RFC7525[19] においても，4.1 節において守秘用途で使用される RSA 暗号方式による鍵の転送（RSA key transport）は利用すべきでないと記載されており，4.2 節において一時的（Ephemeral）な鍵を用いる

[*2] 鍵情報には暗号鍵やメタデータが含まれ [5]，公開鍵暗号の鍵のみではなく共通鍵暗号の鍵に関する情報も含む概念となる。
[*3] 秘密鍵，共通鍵，鍵導出鍵及びそれらの鍵のメタデータを含む。
[*4] ある時点における鍵が漏洩した場合でも，漏洩した鍵とは異なる鍵を使用していた過去の暗号文の復号はできない性質。
[*5] 守秘用途の RSAES-PKCS1-v1_5 は，運用監視暗号リストに記載されており，互換性維持以外での利用は推奨されていない。

暗号スイート[*6]が推奨されている。

2.1.3 鍵共有用途での公開鍵暗号の利用

鍵共有用途での公開鍵暗号は，鍵共有に参加する二者が，同一の鍵情報[*7]を共有するために使用される。近年利用されている二者間鍵共有を目的とした多くの公開鍵暗号プロトコルにおいては，鍵共有に参加する双方が何らかの値を生成し，その値に対して秘密鍵を使用した計算を行う。結果として，共有される鍵には双方の生成した値が影響することとなり，一方のみの計算で暗号鍵を導出することはできない。このため，守秘用途でのデータ送付と異なり，送信者があらかじめ意図した特定の鍵を，共有鍵として利用することはできない。CRYPTREC 暗号リスト [24] には，DH，ECDH，及び PSEC-KEM が鍵共有用途の公開鍵暗号として記載されている。

2.2 PQC の導入における課題

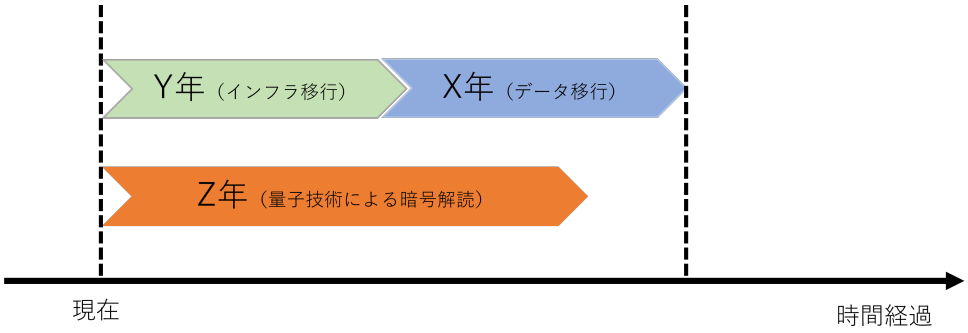


図 2.1: Mosca の発表 [15] より

現在広く利用されている公開鍵暗号が，量子コンピュータを利用した攻撃に起因して，“近い将来”に危殆化する可能性は低い [7] と考えられている。他方で，Mosca [15] が指摘するように，その情報システムで生成されるデータに対して暗号方式による保護が期待される期間（図 2.1 における X）に，暗号処理の実装の置き換えに要する期間（同図における Y）を加えたものが，CRQC による攻撃が実現するまでの期間（同図における Z）よりも長い場合（$X + Y > Z$ の場合）は，当該情報システムで生成されるデータは CRQC による攻撃の脅威にさらされることになる。すなわち，CRQC 実現までの期間（Z）が非常に長く，遠い将来であったとしても，その情報システムの X や Y の値が大きければ，何らかの対応が求められる。なお本章において，特記しない限り以降では，X，Y，Z は図 2.1 における X，Y，Z を示す。

もっとも，CRQC の実現時期は未だ不透明であり，Z を予想することは困難である。また，X は，暗号方式のみならず，保護対象となるデータの性質等によっても大きく異なる。特に，保護対象となるデータに対して，保護期間が設定されていない場合などは，X を導出すること自体が新たな課題となる。同様に，Y も，暗号方式の実装形態によって大きく変化する。さらに，X 及び Y は，情報システムの運用を通して，将来において変動することもありうる。

このように，ある公開鍵暗号アプリケーションが利用されている際に，CRQC による脅威について備える必要性があるか否かを判断しようとした場合，Z は不確定であり，X や Y も変動しうるため，判断が難しいという課題がある [25]。ここで，保護対象となるデータに保護期間が設定されていない場合においては，判断に先駆けて（X 導出のために）データの保護期間を決定することとなり，場合によってはその判断を行うための情報収集に相当の期間を必要と

[*6] 複数の暗号アルゴリズムの組合せ
[*7] 共通鍵暗号の共通鍵，鍵導出機能の鍵やパラメータ等

する。

PQC の導入においては，その情報システムに耐量子計算機性を持たせることが必要なのか，また，いつまでにそれを行う必要があるのか，を判断すること自体が課題となる。

2.2.1 署名用途での課題

署名用途の公開鍵暗号は，コンテンツの改竄防止，認証等に利用されるが，ユースケースによって脅威の性質は大きく異なる。例えば，TLS 通信 [18] におけるクライアント認証やサーバ認証においては，認証用に付与されたデジタル署名の検証を行うのは基本的にその場限りとなるため，X の値は小さくなる。また，Web ブラウザが信用するサーバ認証用の証明書の有効期間は，ごく一部の例外を除いて 1 年程度であり，それほど長い期間利用されることはない。そのため，X の値は，守秘用途や他の認証用途に比べて非常に小さくなる [25]。さらに，ブラウザのアップデートやルート認証局の入れ替えを，より迅速に実施できる体制を整備しており，Y の値も守秘用途や他の認証用途に比べて小さい。

他方で，電子データに対するドキュメント署名や，バイナリデータに対するコードサインであれば，署名対象のデータを利用する人が存在する限り（数十年に渡り）デジタル署名が検証されることもある。特に，コードサインにおいては，仮に電子証明書に有効期間が記載されていたとしても，その有効期間満了後にも検証されることが十分に考えられる。そのため，X の値は，守秘用途や他の認証用途に比べて非常に大きくなる。

このように，署名用途においては，X の値は大きくなりうるものであり，個々のアプリケーションごとに判断する必要がある。また，公開鍵の配布のために PKI を利用した場合，トラストアンカーの置き換え等に時間を要するため，Y が 10 年以上となることも珍しくない。

ここで，アプリケーションごとの判断の一例として，S/MIME プロトコルを利用するメールクライアントソフトウェアにおいて保護が必要な期間について概説する。S/MIME 用に発行された証明書（に対応する秘密鍵）は，通信相手の認証及び通信データの暗号化に利用可能であり，以下の用途での使用が可能である。

1. 通信時に通信相手を認証する
2. 通信時に通信データの暗号化（復号）に利用する
3. 過去に受け取ったメールの通信相手を後から認証する
4. 過去に受け取った通信データを復号する

3 及び 4 は，S/MIME プロトコルを通信プロトコルと捉えると，所管範囲外とも整理できるが，エンドユーザが利用するメールクライアントソフトウェアの中には，通信終了後の保管されたメールに対しても暗号処理を行うものも存在する。このような処理におけるセキュリティは，保存する／されているデータ（data at rest）のセキュリティとなり，データ保護の対象期間は非常に長くなる。

他方で，メールクライアントソフトウェアが 1 のみをサポートする場合[*8]では，S/MIME 証明書（に対応する秘密鍵）は受信者が送信者を認証した後は利用されることはない。このような，通信路上の転送されているデータ（data in transit）の認証におけるセキュリティでは，保護の対象期間は短くなる。このように，メールクライアントソフトウェア間でも，そのソフトウェアがサポートする機能の違いよって X の値は大きく変化する。

[*8] 通信時のメールコンテンツの暗号化は，（S/MIME 以外の）メールサーバ間の通信プロトコルにて実施することも可能である。そのため，送受信者間の送受信者間の E2E 暗号化（End-to-End Encryption）が必要でない場合や，送受信者間の E2E 暗号化が許容されない場合においては，S/MIME による暗号化が行われないこともある。送受信者間の E2E 暗号化が許容されない例としては，メールサーバを運用する組織が，自社ポリシーにて（メールクライアントではなく）メールサーバでのウィルス検知を必須とするケース等が挙げられる。

2.2.2 守秘用途での課題

守秘用途の公開鍵暗号においては，攻撃者が事前に暗号技術で保護されたデータを収集して保存しておき，後からそのデータに対して攻撃を行う攻撃である，Harvest Now Decrypt Later 攻撃（以下，「ハーベスト攻撃」と呼ぶ）[*9]の脅威が指摘されている。

ハーベスト攻撃においては，保護対象となるデータの保護期間，すなわち X の値が大きくなるほど，攻撃者が攻撃可能な期間が長くなる。これは，攻撃者が CRQC の開発を待たずに攻撃（保護された情報の収集）を開始できるためである。一方，防御側は，攻撃者に情報が収集される前に，情報システムに耐量子計算機性を付与することが求められる。保護対象となる情報の保護期間が長くなるほど，この不均衡は大きくなり，攻撃者の攻撃可能期間が長くなる。

守秘用途の公開鍵暗号では，保護対象となるコンテンツや鍵情報の保護期間が非常に長期となることが想定されている場合や，無期限で保護することが想定されている場合も存在する。例えば，患者を特定又は推測可能な形態で保管された遺伝性疾患に関する医療情報や，外交関係の機微な情報，さらには，それらの情報の暗号化に利用される鍵などは長い保護期間を持つ傾向にある。また，ドキュメントの生成時において，無期限に守秘することを前提としており，公開することを想定していない情報も存在する。

これらの情報においては，X の値は非常に大きくなるため，おそらく $X + Y > Z$ が成立することになる。そのため，速やかに CRQC の脅威に対する何らかの対応を行うことで，被害を軽減することが望ましい [25]。

2.2.3 鍵共有用途での課題

鍵共有用途での課題は，守秘用途における課題と同種の課題を含んでいる。例えば，鍵共有で共有された共通鍵が，非常に長い保管期間を持つデータの暗号化に利用されていた場合，X の値は非常に大きくなり，$X + Y > Z$ が成立すると考えられ，速やかに CRQC の脅威に対する何らかの対策が必要となる。

さらに，守秘用途では存在しない新たな懸念も存在する。例えば，一時的（Ephemeral）な鍵情報を用いた DH 鍵共有方式を採用することにより PFS を達成している情報システムが存在し，その情報システムは，PFS であることを前提とした運用ポリシーを策定していたとする。この情報システムの DH 鍵共有処理部分を，耐量子計算機性を持つ標準化された公開鍵暗号方式に置き換える場合，以下の 2 つの方針が考えられる。

1）鍵共有用途の PQC に置き換える
2）守秘用途の PQC に置き換える

標準化された鍵共有用途の PQC が存在するのであれば，1）が選択可能であり，比較的容易に実現可能だと考えられる。しかし，そのような鍵共有用途の PQC が存在せず，守秘用途の PQC しか標準化されていない場合には，2）を選択することとなり，守秘用途の PQC を用いて鍵共有部分を構成することとなる。

2）の選択において，守秘用途の PQC を単純に導入した場合，PFS の性質を持たなくなるおそれがあり，それによりデータ保護及び運用ポリシー策定時に想定していなかった経路からの情報漏洩等が発生する懸念が生じる。また，守秘用途の公開鍵暗号方式に対して何らかの手を加えて，PFS の性質を持つ暗号プロトコルを構成したとしても，その暗号プロトコルが標準化されていない場合は，運用ポリシー上，利用できないこともある。

他方で，2）の選択において，既存の鍵交換及び守秘用途の PQC の両方のハイブリッド構成を用いることによって対応するアプローチも存在する[*10]。ハイブリッド構成を用いることで，既存のアルゴリズムでしか防げない攻撃に対しても，新たなアルゴリズムでしか防げない攻撃に対しても，安全な構成とすることができる [22]。

[*9] Record Now Decrypt Later 攻撃，Store Now Decrypt Later 攻撃等とも呼ばれる。
[*10] TLS における [13, 22]，CMS における [16] 等が当該アプローチとして挙げられる。

例えば，既存の鍵共有方式（ephemeral ECDH 等）を用いた鍵交換で導出された秘密情報と，守秘用途の PQC（ML-KEM 等）を用いて導出された秘密情報の，両方を入力とし，所定のハッシュ計算の出力を暗号鍵とすることにより，PFS の性質を持つアルゴリズムでしか防げない攻撃に対しても，耐量子計算機性を持つアルゴリズムでしか防げない攻撃に対しても安全な構成とすることができる。

もっとも，鍵共有処理を複数回行うことに起因し，処理量及びデータ転送量が増加するため，その増加に対応できるように情報システムや通信プロトコルの修正が必要となりうることには注意が必要である。

2.3 PQC 導入へのアプローチ

2.2 節でも記載したように，CRQC の実現時期（又は実現までの期間 Z）は不透明ながら，X や Y の値が大きな情報システムにおいては，何らかの対応を取ることが望ましい。また，本章冒頭で記載したように，情報システムに耐量子計算機性を持たせる手段は，耐量子計算機性を持つ公開鍵暗号方式の導入だけではないものの，スケーラビリティを考慮すると耐量子計算機性を持つ公開鍵暗号方式の利用が有望である。本節では，耐量子計算機性を持つ公開鍵暗号方式への暗号移行を念頭に，その暗号移行を円滑に行う上での考慮事項について概説する。

2.3.1 プライオリティ設定の重要性

公開鍵暗号は様々な用途において普及している。それらの全ての公開鍵暗号方式を耐量子計算機性を持つ方式へ暗号移行するためには，長い期間及び労力を要する。また，情報システムの中には，そのシステムの利用期間及び生成されるデータの保護期間が短い等の理由により，耐量子計算機性を持たせる必要がないものも存在するかもしれない。

そこで，暗号移行を検討する上では，X,Y,Z を意識して対応することが重要と考えられる。もっとも，X や Y は暗号方式の利用局面ごとに異なることも想定され，またそれらの値は将来において変動する可能性がある。さらに，Z は不確定であり，予想すること自体も困難である。このような状況の下で，全ての暗号モジュールに対して X,Y,Z を分析するアプローチを取ることは，作業量の観点で大きな困難が伴うことが予想され，結果として本当に保護が必要なデータに対する対応に手が回らないおそれがある。そこで，暗号移行を行う担当者は，優先度の高いものを洗い出し，その優先度に応じて対応を行うことが適切である [9, 26, 25]。

PQC への暗号移行を検討するにあたり，あらかじめ優先順位付けを行うことの重要性は，金融庁の報告書 [27] でも触れられており，基本事項は以下のように整理されている。

- 暗号解読可能な量子コンピュータによる既存の暗号危殆化に関連するリスクに基づいて，移行対象の優先順位付けを行う。
- 移行対象の詳細な把握のため，クリプト・インベントリを構築する。
- 暗号危殆化状況に応じて安全かつ迅速に対応できるアーキテクチャを検討する。
- 優先順位の高いものを中心に移行期限を設定し，期限超過の可能性も踏まえたリスク低減策も検討する。

ここで，クリプト・インベントリとは利用している暗号モジュールや暗号方式のリストのことであり，その作成においては，既に管理簿や仕様書等が存在する場合はそれを利用することができる。また，管理簿や仕様書等が存在しない場合は，何らかの自動化ツールを使うことが，効率の観点からもミスを減らす面からも望ましい。そのような自動化ツールの利用を検討する上では，NIST NCCoE の検討 [9] が参考になる。

CRQC による攻撃リスクの評価においては，CRQC による攻撃が成功した場合の影響，暗号方式によって保護される情報の保護期間（X の把握のために必要），情報システムで利用する各暗号モジュールの移行に要する時間（Y），CRQC を利用する攻撃を行うための前提条件の難易度（攻撃対象である暗号化データ取得の難易度や，そのデータを利用した攻撃の難易度）等の把握が有用である。

この優先順位付けに先駆けて，過剰な保護期間が設定されている情報の保管期間短縮，不要な情報の消去，公開可能な情報の公開等を併せて実施する事も望ましい。このような処理により，X の短縮が期待でき，暗号移行の対象となるシステムを削減する効果が期待される。

暗号移行に際しては，速やかに PQC に暗号移行するというアプローチと，あらかじめクリプトグラフィック・アジリティ [12][*11]を向上させつつ，ある程度以上のクリプトグラフィック・アジリティを達成した上で暗号移行するというアプローチが存在する。

クリプトグラフィック・アジリティが向上すると，Y や X の値が小さくなる。このため，例えば，PQC の評価が十分にされておらず，暗号移行開始の妨げとなっている期間においては，当面の間はクリプトグラフィック・アジリティ向上に努めるというアプローチも一定の合理性があるものと考えられる [26]。なお，クリプトグラフィック・アジリティ向上に伴う Y や X の短縮により，該当する情報システムのプライオリティが下がれば，他の相対的にプライオリティが高くなった情報システムへリソースを集中させることも可能となる。

本節では，インベントリ管理，不要な情報の消去，クリプトグラフィック・アジリティの確保等について概説したが，こららのアプローチは，一般的な情報セキュリティの文脈でも有効である。そのため，これらのアプローチの検討においては，PQC への暗号移行の文脈における効果のみを念頭に検討するのではなく，他のセキュリティ上の恩恵も併せて視野に入れて検討を行うべきであろう [25]。

2.3.2 クリプトグラフィック・アジリティの重要性

クリプトグラフィック・アジリティは，文脈によって捕捉範囲が異なり，それに伴って異なる意味合いを持つことがある [2]。しかしながら，それらに通底している性質として，暗号アルゴリズムや暗号プロトコルをより迅速に変更できる点が挙げられる。

暗号移行においては，暗号移行の対象となる情報システムの暗号部分が，情報システムにハードコードされている場合には，暗号アルゴリズムの変更が困難である。このような状態は「クリプトグラフィック・アジリティを持たない」と表現することができる。

他方で，標準プロトコルを採用する情報システム，暗号モジュールにも標準プロトコルを利用している情報システム，その API が適切に定義されている情報システム，相互運用性が確保されている情報システム，及び暗号回路を含むファームウェアアップデートをオンラインで実施できるように設計されている情報システム等では，その暗号移行に要する時間は比較的短くなり，$X + Y > Z$ となる可能性も低くなる。X 及び Y の値が十分に低く，所定の目標期間以内に暗号移行が可能なシステムは，「クリプトグラフィック・アジリティを持つ」と表現することができる [2, 25]。

クリプトグラフィック・アジリティを持たせるための対応は，PQC の実装とは独立して実施することが可能である [3]。また，より短い期間での暗号移行を行うことが可能となれば，移行プロセスを開始するまでの猶予期間（$Z-X-Y$）をより長くすることが期待される。より長い猶予期間での暗号移行が可能となれば，該当猶予期間を CRQC の開発動向調査等に充てることが可能となり，より適切なタイミングでの移行が期待できる。

なお，クリプトグラフィック・アジリティが十分向上した情報システムにおいては，暗号機能以外を変更する場合の対応速度も高くなることが期待できる。すなわち，セキュリティ上の脆弱性が発覚した場合の対応速度や，ビジネス環境に合わせたサービス変更の対応速度も高いことも期待される [25]。

以上を踏まえ，PQC への暗号移行を実施するにあたっては，まずは暗号移行を長期化する要素を排除することを試み，情報システムにおける暗号プロトコルの変更をより迅速にできるようにシフトさせていく対策，すなわちクリプトグラフィック・アジリティを確保する対策を併せて実施することが効果的である [2, 3, 25]。

[*11] 暗号方式を変更可能とする性質。2.3.2 節参照。

2.3.3 既存暗号方式とのハイブリッド構成

暗号移行においては，ハイブリッド構成を採用することができる。PQC への暗号移行の文脈におけるハイブリッド構成とは，既存の公開鍵暗号と，PQC の両方を利用することによって何らかの目標の達成を目指すものであるが，厳密な定義は見当たらない [8]。ハイブリッド構成の目標は，暗号アルゴリズムの切替期間中における相互運用性の確保や，既存暗号方式しか利用できない機器に対する後方互換性の確保であることもあれば，両方のアルゴリズムのうち片方が危殆化した場合の安全性の維持であることもある。

また，ハイブリッドという用語は，単一の暗号モジュールを構成するコンポジット方式 [13, 16] の文脈で使用されることもあれば，複数の暗号モジュールの出力を入力として受け取り，新たな出力を生成するコンバイナー構造に対して使用されることもある [8]。

なお，IETF の標準化活動において，ハイブリッド構成による鍵共有方式に関しては一定の合意が見受けられるが [13, 16]，ハイブリッド構成によるデジタル署名方式 [17] に関しては合意に時間を要している。

2.3.4 署名用途固有の対策

署名用途の公開鍵暗号は様々なユースケースで利用されるが，PKI 等のインフラの移行に要する時間（Y）やコードサイン証明書が利用される期間（X）が比較的長いことから，速やかな PQC への暗号移行が困難である。この場合においても，以下の対応を取ることが望ましい。

PKI においては，一般に Y が長くなる傾向にあるが，電子証明書の有効期間の短縮や，1 枚の電子証明書に対して（既存暗号方式と署名用途の PQC の）2 つの公開鍵及びデジタル署名を付与する方式などを採用することで，Y の短縮が期待できる [21]。なお，後者の 2 つの公開鍵暗号及びデジタル署名を利用する方式においては，実装やポリシー管理の複雑さが大きく増加することから，注意を必要とする。

X を実質的に短縮する技術として，タイムスタンプ更新技術が存在する。例えば，ERS[10] 等を利用することで，タイムスタンプの更新や，暗号方式の更新が可能となる。Z が経過する前に，既存の公開鍵暗号を PQC に更新することが可能であれば，X,Y,Z の関係によらず，データは保護される。ただし，このアプローチでは，データ構造の複雑さが増加する傾向があり，（PQC への即時の暗号移行に比べては小さいものの）情報システムの運用費用が増加する点には注意を必要とする。

2.3.5 守秘及び鍵共有用途固有の対策

既に述べたように，耐量子計算機性を持たせるための一般的な対策は，既存の暗号方式を耐量子計算機性を持つ公開鍵暗号方式に移行することである。ここで，2.2.2 節及び 2.2.3 節で述べたとおり，守秘及び鍵共有用途で保護されたコンテンツや鍵情報は，保護期間が非常に長いことや，場合によっては無期限で保護されることも考えられる。このような情報に対するハーベスト攻撃の脅威を考慮すると，当該情報は，将来における CRQC による解読リスクに既に晒されていることから，一刻も早く耐量子計算機性を持たせる対応を始めることが望ましい。ただし，全ての守秘及び鍵共有用途の公開鍵暗号を移行するためには非常に大きなリソースが要求され，現実的なコストでは実現が困難であるおそれがある。

このような状況においても，守秘及び鍵共有用途固有の対策を効率的に行う方法 [26] として，以下のアプローチがある。

Z に対して X + Y の値が非常に小さく，$X + Y \ll Z$ と予測される暗号文に対しては，CRYPTREC による注意喚起情報 [6] に注意を払いつつ，現在用いている暗号の使用を継続する。また，$X + Y > Z$ となることが十分予想される暗号文に対しては，2.3 節前段で述べた，PQC への暗号移行や，暗号文の保護期間である X の短縮，情報システムの

暗号処理の実装の置き換えに要する期間 Y の短縮を行う。その結果，X や Y の値を十分に小さくすることができるのであれば，現在用いている暗号方式の使用を継続する。

一方で，X + Y > Z と予想される，又は，X + Y > Z となることが避けられない暗号文に対してしては，暗号システムの PQC への暗号移行を進めつつも，既存の公開鍵暗号によって保護されている暗号文は公開ネットワーク等に保管せず，適切にアクセスコントロールを行う。

なお，現在 DH を利用している場合は，2.2.3 節で述べたような検討を行い，DH 固有の性質が必要か否かをあらかじめ検討することが望ましい。

2.4 PQC の活用にむけて

PQC への暗号移行においては，どのようなデータに対して，どのような暗号技術を利用しているのかを把握することが第一歩となる。また，保護対象となるデータの保護期間等をあらかじめ把握しておくことで，より効率的な対応ができる [26]。その上で，公開できるデータは公開し，破棄可能なデータは破棄することも検討すべきである。この検討を進めることで，クリプトグラフィック・アジリティ [12] の確保も見込まれ，より効果的な PQC への移行が期待できる。CRQC の脅威への対策を検討するにあたっては，保護されている情報の価値，CRQC による攻撃が成功した場合の影響，図 2.1 における X,Y,Z の関係等を踏まえ，プライオリティを付けて，そのプライオリティ順に対策を実施することが望ましい [27, 25]。

第 2 章の参照文献

[1] National Security Agency. The Commercial National Security Algorithm Suite 2.0 and Quantum Computing FAQ. 2024-04. https://media.defense.gov/2022/Sep/07/2003071836/-1/-1/1/CSI_CNSA_2.0_FAQ_.PDF. (2025-01-06 閲覧).

[2] N. Alnahawi, N. Schmitt, A. Wiesmaier, A. Heinemann, T. Grasmeyer. On the State of Crypto-Agility. Cryptology ePrint Archive, Paper 2023/487. 2023. https://eprint.iacr.org/2023/487.

[3] A. Amadori et al. The PQC Migration Handbook. https://publications.tno.nl/publication/34643386/fXcPVHsX/TNO-2024-pqc-en.pdf. 2024-12. (2025-01-06 閲覧).

[4] S. Boeyen, S. Santesson, T. Polk, R. Housley, S. Farrell, D. Cooper. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, https://www.rfc-editor.org/info/rfc5280. 2008-05. (2023-04-12 閲覧).

[5] CRYPTREC. TLS 暗号設定ガイドライン. CRYPTREC GL-3001-3.0.1, https://www.cryptrec.go.jp/report/cryptrec-gl-3001-3.0.1.pdf. 2020-07.

[6] CRYPTREC. 注意喚起一覧. https://www.cryptrec.go.jp/er.html. (2024-03-05 閲覧).

[7] CRYPTREC 暗号技術評価委員会. 注意喚起情報 “現在の量子コンピュータによる暗号技術の安全性への影響”. https://www.cryptrec.go.jp/topics/cryptrec-er-0001-2019.html.

[8] F. Driscoll, M. Parsons, B. Hale. Terminology for Post-Quantum Traditional Hybrid Schemes. Internet-Draft. 2024-12. https://datatracker.ietf.org/doc/draft-ietf-pquip-pqt-hybrid-terminology/05/. (2025-02-20 閲覧).

[9] NIST National Cyersecurity Center of Excellence. Migration to Post-Quantum Cryptography Quantum Readiness: Cryptographic Discovery. NIST SP 1800-38B (initial preliminary draft), https://www.nccoe.nist.gov/sites/default/files/2023-12/pqc-migration-nist-sp-1800-38b-preliminary-draft.pdf. 2023-12. (2025-02-17 閲覧).

[10] T. Gondrom, R. Brandner, U. Pordesch. Evidence Record Syntax (ERS). RFC 4998, https://www.rfc-editor.org/info/rfc4998. 2007-08. (2023-04-12 閲覧).

[11] P. E. Hoffman. DNS Security Extensions (DNSSEC). RFC 9364, https://www.rfc-editor.org/info/rfc9364. 2023-02.

[12] R. Housley. Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms. RFC 7696, https://www.rfc-editor.org/info/rfc7696. 2015-11. (2023-04-12 閲覧).

[13] K. Kwiatkowski, P. Kampanakis, B. Westerbaan, D. Stebila. Post-quantum hybrid ECDHE-MLKEM Key Agreement for TLSv1.3. Internet-Draft. 2024-12. https://datatracker.ietf.org/doc/draft-kwiatkowski-tls-ecdhe-mlkem/03/. (2025-02-20 閲覧).

[14] M. Lepinski, S. Kent. An Infrastructure to Support Secure Internet Routing. RFC 6480, https://www.rfc-editor.org/info/rfc6480. 2012-02. (2025-01-15 閲覧).

[15] M. Mosca. Cybersecurity in a quantum world: will we be ready? Workshop on Cybersecurity in a Post-Quantum World. Session 8. 2015-04. (2024-02-29 閲覧).

[16] M. Ounsworth, J. Gray. Composite KEM For Use In Internet PKI. Internet-Draft. 2024-10. https://datatracker.ietf.org/doc/draft-ietf-lamps-pq-composite-kem/. (2025-01-06 閲覧).

[17] M. Ounsworth, J. Gray, M. Pala, J. Klaußner, S. Fluhrer. Composite ML-DSA For use in X.509 Public Key Infrastructure and CMS. Internet-Draft. 2024-10. https://datatracker.ietf.org/doc/draft-ietf-lamps-pq-composite-sigs/03/. (2025-01-15 閲覧).

[18] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, https://www.rfc-editor.org/info/rfc8446. 2018-08. (2023-04-12 閲覧).

[19] Y. Sheffer, R. Holz, P. Saint-Andre. Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS). RFC 7525, https://www.rfc-editor.org/info/rfc7525. 2015-05. (2023-04-12 閲覧).

[20] P. W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput. Vol. 26, Num. 5 (1997), pp. 1484–1509.

[21] D. Stebila, S. Fluhrer, S. Gueron. Hybrid key exchange in TLS 1.3. Internet-Draft. 2024-10. https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/11/. (2025-02-20 閲覧).

[22] D. Stebila, S. Fluhrer, S. Gueron. Hybrid key exchange in TLS 1.3. Internet-Draft. 2025-01. https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/12/. (2025-02-20 閲覧).

[23] R. Zuccherato, P. Cain, Dr. C. Adams, D. Pinkas. Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP). RFC 3161, https://www.rfc-editor.org/info/rfc3161. 2001-08. (2023-04-12 閲覧).

[24] デジタル庁, 総務省, 経済産業省. 電子政府における調達のために参照すべき暗号のリスト（CRYPTREC 暗号リスト）. CRYPTREC LS-0001-2022r1, https://www.cryptrec.go.jp/list/cryptrec-ls-0001-2022r1.pdf. 2024-05.

[25] 伊藤 忠彦. 耐量子計算機暗号への移行へ向けた課題と社会実装への論点整理. 電子情報通信学会誌. Vol. 106, Num. 11 (2023), pp. 1026–1030.

[26] 伊藤 忠彦, 宇根 正志, 清藤 武暢. 量子コンピュータによる脅威を見据えた暗号の移行対応. 2019-08. https://www.imes.boj.or.jp/research/papers/japanese/19-J-15.pdf. (2025-01-06 閲覧).

[27] 預金取扱金融機関の耐量子計算機暗号への対応に関する検討会. 預金取扱金融機関の耐量子計算機暗号への対応に関する検討会報告書. 2024-11. https://www.fsa.go.jp/singi/pqc/houkokusyo.pdf. (2025-01-06 閲覧).

第 3 章

格子に基づく暗号技術

本章では格子に基づく暗号技術についてまとめる。格子に基づく暗号技術の安全性は，LWE (Learning with Errors) 問題，LWR (Learning with Rounding) 問題，NTRU 問題，およびそれらの変種等を含む格子理論に関係する問題を解く計算の困難性に依存している。

3.1 格子に基づく暗号技術の安全性の根拠となる問題

3.1.1 LWE 問題と代表的な求解法

本節では，2005 年 Regev が提案した LWE 問題 [133] を紹介すると共に，格子を利用した LWE 問題に対する求解法を紹介する。また，LWE 問題のいくつかの変種についても言及する。

3.1.1.1 LWE 問題の紹介

LWE 問題は機械学習理論から派生した求解困難な問題で，整数剰余類環 $\mathbb{Z}_q$ 上の秘密ベクトル $\mathbf{s} \in \mathbb{Z}_q^n$ に関するランダムな連立線形「近似」方程式が与えられたとき，その秘密ベクトルを復元する問題である。具体的な数値例として $n = 4,\ q = 17$ に対して，秘密ベクトル $\mathbf{s} = (s_1, s_2, s_3, s_4) \in \mathbb{Z}_{17}^4$ に関する連立線形近似方程式

$$\begin{cases} 14s_1 + 15s_2 + 5s_3 + 2s_4 \approx 8 \pmod{17} \\ 13s_1 + 14s_2 + 14s_3 + 6s_4 \approx 16 \pmod{17} \\ 6s_1 + 10s_2 + 13s_3 + s_4 \approx 12 \pmod{17} \\ \qquad\qquad \vdots \\ 6s_1 + 7s_2 + 16s_3 + 2s_4 \approx 3 \pmod{17} \end{cases}$$

が与えられたとする。(この数値例は [135] から引用した。) ただし，各線形方程式の値は近似値であり，その誤差はこの例では ± 1 以内と仮定する。このとき，この連立線形近似方程式の解 $\mathbf{s}$ を求めるのが LWE 問題である。ここに示した数値例では $\mathbf{s} = (0, 13, 9, 11) \in \mathbb{Z}_{17}^4$ が解となる。LWE 問題で注意すべきことは，連立線形近似方程式に誤差がない場合は，Gauss の消去法により効率的に解を求めることができる点である。逆に言うと，連立線形近似方程式で与えられる誤差の大きさが LWE 問題の求解を困難にする。

■ 離散 Gauss 分布　一般に, LWE 問題における連立線形近似方程式の各行の誤差は, 平均 0, パラメータ $\sigma > 0$ の $\mathbb{Z}$ 上の離散 Gauss 分布 $\chi = D_{\mathbb{Z},\sigma}$ から生成される[*1]。より正確には，χ は各整数 x がサンプルされる確率が $\exp\left(-\dfrac{\pi x^2}{\sigma^2}\right)$ に比例する $\mathbb{Z}$ 上の離散確率分布である。この分布は，数学的な正規分布[*2] とは異なるが，絶対値の大きな値が生成さ

[*1] 本章では，記号 σ を Gauss 分布のパラメータ (標準偏差とは異なる) の意味で使い，署名を表すときには sig を用いる。

[*2] 分散 t^2 に対して数学的な正規分布 $N(0, t^2)$ は，確率密度関数が $\dfrac{1}{\sqrt{2\pi t}} e^{-z^2/(2t^2)}$ により定義されるため，$\sqrt{2\pi}$ 倍のずれがある。暗号の安全性を議論する際に格子上のフーリエ変換が用いられることが多く [133]，本文中の定義を用いることで，数式の表現が簡潔となる。

れる確率が非常に小さいという性質は共通している。例えば，絶対値が 3σ より大きな整数がサンプルされる確率は非常に小さい。離散 Gauss 分布の詳細については [101] などを参照。

離散 Gauss 分布を厳密に出力するルーチンを実装するのは容易ではなく，また生成時間を一定にすることは困難であるため，timing attack などの脆弱性 [40] が生まれる可能性がある。現実の方式 (3.3 節参照) においては，誤差 (ノイズ) として離散 Gauss 分布との統計距離が小さい分布を用いている。

また，記号 $D_{\mathbb{Z},s}^{n\times m}$ を，各要素を $D_{\mathbb{Z},s}$ から独立に生成した $n \times m$ 行列とする。

■ **LWE 問題の定式化**　以下は，定式化された LWE 問題である：

定義 3.1 (LWE 問題 [133]) n を正の整数とし，q を奇素数とする。平均 0，パラメータ σ の $\mathbb{Z}$ 上の離散 Gauss 分布を $\chi = D_{\mathbb{Z},\sigma}$ とする。秘密ベクトル $\mathbf{s} \in \mathbb{Z}_q^n$ を固定する。一様ランダムに選ばれた $\mathbf{a} \in \mathbb{Z}_q^n$ と離散 Gauss 分布 χ からサンプルされた $e \in \mathbb{Z}$ に対して，$(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ の組を出力する確率分布を $L_{\mathbf{s},\chi}$ とする。ただし，$b \equiv \langle \mathbf{a}, \mathbf{s} \rangle + e \pmod q$ とする。(2 つのベクトル $\mathbf{v}$ と $\mathbf{w}$ の内積を $\langle \mathbf{v}, \mathbf{w} \rangle$ で表す。) このとき，次の 2 つの問題を考える：

1. **判定** LWE (Decision-LWE)　与えられた組 $(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ が，確率分布 $L_{\mathbf{s},\chi}$ からサンプルされた元か，$\mathbb{Z}_q^n \times \mathbb{Z}_q$ 上一様ランダムに生成された元かを決定する問題。
2. **探索** LWE (Search-LWE)　確率分布 $L_{\mathbf{s},\chi}$ からサンプルされた組 $(\mathbf{a}, b)$ から秘密ベクトル $\mathbf{s}$ を復元する問題。

一般に，ここに示した 2 つの LWE 問題において確率分布 $L_{\mathbf{s},\chi}$ は任意個の組 $(\mathbf{a}, b)$ をサンプルするオラクルとしてみなす。具体的には，ある固定したサンプル数 $m > 0$ に対して，確率分布 $L_{\mathbf{s},\chi}$ からサンプルされた異なる m 個の組

$$\begin{cases} (\mathbf{a}_1, b_1), & b_1 \equiv \langle \mathbf{a}_1, \mathbf{s} \rangle + e_1 \pmod q \\ (\mathbf{a}_2, b_2), & b_2 \equiv \langle \mathbf{a}_2, \mathbf{s} \rangle + e_2 \pmod q \\ \quad \vdots & \\ (\mathbf{a}_m, b_m), & b_m \equiv \langle \mathbf{a}_m, \mathbf{s} \rangle + e_m \pmod q \end{cases}$$

から LWE 問題を解くことを考える。(解読に要する計算時間が最も短くなるような m を攻撃者が選べることを想定する。) 第 i 行ベクトルを $\mathbf{a}_i$ とする $m \times n$ 行列を $\mathbf{A}$ とし，$\mathbf{b} = (b_1, b_2, \ldots, b_m)$ とおく。このとき，ここに示した m 個の LWE サンプルの組は $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^n$ と簡潔に表せて，関係式 $\mathbf{b} \equiv \mathbf{s}\mathbf{A}^\top + \mathbf{e} \pmod q$ を満たす。ただし，$\mathbf{e} = (e_1, e_2, \ldots, e_m) \in \mathbb{Z}^m$ をノイズベクトルとする。(各 e_i は χ からサンプルされた元であることに注意する。)

■ **LWE 問題の変種**　LWE 問題の変種として，多項式環 $R_q = \mathbb{Z}_q[x]/(\phi)$ 上の LWE である Ring-LWE [149, 107][*3] や Module-LWE [97] がある。Ring-LWE では，3 つの多項式 $s, a_i, e_i \in R_q$ に対する Ring-LWE サンプルとして $\{(a_i, a_i \cdot s + e_i)\}_{i=1}^m$ を考える。(特に，通常の LWE 問題と同じように，ランダムな s と，係数が小さい多項式の集合からサンプリングされた e_i が用いられる。) Ring-LWE の基礎環 R_q を定める多項式として，2 のべき乗の形をした整数 n に対し $\phi = x^n + 1$ がよく用いられる。また，Module-LWE では，多項式ベクトル $\mathbf{s}, \mathbf{a}_i \in R_q^k$ と多項式 $e_i \in R_q$ に対する Module-LWE サンプルとして $\{(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i)\}_{i=1}^m$ を考える。Module-LWE の基礎環 R_q を定める多項式としては $\phi = x^{n/k} + 1$ がよく用いられる。さらに，環上の LWE 以外の LWE 問題の変種として，丸め込み（rounding）でノイズベクトルを生成する LWR[24] や middle-product と呼ばれる多項式演算を用いる Middle-product LWE [136] など数多くの変種が提案されている。

[*3] 文献 [107] ではより一般的に整数環とイデアルを用いて定義されているが，後の文献 [39] ではそれの簡略化として，多項式環 R_q を用いた表現である “polynomial-LWE assumption” が提案された。現在では，後者の表現の方が Ring-LWE と呼ばれている。

3.1.1.2 格子の基本事項と q-ary 格子の紹介

■ **格子の基本事項** m 次元実ベクトル空間 $\mathbb{R}^m$ の一次独立な m 個のベクトル $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m$ の整数係数の線形結合全体 $L = \{\sum_{i=1}^m x_i \mathbf{b}_i : x_i \in \mathbb{Z}, 1 \leq i \leq m\}$ を (完全階数の) m 次元**格子**と呼ぶ。特に，格子 L はベクトル空間 $\mathbb{R}^m$ の (離散) 加法部分群である。また，格子 L を生成する一次独立な m 個のベクトルの組 $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$ を**基底**と呼び，各 $\mathbf{b}_i$ を**基底ベクトル**と呼ぶ。さらに，行ベクトルで表した基底ベクトル $\mathbf{b}_i \in \mathbb{R}^m$ を行として持つ $m \times m$ 行列 $\mathbf{B} = (\mathbf{b}_i)_{i=1}^m$ を格子 L の**基底行列**と呼ぶ。2 次元以上の格子を生成する異なる基底は無限に存在し，同じ格子を生成する 2 つの基底行列 $\mathbf{B}_1$ と $\mathbf{B}_2$ に対し $\mathbf{B}_2 = \mathbf{V}\mathbf{B}_1$ を満たす $m \times m$ のユニモジュラ行列 $\mathbf{V}$ が存在する。また，基底行列 $\mathbf{B}$ を用いて，格子 L の**体積**を $\mathrm{vol}(L) = |\det(\mathbf{B})|$ と定める。(体積は基底の取り方に依存しない。) 格子 L の**第 1 逐次最小**は L 上の最短な非零ベクトルの Euclid ノルムを指し，$\lambda_1(L)$ と表す。この長さを持つベクトル $\mathbf{v} \in L$ を発見する計算問題を最短ベクトル問題 (Shortest Vector Problem: SVP) と書く。定数 $\gamma > 1$ に対して，$\gamma\lambda_1(L)$ よりも短いベクトルを発見する計算問題を近似最短ベクトル問題と呼び，近似因子 γ を明示して γ-SVP と書く。また，点 $\mathbf{w} \in \mathbb{R}^m$ と格子基底が与えられたときに，$\mathbf{w}$ との距離が最小となる格子点を発見する計算問題を最近ベクトル問題（Closest Vector Problem: CVP）と書く。ベクトル空間 $\mathbb{R}^m$ の完全階数の格子 L に対し，集合 $\widehat{L} = \{\mathbf{x} \in \mathbb{R}^m : \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z} \quad (\forall \mathbf{y} \in L)\}$ を格子 L の**双対格子**と呼ぶ。また，格子 L の基底行列 $\mathbf{B}$ に対して，$\widehat{\mathbf{B}} = \left(\mathbf{B}^{-1}\right)^\top$ は双対格子 $\widehat{L}$ の基底行列となり，この $\widehat{\mathbf{B}}$ を**双対基底行列**と呼ぶ。単位行列 $\mathbf{I}_m$ に対し $\mathbf{B}\widehat{\mathbf{B}}^\top = \mathbf{I}_m$ を満たすので，$\mathrm{vol}(L) \times \mathrm{vol}(\widehat{L}) = 1$ が成り立つ。

■ q-ary **格子** ここでは，LWE 問題の求解で利用する特殊な格子を紹介する。正の整数 q に対して，$q\mathbb{Z}^m \subseteq L \subseteq \mathbb{Z}^m$ を満たす完全階数の m 次元格子 L を **q-ary 格子**と呼ぶ。2 つの自然数 $m > n$ に対し，任意の正の整数 q と $n \times m$ 整数行列 $\mathbf{X}$ に対する 2 つの m 次元 q-ary 格子を

$$\Lambda_q(\mathbf{X}) = \{\mathbf{y} \in \mathbb{Z}^m : \exists \mathbf{s} \in \mathbb{Z}^n \text{ s.t. } \mathbf{y} \equiv \mathbf{s}\mathbf{X} \pmod q\}, \quad \Lambda_q^\perp(\mathbf{X}) = \left\{\mathbf{y} \in \mathbb{Z}^m : \mathbf{y}\mathbf{X}^\top \equiv \mathbf{0} \pmod q\right\}$$

と定義する。(これらの集合は $\mathbb{R}^m$ の離散加法部分群なので格子である。) 正規化の差を除き，これら 2 つの q-ary 格子は互いに双対の関係にある。正確には $\Lambda_q^\perp(\mathbf{X}) = q\widehat{\Lambda_q(\mathbf{X})}$ と $\Lambda_q(\mathbf{X}) = q\widehat{\Lambda_q^\perp(\mathbf{X})}$ が成り立つ。また，群準同型写像 $f : \mathbb{Z}^m \longrightarrow (\mathbb{Z}_q)^n, \quad \mathbf{y} \longmapsto \mathbf{y}\mathbf{X}^\top \pmod q$ の核は q-ary 格子 $\Lambda_q^\perp(\mathbf{X})$ なので，群の準同型定理から $\mathrm{vol}(\Lambda_q^\perp(\mathbf{X})) = [\mathbb{Z}^m : \Lambda_q^\perp(\mathbf{X})] = \#\mathrm{Im}(f)$ が成り立つ。(群の指数 $[\mathbb{Z}^m : \Lambda_q^\perp(\mathbf{X})]$ は格子の体積の比 $\dfrac{\mathrm{vol}(\Lambda_q^\perp(\mathbf{X}))}{\mathrm{vol}(\mathbb{Z}^m)}$ に一致することに注意する。) これより，体積 $\mathrm{vol}\left(\Lambda_q^\perp(\mathbf{X})\right)$ は q^n を割る。さらに，元の格子と双対格子の体積の関係から，q^{m-n} は体積 $\mathrm{vol}\left(\Lambda_q(\mathbf{X})\right)$ を割ることが分かる。(ただし，ほとんどの行列 $\mathbf{X}$ に対して写像 f は全射で，その時 $\mathrm{vol}\left(\Lambda_q^\perp(\mathbf{X})\right) = q^n$ と $\mathrm{vol}\left(\Lambda_q(\mathbf{X})\right) = q^{m-n}$ が成り立つ。) q-ary 格子 $\Lambda_q(\mathbf{X})$ 上のベクトルは $\mathbf{y} = \mathbf{s}\mathbf{X} + q\mathbf{z}$ $(\mathbf{s} \in \mathbb{Z}^n, \mathbf{z} \in \mathbb{Z}^m)$ とかけるので，その格子は $(n+m) \times m$ 整数行列 $\begin{pmatrix} \mathbf{X} \\ q\mathbf{I}_m \end{pmatrix}$ の一次従属な $(n+m)$ 個の行ベクトルで生成される。この生成行列の Hermite Normal Form を計算することで，m 次元 q-ary 格子 $\Lambda_q(\mathbf{X})$ の基底行列 $\mathbf{B} \in \mathbb{Z}^{m \times m}$ が得られる。また，双対基底の性質から，もう片方の q-ary 格子 $\Lambda_q^\perp(\mathbf{X})$ の基底行列は $(q\mathbf{B}^{-1})^\top \in \mathbb{Z}^{m \times m}$ で得られる。

3.1.1.3 LWE 問題の代表的な求解法

LWE 問題の代表的な求解法として，最短ベクトル問題や最近ベクトル問題に帰着する方法がある。これらの手法は LWE 問題の困難性に基づいた暗号方式の解読困難性を評価するデファクトスタンダードの手法となっている。LWE 問題のインスタンスから変換される q-ary 格子の種類から，Dual attack，Primal attack という分類がなされ，それぞれ判定版 LWE 問題，探索版 LWE 問題の評価手法として確立されている。

■ **判定 LWE 問題に対する求解（Dual Attack）** 判定 LWE 問題を SIS (Short Integer Solution) 問題に帰着して解く方法を紹介する：正の整数 q と，$0 < \beta < q$ を満たす実数 β を固定する。各成分が剰余環 $\mathbb{Z}/q\mathbb{Z}$ 上一様ランダム

に選ばれた $n \times m$ 整数行列 $\mathbf{X}$ に対して，$\|\mathbf{v}\| \leq \beta$ かつ $\mathbf{v}\mathbf{X}^\top \equiv \mathbf{0} \pmod q$ を満たす非零ベクトル $\mathbf{v} \in \mathbb{Z}^m$ を見つける問題を **SIS 問題**と呼ぶ。つまり，これは q-ary 格子 $\Lambda_q^\perp(\mathbf{X})$ 上の短い非零ベクトルを見つける問題である。剰余パラメータ q における LWE 問題のサンプル数を m とし，m 個の LWE サンプルの組を $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$ とする。ここで，$n \times m$ の転置行列 $\mathbf{A}^\top$ に対する SIS 問題の短い解ベクトル $\mathbf{v} \in \Lambda_q^\perp(\mathbf{A}^\top)$ が得られたとする ($0 < \|\mathbf{v}\| \leq \beta$ と仮定)。このとき，LWE サンプルの組 $(\mathbf{A}, \mathbf{b})$ は関係式 $\mathbf{b} \equiv \mathbf{s}\mathbf{A}^\top + \mathbf{e} \pmod q$ を満たすので，$\langle \mathbf{v}, \mathbf{b}\rangle \equiv \langle \mathbf{v}, \mathbf{s}\mathbf{A}^\top + \mathbf{e}\rangle \equiv \langle \mathbf{v}\mathbf{A}, \mathbf{s}\rangle + \langle \mathbf{v}, \mathbf{e}\rangle \equiv \langle \mathbf{v}, \mathbf{e}\rangle \pmod q$ が成り立つ ($\mathbf{v}\mathbf{A} \equiv \mathbf{0} \pmod q$ に注意) 。さらに，ノイズベクトル $\mathbf{e}$ のすべての成分 e_i は離散 Gauss 分布 χ からサンプルされた元なので，$|\langle \mathbf{v}, \mathbf{e}\rangle| \lesssim \sigma\sqrt{m}\|\mathbf{v}\| \leq \sigma\beta\sqrt{m}$ が期待できる。(離散 Gauss 分布 $\chi = D_{\mathbb{Z},\sigma}$ のサンプル元 e_i の絶対値はおおよそ σ 未満で，多めに見積もって $\|\mathbf{e}\| \lesssim \sigma\sqrt{m}$ とした。) ゆえに，$\sigma\beta\sqrt{m} \ll q$ ならば，$|\langle \mathbf{v}, \mathbf{b}\rangle| \pmod q$ の値の大きさから LWE サンプルの組 $(\mathbf{A}, \mathbf{b})$ は確率分布 $L_{\mathbf{s},\chi}$ からサンプルされたものか判定できる。

■ **探索 LWE 問題に対する求解法 (Primal Attack)** 探索 LWE 問題を BDD (Bounded Distance Decoding) 問題に帰着して解く方法を紹介する：格子 L と目標ベクトル $\mathbf{w}$ に対し，ある $0 < \mu \leq \frac{1}{2}$ が存在し $\mathrm{dist}(\mathbf{w}, L) = \min_{\mathbf{v}\in L}\|\mathbf{w} - \mathbf{v}\| < \mu\lambda_1(L)$ を満たすと仮定する。格子 L の基底が与えられたとき，目標ベクトル $\mathbf{w}$ に最も近い格子ベクトル $\mathbf{v} \in L$ を見つける問題を **BDD 問題**と呼ぶ。m 個の LWE サンプルの組 $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$ は関係式 $\mathbf{b} \equiv \mathbf{s}\mathbf{A}^\top + \mathbf{e} \pmod q$ を満たすので，探索 LWE 問題は $\mathbf{b}$ を目標ベクトルとする q-ary 格子 $\Lambda_q(\mathbf{A}^\top)$ 上の BDD 問題とみなせる。実際, 目標ベクトル $\mathbf{b} = \mathbf{s}\mathbf{A}^\top + \mathbf{e} + q\mathbf{z}$ ($\exists \mathbf{z} \in \mathbb{Z}^m$) に対して, 格子ベクトルを $\mathbf{v} = \mathbf{s}\mathbf{A}^\top + q\mathbf{z} \in \Lambda_q(\mathbf{A}^\top)$ とおくと，$\mathbf{b} - \mathbf{v} = \mathbf{e}$ が成り立つ。ノイズベクトル $\mathbf{e}$ のすべての成分 e_i は離散 Gauss 分布 χ からサンプルされた元であるため，分散と次元が大きい場合にはおおよそスケーリングされたカイ二乗分布に従い，高い確率で $\|\mathbf{e}\| \approx \frac{\sigma}{\sqrt{2\pi}} \cdot \sqrt{m}$ となる。ゆえに，目標ベクトル $\mathbf{b}$ との距離が $\sigma\sqrt{m}$ 以下となる q-ary 格子 $\Lambda_q(\mathbf{A}^\top)$ 上の格子ベクトル $\mathbf{v}$ を見つけることで，ノイズベクトル $\mathbf{e}$ を復元することができる。実用的には，Kannan や Bai-Galbraith らの埋め込み法 [94, 21] により，BDD 問題を unique-SVP に帰着してからノイズベクトル $\mathbf{e}$ を復元する。

注意 3.2 (LWE 問題の変種に対する求解) LWE 問題の代表的な変種である Ring-LWE や Module-LWE では，上述したように多項式環 $R_q = \mathbb{Z}_q[x]/(\phi)$ を基礎環として利用する。n 次多項式 ϕ に対して，基礎環 $R_q = \mathbb{Z}_q[x]/(\phi)$ の任意の元は $n-1$ 次以下の多項式 $f = f_0 + f_1 x + \cdots + f_{n-1}x^{n-1}$ ($f_i \in \mathbb{Z}_q$) と表せ，その係数ベクトル $\mathbf{f} = (f_0, f_1, \ldots, f_{n-1}) \in \mathbb{Z}_q^n$ と一対一に対応する。このように，基礎環 R_q の元をその係数ベクトルに対応させることで，Ring-LWE や Module-LWE 問題は通常の LWE 問題と同じようにベクトル・行列の形で表現できる。(詳細は [5] を参照。また，ベクトル・行列の形の表現については，次で説明する NTRU 問題も参照。) ベクトル・行列の形で表現した Ring-LWE や Module-LWE 問題に対して，上述で説明した通常の LWE 問題の求解法が適用できる。

3.1.2 NTRU 問題と代表的な求解法

ここでは，NTRU 問題とその代表的な求解法を紹介する。まず以下で，NTRU 問題について述べる：

定義 3.3 (NTRU 問題 [87]) 2 つの正の整数 n と q に対し，$\phi \in \mathbb{Z}[x]$ を次数 n の多項式とし，$R_q = \mathbb{Z}_q[x]/(\phi)$ とする。係数が小さい 2 つの多項式 $f \in R_q^\times, g \in R_q$ に対して，$h = g \cdot f^{-1} \in R_q$ とする。(特に，f は環 R_q の可逆元に注意。) このとき，与えられた多項式 h から，f または g の多項式を復元する問題を (探索) NTRU 問題という。

NTRU 問題における多項式 ϕ の選び方として，$\phi = x^n \pm 1, x^n - x - 1, x^n - x^{n/2} + 1, \sum_{i=0}^{n-1} x^i$ などがある [7, Table 1]。(最後の ϕ のみ次数は $n-1$ である。) また，多項式 f (または g) の選び方として，$\{-1, 0, 1\}$ などの小さい係数を持つ多項式や，小さい素数 p と係数が小さい多項式 F に対し $f = pF$ または $f = pF + 1$ と選ぶことが多い。

次に，NTRU 問題の代表的な求解法を紹介する。まず，与えられた多項式 $h \in R_q$ に対して，h の回転行列を $\mathbf{H} \in \mathbb{Z}^{n\times n}$ とする。(具体的には，$n \times n$ 整数行列 $\mathbf{H}$ の i 行ベクトルを多項式 $x^{i-1}h \in R_q$ を次数の昇順に並べた係数ベクトルとする。) このとき

$$\mathbf{H}\begin{pmatrix} 1 \\ x \\ \vdots \\ x^{n-1} \end{pmatrix} = \begin{pmatrix} h \\ xh \\ \vdots \\ x^{n-1}h \end{pmatrix} \in R_q^n$$

が成り立つ。ここで，$2n \times 2n$ 行列 $\mathbf{B} = \begin{pmatrix} \mathbf{I}_n & \mathbf{H} \\ \mathbf{0} & q\mathbf{I}_n \end{pmatrix}$ の行ベクトルで生成される **NTRU 格子**を L とすると，短いベクトル $(\mathbf{f} \mid \mathbf{g}) \in \mathbb{Z}^{2n}$ を含む。(ただし，$\mathbf{f}, \mathbf{g} \in \mathbb{Z}^n$ を多項式 $f, g \in R_q$ の係数ベクトルとする。) 実際，$hf = g \pmod q$ より，$g = hf + qr$ を満たす多項式 $r \in R_q$ が存在する。また，多項式 r の係数ベクトルを $\mathbf{r} \in \mathbb{Z}^n$ とすると，

$$\mathbf{g}\begin{pmatrix} 1 \\ x \\ \vdots \\ x^{n-1} \end{pmatrix} = g = hf + qr = \mathbf{f}\begin{pmatrix} h \\ xh \\ \vdots \\ x^{n-1}h \end{pmatrix} + q\mathbf{r}\begin{pmatrix} 1 \\ x \\ \vdots \\ x^{n-1} \end{pmatrix} = (\mathbf{fH} + q\mathbf{r})\begin{pmatrix} 1 \\ x \\ \vdots \\ x^{n-1} \end{pmatrix} \in R_q$$

となるので，$\mathbf{g} = \mathbf{fH} + q\mathbf{r}$ が成り立つ。これより，$(\mathbf{f} \mid \mathbf{g}) = (\mathbf{f} \mid \mathbf{fH} + q\mathbf{r}) = (\mathbf{f} \mid \mathbf{r})\mathbf{B} \in L$ が成り立つ。(つまり，ベクトル $(\mathbf{f} \mid \mathbf{g})$ が NTRU 格子 L に含まれる。) ベクトル $(\mathbf{f} \mid \mathbf{g}) \in \mathbb{Z}^{2n}$ が十分短く NTRU 格子 L 上の最短ベクトルと仮定すると，これは NTRU 問題を SVP に帰着できることを示している。多くの方式で用いられる $\phi = x^n \pm 1$ に対しては，最短ベクトル $(\mathbf{f} \mid \mathbf{g})$ の各ブロックにおける回転で得られるベクトルも NTRU 格子 L に含まれるので，一般に NTRU 格子 L は複数の最短ベクトルを含む。

3.1.3 格子問題を解くアルゴリズムとその計算量について

SVP・CVP の格子問題やここまでに紹介した LWE 問題・NTRU 問題などの格子問題を解くのに有用な技術として**格子基底簡約**がある。格子基底簡約は，与えられた格子 L の基底から，各ベクトル $\mathbf{b}_i$ が短く・互いのベクトルが直交に近い格子 L の新しい基底 $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$ を見つける操作である。(明確な定義はないが，このような基底を「簡約基底」または「良い基底」と呼ぶ。)

3.1.3.1 代表的な格子基底簡約アルゴリズムの紹介

基底簡約アルゴリズムを紹介するために，Gram-Schmidt の直交化を説明する：基底 $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$ の Gram-Schmidt ベクトル $\mathbf{b}_i^*$ は次のように再帰的に定まる：$\mathbf{b}_1^* = \mathbf{b}_1, \mathbf{b}_i^* = \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j}\mathbf{b}_j^*$, $\mu_{i,j} = \dfrac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\|\mathbf{b}_j^*\|^2}$。また，各 $2 \le \ell \le m$ に対し $\mathbb{R}^m$ から $\mathbb{R}$-ベクトル空間 $\langle \mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{\ell-1} \rangle_{\mathbb{R}}$ の直交補空間への直交射影を π_ℓ とかく。(便宜上，π_1 を恒等写像とする。) 以下で，代表的な 2 つの格子基底簡約アルゴリズムを紹介する。

■ LLL [99] 簡約パラメータ $\dfrac{1}{4} < \delta < 1$ に対し，LLL 基底簡約は次の 2 条件を満たす基底 $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$ を見つける （次元 m に関する）多項式時間アルゴリズムである： (i) 基底 $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$ はサイズ簡約されている：Gram-Schmidt 係数が $|\mu_{i,j}| \le \dfrac{1}{2}$ $(\forall i > \forall j)$ を満たす。 (ii) 基底 $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$ は Lovász 条件を満たす：$\delta\|\mathbf{b}_{k-1}^*\|^2 \le \|\pi_{k-1}(\mathbf{b}_k)\|^2$ $(2 \le \forall k \le m)$ を満たす。入力基底に対して，Lovász 条件が成り立たないとき LLL 基底簡約アルゴリズム内で隣り合う基底ベクトル $\mathbf{b}_{k-1}$ と $\mathbf{b}_k$ の交換を行い，基底をサイズ簡約する。この操作を (i) と (ii) の 2 条件を満たすまで繰り返すことで LLL 簡約基底が計算される。

■ BKZ [142] BKZ 基底簡約アルゴリズムは，ブロックサイズ β による LLL 基底簡約アルゴリズムの一般化である。LLL に比べ，BKZ 基底簡約アルゴリズムでより良い簡約基底を見つけることが可能であるが，その計算量は β に関し

て指数時間である。特に，BKZ 基底簡約アルゴリズムに入力するブロックサイズ β を増やすごとに，実行時間が非常に遅くなるが，より短い基底ベクトルを出力する。具体的には，ブロックサイズ $2 \leq \beta \leq m$ に対して，BKZ 基底簡約アルゴリズムは次の 2 つの条件を満たす格子 L の基底 $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$ を見つける： (i) 基底はサイズ簡約されている。 (ii) すべての $1 \leq j \leq m$ に対し $\|\mathbf{b}_j^*\| = \lambda_1(L_{[j,k]})$ を満たす。ただし，$k = \min(j+\beta-1, m)$ とし，射影ベクトル $\pi_j(\mathbf{b}_j), \ldots, \pi_j(\mathbf{b}_k)$ で生成されるブロック射影格子を $L_{[j,k]}$ とする。入力基底に対して，BKZ 基底簡約アルゴリズム内ではブロック射影格子 $L_{[j:k]}$ 上の SVP オラクルを繰り返し呼びだし，(i) と (ii) の 2 条件を満たす基底を見つける。

3.1.3.2 BKZ 基底簡約アルゴリズムの出力基底と計算量

これまで BKZ2.0 [46] などの効率的な BKZ の改良アルゴリズムが提案され，格子に基づく暗号技術の安全性評価で頻繁に利用されている。以下で，BKZ の出力基底と計算量評価の見積もりについて紹介する。 (詳細は [7] を参照。)

■ BKZ の出力基底の見積もり 格子基底簡約アルゴリズムが出力する簡約基底の「良さ」を測る指標として Hermite 因子がある。m 次元格子 L の基底が与えられたとき，アルゴリズムが出力する最短な基底ベクトルを $\mathbf{b} \in L$ とする。このとき，その基底簡約アルゴリズムの Hermite 因子は $\gamma = \dfrac{\|\mathbf{b}\|}{\mathrm{vol}(L)^{1/m}}$ で定義される。(つまり，Hermite 因子が小さいほど，より短い基底ベクトルの出力を意味する。) 100 以上の高次元のランダム格子に対し，LLL や BKZ などの基底簡約アルゴリズムの Hermite 因子の m 乗根 $\gamma^{1/m}$ は格子の次元 m によらず定数に収束することが実験的に知られている。高い次元 m のランダム格子において，ブロックサイズ $\beta \geq 50$ に対する BKZ 基底簡約アルゴリズムの root Hermite 因子はおおよそ

$$\gamma^{\frac{1}{m}} \approx \left(\nu_\beta^{-\frac{1}{\beta}}\right)^{\frac{1}{\beta-1}} \approx \left(\frac{\beta}{2\pi e}(\pi\beta)^{\frac{1}{\beta}}\right)^{\frac{1}{2(\beta-1)}}$$

に従うことが実験的に知られている [46, 159]。ただし，ν_β は β-次元の単位超球の体積とする。(例えば，$\beta = 85$ で $\gamma^{1/m} \approx 1.01$ となる。) この root Hermite 因子の見積もりを用いて，格子に基づく暗号技術の安全性評価対象の格子問題の求解で必要となる BKZ のブロックサイズ β を求めることができる。

■ BKZ の計算量の見積もり BKZ 基底簡約アルゴリズムの計算量は，β 次元格子上の「SVP オラクルの計算量」と「呼び出し回数」の積で見積もることができる。β 次元格子上の SVP オラクルに適したアルゴリズムとして篩 (sieving) と数え上げ (enumeration) があり，篩の方が漸近計算量が小さい。(ただし，数え上げの空間計算量が β に関して多項式的であるのに対し，篩の空間計算量は β に関して指数関数的である。) 具体的には，β 次元格子上の篩の時間計算量は $2^{c\beta+o(\beta)}$ で，古典計算機上では $c = 0.292$ で，Montanaro の量子木探索アルゴリズムによって量子計算機上で $c = 0.265$ と見積もられている。一方，数え上げの時間計算量は古典計算機上で $2^{c_1\beta\log\beta+c_2\beta+c_3}$ または $2^{c_1\beta^2+c_2\beta+c_3}$ で，Grover アルゴリズムにより量子計算機上ではその指数部分が半分になると見積もられている。(定数 c_1, c_2, c_3 に関しては様々な評価値があり，具体的な値については [7, Table 4] を参照。) また，BKZ 内の SVP オラクルの呼び出し回数については，β または $8m$ と見積もることが多い。 (β は BKZ のブロックサイズで，m は格子の次元とする。)

■ Core-SVP による安全性レベルの見積もり 上述の LWE や NTRU の探索問題の求解において，秘密情報に対応するベクトル $\mathbf{v}$ を帰着先の格子 L の最短ベクトルとして埋め込み，L の基底に BKZ 基底簡約アルゴリズムを適用することで目的の $\mathbf{v}$ を見つけることを考える。(具体的には，目的の $\mathbf{v}$ は，探索 LWE 問題では q-ary 格子上のノイズベクトル $\mathbf{e}$ で，NTRU 問題では NTRU 格子上のベクトル $(\mathbf{f} \mid \mathbf{g})$ を想定する。) ここで，$\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$ を格子 L の β-BKZ 簡約基底とし，$\{\mathbf{b}_1^*, \ldots, \mathbf{b}_m^*\}$ をその Gram-Schmidt ベクトルとする。Gaussian Heuristic と Geometric Series Assumption (GSA) の仮定の下で，目的ベクトル $\mathbf{v}$ の $m-\beta$ の位置における射影ベクトル $\pi_{m-\beta}(\mathbf{v}) \in \pi_{m-\beta}(L)$ の長さが

$$\|\pi_{m-\beta}(\mathbf{v})\| < \|\mathbf{b}_{m-\beta}^*\| \approx \delta_\beta^{2\beta-m-1}\mathrm{vol}(L)^{\frac{1}{m}}, \quad \delta_\beta = \left(\frac{\beta}{2\pi e}(\pi\beta)^{\frac{1}{\beta}}\right)^{\frac{1}{2(\beta-1)}}$$

を満たせば，BKZ 基底簡約の基底ベクトルとして目的の $\mathbf{v}$ を見つけることができる。（探索 LWE 問題に対する BKZ による求解実験については，[10, 129] を参照。）この不等式を満たす BKZ のブロックサイズ β に対して，BKZ 基底簡約アルゴリズムのサブルーチンである β-次元 SVP アルゴリズムの 1 回の計算困難性を Core-SVP 困難性と呼ぶ [12]。格子に基づく暗号方式の具体的な安全性レベルは，Core-SVP 困難性で評価・比較されることが多い。

3.1.3.3 格子問題の公開チャレンジの求解状況

SVP や LWE に対する求解アルゴリズムをテストする目的で，ドイツ・ダルムシュタット工科大学によって「SVP チャレンジ」・「LWE チャレンジ」と呼ばれる求解コンテストがインターネット上で開催されている [56]。2018 年に，篩（ふるい）をベースとした高速な格子アルゴリズム群である General Sieve Kernel (G6K)[9] が提案され，SVP チャレンジ・LWE チャレンジの求解記録が飛躍的に更新された。具体的には，SVP チャレンジにおいては，G6K 内の篩アルゴリズムを GPU 実装することで，180 次元の SVP インスタンスが 4 台の NVIDIA Turing GPU と 1.5TBytes の RAM を搭載した計算機を用いて 51.6 日で求解されたと 2021 年 2 月に報告されている [69]。

（ただし，本報告では Gaussian Heuristic で期待される最短ベクトル長に対する近似因子が 1.04002 なので，今回見つかった格子ベクトルは 180 次元 SVP インスタンスの厳密解ではなく近似解である。）また 2023 年 7 月に，186 次元の SVP インスタンスに対して，次のスペックを持つ計算機システムで約 50 日程度で近似因子が 1.01405 の非常に短い格子ベクトルを見つけることに成功している（計算時間の内訳は，Progressive pnj-BKZ による基底簡約に 12.3 日，Sieving に短い格子ベクトルの探索に 38 日かかったと報告されている）[57]。

- CPU: 1 ∗ Intel Xeon Gold 6330, 56 threads @ 2.00GHz
- GPU: 4 ∗ NVIDIA A100 80GB PCIe
- Max RAM used: 1441.6685 GB

さらに，2024 年 7 月に，190 次元の SVP インスタンスに対する近似因子 1.04237 のベクトルが発見されている。G6K ライブラリによる $\beta = 155$〜158 の篩アルゴリズムを，4 台の NVIDIA GeForce RTX 4090 と 1.5TBytes の RAM を搭載した計算機，および 3 台の NVIDIA GeForce RTX 4090 D と 2.0TBytes の RAM を搭載した計算機を協調させて動かすことで記録を出したと報告されている。

LWE チャレンジでは，$(n, \alpha) = (40, 0.040), (45, 0.030), (50, 0.025), (55, 0.020), (90, 0.005)$ の数多くの LWE インスタンスが G6K 内の progressive-BKZ の改良により求解されたと 2022 年 6〜10 月に報告されている。（n は LWE の秘密ベクトル長で，α はノイズの大きさに関するパラメータで，組 (n, α) のバランスで LWE インスタンスの難しさが大きく変化する。）例えば，$(n, \alpha) = (50, 0.025)$ と $(40, 0.040)$ の 2 つの LWE インスタンスに対して，次のスペックを持つ計算システムでそれぞれ約 592 時間と約 683 時間で求解されている [158]：

- CPU：AMD EPYC 7002 Series 128@2.6GHz
- RAM：1.5TBytes
- GPU：8 ∗ NVIDIA GeForce RTX 3090
- VRAM：8 ∗ 24GB（936.2 GB/s）

2024 年 9 月には，$(n, \alpha) = (95, 0.005)$ の LWE インスタンスに対して，最大 144 の篩次元を用いて約 46 日で求解したと報告されている。使用した計算機は 8 台の NVIDIA RTX 4090, 2 基の Intel Xeon Platinum 8480+, 2TBytes RAM を搭載している。

3.2 格子に基づく代表的な暗号方式

本節では，格子に基づく代表的な暗号方式として，LWE 問題に基づく Regev による公開鍵暗号方式 [133] および Lindner, Peikert による公開鍵暗号方式 [101]，Ring-LWE 問題に基づく Brakerski らによる公開鍵暗号方式 [39]，NTRU 問題に基づく Hoffstein らによる公開鍵暗号方式 [87]，Hash-and-Sign に基づく署名方式の格子問題への拡張，ならびに Fiat-Shamir 署名方式の格子問題への拡張について述べる。

3.2.1 LWE に基づく Regev による公開鍵暗号方式

Regev による公開鍵暗号方式 [133] の構成には，以下の 4 つのパラメータが必要である。

- n: 安全性パラメータ
- m: LWE サンプルの個数 ($m \geq 1.1 \cdot n\log q$ となる最小の整数を選ぶ)
- q: 剰余パラメータ (q として $n^2 \leq q \leq 2n^2$ を満たす素数を選ぶ)
- $\alpha > 0$: ノイズパラメータ ($\alpha = 1/(\sqrt{n} \cdot \log^2 n)$)

以下に具体的な暗号方式の構成を示す。

秘密鍵の生成 一様ランダムに $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ を選ぶ。

公開鍵の生成 秘密鍵 $\mathbf{s}$，剰余パラメータ q，ノイズパラメータ α を持つ LWE 分布から生成した m 個のサンプル $(\mathbf{a}_i, b_i)_{i=1}^m \leftarrow A_{\mathbf{s},\chi}^m$ を公開鍵とする。ただし各 i について，$a \leftarrow \mathbb{Z}_q^n, e_i \leftarrow \chi = D_{\mathbb{Z},\alpha q}$ とした時，$b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i \in \mathbb{Z}_q$ とする。

暗号化 集合 S を $\{1, 2, \ldots, m\}$ の中から一様ランダムに選んだ部分集合とする。平文 $\mu\{0,1\}$ を以下のように暗号化する。平文ビットが 0 のとき，暗号文を $\left(\sum_{i\in S} \mathbf{a}_i, \sum_{i\in S} b_i\right)$ とする。平文ビットが 1 のとき，暗号文を $\left(\sum_{i\in S} \mathbf{a}_i, \left\lfloor \frac{q}{2} \right\rfloor + \sum_{i\in S} b_i\right)$ とする。

復号 暗号文 $(\mathbf{a}, b)$ に対し，$b - \langle \mathbf{a}, \mathbf{s} \rangle \in \mathbb{Z}_q$ が $\left\lfloor \frac{q}{2} \right\rfloor$ より 0 に近い場合，復号結果として 0 を出力し，それ以外の場合は 1 を出力する。

復号の正当性について，平文 0 を暗号化した暗号文 $(\mathbf{a}, b) = \left(\sum_{i\in S} \mathbf{a}, \sum_{i\in S} b_i\right)$ の場合，$b - \langle \mathbf{a}, \mathbf{s} \rangle \in \mathbb{Z}_q = \sum_{i\in S}(b_i - \langle \mathbf{a}_i, \mathbf{s} \rangle) = \sum_{i\in S} e_i$ なので，$-\frac{q}{4} < \sum_{i\in S} e_i < \frac{q}{4}$ であれば復号に成功し，0 が出力される。

各ノイズ e_i は離散 Gauss 分布 $\chi = D_{\mathbb{Z},\alpha q}$ から選ばれているので，$\sum_{i\in S} e_i$ の標準偏差は高々 $\sqrt{m}\alpha q$ となる。

ここで，各パラメータの選択方法から $\sqrt{m}\alpha q < q/\log n$ であり，非常に高い確率で復号に成功することが分かる。また平文 1 を暗号化した暗号文に対しても同様の議論が成り立つ。この暗号方式の安全性については，LWE 仮定の下で IND-CPA 安全であることが証明されている [134]。

ここで紹介した [133] による暗号方式は，公開鍵のサイズが $O(mn\log q) = \widetilde{O}(n^2)$ で，暗号文サイズも平文サイズの $O(n\log q) = \widetilde{O}(n)$ 倍に増加するため，決して効率的ではない。より効率的な方式としては [124] などを参照。

3.2.2 LWE に基づく Lindner, Peikert らによる公開鍵暗号方式

Lindner，Peikert らによる公開鍵暗号方式 [101] の構成には，以下のパラメータが必要である。

- n_1, n_2 LWE 問題の安全性パラメータ
- s_k, s_e：鍵生成時，暗号化時のノイズ付与のための離散 Gauss 分布パラメータ
- q：q-ary 格子を構成する剰余パラメータ $(q > 2)$
- l：平文空間の次元，$\{0,1\}^l$ を平文の対象空間とする

以下に具体的な暗号方式の構成を示す。

鍵生成 一様ランダムに $A \leftarrow \mathbb{Z}_q^{n_1 \times n_2}$ を選ぶ。$S \leftarrow D_{\mathbb{Z},s_k}^{n_2 \times l}$ を秘密鍵とする。$E \leftarrow D_{\mathbb{Z},s_k}^{n_1 \times l}$ を生成し，$P = E - AS \in \mathbb{Z}_q^{n_1 \times l}$ を計算。(A, P) を公開鍵とする。

暗号化 暗号化用のノイズとして $(e_1, e_2, e_3) \leftarrow D_{\mathbb{Z},s_e}^{1 \times n_1} \times D_{\mathbb{Z},s_e}^{1 \times n_2} \times D_{\mathbb{Z},s_e}^{1 \times l}$ をサンプリング。メッセージ $m \in \{0,1\}^l$ に対して $c = (e_1 A + e_2, e_1 P + e_3 + m \left\lfloor \frac{q}{2} \right\rfloor)$ とし c を暗号文とする。

復号 $v = c_1 S + c_2$ を求め，各要素毎に m_i' を $|v_i| < \frac{q}{4}$ であれば 0，それ以外であれば 1 として $m' = (m_1', \ldots, m_l')$ を復号文とする。

復号の正当性について，暗号文 $c = (e_1 A + e_2, e_1 P + e_3)$ に対し，$v = c_1 S + c_2 = e_1 AS + e_2 S + e_1 P + e_3 + m \left\lfloor \frac{q}{2} \right\rfloor = e_2 S + e_1 E + e_3 + m \left\lfloor \frac{q}{2} \right\rfloor$ となる。秘密鍵 S ならびに E の各成分は，Gauss 分布 $D_{\mathbb{Z},s_k}$ から，各ノイズ e_i の成分は Gauss 分布 $D_{\mathbb{Z},s_e}$ から選ばれており，また s_k, s_e の選び方から，高い確率で $|v| < s_e s_k (n_1 + n_2) < \frac{q}{4}$ を満たし，復号に成功することが分かる。この暗号方式の安全性については LWE 仮定の下で IND-CPA 安全であることが証明されている。

3.2.3 Ring-LWE に基づく Brakerski らによる公開鍵暗号方式

Brakerski らによる Ring-LWE 問題に基づく公開鍵暗号方式は，暗号化したまま限定回の加算と乗算が可能な somewhat 準同型暗号として提案されているものである。この暗号方式には，以下の 4 つのパラメータが必要である。

- n: 2 のべき乗の整数で，暗号方式を構成する基礎的な環 $R = \mathbb{Z}[x]/(x^n + 1)$ を定義する (n が 2 べきであることから，多項式 $x^n + 1$ が $\mathbb{Z}$ 上既約となることに注意)。
- q: $q \equiv 1 \pmod{2n}$ を満たす素数で，暗号文空間の基礎環 $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ を定義する。
- t: 条件 $t < q$ を満たす整数で，暗号方式の平文空間 $R_t = \mathbb{Z}_t[x]/(x^n + 1)$ を定義する。
- $\sigma > 0$: ノイズを与えるための離散 Gauss 分布のパラメータ。

以下に具体的な暗号方式の構成を示す。なお，$a_0 + a_1 x + \cdots + a_{n-1} x^{n-1} \to (a_0, a_1, \ldots, a_{n-1})$ によって，環 R を $\mathbb{Z}^n$ と同一視する。また同様に R_q と $\mathbb{Z}_q^n$ を同一視する。

鍵生成 $s \leftarrow D_{\mathbb{Z},\sigma}^n$ を生成する。一様ランダムに $p_1 \leftarrow R_q$ を取りる。小さなエラー $e \leftarrow \chi$ を固定する。([39] では $s \leftarrow \chi$ を一様ランダムに選択するのに対し，[115] では一様ランダムには選択しない点だけが異なる)。公開鍵を $\mathsf{pk} = (p_0 \quad = -(p_1 s + te), p_1)$ とし，秘密鍵を $\mathsf{sk} = s$ とする。

暗号化 平文情報 $m \in R_t$ と公開鍵 $\mathsf{pk} = (p_0, p_1)$ に対し，まず χ から $u, f, g \in R$ をサンプリングし，暗号文を

$$\mathsf{Enc}(m, \mathsf{pk}) = (c_0, c_1) = (p_0 u + tg + m, p_1 u + tf),$$

と定義する。ただし，条件 $t < q$ より，この数式では元 $m \in R_t$ を環 R_q の元として見なして計算する。つまり，暗号文は $(R_q)^2$ の元として表現される。

復号 任意の長さの暗号文 $\mathsf{ct} = (c_0, c_1, \ldots, c_\xi)$ に対して，復号は

$$\mathsf{Dec}(\mathsf{ct}, \mathsf{sk}) = [\tilde{m}]_q \bmod t \in R_t,$$

で計算できる。ただし，$\tilde{m} = \sum_{i=0}^{\xi} c_i s^i \in R_q$ であり，$[\tilde{m}]_q$ は元 $\tilde{m}$ の各係数の $[-q/2, q/2)$ への剰余とする。また，$\mathbf{s} = (1, s, s^2, \ldots, s^\xi)$ としたとき，この復号処理を $\mathsf{Dec}(\mathsf{ct}, \mathsf{sk}) = [\langle \mathsf{ct}, \mathbf{s} \rangle]_q \bmod t$ と書き直すこともできる。

この復号アルゴリズムの正当性については，暗号アルゴリズムで得られる暗号文 $\mathsf{ct} = (c_0, c_1)$ に対し，関係式 $p_0 + p_1 s = -te$ が成り立つことから，$\langle \mathsf{ct}, \mathbf{s} \rangle = (p_0 u + tg + m) + s \cdot (p_1 u + tf) = m + t \cdot (g + sf - ue)$ が環 R_q 上で成り立つ。ここで，元 $m + t \cdot (g + sf - ue)$ を環 R の元と見なしたとき，その各係数が $[-q/2, q/2)$ 内に収まっている限り，$[\langle \mathsf{ct}, \mathbf{s} \rangle]_q = m + t \cdot (g + sf - ue)$ が環 R 上で成立する (元 $e, f, g, u \leftarrow \chi$ が十分小さなノイズとして選択されていることに注意)。この場合，剰余 $\bmod t$ の操作で正しい復号結果 $m \in R_t$ が得られる。

また，この暗号方式の安全性については，Ring-LWE 問題の計算量困難性仮定の下で KDM 安全 (key dependent message security) であることが証明されている [39]。

3.2.4 NTRU 問題に基づく Hoffstein らによる公開鍵暗号方式

Hoffstein らによる NTRU 問題に基づく公開鍵暗号方式 NTRUEncrypt [87] の構成には次のパラメータが必要である。

- n : 正の整数 (セキュリティパラメータ)
- q : 正の整数 (素数である必要はない)
- p : q と互に素で $p \ll q$ である正の整数
- ϕ: 次数 n の多項式であり環 $R_p = \mathbb{Z}_p[x]/(\phi), R_q = \mathbb{Z}_q[x]/(\phi)$ を定義する (ϕ としては例えば $x^n \pm 1, x^n - x - 1$ 等)

以下に具体的な暗号方式の構成を示す。

鍵生成 すべての係数の絶対値が小さい二つの多項式 $f \in R_q, g \in R_q$ を選ぶ。ただし，f は R_p, R_q の両方において可逆な要素とする。すなわち，ある f_p, f_q が存在し，以下を満たす。

$$f_p \cdot f = 1 \in R_p, f_q \cdot f = 1 \in R_q$$

ここで f, f_p を秘密鍵とし，$h = p f_q \cdot g \in R_q$ を公開鍵とする。なお f_p, f_q ならびに g は f と h を用いて復元可能であることに注意する。

暗号化 平文情報として，すべての係数の絶対値が p より小さい (例えば $-1, 0, 1$ のいずれかである) 要素 $m \in R_q$ とし，公開鍵 $\mathsf{pk} = h$ に対し，$r \in R_q$ を係数が小さい多項式からランダムに選び，暗号文を

$$\mathsf{Enc}(m, \mathsf{pk}) = r \cdot h + m \in R_q$$

と定義する。

復号 暗号文 $c \in R_q$ に対し，復号は

$$\mathsf{Dec}(m, \mathsf{sk}) = [f_p \cdot [f \cdot c]_q]_p$$

で求められる。ただし，$[a]_q, [a]_p$ は元 $a \in R_q$ の各係数をそれぞれ $[-q/2, q/2), [-p/2, p/2)$ に収めたものとする。

復号の正当性については, 次のように示される。$[f \cdot c]_q$ は, $f \cdot c = f \cdot (r \cdot h + m) = f(r \cdot p f_q \cdot g + m) = pr \cdot g + f \cdot m \in R_q$ と変形されるが，r, g, f, m 共に，係数が小さいものから抽出しており，また $p \ll q$ であること，更に係数が $[-q/2, q/2)$ に収められていることから適切なパラメータ選択により，$f \cdot c$ は q による剰余を伴わない等式，すなわち

$f \cdot c = pr \cdot g + f \cdot m \in Z[x]/(\phi)$ が満たされる。また右辺第一項は p 倍項であることから，続く p による剰余で消去され，$f_p \cdot (pr \cdot g + f \cdot m) = f_p \cdot f \cdot m = m \in R_p$ となり正しい復号結果 m が得られる。

NTRUEncrypt の安全性についてはアルゴリズム提案当初格子問題への安全性帰着がついていなかったが，Stehlé ら [148] により，standard model での IND-CPA 仮定に基づくイデアル格子上の Ring-SIS 問題，ならびに Ring-LWE 問題に帰着されることが示されている。

3.2.5 Hash-and-Sign に基づく署名方式の格子問題への拡張

Hash-and-Sign に基づく署名方式は，Diffie,Hellman らによってその基本形が示されており，落とし戸つき一方向性関数 $f(x)$ ならびに $f^{-1}(x)$ を用いて署名・検証が行われる。

- M：メッセージ
- $h = hash(M)$: 暗号学的ハッシュ関数
- $\sigma = f^{-1}(h)$：署名
- $h = f(\sigma)$ が成り立つかを確認：署名検証

Diffie,Hellman らによる方式では，一方向性関数 $f(x)$ として，素数 p を法とした離散対数問題に基づく関数 $f(x) = a^x \mod p$ が提示されている。

この署名方式は，さまざまな改良が提案されているが，格子問題の困難性に基づく落とし戸つき関数を用いた Hash-and-Sign 署名方式が，Gentry らによって提案されている [83]。以下にその方式を示す。次のパラメータを準備する。

- m, n：正の整数 (セキュリティパラメータ)
- $hash(M)$: 暗号学的ハッシュ関数
- q：素数
- $L = m^{1+\epsilon}, (\epsilon > 0)$：秘密鍵の Euclide ノルムの上限

以下に具体的な署名方式を示す。

鍵生成 一様ランダムに $A \leftarrow \mathbb{Z}_q^{n \times m}$ を生成する。[83] のサンプリング手法を用いて，$\Lambda_q^{\perp}(\mathbf{A})$ から短いベクトル S を生成する。具体的には $||S|| < L$ かつ $SA^T \equiv 0 \mod q$ を満たす。秘密鍵を S，公開鍵を A とする。

署名生成 メッセージ M のハッシュ値 $H = hash(M)$ を乱数のシードとして $D_{\mathbb{Z},s}^m$ からサンプリングを行う。その値を u とする。$tA = u \mod q$ を満たす t を任意に求める。秘密鍵 S を用いて，$-t$ に近い格子 $\Lambda_q^{\perp}(\mathbf{A})$ 上の点 v を求め，$\sigma = v + t$ とする。σ を署名として出力する。

署名検証 メッセージ M にハッシュ関数を作用させた値 $h = hash(M)$ を $D_{\mathbb{Z},s}^m$ にマッピングし，値を u とする。σ が短いベクトルでありかつ $(\sigma - u)A = 0$ である場合に正当な署名として受理する。

署名の正当性については，次のように示される。構成の仕方から，$\sigma - u = v$ であり，v は格子 $\Lambda_q^{\perp}(\mathbf{A}, \mathbf{q})$ 上の点であるから，$(\sigma - u)A \mod q = vA \mod q = 0$ が成り立つ。また 秘密鍵 S の特徴から，$\sigma \in D_{\mathbb{Z}^m,s}$ であることから，σ は短いベクトルとなる。本署名方式は LWE 仮定の元で SUF-CMA(Strong Existential Unforgeability under Chosen Message Attack) 安全であることが示されている。

3.2.6 Fiat-Shamir 署名方式の格子問題への拡張

Fiat-Shamir 変換 [78] に基づく署名方式を総称して Fiat-Shamir 署名と呼ばれており，現在までさまざまな方式が提案されている。以下に基本となる方式の一つである素因数分解問題をベースとする方式を記す。合成数 $n = pq$ (p, q は素数) を法とするべき乗剰余演算 $g(x) = g^x \mod n$ を一方向性関数として利用し，秘密鍵 s，公開鍵 $a = g(s)$ を準備する。

- M：メッセージ
- $h = hash(M)$: 暗号学的ハッシュ関数
- r：ランダムな値
- $(z, y) = (g(r)h + s, g(r))$：署名
- $g(z) = a^r y$ が成り立つかを確認：署名検証

Lyubashevsky [105] によって，Fiat-Shamir with Aborts 型の格子ベースの署名方式が提案されている。以下にその具体的な署名方式について述べる。次のパラメータを準備する。

- $hash(M)$: 入力を係数の小さい多項式に写す暗号学的ハッシュ関数
- m：正の整数 (セキュリティパラメータ)
- n：2 のべき乗 (セキュリティパラメータ)
- σ：正の整数 (セキュリティパラメータ)
- κ：${}_{2^\kappa}C_\kappa > 160$ を満たす整数
- p：$(2\sigma + 1)^m 2^{-128/n}$ 程度の素数
- $R = \mathbb{Z}_p[x]/(x^n + 1)$：多項式剰余環
- $D = \{z \in R \mid ||g||_\infty \leq mn\sigma\kappa\}$：　ハッシュ関数 $h_{\hat{a}}$ の定義域を指定する集合
- $G = \{g \in R \mid ||g||_\infty \leq mn\sigma\kappa - \sigma\kappa\}$：署名空間

ただし，$||z||_\infty$ は z の最大値ノルムとする。以下に具体的な署名方式を示す。

R に属する m 個の多項式の集合 R^m の要素 $\hat{a}$ に対し，D^m 上のハッシュ関数 $h_{\hat{a}}(\hat{z}), (\hat{z} \in D^m)$ を以下のように定める。$h_{\hat{a}}(\hat{z}) = \hat{a} \cdot \hat{z} = a_1 z_1 + \cdots + a_m z_m \in R$。

鍵生成 係数の絶対値の最大値が σ 以下の多項式をランダムに m 個とり，多項式成分のベクトルとして並べたものを $\hat{s}$ とする。D^m のランダムなベクトル $\hat{a}$ によるハッシュ関数 $h_{\hat{a}}(\cdot)$ を作用させた値 $S = h_{\hat{a}}(\hat{s})$ を求め，$\hat{s}$ を秘密鍵，S を公開鍵とする。

署名生成 メッセージを M とする。

多項式を成分とするベクトル $\hat{y} \in D^m$ をランダムに選択し，$c = hash(h_{\hat{a}}(\hat{y})||M), \hat{z} = \hat{y} + c\hat{s}$ を求める。$\hat{z} \in G^m$ となるまで，ベクトル $\hat{y}$ の選択をくりかえす。$\sigma = (\hat{z}, c)$ を署名として出力する。

署名検証 $\hat{z} \in G^m$ ならびに $c = hash(h_{\hat{a}}(\hat{z}) - Sc||M)$ が成り立つ場合に署名を受理する。

この署名方式の正当性は，$h_{\hat{a}}(\hat{z}) - Sc = h_{\hat{a}}(\hat{y} + c\hat{s}) - h_{\hat{a}}(\hat{s})c = h_{\hat{a}}(\hat{y})$ が成り立つことから保証される。安全性については，環 R 上のイデアル格子での γ-SVP の困難性と等価であることが示されている。

3.3 格子に基づく主要な暗号方式

本節では，格子に基づく主要な暗号方式として，6 つの公開鍵暗号方式と 3 つの署名方式を取り上げ，その概要と設計原理を説明する。

格子を用いた主な公開鍵暗号方式の構成として，最初期の Ajtai-Dwork 型 [3]，GGH 型 [84] から近年の [133] による LWE 型 (Regev 型)，[83, 101] に代表される dual-LWE 型，[87] に代表される NTRU 型が存在する。格子を用いた署名の構成としては主に GGH/NTRUSign 型 [84, 87]，Fiat-Shamir with abort 型 [105, 106]，Hash-and-Sign 型 [83, Sect.6]，Plantard-Susilo-Win 型 [126] 等が知られている[*4]。

また，安全性の根拠となる計算問題に関しても，最短ベクトル問題に直接還元するもの，LWE 問題，SIS 問題，LWR 問題およびそれらの Module 版，Ring 版へと還元するもの，NTRU 問題に還元するものへと分類できる。

これらの構成は安全性，実装時の性能，使いやすさなどの面から様々な長所・短所を持つが，できる限り幅広くそれらを解説するため，以下の表 3.1 に挙げる 9 つの方式を紹介する。

表 3.1: 格子に基づく暗号の分類

文献	暗号化	鍵交換	署名
ML-KEM (FIPS 203) [120]	○	○	
ML-DSA (FIPS 204) [119]			○
CRYSTALS-Kyber [19]	○	○	
CRYSTALS-Dilithium [20]			○
FALCON [81]			○
FrodoKEM [13]	○	○	
NewHope [14]	○	○	
NTRU [43]	○	○	
SABER [28]	○	○	

- ML-KEM は CRYSTALS-Kyber に基づく dual-LWE 型の公開鍵暗号である。NIST により FIPS 標準アルゴリズムとして制定されたことから，取り上げる。
- ML-DSA は CRYSTALS-Dilithium に基づく署名方式である。NIST により FIPS 標準アルゴリズムとして制定されたことから，取り上げる。
- CRYSTALS-Kyber は dual-LWE 型の公開鍵暗号方式であり，安全性の根拠に $x^n + 1, n = 2^k$ の形の多項式により定義される環上の Module-LWE 問題の困難性を置いている。NIST PQC 標準化プロジェクトの Selected Algorithm となったことから取り上げる。
- CRYSTALS-Dilithium は Fiat-Shamir 型の署名方式であり，$x^{256}+1$ を定義多項式とする環上の Module-LWE 問題の計算困難性を安全性の根拠としている。 環の性質を用いた数論変換による高速処理とサイズの圧縮が可能であり，公開鍵サイズと署名サイズの和を最小化することを目的としてパラメータ設計を行っている。環の性質を用いた処理の効率化の観点から取り上げる。
- FALCON は Hash-and-Sign 型の署名方式であり，$x^n + 1$ を定義多項式とする NTRU 格子上の SIS 問題の困難性を安全性の根拠としている。格子上の高速フーリエサンプリングを用いた高速な署名生成を特徴とし，方式

[*4] この分類に関しては例えば [90, Sect. 3]，[75, Sect. 5.5] 等を参照。

提案後も数多くの改良が提案されていることから取り上げる。

- FrodoKEM は dual-LWE 型の公開鍵暗号方式であり，安全性の根拠に LWE 問題の計算困難性を仮定している。保守的な構成を設計指針としており，将来 Ring 型や Module 型の構造を持つ格子問題に効率的な解法が発見された場合でも安全性が確保されると期待される。保守的な構成という観点から取り上げる。
- NewHope は dual-LWE 型の公開鍵暗号方式であり，安全性の根拠に $x^n + 1, n = 2^k$ の形の多項式により定義される環上の Ring-LWE 問題の困難性を置いている。環の構造を活用した数論変換による高速実装，サイズの低減という観点から取り上げる。
- NTRU は NTRU 型の公開鍵暗号方式であり，NTRU 格子上の格子問題の計算困難性を安全性の根拠としている。 NTRU 暗号は 1996 年の提案依頼改良が続けられてきたが，本方式は近年の研究成果が数多く盛り込まれた形となるため取り上げる。
- SABER は LWE 型の公開鍵暗号方式であり[*5]，$x^{256} + 1$ を定義多項式とする環上の Module-LWR 問題の計算困難性を安全性の根拠としている。LWR 問題を用いることで実装時のサンプリングを極力負担の少ないものとしている。この観点から取り上げる。

3.3.1 FIPS 203：Module-Lattice-Based Key-Encapsulation Mechanism Standard (ML-KEM)

KEM とは公開されたチャンネル上で 2 者が秘密を共有するアルゴリズム群である。KEM で安全に生成された共有の秘密は共通鍵暗号で用いられ，暗号や認証などの安全なやり取りの中で重要な役割を果たす。ML-KEM [120] は CRYSTALS-Kyber に基づく KEM で，その安全性は Module-LWE 問題の計算量困難性に基づく。具体的には，$n = 256$ に対し $R := \mathbb{Z}[X]/(X^n + 1)$ を基本環とし，素数 $q = 3329$ に対し $R_q := R/qR = \mathbb{Z}_q[X]/(X^n + 1)$ をその剰余環とする。環 R_q の元は $\mathbb{Z}_q$ を係数とする $n - 1$ 以下の次数の多項式 $f = f_0 + f_1 X + \cdots + f_{n-1}X^{n-1}$ と表せ，その係数ベクトル $(f_0, f_1, \ldots, f_{n-1})$ を対応させることで，$\mathbb{Z}_q$ 加群として R_q は $\mathbb{Z}_q^n$ と同型である。ML-KEM は，階数パラメータ $k \in \{2, 3, 4\}$ に対し，$\mathbb{Z}_q$ 加群 $R_q^k \simeq (\mathbb{Z}_q^n)^k$ 上の LWE 問題を安全性の根拠とした KEM である。特に，R_q における乗算を高速化するために，数論変換（Number-Theoretic Transform: NTT）を利用する。ここでは，ML-KEM の最も基本となる構成要素である NTT を説明したのちに，ML-KEM の基本構成について説明する。

3.3.1.1 ML-KEM における数論変換

NTT は，環 R_q の元 f を R_q と同型な環 T_q の元 $\widehat{f}$ に写し，T_q における乗算を利用して効率的に R_q の 2 つの元の乗算を行う手法である。これは $\mathbb{C}$ 上の高速フーリエ変換による多項式乗算と同じアイデアで，NTT はその $\mathbb{Z}_q$ 上版とみなせる。ML-KEM では，$n = 2^8 = 256$ と素数 $q = 3329$ で定まる剰余環 $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ を用いる（ML-KEM の暗号パラメータについては，後述の 3.3.1.3 節を参照）。これらの暗号パラメータ (n, q) において，$\mathbb{Z}_q^\times := \mathbb{Z}_q \setminus \{0\}$ は位数 $q - 1 = 3328 = 2^8 \cdot 13$ の巡回群で，位数 $2^8 = 256 = n$ の巡回部分群 $\langle\zeta\rangle$ を唯一つ含む。具体的には，$\mathbb{Z}_q$ において $\zeta := 17 \bmod q$ が 1 の原始 n 乗根で，$\{\zeta, \zeta^3, \ldots, \zeta^{n-1}\}$ が $\mathbb{Z}_q$ に含まれる 1 の原始 n 乗根のすべてである。ここで，$N = \dfrac{n}{2} = 128$ とおくと，各 $i = 0, 1, \ldots, N - 1$ に対して，$\zeta^{(2i+1)N} \equiv -1 \pmod q$ である。ゆえに，多項式環 $\mathbb{Z}_q[X]$ において，$X^n + 1$ は次のように N 個の 2 次式の積に分解できる。

$$X^n + 1 = \prod_{i=0}^{N-1} \left(X^2 - \zeta^{2i+1}\right) = \prod_{i=0}^{N-1} \left(X^2 - \zeta^{2\mathsf{BitRev}_7(i)+1}\right) \in \mathbb{Z}_q[X]$$

[*5] 仕様書 [28] の設計原理の項には “Encryption: we use a simple LWR version of Regev’s LWE encryption scheme [35], where the encryption part is compressed (using the parameter T) to save on bandwith.” とあるが，Second PQC Standardization Conference の発表スライド [53, p. 5] においてはひな型を dual-LWE 型暗号としている。数式の比較から，どちらも原型と考えることが可能であるため，本報告書では仕様書に従い LWE 型であるとした。

ただし，$\mathsf{BitRev}_7(i)$ は符号なし 7 ビット整数 i のビット逆順整数を表し，実装上の都合のため ML-KEM ではこの順序を利用する。以下では，数論変換の原理を説明するために，$i = 0, 1, \cdots, N-1$ の単純な順序を用いる。上に示した $X^n + 1$ の分解により，次の（$\mathbb{Z}_q$ 加群としての）同型を得る。

$$R_q = \mathbb{Z}_q[X]/(X^n + 1) \simeq \bigoplus_{i=0}^{N-1} \mathbb{Z}_q[X]/\left(X^2 - \zeta^{2i+1}\right) =: T_q$$

具体的には，この同型は

$$\mathsf{NTT} : R_q \longrightarrow T_q, \quad f \longmapsto \widehat{f} := \left(f \bmod (X^2 - \zeta^{2i+1})\right)_{i=0}^{N-1} \tag{3.1}$$

で定まる。特に，T_q を **NTT 空間**，$\widehat{f} = \mathsf{NTT}(f) \in T_q$ を $f \in R_q$ の **NTT 表現**とよぶ。

■NTT 表現について　$f = f_0 + f_1 X + \cdots + X^{n-1} \in R_q$ の偶数と奇数の次数に関する多項式をそれぞれ

$$f_e := f_0 + f_2 Y + f_4 Y^2 + \cdots + f_{2N-2} Y^{N-1}, \quad f_o := f_1 + f_3 Y + f_5 Y^2 + \cdots + f_{2N-1} Y^{N-1} \in \mathbb{Z}_q[Y]$$

とおく。構成から $f = f_e(X^2) + f_o(X^2)X$ なので，各 $i = 0, 1, \ldots, N-1$ に対して，

$$\widehat{f}_{2i} := f_e(\zeta^{2i+1}) = \sum_{j=0}^{N-1} f_{2j} \zeta^{(2i+1)j}, \quad \widehat{f}_{2i+1} := f_o(\zeta^{2i+1}) = \sum_{j=0}^{N-1} f_{2j+1} \zeta^{(2i+1)j}$$

とおくと，

$$f \equiv \widehat{f}_{2i} + \widehat{f}_{2i+1} X \pmod{\left(X^2 - \zeta^{2i+1}\right)} \tag{3.2}$$

が成り立つ。これより，f の NTT 表現は $\widehat{f} = \left(\widehat{f}_{2i} + \widehat{f}_{2i+1} X\right)_{i=0}^{N-1} \in T_q$ とかける。

■NTT 表現の行列表示　$\mathbb{Z}_q$ の元を成分とする $N \times N$ 行列を

$$\mathbf{B} = A(\zeta) := \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \zeta^3 & \zeta^6 & \cdots & \zeta^{3(N-1)} \\ 1 & \zeta^5 & \zeta^{10} & \cdots & \zeta^{5(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta^{2N-1} & \zeta^{(2N-1)\cdot 2} & \cdots & \zeta^{(2N-1)(N-1)} \end{pmatrix} \in (\mathbb{Z}_q)^{N \times N}$$

とおく。R_q の元 $f = f_0 + f_1 X + \cdots + X^{n-1}$ の偶数と奇数の次数に関するそれぞれの係数ベクトル $(f_0, f_2, \ldots, f_{2N-2}), (f_1, f_3, \ldots, f_{2N-1}) \in \mathbb{Z}_q^N$ に対して

$$\begin{pmatrix} \widehat{f}_0 \\ \widehat{f}_2 \\ \widehat{f}_4 \\ \vdots \\ \widehat{f}_{2N-2} \end{pmatrix} = \begin{pmatrix} f_e(1) \\ f_e(\zeta^3) \\ f_e(\zeta^5) \\ \vdots \\ f_e(\zeta^{2N-1}) \end{pmatrix} = \mathbf{B} \begin{pmatrix} f_0 \\ f_2 \\ f_4 \\ \vdots \\ f_{2N-2} \end{pmatrix}, \quad \begin{pmatrix} \widehat{f}_1 \\ \widehat{f}_3 \\ \widehat{f}_5 \\ \vdots \\ \widehat{f}_{2N-1} \end{pmatrix} = \begin{pmatrix} f_o(1) \\ f_o(\zeta^3) \\ f_o(\zeta^5) \\ \vdots \\ f_o(\zeta^{2N-1}) \end{pmatrix} = \mathbf{B} \begin{pmatrix} f_1 \\ f_3 \\ f_5 \\ \vdots \\ f_{2N-1} \end{pmatrix}$$

が成り立つ。つまり，$f \in R_q$ の偶数と奇数の次数の係数ベクトルはそれぞれ $\mathbf{B}$ による線形変換（つまり，離散フーリエ変換）で $\widehat{f} \in T_q$ の偶数と奇数の添え字番号のベクトルに写る。$\mathbf{B}$ の逆行列は $\mathbf{C} := \frac{1}{N} A(\zeta^{-1})$ なので，式 (3.1) の NTT 写像の逆写像 NTT^{-1} は行列 $\mathbf{C}$ を用いて計算可能である（つまり，逆離散フーリエ変換から計算可能）。

■NTT 空間における乗算　R_q の 2 つの元 f, g に対して，その積を $h := f \cdot g \in R_q$ とおく。h の NTT 表現 $\widehat{h} \in T_q$ について，式 (3.2) から，各 $i = 0, 1, \ldots, N-1$ に対して

$$\widehat{h}_{2i} + \widehat{h}_{2i+1} X \equiv h = f \cdot g \equiv (\widehat{f}_{2i} + \widehat{f}_{2i+1} X)\,(\widehat{g}_{2i} + \widehat{g}_{2i+1} X) \bmod \left(X^2 - \zeta^{2i+1}\right)$$

が成り立つ。ここで，2 つの NTT 表現 $\widehat{f} = \left(\widehat{f}_{2i} + \widehat{f}_{2i+1} X\right)_{i=0}^{N-1}, \widehat{g} = (\widehat{g}_{2i} + \widehat{g}_{2i+1} X)_{i=0}^{N-1} \in T_q$ の積を

$$\begin{aligned}\widehat{f} \circ \widehat{g} &:= \left(\left(\widehat{f}_{2i} + \widehat{f}_{2i+1} X\right) \cdot (\widehat{g}_{2i} + \widehat{g}_{2i+1} X) \bmod \left(X^2 - \zeta^{2i+1}\right)\right)_{i=0}^{N-1} \\ &= \left(\widehat{f}_{2i}\widehat{g}_{2i} + \widehat{f}_{2i+1}\widehat{g}_{2i+1}\zeta^{2i+1} + \left(\widehat{f}_{2i}\widehat{g}_{2i+1} + \widehat{f}_{2i+1}\widehat{g}_{2i}\right) X\right)_{i=0}^{N-1} \in T_q\end{aligned}$$

と定めると，

$$\mathsf{NTT}(f \cdot g) = \mathsf{NTT}(f) \circ \mathsf{NTT}(g) \iff f \cdot g = \mathsf{NTT}^{-1}\left(\widehat{f} \circ \widehat{g}\right) \in R_q$$

が成り立つ（つまり，式 (3.1) の NTT 写像は環の同型写像である）。特に，NTT 空間 T_q における乗算は，成分ごとの演算であるため，（R_q における乗算に比べて）効率的に計算可能である。

3.3.1.2　ML-KEM の基本構成と処理概要

加群 $R_q^k \simeq (\mathbb{Z}_q^n)^k$ 上の LWE 問題に基づく ML-KEM は 2 つのステップで構成される。まず，R_q^k 上の LWE 問題から公開鍵暗号（K-PKE）を構成し，次に藤崎-岡本変換により IND-CCA2 KEM に変換する。

■K-PKE の処理概要　ここでは，K-PKE の処理概要とその原理が分かるように，簡略化した形で各アルゴリズムの処理を説明する。特に，処理の高速化のために，NTT 変換を適宜利用する。

K-PKE 鍵生成　鍵生成アルゴリズム（FIPS 203 の Algorithm 13, $\mathsf{K\text{-}PKE.KeyGen}(d)$）では，乱数 d を入力として，暗号鍵 $\mathtt{ek}_{\mathtt{PKE}}$ と復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ を次のように出力する。

- 入力：乱数 d
- 出力：暗号鍵 $\mathtt{ek}_{\mathtt{PKE}}$ と復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$
  1. $(\rho, \sigma) \leftarrow \mathsf{G}(d \| k)$：ハッシュ関数 G を用いて擬似ランダムな乱数の組 (ρ, σ) を生成。k は Module の階数で，各パラメータごとのドメインセパレーションのために追加されている。
  2. $\widehat{\mathbf{A}} = \left(\widehat{\mathbf{A}}[i,j]\right)_{0 \le i,j < k} \in (T_q)^{k \times k}$: 乱数 ρ を用いて，NTT 表現の公開鍵行列を生成
  3. $\mathbf{s} = (\mathbf{s}[i])_{0 \le i < k} \in R_q^k$：各 $\mathbf{s}[i] \in R_q$ のすべて $\mathbb{Z}_q$ 係数は中心二項分布 CBD_η からサンプルする
  4. $\mathbf{e} = (\mathbf{e}[i])_{0 \le i < k} \in R_q^k$：各 $\mathbf{e}[i] \in R_q$ のすべての $\mathbb{Z}_q$ 係数は CBD_η からサンプルする
  5. $\widehat{\mathbf{s}} = (\mathsf{NTT}(\mathbf{s}[i]))_{0 \le i < k} \in T_q^k$：各 $\mathbf{s}[i]$ を NTT 変換
  6. $\widehat{\mathbf{e}} = (\mathsf{NTT}(\mathbf{e}[i]))_{0 \le i < k} \in T_q^k$：前のステップ同様，各 $\mathbf{e}[i]$ を NTT 変換
  7. $\widehat{\mathbf{t}} = \widehat{\mathbf{A}} \circ \widehat{\mathbf{s}} + \widehat{\mathbf{e}} = \left(\sum_{j=0}^{k-1} \widehat{\mathbf{A}}[i,j] \circ \widehat{\mathbf{s}}[j] + \widehat{\mathbf{e}}[i]\right)_{0 \le i < k} \in T_q^k$：NTT 空間上で LWE 関係式を生成
  8. $\mathtt{ek}_{\mathtt{PKE}} = \left(\widehat{\mathbf{t}}, \rho\right), \mathtt{dk}_{\mathtt{PKE}} = \widehat{\mathbf{s}}$（公開鍵行列 $\widehat{\mathbf{A}}$ は ρ から復元可能であることに注意）
  9. $(\mathtt{ek}_{\mathtt{PKE}}, \mathtt{dk}_{\mathtt{PKE}})$ を出力

ステップ 2 において，NTT 表現の公開鍵行列の各成分 $\widehat{\mathbf{A}}[i,j]$ は，入力する乱数から擬似ランダムな T_q の元を出力する $\mathsf{SampleNTT}$ 関数（FIPS 203 の Algorithm 7）を用いて生成する（具体的には，$\widehat{\mathbf{A}}[i,j] \leftarrow \mathsf{SampleNTT}(\rho \| i \| j)$ と生成）。ステップ 3, 4 において $\mathbf{s}[i]$，$\mathbf{e}[i]$ を $\mathsf{SamplePolyCBD}$ 関数（FIPS 203 の Algorithm 8）を用いて生成する。この関数の中では R_q の元の各係数を，Z_q 上の二項分布を出力する関数 CBD_η の出力として設定する。このとき，関数はステップ 1 で生成した σ を乱数シードとし，以下の順序で実行される。

(i) $(x_1, \ldots, x_\eta, y_1, \ldots, y_\eta) \in \{0,1\}^{2\eta}$ を一様ランダムにサンプルする
(ii) $\sum_{i=1}^{\eta}(x_i - y_i) \bmod q \in \mathbb{Z}_q$ を出力

ステップ 8 において，FIPS 203 では $\left(\widehat{\mathbf{t}}, \rho\right)$ と $\widehat{\mathbf{s}}$ をそれぞれ符号化関数 ByteEncode（FIPS 203 の Algorithm 5）で符号化したものを暗号鍵 $\mathtt{ek}_{\mathtt{PKE}}$ と復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ とする。

鍵生成アルゴリズムにおいて，ρ から NTT 表現の公開鍵行列 $\widehat{\mathbf{A}}$ が復元可能なので，暗号鍵 $\mathtt{ek}_{\mathtt{PKE}}$ は NTT 表現の LWE インスタンスの組 $\left(\widehat{\mathbf{A}}, \widehat{\mathbf{t}}\right)$ と等価なデータとなる。特に，$\mathbf{t} := \mathsf{NTT}^{-1}\left(\widehat{\mathbf{t}}\right) \in R_q^k, \mathbf{A} := \mathsf{NTT}^{-1}\left(\widehat{\mathbf{A}}\right) \in (R_q)^{k\times k}$ とおくと，R_q^k 上の LWE 関係式 $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}$ が成り立つ。一方，復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ は NTT 表現の LWE の秘密 $\widehat{\mathbf{s}}$ であるので，暗号鍵から復号鍵を見つけるのは $T_q^k \simeq R_q^k$ 上の探索 LWE 問題である。特に，適切な暗号パラメータ（後述の 3.3.1.3 節を参照）を利用した場合，その LWE 問題を解くのは計算量的に非常に困難である。また，鍵生成アルゴリズムにおいて，NTT 空間上で公開鍵行列 $\widehat{\mathbf{A}}$ を直接生成すると共に，ステップ 7 で NTT 空間上で LWE 関係式を生成することで，計算の高速化を図る。

K-PKE 暗号化　暗号化アルゴリズム（FIPS 203 の Algorithm 14, K-PKE.Encrypt）では，暗号化鍵 $\mathtt{ek}_{\mathtt{PKE}}$，平文 m と乱数 r を入力として，次のように暗号文 c を出力する。

- 入力：暗号化鍵 $\mathtt{ek}_{\mathtt{PKE}} = \left(\widehat{\mathbf{t}}, \rho\right)$，平文 m と乱数 r
- 出力：暗号文 c
  1. ρ から NTT 表現の公開鍵行列 $\widehat{\mathbf{A}} \in (T_q)^{k\times k}$ を復元
  2. $\mathbf{y} = (\mathbf{y}[i])_{0\le i<k} \in R_q^k$：各 $\mathbf{y}[i] \in R_q$ のすべての $\mathbb{Z}_q$ 係数は中心二項分布 CBD_η からサンプルする
  3. $\mathbf{e}_1 = (\mathbf{e}_1[i])_{0\le i<k} \in R_q^k$：各 $\mathbf{e}[i] \in R_q$ のすべての $\mathbb{Z}_q$ 係数は CBD_η からサンプルする
  4. $e_2 \in R_q$：すべての $\mathbb{Z}_q$ 係数は CBD_η からサンプルする
  5. $\widehat{\mathbf{y}} = (\mathsf{NTT}\,(\mathbf{y}[i]))_{0\le i<k} \in T_q^k$
  6. $\mathbf{u} = \mathsf{NTT}^{-1}\left(\widehat{\mathbf{A}}^\top \circ \widehat{\mathbf{y}}\right) + \mathbf{e}_1 = \mathbf{A}^\top\mathbf{y} + \mathbf{e}_1 = \left(\sum_{j=0}^{k-1}\mathbf{A}[j,i]\mathbf{y}[j] + \mathbf{e}_1[i]\right)_{0\le i<k} \in R_q^k$
     （ただし，$\mathbf{A} = \mathsf{NTT}^{-1}(\widehat{\mathbf{A}}) = (\mathbf{A}[i,j])_{0\le i,j<k} \in (R_q)^{k\times k}$ とする）
  7. $\mu = \mathsf{Decompress}\,(\mathsf{ByteDecode}(m)) \in R_q$：平文 m をビット列化した後に R_q の元に変換
  8. $v = \mathsf{NTT}^{-1}\left(\widehat{\mathbf{t}}^\top \circ \widehat{\mathbf{y}}\right) + e_2 + \mu = \mathbf{t}^\top\mathbf{y} + e_2 + \mu \in R_q$
  9. $c = (\mathbf{u}, v) \in R_q^k \times R_q$ を出力

ステップ 2, 3, 4 において，r をシードとした擬似乱数を引数とした SamplePolyCBD 関数で，すべての $\mathbb{Z}_q$ 係数が十分小さい多項式を生成する。ステップ 7 では，バイト列で表現された平文 m を ByteDecode 関数（FIPS 203, Algorithm 6）でビット列 $(m_0, m_1, \ldots, m_{n-1})$ に変換した後に，各ビット $m_i \in \{0,1\}$ を Decompress 関数で $\mu_i := \left\lceil \frac{q}{2} \cdot m_i \right\rfloor \in \mathbb{Z}_q$ に変換する。また，各 μ_i を係数とする多項式を $\mu = \mu_0 + \mu_1 x + \cdots + \mu_{n-1}x^{n-1} \in R_q$ とする。ステップ 9 において，FIPS 203 では $\mathbf{u}$ と v はそれぞれ Compress 関数で圧縮した後，ByteEncode 関数で符号化する。

暗号文は $c = (\mathbf{u}, v) = \left(\mathbf{A}^\top\mathbf{y} + \mathbf{e}_1, \mathbf{t}^\top\mathbf{y} + e_2 + \mu\right) \in R_q^k \times R_q$ の形で，LWE に基づく Lindner-Peikert による暗号方式と同様，R_q^k 上の LWE 問題が計算困難であれば，暗号文から μ（つまり，平文 m）の情報が洩れない。また，ステップ 6 と 8 において，NTT 空間上で $\mathbf{A}^\top\mathbf{y}$ と $\mathbf{t}^\top\mathbf{y}$ を計算することで，計算の高速化を図る。

K-PKE 復号　復号アルゴリズム（FIPS 203 の Algorithm 15, K-PKE.Decrypt）では，復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ と暗号文 c を入力とし，次のように復号文 m' を出力する。

- 入力：復号鍵 $\mathtt{dk}_{\mathtt{PKE}} = \widehat{\mathbf{s}}$ と暗号文 $c = (\mathbf{u}, v)$
- 出力：復号文 m'
  1. $w = v - \mathsf{NTT}^{-1}\left(\widehat{\mathbf{s}}^\top \circ \mathsf{NTT}(\mathbf{u})\right) = v - \mathbf{s}^\top\mathbf{u} \in R_q$
  2. $m' = \mathsf{ByteEncode}\,(\mathsf{Compress}(w))$ を出力

ステップ 2 において，多項式表現の R_q の元 $w = w_0 + w_1 x + \cdots + w_{n-1}x^{n-1}$ に対して，各係数 $w_i \in \mathbb{Z}_q$ を Compress 関数で $z_i := \left\lceil \frac{2}{q} \cdot w_i \right\rfloor \bmod 2 \in \{0,1\}$ に変換する。また，ビット列 $(z_0, z_1, \ldots, z_{n-1})$ を ByteEncode 関数（FIPS 203, Algorithm 5）でバイト列に変換する。特に，ByteEncode 関数と ByteDecode 関数は互いの逆関数である。

復号アルゴリズムにおいて，暗号文 $c = (\mathbf{u}, v) = \left(\mathbf{A}^\top \mathbf{y} + \mathbf{e}_1, \mathbf{t}^\top \mathbf{y} + e_2 + \mu\right) \in R_q^k \times R_q$ に対して，R_q^k 上の LWE 関係式 $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}$ から，

$$\begin{aligned} w = v - \mathbf{s}^\top \mathbf{u} &= \mathbf{t}^\top \mathbf{y} + e_2 + \mu - (\mathbf{A}\mathbf{s})^\top \mathbf{y} - \mathbf{s}^\top \mathbf{e}_1 \\ &= \mathbf{t}^\top \mathbf{y} + e_2 + \mu - \left(\mathbf{t}^\top + \mathbf{e}^\top\right)\mathbf{y} - \mathbf{s}^\top \mathbf{e}_1 = \mu + \underbrace{e_2 - \mathbf{e}^\top \mathbf{y} - \mathbf{s}^\top \mathbf{e}_1}_{\text{すべての } \mathbb{Z}_q \text{ 係数が十分小さい}} \in R_q \end{aligned}$$

が成り立つ。ここで，$\mathbf{s}, \mathbf{e}, \mathbf{e}_1, \mathbf{y} \in R_q^k$ の各成分 $\mathbf{s}[i], \mathbf{e}[i], \mathbf{e}_1[i], \mathbf{y}[i] \in R_q$ と $\mu \in R_q$ のすべての $\mathbb{Z}_q$ 係数は十分小さいことに注意する。よって，Compress 関数による各 $\mathbb{Z}_q$ 係数におけるノイズ補正により

$$\mathsf{Compress}(w) = \mathsf{Compress}(\mu) = (m_0, m_1, \ldots, m_{n-1}) \in \{0,1\}^{n-1}$$

が成り立つ。最後に，ByteEncode 関数により，平文のビット列 $(m_0, m_1, \ldots, m_{n-1})$ をバイト列に変換することで，元の平文 m に復号できる（つまり，復号文 m' は平文 m に一致する）。また，ステップ 1 において，NTT 空間上で $\mathbf{s}^\top \mathbf{u}$ を計算することで，計算の高速化を図る。

■ML-KEM の処理概要　K-PKE 方式を用いて，ML-KEM を以下のように構成する。

ML-KEM 鍵生成　鍵生成アルゴリズム（FIPS 203, Algorithm 16）では，K-PKE 鍵生成アルゴリズムを用いて，2 つの乱数 d, z から鍵カプセル化鍵 $\mathtt{ek}$ とデカプセル化鍵 $\mathtt{dk}$ を次のように出力する。

- 入力：2 つの乱数 d, z
- 出力：鍵カプセル化鍵 $\mathtt{ek}$ とデカプセル化鍵 $\mathtt{dk}$
  1. K-PKE 鍵生成アルゴリズムで，乱数 d から $(\mathtt{ek}_{\mathtt{PKE}}, \mathtt{dk}_{\mathtt{PKE}})$ を生成
  2. $\mathtt{ek} = \mathtt{ek}_{\mathtt{PKE}}$
  3. $\mathtt{dk} = (\mathtt{dk}_{\mathtt{PKE}}, \mathtt{ek}, \mathsf{H}(\mathtt{ek}), z)$：$\mathsf{H}$ はハッシュ関数
  4. $(\mathtt{ek}, \mathtt{dk})$ を出力

ML-KEM 鍵カプセル化　鍵カプセル化アルゴリズム（FIPS 203, Algorithm 17）では，K-PKE 暗号化アルゴリズムを用いて，鍵カプセル化鍵 $\mathtt{ek}$ と乱数 m から共有の秘密鍵 K と暗号文 c を次のように出力する。

- 入力：鍵カプセル化鍵 $\mathtt{ek}$ と乱数 m
- 出力：共有の秘密鍵 K と暗号文 c
  1. $(K, r) = \mathsf{G}\left(m \| \mathsf{H}(\mathtt{ek})\right)$：$\mathsf{G}$ はハッシュ関数
  2. K-PKE 暗号化アルゴリズムで，$(\mathtt{ek}, m, r)$ から暗号文 c を生成
  3. (K, c) を出力

ML-KEM デカプセル化　デカプセル化アルゴリズム (FIPS 203, Algorithm 18) では，K-PKE 復号アルゴリズムを用いて，デカプセル化 $\mathtt{dk}$ と暗号文 c から，共有の秘密鍵 K を次のように出力する。また，c が改竄されていないことを保証するために，K-PKE 暗号化アルゴリズムで復号文から暗号文 c' を生成し，c と c' が一致するか検証する。

- 入力：デカプセル化 $\mathtt{dk} = (\mathtt{dk}_{\mathtt{PKE}}, \mathtt{ek}, \mathsf{H}(\mathtt{ek}), z)$ と暗号文 c
- 出力：共有の秘密鍵 K
  1. K-PKE 復号アルゴリズムで，復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ と暗号文 c から，復号文 m' を生成
  2. $(K', r') = \mathsf{G}\left(m' \| \mathsf{H}(\mathtt{ek})\right)$

3. $\bar{K} = \mathsf{J}(z\|c)$：J はハッシュ関数
4. K-PKE 暗号化アルゴリズムで，$(\mathtt{ek}, m', r')$ から暗号文 c' を生成
5. $c \neq c'$ の場合は，$K' = \bar{K}$ とおく
6. K' を出力

3.3.1.3 暗号パラメータ

ML-KEM における主な暗号パラメータと対応する鍵や暗号文のサイズと安全性レベルは以下である。具体的には，LWE の次元 $n = 256$ と剰余素数 $q = 3329$ は ML-KEM-512, -768, -1024 の 3 種類の暗号パラメータで共通であるが，主に 3 種類の階数パラメータ $k \in \{2, 3, 4\}$ により安全性レベルが異なる。(ML-KEM のパラメータ名は，$n \times k \in \{512, 768, 1024\}$ の値により名づけられている。)

表 3.2: ML-KEM の暗号パラメータ

	暗号パラメータ			サイズ（単位：Bytes）				安全性
	n	q	k	カプセル化鍵	デカプセル化鍵	暗号文	共有の秘密鍵	レベル
ML-KEM-512	256	3329	2	800	1,632	768	32	レベル 1
ML-KEM-768	256	3329	3	1,184	2,400	1,088	32	レベル 3
ML-KEM-1024	256	3329	4	1,568	3,168	1,568	32	レベル 5

3.3.1.4 CRYSTALS-Kyber との違い

- Kyber の第 3 ラウンド提出版では，共有する秘密鍵は長さが可変な値として扱われていた。一方，ML-KEM では，その長さは 256 ビットに固定している。また，その鍵は直接共通鍵として利用できる。
- ML-KEM.Encaps と ML-KEM.Decaps のアルゴリズムでは，第 3 ラウンド仕様とは異なる藤崎-岡本変換を利用する。具体的には，ML-KEM.Encaps は共有する秘密の導出において暗号文のハッシュ値を含まず，ML-KEM.Decaps ではその変更に合わせている。
- 第 3 ラウンドの仕様では，ML-KEM.Ecaps アルゴリズム内の初期乱数 m は使う前にハッシュ化される。具体的には，アルゴリズム 16 の 1 と 2 行目の間に，$m \leftarrow H(m)$ のステップがあったが，ML-KEM ではその処理は不必要で行わない。
- ML-KEM では, 第 3 ラウンドの仕様にはなかった入力データの検証ステップを含む。例えば, ML-KEM.Encaps では，カプセル化キーを含むバイト配列が，モジュラー還元なしで q を法とする整数配列に正しくデコードされることを必要とする。

3.3.2 FIPS 204: Module-Lattice-Based Digital Signature Standard（ML-DSA）

ML-DSA [119] は CRYSTALS-Dilithium に基づく署名方式である。ML-KEM と同じように，2 のべき数 $n = 256$ に対し $R := \mathbb{Z}[X]/(X^n + 1)$ を基本環とし，素数 $q = 8380417$ に対し $R_q := R/qR = \mathbb{Z}_q[X]/(X^n + 1)$ をその剰余環とする。階数パラメータ $k \in \{2, 3, 4\}$ に対し，ML-DSA の安全性は $\mathbb{Z}_q$ 加群 $R_q^k \simeq (\mathbb{Z}_q^n)^k$ 上の Module-LWE 問題の計算困難性に依存する。また，ML-KEM と同様に，R_q における乗算を高速化するために，NTT を利用する。ここでは，主に ML-DSA の構成と処理概要について説明する。

3.3.2.1 ML-DSA における数論変換

ML-DSA では, 2 のべき数 $n = 2^8 = 256$ と素数 $q = 2^{23} - 2^{13} + 1 = 8380417$ で定まる剰余環 $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ を用いる（ML-DSA の暗号パラメータについては, 後述の 3.3.2.3 節を参照)。これらの暗号パラメータの組 (n, q) において, $\mathbb{Z}_q^\times$ は位数 $q - 1 = 2^{13} \cdot 1023$ の巡回群である。ML-DSA では, $\mathbb{Z}_q$ における 1 の原始 512 乗根 $\zeta := 1753 \bmod q$ をとる。このとき, 多項式環 $\mathbb{Z}_q[X]$ において, $X^n + 1$ は次のように n 個の 1 次式の積に分解できる。

$$X^n + 1 = \prod_{i=0}^{n-1} \left(X - \zeta^{2i+1}\right) = \prod_{i=0}^{n-1} \left(X - \zeta^{2\mathsf{BitRev}_8(i)+1}\right) \in \mathbb{Z}_q[X]$$

ただし, $\mathsf{BitRev}_8(i)$ は符号なし 8 ビット整数 i のビット逆順整数とし, ML-DSA ではこの順序を利用する。具体的には, 各 $i = 0, 1, \ldots, n-1$ に対し $\zeta_i := \zeta^{2\mathsf{BitRev}_8(i)+1}$ とおき, 環としての同型

$$\mathsf{NTT} : R_q \simeq \bigoplus_{i=0}^{n-1} \mathbb{Z}_q[X]/(X - \zeta_i) \simeq \bigoplus_{i=0}^{n-1} \mathbb{Z}_q =: T_q, \quad f \mapsto \widehat{f} := (f(\zeta_0), f(\zeta_1), \ldots, f(\zeta_{n-1}))$$

を用いて, R_q における乗算を効率的に行う。

3.3.2.2 ML-DSA の構成と処理概要

加群 $R_q^k \simeq (\mathbb{Z}_q^n)^k$ 上の LWE 問題に基づく ML-DSA は, 以下に示すアルゴリズム群で構成される。ただし, ML-DSA の処理概要とその原理が分かるように, 簡略化した形で各アルゴリズムの処理を説明する。

■ML-DSA **鍵生成**　鍵生成アルゴリズム（FIPS 204, Algorithm 6）では, 乱数 ξ を入力として, 公開鍵 pk と秘密鍵 sk を次のように出力する（ただし, ℓ は次元パラメータとする)。

- 入力：乱数 ξ
- 出力：公開鍵 pk と秘密鍵 sk
  1. $(\rho, \rho', K) = \mathsf{H}(\xi)$：ハッシュ関数 H で乱数 ξ から 3 つのデータの組 (ρ, ρ', K) を一意的に生成
  2. $\widehat{\mathbf{A}} = \mathsf{ExpandA}(\rho) \in (T_q)^{k \times \ell}$：擬似ランダムな行列 $\mathbf{A} \in (R_q)^{k \times \ell}$ を生成し, その NTT 表現を $\widehat{\mathbf{A}}$ を計算
  3. $(\mathbf{s}_1, \mathbf{s}_2) = \mathsf{ExpandS}(\rho') \in S_\eta^\ell \times S_\eta^k$：$S_\eta$ はすべての係数が $[-\eta, \eta]$ 内の R の元全体の集合（例：$\eta \in \{2, 4\}$）
  4. $\mathbf{t} = \mathsf{NTT}^{-1}\left(\widehat{\mathbf{A}} \circ \mathsf{NTT}(\mathbf{s}_1)\right) + \mathbf{s}_2 \in R_q^k \left(= \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2\right)$
  5. $(\mathbf{t}_1, \mathbf{t}_0) = \mathsf{Power2Round}(\mathbf{t}) \in R_q^k \times R_q^k$：$\mathbf{t} \in R_q^k$ を上位と下位ビットに分割
  6. $\mathtt{pk} = (\rho, \mathbf{t}_1)$ とおき, そのハッシュ値 $tr = \mathsf{H}(\mathtt{pk})$ を計算
  7. pk と $\mathtt{sk} = (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$ を出力

ステップ 2 において, ExpandA 関数（FIPS 204, Algorithm 32）は, 乱数シード ρ から擬似ランダムな $\mathbf{A}$ を生成し, その NTT 表現 $\widehat{A}$ を計算する。ステップ 3 において, ExpandS 関数（FIPS 204, Algorithm 33）は, 棄却サンプリングを用いてある範囲 $[-\eta, \eta]$ 内の係数をもつ R の元の組を生成する（$\eta \in \{2, 4\}$)。ステップ 5 において, Power2Round 関数（FIPS 204, Algorithm 35）を用いて, $\mathbf{t} \in R_q^k$ の各成分のすべての係数を上位と下位のビットに分割する。

鍵生成アルゴリズムにおいて, 本質的に公開鍵 pk は $(\mathbf{A}, \mathbf{t})$ に対応する。公開鍵に関する付属情報をいくつか含むが秘密鍵 sk は $(\mathbf{s}_1, \mathbf{s}_2)$ に対応する。公開鍵と秘密鍵の間に, R_q^k 上の LWE 関係式 $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$ が成り立つ。これより, 公開鍵から秘密鍵を見つけるのは R_q^k 上の探索 LWE 問題となり, 適切な暗号パラメータ（後述の 3.3.2.3 節を参照）を利用した場合, その LWE 問題を解くのは計算量的に非常に困難である。

■ML-DSA **署名生成**　署名生成アルゴリズム（FIPS 204, Algorithm 7）では, 秘密鍵 sk と平文 M' を入力として, 平文に対応する署名 σ を次のように出力する。

- 入力：秘密鍵 $\mathtt{sk} = (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$ と平文 M'
- 出力：署名 σ
  1. $\widehat{\mathbf{s}}_1 = \mathsf{NTT}(\mathbf{s}_1) \in T_q^\ell, \widehat{\mathbf{s}}_2 = \mathsf{NTT}(\mathbf{s}_2) \in T_q^k, \widehat{\mathbf{t}}_0 = \mathsf{NTT}(\mathbf{t}_0) \in T_q^k$：NTT 表現を計算
  2. $\widehat{\mathbf{A}} = \mathsf{ExpandA}(\rho) \in (T_q)^{k \times \ell}$：$\rho$ から $\widehat{\mathbf{A}}$ を復元
  3. $\mu = \mathsf{H}(tr \| M')$：秘密鍵の一部 tr と平文 M' から定まるハッシュ値
  4. 次を繰り返す：
     (a) $\mathbf{y} = (\mathbf{y}[i])_{i=0}^{\ell} \in R_q^\ell$：各 $\mathbf{y}[i] \in R_q$ の各 $\mathbb{Z}_q$ 係数をある小さい範囲で擬似ランダムにサンプル
     (b) $\mathbf{w} = \mathsf{NTT}^{-1}\left(\widehat{\mathbf{A}} \circ \mathsf{NTT}(\mathbf{y})\right) = \mathbf{A}\mathbf{y} \in R_q^k$：NTT 変換を利用
     (c) $\mathbf{w}_1 = \mathsf{HighBits}(\mathbf{w}) \in R_q^k$：$\mathbf{w}$ の各成分の上位ビット
     (d) $\tilde{c} = \mathsf{H}(\mu \| \mathbf{w}_1)$
     (e) $c = \mathsf{SampleInBall}(\tilde{c}) \in R_q$：各係数を $\{-1, 0, 1\}$ からサンプルする（十分小さい）
     (f) $\widehat{c} = \mathsf{NTT}(c) \in T_q$
     (g) $c\mathbf{s}_1 = \mathsf{NTT}^{-1}(\widehat{c} \circ \widehat{\mathbf{s}}_1) \in R_q^\ell, c\mathbf{s}_2 = \mathsf{NTT}^{-1}(\widehat{c} \circ \widehat{\mathbf{s}}_2) \in R_q^k$：NTT 空間の乗算を利用
     (h) $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1 \in R_q^\ell$
     (i) $\mathbf{r}_0 = \mathsf{LowBits}(\mathbf{w} - c\mathbf{s}_2) \in R_q^k$：$\mathbf{w} - c\mathbf{s}_2$ の各成分の下位ビット
     (j) $\mathbf{z}$ と $\mathbf{r}_0$ のすべての $\mathbb{Z}_q$ 係数が十分小さい場合，次の処理を行う：
        i. $c\mathbf{t}_0 = \mathrm{NTT}^{-1}(\widehat{c} \circ \widehat{\mathbf{t}}_0) \in R_q^k$：NTT 空間の乗算を利用
        ii. $\mathbf{h} = \mathsf{MakeHint}\left(-c\mathbf{t}_0, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0\right)$：長さ k の不一致真理値ベクトル
        $c\mathbf{t}_0$ のすべての $\mathbb{Z}_q$ 係数が十分小さく，かつ $\mathbf{h}$ 内の 1 の個数が十分少ないとき，ステップ 5 に進む
  5. $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$ を出力

ステップ 4 (e) において，乱数 $\tilde{c}$ を引数とする SampleInBall 関数（FIPS 204, Algorithm 29）で，すべての $\mathbb{Z}_q$ 係数を $\{-1, 0, 1\}$ からサンプルした多項式 $c \in R_q$ を生成する（ただし，係数ベクトルのハミング重みは 64 以下）。ステップ 4 (j) ii において，MakeHint 関数（FIPS 204, Algorithm 39）は，$\mathsf{HighBits}(\mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0)$ と $\mathsf{HighBits}(\mathbf{w} - c\mathbf{s}_2)$ の $\mathbb{Z}_q$ 係数の不一致真理値による長さ k のベクトル $\mathbf{h}$ を計算する。次の署名検証時で $\mathbf{w}_1$ を復元するために $\mathbf{h}$ を用いる。

署名生成アルゴリズムにおいて，ステップ 4 が主処理で，すべての $\mathbb{Z}_q$ 係数が十分小さい $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1 \in R_q^\ell$ を見つけるまで $\mathbf{y} \in R_q^\ell$ を取り直す。具体的には，擬似ランダムにサンプルしたすべての $\mathbb{Z}_q$ 係数が十分小さい $\mathbf{y} \in R_q^\ell$ から，コミットメント $\mathbf{w}_1$ を生成し，$\mathbf{w}_1$ と μ から定まるハッシュ値であるチャレンジ $\tilde{c}$ を求める。また，レスポンスとして，すべての $\mathbb{Z}_q$ 係数が十分小さい $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$ を生成する。チャレンジ $\tilde{c}$，レスポンス $\mathbf{z}$，コミットメント $\mathbf{w}_1$ のヒント $\mathbf{h}$ の 3 つの組 $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$ を平文に対応する署名とする。

■ML-DSA 署名検証　署名検証アルゴリズム（FIPS 204, Algorithm 8）では，公開鍵 $\mathtt{pk} = (\rho, \mathbf{t}_1)$ と署名 $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$ 付きの平文 M' を入力として，署名検証の結果を次のように真偽値で出力する。

- 入力：公開鍵 $\mathtt{pk} = (\rho, \mathbf{t}_1)$，署名 $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$ 付きの平文 M'
- 出力：真偽値
  1. $\widehat{\mathbf{A}} = \mathsf{ExpandA}(\rho)$：$\rho$ から $\widehat{\mathbf{A}}$ を復元
  2. $tr = \mathsf{H}(\mathtt{pk})$：pk のハッシュ値
  3. $\mu = \mathsf{H}(tr \| M')$：秘密鍵の一部 tr と平文 M' から定まるハッシュ値
  4. $c' = \mathsf{SampleInBall}(\tilde{c}) \in R_q$：各係数を $\{-1, 0, 1\}$ からサンプルする
  5. $\mathbf{w}'_{\mathrm{Approx}} = \mathsf{NTT}^{-1}\left(\widehat{\mathbf{A}} \circ \mathsf{NTT}(\mathbf{z}) - \mathsf{NTT}(c') \circ \mathsf{NTT}(\mathbf{t}_1 \cdot 2^d)\right) = \mathbf{A}\mathbf{z} - c'\mathbf{t}_1 \cdot 2^d \in R_q^\ell$
     （ただし，d は上位と下位ビットを分割する閾値）
  6. $\mathbf{w}'_1 = \mathsf{UseHint}(\mathbf{h}, \mathbf{w}'_{\mathrm{Approx}})$：署名生成時のコミットメントを復元

7. $\tilde{c}' = \mathsf{H}(\mu \| \mathbf{w}_1')$：$\mu$ と $\mathbf{w}_1'$ から定まるハッシュ値
8. $\mathbf{z}$ のすべての $\mathbb{Z}_q$ 係数が十分小さく，かつ $\tilde{c} = \tilde{c}'$ のとき署名を受理し，それ以外は棄却とする。

ステップ 6 において，UseHint 関数（FIPS 204, Algorithm 40）で，$\mathbf{w}'_{\mathrm{Approx}}$ が $\mathbf{w}$ に十分近いとき，ヒント $\mathbf{h}$ を元に署名生成時のコミットメント $\mathbf{w}_1$ を復元する（つまり，$\mathbf{w}_1' = \mathbf{w}_1$）。具体的には，$\sigma$ が正当な署名であれば，$c' = c$ で $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$ なので，LWE 関係式 $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$ と $\mathbf{t}_1 \cdot 2^d \approx \mathbf{t}$（$\mathbf{t}_1$ は $\mathbf{t}$ の上位ビット）より

$$\begin{aligned}\mathbf{w}'_{\mathrm{Approx}} = \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d &= \mathbf{A}\mathbf{y} + c\mathbf{A}\mathbf{s}_1 - c\mathbf{t}_1 \cdot 2^d \\ &= \mathbf{w} + c(\mathbf{t} - \mathbf{s}_2) - c\mathbf{t}_1 \cdot 2^d \approx \mathbf{w} - c\mathbf{s}_2 \approx \mathbf{w}\end{aligned}$$

が成り立つ（$c\mathbf{s}_2 \in R_q^k$ のすべての $\mathbb{Z}_q$ 係数は十分小さいことに注意）。このとき，ステップ 7 で $\tilde{c}' = \tilde{c}$ となり検証に成功する。一方，平文 M' が改竄または署名 σ が偽造された場合は，非常に高い確率で $\tilde{c} \neq \tilde{c}'$ となり，検証に失敗する。

3.3.2.3 暗号パラメータ

ML-DSA における主な暗号パラメータと対応する鍵や署名のサイズと安全性レベルは以下である。具体的には，LWE の次元 $n = 256$ と剰余素数 $q = 8380417$ は ML-DSA-44, -65, -87 の 3 種類の暗号パラメータで共通であるが，主に公開鍵行列 $\mathbf{A} \in (R_q)^{k \times \ell}$ のサイズ (k, ℓ) により安全性レベルが異なる（特に，ML-DSA のパラメータ名は，(k, ℓ) により名づけられている）。

表 3.3: ML-DSA の暗号パラメータ

	暗号パラメータ			サイズ（単位：バイト）			安全性
	n	q	(k, ℓ)	秘密鍵	公開鍵	署名	レベル
ML-DSA-44	256	8380417	(4, 4)	2,560	1,312	2,420	レベル 2
ML-DSA-65	256	8380417	(6, 5)	4,032	1,952	3,309	レベル 3
ML-DSA-87	256	8380417	(8, 7)	4,896	2,592	4,627	レベル 5

3.3.2.4 CRYSTALS-Dilithium との違い

- CRYSTALS-Dilithum の version 3.1 と第 3 ラウンド提出版との違いは，安全性を確保するために，署名アルゴリズム内の秘密ランダムシード ρ' とメッセージ表現 μ の長さが 384 から 512bits への増大である。加えて，公開鍵のハッシュに関する変数 tr のサイズを 384 から 256 ビットに減少させる一方，鍵生成において変数 ζ を ρ' に再ラベル付けし，そのサイズを 256 から 512bits 増大させている。
- ML-DSA と CRYSTALS-Dilithum の version 3.1 との違いについて，ML-DSA では tr の長さを 512bits に増やし，ML-DSA-65 と ML-DSA-87 のパラメータ設定それぞれで $\tilde{c}$ の長さを 384 と 512bits に増大している。CRYSTALS-Dilithum version 3.1 では，デフォルトの署名アルゴリズムは署名者の秘密鍵とメッセージから疑似ランダム生成された ρ' について確定的で，optional version では ρ' は 512bits のランダム列としてサンプリングされる。一方，ML-DSA では，ρ' は署名者の秘密鍵，メッセージ，と Approved RBG[*6]から生成された 256bits の文字列 rnd から生成される。また，ML-DSA 標準では，rnd が 256bits の定数文字列である optional deteministic version を許可している。

[*6] NIST SP 800-90 シリーズ [26, 151, 27] で規定されたランダムビット生成器 (Random Bit Generator: RBG) を指す。

3.3.3 CRYSTALS-Kyber

歴史: CRYSTALS-Kyber は NIST PQC 標準化プロジェクトへの応募方式の一つとして 2017 年 11 月に Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancréde Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, Damien Stehlé の 10 名により共同で発表され [17]，その後 2018 年 4 月の国際会議 Euro S&P に Roberto Avanzi を除いた 9 名の共著により査読付き論文として発表された [34]。NIST PQC 標準化プロジェクトの第 3 ラウンドからは Jintai Ding が加わり 11 名での提案となった。NIST の耐量子計算機暗号標準化において唯一暗号化・鍵交換目的での Selected Algorithm として残った方式である [144]。

NIST PQC 標準化プロジェクトのラウンドが進むごとに主に暗号化処理のパラメータに関して修正が行われ，現在の最新版は 2021 年 8 月に公開されたバージョン 3.02[19] である。以下の記述はこの仕様書に従う。

参照 URL: 開発者による公式ページ `https://pq-crystals.org/kyber/` および GitHub のリファレンスコード `https://github.com/pq-crystals/kyber` を参照した。

設計原理: CRYSTALS-Kyber は Module-LWE 問題を安全性の根拠とする公開鍵暗号方式であり，dual-LWE 暗号方式をひな型[*7]として $x^{256}+1$ を定義多項式とした環上で処理を行うことで効率化している。

ベースとして IND-CPA 安全な公開鍵暗号を構成し，それを藤崎-岡本変換のデカプセル化失敗時の戻り値を調整した Hofheinz らの変種 [89] により IND-CCA2 安全な KEM へと変換している。

アルゴリズムの詳細: 表 3.5，3.6，3.7 に Lindner-Peikert[101] による格子ベース公開鍵暗号と CRYSTALS-Kyber の鍵生成，暗号化，復号アルゴリズムを並置する。

パブリックパラメータは以下で与えられる。

- n, q: 環を定義するための多項式 x^n+1 の次数と法を示す。用いられる多項式環は $R := \mathbb{Z}[x]/(x^n+1), R_q := \mathbb{Z}_q[x]/(x^n+1)$ であり，常に $n=256, q=3329=2^8\cdot 13+1$ と固定されている[*8]。
- k: Module 格子のランクとする。
- η_1, η_2: 鍵生成および暗号化時に生成するノイズベクトルの大きさを指定する。
- d_u, d_v: 暗号文多項式 $(\boldsymbol{u}, v) \in R_q^k \times R_q$ を表現するためのビット数を指定する。

用いられるサブルーチンのうち主なものを以下に列挙する。

- $\mathsf{NTT}(f)$ は $f=\sum_{i=0}^{255} f_i x^i \in R_q$ の NTT 表現 $\hat{f}=\sum_{i=0}^{255} \hat{f}_i x^i \in R_q$ を求める関数で，

$$\hat{f}_{2i}=\sum_{j=0}^{127} f_{2j}\zeta^{(2\mathsf{br}_7(i)+1)j} \quad \text{および} \quad \hat{f}_{2i+1}=\sum_{j=0}^{127} f_{2j+1}\zeta^{(2\mathsf{br}_7(i)+1)j}$$

  により定義される。ただし，$\mathsf{br}_7(i)$ は 7bits の整数を引数にとり，そのビット順序を反転した整数を出力する関数である。$\zeta=17$ は $\mathbb{Z}_q$ における原始元である。
  この表記は複数の R_q の元を並べたベクトル $\boldsymbol{s}=(s_0, s_1, \ldots, s_{k-1}) \in R_q^k$ にも有効で，$\mathsf{NTT}(\boldsymbol{s}) = (\mathsf{NTT}(s_0), \mathsf{NTT}(s_1), \ldots, \mathsf{NTT}(s_{k-1}))$ 等と解釈する。
- $\mathsf{Parse}(\mathsf{XOF}(\rho, i, j))$: XOF (extendable output function) を用いてシードの ρ, i, j から十分な長さの擬似乱数列を生成し，それを Parse 関数により R_q の元に変換する。

[*7] 仕様書では，アルゴリズムの形が Lyubashevsky-Peikert-Rosen の Ring-LWE ベース暗号 [107] に似ているとしている。
[*8] NIST PQC 標準化プロジェクト 第 1 ラウンド提出時には $q=7681$ であったが，第 2 ラウンドからはこの値に変更された。

- $\mathsf{CBD}_\eta(\mathsf{PRF}(\sigma, i))$: 大きさ $\eta \in \mathbb{N}$ の Central Binomial Distribution (CBD) を生成する。擬似乱数生成器 PRF は長さ 32Bytes の σ と 1Byte の i をシードとして 512ηbits の擬似乱数列 $\beta_0\beta_1\cdots\beta_{512\eta-1}$ へと変換する。この列を 2ηbits ごとに切り分け $i = 0, \ldots, 255$ に対して $f_i = \sum_{j=0}^{\eta-1} \beta_{i\cdot 2\eta+j} - \sum_{j=0}^{\eta-1} \beta_{i\cdot 2\eta+\eta+j}$ を計算。i 次の係数を f_i とした 255 次多項式を CBD_η 関数の出力とする。
- $\mathsf{Encode}_\ell(\hat{\boldsymbol{s}}), \mathsf{Decode}_\ell(\boldsymbol{b})$: Encode_ℓ 関数は 255 次の多項式 $\hat{\boldsymbol{s}} \in R_q$ を入力とし，各係数を ℓbits のビット列に直したものを結合した 256ℓbits のビット列を出力とする。Decode 関数はその逆を行う関数で，ビット列を多項式環の元に変換する。
- $\mathsf{Compress}_q(x, d), \mathsf{Decompress}_q(x, d)$: $x \in \mathbb{Z}_q$ を近似的に $d$bits に変換，逆変換を行う関数であり，暗号文のサイズ削減に用いられる。具体的には

$$\begin{aligned} \mathsf{Compress}_q(x, d) &:= \lceil (2^d/q) \cdot x \rfloor \bmod 2^d, \text{および} \\ \mathsf{Decompress}_q(x, d) &:= \lceil (q/2^d) \cdot x \rfloor \end{aligned}$$

で定義される。

擬似乱数生成器の実装について: アルゴリズムの仕様の中で用いられる擬似乱数生成器 XOF, PRF, G, H, KDF について，元々の SHAKE ハッシュ関数などを用いたものに加え，NIST PQC 標準化プロジェクト 第 2 ラウンドに合わせてアップデートされたバージョン 2.0[18] からは “90s version” として AES と SHA のみを用いたものが提案されている。これらの関数がデファクトスタンダードとして既に多くのハードウェア上で実装されている事から，高速化を狙ったものである。以下の表 3.4 に用いられる関数をまとめる。なお，本節で紹介する IND-CPA 安全な方式の中では XOF, PRF および G のみが用いられ，他の 2 つは IND-CCA2 安全な方式の構成において呼び出される。

90s version の XOF 関数では CTR モードの AES-256 を，ρ を鍵，12Bytes の nonce を $\text{nonce}[0] = i$, $\text{nonce}[1] = j$, $\text{nonce}[\ell] = 0$ for $\ell = 2, \ldots, 11$ とパディングして用いる。同様に PRF 関数では AES-256 の CTR モードを ρ を鍵，12Bytes の nonce を $\text{nonce}[0] = i$, $\text{nonce}[\ell] = 0$ for $\ell = 1, \ldots, 11$ として用いる。オリジナルバージョンの SHAKE-128 の呼び出し方に関してはリファレンス実装[*9] を参照した。

表 3.4: CRYSTALS-Kyber における擬似乱数生成器の実装 [19, Sect. 1.4]

	$\mathsf{XOF}(\rho, i, j)$	$\mathsf{PRF}(\sigma, i)$	$\mathrm{H}(\boldsymbol{b})$	$\mathrm{G}(\boldsymbol{b})$	$\mathsf{KDF}(\boldsymbol{b})$
オリジナル	$\mathsf{SHAKE\text{-}128}(\rho\|\|i\|\|j)$	$\mathsf{SHAKE\text{-}256}(\sigma\|\|i)$	$\mathsf{SHA3\text{-}256}(\boldsymbol{b})$	$\mathsf{SHA3\text{-}512}(\boldsymbol{b})$	$\mathsf{SHAKE\text{-}256}(\boldsymbol{b})$
90s	AES-256	AES3-256	$\mathsf{SHA\text{-}256}(\boldsymbol{b})$	$\mathsf{SHA\text{-}512}(\boldsymbol{b})$	$\mathsf{SHA\text{-}256}(\boldsymbol{b})$

CRYSTALS-Kyber の鍵生成関数（表 3.5 右）を説明する。表の中で $\mathcal{B}$ は 1Byte 分の情報を表す集合 $\{0, 1, \ldots, 255\}$ を表す。ランダムに生成した 32Bytes の d をシードとして，ハッシュ関数 G を用いて 512Bytes の擬似ランダムビットの組 (ρ, σ) を生成する。これらはそれぞれ，行列 $A \in R_q^{k\times k}$ とノイズ多項式 $\boldsymbol{s}, \boldsymbol{e} \in R_q^k$ をサンプリングするためのシードとして用いられる。通常空間で R_q を一様ランダムにサンプルしたものに NTT をかけた後の分布はまた R_q 内の一様分布となるため，A は最初から NTT 空間でサンプリングされているものとみなされる。

$\boldsymbol{s}, \boldsymbol{e} \in R_q^k$ については CBD_{η_1} を用いて通常空間でのサンプリングを行い，その成分を個別に数論変換する。数論変換の性質により，最後の $\hat{\boldsymbol{t}}$ は $\mathsf{NTT}(A\boldsymbol{s} + \boldsymbol{e})$ となる。公開鍵サイズを圧縮するため，$A, \hat{\boldsymbol{t}}$ をそれぞれシード ρ，Encode 関数による圧縮形式で保存する。秘密鍵の $\hat{\boldsymbol{s}}$ に関しても同様である。

CRYSTALS-Kyber の暗号化関数（表 3.6 右）を説明する。圧縮形で入力された公開鍵から $\hat{\boldsymbol{t}}, \hat{A}$ を復元する。このとき，処理の効率化のために行列は転置された形で復元される。

[*9] https://github.com/pq-crystals/kyber/blob/master/ref/symmetric-shake.c, 2024/12/24 参照

表 3.5: Lindner-Peikert 格子ベース暗号および CRYSTALS-Kyber における鍵生成関数の比較

	Lindner-Peikert [101, Sect. 3.1] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$	CRYSTALS-Kyber [19, Algorithm 4] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$
0:		$d \xleftarrow{\$} \mathcal{B}^{32}$ $(\rho, \sigma) \leftarrow \mathrm{G}(d)$ // $\mathcal{B}^{256} \times \mathcal{B}^{256}$ の疑似ランダムビット列
1:	A: $n_1 \times n_2$ ランダム行列	$\hat{A}[i][j] \leftarrow \mathsf{Parse}(\mathsf{XOF}(\rho, j, i))$ for $i = 0, \ldots, k-1$ and $j = 0, \ldots, k-1$
2:	S: 成分の小さい $n_2 \times \ell$ 行列	$\boldsymbol{s}[i] \leftarrow \mathsf{CBD}_{\eta_1}(\mathsf{PRF}(\sigma, i))$ for $i = 0, \ldots, k-1$ $\hat{\boldsymbol{s}} \leftarrow \mathsf{NTT}(\boldsymbol{s})$
3:	E: 成分の小さい $n_1 \times \ell$ 行列	$\boldsymbol{e}[i] \leftarrow \mathsf{CBD}_{\eta_1}(\mathsf{PRF}(\sigma, i+k))$ for $i = 0, \ldots, k-1$ $\hat{\boldsymbol{e}} \leftarrow \mathsf{NTT}(\boldsymbol{e})$
4:	$B = AS + E$	$\hat{\boldsymbol{t}} \leftarrow \hat{A} \circ \hat{\boldsymbol{s}} + \hat{\boldsymbol{e}}$
return	$pk = (A, B), sk = S$	$pk = (\mathsf{Encode}_{12}(\hat{\boldsymbol{t}} \bmod q) \| \rho), sk = \mathsf{Encode}_{12}(\hat{\boldsymbol{s}} \bmod q)$

表 3.6: Lindner-Peikert 格子ベース暗号および CRYSTALS-Kyber における暗号化関数の比較

	Lindner-Peikert[101, Sect. 3.1] $\mathsf{Enc}(pk = (A, B), \boldsymbol{m} \in \{0,1\}^\ell) \to \mathsf{ct}$	CRYSTALS-Kyber [19, Algorithm 5] $\mathsf{Enc}(pk = (T \| \rho), m \in \mathcal{B}^{32}) \to \mathsf{ct}$
0:		$\hat{\boldsymbol{t}} \leftarrow \mathsf{Decode}_{12}(T)$ $\hat{A}^T[i][j] \leftarrow \mathsf{Parse}(\mathsf{XOF}(\rho, i, j))$ //行列 $\hat{\boldsymbol{A}}$ の転置の形での復元
1:	$\boldsymbol{s}', \boldsymbol{e}', \boldsymbol{e}''$: 成分の小さいベクトル それぞれ Kyber の $\boldsymbol{r}, \boldsymbol{e}_1, \boldsymbol{e}_2$ に対応	$\boldsymbol{r}[i] \leftarrow \mathsf{CBD}_{\eta_1}(\mathsf{PRF}(r, i))$ for $i = 0, \ldots, k-1$ $\boldsymbol{e}_1[i] \leftarrow \mathsf{CBD}_{\eta_2}(\mathsf{PRF}(r, i+k))$ for $i = 0, \ldots, k-1$ $e_2 \leftarrow \mathsf{CBD}_{\eta_2}(\mathsf{PRF}(r, 2k))$
2:	$\boldsymbol{u} = \boldsymbol{s}'A + \boldsymbol{e}'$ $\boldsymbol{v} = \boldsymbol{s}'B + \boldsymbol{e}'' + \boldsymbol{m} \cdot \lfloor \frac{q}{2} \rfloor$	$\hat{\boldsymbol{r}} \leftarrow \mathsf{NTT}(\boldsymbol{r})$ $\boldsymbol{u} \leftarrow \mathsf{NTT}^{-1}(\hat{\boldsymbol{A}}^T \circ \hat{\boldsymbol{r}}) + \boldsymbol{e}_1$ $v \leftarrow \mathsf{NTT}^{-1}(\hat{\boldsymbol{t}}^T \circ \hat{\boldsymbol{r}}) + e_2 + \mathsf{Decompress}_q(\mathsf{Decode}_1(m), 1)$ $c_1 \leftarrow \mathsf{Encode}_{d_u}(\mathsf{Compress}_q(\boldsymbol{u}, d_u))$ $c_2 \leftarrow \mathsf{Encode}_{d_v}(\mathsf{Compress}_q(v, d_v))$
return	$\mathsf{ct} = (\boldsymbol{u}, \boldsymbol{v})$	$\mathsf{ct} = (c_1 \| c_2)$

暗号化のため成分の小さい $\boldsymbol{r}, \boldsymbol{e}_1 \in R_q^k$ と $e_2 \in R_q$ をサンプリングする。通常空間と NTT 空間を使い分けて処理を効率化しているが，最終的な暗号文 $c_1 \| c_2$ は通常空間でのベクトル $\boldsymbol{u} \in R_q^k$ と多項式 $v \in R_q$ を $\mathsf{Compress}_q$ 関数で圧縮したものとなる。ここで，2 種類のノイズ η_1, η_2 を使い分けるのは，η_1 のみによるノイズの大きさと，最後の Encode 関数によるラウンディングからの決定的ノイズと η_2 のノイズを合成したものの大きさが釣り合うように調整するためである [19, Sect. 1.5]。

CRYSTALS-Kyber の復号関数（表 3.7 右）は圧縮されたビット列の展開，NTT 空間の利用などで表現が煩雑になっているが，Lindner-Peikert 暗号の復号処理と本質的に同様である。最後の $\mathsf{Compress}_q(\cdot, 1)$ 関数が Lindner-Peikert 暗号における $\overline{\boldsymbol{m}}$ から $\boldsymbol{m}'$ への変換に対応している。

安全性とパラメータ: ベースとなる IND-CPA 安全な公開鍵暗号の安全性は多項式環 $R_q := \mathbb{Z}_q[x]/(x^n + 1)$ 上の判定版 Module-LWE 問題へと ROM，QROM モデルの下で帰着される。

パラメータの設定は Module-LWE 問題を構造の無い LWE 問題とみなし Primal, Dual の双方の攻撃を BKZ アルゴ

表 3.7: Lindner-Peikert 格子ベース暗号および CRYSTALS-Kyber における復号関数の比較

	Lindner-Peikert[101, Sect. 3.1] $\mathsf{Dec}(sk, \mathsf{ct}) \to \boldsymbol{m}'$	CRYSTALS-Kyber [19, Algorithm 6] $\mathsf{Dec}(sk, \mathsf{ct} = (c_1 \| \| c_2)) \to m' \in \mathcal{B}^{32}$
1:	$\overline{\boldsymbol{m}} = \boldsymbol{v} - \boldsymbol{u}S$ $m'_i = \begin{cases} 0 & \lvert\overline{m}_i\rvert \leq \lfloor q/4 \rceil \\ 1 & \text{それ以外} \end{cases}$	$\boldsymbol{u} \leftarrow \mathsf{Decompress}_q(\mathsf{Decode}_{d_u}(c_1), d_u)$ $v \leftarrow \mathsf{Decompress}_q(\mathsf{Decode}_{d_v}(c_2), d_v)$ $\hat{\boldsymbol{s}} \leftarrow \mathsf{Decode}_{12}(sk)$ $m' \leftarrow \mathsf{Encode}_1(\mathsf{Compress}_q(v - \mathsf{NTT}^{-1}(\hat{\boldsymbol{s}}^T \circ \mathsf{NTT}(\boldsymbol{u})), 1))$
return	$\boldsymbol{m}' = (m'_1, \ldots, m'_\ell)$	m'

リズムを用いて解いた場合の必要ブロックサイズに対応する Core SVP 計算量を通じて行われている。Module-LWE 問題へと帰着する際に，二項分布によるノイズと $\mathsf{Compress}_q$ 関数の四捨五入によるノイズを総合して詳細な解析を行っている。また，パラメータ設定用のスクリプトは [68] で公開されている。

暗号の性能を決めるパラメータは $n, k, q, \eta_1, \eta_2, d_u, d_v$ の 7 個であり，大まかに以下の特徴を持つ。格子の次元は多項式の次数 n と Module-LWE 問題のランク k の積であり，これらのパラメータを大きくとることで暗号の安全性が上がるが処理速度が低下し，鍵と暗号文のサイズが膨らむ。法 q を大きくとることでノイズ耐性が上がり復号エラー率が下がるが，格子が疎になり暗号の安全性が低下する。

(η_1, η_2) は鍵生成と暗号化に用いられるノイズ多項式の大きさで，大きくとることで暗号の安全性が上がるが復号エラー率も上がる。また，ノイズの中心二項分布を生成する際に必要とされるランダムビットの長さが増える。

(d_u, d_v) は暗号文 $(\boldsymbol{u}, v)$ をビット列で表現するための精度を指定する。小さくとることで暗号文サイズが削減できるが，桁落ちが発生し復号エラー率が上がる。また，これらの値を小さくとることは暗号文にノイズを与えることになり，安全性が僅かではあるが向上するが，復号エラー率への影響の方が大きい。

安全性レベル 1,3,5 に対応するパラメータの値を表 3.8 に示す。また，δ は IND-CCA2 KEM おける復号エラー率を示す。正しい暗号文が KEM のデカプセル化 [19, Algorithm 9] で棄却される確率である。

表 3.8: CRYSTALS-Kyber のパラメータ [19, Table 1] および [4, Sect. D]. 公開鍵，秘密鍵，平文，暗号文サイズの単位はそれぞれ Byte である。

(n, k, q)	(η_1, η_2)	(d_u, d_v)	安全性 レベル	公開鍵 サイズ	秘密鍵 サイズ	平文 サイズ	暗号文 サイズ	復号 エラー率 δ
$(256, 2, 3329)$	$(3, 2)$	$(10, 4)$	レベル 1	800	1,632	32	768	2^{-139}
$(256, 3, 3329)$	$(2, 2)$	$(10, 4)$	レベル 3	1,184	2,400	32	1,088	2^{-164}
$(256, 4, 3329)$	$(2, 2)$	$(11, 5)$	レベル 5	1,568	3,168	32	1,568	2^{-174}

3.3.4 CRYSTALS-Dilithium

歴史: CRYSTALS-Dilithium は 2017 年 6 月に Cryptology ePrint Archive において Léo Ducas，Tancrède Lepoint，Vadim Lyubashevsky，Peter Schwabe，Gregor Seiler，Damien Stehlé の 6 名の連名で公表 [65] され，その後論文内での予告通りに 2017 年 11 月に NIST PQC 標準化プロジェクトへの応募方式 [63] として Eike Kiltz を加えた 7 名を開発者として提出された。査読付き論文としては国際会議 CHES 2018 において公開された版 [64] が存在

する。

NIST PQC 標準化プロジェクトのラウンドが進むごとに微修正が行われ，現在の最新版は 2021 年 2 月に公開された仕様書 V3.1[20] である。本節の記述はこの仕様書に従う。

参照 URL: 開発者による公式ページ `https://pq-crystals.org/dilithium/` を参照した。

設計原理: CRYSTALS-Dilithium は格子ベースの署名方式であり，Lyubashevsky[105] による Fiat-Shamir with Aborts 型の構成を行っている。秘密鍵復元問題の安全性の根拠を，$x^{256}+1$ を定義多項式とした環上における Module-LWE 問題に，署名の強偽造不可能性の根拠を SelfTargetMSIS 問題に置いている。通信コストを下げるため，公開鍵サイズと署名サイズの和の最小化を目的としてパラメータの設計を行っている。

最新の実装では，署名の検証にかかる計算時間の 80% はハッシュ関数 Keccak の処理時間であり，速度的にはこれ以上改良できない限界であるとしている [104]。

アルゴリズムの詳細: 表 3.9，3.10，3.12 に Lyubashevsky による Fiat-Shamir with Aborts 型の格子ベース署名，CRYSTALS-Dilithium のテンプレートアルゴリズム [20, Fig. 1] および実装のための擬似コード [20, Fig.4] を並置して記述する。

パブリックパラメータは以下で与えられる。

- n, q: 環を定義するための多項式 x^n+1 の次数と法を示す。用いられる多項式環は $R := \mathbb{Z}[x]/(x^n+1), R_q := \mathbb{Z}_q[x]/(x^n+1)$ であり，提案方式の中では常に $n = 256, q = 2^{23} - 2^{13} + 1 = 8380417$ を用いる。
- k: モジュール格子のランクとする。
- l: ハッシュの (R_q における) 次元パラメータとする。
- d: 鍵生成時に $\boldsymbol{t}$ から分離する下位ビットの長さ
- η: 秘密鍵ベクトルのサンプリング空間の大きさ。
- τ: 署名生成時のベクトル $\boldsymbol{c}$ のサンプリング空間の大きさ。$\beta := \eta \cdot \tau$
- γ_1: 署名生成用ベクトル $\boldsymbol{y}$ のサンプリング空間の大きさ。
- γ_2: 署名生成用ベクトル $\boldsymbol{w}$ から取り出す上位ビットの長さ。

用いられるサブルーチンのうち主なものを以下に列挙する。

- $\mathsf{NTT}(\mathsf{a})$ は $\boldsymbol{a} = \sum_{i=0}^{255} a_i x^i$ の NTT 表現 $\hat{a} \in \mathbb{Z}_q^{256}$ を求める関数で，

$$\hat{a} = (a(r_0), a(-r_0), a(r_1), a(-r_1), \ldots, a(r_{127}), a(-r_{127}))$$

  で計算される。ただし，$r = 1753, r_i = r^{\mathrm{brv}(128+i)} \bmod q, \mathrm{brv}(k)$ 関数は k を 8bits の 2 進数としてみたときのビット順序を反転された数を出力するものとする [20, Sect. 2.2]。
- H: ビット列の伸長のためのハッシュ関数。CRYSTALS-Dilithium の実装では SHAKE256 ハッシュ関数を用いる。
- $\mathsf{ExpandA}(\rho)$: 乱数生成のシード ρ を用いて，ランダム行列 $A \in R_q^{k \times l}$ を生成し，その NTT 表現

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,l} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k,1} & a_{k,2} & \cdots & a_{k,l} \end{bmatrix} \to \hat{A} = \begin{bmatrix} \mathsf{NTT}(a_{1,1}) & \mathsf{NTT}(a_{1,2}) & \cdots & \mathsf{NTT}(a_{1,l}) \\ \mathsf{NTT}(a_{2,1}) & \mathsf{NTT}(a_{2,2}) & \cdots & \mathsf{NTT}(a_{2,l}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathsf{NTT}(a_{k,1}) & \mathsf{NTT}(a_{k,2}) & \cdots & \mathsf{NTT}(a_{k,l}) \end{bmatrix}$$

  を計算し出力する。
- $\mathsf{ExpandS}(\rho')$: 署名に用いる多項式 $\boldsymbol{s}_1, \boldsymbol{s}_2$ を生成するための関数で，512bits のシードを入力とする。

- $\mathsf{Power2Round}_q(t,d), \mathsf{HighBits}_q(t,\alpha), \mathsf{LowBits}_q(t,\alpha)$: $\mathbb{Z}_q$ の元 t で，$0 \le t < q$ を満たすものを $t = r_1 \cdot 2^d + r_0, -q/2 < r \le q/2$ と分解したときに $\mathsf{Power2Round}_q(t,d) = (r_1, r_0)$ と定義する。$\mathbb{Z}_q$ 成分の多項式 $t \in R_q$，および R_q 成分のベクトル $\boldsymbol{t}$ に対しても成分ごとに同様の操作を行うものとして定義する。具体的には，$\boldsymbol{t} = \left(\sum t_{j,i}x^i\right)_{j=1,\dots,k}$ と書いたときに $\mathsf{Power2Round}_q(t_{j,i}, d) \to (t_{j,i,1}, t_{j,i,0})$ とすれば，$\mathsf{Power2Round}_q(\boldsymbol{t}, d) \to (\boldsymbol{t}_1, \boldsymbol{t}_0)$ は $\boldsymbol{t}_1 = \left(\sum q_{j,i}x^i\right)_{j=1,\dots,k}, \boldsymbol{t}_0 = \left(\sum r_{j,i}x^i\right)_{j=1,\dots,k}$，ただし $t_{j,i} = q_{j,i} \cdot 2^d + r_{j,i}$ と定義したものとする。

  また，α を $q-1$ の約数としたとき，同様に整数 t を $t = r_1 \cdot \alpha + r_0, -q/2 < r_0 \le q/2$ の形で分解し，$\mathsf{HighBits}_q(t,\alpha), \mathsf{LowBits}_q(t,\alpha)$ をそれぞれ r_1, r_0 で定義する。
- $\mathsf{MakeHint}_q(z,r,\alpha), \mathsf{UseHint}_q(h,r,\alpha)$: $\mathsf{MakeHint}_q$ 関数は $\mathsf{HighBits}_q(r,\alpha) \neq \mathsf{HighBits}_q(r+z,\alpha)$ であれば 1 を，そうでなければ 0 を返す関数である。$\mathsf{UseHint}_q$ 関数は引数から $\mathsf{HighBits}_q(r+z,\alpha)$ を復元する関数である。また，ベクトル $\mathbf{z} = (z_1, \dots, z_k), \mathbf{r} = (r_1, \dots, r_k)$ に対して，$\mathsf{MakeHint}_q(\mathbf{z}, \mathbf{r}, \alpha)$ は $\mathsf{MakeHint}_q(z_i, r_i, \alpha)$ を第 i 成分としたベクトルとする。復元成功の十分条件は [20, Lemma 4] で与えられている。
- $\mathsf{SampleInBall}(\tilde{c})$ 関数は係数のうち τ 個が ± 1 で，それ以外が 0 である多項式の集合 B_τ から一様サンプリングを行う。τ はパブリックパラメータとして与えられており，引数の $\tilde{c}$ はサンプリングのシードとして用いられる。生成された多項式 $c \in R$ の NTT 表現 $\hat{c} = \mathsf{NTT}(c)$ が出力される。
- $\#_1\boldsymbol{h}$ はベクトル $\boldsymbol{h} = (h_0, \dots, h_{255})$ の中で，$h_i = 1$ となる成分の個数を返す。Dilithium の中では MakeHint 関数の出力となる 0-1 ベクトルであるため，ベクトルのハミング重みである。

表 3.9: CRYSTALS-Dilithium における鍵生成関数の比較

	格子ベース署名 [105, Fig. 4] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$	CRYSTALS-Dilithium テンプレート [20, Fig. 1] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$	CRYSTALS-Dilithium 実装のための擬似コード [20, Fig. 4] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$
1:	$\hat{\boldsymbol{s}}$: 短い多項式を 成分とするベクトル	$\boldsymbol{s}_1 \leftarrow S_\eta^l, \boldsymbol{s}_2 \leftarrow S_\eta^k$	$\zeta \xleftarrow{\$} \{0,1\}^{256}$ $\mathsf{H}(\zeta) \to (\rho, \rho', K) \in \{0,1\}^{256} \times \{0,1\}^{512} \times \{0,1\}^{256}$ $\mathsf{ExpandS}(\rho') \to (\boldsymbol{s}_1, \boldsymbol{s}_2) \in S_\eta^\ell \times S_\eta^k$
2:	a: ハッシュ関数	$\hat{A} \xleftarrow{\$} R_q^{k \times l}$	$\mathsf{ExpandA}(\rho) \to \hat{A} \in R_q^{k \times \ell}$
3:	$t \leftarrow a(\hat{\boldsymbol{s}})$	$\boldsymbol{t} = A\boldsymbol{s}_1 + \boldsymbol{s}_2$	$\boldsymbol{t} \leftarrow \mathsf{NTT}^{-1}(\hat{A} \cdot \mathsf{NTT}(\boldsymbol{s}_1)) = A\boldsymbol{s}_1 + \boldsymbol{s}_2$ $\mathsf{Power2Round}_\mathsf{q}(\boldsymbol{t}, d) \quad \to (\boldsymbol{t}_1, \boldsymbol{t}_0)$ $\mathsf{H}(\rho \| \boldsymbol{t}_1) \to tr \in \{0,1\}^{256}$
return	$sk = (a, \hat{\boldsymbol{s}}), pk = (a, \boldsymbol{t})$	$sk = (A, \boldsymbol{t}, s_1, s_2), pk = (A, \boldsymbol{t})$	$sk = (\rho, K, tr, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{t}_0), pk = (\rho, \boldsymbol{t}_1)$

表 3.9 の鍵生成関数について記述する。256bits のシード ζ をハッシュ関数 H により合計 1024bits に伸長し，そのうち ρ, ρ' をそれぞれ公開鍵 A のシード，秘密鍵 $\boldsymbol{s}_1, \boldsymbol{s}_2$ のシードとして用いる。鍵サイズ圧縮のため，行列 A はシード ρ の形で表現され，必要に応じて展開される。秘密鍵 $\boldsymbol{s}_1, \boldsymbol{s}_2$ は R の元をそれぞれ ℓ, k 個並べたベクトルであり，各成分は集合 $S_\eta = \{\boldsymbol{w} \in R : \|\boldsymbol{w}\|_\infty \le \eta\}$ から一様ランダムにサンプリングされる。

Step 2 および 3 では Fiat-Shamir 型格子署名における秘密鍵 $\hat{\boldsymbol{s}}$ のハッシュ関数 $a(\hat{\boldsymbol{s}})$ の計算が，ベクトル $(\boldsymbol{s}_1, \boldsymbol{s}_2)$ と行列 A を用いた $A\boldsymbol{s}_1 + \boldsymbol{s}_2$ の計算に対応している。計算されたベクトル $\boldsymbol{t} \in R_q^k$ に対して，$\mathsf{Power2Round}_q$ 関数により上位ビットと下位ビットに分割する。

最後に，メッセージに連結するためのランダムビット tr をハッシュ関数 H を用いて生成する。

表 3.10 の署名生成関数について記述する。処理の準備として，ρ から行列 A の NTT 表現 $\hat{A}$ を復元する。ランダムビット tr を用いてメッセージのハッシュ値 μ を計算し，この値に署名をつける。κ は ExpandMask 関数の中で呼び出す SHAKE256 のシードとなる値で，$\mathsf{H}(K||\mu) \to \rho'$ とともに用いられる。計算効率化の目的で R_q の元の乗算には

表 3.10: CRYSTALS-Dilithium における署名生成関数の比較

	格子ベース署名 [105, Fig. 4] $\mathsf{Sign}(sk=(a,\hat{\boldsymbol{s}}),$ $\mu\in\{0,1\}^*)\to\sigma$	CRYSTALS-Dilithium テンプレート [20, Fig. 1] $\mathsf{Sign}(sk=(A,t,\boldsymbol{s}_1,\boldsymbol{s}_2),$ $\mu\in\{0,1\}^*)\to\sigma$	CRYSTALS-Dilithium 実装のための擬似コード [20, Fig. 4] $\mathsf{Sign}(sk=(\rho,K,tr,\boldsymbol{s}_1,\boldsymbol{s}_2,\boldsymbol{t}_0),$ $M\in\{0,1\}^*\})\to\sigma$
0:			$\mathsf{ExpandA}(\rho)\to\hat{A}$ $\mathsf{H}(tr\|\|M)\to\mu\in\{0,1\}^{512}$
1:	$z\leftarrow\perp$	$z\leftarrow\perp$	$\kappa\leftarrow 0,(\boldsymbol{z},\boldsymbol{h})\leftarrow\perp$ $\mathsf{H}(K\|\|\mu)\to\rho'\in\{0,1\}^{512}$ $\hat{\boldsymbol{s}}_1\leftarrow\mathsf{NTT}(\boldsymbol{s}_1);\ \hat{\boldsymbol{s}}_2\leftarrow\mathsf{NTT}(\boldsymbol{s}_2)$ $\hat{\boldsymbol{t}}_0\leftarrow\mathsf{NTT}(\boldsymbol{t}_0)$
2:	**while** $z=\perp$ **do**	**while** $z=\perp$ **do**	**while** $(\boldsymbol{z},\boldsymbol{h})=\perp$ **do**
3:	$\hat{\boldsymbol{y}}$: 短い多項式を 成分とするベクトル	$\boldsymbol{y}\leftarrow D^{l\times 1}_{\gamma_1-1}$	$\mathsf{ExpandMask}(\rho',\kappa)\to\boldsymbol{y}\in\tilde{S}^l_{\gamma_1}$
4:	$c\leftarrow H(a(\hat{\boldsymbol{y}})\|\|\mu)$	$\boldsymbol{w}_1\leftarrow\mathsf{HighBits}(A\boldsymbol{y},2\gamma_2)$ $c=H(\mu\|\|\boldsymbol{w}_1)$	$\boldsymbol{w}\leftarrow\mathsf{NTT}^{-1}(\hat{A}\cdot\mathsf{NTT}(\boldsymbol{y}))=\boldsymbol{Ay}$ $\boldsymbol{w}_1\leftarrow\mathsf{HighBits}_q(\boldsymbol{w},2\gamma_2)$ $H(\mu\|\|\boldsymbol{w}_1)\to\tilde{c}\in\{0,1\}^{256}$ $\mathsf{SampleInBall}(\tilde{c})\to\hat{c}\in B_\tau\subset R_q$
5:	$\hat{\boldsymbol{z}}\leftarrow\hat{\boldsymbol{y}}+c\hat{\boldsymbol{s}}$ if $\hat{\boldsymbol{z}}\notin G^m$ **then** $z\leftarrow\perp$	$\boldsymbol{z}\leftarrow\boldsymbol{y}+c\boldsymbol{s}_1$ $\boldsymbol{r}_0\leftarrow\mathsf{LowBits}(A\boldsymbol{y}-c\boldsymbol{s}2,2\gamma_2)$ **if** $(\\|\boldsymbol{z}\\|_\infty\geq\gamma_1-\beta)$ OR $(\\|\boldsymbol{r}_0\\|_\infty\geq\gamma_2-\beta)$ **then** $z\leftarrow\perp$	$\boldsymbol{z}\leftarrow\boldsymbol{y}+\mathsf{NTT}^{-1}(\hat{c}\cdot\hat{\boldsymbol{s}}_1)$ $\boldsymbol{r}_0\leftarrow\mathsf{LowBits}_q(\boldsymbol{w}-\mathsf{NTT}^{-1}(\hat{c}\cdot\hat{\boldsymbol{s}}_2),2\gamma_2)$ **if** $(\\|\boldsymbol{z}\\|_\infty\geq\gamma_1-\beta)$ OR $(\\|\boldsymbol{r}_0\\|_\infty\geq\gamma_2-\beta)$ **then** $(\boldsymbol{z},\boldsymbol{h})\leftarrow\perp$ **else** $\boldsymbol{h}\leftarrow\mathsf{MakeHint}_q(\cdot)$ ... (*) $\kappa\leftarrow\kappa+l$
return	$\sigma=(\hat{\boldsymbol{z}},c)$	$\sigma=(\boldsymbol{z},c)$	$\sigma=(\boldsymbol{z},\boldsymbol{h},\tilde{c})$

NTT 表現を用いるため，予め $\boldsymbol{s}_1,\boldsymbol{s}_2,\boldsymbol{t}_0$ を NTT 表現に変換しておく。

Fiat-Shamir 型署名の標準的な構成方法と同様に，署名の初期値 $(\boldsymbol{z},\boldsymbol{h})$ を $\perp$ とし，**while** ループの中で生成された署名が集合 G に含まれているかどうかを検査し含まれていない場合にはループをやり直す。

ExpandMask 関数の中では，(ρ',κ) をシードとしてランダムベクトル $\boldsymbol{y}\in R_q^l$ をサンプリングする。ここで，各成分は $\tilde{S}_{\gamma_1}=\left\{\sum_{i=0}^{255}w_ix^i:-\gamma_1<w_i\leq\gamma_1\right\}$ から一様ランダムにサンプリングされる。このサンプリングは表 3.10 中央の $\boldsymbol{y}\leftarrow D^{l\times 1}_{\gamma_1}$ に対応する。

署名生成のためのベクトル $c\in R_q^l$ は 256bits のシード $\tilde{c}$ により表現され，この値自体は μ と $\boldsymbol{w}_1$ を連結したハッシュ値から計算される。ここで，μ はメッセージからの要素であり，$\boldsymbol{w}_1$ は公開鍵 A と直前でサンプリングした $\boldsymbol{y}$ から来る要素である。計算効率のため，内積 $c\cdot\boldsymbol{s}_1$ は NTT 表現で計算された後に逆変換をかけ $\boldsymbol{z}=\boldsymbol{y}+c\cdot\boldsymbol{s}_1$ となる。

ステップ 5 では $z\notin G$ のチェックのため，$\boldsymbol{z}$ と $\boldsymbol{w}-c\boldsymbol{s}_2$ の下位ビットの ℓ_∞ ノルムがそれぞれ比較される。両方が閾値よりも小さい場合には次のヒント生成関数 $(*)$ が実行される。ヒント生成関数は表 3.11 により示され，$\mathsf{MakeHint}_q$ 実行後に再びノルムの大きさがチェックされ，閾値よりも大きな場合には $(\boldsymbol{z},\boldsymbol{h})\leftarrow\perp$ となる。つまり，2 回の **if** 文の中での 4 回の不等号検査のうち一つでも満たされない条件があれば，シード κ を増やし $\boldsymbol{y}$ の生成からやり直すことになる。ここで，$\mathsf{MakeHint}_q$ 関数の中での $-c\boldsymbol{s}_2+c\boldsymbol{t}_0$ の計算は前半を $\boldsymbol{r}_0$ の計算で用いたものを使いまわし，後半を $\mathsf{NTT}^{-1}(\hat{c}\cdot\hat{\boldsymbol{t}}_0)$ の形で計算することで効率化可能である。

表 3.12 の署名検証関数について記述する。公開鍵，署名に含まれる乱数のシード $\rho,\tilde{c}$ から $\hat{A},\hat{c}$ を復元し，メッセージに対応するハッシュ値 μ を計算する。$A\boldsymbol{z}-c\boldsymbol{t}_1\cdot 2^d$ は $\hat{A}\cdot\mathsf{NTT}(\boldsymbol{z})-\mathsf{NTT}(c)\cdot\mathsf{NTT}(\boldsymbol{t}_1\cdot 2^d)$ の形で計算する。これ

表 3.11: 署名生成関数におけるヒント生成時のチェック関数

$\boldsymbol{h} \leftarrow \mathsf{MakeHint}_q(-c\boldsymbol{t}_0, \boldsymbol{w} - c\boldsymbol{s}_2 + c\boldsymbol{t}_0, 2\gamma_2)$ **if** $\|c\boldsymbol{t}_0\|_\infty \geq \gamma_2$ OR $\#_1\boldsymbol{h} > \omega$ **then** $(\boldsymbol{z}, \boldsymbol{h}) \leftarrow \perp$

表 3.12: CRYSTALS-Dilithium における署名検証関数の比較

	格子ベース署名 [105, Fig. 4] $\mathsf{Vrfy}(pk = (a, \boldsymbol{t}), \mu \in \{0,1\}^*,$ $\sigma = (\hat{\boldsymbol{z}}, c))$	CRYSTALS-Dilithium テンプレート [20, Fig. 1] $\mathsf{Vrfy}(pk = (A, \boldsymbol{t}), \mu \in \{0,1\}^*,$ $\sigma = (\boldsymbol{z}, c))$	CRYSTALS-Dilithium 実装のための擬似コード [20, Fig. 4] $\mathsf{Vrfy}(pk = (\rho, \boldsymbol{t}_1), M \in \{0,1\}^*,$ $\sigma = (\boldsymbol{z}, \boldsymbol{h}, \tilde{c}))$
0:			$\mathsf{ExpandA}(\rho) \rightarrow \hat{A}$ $\mathsf{H}(\mathsf{H}(\rho\|\|\boldsymbol{t}_1)\|\|M) \rightarrow \mu \in \{0,1\}^{512}$ $\mathsf{SampleInBall}(\tilde{c}) \rightarrow \hat{c}$
1:	**if** $\hat{\boldsymbol{z}} \in G^m$ AND $c = H(a(\hat{\boldsymbol{z}}) - \boldsymbol{t}c, \mu)$ **then** accept **else** reject	$\boldsymbol{w}_1' = \mathsf{HighBits}(A\boldsymbol{z} - c\boldsymbol{t}, 2\gamma_2)$ **if** $\|\boldsymbol{z}\|_\infty < \gamma_1 - \beta$ AND $c = H(M\|\|\boldsymbol{w}_1')$ **then** accept **else** reject	$\boldsymbol{w}_1' \leftarrow \mathsf{UseHint}_q(\boldsymbol{h}, A\boldsymbol{z} - c\boldsymbol{t}_1 \cdot 2^d, 2\gamma_2)$ **if** $\|\boldsymbol{z}\|_\infty < \gamma_1 - \beta$ AND $\tilde{c} = H(\mu\|\|\boldsymbol{w}_1')$ AND $\#_1\boldsymbol{h} \leq \omega$ **then** accept **else** reject

らの値から $\mathsf{UseHint}_q$ を用いて $\boldsymbol{w}_1'$ を復元し，$\boldsymbol{z}$ のノルム，$\boldsymbol{h}$ の 1 の数の確認を行い，正しければ accept を出力する。

安全性とパラメータ: CRYSTALS-Dilithium の安全性は，$x^n + 1$ を定義多項式とする環上のモジュール格子問題である。ROM の下で，秘密鍵復元の困難性が Module-LWE 問題に，署名の強偽造不可能性が SelfTargetMSIS 問題にそれぞれ帰着される。SelfTargetMSIS 問題は Module-SIS 問題の変種であり，署名の偽造不可能性からのタイトな古典帰着が知られている。一方で，Module-SIS 問題への古典帰着もタイトではないものの証明が与えられており，その意味では Module-SIS 問題を安全性の根拠と捉えることもできる。

方式の発表後，仕様書内の安全性証明における理論の飛躍が発見された [25, p. 3]。具体的には署名の受動的攻撃下における強偽造不可能性 (EUF-NMA) から選択文書攻撃下における強偽造不可能性 (EUF-CMA 安全性) への帰着を行う際に，アドバーサリーが想定された挙動を行うためには秘密鍵の情報を必要とするため，確率的多項式時間であることが保証できない。Barbosa らは 2023 年に問題点と修正，およびコンピュータ支援による安全性証明の論文 [25] を発表している。なお，証明の修正であるため方式自体の変更は行われていない。

一方で，QROM においても鍵復元，署名偽造が同様に Module-LWE 問題，SelfTargetMSIS 問題それぞれ帰着されるものの Module-SIS 問題までの量子帰着が知られていない。

具体的なパラメータは，LWE 問題と SIS 問題の双方に対して BKZ アルゴリズムで解いた際の必要ブロックサイズと Core-SVP の見積から求められている。

仕様書に掲載されたパラメータセットを表 3.13 に示す。セキュリティ強度を規定するパラメータのうち，問題が定義される環とモジュールのランクに関わるものが (n, k, l, q) の 4 個，ノイズに関わるものが $(\eta, \gamma_1, \gamma_2, \beta, \tau, d)$ の 6 個である。

変種: [20, Table 3] には NIST の提唱する安全性レベル 1 よりも弱いパラメータ，安全性レベル 5 よりも強いパラメータが掲載されている。NIST PQC 標準化が決定して以降仕様書のバージョン [20] から様々な変更が加えられ，標準方式 ML-DSA (3.3.2 節も参照) が公開された。変更の概要は本報告書 3.3.2.4 節を参照。

補足情報: NIST PQC 標準化プロジェクトの第 3 ラウンド報告レポートにおいて署名方式 FALCON との比較が行われ，CRYSTALS-Dilithium はそのシンプルさから一般的な実装に向いているが，FALCON は署名の短さからリソースの制限されたデバイスで使われることが期待されている [4, p. 19]。

表 3.13: CRYSTALS-Dilithium 署名方式のパラメータ [20, Table 1] , [4, Table 8]。公開鍵，秘密鍵，署名サイズの単位はそれぞれ Byte である。

(n, k, l, q)	$(\eta, \gamma_1, \gamma_2, \beta, \tau, d)$	安全性レベル	公開鍵サイズ	秘密鍵サイズ	署名サイズ
$(256, 4, 4, 8380417)$	$(2, 2^{17},\ 95232,\ 78, 49, 13)$	レベル 2	$1,312$	$2,528$	$2,420$
$(256, 6, 5, 8380417)$	$(4, 2^{19}, 261888, 196, 49, 13)$	レベル 3	$1,952$	$4,000$	$3,293$
$(256, 8, 7, 8380417)$	$(2, 2^{19}, 261888, 120, 60, 13)$	レベル 5	$2,592$	$4,864$	$4,595$

注： 秘密鍵サイズは仕様書 [20] には掲載されていないが，NIST の第 3 ラウンド報告レポート [4] を参照した。

3.3.5 FALCON

歴史: FALCON は 2017 年 11 月の NIST PQC 標準化プロジェクトの公募に Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, Zhenfei Zhang の 10 名を開発者として提出された [80]。その後修正が加えられ，現在の最新版は 2020 年 10 月に公開された v1.2[81] である。以下の記述はこの仕様書に従う。

参照 URL: 開発者による公式ページ `https://falcon-sign.info/` を参照した。

設計原理: FALCON は多項式 $x^n + 1, n = 2^k$ により定義される NTRU 格子上の SIS 問題の困難性を安全性の根拠とした格子ベースの署名方式であり，形式的には Gentry ら [83] の Hash-and-Sign 型の格子ベース署名をひな型としている。高速フーリエサンプリングを用いるため，定義多項式の次数を 2^k の形としていることからパラメータ選択の自由度に制限があり，NIST PQC 標準化プロジェクトの提案方式では安全性レベル 1 および 5 のパラメータセットのみが提案されている。

アルゴリズムの詳細: 表 3.14，3.15，3.16 に，Gentry ら [83] の Hash-and-Sign 型格子ベース署名と FALCON の鍵生成，署名生成，署名検証関数を並置する。

パブリックパラメータは以下で与えられる。

- n, q: 環を定義する多項式 $\phi(x) = x^n + 1$ と法 q で，演算は $\mathbb{Z}_q[x]/(\phi)$ で行われる。
- σ: 離散 Gauss 分布の大きさを指定する。
- β: 有効な署名のノルムの上限を指定する。

アルゴリズム中で用いられるサブルーチンのうち，主なものを列挙する。

- $\mathsf{FFT}(f), \mathsf{invFFT}(\boldsymbol{s})$: 多項式 $f \in \mathbb{R}[x]/(\phi)$ に対して，そのフーリエ変換 $\mathsf{FFT}(f)$ を n 次元ベクトル $(f(\zeta_k))_{k=0,\ldots,n-1}$ で定義する[*10]。ただし，$\zeta_k := \exp((2k+1)\pi i/n)$。逆演算を $\mathsf{invFFT} : \mathbb{R}^n \to \mathbb{R}[x]/(\phi)$ で示す。変換，逆変換ともに標準的な高速フーリエ変換の手法が利用可能である。コンピュータ上での計算には浮動小数点演算を用いるため，実行環境ごとに差が出ないように IEEE754 で規定される浮動小数点の表現と演算を用いることが指定されている。
  多項式を成分とするベクトル，行列に対しても FFT は成分ごとのフーリエ変換と定義し，invFFT も適切な切り分けにより実数成分の行列，ベクトルから多項式成分の行列，ベクトルへ変換するものとする。

[*10] 数式上は差が無いが高速フーリエ変換による実装を行ったサブルーチンも同じ記号で示すため，Fast Fourier の意味で FFT と名づけられている。

また，演算 $\mathsf{FFT}(f) \odot \mathsf{FFT}(g)$ を成分ごとの積と定義する。FFT 表現での多項式の積 $\mathsf{FFT}(fg)$ の計算に対応する。

- $\mathsf{HashToPoint}(\mathsf{str}, q, n)$: ビット列 str を多項式 $c \in \mathbb{Z}_q[x]/(\phi)$ に SHAKE256 ハッシュ関数を用いて写像する。
- Compress, Decompress: 多項式 $s \in \mathbb{Z}[x]$ を文字列に変換する関数とその逆関数とする。
- $\mathsf{NTRUGen}(\phi, q)$: 計算が行われる環 $\mathbb{Z}_q[x]/(\phi)$ を指定するパラメータを入力とし，秘密鍵 $\hat{B}$ の元となる多項式 f, g, F, G を出力する。このとき，f, g は係数が離散 Gauss 分布の n 次多項式，F, G は $fG - gF \equiv q \mod \phi$ を満たすように計算される。

表 3.14: Hash-and-Sign 型格子ベース署名および FALCON における鍵生成関数の比較

	Gentry らの格子ベース署名 [83, Sect. 7.1] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$	FALCON[81, Algorithm 4] $\mathsf{KeyGen}(\phi, q) \to (pk, sk)$
1:	$BA \equiv 0 \pmod q$ を満たす行列の組 (A, B) を生成 B: 成分の小さい行列 A: ランダム行列	$f, g, F, G \leftarrow \mathsf{NTRUGen}(\phi, q)$ $B \leftarrow \begin{bmatrix} g & -f \\ G & -F \end{bmatrix}$ $\hat{B} \leftarrow \mathsf{FFT}(B)$ $G \leftarrow \hat{B} \times \hat{B}^*$ $T \leftarrow \mathsf{ffLDL}^*(G)$ **for** each leaf leaf of T **do** $\quad \mathsf{leaf.value} \leftarrow \sigma/\sqrt{\mathsf{leaf.value}}$ $h \leftarrow gf^{-1} \bmod q$
return	$pk = A, sk = B$	$pk = h, sk = (\hat{B}, T)$

NTRU 型暗号の秘密鍵 (f, g) のうち，f は環 $\mathbb{Z}_q/(\phi)$ の中で逆元を持つため，適当な $F, G \in \mathbb{Z}[x]$ を用いて

$$fG - gF = q \bmod \phi \tag{3.3}$$

と書くことができる。この関係式と公開鍵 $h = f^{-1}g$ を Hash-and-Sign フレームワーク [83] における行列 A, B と捉えると，

$$A = \begin{bmatrix} 1 \\ h \end{bmatrix},\ B = \begin{bmatrix} g & -f \\ G & -F \end{bmatrix} \tag{3.4}$$

と表現することができる。このとき，行列 A は多項式 h の情報のみで表現可能であるため，$pk = h$ となる。

また，署名の生成には $\boldsymbol{s}A \equiv H(m)$ を満たす短いベクトル $\boldsymbol{s}$ を生成する必要があり，効率化のため Ducas-Prest[67] の高速フーリエサンプリングを用いる。サンプリングアルゴリズムに必要な情報が B の FFT 表現

$$\mathsf{FFT}(B) = \begin{bmatrix} \mathsf{FFT}(g) & \mathsf{FFT}(-f) \\ \mathsf{FFT}(G) & \mathsf{FFT}(-F) \end{bmatrix} \tag{3.5}$$

およびそれを元にした LDL 木と呼ばれる木構造 T である。木の中には $\hat{B}$ のグラム行列 $G = \hat{B} \times \hat{B}^\star$ の[*11] LDL 分解における L の情報が格納され，それを用いて Babai の最近平面アルゴリズムの高速化および離散 Gauss 分布の高速なサンプリングが可能となる。サンプリングを行うための付加情報として，木の全ての葉にある値を $\mathsf{leaf.value}$ から $\sigma/\sqrt{\mathsf{leaf.value}}$ に書き換えることで鍵生成が完了する。

[*11] $B^\star$ は体 $\mathbb{Q}[x]/(\phi)$ におけるエルミート共役。詳細は [81, p. 23]

表 3.15: Hash-and-Sign 型格子ベース署名および FALCON における署名生成関数の比較

	Gentry らの格子ベース署名 [83, Sect. 7.1] $\mathsf{Sign}(sk=(\hat{B},T), m\in\{0,1\}^*)\to\sigma$	FALCON[81, Algorithm 10] $\mathsf{Sign}(sk=(\hat{B},T), m\in\{0,1\}^*, \lfloor\beta^2\rfloor)\to\sigma$
1:	$\boldsymbol{c}\leftarrow H(m)$ //平文のハッシュ値をベクトル化	$r\leftarrow\{0,1\}^{320}$ $c\leftarrow\mathsf{HashToPoint}(r\|\|m,q,n)$ $\hat{\boldsymbol{t}}\leftarrow\left(-\frac{1}{q}\mathsf{FFT}(c)\odot\mathsf{FFT}(F), \frac{1}{q}\mathsf{FFT}(c)\odot\mathsf{FFT}(f)\right)$
2:	T を使い，$\boldsymbol{s}A\equiv\boldsymbol{c}\pmod q$ を 満たすベクトル $\boldsymbol{s}$ をサンプリング	**do** 　**do** 　　$\boldsymbol{z}\leftarrow\mathsf{ffSampling}_n(\hat{\boldsymbol{t}},T)$ 　　$\hat{\boldsymbol{s}}\leftarrow(\hat{\boldsymbol{t}}-\hat{\boldsymbol{z}})\hat{B}$ 　**while** $\|\boldsymbol{s}\|^2>\lfloor\beta^2\rfloor$ 　$(s_1,s_2)\leftarrow\mathsf{invFFT}(\hat{\boldsymbol{s}})$ 　$\mathsf{s}\leftarrow\mathsf{Compress}(s_2, 8\cdot\mathsf{sbytelen}-328)$ **while** ($\mathsf{s}=\perp$)
return	$\sigma=\boldsymbol{s}$	$\sigma=(r,\mathsf{s})$

表 3.15 の署名生成関数の説明を記述する。平文にランダムビット r を結合した後，HashToPoint 関数で多項式 $c\in\mathbb{Z}_q/(\phi)$ を出力する。関係式 (3.3), (3.4) より，ベクトル $\hat{\boldsymbol{t}}$ は $(\mathsf{FFT}(c),\mathsf{FFT}(0))\hat{B}^{-1}$ と等しい事がわかる。これらの情報を用いて，署名ベクトルのサンプリングを行う。

関数 $\mathsf{ffSampling_n}$ は，離散 Gauss 分布のサンプリングを行い，FFT 表現で出力するサブルーチンである。具体的には，整数ベクトル $\boldsymbol{z}\in\mathbb{Z}^{2n}$ を，$\boldsymbol{t}=[c,0]B^{-1}$ を中心として $\exp(-\|(\boldsymbol{z}-\boldsymbol{t})B\|^2/2\sigma^2)$ に比例した確率でサンプリングを行う。実装の効率化のため，実際には近似を行っている [81, Sect. 3.9.1, 3.9.2]。このとき，$(\boldsymbol{t}-\boldsymbol{z})B$ は原点を中心とした集合

$$\boldsymbol{t}+\Lambda(B)=\{(c,0)+x\in(\mathbb{Z}[x]/(\phi))^2 : x\in\Lambda(B)\}$$

上の離散 Gauss 分布となるため，$\boldsymbol{s}$ は短く，かつ

$$\boldsymbol{s}A\equiv([c,0]B^{-1}-\boldsymbol{z})BA\equiv[c,0]\begin{bmatrix}1\\ h\end{bmatrix}=c \text{ in } \mathbb{Z}_q[x]/(\phi)$$

が成り立つ。このとき，$\boldsymbol{s}A=c$ の関係から $s_1+s_2h=c$ が成り立つ。この関係式が署名の検証時に用いられる。

サンプリングされた $\hat{\boldsymbol{s}}$ が $\|\hat{\boldsymbol{s}}\|^2\le\lfloor\beta^2\rfloor$ を満たしていれば invFFT により通常空間の表現に戻し，Compress 関数を用いて圧縮された文字列 s を生成し，ハッシュ関数のシード r とともに署名とする。

表 3.16 の署名検証関数の説明を記述する。平文，ハッシュ関数のシード値，署名文字列から各要素を復元し，$s_1=c-s_2h$ を計算する。署名が正しく生成されていれば $\boldsymbol{s}A=c$ の関係から，s_1 は短い元となるはずなので，$\|(s_1,s_2)\|^2\le\lfloor\beta^2\rfloor$ が満たされ検証が完了する。

安全性とパラメータ: FALCON の安全性は $\phi(x)=x^n+1, q=12289$ を定義多項式とする NTRU 格子上の計算問題として表現される。鍵復元の困難性は SIS 問題，署名偽造はターゲットベクトルに近い点を求める計算問題として定式化される。後者は Kannan の埋め込みにより短いベクトルを求める計算問題に変換される。セキュリティに関わるパラメータは n,q,σ,β の 4 個で，n は格子の次元を表し，大きく取ることで安全性が上がるが処理速度が低下する。q は環を定義するための法で，大きくとることでノイズ耐性が上がるが格子が疎になり安全性が低下する。σ は Gauss 分布の大きさを指定するパラメータで，大きくとることで安全性が上がるがエラー率が上がる。β は署名ベクトルの長さの上限を指定するパラメータで，大きくとることで署名生成時のやり直し回数が下がるが，安全性が低下する。

表 3.16: Hash-and-Sign 型格子ベース署名および FALCON における署名検証関数の比較

	Gentry らの格子ベース署名 [83, Sect. 7.1]	FALCON[81, Algorithm 16]
	$\mathsf{Vrfy}(m \in \{0,1\}^*, \sigma = s, pk = A)$	$\mathsf{Vrfy}(m \in \{0,1\}^*, \sigma = (r, \mathsf{s}), pk = h, \lfloor \beta^2 \rfloor)$
1:	$\boldsymbol{t} \leftarrow H(m)$	$c \leftarrow \mathsf{HashToPoint}(r \| \| m, q, n)$
2:	**if** $\boldsymbol{t} - \boldsymbol{s}A \equiv 0 \pmod q$	$s_2 \leftarrow \mathsf{Decompress}(\mathsf{s}, 8 \cdot \mathsf{sbytelen} - 328)$
	AND $\boldsymbol{s}$ が短い **then return** accept	**if** $(s_2 = \perp)$
		return reject
		$s_1 \leftarrow c - s_2 h \bmod q$
		if $\|\|(s_1, s_2)\|\|^2 \leq \lfloor \beta^2 \rfloor$
		return accept
		else
		return reject

具体的な困難性の評価およびパラメータ設定は，SIS 問題を BKZ アルゴリズムを用いて解いた場合の Core-SVP 計算量により導出している。

表 3.17: FALCON のパラメータ [81, Table 3.3], [4, Table 8] 公開鍵，秘密鍵，署名サイズの単位はそれぞれ Byte である。

$(n, q, \sigma, \lfloor \beta^2 \rfloor)$	安全性レベル	公開鍵サイズ	秘密鍵サイズ [*12]	署名サイズ
$(512, 12289, 165.736617183, 34034726)$	レベル 1	897	7,553	666
$(1024, 12289, 168.388571447, 70265242)$	レベル 5	1,793	13,953	1,280

変種: 実装の複雑さによるサイドチャネル攻撃からの防御，セキュリティパラメータの多様性確保などを目的とした改良が多数提案されている。

特に，鍵生成と署名生成における離散 Gauss 分布生成の改良が多い。一例として，Gauss 分布生成の演算を浮動小数点から整数演算に変更した Zalcon[79]，Gauss 分布の代わりに中心二項分布とした Peregrine [145]，実装が複雑な高速フーリエサンプリングを環上の CVP アルゴリズムをベースとしたより単純なものに置き換えた Mitaka [72] などが存在する。Peregrine は韓国の耐量子計算機暗号公募 KpqC[161] へと提出されているものの，同じ秘密鍵から生成した署名に対する統計的攻撃法による実時間での鍵復元手法が知られている [100]。

また，FALCON では環の定義多項式が $\phi(x) = x^n + 1, n = 2^k$ の形に制限されていることから安全性レベル 1,5 のパラメータのみが提案されていたが，NTRU 格子をモジュール格子とすることでパラメータ設定の多様性を確保した Mod Falcon [47] も存在する。

Mitaka 内で用いられる離散 Gauss 分布生成アルゴリズムは実装が比較的単純である反面，生成された鍵および署名ベクトルのノルムが大きく鍵長と署名長が長いという欠点があった。近年では Antrag[118] が両者の中間的な手法として，FFT 表現でのサンプリングを通じて鍵生成における離散 Gauss 分布のノルムを下げ鍵長と署名長を短くする戦略を取っている。また，SOLMAE[95] も同様のサンプリング手法を用いた上で，エラーベクトルの圧縮表現などを用いて署名長を短縮する技術 [74] と組み合わせ KpqC へと提案されている。

[*12] 秘密鍵サイズは仕様書には掲載されていないが，NIST の第 3 ラウンド報告レポート [4, Sect. D] を参照した。

補足情報: 2022 年に NIST より標準化がアナウンスされ，将来的に NIST FIPS 206 (FN-DSA) として出版される予定であるが，他の格子暗号方式 (FIPS 203 および 204) と比較して発表が遅れている。これは基準となる仕様書版 [81] からの修正箇所 [131] が多いことが原因であると考えられる。

鍵生成および署名生成アルゴリズムの中で浮動小数点演算が用いられているため実行環境ごとの結果の不安定性，定数時間での実装が難しいことによるタイミング攻撃の可能性がある。対策として固定小数点を用いた実装への変更が検討されている [131, p. 13]。

また，ML-DSA と比較して beyond unforgeability [52] の性質を完全には持たないことから，署名生成におけるハッシュ値の計算方法の変更が検討されている（[131, p. 15] および [71] を参照）。

3.3.6 FrodoKEM

歴史: 2016 年の国際会議 CCS において LWE ベースの鍵共有プロトコル Frodo が Joppe Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, Douglas Stebila の 8 名の連名で公表された [33]。

2017 年 11 月の NIST PQC 標準化プロジェクトの公募に提出された LWE ベースの公開鍵暗号方式 FrodoPKE および鍵カプセル化メカニズム Frodo KEM[116] では [33] の著者 8 名から Craig Costello が抜け，Erdem Alkim, Patrick Longa, Christopher Peikert の 3 名を加えた 10 名が inventors，Karen Easterbrook, Brian LaMacchia を Additional submitters とした合計 12 名での提案となっている。

NIST PQC 標準化プロジェクトへの提出後のディスカッションを通じて修正が加えられた後に ISO 標準化に提案されている。現在の最新版は 2024 年 12 月に公表された [82] である。本節の記述は NIST PQC 標準化プロジェクトを通じてアップデートされた 2021 年 6 月の第 3 ラウンド版 [13]，および 2024 年 12 月に公表された ISO 標準への提出版 [82] に従う。

参照 URL: 開発者による公式ページ `https://frodokem.org/` およびリファレンスコード `https://github.com/Microsoft/PQCrypto-LWEKE` を参照した。

設計原理: FrodoKEM は LWE 問題を安全性の根拠とする公開鍵暗号方式であり，将来 Ring 型や Module 型の構造を持つ格子問題を用いた暗号に致命的な脆弱性が発見された場合でも安全性が確保されると期待されることを特徴としている[*13]。LWE 問題自体の単純さからパラメータ設定の小回りが利くことも長所の一つとしている。形式的には Gentry-Peikert-Vaikuntanathan の [83, Sect. 7.1]，Lindner-Peikert [101] をひな型とする dual-LWE 暗号に分類される。IND-CPA 安全な公開鍵暗号方式を構成した後に，Hofheinz ら [89] のモジュール化藤崎-岡本変換 $\mathsf{FO}^{\not\perp}$ に Bos らの [34] の修正を施した変換手法 [13, Def. 2.19] を適用し IND-CCA2 安全な KEM を構成している。

3.3.6.1 NIST PQC 第 3 ラウンド版

アルゴリズムの詳細: 表 3.18，3.19，3.20 に Lindner-Peikert[101] による格子ベース公開鍵暗号と FrodoKEM の構成の基礎となる FrodoPKE の鍵生成，暗号化，復号アルゴリズムを並置する。

パブリックパラメータは以下で与えられる。

- q: 計算の剰余環 $\mathbb{Z}_q$ を指定する。ここでは $q = 2^D$ の形とし，$D = 15, 16$ に固定される。
- $n, \overline{m}, \overline{n}$: 行列のサイズを指定する。$n$ は 8 の倍数とする。また，平文は $\overline{m} \times \overline{n} = 8 \times 8$ 行列に符号化される。
- B, ℓ: 平文行列に符号化する情報量を指定する。行列の各成分は $0, \ldots, 2^B - 1$ の整数で表現され，合計で $\ell = B \cdot \overline{m} \cdot \overline{n}$ ビットの情報を埋め込むことができる。

[*13] FrodoKEM の仕様書 [13] では構造を持たないことを “algebraically unstructured” と表現している。

- $\mathsf{len}_{\mathsf{seed}_A}, \mathsf{len}_{\mathsf{seed}_{SE}}$: 擬似ランダム行列 A, S, E を生成するためのシードとなるビット列の長さで，提案パラメータセットでは $\mathsf{len}_{\mathsf{seed}_A}$ は 128 に固定され，$\mathsf{len}_{\mathsf{seed}_{SE}}$ はセキュリティレベルに合わせて $128, 192, 256$ の値を取る。
- T_χ: SampleMatrix 関数で用いられる確率分布の表で，セキュリティレベルごとに離散 Gauss 分布からの Rényi ダイバージェンスが小さくなるように設計されている。具体的な値は [13, Table 3] を参照。

関数内で用いられるサブルーチン群を以下に記述する。

- Frodo.Gen 関数はシードとなるビット列 seed と SHAKE ハッシュ関数を用いて擬似ランダム行列 $A \in \mathbb{Z}_q^{n\times n}$ を生成する関数である。i 行目を生成する際に整数 i を 16 ビットのビット列にエンコードした $\langle i\rangle$ を用いて $\mathsf{SHAKE}(\langle i\rangle||\mathsf{seed}, 16n)$ を呼び出し，得られた $16n$ ビットを 16 ビットごとに分割することで n 個の整数 $c_0, \ldots, c_{n-1} \in \{0, \ldots, 2^{16}-1\}$ として，$A_{i,j} = c_j \mod q$ の形で各成分に振り分けている。このように関数を構成することで，各 i 行目を生成する操作がハードウェアによる並列実装に適した形となる。また，q は 2 べきの形で取られるため，各 $A_{i,j}$ の分布に偏りは生じない。
- $\mathsf{Frodo.SampleMatrix}((\boldsymbol{r}^{(0)}, \ldots, \boldsymbol{r}^{(nm-1)}), n, m, T)$ 関数は行列のサイズ $n \times m$ と (i, j) 成分の生成に用いるシード $\boldsymbol{r}^{(in+j)}$ の列，乱数の確率分布を示すテーブル T を入力とする。それぞれのシードの長さは 16 ビットに固定されている。T は整数上の中心対称な確率分布が $\Pr[|T| = t]$ のテーブルとして与えられており，シード $\boldsymbol{r}^{(in+j)}$ の先頭 15 ビットで (i, j) 成分の絶対値を，残りの 1 ビットで符号を決定しサンプリングを行う。
- Frodo.Encode, Frodo.Decode 関数は $\ell = B \cdot \overline{m} \cdot \overline{n}$ ビットの平文を $\overline{m} \times \overline{n}$ 行列に埋め込む関数とその逆演算を行う関数である。B ビットの整数 k を $\mod q$ に埋め込むため，行列の成分を $k \cdot \lfloor q/2^B \rfloor$ とする。

表 3.18: Lindner-Peikert 格子ベース暗号および FrodoKEM における鍵生成関数の比較

	Lindner-Peikert[101, Sect. 3.1] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$	FrodoKEM[13, Algorithm 9] $\mathsf{FrodoPKE.KeyGen}(1^\lambda) \to (pk, sk)$
1:	A: $n_1 \times n_2$ ランダム行列	$\mathsf{seed}_A \xleftarrow{\$} \{0,1\}^{\mathsf{len}_{\mathsf{seed}_A}}$ $A \leftarrow \mathsf{Frodo.Gen}(\mathsf{seed}_A)$
2:	S: 成分の小さい $n_2 \times \ell$ 行列	$\mathsf{seed}_{SE} \xleftarrow{\$} \{0,1\}^{\mathsf{len}_{\mathsf{seed}_{SE}}}$ //擬似乱数ビットの生成 $(\boldsymbol{r}^{(0)}, \ldots, \boldsymbol{r}^{(2n\overline{n}-1)}) \leftarrow \mathsf{SHAKE}(\mathsf{0x5F}||\mathsf{seed}_{SE}, 2n\overline{n} \cdot \mathsf{len}_\chi)$ $S^T \leftarrow \mathsf{Frodo.SampleMatrix}((\boldsymbol{r}^{(0)}, \ldots, \boldsymbol{r}^{(n\overline{n}-1)}), \overline{n}, n, T_\chi)$
3:	E: 成分の小さい $n_1 \times \ell$ 行列	$E \leftarrow \mathsf{Frodo.SampleMatrix}((\boldsymbol{r}^{(n\overline{n})}, \ldots, \boldsymbol{r}^{(2n\overline{n}-1)}), \overline{n}, n, T_\chi)$
4:	$B = AS + E$	$B = AS + E$
return	$pk = (A, B), sk = S$	$pk = (\mathsf{seed}_A, B), sk = S^T$

FrodoKEM の鍵生成関数（表 3.18 右）を説明する。鍵生成のためのシード seed_A と seed_{SE} を生成した後, Frodo.Gen 関数と Frodo.SampleMatrix 関数を用いて A, S, E を生成する。このとき，行列乗算時のメモリアクセスの順序を考えて S は転置の形で格納される。公開鍵行列 A は鍵サイズ圧縮のために成分ではなくシード seed_A の形で格納される。

FrodoKEM の暗号化関数（表 3.19 右）を説明する。seed_A から行列 A を復元した後, 暗号化用の乱数行列 S', E', E'' を擬似乱数列 $\boldsymbol{r}^{(i)}$ から生成する。擬似乱数列の生成には SHAKE ハッシュ関数を用いるが，パディング値 0x96 が鍵生成で用いられた 0x5F と異なるため鍵生成の S, E とは異なる行列が得られることに注意。残りの処理はひな型の Lindner-Peikert 暗号を行列化したものである。

FrodoKEM の復号関数（表 3.20 右）は Lindner-Peikert 暗号の復号処理を行列化したものである。

安全性とパラメータ: ベースとなる IND-CPA 安全な公開鍵暗号の安全性は判定版 LWE 問題に帰着される。実装上の

表 3.19: Lindner-Peikert 格子ベース暗号および FrodoKEM における暗号化関数の比較

	Lindner-Peikert[101, Sect. 3.1] $\mathsf{Enc}(pk=(A,B),$ $\boldsymbol{m}\in\{0,1\}^{\ell})\to\mathsf{ct}$	FrodoKEM[13, Algorithm 10] $\mathsf{FrodoPKE.Enc}(pk=(A,B),\mu\in\{0,1\}^{\ell}))\to\mathsf{ct}$
0:		$A\leftarrow\mathsf{Frodo.Gen}(\mathsf{seed}_A)$ // A の復元
1:	$\boldsymbol{s}',\boldsymbol{e}',\boldsymbol{e}''$: 成分の小さいベクトル	$\mathsf{seed}_{SE}\xleftarrow{\$}\{0,1\}^{\mathsf{len}_{\mathsf{seed}SE}}$ $(\boldsymbol{r}^{(0)},\ldots,\boldsymbol{r}^{(2n\overline{n}-1)})\leftarrow\mathsf{SHAKE}(\mathsf{0x96}\|\mathsf{seed}_{SE},$ $(2\overline{m}\cdot n+\overline{m}\cdot\overline{n})\cdot\mathsf{len}_{\chi})$ //擬似乱数ビットの生成 $S'\leftarrow\mathsf{Frodo.SampleMatrix}((\boldsymbol{r}^{(0)},\ldots,\boldsymbol{r}^{(\overline{m}\cdot n-1)}),\overline{m},n,T_{\chi})$ $E'\leftarrow\mathsf{Frodo.SampleMatrix}((\boldsymbol{r}^{(\overline{m}\cdot n)},\ldots,\boldsymbol{r}^{(2\overline{m}\cdot n-1)}),\overline{m},n,T_{\chi})$ $E''\leftarrow\mathsf{Frodo.SampleMatrix}((\boldsymbol{r}^{(2\overline{m}\cdot n)},\ldots,\boldsymbol{r}^{(2\overline{m}\cdot n+\overline{m}\cdot\overline{n}-1)}),\overline{m},\overline{n},T_{\chi})$
2:	$\boldsymbol{u}=\boldsymbol{s}'A+\boldsymbol{e}'$ $\boldsymbol{v}=\boldsymbol{s}'B+\boldsymbol{e}''+\boldsymbol{m}\cdot\left\lfloor\frac{q}{2}\right\rfloor$	$B'=S'A+E'$; $V=S'B+E''$ $C_1=B'$; $C_2=S'B+E''+\mathsf{Frodo.Encode}(\mu)$
return	$\mathsf{ct}=(\boldsymbol{u},\boldsymbol{v})$	$\mathsf{ct}=(C_1,C_2)$

表 3.20: Lindner-Peikert 格子ベース暗号および FrodoKEM における復号関数の比較

	Lindner-Peikert[101, Sect. 3.1] $\mathsf{Dec}(sk,\mathsf{ct})\to\boldsymbol{m}'$	FrodoKEM[13, Algorithm 11] $\mathsf{FrodoPKE.Dec}(sk,\mathsf{ct})\to\boldsymbol{m}'$
1:	$\overline{\boldsymbol{m}}=\boldsymbol{v}-\boldsymbol{u}S$ $m'_i=\begin{cases}0 & \lvert m_i\rvert\le\lfloor q/4\rfloor\\ 1 & \text{それ以外}\end{cases}$	$M=C_1-C_2S$ $\boldsymbol{m}'=\mathsf{Frodo.Decode}(M)$
return	$\boldsymbol{m}'=(m'_1,\ldots,m'_{\ell})$	$\boldsymbol{m}'$

効率化のため，鍵生成，暗号化処理において離散 Gauss 分布を近似した確率分布 T_{χ} を用いているが，その際の安全性の低下は Rényi ダイバージェンスを用いた議論により評価されている [13, Sect. 5.1]。n は格子の次元で，大きくとることで安全性レベルが上がるが処理コストも上がる。q は環を定義する法で，大きく取ることで平文空間も大きくなるが，格子が疎になり安全性が下がる。σ は離散 Gauss 分布の大きさを決定するパラメータで，大きくとることで安全性が上がるが，復号エラー率が上がる。

FrodoKEM のパラメータは LWE 問題の Primal 攻撃，Dual 攻撃双方での BKZ アルゴリズムを用いた計算量評価から求められている。保守的なパラメータ設定のため，Core-SVP, BKZ の計算量評価を，計算量の上界を示す既存の攻撃手法のみではなく，下界からの議論が行われていることも方式の特徴である。

変種: 行列 A の生成に SHAKE-256 ではなく，AES-128 を使ったバージョンも提案されている。AES-NI 命令を用いた Intel CPU による実装では SHAKE を用いたものよりも 2.5 倍程度高速である [13, Sect. 3.2]。

また，Encode Decode 関数に誤り訂正符号を用いて暗号文サイズを 1 割ほど削減したバージョンが提案されている [141, 140]。

補足情報: 構造を持つ格子 (structured lattice) でないという理由から NIST PQC 標準化プロジェクトの 第 3 ラウンド候補として残っていた。構造付き格子ではない他の方式では他に符号ベース暗号の BIKE,HQC や同種写像ベース暗号の SIKE があり，それらと比較するとパフォーマンスの観点から不利であったため標準化から漏れたと NIST の標

表 3.21: FrodoKEM CCA のパラメータ [13, Table 5]。σ の値は T_χ の元となる離散 Gauss 分布の標準偏差を示す。秘密鍵サイズはデカプセル化時に用いられる鍵情報の中から，公開鍵に相当するものを除いた分である。公開鍵，秘密鍵，平文，暗号文サイズの単位はそれぞれ Byte である。

(n, q, σ)	安全性レベル	公開鍵サイズ	秘密鍵サイズ	平文サイズ	暗号文サイズ
$(\ 640, 2^{15}, 2.8)$	レベル 1	9,616	10,272	16	9,720
$(\ 976, 2^{16}, 2.3)$	レベル 3	15,632	15,664	24	15,744
$(1344, 2^{16}, 1.4)$	レベル 5	21,520	21,568	32	21,632

準化レポート [4, p. 17] に述べられている。

NIST PQC 標準化プロジェクトの 第 2 ラウンドのバージョンでは，藤崎-岡本変換を行った IND-CCA2 KEM の実装において再暗号化後のチェックが定数時間ではないことから鍵復元攻撃が可能であることが示され [86]，修正されている。実装にかかわる攻撃として，ロウハンマー (Rowhammer) 攻撃による鍵復元攻撃が新たに発見された [76]。

NIST PQC 標準化プロセスの中で第 3 ラウンド候補であり，標準化には至らなかったものの，フランス，ドイツ，オランダ，チェコ等の国で耐量子計算機暗号の推奨・許容リストに入っている[*14]。開発者らは ISO での標準化を目指し，2024 年に予備提案 (Preliminary Standardization Proposal) [16, 82] を提出している。次節に詳細を述べる。

3.3.6.2 ISO 標準への予備提案版

2024 年 12 月に公開された ISO への予備提案版 [16, 82] では，NIST PQC 版からセキュリティ強化のための修正が行われている。1 つ目は multi-target security と呼ばれる，複数の公開鍵が与えられたときに攻撃者がどれかひとつの鍵に関して IND-CCA 安全性を破ることすら困難であるという要件である。通常の IND-CCA 安全性では 1 つの公開鍵に対する識別を行うのに対し，multi-target では攻撃者が自身に都合の良い鍵を選ぶことができるため，攻撃者に有利な設定となる。

2 つ目は multi-ciphertext security と呼ばれる，1 つの公開鍵で暗号化された複数の暗号文が与えられたときに，攻撃者がどれか 1 つの暗号文に関する安全性を破ることが困難であるという要件である。これら 2 つの要件に対応するため，鍵カプセル化時に用いられる疑似乱数 seed_{SE} の長さに変更が加えられ，新たな乱数列 salt が導入されている。

表 3.22 から 3.24 に NIST 版 FrodoKEM と ISO 版 FrodoKEM の比較を示す。データの型変換を行う Pack, Unpack 関数の呼び出しは本質的でないため省略した。なお，2023 年，2024 年に公開された著者らの ISO 向け仕様書 [16, 82] では NIST 版（前節の表 3.18 から 3.20 で示したもの）を eFrodoKEM，新たなバージョンを FrodoKEM としているため表記に注意。

表 3.25 に各安全性レベルごとの乱数 seed と seed_{SE} の長さをまとめる。$\mathsf{len}_{\mathsf{salt}} = 0$ の場合，salt は空となり NIST 第 3 ラウンド版と ISO 版は同じプロトコルとなる。seed を追加したことにより，暗号文長はレベル 1,3,5 の場合表 3.21 と比較して 32,48,64Bytes 長くなる。

3.3.7 NewHope

歴史: NewHope の最初のバージョンは 2016 年に国際会議 USENIX Security において Erdem Alkim，Léo Ducas，Thomas Pöppelmann，Peter Schwabe により鍵共有プロトコルとして発表された [12]。また，直後に reconciliation によるエラー訂正プロセスを省略し簡略化した NewHope-Simple[11] が ePrint Archive において発表されている。

[*14] 各国の PQC 推奨・許容暗号リストの状況は第 1 章も参照。

表 3.22: FrodoKEM の NIST PQC 標準化プロジェクト 第 3 ラウンド版 [13, Algorithm 11] と ISO 版 [82, Sec. 8.1] の鍵生成関数の比較。両者ともに同じ関数であるが，表 3.25 に示すようにパラメータ $\mathsf{len}_{\mathsf{seed}_{SE}}$ の値が異なる。FrodoKEM.KeyGen 関数は表 3.18 の右側の関数内で乱数 $\mathsf{seed}_A, \mathsf{seed}_{SE}$ をランダムとせず，引数の値として実行した結果を示す。

NIST 版 FrodoKEM[13, Algorithm 12]	ISO 版 FrodoKEM [82, Sec. 8-1]
$\mathsf{KeyGen}(1^\lambda) \rightarrow (pk', sk')$	$\mathsf{KeyGen}(1^\lambda) \rightarrow (pk', sk')$
乱数を生成 $s, \mathsf{seed}_{SE}, z \xleftarrow{\$} \{0,1\}^{\mathsf{len}_s + \mathsf{len}_{\mathsf{seed}_{SE}} + \mathsf{len}_z}$	乱数を生成 $s, \mathsf{seed}_{SE}, z \xleftarrow{\$} \{0,1\}^{\mathsf{len}_s + \mathsf{len}_{\mathsf{seed}_{SE}} + \mathsf{len}_z}$
$\mathsf{seed}_A \leftarrow \mathsf{SHAKE}(z, \mathsf{len}_{\mathsf{seed}_A})$	$\mathsf{seed}_A \leftarrow \mathsf{SHAKE}(z, \mathsf{len}_{\mathsf{seed}_A})$
$(pk, sk) \leftarrow \mathsf{FrodoKEM.KeyGen}(\mathsf{seed}_A, \mathsf{seed}_{SE})$	$(pk, sk) \leftarrow \mathsf{FrodoKEM.KeyGen}(\mathsf{seed}_A, \mathsf{seed}_{SE})$
$//pk = (\mathsf{seed}_\mathsf{A}, B)$ および $sk = S^T$	$//pk = (\mathsf{seed}_\mathsf{A}, B)$ および $sk = S^T$
$pkh \leftarrow \mathsf{SHAKE}(\mathsf{seed}_A \|\| B, \mathsf{len}_{pkh})$	$pkh \leftarrow \mathsf{SHAKE}(\mathsf{seed}_A \|\| B, \mathsf{len}_{pkh})$
$pk' = (\mathsf{seed}_\mathsf{A}, B), sk' = (s, \mathsf{seed}_A, b, S^T, pkh)$	$pk' = (\mathsf{seed}_\mathsf{A}, B), sk' = (s, \mathsf{seed}_A, b, S^T, pkh)$

表 3.23: FrodoKEM の NIST PQC 標準化プロジェクト 第 3 ラウンド版 [13, Algorithm 12] と ISO 版 [82, Sec. 8.2] の鍵カプセル化関数の比較。$\mathsf{FrodoPKE.Enc}(\mathsf{seed}_{SE}, pk', \mu)$ 関数は表 3.19 右側の関数内で乱数 seed_{SE} をランダムとせず，引数の値を用いて実行した結果を示す。ISO 版では新たに salt が追加されている。

NIST 版 FrodoKEM[13, Algorithm 12]	ISO 版 FrodoKEM [82, Sec. 8-1]
$\mathsf{Encaps}(pk') \rightarrow (\mathsf{ct}, ss)$	$\mathsf{Encaps}(pk') \rightarrow (\mathsf{ct}, ss)$
$\mu \xleftarrow{\$} \{0,1\}^{\mathsf{len}_\mu}$ // ランダムな鍵を生成	$\mu \xleftarrow{\$} \{0,1\}^{\mathsf{len}_\mu}$ // ランダムな鍵を生成
	$\mathsf{salt} \xleftarrow{\$} \{0,1\}^{\mathsf{len}_{\mathsf{salt}}}$ // ランダムな salt 値を生成
$pkh \leftarrow \mathsf{SHAKE}(pk, \mathsf{len}_{pkh})$	$pkh \leftarrow \mathsf{SHAKE}(pk, \mathsf{len}_{pkh})$
$(\mathsf{seed}_{SE}, k) \leftarrow \mathsf{SHAKE}(pkh, \mu, \mathsf{len}_{\mathsf{seed}_{SE}} + \mathsf{len}_k)$	$(\mathsf{seed}_{SE}, k) \leftarrow \mathsf{SHAKE}(pkh, \mu, \mathsf{salt}, \mathsf{len}_{\mathsf{seed}_{SE}} + \mathsf{len}_k)$
$\mathsf{FrodoPKE.Enc}(\mathsf{seed}_{SE}, pk', \mu) \rightarrow (C_1, C_2)$	$\mathsf{FrodoPKE.Enc}(\mathsf{seed}_{SE}, pk', \mu) \rightarrow (C_1, C_2)$
$\mathsf{SHAKE}(C_1 \|\| C_2 \|\| k, \mathsf{len}_{ss}) \rightarrow ss$	$\mathsf{SHAKE}(C_1 \|\| C_2 \|\| \mathsf{salt} \|\| k, \mathsf{len}_{ss}) \rightarrow ss$
$\mathsf{ct} = (C_1, C_2), ss$	$\mathsf{ct} = (C_1, C_2, \mathsf{salt}), ss$

2017 年 11 月の NIST PQC 標準化プロジェクトの公募に応募された Version 1.0[127] は新たに Roberto Avanzi, Joppe Bos，Antonio de la Piedra，Douglas Stebila の 4 人が加わった合計 8 人での提案とし，[11] をベースとして公開鍵暗号方式を構成している。NIST PQC 標準化プロジェクトへ提出後のディスカッションを通じて修正が加えられ，現在の最新版は 2020 年 4 月に公表された Version 1.1[14] である。

NIST PQC 標準化プロジェクトの 第 2 ラウンドに提出された Version 1.02[128] では，Martin R. Albrecht, Emmanuela Orsini，Valery Osheter，Kenneth G. Paterson，Guy Peer，Nigel P. Smart の 6 人が Contributor として列挙されている。

本節の記述は最新版の仕様書 [14] に従う。

参照 URL: 開発者による公式ページ `https://newhopecrypto.org/` および GitHub 上のリファレンス実装 `https://github.com/newhopecrypto/newhope` を参照した。

設計原理: NewHope は環 $\mathbb{Z}[x]/(x^n + 1), n = 2^k$ 上の Ring-LWE 問題の計算困難性を安全性の根拠とする公開鍵暗号方式であり，形式的には Gentry-Peikert-Vaikuntanathan の [83, Sect. 7.1]，Lindner-Peikert [101] をひな型とする dual-LWE 暗号に分類される。ベースとなる暗号方式におけるベクトルと行列の演算を多項式環の要素に置き換え

表 3.24: FrodoKEM の NIST PQC 標準化プロジェクト 第 3 ラウンド版 [13, Algorithm 13] と ISO 版 [82, Sec. 8.3] のデカプセル化関数の比較。$\mathsf{FrodoPKE.Enc}(\mathsf{seed}'_{SE}, pk', \mu')$ 関数は表 3.19 右側の関数内で乱数 seed_{SE} をランダムとせず，引数の値を用いて実行した結果を示す。ISO 版では新たに salt が追加されている。

NIST 版 FrodoKEM[13, Algorithm 14]	ISO 版 FrodoKEM [82, Sec. 8-2]
$\mathsf{Decaps}(sk', \mathsf{ct}) \rightarrow ss$	$\mathsf{Decaps}(sk', \mathsf{ct}) \rightarrow ss$
$\mu' \leftarrow \mathsf{FrodoPKE.Dec}(sk = S^T, \mathsf{ct})$	$\mu' \leftarrow \mathsf{FrodoPKE.Dec}(sk = S^T, \mathsf{ct})$
$(\mathsf{seed}'_{SE}, k') \leftarrow \mathsf{SHAKE}(pkh, \mu, \mathsf{len}_{\mathsf{seed}_{SE}} + \mathsf{len}_k)$	$(\mathsf{seed}'_{SE}, k') \leftarrow \mathsf{SHAKE}(pkh, \mu', \mathsf{salt}, \mathsf{len}_{\mathsf{seed}_{SE}} + \mathsf{len}_k)$
$\mathsf{FrodoPKE.Enc}(\mathsf{seed}'_{SE}, pk', \mu')) \rightarrow (C'_1, C'_2) =: \mathsf{ct}'$	$\mathsf{FrodoPKE.Enc}(\mathsf{seed}'_{SE}, pk', \mu)) \rightarrow (C'_1, C'_2) =: \mathsf{ct}'$
if $(\mathsf{ct} = \mathsf{ct}')$ $\bar{k} \leftarrow k'$ **else** $\bar{k} \leftarrow s$	**if** $(\mathsf{ct} = \mathsf{ct}')$ $\bar{k} \leftarrow k'$ **else** $\bar{k} \leftarrow s$
$\mathsf{SHAKE}(C_1 \| C_2 \| \bar{k}, \mathsf{len}_{ss}) \rightarrow ss$	$\mathsf{SHAKE}(C_1 \| C_2 \| \mathsf{salt} \| \bar{k}, \mathsf{len}_{ss}) \rightarrow ss$

表 3.25: NIST 版と ISO 版におけるパラメータ $\mathsf{len}_{\mathsf{salt}}$ と $\mathsf{len}_{\mathsf{seed}_{SE}}$ の違い。セキュリティ強化のため salt が追加され，seed_{SE} の長さが変更されている。

	eFrodoKEM (NIST 第 3 ラウンド版)		FrodoKEM (ISO 版)	
安全性レベル	$\mathsf{len}_{\mathsf{salt}}$	$\mathsf{len}_{\mathsf{seed}_{SE}}$	$\mathsf{len}_{\mathsf{salt}}$	$\mathsf{len}_{\mathsf{seed}_{SE}}$
レベル 1	0	128	256	256
レベル 3	0	192	384	384
レベル 5	0	256	512	512

IND-CPA 安全な公開鍵暗号方式を提案している。その際，数論変換を用いた乗算などの実装テクニックを用いて処理を高速化している。環の定義多項式を $x^n + 1, n = 2^k$ の形としている点も高速化に寄与しているが，その一方でパラメータ選択の自由度に制限があり，NIST PQC 標準化プロジェクトの提案方式では安全性レベル 1 およびレベル 5 のパラメータセットのみが提案されている。

IND-CPA 安全な公開鍵暗号から IND-CCA 安全な KEM の構成には [89] のモジュール化された藤崎-岡本変換 $\mathsf{QFO}^{\not\perp}_m$ を用いているが，その際に公開鍵暗号 CRYSTALS-Kyber（本報告書の 3.3.3 節も参照）の IND-CCA 安全な KEM の構成 [34, Sect. 4] に用いられた手法を取り入れ微修正を施している。結果として，構成された KEM が ROM, QROM の双方のモデルにおいて IND-CCA 安全であることが保証されている。

アルゴリズムの詳細: 表 3.26，3.27，3.28 に Lindner-Peikert[101] による格子ベース公開鍵暗号と NewHope の鍵生成，暗号化，復号アルゴリズムを並置する。

パブリックパラメータは以下で与えられる。

- n, q: 演算を行う環 $R_q := \mathbb{Z}_q[x]/(x^n + 1)$ を定義する。特に指定のない場合には多項式の係数は自動的に区間 $[-q/2, q/2)$ 内に収められるものとする。高速数論変換のため，n は 2 のべき乗の形であり，さらにアルゴリズム中で用いられる ω, γ が存在するために q は $q \equiv 1 \pmod{2n}$ を満たす素数として選ばれる。NewHope のパラメータ設定では $n = 512, 1024, q = 12289$ が選ばれている。
- k: ノイズの大きさを設定する。
- ω, γ: 数論変換で用いる。$\mathbb{Z}_q^\times$ における 1 の原始 n 乗根を ω, その平方根を $\gamma := \sqrt{\omega} \mod q$ とする。NewHope のパラメータ設定では $n = 512$ に対して $(\omega, \gamma) = (3, 10968)$，$n = 1024$ に対して $(\omega, \gamma) = (49, 7)$ が取られている。

関数内で用いられるサブルーチン群を以下に記述する。

- $\mathsf{Sample}(\mathsf{seed}, \mathsf{nonce})$ 関数は，各係数を平均ゼロに調整した二項分布 ψ_8 から独立にサンプリングした n 次多項式を 32Bytes の seed と非負整数値 nonce から生成する。自然数 k に対して，ψ_k の出力は独立にサンプリングした $2k$ 個のビット $b_i, b'_i \xleftarrow{\$} \{0,1\}$ $(k = 1, \ldots, k)$ に対する $\sum_{i=1}^{k}(b_i - b'_i)$ として定義される。

- $\mathsf{PolyBitRev}(a \in R_q)$: 高速数論変換を用いた乗算の場合，結果のインデックス順序が入れ替わるため配列の要素 $c[i]$ を $x^{\mathsf{BitRev}}(i)$ として解釈しなければならない。ただし，ビット順序反転は $h = \log_2(n), i = \sum_{j=0}^{h-1} b_j 2^j$ と 2 進数展開したときに，$\mathsf{BitRev}(i) := \sum_{j=0}^{h-1} b_j 2^{h-j-1}$ 計算される。関数は多項式 $a(x) = \sum_{i=0}^{n-1} a_i x^i$ に対して，指数部分をビット順序反転した多項式 $\sum_{i=0}^{n-1} a_i x^{\mathsf{BitRev}(i)}$ を出力する。

- $\mathsf{NTT}(a), \mathsf{NTT}^{-1}(\hat{a})$: R_q の多項式同士の乗算を高速化するため，数論変換 (Number Theoretic Transform: NTT)[14, p. 7-9] を用い，鍵と暗号文の処理を極力 NTT 空間で行う工夫がなされている。パラメータ n, q, ω, γ を固定したとき，多項式 $a(x) = \sum_{i=0}^{n-1} a_i x^i \in R_q$ の数論変換

$$\hat{a} = \mathsf{NTT}(a) := \sum_{i=0}^{n-1} \hat{a}_i x^i,\ \hat{a}_i := \sum_{j=0}^{n-1} \gamma^j a_j \omega^{ij} \pmod{q}$$

および逆変換を

$$a = \mathsf{NTT}^{-1}(\hat{a}) := \sum_{i=0}^{n-1} a_i x^i,\ a_i := \left(n^{-1}\gamma^{-i} \sum_{j=0}^{n-1} \hat{a}_j \omega^{-ij} \right) \pmod{q}$$

とする。これらは線形変換であるので，$a, b \in R_q$ に対して $\mathsf{NTT}(a) + \mathsf{NTT}(b) = \mathsf{NTT}(a+b)$ 等の性質がなりたつ。また，$a * b := \sum_{i=0}^{n-1} c_i x^i, c_i = \sum_{j=0}^{n-1} a_j b_{i-j \mod n} \mod q$ は $a * b = \mathsf{NTT}^{-1}(\mathsf{NTT}(a) \circ \mathsf{NTT}(b))$ を満たす。ただし，記号 $\circ$ は多項式の係数同士の積を取ることを表す。
$a * b$ を定義式通りに計算すると mod q での演算が $O(n^2)$ 回必要であるのに対して，高速数論変換を用いた方法では $O(n\log n)$ 回の演算で可能である。

数論変換を用いた高速乗算ではどこかのタイミングで添え字のビット順序を反転する必要がある。単純な IND-CPA PKE の実装における最適なのみを考えるのであればこのような置換は必要ないが，IND-CCA KEM のデカプセル化処理内での再暗号化まで含めて実装を最適化した結果，NewHope では最初の $\mathsf{KeyGen}, \mathsf{Enc}$ 関数の中で置換が行われている [14, p. 8]。

なお，仕様書 [14] には多項式とバイト列の相互変換を行う関数 $\mathsf{EncodePK}, \mathsf{EncodePolynomial}, \mathsf{EncodeC}, \mathsf{Compress}, \mathsf{DecodePK}, \mathsf{DecodePolynomial}, \mathsf{DecodeC}, \mathsf{Decompress}$ が定義されているが，何れもデータの再配置を行う関数であり方式を説明する上では本質的ではないため省略した。

NewHope の鍵生成関数（表 3.26 右）を説明する。ステップ 1 では 32Bytes(= 256bits) の乱数のシードを SHAKE-256 ハッシュ関数を用いて 64Bytes の擬似乱数列 z を生成し，それを前半の $\mathsf{z}[0:31]$ と後半の $\mathsf{z}[32:63]$ に分割する。Lindner-Peikert 型暗号（左側）におけるランダム行列 A の生成に対応して，$\mathsf{GenA}(\cdot)$ 関数は 32Bytes の列をシードとして，ランダムな R_q の元を生成する。各係数が一様独立に Z_q の元からサンプリングされる。実装は [14,

表 3.26: Lindner-Peikert 格子ベース暗号および NewHope における鍵生成関数の比較

	Lindner-Peikert[101, Sect. 3.1] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$	NewHope[14, Algorithm 1] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$
1:	A: $n_1 \times n_2$ ランダム行列	$\mathsf{seed} \xleftarrow{\$} \{0,1,\ldots,255\}^{32}, z = \mathsf{SHAKE256}(64, \mathsf{seed})$ //32Bytes の seed を 64Bytes に伸長 $\hat{a} = \mathsf{GenA}(\mathsf{z}[0:31]) \in R_q$
2:	S: 成分の小さい $n_2 \times \ell$ 行列	$s = \mathsf{PolyBitRev}(\mathsf{Sample}(\mathsf{z}[32:63], 0)) \in R_d$; $\hat{s} = \mathsf{NTT}(s)$
3:	E: 成分の小さい $n_1 \times \ell$ 行列	$e = \mathsf{PolyBitRev}(\mathsf{Sample}(\mathsf{z}[32:63], 1)) \in R_d$; $\hat{e} = \mathsf{NTT}(e)$
4:	$B = E - AS$	$\hat{b} = \hat{a} \circ \hat{s} + \hat{e}$ //$b = \mathsf{NTT}(a * s + e)$
return	$pk = (A, B), sk = S$	$pk = (\hat{a}, \hat{b}), sk = \hat{s}$

Algorithm 5] を参照。出力された $\hat{a}$ がランダムな多項式 a の数論変換であることは, ランダム多項式の数論変換がまたランダム多項式となることから従う。

ステップ 2,3 ではそれぞれ係数の小さい多項式 s, e の数論変換を計算する。ステップ 1 で生成した擬似乱数の後半 $z[32:63]$ をシードに用いて小さい値を係数に持つ多項式のサンプリングを行い，数論変換のためのビット順序の反転処理をしたものを s，その数論変換を $\hat{s}$ とする。$e, \hat{e}$ についても同様。

ステップ 4 では数論変換後の多項式から公開鍵 $\hat{b}$ を計算する。数論変換の性質より，これは $a * b + e$ の NTT 表現となる。

表 3.27: Lindner-Peikert 格子ベース暗号および NewHope における暗号化関数の比較

	Lindner-Peikert[101, Sect. 3.1] $\mathsf{Enc}(pk = (A, B), \boldsymbol{m} \in \{0,1\}^\ell) \to \mathsf{ct}$	NewHope[14, Algorithm 1] $\mathsf{Enc}(pk = (A, B), M \in \{0,1,\ldots,255\}^{32}) \to \mathsf{ct}$
1:	$\boldsymbol{t}, \boldsymbol{e}', \boldsymbol{e}''$: 成分の小さいベクトル	$\mathsf{coin} \xleftarrow{\$} \{0,1,\ldots,255\}^{32}$ // ランダムシード $s' = \mathsf{PolyBitRev}(\mathsf{Sample}(\mathsf{coin}, 0)) \in R_d$; $\hat{t} = \mathsf{NTT}(s')$ $e' = \mathsf{PolyBitRev}(\mathsf{Sample}(\mathsf{coin}, 1)) \in R_d$ $e'' = \mathsf{Sample}(\mathsf{coin}, 2) \in R_d$
2:	$\boldsymbol{u} = \boldsymbol{t}A + \boldsymbol{e}'$ $\boldsymbol{v} = \boldsymbol{t}B + \boldsymbol{e}'' + \boldsymbol{m} \cdot \lfloor \frac{q}{2} \rfloor$	$\hat{u} = \hat{a} \circ \hat{t} + \mathsf{NTT}(e')$ $v' = \mathsf{NTT}^{-1}(\hat{b} \circ \hat{t}) + e'' + \mathsf{Encode}(M)$
return	$\mathsf{ct} = (\boldsymbol{u}, \boldsymbol{v})$	$\mathsf{ct} = (\hat{u}, v')$

NewHope の暗号化関数（表 3.27 右）を説明する。暗号化のためのランダム多項式 $\hat{t}, e', e''$ を生成するため，鍵生成のステップ 2-3 と同様の処理を行う。$\hat{t}$ は数論変換後の形式であるが，e', e'' は通常の形式のまま用いる。

Encode 関数は 32Bytes(=256bits) の平文を n 次多項式の係数として埋め込む。NewHope のパラメータでは $n = 512, 1024$ が用いられるため，1bit の情報が複数箇所に埋め込まれることになる。具体的には平文の $i$bit 目を b_i, 多項式の j 次の係数を v_j としたときに $j = 0, \ldots, n-1$ に対して

$$v_j = \begin{cases} 0 & (b_{j \bmod 256} = 0) \\ \lfloor \frac{q}{2} \rfloor & (b_{j \bmod 256} = 1) \end{cases}$$

とする。

NewHope の復号関数（表 3.35 右）を説明する。健全性の証明より, $v - \mathsf{NTT}^{-1}(\hat{u} \circ \hat{s}) = v - u * s$ が $\mathsf{Encode}(M)$ と小さいノイズ和であることが示されるため, Decode 関数はノイズの除去と平文 M' の復元を同時に行う [14, Algorithm 11]。

表 3.28: Lindner-Peikert 格子ベース暗号および NewHope における復号関数の比較

	Lindner-Peikert[101, Sect. 3.1] $\mathsf{Dec}(sk, \mathsf{ct}) \to m'$	NewHope[14, Algorithm 1] $\mathsf{Dec}(sk, \mathsf{ct}) \to M'$
1:	$\overline{\boldsymbol{m}} = \boldsymbol{v} + \boldsymbol{u}S$ $m'_i = \begin{cases} 0 & \lvert m_i \rvert \leq \lfloor q/4 \rceil \\ 1 & \text{それ以外} \end{cases}$	$\overline{m} = v - \mathsf{NTT}^{-1}(\hat{u} \circ \hat{s})$ $M' = \mathsf{Decode}(\overline{m})$
return	$\boldsymbol{m}' = (m'_1, \ldots, m'_\ell)$	M'

多項式 $\overline{m}$ の係数の中で，$\overline{m}_{i+256k}$ $(k = 0, \ldots, n/256 - 1)$ の中にビット M'_i の情報が埋め込まれているため，それらを多数決で決定する。具体的には，$\sum_{k=0}^{n/256-1} |\overline{m}_{i+256k} - (q-1)/2|$ が $M'_i = 0$ の場合には $(n/256) \cdot (q-1)/2 \approx (n/512)q$ に近く，$M'_i = 1$ の場合には 0 に近い値を取るため，和から $(n/1024)q$ を引いた後に符号を見ることでビット列の復元が完了する。

安全性とパラメータ: ベースとなる IND-CPA 安全な公開鍵暗号の安全性は環 $Z_q[x]/(x^n + 1)$ 上の判定版 LWE 問題に量子帰着されることが示されている。実装上の効率化のため，NewHope では鍵生成，暗号化の際に離散 Gauss 分布の替わりに中心を 0 とした二項分布を用いているが，その分の安全性の低下は [14, Theorem 4.1] で Rényi ダイバージェンスを用いた議論により評価されている。

Ring-LWE 問題の具体的な困難性の評価には，LWE 問題に対する Primal 攻撃，Dual 攻撃双方での，BKZ アルゴリズムを用いた必要ブロックサイズから導き出した CoreSVP 計算量による評価を用いている。

表 3.29: NewHope CPA-KEM, CCA-KEM のパラメータ [14, Table 2, 3]。2 番目のパラメータは NIST 耐量子計算機暗号 Call for proposals[121] の基準でレベル 5 であると主張されているが，表 [14, Table 3] では 233-bit 安全性となっている。公開鍵，秘密鍵，平文，暗号文サイズの単位はそれぞれ Byte である。

(n, q, k, γ)	安全性 レベル	公開鍵 サイズ	秘密鍵サイズ (鍵カプセル化後)	平文 サイズ	暗号文サイズ (鍵カプセル化後)
$(512, 12289, 8, 10968)$	レベル 1	928	869 (1,888)	32	1,088 (1,120)
$(1024, 12289, 8, 7)$	レベル 5	1,824	1,792 (3,680)	32	2,176 (2,208)

変種: 鍵共有を目的とした USENIX 版 [12]，および reconciliation によるエラー訂正プロセスを省略し簡略化した NewHope-Simple[11] が存在する。

補足情報: NIST PQC 標準化プロジェクトの 第 2 ラウンドの選定レポート [113, p. 16] によると，NewHope と CRYSTALS-Kyber の間で比較が行われた。双方ともに dual-LWE 形式の構造を持つ格子上で考え，数論変換を用いた高速化を行うという方針で設計されている。Core-SVP ベースの困難性評価では双方とも同程度の強度であったが，実装時のベンチマークの結果は CRYSTALS-Kyber の方が若干良かった。また，安全性の根拠に用いている問題が Ring-LWE と Module-LWE という違いがあり，パラメータ設定の自由度において不利であったようである。

3.3.8 NTRU

歴史: NTRU 暗号方式自体の歴史は長く，1996 年に国際会議 CRYPTO の Rump Session において発表され，その後 1998 年に国際会議 ANTS において発表された論文 [87] が方式の源流となる。本節では，歴史的な NTRU ではなく，NIST PQC 標準化プロジェクトの第 3 ラウンド Finalists に選定された公開鍵暗号方式 NTRU について説明する[*15]。

2017 年 11 月の NIST PQC 標準化プロジェクトの公募に提出された 2 件の方式, Andreas Hülsing, Joost Rijneveld, John M. Schanck, Peter Schwabe らにより提案された NTRU-HRSS-KEM[93] と，Cong Chen, Jeffrey Hoffstein, William Whyte, Zhenfei Zhang らにより提案された NTRUEncrypt[42] が Round 2 に進む際にマージされ名称が NTRU と変更，Round 3 にかけてさらに修正が加えられたものが現在の方式となる。第 2 ラウンド提出版は [93] と [42] の著者 8 名に Oussama Danba を加えた合計 9 名での提案，第 3 ラウンド提出版はさらに Tsunekazu Saito, Keita Xagawa，Takashi Yamakawa の 3 名を加えた合計 12 名での提案となった。

現在の最新版は 2020 年 9 月に公表された仕様書 [43] である。本節の記述はこの仕様書に従う。

参照 URL: 開発者による公式ページ `https://ntru.org/` を参照した。

設計原理: NTRU は NTRU 格子上の計算困難問題に安全性の根拠を置く公開鍵暗号方式である。具体的には鍵復元攻撃の困難性が NTRU 格子上の短いベクトルを求める問題，平文復元攻撃の困難性が，ターゲットベクトルに近い NTRU 格子上の点を求める問題[*16]として捉えられる。

ベースとなる公開鍵暗号方式は ANTS バージョン [87] の NTRU と比較して，鍵 f, g のサンプリング空間の変更，メッセージ多項式のマスキング手法の変更，暗号化関数の脱乱択化[*17] などの改良が行われている。暗号化関数が決定的であるためベースの方式自体は IND-CPA 安全性を持たないが，この性質を用いることで IND-CCA2 KEM の構成において，単純な藤崎-岡本変換を用いた構成と比較してハッシュ関数の呼び出し回数を削減することが可能である。

提案ではベースとなる公開鍵暗号方式の OW-CPA 安全性を格子問題の困難性に還元した後に，Saito ら [139] において提案された implicit rejection の導入による NTRU-HRSS-KEM の改良の構成をベースとして，デカプセル化時の再暗号化処理のスキップによる IND-CCA2 KEM の構成を行っている。最終的な方式の IND-CCA2 安全性は適当な仮定を置くことで ROM, QROM モデルにおいて元の方式の OW-CPA 安全性に帰着される。ただし標準的な仮定においては QROM モデルでの還元はタイトではなく，いくつかの non-standard な仮定を置くことでタイトになる [4, p. 39]。

アルゴリズムの詳細:

以下，表 3.31，3.32，3.33 に ANTS 版の NTRU と NIST PQC 標準化プロジェクト第 3 ラウンド提出版の NTRU の公開鍵暗号方式を並置して解説する。

パブリックパラメータは以下で与えられる。仕様書 [43] には NTRU-HPS と NTRU-HRSS の 2 系統のパラメータセットが存在し，それぞれ $\boldsymbol{f}, \boldsymbol{g}$ のサンプリング空間，Lift 関数の構成などが異なる。

- n を定義多項式の次数，$\Phi_1 = x - 1, \Phi_n = (x^n - 1)/(x - 1)$ を円分多項式，p, q を素数の法とする。多項式の次数 n は素数で，$2, 3$ が $\mathbb{Z}_n$ の原始元となるように選ぶ。
- 素数 q と多項式 $F(x)$ に対して，記号 $\mathbb{Z}[x]/(q, F)$ で剰余環 $\mathbb{Z}_q[x]/(F(x))$ を表す。特に $R/q, S/q$ はそれぞれ $\mathbb{Z}_q[x]/(\Phi_1\Phi_n), \mathbb{Z}_q[x]/(\Phi_n)$ を定義する。n の取り方より，$S/2, S/3$ は有限体となるため，0 以外の元に常に逆元

[*15] 第 3 ラウンド Alternative Candidates の中の NTRU Prime とは異なる方式であることに注意。

[*16] 最短・最近ベクトル問題や限界距離復号問題と似ているが，設定が少し異なるため，このように表現する。

[*17] 一般に，脱乱択化 (derandomizing) とはアルゴリズム中で乱数を用いるサブルーチンを，ほぼ同等の性能を保ったままに乱数を用いない決定的なサブルーチンに置き換えることを示す。元々の NTRU では暗号化関数内で平文をマスキングする多項式をランダムに生成していたが，提案バージョンでは多項式を関数の引数とし関数自体は決定的なものとなっている。

が存在することになる。

- 集合 $\mathcal{T}$, を係数が $\{0, \pm 1\}$ の多項式で次数が $n-2$ 以下のものの全体とし，

$$\mathcal{T}' := \left\{ \boldsymbol{v} = \sum_{i=0}^{n-2} v_i x^i \in \mathcal{T} : \sum_{i=0}^{n-3} v_i v_{i+1} \geq 0 \right\}$$

とする。

- $\mathcal{L}(d_1, d_2)$ は ANTS 版 NTRU のサンプリング空間を定義するために用いられる。次数 $n-1$ 以下の多項式で d_1 個の係数が $+1$，d_2 個の係数が -1，残りは 0 であるものの集合とする。

暗号アルゴリズムの中で用いられるサブルーチン群は以下で与えられる。

- 多項式 $\boldsymbol{a}$ に対して，関数 $\underline{\mathsf{S3}}(\boldsymbol{a})$ を $\boldsymbol{b} \equiv \boldsymbol{a} \bmod (3, \Phi_n)$ を満たすもので，次数 $n-2$ 以下かつ係数が $\{0, \pm 1\}$ となるものとする。これを $S/3$ の代表元とする。
- $\mathsf{Lift}(\boldsymbol{m})$ 関数は，メッセージ $\boldsymbol{m}$ のマスキングに用いられる。暗号文を $\boldsymbol{c} = \boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m}$ にって計算する NTRU 系の暗号の場合，$\mathcal{L}_f, \mathcal{L}_g$ の取り方によっては IND-CPA 安全性を持たない可能性がある [43, p. 22]。そのため，メッセージ多項式 m を一度別の形にマスキングする必要がある。NTRUEncrypt[42] では，ランダム多項式 t を足しこむことで実現していたのだが，NTRU-HRSS[93] ではこの機能を Lift を用いて実現している。Lift 関数は $\underline{\mathsf{S3}}$ 関数を用いて実現され高速実装が可能であり，しかもマスキングのための多項式 t をサンプリングするという手間がなくなるため，より実装が単純・高速となる。そのため，第 2 ラウンドでのマージにおいて NTRU-HRSS のアイディアが残った形である。

以下の表 3.30 に鍵多項式 f, g，暗号化に用いるランダム多項式 r，平文多項式 m の空間と Lift 関数の違いをまとめる。$\underline{\mathsf{S3}}(\boldsymbol{m}/(\Phi_1))$ の高速計算法は文献 [92, Append. B] に掲載されている。ANTS 版 NTRU では Lift 関数は明示されていないが，[43, Sect. 1.3.1] によると $\underline{\mathsf{S3}}(\mathsf{Lift}(\boldsymbol{m})) = \boldsymbol{m}$ を満たす単射 $\mathcal{L}_m \to \mathbb{Z}[x]$ として解釈できる。

表 3.30: 方式ごとのサンプリング空間，Lift 関数の違い

	ANTS 版 NTRU [43, Sect. 1.3.1]	NTRU-HPS [43, Sect. 1.3.2]	NTRU-HRSS [43, Sect. 1.3.3]
$\mathcal{L}_f$	$\mathcal{L}(d_f, d_f - 1)$	$\mathcal{T}$	$\mathcal{T}_+$
$\mathcal{L}_g$	$\mathcal{L}(d_f, d_f - 1)$	$\mathcal{T}(q/8 - 2)$	$\{\Phi_1 \cdot \boldsymbol{v} : \boldsymbol{v} \in \mathcal{T}_+\}$
$\mathcal{L}_r$	$\{p \cdot \phi : \phi \in \mathcal{L}(d, d)\}$	$\mathcal{T}$	$\mathcal{T}$
$\mathcal{L}_m$	係数が $[-p/2, p/2]$ に含まれる多項式の集合	$\mathcal{T}(q/8 - 2)$	$\mathcal{T}$
$\mathsf{Lift}(\boldsymbol{m})$ 関数	-	$\underline{\mathsf{S3}}(\boldsymbol{m})$	$\Phi_1 \cdot \underline{\mathsf{S3}}(\boldsymbol{m}/\Phi_1)$

NTRU の鍵生成関数（3.31）の詳細を記述する。最初に Sample_fg 関数により $(\boldsymbol{f}, \boldsymbol{g})$ を $\mathcal{L}_f \times \mathcal{L}_g$ から一様ランダムにサンプリングする。ANTS 版では $\mathcal{L}_f$ からランダムにサンプリングを行い，$\bmod (2, \Phi_1 \Phi_n), \bmod (3, \Phi_1 \Phi_n)$ の双方で可逆であることを確認し，可逆でない場合にはサンプリングをやり直していた。一方で，NTRU-HPS,NTRU-HRSS では構成から可逆性が保証されているため，可逆性検査は行わない。

秘密鍵の中に $\boldsymbol{h}_q$ が含まれるのは復号関数の中で暗号化に用いた多項式 $\boldsymbol{r}$ を復元するためである。

表 3.32 に暗号化関数を記述する。この部分はオリジナルの ANTS 版 NTRU とほぼ同様であるが，暗号化に用いるランダム多項式 $\boldsymbol{r}$ が関数の入力として明示されている点，平文多項式の表現を Lift 関数によって変えている点が異なる。ANTS 版 NTRU では暗号化時に乱数として生成された $\boldsymbol{r}$ を平文の一部として扱うことで，暗号化関数が決定的なものとなる。これにより，IND-CCA2 KEM を構成する際のタイトな還元を実現している [139]。

表 3.31: ANTS 版 NTRU および NIST PQC 版 NTRU における鍵生成関数の比較

	ANTS NTRU [87, Sect. 1.2] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$	NIST PQC NTRU [43, Figure 9] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$
1:	$\boldsymbol{f} \leftarrow \mathsf{Sample\_f}()$ // $\Phi_1\Phi_n$ の中で可逆な元をサンプリングする $\boldsymbol{g} \leftarrow \mathsf{Sample\_g}()$	$(\boldsymbol{f}, \boldsymbol{g}) \leftarrow \mathsf{Sample\_fg}()$ $\boldsymbol{f}_q \leftarrow (1/\boldsymbol{f}) \bmod (q, \Phi_n)$
2:	$\boldsymbol{h} \leftarrow (3\boldsymbol{g}/\boldsymbol{f}) \bmod (q, \Phi_1\Phi_n)$	$\boldsymbol{h} \leftarrow (3\boldsymbol{g} \cdot \boldsymbol{f}_q) \bmod (q, \Phi_1\Phi_n)$ $\boldsymbol{h}_q \leftarrow (1/\boldsymbol{h}) \bmod (q, \Phi_n)$
3:	$\boldsymbol{f}_p \leftarrow (1/\boldsymbol{f}) \bmod (3, \Phi_1\Phi_n)$	$\boldsymbol{f}_p \leftarrow (1/\boldsymbol{f}) \bmod (3, \Phi_n)$
return	$pk = \boldsymbol{h}, sk = (\boldsymbol{f}, \boldsymbol{f}_p)$	$pk = \boldsymbol{h}, sk = (\boldsymbol{f}, \boldsymbol{f}_p, \boldsymbol{h}_q)$

表 3.32: ANTS 版 NTRU および NIST PQC 版 NTRU における暗号化関数の比較

	ANTS NTRU [87, Sect. 1.3] $\mathsf{Enc}(pk = h, \boldsymbol{m} \in \mathcal{L}_m) \to \boldsymbol{c}$	NIST PQC NTRU [43, Figure 9] $\mathsf{Enc}(pk = h, \boldsymbol{m} \in \mathcal{L}_m; \boldsymbol{r}) \to \boldsymbol{c}$
1:	$\boldsymbol{r} \leftarrow \mathsf{Sample\_r}()$	
2:	$\boldsymbol{c} \leftarrow (\boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m}) \bmod (q, \Phi_1\Phi_n)$	$\boldsymbol{m}' \leftarrow \mathsf{Lift}(\boldsymbol{m})$ $\boldsymbol{c} \leftarrow (\boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m}') \bmod (q, \Phi_1\Phi_n)$
return	$\boldsymbol{c}$	$\boldsymbol{c}$

表 3.33: ANTS 版 NTRU および NIST PQC 版 NTRU における復号関数の比較

	ANTS NTRU [87, Sect. 1.4] $\mathsf{Dec}(sk = (\boldsymbol{f}, \boldsymbol{f}_p), \boldsymbol{c}) \to \boldsymbol{m}'$	NIST PQC NTRU [43, Figure 9] $\mathsf{Dec}(sk = (\boldsymbol{f}, \boldsymbol{f}_p, \boldsymbol{h}_q), \boldsymbol{c}) \to (\boldsymbol{r}, \boldsymbol{m}, \mathsf{flag})$
0:		**if** $\boldsymbol{c} \not\equiv 0 \pmod{(q, \Phi_1)}$ **return** $(0, 0, 1)$
1:	$\boldsymbol{a} \leftarrow (\boldsymbol{c} \cdot \boldsymbol{f}) \bmod (q, \Phi_1\Phi_n)$	$\boldsymbol{a} \leftarrow (\boldsymbol{c} \cdot \boldsymbol{f}) \bmod (q, \Phi_1\Phi_n)$
2:	$\boldsymbol{m}' \leftarrow (\boldsymbol{a} \cdot \boldsymbol{f}_p) \bmod (3, \Phi_1\Phi_n)$	$\boldsymbol{m} \leftarrow (\boldsymbol{a} \cdot \boldsymbol{f}_p) \bmod (3, \Phi_n)$ $\boldsymbol{m}' \leftarrow \mathsf{Lift}(\boldsymbol{m})$ $\boldsymbol{r} \leftarrow ((\boldsymbol{c} - \boldsymbol{m}') \cdot \boldsymbol{h}_q) \bmod (q, \Phi_n)$ **if** $(\boldsymbol{r}, \boldsymbol{m}) \in \mathcal{L}_r \times \mathcal{L}_m$ **return** $(\boldsymbol{r}, \boldsymbol{m}, 0)$ **else return** $(0, 0, 1)$
return	$\boldsymbol{m}'$	$(\boldsymbol{r}, \boldsymbol{m}, \mathsf{flag})$

表 3.33 の復号関数の説明を行う。暗号化関数が脱乱択化されたことで，復号関数は ANTS 版のものとは大きく異なるものになる。出力が平文の $(\boldsymbol{r}, \boldsymbol{m})$ の他に，復号が失敗したかどうかを示すフラグ flag を返す。このフラグがデカプセル化時の implicit rejection に用いられる。

暗号化時の $\mathcal{L}_g$ の取り方，Lift 関数の性質から正しく作られた暗号文の場合ステップ 0 の $\boldsymbol{c} \equiv 0 \mod (q, \Phi_1)$ が保証される。$\mathsf{flag} = 0$ の場合にこの等式が保証されることは，デカプセル化関数内での再暗号化スキップのために必要である。ステップ 2 で復元された $\boldsymbol{m}$ が $S/3$ の代表元となっていることは保証できないため，Lift 関数を用いて $\boldsymbol{m}'$ の復元を行い，それを用いて $\boldsymbol{r}$ を復元する。$(\boldsymbol{r}, \boldsymbol{m})$ が正常な平文空間 $\mathcal{L}_r \times \mathcal{L}_m$ に含まれているならば $\mathsf{flag} = 0$ をセットして復号結果を返し，そうでなければ失敗として $\boldsymbol{r} = \boldsymbol{m} = 0$ とし，失敗フラグを立てて値を返す。

安全性とパラメータ: ベースとなる公開鍵暗号方式の OW-CPA 安全性の具体的な困難性を評価するために鍵復元攻撃と平文復元攻撃が考えられている。公開鍵から秘密鍵を復元する問題は，環の定義多項式と公開鍵多項式によって定義される格子内において，秘密鍵 $(\boldsymbol{f}, \boldsymbol{g})$ に対応する短いベクトル[*18]を発見する問題として捉えることができる。また，暗号文から平文を復元する問題も $\boldsymbol{h}_q$ から定義される格子内で，$(0, \boldsymbol{c})$ に近いベクトルを探索する問題として捉えられるため，最近ベクトル問題の埋め込みによりこちらも格子内の短いベクトルを求める問題として定式化することが可能である。

以上により，暗号方式のパラメータ設定には与えられた格子内の短いベクトルを BKZ アルゴリズムを用いて発見するために必要なブロックサイズ β を求め，具体的な計算量は Core-SVP による評価を行っている。

Core-SVP の計算量評価には，篩アルゴリズムで用いる巨大なメモリ空間にアクセスするためのコストを定数と仮定した non-local model およびそうでないと仮定した local model の双方を用いた個別の評価 [43, Sect 5.3] を行っている。

表 3.34: NTRU のパラメータ [43, Sect. 1.6, 3.2]. 公開鍵，秘密鍵，平文，暗号文サイズの単位はそれぞれ Byte である。

パラメータ名	(n, p, q)	安全性レベル		公開鍵	秘密鍵	平文	暗号文
		non-local	local	サイズ	サイズ	サイズ	サイズ
ntruhps2048509	$(509, 3, 2048)$	-	レベル 1	699	903(935)	204	699
ntruhps2048677	$(677, 3, 2048)$	レベル 1	レベル 3	930	1,202(1,234)	272	930
ntruhps4096821	$(821, 3, 4096)$	レベル 3	レベル 5	1,230	1,558(1,590)	328	1,230
ntruhrss701	$(701, 3, 8192)$	レベル 1	レベル 3	1,138	1,418(1,450)	280	1,138

秘密鍵サイズの括弧内は KEM のもので，32Bytes の乱数列 s の分大きくなる。

3.3.9 SABER

歴史: SABER は NIST PQC 標準化プロジェクトの公募への応募方式の一つ 2017 年 11 月に Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren の 4 名により公表され，その後同著者により 2018 年 5 月に国際会議 AFRICACRYPT において査読付き国際会議論文として発表された [54]。

NIST PQC 標準化プロジェクトの 第 2 ラウンド提出時において安全性証明に関わる微修正が行われ，第 3 ラウンドからは新たに Andrea Basso, Jose Maria Bermudo Mera, Michiel Van Beirendonck の 3 名が加わり，開発者は合計 7 名となっている。現在の最新版は第 3 ラウンド Finalists に提出された [28] であり，以下の記述はこの仕様書に従う。

参照 URL: 開発者による公式ページ `https://www.esat.kuleuven.be/cosic/pqcrypto/saber/` およびリファレンス実装 `https://github.com/KULeuven-COSIC/SABER` を参照した。

設計原理: SABER は Module-LWR 問題を安全性の根拠とする公開鍵暗号方式であり，LWE 暗号におけるノイズ付加計算をラウンディング演算に置き換えた暗号方式を構成のひな型としている。基本となる方式に対して Module 化と実装上の改良のための修正を行い IND-CPA 安全な暗号方式を構成，藤崎-岡本変換により IND-CCA2 KEM としたものである。仕様書の設計原理 [28, Sect. 4] の項には Regev 暗号 [133] の "LWR version" であると記述されている一方で，Second PQC Standardization Conference の発表スライド [53, p. 5] では Lindner-Peikert 型（dual-LWE 型）

[*18] 多項式環や格子の構成等の違いにより秘密鍵と最短ベクトルが対応しない，最短ベクトルが複数存在するなどの状況があるため，短いベクトルという表現としている。

[101] の構成を原型としている。数式の比較から，どちらも原型と捉えることが可能であり，本報告書では仕様書に従い LWE 型に分類する。

LWR 型暗号方式の利点として，LWE 暗号の実装時に必要とされる離散 Gauss 分布等からのサンプリング計算の回避が挙げられる。また，処理を行う際の法 p, q を 2 のべき乗とすることでラウンディング処理がビット列の部分的なコピーのみで完了すること，同様に鍵となる行列 $A \in R_q^{\ell \times \ell}$ の生成が乱数生成ルーチンからのビット列のコピーのみで完了することから，高速処理が可能であるという特徴がある。

一方で，2 のべき乗の形で p, q をとることにより，数論変換を用いた多項式同士の乗算が単純に適用できなくなるという欠点があるが，仕様書 [28, Sect.4] によると，SABER で用いられる多項式は 256 次であり，通常の乗算方法を用いても NTT によるものと処理時間に大きな差は無いと主張されている。また，[48] のように一度大きな剰余空間で乗算を計算した後に $\mathbb{Z}_p, \mathbb{Z}_q$ の世界に引き戻す実装テクニックも開発されており，多項式の乗算による効率の低下に関しての大きな問題は無いと考えられる。

IND-CPA 安全な公開鍵暗号から IND-CCA2 KEM の構成には Hofheinz ら [89] による藤崎-岡本変換の変種を用いており，ROM,QROM モデルの双方で安全であることが示されている [28, Sect. 6]。公開鍵暗号から KEM の具体的な構成は [28, Algorithm 4-6] 参照。

アルゴリズムの詳細: 表 3.35，3.36，3.37 に Regev 暗号の LWR 版と SABER（IND-CPA 安全な基本バージョン）の鍵生成，暗号化，復号アルゴリズムを並置する。

パブリックパラメータは以下で与えられる。

- n: 環を定義するための多項式 $x^n + 1$ の次数であり，全てのパラメータセットで $n = 256$ とする。
- p, q, T: ラウンディングの大きさを決定するパラメータ。全て 2 のべき乗の形で，$\epsilon_q = \log_2(q), \epsilon_p = \log_2(p), \epsilon_T = \log_2(T)$ とし，$\epsilon_q > \epsilon_p > \epsilon_T$ とする。つまり $T|p|q$ の関係がある。計算を行う環は $R_q := \mathbb{Z}_q[x]/(x^n + 1)$ とし，R_q の元を成分とする。
- l: モジュール格子のランクである。ベクトル，行列をそれぞれ $R_q^{l\times 1}, R_q^{l\times l}$ 等で表現する。
- 平文空間は $R_2 := \mathbb{Z}_2[x]/(x^n + 1)$ であり，256bits の情報を格納できる。

表 3.35: Regev 暗号の LWR 版および SABER における鍵生成関数の比較

	Regev 暗号の LWR 版 [53, p. 7] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$	SABER[28, Algorithm 1] $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$
1:	A: ランダム行列	$\mathsf{seed}_A \xleftarrow{\$} \{0,1\}^{256}$; $A \leftarrow \mathsf{gen}(\mathsf{seed}_A) \in R_q^{l\times l}$
2:	$\boldsymbol{s}$: 短いランダムベクトル	$\mathsf{r} \xleftarrow{\$} \{0,1\}^{256}$; $\boldsymbol{s} \leftarrow \beta_\mu(R_q^{l\times 1}; \mathsf{r})$
3:	$\boldsymbol{b} = \lfloor A\boldsymbol{s} \rceil_{p/q}$	$\boldsymbol{b} = ((A^T\boldsymbol{s} + \boldsymbol{h}) \bmod q) \gg (\epsilon_q - \epsilon_p) \in R_p^{l\times 1}$
return	$pk = (A, \boldsymbol{b}), sk = \boldsymbol{s}$	$pk = (\mathsf{seed}_A, \boldsymbol{b}), sk = \boldsymbol{s}$

表 3.35 の左側は Regev による LWE 暗号 [133] におけるノイズ付加関数 $A\boldsymbol{s} + \boldsymbol{e}$ をラウンディング関数 $\lfloor A\boldsymbol{s} \rceil_{p/q}$ へと置き換えたものである。ただし，実数 x に対して $\lfloor x \rceil_{p/q} := \lfloor x \cdot (p/q) \rceil$ とし，ベクトル，行列に対しては成分ごとにその操作を行うものとする。また，自然数 a, ϵ に対して $a \gg \epsilon$ は右シフト演算の結果 $a \cdot 2^{-\epsilon}$ の整数部分を返すものであり，$b \in R_q$ に対しては $b \gg \epsilon$ は係数ごとの右シフトを，ベクトル $\boldsymbol{b} \in R_p^{l\times 1}$ に関しても成分ごとの右シフトを行った結果とする。

表 3.35 の右側，SABER の鍵生成関数を説明する。ステップ 1 の gen 関数は 256bits の列をシードとして，R_q を成分とした擬似ランダムな $l \times l$ 行列を生成する。各成分は R_q 内の一様分布とする。

ステップ 2 の $\beta_\mu(R_q^{l\times 1}; r)$ はビット列 r をシードとして R_q 成分 l 次元の擬似ランダムベクトルを出力する関数であ

る。ここで，μ は偶数であるとし，各成分の多項式は係数を独立に，パラメータ μ の二項分布の出力から平均値 $\mu/2$ を引いた値をサンプリングしたものとする。

ステップ 3 の公開鍵ベクトル $\boldsymbol{b}$ の生成は $A^T\boldsymbol{s}$ のラウンディングによるものだが，p, q がともに 2 のべき乗 $\varepsilon_p, \varepsilon_q$ であることから p/q を掛けた後のラウンディング処理が

$$\lfloor x \rceil_{p/q} := \left\lfloor \frac{p}{q} x \right\rceil = \left\lfloor \frac{p}{q} x + \frac{1}{2} \right\rfloor = \left\lfloor \frac{p}{q} \left(x + \frac{q}{2p} \right) \right\rfloor = \left\lfloor \left(x + 2^{\varepsilon_q - \varepsilon_p - 1} \right) 2^{\varepsilon_p - \varepsilon_q} \right\rfloor \tag{3.6}$$

と表現され，入力 x に定数 $h = 2^{\varepsilon_q - \varepsilon_p - 1}$ を加えた後，$2^{\varepsilon_p - \varepsilon_q}$ による乗算と切り捨て処理が $(\varepsilon_q - \varepsilon_p)$ ビットの右シフトで実現される。表中の $\boldsymbol{h}$ は，係数が全て h の多項式を成分として持つ $R_q^{l \times 1}$ のベクトルを示す。

出力される公開鍵の形式はデータ量削減のため $(A, \boldsymbol{b})$ の替わりに，行列 A を生成するためのシードを用いて $pk = (\mathsf{seed}_A, \boldsymbol{b})$ としている。

表 3.36: Regev 暗号の LWR 版および SABER における暗号化関数の比較

	Regev 暗号の LWR 版 [53, p. 7] $\mathsf{Enc}(pk = (A, b), m \in \{0,1\}) \to \mathsf{ct}$	SABER[28, Algorithm 2] $\mathsf{Enc}(pk = (\mathsf{seed}_A, b), m \in R_2; r) \to \mathsf{ct}$
0:		$A \leftarrow \mathsf{gen}(\mathsf{seed}_A) \in R_q^{l \times l}$ //行列 A の復元
1:	$\boldsymbol{s}'$: 短いランダムベクトル	$\boldsymbol{s}' \leftarrow \beta_\mu(R_q^{l \times 1}; r)$
2:	$\boldsymbol{b}' = \lfloor A^T \boldsymbol{s}' \rceil_{p,q}$	$\boldsymbol{b}' = ((A\boldsymbol{s}' + \boldsymbol{h}) \bmod q) \gg (\epsilon_q - \epsilon_p) \in R_p^{\ell \times 1}$
3:	$c_m = \left\lfloor \boldsymbol{b}^T \boldsymbol{s}' - m \cdot \left\lfloor \frac{p}{2} \right\rceil \right\rceil_{T/p}$	$v' = \boldsymbol{b}^T (\boldsymbol{s}' \bmod p) \in R_p$ $c_m = (v' + h_1 - 2^{\epsilon_p - 1} m \bmod p) \gg (\epsilon_p - \epsilon_T) \in R_T$
return	$\mathsf{ct} = (c_m, \boldsymbol{b}')$	$\mathsf{ct} = (c_m, \boldsymbol{b}')$

表 3.36 の右側，SABER の暗号化関数を説明する。平文空間は $R_2 = \mathbb{Z}_2[x]/(x^n + 1)$ で，$n = 256$bits の情報を格納する。最初にステップ 0 として，暗号化処理に用いる行列 A を seed_A から復元する。次にステップ 1 では暗号化用のランダムベクトル $\boldsymbol{s}'$ を種となるビット列 r を用いて生成するが，与えられていなかった場合には 256bits の乱数列を擬似乱数生成器によって生成する。Enc 関数に種となるビット列が与えられているのは後にこの関数が藤崎-岡本変換を用いた IND-CCA KEM の構成に使われるためである。$\boldsymbol{s}'$ を生成する関数 β_μ，およびステップ 2 での $A\boldsymbol{s}'$ のラウンディングによる $\boldsymbol{b}'$ の生成は鍵生成と同様である。ステップ 3 では $\boldsymbol{b}, \boldsymbol{s}'$ を用いて暗号化を行うが，$(\cdots) \gg (\epsilon_p - \epsilon_T)$ の計算は (3.6) 式における p/q の役割を T/p に置き換えたものである。$2^{\epsilon_p - 1} m$ は係数が 0 もしくは $2^{\epsilon_p - 1}$ の多項式 $(\in R_p)$ となるため，復号関数内で $v' + h_1$ のラウンディングを鍵を用いて小さいノイズに変換することで平文が復元可能となる。

表 3.37: Regev 暗号の LWR 版および SABER における復号関数の比較

	Regev 暗号の LWR 版 [53, p. 7] $\mathsf{Dec}(sk = \boldsymbol{s}, \mathsf{ct} = (c_m, \boldsymbol{b}')) \to m'$	SABER[28, Algorithm 3] $\mathsf{Dec}(sk = \boldsymbol{s}, \mathsf{ct} = (c_m, \boldsymbol{b}')) \to m'$
1:	$v = (\boldsymbol{b}')^T \boldsymbol{s}$	$v = (\boldsymbol{b}')^T (\boldsymbol{s} \bmod p) \in R_p$
2:	$m' = \left\lfloor \frac{2}{q} \cdot \left(\frac{T}{p} v - c_m \right) \right\rceil$	$m' = ((v - 2^{\epsilon_p - \epsilon_r} c_m + h_2) \bmod p) \gg (\epsilon_p - 1) \in R_2$
return	m'	m'

表 3.37 の右側，SABER の復号関数では (3.6) 式と同様の原理により $\left\lfloor \frac{2}{q} \cdot \left(\frac{T}{p} v - c_m \right) \right\rceil$ が計算される。$h_2 \in R_q$ は全ての係数が $2^{\epsilon_p - 2} - 2^{\epsilon_p - \epsilon_T - 1} + 2^{\epsilon_q - \epsilon_p - 1}$ である多項式とする。

安全性とパラメータ: ベースとなる IND-CPA 安全な公開鍵暗号の安全性は環 $\mathbb{Z}[x]/(x^{256}+1)$ 上の Module-LWR 問題に帰着されることが示されている。Module-LWR 問題の具体的な困難性評価は，Albrecht らによる LWE 問題，NTRU 問題の困難性シミュレータ [7] を LWR 問題向けに修正したものを用いている。

SABER の安全性を決めるパラメータは l, n, q, p, T, μ の 6 個である。l が Module-LWR 問題のランク，n が多項式の次数であり，大きいほど安全性が上がるがラウンディング時のノイズの蓄積により復号エラー率が上がる。法 q を大きくとることで Module-LWR の格子体積が大きくなり安全性が下がるがラウンディング時のノイズに強くなるため復号エラー率が下がる。ラウンディングパラメータ p, T を大きくとることでラウンディング時のノイズに相当するものが大きくなり，安全性が上がると同時に復号エラー率も上がる。秘密鍵サンプリングの範囲を指定する μ は大きくとることで秘密鍵ベクトルのサンプリング空間が広がり安全性が上がるが，ラウンディング時のノイズが大きくなり復号エラー率が上がる。

表 3.38: SABER のパラメータ [28, Table 1]。公開鍵，秘密鍵，平文，暗号文サイズの単位はそれぞれ Byte である。

(l, n, q, p, T, μ)	安全性レベル	公開鍵暗号			鍵カプセル化後		
		公開鍵サイズ	秘密鍵サイズ	暗号文サイズ	公開鍵サイズ	秘密鍵サイズ	暗号文サイズ
$(2, 256, 2^{13}, 2^{10}, 2^3, 10)$	レベル 1	672	832(256)	736	672	1,568(992)	736
$(3, 256, 2^{13}, 2^{10}, 2^4, 8)$	レベル 3	992	1,248(288)	1,088	992	2,304(1,344)	1,088
$(4, 256, 2^{13}, 2^{10}, 2^6, 6)$	レベル 5	1,312	1,664(384)	1,472	1,312	3,040(1,760)	1,472

秘密鍵サイズは一つ目の数字が圧縮前，括弧内の数字が圧縮後のものを示す。秘密鍵ベクトル $\boldsymbol{s} \in R_q$ の各成分が $[-\mu/2, \mu/2]$ の範囲であることから，$\lceil \log_2 q \rceil$ bits の整数として保存するのではなく，下位 $\lceil \log_2 \mu \rceil$ bits のみを保存することで圧縮できる。

3.4 格子に基づく暗号技術に関するまとめ

格子に基づく暗号技術は，LWE 問題，Ring-LWE 問題，NTRU 問題を安全性の根拠とする方式をはじめ，これまで数多く提案されており，米国 NIST PQC 標準化プロジェクトで提案された暗号技術としては最も多くの暗号がこのカテゴリーに分類されている。

この米国 NIST PQC 標準化プロジェクトを通じて 2022 年 7 月に CRYSTALS-Kyber が標準的な暗号方式として，CRYSTALS-Dilithium および FALCON が標準的な署名方式として選定され，CRYSTALS-Kyber と CRYSTALS-Dilithium については，2024 年 8 月に FIPS 203, FIPS 204 として公開されている [120, 119]。また，CRYSTALS-Kyber と CRYSTALS-Dilithium は 2022 年 9 月に米国国家安全保障局の Commercial National Security Algorithm Suite 2.0 (CNSA2.0) にも選定されている [1]。NIST PQC 標準化プロジェクトの選考プロセスから漏れた方式の中でも，米国以外の公的機関において推奨暗号とされているものが存在する。一例として，FrodoKEM が 2020 年 8 月よりドイツ情報セキュリティ庁 (BSI) の推奨暗号に [143]，2022 年 1 月にはオランダ通信・安全委員会 (NLNCSA) により最も安全な暗号の例として推奨されている [15]。Google 社の Chrome ブラウザには，TLS レイヤーの性能試験目的で搭載された耐量子計算機暗号プロトコル CECPQ1[37] および CECPQ2[132] にそれぞれ NewHope の USENIX 発表バージョン [12] と NTRU が実装されていたが，2023 年 1 月現在ではともに削除されている。IBM 製テープドライブのプロトタイプとして，CRYSTALS-Kyber と CRYSTALS-Dilithium の組み合わせにより暗号化を行うものが制作されている [98]。DNS サーバの一種である PowerDNS において，耐量子計算機性を実現する署名として FALCON のテスト用の実装が行われている [85]。オープンソースライブラリへの導入として，WireGuard VPN protocol への

SABER の実装 [91]，WolfSSL への CRYSTALS-Kyber，FALCON の実装 [156]，OpenSSH への Streamlined NTRU Prime の実装 [122] などが存在する他，Open Quantum Safe (OQS) プロジェクトによる liboqs ライブラリには暗号化・鍵交換の方式として CRYSTALS-Kyber，NTRU，SABER，FALCON，FrodoKEM，NTRU-Prime が，署名方式として CRYSTALS-Dilithium と FALCON が実装されている [138]。このように格子に基づく暗号技術の社会実装が徐々に進みつつある。特に，標準化が先行する CRYSTALS-Kyber，CRYSTALS-Dilithium に対するサイドチャネル攻撃とその対策としてマスキング実装が検討されている [146, 153, 49]。

格子に基づく暗号技術の安全性の根拠となる問題としては，先に挙げた LWE 問題，Ring-LWE 問題，NTRU 問題以外にも Compact LWE 問題，Module-LWE 問題，LWR 問題，BDD 問題，SIS 問題他，多くのバリエーションが存在している。一般的な格子問題を解く手法としては，LLL アルゴリズム，BKZ アルゴリズムなどの基底簡約アルゴリズムや，篩型のアルゴリズムが集中的に研究されている。格子に関する計算問題の間数多くの帰着関係が知られており，それらを用いて計算問題の困難性評価が行われている。例えば，LWE 問題は 3.1.1.3 節に挙げたような uSVP 問題や BDD 問題に変換する手法が知られている。

SVP や LWE/NTRU などの格子問題の解析やそれらの求解アルゴリズムに関する最新研究については [36, 30, 77, 102, 137, 112, 55, 150, 58, 114, 130, 41, 100, 32, 51] を参照。近年，新しい格子問題として格子同型問題 [70] が提案された。（格子同型問題の性質については [29] を参照。）また，格子同型問題の困難性を安全性の根拠とする署名方式 HAWK[66] は，NIST PQC 標準化プロジェクトにおける署名方式の追加公募において，格子に基づく方式の中で第 2 ラウンドにおいて進むことが許された方式である（2024 年 10 月時点）。さらに，量子紛失 LWE サンプリング [59] や，格子問題に対する量子アルゴリズムに関する研究 [45, 50] も近年進展している。

格子問題の困難性をベースとした暗号方式で最初のものは，Ajtai[2] により 1996 年に行われた，SIS 問題が格子問題の最悪時と同等かそれ以上に困難であることの証明およびそれを用いた暗号学的ハッシュ関数の構成である。また，1997 年には Ajtai と Dwork[3] により，unique SVP の最悪困難性を安全性の根拠とした公開鍵暗号が提案されている。この公開鍵暗号方式は翌年，Nguyen らによる解読実験 [117] により必要なパラメータが長大となり実用的でないことが明らかにされたものの，その後の格子に基づく暗号構成の基礎となっている。

1996 年に Hoffstein らによって提案された NTRU 暗号 [87][*19] は，発表当初安全性証明が付けられておらず，攻撃と修正が繰り返されていたが，2011 年 Stehlé ら [147] により，IND-CPA 安全性が Ring-LWE 問題に帰着可能な方式が示された。一方で，2016 年には subfield attack[6] のような体の構造を使って格子の次元を圧縮する攻撃も提案されており，暗号の構成のためには次元や法の大きさだけでなく，環・体の構造にも注意を払う必要がある。NTRU 格子上の署名方式のサイズ改良 [74]・トラップドア生成 [73] や，NTRU に対する鍵ミスマッチ攻撃の改良 [103]・NTRU 格子の簡約 [23] に関する最新の研究がある。

2005 年に Regev[133] により提案された LWE 問題は，論文発表と同時にそれを暗号の安全性根拠として保障する重要な三つの性質が示された。一つは問題の average-case to worst case reduction，つまりパラメータを固定した際，問題の (秘密ベクトル $\mathbf{s}$ に関する) 平均的な計算量が，最悪計算量 (難しいインスタンスを生成するような $\mathbf{s}$ の集合に対する計算量) と多項式倍の違いしか無いことであり，残りの二つは判定 LWE と探索 LWE の等価性，および量子アルゴリズムによる困難な格子問題への還元である。これらの定理を組み合わせることにより，Regev 自身により提案された公開鍵暗号を解読することが平均的に難しいことが示され，その後の様々な LWE ベース暗号の構成の基礎なった。LWE 格子問題への還元に関して，2013 年には古典計算機による還元も示されている [38]。

LWE 問題の欠点である鍵サイズの大きさを改善するため，2010 年には Lyubashevsky ら [107, 108] により Ring-LWE 問題が，2015 年には Langlois ら [97] により Module-LWE 問題が公開鍵暗号と同時に提案され，LWE 問題における関係と類似の，解読の平均的な困難さが証明されている。一方で，これらの変種とオリジナルの LWE 問題との関係性は自明ではなく，同程度の難しさを持つかどうかは未解決問題である。一般的に Ring(Module)-LWE 問題

[*19] 文献上は 1998 年の国際会議 ANTS だが，初出は CRYPTO1996 の Rump Session である。

のインスタンスは LWE 問題のインスタンスとして書きなおすことができるため，LWE 問題は Ring(Module)-LWE 問題よりも困難であるという関係は自明であるが，逆の関係は知られていない。法 q が大きい場合には，Ring-LWE は Module-LWE よりも困難であることが知られている [8]。（Ring/Module-LWE 問題の理論解析の最新研究について [154] を参照。）

実装時の問題として，離散 Gauss 分布を正確に生成することは難しいことが挙げられる。ノイズをある整数区間から一様分布として取った場合でも，格子問題へと量子帰着が可能であることが 2013 年に Döttling ら [62] により示された。この方向性の研究として，Bai ら [22] により提案された，理想的な Gauss 分布を用いた暗号方式とそれを近似的な分布に置き換えた方式の間での安全性の低下を Rényi ダイバージェンスを用いて議論するものがある。

格子に基づく暗号技術は，耐量子計算機暗号としてだけでなく，完全準同型暗号や多重署名などの高機能な暗号方式に応用する研究も数多くある [31, 35, 125, 155, 61, 160, 123, 109, 44, 96, 60, 88, 110, 111]。

また，格子問題の計算機による具体的な求解に関して，2016 年より暗号解読コンテスト LWE Challenge[56] が開催されている。3.1 節に，2024 年 11 月現在の状況について記載した。特に 3.3 節で示された各暗号方式のパラメータから見ると，解が得られている値からは，大きな隔たりがみられる。格子に基づく暗号技術は，各方式毎にパラメータ設定手法に対する制約が異なっていることから，解読コンテストのサイズに基づく解読到達レベルを，具体的な暗号方式の安全性の根拠とすることは，難しいところではあるものの，古典計算機での解読困難性を測る上での検討の一つに値すると考えられる。（最新の BKZ の改良や LWE の解読計算量見積もりについては [152, 157] を参照）

格子に基づく暗号技術の安全性の根拠となる問題は，古典計算機・量子計算機のいずれにおいても現時点で効率的な解読手法は見つかっていないが，格子に基づく暗号技術は未だ研究途上にあり，今後も研究の進捗を注視する必要がある。

第 3 章の参照文献

[1] National Security Agency. Announcing the Commercial National Security Algorithm Suite 2.0. https://media.defense.gov/2022/Sep/07/2003071834/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS_.PDF. 2022-09. (2024-12-06 閲覧).

[2] M. Ajtai. Generating Hard Instances of Lattice Problems (Extended Abstract). STOC. ACM, 1996, pp. 99–108.

[3] M. Ajtai, Cynthia Dwork. A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence. STOC. ACM, 1997, pp. 284–293.

[4] G. Alagic et al. Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process. NIST IR 8413, https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8413-upd1.pdf. 2022-07.

[5] M. Albrecht, L. Ducas. Lattice attacks on NTRU and LWE: A history of refinements. London Mathematical Society Lecture Notes 469. Cambridge University Press, 2021, pp. 15–40. Chapter 2.

[6] M. R. Albrecht, S. Bai, L. Ducas. A Subfield Lattice Attack on Overstretched NTRU Assumptions – Cryptanalysis of Some FHE and Graded Encoding Schemes. CRYPTO (1). Vol. 9814. Lecture Notes in Computer Science. Springer, 2016, pp. 153–178.

[7] M. R. Albrecht, B. R. Curtis, A. Deo, A. Davidson, R. Player, E. W. Postlethwaite, F. Virdia, T. Wunderer. Estimate All the {LWE, NTRU} Schemes! SCN. Vol. 11035. Lecture Notes in Computer Science. Springer, 2018, pp. 351–367.

[8] M. R. Albrecht, A. Deo. Large Modulus Ring-LWE $\geq$ Module-LWE. ASIACRYPT (1). Vol. 10624. Lecture Notes in Computer Science. Springer, 2017, pp. 267–296.

[9] M. R. Albrecht, L. Ducas, G. Herold, E. Kirshanova, E. W. Postlethwaite, M. Stevens. The General Sieve Kernel and New Records in Lattice Reduction. EUROCRYPT (2). Vol. 11477. Lecture Notes in Computer Science. Springer, 2019, pp. 717–746.

[10] M. R. Albrecht, F. Göpfert, F. Virdia, T. Wunderer. Revisiting the Expected Cost of Solving uSVP and Applications to LWE. ASIACRYPT (1). Vol. 10624. Lecture Notes in Computer Science. Springer, 2017, pp. 297–322.

[11] E. Alkim, L. Ducas, T. Pöppelmann, P. Schwabe. NewHope without reconciliation. (2016). https://eprint.iacr.org/2016/1157.

[12] E. Alkim, L. Ducas, T. Pöppelmann, P. Schwabe. Post-quantum Key Exchange - A New Hope. USENIX Security Symposium. USENIX Association, 2016, pp. 327–343.

[13] E. Alkim et al. FrodoKEM – Learning with errors key encapsulation. Algorithm specifications and supporting documentation (June 4, 2021). https://frodokem.org/files/FrodoKEM-specification-20210604.pdf. 2021-06. (2024-03-04 閲覧).

[14] E. Alkim et al. NewHope algorithm specifications and supporting documentation Version 1.1 (Updated April 10, 2020). https://newhopecrypto.org/data/NewHope_2020_04_10.pdf. 2020-04. (2024-03-04 閲覧).

[15] General intelligence and security service. Prepare for the threat of quantum computers. https://english.aivd.nl/publications/publications/2022/01/18/prepare-for-the-threat-of-quantumcomputers. 2022-01. (2024-03-04 閲覧).

[16] Annex on FrodoKEM updates. https://frodokem.org/files/FrodoKEM-annex-20230418.pdf. 2024-04. (2024-12-01 閲覧).

[17] R. Avanzi et al. CRYSTALS-Kyber – Algorithm specifications and supporting documentation. https://pq-crystals.org/kyber/data/kyber-specification.pdf. 2017-11. (2024-03-04 閲覧).

[18] R. Avanzi et al. CRYSTALS-Kyber – Algorithm specifications and supporting documentation (version 2.0). https://pq-crystals.org/kyber/data/kyber-specification-round2.pdf. 2019-04. (2024-03-04 閲覧).

[19] R. Avanzi et al. CRYSTALS-Kyber – Algorithm specifications and supporting documentation (version 3.02). https://pq-crystals.org/kyber/data/kyber-specification-round3-20210804.pdf. 2021-08. (2024-03-04 閲覧).

[20] S. Bai, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, D. Stehlé. CRYSTALS-Dilithium algorithm specifications and supporting documentation (Version 3.1). https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf. 2021-02. (2024-03-04 閲覧).

[21] S. Bai, S. D. Galbraith. Lattice Decoding Attacks on Binary LWE. ACISP. Vol. 8544. Lecture Notes in Computer Science. Springer, 2014, pp. 322–337.

[22] S. Bai, T. Lepoint, A. Roux-Langlois, A. Sakzad, D. Stehlé, R. Steinfeld. Improved Security Proofs in Lattice-Based Cryptography: Using the Rényi Divergence Rather than the Statistical Distance. J. Cryptol. Vol. 31, Num. 2 (2018), pp. 610–640.

[23] H. Bambury, P. Q. Nguyen. Improved Provable Reduction of NTRU and Hypercubic Lattices. PQCrypto (1). Vol. 14771. Lecture Notes in Computer Science. Springer, 2024, pp. 343–370.

[24] A. Banerjee, C. Peikert, A. Rosen. Pseudorandom Functions and Lattices. EUROCRYPT. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 719–737.

[25] M. Barbosa et al. Fixing and Mechanizing the Security Proof of Fiat-Shamir with Aborts and Dilithium. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 358–389.

[26] E. Barker, J. Kelsey. Recommendation for Random Number Generation Using Deterministic Random Bit Generators. NIST SP 800-90A Rev. 1, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf. 2015-06.

[27] E. Barker, J. Kelsey, K. McKay, A. Roginsky, M. S. Turan. Recommendation for Random Bit Generator (RBG) Constructions. NIST SP 800-90C (4th public draft), https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90C.4pd.pdf. 2024-07. (2025-02-17 閲覧).

[28] A. Basso, J. M. Bermudo Mera, J.-P. D'Anvers, A. Karmakar, S. S. Roy, M. Van Beirendonck, F. Vercauteren. SABER: Mod-LWR based KEM (Round 3 Submission). https://www.esat.kuleuven.be/cosic/pqcrypto/saber/files/saberspecround3.pdf. (2024-03-04 閲覧).

[29] B. Bencina, A. Budroni, J.-J. Chi-Domínguez, M. Kulkarni. Properties of Lattice Isomorphism as a Cryptographic Group Action. PQCrypto (1). Vol. 14771. Lecture Notes in Computer Science. Springer, 2024, pp. 170–201.

[30] O. Bernard, A. Lesavourey, TH Nguyen, A. Roux-Langlois. Log-S-unit Lattices Using Explicit Stickelberger Generators to Solve Approx Ideal-SVP. ASIACRYPT (3). Vol. 13793. Lecture Notes in Computer Science. Springer, 2022, pp. 677–708.

[31] W. Beullens, S. Dobson, S. Katsumata, Y.-F. Lai, F. Pintore. Group signatures and more from isogenies and lattices: generic, simple, and efficient. Vol. 91. 6. 2023, pp. 2141–2200.

[32] M. Bolboceanu, Z. Brakerski, D. Sharma. On Algebraic Embedding for Unstructured Lattices. Public Key Cryptography (3). Vol. 14603. Lecture Notes in Computer Science. Springer, 2024, pp. 123–154.

[33] J. W. Bos, C. Costello, L. Ducas, I. Mironov, M. Naehrig, V. Nikolaenko, A. Raghunathan, D. Stebila. Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. CCS. ACM, 2016, pp. 1006–1018.

[34] J. W. Bos et al. CRYSTALS – Kyber: A CCA-Secure Module-Lattice-Based KEM. EuroS&P. IEEE, 2018, pp. 353–367.

[35] C. Boschini, A. Takahashi, M. Tibouchi. MuSig-L: Lattice-Based Multi-signature with Single-Round Online Phase. CRYPTO (2). Vol. 13508. Lecture Notes in Computer Science. Springer, 2022, pp. 276–305.

[36] K. Boudgoust, E. Gachon, A. Pellet-Mary. Some Easy Instances of Ideal-SVP and Implications on the Partial Vandermonde Knapsack Problem. CRYPTO (2). Vol. 13508. Lecture Notes in Computer Science. Springer, 2022, pp. 480–509.

[37] M. Braithwaite. Experimenting with post-quantum cryptography. `https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html`. 2023-04. (2024-03-04 閲覧).

[38] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, D. Stehlé. Classical hardness of learning with errors. STOC. ACM, 2013, pp. 575–584.

[39] Z. Brakerski, V. Vaikuntanathan. Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages. CRYPTO. Vol. 6841. Lecture Notes in Computer Science. Springer, 2011, pp. 505–524.

[40] L. G. Bruinderink, A. Hülsing, T. Lange, Y. Yarom. Flush, Gauss, and Reload - A Cache Attack on the BLISS Lattice-Based Signature Scheme. CHES. Vol. 9813. Lecture Notes in Computer Science. Springer, 2016, pp. 323–345.

[41] K. Carrier, T. Debris-Alazard, C. Meyer-Hilfiger, J.-P. Tillich. Reduction from Sparse LPN to LPN, Dual Attack 3.0. EUROCRYPT (6). Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 286–315.

[42] C. Chen, J. Hoffstein, W. Whyte, Z. Zhang. NIST PQ Submission: NTRUEncrypt A lattice based encryption algorithm. `https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/NTRUEncrypt.zip`. (2024-03-04 閲覧).

[43] C. Chen et al. NTRU algorithm specifications and supporting documentation (September 30,2020). `https://csrc.nist.gov/CSRC/media/Projects/post-quantum-cryptography/documents/round-3/submissions/NTRU-Round3.zip`. 2020-09. (2024-03-04 閲覧).

[44] Y. Chen. sfDualMS: Efficient Lattice-Based Two-Round Multi-signature with Trapdoor-Free Simulation. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 716–747.

[45] Y. Chen, Q. Liu, M. Zhandry. Quantum Algorithms for Variants of Average-Case Lattice Problems via Filtering. EUROCRYPT (3). Vol. 13277. Lecture Notes in Computer Science. Springer, 2022, pp. 372–401.

[46] Y. Chen, P. Q. Nguyen. BKZ 2.0: Better Lattice Security Estimates. ASIACRYPT. Vol. 7073. Lecture Notes in Computer Science. Springer, 2011, pp. 1–20.

[47] C. Chuengsatiansup, T. Prest, D. Stehlé, A. Wallet, K. Xagawa. ModFalcon: Compact Signatures Based On Module-NTRU Lattices. AsiaCCS. ACM, 2020, pp. 853–866.

[48] C.-M. M. Chung, V. Hwang, M. J. Kannwischer, G. Seiler, C.-J. Shih, B.-Y. Yang. NTT Multiplication for NTT-unfriendly Rings New Speed Records for Saber and NTRU on Cortex-M4 and AVX2. IACR Trans. Cryptogr. Hardw. Embed. Syst. Vol. 2021, Num. 2 (2021), pp. 159–188.

[49] J.-S. Coron, F. Gérard, M. Trannoy, R. Zeitoun. Improved Gadgets for the High-Order Masking of Dilithium. Vol. 2023. 4. 2023, pp. 110–145.

[50] R. Cramer, L. Ducas, B. Wesolowski. Mildly Short Vectors in Cyclotomic Ideal Lattices in Quantum Polynomial Time. J. ACM. Vol. 68, Num. 2 (2021), 8:1–8:26.

[51] R. Cramer, L. Ducas, B. Wesolowski. Short Stickelberger Class Relations and Application to Ideal-SVP. EUROCRYPT (1). Vol. 10210. Lecture Notes in Computer Science. 2017, pp. 324–348.

[52] C. Cremers, S. Düzlü, R. Fiedler, M. Fischlin, C. Janson. BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures. Symposium on Security and Privacy (SP). IEEE, 2021, pp. 1696–1714.

[53] J.-P. D'Anvers. SABER: Module-LWR based KEM. Second PQC Standardization Conference. 2019-08. `https://csrc.nist.gov/Presentations/2019/saber-round-2-presentation`. (2024-03-04 閲覧).

[54] J.-P. D'Anvers, A. Karmakar, S. Sinha Roy, F. Vercauteren. Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM. AFRICACRYPT. Vol. 10831. Lecture Notes in Computer Science. Springer, 2018, pp. 282–305.

[55] D. Dachman-Soled, H. Gong, T. Hanson, H. Kippen. Revisiting Security Estimation for LWE with Hints from a Geometric Perspective. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 748–781.

[56] TU Darmstadt, UC San Diego. LWE Challenge. `https://www.latticechallenge.org/lwe_challenge/challenge.php`. (2024-03-04 閲覧).

[57] TU Darmstadt, UC San Diego. SVP Challenge, Hall Of Fame. `https://www.latticechallenge.org/svp-challenge/halloffame.php`. (2024-03-04 閲覧).

[58] D. Das, A. Joux. Key Recovery Attack on the Partial Vandermonde Knapsack Problem. EUROCRYPT (6). Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 205–225.

[59] T. Debris-Alazard, P. Fallahpour, D. Stehlé. Quantum Oblivious LWE Sampling and Insecurity of Standard Model Lattice-Based SNARKs. STOC. ACM, 2024, pp. 423–434.

[60] J. Devevey, A. Passelègue, D. Stehlé. G+G: A Fiat-Shamir Lattice Signature Based on Convolved Gaussians. ASIACRYPT (7). Vol. 14444. Lecture Notes in Computer Science. Springer, 2023, pp. 37–64.

[61] N. Döttling, D. Kolonelos, R. W. F. Lai, C. Lin, G. Malavolta, A. Rahimi. Efficient Laconic Cryptography from Learning with Errors. EUROCRYPT (3). Vol. 14006. Lecture Notes in Computer Science. Springer, 2023, pp. 417–446.

[62] N. Döttling, J. Müller-Quade. Lossy Codes and a New Variant of the Learning-With-Errors Problem. EUROCRYPT. Vol. 7881. Lecture Notes in Computer Science. Springer, 2013, pp. 18–34.

[63] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, D. Stehlé. CRYSTALS-Dilithium algorithm specifications and supporting documentation. https://pq-crystals.org/dilithium/data/dilithium-specification.pdf. 2017-11. (2024-03-04 閲覧).

[64] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, D. Stehlé. CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme. IACR Trans. Cryptogr. Hardw. Embed. Syst. Vol. 2018, Num. 1 (2018), pp. 238–268.

[65] L. Ducas, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, D. Stehle. CRYSTALS-Dilithium: Digital Signatures from Module Lattices. Cryptology ePrint Archive, Paper 2017/633. 2017. https://eprint.iacr.org/2017/633.

[66] L. Ducas, E. W. Postlethwaite, L. N. Pulles, W. P. J. van Woerden. HAWK: Module LIP Makes Lattice Signatures Fast, Compact and Simple. ASIACRYPT (4). Vol. 13794. Lecture Notes in Computer Science. Springer, 2022, pp. 65–94.

[67] L. Ducas, T. Prest. Fast Fourier Orthogonalization. ISSAC. ACM, 2016, pp. 191–198.

[68] L. Ducas, J. Schanck. Security estimation scripts for Kyber and Dilithium. https://github.com/pq-crystals/security-estimates. 2019-03. (2024-03-04 閲覧).

[69] L. Ducas, M. Stevens, W. P. J. van Woerden. Advanced Lattice Sieving on GPUs, with Tensor Cores. EUROCRYPT (2). Vol. 12697. Lecture Notes in Computer Science. Springer, 2021, pp. 249–279.

[70] L. Ducas, W. P. J. van Woerden. On the Lattice Isomorphism Problem, Quadratic Forms, Remarkable Lattices, and Cryptography. EUROCRYPT (3). Vol. 13277. Lecture Notes in Computer Science. Springer, 2022, pp. 643–673.

[71] S. Düzlü, R. Fiedler, M. Fischlin. BUFFing FALCON without Increasing the Signature Size. IACR Cryptol. ePrint Arch. (2024), p. 710.

[72] T. Espitau, P.-A. Fouque, F. Gérard, M. Rossi, A. Takahashi, M. Tibouchi, A. Wallet, Y. Yu. MITAKA: A Simpler, Parallelizable, Maskable Variant of FALCON. EUROCRYPT (3). Vol. 13277. Lecture Notes in Computer Science. Springer, 2022, pp. 222–253.

[73] T. Espitau, T. Thu Quyen Nguyen, C. Sun, M. Tibouchi, A. Wallet. ANTRAG: Annular NTRU trapdoor generation - Making MITAKA as secure as FALCON. ASIACRYPT (7). Vol. 14444. Lecture Notes in Computer Science. Springer, 2023, pp. 3–36.

[74] T. Espitau, M. Tibouchi, A. Wallet, Y. Yu. Shorter Hash-and-Sign Lattice-Based Signatures. CRYPTO (2). Vol. 13508. Lecture Notes in Computer Science. Springer, 2022, pp. 245–275.

[75] ETSI TR 103 616 V1.1.1 (2021-09) CYBER; Quantum-safe signatures. https://www.etsi.org/deliver/etsi_tr/103600_103699/103616/01.01.01_60/tr_103616v010101p.pdf. 2021-09. (2024-03-04 閲覧).

[76] M. Fahr et al. When Frodo Flips: End-to-End Key Recovery on FrodoKEM via Rowhammer. CCS. ACM, 2022, pp. 979–993.

[77] J. Felderhoff, A. Pellet-Mary, D. Stehlé. On Module Unique-SVP and NTRU. ASIACRYPT (3). Vol. 13793. Lecture Notes in Computer Science. Springer, 2022, pp. 709–740.

[78] A. Fiat, A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. CRYPTO. Vol. 263. Lecture Notes in Computer Science. Springer, 1986, pp. 186–194.

[79] P.-A. Fouque, F. Gérard, M. Rossi, Y. Yu. ℤalcon: An alternative FPA-free NTRU sampler for Falcon. Third PQC Standardization Conference. 2021-06. (2024-03-04 閲覧).

[80] P.-A. Fouque et al. FALCON: Fast-Fourier lattice-based compact signatures over NTRU. Specification v1.0. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/Falcon.zip. (2024-03-04 閲覧).

[81] P.-A. Fouque et al. FALCON: Fast-Fourier lattice-based compact signatures over NTRU. Specification v1.2 – 01/10/2020. https://falcon-sign.info/falcon.pdf. 2020-10. (2024-03-04 閲覧).

[82] FrodoKEM: Learning With Errors Key Encapsulation – Preliminary Standardization Proposal. https://frodokem.org/files/FrodoKEM_standard_proposal_20241205.pdf. 2024-12. (2025-01-27 閲覧).

[83] C. Gentry, C. Peikert, V. Vaikuntanathan. How to Use a Short Basis: Trapdoors for hard lattices and new cryptographic constructions. STOC. ACM, 2008, pp. 197–206.

[84] O. Goldreich, S. Goldwasser, S. Halevi. Public-Key Cryptosystems from Lattice Reduction Problems. CRYPTO. Vol. 1294. Lecture Notes in Computer Science. Springer, 1997, pp. 112–131.

[85] M. Grillere, P. Thomassen, N. Wisiol. FALCON-512 in PowerDNS. https://blog.powerdns.com/2022/04/07/falcon-512-in-powerdns/. 2022-04. (2024-03-04 閲覧).

[86] Q. Guo, T. Johansson, A. Nilsson. A Key-Recovery Timing Attack on Post-quantum Primitives Using the Fujisaki-Okamoto Transformation and Its Application on FrodoKEM. CRYPTO (2). Vol. 12171. Lecture Notes in Computer Science. Springer, 2020, pp. 359–386.

[87] J. Hoffstein, J. Pipher, J. H. Silverman. NTRU: A Ring-Based Public Key Cryptosystem. ANTS. Vol. 1423. Lecture Notes in Computer Science. Springer, 1998, pp. 267–288.

[88] D. Hofheinz, K. Hostáková, R. Langrehr, B. Ursu. On Structure-Preserving Cryptography and Lattices. Public Key Cryptography (3). Vol. 14603. Lecture Notes in Computer Science. Springer, 2024, pp. 255–287.

[89] D. Hofheinz, K. Hövelmanns, E. Kiltz. A Modular Analysis of the Fujisaki-Okamoto Transformation. TCC (1). Vol. 10677. Lecture Notes in Computer Science. Springer, 2017, pp. 341–371.

[90] J. Howe, T. Pöppelmann, M. O'Neill, E. O'Sullivan, T. Güneysu. Practical Lattice-Based Digital Signature Schemes. ACM Trans. Embed. Comput. Syst. Vol. 14, Num. 3 (2015), 41:1–41:24.

[91] A. Hülsing, K.-C. Ning, P. S., F. Weber, P. R. Zimmermann. Post-quantum WireGuard. SP. IEEE, 2021, pp. 304–321.

[92] A. Hülsing, J. Rijneveld, J. M. Schanck, P. Schwabe. High-Speed Key Encapsulation from NTRU. CHES. Vol. 10529. Lecture Notes in Computer Science. Springer, 2017, pp. 232–252.

[93] A. Hülsing, J. Rijneveld, J. M. Schanck, P. Schwabe. NTRU-HRSS-KEM Algorithm Specifications And Supporting Documentation (November 30, 2017). https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/NTRU_HRSS_KEM.zip. 2017-11. (2024-03-04 閲覧).

[94] R. Kannan. Improved Algorithms for Integer Programming and Related Lattice Problems. STOC. ACM, 1983, pp. 193–206.

[95] K. Kim, M. Tibouchi, A. Wallet, T. Espitau, A. Takahashi, Y. Yu, S. Guilley. SOLMAE – Algorithm specifications. https://kpqc.or.kr/images/pdf/SOLMAE.pdf. (2024-03-04 閲覧).

[96] R. W. F. Lai, G. Malavolta. Lattice-Based Timed Cryptography. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 782–804.

[97] A. Langlois, D. Stehlé. Worst-case to average-case reductions for module lattices. Des. Codes Cryptogr. Vol. 75, Num. 3 (2015), pp. 565–599.

[98] M. Lantz. World's first quantum computing safe tape drive. https://www.ibm.com/blogs/research/2019/08/crystals/. 2019-08. (2024-03-04 閲覧).

[99] A. K. Lenstra, H. W. Lenstra, L. Lovász. Factoring polynomials with rational coefficients. Mathematische Annalen. Vol. 261, Num. 4 (1982), pp. 515–534.

[100] X. Lin, M. Suzuki, S. Zhang, T. Espitau, Y. Yu, M. Tibouchi, M. Abe. Cryptanalysis of the Peregrine Lattice-Based Signature Scheme. Public Key Cryptography (1). Vol. 14601. Lecture Notes in Computer Science. Springer, 2024, pp. 387–412.

[101] R. Lindner, C. Peikert. Better Key Sizes (and Attacks) for LWE-Based Encryption. CT-RSA. Vol. 6558. Lecture Notes in Computer Science. Springer, 2011, pp. 319–339.

[102] H. Liu, Y. Yu. A Non-heuristic Approach to Time-Space Tradeoffs and Optimizations for BKW. ASIACRYPT (3). Vol. 13793. Lecture Notes in Computer Science. Springer, 2022, pp. 741–770.

[103] Z. Liu, V., J. Ding, C. Cheng, Y. Pan. An Improved Practical Key Mismatch Attack Against NTRU. PQCrypto (1). Vol. 14771. Lecture Notes in Computer Science. Springer, 2024, pp. 322–342.

[104] V. Lyubashevsky. CRYSTALS-Dilithium Round 3 presentation. Third PQC Standardization Conference. 2021-06. https://csrc.nist.gov/Presentations/2021/crystals-dilithium-round-3-presentation. (2024-03-04 閲覧).

[105] V. Lyubashevsky. Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures. ASIACRYPT. Vol. 5912. Lecture Notes in Computer Science. Springer, 2009, pp. 598–616.

[106] V. Lyubashevsky. Lattice Signatures without Trapdoors. EUROCRYPT. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 738–755.

[107] V. Lyubashevsky, C. Peikert, O. Regev. On Ideal Lattices and Learning with Errors over Rings. EUROCRYPT. Vol. 6110. Lecture Notes in Computer Science. Springer, 2010, pp. 1–23.

[108] V. Lyubashevsky, C. Peikert, O. Regev. On Ideal Lattices and Learning with Errors over Rings. J. ACM. Vol. 60, Num. 6 (2013), 43:1–43:35.

[109] D. Micciancio, M. Schultz. Error Correction and Ciphertext Quantization in Lattice Cryptography. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 648–681.

[110] D. Micciancio, V. Vaikuntanathan. SoK: Learning with Errors, Circular Security, and Fully Homomorphic Encryption. Public Key Cryptography (4). Vol. 14604. Lecture Notes in Computer Science. Springer, 2024, pp. 291–321.

[111] G. De Micheli, D. Kim, D. Micciancio, A. Suhl. Faster Amortized FHEW Bootstrapping Using Ring Automorphisms. Public Key Cryptography (4). Vol. 14604. Lecture Notes in Computer Science. Springer, 2024, pp. 322–353.

[112] G. De Micheli, D. Micciancio, A. Pellet-Mary, N. Tran. Reductions from Module Lattices to Free Module Lattices, and Application to Dequantizing Module-LLL. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 836–865.

[113] D. Moody et al. Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process. NIST IR 8309, https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8309.pdf. 2020-07.

[114] G. Mureau, A. Pellet-Mary, G. Pliatsok, A. Wallet. Cryptanalysis of Rank-2 Module-LIP in Totally Real Number Fields. EUROCRYPT (6). Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 226–255.

[115] M. Naehrig, K. E. Lauter, V. Vaikuntanathan. Can homomorphic encryption be practical? CCSW. ACM, 2011, pp. 113–124.

[116] M. Naehrig et al. FrodoKEM – learning with errors key encapsulation. Algorithm specifications and supporting documentation (November 30, 2017). https://frodokem.org/files/FrodoKEM-specification-20171130.pdf. 2017-11. (2024-03-04 閲覧).

[117] P. Q. Nguyen, J. Stern. Cryptanalysis of the Ajtai-Dwork Cryptosystem. CRYPTO. Vol. 1462. Lecture Notes in Computer Science. Springer, 1998, pp. 223–242.

[118] Q. Nguyen. ANTRAG: Simplifying and improving Falcon Without Compromising Security. https://csrc.nist.gov/csrc/media/Presentations/2024/antrag-simplifying-and-improving-falcon/images-media/nguyen-antrag-pqc2024.pdf. 2024-04. (2024-12-30 閲覧).

[119] NIST. Module-Lattice-Based Digital Signature Standard. NIST FIPS 204, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.204.pdf. 2024-08.

[120] NIST. Module-Lattice-Based Key-Encapsulation Mechanism Standard. NIST FIPS 203, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.pdf. 2024-08.

[121] NIST. Submission requirements and evaluation criteria for the post-quantum cryptography standardization process. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf. 2016-12. (2024-03-05 閲覧).

[122] OpenSSH 8.9 was released on 2022-02-23. https://www.openssh.com/txt/release-8.9. (2024-03-04 閲覧).

[123] J. Pan, B. Wagner, R. Zeng. Lattice-Based Authenticated Key Exchange with Tight Security. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 616–647.

[124] C. Peikert, V. Vaikuntanathan, B. Waters. A Framework for Efficient and Composable Oblivious Transfer. CRYPTO. Vol. 5157. Lecture Notes in Computer Science. Springer, 2008, pp. 554–571.

[125] R. del Pino, S. Katsumata. A New Framework for More Efficient Round-Optimal Lattice-Based (Partially) Blind Signature via Trapdoor Sampling. CRYPTO (2). Vol. 13508. Lecture Notes in Computer Science. Springer, 2022, pp. 306–336.

[126] T. Plantard, W. Susilo, K. Than Win. A Digital Signature Scheme Based on CVP_{∞}. Public Key Cryptography. Vol. 4939. Lecture Notes in Computer Science. Springer, 2008, pp. 288–307.

[127] T. Pöppelmann, E. Alkim, R. Avanzi, J. Bos, L. Ducas, A. de la Piedra, P. Schwabe, D. Stebila. NewHope algorithm specifications and supporting documentation Version 1.0. https://newhopecrypto.org/data/NewHope_2017_12_21.pdf. 2017-11. (2024-03-04 閲覧).

[128] T. Pöppelmann et al. NewHope algorithm specifications and supporting documentation Version 1.02 (Updated March 15, 2019). https://newhopecrypto.org/data/NewHope_2019_04_10.pdf. 2019-03. (2024-03-04 閲覧).

[129] E. W. Postlethwaite, F. Virdia. On the Success Probability of Solving Unique SVP via BKZ. Public Key Cryptography (1). Vol. 12710. Lecture Notes in Computer Science. Springer, 2021, pp. 68–98.

[130] A. Pouly, Y. Shen. Provable Dual Attacks on Learning with Errors. EUROCRYPT (6). Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 256–285.

[131] T. Prest. FALCON Update (2024). https://csrc.nist.gov/csrc/media/Presentations/2024/falcon/images-media/prest-falcon-pqc2024.pdf. 2024-04. (2024-12-30 閲覧).

[132] The Chromium Projects. CECPQ2. https://www.chromium.org/cecpq2/. (2024-03-04 閲覧).

[133] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. STOC. ACM, 2005, pp. 84–93.

[134] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. J. ACM. Vol. 56, Num. 6 (2009), 34:1–34:40.

[135] O. Regev. The Learning with Errors Problem (Invited Survey). CCC. IEEE Computer Society, 2010, pp. 191–204.

[136] M. Rosca, A. Sakzad, D. Stehlé, R. Steinfeld. Middle-Product Learning with Errors. CRYPTO (3). Vol. 10403. Lecture Notes in Computer Science. Springer, 2017, pp. 283–297.

[137] K. Ryan, N. Heninger. Fast Practical Lattice Reduction Through Iterated Compression. CRYPTO (3). Vol. 14083. Lecture Notes in Computer Science. Springer, 2023, pp. 3–36.

[138] Open Quantum Safe. Algorithms in liboq. https://openquantumsafe.org/liboqs/algorithms/. (2024-03-04 閲覧).

[139] T. Saito, K. Xagawa, T. Yamakawa. Tightly-Secure Key-Encapsulation Mechanism in the Quantum Random Oracle Model. EUROCRYPT (3). Vol. 10822. Lecture Notes in Computer Science. Springer, 2018, pp. 520–551.

[140] C. Saliba. Error correction and reconciliation techniques for lattice-based key generation protocols. Ph. D. Theses. CY Cergy Paris Université, 2022-05. https://theses.hal.science/tel-03718212/file/SALIBA_2022.pdf.

[141] C. Saliba, L. Luzzi, C. Ling. Error Correction for FrodoKEM Using the Gosset Lattice. 2021.

[142] C.-P. Schnorr, M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. Math. Program. Vol. 66 (1994), pp. 181–199.

[143] Federal office for information security. BSI – Technical guideline (Cryptographic mechanisms: Recommendations and key lengths). https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf?__blob=publicationFile&v=10. 2023-01. (2024-03-04 閲覧).

[144] Selected Algorithms 2022. 2022-07. https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. (2024-03-04 閲覧).

[145] E.-Y. Seo, Y.-S. Kim, J.-W. Lee, J.-S. No. Peregrine: Toward Fastest FALCON Based on GPV Framework. (2022). https://eprint.iacr.org/2022/1495.

[146] H. M. Steffen, G. Land, L. J. Kogelheide, T. Güneysu. Breaking and Protecting the Crystal: Side-Channel Analysis of Dilithium in Hardware. PQCrypto. Vol. 14154. Lecture Notes in Computer Science. Springer, 2023, pp. 688–711.

[147] D. Stehlé, R. Steinfeld. Making NTRU as Secure as Worst-Case Problems over Ideal Lattices. EUROCRYPT. Vol. 6632. Lecture Notes in Computer Science. Springer, 2011, pp. 27–47.

[148] D. Stehlé, R. Steinfeld. Making NTRUEncrypt and NTRUSign as Secure as Standard Worst-Case Problems over Ideal Lattices. Cryptology ePrint Archive, Paper 2013/004. 2013. https://eprint.iacr.org/2013/004.

[149] D. Stehlé, R. Steinfeld, K. Tanaka, K. Xagawa. Efficient Public Key Encryption Based on Ideal Lattices. ASIACRYPT. Vol. 5912. Lecture Notes in Computer Science. Springer, 2009, pp. 617–635.

[150] M. J. Steiner. The Complexity of Algebraic Algorithms for LWE. EUROCRYPT (3). Vol. 14653. Lecture Notes in Computer Science. Springer, 2024, pp. 375–403.

[151] M. S. Turan, E. Barker, J. Kelsey, K. McKay, M. Baish, M. Boyle. Recommendation for the Entropy Sources Used for Random Bit Generation. NIST SP 800-90B, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf. 2018-01.

[152] L. Wang. Analyzing Pump and Jump BKZ Algorithm Using Dynamical Systems. PQCrypto (1). Vol. 14771. Lecture Notes in Computer Science. Springer, 2024, pp. 406–432.

[153] R. Wang, M. Brisfors, E. Dubrova. A Side-Channel Attack on a Higher-Order Masked CRYSTALS-Kyber Implementation. ACNS (3). Vol. 14585. Lecture Notes in Computer Science. Springer, 2024, pp. 301–324.

[154] Z. Wang, Q. Lai, F.-H. Liu. Ring/Module Learning with Errors Under Linear Leakage – Hardness and Applications. Public Key Cryptography (2). Vol. 14602. Lecture Notes in Computer Science. Springer, 2024, pp. 275–304.

[155] H. Wee, D. J. Wu. Succinct Vector, Polynomial, and Functional Commitments from Lattices. EUROCRYPT (3). Vol. 14006. Lecture Notes in Computer Science. Springer, 2023, pp. 385–416.

[156] wolfSSL. wolfSSL support for Apache httpd and curl (Post-Quantum Edition). https://github.com/wolfSSL/osp/blob/master/apache-httpd/README_post_quantum.md. (2024-03-04 閲覧).

[157] W. Xia, L. Wang, G. Wang, D. Gu, B. Wang. A Refined Hardness Estimation of LWE in Two-Step Mode. Public Key Cryptography (3). Vol. 14603. Lecture Notes in Computer Science. Springer, 2024, pp. 3–35.

[158] W. Xia, L. Wang, G. Wang, D. Gu, B. Wang. Refined Strategy for Solving LWE in Two-step Mode. Cryptology ePrint Archive, Paper 2022/1343. 2022. https://eprint.iacr.org/2022/1343.

[159] Y. Yu, L. Ducas. Second Order Statistical Behavior of LLL and BKZ. SAC. Vol. 10719. Lecture Notes in Computer Science. Springer, 2017, pp. 3–22.

[160] Y. Yu, H. Jia, X. Wang. Compact Lattice Gadget and Its Applications to Hash-and-Sign Signatures. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 390–420.

[161] 量子耐性暗号研究団. KpqC. https://kpqc.or.kr/. (2024-12-06 閲覧).

第 4 章

符号に基づく暗号技術

本章では符号に基づく暗号技術についてまとめる。符号に基づく暗号技術の安全性はシンドローム復号（Syndrome Decoding: SD）問題や Learning Parity with Noise（LPN）問題を解く計算の困難性に依存している。

■準備: 本章で使用する記号・用語を以下にまとめる。以下では，q を素数 p のべきとする。すなわち，ある正整数 k が存在して $q = p^k$ である。以下では log の底が省略されている場合は底を 2 とする。自然対数を用いる場合は ln と書く。

有限体: $\mathbb{F}_q$ で位数が q の有限体を表す。

ハミング重みとハミング距離: V_n を有限体 $\mathbb{F}_q$ 上の n 次元ベクトル空間とする。

- 行ベクトル $\boldsymbol{v} = (v_1, v_2, \ldots, v_n) \in V_n$ のハミング重みとは，非ゼロの成分の数である。すなわち，有限集合 X に対して $|X|$ で X の要素数を表すとき，$\mathsf{HW}(\boldsymbol{v}) = |\{i \mid v_i \neq 0\}|$ である。
- ハミング距離を $d_H(\boldsymbol{x}, \boldsymbol{y}) = \mathsf{HW}(\boldsymbol{x} - \boldsymbol{y})$ で定義する。
- $\mathcal{S}_H(n, w)$ でハミング重みが w の n 次元ベクトル全体の集合を表す。
- $\mathcal{S}_H^{\leq}(n, w)$ でハミング重みが w 以下の n 次元ベクトル全体の集合を表す。

■線形符号: 線形符号とは，誤りが発生する通信路において，メッセージを相手に正しく伝えるための技術である。メッセージを冗長にして（符号化という）送信し，受信時に伝送中に生じた誤りを訂正する（復号という）ことで，正しいメッセージを得ることができる。自然数 n および 素数べき q について，$\mathbb{F}_q$ 上の n 次元ベクトル空間の線形部分空間を $\mathbb{F}_q$ 上の線形符号と呼び，$\mathcal{C}$ で表す。$\mathcal{C}$ の要素を符号語と呼び，n を符号長という。このとき $[\boldsymbol{c}_1, \ldots, \boldsymbol{c}_k]$ を $\mathcal{C}$ の基底とする。k を線形符号の次元と呼ぶ。$\mathbb{F}_q$ 上の線形符号の符号長が n，次元が k であるとき，$[n, k]_q$ -線形符号とよぶ。$[n, k]_q$ -線形符号 $\mathcal{C}$ の最小距離 d とは，2 つの異なる符号語間のハミング距離の最小値である。d は符号 $\mathcal{C}$ の全ての非ゼロ符号語のハミング重みの最小値に一致する。すなわち，$d = \min_{\boldsymbol{c} \in \mathcal{C} \setminus \{\boldsymbol{0}\}} \mathsf{HW}(\boldsymbol{c})$ である。

$[n, k]_q$ -線形符号 $\mathcal{C}$ の生成行列とは，符号 $\mathcal{C}$ の基底ベクトルを行とする行列 $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$ であり，メッセージの符号化に用いられる。メッセージ $\boldsymbol{s} \in \mathbb{F}_q^k$ に対して，$\boldsymbol{sG} \in \mathbb{F}_q^n$ は符号語である。メッセージと符号語は一対一対応させることができる。$[n, k]_q$ -線形符号 $\mathcal{C}$ のパリティ検査行列とは，行列 $\boldsymbol{H} \in \mathbb{F}_q^{r \times n}$ で，$\boldsymbol{c} \in \mathbb{F}_q^n$ に対して，$\boldsymbol{c} \in \mathcal{C}$ ならばかつその時に限り $\boldsymbol{cH}^\top = \boldsymbol{0}$ となるものである。$\boldsymbol{H}$ の行が一次独立であれば，$r = n - k$ である。組織符号化とは，行列 $\boldsymbol{H}$ に対して，行列 $\boldsymbol{S} \in \mathbb{F}_q^{(n-k) \times (n-k)}$ を適用し，$\boldsymbol{SH} = [\boldsymbol{I}_{n-k} \mid \boldsymbol{Z}]$ を得る操作を指す。ここで，$\boldsymbol{Z} \in \mathbb{F}_q^{(n-k) \times k}$ である。

線形符号は，生成行列やパリティ検査行列をうまく設計することで，受信時に符号語に加えられた誤りを訂正することができる。誤り訂正には，復号アルゴリズムが用いられる。送信する符号語を $\boldsymbol{c}$ とし，通信路上で乗った誤りを $\boldsymbol{e}$ とする。受信者側は，受信語として $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$ を得る。受信者は，復号アルゴリズムを用いて $\boldsymbol{y}$ から $\boldsymbol{c}$ を得る。受信者がハミング重み t までの誤り $\boldsymbol{e}$ を一意に訂正できるとき，符号の訂正能力が t であるという。一般に，$t \leq \lfloor (d-1)/2 \rfloor$ が成り立つ。復号アルゴリズムには，符号の構造を用いる方式や，パリティ検査行列を用いる方式がある。後者の場

合，受信語 $\boldsymbol{y}$ に対して $\boldsymbol{s} = \boldsymbol{y}\boldsymbol{H}^\top$ を計算する。$\boldsymbol{s}$ はシンドロームと呼ばれる。$\boldsymbol{s} = \boldsymbol{e}\boldsymbol{H}^\top$ となることから，$\boldsymbol{s} \neq \boldsymbol{0}$ であれば，誤りを検出・訂正できる。

本稿では，具体的な線形符号（リード・ソロモン符号，リード・マラー符号，Goppa 符号）の詳細については扱わない。符号理論の教科書や，電子情報通信学会 知識の森 「1 群 2 編 符号理論」 [57] などを参照されたい。

4.1 符号に基づく暗号技術の安全性の根拠となる問題

本節では，SD 問題と LPN 問題およびその困難性について報告する。

4.1.1 SD 問題とその拡張

4.1.1.1 SD 問題

SD 問題とは，解のハミング重みが指定された条件のもとで，$\mathbb{F}_2$ 上の線形方程式を解けるかどうかという問題である。また，$[n,k]_2$ -線形符号において，パリティ検査行列とシンドローム，受信語に乗ったエラーのハミング重みが与えられたときに，エラーを求める問題とみなすことができる。本問題は符号暗号の安全性の根拠として非常に重要である。実際に， 4.3 節で説明する NIST PQC 標準化プロジェクトの第 4 ラウンドの 3 種類の符号暗号いずれの方式においても，SD 問題が安全性の根拠である。また，近年は SD 問題の拡張問題を安全性の根拠とする暗号や署名方式が多数提案されていることから，主要な問題について説明する。

定義 4.1 (SD 問題) $k, w \leq n$ を満たす正の整数 n, k, w に対して，行列 $\boldsymbol{H} \in \mathbb{F}_2^{(n-k)\times n}$ とベクトル $\boldsymbol{s} \in \mathbb{F}_2^{n-k}$ が与えられる。$\mathsf{SD}_{n,k,w}$ 問題は，$\boldsymbol{e}\boldsymbol{H}^\top = \boldsymbol{s}$ かつ $\mathsf{HW}(\boldsymbol{e}) = w$ を満たすベクトル $\boldsymbol{e} \in \mathbb{F}_2^n$ を求める問題である。

4.1.1.2 SD 問題の拡張

SD 問題の拡張として，Regular-SD 問題，Restricted-SD 問題，Rank-SD 問題について述べる。

■Regular-SD 問題: ベクトル $\boldsymbol{e} \in \mathbb{F}_2^n$ が w-正則であるとは，$\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_w)$ であり，各 $\boldsymbol{e}_i \in \mathbb{F}_2^{n/w}$ がハミング重み $\mathsf{HW}(\boldsymbol{e}_i) = 1$ を満たすことを指す。$\text{Regular-SD}_{n,k,w}$ 問題は，$\boldsymbol{e}\boldsymbol{H}^\top = \boldsymbol{s}$ かつ $\mathsf{HW}(\boldsymbol{e}) = w$ を満たす w-正則なベクトル $\boldsymbol{e} \in \mathbb{F}_2^n$ を求める問題である。Carozza，Couteau，Joux [16] によると，Regular-SD 問題に基づく署名は，通常の SD 問題に基づく署名と比較して，署名長や署名時間がより短くなる場合があるとのことである。

■Restricted-SD 問題: Restricted-SD 問題は，体を $\mathbb{F}_2$ から $\mathbb{F}_q$ に変更し，解 $\boldsymbol{e}$ の取りうる値を 0 を含む $\mathbb{F}_q$ の部分集合に制限したものである。

定義 4.2 (Restricted-SD 問題) $k, w \leq n$ を満たす正の整数 n, k, w に対して，行列 $\boldsymbol{H} \in \mathbb{F}_q^{(n-k)\times n}$ とベクトル $\boldsymbol{s} \in \mathbb{F}_q^{n-k}$ が与えられる。部分群 $\mathbb{E} \subseteq \mathbb{F}_q^\times$ に対して，$\mathbb{E}_0 = \mathbb{E} \cup \{0\}$ とする。$\text{Restricted-SD}_{n,k,w}$ 問題は，$\boldsymbol{e}\boldsymbol{H}^\top = \boldsymbol{s}$ かつ $\mathsf{HW}(\boldsymbol{e}) = w$ を満たすベクトル $\boldsymbol{e} \in \mathbb{E}_0^n$ を求める問題である。

Restricted-SD 問題は，NIST PQC 標準化プロジェクト追加署名第 2 ラウンドの CROSS [7] の安全性の根拠である。

■Rank-SD 問題: Rank-SD 問題は，SD 問題を拡大体 $\mathbb{F}_{q^m}$ に拡張し，重みの指標として行列のランクを用いたものである。$\mathbb{F}_{q^m}$ の $\mathbb{F}_q$ 基底を $(b_1, \ldots, b_m)$ とする。ベクトル $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ を考える。各座標 x_i について，

$$x_i = \sum_{j=1}^{m} x_{i,j} b_j$$

となるようなベクトル $(x_{i,1},\ldots,x_{i,m}) \in \mathbb{F}_q^m$ と関連付けることができる。ベクトル $\boldsymbol{x}$ の関連行列 $\boldsymbol{M}$ を，$\boldsymbol{M} = (x_{i,j})_{(i,j)\in[1,n]\times[1,m]}$ で定義する。このとき，$\boldsymbol{x}$ のランク重み $\mathsf{RW}(\boldsymbol{x})$ を $\mathsf{Rank}(\boldsymbol{M})$ で定義する。

定義 4.3 (Rank-SD 問題) $k, w \leq n$ を満たす正の整数 n, k, w に対して，行列 $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ とベクトル $\boldsymbol{s} \in \mathbb{F}_{q^m}^{n-k}$ が与えられる。$\mathsf{Rank\text{-}SD}_{n,k,w}$ 問題は，$\boldsymbol{e}\boldsymbol{H}^\top = \boldsymbol{s}$ かつ $\mathsf{RW}(\boldsymbol{e}) = w$ を満たすベクトル $\boldsymbol{e} \in \mathbb{F}_{q^m}^n$ を求める問題である。

Rank-SD 問題は，NIST PQC 標準化プロジェクト追加署名第 2 ラウンドの RYDE [5] の安全性の根拠である。

4.1.2 SD 問題に対する評価

SD 問題の計算の困難性に関して，Berlekamp，McEliece，van Tilborg [9] によって，NP 困難な 3 次元マッチング問題から SD 問題への多項式時間帰着が示されている。これにより，SD 問題が NP 困難であることが判明している。$\mathsf{SD}_{n,k,w}$ 問題や $[n,k]_2$ -線形符号における k/n は符号化レートと呼ばれており，符号化レートが 1/2 付近で SD 問題が最も難しくなることが知られている。また，符号化レートを増加させると，公開鍵に相当する入力行列 $\boldsymbol{H} \in \mathbb{F}_2^{(n-k)\times n}$ のサイズが減少することから，暗号の設計においては，符号化レート 1/2 以上 1 未満の値が採用されることが多い。

以降では，SD 問題の求解手法として最も研究が進んでいる Information Set Decoding（ISD）を取り上げる。ISD は，符号化レート 0.42 以上において既存方式の中で漸近計算量が最も小さく，SD 問題の解読チャレンジを通して実時間での計算量解析が進展している。

近年，Carrier，Debris-Alazard，Meyer-Hilfiger，Tillich [18, 17] が提案した Reduction-to-LPN（R-LPN）と呼ばれる手法が，符号化レート 0.42 未満において ISD の漸近計算量を下回るとする解析がなされた。R-LPN は，SD 問題を後述する Sparse-LPN 問題や LPN 問題に帰着して解く手法であるが，実時間での計算量に関して評価が不足しているため，本稿では取り上げない。

4.1.2.1 Information Set Decoding

$\mathsf{SD}_{n,k,w}$ 問題と対応する $[n,k]$-線形符号の最小距離を d と置く。2 進符号の場合，Gilbert-Varshamov 限界により，$k/n \approx 1 - H(d/n)$ である[*1]。$w \approx d$ の場合の $\mathsf{SD}_{n,k,w}$ 問題を Full Distance Decoding と呼ぶ。$w \approx d$ のとき，$\mathsf{SD}_{n,k,w}$ 問題は解くのが最も難しくなる。$w \gg d$ のとき，$\mathsf{SD}_{n,k,w}$ 問題には複数の解が存在することが期待され，$w \leq d$ のとき，$\mathsf{SD}_{n,k,w}$ 問題の解の個数の期待値は 1 以下である。暗号利用においては，$w \ll d$ が選ばれ，トラップドアを通して唯一解が存在するように設計される。以降は $w \leq d$ の場合を考える。

$\mathsf{SD}_{n,k,w}$ 問題を総当りで解くには，ハミング重みが w の n 次元ベクトル $\boldsymbol{e}$ を列挙すればよい。そのため，時間計算量は $O\left(\binom{n}{w}\right)$ となる。より効率的な手法として，Prange は Information Set Decoding と呼ばれる手法 [53] を提案した。基本アイデアは以下である:

1. 一様ランダムに $\boldsymbol{H} \in \mathbb{F}_2^{(n-k)\times n}$ の列ベクトルを入れ替え，$\tilde{\boldsymbol{H}} = \boldsymbol{H}\boldsymbol{P}$ とする。（$\boldsymbol{P} \in \mathbb{F}_2^{n\times n}$ は置換行列。）
2. ガウスの消去法と対応する行列 $\boldsymbol{S} \in \mathbb{F}_2^{(n-k)\times(n-k)}$ によって $\tilde{\boldsymbol{H}}$ を $[\boldsymbol{I}_{n-k} \mid \boldsymbol{Z}] = \boldsymbol{S}\tilde{\boldsymbol{H}}$ とする。（組織符号化）
3. シンドローム $\boldsymbol{s} \in \mathbb{F}_2^{n-k}$ に対して，$\boldsymbol{s}' = \boldsymbol{s}\boldsymbol{S}^\top$ を計算する。
4. $\boldsymbol{s}'$ のハミング重みが w ならば，$\boldsymbol{e} = (\boldsymbol{s}', \boldsymbol{0}_k)\boldsymbol{P}^\top$ を出力する。そうでなければ，1. に戻る。

$\mathsf{HW}(\boldsymbol{s}') = w$ ならば，$\mathsf{HW}(\boldsymbol{e}) = w$ である。また，$\boldsymbol{e}\boldsymbol{H}^\top = (\boldsymbol{s}', \boldsymbol{0}_k)\boldsymbol{P}^\top\boldsymbol{H}^\top = (\boldsymbol{s}', \boldsymbol{0}_k)\tilde{\boldsymbol{H}}^\top = (\boldsymbol{s}, \boldsymbol{0}_k)\boldsymbol{S}^\top\tilde{\boldsymbol{H}}^\top = (\boldsymbol{s}, \boldsymbol{0}_k)[\boldsymbol{I}_{n-k}\boldsymbol{Z}]^\top = \boldsymbol{s}$ が成立する。よって，ステップ 4 のチェックが通るならば，$\boldsymbol{e}$ は SD 問題の解である。末尾 k 列が全て 0 であるような $\boldsymbol{e}\boldsymbol{P}$ は $\binom{n-k}{w}$ 通りあるため，ある置換行列に対して解が出力される確率は $\binom{n-k}{w}/\binom{n}{w}$

[*1] ここで $H(p) = -p\log(p) - (1-p)\log(1-p)$。

表 4.1: 確率 1/2 以上で SD 問題を解く場合の漸近計算量（Full Distance Decoding の場合）

	$\log(\text{Time})/n$	$\log(\text{Space})/n$	備考
Pra62（Lee-Brickel）	0.121	–	[53, 39]
Stern89	0.117	0.0135	[55]
MMT11	0.112	0.0530	[43]; [30] によって空間計算量が改良された
BJMM12	0.102	0.0769	[8]
MO15	0.0967	0.0890	[44]
BM17	0.0953	0.0910	[15]; MO15 を最適化したもの
BM18	0.0951	0.0760	[14]; [18, 25] によって時間・空間計算量が修正された
Sieving ISD	0.101	0.0636	[24]

となる。期待計算量は $\mathrm{poly}(n,k)\cdot O\left(\binom{n}{w}/\binom{n-k}{w}\right)$ となり，先ほどの列挙法よりも速くなる。

Stern [55] 以降，空間計算量を犠牲にすることで時間計算量を引き下げる ISD の改良アルゴリズムが多数提案されている。Both と May [14] による時間計算量の表を，表 4.1 に示す。この表は，時間計算量を最小化した場合の符号化レート k/n の最悪時（1/2 の少し下）の漸近計算量についてまとめられている。したがって，問題のパラメータによっては，表の数値よりも速く解くことが可能となる。

近年は，漸近計算量のみならず，具体的なパラメータに対する $\mathsf{SD}_{n,k,w}$ 問題を求解するために必要なビット計算量を見積もる研究もなされている。Esser，Verbel，Zweydinger，Bellini [29] は，CryptographicEstimators と呼称する符号暗号や多変数多項式暗号のビット計算量を推定するソフトウェアを開発した。Narisada ら [48] は，Becker らの ISD [8] の実用的な改良方式を提案し，NIST PQC 標準化プロジェクト第 4 ラウンドの 3 種類の符号暗号のビット計算量と実時間の計算量を算出した。

■量子アルゴリズム　現在のところ多項式時間で SD 問題を解く量子アルゴリズムは提案されていない。しかし，量子アルゴリズムを利用して，いくつかの古典 ISD を高速化する方法を Kachigar と Tillich [36] が提案している[*2]。2024 年現在，最良の漸近計算量が得られているのは，BJMM 法 [8] の量子アルゴリズムである量子 BJMM 法であり，時間計算量が $2^{0.0587n}$，空間計算量が $2^{0.0188n}$ となっている。

量子回路設計の研究に関しては，量子 Prange 法に対して，Perriello，Barenghi，Pelosi [51] がグローバーのアルゴリズムを用いた量子回路を提案した。Esser ら [28] は，量子 Prange 法に対して，一部の演算を古典コンピュータ上で行う量子と古典のハイブリッド法を提案した。Perriello，Barenghi，Pelosi [52] は，量子 Prange におけるガウスの消去法の量子回路を改良し，NIST PQC 標準化プロジェクト第 4 ラウンドの 3 種類の符号暗号の解読に必要な量子回路の深さを最大で 2^{30} 削減した。Chevignard，Fouque，Schrottenloher [20] は，量子 Prange 法に対して，量子回路の深さを犠牲にすることで量子ビット数を削減する，深さと幅のトレードオフ手法を提案した。Stern の ISD [55] 以降に提案されたリスト探索を伴う ISD については，グローバーのアルゴリズムと量子ウォーク探索を組み合わせた複雑な量子回路が必要になると考えられている [36]。現在のところ，これらの量子 ISD に対する量子回路は提案されていない。

■現状の進展　格子の場合と同様に “Decoding Challenge”（`https://decodingchallenge.org/`）というウェブサイトが作成された。実時間での計算量解析を通じて，符号に基づく暗号技術の信頼性を向上させることが目的である。現在，以下 5 つのカテゴリが用意されている。

[*2] Kirshanova [38] が Kachigar と Tillich の結果 [36] の改良を提案していたが，誤りがあったことが報告されている。そのため，2024 年時点でのベストな量子アルゴリズムは Kachigar と Tillich [36] であると考えられる。

1. $\mathbb{F}_2$ 係数の一様ランダムな線形符号に対する SD 問題
2. $\mathbb{F}_2$ 係数の一様ランダムな線形符号に対するハミング重みが小さい符号語を探索する問題
3. $\mathbb{F}_3$ 係数の一様ランダム線形符号に対する SD 問題
4. Goppa 符号を用いた Niederreiter 暗号の場合の SD 問題。Classic McEliece（4.3.1 節）に対応
5. QC-MDPC 符号に基づく SD 問題。BIKE（4.3.2 節）や HQC（4.3.3 節）に対応

各問題に対して研究および解読が進んでおり，2025 年 1 月現在，解読に成功した最も困難な問題は次のとおりである。

- 1. $n = 570, k = n/2$ に対して $w = 70$（成定，福島，清本，2023/04）
- 2. $n = 1280, k = n/2$ の場合に $w = 204$（成定，岡田，上村，相川，福島，清本，2024/09）
- 3. $n = 200, k = \log_3(n)$ の場合に $w = 198$（Esser，May，Zweydinger，2021/12）
- 4. $n = 1409, k = 0.8n$ に対して $w = 26$（成定，古江，相川，福島，清本，2023/11）
- 5. $n = 3602, k = n/2$ に対して $w = 60$（成定，岡田，上村，相川，福島，清本，2025/1）

1 の結果については，Narisada，Fukushima，Kiyomoto [47] を，4 の結果については，Narisada ら [48] を参照されたい。2，3，5 についての詳細は，Decoding Challenge 上に掲載されている各記録の詳細を参照されたい。

4.1.3 LPN 問題とその拡張

4.1.3.1 LPN 問題

LPN 問題とは，$\mathbb{F}_2$ 上の誤差付きの線形方程式を解けるかどうかという問題である。また，$[n, k]_2$ -線形符号において，生成行列と受信語が与えられたときに，メッセージを復号する問題とみなすことができる。1993 年に，Blum，Furst，Kearns，Lipton [11] が困難と思われる問題として挙げ，定式化を行った。第 3.1.1 章において，この問題を $\mathbb{F}_q$ に一般化した LWE 問題を既に扱っている。Ber_τ でパラメータ τ のベルヌーイ分布を表す。（確率 τ で 1，確率 $1-\tau$ で 0 となる $\mathbb{F}_2$ 上の分布である。）また，自然数 $k \geq 1$ について，Ber_τ^k で，Ber_τ から独立に k 個サンプルを取ったときの $\mathbb{F}_2^k$ 上の分布を表す。

定義 4.4 (LPN 問題) $\mathbb{F}_2^k$ から一様ランダムに選ばれた秘密鍵 $\boldsymbol{s}$ およびエラー比 $\tau \in [0, 1/2)$ に対して，以下の LPN サンプルを出力する LPN オラクルを考える。

$$(\boldsymbol{a}, b) = (\boldsymbol{a}, \boldsymbol{s} \cdot \boldsymbol{a}^\top + e),$$

ここで，$\boldsymbol{a}$ は $\mathbb{F}_2^k$ から一様ランダムに選び，e は分布 Ber_τ に従い選ぶ。LPN オラクルを n 回呼び出すとき，$(\boldsymbol{A}, \boldsymbol{b}) \leftarrow \mathsf{LPN}_{k,\tau}^n$ と表記する。これは，n 個の LPN サンプル $(\boldsymbol{a}_1, b_1), (\boldsymbol{a}_2, b_2), \ldots, (\boldsymbol{a}_n, b_n)$ を行列・ベクトル表示して，

$$\boldsymbol{A} = [\boldsymbol{a}_1^\top \boldsymbol{a}_2^\top \ldots \boldsymbol{a}_n^\top] \in \mathbb{F}_2^{k \times n}, \qquad \boldsymbol{b} = \boldsymbol{s}\boldsymbol{A} + \boldsymbol{e} \in \mathbb{F}_2^n$$

としたものである。n をサンプル数と呼ぶ。$\boldsymbol{e}$ は，n 個の LPN サンプルのエラー e を成分とするベクトルである。$\mathsf{LPN}_{k,\tau}$ 問題とは，LPN オラクルへのアクセスが可能なときに，$\boldsymbol{s}$ を求める問題である。

サンプル数が n の $\mathsf{LPN}_{k,\tau}$ 問題は，$\mathsf{SD}_{n,k,n\tau}$ 問題に変換することができる。変種として，体を $\mathbb{F}_q$ に変更した LPN 問題・仮定が用いられることもある。LPN 問題の安全性仮定については，2022 年度版の CRYPTREC 耐量子計算機暗号の研究動向調査報告書 [1, Section 3.1] も参照。

4.1.3.2 LPN 問題の拡張

以下では LPN 問題の拡張について述べる。

■Exact-LPN **問題**　誤差分布として，$\boldsymbol{e} \leftarrow \mathsf{Ber}_\tau^n$ ではなく，ハミング重みが厳密に w のものだけを考える（すなわち $\boldsymbol{e} \leftarrow \mathcal{S}_H(n, w)$）。このように誤差分布を変えた問題を Exact-LPN 問題と呼ぶ。

■Sparse-LPN **問題**　一部の暗号方式では，$\boldsymbol{s}$ のハミング重みが小さい，すなわち，疎（スパース，sparse）であることを要求する。Applebaum ら [3] は $\boldsymbol{s}$ を誤差分布である χ^k から選んだ場合の LPN 問題と $\boldsymbol{s}$ を $\mathbb{F}^k$ からランダムに選んだ LPN 問題の等価性を示している。このように $\boldsymbol{s}$ の分布を変えた問題を Sparse-LPN 問題と呼ぶ。

■Ring-LPN **問題**　Heyse，Kiltz，Lyubashevsky，Paar，Pietrzak [35] は，Ring-LPN 問題を定義した。この問題は Ring-LWE 問題（3.1.1.1 節）と同様に定義される。

定義 4.5 (探索版 Ring-LPN 問題) 適当な k 次の $\mathbb{F}_q$ 係数多項式 $f(x)$ を考え，環 $R_q = \mathbb{F}_q[x]/(f(x))$ を固定する。R_q 上の確率分布 χ を固定する。χ と $s \in R_q$ に対して，オラクル $\mathcal{O}_{s,\chi}$ を以下で定義する。(1) a を R_q から一様ランダムに選び，(2) e を分布 χ に従い選び，(3) $b = sa + e$ と計算し，(4) $(a, b) \in R_q^2$ を出力する。

探索版 Ring-LPN 問題とは，オラクル $\mathcal{O}_{s,\chi}$ へのアクセスが可能なときに，$s \in R_q$ を求める問題である。

■Module-LPN **問題**　Module-LWE 問題（3.1.1.1 節）と同様に，Module-LPN 問題を定義することができる。

定義 4.6 (探索版 Module-LPN 問題) 適当な k_0 次の $\mathbb{F}_q$ 係数多項式 $f(x)$ を考え，環 $R_q = \mathbb{F}_q[x]/(f(x))$ を固定する。R_q 上の確率分布 χ を固定する。

R_q 上の誤差分布 χ および $\boldsymbol{s} \in R_q^k$ について，オラクル $\mathcal{O}_{\boldsymbol{s},\chi}$ を以下で定義する。(1) $\boldsymbol{a}$ を R_q^k から一様ランダムに選び，(2) e を分布 χ に従い選び，(3) $b = \boldsymbol{s} \cdot \boldsymbol{a}^\top + e$ と計算し，(4) $(\boldsymbol{a}, b) \in R_q^{(k+1)}$ を出力する。

探索版 Module-LPN 問題とは，オラクル $\mathcal{O}_{\boldsymbol{s},\chi}$ へのアクセスが可能なときに，$\boldsymbol{s} \in R_q^k$ を求める問題である。

4.1.4　LPN 問題に対する評価

LPN 問題の計算の困難性に関して，サンプル数を固定した場合，NP 困難になることが Berlekamp，McEliece，van Tilborg [9] によって示されている[*3]。また，Håstad [34] により近似版 LPN 問題[*4]の NP 困難性も示されている。しかし，平均時の困難性についてはよく分かっていない。

LPN 問題の古典求解手法として，現在，大別して以下の 5 つのアルゴリズムが知られている。

1. ガウスの消去法に基づく手法 [19]
2. SD 問題における Information Set Decoding に基づく手法 [27]
3. Blum，Kalai，Wasserman [12] の BKW アルゴリズムに基づく手法
4. Arora，Ge [6] の「再線形化」アルゴリズム
5. 2. と 3. を組み合わせたハイブリッド法 [27]

このうち，漸近的に時間計算量が最も小さい手法は BKW アルゴリズムであり，実用上最も高速な手法はハイブリッド法である。以降で各手法の概要を説明する。

4.1.4.1　ガウスの消去法に基づく手法

ガウスの消去法に基づく手法は，2008 年に Carrijo，Tonicelli，Imai，Nascimento [19] によって初めて提案された LPN 問題に対する多項式空間・指数時間アルゴリズムである。この手法は指数回数の LPN オラクルの呼び出しが必要

[*3] $\boldsymbol{A}$，$\boldsymbol{b}$，自然数 w が与えられたときに，線形方程式 $\boldsymbol{s}\boldsymbol{a}_i^\top = b_i$ を満たすハミング重み w 以下の $\boldsymbol{s}$ が存在するかどうかを判定する問題。
[*4] $\boldsymbol{A}$ および $\boldsymbol{b}$ を与えられたときに，線形方程式 $\boldsymbol{s}\boldsymbol{a}_i^\top = b_i$ を近似度×最大値以上満たす $\boldsymbol{s}$ を探索する問題。

であるが，k 個の LPN サンプルを格納するメモリがあれば良いので，必要な計算資源が少なく，実装が容易である。アルゴリズムの概要は以下である。

1. LPN オラクルを k 回呼び出す: $(\boldsymbol{A}, \boldsymbol{b}) \leftarrow \mathsf{LPN}_{k,\tau}^{k}$
2. $\boldsymbol{A} \in \mathbb{F}_2^{k\times k}$ が可逆行列なら，$\boldsymbol{s}' = \boldsymbol{b}\boldsymbol{A}^{-1}$ を秘密鍵の候補とする。
3. LPN オラクルを $m = O(k)$ 回呼び出す: $(\boldsymbol{A}', \boldsymbol{b}') \leftarrow \mathsf{LPN}_{k,\tau}^{m}$
4. 閾値 $c \geq m\tau$ に対して，$\mathsf{HW}(\boldsymbol{s}'\boldsymbol{A}' + \boldsymbol{b}') \leq c$ なら, $\boldsymbol{s}'$ を解として出力する。それ以外の場合，1. からやり直す。

本アルゴリズムは，$\boldsymbol{b}$ にエラーが含まれている時とそうでない時のハミング重みの違いを用いて，秘密鍵を抽出する。エラーが全く含まれていない場合，$\boldsymbol{s}'$ は秘密鍵である。このとき，$\mathsf{HW}(\boldsymbol{s}'\boldsymbol{A}' + \boldsymbol{b}')$ の分布は $\mathsf{Ber}_{m,\tau}$ に従う。一方，エラーが含まれているとき，$\boldsymbol{s}'$ は秘密鍵ではない。このとき，$\mathsf{HW}(\boldsymbol{s}'\boldsymbol{A}' + \boldsymbol{b}')$ の分布は $\mathsf{Ber}_{m,1/2}$ に従う。この分布の違いを使えば，適切にパラメータ m, c を設定することで，高い確率で秘密鍵のみを出力できる。本手法の計算量は，k 個の LPN サンプルのエラーが全て 0 である確率が $(1-\tau)^k$ であることから，時間計算量が $\mathrm{poly}(k) \cdot O(2^k)$，空間計算量が $O(k^2)$，サンプル数が $n = O(2^k)$ となる。

4.1.4.2 Information Set Decoding に基づく手法

$\mathsf{LPN}_{k,\tau}$ は任意のサンプル数 n に対して SD 問題（$\mathsf{SD}_{n,k,\tau n}$）に変換できるため，LPN 問題は SD 問題の効率的な求解手法である Information Set Decoding を用いて解くことができる。Esser，Kübler，May [27] によって提案された実用的なアルゴリズムは，ガウスの消去法に基づく手法を拡張したものである。基本アイデアとして，ガウスの消去法に基づく手法は k 個の LPN サンプルのエラーが全て 0 の場合を考えるが，その拡張として n 個の LPN サンプルのうち k 個の LPN サンプルのエラーが全て 0 となる組み合わせを考える。$n = O(k^2)$ に設定することで，高い確率でこのような組み合わせが存在する。ガウスの消去法に基づく手法と比較して，LPN サンプルの数を $O(2^k)$ から $O(k^2)$ まで減らせる点が利点である。アルゴリズムの概要は以下である。

1. LPN オラクルを $n+m$ 回呼び出す: $(\boldsymbol{A}, \boldsymbol{b}) \leftarrow \mathsf{LPN}_{k,\tau}^{n+m}$
2. 集合 $[n] = \{1, \ldots, n\}$ から一様ランダムに k 要素を抽出した部分集合 I を 1 つ選ぶ。
3. $(\boldsymbol{A}, \boldsymbol{b})$ から集合 I の要素に対応する k 列を抽出したサンプル $(\boldsymbol{A}_I, \boldsymbol{b}_I)$ に対して，$\boldsymbol{A}_I \in \mathbb{F}_2^{k\times k}$ が可逆行列なら，$\boldsymbol{s}' = \boldsymbol{b}_I\boldsymbol{A}_I^{-1}$ を秘密鍵の候補とする。
4. $(\boldsymbol{A}', \boldsymbol{b}') \leftarrow \mathsf{LPN}_{k,\tau}^{m}$ と閾値 $c \geq m\tau$ に対して，$\mathsf{HW}(\boldsymbol{s}'\boldsymbol{A}' + \boldsymbol{b}') \leq c$ なら，$\boldsymbol{s}'$ を解として出力する。それ以外の場合，2. からやり直す。

この手法のステップ 2 以降では，$\mathsf{LPN}_{k,\tau}^{n}$ 問題に対応する $\mathsf{SD}_{n,k,\tau n}$ 問題を考える。Information Set Decoding における Prange 法 [53] を使い，$\mathsf{SD}_{n,k,\tau n}$ 問題を解く [27]。Prange 法は，$\mathsf{HW}(\boldsymbol{e}_I) = 0$ ならば，$\boldsymbol{e} = \boldsymbol{b} - \boldsymbol{s}\boldsymbol{A}$ かつ $\mathsf{HW}(\boldsymbol{e}) = \tau n$ である $\boldsymbol{e} \in \mathbb{F}_2^n$ を出力する。集合 I は，ISD における Information Set と呼ばれる集合に対応する。上のアルゴリズムは，任意の ISD に一般化できる:

1. LPN オラクルを $n+m$ 回呼び出す: $(\boldsymbol{A}, \boldsymbol{b}) \leftarrow \mathsf{LPN}_{k,\tau}^{n+m}$
2. $\mathsf{LPN}_{k,\tau}^{n}$ に対して，ISD を使って $\mathsf{SD}_{n,k,\tau n}$ を解く。出力として，解 $\boldsymbol{e} \in \mathbb{F}_2^n$ と Information Set I を得る。なお，残りの m サンプル $\mathsf{LPN}_{k,\tau}^{m}$ は ISD の内部で解の検証に用いられる。
3. $(\boldsymbol{A}_I, \boldsymbol{b}_I)$ に対して，$\boldsymbol{s} = (\boldsymbol{b}_I - \boldsymbol{e}_I)\boldsymbol{A}_I^{-1}$ は高い確率で解である。

ISD には様々な手法があるが，[27] において，MMT 法が実用上最適であることが示されている。MMT 法のような Prange 法より後に提案された ISD は，$\mathsf{HW}(\boldsymbol{e}_I) \leq p$ ならば，$\boldsymbol{e} = \boldsymbol{b} - \boldsymbol{s}\boldsymbol{A}$ かつ $\mathsf{HW}(\boldsymbol{e}) = \tau n$ である $\boldsymbol{e} \in \mathbb{F}_2^n$ を出力する。よって，Information Set にエラーが含まれている場合があるため，ステップ 3 でエラーを打ち消す必要がある。

本手法の計算量は，ISD の手法によって異なる関数 $c(\tau)$ に対して，時間計算量が $2^{c(\tau)k}$，空間計算量が $O(k^3)$，サンプル数が $O(k^2)$ となる。

4.1.4.3 BKW アルゴリズムに基づく手法

BKW アルゴリズム [12] は，LPN 問題に対する最も著名な手法である。基本アイデアは以下である。オラクルからのサンプル $(\boldsymbol{a}, b)$ が $\boldsymbol{a} = (1, 0, \ldots, 0)$ という形であれば，$b = s_1 + e$ となる。このようなサンプルを大量に集めれば，s_1 を多数決法で求めることが出来る。一般に $\boldsymbol{u}_j$ を j 番目の単位ベクトルとして，$(\boldsymbol{u}_j, b)$ という形のサンプルを集めれば s_j を多数決法で求められる。そこで，LPN オラクルからのサンプルを用いて，このようなサンプルを生成することを目指す。s_1 を求める BKW アルゴリズムの概要は以下である。なお，$s_2, \ldots, s_k$ についても同様に求められる。

1. LPN オラクルを $N = 2^{O(k/\log k)}$ 回呼び出す。
2. $\boldsymbol{a} \in \mathbb{F}_2^k$ を長さ $\ell = k/\log k$ の $t = \log k$ 個のブロックに分割する。
3. $i = 1, \ldots, t-1$ および各 $j \in \mathbb{F}_2^\ell$ について，$\boldsymbol{a}$ の接尾辞 $j|0^{(i-1)d}$ の j 毎に，全サンプルに対して 2^ℓ 個のバケツに分類する。
4. 各バケツ内で代表 $(\boldsymbol{a}^*, b^*)$ を 1 つ選び，他のサンプル $(\boldsymbol{a}, b)$ に対して $(\boldsymbol{a} + \boldsymbol{a}^*, b + b^*)$ で置換する。代表 $(\boldsymbol{a}^*, b^*)$ はバケツから除去する。これにより，$\boldsymbol{a}$ の末尾 i ブロックが全て 0 となるサンプルが得られる。
5. $\boldsymbol{a} = (a_1, \ldots, a_\ell, 0, \ldots, 0)$ であるサンプル集合に対してガウスの消去法を行い，$\boldsymbol{a} = (1, 0, \ldots, 0)$ を得る。いま，$b = \boldsymbol{s} \cdot \boldsymbol{a}^\top + e$ に対して，$b = s_1 + e$ である。
6. b について多数決を行い，多い方を s_1 とする。

BKW アルゴリズムでは，ステップ 4 で LPN サンプル同士の加算を実施することに起因してノイズが増加するため，ステップ 5 の e の分布は Ber_τ とは異なる点に注意されたい。

BKW アルゴリズムの計算量は，時間計算量・空間計算量・サンプル数いずれも $2^{O(k/\log k)}$ である。よって，大きな次元の LPN 問題に対しては，メモリ量の増加が課題となる。後述するように，良好なタイムメモリトレードオフ（時間計算量と空間計算量のトレードオフ）を持つ BKW アルゴリズムの研究開発も進められている。

■LF アルゴリズム: Levieil と Fouque [40] は BKW アルゴリズムの一部を変更し LF1 アルゴリズムを提案した。変更点は，BKW アルゴリズムの 5 行目において，$\boldsymbol{a} = (a_1, \ldots, a_\ell, 0, \ldots, 0)$ から $s_1, \ldots, s_\ell$ を総当りで計算することである。

Levieil と Fouque は，LF1 アルゴリズムに一部のヒューリスティクを組み合わせた LF2 アルゴリズムも提案している。報告によれば，$k = 99, \tau = 1/4, n = 10000$ の LPN 問題を CPU: Pentium 4（3GHz），RAM: 1GB のマシンで解くことが可能である。Devadas，Ren，Xiao [22] は LF2 アルゴリズムについて詳細な解析を与え，BKW アルゴリズムとの比較を行っている。Devadas らの報告によれば，LF2 アルゴリズムは BKW アルゴリズムより時間計算量が少ない分，メモリ使用量が増加するとのことである。

■Kirchner の手法: Kirchner [37] は一様ランダムに選ばれた $\boldsymbol{s}$ よりは Ber_τ に従って選ばれる誤りベクトル $\boldsymbol{e}$ の方が，ハミング重みが小さく，取りうる値が少ないことに着目した。そこで，LPN 問題を Sparse-LPN 問題に置き換えた上で問題を解く手法を提案している。一般の $\boldsymbol{s}$ であれば，総当りに必要な回数は 2^ℓ となる。一方，$\boldsymbol{e}$ は疎であることが期待されるため，$\boldsymbol{e}$ の総当りに必要な回数が削減される。

■Ring-LPN 問題への応用: Bernstein と Lange [10] は Levieil と Fouque の高速化手法および Kirchner のアイデアを用いることにより，Ring-LPN 問題の解法が高速化できることを示している。

■Coded-BKW: Guo，Johansson，Löndahl [33] は，covering codes と呼ばれる符号を用いて，Kirchner の手法 [37] および Bernstein と Lange [10] の手法を改良した Coded-BKW を提案した。Kirchner の手法では，BKW アルゴ

リズムのステップ 5 において，$\boldsymbol{a}$ を covering code の受信語とみなすことで探索空間の圧縮を行い，高速化を行っている[*5]。Zhang，Jiao，Wang [56] は別の符号を用いて Coded-BKW を改良している。Bogos と Vaudenay [13] は Coded-BKW の解析が一部欠けていることを分析し，最適化を行いつつ詳細な計算量評価を与えた。

■Dissection-BKW: Esser，Heuer，Kübler，May，Sohler [26] は BKW アルゴリズムのタイムメモリトレードオフ手法である Dissection-BKW を提案した。Dissection-BKW は，Dinur，Dunkelman，Keller，Shamir [23] によって提案された部分和問題に対するタイムメモリトレードオフ手法である Dissection を，BKW アルゴリズムに適用したものである。

4.1.4.4 Arora–Ge アルゴリズム

Arora と Ge [6] は多変数多項式問題で古くから用いられている再線形化と呼ばれる手法を用いて，LPN 問題を解くアルゴリズムを提案した。このアルゴリズムをサンプル数 n の $\mathsf{LPN}_{k,\tau}$ 問題に用いた場合，$w = \tau n$ として，$\mathrm{poly}(k^w)$ 時間で解くことができる。$\mathrm{poly}(k^w) = 2^{O(\tau n \log k)}$ であるから，$\tau = o(k/(n\log^2 k))$ のようにエラーが疎であれば，BKW アルゴリズムよりも効率が良い。実際の符号暗号のパラメータ設定では，エラーをこのように疎に設定することはないため，暗号の攻撃アルゴリズムとして用いるには重要度が低い。

4.1.4.5 Information Set Decoding と BKW を組み合わせたハイブリッド法

Esser，Kübler，May [27] は，BKW アルゴリズムと Information Set Decoding を組み合わせた実用上高速な手法を提案した。ハイブリッド法のアルゴリズムの概要は以下である。

1. 次元削減パラメータ $k_1, k_2, k' = k - k_1 - k_2$ を決定する。
2. $\boldsymbol{a}$ の末尾 k_1 列が 0 である LPN サンプルを一定数集める。
3. 集められたサンプルの k_2 列に対して BKW アルゴリズムを行い，$\boldsymbol{a}$ の末尾 $k_1 + k_2$ 列が 0 となるサンプルを一定数集める。これによって，$s_{k'+1}, \ldots, s_{k'+k_2}$ が求まる。
4. 集められたサンプルからなる LPN 問題を SD 問題に変換し，ISD で解く。これによって，$s_1, \ldots, s_{k'}$ が求まる。
5. 残りの $s_{k'+k_2+1}, \ldots, s_k$ を求める。k_1 の値の大きさに応じて，全探索・ISD・再度ステップ 1 から処理をやり直すといった方策を取る。

本手法の計算量は，Information Set Decoding に基づく手法の計算量と BKW アルゴリズムに基づく手法の計算量との中間である。報告によれば，$k = 135, \tau = 1/4$ の LPN 問題に対して，$k_1 = 10, k_2 = 99, k' = 26$ のパラメータを使用し，16 コアの CPU および 256GB の RAM を搭載したサーバ 1 台を用いて，5.69 日での求解に成功した。また，$k = 243, \tau = 1/8$ の LPN 問題に対して，$k_1 = 35, k_2 = 0, k' = 208$ のパラメータを使用し，同じサーバ 1 台を用いて，15.07 日での求解に成功した。

また，暗号設計に用いられるパラメータを持つ LPN 問題に対して，空間計算量を現実的な値にセキュリティマージンを加えたものに制限（2^{60}bit = 128PBytes および 2^{80}bit = 128ZBytes）した時のビット計算量が推定された。報告によれば，$k = 512, \tau = 1/8$ の LPN 問題に対するハイブリッド法のビット計算量は 2^{102} であり，$k = 512, \tau = 1/4$ の LPN 問題に対するビット計算量は 2^{151} である。一方で，この空間計算量の範囲では Coded-BKW [33] は動作しないと報告されている。

[*5] ただし，国際会議でのプレゼンテーションではサンプル数が不足していたとの報告があり，計算量・メモリ・サンプル数の評価は見直されている。詳しくは，[56] および [13] を参照のこと

4.1.4.6 量子アルゴリズム

現在のところ，多項式時間で LPN 問題を解く量子アルゴリズムは提案されていない。Esser，Kübler，May [27] は，上述する ISD に基づく手法やハイブリッド法に対して，グローバーのアルゴリズムや量子ウォーク探索を用いることで高速化できる点を指摘している。$k = 512, \tau = 1/8$ の LPN 問題に対する量子ハイブリッド法のビット計算量は 2^{69} であり，$k = 512, \tau = 1/4$ の LPN 問題に対するビット計算量は 2^{112} と推定されている。

4.2 符号に基づく代表的な暗号方式

本節では，符号に基づく代表的な暗号方式と署名方式の説明を行う。以下では，$\mathrm{GL}_k(\mathbb{F}_q)$ で k 次の $\mathbb{F}_q$ 要素正則行列全体がなす群を表す。また，S_n で n 次対称群を表す。S_n の要素である置換を $\mathrm{GL}_n(\mathbb{F}_q)$ 中の置換行列と同一視する。

4.2.1 McEliece 公開鍵暗号方式

McEliece [45] が提案した古典的な暗号方式である。以下では $q = 2$ とする。

- k：安全性パラメータ
- n：サンプルの個数
- τ：誤差パラメータ（例: $\tau n = O(k)$）
- t：線形符号の誤り訂正能力（$t = \Omega(\tau n)$）

鍵生成: 誤り訂正能力が t である $[n, k]_2$-線形符号の生成行列 $\boldsymbol{G} \in \mathbb{F}_2^{k \times n}$ を生成する。$\boldsymbol{S} \leftarrow \mathrm{GL}_k(\mathbb{F}_2)$ を一様ランダムに選ぶ。$\boldsymbol{P} \leftarrow S_n$ を一様ランダムに選ぶ。$\tilde{\boldsymbol{G}} = \boldsymbol{SGP}$ とする。
公開鍵を $\tilde{\boldsymbol{G}}$ とし，秘密鍵を $(\boldsymbol{S}, \boldsymbol{G}, \boldsymbol{P})$ とする。
暗号化: 平文を $\boldsymbol{m} \in \mathbb{F}_2^k$ とする。乱数 $\boldsymbol{e} \leftarrow \mathsf{Ber}_\tau^n$ を選び，暗号文 $\boldsymbol{c} = \boldsymbol{m}\tilde{\boldsymbol{G}} + \boldsymbol{e} \in \mathbb{F}_2^n$ を計算する。
復号: $\hat{\boldsymbol{v}} = \boldsymbol{cP}^{-1}$ を計算する。$\hat{\boldsymbol{v}}$ を線形符号で訂正し $\boldsymbol{m}' \in \mathbb{F}_2^k$ を得る。$\boldsymbol{m} = \boldsymbol{m}'\boldsymbol{S}^{-1}$ を出力する。

復号の正当性は以下で確認される。$\boldsymbol{c} = \boldsymbol{m}\tilde{\boldsymbol{G}} + \boldsymbol{e}$ として，$\hat{\boldsymbol{v}} = \boldsymbol{cP}^{-1}$ を計算すると，

$$\hat{\boldsymbol{v}} = \boldsymbol{m}\tilde{\boldsymbol{G}}\boldsymbol{P}^{-1} + \boldsymbol{e}\boldsymbol{P}^{-1} = \boldsymbol{mSG} + \boldsymbol{e}\boldsymbol{P}^{-1}$$

を得る。$\boldsymbol{mSG}$ はランダム化されたメッセージ $\boldsymbol{mS}$ の符号語であり，$\boldsymbol{e}\boldsymbol{P}^{-1}$ は誤りである。$\boldsymbol{e}\boldsymbol{P}^{-1}$ のハミング重みが t 以下であれば，線形符号の復号により，$\boldsymbol{m}' = \boldsymbol{mS}$ を得る。よって，高い確率で復号に成功する。平文 $\boldsymbol{m}$ および生成行列 $\tilde{\boldsymbol{G}}$ が一様ランダムであれば，暗号文 $\boldsymbol{c} \in \mathbb{F}_2^n$ はランダムな n 次元のベクトルと見分けが付かないと考えられている。一方で，平文 $\boldsymbol{m}$ が零ベクトルのとき，暗号文はランダムなベクトルと区別されてしまう。このことから，オリジナルの McEliece 暗号にはセキュリティ上の課題が存在することがわかる。

4.2.2 Niederreiter 公開鍵暗号方式

Niederreiter [49] が 1986 年に提案した。のちに McEliece 暗号と安全性が等価であることが示された。詳しくは [41] を参照のこと。以下では $q = 2$ とする。

- k：安全性パラメータ
- n：サンプルの個数
- t：線形符号の誤り訂正能力

鍵生成: 誤り訂正能力が t である $[n, k]_2$ -線形符号のパリティ検査行列 $\boldsymbol{H} \in \mathbb{F}_2^{(n-k) \times n}$ を生成する。$\boldsymbol{T} \leftarrow \mathrm{GL}_{n-k}(\mathbb{F}_2)$

を一様ランダムに選ぶ。$\boldsymbol{Q} \leftarrow S_n$ を一様ランダムに選ぶ。$\tilde{\boldsymbol{H}} = \boldsymbol{THQ}$ とする。
公開鍵を $\tilde{\boldsymbol{H}}$ とし，秘密鍵を $(\boldsymbol{T}, \boldsymbol{H}, \boldsymbol{Q})$ とする。

暗号化: 平文を $\boldsymbol{e} \in \mathcal{S}_H(n,t)$ とする。暗号文 $\boldsymbol{d} = \boldsymbol{e}\tilde{\boldsymbol{H}}^\top \in \mathbb{F}_2^{n-k}$ を計算する。

復号: $\hat{\boldsymbol{w}} = \boldsymbol{dT}^{-\top}$ を計算する。$\hat{\boldsymbol{w}}$ を線形符号で訂正し復号し，誤りとして $\boldsymbol{e}'$ を得る。$\boldsymbol{e} = \boldsymbol{e}'\boldsymbol{Q}^{-\top}$ を出力する。

復号の正当性は以下で確認される。$\boldsymbol{d} = \boldsymbol{e}\tilde{\boldsymbol{H}}^\top$ として，$\hat{\boldsymbol{w}} = \boldsymbol{dT}^{-\top}$ を計算すると，

$$\hat{\boldsymbol{v}} = \boldsymbol{e}\tilde{\boldsymbol{H}}^\top\boldsymbol{T}^{-\top} = \boldsymbol{eQ}^\top\boldsymbol{H}^\top\boldsymbol{T}^\top\boldsymbol{T}^{-\top} = \boldsymbol{eQ}^\top\boldsymbol{H}^\top$$

を得る。$\boldsymbol{eQ}^\top$ はランダムに置換されたエラーであり，$\boldsymbol{eQ}^\top\boldsymbol{H}^\top$ はシンドロームである。$\boldsymbol{eQ}^\top$ のハミング重みが t 以下であれば，線形符号の復号により，$\boldsymbol{e}' = \boldsymbol{eQ}^\top$ を得る。よって，高い確率で復号に成功する。平文 $\boldsymbol{e}$ およびパリティ検査行列 $\tilde{\boldsymbol{H}}$ が一様ランダムであれば，暗号文 $\boldsymbol{d} \in \mathbb{F}_2^{n-k}$ はランダムな $n-k$ 次元のベクトルと見分けが付かないと考えられている。また，$\tilde{\boldsymbol{H}}$ が一様ランダムであり，適切な t が選択されていれば，暗号文は統計的にランダムなベクトルと見分けが付かないとされている。一方で，オリジナルの Niederreiter 暗号は適応的選択暗号文攻撃（CCA）に対して安全ではないため，次節で示すより安全な方式が提案されている。

4.2.3 符号版 Lyubashevsky-Peikert-Regev（LPR）公開鍵暗号方式

符号版 LPR 暗号は，Lyubashevsky，Peikert，Regev が 2010 年に提案した Ring-LWE 問題に基づく暗号方式 [42] を LPN 問題に基づく方式に変更したものである。以下では $q = 2$ とする。

- k：安全性パラメータ
- $n = n_1 + n_2$：サンプルの個数
- ℓ：平文長
- τ：誤差パラメータ（例: $\tau n = O(\sqrt{k})$）
- t：線形符号の誤り訂正能力（$t = \Omega\big((\tau n)^2\big)$）

鍵生成: 誤り訂正能力が t である $[n_2, \ell]_2$ -線形符号の生成行列 $\boldsymbol{G}_c$ を生成する。$\boldsymbol{A} \leftarrow \mathbb{F}_2^{k\times n_1}$ とする。$\boldsymbol{X} \leftarrow \mathsf{Ber}_\tau^{n_1\times n_2}, \boldsymbol{Y} \leftarrow \mathsf{Ber}_\tau^{k\times n_2}$ とし，$\boldsymbol{B} = \boldsymbol{AX} + \boldsymbol{Y} \in \mathbb{F}_2^{k\times n_2}$ とする。
公開鍵を $\tilde{\boldsymbol{G}} = [\boldsymbol{A} \mid \boldsymbol{B}] \in \mathbb{F}_2^{k\times n}$ とし，秘密鍵を $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{X})$ とする。

暗号化: 平文を $\boldsymbol{m} \in \mathbb{F}_2^\ell$ とする。乱数 $\boldsymbol{s} \leftarrow \mathsf{Ber}_\tau^k$ と 乱数 $\boldsymbol{e} \leftarrow \mathsf{Ber}_\tau^n$ を選び，暗号文 $\boldsymbol{c} = \boldsymbol{s}\tilde{\boldsymbol{G}} + \boldsymbol{e} + (\boldsymbol{0}_{n_1}, \boldsymbol{mG}_c) \in \mathbb{F}_2^n$ を計算する。

復号: $\boldsymbol{d} = \boldsymbol{c}\binom{-\boldsymbol{X}}{\boldsymbol{I}_{n_2}}$ を計算する。$\boldsymbol{d}$ を線形符号で訂正し復号すると $\boldsymbol{m}$ を得る。

復号の正当性は以下で確認される。$\boldsymbol{c} = \boldsymbol{s}\tilde{\boldsymbol{G}} + \boldsymbol{e} + (\boldsymbol{0}_{n_1}, \boldsymbol{mG}_c)$ なので，前半部を $\boldsymbol{u} = \boldsymbol{sA} + \boldsymbol{e}_1$，後半部を $\boldsymbol{v} = \boldsymbol{sB} + \boldsymbol{e}_2 + \boldsymbol{mG}_c$ と書く。
$\boldsymbol{d} = \boldsymbol{c}\binom{-\boldsymbol{X}}{\boldsymbol{I}_{n_2}}$ を計算すると，

$$\boldsymbol{d} = \boldsymbol{v} - \boldsymbol{uX} = \boldsymbol{sB} + \boldsymbol{e}_2 + \boldsymbol{mG}_c - \boldsymbol{sAX} - \boldsymbol{e}_1\boldsymbol{X} = \boldsymbol{mG}_c + (\boldsymbol{e}_2 - \boldsymbol{e}_1\boldsymbol{X} + \boldsymbol{sY})$$

を得る。$\boldsymbol{mG}_c$ は符号語であり，$\boldsymbol{e}_2 - \boldsymbol{e}_1\boldsymbol{X} + \boldsymbol{sY}$ は誤りベクトルである。よって，$\boldsymbol{e}_2 - \boldsymbol{e}_1\boldsymbol{X} + \boldsymbol{sY}$ のハミング重みが t 以下であれば，線形符号の復号により，$\boldsymbol{m}$ を得る。高い確率で $\boldsymbol{e}_2 - \boldsymbol{e}_1\boldsymbol{X} + \boldsymbol{sY}$ のハミング重みが t 以下になるように τ を設定しているため，高い確率で復号に成功する。

表 4.2: 符号に基づく暗号の分類

文献	暗号化	鍵交換	署名
Classic McEliece [2]	○	○	–
BIKE [4]	○	○	–
HQC [46]	○	○	–

4.2.4 CFS 署名方式

CFS 署名方式は Courtois，Finiasz，Sendrier が 2001 年に提案した署名方式である [21]。のちに，安全性仮定が提案パラメータセットでは成り立たないことが示された [31, 32]。しかし後の方式に大きな影響を与えたため，ここに記す。Niederreiter 暗号では秘密鍵を持っている場合，ハミング重みが t 以下の誤りは訂正できる。一方，訂正可能なシンドロームの集合 $\{\boldsymbol{e}\tilde{\boldsymbol{H}} \in \mathbb{F}_2^{n-k} \mid \boldsymbol{e} \in \mathcal{S}_{\tilde{H}}^{\leq}(n,t)\}$ のサイズは $\mathbb{F}_2^{n-k}$ のサイズに比べれば圧倒的に少ない。

3.2.5 節のように Hash-and-Sign に基づいた構成を考える。メッセージ M のハッシュ値を シンドローム $\boldsymbol{u} \in \mathbb{F}_2^{n-k}$ と捉えた場合，正しく復号できないシンドロームになることが多い。そこで CFS 署名では，ハッシュ値を $\boldsymbol{u} = \mathsf{Hash}(M,i)$ と i をインクリメントしながら計算し，ハッシュ値が $\{\boldsymbol{e}\tilde{\boldsymbol{H}} \in \mathbb{F}_2^{n-k} \mid \boldsymbol{e} \in \mathcal{S}_{\tilde{H}}^{\leq}(n,t)\}$ に入るものを採用する。

署名鍵と検証鍵: パリティ検査行列 $\tilde{\boldsymbol{H}} \in \mathbb{F}_2^{(n-k)\times n}$ を検証鍵とする。また署名鍵を用いると，ハミング重み t 以下の符号語を訂正できることとする。

署名: 文書 M について，

1. $i = 0$ とする。
2. $\boldsymbol{u} = \mathsf{Hash}(M,i)$ を計算する。
3. ハミング重み t 以下の $\boldsymbol{e}$ で，$\boldsymbol{e}\tilde{\boldsymbol{H}}^\top = \boldsymbol{u}$ となるものを計算する。なければ $i \leftarrow i+1$ としてステップ 2 に戻る。
4. $\sigma = (\boldsymbol{e}, i)$ を出力する。

検証: 文書 M と $\sigma = (\boldsymbol{e}, i)$ について，$\mathsf{HW}(\boldsymbol{e}) \leq t$ と $\boldsymbol{e}\tilde{\boldsymbol{H}}^\top = \mathsf{Hash}(M,i)$ ならば，受理する。そうでないならば，棄却とする。

4.3 符号に基づく主要な暗号方式

本稿では以下の暗号方式を取り上げる。いずれも NIST PQC 標準化プロジェクトにおいて第 4 ラウンドに進んだものである。

1. Classic McEliece: Niederreiter 暗号を採用し，符号の構成が非常に保守的という観点からこれを取り上げる。
2. BIKE: Niederreiter 暗号を採用し，QC-MDPC 符号を用いて鍵を圧縮している，という観点からこれを取り上げる。
3. HQC: 符号版の LPR 暗号を採用，Quasi-Cyclic 符号を用いて鍵を圧縮している，という特徴からこれを取り上げる。

4.3.1 Classic McEliece

- 提案者: Albrecht, Bernstein, Chou, Cid, Gilcher, Lange, Maram, von Maurich, Misoczki, Niederhagen, Paterson, Persichetti, Peters, Schwabe, Sendrier, Szefer, Tjhai, Tomlinson, Wang
- 基本方式の説明: Niederreiter 暗号方式に基づいている。基本符号方式として $\mathbb{F}_2$ 上の Goppa 符号を利用している。（具体的な Goppa 符号の生成方法や符号化および復号の方法については提案方式の仕様書 [2] を参照のこと。）$q = 2^m$ とし，$n \leq q$ を用いる。2 以上の t を $mt < n$ となるように取り，$k = n - mt$ とする。

  鍵生成: 誤り訂正能力が t である Goppa 符号のパリティ検査行列 $\boldsymbol{H} \in \mathbb{F}_2^{(n-k)\times n}$ をランダムに生成する。組織符号化し，$\tilde{\boldsymbol{H}} = [\boldsymbol{I}_{n-k} \mid \boldsymbol{T}]$ とする。公開鍵を $pk = \boldsymbol{T} \in \mathbb{F}_2^{(n-k)\times k}$ とする。符号生成に使ったパラメータを Γ（$\mathbb{F}_q$ 係数の t 次モニック既約多項式と互いに異なる $\alpha_0, \ldots, \alpha_{n-1} \in \mathbb{F}_q$）とする。秘密鍵を $sk = \Gamma$ とする。

  暗号化 $\mathsf{Encrypt}(pk, \boldsymbol{e})$：入力を $\boldsymbol{e} \in \mathcal{S}_H(n, t)$ とする。$\tilde{\boldsymbol{H}} = [\boldsymbol{I}_{n-k} \mid \boldsymbol{T}]$ とし，暗号文として $\boldsymbol{c} = \tilde{\boldsymbol{H}}\boldsymbol{e} \in \mathbb{F}_2^{n-k}$ を出力する。

  復号 $\mathsf{Decrypt}(sk, \boldsymbol{c})$：ハミング重み t のベクトル $\boldsymbol{e}$ を復号する。

  1. $\boldsymbol{c}$ に k 個ゼロを加え，$\boldsymbol{v} = (\boldsymbol{c}, \boldsymbol{0}_k) \in \mathbb{F}_2^n$ を考える。
  2. Goppa 符号の復号アルゴリズムを用いて，$\boldsymbol{v}$ と距離 t 以下にある符号語 $\boldsymbol{d}$ を計算する。（なければ $\perp$ を出力する。）
  3. $\boldsymbol{e} = \boldsymbol{v} + \boldsymbol{d}$ とする。
  4. $\mathsf{HW}(\boldsymbol{e}) = t$ かつ $\boldsymbol{c} = \tilde{\boldsymbol{H}}\boldsymbol{e}$ ならば $\boldsymbol{e}$ を出力する。（そうでなければ $\perp$ を出力する。）
- 鍵カプセル化方式の説明: 基本方式を決定性の公開鍵暗号とみなし，藤崎–岡本変換の変種をかけたものとみなせる。以下ではハッシュ関数 $\mathsf{H}\colon \{0,1\}^* \to \{0,1\}^{256}$ を用いる。

  鍵生成: ℓ ビットのシード δ から乱数を生成し，鍵生成を行う。（乱数の生成方法は省略する。）公開鍵は同じく $pk = \boldsymbol{T}$ である。n ビットの一様ランダムな文字列 $\boldsymbol{s}$ を生成する。秘密鍵は $sk = (\Gamma, \boldsymbol{s})$ である。

  鍵カプセル化:

  1. $\boldsymbol{e} \leftarrow \mathcal{S}_H(n, t)$ をランダムにサンプリングする。
  2. $\boldsymbol{c} = \mathsf{Encrypt}(pk, \boldsymbol{e})$ を計算する。
  3. $K = \mathsf{H}(1, \boldsymbol{e}, \boldsymbol{c})$ とする。
  4. 暗号文を $\boldsymbol{c}$ とし，セッション鍵 K を出力する。

  デカプセル化:

  1. $b = 1$ とする。
  2. 受信した $\boldsymbol{c}$ に対して，$\boldsymbol{e} = \mathsf{Decrypt}(sk, \boldsymbol{c})$ とする。$\boldsymbol{e} = \perp$ であれば，$b = 0, \boldsymbol{e} = \boldsymbol{s}$ と上書きする。
  3. $K = \mathsf{H}(b, \boldsymbol{e}, \boldsymbol{c})$ を計算する。
  4. K を出力する。

  以上より，セッション鍵（共通鍵）K を安全に共有することができる。

パラメータセットとして mceliece348864，mceliece348864f，mceliece460896，mceliece460896f，mceliece6688128，mceliece6688128f, mceliece6960119, mceliece6960119f, mceliece8192128, mceliece8192128f が提案されている。表 4.3 に鍵カプセル化方式のパラメータ，鍵長および暗号文長，想定セキュリティレベル，復号エラー率をまとめた。末尾に f が付くものは扱っていないが，鍵長・暗号文長は f 無しのものと同一である。Classic McEliece は公開鍵長が非常に大きく，レベル 5 では 1MBytes を超える。一方で，暗号文長は非常に小さく，格子暗号に基づく FIPS 標準（FIPS 203）である ML-KEM [50] の暗号文サイズよりも小さい。例えば，[50, Table 3] によれば，ML-KEM のレベル 1 の

表 4.3: Classic McEliece のパラメータ。公開鍵長，秘密鍵長，暗号文長の単位はそれぞれ Byte とする。

パラメータ名	(m, n, t)	安全性レベル	公開鍵長	秘密鍵長	暗号文長	復号エラー率
mceliece348864	$(12, 3488, 64)$	レベル 1	$261,120$	$6,492$	96	0
mceliece460896	$(13, 4608, 96)$	レベル 3	$524,160$	$13,608$	156	0
mceliece6688128	$(13, 6688, 128)$	レベル 5	$1,044,992$	$13,932$	208	0
mceliece6960119	$(13, 6960, 119)$	レベル 5	$1,047,319$	$13,948$	194	0
mceliece8192128	$(13, 8192, 128)$	レベル 5	$1,357,824$	$14,120$	208	0

公開鍵長は 800 Bytes，秘密鍵長は 1632 Bytes，暗号文長は 768 Bytes となっている。

mceliece348864 の速度に関しては，鍵生成に必要な平均 CPU サイクル数が $60,333,686$ Cycle，鍵カプセル化が $37,585$ Cycle，デカプセル化が $127,668$ Cycle である。参考までに，ML-KEM（Kyber-512）の速度は，鍵生成が $33,428$ Cycle，鍵カプセル化が $49,184$ Cycle，デカプセル化が $40,564$ Cycle である [50]。なお，いずれも Haswell CPU 搭載のサーバ上で AVX 命令を使用した C 言語実装を動作させた時の記録である。他のパラメータに関しては，仕様書を参照されたい。

4.3.2 BIKE

- 提案者: Aragon, Barreto, Bettaieb, Bidoux, Blazy, Deneuville, Gaborit, Gueron, Güneysu, Aguilar Melchor, Misoczki, Persichetti, Sendrier, Tillich, Zémor, Vasseur, Ghosh, Richter-Brokmann
- 基本方式の説明: Niederreiter 暗号方式に基づいている。基本となる符号に QC-MDPC 符号を採用し，公開鍵サイズを圧縮している。そのため，鍵や暗号化は格子暗号の一種の NTRU 暗号と非常に近い形をしている点が特徴である。具体的な符号化および復号の方法については提案方式の仕様書 [4] を参照のこと。以下では，$\mathcal{R} = \mathbb{F}_2[X]/(X^n - 1)$ とする。

  鍵生成: $\boldsymbol{h}_0 \in \mathcal{R}$ および $\boldsymbol{h}_1 \in \mathcal{R}$ を $\mathcal{S}_H(n, w/2)$ から一様ランダムに選ぶ。$\boldsymbol{h} = \boldsymbol{h}_1/\boldsymbol{h}_0 \in \mathcal{R}$ とする。$(\boldsymbol{h}_0, \boldsymbol{h}_1)$ を QC-MDPC 符号のパリティ検査行列とし，$(\mathbf{1}, \boldsymbol{h})$ をその組織符号化したものとみなすことができる。公開鍵を $pk = \boldsymbol{h}$ とし，秘密鍵を $sk = (\boldsymbol{h}_0, \boldsymbol{h}_1)$ とする。

  暗号化 $\mathsf{Encrypt}(pk, (\boldsymbol{e}_0, \boldsymbol{e}_1))$: $(\boldsymbol{e}_0, \boldsymbol{e}_1) \in \mathcal{R}^2$ を $\mathcal{S}_H(2n, t)$ 中のベクトルとみなす。$\boldsymbol{c} = \boldsymbol{e}_0 + \boldsymbol{e}_1\boldsymbol{h} \in \mathcal{R}$ を出力する。

  復号 $\mathsf{Decrypt}(sk, \boldsymbol{c})$: ハミング重み t 以下のベクトル $(\boldsymbol{e}_0, \boldsymbol{e}_1)$ を復号する。

  1. $\boldsymbol{c}\boldsymbol{h}_0$ を計算する。
  2. QC-MDPC 符号の復号アルゴリズムを用いて，$\boldsymbol{c}\boldsymbol{h}_0$ をシンドロームとするベクトル $(\boldsymbol{e}_0, \boldsymbol{e}_1)$ を計算する。
- 鍵カプセル化方式の説明: 基本方式を決定性公開鍵暗号方式とみなす。基本方式とハッシュ関数 $\mathsf{L}\colon \{0,1\}^* \to \{0,1\}^{256}$ を用いて，平文 $\boldsymbol{m} \in \{0,1\}^{256}$ と乱数 $(\boldsymbol{e}_0, \boldsymbol{e}_1)$ に対して暗号化 $(\boldsymbol{c}_0 = \mathsf{Encrypt}(pk, (\boldsymbol{e}_0, \boldsymbol{e}_1)))$ および $\boldsymbol{m}$ のマスキング $(\boldsymbol{c}_1 = \boldsymbol{m} \oplus \mathsf{L}(\boldsymbol{e}_0, \boldsymbol{e}_1))$ とを行う IND-CPA 安全な乱択公開鍵暗号を構成する。鍵カプセル化方式は，この乱択公開鍵暗号に藤崎–岡本変換の変種を適用したものとみなせる。以下ではハッシュ関数 $\mathsf{H}, \mathsf{L}\colon \{0,1\}^* \to \{0,1\}^{256}$ と $\mathsf{G}\colon \{0,1\}^* \to \mathcal{S}_H(2n, t)$ を用いる。

  鍵生成: 適切な長さのシード δ から乱数を生成し，鍵生成を行う。公開鍵は同じく $pk = \boldsymbol{h}$ である。ℓ ビットの一様ランダムな文字列 $\boldsymbol{s} \in \{0,1\}^\ell$ を生成する。秘密鍵は $sk = (\boldsymbol{h}_0, \boldsymbol{h}_1, \boldsymbol{s})$ である。

鍵カプセル化:

1. $\boldsymbol{m} \leftarrow \{0,1\}^{256}$ を一様ランダムに選ぶ。
2. $(\boldsymbol{e}_0, \boldsymbol{e}_1) = \mathsf{G}(\boldsymbol{m})$ を計算する。
3. $\boldsymbol{c}_0 = \mathsf{Encrypt}(pk, (\boldsymbol{e}_0, \boldsymbol{e}_1))$ と，$\boldsymbol{c}_1 = \boldsymbol{m} \oplus \mathsf{L}(\boldsymbol{e}_0, \boldsymbol{e}_1)$ を計算する。
4. $K = \mathsf{H}(\boldsymbol{m}, \boldsymbol{c})$ を計算する。
5. 暗号文を $C = (\boldsymbol{c}_0, \boldsymbol{c}_1)$ とし，セッション鍵 K を出力する。

デカプセル化:

1. 受信した C に対して，$(\boldsymbol{e}'_0, \boldsymbol{e}'_1) = \mathsf{Decrypt}(sk, \boldsymbol{c}_0)$ を計算する。
2. 復号に失敗したら，$\perp$ を出力して停止する。
3. $\boldsymbol{m}' = \boldsymbol{c}_1 \oplus \mathsf{L}(\boldsymbol{e}'_0, \boldsymbol{e}'_1)$ を計算する。
4. $(\boldsymbol{e}'_0, \boldsymbol{e}'_1) = \mathsf{G}(\boldsymbol{m}')$ ならば，$K = \mathsf{H}(\boldsymbol{m}', \boldsymbol{c})$ を出力して停止する。
5. そうでなければ，$K = \mathsf{H}(\boldsymbol{s}, \boldsymbol{c})$ を計算し，出力する。

以上より，セッション鍵（共通鍵）K を安全に共有することができる。

表 4.4 に鍵カプセル化方式のパラメータ，鍵長，暗号文長および復号エラー率をまとめた。3 つのパラメータセットがそれぞれレベル 1，3，5 相当として提案された。BIKE-Level1 の速度に関しては，鍵生成が 589,000 Cycle，鍵カプセル化が 97,000 Cycle，デカプセル化が 1,135,000 Cycle である（Skylake CPU 搭載のサーバ，AVX 命令を使用）。他のパラメータに関しては，仕様書を参照されたい。

表 4.4: BIKE のパラメータ。公開鍵長，秘密鍵長，暗号文長の単位はそれぞれ Byte とする。

パラメータ名	(n, w, t)	安全性レベル	公開鍵長	秘密鍵長	暗号文長	復号エラー率
BIKE-Level1	$(12323, 142, 134)$	レベル 1	$1,541$	281	$1,573$	2^{-128}
BIKE-Level3	$(24659, 206, 199)$	レベル 3	$3,083$	419	$3,115$	2^{-192}
BIKE-Level5	$(40973, 274, 264)$	レベル 5	$5,122$	580	$5,154$	2^{-256}

4.3.3 HQC

- 提案者: Aguilar Melchor，Aragon，Bettaieb，Bidoux，Blazy，Deneuville，Gaborit，Persichetti，Zémor，Bos，Dion，Lacan，Robert，Veron
- 基本方式の説明: 符号版の LPR 暗号に基づき，公開鍵暗号を構成している。
  $\mathcal{R} = \mathbb{F}_2[X]/(X^n - 1)$ とする。$n' = n_1 n_2$ とし，$[n', k]$ 線形符号 $\mathcal{C}$ を採用する。具体的な符号化および復号の方法については提案方式の仕様書 [46] を参照のこと。線形符号 $\mathcal{C}$ の符号化・復号アルゴリズムを $\mathsf{encode}, \mathsf{decode}$ とする。$n \geq n'$ を仮定する。以下では，暗号文の第二要素 $\boldsymbol{v}$ を $\mathcal{R}$ 要素（n ビットベクトル）として扱っているが，実際には n' ビットに縮めて用いる。
  鍵生成: $\boldsymbol{x} \in \mathcal{R}$ および $\boldsymbol{y} \in \mathcal{R}$ を $\mathcal{S}_H(n, w)$ から一様ランダムに選び，$\boldsymbol{h} \leftarrow \mathcal{R}$ に対して公開鍵を $pk = (\boldsymbol{h}, \boldsymbol{s}) \in \mathcal{R}^2$ とし，秘密鍵を $sk = (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{R}^2$ とする。
  暗号化 $\mathsf{Encrypt}(pk, \boldsymbol{m}, \boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{e})$: $\boldsymbol{r}_1 \in \mathcal{R}$ および $\boldsymbol{r}_2 \in \mathcal{R}$ を $\mathcal{S}_H(n, w_r)$ から一様ランダムに選び，$\boldsymbol{e} \in \mathcal{R}$ を $\mathcal{S}_H(n, w_e)$ から一様ランダムに選ぶ。$\boldsymbol{u} = \boldsymbol{r}_1 + \boldsymbol{h} \cdot \boldsymbol{r}_2$ および $\boldsymbol{v} = \mathsf{encode}(m) + \boldsymbol{s} \cdot \boldsymbol{r}_2 + \boldsymbol{e}$ を計算する。$\boldsymbol{c} = (\boldsymbol{u}, \boldsymbol{v})$ を暗号文として出力する。
  復号 $\mathsf{Decrypt}(sk, \boldsymbol{c})$: $\mathsf{decode}(\boldsymbol{v} - \boldsymbol{u} \cdot \boldsymbol{y})$ を出力する。
- 鍵カプセル化方式: 基本方式を乱択な公開鍵暗号とみなし，藤崎–岡本変換の変種を適用したものとみなせる。

表 4.5: HQC のパラメータ。公開鍵長，秘密鍵長，暗号文長の単位はそれぞれ Byte とする。

パラメータ名	$(n_1, n_2, n, w, w_r = w_e)$	安全性レベル	公開鍵長	秘密鍵長	暗号文長	復号エラー率
hqc-128	$(46, 384, 17669,\ 66,\ 75)$	レベル 1	$2,249$	40	$4,497$	2^{-128}
hqc-192	$(56, 640, 35851, 100, 114)$	レベル 3	$4,522$	40	$9,042$	2^{-192}
hqc-256	$(90, 640, 57637, 131, 149)$	レベル 5	$7,245$	40	$14,485$	2^{-256}

以下ではハッシュ関数 $\mathsf{H}, \mathsf{H}' \colon \{0,1\}^* \to \{0,1\}^{256}$ を用いる。また，XOF [*6] として $\mathsf{H}_G \colon \{0,1\}^* \to \{0,1\}^*$ も用いる。（第 4 ラウンドで G への入力に seed $\in \{0,1\}^{128}$ と salt $\in \{0,1\}^{128}$ が追加された。）

鍵生成: 基本方式の鍵生成と同様。ただし $\boldsymbol{h}$ の生成をシード seed から行うこととし，公開鍵を $pk = (\boldsymbol{s}, \text{seed})$ とする。また，秘密鍵にもシードを加え，$sk = (\boldsymbol{x}, \boldsymbol{y}, \text{seed})$ とする。

鍵カプセル化:

1. $\boldsymbol{m} \leftarrow \mathbb{F}_2^k$ を一様ランダムにとる。
2. salt $\leftarrow \mathbb{F}_2^{128}$ を一様ランダムにとる。
3. $\theta = \mathsf{H}_G(\boldsymbol{m}, \text{seed}, \text{salt})$ を計算する。θ から $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{e}$ を生成する。
4. $\boldsymbol{c} = \mathsf{Encrypt}(pk, \boldsymbol{m}, \boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{e})$ を計算する。$\boldsymbol{d} = \mathsf{H}'(\boldsymbol{m})$ とする。$K = \mathsf{H}(\boldsymbol{m}, \boldsymbol{c})$ とする。
5. 暗号文を $C = (\boldsymbol{c}, \boldsymbol{d}, \text{salt})$ とし，セッション鍵 K を出力する。

デカプセル化:

1. 受信した C に対して，$\boldsymbol{m}' = \mathsf{Decrypt}(sk, \boldsymbol{c})$ を計算する。
2. $\theta' = \mathsf{H}_G(\boldsymbol{m}', \text{seed}, \text{salt})$ を計算する。θ' から $\boldsymbol{r}_1', \boldsymbol{r}_2', \boldsymbol{e}'$ を生成する。
3. $\boldsymbol{c}' = \mathsf{Encrypt}(pk, \boldsymbol{m}', \boldsymbol{r}_1', \boldsymbol{r}_2', \boldsymbol{e}')$ を計算する。$\boldsymbol{c} \neq \boldsymbol{c}'$ もしくは $\boldsymbol{d} \neq \boldsymbol{d}'$ ならば $\perp$ を出力して停止する。
4. $K = \mathsf{H}(\boldsymbol{m}, \boldsymbol{c})$ を出力する。

以上より，セッション鍵（共通鍵）K を安全に共有することができる。

3 つのパラメータセットがそれぞれレベル 1，3，5 相当として提案された。表 4.5 に鍵カプセル化方式のパラメータ，鍵長，暗号文長および復号エラー率をまとめた。表中では，秘密鍵はシードだけ記憶していることにされており，40Bytes しかない。また公開鍵の $\boldsymbol{h}$ の部分もシードから再生成されることと定義されている点に注意されたい。hqc-128 の速度に関しては，鍵生成が $87,000$ Cycle，鍵カプセル化が $204,000$ Cycle，デカプセル化が $362,000$ Cycle である（Skylake CPU 搭載のデスクトップ PC，AVX 命令を使用）。他のパラメータに関しては，仕様書を参照されたい。

4.4 符号に基づく暗号技術に関するまとめ

基本となる McEliece 暗号方式 [45] は，McEliece により 40 年以上前に提案されており，パラメータは改訂されているものの，いまだに破られていない。Classic McEliece などのように，公開鍵や秘密鍵は長いものの，暗号文は短い方式が多い。LPN 問題は学習理論や符号理論から派生した問題であり，SD 問題は LPN 問題の特殊な場合である。誤り確率 τ が十分大きい場合の LPN 問題や，重み w が一定の大きさの SD 問題を確率的多項式時間で効率的に解くことは，量子コンピュータを用いても困難であると予想されている。

共通鍵暗号や公開鍵暗号の分野で多くの方式が LPN 問題や SD 問題に基づいて提案されている。LWE 問題と比較した場合，利点としては，

[*6] eXtendable-Output Function の略。SHAKE128 や SHAKE256 が例として知られている。

- $\mathbb{F}_2$ およびその拡大体を基に構成するため，ハードウェア構成との相性が良い点
- 誤差分布としてベルヌーイ分布やその一般化した分布を用いるため，誤差のサンプリングが容易である点

が挙げられる。一方，欠点として，

- 鍵や暗号文のサイズが大きくなりやすい点
- 符号の復号アルゴリズムが複雑になりがちな点
- 完全準同型暗号といった発展的な応用が少ない点

が挙げられる。暗号方式のパラメータ設定の際には，4.1 節で挙げたさまざまなアルゴリズムを考慮する必要がある。アルゴリズムの高速化について盛んに研究されており，動向を注視する必要がある。また，符号に基づく暗号技術の信頼性を向上させるためには，理論面における研究だけではなく，実時間の計算量に関する研究も重要である。公開鍵や秘密鍵を圧縮しようと特殊な符号を採用したり，距離の定義を変える提案も多くある。これらは解読攻撃を受けることも多く，評価が確定していない暗号・署名方式については注視が必要である。

本報告書は 2024 年 9 月 30 日時点の情報に基づいているが，2025 年 3 月に NIST から，HQC が標準化方式として選ばれたことが発表された [54]。

第 4 章の参照文献

[1] CRYPTREC 暗号技術調査 WG (耐量子計算機暗号). CRYPTREC 耐量子計算機暗号の研究動向調査報告書. CRYPTREC TR-2001-2022, https://www.cryptrec.go.jp/report/cryptrec-tr-2001-2022.pdf. 2023-03.

[2] M. R. Albrecht et al. Classic McEliece: conservative code-based cryptography. https://csrc.nist.gov/csrc/media/Projects/post-quantum-cryptography/documents/round-4/submissions/mceliece-Round4.tar.gz. 2022-10. (2024-03-05 閲覧).

[3] B. Applebaum, D. Cash, C. Peikert, A. Sahai. Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems. CRYPTO. Vol. 5677. Lecture Notes in Computer Science. Springer, 2009, pp. 595–618.

[4] N. Aragon et al. BIKE: Bit flipping key encapsulation (Round 4 submission). https://csrc.nist.gov/csrc/media/Projects/post-quantum-cryptography/documents/round-4/submissions/BIKE-Round4.zip. 2022-10. (2024-03-05 閲覧).

[5] N. Aragon et al. RYDE. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/submission-pkg/ryde-submission.zip. 2024-11. (2024-11-15 閲覧).

[6] S. Arora, R. Ge. New Algorithms for Learning in Presence of Errors. ICALP (1). Vol. 6755. Lecture Notes in Computer Science. Springer, 2011, pp. 403–415.

[7] M. Baldi et al. CROSS (Codes and Restricted Objects Signature Scheme). https://cross-crypto.com/. 2024-11. (2024-11-15 閲覧).

[8] A. Becker, A. Joux, A. May, A. Meurer. Decoding Random Binary Linear Codes in $2^{n/20}$: How $1 + 1 = 0$ Improves Information Set Decoding. EUROCRYPT. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 520–536.

[9] E. R. Berlekamp, R. J. McEliece, H. C. A. van Tilborg. On the inherent intractability of certain coding problems (Corresp.) IEEE Trans. Inf. Theory. Vol. 24, Num. 3 (1978), pp. 384–386.

[10] D. J. Bernstein, T. Lange. Never Trust a Bunny. RFIDSec. Vol. 7739. Lecture Notes in Computer Science. Springer, 2012, pp. 137–148.

[11] A. Blum, M. L. Furst, M. J. Kearns, R. J. Lipton. Cryptographic Primitives Based on Hard Learning Problems. CRYPTO. Vol. 773. Lecture Notes in Computer Science. Springer, 1993, pp. 278–291.

[12] A. Blum, A. Kalai, H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. J. ACM. Vol. 50, Num. 4 (2003), pp. 506–519.

[13] S. Bogos, S. Vaudenay. Optimization of LPN Solving Algorithms. ASIACRYPT (1). Vol. 10031. Lecture Notes in Computer Science. 2016, pp. 703–728.

[14] L. Both, A. May. Decoding Linear Codes with High Error Rate and Its Impact for LPN Security. PQCrypto. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 25–46.

[15] L. Both, A. May. Optimizing BJMM with nearest neighbors: Full decoding in $2^{2n/21}$ and McEliece security. Workshop on Coding and Cryptography. 2017. https://www.cits.ruhr-uni-bochum.de/imperia/md/content/may/paper/bjmm+.pdf.

[16] E. Carozza, G. Couteau, A. Joux. Short Signatures from Regular Syndrome Decoding in the Head. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 532–563.

[17] K. Carrier, T. Debris-Alazard, C. Meyer-Hilfiger, J.-P. Tillich. Reduction from Sparse LPN to LPN, Dual Attack 3.0. EUROCRYPT (6). Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 286–315.

[18] K. Carrier, T. Debris-Alazard, C. Meyer-Hilfiger, J.-P. Tillich. Statistical Decoding 2.0: Reducing Decoding to LPN. ASIACRYPT (4). Vol. 13794. Lecture Notes in Computer Science. Springer, 2022, pp. 477–507.

[19] J. Carrijo, R. Tonicelli, H. Imai, A. C. A. Nascimento. A Novel Probabilistic Passive Attack on the Protocols HB and HB^{+}. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. Vol. 92-A, Num. 2 (2009), pp. 658–662.

[20] C. Chevignard, P.-A. Fouque, A. Schrottenloher. Reducing the Number of Qubits in Quantum Information Set Decoding. ASIACRYPT (8). Vol. 15491. Lecture Notes in Computer Science. Springer, 2024, pp. 299–329.

[21] N. T. Courtois, M. Finiasz, N. Sendrier. How to Achieve a McEliece-Based Digital Signature Scheme. ASIACRYPT. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 157–174.

[22] S. Devadas, L. Ren, H. Xiao. On Iterative Collision Search for LPN and Subset Sum. TCC (2). Vol. 10678. Lecture Notes in Computer Science. Springer, 2017, pp. 729–746.

[23] I. Dinur, O. Dunkelman, N. Keller, A. Shamir. Efficient Dissection of Composite Problems, with Applications to Cryptanalysis, Knapsacks, and Combinatorial Search Problems. CRYPTO. Vol. 7417. Lecture Notes in Computer Science. Springer, 2012, pp. 719–740.

[24] L. Ducas, A. Esser, S. Etinski, E. Kirshanova. Asymptotics and Improvements of Sieving for Codes. EUROCRYPT (6). Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 151–180.

[25] A. Esser. Revisiting Nearest-Neighbor-Based Information Set Decoding. IMACC. Vol. 14421. Lecture Notes in Computer Science. Springer, 2023, pp. 34–54.

[26] A. Esser, F. Heuer, R. Kübler, A. May, C. Sohler. Dissection-BKW. CRYPTO (2). Vol. 10992. Lecture Notes in Computer Science. Springer, 2018, pp. 638–666.

[27] A. Esser, R. Kübler, A. May. LPN Decoded. CRYPTO (2). Vol. 10402. Lecture Notes in Computer Science. Springer, 2017, pp. 486–514.

[28] A. Esser, S. Ramos-Calderer, E. Bellini, J. I. Latorre, M. Manzano. Hybrid Decoding – Classical-Quantum Trade-Offs for Information Set Decoding. PQCrypto. Vol. 13512. Lecture Notes in Computer Science. Springer, 2022, pp. 3–23.

[29] A. Esser, J. A. Verbel, F. Zweydinger, E. Bellini. SoK: CryptographicEstimators – a Software Library for Cryptographic Hardness Estimation. AsiaCCS. ACM, 2024.

[30] A. Esser, F. Zweydinger. New Time-Memory Trade-Offs for Subset Sum – Improving ISD in Theory and Practice. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 360–390.

[31] J.-C. Faugère, V. Gauthier-Umaña, A. Otmani, L. Perret, J.-P. Tillich. A distinguisher for high rate McEliece cryptosystems. ITW. IEEE, 2011, pp. 282–286.

[32] J.-C. Faugère, V. Gauthier-Umaña, A. Otmani, L. Perret, J.-P. Tillich. A Distinguisher for High-Rate McEliece Cryptosystems. IEEE Trans. Inf. Theory. Vol. 59, Num. 10 (2013), pp. 6830–6844.

[33] Q. Guo, T. Johansson, C. Löndahl. Solving LPN Using Covering Codes. J. Cryptol. Vol. 33, Num. 1 (2020), pp. 1–33.

[34] J. Håstad. Some optimal inapproximability results. J. ACM. Vol. 48, Num. 4 (2001), pp. 798–859.

[35] S. Heyse, E. Kiltz, V. Lyubashevsky, C. Paar, K. Pietrzak. Lapin: An Efficient Authentication Protocol Based on Ring-LPN. FSE. Vol. 7549. Lecture Notes in Computer Science. Springer, 2012, pp. 346–365.

[36] G. Kachigar, J.-P. Tillich. Quantum Information Set Decoding Algorithms. PQCrypto. Vol. 10346. Lecture Notes in Computer Science. Springer, 2017, pp. 69–89.

[37] P. Kirchner. Improved Generalized Birthday Attack. Cryptology ePrint Archive, Paper 2011/377. 2011. https://eprint.iacr.org/2011/377.

[38] E. Kirshanova. Improved Quantum Information Set Decoding. PQCrypto. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 507–527.

[39] P. J. Lee, E. F. Brickell. An Observation on the Security of McEliece's Public-Key Cryptosystem. EUROCRYPT. Vol. 330. Lecture Notes in Computer Science. Springer, 1988, pp. 275–280.

[40] É. Levieil, P.-A. Fouque. An Improved LPN Algorithm. SCN. Vol. 4116. Lecture Notes in Computer Science. Springer, 2006, pp. 348–359.

[41] Y. Li, R. H. Deng, X. Wang. On the equivalence of McEliece's and Niederreiter's public-key cryptosystems. IEEE Trans. Inf. Theory. Vol. 40, Num. 1 (1994), pp. 271–273.

[42] V. Lyubashevsky, C. Peikert, O. Regev. On Ideal Lattices and Learning with Errors over Rings. EUROCRYPT. Vol. 6110. Lecture Notes in Computer Science. Springer, 2010, pp. 1–23.

[43] A. May, A. Meurer, E. Thomae. Decoding Random Linear Codes in $\tilde{\mathcal{O}}(2^{0.054n})$. ASIACRYPT. Vol. 7073. Lecture Notes in Computer Science. Springer, 2011, pp. 107–124.

[44] A. May, I. Ozerov. On Computing Nearest Neighbors with Applications to Decoding of Binary Linear Codes. EUROCRYPT (1). Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 203–228.

[45] R. J. McEliece. A Public-Key Cryptosystem Based On Algebraic Coding Theory. Deep Space Network Progress Report. Vol. 44 (1978), pp. 114–116.

[46] C. Aguilar Melchor et al. Hamming Quasi-Cyclic (HQC) – Fourth round version (Updated version 01/10/2022). https://csrc.nist.gov/csrc/media/Projects/post-quantum-cryptography/documents/round-4/submissions/HQC-Round4.zip. 2022-10. (2024-03-05 閲覧).

[47] S. Narisada, K. Fukushima, S. Kiyomoto. Multiparallel MMT: Faster ISD Algorithm Solving High-Dimensional Syndrome Decoding Problem. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. Vol. 106, Num. 3 (2023), pp. 241–252.

[48] S. Narisada, S. Uemura, H. Okada, H. Furue, Y. Aikawa, K. Fukushima. Solving McEliece-1409 in One Day – Cryptanalysis with the Improved BJMM Algorithm. ISC (2). Vol. 15258. Lecture Notes in Computer Science. Springer, 2024, pp. 3–23.

[49] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. Problemy Upravleniia i Teorii Informatsii (Problems of Control and Information Theory). Vol. 15, Num. 2 (1986), pp. 157–166.

[50] NIST. Module-Lattice-Based Key-Encapsulation Mechanism Standard. NIST FIPS 203, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.pdf. 2024-08.

[51] S. Perriello, A. Barenghi, G. Pelosi. A Complete Quantum Circuit to Solve the Information Set Decoding Problem. QCE. IEEE, 2021, pp. 366–377.

[52] S. Perriello, A. Barenghi, G. Pelosi. Improving the Efficiency of Quantum Circuits for Information Set Decoding. ACM Transactions on Quantum Computing. Vol. 4, Num. 4 (2023). https://doi.org/10.1145/3607256.

[53] E. Prange. The use of information sets in decoding cyclic codes. IRE Trans. Inf. Theory. Vol. 8, Num. 5 (1962), pp. 5–9.

[54] Selected Algorithms. 2025-03. https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms. (2025-03-30 閲覧).

[55] J. Stern. A method for finding codewords of small weight. Coding Theory and Applications. Vol. 388. Lecture Notes in Computer Science. Springer, 1988, pp. 106–113.

[56] B. Zhang, L. Jiao, M. Wang. Faster Algorithms for Solving LPN. EUROCRYPT (1). Vol. 9665. Lecture Notes in Computer Science. Springer, 2016, pp. 168–195.

[57] 電子情報通信学会. 知識ベース 知識の森 1 群 (信号・システム) 2 編 (符号理論). https://www.ieice-hbkb.org/portal/01-2/01_02/. (2024-03-05 閲覧).

第 5 章

多変数多項式に基づく暗号技術

多変数公開鍵暗号（Multivariate Public Key Cryptosystems: MPKC）における暗号方式の特徴は，有限体上の多変数多項式を用いた連立方程式

$$\begin{cases} p_1(x_1, x_2, \ldots, x_n) = 0, \\ p_2(x_1, x_2, \ldots, x_n) = 0, \\ \qquad\qquad \vdots \\ p_m(x_1, x_2, \ldots, x_n) = 0 \end{cases}$$

の求解問題（MP 問題）を解く計算の困難性が安全性の根拠として必要ということである。連立線形方程式は多項式時間で求解可能であるから，多変数公開鍵暗号に現れる MP 問題における多項式の最大次数は 2 次以上に限定される。本報告書では，多変数公開鍵暗号の多くの方式で採用されている双極型システムを中心に解説する。

5.1 多変数多項式に基づく暗号技術の安全性の根拠となる問題

$\mathbb{F}_q$ で位数 q の有限体を表し，$\mathbf{x} = (x_1, x_2, \ldots, x_n)$ で（代数的に独立な）変数の集合を表すものとする。$\mathbf{x}$ に関する $\mathbb{F}_q$ 上の多変数多項式の組，すなわち，多変数多項式 $p_i(\mathbf{x})$ $(i = 1, \ldots, m)$ により，$P(\mathbf{x}) = (p_1(\mathbf{x}), p_2(\mathbf{x}), \ldots, p_m(\mathbf{x}))$ と表されるものを（$\mathbb{F}_q$ 上の）多変数多項式系と呼ぶことにする。この多変数多項式系 $P(\mathbf{x})$ は代入評価により，$\mathbb{F}_q^n$ から $\mathbb{F}_q^m$ への写像を構成する。この（多変数多項式）写像を $P : \mathbb{F}_q^n \to \mathbb{F}_q^m$ と表すことにする。

5.1.1 MP 問題（MQ 問題）

MP 問題は次のように定義される。

MP 問題 多変数多項式系 $P(\mathbf{x}) = (p_1(\mathbf{x}), p_2(\mathbf{x}), \ldots, p_m(\mathbf{x}))$ と $\mathbf{d} = (d_1, d_2, \ldots, d_m) \in \mathbb{F}_q^m$ に対して，変数 $\mathbf{x}$ に関する連立方程式

$$\begin{cases} p_1(x_1, x_2, \ldots, x_n) = d_1, \\ p_2(x_1, x_2, \ldots, x_n) = d_2, \\ \qquad\qquad \vdots \\ p_m(x_1, x_2, \ldots, x_n) = d_m \end{cases} \tag{5.1}$$

の解（が存在するなら）少なくとも 1 つ求めよ。

連立方程式 (5.1) の右辺の各 d_i を左辺に移項して $p_i(\mathbf{x})$ に吸収させることができるので，右辺を 0 として MP 問題を表現する場合もある。MP 問題において，$P(\mathbf{x})$ の全ての成分 $p_i(\mathbf{x})$ が 1 次以下となる場合，MP 問題は単に線形方程式を解く問題となり，ガウスの消去法などで m, n に関し多項式時間で求解することが可能である。よって，MP 問題

を考える場合は通常，各 $p_i(\mathbf{x})$ の次数は 2 以上であると仮定する。特に，$p_i(\mathbf{x})$ の次数が全て 2 となるとき，MP 問題は MQ 問題と呼ばれる。$\mathbb{F}_q = \mathbb{F}_2$ の場合，MQ 問題は NP 完全であることが知られている [26, p. 251, AN9]。

MQ 問題を解くコンテストとして Fukuoka MQ challenge [23] が知られている。扱われている MQ 問題は，有限体は $q = 2, 31, 256$ の 3 種類と m, n に関しては $m = 2n$，$n \approx 1.5m$ の 2 種類の計 6 種類である。投稿され解かれた問題の (m, n) の値の最大は表 5.1 のようになっている。

表 5.1: Fukuoka MQ challenge で解かれた MQ 問題のパラメータの最大値（2024/9/30 時点）

タイプ	I	II	III	IV	V	VI
$\mathbb{F}_q$	$\mathbb{F}_2$	$\mathbb{F}_{31}$	$\mathbb{F}_{256}$	$\mathbb{F}_2$	$\mathbb{F}_{31}$	$\mathbb{F}_{256}$
(m, n)	$m = 2n$	$m = 2n$	$m = 2n$	$n \approx 1.5m$	$n \approx 1.5m$	$n \approx 1.5m$
(m, n) の最大	$(166, 83)$	$(74, 37)$	$(76, 38)$	$(76, 114)$	$(20, 30)$	$(22, 33)$

5.1.2 MP 問題を解く計算の計算量

MP 問題 (5.1) は，右辺の $\mathbf{d}$ を左辺に移行し，$P(\mathbf{x})$ の中に吸収させてしまうことにより，

$$P(\mathbf{x}) = (p_1(\mathbf{x}), \ldots, p_m(\mathbf{x})) = \mathbf{0}_m \tag{5.2}$$

の形の求解問題に変形できる。MP 問題 (5.2) に対する一般的な解き方として，総当たり法や XL [38], Gröbner 基底攻撃 [16] が知られている。

XL は，$(\prod_{j=1}^{k} x_{i_j})\, p_j(\mathbf{x}) = 0$ $(j = 1, \ldots, m)$ の形の方程式をたくさん集め，連立線型方程式の簡約操作を用いて MP 問題の解を求める。MP 問題の解の x_n の値を求める基本的な手順は以下のようになる。

1. $(\prod_{j=1}^{k} x_{i_j})\, p_j(\mathbf{x}) = 0$ $(j = 1, \ldots, m)$ の形の方程式をたくさん集める。
2. これらの方程式内に現れる単項式を新たな変数で置き直し，連立線型方程式を立てる。
3. 立てた連立線型方程式を簡約化することで，x_n^ℓ $(\ell = 0, 1, 2, \ldots)$ 以外の変数を消去する。
4. 得られた 1 変数 x_n に関する多項方程式を解いて x_n の値を求める。

この手順で得られた x_n の値を (5.2) に代入すると，$x_1, \ldots, x_{n-1}$ に関する MP 問題が得られる。この MP 問題に対して，上の手順と同様のことを行うと，今度は x_{n-1} の値を得ることができる。これを繰り返せば，最終的に MP 問題の解のすべての成分が得られる。十分大きい正の整数 D を取り，D 次以下の $(\prod_{\ell=1}^{k} x_{i_\ell})\, p_j(\mathbf{x}) = 0$ $(j = 1, \ldots, m)$ の形の方程式をすべて集めると，連立線型方程式が退化し，必ず解を持つようにすることができる。

Gröbner 基底攻撃は，イデアルの Gröbner 基底 [1] を計算して，MP 問題の解を求める。MP 問題 (5.2) の $p_1(\mathbf{x}), \ldots, p_m(\mathbf{x})$ の列を延長するように $p_{m+1}(\mathbf{x}) = x_1^q - x_1, \ldots, p_{m+n}(\mathbf{x}) = x_n^q - x_n$ とおき，イデアル $I \subset \mathbb{F}_q[\mathbf{x}]$ を $I = \langle p_1(\mathbf{x}), \ldots, p_{m+n}(\mathbf{x}) \rangle$ とおく。イデアル I の（ある項順序に関する）Gröbner 基底が計算できたとして，それを $g_1(\mathbf{x}), \ldots, g_\ell(\mathbf{x}) \in \mathbb{F}_q[\mathbf{x}]$ とすると，MP 問題 (5.2) の解集合と，方程式 $(g_1(\mathbf{x}), \ldots, g_\ell(\mathbf{x})) = \mathbf{0}_\ell$ の解集合は一致する。項順序を辞書式順序にした場合，I の Gröbner 基底は，

$$g_1(x_1, \ldots, x_n), \ldots, g_{i_2-1}(x_1, \ldots, x_n), g_{i_2}(x_2, \ldots, x_n), \ldots, g_{i_3}(x_2, \ldots, x_n), \ldots, g_{\ell-1}(x_{n-1}, x_n), g_\ell(x_n)$$

という風に，i が大きくなるにつれ，g_i の変数の個数が（広義単調に）減るという形にできる。すると，$g_\ell(x_n) = 0$ を解いて x_n の値を求めることができ，さらに，$g_{\ell-1}(x_{n-1}, x_n) = 0$ などに求めた x_n の値を代入することで，x_{n-1} に関する 1 変数の多項方程式が得られ，x_{n-1} の値を求めることができる。これを繰り返すことで，すべての x_i の値が

特定でき，MP 問題 (5.2) の解を求められる。これが Gröbner 基底攻撃の基本戦略である。Gröbner 基底の効率的計算方法としては，F4/F5 アルゴリズム [19, 20] が有名である。

Gröbner 基底攻撃と XL の攻撃計算量は，アルゴリズム内に現れる（最も計算が重い）連立線型方程式の簡約操作の計算量で見積もられる。よって，攻撃計算量を求めるには，アルゴリズム内に現れる行列のサイズを知る必要があるが，それには攻撃アルゴリズム中に現れる多項式の次数の上限を見積もる必要がある。この上限の見積もり方について説明する。各 $p_i(\mathbf{x})$ $(i = 1, \ldots, m+n)$ に対し，その最高次斉次部分を $p_i^h(\mathbf{x})$ （d_i 次斉次多項式）と表し，$\mathbb{F}_q[\mathbf{x}]$ の斉次イデアル J を

$$J = \langle p_1^h(\mathbf{x}), \ldots, p_{m+n}^h(\mathbf{x}) \rangle$$

で定める。$d \geq 0$ に対し， $\mathbb{F}_q[\mathbf{x}]_d$ で d-次斉次多項式のなす $\mathbb{F}_q[\mathbf{x}]$ の部分ベクトル空間を表し，$J_d := J \cap \mathbb{F}_q[\mathbf{x}]_d$ とする。次数環 $\mathbb{F}_q[\mathbf{x}]/J = \bigoplus_{d=0}^{\infty} \mathbb{F}_q[\mathbf{x}]_d/J_d$ の Hilbert 級数は

$$\mathrm{HS}_{\mathbb{F}_q[\mathbf{x}]/J}(t) = \sum_{d=0}^{\infty} \dim_{\mathbb{F}_q}(\mathbb{F}_q[\mathbf{x}]_d/J_d)\, t^d \in \mathbb{Z}[[t]] \quad \text{(形式的べき級数)}$$

で定義される。J の Krull-次元が 0，すなわち，J が $\mathbb{F}_q[\mathbf{x}]$ の極大イデアルとなるとき，$\mathrm{HS}_{\mathbb{F}_q[\mathbf{x}]/J}(t)$ は多項式となる。このとき，$d_{\mathrm{reg}} = \deg(\mathrm{HS}_{\mathbb{F}_q[\mathbf{x}]/J}(t)) + 1$ とおき，これを正則性の次数 (degree of regularity) と呼ぶ。これ以外にも，Gröbner 基底計算と関係のある不変量として，solving degree d_{sol} や first fall degree d_{ff} などが存在する [13, 18]。これらの不変量 d_{reg}，d_{sol}，d_{ff} はいずれも Gröbner 基底計算中に現れる多項式の次数の上限を評価する値である。一般に，これらの不変量を求めることは Gröbner 基底計算と同程度困難であろうと考えられている。d をこれら不変量のうちの 1 つとしたとき，Gröbner 基底攻撃の計算量は以下のようになる [5]：

$$\mathcal{O}\left(\left(\begin{array}{c} n+d \\ d \end{array} \right)^{\omega} \right). \tag{5.3}$$

ここで，$2 \leq \omega \leq 3$ は行列乗算指数。

任意の $S(t) \in \mathbb{Z}[[t]]$ に対し，$[S(t)]_+ \in \mathbb{Z}_{>0}[[t]]$ で，$S(t)$ の最初に現れる非正係数の次数以降（この項も含む）を切り捨てた多項式を表すことにする。もし，

$$\mathrm{HS}_{\mathbb{F}_q[\mathbf{x}]/J}(t) = \left[\frac{\prod_{i=1}^{m+n}(1-t^{d_i})}{(1-t)^n} \right]_+ = \left[\left(\prod_{i=1}^{n}(1-t^{d_i}) \right) \left(\frac{1-t^q}{1-t} \right)^n \right]_+$$

を満たすならば，$p_1(\mathbf{x}), \ldots, p_{m+n}(\mathbf{x})$ は半正則であるという。任意の m, n に対して, $p_1(\mathbf{x}), \ldots, p_m(\mathbf{x})$ の係数をランダムに選ぶと，多くの場合に $p_1(\mathbf{x}), \ldots, p_{m+n}(\mathbf{x})$ は半正則となることが実験的に知られている。半正則であれば，正則性の次数 d_{reg} は容易に計算可能である。

XL で解く連立線形方程式の行列部分は疎行列である。実際，$p_i(\mathbf{x})$ $(i = 1, \ldots, m)$ が含む単項式の個数の最大を L とすると，行列の各行の非零成分の個数も L 個以下となる。従って，

$$d_{\mathrm{XL}} = \deg\left(\left[\frac{\prod_{i=1}^{m}(1-t^{d_i})}{(1-t)^{n+1}} \right]_+ \right) + 1$$

とおくと，q がある程度大きい場合，XL の計算量は以下のようになる：

$$\mathcal{O}\left(\left(\begin{array}{c} n+d_{\mathrm{XL}} \\ d_{\mathrm{XL}} \end{array} \right)^2 L \right)$$

5.1.3 MinRank 問題

MinRank 問題 正の整数 r と行列 $\mathrm{M}_1, \ldots, \mathrm{M}_k \in \mathbb{F}_q^{m \times n}$ に対し，$\alpha_1, \ldots, \alpha_k \in \mathbb{F}_q$ で，$(\alpha_1, \ldots, \alpha_k) \neq (0, \ldots, 0)$ かつ

$$\mathrm{Rank}\left(\sum_{i=1}^{k} \alpha_i \mathrm{M}_i\right) \leq r$$

なるものを求めよ。(Rank(M) は行列 M のランクを表す。)

MinRank 問題は HFE_v^- や Rainbow など様々な方式の安全性に関わっている。また，MinRank 問題を解く計算の困難性をベースとした署名方式などがいくつか提案されている [15, 6, 35, 2]。MinRank 問題は MP 問題に帰着できることが知られている [30, 21, 4]。

例えば，Support minor modeling [4] では以下のように MinRank 問題が MP 問題に帰着される。$\alpha_1, \ldots, \alpha_k$ が MinRank 問題の解であるとするならば，$(\mathrm{S}, \mathrm{C}) \in \mathbb{F}_q^{m \times r} \times \mathbb{F}_q^{r \times n}$ で

$$\mathrm{S}\,\mathrm{C} = \sum_{i=1}^{k} \alpha_i \mathrm{M}_i \tag{5.4}$$

なるものが存在する。$\mathbf{r}_j$ を $\sum_{i=1}^{k} \alpha_i \mathrm{M}_i$ の第 j 行とすると，(5.4) より $\mathbf{r}_j$ は C の行ベクトルが張る空間に属する。よって，行列 $\mathrm{C}'_j \in \mathbb{F}_q^{(r+1) \times n}$ を

$$\mathrm{C}'_j = \begin{pmatrix} \mathbf{r}_j \\ \mathrm{C} \end{pmatrix}$$

で定めると，各 $j = 1, \ldots, m$ に対して，$\mathrm{Rank}\,\mathrm{C}'_j \leq r$ を満たす。従って，C'_j の任意の $(r+1) \times (r+1)$ 小行列の行列式 $= 0$ という関係式が得られるが，このような関係式は j と小行列を動かすことにより $m \begin{pmatrix} n \\ r+1 \end{pmatrix}$ 個存在する。r 個の元からなる $T\,(\subset \{1, 2, \ldots, n\})$ に対して，T に属する列番号からなる C の $r \times r$ 小行列を C_T と表し，さらにその行列式を c_T と表すと，C'_j の任意の $(r+1) \times (r+1)$ 小行列の行列式は，$\alpha_1, \ldots, \alpha_k$ と c_T ($T \subset \{1, \ldots, m\}$, T の元の個数は r) に関する多項式で表すことができる。これらの変数の個数は $k + \begin{pmatrix} n \\ r \end{pmatrix}$ である。つまり，MinRank 問題は $k + \begin{pmatrix} n \\ r \end{pmatrix}$ 個の変数の $m \begin{pmatrix} n \\ r+1 \end{pmatrix}$ 個の方程式からなる MP 問題に帰着される。

5.1.4 IP 問題，EIP 問題

Isomorphism of Polynomials（IP）問題は以下のように定義される。

IP 問題 S, T をそれぞれ，$\mathbb{F}_q^n$, $\mathbb{F}_q^m$ 上のアフィン同型写像とする。多変数多項式系 $P(\mathbf{x}) = (p_1(\mathbf{x}), p_2(\mathbf{x}), \ldots, p_m(\mathbf{x}))$ に対し，多変数多項式系 $\tilde{P}(\mathbf{x})$ を合成により，$\tilde{P}(\mathbf{x}) = T \circ P(\mathbf{x}) \circ S$ で定める。このとき，$P(\mathbf{x})$, $\tilde{P}(\mathbf{x})$ の情報から S, T を求めよ。

IP 問題において，S や T の行列成分やベクトル成分をすべて独立な変数と見た場合，等式 $\tilde{P}(\mathbf{x}) = T \circ P(\mathbf{x}) \circ S$ は連立多項式方程式と見ることができる。すなわち，IP 問題は MP 問題に変換される。

多変数多項式系のクラス $\mathcal{C}$ を 1 つ固定する。ここで多変数多項式系のクラスとは多変数多項式系の集合 $\mathbb{F}_q[\mathbf{x}]^m$ の部分集合のことである。このとき，(クラス $\mathcal{C}$ に関する) Extended Isomorphism of Polynomials（EIP）問題は以下のように定義される。

EIP 問題 多変数多項式系 $\tilde{P}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ は，$\mathbb{F}_q^n, \mathbb{F}_q^m$ 上のアフィン同型写像 S, T とクラス $\mathcal{C}$ に属する多変数多項式系 $P(\mathbf{x})$ により，$\tilde{P}(\mathbf{x}) = T \circ P(\mathbf{x}) \circ S$ で表されるとする。このとき，分解 $\tilde{P}(\mathbf{x}) = T' \circ P'(\mathbf{x}) \circ S'$ で，S', T' は $\mathbb{F}_q^n, \mathbb{F}_q^m$ 上のアフィン同型写像，$P'(\mathbf{x}) \in \mathcal{C}$ なるものを見つけよ。

$\mathcal{C} = \{P(\mathbf{x})\}$ に関する EIP 問題が通常の IP 問題であるから，EIP 問題は IP 問題の拡張である。5.2 節で述べるように，EIP 問題は双極型システムで構成される公開鍵暗号方式，署名方式の鍵復元攻撃に対する安全性に関わる。EIP 問題を解く方法はクラス $\mathcal{C}$ の取り方（あるいは方式）に依存する。

5.2 多変数多項式に基づく代表的な暗号方式

5.2.1 双極型システム

IP 問題ベース [33] や MinRank 問題ベース [15, 2, 6, 35] の方式も存在するが，多変数公開鍵暗号の多くの方式が MP 問題をベースとして構成されている。中でも双極型システム [16] と呼ばれる構成方法が多く利用されているため，この構成方法について説明する。（1 次多項式で構成されてなくても）多変数多項式系 $P(\mathbf{x})$ によっては，多くの $\mathbf{d} \in \mathbb{F}_q^m$ に対して MP 問題が効率的に計算できる場合がある。例えば, $n = m$ とし, $P(\mathbf{x}) = (p_1(\mathbf{x}), p_2(\mathbf{x}), \ldots, p_m(\mathbf{x}))$ が三角型多変数多項式系である，すなわち，

$$
\begin{aligned}
&p_1(\mathbf{x}) = x_1, \\
&p_2(\mathbf{x}) = x_2 + g_2(x_1) \quad (g_2(x_1) \in \mathbb{F}_q[x_1]), \\
&p_3(\mathbf{x}) = x_3 + g_3(x_1, x_2) \quad (g_2(x_1, x_2) \in \mathbb{F}_q[x_1, x_2]), \\
&\qquad \vdots \\
&p_m(\mathbf{x}) = x_m + g_m(x_1, \ldots, x_{m-1}) \quad (g_m(x_1, \ldots, x_{m-1}) \in \mathbb{F}_q[x_1, \ldots, x_{m-1}])
\end{aligned}
$$

の形で表されるとすると，任意の $\mathbf{d} \in \mathbb{F}_q^m$ に対して $P(\mathbf{x}) = \mathbf{d}$ の（唯 1 つの）解が，x_1 から逐次的に求められることが分かる。このことはすなわち，多変数多項式系のクラス $\mathcal{C}$ を三角型多変数多項式系の全体で定めると，任意の $P \in \mathcal{C}$ に対して，$P(\mathbf{x}) = \mathbf{d}$ $(\mathbf{d} \in \mathbb{F}_q^m)$ の解が効率的に計算可能ということである。

双極型システムでは，まず，MP 問題が効率的に計算できる多変数多項式系のクラス $\mathcal{C}_{\text{cent}}$ を見つけ固定する。（例えば，$\mathcal{C}_{\text{cent}}$ として三角型多変数多項式系の集合を取れる。）$G(\mathbf{x}) \in \mathcal{C}_{\text{cent}}$ と $\mathbb{F}_q^n, \mathbb{F}_q^m$ 上のアフィン同型写像 S, T をそれぞれ任意にとり，これらを合成した多変数多項式系 $F(\mathbf{x}) = T \circ G(\mathbf{x}) \circ S$ をトラップドア付き一方向関数として利用するのが，双極型システムのアイデアである。ただし，$F(\mathbf{x})$ が実際にトラップドア付き一方向関数となるかどうかは $\mathcal{C}_{\text{cent}}$ のとり方に依存する。

双極型システムの鍵生成は次のように行う。

鍵生成

1. $G(\mathbf{x}) \in \mathcal{C}_{\text{cent}}$ をランダムに選ぶ。
2. $\mathbb{F}_q^n, \mathbb{F}_q^m$ 上のアフィン同型写像 S, T をランダムに選ぶ。
3. $F(\mathbf{x}) = T \circ G(\mathbf{x}) \circ S$ とする。

このとき，公開鍵は $F(\mathbf{x})$，秘密鍵は $G(\mathbf{x}), S, T$ となる。$F(\mathbf{x})$ はその係数集合が公開鍵として保管される。また，$G(\mathbf{x})$ を（この方式の）**中心写像**とよぶ。中心写像のクラス $\mathcal{C}_{\text{cent}}$ は 2 次の多変数多項式系で構成されることが多い。これは，公開鍵長（や秘密鍵長）を出来るだけ小さくするためである。双極型システムは公開鍵暗号方式，署名方式両方の構成に用いることができる。

公開鍵暗号方式の暗号化・復号は次のように行う。

暗号化 平文 $M \in \mathbb{F}_q^n$ に対し，$C = F(M)$ を計算する。C が暗号文となる。

復号 暗号文 $C \in \mathbb{F}_q^m$ に対し，(1) $B_1 = T^{-1}(C)$, (2) $G(B_2) = B_1$ なる B_2 を計算, (3) $M' = S^{-1}(B_2)$ の順に計算する。M' が平文と一致する。

復号が成功するためには，$G(\mathbf{x})$（あるいは $F(\mathbf{x})$）が単射である必要がある。単射の条件を少し緩めて，「$G(\mathbf{x})$（あるいは $F(\mathbf{x})$）の逆像の個数が十分少ない」とすることもできる。この場合，M' が複数得られることになるので，ハッシュ値などを用いて平文 M と一致する M' を特定する。

双極型システムの署名方式の署名生成・検証は次のように行う。

署名生成 メッセージ（のハッシュ値） $M \in \mathbb{F}_q^m$ に対し，(1) $B_1 = T^{-1}(M)$, (2) $G(B_2) = B_1$ なる B_2 を計算, (3) $\sigma = S^{-1}(B_2)$ の順に計算する。σ が署名となる。

検証 署名 $\sigma \in \mathbb{F}_q^n$ に対し，$M' = F(\sigma)$ を計算する。$M = M'$ ならば署名を受理，それ以外は棄却する。

署名生成がいつでも実行できるためには，どのような $M \in \mathbb{F}_q^m$ に対しても，$B_2 = G^{-1}(B_1)$ の計算ができる，すなわち，$G(\mathbf{x})$（あるいは $F(\mathbf{x})$）が全射である必要がある。

双極型システムでは，中心写像のクラス $\mathcal{C}_{\mathrm{cent}}$ の取り方を変えることで幅広い方式の構成が可能である。例えば，$\mathcal{C}_{\mathrm{cent}} = \{$ 三角型多変数多項式系 $\}$ とすると公開鍵暗号方式が得られる。双極型システムにおいては，$\mathcal{C}_{\mathrm{cent}}$ に関する EIP 問題がその安全性に大きく関わってくる。実際，EIP 問題を解けた場合，$F(\mathbf{x}) = T \circ G(\mathbf{x}) \circ S$ の代わりに分解 $F(\mathbf{x}) = T' \circ G'(\mathbf{x}) \circ S'$ を用いても，公開鍵暗号方式における復号および，署名方式における署名生成（偽造）が実行可能となる。EIP 問題の困難性はクラス $\mathcal{C}$ の選び方に依存するので，$\mathcal{C}$ の選び方に応じて個々に解析される必要がある。例えば，$\mathcal{C}_{\mathrm{cent}} = \{$ 三角型多変数多項式系 $\}$ としたときの EIP 問題は効率的に解けることが知られている [27]。

双極型システムの代表的な構成法として，simple field 方式と big field 方式がある。simple field 方式は中心写像の構成に $\mathbb{F}_q$ 以外の有限体を利用しない。big field 方式は中心写像の構成に $\mathbb{F}_q$ の n 次拡大体 $\mathbb{F}_{q^n}$ を利用する。big field 方式は中心写像を構成しやすいが，Gröbner 基底攻撃が効果的となる場合が多いという性質を持つ。5.2.3 節では，big field 方式の代表として署名方式 HFE および HFE$_v^-$，5.2.4 節では，simple field 方式の代表として署名方式 UOV について説明する。

5.2.2 双極型システムの modifier

modifier [37, 16] は双極型システムの方式からその変種方式を構成する。様々な modifier があり，それぞれに安全性を強化したり，効率性を向上させたりといった効果がある。以下では代表的な modifier を 4 つ紹介する。5.2.3 節で説明する HFEv^- 方式では，2 つの modifier（マイナス手法と External Perturbation）が利用されている。

5.2.2.1 マイナス手法 “ − ”

マイナス手法は，公開鍵 $F(\mathbf{x})$ のいくつかの成分を削除する方法である。すなわち，$F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ と表されるとき，r 個の成分を削除し，$\tilde{F}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_{m-r}(\mathbf{x}))$ を新たな公開鍵とする方式である。$F(\mathbf{x})$ に対して有効な秘密鍵復元攻撃がある場合でも，$\tilde{F}(\mathbf{x})$ は $F(\mathbf{x})$ よりも情報が欠落しているため，$\tilde{F}(\mathbf{x})$ に対しては同じ秘密鍵復元攻撃が適用できなくなる可能性があり，安全性強化につながる。公開鍵暗号方式の構成では公開鍵の単射性が失われたり，復号が困難になるといった理由により，マイナス手法はあまり用いられない。署名方式では，$F(\mathbf{x})$ に対する署名生成を利用して，$\tilde{F}(\mathbf{x})$ に対する署名生成ができる。

5.2.2.2 プラス手法“+”

プラス手法は，中心写像 $G(\mathbf{x})$ にランダムな多項式を成分として加える方法である。すなわち，中心写像 $G(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x}))$ に対し，r 個のランダムな多項式 $g_{m+1}(\mathbf{x}), \ldots, g_{m+r}(\mathbf{x})$ を用意し，$\tilde{G}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_{m+r}(\mathbf{x}))$ を新たな中心写像とする方式である。中心写像に特定の構造を持たない多項式が混ざることにより，秘密鍵復元攻撃の成功率を下げることができる。署名方式では公開鍵の全射性が失われたり，署名生成が困難になるといった理由により，プラス手法はあまり用いられない。公開鍵暗号方式では，復号において $\tilde{G}(\mathbf{x})$ の逆写像を計算しなければならないが，その計算に $G(\mathbf{x})$ の逆写像計算が利用できる。

5.2.2.3 External Perturbation“v”

この modifier は，元々の変数 $\mathbf{x} = (x_1, \ldots, x_n)$ に新たな変数 $\mathbf{v} = (x_{n+1}, \ldots, x_{n+v})$（vinegar 変数）を加える方法である。この modifier は主に署名方式で利用される。署名方式を定める中心写像のクラスを $\mathcal{C}$ とする。新たな中心写像のクラス $\mathcal{C}'$ を多項式写像 $G(\mathbf{x}, \mathbf{v}) : \mathbb{F}_q^{n+v} \to \mathbb{F}_q^m$ で，任意の $\mathbf{v}_0 \in \mathbb{F}_q^v$ に対し，$G(\mathbf{x}, \mathbf{v}_0) \in \mathcal{C}$ なるもの全体として定める。すると，$G(\mathbf{x}, \mathbf{v}) \in \mathcal{C}'$ に対し，$G(\mathbf{x}, \mathbf{v}) = \mathbf{d}$ $(\mathbf{d} \in \mathbb{F}_q^m)$ の解が次のように得られる。

1. $\mathbf{v}_0 \in \mathbb{F}_q^v$ をランダムに選ぶ。
2. $G(\mathbf{x}, \mathbf{v}_0) = \mathbf{d}$ を $\mathbf{x}$ に関して解く。（解を $\mathbf{x}^*$ とする。）
3. $(\mathbf{x}, \mathbf{v}) = (\mathbf{x}^*, \mathbf{v}_0)$ を出力。

この計算を利用して，新たな署名方式が構成できる。vinegar 変数は $\mathcal{C}$ とは無関係な変数なので追加することで安全性強化が期待できる。

5.2.2.4 Internal Perturbation“I”

この modifier は，中心写像 $G(\mathbf{x})$ にノイズを加えて安全性を強化する方法である。変数 $\mathbf{z} = (z_1, \ldots, z_w)$ と多項式写像 $H(\mathbf{z}) : \mathbb{F}_q^w \to \mathbb{F}_q^m$，および，アフィン写像 $S : \mathbb{F}_q^n \to \mathbb{F}_q^w$ を用意する。また，$H(\mathbf{z})$ の像 $W \subset \mathbb{F}_q^m$ が分かっており，W に属する元の個数は十分少ないと仮定する。新たな中心写像 $\tilde{G}(\mathbf{x}) : \mathbb{F}_q^n \to \mathbb{F}_q^m$ を $G(\mathbf{x}) + H(S(\mathbf{x}))$ で定める。このとき，$\tilde{G}(\mathbf{x}) = \mathbf{d}$ $(\mathbf{d} \in \mathbb{F}_q^m)$ の解が次のように得られる。

1. $\mathbf{w}_0 \in W$ をランダムに選ぶ。
2. 次の方程式の解 $\mathbf{x}^*$ を求める。

$$\begin{cases} G(\mathbf{x}) = \mathbf{d} - \mathbf{w}_0, \\ H(S(\mathbf{x})) = \mathbf{w}_0 \end{cases}$$

   もし，解が得られなかったら 1. に戻る。
3. $\mathbf{x}^*$ を出力。

よって，この $\tilde{G}(\mathbf{x})$ を中心写像として方式が構成できる。

5.2.3 公開鍵暗号方式 HFE，署名方式 HFE v^-

5.2.3.1 公開鍵暗号方式 HFE

$K = \mathbb{F}_{q^n}$ を $\mathbb{F}_q$ の n 次拡大体とし，$\mathbb{F}_q$-線形同型写像 $\phi : \mathbb{F}_q^n \xrightarrow{\sim} K$ を 1 つ固定する。D を正の整数として，K 上の 1 変数多項式

$$\mathcal{G}(X) = \sum_{0 \le i \le j}^{q^i+q^j \le D} \alpha_{i,j} X^{q^i+q^j} + \sum_{0 \le i}^{q^i \le D} \beta_i X^{q^i} + \gamma \qquad (\alpha_{i,j},\ \beta_i,\ \gamma \in K)$$

をとる。$\mathcal{G}(X)$ の形の 1 変数多項式は HFE 多項式と呼ばれる。このとき，多変数多項式写像 $G : \mathbb{F}_q^n \to \mathbb{F}_q^n$ を $G = \phi^{-1} \circ \mathcal{G} \circ \phi$ と定めると，対応する多変数多項式系 $G(\mathbf{x})$ の成分は全て 2 次多項式となる。$\mathbf{d} \in \mathbb{F}_q^n$ に対して，$G(\mathbf{x}) = \mathbf{d}$ が解を持つならば，この解は全て効率的に計算することができる。実際，次の手順で計算できる。

1. $B = \phi(\mathbf{d}) \in K$ を計算する。
2. $A = \mathcal{G}^{-1}(B)$ を Cantor-Zassenhaus アルゴリズムなどの因数分解アルゴリズムを用いて計算する。
3. $\phi^{-1}(A)$ を計算する。

但し，Step 2 の計算が効率的に実行できるためには D をある程度小さくとる必要がある。以上のことを踏まえ，$\alpha_{i,j},\ \beta_i,\ \gamma \in K$ を動かしてできる $G(\mathbf{x})$ のなすクラス $\mathcal{C}_{\mathrm{HFE}}$ に対し，$\mathcal{C}_{\mathrm{cent}} = \mathcal{C}_{\mathrm{HFE}}$ とした双極型システムにより公開鍵暗号方式が構成できる。この公開鍵暗号方式を HFE [33] と呼ぶ。HFE 自体は 1999 年，Kipnis と Shamir により効果的な攻撃が発見されている [30]。その後，HFE から派生した変種方式がいくつか提案されており，以下の HFE_v^- もその 1 つである。

5.2.3.2 署名方式 HFE_v^-

HFE_v^- [33, 34] は，公開鍵暗号方式 HFE を署名方式に応用したものである。HFE と同様に $\mathbb{F}_q$ の n 次拡大体 $K = \mathbb{F}_{q^n}$ をとり，$\mathbb{F}_q$-線形同型写像 $\phi : \mathbb{F}_q^n \to K$ を固定する。正の整数 a $(a < n)$ と v を固定する。まず，$\mathcal{G}(X)$ は次のように変更される。

$$\mathcal{G}(X) = \sum_{0 \le i \le j}^{q^i+q^j \le D} \alpha_{i,j} X^{q^i+q^j} + \sum_{0 \le i}^{q^i \le D} \beta_i(x_{n+1}, \ldots, x_{n+v}) X^{q^i} + \gamma(x_{n+1}, \ldots, x_{n+v}) \qquad (\alpha_{i,j} \in K). \tag{5.5}$$

ここで, $\beta_i(x_{n+1}, \ldots, x_{n+v}), \gamma(x_{n+1}, \ldots, x_{n+v})$ は共に $\mathbb{F}_q^v$ から K への多項式写像であり, $\beta_i(x_{n+1}, \ldots, x_{n+v})$ は 1 次多項式, $\gamma(x_{n+1}, \ldots, x_{n+v})$ は 2 次多項式である。多変数多項式系 $G(\mathbf{x})$ は, 多変数多項式写像 $G = \phi^{-1} \circ \mathcal{G} \circ (\phi \times id_v) : \mathbb{F}_q^{n+v} \to \mathbb{F}_q^n$ により定める。$\alpha_{i,j} \in K$ と $\beta_i(x_{n+1}, \ldots, x_{n+v})$, $\gamma(x_{n+1}, \ldots, x_{n+v})$ を動かしてできる $G(\mathbf{x})$ のなすクラスを $\mathcal{C}_{\mathrm{HFE}_v^-}$ と定める。基本的には，これを $\mathcal{C}_{\mathrm{cent}} = \mathcal{C}_{\mathrm{HFE}_v^-}$ として構成される双極型システムを考えるのであるが，双極型システムを若干変更する。S は $\mathbb{F}_q^{n+v}$ 上のアフィン同型写像のままでよいが，T は $\mathbb{F}_q^n$ から $\mathbb{F}_q^{n-a}$ への最大ランクのアフィン写像と変更する。公開鍵は通常の双極型システムと同じように，$F(\mathbf{x}) = T \circ G(\mathbf{x}) \circ S$ と定める。よって，F は $\mathbb{F}_q^{n+v}$ から $\mathbb{F}_q^{n-a}$ への多変数多項式写像となる。メッセージ（のハッシュ値） $M \in \mathbb{F}_q^{n-a}$ に対する署名 $\sigma = F^{-1}(M)$ は以下のよう計算される。

1. $\mathbf{c} = T^{-1}(M) \in \mathbb{F}_q^n$ （の 1 つ）を計算する。
2. $B = \phi(\mathbf{c}) \in K$ を計算する。
3. $B' \in \mathbb{F}_q^v$ をランダムに選び， $A = \mathcal{G}^{-1}(B \,\|\, B')$ を Cantor-Zassenhaus アルゴリズムなどを用いて計算する。$\mathcal{G}^{-1}(B \,\|\, B')$ が存在しない場合は，B' の選択からやり直す。
4. $\mathbf{e} = \phi^{-1}(A)$ を計算する。

5. $\sigma = S^{-1}(\mathbf{e})$ を計算する。

HFE_v^- と同じ構造を持つ署名方式 GeMSS [14] は NIST PQC 標準化プロジェクト 第 3 ラウンドに選ばれたが，効率的な攻撃法が提案されたため [36]，第 4 ラウンドに進むことはできなかった。

5.2.4 署名方式 UOV

5.2.4.1 UOV の概要

UOV [29, 11] は，双極型システムを用いた署名方式である。UOV の中心写像は，決まったいくつかの変数に値を代入することで 1 次式に変形でき，連立線形方程式の求解手法を用いて，効率的に署名生成が可能である。v, m を正の整数とし，$n = v + m$ とする。2 次多項式からなる多変数多項式系 $G(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x}))$ を次の形で与える。

$$g_k(\mathbf{x}) = \sum_{\substack{1 \le i \le v \\ v+1 \le j \le n}} \alpha_{i,j}^{(k)} x_i x_j + \sum_{1 \le i \le j \le v} \beta_{i,j}^{(k)} x_i x_j + \sum_{1 \le i \le n} \gamma_i^{(k)} x_i + \eta^{(k)} \quad (k = 1, \ldots, m)$$

ここで，$\alpha_{i,j}^{(k)}, \beta_{i,j}^{(k)}, \gamma_i^{(k)}, \eta^{(k)} \in \mathbb{F}_q$ である。$G(\mathbf{x})$ の形で定義される多変数多項式写像を **UOV 多項式写像**と呼ぶ。$g_k(\mathbf{x})$ の 2 次多項式部分を $\tilde{g}_k(\mathbf{x})$ とすると，

$$\tilde{g}_k(\overbrace{0, \ldots, 0}^{v}, \overbrace{*, \ldots, *}^{m}) = 0 \tag{5.6}$$

となることが，UOV 多項式写像の特徴である。$x_1, \ldots, x_v$ を vinegar 変数，$x_{v+1}, \ldots, x_n$ を oil 変数と呼ぶ。$G(\mathbf{x})$ の vinegar 変数に（ランダムな）値を代入すると，(5.6) により oil 変数に関する 1 次式が得られる。$G(\mathbf{x})$ の逆写像は，連立線形方程式の求解手法を用いて効率的に計算できる。具体的に，任意の $\mathbf{c} = (c_1, \ldots, c_m) \in \mathbb{F}_q^m$ に対し，$\mathbf{b} = G^{-1}(\mathbf{c})$（の一つ）が以下のように計算できる。

1. $b_1, \ldots, b_v \in \mathbf{F}_q$ をランダムにとる。
2. $g_1(\mathbf{x}), \ldots, g_m(\mathbf{x})$ に $(x_1, \ldots, x_v) = (b_1, \ldots, b_v)$ を代入して得られる $x_{v+1}, \ldots, x_n$ に関する 1 次式をそれぞれ $\bar{g}_1(x_{v+1}, \ldots, x_n), \ldots, \bar{g}_m(x_{v+1}, \ldots, x_n)$ とする。連立線形方程式

$$\begin{cases} \bar{g}_1(x_{v+1}, \ldots, x_n) = c_1 \\ \qquad\qquad\qquad \vdots \\ \bar{g}_m(x_{v+1}, \ldots, x_n) = c_m \end{cases}$$

   の解を計算し，それを $b_{v+1}, \ldots, b_n$ と置く。もし解がなければ Step 1 に戻る。
3. $\mathbf{b} = (b_1, \ldots, b_n)$

$\alpha_{i,j}^{(k)}, \beta_{i,j}^{(k)}, \gamma_i^{(k)}, \eta^{(k)} \in \mathbb{F}_q$ を動かしてできる $G(\mathbf{x})$ の集合を $\mathcal{C}_{\mathrm{UOV}}$ としたとき，$\mathcal{C}_{\mathrm{cent}} = \mathcal{C}_{\mathrm{UOV}}$ として構成される双極型システムの署名方式を UOV と呼ぶ。但し，通常の双極型システムで使用するアフィン同型写像 T は，UOV の安全性には貢献しないので必要ない。すなわち，秘密鍵は $G(\mathbf{x}) \in \mathcal{C}_{\mathrm{UOV}}$ と $\mathbb{F}_q^n$ 上のアフィン同型写像 S で，公開鍵は $F(\mathbf{x}) = G(\mathbf{x}) \circ S$ となる。$F(\mathbf{x})$ の成分の 2 次多項式部分を $\tilde{f}_1(\mathbf{x}), \ldots, \tilde{f}_m(\mathbf{x})$ とし，部分空間 $O\,(\subset \mathbb{F}_q^n)$ を

$$O = S^{-1}(\{(\overbrace{0, \ldots, 0}^{v}, \mathbf{a}) \in \mathbb{F}_q^n \mid \mathbf{a} \in \mathbb{F}_q^m\})$$

とすると，(5.6) により $\tilde{f}_i(\mathbf{o}) = 0 \ \ (i = 1, \ldots, m,\ \mathbf{o} \in O)$ を満たす。このような性質を持つ部分空間を oil 空間という。逆に，多変数 2 次多項式系 $H(\mathbf{x})$ が o 次元の oil 空間 $O\,(\subset \mathbb{F}_q^n)$ を持ち，$o \ge m$ を満たすならば，$H(\mathbf{x})$ は UOV の公開鍵として使用できる。

5.2.4.2 UOV の公開鍵長の削減

双極型システムの公開鍵 $F(\mathbf{x})$ は，通常，その係数集合の形で記述され，その中でも，2 次多項式部分の係数集合が公開鍵の大部分を占める。$P(\mathbf{x})$ を多変数 2 次多項式系とし，$\tilde{p}_1(\mathbf{x}), \ldots, \tilde{p}_m(\mathbf{x})$ をその 2 次多項式部分とすると，ある行列 $\mathrm{P}_1, \ldots, \mathrm{P}_m \in \mathbb{F}_q^{n\times n}$ により，

$$\tilde{p}_i(\mathbf{x}) = \mathbf{x}\,\mathrm{P}_i\,\mathbf{x}^\top \quad (i = 1, \ldots, m)$$

と表すことができる。一般に，行列 $\mathrm{P} = (p_{ij}) \in \mathbb{F}_q^{h\times h}$ に対し，上三角行列 $\mathrm{upper}(\mathrm{P}) = (c_{ij}) \in \mathbb{F}_q^{h\times h}$ を

$$c_{ij} = \begin{cases} p_{ii} & i = j, \\ p_{ij} + p_{ji} & i < j, \\ 0 & i > j \end{cases}$$

で定義すると，$\tilde{p}_i(\mathbf{x}) = \mathbf{x}\,\mathrm{P}_i\,\mathbf{x}^\top = \mathbf{x}\,\mathrm{upper}(\mathrm{P}_i)\,\mathbf{x}^\top$ が成り立つ。特に，P_i はすべて上三角行列で選ぶことができる。

UOV の中心写像 $G(\mathbf{x})$ の 2 次多項式部分に対応する上三角行列を $\mathrm{G}_1, \ldots, \mathrm{G}_m$ とし，公開鍵 $F(\mathbf{x})$ の 2 次多項式部分に対応する上三角行列を $\mathrm{F}_1, \ldots, \mathrm{F}_m$ とすると，

$$\mathrm{G}_i = \begin{pmatrix} \mathrm{G}_{i,1} & \mathrm{G}_{i,2} \\ \mathbf{0}_{o\times v} & \mathbf{0}_{o\times o} \end{pmatrix}, \quad \mathrm{F}_i = \begin{pmatrix} \mathrm{F}_{i,1} & \mathrm{F}_{i,2} \\ \mathbf{0}_{o\times v} & \mathrm{F}_{i,3} \end{pmatrix} \quad (\mathrm{G}_{i,1}, \mathrm{F}_{i,1} \in \mathbb{F}_q^{v\times v},\ \mathrm{G}_{i,2}, \mathrm{F}_{i,2} \in \mathbb{F}_q^{v\times o},\ \mathrm{F}_{i,3} \in \mathbb{F}_q^{o\times o})$$

の形で表すことができる。ここで，$\mathrm{G}_{i,1}, \mathrm{G}_{i,3}, \mathrm{F}_{i,1}, \mathrm{F}_{i,3}$ は上三角行列である。今，アフィン同型写像 S の線形部分を表す行列 S （線形写像は $\mathbf{x} \mapsto \mathbf{x}\,\mathrm{S}$ の形）が

$$\mathrm{S} = \begin{pmatrix} \mathbf{I}_v & \mathbf{0}_{v\times o} \\ \mathrm{S}_0 & \mathbf{I}_o \end{pmatrix} \quad (\mathrm{S}_0 \in \mathbb{F}_q^{o\times v})$$

の形で表される場合に限定する。（但し，$\mathbf{I}_\ell$ は ℓ 次単位行列を表す。）すると，$\mathrm{F}_i = \mathrm{upper}(\mathrm{S}\,\mathrm{G}_i\,\mathrm{S}^\top)$ となるので，次の関係が成り立つ。

$$\begin{aligned} &\cdot\ \mathrm{G}_{i,1} = \mathrm{F}_{i,1},\ \ \mathrm{G}_{i,2} = \mathrm{F}_{i,2} - (\mathrm{F}_{i,1} + \mathrm{F}_{i,1}^\top)\,\mathrm{S}_0^\top, \\ &\cdot\ \mathrm{F}_{i,3} = \mathrm{upper}(-\mathrm{S}_0\,\mathrm{F}_{i,1}^\top\,\mathrm{S}_0^\top + \mathrm{S}_0\,\mathrm{F}_{i,2}) = \mathrm{upper}(-\mathrm{S}_0\,\mathrm{F}_{i,1}\,\mathrm{S}_0^\top + \mathrm{S}_0\,\mathrm{F}_{i,2}). \end{aligned} \tag{5.7}$$

これより，$\mathrm{F}_{i,1}$ は任意の上三角行列, $\mathrm{F}_{i,2}$ は任意の行列で選べることが分かる。そこで，$\mathrm{F}_{i,1}, \mathrm{F}_{i,2}$ の成分すべてを公開鍵として記述する代わりに，$\mathrm{F}_{i,1}, \mathrm{F}_{i,2}$ を疑似乱数生成器を用いて構成することにして，そのシードのみを公開鍵として記述することにより公開鍵長を削減できる。このようにして，UOV の公開鍵の 2 次多項式部分は，シードと (5.7) で求められた $\mathrm{F}_{i,3}$ $(i = 1, \ldots, m)$ だけで記述できる。

一般に，双極型システムの公開鍵長は n, m に関して，$\mathcal{O}(mn^2)$ の増大度を持ち，大きくなりやすい。UOV では，上の公開鍵の記述方法を使うことにより，公開鍵長の増大度は $\mathcal{O}(m^3)$ となり，UOV のパラメータが $2m < n$ で選ばれることを踏まえると，一般の双極型システムの公開鍵の記述方法よりも，公開鍵長を削減できる。この削減方法は，5.3 節で記述する（UOV の変種である）QR-UOV や MAYO にも利用されている。

5.2.4.3 署名方式 Rainbow

署名方式 Rainbow [17] は UOV を多層化して作られる。正の整数 $t, v_1, o_1, \ldots, o_t$ に対し，$v_{i+1} = v_i + o_i$ により，$v_2, \ldots, v_{t+1}$ を順次定める。また，$i = 1, \ldots, t$ に対し，$S_i = \{1, \ldots, v_i\}$, $O_i = \{v_i + 1, \ldots, v_{i+1}\}$ とおく。S_i の元の個数は v_i で，O_i の元の個数は o_i である。変数の個数を $n = v_{t+1}$，式数を $m = n - v_1$ とする多変数多項式系 $G(\mathbf{x}) = (g_{v_1+1}(\mathbf{x}), \ldots, g_n(\mathbf{x}))$ を次の形で与える：

$$g_k(x_1, \ldots, x_n) = \sum_{i\in O_h, j\in S_h} \alpha_{i,j}^{(k)} x_i x_j + \sum_{i,j\in S_h,\, i\le j} \beta_{i,j}^{(k)} x_i x_j + \sum_{i\in S_{h+1}} \gamma_i^{(k)} x_i + \eta^{(k)} \quad (k = v_1{+}1, \ldots, n)$$

但し，h は k が属する層番号，すなわち，“$k \in O_h$” で定まる整数 $1 \leq h \leq t$ である。$\alpha_{i,j}^{(k)}, \beta_{i,j}^{(k)}, \gamma_i^{(k)}, \eta^{(k)} \in \mathbb{F}_q$ を動かしてできる $G(\mathbf{x})$ のなすクラスを $\mathcal{C}_{\text{Rainbow}}$ と定め，これを Rainbow の中心写像のクラスとする。任意の $\mathbf{c} = (c_1, \ldots, c_m) \in \mathbb{F}_q^m$ に対し，$\mathbf{b} = G^{-1}(\mathbf{c})$（の一つ）が以下のように計算できる。

1. $b_1, \ldots, b_{v_1} \in \mathbb{F}_q$ をランダムにとる。
2. $h = 1, 2, \ldots, t$ に対し，以下を実行：

   $g_{v_h+1}(\mathbf{x}), \ldots, g_{v_{h+1}}(\mathbf{x})$ に $(x_1, \ldots, x_{v_h}) = (b_1, \ldots, b_{v_h})$ を代入して得られる $x_{v_h+1}, \ldots, x_{v_{h+1}}$ に関する 1 次式をそれぞれ $\bar{g}_{v_h+1}(x_{v_h+1}, \ldots, x_{v_{h+1}}), \ldots, \bar{g}_{v_{h+1}}(x_{v_h+1}, \ldots, x_{v_{h+1}})$ とする。連立線形方程式

   $$\begin{cases} \bar{g}_{v_h+1}(x_{v_h+1}, \ldots, x_{v_{h+1}}) = c_{v_h+1} \\ \qquad\qquad \vdots \\ \bar{g}_{v_{h+1}}(x_{v_h+1}, \ldots, x_{v_{h+1}}) = c_{v_{h+1}} \end{cases}$$

   の解を計算し，それを $b_{v_h+1}, \ldots, b_{v_{h+1}}$ と置く。もし解がなければ Step 1 に戻る。
3. $\mathbf{b} = (b_1, \ldots, b_n)$

このアルゴリズムの Step 2 の第 h ループ内では，S_h を添字とする変数を vinegar 変数，O_h を添字とする変数を oil 変数とし，UOV の中心写像と同じ逆写像計算を行っている。このことから，Rainbow を UOV の多層化と見ることができる。Rainbow は NIST PQC 標準化プロジェクト 第 3 ラウンドに選ばれたが，Rainbow の EIP 問題を解くことにより，小さなサイズの UOV への攻撃に帰着する攻撃が提案され [7, 12]，その結果，level I, III, V として提案されていたパラメータがその安全性レベルに到達しないことになり（安全性レベル 143 bits が 69 bits に，207 bits が 157bits に，272 bits が 206 bits に下がった），第 4 ラウンドに進むことはできなかった。

5.2.5 MPC-in-the-Head による署名方式の構成

5.2.5.1 秘匿マルチパーティ計算

Ishai らによって導入された MPC-in-the-Head [28] は，秘匿マルチパーティ計算からゼロ知識証明を構成し，さらに Fiat-Shamir 変換により署名方式を構成する枠組みである。本来，MPC-in-the-Head の枠組みは広く，多変数公開鍵暗号に限定された技術ではないが，多変数公開鍵暗号においては，MQ 問題に付随する MPC-in-the-Head と，MinRank 問題に付随する MPC-in-the-Head が重要であるため，この 2 つの場合に限定して説明する。

$\mathcal{R} \subset \{0,1\}^* \times \{0,1\}^*$ を関係とする。命題 a に対して，$(a, x) \in \mathcal{R}$ であるとき，x は a の証拠（witness）であるという。ここでは，$\mathcal{R}$ として，MQ 問題（あるいは，MinRank 問題）のインスタンスを命題とし，その解を証拠とする関係に限定する。N 組のパーティ $\mathcal{P}_1, \ldots, \mathcal{P}_N$ が存在するとする。加法構造を持つ代数系の元 b に対し，分散 $[\![b]\!]$ は

$$[\![b]\!] = ([\![b]\!]_1, \ldots, [\![b]\!]_N) \text{ であり，} [\![b]\!]_1 + \cdots + [\![b]\!]_N = b$$

なるものを意味するとする。命題 a に対し，以下のような性質を持つ秘匿マルチパーティ計算 f を考える。

- 秘密情報 x の分散 $[\![x]\!]$ に対し，$\mathcal{P}_j$ は $[\![x]\!]_j$ を入力として受け取る。
- f は ‘受理’ か ‘棄却’ を返す。$(a, x) \in \mathcal{R}$ ならば，‘受理’ が返される。
- $N - 1$ 組以下のパーティのビューが集まっても x の情報は全く漏れない。

命題が以下のような MQ 問題のインスタンスである場合を考える。

$$\begin{cases} \mathbf{x}\,\mathrm{A}_1\,\mathbf{x}^\top + \mathbf{x}\,\mathbf{b}_1^\top = y_1 \\ \mathbf{x}\,\mathrm{A}_2\,\mathbf{x}^\top + \mathbf{x}\,\mathbf{b}_2^\top = y_2 \\ \qquad\qquad \vdots \\ \mathbf{x}\,\mathrm{A}_m\,\mathbf{x}^\top + \mathbf{x}\,\mathbf{b}_m^\top = y_m \end{cases} \tag{5.8}$$

ここで，$\mathrm{A}_1, \ldots, \mathrm{A}_m \in \mathbb{F}_q^{n\times n}$，$\mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbb{F}_q^n$ である。この場合の（MQOM [22] で利用されている）秘匿マルチパーティ計算 f_{MQ} の入力は，この MQ 問題の解 $\mathbf{x}^*$ の分散 $[\![\mathbf{x}^*]\!]$ である。η を正の整数とする。正の整数 n_1, n_2 を $n_1 n_2 \geq n$ なるように選んでおく。また，$u_1, \ldots, u_{n_2} \in \mathbb{F}_q$ を相異なる元として固定しておく。このとき，f_{MQ} の計算手順は以下の通りである。

秘匿マルチパーティ計算 f_{MQ}

1. 乱数生成オラクル[*1] $\mathcal{O}_R$ により $\gamma_1, \ldots, \gamma_m \in \mathbb{F}_{q^\eta}$ が作られ，全パーティに送信する。
2. 各パーティ $\mathcal{P}_j$ は $[\![z]\!]_j = \sum_{i=1}^m \gamma_i \left([\![y_i]\!]_j - [\![\mathbf{x}^*]\!]_j \, \mathbf{b}_i^\top\right)$ を計算する。ここで，$[\![y_i]\!] = (y_i, 0, 0, \ldots, 0)$ である。
3. 各パーティ $\mathcal{P}_j$ は $[\![\mathbf{w}]\!]_j = [\![\mathbf{x}^*]\!]_j \left(\sum_{i=1}^m \gamma_i \, \mathrm{A}_i^\top\right)$ を計算する。
4. ヒントオラクル $\mathcal{O}_H$ により，$a_1, \ldots, a_{n_2} \in \mathbb{F}_{q^\eta}$，$Q'(u) \in \mathbb{F}_{q^\eta}[u]$ の分散 $[\![a_1]\!], \ldots, [\![a_{n_2}]\!]$，$[\![Q'(u)]\!]$ が作られ，対応するパーティに配布される。ここで，$a_1, \ldots, a_{n_2}$ はランダムに選ばれ，$Q'(u)$ は次のように定められる。まず，$n_1 - 1$ 次以下の多項式 $X_\ell(u) \in \mathbb{F}_q[u]$，$W_\ell(u) \in \mathbb{F}_{q^\eta}[u]$ $(\ell = 1, \ldots, n_2)$ を
$$\begin{cases} X_\ell(u_1) = x^*_{(\ell-1)n_1+1} \\ \quad\vdots \\ X_\ell(u_{n_1}) = x^*_{(\ell-1)n_1+n_1} \end{cases} \qquad \begin{cases} W_\ell(u_1) = w_{(\ell-1)n_1+1} \\ \quad\vdots \\ W_\ell(u_{n_1}) = w_{(\ell-1)n_1+n_1} \end{cases}$$
を満たす補間多項式によって計算する。次に，$\tilde{W}_\ell(u) \in \mathbb{F}_{q^\eta}[u]$ $(\ell = 1, \ldots, n_2)$ を
$$\tilde{W}_\ell(u) = W_\ell(u) + a_j \, (u - u_1)(u - u_2) \cdots (u - u_{n_1})$$
とし，$Q(u) \in \mathbb{F}_{q^\eta}[u]$ を $Q(u) = \sum_{\ell=1}^{n_2} X_\ell(u) \, \tilde{W}_\ell(u)$ で定める。最後に，q_0 を $Q(u)$ の定数項とし，$Q(u) = u \cdot Q'(u) + q_0$ で $Q'(u)$ を定める。
5. 各パーティ $\mathcal{P}_j$ は $n_1 - 1$ 次以下の多項式 $[\![X_\ell]\!]_j(u) \in \mathbb{F}_q[u]$，$[\![W_\ell]\!]_j(u) \in \mathbb{F}_{q^\eta}[u]$ $(\ell = 1, \ldots, n_2)$ を
$$\begin{cases} [\![X_\ell]\!]_j(u_1) = [\![x^*_{(\ell-1)n_1+1}]\!]_j \\ \quad\vdots \\ [\![X_\ell]\!]_j(u_{n_1}) = [\![x^*_{(\ell-1)n_1+n_1}]\!]_j \end{cases} \qquad \begin{cases} [\![W_\ell]\!]_j(u_1) = [\![w_{(\ell-1)n_1+1}]\!]_j \\ \quad\vdots \\ [\![W_\ell]\!]_j(u_{n_1}) = [\![w_{(\ell-1)n_1+n_1}]\!]_j \end{cases}$$
を満たす補間多項式によって計算する。（$[\![X_\ell]\!](u)$, $[\![W_\ell]\!](u)$ は，それぞれ $X_\ell(u)$, $W_\ell(u)$ の分散となる。）
6. 各パーティ $\mathcal{P}_j$ は $[\![\tilde{W}_\ell]\!]_j(u) \in \mathbb{F}_{q^\eta}[u]$ $(\ell = 1, \ldots, n_2)$ を
$$[\![\tilde{W}_\ell]\!]_j(u) = [\![W_\ell]\!]_j(u) + [\![a_\ell]\!]_j \, (u - u_1)(u - u_2) \cdots (u - u_{n_1})$$
で定める。（$[\![\tilde{W}_\ell]\!](u)$ は，$\tilde{W}_\ell(u)$ の分散となる。）
7. 各パーティ $\mathcal{P}_j$ は $[\![q_0]\!]_j = n_1^{-1} \cdot ([\![z]\!]_j - \sum_{i=1}^{n_1} u_i \, [\![Q']\!]_j(u_i))$ を計算する。
8. 乱数生成オラクルを用いて $r \in \mathbb{F}_{q^\eta} \backslash \{u_1, \ldots, u_{n_1}\}$ を生成し，全パーティに送信する。
9. 各パーティ $\mathcal{P}_j$ は $[\![c_\ell]\!]_j = [\![\tilde{W}_\ell]\!]_j(r)$ $(\ell = 1, \ldots, n_2)$ を計算する。
10. 全パーティは $[\![c_\ell]\!]$ $(\ell = 1, \ldots, n_2)$ を共有し，$c_\ell \in \mathbb{F}_{q^\eta}$ $(\ell = 1, \ldots, n_2)$ を計算する。
11. 各パーティ $\mathcal{P}_j$ は $[\![v]\!]_j = r \cdot [\![Q']\!]_j(r) + [\![q_0]\!]_j - \sum_{\ell=1}^{n_2} c_\ell \, [\![X_\ell]\!]_j(r)$ を計算する。
12. 全パーティは $[\![v]\!]$ を共有し，v を計算する。
13. $v = 0$ なら ‘受理’，それ以外は ‘棄却’ を出力する。

この計算について補足する。この計算では，$\mathbf{x}^*$ が (5.8) の解であることを確認する代わりに，
$$\sum_{i=1}^m \gamma_i \left(y_i - \mathbf{x}^* \, \mathrm{A}_i \, {\mathbf{x}^*}^\top - \mathbf{x}^* \, \mathbf{b}_i^\top\right) = 0$$

[*1] Randomness Oracle で，安全性証明に用いられるランダムオラクルとは異なる。

であることを確認している。$\mathbf{x}^*$ が (5.8) の解でなくても，この等式は $1/q^\eta$ の確率で成り立つ。等式を書き直すと，

$$\sum_{i=1}^{m}\gamma_i\,(y_i-\mathbf{x}^*\,\mathbf{b}_i^\top)=\sum_{i=1}^{m}\gamma_i\,(\mathbf{x}^*\,\mathrm{A}_i\,\mathbf{x}^{*\top})=\mathbf{x}^*\left(\sum_{i=1}^{m}\gamma_i\,\mathrm{A}_i\right)\mathbf{x}^{*\top}=\langle\mathbf{x}^*,\,\mathbf{w}\rangle,\qquad\left(\mathbf{w}=\mathbf{x}^*\left(\sum_{i=1}^{m}\gamma_i\,\mathrm{A}_i^\top\right)\right)$$

となる。よって，$z=\sum_{i=1}^{m}\gamma_i\,(y_i-\mathbf{x}^*\,\mathbf{b}_i^\top)$ とおくならば，$z=\langle\mathbf{x}^*,\,\mathbf{w}\rangle$ を確かめればよい。これは，

$$z=\sum_{i=1}^{n_1}\sum_{\ell=1}^{n_2}X_\ell(u_i)\tilde{W}_\ell(u_i)=\sum_{i=1}^{n_1}Q(u_i)\tag{5.9}$$

と同値である。$v=0$ であれば，Schwartz-Zippel の補題により，高い確率で (5.9) が満たされることになる。η を大きくすることで，この確率を 1 に近づけることができる。

5.2.5.2 ゼロ知識証明への変換

MPC-in-the-Head では秘匿マルチパーティ計算からゼロ知識証明を構成する。このゼロ知識証明は，命題 a に対して証拠 x を知っているかどうかを検証するものである。秘匿マルチパーティ計算 f_{MQ} に対応するゼロ知識証明の基本設計は以下の通りである。

f_{MQ} に対応するゼロ知識証明

1. 証明者は，証拠 $\mathbf{x}^*$ の分散 $[\![\mathbf{x}^*]\!]$ を作成する。そして，すべての $j\in\{1,\ldots,N\}$ に対し，$[\![\mathbf{x}^*]\!]_j$ のコミットメントを作成し，検証者に送る。
2. 検証者は，ランダムに $\gamma_1,\ldots,\gamma_m\in\mathbb{F}_{q^\eta}$ を生成し，これらをチャレンジとして証明者に送る。
3. 証明者は，$a_1,\ldots,a_{n_2}$, $Q'(u)$ を生成し，これらの分散 $[\![a_1]\!],\ldots,[\![a_{n_2}]\!]$, $[\![Q']\!](u)$ を作成する。そして，すべての $j\in\{1,\ldots,N\}$ に対し，$[\![a_1]\!]_j,\ldots,[\![a_{n_2}]\!]_j$, $[\![Q']\!]_j(u)$ のコミットメントを作成し，検証者に送る。
4. 検証者は，ランダムに $r\in\mathbb{F}_{q^\eta}\backslash\{u_1,\ldots,u_{n_1}\}$ を生成し，これらをチャレンジとして証明者に送る。
5. 証明者は，（全てのパーティの計算を “頭の中で” 行い，）$[\![c_1]\!],\ldots,[\![c_{n_2}]\!]$，および，$[\![v]\!]$ を作成する。そして，これらを検証者に送る。
6. 検証者は，ランダムに $i^*\in\{1,2,\ldots,N\}$ を生成し，これをチャレンジとして証明者に送る。
7. 証明者は，すべての $j\in\{1,2,\ldots,N\}\backslash\{i^*\}$ に対し，$[\![\mathbf{x}]\!]_j$, $[\![a_1]\!]_j,\ldots,[\![a_{n_2}]\!]_j$, $[\![Q']\!]_j(u)$ を検証者に開示する。
8. 検証者は，以下を検証し，全て正しければ ‘受理’ を，それ以外は ‘棄却’ を出力する。
   - ✓ すべての $j\in\{1,2,\ldots,N\}\backslash\{i^*\}$ に対し，$[\![\mathbf{x}^*]\!]_j$ のコミットメント，および，$[\![a_1]\!]_j,\ldots,[\![a_{n_2}]\!]_j$, $[\![Q']\!]_j(u)$ のコミットメントが正しいこと
   - ✓ すべての $j\in\{1,2,\ldots,N\}\backslash\{i^*\}$ に対し，$[\![\mathbf{x}^*]\!]_j$, $[\![a_1]\!]_j,\ldots,[\![a_{n_2}]\!]_j$, $[\![Q']\!]_j(u)$ から $\mathcal{P}_j$ と同じ計算を行ったとき，$[\![c_1]\!]_j,\ldots,[\![c_{n_2}]\!]_j$, $[\![v]\!]_j$ の計算結果が一致すること
   - ✓ $v=0$ であること

このゼロ知識証明について補足する。f_{MQ} における全てのパーティのビューは（もともと開示されるものを除き）コミットメントが作成され，検証者に送られている。また，f_{MQ} において乱数生成オラクルが介入する部分は，検証者のチャレンジに置き換えられている。そして，1 つのパーティ以外のすべてのパーティに対するビューが開示され，そのビューから各パーティと同じ計算を行って得られる結果と開示情報が一致すること，および，$v=0$ であることにより検証者は ‘受理’ を行っている。このゼロ知識証明の健全性誤差（soundness error）ε は約 $1/N$ である。このゼロ知識証明を τ 回繰り返すことにより，全体の健全性誤差を ε^τ にすることができる。

このゼロ知識証明に Fiat-Shamir 変換を施すことで署名方式 MQOM [22] が構成できる。MQOM ついては，5.3.4 節で詳しく述べる。また，5.3.5 節で詳しく述べる MiRitH は MinRank 問題に付随する秘匿マルチパーティ計算から構成される署名方式である。

5.3　多変数多項式に基づく主要な暗号方式

多変数公開鍵暗号で標準化が有力視されるのは効率的な検証と短い署名長を持つ署名方式の UOV である。但し，UOV は公開鍵長が大きくなりやすいという性質を持つため，公開鍵長の削減手法を取り入れている UOV の変種である QR-UOV と MAYO も標準化の有力候補である。

また，MPC-in-the-Head では，MQ 問題に関するマルチパーティ計算に基づく署名方式 MQOM と，MinRank 問題に関するマルチパーティ計算に基づく署名方式 MiRitH が注目されている。

表 5.2: 多変数多項式に基づく暗号の分類

文献	暗号化	鍵交換	署名
UOV [29, 11]			○
QR-UOV [24, 25]			○
MAYO [8, 9]			○
MQOM [22]			○
MiRitH [3]			○

5.3.1　署名方式 UOV

5.3.1.1　UOV の概要

5.2.4.1 節で UOV の基本アルゴリズムは述べたため，この節でのアルゴリズムの記述は割愛する。NIST PQC 標準化プロジェクト追加署名第 1 ラウンドに提出された UOV [11] のアルゴリズムには，さらに，5.2.4.2 節で述べた公開鍵長の削減手法が取り入れられている。

5.3.1.2　UOV のパラメータ選択

UOV の設計に必要なパラメータは，q, m, n である。NIST PQC 標準化プロジェクト追加署名第 1 ラウンドに提出されたドキュメント [11] では，以下のように UOV のパラメータ見積もりが公開されている。

表 5.3: UOV のパラメータと鍵および署名のサイズ

(q, m, n)	安全性レベル	公開鍵サイズ	秘密鍵サイズ	署名サイズ
$(256, 44, 112)$	レベル 1	$43,576$ Bytes	48 Bytes	128 Bytes
$(\ 16, 64, 160)$	レベル 1	$66,576$ Bytes	48 Bytes	96 Bytes
$(256, 72, 184)$	レベル 3	$189,232$ Bytes	48 Bytes	200 Bytes
$(256, 96, 244)$	レベル 5	$446,992$ Bytes	48 Bytes	260 Bytes

5.3.2 署名方式 QR-UOV

5.3.2.1 QR-UOV の概要

QR-UOV [24, 25] は UOV の変種である。$\mathbb{F}_{q^\ell}$ 上の行列の集合 $\mathbb{F}_{q^\ell}^{n'\times n'}$ は，$\mathbb{F}_q$ 上の行列の集合 $\mathbb{F}_q^{n'\ell\times n'\ell}$ の部分集合と見なすことができる。この部分集合に属する行列は，$\mathbb{F}_q^{n'\ell\times n'\ell}$ の元として表示するよりも，$\mathbb{F}_{q^\ell}^{n'\times n'}$ の元として表示する方が，サイズを $1/\ell$ 倍に圧縮できる。QR-UOV は，この性質を利用して UOV の公開鍵長を削減している。

ℓ, V, M を正の整数とし，$v=\ell\cdot V$, $m=\ell\cdot M$, $n=v+m$ とする。次数 ℓ の既約多項式 $f\in\mathbb{F}_q[t]$ を取り，$\mathbb{F}_q$ の拡大体 $E_f=\mathbb{F}_q[t]/(f)$ に，$\mathbb{F}_q$-基底を $1,t,t^2,\ldots,t^{\ell-1}$ で入れ，$\mathbb{F}_q$ 上のベクトル空間として $\mathbb{F}_q^\ell$ と同一視する。任意の $g\in\mathbb{F}_q[t]$ に対し，写像 $E_f\ni x\mapsto xg\in E_f$ は $\mathbb{F}_q$ 上の線形写像となる。よって，この写像は $\mathbb{F}_q$ 上の $\ell\times\ell$ 行列として表すことができる。この行列を $\Phi_g^f\in\mathbb{F}_q^{\ell\times\ell}$ と表し，$\mathcal{A}_f=\{\Phi_g^f\,|\,g\in E_f\}(\subset\mathbb{F}_q^{\ell\times\ell})$ とおく。$\phi:E_f\to\mathbb{F}_q$ を非自明な $\mathbb{F}_q$-線形写像で固定し，$\mathrm{W}=(\phi(t^{i+j-2}))_{ij}\in\mathbb{F}_q^{\ell\times\ell}$ とすると，任意の $\mathrm{X}\in\mathcal{A}_f$ に対して，$\mathrm{WX}\in\mathbb{F}_q^{\ell\times\ell}$ は対称行列になることが知られている [24, Theorem 1]。正の整数 a,b に対し，$\mathcal{A}_f^{a,b}$ を以下の形で表される $a\ell\times b\ell$ 行列の集合とする：

$$\begin{pmatrix}\mathrm{X}_{11} & \mathrm{X}_{12} & \cdots & \mathrm{X}_{1b}\\ \mathrm{X}_{21} & \mathrm{X}_{22} & \cdots & \mathrm{X}_{2b}\\ \vdots & \vdots & \ddots & \vdots\\ \mathrm{X}_{a1} & \mathrm{X}_{a2} & \cdots & \mathrm{X}_{ab}\end{pmatrix}\quad(X_{11},X_{12},\ldots,\mathrm{X}_{ab}\in\mathcal{A}_f)$$

$\mathrm{W}^{(a)}\in\mathbb{F}_q^{a\ell\times a\ell}$ を W が主対角線に a 個並ぶ対角行列とし，$\mathrm{W}^{(a)}\mathcal{A}_f^{a,b}=\{\mathrm{W}^{(a)}\,\mathrm{X}\,|\,\mathrm{X}\in\mathcal{A}_f^{a,b}\}$ $(\subset\mathbb{F}_q^{a\ell\times b\ell})$ とする。$\mathbb{F}_q^{a\ell\times b\ell}$ に属する一般の行列を記述するには，$\mathbb{F}_q$ の元が $a\,b\,\ell^2$ 個必要であるが，それが $\mathrm{W}^{(a)}\mathcal{A}_f^{a,b}$ に属する場合は，$a\,b\,\ell$ 個で記述できることに注意してほしい。

QR-UOV では，公開鍵 $F(\mathbf{x})$ の成分として 2 次斉次多項式を用いる。QR-UOV は双極型システムであるため，5.2.4.2 節で述べたように，$F(\mathbf{x})$ は，m 個の行列 $(\in\mathbb{F}_q^{n\times n})$ を用いて記述することができる。QR-UOV では，これらの行列がすべて $\mathrm{W}^{(V+M)}\mathcal{A}_f^{V+M,V+M}$ に属するような $F(\mathbf{x})$ だけを用いる。但し，これらの行列は上三角行列の形にはできないので，対称行列で記述する。この影響で，有限体の位数 q は奇数にする必要がある。$\mathrm{W}^{(V+M)}\mathcal{A}_f^{V+M,V+M}$ に属する行列は，一般の $\mathbb{F}_q^{n\times n}$ に属する行列よりも小さいサイズで記述できるため，オリジナルの UOV よりも QR-UOV の公開鍵の方が小さいサイズで記述できる。また，5.2.4.2 節で述べた UOV に対して適用できる公開鍵長削減手法は，QR-UOV に対しても適用可能である。

安全性パラメータを λ とし，以下の関数を用意する。

- $\mathrm{Expand}_{\mathrm{sk}}$： 任意の 2λ-ビット列から 1 個の $\mathcal{A}_f^{V,M}$ に属する行列を生成する疑似乱数生成関数
- $\mathrm{Expand}_{\mathrm{pk}}$： 任意の 2λ-ビット列から m 個の $\mathrm{W}^{(V)}\mathcal{A}_f^{V,V}$ に属する対称行列と，m 個の $\mathrm{W}^{(V)}\mathcal{A}_f^{V,M}$ に属する行列を生成する疑似乱数生成関数
- $\mathcal{H}:\{0,1\}^*\to\mathbb{F}_q^m$： 暗号学的ハッシュ関数

鍵生成

1. $\mathrm{seed}_{\mathrm{pk}}$, $\mathrm{seed}_{\mathrm{sk}}\in\{0,1\}^{2\lambda}$ をランダムに選ぶ。
2. $\mathrm{Expand}_{\mathrm{pk}}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{P}_{i,1}\in\mathrm{W}^{(V)}\mathcal{A}_f^{V,V}$ (対称行列)，$\mathrm{P}_{i,2}\in\mathrm{W}^{(V)}\mathcal{A}_f^{V,M}$ $(i=1,\ldots,m)$ を得る。
3. $\mathrm{Expand}_{\mathrm{sk}}(\mathrm{seed}_{\mathrm{sk}})$ の計算により，$\mathrm{S}_0\in\mathcal{A}_f^{V,M}$ を得る。
4. $\mathrm{P}_{i,3}=-\mathrm{S}_0^\top\,\mathrm{P}_{i,1}\mathrm{S}_0+\mathrm{P}_{i,2}^\top\,\mathrm{S}_0+\mathrm{S}_0^\top\,\mathrm{P}_{i,2}\in\mathbb{F}_q^{m\times m}\quad(i=1,\ldots,m)$ を計算する。

公開鍵は $\mathrm{pk}=(\mathrm{seed}_{\mathrm{pk}},\{\mathrm{P}_{i,3}\}_{i=1,\ldots,m})$，秘密鍵は $\mathrm{sk}=\mathrm{seed}_{\mathrm{sk}}$ である。次に，署名生成である。メッセージを $M\in\{0,1\}^*$ とする。

署名生成

1. pk から $(\mathrm{seed}_{\mathrm{pk}}, \{\mathrm{P}_{i,3}\}_{i=1,\ldots,m})$ を取り出す。
2. sk から $\mathrm{seed}_{\mathrm{sk}}$ を取り出す。
3. $\mathrm{Expand}_{\mathrm{pk}}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{P}_{i,1} \in \mathrm{W}^{(V)}\mathcal{A}_f^{V,V}$ (対称行列)，$\mathrm{P}_{i,2} \in W^{(V)}\mathcal{A}_f^{V,M}$ $(i = 1, \ldots, m)$ を得る。
4. $\mathrm{Expand}_{\mathrm{sk}}(\mathrm{seed}_{\mathrm{sk}})$ の計算により，$\mathrm{S}_0 \in \mathcal{A}_f^{V,M}$ を得る。
5. $\mathrm{G}_i = -\mathrm{P}_{i,1}\mathrm{S}_0 + \mathrm{P}_{i,2} \in \mathbb{F}_q^{v\times m}$ $(i = 1, \ldots, m)$ を計算する。
6. $\mathrm{U} = \begin{pmatrix} \mathbf{I}_v & \mathbf{0}_{v\times m} \\ -\mathrm{S}_0^\top & \mathbf{I}_m \end{pmatrix} \in \mathbb{F}_q^{n\times n}$ とおく。
7. $\mathbf{y} = (y_1, \ldots, y_v) \in \mathbb{F}_q^v$ をランダムに選ぶ。
8. $\mathrm{L} = (2\,(\mathbf{y}\,\mathrm{G}_1)^\top, \ldots, 2\,(\mathbf{y}\,\mathrm{G}_m)^\top) \in \mathbb{F}_q^{m\times m}$ を計算する。(縦ベクトルを m 列並べて行列を作る。)
9. $\mathbf{u} = (\mathbf{y}\,\mathrm{P}_{1,1}\,\mathbf{y}^\top, \ldots, \mathbf{y}\,\mathrm{P}_{m,1}\,\mathbf{y}^\top) \in \mathbb{F}_q^m$ を計算する。
10. $\mathrm{salt} \in \{0,1\}^\lambda$ をランダムに選び，$\mathbf{t} = \mathcal{H}(M \,\|\, \mathrm{salt})$ を計算する。
11. 連立線形方程式 $\mathbf{x}\,\mathrm{L} = \mathbf{t} - \mathbf{u}$ の解を計算し，解 $\mathbf{x} = (y_{v+1}, \ldots, y_n) \in \mathbb{F}_q^m$ を得る。もし解がなければ，Step 10 に戻る。
12. $\mathbf{s} = (y_1, \ldots, y_n)\,\mathrm{U}$ を計算する。

$\sigma = (\mathrm{salt}, \mathbf{s})$ が署名となる。最後に検証である。

検証

1. pk から $(\mathrm{seed}_{\mathrm{pk}}, \{\mathrm{P}_{i,3}\}_{i=1,\ldots,m})$ を取り出す。
2. σ から $(\mathrm{salt}, \mathbf{s})$ を取り出す。
3. $\mathrm{Expand}_{\mathrm{pk}}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{P}_{i,1} \in \mathrm{W}^{(V)}\mathcal{A}_f^{V,V}$ (対称行列)，$\mathrm{P}_{i,2} \in \mathrm{W}^{(V)}\mathcal{A}_f^{V,M}$ $(i = 1, \ldots, m)$ を得る。
4. $\mathrm{F}_i = \begin{pmatrix} \mathrm{P}_{i,1} & \mathrm{P}_{i,2} \\ \mathrm{P}_{i,2}^\top & \mathrm{P}_{i,3} \end{pmatrix}$ $(i = 1, \ldots, m)$ とおく。
5. $\mathbf{t} = \mathcal{H}(M \,\|\, \mathrm{salt})$ を計算する。
6. $\mathbf{t}' = (\mathbf{s}\,\mathrm{F}_1\,\mathbf{s}^\top, \ldots, \mathbf{s}\,\mathrm{F}_m\,\mathbf{s}^\top)$ を計算する。
7. $\mathbf{t} = \mathbf{t}'$ ならば '受理' を，それ以外は '棄却' を返す。

5.3.2.2 QR-UOV のパラメータ選択

QR-UOV の設計に必要なパラメータは，λ, q, v, m, ℓ である。NIST PQC 標準化プロジェクト追加署名第 1 ラウンドに提出されたドキュメント [25] では，以下のように QR-UOV のパラメータ見積もりが公開されている。

表 5.4: QR-UOV のパラメータと鍵および署名のサイズ

(λ, q, v, m, ℓ)	安全性レベル	公開鍵サイズ	秘密鍵サイズ	署名サイズ
$(128,\ 7,\ 740, 100, 10)$	レベル 1	$20,657$ Bytes	32 Bytes	331 Bytes
$(128,\ 31,\ 165,\ 60,\ 3)$	レベル 1	$23,657$ Bytes	32 Bytes	157 Bytes
$(128,\ 31,\ 600,\ 70, 10)$	レベル 1	$12,282$ Bytes	32 Bytes	435 Bytes
$(128, 127,\ 156,\ 54,\ 3)$	レベル 1	$24,271$ Bytes	32 Bytes	200 Bytes
$(192,\ 7, 1100, 140, 10)$	レベル 3	$55,173$ Bytes	48 Bytes	489 Bytes
$(192,\ 31,\ 246,\ 87,\ 3)$	レベル 3	$71,007$ Bytes	48 Bytes	232 Bytes
$(192,\ 31,\ 890, 100, 10)$	レベル 3	$34,423$ Bytes	48 Bytes	643 Bytes
$(192, 127,\ 228,\ 78,\ 3)$	レベル 3	$71,915$ Bytes	48 Bytes	292 Bytes
$(256,\ 7, 1490, 190, 10)$	レベル 5	$135,439$ Bytes	64 Bytes	662 Bytes
$(256,\ 31,\ 324, 114,\ 3)$	レベル 5	$158,453$ Bytes	64 Bytes	306 Bytes
$(256,\ 31, 1120, 120, 10)$	レベル 5	$58,564$ Bytes	64 Bytes	807 Bytes
$(256, 127,\ 306, 105,\ 3)$	レベル 5	$173,708$ Bytes	64 Bytes	392 Bytes

5.3.3 署名方式 MAYO

5.3.3.1 MAYO の概要

MAYO [8, 9] は UOV の変種である。公開鍵を作る基となる変数の個数が少ない多変数多項式系 $P(\mathbf{x})$ を用意しておき，検証者は，検証時（あるいはそれ以前）に $P(\mathbf{x})$ から MAYO の公開鍵 $F(\mathbf{x})$ を構成する。そのため，MAYO の公開鍵は，$F(\mathbf{x})$ の係数集合ではなく，$P(\mathbf{x})$ の係数集合となる。これにより，MAYO は，オリジナルの UOV に比べて公開鍵長を小さくすることができる。

m, v, o, k を正の整数とし，$o < m,\ n = v + o$ とする。2 次斉次多変数多項式写像 $P(\mathbf{x}) = (p_1(\mathbf{x}), \ldots, p_m(\mathbf{x})) : \mathbb{F}_q^n \to \mathbb{F}_q^m$ に対し，ある o 次元部分空間 $O\,(\subset \mathbb{F}_q^n)$ があり，

$$P(\mathbf{o}) = \mathbf{0}_m \quad (\mathbf{o} \in O)$$

を満たすとする。5.2.4.1 節の言葉を使えば，O は oil 空間である。もし $o \geq m$ であれば，$P(\mathbf{x})$ は UOV の公開鍵として使用できるが，$o < m$ なのでそれはできない。$P^\star(\mathbf{x}_1, \ldots, \mathbf{x}_k) : \mathbb{F}_q^{kn} \to \mathbb{F}_q^m$ を次のようにおく。

$$P^\star(\mathbf{x}_1, \ldots, \mathbf{x}_k) = \sum_{i=1}^{k} P(\mathbf{x}_i)\,\mathrm{E}_{i,i} + \sum_{1 \leq i < j \leq k} P'(\mathbf{x}_i, \mathbf{x}_j)\,\mathrm{E}_{i,j} \tag{5.10}$$

ここで，$\mathrm{E}_{i,j} \in \mathbb{F}_q^{m \times m}\ (1 \leq i \leq j \leq k)$ は正則行列であり，$P'(\mathbf{x}, \mathbf{y})$ は $P(\mathbf{x} + \mathbf{y}) - P(\mathbf{x}) - P(\mathbf{y})$ で定まる双線形写像である。すると，

$$P^\star(\mathbf{o}_1, \ldots, \mathbf{o}_k) = \mathbf{0}_m \quad (\mathbf{o}_1, \ldots, \mathbf{o}_k \in O)$$

を満たすので，O^k が $P^\star(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ の oil 空間となる。$ko = \dim_{\mathbb{F}_q} O^k$ なので，$ko \geq m$ を満たせば，$P^\star(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ は UOV の公開鍵として使用できる。$\mathrm{E}_{i,j}\ (1 \leq i \leq j \leq n)$ をシステムパラメータとしておけば，$P^\star(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ は (5.10) により，$P(\mathbf{x})$ だけから構成できる。公開鍵を $P(\mathbf{x})$ の係数集合だけで記述することで，公開鍵のサイズを小さくした UOV が MAYO である。さらに，5.2.4.2 節で述べた UOV に対して適用できる公開鍵長削減手法は，MAYO に対しても適用可能である。

安全性パラメータを λ とし，以下の行列，関数を用意する。

- 正則行列 $\mathrm{E}_{i,j} \in \mathbb{F}_q^{m \times m}$ $(1 \le i \le j \le k)$
- $\mathrm{Expand}_{\mathrm{sk}}$： 任意の λ-ビット列から 1 個の $\mathbb{F}_q^{o \times v}$ に属する行列を生成する疑似乱数生成関数
- $\mathrm{Expand}_{\mathrm{pk}}$： 任意の λ-ビット列から m 個の $\mathbb{F}_q^{v \times v}$ に属する上三角行列と，m 個の $\mathbb{F}_q^{v \times o}$ に属する行列を生成する疑似乱数生成関数
- $\mathcal{H} : \{0,1\}^* \to \mathbb{F}_q^m$： 暗号学的ハッシュ関数

鍵生成

1. $\mathrm{seed}_{\mathrm{pk}},\ \mathrm{seed}_{\mathrm{sk}} \in \{0,1\}^\lambda$ をランダムに選ぶ。
2. $\mathrm{Expand}_{\mathrm{pk}}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{P}_{i,1} \in \mathbb{F}_q^{v \times v}$（上三角行列），$\mathrm{P}_{i,2} \in \mathbb{F}_q^{v \times o}$ $(i = 1, \ldots, m)$ を得る。
3. $\mathrm{Expand}_{\mathrm{sk}}(\mathrm{seed}_{\mathrm{sk}})$ の計算により，$\mathrm{R} \in \mathbb{F}_q^{o \times v}$ を得る。
4. $\mathrm{P}_{i,3} = \mathrm{upper}(-\mathrm{R}\,\mathrm{P}_{i,1}\mathrm{R}^\top - \mathrm{R}\,\mathrm{P}_{i,2}) \in \mathbb{F}_q^{o \times o}$ $(i = 1, \ldots, m)$ を計算する。

公開鍵は $\mathrm{pk} = (\mathrm{seed}_{\mathrm{pk}}, \{\mathrm{P}_{i,3}\}_{i=1,\ldots,m})$，秘密鍵は $\mathrm{sk} = \mathrm{seed}_{\mathrm{sk}}$ である。次に，署名生成である。メッセージを $M \in \{0,1\}^*$ とする。

署名生成

1. pk から $(\mathrm{seed}_{\mathrm{pk}}, \{\mathrm{P}_{i,3}\}_{i=1,\ldots,m})$ を取り出す。
2. sk から $\mathrm{seed}_{\mathrm{sk}}$ を取り出す。
3. $\mathrm{Expand}_{\mathrm{pk}}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{P}_{i,1} \in \mathbb{F}_q^{v \times v}$（上三角行列），$\mathrm{P}_{i,2} \in \mathbb{F}_q^{v \times o}$ $(i = 1, \ldots, m)$ を得る。
4. $\mathrm{Expand}_{\mathrm{sk}}(\mathrm{seed}_{\mathrm{sk}})$ の計算により，$\mathrm{R} \in \mathbb{F}_q^{o \times v}$ を得る。
5. $\mathrm{F}_i = (\mathrm{P}_{i,1} + \mathrm{P}_{i,1}^\top)\,\mathrm{R}^\top + \mathrm{P}_{i,2} \in \mathbb{F}_q^{v \times o}$ $(i = 1, \ldots, m)$ を計算する。
6. $\mathrm{U} = \begin{pmatrix} \mathbf{I}_v & \mathbf{0}_{v \times o} \\ \mathrm{R} & \mathbf{I}_o \end{pmatrix} \in \mathbb{F}_q^{n \times n}$ とおく。
7. $\mathbf{y}_1, \ldots, \mathbf{y}_k \in \mathbb{F}_q^v$（横ベクトル）をランダムに選ぶ。
8. $\mathrm{L}_i = \sum_{j=1}^{i}((\mathbf{y}_j\,\mathrm{F}_1)^\top, \ldots, (\mathbf{y}_j\,\mathrm{F}_m)^\top)\,\mathrm{E}_{j,i} + \sum_{j=i+1}^{k}((\mathbf{y}_j\,\mathrm{F}_1)^\top, \ldots, (\mathbf{y}_j\,\mathrm{F}_m)^\top)\,\mathrm{E}_{i,j} \in \mathbb{F}_q^{o \times m}$ $(i = 1, \ldots, k)$ を計算する。
9. $\mathrm{L} = \begin{pmatrix} \mathrm{L}_1 \\ \vdots \\ \mathrm{L}_k \end{pmatrix} \in \mathbb{F}_q^{ko \times m}$ とおく。
10. $\mathbf{u} = \sum_{i=1}^{k}(\mathbf{y}_i\,\mathrm{P}_{1,1}\,\mathbf{y}_i^\top, \ldots, \mathbf{y}_i\,\mathrm{P}_{m,1}\,\mathbf{y}_i^\top)\,\mathrm{E}_{i,i} + \sum_{1 \le i < j \le k}(\mathbf{y}_i\,(\mathrm{P}_{1,1} + \mathrm{P}_{1,1}^\top)\,\mathbf{y}_j^\top, \ldots, \mathbf{y}_i\,(\mathrm{P}_{m,1} + \mathrm{P}_{m,1}^\top)\,\mathbf{y}_j^\top)\,\mathrm{E}_{i,j} \in \mathbb{F}_q^m$ を計算する。
11. $\mathrm{salt} \in \{0,1\}^\lambda$ をランダムに選び，$\mathbf{t} = \mathcal{H}(M \,\|\, \mathrm{salt})$ を計算する。
12. 連立線形方程式 $(\mathbf{x}_1, \ldots, \mathbf{x}_k)\,\mathrm{L} = \mathbf{t} - \mathbf{u}$ の解を計算し，解 $(\mathbf{x}_1, \ldots, \mathbf{x}_k) = (\mathbf{z}_1, \ldots, \mathbf{z}_k) \in \mathbb{F}_q^{ko}$ を得る。もし解がなければ，Step 11 に戻る。
13. $\mathbf{s} = ((\mathbf{y}_1, \mathbf{z}_1)\,\mathrm{U}, \ldots, (\mathbf{y}_k, \mathbf{z}_k)\,\mathrm{U})$ を計算する。

$\sigma = (\mathrm{salt}, \mathbf{s})$ が署名となる。最後に検証である。

検証

1. pk から $(\mathrm{seed}_{\mathrm{pk}}, \{\mathrm{P}_{i,3}\}_{i=1,\ldots,m})$ を取り出す。
2. σ から $(\mathrm{salt}, (\mathbf{s}_1, \ldots, \mathbf{s}_k))$ を取り出す。
3. $\mathrm{Expand}_{\mathrm{pk}}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{P}_{i,1} \in \mathbb{F}_q^{v \times v}$（上三角行列），$\mathrm{P}_{i,2} \in \mathbb{F}_q^{v \times o}$ $(i = 1, \ldots, m)$ を得る。

4. $\mathrm{P}_i = \begin{pmatrix} \mathrm{P}_{i,1} & \mathrm{P}_{i,2} \\ \mathbf{0}_{v\times o} & \mathrm{P}_{i,3} \end{pmatrix}$ $(i = 1, \ldots, m)$ とおく。
5. $\mathbf{t} = \mathcal{H}(M \,\|\, \mathrm{salt})$ を計算する。
6. $\mathbf{t}' = \sum_{i=1}^{k} (\mathbf{s}_i \mathrm{P}_1 \mathbf{s}_i^\top, \ldots, \mathbf{s}_i \mathrm{P}_k \mathbf{s}_i^\top)\, \mathrm{E}_{i,i} + \sum_{1\le i<j\le k} (\mathbf{s}_i(\mathrm{P}_1 + \mathrm{P}_1^\top)\mathbf{s}_j^\top, \ldots, \mathbf{s}_i(\mathrm{P}_k + \mathrm{P}_k^\top)\mathbf{s}_j^\top)\, \mathrm{E}_{i,j}$ を計算する。
7. $\mathbf{t} = \mathbf{t}'$ ならば '受理' を，それ以外は '棄却' を返す。

5.3.3.2 MAYO のパラメータ選択

MAYO の設計に必要なパラメータは，λ, q, n, m, o, k である。MAYO の情報を発信しているウェブサイトで掲載されているドキュメント [10] では，以下のように MAYO のパラメータ見積もりが公開されている。

表 5.5: MAYO のパラメータと鍵および署名のサイズ

(λ, q, n, m, o, k)	安全性レベル	公開鍵サイズ	秘密鍵サイズ	署名サイズ
$(128, 16, 86, 78, 8, 10)$	レベル 1	1,420 Bytes	24 Bytes	454 Bytes
$(128, 16, 81, 64, 17, 4)$	レベル 1	4,912 Bytes	24 Bytes	186 Bytes
$(192, 16, 118, 108, 10, 11)$	レベル 3	2,986 Bytes	32 Bytes	681 Bytes
$(256, 16, 154, 142, 12, 12)$	レベル 5	5,554 Bytes	40 Bytes	964 Bytes

5.3.4 署名方式 MQOM

5.3.4.1 MQOM の概要

MQOM [22] は，5.2.5.1 節で説明した MQ 問題に関する秘匿マルチパーティ計算 f_{MQ} から MPC-in-the-Head で構成された署名方式である。5.2.5.2 節で述べたように，f_{MQ} はゼロ知識証明に変換することができる。さらに，Fiat-Shamir 変換により署名方式が構成できる。以下では，5.2.5.1 節の設定や記号を用いる。安全性パラメータを λ とし，以下の関数を用意する。

- Expand： 任意の λ-ビット列を入力とする疑似乱数生成関数
- $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3 : \{0,1\}^* \to \{0,1\}^{2\lambda}$ ：暗号学的ハッシュ関数
- Commit：コミットメント関数

鍵生成

1. $\mathrm{seed}_{\mathrm{pk}}$, $\mathrm{seed}_{\mathrm{sk}} \in \{0,1\}^\lambda$ をランダムに選ぶ。
2. $\mathrm{Expand}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{A}_i \in \mathbb{F}_q^{n\times n}$（上三角行列），$\mathbf{b}_i \in \mathbb{F}_q^n$ $(i = 1, \ldots, m)$ を得る。
3. $\mathrm{Expand}(\mathrm{seed}_{\mathrm{sk}})$ の計算により，$\mathbf{x}^* \in \mathbb{F}_q^n$ を得る。
4. $y_i = \mathbf{x}^* \mathrm{A}_i \mathbf{x}^{*\top} + \mathbf{x}^* \mathbf{b}_i^\top$ $(i = 1, \ldots, m)$ を計算する。

公開鍵は $\mathrm{pk} = (\mathrm{seed}_{\mathrm{pk}}, \mathbf{y} = (y_1, \ldots, y_m))$，秘密鍵は $\mathrm{sk} = \mathrm{seed}_{\mathrm{sk}}$ である。次に，署名生成である。メッセージを $M \in \{0,1\}^*$ とする。

署名生成

1. pk から $(\mathrm{seed}_{\mathrm{pk}}, \mathbf{y})$ を取り出す。
2. sk から $\mathrm{seed}_{\mathrm{sk}}$ を取り出す。

3. Expand($\mathrm{seed}_{\mathrm{pk}}$) の計算により，$\mathrm{A}_i \in \mathbb{F}_q^{n\times n}$（上三角行列），$\mathbf{b}_i \in \mathbb{F}_q^n$ $(i=1,\ldots,m)$ を得る。
4. salt $\in \{0,1\}^{2\lambda}$, mseed $\in \{0,1\}^{\lambda}$ をランダムに選ぶ。
5. Expand(salt, mseed) の計算により，$\mathrm{rseed}^{[e]} \in \{0,1\}^{\lambda}$ $(e=1,\ldots,\tau)$ を得る。
6. $e=1,\ldots,\tau$ に対して以下を行う：
   6-1. Expand(salt, $\mathrm{rseed}^{[e]}$) の計算により，$\mathrm{seed}_j^{[e]} \in \{0,1\}^{\lambda}$ $(j=1,\ldots,N)$ を得る。
   6-2. すべての $j=1,\ldots,N$ に対し，Expand(salt, $\mathrm{seed}_j^{[e]}$) の計算により以下を得る：
   · $j<N$ ならば，$[\![\mathbf{x}^{*[e]}]\!]_j$, $[\![a_1^{[e]}]\!]_j,\ldots,[\![a_{n_2}^{[e]}]\!]_j$, $[\![Q'^{[e]}]\!]_j(u)$
   · $j=N$ ならば，$[\![a_1^{[e]}]\!]_N,\ldots,[\![a_{n_2}^{[e]}]\!]_N$
   6-3. $[\![\mathbf{x}^{*[e]}]\!]_N = \mathbf{x}^* - \sum_{j=1}^{N-1}[\![\mathbf{x}^{*[e]}]\!]_j$ を計算する。
   6-4. コミットメントを計算する：

$$\mathrm{com}_j^{[e]} = \begin{cases} \mathrm{Commit}(\mathrm{salt},\, e,\, j,\, \mathrm{seed}_j^{[e]}) & j=1,\ldots,N-1 \\ \mathrm{Commit}(\mathrm{salt},\, e,\, N,\, \mathrm{seed}_N^{[e]},\, [\![\mathbf{x}^{*[e]}]\!]_N) & j=N \end{cases}$$

7. $h_1 = \mathcal{H}_1(M,\, \mathrm{salt},\, \mathrm{com}_1^{[1]},\ldots,\mathrm{com}_N^{[\tau]})$ を計算する。
8. Expand(h_1) の計算により，$\gamma_1^{[e]},\ldots,\gamma_m^{[e]}$ $(e=1,\ldots,\tau)$ を得る。
9. $e=1,\ldots,\tau$ に対して以下を行う：
   9-1. f_{MQ} のヒントオラクルと同じ計算により，$\mathbf{x}^*$, $\{\mathrm{A}_i,\mathbf{b}_i\}_{i=1,\ldots,m}$, $\gamma_1^{[e]},\ldots,\gamma_m^{[e]}$, $a_1^{[e]},\ldots,a_{n_2}^{[e]}$（但し，$a_i^{[e]} = \sum_{j=1}^{N}[\![a_i^{[e]}]\!]_j$）から，$Q'^{[e]}(u) \in \mathbb{F}_{q^\eta}[u]$ を計算する。
   9-2. $[\![Q'^{[e]}]\!]_N(u) = Q'^{[e]}(u) - \sum_{j=1}^{N-1}[\![Q'^{[e]}]\!]_j(u)$ を計算する。
   9-3. コミットメント $\mathrm{com}'^{[e]}_N = \mathrm{Commit}(\mathrm{salt},\, e\,,0\,,\, [\![Q'^{[e]}]\!]_N(u))$ を計算する。
10. $h_2 = \mathcal{H}_2(M,\, \mathrm{salt},\, h_1,\, \mathrm{com}'^{[1]}_N,\ldots,\mathrm{com}'^{[\tau]}_N)$ を計算する。
11. Expand(h_2) の計算により，$r^{[1]},\ldots,r^{[\tau]} \in \mathbb{F}_{q^\eta}$ を得る。
12. $e=1,\ldots,\tau$ に対して以下を行う：
   12-1. f_{MQ} における全パーティと同じ計算により，$[\![\mathbf{x}^{*[e]}]\!]$, $[\![a_1^{[e]}]\!],\ldots,[\![a_{n_2}^{[e]}]\!]$, $[\![Q'^{[e]}]\!](u)$ から，$[\![\mathrm{broad}^{[e]}]\!] = ([\![c_1^{[e]}]\!],\ldots,[\![c_{n_2}^{[e]}]\!],\, [\![v^{[e]}]\!])$ を計算する。
   12-2. $\mathrm{broad}^{[e]} = (c_1^{[e]},\ldots,c_{n_2}^{[e]},\, v^{[e]}) = \sum_{j=1}^{N}[\![\mathrm{broad}^{[e]}]\!]$ を計算する。
13. $h_3 = \mathcal{H}_3(M,\, \mathrm{salt},\, h_2,\, [\![\mathrm{broad}^{[1]}]\!],\ldots,[\![\mathrm{broad}^{[\tau]}]\!])$ を計算する。
14. Expand(h_3) の計算により，$i^{*[1]},\ldots,i^{*[\tau]} \in \{1,\ldots,N\}$ を得る。
15. $\mathrm{view}^{[e]}$ $(e=1,\ldots,\tau)$ を以下のようにおく：

$$\mathrm{view}^{[e]} = \begin{cases} (\{\mathrm{seed}_j^{[e]}\}_{j\in\{1,\ldots,N\}\setminus\{i^{*[e]}\}},\, [\![\mathbf{x}^{*[e]}]\!]_N,\, [\![Q'^{[e]}]\!]_N(u)) & (i^{*[e]}\neq N) \\ (\{\mathrm{seed}_j^{[e]}\}_{j\in\{1,\ldots,N\}\setminus\{i^{*[e]}\}}) & (i^{*[e]}=N) \end{cases}$$

16. $\sigma = (\mathrm{salt},\, h_1,\, h_2,\, h_3,\, \{\mathrm{view}^{[e]},\, \mathrm{broad}^{[e]},\, \mathrm{com}^{[e]}_{i^{*[e]}},\, \mathrm{com}'^{[e]}_N\}_{e=1,\ldots,\tau})$ とおく。

σ が署名となる。最後に検証である。

検証

1. pk から $(\mathrm{seed}_{\mathrm{pk}},\, \mathbf{y})$ を取り出す。
2. σ から $(\mathrm{salt},\, h_1,\, h_2,\, h_3,\, \{\mathrm{view}^{[e]},\, \mathrm{broad}^{[e]},\, \mathrm{com}^{[e]}_{i^{*[e]}},\, \mathrm{com}'^{[e]}_N\}_{e=1,\ldots,\tau})$ を取り出す。
3. Expand($\mathrm{seed}_{\mathrm{pk}}$) の計算により，$\mathrm{A}_i \in \mathbb{F}_q^{n\times n}$（上三角行列），$\mathbf{b}_i \in \mathbb{F}_q^n$ $(i=1,\ldots,m)$ を得る。
4. Expand(h_1) の計算により，$\gamma_1^{[e]},\ldots,\gamma_m^{[e]}$ $(e=1,\ldots,\tau)$ を得る。
5. Expand(h_2) の計算により，$r^{[1]},\ldots,r^{[\tau]} \in \mathbb{F}_{q^\eta}$ を得る。
6. Expand(h_3) の計算により，$i^{*[1]},\ldots,i^{*[\tau]} \in \{1,\ldots,N\}$ を得る。

7. $e = 1, \ldots, \tau$ に対して以下を行う：
   7-1. $\mathrm{broad}^{[e]}$ から $(c_1^{[e]}, \ldots, c_{n_2}^{[e]}, v^{[e]})$ を取り出す。
   7-2. $\mathrm{view}^{[e]}$ から以下を取り出す：
      · $i^{*[e]} \neq N$ ならば，$\{\mathrm{seed}_j^{[e]}\}_{j \in \{1,\ldots,N\} \setminus \{i^{*[e]}\}}$, $[\![\mathbf{x}^{*[e]}]\!]_N$, $[\![Q'^{[e]}]\!]_N(u)$
      · $i^{*[e]} = N$ ならば，$\{\mathrm{seed}_j^{[e]}\}_{j \in \{1,\ldots,N\} \setminus \{i^{*[e]}\}}$
   7-3. すべての $j \in \{1, \ldots, N\} \setminus \{i^{*[e]}\}$ に対し，$\mathrm{Expand}(\mathrm{salt}, \mathrm{seed}_j^{[e]})$ の計算により以下を得る：
      · $j < N$ ならば，$[\![\mathbf{x}^{*[e]}]\!]_j$, $[\![a_1^{[e]}]\!]_j, \ldots, [\![a_{n_2}^{[e]}]\!]_j$, $[\![Q'^{[e]}]\!]_j(u)$
      · $j = N$ ならば，$[\![a_1^{[e]}]\!]_N, \ldots, [\![a_{n_2}^{[e]}]\!]_N$
   7-4. すべての $j \in \{1, \ldots, N\} \setminus \{i^{*[e]}\}$ に対し，f_{MQ} における $\mathcal{P}_j$ と同じ計算より，$[\![\mathbf{x}^{*[e]}]\!]_j$, $[\![a_1^{[e]}]\!]_j, \ldots, [\![a_{n_2}^{[e]}]\!]_j$, $[\![Q'^{[e]}]\!]_j(u)$, $c_1^{[e]}, \ldots, c_{n_2}^{[e]}$ から，$[\![\mathrm{broad}^{[e]}]\!]_j = ([\![c_1^{[e]}]\!]_j, \ldots, [\![c_{n_2}^{[e]}]\!]_j, [\![v^{[e]}]\!]_j)$ を計算する。
   7-5. $[\![\mathrm{broad}^{[e]}]\!]_{i^{*[e]}} = \mathrm{broad}^{[e]} - \sum_{j \in \{1,\ldots,N\} \setminus \{i^{*[e]}\}} [\![\mathrm{broad}^{[e]}]\!]_j$ を計算する。
   7-6. すべての $j \in \{1, \ldots, N\} \setminus \{i^{*[e]}\}$ に対し，以下のようにコミットメントを計算する：
      · $j < N$ ならば，$\mathrm{com}_j^{[e]} = \mathrm{Commit}(\mathrm{salt}, e, j, \mathrm{seed}_j^{[e]})$
      · $j = N$ ならば，$\mathrm{com}_j^{[e]} = \mathrm{Commit}(\mathrm{salt}, e, N, \mathrm{seed}_N^{[e]}, [\![\mathbf{x}^{*[e]}]\!]_N)$
      $\mathrm{com}''^{[e]}_j = \mathrm{Commit}(\mathrm{salt}, e, 0, [\![Q'^{[e]}]\!]_N(u))$
8. $h'_1 = \mathcal{H}_1(M, \mathrm{salt}, \mathrm{com}_1^{[1]}, \ldots, \mathrm{com}_N^{[\tau]})$ を計算する。
9. $h'_2 = \mathcal{H}_2(M, \mathrm{salt}, h'_1, \mathrm{com}'^{[1]}_N, \ldots, \mathrm{com}'^{[\tau]}_N)$ を計算する。
10. $h'_3 = \mathcal{H}_3(M, \mathrm{salt}, h'_2, [\![\mathrm{broad}^{[1]}]\!], \ldots, [\![\mathrm{broad}^{[\tau]}]\!])$ を計算する。
11. $i^{*[e]} \neq N$ なる $e \in \{1, \ldots, \tau\}$ で，$\mathrm{com}''^{[e]}_N \neq \mathrm{com}'^{[e]}_N$ となるものがあれば，‘棄却’ を返す。
12. $v^{[e]} \neq 0$ なる $e \in \{1, \ldots, \tau\}$ があれば，‘棄却’ を返す。
13. $(h'_1, h'_2, h'_3) \neq (h_1, h_2, h_3)$ ならば，‘棄却’ を返す。それ以外は ‘受理’ を返す。

5.3.4.2 MQOM のパラメータ選択

MQOM の設計に必要なパラメータは，$\lambda, q, m, n, n_1, n_2, \eta, N, \tau$ である。NIST PQC 標準化プロジェクト追加署名第 1 ラウンドに提出されたドキュメント [22] では，さらに効率性向上のテクニック（hypercube optimization，seed tree など）が追加されており，それを踏まえて以下のように MQOM のパラメータの見積もりが公開されている。

表 5.6: MQOM のパラメータと鍵および署名のサイズ。署名中の view$^{[e]}$ に含まれる要素数が署名生成ごとに異なるため，署名サイズは平均値が与えられている。

$(\lambda, q, m(=n), n_1, n_2, \eta, N, \tau)$	安全性レベル	公開鍵サイズ	秘密鍵サイズ	署名サイズ（平均）
$(128,\ 31,\ 49, 5, 10, 10, 256, 20)$	レベル 1	47 Bytes	78 Bytes	6,348 Bytes
$(128,\ 31,\ 49, 5, 10,\ 6,\ 32, 35)$	レベル 1	59 Bytes	102 Bytes	6,575 Bytes
$(128, 251,\ 43, 4, 11,\ 5, 256, 22)$	レベル 1	47 Bytes	78 Bytes	7,621 Bytes
$(128, 251,\ 43, 4, 11,\ 4,\ 32, 34)$	レベル 1	59 Bytes	102 Bytes	7,809 Bytes
$(192,\ 31,\ 77, 6, 13, 11, 256, 30)$	レベル 3	73 Bytes	122 Bytes	13,837 Bytes
$(192,\ 31,\ 77, 6, 13,\ 7,\ 32, 51)$	レベル 3	92 Bytes	160 Bytes	14,257 Bytes
$(192, 251,\ 68, 5, 14,\ 7, 256, 30)$	レベル 3	73 Bytes	122 Bytes	16,590 Bytes
$(192, 251,\ 68, 5, 14,\ 4,\ 32, 52)$	レベル 3	92 Bytes	160 Bytes	17,161 Bytes
$(256,\ 31, 106, 6, 18, 10, 256, 42)$	レベル 5	99 Bytes	166 Bytes	24,147 Bytes
$(256,\ 31, 106, 6, 18,\ 8,\ 32, 66)$	レベル 5	125 Bytes	218 Bytes	24,926 Bytes
$(256, 251,\ 93, 6, 16,\ 7, 256, 41)$	レベル 5	99 Bytes	166 Bytes	28,917 Bytes
$(256, 251,\ 93, 6, 16,\ 5,\ 32, 66)$	レベル 5	125 Bytes	218 Bytes	29,919 Bytes

5.3.5 署名方式 MiRitH

5.3.5.1 MiRitH の概要

MiRitH [3] は，MinRank 問題に関する秘匿マルチパーティ計算から MPC-in-the-Head で構成された署名方式である。MinRank 問題は 5.1.3 節でも述べたが，次のように表現することもできる。

MinRank 問題（別バージョン） 正の整数 r と行列 $\mathrm{M}_0, \ldots, \mathrm{M}_k \in \mathbb{F}_q^{m\times n}$ に対し，$\alpha_1, \ldots, \alpha_k \in \mathbb{F}_q$ で，

$$\mathrm{Rank}\left(\mathrm{M}_0 + \sum_{i=1}^{k} \alpha_i \mathrm{M}_i\right) \leq r$$

なるものを求めよ。

もし，$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k) \in \mathbb{F}_q^k$ と $\mathrm{K} \in \mathbb{F}_q^{r\times(n-r)}$ が存在し，

$$\left(\mathrm{M}_0 + \sum_{i=1}^{k} \alpha_i \mathrm{M}_i\right) \cdot \begin{pmatrix} \mathbf{I}_{n-r} \\ \mathrm{K} \end{pmatrix} = \mathbf{0}_{m\times(n-r)} \tag{5.11}$$

となるならば，$\boldsymbol{\alpha}$ は MinRank 問題の解である。$\mathbf{M} = (\mathrm{M}_0, \ldots, \mathrm{M}_k)$ に対し，$\mathbf{M}_{\boldsymbol{\alpha}} \in \mathbb{F}_q^{m\times n}$ を

$$\mathbf{M}_{\boldsymbol{\alpha}} = \mathrm{M}_0 + \sum_{i=1}^{k} \alpha_i \mathrm{M}_i$$

とし，$\mathbf{M}_{\boldsymbol{\alpha}}^L \in \mathbb{F}_q^{m\times(n-r)}$, $\mathbf{M}_{\boldsymbol{\alpha}}^R \in \mathbb{F}_q^{m\times r}$ をそれぞれ，$\mathbf{M}_{\boldsymbol{\alpha}}$ の左側 $n-r$ 列，$\mathbf{M}_{\boldsymbol{\alpha}}$ の右側 r 列で定めると，(5.11) は $\mathbf{M}_{\boldsymbol{\alpha}}^L = \mathbf{M}_{\boldsymbol{\alpha}}^R \cdot \mathrm{K}$ と同値である。そこで，（秘匿マルチパーティ計算において設定される関係 $\mathcal{R}$ の）命題を MinRank 問題のインスタンスとし，その証拠を $\boldsymbol{\alpha}, \mathrm{K}$ としておけば，$\boldsymbol{\alpha}$ が MinRank 問題の解であることが（$\mathbf{M}_{\boldsymbol{\alpha}}$ のランクを直接計算をせずとも）効率的に検証できる。以下の検証プロトコルでは，ランダム行列 $\mathrm{R} \in \mathbb{F}_q^{s\times m}$ を用意

し，$\mathrm{V} = \mathrm{R}\,(\mathbf{M}_{\boldsymbol{\alpha}}^{L} - \mathbf{M}_{\boldsymbol{\alpha}}^{R} \cdot \mathrm{K})$ とおき，$\mathrm{V} = \mathbf{0}_{s\times(n-r)}$ となることで (5.11) を確認する。(5.11) が成り立つときは $\mathrm{V} = \mathbf{0}_{s\times(n-r)}$ が必ず成り立つが，(5.11) が成り立たないときに $\mathrm{V} = \mathbf{0}_{s\times(n-r)}$ となる確率は約 $1/q^s$ である。

安全性パラメータを λ とし，以下の関数を用意する。

- Expand： 任意の λ-ビット列を入力とする疑似乱数生成関数
- $\mathcal{H}_1, \mathcal{H}_2 : \{0,1\}^* \to \{0,1\}^{2\lambda}$ ：暗号学的ハッシュ関数
- Commit：コミットメント関数

鍵生成

1. $\mathrm{seed}_{\mathrm{pk}},\ \mathrm{seed}_{\mathrm{sk}} \in \{0,1\}^{\lambda}$ をランダムに選ぶ。
2. $\mathrm{Expand}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{M}_1, \ldots, \mathrm{M}_k \in \mathbb{F}_q^{m\times n}$ を得る。
3. $\mathrm{Expand}(\mathrm{seed}_{\mathrm{sk}})$ の計算により，$\alpha_1, \ldots, \alpha_k \in \mathbb{F}_q$, $\mathrm{K} \in \mathbb{F}_q^{r\times(n-r)}$, $\mathrm{E}^R \in \mathbb{F}_q^{m\times r}$ を得る。
4. $\mathrm{E}^R\,\mathrm{K}$ を計算し，左側の $n-r$ 列を $\mathrm{E}^R\,\mathrm{K}$，右側の r 列を E^R として定まる行列を $\mathrm{E} \in \mathbb{F}_q^{m\times n}$ とする。
5. $\mathrm{M}_0 = \mathrm{E} - \sum_{\ell=1}^{k} \alpha_\ell\,\mathrm{M}_\ell$ を計算する。

公開鍵は $\mathrm{pk} = (\mathrm{seed}_{\mathrm{pk}}, \mathrm{M}_0)$，秘密鍵は $\mathrm{sk} = \mathrm{seed}_{\mathrm{sk}}$ である。次に，署名生成である。メッセージを $M \in \{0,1\}^*$ とする。

署名生成

1. pk から $(\mathrm{seed}_{\mathrm{pk}}, \mathrm{M}_0)$ を取り出す。
2. sk から $\mathrm{seed}_{\mathrm{sk}}$ を取り出す。
3. $\mathrm{Expand}(\mathrm{seed}_{\mathrm{pk}})$ の計算により，$\mathrm{M}_1, \ldots, \mathrm{M}_k \in \mathbb{F}_q^{m\times n}$ を得る。
4. $\mathrm{salt} \in \{0,1\}^{2\lambda}$ をランダムに選ぶ。
5. $e = 1, \ldots, \tau$ に対して以下を計算する：
   - 5-1. $\mathrm{seed}^{[e]} \in \{0,1\}^{\lambda}$ をランダムに選ぶ。
   - 5-2. $\mathrm{Expand}(\mathrm{salt},\ \mathrm{seed}^{[e]})$ の計算により，$\mathrm{seed}_j^{[e]} \in \{0,1\}^{\lambda}\ \ (j = 1, \ldots, N)$ を得る。
   - 5-3. すべての $j = 1, \ldots, N$ に対し，$\mathrm{Expand}(\mathrm{salt}, \mathrm{seed}_j^{[e]})$ の計算により以下を得る：
     - $j < N$ ならば，$[\![\mathrm{A}^{[e]}]\!]_j \in \mathbb{F}_q^{s\times r}$, $[\![\alpha_1^{[e]}]\!]_j, \ldots, [\![\alpha_k^{[e]}]\!]_j\ (\in \mathbb{F}_q)$, $[\![\mathrm{K}^{[e]}]\!]_j \in \mathbb{F}_q^{r\times(n-r)}$, $[\![\mathrm{C}^{[e]}]\!]_j \in \mathrm{F}_q^{s\times(n-r)}$
     - $j = N$ ならば，$[\![\mathrm{A}^{[e]}]\!]_N \in \mathbb{F}_q^{s\times r}$
   - 5-4. $[\![\alpha_\ell^{[e]}]\!]_N = \alpha_\ell - \sum_{j=1}^{N-1} [\![\alpha_\ell^{[e]}]\!]_j\ \ (\ell = 1, \ldots, k)$ を計算する。
   - 5-5. $[\![\mathrm{K}^{[e]}]\!]_N = \mathrm{K} - \sum_{j=1}^{N-1} [\![\mathrm{K}^{[e]}]\!]_j$，$[\![\mathrm{C}^{[e]}]\!]_N = \mathrm{A}^{[e]}\,\mathrm{K} - \sum_{j=1}^{N-1} [\![\mathrm{C}^{[e]}]\!]_j$ を計算する。
   - 5-6. $\mathrm{state}_j^{[e]} = \begin{cases} (\mathrm{seed}_j^{[e]}) & (j = 1, \ldots, N-1) \\ (\mathrm{seed}_N^{[e]}, [\![\alpha_1^{[e]}]\!]_N, \ldots, [\![\alpha_k^{[e]}]\!]_N, [\![\mathrm{K}^{[e]}]\!]_N, [\![\mathrm{C}^{[e]}]\!]_N) & (j = N) \end{cases}$ とおく。
   - 5-7. コミットメント $\mathrm{com}_j^{[e]} = \mathrm{Commit}(\mathrm{salt}, e, j, \mathrm{state}_j^{[e]})\ \ (j = 1, \ldots, N)$ を計算する。
6. $h_1 = \mathcal{H}_1(M, \mathrm{salt}, \mathrm{com}_1^{[1]}, \ldots, \mathrm{com}_N^{[\tau]})$ を計算する。
7. $\mathrm{Expand}(h_1)$ の計算により，$\mathrm{R}^{[e]} \in \mathbb{F}_q^{s\times m}\ \ (e = 1, \ldots, \tau)$ を得る。
8. $e = 1, \ldots, \tau$ に対して以下を計算する：
   - 8-1. $[\![\alpha_1^{[e]}]\!], \ldots, [\![\alpha_k^{[e]}]\!]$ より，$[\![\mathbf{M}_{\boldsymbol{\alpha}}^{L\,[e]}]\!]$, $[\![\mathbf{M}_{\boldsymbol{\alpha}}^{R\,[e]}]\!]$ を計算する。
   - 8-2. $[\![\mathrm{S}^{[e]}]\!] = \mathrm{R}^{[e]}\,[\![\mathbf{M}_{\boldsymbol{\alpha}}^{R\,[e]}]\!] + [\![\mathrm{A}^{[e]}]\!]$ を計算する。
   - 8-3. $\mathrm{S}^{[e]} = \sum_{j=1}^{N} [\![\mathrm{S}^{[e]}]\!]_j$ を計算する。
   - 8-4. $[\![\mathrm{V}^{[e]}]\!] = \mathrm{S}^{[e]}\,[\![\mathrm{K}^{[e]}]\!] - \mathrm{R}^{[e]}\,[\![\mathbf{M}_{\boldsymbol{\alpha}}^{L\,[e]}]\!] - [\![\mathrm{C}^{[e]}]\!]$ を計算する。
   - 8-5. $\mathrm{V}^{[e]} = \sum_{j=1}^{N} [\![\mathrm{V}^{[e]}]\!]_j$ を計算する。
   - 8-6. $[\![\mathrm{broad}^{[e]}]\!] = ([\![\mathrm{S}^{[e]}]\!], [\![\mathrm{V}^{[e]}]\!])$, $\mathrm{broad}^{[e]} = (\mathrm{S}^{[e]}, \mathrm{V}^{[e]})$ とおく。

9. $h_2 = \mathcal{H}_2(M, \text{salt}, h_1, [\![\text{broad}^{[1]}]\!], \cdots, [\![\text{broad}^{[\tau]}]\!])$ を計算する。
10. $\text{Expand}(h_2)$ の計算により，$i^{*[1]}, \ldots, i^{*[\tau]} \in \{1, \ldots, N\}$ を得る。
11. $\text{view}^{[e]}$ をすべての $j \in \{1, \ldots, N\} \backslash \{i^{*[e]}\}$ に対する $\text{state}_j^{[e]}$ のリストとする。
12. $\sigma = (\text{salt}, h_1, h_2, \{\text{view}^{[e]}, \text{broad}^{[e]}, \text{com}_{i^{*[e]}}\}_{e=1,\ldots,\tau})$ とおく。

σ が署名となる。最後に検証である。

検証

1. pk から $(\text{seed}_{\text{pk}}, \text{M}_0)$ を取り出す。
2. σ から $(\text{salt}, h_1, h_2, \{\text{view}^{[e]}, \text{broad}^{[e]}, \text{com}_{i^{*[e]}}\}_{e=1,\ldots,\tau})$ を取り出す。
3. $\text{Expand}(\text{seed}_{\text{pk}})$ の計算により，$\text{M}_1, \ldots, \text{M}_k \in \mathbb{F}_q^{m \times n}$ を得る。
4. $\text{Expand}(h_1)$ の計算により，$\text{R}^{[e]} \in \mathbb{F}_q^{s \times m}$ $(e = 1, \ldots, \tau)$ を得る。
5. $\text{Expand}(h_2)$ の計算により，$i^{*[1]}, \ldots, i^{*[\tau]} \in \{1, \ldots, N\}$ を得る。
6. $e = 1, \ldots, \tau$ に対して以下を計算する：
   6-1. $\text{broad}^{[e]}$ から $(\text{S}^{[e]}, \text{V}^{[e]})$ を取り出す。
   6-2. $\text{view}^{[e]}$ から $(\text{state}_j^{[e]})_{j \in \{1,\ldots,N\} \backslash \{i^{*[e]}\}}$ を取り出す。
   6-3. コミットメント $\text{com}_j^{[e]} = \text{Commit}(\text{salt}, e, j, \text{state}_j^{[e]})$ $(j \in \{1, \ldots, N\} \backslash \{i^{*[e]}\})$ を計算する。
   6-4. すべての $j \in \{1, \ldots, N\} \backslash \{i^{*[e]}\}$ に対し，$\text{state}_j^{[e]}$ により以下を得る：
      · $j < N$ ならば，$\text{seed}_j^{[e]}$
      · $j = N$ ならば，$\text{seed}_N^{[e]}, [\![\alpha_1^{[e]}]\!]_N, \ldots, [\![\alpha_k^{[e]}]\!]_N, [\![\text{K}^{[e]}]\!]_N, [\![\text{C}^{[e]}]\!]_N$
   6-5. すべての $j \in \{1, \ldots, N\} \backslash \{i^{*[e]}\}$ に対し，$\text{Expand}(\text{salt}, \text{seed}_j^{[e]})$ の計算により以下を得る：
      · $j < N$ ならば，$[\![\text{A}^{[e]}]\!]_j, [\![\alpha_1^{[e]}]\!]_j, \ldots, [\![\alpha_k^{[e]}]\!]_j, [\![\text{K}^{[e]}]\!]_j, [\![\text{C}^{[e]}]\!]_j$
      · $j = N$ ならば，$[\![\text{A}^{[e]}]\!]_N$
   6-6. すべての $j \in \{1, \ldots, N\} \backslash \{i^{*[e]}\}$ に対し，$[\![\alpha_1^{[e]}]\!]_j, \ldots, [\![\alpha_k^{[e]}]\!]_j$ より，$[\![\mathbf{M}_{\boldsymbol{\alpha}}^{L\,[e]}]\!]_j, [\![\mathbf{M}_{\boldsymbol{\alpha}}^{R\,[e]}]\!]_j$ を計算する。
   6-7. $[\![\text{S}^{[e]}]\!]_j = \text{R}^{[e]} [\![\mathbf{M}_{\boldsymbol{\alpha}}^{R\,[e]}]\!]_j + [\![\text{A}^{[e]}]\!]_j$ $(j \in \{1, \ldots, N\} \backslash \{i^{*[e]}\})$ を計算する。
   6-8. $[\![\text{V}^{[e]}]\!]_j = \text{S}^{[e]} [\![\text{K}^{[e]}]\!]_j - \text{R}^{[e]} [\![\mathbf{M}_{\boldsymbol{\alpha}}^{L\,[e]}]\!]_j - [\![\text{C}^{[e]}]\!]_j$ $(j \in \{1, \ldots, N\} \backslash \{i^{*[e]}\})$ を計算する。
   6-9. $[\![\text{broad}^{[e]}]\!]_j = ([\![\text{S}^{[e]}]\!]_j, [\![\text{V}^{[e]}]\!]_j)$ $(j \in \{1, \ldots, N\} \backslash \{i^{*[e]}\})$ とおく。
   6-10. $[\![\text{broad}^{[e]}]\!]_{i^{*[e]}} = \text{broad}^{[e]} - \sum_{j \in \{1,\ldots,N\} \backslash \{i^{*[e]}\}} [\![\text{broad}^{[e]}]\!]_j$ を計算する。
7. $h_1' = \mathcal{H}_1(M, \text{salt}, \text{com}_1^{[1]}, \ldots, \text{com}_N^{[\tau]})$ を計算する。
8. $h_2' = \mathcal{H}_2(M, \text{salt}, h_1', [\![\text{broad}^{[1]}]\!], \cdots, [\![\text{broad}^{[\tau]}]\!])$ を計算する。
9. $\text{V}^{[e]} \neq \mathbf{0}_{s \times (n-r)}$ なる $e \in \{1, \ldots, \tau\}$ があれば，'棄却' を返す。
10. $(h_1', h_2') \neq (h_1, h_2)$ ならば，'棄却' を返す。それ以外は '受理' を返す。

5.3.5.2 MiRitH のパラメータ選択

MiRitH の設計に必要なパラメータは，$\lambda, q, m, n, k, r, s, N, \tau$ である。NIST PQC 標準化プロジェクト追加署名第 1 ラウンドに提出されたドキュメント [3] では，さらに効率性向上のテクニック（hypercube optimization，seed tree など）が追加されており，それを踏まえて以下のように MiRitH のパラメータの見積もりが公開されている。署名中の $\text{view}^{[e]}$ に含まれる要素数が署名生成ごとに異なるため，署名長は平均値が与えられている。

表 5.7: MiRitH のパラメータと鍵および署名のサイズ。署名中の view$^{[e]}$ に含まれる要素数が署名生成ごとに異なるため，署名サイズは平均値が与えられている。

$(\lambda, q, m(=n), k, r, s, N, \tau)$	安全性レベル	公開鍵サイズ	秘密鍵サイズ	署名サイズ（平均）
$(128, 16, 15, 78, 6, 5, 16, 39)$	レベル 1	129 Bytes	16 Bytes	7,661 Bytes
$(128, 16, 15, 78, 6, 9, 256, 19)$	レベル 1	129 Bytes	16 Bytes	5,665 Bytes
$(128, 16, 16, 142, 4, 5, 16, 39)$	レベル 1	144 Bytes	16 Bytes	8,800 Bytes
$(128, 16, 16, 142, 4, 9, 256, 19)$	レベル 1	144 Bytes	16 Bytes	6,298 Bytes
$(192, 16, 19, 109, 8, 7, 16, 55)$	レベル 3	205 Bytes	24 Bytes	16,668 Bytes
$(192, 16, 19, 109, 8, 9, 256, 29)$	レベル 3	205 Bytes	24 Bytes	12,423 Bytes
$(192, 16, 19, 167, 6, 7, 16, 55)$	レベル 3	205 Bytes	24 Bytes	17,882 Bytes
$(192, 16, 19, 167, 6, 9, 256, 29)$	レベル 3	205 Bytes	24 Bytes	13,115 Bytes
$(256, 16, 21, 189, 7, 7, 16, 74)$	レベル 5	253 Bytes	32 Bytes	29,568 Bytes
$(256, 16, 21, 189, 7, 10, 256, 38)$	レベル 5	253 Bytes	32 Bytes	21,763 Bytes
$(256, 16, 22, 254, 6, 7, 16, 74)$	レベル 5	274 Bytes	32 Bytes	31,980 Bytes
$(256, 16, 22, 254, 6, 10, 256, 38)$	レベル 5	274 Bytes	32 Bytes	23,144 Bytes

5.4 多変数多項式に基づく暗号技術に関するまとめ

1984 年に，Ong と Schnorr が多変数 2 次多項式を利用した署名方式 [32] を提案した。したがって，多変数多項式を利用した暗号技術は，既に 40 年以上の歴史を持つことになる。Ong と Schnorr の署名方式は，合成数を法とする剰余環を係数としており，合成数の素因数分解ができないことを安全性の仮定としていたが，1988 年に，松本と今井により，初めて有限体を係数とした多変数多項式を利用した公開鍵暗号方式 [31] が提案された[*2]。これ以降，MQ 問題の解読困難性を安全性の仮定とする方式が数多く提案されており，現在に至る。

多変数多項式に基づく公開鍵暗号方式，および，署名方式の多くは双極型システムを用いて構成されている。双極型システムは，暗号化や検証が効率的に実行できるという特徴を持つ。双極型システムを用いて構成されている署名方式 UOV も，この特徴を持ち，さらに，署名長が短いという特徴も持っている。一方で，双極型システムは公開鍵長が大きくなりやすいという課題もある。UOV に対しては，公開鍵長を削減する改良を加えた変種として QR-UOV や MAYO が提案されている。

一方で，MQ 問題や MinRank 問題に付随する秘匿マルチパーティ計算から MPC-in-the-Head の枠組みを利用して署名方式を構成することができる。こちらは方式が提案されてからまだ数年しかたっていないということもあり，今後の研究動向を見守る必要がある。

[*2] かつて国内では「多次多変数暗号」と呼ばれていた

第 5 章の参照文献

[1] W. W. Adams, P. Loustaunau. An Introduction to Gröbner Bases. Vol. 3. Graduate Studies in Mathematics. American Mathematical Society.

[2] G. Adj, L. Rivera-Zamarripa, J. A. Verbel. MinRank in the Head: Short Signatures from Zero-Knowledge Proofs. (2022). https://eprint.iacr.org/2022/1501.

[3] G. Adj, L. Rivera-Zamarripa, J. A. Verbel, E. Bellini, S. Barbero, A. Esser, C. Sanna, F. Zweydinger. MiRitH. 2022-08. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/submission-pkg/mirith-submission.zip. Submission to the NIST's Post-Quantum Cryptography: Additional Digital Signature Schemes, round 1.

[4] M. Bardet, M. Bros, D. Cabarcas, P. Gaborit, R. A. Perlner, D. Smith-Tone, J.-P. Tillich, J. A. Verbel. Improvements of Algebraic Attacks for Solving the Rank Decoding and MinRank Problems. ASIACRYPT (1). Vol. 12491. Lecture Notes in Computer Science. Springer, 2020, pp. 507–536.

[5] M. Bardet, J.-C. Faugère, B. Salvy, B.-Y. Yang. Asymptotic expansion of the degree of regularity for semi-regular systems of equations. Effective Methods in Algebraic Geometry (MEGA). 2004, pp. 71–74.

[6] E. Bellini, A. Esser, C. Sanna, J. Verbel. MR-DSS – Smaller MinRank-based (Ring-)Signatures. (2022). https://eprint.iacr.org/2022/973.

[7] W. Beullens. Improved cryptanalysis of UOV and Rainbow. Cryptology ePrint Archive, Paper 2020/1343. 2020. https://eprint.iacr.org/2020/1343.

[8] W. Beullens. MAYO: Practical Post-quantum Signatures from Oil-and-Vinegar Maps. SAC. Vol. 13203. Lecture Notes in Computer Science. Springer, 2021, pp. 355–376.

[9] W. Beullens, F. Campos, S. Celi, B. Hess, M. J. Kannwischer. MAYO. 2022-08. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/submission-pkg/mayo-submission.zip. Submission to the NIST's Post-Quantum Cryptography: Additional Digital Signature Schemes, round 1.

[10] W. Beullens, F. Campos, S. Celi, B. Hess, M. J. Kannwischer. MAYO Round 2 Version. 2025-02. https://pqmayo.org/assets/specs/mayo-round2.pdf.

[11] W. Beullens et al. UOV. 2022-08. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/submission-pkg/UOV-submission.zip. Submission to the NIST's Post-Quantum Cryptography: Additional Digital Signature Schemes, round 1.

[12] Ward Beullens. Breaking Rainbow Takes a Weekend on a Laptop. CRYPTO (2). Vol. 13508. Lecture Notes in Computer Science. Springer, 2022, pp. 464–479.

[13] A. Caminata, E. Gorla. Solving degree, last fall degree, and related invariants. Journal of Symbolic Computation. Vol. 114 (2023), pp. 322–335.

[14] A. Casanova, J.-C. Faugere, G. Macario-Rat, J. Patarin, L. Perret, J. Ryckeghem. GeMSS. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-2/submissions/GeMSS-Round2.zip. Submission to the NIST's Post-Quantum Cryptography, round 2.

[15] N. T. Courtois. Efficient Zero-Knowledge Authentication Based on a Linear Algebra Problem MinRank. ASIACRYPT. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 402–421.

[16] J. Ding, J. E. Gower, D. S. Schmidt. Multivariate Public Key Cryptosystems. Vol. 25. Advances in Information Security. Springer, 2006.

[17] J. Ding, D. Schmidt. Rainbow, a New Multivariable Polynomial Signature Scheme. ACNS. Vol. 3531. Lecture Notes in Computer Science. 2005, pp. 164–175.

[18] V. Dubois, N. Gama. The Degree of Regularity of HFE Systems. ASIACRYPT. Vol. 6477. Lecture Notes in Computer Science. Springer, 2010, pp. 557–576.

[19] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases (F4). J. Pure Appl. Algebra. Vol. 139 (1999), pp. 61–88.

[20] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). ISSAC. ACM, 2002, pp. 75–83.

[21] J.-C. Faugère, M. S. El Din, P.-J. Spaenlehauer. Computing loci of rank defects of linear matrices using Gröbner bases and applications to cryptology. ISSAC. ACM, 2010, pp. 257–264.

[22] T. Feneuil, M. Rivain. MQOM (MQ on my mind). 2022-08. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/submission-pkg/mqom-submission.zip. Submission to the NIST's Post-Quantum Cryptography: Additional Digital Signature Schemes, round 1.

[23] Fukuoka MQ Challenge. https://www.mqchallenge.org/. (2025-03-04 閲覧).

[24] H. Furue, Y. Ikematsu, Y. Kiyomura, T. Takagi. A New Variant of Unbalanced Oil and Vinegar Using Quotient Ring: QR-UOV. ASIACRYPT (4). Vol. 13093. Lecture Notes in Computer Science. Springer, 2021, pp. 187–217.

[25] H. Furue, Y. Ikematsu, Y. Kiyomura, T. Takagi, K. Yasuda, T. Miyazawa, T. Saito, A. Nagai. QR-UOV. 2022-08. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/submission-pkg/QR-UOV-submission.zip. Submission to the NIST's Post-Quantum Cryptography: Additional Digital Signature Schemes, round 1.

[26] M. R. Garay, D. S. Johnson. A Guide to the Theory of NP-Completeness. In Computers and Intractability. W.H. Freeman, 1979.

[27] L. Goubin, N.T. Courtois. Cryptanalysis of the TTM Cryptosystem. ASIACRYPT. Vol. 1976. Lecture Notes in Computer Science. Springer, 2000, pp. 44–57.

[28] Y. Ishai, E. Kushilevitz, R. Ostrovsky, A. Sahai. Zero-knowledge from secure multiparty computation. STOC. ACM, 2007, pp. 21–30.

[29] A. Kipnis, L. Patarin, L. Goubin. Unbalanced Oil and Vinegar Signature Schemes. EUROCRYPT. Vol. 1592. Lecture Notes in Computer Science. Springer, 1999, pp. 206–222.

[30] A. Kipnis, A. Shamir. Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization. CRYPTO. Vol. 1666. Lecture Notes in Computer Science. Springer, 1999, pp. 19–30.

[31] T. Matsumoto, H. Imai. Public Quadratic Polynominal-Tuples for Efficient Signature-Verification and Message-Encryption. EUROCRYPT. Vol. 330. Lecture Notes in Computer Science. Springer, 1988, pp. 419–453.

[32] H. Ong, C.-P. Schnorr. Signatures through Approximate Representation by Quadratic Forms. CRYPTO. Plenum Press, New York, 1983, pp. 117–131.

[33] J. Patarin. Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. EUROCRYPT. Vol. 1070. Lecture Notes in Computer Science. Springer, 1996, pp. 33–48.

[34] A. Petzoldt, M.-S. Chen, B.-Y. Yang, C. Tao, J. Ding. Design Principles for HFEv-Based Multivariate Signature Schemes. ASIACRYPT (1). Vol. 9452. Lecture Notes in Computer Science. Springer, 2015, pp. 311–334.

[35] B. Santoso, Y. Ikematsu, S. Nakamura, T. Yasuda. Three-Pass Identification Scheme Based on MinRank Problem with Half Cheating Probability. arXiv: 2205.03255.

[36] C. Tao, A. Petzoldt, J. Ding. Efficient Key Recovery for All HFE Signature Variants. CRYPTO (1). Vol. 12825. Lecture Notes in Computer Science. Springer, 2021, pp. 70–93.

[37] C. Wolf. Taxonomy of public key schemes based on the problem of Multivariate Quadratic equations. Cryptology ePrint Archive, Paper 2005/077. 2005. https://eprint.iacr.org/2005/077.

[38] B.-Y. Yang, J.-M. Chen. All in the XL Family: Theory and Practice. ICISC. Vol. 3506. Lecture Notes in Computer Science. Springer, 2004, pp. 67–86.

第 6 章

同種写像に基づく暗号技術

本章では同種写像に基づく暗号技術についてまとめる。同種写像に基づく暗号技術の安全性は，同種写像問題を解く計算の困難性及び（それと同値な）自己準同型環計算問題の困難性に依存しており，同種写像暗号に関する研究はこれまで継続して進められている。特に，本報告書の 2022 年度版 [1] と比べて，高次元同種写像計算を利用した鍵共有・署名構成に進展が見られている（6.2.2.2 節，6.3.1.2 節及び 6.4 節参照）。

6.1 節では, 安全性の根拠となる問題として, 同種写像問題の一般形を述べた後, 最近発見された SIDH（Supersingular Isogeny Diffie–Hellman）同種写像問題 [59] に対する解法について述べる。そして，その攻撃法を回避する計算問題として，レベル構造付き同種写像問題，同種写像に基づく一方向性群作用に関する計算問題，自己準同型環計算問題及び SQIsign（Short Quaternion and Isogeny Signature）署名方式 [61] の安全性に関する計算問題の順に，その概要を記述していく。6.2 節では，代表的な暗号方式として，一方向性群作用に基づく CSIDH （Commutative Supersingular Isogeny Diffie–Hellman）鍵共有 [29] とその変種，SIDH 型の鍵共有方式，CSIDH ベースの SeaSign 署名方式 [58], CSI-FiSh（Commutative Supersingular Isogeny Fiat–Shamir）署名方式 [16]，そして自己準同型環計算問題に基づく GPS（Galbraith–Petit–Silva）署名方式 [72] を取り上げる。6.3 節では，主要な暗号方式として，GPS 署名方式を改良した SQIsign 署名方式を解説する。

本章では，超特異楕円曲線を用いた暗号技術を主に扱う。しかし，通常楕円曲線に基づく CRS （Couveignes–Rostovtsev–Stolbunov）鍵共有法 [42, 109] を改良した De Feo らの方式 [60] は，それ自体は実用的な性能にはまだ遠いが，6.2.1.1 節で説明する CSIDH 鍵共有の原型を与えているという点で重要である。また，群作用暗号を量子マネーへ応用した [120, 90] では，通常楕円曲線が用いられている。

同種写像の数学的詳細については，De Feo の概説記事 [55] や Washington の楕円曲線の教科書 [118] を参照のこと。和書では，相川らによる概説書 [121] において，同種写像暗号に必要な数学も詳しく説明されている。また，Galbraith–Vercauteren による同種写像関連問題のサーベイ [73] も参照する。

■記法　$x \leftarrow_R X$ は，x を有限集合 X から一様ランダムにサンプリングすることを表す。以下では，有限体上に定義された楕円曲線のみを扱い，同種写像暗号では，多くの場合，モンゴメリ型の楕円曲線定義式 $E_{a,b} : by^2 = x^3 + ax^2 + x$ が用いられる。標数 p の有限体 $\mathbb{F}$ 上定義された楕円曲線 E に対し，O_E は E の無限遠点であり，$\mathbb{F}$ の拡大体 $\mathbb{K}$ に対して，$\mathbb{K}$-有理点群は $E(\mathbb{K}) := \{(x, y) \in \mathbb{K}^2 \,|\, (x, y) \text{ は } E \text{ の定義式を満たす}\} \cup \{O_E\}$ で与えられる。また，正整数 r に対して E の r-ねじれ部分群は $E[r] := \{P \in E(\overline{\mathbb{F}}_p) \,|\, rP = O_E\}$ で与えられる。ここで $\overline{\mathbb{F}}_p$ は有限体 $\mathbb{F}_p$ の代数閉包を表す。

6.1 同種写像に基づく暗号技術の安全性の根拠となる問題

6.1.1 節で同種写像問題の一般形を述べた後，6.1.2 節で 2022 年に発見された SIDH 同種写像問題に対する解法について述べ，そして，その攻撃法を回避する計算問題として，レベル構造付き同種写像問題（6.1.3 節），同種写像に基づく一方向性群作用に関する計算問題（6.1.4 節），自己準同型環計算問題と SQIsign 署名方式の安全性に関する計算問題（6.1.5 節）の順に，その概要及びそれら問題に対する解析状況について記述していく。

レベル構造付き同種写像問題に基づく QFESTA 暗号方式・鍵共有 [96] は，現時点で効率面で最も良い同種写像ベース鍵共有法を実現しており，6.2.2.2 節で紹介される。また，同種写像ベースの一方向性群作用に関する計算問題は，様々な暗号応用に対して有用であり，6.2.1 節及び 6.2.3 節でそれぞれ基本的な CSIDH 鍵共有と CSI-FiSh 署名方式が示される。そして，自己準同型環計算問題と関連する計算問題は，NIST PQC 標準化プロジェクト追加署名第 1 ラウンドを通過した SQIsign 署名方式（6.3.1 節）を基礎付けるために重要である。

つまり，6.1.3 節から 6.1.5 節までで示される 3 種の同種写像問題群はそれぞれに重要な応用先を有しており，それらの安全性に関して検討することが同種写像暗号を評価する第一歩である。

6.1.1 同種写像問題の一般形

同種写像とは，2 つの楕円曲線 E, E' の間の写像 φ であり，E の座標 (x, y) の有理式で与えられると共に，楕円曲線の加法構造に関する準同型性，即ち $\varphi(P+Q) = \varphi(P) + \varphi(Q)$，を有する非零写像である。（その正確な定義は，前掲の各文献を参照のこと。）また，E, E' の間に，同種写像 φ が存在する時に，E と E' は同種であるという。

同種写像 φ は，その核 $C = \ker(\varphi)$ によって決まるので，φ の定義域曲線（始点曲線）E に対して φ の値域となる楕円曲線を E/C と書き表す，すなわち，$\varphi : E \to E/C$。核 $C = \ker(\varphi)$ の位数がセキュリティパラメータ λ の多項式サイズであれば，$C = \ker(\varphi)$ の生成元から φ を効率的に計算するアルゴリズムが Vélu によって与えられている [116]。（モンゴメリ型楕円曲線に対する Vélu の公式に関しては，[105] を参照のこと。） 特に核の位数 $\#C$ が小素数になる同種写像の計算を同種写像基本演算として，それらの合成が同種写像暗号での基本的な暗号演算を与えることになる。そして，その合成における基本演算の組み合わせ方法が，秘密鍵情報を与える。

つまり，同種な楕円曲線の間の同種写像を計算することを要求する次の同種写像問題が，具体的な暗号方式の安全性を根拠づける次節以降の諸問題の基本形となる。（超特異同種写像問題と自己準同型環計算問題との計算量的同値性に関しては 6.1.5 節で触れる。）

定義 6.1 (一般形同種写像問題 [73]) *2* つの同種な楕円曲線 E, E' に対して，同種写像 φ を計算せよ。（φ のコンパクトな表現を与えよ。）

ここで，「φ のコンパクトな表現」とは，様々な表現方法が考えられる。例えば，$\deg(\varphi)$ が小素数 ℓ_i によって $\prod_i \ell_i^{e_i}$ となっている場合には，この分解に沿って φ を分解した各 ℓ_i 次同種写像の像に現れる値域楕円曲線（又は j 不変量）の列挙で与えることができる。また，6.1.2 節にて後述する SIDH 同種写像問題（定義 6.2）の設定では，核の生成点が，同種写像のコンパクトな表現を与える。そして，6.2.1.1 節で与えられる CSIDH 鍵共有では虚 2 次整環（オーダー）のイデアル類によって同種写像が表現される。6.1.2 節で述べられる SIDH 鍵共有に対する攻撃法は，楕円曲線同種写像に対する新しい表現法を与えた [107]。最近の高次元同種写像を用いた同種写像暗号の進展は，この新しい同種写像表現法に基づいて行われている。SIDH 同種写像問題，6.1.4 節にて後述する CSIDH-(R)EGA-DL 問題（定義 6.5，6.7）は，定義 6.1 の同種写像問題に基づいて定義される。

定義 6.1 において，φ の次数が多項式サイズであれば，この問題は簡単に解けるので，φ の次数は指数サイズとする。また，CSIDH 鍵共有では $\mathbb{F}_p$-有理な楕円曲線のみを対象とするので，$\overline{\mathbb{F}}_p$-同型であるが $\mathbb{F}_p$-同型でないツイスト曲線を

判別する必要性がある。しかしながら，ツイスト曲線は j 不変量では判別できない。この理由から，Galbraith ら [73] は j 不変量を使って同種写像問題を定式化しているが，上ではあえて，より素朴な形を採用して，2 つの同種な楕円曲線 E, E' を使って同種写像問題を提示した。

同種写像問題の初期の考察には，自己準同型環計算を扱った Kohel の博士論文 [80] や Galbraith による同種写像問題に関する研究 [69] 及び Couveignes と Rostovtsev–Stolbunov による初期の暗号応用への提案 [42, 109] がある。その後，Charles らによる同種写像に基づいたハッシュ関数の提案 [33] は，同種写像一方向性関数を一方向性の観点からだけでなく，衝突困難性の観点からも見直すことになり，初期の同種写像暗号の研究では重要な役割を果たした。特に，同種写像グラフがエクスパンダーグラフであることに着目して暗号に応用した意義は大きい。

■超特異同種写像問題と通常同種写像問題　標数 p の有限体上の楕円曲線 E の p-ねじれ部分群 $E[p]$ が，$E[p] = \{O_E\}$ の時，E を超特異楕円曲線といい，そうでない時，E を通常楕円曲線という。超特異楕円曲線の j 不変量は，$\mathbb{F}_{p^2}$ の要素である。つまり，超特異 j 不変量の個数は，有限個であり，具体的に $\lfloor p/12 \rfloor + \epsilon$（但し $\epsilon \in \{0, 1, 2\}$）で与えられる。超特異，通常という楕円曲線の性質は，同種写像によって保存されるため，同種写像問題も，この 2 つの性質によって，超特異同種写像問題と通常同種写像問題という 2 つの問題に分類される。

■超特異同種写像問題の計算困難性　超特異同種写像問題は，6.1.4.2 節で述べられる CSIDH 鍵共有や CSI-FiSh 署名方式の安全性に関する計算問題の一般形であり，その計算困難性を評価することは重要である。また，自己準同型環計算問題との関係性については 6.1.5 節を参照のこと。

超特異同種写像問題の古典計算機による解読時間は $\tilde{O}(\sqrt{p})$，量子計算機による解読時間は $\tilde{O}(\sqrt[4]{p})$ と見積もられている。Kohel [80] による超特異同種写像グラフ上のアルゴリズム解析に基づいて，現在最良の古典解読アルゴリズムは Delfs–Galbraith [49] によるもの及びその改良 [111] で，解読時間は $\tilde{O}(\sqrt{p})$ である。Delfs–Galbraith アルゴリズムでは $\mathbb{F}_p$ 上の超特異楕円曲線からなる部分グラフが利用されている。量子解読アルゴリズムは Biasse ら [17] によって時間計算量が $\tilde{O}(\sqrt[4]{p})$ の量子アルゴリズムが知られている。これは，$\mathbb{F}_p$ 上の超特異楕円曲線の同種写像問題に対する準指数時間量子アルゴリズム [37] と Grover アルゴリズムに基づく $\tilde{O}(\sqrt[4]{p})$ の道探索アルゴリズムを結合したものである。

また，Costello ら [41]，Longa ら [86] による報告，Udovenko–Vitto [115] による$IKEp182 Challenge [40] 解読報告，Jaques–Schanck [76] による同種写像問題に対する（量子）安全性評価報告は，いずれも SIDH 鍵共有（及び SIKE 暗号方式 [75]）法への攻撃として提案されているが，多くの部分は一般的な超特異同種写像問題に関する知見としても有効であることに注意する。更に，固定次数の同種写像を計算する問題に対して，CRYPTO 2024 において Benčina ら [11] により改善された古典/量子アルゴリズムが提案されている。

6.1.2　SIDH 同種写像問題とその解法

SIDH 鍵共有（及び SIKE 暗号方式）は，これまで同種写像暗号の中核方式と位置づけられてきたが，SIDH 同種写像問題（定義 6.2）に対して，2022 年に Castryck–Decru [23] に始まる一連の鍵導出解法が発表されて，暗号として完全に破れてしまった。しかし，これらの解法は，SIDH 同種写像問題という補助点の情報を入力に含む問題に対する解法であって，一般の同種写像問題（定義 6.1）には適用できないことに注意する。B-SIDH 鍵共有 [39]，Séta 暗号方式 [63] も本攻撃法により同様に解読可能である。SIDH 鍵共有に対して修正を図ろうとする試みについては，6.2.2 節で述べる。

■SIDH 同種写像問題　SIDH 鍵共有の公開パラメータは $pp_{\mathtt{sidh}} := (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}}, e_{\mathtt{B}}, f; E, P_{\mathtt{A}}, Q_{\mathtt{A}}, P_{\mathtt{B}}, Q_{\mathtt{B}})$ で与えられる。ここで，$p + 1 = f \cdot \ell_{\mathtt{A}}^{e_{\mathtt{A}}} \cdot \ell_{\mathtt{B}}^{e_{\mathtt{B}}}$ で，p は素数，$\ell_{\mathtt{A}}, \ell_{\mathtt{B}}$ は 2 つの異なる小素数である（f は小さい正整数で，多くの場合 $f = 1$）。例えば，$\ell_{\mathtt{A}} = 2, \ell_{\mathtt{B}} = 3$。$E$ は，$\mathbb{F}_{p^2}$ 上定義された超特異楕円曲線であり，$P_{\mathtt{A}}, Q_{\mathtt{A}}$ は，$E[\ell_{\mathtt{A}}^{e_{\mathtt{A}}}]$ の基底，$P_{\mathtt{B}}, Q_{\mathtt{B}}$ は，$E[\ell_{\mathtt{B}}^{e_{\mathtt{B}}}]$ の基底である。

定義 6.2 (SIDH 同種写像問題 [59, 73]) *SIDH* 鍵共有公開パラメータ $pp_{\mathtt{sidh}}$ と，そこで定義された E と $\ell_{\mathtt{B}}^{e_{\mathtt{B}}}$-同種な $E_{\mathtt{B}}$ と $P'_{\mathtt{A}}, Q'_{\mathtt{A}} \in E_{\mathtt{B}}[\ell_{\mathtt{A}}^{e_{\mathtt{A}}}]$ が与えられた時，$P'_{\mathtt{A}} = \varphi_{\mathtt{B}}(P_{\mathtt{A}}), Q'_{\mathtt{A}} = \varphi_{\mathtt{B}}(Q_{\mathtt{A}})$ となる次数 $\ell_{\mathtt{B}}^{e_{\mathtt{B}}}$ の同種写像 $\varphi_{\mathtt{B}} : E \to E_{\mathtt{B}}$ を計算せよ。特に，$\varphi_{\mathtt{B}}$ の核 $\ker(\varphi_{\mathtt{B}})$ の生成元 $R_{\mathtt{B}} \in E[\ell_{\mathtt{B}}^{e_{\mathtt{B}}}]$ を計算せよ。

以下，SIDH 同種写像問題に対して，Castryck–Decru による鍵導出攻撃 [23]，Maino らによる始点曲線に依らない攻撃法 [87]，Robert による SIDH 問題に対する多項式時間攻撃 [106] の順に概説する。

■Castryck–Decru による鍵導出攻撃 [23]　始点曲線 E が特殊極値整環 $\mathcal{O}_0$ を自己準同型環にもつ場合に主に限られるが[*1]，アーベル曲面間の同種写像が分解するかどうかという事実を使って SIDH 問題にアプローチした最初の論文である。特筆すべきは，SIKE Challenge [40] に挙げられていたパラメータ $IKE217 や SIKE パラメータ SIKEp434, SIKEp503，SIKEp610，SIKEp751 をいずれも現実的な時間内で解読することに成功したことである。後続の実装報告 [99] では，NIST 安全性レベル 5 に相当するとされていたパラメータ SIKEp751 が通常の PC（Intel Core i7-9750H CPU）で 1-2 時間程度で解けることが報告されている。

以下に示す Kani の補題が鍵となっている： 秘密同種写像の次数 $N_{\mathtt{B}} := \ell_{\mathtt{B}}^{e_{\mathtt{B}}}$ と互いに素な次数 a の同種写像 $\alpha : E \to E'$ が与えられれば，α と SIDH 同種写像問題の入力から楕円曲線積の間の $(a + N_{\mathtt{B}}, a + N_{\mathtt{B}})$-同種写像 $F : E \times E'' \to E_{\mathtt{B}} \times E'$ が構成可能になる。ここで Castryck–Decru [23] は，$N_{\mathtt{A}} := \ell_{\mathtt{A}}^{e_{\mathtt{A}}}, \ell_{\mathtt{A}} = 2, \ell_{\mathtt{B}} = 3$ として $N_{\mathtt{A}} > N_{\mathtt{B}}$ の下で，$a := N_{\mathtt{A}} - N_{\mathtt{B}}$ が $a = a_1^2 + 4a_2^2$ と 2 整数 a_1, a_2 による表現をもつ場合に有効な攻撃法を始点曲線 E が特殊極値的な場合に示した（特殊極値的楕円曲線に関しては 6.1.5.2 節を参照）。更に，その攻撃では，$a + N_{\mathtt{B}} = (N_{\mathtt{A}} - N_{\mathtt{B}}) + N_{\mathtt{B}} = N_{\mathtt{A}} = 2^{e_{\mathtt{A}}}$ であるので，F は $(2^{e_{\mathtt{A}}}, 2^{e_{\mathtt{A}}})$-同種写像，つまり Richelot 同種写像の列となっている。Richelot 同種写像列の計算や Richelot 同種写像のなすグラフに関する研究はこれまでも種数 2 同種写像暗号と関連して行われてきており [114, 79, 25, 64, 46, 43]，それらの研究と関連した形で Castryck–Decru 攻撃法が実現されている。

■Maino らによる始点曲線に依らない攻撃法 [87]　Maino らによる論文 [87] では，始点曲線に依らない攻撃法を実現するために，Kani の補題で必要な次数 a が平滑（smooth）になる場合の解析を進めて，De Feo の寄与も取り込んで準指数時間攻撃法を実現した。更に，Castryck–Decru 法では秘密同種写像 $\varphi_{\mathtt{B}}$ を徐々に部分的な同種写像を決定していく方法であったが，Maino ら [87] は，より直接的に 1 度の Kani の補題適用により $\varphi_{\mathtt{B}}$ を見つけ出すアルゴリズムに改良した。

■Robert による SIDH 問題に対する多項式時間攻撃 [106]　Robert [106] では，楕円曲線 8 つの直積の間の同種写像 F を上述の Kani の補題の一般化より得て，その分解性を使って多項式時間が証明可能な SIDH 攻撃アルゴリズムを構成した。さらに，[106] では，Castryck–Decru 法，Maino らによる方法を「$2g$ 次元攻撃法」という形で統合した。それによりそれぞれの方法の利点と課題が分かりやすい形で把握できるようになった。

例えば，[106] の主結果は，上述のように 8 次元アーベル多様体間の同種写像計算に基づいた証明可能多項式時間「8 次元攻撃法」アルゴリズムの提示にあるが，一方，ヒューリスティック多項式時間の「4 次元攻撃法」アルゴリズム [106, 系 4.5, 命題 4.6] についても詳細な解析を与えており，実装可能性という点から貴重な考察が含まれている。そして，2024 年に Dartois [43] により 4 次元攻撃法の効率的な実装結果が報告された。

また，Robert の論文 [106] においては，Petit [102] から始まり de Quehen ら [103] により発展させられた SIDH に対する「ねじれ点攻撃（torsion point attack）」との関連性もまとめられており，さらに広い見地から補助点情報を使った攻撃法の全体像も概観できるようになっている。

[*1] 特殊極値整環 $\mathcal{O}_0$ に関しては 6.1.5.2 節を参照のこと。

6.1.3 レベル構造付き同種写像問題

定義 6.2 の SIDH 同種写像問題では，$P'_{\mathtt{A}} = \varphi_{\mathtt{B}}(P_{\mathtt{A}}), Q'_{\mathtt{A}} = \varphi_{\mathtt{B}}(Q_{\mathtt{A}})$ となる $E[\ell_{\mathtt{A}}^{e_{\mathtt{A}}}]$ の基底 $P_{\mathtt{A}}, Q_{\mathtt{A}}$ および $E_{\mathtt{B}}[\ell_{\mathtt{A}}^{e_{\mathtt{A}}}]$ の基底 $P'_{\mathtt{A}}, Q'_{\mathtt{A}}$ が補助点情報として与えられたことにより多項式時間攻撃を受けた。それを回避するために M-SIDH 鍵共有や FESTA 鍵共有が提案されたが，De Feo ら [57] によって，それらの鍵共有の安全性を保証する問題として，SIDH 同種写像問題を一般化したレベル構造付き同種写像問題（Γ-SIDH 問題）が提案された。

まず，レベル構造を定義する。E が定義されている有限体の標数を p として，N を p と互いに素な正整数とする。$E[N]$ を生成する 2 点 (P, Q) を $E[N]$ の基底と呼び，$E[N]$ の全ての基底からなる集合を $\mathcal{B}_E[N]$ で表す。2×2 の可逆行列全体 $\mathrm{GL}_2(\mathbb{Z}/N\mathbb{Z})$ の $\mathcal{B}_E[N]$ への作用を

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot (P, Q) = (aP + bQ, cP + dQ)$$

で定める。$\mathrm{GL}_2(\mathbb{Z}/N\mathbb{Z})$ の部分群 Γ に対して，$\Gamma \cdot (P, Q) := \{\gamma \cdot (P, Q) \,|\, \gamma \in \Gamma\}$ を Γ-軌道とすると基底集合 $\mathcal{B}_E[N]$ は Γ-軌道によって分割される。Γ-軌道を（レベル N の）Γ-レベル構造と呼び，全ての Γ-レベル構造からなる集合を $\mathcal{B}_E[\Gamma]$ と表す。すなわち，基底集合 $\mathcal{B}_E[N]$ は $\mathcal{B}_E[N] = \bigcup_{S \in \mathcal{B}_E[\Gamma]} S$ と Γ-レベル構造 $S \in \mathcal{B}_E[\Gamma]$ によって交わりなく分割される。

定義 6.3 ((d,Γ)-SIDH 問題，Γ-SIDH 問題 [57]) 楕円曲線 E と d 次の同種写像 ϕ によって d-同種な $E'(= \phi(E))$，Γ-レベル構造 $S \in \mathcal{B}_E[\Gamma]$ に対して，$(E, S, E', \phi(S))$ が与えられて ϕ を計算せよ（(d,Γ)-*SIDH* 問題）。次数 d が文脈から明らかな場合は，単に Γ-*SIDH* 問題と呼ぶ。

多くの暗号応用においては [57, 補題 6] に示されるように Γ を特殊線型群 $\mathrm{SL}_2(\mathbb{Z}/N\mathbb{Z})$ の部分群としてよいので，以下でも $\Gamma \subseteq \mathrm{SL}_2(\mathbb{Z}/N\mathbb{Z})$ とする。特に，Γ-SIDH 問題は，$\Gamma = \mathrm{I} = \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right\}$ の時には，定義 6.2 の SIDH 同種写像問題と等しくなり多項式時間解法を有する一方，$\Gamma = \mathrm{SL}_2(\mathbb{Z}/N\mathbb{Z})$ の時には，一般の同種写像問題と等しくなり現状では効率的な解法が知られていない。よって，$\mathrm{I} \subsetneq \Gamma \subsetneq \mathrm{SL}_2(\mathbb{Z}/\mathrm{N}\mathbb{Z})$ となる Γ によって暗号構成に有用な Γ-SIDH 問題を見つけることが重要な課題である。

そのような Γ の例として，$\Gamma_{\mathrm{M}} := \left\{ \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix} \right\} (\subsetneq \mathrm{SL}_2(\mathbb{Z}/N\mathbb{Z}))$ に対する Γ_{M}-SIDH 問題の困難性が M-SIDH 鍵共有の安全性の根拠となるので，Γ_{M}-SIDH 問題は M-SIDH 問題と呼ばれる [57]。また，FESTA 鍵共有は $\Gamma_{\mathrm{D}} = \left\{ \begin{pmatrix} \lambda & 0 \\ 0 & \lambda^{-1} \end{pmatrix} \right\} (\subsetneq \mathrm{SL}_2(\mathbb{Z}/N\mathbb{Z}))$ に対する Γ_{D}-SIDH 問題の困難性を安全性の根拠としており，Γ_{D}-SIDH 問題は，対角-SIDH 問題（Diagonal-SIDH 問題）と呼ばれている [57]。6.2.2 節での参照のため，定義 6.4 で，改めて M-SIDH 問題，対角-SIDH 問題を定義する。

定義 6.4 (M-SIDH 問題，対角-SIDH 問題 [57]) 上で与えた $\mathrm{SL}_2(\mathbb{Z}/N\mathbb{Z})$ の部分群 $\Gamma_{\mathrm{M}}, \Gamma_{\mathrm{D}}$ に対して，Γ_{M}-*SIDH* 問題を *M-SIDH* 問題と定義して，Γ_{D}-*SIDH* 問題を対角-*SIDH* 問題と定義する。

M-SIDH 鍵共有においては, 相異なる小素数 $q_1, \ldots, q_t$ を用いて $N = q_1 \cdots q_t$ とした Γ_{M} を用いることで, $\#\Gamma_{\mathrm{M}} = 2^t$ となることを安全性の根拠にしている。また，FESTA では，N を 2 のべき乗とした Γ_{D} を用いる。それらのパラメータに対して，M-SIDH 問題，対角-SIDH 問題共に，現状，指数関数時間の攻撃法しか知られていない [57, 表 1]。

6.1.4 同種写像に基づく一方向性群作用（暗号学的群作用）に関する計算問題

素体 $\mathbb{F}_p$ 上定義された超特異楕円曲線間の同種写像問題の困難性に基づく鍵共有法として，CSIDH 鍵共有（6.2.1 節参照）が 2018 年になって Castryck らによって提案された [29]。その安全性の根拠となる計算問題を [42, 4] に従ってまとめる。以降では，整数 a, b（$a < b$）に対して $\mathbb{Z}$ の部分集合 $[a, b]$ を $[a, b] := \{a, a+1, \ldots, b-1, b\}$ とする。

6.1.4.1 2 種の一方向性群作用：REGA と EGA

■CSIDH 鍵共有・CSI-FiSh 署名方式の公開パラメータ CSIDH 鍵共有で，公開パラメータは $pp_{\texttt{csidh}} := (\mathfrak{O}, (\mathfrak{l}_1, \mathfrak{l}_2, \ldots, \mathfrak{l}_n),\ E, B)$ で与えられる。ここで，$\mathfrak{O}$ は虚 2 次代数体の整環，$\mathfrak{l}_1, \mathfrak{l}_2, \ldots, \mathfrak{l}_n$ はノルムが小さい奇素数 ℓ_i になる $\mathfrak{O}$ の素イデアルで，ℓ_i は $(\ell_i) := \mathfrak{l}_i\bar{\mathfrak{l}}_i$（$i = 1, 2, \ldots, n$）と 2 個の異なる素イデアル $\mathfrak{l}_i, \bar{\mathfrak{l}}_i$ の積に分解している。そして $p + 1 = 4 \cdot \ell_1 \cdots \ell_n$ とした時，p は素数である必要がある。小奇素数 ℓ_i は，例えば，$\ell_1 = 3, \ell_2 = 5, \ldots$ である。E は，$\mathbb{F}_p$ 上定義されて，$\mathfrak{O}$ を $\mathbb{F}_p$-自己準同型環にもつ超特異楕円曲線である。B は指数 e_i の絶対値の上限値，すなわち $-B \le e_i \le B$ となる指数 e_i を CSIDH 鍵共有では使う。

CSI-FiSh 署名方式の公開パラメータには，イデアル類群 $\mathrm{cl}(\mathfrak{O})$ の構造計算がなされた $pp_{\texttt{csidh}}$ が含まれる。そして，$\mathrm{cl}(\mathfrak{O})$ の生成元の間の関係式の情報 $\mathcal{R}$ も付された公開パラメータ $pp_{\texttt{csi-fish}} := (pp_{\texttt{csidh}}, \mathcal{R})$ を用いることで，有限アーベル群 $\mathrm{cl}(\mathfrak{O})$ 上での効率的な一様サンプリングが可能になる。特にその代表的なパラメータである CSIDH-512 では，$\mathrm{cl}(\mathfrak{O})$ がノルム 3 のイデアル $\mathfrak{l}_1$ により生成される巡回群であることが [16] において示された。現在，イデアル類群 $\mathrm{cl}(\mathfrak{O})$ の構造計算は入力 $\mathfrak{O}$ のサイズに対して（古典）準指数時間必要であるので，CSIDH-512 のように生成元が既知の十分大きいイデアル類群を得ることは一般に困難で，公開パラメータ $pp_{\texttt{csi-fish}}$ 生成に対する大きな制約となっている。

■CSIDH・CSI-FiSh 群作用の抽象形としての REGA・EGA CSIDH 鍵共有，及び CSI-FiSh 署名方式での基本演算は，$\mathbb{F}_p$-自己準同型環に虚 2 次代数体の整環 $\mathfrak{O}$ をもつ楕円曲線集合 X に対する $\mathfrak{O}$ のイデアル類群 $G := \mathrm{cl}(\mathfrak{O})$ の群作用 $(g, x) \mapsto gx \in X$ (但し $g \in G, x \in X$) として理解できる。その群作用は，自由かつ推移的である。群作用の詳細については 6.2.1.1 節を参照のこと。その記法に従えば，CSIDH・CSI-FiSh における同種写像問題は，この群作用の（G に関する）逆関数 $(x, gx) \mapsto g$ を計算する問題と理解できる。このことから，CSIDH・CSI-FiSh における群作用は，一方向性群作用（または暗号学的群作用）と呼ばれる [42, 113]。

但し，CSIDH 鍵共有と CSI-FiSh 署名方式では，$pp_{\texttt{csidh}}$ と $pp_{\texttt{csi-fish}}$ の違いに応じて G からのサンプリング方法が異なる。CSI-FiSh 署名方式ではすでに述べたように G から効率的に一様サンプリングするのに対して，CSIDH では $[-B, B]^n \subset \mathbb{Z}^n$ から適切に選んだ $(e_1, e_2, \ldots, e_n)$ により計算した $\prod_{i=1}^{n} \mathfrak{l}_i^{e_i}$ によって G からのサンプリングを行う。

これらサンプリング方法の違いに基づいて，最近の一方向性群作用の研究 [4, 91] では，CSI-FiSh 署名方式の場合の一方向性群作用を EGA（Effective Group Action：有効群作用）と呼び，CSIDH 鍵共有の場合の群作用を REGA（Restricted Effective Group Action：制限的有効群作用）と呼んでいる。以降では，基本構成としての「CSIDH」を接頭辞に付けて，それぞれの一方向性群作用を CSIDH-EGA，CSIDH-REGA と呼ぶことにする。

6.1.4.2 CSIDH-(R)EGA 上の計算問題

■CSIDH-EGA 上の DL,CDH 計算問題とそれらの量子帰着同値性 CSIDH-EGA に関して離散対数問題（Discrete Logarithm: DL）にあたる基本問題は，以下の CSIDH-EGA-DL 問題であり，更に，それに基づいた CDH（Computational Diffie–Hellmann）問題は，CSIDH-EGA-CDH 問題である。それらの問題は，それぞれ，CSIDH ベクトル化問題及び CSIDH 並列化問題と呼ばれることもある [42, 113, 30]。以下，イデアル類 $[\mathfrak{a}]$ の群作用 $[\mathfrak{a}]E$ に関しては 6.2.1.1 節を参照のこと。

定義 6.5 (CSIDH-EGA-DL 問題 [42, 29, 4]) *CSI-FiSh* 署名公開パラメータ $pp_{\texttt{csi-fish}}$ と，$\mathbb{F}_p$ 上定義されており $\mathbb{F}_p$-自己準同型環 $\mathfrak{O}$ をもつ超特異楕円曲線 $E, E_{\mathtt{A}}$ が与えられた時，$E_{\mathtt{A}} = [\mathfrak{a}]E$ となる $\mathfrak{O}$ のイデアル $\mathfrak{a}$ を計算せよ。但し，$\mathfrak{a}$ の E への作用が効率的に計算可能な場合に限る。例えば，$\mathfrak{a}$ が小さい次数のイデアル積で与えられる場合などである。

定義 6.6 (CSIDH-EGA-CDH 問題 [42, 29, 4]) *CSI-FiSh* 署名公開パラメータ $pp_{\texttt{csi-fish}}$ と，$\mathbb{F}_p$ 上定義されており $\mathbb{F}_p$-自己準同型環 $\mathfrak{O}$ をもつ超特異楕円曲線 $E, E_{\mathtt{A}} := [\mathfrak{a}]E, E_{\mathtt{B}} := [\mathfrak{b}]E$（但し，$\mathfrak{a}, \mathfrak{b}$ は群作用が効率的に計算できる $\mathfrak{O}$ のイデアル）が与えられた時，$[\mathfrak{ab}]E = [\mathfrak{b}]E_{\mathtt{A}} = [\mathfrak{a}]E_{\mathtt{B}}$ を計算せよ。

通常の DL 問題と CDH 問題の場合のように，CSIDH-EGA-CDH 問題を CSIDH-EGA-DL 問題に帰着させることができるが，[68] において，その逆，CSIDH-EGA-CDH 問題を解くオラクルを用いて CSIDH-EGA-DL 問題を解く多項式時間量子帰着アルゴリズムが提案されている。[68] の帰着では，CSIDH-EGA-CDH 問題を完全に解くオラクルを仮定していた。Montgomery ら [91, 70] により有意な（non-negligible）確率で CSIDH-EGA-CDH 問題に答えるオラクルを用いても [68] の帰着が成り立つことが示された。

更に，[91] では，CSIDH-REGA 上では [68] の帰着結果が成り立たないことも示されており，判定版の CSIDH-EGA-DDH 問題へ拡張する事に関しても否定的な結果が示されている。また，CSIDH-EGA-DL 問題と CSIDH-EGA-CDH 問題の古典帰着に関して，Castryck ら [30] は，（古典）一方向性の準同型写像が存在するという妥当な仮定の下に，一般には EGA-DL 問題を EGA-CDH 問題に古典帰着させることができないことを簡潔な反例によって示した。

■CSIDH-REGA 上の DL,CDH 計算問題　既に述べたように，現在，イデアル類群 $G := \mathrm{cl}(\mathfrak{O})$ の構造計算を多項式時間で行う（古典）アルゴリズムは知られていないため，G 上の一様分布からの効率的なサンプリング法も知られていない。よって，近似的にその一様サンプリングを行う効率的な（秘密鍵）サンプリング法を用いて CSIDH 鍵共有は与えられる（6.2.1 節参照）。それに従って，CSIDH-EGA-DL 問題と CSIDH-EGA-CDH 問題もそれぞれ修正されて，それらを CSIDH-REGA-DL 問題，CSIDH-REGA-CDH 問題として以下に与える。これらの問題も，前段落と同様に，CSIDH ベクトル化問題及び CSIDH 並列化問題と呼ばれることもある [42, 113, 30]。

定義 6.7 (CSIDH-REGA-DL 問題 [29, 4]) *CSIDH* 鍵共有公開パラメータ $pp_{\texttt{csidh}}$ と，$\mathbb{F}_p$ 上定義されており $\mathbb{F}_p$-自己準同型環 $\mathfrak{O}$ をもつ超特異楕円曲線 E，及び $[-B, B]^n \subset \mathbb{Z}^n$ から一様ランダムに選んだ $(e_1, e_2, \ldots, e_n)$ により $\mathfrak{a} := \prod_{i=1}^{n} \mathfrak{l}_i^{e_i}$ となる $\mathfrak{a}$ によって $E_{\mathtt{A}} = [\mathfrak{a}]E$ となる $E_{\mathtt{A}}$ が与えられた時，$\mathfrak{a}$ と同じイデアル類に属する $\mathfrak{a}' \in [\mathfrak{a}]$ を計算せよ。

定義 6.8 (CSIDH-REGA-CDH 問題 [29, 4]) *CSIDH* 鍵共有公開パラメータ $pp_{\texttt{csidh}}$ と，$\mathbb{F}_p$ 上定義されており $\mathbb{F}_p$-自己準同型環 $\mathfrak{O}$ をもつ超特異楕円曲線 $E, E_{\mathtt{A}} := [\mathfrak{a}]E, E_{\mathtt{B}} := [\mathfrak{b}]E$（但し，$\mathfrak{a}, \mathfrak{b}$ は共に，$[-B, B]^n \subset \mathbb{Z}^n$ から一様ランダムに選んだ $(e_1, e_2, \ldots, e_n)$ により $\prod_{i=1}^{n} \mathfrak{l}_i^{e_i}$ と表されるイデアル）が与えられた時，$[\mathfrak{ab}]E = [\mathfrak{b}]E_{\mathtt{A}} = [\mathfrak{a}]E_{\mathtt{B}}$ を計算せよ。

■CSIDH-(R)EGA-DL 問題の古典計算機による計算困難性　CSIDH-(R)EGA-DL 問題に関しては，$\mathbb{F}_p$ 上の超特異楕円曲線の同種写像問題に対する Delfs–Galbraith [49] の（古典）アルゴリズムを適用するのが，漸近的解読時間が最速の古典アルゴリズムとされており，その解読時間は $\tilde{O}(\sqrt[4]{p})$ である。

■CSIDH-(R)EGA-DL 問題に対する準指数時間での量子攻撃　G の X への作用が自由かつ推移的であるなら，群作用 DL 問題は，隠れシフト問題に帰着されて，それは更に二面体群に関する隠れ部分群問題（Dihedral Hidden Subgroup Problem: DHSP）に帰着する。DHSP には，準指数時間で動く量子アルゴリズムが知られているので，一般に一方向性群作用に基づいた暗号方式は，量子計算機に対して準指数時間安全性しかもたない。つまり，Childs ら [37] による

通常同種写像問題に対する量子準指数時間アルゴリズムは，CSIDH-(R)EGA-DL 問題に対しても有効であり，CSIDH 群作用に関する DL 問題に対する量子計算機による漸近的な解読時間は準指数関数 $L_p[1/2, \sqrt{3}/2]$ で与えられる[*2]。

実用的には，漸近的ではないその正確な見積もりが，与えられた安全性レベルを達成する p の bit 長を決めるのに重要である。まず，Eurocrypt 2019 において，Bernstein ら [13] は，CSIDH 群作用を行う量子回路のサイズを具体的に見積もることで，上述の準指数時間アルゴリズムが，従来考えられていたより計算オーバーヘッドが大きいのでないか，つまり，攻撃するのはより困難であろうと主張している。

更に，Eurocrypt 2020 において，Bonnetain–Schrottenloher [19] と Peikert [101] により独立に 2 つの研究成果が報告された。[19] では，詳細に CSIDH 攻撃量子アルゴリズムを検討して，これまで考えられていたより効率的に攻撃可能であると主張している。それにより，彼らは，Castryck ら [29] が 56 bits 量子安全性レベルと主張していたパラメータが，実際には 38 bits レベルの量子安全性しか確保できないのでないか，という試算を述べている。また，[19] では，Kuperberg の 2005 年の論文 [82] に基づいた攻撃法に関して特に詳細な解析がなされたが，Peikert [101] では，その後の DHSP 解法の進展 [104, 83] も取り入れた安全性評価の改善がなされた。

そして，Chávez-Saab ら [35] の最近の評価結果では，NIST 安全性レベル 1 を満たすために素数 p を 4096 bits または 5120 bits 程度に大きくする必要性が示されており，安全性レベル 2 には 6144 bits，安全性レベル 3 には 8192 bits または 9216 bits 程度の大きさが必要であるという評価結果も報告されている。

■CSIDH・CSI-FiSh パラメータ以外のイデアル類群作用 DDH 問題の古典解法　上で見た EGA 上の DL 問題と CDH 問題の同値性と関係した DDH（Decisional Diffie–Hellman）問題に関する研究成果が Castryck ら [31, 28, 27] により発表された。ある種のイデアル類群作用においては，虚 2 次整環における「種の理論（genus theory）」を用いて，その作用に関する DDH 問題を効率的に解くことができることが示された。

但し，CSIDH 鍵共有・CSI-FiSh 署名方式に対して重要なこととして，その攻撃が有効になるためにはイデアル類群 $G := \mathrm{cl}(\mathfrak{O})$ が 2-ねじれ点を持つ必要があり，$p \equiv 3 \bmod 4$ である CSIDH パラメータに対しては無効であることも示されている。

6.1.4.3　イデアル類群作用に基づく量子マネーの安全性に関する計算問題

また，現在，イデアル類群の群作用に基づいて公開検証可能な量子マネーを構成する研究も活発に行なわれている [85, 120, 90, 95]。特に，Montgomery–Sharif [95] では，量子マネー（quantum money/quantum lightning）の安全性を示すために，楕円曲線重ね合わせ複製問題（Elliptic Curve Superposition Duplication Problem: ECSDP）や楕円曲線重ね合わせ衝突問題（Elliptic Curve Superposition Collision Problem: ECSCP）といった計算問題が定義されて，その計算困難性が論じられている。

6.1.5　自己準同型環計算問題と SQIsign 署名方式の安全性に関する計算問題

6.1.5.1　自己準同型環計算問題

同種写像暗号は，Kohel [80]，Galbraith [69]，Couveignes [42] らの先駆的研究にその起源をもつが，特に，Kohel は有限体上の楕円曲線の自己準同型環を計算するアルゴリズムを探求しており，そのために楕円曲線の同種写像からなる「同種写像グラフ」の性質を見極めることから始めて，目的とする自己準同型環計算を同種写像グラフ上のアルゴリズム構成に帰着していく。その後，Kohel–Lauter–Petit–Tignol [81] は，この「同種写像計算」と「自己準同型環計算」を並置しながら考察する視点を，「構成的 Deuring 対応」として計算論的観点から捉え直した（表 6.1 参照）。そこでは，四元数環側での ℓ-同種写像道探索問題を解く KLPT アルゴリズムが鍵となるアルゴリズムである（[78, 54] 参照）。そして，この構成的 Deuring 対応に基づき「同種写像計算」と「自己準同型環計算」の等価性が示されており [51, 52,

[*2] ここで，$L_p[\alpha, c] := \exp\left((c + o(1))\,(\log p)^{\alpha}(\log\log p)^{1-\alpha}\right)$ とする。

119]，現在，自己準同型環計算問題の困難性に基づいた暗号構成の研究が進められている [72, 61, 62]。

■自己準同型環計算問題とその超特異同種写像計算問題との同値性　以下の記述に関しては，例えば [84] を参照。また，四元数環については Voight の教科書 [117] に詳しい説明がある。有理数体 $\mathbb{Q}$ 上 $\{1, i, j, k\}$ を基底とするベクトル空間でありかつ $a, b \in \mathbb{Q}$ により $i^2 = a, j^2 = b, k = ij = -ji$ という積構造が入った $\mathbb{Q}$ 上の代数（環）を四元数環 $\mathcal{B}$ と呼ぶ。各素点 ν（素数または ∞）における $\mathbb{Q}$ の完備化 $\mathbb{Q}_\nu$ による $\mathcal{B} \otimes \mathbb{Q}_\nu$ が $\nu = p, \infty$ の時にのみ斜体（可除環）になる四元数環 $\mathcal{B} = \mathcal{B}_{p,\infty}$ を扱う。これを，$\mathcal{B}_{p,\infty}$ は p, ∞ の 2 点のみで分岐する四元数環であるといい，$\mathcal{B}_{p,\infty}$ は同型を除いて一意的に決まる。この同じ素数 p を標数とする有限体上の超特異楕円曲線 E の自己準同型写像がなす環 $\mathrm{End}(E)$ は E の自己準同型環と呼ばれて，$\mathrm{End}(E)$ は $\mathcal{B}_{p,\infty}$ の極大整環 $\mathcal{O}$ になっている[*3]。ここで，（四元数環の）整環とは $\mathbb{Z}$ 上階数 4 の加群でありかつ環であるものであり，極大整環とは，そのような整環の中で包含関係に関して極大になっているものを指す。この自己準同型環 $\mathrm{End}(E)$ を計算する以下の問題が基本である。

定義 6.9 (自己準同型環計算問題 [80]) 超特異楕円曲線 E が与えられて，E の自己準同型環 $\mathrm{End}(E)$ を計算せよ。

Eisenträger らの研究 [51, 52] により，超特異同種写像計算問題と（超特異）自己準同型環計算問題の間に多項式時間帰着による計算問題としての同値性が示された。そこではヒューリスティックな仮定が使われていたが，Wesolowski [119] は，一般化されたリーマン予想に基づいて，その同値性に対して厳密な証明を与えた。また，[100, 88] においては，非スカラー自己準同型写像を計算する問題（One Endomorphism Problem）と自己準同型環計算問題の等価性も示されている。

6.1.1 節で，超特異同種写像問題の古典計算機による現在最速の解読時間は $\tilde{O}(\sqrt{p})$ と見積もられていたので，この同値性により，自己準同型環計算問題も同等の計算時間であるが，直接に，自己準同型環計算問題を解く研究も進められており，[52] において，$\tilde{O}(\sqrt{p})$ 時間の自己準同型環計算（古典）アルゴリズムが報告されており，その後 [67] により改良されている。また，Kambe ら [77] によって，10 から 30 bits の素数 p に対する自己準同型環計算の実装報告がなされている。

■Deuring 対応　自己準同型環計算問題で与えられる楕円曲線 E から極大整環 $\mathcal{O}$ への対応は，表 6.1 に掲げたように，楕円曲線に関する様々な概念から四元数環に関する概念への対応に拡張される。その詳細に関しては，例えば [84, 第 2 章] に記述があるが，特に基本的な対応としては，同種写像 $\varphi : E \to E_1$ が，極大整環の間の同型 $\mathcal{O} \cong \mathrm{End}(E), \mathcal{O}_1 \cong \mathrm{End}(E_1)$ を通して，左 $\mathcal{O}$-整イデアルかつ右 $\mathcal{O}_1$-整イデアルである I_φ に対応していることである。これにより始点曲線 E を固定すると，同種写像 $\varphi : E \to E_1$ の終点曲線 E_1 が $\mathcal{O}$ のイデアル類と対応することがわかり，超特異 j 不変量（$\in \mathbb{F}_{p^2}$）の集合がイデアル類集合 $\mathrm{cl}(\mathcal{O})$ と一対一に対応していることもわかる。

一般に表 6.1 に示されるように，幾何的な情報から成る楕円曲線側のデータと代数的な情報から成る四元数環側のデータの間に対応関係が存在しており，Deuring 対応と呼ばれる。自己準同型環計算問題（定義 6.9）は Deuring 対応に基づいた問題であり，楕円曲線側の超特異 j 不変量 $j(E)$ から対応する四元数環側の極大整環 $\mathcal{O} = \mathrm{End}(E)$ を計算する問題となっている。そして，この Deuring 対応は，6.2.4 節及び 6.3.1 節での暗号構成を理解する際にも重要な鍵となっている。

6.1.5.2　SQIsign 署名方式の安全性に関する計算問題

次に，SQIsign 署名方式 [61, 34] の安全性を示すために必要な計算問題を述べる。近年進展が著しい SQIsign2D 署名方式の安全性に関しては [8, 97] などを参照のこと。

■SQIsign 署名方式の健全性に関する計算問題　まずは，SQIsign 署名方式の健全性（偽造不可能性）を示すための計算問題である超特異平滑自己準同型写像計算問題（Supersingular Smooth Endomorphism Problem）を定義する。以

[*3] 自己準同型写像は英語で endomorphism であるので，その全体を $\mathrm{End}(E)$ で表す。

表 6.1: Deuring 対応

楕円曲線側	四元数環側
超特異 j 不変量 $j(E) \in \mathbb{F}_{p^2}$（の $\mathbb{F}_{p^2}/\mathbb{F}_p$-Galois 共役類）	$\mathcal{B}_{p,\infty}$ 内の極大整環 $\mathcal{O} = \mathrm{End}(E)$ の自己同型類（タイプ）
同種写像 $\varphi : E \to E_1$ で定まる (E_1, φ)	左 $\mathcal{O}$-整イデアルかつ右 $\mathcal{O}_1$-整イデアルである I_φ
自己準同型写像 $\theta \in \mathrm{End}(E)$	主イデアル $\mathcal{O}\theta$
同種写像の次数 $\deg(\varphi)$	イデアルのノルム $n(I_\varphi)$
双対同種写像 $\hat{\varphi}$	共役イデアル $\overline{I_\varphi}$
同じ定義域・値域の同種写像 $\varphi : E \to E_1, \psi : E \to E_1$	同値なイデアル $I_\varphi \sim I_\psi$
超特異 j 不変量 $j(E) \in \mathbb{F}_{p^2}$ の集合	イデアル類の集合 $\mathrm{cl}(\mathcal{O})$
同種写像の合成 $\tau \circ \rho : E \to E_1 \to E_2$	イデアル積 $I_{\tau\circ\rho} = I_\rho \cdot I_\tau$
N-同種写像の同型類	レベル N の Eichler 整環の類集合

下では，核が巡回群となる自己準同型写像を巡回自己準同型写像と呼ぶ。

定義 6.10 (超特異平滑自己準同型写像計算問題 [61, 34]) 超特異楕円曲線 E が与えられて，平滑な整数を次数にもつ E 上の（非自明な）巡回自己準同型写像を見つけよ。

この問題で問うているような非自明な自己準同型写像が計算できれば，[52] で見るように，自己準同型環 $\mathrm{End}(E)$ 全体も計算できることが知られているので，この問題は，本質的に自己準同型環計算問題と同値である [61]。よって，$\tilde{O}(\sqrt{p})$ 時間での古典アルゴリズム [52] が現状最速と見積もられる。

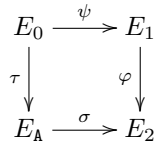
■特殊極値的楕円曲線 次に，SQIsign 署名方式のゼロ知識性を示すための計算問題を述べるが，公開パラメータで重要となる楕円曲線 E_0 を示す。$p = 3 \bmod 4$ の時，j 不変量 $j = 1728$ となる $E_0 : y^2 = x^3 + x$ の $\mathcal{O}_0 = \mathrm{End}(E_0)$ は $i^2 = -1, j^2 = -p$ となる $\mathcal{O}_0 = \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}\frac{i+j}{2} + \mathbb{Z}\frac{1+ij}{2}$ となることが知られている。更に具体的に自己準同型写像 $\iota : (x, y) \mapsto (-x, \sqrt{-1}y), \pi : (x, y) \mapsto (x^p, y^p)$ により $\mathrm{End}(E_0) = \mathbb{Z} + \mathbb{Z}\iota + \mathbb{Z}\frac{\iota+\pi}{2} + \mathbb{Z}\frac{1+\iota\pi}{2}$ で与えられる。

標数 p と ∞ のみで分岐する四元数環 $\mathcal{B}_{p,\infty} := \mathbb{Q}[i, j]$ における極大整環 $\mathcal{O}_0 \subset \mathcal{B}_{p,\infty}$ は，最小判別式の 2 次整環である $\mathfrak{O} \subset \mathcal{O}_0 \cap \mathbb{Q}[i]$ による $\mathfrak{O} + j\mathfrak{O} \subset \mathcal{O}_0$ が部分整環であり $\mathfrak{O} \subset (j\mathfrak{O})^\perp$ と直交分解しているとき[*4]，特殊極値的（special extremal）であるという。詳細は [61, 84] を参照。$p = 7 \bmod 12$ の時，上述の E_0 に対して $\mathrm{End}(E_0)$ は特殊極値的であり，この時，E_0 は特殊極値的曲線と呼ばれる。特殊極値的曲線 E_0 は，その自己準同型環の構造が簡単で計算上扱いやすいため GPS 署名方式 及び SQIsign 署名方式の公開パラメータの一部として必要である。

[*4] $\mathcal{B}_{p,\infty}$ における内積は $\alpha, \beta \in \mathcal{B}_{p,\infty}$ に対して $\frac{1}{2}\mathrm{tr}(\alpha\bar{\beta})$ で与えられて，ここは，その内積に関する直交分解である（$\mathcal{B}_{p,\infty}$ 内のトレース，共役の定義は，例えば [84] を参照のこと）。

■SQIsign 署名方式のゼロ知識性に関する計算問題　SQIsign 署名方式では，右図の同種写像 τ が秘密鍵で，超特異楕円曲線 $E_{\mathtt{A}}$ が公開鍵（の主要な一部）である。署名生成では，同種写像 ψ, φ を適切に生成して得られた合成写像 $\varphi \circ \psi \circ \hat{\tau}$ を「ランダム化」した同種写像 σ を署名とする[*5]。その詳細は 6.3.1.1 節を参照。[61, 62] において定義された E_0 を始点とする同種写像から成るある集合 $\mathcal{P}_{N_\tau}$ を τ によって $E_{\mathtt{A}}$ を始点とした同種写像に移した集合 $[\tau]_*\mathcal{P}_{N_\tau}$（$\mathcal{P}_{N_\tau}$ の τ による pushforward）を考える。正しく生成された署名同種写像 σ は $[\tau]_*\mathcal{P}_{N_\tau}$ に属するのであるが，それが $E_{\mathtt{A}}$ を始点とした 2 べき次数 $D\,(=2^e)$ の巡回同種写像全体 $\mathtt{Iso}_{D,j(E_{\mathtt{A}})}$ から一様ランダムにサンプリングしたのと区別が付くかという問題が以下であり，SQIsign 署名方式のゼロ知識性を示すために必要である。

SQIsign 同種写像図式



CSI-FiSh，GPS 図式と同様に可換図式ではない。

定義 6.11 (SQIsign 署名方式のランダム識別問題 [61, 34]) $\tau : E_0 \to E_{\mathtt{A}}$ を秘密同種写像として，楕円曲線 E_0 を含む *SQIsign* 署名方式の公開パラメータ $pp_{\mathtt{sqisign}}$（詳しくは *6.3.1* 節参照）と公開鍵 $E_{\mathtt{A}}$ が入力として与えられると共に，$[\tau]_*\mathcal{P}_{N_\tau}$ から一様サンプリングして返すオラクル O_τ への多項式回のアクセスが許される時に，$E_{\mathtt{A}}$ を始点とする同種写像 σ が与えられて σ が $\mathtt{Iso}_{D,j(E_{\mathtt{A}})}$ から一様ランダムに選ばれたか，$[\tau]_*\mathcal{P}_{N_\tau}$ から一様ランダムに選ばれたかを判定せよ。

SQIsign 署名方式の提案者によると，現在のところ，SQIsign 署名方式のランダム識別問題を解くのに，E_0 と $E_{\mathtt{A}}$ の情報から τ を暴く攻撃法より効率の良い攻撃法はまだ知られていないとのことである [61, 84]。つまり，$\tilde{O}(\sqrt{p})$ 時間を必要とすると見積もられている。また，上述の SQIsign 署名方式に関する計算問題は，どちらも補助点を問題に含まないことにより，6.1.2 節で見た最近の SIDH 同種写像問題に対する攻撃法が適用できないことに注意する。

6.2　同種写像に基づく代表的な暗号方式

以下では，6.2.1 節で CSIDH 鍵共有と CSIDH 鍵共有以外の群作用暗号を，6.2.2 節でレベル構造付き同種写像問題に基づく鍵共有方式を，6.2.3 節で暗号学的群作用に基づく署名方式を，6.2.4 節で GPS 署名方式を述べる。

また，ここで述べた方式以外にも POKE [7]，IS-CUBE [92]，LIT-SiGamal [93] など新たな鍵共有方式が提案されていることにも注意する。

6.2.1　暗号学的群作用に基づく鍵共有方式

6.2.1.1　CSIDH 鍵共有

Castryck ら [29] により提案された CSIDH 鍵共有を記述する。CSIDH 鍵共有は，有限アーベル群 G による一方向性群作用をもつ空間 X 上で構成される。ここで，$X = \mathcal{E}\ell\ell_p(\mathfrak{O}, \pi)$ は，$\mathbb{F}_p$ 上定義されて $\mathbb{F}_p$-有理自己準同型環が固定された虚 2 次整環 $\mathfrak{O}$ と同型であり，かつその同型により p 乗フロベニウス写像が $\pi \in \mathfrak{O}$ に移されるような超特異楕円曲線の $\mathbb{F}_p$-同型類の集合であり，$G = \mathrm{cl}(\mathfrak{O})$ は $\mathfrak{O}$ のイデアル類群である。以下では，$\mathcal{E}\ell\ell_p(\mathfrak{O}, \pi)$ が空でないと仮定する。Castryck ら [29] は，6.1.4.2 節で定義した CSIDH-REGA-CDH の判定版問題の困難性に基づいて，CSIDH 鍵共有方式を提案した。

K を虚 2 次代数体，$\mathfrak{O} \subset K$ をその整環とする，すなわちランク 2 の自由 $\mathbb{Z}$-加群である K の部分環である。$\mathfrak{O}$-分数イデアルは，$\alpha \in K^*$ と $\mathfrak{O}$-イデアル $\mathfrak{a}$ によって $\alpha\mathfrak{a}$ と表される K 内の $\mathfrak{O}$-部分加群である。$\mathfrak{a}\mathfrak{b} = \mathfrak{O}$ となる $\mathfrak{O}$-分数イデアル $\mathfrak{b}$ が存在する時に（$\mathfrak{O}$-分数イデアル）$\mathfrak{a}$ は可逆であるという。そして，そのような $\mathfrak{b}$ が存在するならば，$\mathfrak{a}^{-1} = \mathfrak{b}$

[*5] $\hat{\tau}$ は τ の双対同種写像である。表 6.1 も参照のこと。

と定義する。可逆分数イデアルの集合 $I(\mathfrak{O})$ はイデアル積に関してアーベル群をなす。この群には主イデアルからなる部分群 $P(\mathfrak{O})$ が含まれており，$\mathfrak{O}$ のイデアル類群は商群 $\mathrm{cl}(\mathfrak{O}) = I(\mathfrak{O})/P(\mathfrak{O})$ によって定義される。どのイデアル類 $[\mathfrak{a}] \in \mathrm{cl}(\mathfrak{O})$ にも整イデアルが存在してその代表として使うことができる。$\mathfrak{O}$ のどの整イデアル $\mathfrak{a}$ も $\mathfrak{O}$-イデアルの積として $\mathfrak{a}_s \not\subseteq \pi\mathfrak{O}$ となる整イデアル $\mathfrak{a}_s$ によって $(\pi\mathfrak{O})^r\mathfrak{a}_s$ と表せる。ここで，π は，p 乗フロベニウス写像。この表示により，整イデアル $\mathfrak{a}$ に対して楕円曲線 $E/E[\mathfrak{a}]$ とそこへの $N(\mathfrak{a})$ 次同種写像 $\varphi_{\mathfrak{a}} : E \to E/E[\mathfrak{a}]$ が以下のように定義される。ここで，$N(\mathfrak{a}) := \#(\mathfrak{O}/\mathfrak{a})$ は $\mathfrak{a}$ のノルムである。$\varphi_{\mathfrak{a}}$ の分離的な部分は $E[\mathfrak{a}] = \bigcap_{\alpha \in \mathfrak{a}_s} \ker \alpha$ を核にもつ同種写像であり，純非分離的な部分はフロベニウス写像 π の r 回の繰り返しで与えられる。同種写像 $\varphi_{\mathfrak{a}}$ 及び値域曲線 $E/E[\mathfrak{a}]$ は共に $\mathbb{F}_p$ 上定義されており $\mathbb{F}_p$-同型を除いて一意的に決まる。ここで主イデアルにより定義される同種写像は E 上の自己準同型写像になるので，2 つのイデアルが同じイデアル類に属することと，対応する同種写像が $\mathbb{F}_p$-同型な値域曲線を与えることとは同値である。つまり，$E/E[\mathfrak{a}]$ の $\mathbb{F}_p$-同型類はイデアル類 $[\mathfrak{a}]$ のみにより決まり，特に，この対応はイデアル類群 $\mathrm{cl}(\mathfrak{O})$ の $\mathcal{E}\ell\ell_p(\mathfrak{O}, \pi)$ への作用を与える。更に，$\mathcal{E}\ell\ell_p(\mathfrak{O}, \pi)$ に属する 2 つの楕円曲線間の $\mathbb{F}_p$-同種写像 ψ はすべてこの対応により可逆な $\mathfrak{O}$-イデアルから得られる。そして分離部分 $\mathfrak{a}_s$ は ψ から $\mathfrak{a}_s = \{\alpha \in \mathfrak{O} \mid \ker \alpha \supseteq \ker \psi\}$ によって復元できる。その対応は以下の定理にまとめられる。

定理 6.12 ([29]) 虚 2 次代数体内の整環 $\mathfrak{O}$ と $\pi \in \mathfrak{O}$ を $\mathcal{E}\ell\ell_p(\mathfrak{O}, \pi)$ が空集合でないものとする。その時，以下で与えられるイデアル類群 $\mathrm{cl}(\mathfrak{O})$ の $\mathcal{E}\ell\ell_p(\mathfrak{O}, \pi)$ への作用は自由かつ推移的である。

$$\begin{array}{ccc} \mathrm{cl}(\mathfrak{O}) \times \mathcal{E}\ell\ell_p(\mathfrak{O}, \pi) & \to & \mathcal{E}\ell\ell_p(\mathfrak{O}, \pi) \\ ([\mathfrak{a}], E) & \mapsto & E/E[\mathfrak{a}], \end{array}$$

ここで，$\mathfrak{a}$ は類 $[\mathfrak{a}]$ を代表する整イデアルである。

以下では $E/E[\mathfrak{a}]$ を $[\mathfrak{a}]E$ と書くことにする。定理 6.12 で述べた群作用に基づいて，以下のように CSIDH 鍵共有プロトコル（図 6.1）を定義する。下の図で，$\mathfrak{a} \leftarrow \mathrm{cl}(\mathfrak{O})$ と書かれている部分は，実際にはイデアル類群 $\mathrm{cl}(\mathfrak{O})$ からのサンプリングとして，定義 6.7 の CSIDH-REGA-DL 問題及び定義 6.8 の CSIDH-REGA-CDH 問題に記載された REGA としての $\mathfrak{a}$ のサンプリング法を用いる。モンゴメリ型楕円曲線 $E : y^2 = x^3 + ax^2 + x$ に対して，係数 a は，E のモンゴメリ係数と呼ばれる。CSIDH 鍵共有では，始点曲線 $E : y^2 = x^3 + x$ に対して，アリスとボブによって計算される楕円曲線はすべてモンゴメリ型楕円曲線である。

アリス		**ボブ**
$\mathfrak{a} \leftarrow \mathrm{cl}(\mathfrak{O})$: アリスの秘密鍵		$\mathfrak{b} \leftarrow \mathrm{cl}(\mathfrak{O})$: ボブの秘密鍵
$[\mathfrak{a}]E$（のモンゴメリ係数）を計算	$\xrightarrow{[\mathfrak{a}]E}$	$[\mathfrak{b}]E$（のモンゴメリ係数）を計算
$SK_{\mathtt{Alice}} := [\mathfrak{a}]([\mathfrak{b}]E)$ のモンゴメリ係数	$\xleftarrow{[\mathfrak{b}]E}$	$SK_{\mathtt{Bob}} := [\mathfrak{b}]([\mathfrak{a}]E)$ のモンゴメリ係数

図 6.1: CSIDH 鍵共有の概要

イデアル類群 $\mathrm{cl}(\mathfrak{O})$ は可換なので，$[\mathfrak{a}]([\mathfrak{b}]E) = [\mathfrak{a}\mathfrak{b}]E = [\mathfrak{b}\mathfrak{a}]E = [\mathfrak{b}]([\mathfrak{a}]E)$ であり，そのモンゴメリ係数を考えれば $SK_{\mathtt{Alice}} = SK_{\mathtt{Bob}}$ となるので，アリスとボブは同じ鍵を共有できる。その計算アルゴリズムについては，Castryck ら [29]，Meyer ら [89]，Onuki ら [98] などを参照のこと。CSIDH 鍵共有の安全性は，CSIDH-REGA-CDH 問題の困難性に基づく。

最近, 主に CSIDH 系の方式で使う演算を効率化するべき根同種写像（radical isogeny）計算法 [26, 24] や square-root Vélu 計算法 [12]，さらにそれらを組み合わせた計算法 [47] が提案されている。また，Chávez-Saab らによって，効率化された SQALE 鍵共有方式 [35] も提案されており，4096 bits 以上の CSIDH 素数パラメータに関する SQALE 鍵共有の実装報告もなされている。2024 年に出版された CSIDH 実装報告 [21] も参照のこと。

6.2.1.2 群作用に基づく CSIDH 以外の鍵共有方式

CSIDH 鍵共有の変種には，OSIDH 鍵共有 [38, 44] や SiGamal 暗号方式 [94]・Sims 暗号方式 [66] がある。OSIDH 鍵共有では，楕円曲線以外に「向き」（orientation）と呼ばれる付加情報への群作用も考慮しており，SiGamal 暗号方式では，楕円曲線とその上のねじれ点への群作用を考慮した暗号方式の設計になっている。特に，SiGamal 暗号方式では，ねじれ点への群作用を取り入れることで，CSIDH 共有鍵であるモンゴメリ係数（図 6.1 参照）ではなく，ねじれ点の離散対数からなる一様ランダムな乱数を送受信者間で共有できるようになっている。

2023 年に入ってから，CSIDH 鍵共有以外で向き付けられた楕円曲線に対して新たに SCALLOP [56] と呼ばれる群作用暗号が De Feo らによって提案された。SCALLOP により，利用できる「向き」の取り方を増やすことができて，EGA パラメータの選択肢を増やすことが可能となる。その後，SCALLOP を高次元同種写像を用いて効率化した SCALLOP-HD [36] や，実用的なパラメータ選択のための改良を施した PEARL-SCALLOP [5] が提案された。PEARL-SCALLOP により，CSIDH-512，CSIDH-1024，CSIDH-1536 と同等の安全性レベルの EGA が得られて，それらのパラメータに関して実際に格段の計算効率化が実現できたことが [5] で報告されている。

また，レベル構造付きの群作用を考察した論文 [71, 6] や，従来の群作用をランク 1 加群作用と位置付けてランク 2 加群作用に拡張した論文 [108] など，最近になっても引き続いて，暗号学的群作用の新しい構成法が提案されていることにも注意する。

6.2.2 レベル構造付き同種写像問題に基づく鍵共有

定義 6.3，6.4 で定めたレベル構造付き同種写像問題の困難性に安全性の根拠を置いた鍵共有を述べる。

6.2.2.1 M-SIDH 鍵共有と MD-SIDH 鍵共有

6.1.2 節で見たように，SIDH 鍵共有に対して多項式時間の攻撃法が発見されたが，その直後に，それら攻撃法を回避する SIDH 鍵共有の変種方式が 2 つ提案された。1 つ目は，Moriya による「同種写像次数」を隠すことによる次数隠蔽型（masked-degree）SIDH であり MD-SIDH 鍵共有と呼ばれ，2 つ目は，Fouotsa によるねじれ点隠蔽型（masked torsion point images）SIDH であり M-SIDH 鍵共有と呼ばれる [65]。Eurocrypt 2023 で発表された [65] において，M-SIDH 方式の方が，MD-SIDH 方式より小さい素数 p によって同等の安全性が得られることが述べられている。M-SIDH 鍵共有の安全性は定義 6.4 の M-SIDH 問題の困難性に基づく。一方，これらの方式は，SIDH 方式より効率面では格段に劣ることも指摘されている。例えば，NIST レベル 1,3,5 に対応する M-SIDH 素数 p の bit 長は，それぞれ 5911，9382，13000 bits となることが示されており，SIKE 暗号方式に提案された素数 bit 長（434，610，751 bits）と比べて格段に大きく，それにより計算効率も劣ることになる。また，Castryck–Vercauteren[32] は特殊なパラメータを用いた M-SIDH に対する攻撃法を発表したが，[65] に述べられているパラメータ設定においては有効な攻撃にはなっていない。

6.2.2.2 (Q)FESTA 鍵共有と binSIDH 鍵共有（terSIDH 鍵共有）

M-SIDH 鍵共有以外にも，Asiacrypt 2023 において，レベル構造付き同種写像問題困難性に基づいた 2 方式が発表された。Basso–Maino–Pope による FESTA 鍵共有 [10] と Basso–Fouotsa による binSIDH 鍵共有（及び terSIDH 鍵共有）[9] である。それらの FESTA 鍵共有，binSIDH 鍵共有共に，対角-SIDH 問題（6.4）の困難性に安全性の根拠を置いている [57, 図 1]。

CRYPTO 2024 において，Nakagawa–Onuki により，2 次元同種写像の使い方を改良して，FESTA の効率を高めた QFESTA[96] が提案された。[96] で NIST 安全性レベル 1 パラメータである QFESTA-128 では，公開鍵サイズは 247 Bytes で SIKEp434 と同程度，暗号文サイズは 494 Bytes で SIKEp434 の約 2 倍となっている。そして，NIST

安全性レベル 3,5 の QFESTA-192，QFESTA-256 に関しても対応する SIKE パラメータ SIKEp610，SIKEp751 と同様のデータサイズ比率（公開鍵サイズは同程度で，暗号文サイズは約 2 倍）をほぼ達成しており，現状，効率面で最も良い同種写像ベース鍵共有法を実現している。

また，[32] では，M-SIDH の時と同様に特殊なパラメータの FESTA に対する攻撃法が発表されたが [10] のパラメータ設定においては有効な攻撃にはなっていない。

6.2.3 暗号学的群作用に基づく署名方式

本節では，暗号学的群作用に基づく署名方式として，6.2.3.1 節で SeaSign 署名方式を，6.2.3.2 節で CSI-Fish 署名方式を述べる。また，6.1.4.3 節でも述べたように，群作用暗号に基づく署名・認証系の研究として，最近，量子マネー構成への応用が活発に行なわれている [85, 120, 90, 95]。

6.2.3.1 SeaSign 署名方式

De Feo と Galbraith [58] により，CSIDH ベースの SeaSign 署名方式が提案された。これは，CSIDH 鍵共有の数学的な構造を利用したものであり，まだ実用的とは言い難いが，現実的な計算時間に収まる署名方式となっている。以下では，[58] で，「基本形」と呼ばれる SeaSign 署名方式を記載する。[58] では，更に，基本形でも使われたパラメータ t と共に，パラメータ s を導入して，署名が短い方式や公開鍵が短い方式といった変形方式を定義している。また，後続研究 [48] において，実用化を目指して SeaSign 署名方式の高速化が図られた。

以下では，ベクトル $\mathbf{e}$ の各成分は e_i（$i = 1, 2, \ldots, n$）とする，即ち，$\mathbf{e} = (e_1, e_2, \ldots, e_n)$ である。ベクトル $\mathbf{f}_k, \mathbf{z}_k$ についても同様の記法を用いる。H は t bit 出力のハッシュ関数とする。また，整数 a, b（$a < b$）に対して $[a, b] := \{a, a+1, \ldots, b-1, b\} \subset \mathbb{Z}$ とする。

鍵生成： 公開パラメータ $pp_{\mathtt{csidh}} := (\mathfrak{O}, (\mathfrak{l}_1, \mathfrak{l}_2, \ldots, \mathfrak{l}_n), E, B)$，すなわち，虚 2 次整環 $\mathfrak{O}$，イデアル $\mathfrak{l}_1, \mathfrak{l}_2, \ldots, \mathfrak{l}_n$，$\mathbb{F}_p$-有理な超特異楕円曲線 E，上限値 B を入力とする。係数ベクトル $\mathbf{e} \leftarrow_R [-B, B]^n$ を生成する。$E_{\mathtt{A}} := \left[\prod_{i=1}^{n} \mathfrak{l}_i^{e_i}\right] E$ を計算して，秘密鍵 $sk := \mathbf{e}$，公開鍵 $pk := E_{\mathtt{A}}$ とする。

署名生成： 公開パラメータ $pp_{\mathtt{csidh}}$, メッセージ $\mathtt{msg}$, 公開鍵 $pk := E_{\mathtt{A}}$, 秘密鍵 $sk := \mathbf{e}$ を入力とする。各 $k = 1, 2, \ldots, t$ に対して，係数ベクトル $\mathbf{f}_k \leftarrow_R [-(nt+1)B, (nt+1)B]^n$ を生成して，$\mathcal{E}_k := \left[\prod_{i=1}^{n} \mathfrak{l}_i^{f_{k,i}}\right] E$ を計算する。ハッシュ値を $b_1 \| \cdots \| b_t := H(j(\mathcal{E}_1), \ldots, j(\mathcal{E}_t), \mathtt{msg})$ と bit 分解する。各 $k = 1, 2, \ldots, t$ に対して，もし $b_k = 0$ であれば，$\mathbf{z}_k := \mathbf{f}_k$ として，もし $b_k = 1$ であれば，$\mathbf{z}_k := \mathbf{f}_k - \mathbf{e}$ として，もし $\mathbf{z}_k \notin [-ntB, ntB]^n$ であれば $\perp$ を出力する。そうでなければ，署名 $\sigma := (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t, b_1, b_2, \ldots, b_t)$ を出力する。

署名検証 公開パラメータ $pp_{\mathtt{csidh}}$，メッセージ $\mathtt{msg}$，公開鍵 $pk := E_{\mathtt{A}}$，署名 σ を入力とする。まず，署名 σ が，$\sigma := (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t, b_1, b_2, \ldots, b_t)$ というデータになっていることを確認する。各 $k = 1, 2, \ldots, t$ に対して，もし $b_k = 0$ であれば，$\mathcal{E}_k := \left[\prod_{i=1}^{n} \mathfrak{l}_i^{z_{k,i}}\right] E$ を計算して，もし $b_k = 1$ であれば，$\mathcal{E}_k := \left[\prod_{i=1}^{n} \mathfrak{l}_i^{z_{k,i}}\right] E_{\mathtt{A}}$ を計算する。ハッシュ値を $b'_1 \| \cdots \| b'_t := H(j(\mathcal{E}_1), \ldots, j(\mathcal{E}_t), \mathtt{msg})$ と bit 分解する。もし，$(b'_1, b'_2, \ldots, b'_t) = (b_1, b_2, \ldots, b_t)$ であれば，受理を出力して，そうでなければ，棄却とする。

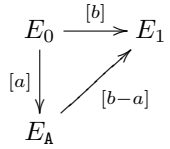
SeaSign 署名方式は，ランダムオラクルモデルにおいて，CSIDH-REGA-CDH 問題困難性の仮定の下で，選択文書攻撃に対して存在的偽造不可（EUF-CMA）安全であることが示されている [58]。

6.2.3.2 CSI-FiSh 署名方式

Beullens–Kleinjung–Vercauteren [16] は，CSIDH-512 パラメータに関するイデアル類群 $\mathrm{cl}(\mathfrak{O})$ の構造計算を遂行することで，効率を高めた CSIDH ベースの CSI-FiSh 署名方式を提案した。CSIDH-512 パラメータでは，512 bits 素数 $p = 4 \cdot \ell_1 \cdot \cdots \cdot \ell_{74} - 1$，$\ell_{74} = 587$ を用いており，Beullens ら [16] は，$\mathbb{Q}(\sqrt{-p})$ のイデアル類群 $\mathrm{cl}(\mathfrak{O})$ が，ノルム 3 のイデアル $\mathfrak{l}_1$ により生成される位数 $N := \#\mathrm{cl}(\mathfrak{O}) = 37 \cdot 1407181 \cdot 51593604295295867744293584889 \cdot 315994145046819958530082787455878322049 09$ の有限巡回群になることを示した。つまり，$[a] \in \mathbb{Z}/N\mathbb{Z}$ に対してイデアル類 $[\mathfrak{l}_1]^a = [\mathfrak{l}_1^a]$ を対応させることで同型 $\mathbb{Z}/N\mathbb{Z} \cong \mathrm{cl}(\mathfrak{O})$ が得られる。更に，イデアル類群の作用 $[\mathfrak{l}_1^a]E$ を同型 $\mathbb{Z}/N\mathbb{Z} \cong \mathrm{cl}(\mathfrak{O})$ を使って $[a]E$ と表し，$\mathbb{Z}/N\mathbb{Z}$ の作用と見なすことにする。これにより 6.1.4.1 節に述べた，より望ましい CSIDH ベースの一方向性群作用 EGA が得られた。

CSI-FiSh 署名方式の構成法は SeaSign 署名方式と変わらないので，Σ-プロトコルにおける記述の差異のみを右図に従って以下に述べる： 位数 N の巡回群 $\mathbb{Z}/N\mathbb{Z}\,(\cong \mathrm{cl}(\mathfrak{O}))$ の作用に基づき，秘密鍵（証拠）を乱数 $a \in \mathbb{Z}/N\mathbb{Z}$，公開鍵 $E_0, E_A := [a]E_0$ とする。証明者は乱数 $b \in \mathbb{Z}/N\mathbb{Z}$ によりコミットメント $E_1 := [b]E_0$ を計算して検証者に送る。検証者はチャレンジ $c \in \{0,1\}$ をランダムに選び証明者に送付，証明者はレスポンス $r := b - ca \bmod N$ を検証者に送る。最後に，検証者は $c = 0$ であれば $E_1 = [r]E_0$ であること，$c = 1$ であれば $E_1 = [r]E_A$ であることが成り立つかどうか検証して検証結果を出力する。

CSI-FiSh 同種写像図式



実際に，CSI-FiSh 署名方式にするには，Fiat–Shamir 変換に則り，チャレンジを $(c_i \in \{0,1\})_{i\in[t]}$ と t 個にするとともに，それらをハッシュ関数を用いて計算して非対話化することで得られる。

上に示したフレームワークをより広範囲の CSIDH パラメータに拡張できれば望ましいが，6.1.4.2 節に述べたように，CSIDH 問題の最近の安全性検討状況によると，NIST 安全性レベル 1 を満たすためには，素数 p を 4096 bits または 5120 bits 程度に大きくする必要性が示されている。そして，それに対応するイデアル類群計算を遂行するのは現状では困難と思われており，このことが，実適用における CSI-FiSh 署名アプローチの限界を示していた。しかし，2023 年に入って，6.2.1.2 節で述べたように，新たな群作用として，SCALLOP [56] が提案されて，EGA パラメータの選択肢を増やすことができるようになり，CSI-FiSh 署名方式の柔軟なパラメータ選択への新しい道が開かれた。その後，実用性を高めるために SCALLOP-HD [36] や PEARL-SCALLOP [5] が提案されたことは 6.2.1.2 節で述べた通りであり，今後の研究に注目する必要がある。

また，Boneh ら [18] により，（ジェネリック）群作用に基づく署名サイズがセキュリティパラメータ λ に対して $\Omega(\lambda^2/\log\lambda)$ となることが示されていることにも注意する。

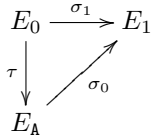
CSI-FiSh 署名提案後に，El Kaafarani ら [53] により，タイト安全な Lossy CSI-FiSh 署名方式の提案もなされた。そして，リング署名・グループ署名など高機能暗号系への拡張研究 [15, 14] があるのが CSI-FiSh 署名方式の利点の一つとなっている。

6.2.4 GPS 署名方式

Galbraith–Petit–Silva（GPS）[72] によって始めて自己準同型環のゼロ知識証明に基づく署名方式が提案された。GPS 署名方式は 1 bit チャレンジ空間のゼロ知識証明プロトコルに基づいているため実際に利用するのは困難であろうと思われているが，現在，GPS 署名方式は，SQIsign 署名方式の原型を与えているという点で重要である。6.1.5 節で述べた Deuring 対応と KLPT アルゴリズム [81] が GPS 署名方式の理論的基礎を与える。

右図において E_0 は 6.1.5.2 節で与えた $j(E_0) = 1728$ なる楕円曲線（特殊極値的楕円曲線）であり，そこで見たようにその E_0 に関しては $\mathrm{End}(E_0)$ の構造が簡明な形で与えられている。その楕円曲線 E_0 からの秘密鍵同種写像 $\tau : E_0 \to E_{\mathtt{A}}$ を知っている証明者（署名生成者）は，$E_{\mathtt{A}}$ から別の楕円曲線 E_1 への同種写像 $\sigma_0 : E_{\mathtt{A}} \to E_1$ と τ との合成 $\sigma_0 \circ \tau : E_0 \to E_1$ を KLPT アルゴリズムに基づいて「ランダム化」して同じ始点 E_0 と終点 E_1 をもつ $\sigma_0 \circ \tau$ とは異なる同種写像 $\sigma_1 : E_0 \to E_1$ を得ることができる。

GPS 同種写像図式



さらに，自己準同型環 $\mathrm{End}(E_{\mathtt{A}})$ を計算する問題の困難性に基づけば，このようなランダム化ができるのは，τ を知っている証明者に限られるので，チャレンジ bit $c \in \{0, 1\}$ を送って証明者に同種写像 σ_c を答えさせることにより，τ に関する知識の有無を検査することができて，認証・署名方式が構成できる。それが GPS 認証方式，そしてその Fiat–Shamir 変換署名が GPS 署名方式である。ここでは [72, 第 4 章] と [84, 5.1.2 節] に基づいて GPS 署名方式を記述する。また，[72, 第 4 章] では，通常の Fiat–Shamir 変換を施した署名方式と Unruh 変換を施した署名方式の 2 方式が記述されているが，ここでは記述の簡便さを考慮して前者の記述を基にして以下に署名方式を与える。

鍵生成： 既知の特殊極値的自己準同型環 $\mathcal{O}_0$ をもつ超特異楕円曲線 E_0 とする。互いに素な B-べき平滑数 S_1, S_2[*6]を，S_1, S_2 次の同種写像グラフ上ランダムウォークがグラフのエクスパンダー性により一様分布を導く程度に十分大きくとる。セキュリティパラメータ λ に対して $t := \lambda$（または $t := 2\lambda$）として，t bits 出力のハッシュ関数 H を選ぶ。$pp_{\mathtt{gps}} := (E_0, S_1, S_2, H)$ を公開パラメータとする。さらに，E_0 を始点とする S_1 次のランダムな同種写像 $\tau : E_0 \to E_{\mathtt{A}}$ を計算して，$pp_{\mathtt{gps}}$ と $E_{\mathtt{A}}$ を公開鍵として，τ を秘密鍵とする。

署名生成： 各 $i = 1, \ldots, t$ に関して $E_{\mathtt{A}}$ を始点とする S_2 次のランダムな同種写像 $\sigma_{0,i} : E_{\mathtt{A}} \to E_{1,i}$ を計算する。署名対象メッセージ $\mathtt{msg}$ に対してチャレンジ bit 列 $h := b_1 \| \cdots \| b_t := H(j(E_{1,1}), \ldots, j(E_{1,t}), \mathtt{msg}) \in \{0, 1\}^t$ をハッシュ関数 H で計算する。各 $i = 1, \ldots, t$ に対して，もし $b_i = 1$ なら KLPT アルゴリズムに基づいて「ランダム化」したランダム同種写像 $\sigma_{1,i} : E_0 \to E_{1,i}$ を計算する。署名を $\sigma := (h, \sigma_{b_1,1}, \ldots, \sigma_{b_t,t})$ とする。

署名検証： 公開鍵 $(pp_{\mathtt{gps}}, E_{\mathtt{A}})$，メッセージ $\mathtt{msg}$ と署名 $\sigma = (h, \sigma_1, \ldots, \sigma_t)$ を入力として，各 $i = 1, \ldots, t$ に対して，同種写像 σ_i を計算して，その終点曲線 $E_{1,i}$ を得る。次に $H(j(E_{1,1}), \ldots, j(E_{1,t}), \mathtt{msg})$ を計算して署名内の h と一致するかどうか検証して，全ての $i = 1, \ldots, t$ に対して検証が成功すれば受理を出力して，そうでなければ，棄却とする。

GPS 署名方式は，超特異楕円曲線同種写像計算問題またはそれと同値な自己準同型環計算問題（定義 6.9）の困難性を仮定すればランダムオラクルモデルの下で EUF-CMA 安全であることが示されている [72, 定理 10]。GPS 署名方式では，1 bit のチャレンジを用いた Σ-プロトコルに基づいているため，署名サイズが大きくなるのが欠点である。また，署名生成で使われた KLPT アルゴリズムの計算時間改善も課題であった [84, 5.1.2 節]。以上，GPS 署名方式には (1) 署名サイズ 及び (2) KLPT アルゴリズム計算時間 に関する 2 つの課題が存在する。

6.3 同種写像に基づく主要な暗号方式

本節では，公開鍵と署名サイズが小さいことを特長にもつ SQIsign 署名方式について述べる（表 6.2 参照）。

[*6] S_k（$k = 1, 2$）が B-べき平滑数（powersmooth number）とは，S_k が $\ell_{k,i}^{e_{k,i}} < B$ なる $\ell_{k,i}^{e_{k,i}}$ の積で表される（つまり，$S_k = \prod_i \ell_{k,i}^{e_{k,i}}$）ことである。ただし $\ell_{k,i}$ は互いに異なるものとする。

表 6.2: 同種写像に基づく暗号の分類

文献	暗号化	鍵交換	署名
SQIsign [61, 34, 8]			○

6.3.1 SQIsign 署名方式

以下，自己準同型環計算問題（定義 6.9）の困難性に安全性の根拠を置く SQIsign 署名方式を概説する。SQIsign 署名方式は公開鍵と署名を合わせたサイズが小さい方式として注目されている。また，2024 年 10 月に，SQIsign 署名方式が NIST PQC 標準化プロジェクト追加署名第 2 ラウンドに進むことが発表された [3]。以下では，KLPT アルゴリズムに基づいた SQIsign 署名方式 [61, 34] のアルゴリズムとパラメータを述べた後，最新の改良版である SQIsign2D 署名方式 [8] について報告する。

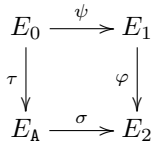
6.3.1.1 KLPT アルゴリズムに基づく SQIsign 署名方式

6.2.4 節で述べた GPS 署名方式を基にして改良を加えた署名方式が SQIsign 署名方式であり，Asiacrypt 2020 で De Feo–Kohel–Leroux–Petit–Wesolowski [61] により提案された。6.2.4 節末尾に付した GPS 署名方式の 2 つの課題を克服している。チャレンジ空間に同種写像の空間を用いることで，そのサイズをセキュリティパラメータ λ まで大きくして，Σ-プロトコルを 1 度適用するだけで十分な Fiat–Shamir 署名構成とした。これで署名サイズが格段に小さくなった。また，GPS 署名生成においては，表 6.1 の Deuring 対応に基づいて，同種写像のイデアル表現（表 6.1 の四元数環側）をねじれ点を使った表現（表 6.1 の楕円曲線側）に変換する部分で時間が費やされていたが，SQIsign 署名方式ではその処理を速度改善したサブルーチン（IdealToIsogeny）に置き換えるのに成功して現実的な演算効率を達成した（詳細は [61, 62] を参照）。

また，安全性に関しては，健全性は超特異平滑自己準同型写像計算問題（定義 6.10）の困難性に基づき，ゼロ知識性は定義 6.11 で述べた SQIsign 署名方式のランダム識別問題の困難性に基づいている。初期提案 [61] では，ノルム方程式を解くサブルーチンに不備があり，生成される署名同種写像 σ に偏りが生じていたことが [62] において指摘された。そして，更に [62] でその不備を除去したアルゴリズム提案が行われた。

■SQIsign 署名アルゴリズム　SQIsign 署名方式では，右図の同種写像 τ が秘密鍵で，超特異楕円曲線 $E_{\mathtt{A}}$ が公開鍵（の主要な一部）である。署名生成では，コミットメント同種写像 ψ とチャレンジ同種写像 φ を適切に生成して得られた合成写像 $\varphi \circ \psi \circ \hat{\tau}$ を一般化された KLPT アルゴリズムに基づいてランダム化した同種写像 σ を署名（Σ-プロトコルのレスポンス）とする。一般化 KLPT アルゴリズムに関しては [34, 2.5.2.2 節] を参照。チャレンジ φ によりセキュリティパラメータ分のランダムネスを与えることができるので，1 度の Σ-プロトコル適用で十分な安全性が達成できる。よって，GPS 署名方式と比べて格段に短い署名サイズが実現できる。

SQIsign 同種写像図式



鍵生成： 既知の特殊極値的自己準同型環 $\mathcal{O}_0$ をもつ超特異楕円曲線 E_0，λ bits の平滑な奇数 D_c（λ はセキュリティパラメータ），超特異 2-同種写像グラフの直径より大きな e による $D := 2^e$ を生成して，$pp_{\mathtt{sqisign}} := (E_0, D_c, D)$ を公開パラメータとする。さらに，E_0 を始点とするランダムな同種写像 $\tau : E_0 \to E_{\mathtt{A}}$ を計算して，$pp_{\mathtt{sqisign}}$ と $E_{\mathtt{A}}$ を公開鍵として，τ を秘密鍵とする。

署名生成： E_0 を始点とするランダムな同種写像 $\psi : E_0 \to E_1$ を計算。署名対象メッセージ $\mathtt{msg}$ に対してハッシュ関数 H で計算した $H(j(E_1), \mathtt{msg})$ から決まる D_c 次の巡回同種写像 $\varphi : E_1 \to E_2$ を計算。同種写像の合成 $\varphi \circ \psi \circ \hat{\tau} : E_{\mathtt{A}} \to E_2$ から（一般化された KLPT アルゴリズムを用いて）同じ始点・終点を有して $\hat{\varphi} \circ \sigma$ が巡回同種写像になる D 次のランダム同種写像 $\sigma : E_{\mathtt{A}} \to E_2$ を計算。(E_1, E_2, σ) を $\mathtt{msg}$ の署名として出力。

署名検証： 公開鍵 $(pp_{\mathtt{sqisign}}, E_{\mathtt{A}})$，メッセージ $\mathtt{msg}$ と署名 (E_1, E_2, σ) を入力として，E_1 から E_2 への同種写像 $\varphi := H(j(E_1), \mathtt{msg})$ を計算する。σ が $E_{\mathtt{A}}$ から E_2 への D 次同種写像であることと $\hat{\varphi} \circ \sigma$ が $E_{\mathtt{A}}$ から E_1 への巡回同種写像であることを検証して，共に成立すれば受理を出力して，そうでなければ，棄却とする。

既に述べたように，SQIsign 署名方式の安全性は，超特異平滑自己準同型写像計算問題（定義 6.10）の困難性と，定義 6.11 で述べた SQIsign 署名 σ のランダム識別問題の困難性に基づいている。また，Santos–Eriksen–Meyer–Reijnders [112] は有限拡大体を活用して署名検証を高速に行う方法を提案している。

■SQIsign 署名パラメータ　署名同種写像 σ の次数は $D = 2^e$，チャレンジ同種写像 φ の次数は平滑な奇数 D_c である。$\mathbb{F}_p$ 上の超特異楕円曲線 E の位数 $p+1$ のねじれ点及びそのツイスト曲線上の位数 $p-1$ のねじれ点を利用して次数 D, D_c の同種写像を小さい拡大次数の有限体で効率的に計算するために，できるだけ大きい正整数 f，正奇数 T に関して $2^f \cdot T \,|\, p^2 - 1$ が満たされる素数 p（SQIsign 素数）を生成することが必要である。具体的には，ある B に対して B-平滑な T，$T \approx p^{5/4+\epsilon}$（[20] では例えば $0.02 < \epsilon < 0.1$ とする）に対して $2^f \cdot T \,|\, p^2 - 1$ となる素数 p を探索する必要がある。SQIsign 素数の選択基準として，署名検証の効率化には f をできるだけ大きくして，署名生成の効率性にとっては $\sqrt{B}/f$ をできるだけ小さくするのが望ましい [62]。

6.3.1.2　SQIsign2D 署名方式

Dartois ら [45] により高次元同種写像を用いて改善を図った SQIsignHD 署名方式が提案された。さらに 2 次元同種写像によってデータサイズ，演算時間，安全性に関して改善された複数の方式が相次いで発表されている [8, 97, 50, 22]。以下では，それらの中で，特に，SQIsign2D-West 署名方式 [8] に関して，[8] で述べられたパラメータ，データサイズ及び性能報告に関して述べる。SQIsign 署名方式を 2 次元同種写像を用いて改善することができたのは Nakagawa–Onuki [96, 97] の貢献が大きい。

特筆すべきは，素数 p の選択である。上に述べたように，従来の SQIsign 署名方式では B-平滑な T を適切に設定する必要があるなど素数 p の選択には限界が伴っていた。しかし，SQIsign2D-West では，できるだけ小さな c により $p + 1 = c \times 2^e$ となる素数 p を用いるため，セキュリティレベルに応じて柔軟なパラメータ選択がしやすい。また，一般化メルセンヌ素数 $p = c \times 2^e - 1$ を用いることで高速実装も可能になり，表 6.3 に示すように，鍵生成・署名生成・署名検証において実用的な実行時間が達成できることが報告されている [8]。

そして，表 6.3 に示されているように，セキュリティパラメータ λ ($\sim \frac{1}{2}\log_2 p$) に対して公開鍵サイズを $4\lambda + 16$ bits，署名サイズを $9\lambda + 16 + 2\log_2(2\lambda)$ bits と小さく抑えることができるのも特長である。

6.4　同種写像に基づく暗号技術に関するまとめ

本章では，同種写像に基づいた暗号技術をまとめた。NIST PQC 標準化プロジェクト追加署名第 2 ラウンドに進んだ SQIsign 署名方式，CSIDH 鍵共有に代表される群作用暗号，ここ数年進展著しいレベル構造付き同種写像問題に基づく鍵共有方式などに関して方式記述と安全性研究についてまとめてきた。また，SIDH 攻撃に端を発した高次元同種写像の暗号応用に関しても調査結果を報告した。

[42] によると，Couveignes は，1997 年の École Normale Supérieure でのセミナーで既に同種写像に基づく暗号技術を提案しており，ほぼ同時期に Kohel [80] や Galbraith [69] も，同種写像問題に関する研究を始めていた。つまり，同種写像暗号技術の研究は既に 27 年の歴史をもつ。そして，最近になり，耐量子計算機暗号の必要性が高まること

表 6.3: SQIsign2D-West 素数パラメータ p 及び公開鍵・署名サイズ（Bytes），Intel Xeon Gold 6338（Ice Lake, 2GHz）上での鍵生成・署名生成・署名検証の実行時間（ms）[8]

NIST 安全性レベル	1	3	5
素数 p	$5 \cdot 2^{248} - 1$	$65 \cdot 2^{376} - 1$	$27 \cdot 2^{500} - 1$
公開鍵サイズ	66	98	130
署名サイズ	148	222	294
鍵生成	30	85	180
署名生成	80	230	470
署名検証	4.5	14.5	31

で，同種写像暗号技術は注目されて研究が進み，NIST PQC 標準化プロジェクト第 4 ラウンドにも選ばれた SIKE 暗号方式及びその基本形である SIDH 鍵共有は，最近まで堅調に安全性評価を積み重ねてきた。しかし，2022 年の Castryck–Decru の攻撃法 [23] を始めとする一連の攻撃法 [87, 106] は SIDH 鍵共有に対して決定的な結果をもたらした。

一方，本章においても随所に見られるように，Kani の補題に基づいて楕円曲線同種写像を高次元同種写像に埋め込むことで，次数が平滑でない同種写像も暗号演算に取り込むことが可能になるなど，SIDH 攻撃法に端を発した全く新しい同種写像暗号研究が現在展開されつつある。例えば，SIDH 攻撃の発案者である Castryck は，“An Attack Became a Tool: Isogeny-based Cryptography 2.0” と題する Eurocrypt 2024 の招待講演において，同種写像暗号研究が今新しい転換点に差し掛かっており，その技術的な核となるのが高次元同種写像の利用であると述べている。更に，6.1.3 節で示したレベル構造付き同種写像問題などの新たな安全性解析の枠組みに関しても研究が進んでおり，そのような理論的基盤に基づいて，新しい方式提案も含む活発な研究活動が引き続いて行われている。

現在，特に，公開鍵と署名を合わせたサイズが小さい SQIsign 署名方式が注目されていると共に，6.1.4 節で見たような CSIDH ベースの一方向性群作用に関する研究も注目されており，種々の暗号プロトコルへの応用も視野に入れた研究も進んでいる。それらも含めて，今後，特に注意すべきこと数点について以下にまとめておく。

- SQIsign 署名方式は，公開鍵と署名のサイズの小ささ，補助点なしの署名構成，そして短署名に対する強い社会的ニーズなどを踏まえると，現在有望な同種写像暗号技術と思われる。その一方，ゼロ知識性に関する計算問題（定義 6.11）の安全性検討などに関して，まだ安全性評価が不十分であり，その安全性評価は今後の重要な課題の一つである。さらに，今後は，実装研究を進める必要もあり，特にさまざまなプラットフォームでの実装結果を蓄えていく必要がある。また，SQIsign2D-West 論文 [8]（Asiacrypt 2024）の副題は “The Fast, the Small, and the Safer” となっており，2 次元同種写像の利用により演算速度，データサイズ，安全性と多方面での改善が図られており，この方向性での今後の研究進展に注目していく必要がある。
- SQIsign 署名方式は NIST PQC 標準化プロジェクト追加署名第 2 ラウンドに進むことが決定しており [3]，SQIsign（及び SQIsign2D）署名パラメータに対して，（一般的な）超特異同種写像問題及びそれと同値な自己準同型環計算問題に対する古典・量子アルゴリズムの詳細な解析・見積もりを行うことが今後の重要な課題である。
- 鍵共有方式として，レベル構造付き同種写像問題に基づく M-SIDH 鍵共有，(Q)FESTA 鍵共有，binSIDH 鍵共有を 6.2.2 節で取り上げた。群作用ベースの鍵共有には，例えば，CSIDH，SCALLOP，SiGamal などがあるが，他にも POKE，IS-CUBE，LIT-SiGamal など新たな鍵共有・暗号方式が提案されてきており，これからも同種写像に基づく鍵共有・暗号方式の安全性解析と方式改良（及び新規提案）は大変重要な課題である。
- 6.2.3 節で述べたリング署名・グループ署名の他にもパスワード認証鍵共有（PAKE）[2, 74] や紛失疑似ランダム関数（OPRF）[110] などといった一方向性群作用の暗号応用に関する研究が進められており，耐量子計算機

性をもつ方式として注目する必要がある。更に，近年では 6.1.4.3 節で見たように量子マネーなどの新しい応用研究も進んでおり，一方向性群作用の新たな暗号応用を探ることも今後の重要な課題の一つである。

- 上で述べたように高次元同種写像を利用した暗号・署名構成，及び安全性解析は，現在も研究が進展し続けている。全体に，同種写像暗号技術は，まだまだ研究の余地があり，鍵・暗号文・署名サイズの小ささの点で他の耐量子計算機暗号にない特長があるので，さまざまな利用用途を見据えて今後も継続的な研究が望まれる。

第 6 章の参照文献

[1] CRYPTREC 暗号技術調査 WG (耐量子計算機暗号). CRYPTREC 耐量子計算機暗号の研究動向調査報告書. CRYPTREC TR-2001-2022, https://www.cryptrec.go.jp/report/cryptrec-tr-2001-2022.pdf. 2023-03.

[2] M. Abdalla, T. Eisenhofer, E. Kiltz, S. Kunzweiler, D. Riepel. Password-Authenticated Key Exchange from Group Actions. CRYPTO (2). Vol. 13508. Lecture Notes in Computer Science. Springer, 2022, pp. 699–728.

[3] G. Alagic et al. Status Report on the First Round of the Additional Digital Signature Schemes for the NIST Post-Quantum Cryptography Standardization Process. NIST IR 8528, https://nvlpubs.nist.gov/nistpubs/ir/2024/NIST.IR.8528.pdf. 2024-10.

[4] N. Alamati, L. De Feo, H. Montgomery, S. Patranabis. Cryptographic Group Actions and Applications. ASIACRYPT (2). Vol. 12492. Lecture Notes in Computer Science. Springer, 2020, pp. 411–439.

[5] B. Allombert, J.-F. Biasse, J. Komada Eriksen, P. Kutas, C. Leonardi, A. Page, R. Scheidler, M. Tot Bagi. PEARL-SCALLOP: Parameter Extension Applicable in Real-Life SCALLOP. Cryptology ePrint Archive, Paper 2024/1744. 2024. https://eprint.iacr.org/2024/1744.

[6] S. Arpin, W. Castryck, J. Komada Eriksen, G. Lorenzon, F. Vercauteren. Generalized class group actions on oriented elliptic curves with level structure. Cryptology ePrint Archive, Paper 2024/1172. 2024. https://eprint.iacr.org/2024/1172. to appear in the proceedings of WAIFI 2024.

[7] A. Basso. POKE: A Framework for Efficient PKEs, Split KEMs, and OPRFs from Higher-dimensional Isogenies. Cryptology ePrint Archive, Paper 2024/624. 2024. https://eprint.iacr.org/2024/624.

[8] A. Basso, L. De Feo, P. Dartois, A. Leroux, L. Maino, G. Pope, D. Robert, B. Wesolowski. SQIsign2D-West - The Fast, the Small, and the Safer. 2024.

[9] A. Basso, T. B. Fouotsa. New SIDH Countermeasures for a More Efficient Key Exchange. ASIACRYPT (8). Vol. 14445. Lecture Notes in Computer Science. Springer, 2023, pp. 208–233.

[10] A. Basso, L. Maino, G. Pope. FESTA: Fast Encryption from Supersingular Torsion Attacks. ASIACRYPT (7). Vol. 14444. Lecture Notes in Computer Science. Springer, 2023, pp. 98–126.

[11] B. Bencina, P. Kutas, S.-P. Merz, C. Petit, M. Stopar, C. Weitkämper. Improved Algorithms for Finding Fixed-Degree Isogenies Between Supersingular Elliptic Curves. CRYPTO (5). Vol. 14924. Lecture Notes in Computer Science. Springer, 2024, pp. 183–217.

[12] D. J. Bernstein, L. De Feo, A. Leroux, B. Smith. Faster computation of isogenies of large prime degree. ANTS 2020. Vol. 4. The Open Book Series 1. Mathematical Sciences Publishers, 2020, pp. 39–55.

[13] D. J. Bernstein, T. Lange, C. Martindale, L. Panny. Quantum Circuits for the CSIDH: Optimizing Quantum Evaluation of Isogenies. EUROCRYPT (2). Vol. 11477. Lecture Notes in Computer Science. Springer, 2019, pp. 409–441.

[14] W. Beullens, S. Dobson, S. Katsumata, Y.-F. Lai, F. Pintore. Group Signatures and More from Isogenies and Lattices: Generic, Simple, and Efficient. EUROCRYPT (2). Vol. 13276. Lecture Notes in Computer Science. Springer, 2022, pp. 95–126.

[15] W. Beullens, S. Katsumata, F. Pintore. Calamari and Falafl: Logarithmic (Linkable) Ring Signatures from Isogenies and Lattices. ASIACRYPT (2). Vol. 12492. Lecture Notes in Computer Science. Springer, 2020, pp. 464–492.

[16] W. Beullens, T. Kleinjung, F. Vercauteren. CSI-FiSh: Efficient Isogeny Based Signatures Through Class Group Computations. ASIACRYPT (1). Vol. 11921. Lecture Notes in Computer Science. Springer, 2019, pp. 227–247.

[17] J.-F. Biasse, D. Jao, A. Sankar. A Quantum Algorithm for Computing Isogenies between Supersingular Elliptic Curves. INDOCRYPT. Vol. 8885. Lecture Notes in Computer Science. Springer, 2014, pp. 428–442.

[18] D. Boneh, J. Guan, M. Zhandry. A Lower Bound on the Length of Signatures Based on Group Actions and Generic Isogenies. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 507–531.

[19] X. Bonnetain, A. Schrottenloher. Quantum Security Analysis of CSIDH. EUROCRYPT (2). Vol. 12106. Lecture Notes in Computer Science. Springer, 2020, pp. 493–522.

[20] G. Bruno, M. Corte-Real Santos, C. Costello, J. Komada Eriksen, M. Meyer, M. Naehrig, B. Sterner. Cryptographic Smooth Neighbors. Cryptology ePrint Archive, Paper 2022/1439. 2022. `https://eprint.iacr.org/2022/1439`.

[21] F. Campos, J. Chávez-Saab, J.-J. Chi-Domínguez, M. Meyer, K. Reijnders, F. Rodríguez-Henríquez, P. Schwabe, T. Wiggers. Optimizations and Practicality of High-Security CSIDH. IACR Commun. Cryptol. Vol. 1, Num. 1 (2024), p. 5.

[22] W. Castryck, M. Chen, R. Invernizzi, G. Lorenzon, F. Vercauteren. Breaking and Repairing SQIsign2D-East. Cryptology ePrint Archive, Paper 2024/1453. 2024. `https://eprint.iacr.org/2024/1453`. to appear in the proceedings of ASIACRYPT 2024 merging with [97].

[23] W. Castryck, T. Decru. An Efficient Key Recovery Attack on SIDH. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 423–447.

[24] W. Castryck, T. Decru, M. Houben, F. Vercauteren. Horizontal Racewalking Using Radical Isogenies. ASIACRYPT (2). Vol. 13792. Lecture Notes in Computer Science. Springer, 2022, pp. 67–96.

[25] W. Castryck, T. Decru, B. Smith. Hash functions from superspecial genus-2 curves using Richelot isogenies. J. Math. Cryptol. Vol. 14, Num. 1 (2020), pp. 268–292.

[26] W. Castryck, T. Decru, F. Vercauteren. Radical Isogenies. ASIACRYPT (2). Vol. 12492. Lecture Notes in Computer Science. Springer, 2020, pp. 493–519.

[27] W. Castryck, M. Houben, S.-P. Merz, M. Mula, S. van Buuren, F. Vercauteren. Weak Instances of Class Group Action Based Cryptography via Self-pairings. CRYPTO (3). Vol. 14083. Lecture Notes in Computer Science. Springer, 2023, pp. 762–792.

[28] W. Castryck, M. Houben, F. Vercauteren, B. Wesolowski. On the decisional Diffie–Hellman problem for class group actions on oriented elliptic curves. ANTS 2022. Vol. 8. Research in Number Theory 99. Springer, 2022, pp. 39–55.

[29] W. Castryck, T. Lange, C. Martindale, L. Panny, J. Renes. CSIDH: An Efficient Post-Quantum Commutative Group Action. ASIACRYPT (3). Vol. 11274. Lecture Notes in Computer Science. Springer, 2018, pp. 395–427.

[30] W. Castryck, N. Vander Meeren. Two Remarks on the Vectorization Problem. INDOCRYPT. Vol. 13774. Lecture Notes in Computer Science. Springer, 2022, pp. 658–678.

[31] W. Castryck, J. Sotáková, F. Vercauteren. Breaking the Decisional Diffie-Hellman Problem for Class Group Actions Using Genus Theory: Extended Version. J. Cryptol. Vol. 35, Num. 4 (2022), p. 24.

[32] W. Castryck, F. Vercauteren. A Polynomial Time Attack on Instances of M-SIDH and FESTA. ASIACRYPT (7). Vol. 14444. Lecture Notes in Computer Science. Springer, 2023, pp. 127–156.

[33] D. X. Charles, K. E. Lauter, E. Z. Goren. Cryptographic Hash Functions from Expander Graphs. J. Cryptol. Vol. 22, Num. 1 (2009), pp. 93–113.

[34] J. Chavez-Saab et al. SQIsign: Algorithm specifications and supporting documentation. submission to the NIST's PQC standardization. (2023).

[35] J. Chávez-Saab, J.-J. Chi-Domínguez, S. Jaques, F. Rodríguez-Henríquez. The SQALE of CSIDH: sublinear Vélu quantum-resistant isogeny action with low exponents. J. Cryptogr. Eng. Vol. 12, Num. 3 (2022), pp. 349–368.

[36] M. Chen, A. Leroux, L. Panny. SCALLOP-HD: Group Action from 2-Dimensional Isogenies. Public Key Cryptography (3). Vol. 14603. Lecture Notes in Computer Science. Springer, 2024, pp. 190–216.

[37] A. M. Childs, D. Jao, V. Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. J. Math. Cryptol. Vol. 8, Num. 1 (2014), pp. 1–29.

[38] L. Colò, D. Kohel. Orienting supersingular isogeny graphs. J. Math. Cryptol. Vol. 14, Num. 1 (2020), pp. 414–437.

[39] C. Costello. B-SIDH: Supersingular Isogeny Diffie-Hellman Using Twisted Torsion. ASIACRYPT (2). Vol. 12492. Lecture Notes in Computer Science. Springer, 2020, pp. 440–463.

[40] C. Costello. The Case for SIKE: A Decade of the Supersingular Isogeny Problem. Cryptology ePrint Archive, Paper 2021/543. 2021. https://eprint.iacr.org/2021/543.

[41] C. Costello, P. Longa, M. Naehrig, J. Renes, F. Virdia. Improved Classical Cryptanalysis of SIKE in Practice. Public Key Cryptography (2). Vol. 12111. Lecture Notes in Computer Science. Springer, 2020, pp. 505–534.

[42] J.-M. Couveignes. Hard Homogeneous Spaces. Cryptology ePrint Archive, Paper 2006/291. 2006. https://eprint.iacr.org/2006/291.

[43] P. Dartois. Fast computation of 2-isogenies in dimension 4 and cryptographic applications. Cryptology ePrint Archive, Paper 2024/1180. 2024. https://eprint.iacr.org/2024/1180.

[44] P. Dartois, L. De Feo. On the Security of OSIDH. Public Key Cryptography (1). Vol. 13177. Lecture Notes in Computer Science. Springer, 2022, pp. 52–81.

[45] P. Dartois, A. Leroux, D. Robert, B. Wesolowski. SQIsignHD: New Dimensions in Cryptography. EUROCRYPT (1). Vol. 14651. Lecture Notes in Computer Science. Springer, 2024, pp. 3–32.

[46] P. Dartois, L. Maino, G. Pope, D. Robert. An Algorithmic Approach to $(2, 2)$-isogenies in the Theta Model and Applications to Isogeny-based Cryptography. Cryptology ePrint Archive, Paper 2023/1747. 2023. https://eprint.iacr.org/2023/1747.

[47] T. Decru. Radical $\sqrt[N]{\text{élu}}$ Isogeny Formulae. CRYPTO (5). Vol. 14924. Lecture Notes in Computer Science. Springer, 2024, pp. 107–128.

[48] T. Decru, L. Panny, F. Vercauteren. Faster SeaSign Signatures Through Improved Rejection Sampling. PQCrypto. Vol. 11505. Lecture Notes in Computer Science. Springer, 2019, pp. 271–285.

[49] C. Delfs, S. D. Galbraith. Computing isogenies between supersingular elliptic curves over F_p. Des. Codes Cryptogr. Vol. 78, Num. 2 (2016), pp. 425–440.

[50] M. Duparc, T. B. Fouotsa. SQIPrime: A Dimension 2 Variant of SQISignHD with Non-smooth Challenge Isogenies. 2024.

[51] K. Eisenträger, S. Hallgren, K. E. Lauter, T. Morrison, C. Petit. Supersingular Isogeny Graphs and Endomorphism Rings: Reductions and Solutions. EUROCRYPT (3). Vol. 10822. Lecture Notes in Computer Science. Springer, 2018, pp. 329–368.

[52] K. Eisenträger, S. Hallgren, C. Leonardi, T. Morrison, J. Park. Computing endomorphism rings of supersingular elliptic curves and connections to pathfinding in isogeny graphs. ANTS 2020. Vol. 4. The Open Book Series 1. Mathematical Sciences Publishers, 2020, pp. 215–232.

[53] A. El Kaafarani, S. Katsumata, F. Pintore. Lossy CSI-FiSh: Efficient Signature Scheme with Tight Reduction to Decisional CSIDH-512. Public Key Cryptography (2). Vol. 12111. Lecture Notes in Computer Science. Springer, 2020, pp. 157–186.

[54] J. Komada Eriksen, L. Panny, J. Sotáková, M. Veroni. Deuring for the People: Supersingular Elliptic Curves with Prescribed Endomorphism Ring in General Characteristic. LuCaNT: LMFDB, Computation, and Number Theory. Vol. 796. Contemporary Mathematics. AMS, 2024. `https://www.ams.org/books/conm/796/16008/conm796-16008.pdf`.

[55] L. De Feo. Mathematics of Isogeny Based Cryptography. 2017. arXiv: `1711.04062`.

[56] L. De Feo, T. B. Fouotsa, P. Kutas, A. Leroux, S.-P. Merz, L. Panny, B. Wesolowski. SCALLOP: Scaling the CSI-FiSh. Public Key Cryptography (1). Vol. 13940. Lecture Notes in Computer Science. Springer, 2023, pp. 345–375.

[57] L. De Feo, T. B. Fouotsa, L. Panny. Isogeny Problems with Level Structure. EUROCRYPT (6). Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 181–204.

[58] L. De Feo, S. D. Galbraith. SeaSign: Compact Isogeny Signatures from Class Group Actions. EUROCRYPT (3). Vol. 11478. Lecture Notes in Computer Science. Springer, 2019, pp. 759–789.

[59] L. De Feo, D. Jao, J. Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. J. Math. Cryptol. Vol. 8, Num. 3 (2014), pp. 209–247.

[60] L. De Feo, J. Kieffer, B. Smith. Towards Practical Key Exchange from Ordinary Isogeny Graphs. ASIACRYPT (3). Vol. 11274. Lecture Notes in Computer Science. Springer, 2018, pp. 365–394.

[61] L. De Feo, D. Kohel, A. Leroux, C. Petit, B. Wesolowski. SQISign: Compact Post-quantum Signatures from Quaternions and Isogenies. ASIACRYPT (1). Vol. 12491. Lecture Notes in Computer Science. Springer, 2020, pp. 64–93.

[62] L. De Feo, A. Leroux, P. Longa, B. Wesolowski. New Algorithms for the Deuring Correspondence – Towards Practical and Secure SQISign Signatures. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 659–690.

[63] L. De Feo, C. D. de Saint Guilhem, T. B. Fouotsa, P. Kutas, A. Leroux, C. Petit, J. Silva, B. Wesolowski. Séta: Supersingular Encryption from Torsion Attacks. ASIACRYPT (4). Vol. 13093. Lecture Notes in Computer Science. Springer, 2021, pp. 249–278.

[64] E. Florit, B. Smith. An atlas of the Richelot isogeny graph. RIMS Kôkyûroku Bessatsu. Vol. B90 (2022), pp. 195–219. https://repository.kulib.kyoto-u.ac.jp/dspace/handle/2433/276282.

[65] T. B. Fouotsa, T. Moriya, C. Petit. M-SIDH and MD-SIDH: Countering SIDH Attacks by Masking Information. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 282–309.

[66] T. B. Fouotsa, C. Petit. SimS: A Simplification of SiGamal. PQCrypto. Vol. 12841. Lecture Notes in Computer Science. Springer, 2021, pp. 277–295.

[67] J. Fuselier, A. Iezzi, M. Kozek, T. Morrison, C. Namoijam. Computing supersingular endomorphism rings using inseparable endomorphisms. 2023. arXiv: 2306.03051.

[68] S. Galbraith, L. Panny, B. Smith, F. Vercauteren. Quantum equivalence of the DLP and CDHP for group actions. Mathematical Cryptology. Vol. 1, Num. 1 (2021), pp. 40–44. https://journals.flvc.org/mathcryptology/article/view/122741.

[69] S. D. Galbraith. Constructing Isogenies between Elliptic Curves Over Finite Fields. LMS Journal of Computation and Mathematics. Vol. 2 (1999), pp. 118–138.

[70] S. D. Galbraith, Y.-F. Lai, H. Montgomery. A Simpler and More Efficient Reduction of DLog to CDH for Abelian Group Actions. Public Key Cryptography (3). Vol. 14603. Lecture Notes in Computer Science. Springer, 2024, pp. 36–60.

[71] S. D. Galbraith, D. Perrin, J. F. Voloch. CSIDH with Level Structure. Cryptology ePrint Archive, Paper 2023/1726. 2023. https://eprint.iacr.org/2023/1726.

[72] S. D. Galbraith, C. Petit, J. Silva. Identification Protocols and Signature Schemes Based on Supersingular Isogeny Problems. J. Cryptol. Vol. 33, Num. 1 (2020), pp. 130–175.

[73] S. D. Galbraith, F. Vercauteren. Computational problems in supersingular elliptic curve isogenies. Quantum Inf. Process. Vol. 17, Num. 10 (2018), p. 265.

[74] R. Ishibashi, K. Yoneyama. Compact Password Authenticated Key Exchange from Group Actions. ACISP. Vol. 13915. Lecture Notes in Computer Science. Springer, 2023, pp. 220–247.

[75] D. Jao et al. Supersingular Isogeny Key Encapsulation. https://sike.org/files/SIDH-spec.pdf. 2022-09. (2024-11-12 閲覧).

[76] S. Jaques, J. M. Schanck. Quantum Cryptanalysis in the RAM Model: Claw-Finding Attacks on SIKE. CRYPTO (1). Vol. 11692. Lecture Notes in Computer Science. Springer, 2019, pp. 32–61.

[77] Y. Kambe, A. Katayama, Y. Aikawa, Y. Ishihara, M. Yasuda, K. Yokoyama. Computing Endomorphism Rings of Supersingular Elliptic Curves by Finding Cycles in Concatenated Supersingular Isogeny Graphs. Commentarii Mathematici Universitatis Sancti Pauli. Vol. 72, Num. 1 (2024), pp. 19–42.

[78] Y. Kambe, Y. Takahashi, M. Yasuda, K. Yokoyama. On the feasibility of computing constructive Deuring correspondence. NuTMiC 2021. Vol. 126. Banach Center Publications. Institute of Mathematics, Polish Academy od Sciences, 2023. https://www.impan.pl/en/publishing-house/banach-center-publications/all/126/0/115356/on-the-feasibility-of-computing-constructive-deuring-correspondence.

[79] T. Katsura, K. Takashima. Counting Richelot isogenies between superspecial abelian surfaces. ANTS 2020. Vol. 4. The Open Book Series 1. Mathematical Sciences Publishers, 2020, pp. 283–300.

[80] D. Kohel. Endomorphism rings of elliptic curves over finite fields. PhD thesis. University of California at Berkeley, 1996.

[81] D. Kohel, K. Lauter, C. Petit, J.-P. Tignol. On the quaternion ℓ-isogeny path problem. LMS Journal of Computation and Mathematics. Vol. 17 (2014), pp. 418–432. Special Issue A: Algorithmic Number Theory Symposium XI.

[82] G. Kuperberg. A Subexponential-Time Quantum Algorithm for the Dihedral Hidden Subgroup Problem. SIAM J. Comput. Vol. 35, Num. 1 (2005), pp. 170–188.

[83] G. Kuperberg. Another Subexponential-time Quantum Algorithm for the Dihedral Hidden Subgroup Problem. 8th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2013, May 21-23, 2013, Guelph, Canada. Ed. by S. Severini, F. G. S. L. Brandão. Vol. 22. LIPIcs. 2013, pp. 20–34.

[84] A. Leroux. Quaternion algebras and isogeny-based cryptography. PhD thesis. Ecole Polytechnique, 2022.

[85] J. Liu, H. Montgomery, M. Zhandry. Another Round of Breaking and Making Quantum Money: How to Not Build It from Lattices, and More. EUROCRYPT (1). Vol. 14004. Lecture Notes in Computer Science. Springer, 2023, pp. 611–638.

[86] P. Longa, W. Wang, J. Szefer. The Cost to Break SIKE: A Comparative Hardware-Based Analysis with AES and SHA-3. CRYPTO (3). Vol. 12827. Lecture Notes in Computer Science. Springer, 2021, pp. 402–431.

[87] L. Maino, C. Martindale, L. Panny, G. Pope, B. Wesolowski. A Direct Key Recovery Attack on SIDH. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 448–471.

[88] A. Herlédan Le Merdy, B. Wesolowski. The supersingular endomorphism ring problem given one endomorphism. Cryptology ePrint Archive, Paper 2023/1448. 2023. `https://eprint.iacr.org/2023/1448`.

[89] M. Meyer, S. Reith. A Faster Way to the CSIDH. INDOCRYPT. Vol. 11356. Lecture Notes in Computer Science. Springer, 2018, pp. 137–152.

[90] H. Montgomery, S. Sharif. Quantum Money from Class Group Actions on Elliptic Curves. 2024.

[91] H. Montgomery, M. Zhandry. Full Quantum Equivalence of Group Action DLog and CDH, and More. ASIACRYPT (1). Vol. 13791. Lecture Notes in Computer Science. Springer, 2022, pp. 3–32.

[92] T. Moriya. IS-CUBE: An isogeny-based compact KEM using a boxed SIDH diagram. Cryptology ePrint Archive, Paper 2023/1506. 2023. `https://eprint.iacr.org/2023/1506`.

[93] T. Moriya. LIT-SiGamal: An efficient isogeny-based PKE based on a LIT diagram. Cryptology ePrint Archive, Paper 2024/521. 2024. `https://eprint.iacr.org/2024/521`.

[94] T. Moriya, H. Onuki, T. Takagi. SiGamal: A Supersingular Isogeny-Based PKE and Its Application to a PRF. ASIACRYPT (2). Vol. 12492. Lecture Notes in Computer Science. Springer, 2020, pp. 551–580.

[95] S. Mutreja, M. Zhandry. Quantum State Group Actions. Cryptology ePrint Archive, Paper 2024/1636. 2024. `https://eprint.iacr.org/2024/1636`.

[96] K. Nakagawa, H. Onuki. QFESTA: Efficient Algorithms and Parameters for FESTA Using Quaternion Algebras. CRYPTO (5). Vol. 14924. Lecture Notes in Computer Science. Springer, 2024, pp. 75–106.

[97] K. Nakagawa, H. Onuki. SQIsign2D-East: A New Signature Scheme Using 2-dimensional Isogenies. Cryptology ePrint Archive, Paper 2024/771. 2024. https://eprint.iacr.org/2024/771. to appear in the proceedings of ASIACRYPT 2024 merging with [22].

[98] H. Onuki, Y. Aikawa, T. Yamazaki, T. Takagi. A Constant-Time Algorithm of CSIDH Keeping Two Points. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. Vol. 103-A, Num. 10 (2020), pp. 1174–1182.

[99] R. Oudompheng, G. Pope. A Note on Reimplementing the Castryck–Decru Attack and Lessons Learned for SageMath. Cryptology ePrint Archive, Paper 2022/1283. 2022. https://eprint.iacr.org/2022/1283.

[100] A. Page, B. Wesolowski. The Supersingular Endomorphism Ring and One Endomorphism Problems are Equivalent. EUROCRYPT (6). Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 388–417.

[101] C. Peikert. He Gives C-Sieves on the CSIDH. EUROCRYPT (2). Vol. 12106. Lecture Notes in Computer Science. Springer, 2020, pp. 463–492.

[102] C. Petit. Faster Algorithms for Isogeny Problems Using Torsion Point Images. ASIACRYPT (2). Vol. 10625. Lecture Notes in Computer Science. Springer, 2017, pp. 330–353.

[103] V. de Quehen, P. Kutas, C. Leonardi, C. Martindale, L. Panny, C. Petit, K. E. Stange. Improved Torsion-Point Attacks on SIDH Variants. CRYPTO (3). Vol. 12827. Lecture Notes in Computer Science. Springer, 2021, pp. 432–470.

[104] O. Regev. A Subexponential Time Algorithm for the Dihedral Hidden Subgroup Problem with Polynomial Space. 2004. arXiv: quant-ph/0406151.

[105] J. Renes. Computing Isogenies Between Montgomery Curves Using the Action of $(0, 0)$. PQCrypto. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 229–247.

[106] D. Robert. Breaking SIDH in Polynomial Time. EUROCRYPT (5). Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 472–503.

[107] D. Robert. On the efficient representation of isogenies (a survey). Cryptology ePrint Archive, Paper 2024/1071. 2024. https://eprint.iacr.org/2024/1071.

[108] D. Robert. The module action for isogeny based cryptography. Cryptology ePrint Archive, Paper 2024/1556. 2024. https://eprint.iacr.org/2024/1556.

[109] A. Rostovtsev, A. Stolbunov. Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Paper 2006/145. 2006. https://eprint.iacr.org/2006/145.

[110] C. D. de Saint Guilhem, R. Pedersen. New Proof Systems and an OPRF from CSIDH. Public Key Cryptography (3). Vol. 14603. Lecture Notes in Computer Science. Springer, 2024, pp. 217–251.

[111] M. Corte-Real Santos, C. Costello, J. Shi. Accelerating the Delfs-Galbraith Algorithm with Fast Subfield Root Detection. CRYPTO (3). Vol. 13509. Lecture Notes in Computer Science. Springer, 2022, pp. 285–314.

[112] M. Corte-Real Santos, J. Komada Eriksen, M. Meyer, K. Reijnders. AprèsSQI: Extra Fast Verification for SQIsign Using Extension-Field Signing. EUROCRYPT (1). Vol. 14651. Lecture Notes in Computer Science. Springer, 2024, pp. 63–93.

[113] B. Smith. Pre- and Post-quantum Diffie-Hellman from Groups, Actions, and Isogenies. WAIFI. Vol. 11321. Lecture Notes in Computer Science. Springer, 2018, pp. 3–40.

[114] K. Takashima. Efficient Algorithms for Isogeny Sequences and Their Cryptographic Applications. *CREST Crypto-Math Project*. Mathematics for Industry. Springer Singapore, 2017, pp. 97–114.

[115] A. Udovenko, G. Vitto. Revisiting Meet-in-the-Middle Cryptanalysis of SIDH/SIKE with Application to the $IKEp182 Challenge. Cryptology ePrint Archive, Paper 2021/1421. 2021. https://eprint.iacr.org/2021/1421.

[116] J. Vélu. Isogénies entre courbes elliptiques. C. R. Acad. Sci. Paris, Sér. A. Vol. 273 (1971), pp. 305–347.

[117] J. Voight. Quaternion algebras. Springer International Publishing, 2021-06.

[118] L. C. Washington. Elliptic curves : number theory and cryptography. 2nd ed. Discrete mathematics and its applications. Chapman & Hall/CRC, 2008.

[119] B. Wesolowski. The supersingular isogeny path and endomorphism ring problems are equivalent. FOCS. IEEE, 2021, pp. 1100–1111.

[120] M. Zhandry. Quantum Money from Abelian Group Actions. ITCS. Vol. 287. LIPIcs. 2024, 101:1–101:23.

[121] 相川 勇輔, 神戸 祐太, 工藤 桃成, 高島 克幸, 安田 雅哉. 代数曲線の計算理論と暗号への応用. 数学メモアール 10. 日本数学会, 2024.

第 7 章

ハッシュ関数に基づく署名技術

本章ではハッシュ関数に基づく署名技術についてまとめる。ハッシュ関数に基づく署名技術の安全性はハッシュ関数の第二原像攻撃に対する安全性に依存している。

ハッシュ関数に基づく署名技術は，最初に Lamport により one-time signature として提案された [15, 31]。また，この方式を改良した Winternitz one-time signature が Merkle [35] により述べられている。これらの方式は一組の公開鍵と秘密鍵を用いて一つのメッセージに署名を行う 1 回署名方式である。1 回署名方式とマークル木とを用いて複数回署名を行うことを可能とする方式が Merkle [34, 35] により述べられている。

7.1 ハッシュ関数に基づく署名技術の安全性の根拠となる問題

ハッシュ関数は任意長あるいは実用上十分な長さ以下の入力 $\{0,1\}$ 系列に対して固定長の $\{0,1\}$ 系列を出力する関数である。ハッシュ関数を $H : \mathcal{D} \to \mathcal{R}$ とする。ここで，$\mathcal{D}$ は任意長の $\{0,1\}$ 系列の集合 $\{0,1\}^*$ の部分集合であり，$\mathcal{R}$ は固定長の $\{0,1\}$ 系列の集合である。ハッシュ関数の第二原像攻撃は，第一原像 $X \in \mathcal{D}$ が与えられたとき，$X \neq X'$ かつ $H(X) = H(X')$ を満たす第二原像 $X' \in \mathcal{D}$ を求めるという問題を解くことを目的とする攻撃である。なお，第二原像攻撃に対する安全性は，しばしばハッシュ関数の各入力に対する出力がランダムであると仮定して評価される。このようなランダム関数はランダムオラクルとも呼ばれる。H がランダムオラクルであるとき，第二原像の計算時間は $\Theta(|\mathcal{R}|)$ である。また，量子コンピュータでは，Grover の探索アルゴリズム [20] を用いることにより，第二原像の計算時間は $\Theta(\sqrt{|\mathcal{R}|})$ となる。

本章で取り上げるハッシュ関数に基づく署名技術では, 米国 NIST の指定する標準ハッシュ関数族である SHA-2 [38], SHA-3 [39] のうちのいくつかのハッシュ関数を用いることが想定されている。

SHA-2 は固定長入出力の圧縮関数からなる Merkle-Damgård 構造 [14, 36] を有するハッシュ関数の族であり, Secure Hash Standard [38] のうち，SHA-1 を除く SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256 からなる。SHA-2 の各ハッシュ関数の名称の末尾の数値は出力の bit 長を表す。SHA-3 は固定長入出力の置換を用いたスポンジ構造 [9] を有するハッシュ関数の族であり，SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256 からなる。SHA3-224, SHA3-256, SHA3-384, SHA3-512 については，末尾の数値は出力の bit 長を表す。なお，出力長は内部状態のうちスポンジ構造を有するハッシュ関数の安全性に大きく関わるキャパシティと呼ばれる部分の bit 長の半分である。SHAKE128,SHAKE256 については，出力長は任意に設定できる。なお，末尾の数値はキャパシティの bit 長の半分である。SHA-2, SHA-3 の出力長および安全性の一覧を表 7.1 に示す。この表では，第二原像攻撃に対する安全性のみでなく，それとともにハッシュ関数の主な安全性要件である衝突攻撃，原像攻撃に対する安全性も示されている。この表で，攻撃に対する安全性を表す数値 μ は bit 安全性と呼ばれ，攻撃に成功するために必要な計算時間（ハッシュ関数を構成する圧縮関数あるいは置換の計算回数）がおよそ 2^{μ} であることを示している。なお，SHA-224, SHA-256, SHA-512 の第二原像攻撃に対する安全性は Kelsey と Schneier により提案された攻撃 [29] に基

づいて評価されており，$L(m) := \lceil \log_2(m/B) \rceil$ である。ここで，m は第一原像の bit 長であり，SHA-224, SHA-256 については $B = 512$，SHA-512 については $B = 1024$ である。

表 7.1: SHA-2, SHA-3 の安全性 [39]

ハッシュ関数	出力長	攻撃に対する安全性（ビットセキュリティ）		
		衝突攻撃	原像攻撃	第二原像攻撃
SHA-224	224	112	224	$\min\{224, 256 - L(m)\}$
SHA-512/224	224	112	224	224
SHA-256	256	128	256	$256 - L(m)$
SHA-512/256	256	128	256	256
SHA-384	384	192	384	384
SHA-512	512	256	512	$512 - L(m)$
SHA3-224	224	112	224	224
SHA3-256	256	128	256	256
SHA3-384	384	192	384	384
SHA3-512	512	256	512	512
SHAKE128	d	$\min\{d/2, 128\}$	$\min\{d, 128\}$ 以上	$\min\{d, 128\}$
SHAKE256	d	$\min\{d/2, 256\}$	$\min\{d, 256\}$ 以上	$\min\{d, 256\}$

本章で使用する記号・用語を以下にまとめる。

- $\{0,1\}$ 系列 α, β の連接を $\alpha \| \beta$ と表記する。
- $\ll$ は左論理シフトを表す。
- 整数 ν について $[\nu]_l$ は ν の長さ l Bytes の 2 進数表記を表す。
- $\mathbb{B} := \{0,1\}^8$ とする。
- `8080` のように typewriter font で書かれている数字は 16 進数として解釈する。

7.2 ハッシュ関数に基づく代表的な署名方式

7.2.1 Winternitz One-Time Signature

Winternitz one-time signature [35] は，一組の公開鍵と秘密鍵を用いて一つのメッセージに署名を行う 1 回署名方式である。この方式では，署名対象のメッセージのハッシュ値 N を b 進数表記の整数とみなす。N が ℓ_{m} 桁の b 進数 $N_{\ell_{\mathrm{m}}-1} N_{\ell_{\mathrm{m}}-2} \cdots N_1 N_0$ で表記されるとする。このとき，$0 \leq k \leq \ell_{\mathrm{m}} - 1$ について $N_k \in \{0, 1, \ldots, b-1\}$ であり，$N = \sum_{k=0}^{\ell_{\mathrm{m}}-1} N_k 2^k$ である。さらに，N のチェックサムを $C := \sum_{k=0}^{\ell_{\mathrm{m}}-1} (b - 1 - N_k)$ と定義する。C が ℓ_{c} 桁の b 進数 $N_{\ell_{\mathrm{m}}+\ell_{\mathrm{c}}-1} N_{\ell_{\mathrm{m}}+\ell_{\mathrm{c}}-2} \cdots N_{\ell_{\mathrm{m}}+1} N_{\ell_{\mathrm{m}}}$ で表記されるとする。$\ell := \ell_{\mathrm{m}} + \ell_{\mathrm{c}}$ とする。

■鍵生成アルゴリズム　秘密鍵 $(x_0, x_1, \ldots, x_{\ell-1})$，公開鍵 $(pub_0, pub_1, \ldots, pub_{\ell-1})$ は以下のように生成される。

1. 一様ランダムに $x_0, x_1, \ldots, x_{\ell-1} \in \mathcal{D}$ を取る。
2. $0 \leq k \leq \ell - 1$ について $pub_k := H^{b-1}(x_k) := \underbrace{H(H(\cdots(H}_{b-1 \text{ times}}(x_k))\cdots))$ とする。

■署名アルゴリズム　メッセージのハッシュ値 N の署名 $(s_0, s_1, \ldots, s_{\ell-1})$ は以下のように生成される。

1. $0 \leq k \leq \ell - 1$ について $s_k := H^{N_k}(x_k)$ とする。

■検証アルゴリズム　メッセージのハッシュ値 N とその署名 $(s_0, s_1, \ldots, s_{\ell-1})$ の検証は以下のように行われる。

1. $0 \leq k \leq \ell - 1$ について $pub_k = H^{b-1-N_k}(s_k)$ かつそのときに限り，$(s_0, s_1, \ldots, s_{\ell-1})$ は N の正しい署名である。

仮にチェックサムが導入されていないとすると，N の署名 $(s_0, s_1, \ldots, s_{\ell_{\mathrm{m}}-1})$ が得られたとき，$0 \leq k \leq \ell_{\mathrm{m}} - 1$ について $N'_k \geq N_k$ を満たす N' について，$s'_k := H^{N'_k - N_k}(s_k)$ によって，署名 $(s'_0, s'_1, \ldots, s'_{\ell_{\mathrm{m}}-1})$ が容易に偽造できる。

Winternitz one-time signature の偽造不能性は，Dods ら [16] により論じられている。Winternitz one-time signature に基づく方式については，Lafrance と Menezes [30] によりまとめられている。

7.2.2　マークル木を用いた署名方式

1 回署名方式を用いて複数のメッセージに署名を行う場合，メッセージの個数と同じ個数の公開鍵と秘密鍵の組が必要となる。マークル木を用いることにより，このような複数回署名方式の公開鍵の大きさを削減できる [34]。

2^h 個のメッセージに署名を行うための 1 回署名の公開鍵を $pk_0, pk_1, \ldots, pk_{2^h-1}$ とする。このとき，高さが h，すなわち，葉の個数が 2^h のマークル木は以下のように構成される。高さ $j(\geq 0)$ の左から $i(\geq 0)$ 番目の節点を $v_{i,j}$ と表記する。$v_{i,j}$ は以下のように計算される。

1. $0 \leq i \leq 2^h - 1$ について，$v_{i,0} := H(pk_i)$ とする。
2. $1 \leq j \leq h$ に対し，$0 \leq i \leq 2^{h-j} - 1$ について，$v_{i,j} := H(v_{2i,j-1} \| v_{2i+1,j-1})$ とする。

この署名方式の公開鍵は $v_{0,h}$ である。秘密鍵は 1 回署名の公開鍵 $pk_0, pk_1, \ldots, pk_{2^h-1}$ に対応するすべての秘密鍵である。i 個目のメッセージの署名を検証するためには，$v_{0,h}$ を用いて pk_i が正しいことを検証する必要がある。このために，i 個目のメッセージの署名には，マークル木の $v_{i,0}$ から $v_{0,h}$ に至る経路上の各節点の，経路上にない子節点が含まれる。これらの節点の列は認証パスと呼ばれる。

7.2.3　マークル木の階層構造による署名方式

前節で述べた一つのマークル木を用いた署名方式では，鍵生成時にすべての 1 回署名の公開鍵と秘密鍵を生成する必要があり，例えば，2^{50} 個の署名を行うために高さ 50 のマークル木を構成することは，所要計算時間の観点から非実用的である。このような多数のメッセージに署名を行う際には，マークル木を用いた署名方式の階層構造による署名方式が提案されている [24]。

この署名方式で構成されるマークル木を用いた署名方式の階層構造の階層数を L とし，根に相当する最上層を第 $(L-1)$ 層，葉に相当する最下層を第 0 層とする。さらに，$0 \leq i \leq L-1$ について，第 i 層のマークル木の高さはすべて等しく h_i であると仮定する。このとき，第 i 層のマークル木は $2^{\sum_{j=i+1}^{L-1} h_j}$ 個存在する。この署名方式では $2^{\sum_{j=0}^{L-1} h_j}$ 個のメッセージに署名できる。

この署名方式では，第 $(L-1)$ 層のマークル木の根が公開鍵となる。この公開鍵を生成する際には，1 回署名の公開鍵と秘密鍵の組を $2^{h_{L-1}}$ 個だけ生成すれば良い。$0 < i \leq L-1$ について，第 i 層の各マークル木は第 $(i-1)$ 層の 2^{h_i} 個のマークル木の根を署名するために使用される。第 0 層のマークル木は，それぞれ 2^{h_0} 個のメッセージの署名に使用される。

この署名方式では，一つのメッセージの署名の際に，各層についてそれぞれ一つのマークル木を生成しておけば十分である。各メッセージの署名は，そのメッセージに対する第 0 層のマークル木による署名と，$0 < i \leq L-1$ について，そのメッセージの署名の際に使用された第 i 層のマークル木による第 $(i-1)$ 層のマークル木の根の署名からなる。この署名方式について，階層数 $L = 3$，各階層のマークル木の高さ $h_0 = h_1 = h_2 = 3$ の模式図を図 7.1 に示す。灰色の節点は認証パスをなす節点である。

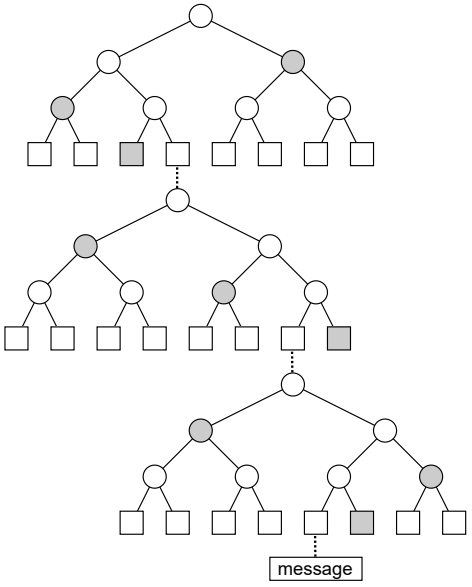


図 7.1: マークル木の階層構造による署名方式

7.2.4 プレフィクスとビットマスク

プレフィクスは，ハッシュ関数に基づく署名方式の処理において，すべてのハッシュ関数の計算がそれぞれ異なる入力に対して行われるよう入力に付加される系列である。プレフィクスは，Lighton と Micali [32] により，security string という名称で，ハッシュ関数に基づく署名方式の安全性をハッシュ関数の第二原像攻撃に対する安全性にタイトに帰着するために導入された。なお，プレフィクスは，ハッシュ関数の用途とそれが用いられる位置（例えば，どの 1 回署名方式か，どのマークル木のどの節点か）により自然に定義できることから，現在は通常，アドレスと呼ばれる。

ビットマスクは，Dahmen ら [13] により，ハッシュ関数に基づく署名方式の安全性をハッシュ関数の第二原像攻撃に対する安全性に帰着するために導入された。ビットマスクは乱数系列であり，ハッシュ関数への入力をランダム化するために，bit ごとの排他的論理和により入力に加えられる。

7.3 ハッシュ関数に基づく主要な署名方式

本章で取り上げるハッシュ関数に基づく署名方式を表 7.2 に示す。

NIST SP 800-208 [12] は，以下のハッシュ関数に基づく stateful な署名方式を規定している。

- Lighton-Micali Signatures (LMS), Hierarchical Signature System (HSS) [33]
- eXtended Merkle Signature Scheme (XMSS), multi-tree XMSS (XMSS^{MT}) [21]

LMS は Lighton と Micali による署名方式 [32] に基づく。HSS, XMSS^{MT} はそれぞれ，7.2.3 節で述べられたような，LMS, XMSS の階層構造による署名方式である。ハッシュ関数に基づく stateful な署名方式では，同一の秘密鍵が複

表 7.2: ハッシュ関数に基づく署名方式

文献	暗号化	鍵交換	署　名
Lighton-Micali Hash-Based Signatures [33, 12]			○
eXtended Merkle Signature Scheme (XMSS) [21, 12]			○
Stateless Hash-Based Digital Signature Algorithm (SLH-DSA) [40]			○

数のメッセージの署名に使用されることがないように秘密鍵を管理することが必須である。

NIST FIPS 205 [40] はハッシュ関数に基づく stateless な署名方式 SLH-DSA (Stateless Hash-Based Digital Signature Algorithm) を規定している。stateless な署名方式では，stateful な方式に求められるような秘密鍵の管理は不要である。SLH-DSA は，2022 年 7 月に NIST PQC 標準化プロジェクトで標準化候補アルゴリズムの一つに選出された SPHINCS$^+$ v.3.1 [2] に基づく。SPHINCS$^+$ は SPHINCS [8] の改良版として提案され [6, 7]，その後も NIST PQC 標準化プロジェクトで改良が行われ，v.3.1 となった。

以下では，Lighton-Micali Hash-Based Signatures，XMSS，SLH-DSA についてそれぞれ 7.3.1 節，7.3.2 節，7.3.3 節で述べるが，どの順番で読んでも差し支えない。

7.3.1 Lighton-Micali Hash-Based Signatures

IRTF RFC 8554 [33] では，LMS, HSS が述べられている。LMS, HSS では Winternitz one-time signature に基づく LM-OTS が用いられる。LMS は LM-OTS とマークル木とを用いて構成され，この構造は LMS 木と呼ばれる。HSS は LMS 木の階層構造による署名方式である。LM-OTS, LMS, HSS にはそれぞれ，それらのアルゴリズムで用いられるハッシュ関数，パラメータセットに対応する長さ 4 Bytes の符号なし整数が割り当てられる。これは typecode と呼ばれる。

本節では，ハッシュ関数 $H : \mathcal{D} \to \mathcal{R}$ について，$\mathcal{D} = \bigcup_{i \geq 0} \{0,1\}^{8i}, \mathcal{R} = \{0,1\}^{8n}$ とする。すなわち，H は任意長の byte 系列を入力とし，長さ n Bytes の系列を出力する。なお，本節の表記は概ね [18] の表記法に基づいている。

7.3.1.1 LM-OTS

$w \in \{1, 2, 4, 8\}$ を Winternitz 係数の幅（bit 長）とする。p を LM-OTS を構成する長さ n Bytes の系列の個数とする。*type* を typecode とする。ハッシュ関数 H を用いて以下の関数が定義される。

$$H^i_{I,q,d}(x; j) := \begin{cases} x & i = 0 \text{ のとき} \\ H(I \| [q]_4 \| [d]_2 \| [i + j - 1]_1 \| H^{i-1}_{I,q,d}(x; j)) & i \geq 1 \text{ のとき} \end{cases}$$

ここで，I は長さ 16 Bytes の系列であり，LM-OTS が，LMS や HSS においてどのマークル木で使用されるかを表す。q は長さ 4 Bytes の整数であり，LM-OTS の公開鍵が対応するマークル木の葉を表す。

■鍵生成アルゴリズム　与えられた I, q に対応する秘密鍵と公開鍵の組を生成するアルゴリズムを以下に示す。

1. $x_0, x_1, \ldots, x_{p-1} \in \{0,1\}^{8n}$ を一様ランダムに取る。
2. $0 \leq i \leq p-1$ について，$y_i := H^{2^w-1}_{I,q,i}(x_i; 0)$ とする（図 7.2）。
3. $K := H(I \| [q]_4 \| [8080]_2 \| y_0 \| y_1 \| \cdots \| y_{p-1})$ とする。

秘密鍵は $(type, I, q, x_0, x_1, \ldots, x_{p-1})$ である。公開鍵は $[type]_4 \| I \| [q]_4 \| K$ である。

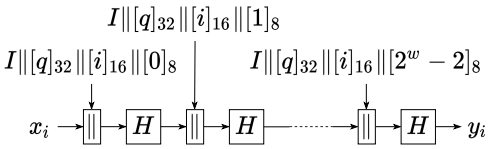


図 7.2: y_i の計算

■署名アルゴリズム　$\mathsf{Checksum} : \{0,1\}^{8n} \to \{0,1\}^{16}$ は以下のように定義される関数である。

$$\mathsf{Checksum}(S) := \Big(\sum_{i=0}^{8n/w-1} (2^w - 1 - d_i) \Big) \ll ls$$

ここで，$S = d_0 \| d_1 \| \cdots \| d_{8n/w-1}$ であり，$0 \le i \le 8n/w - 1$ について $d_i \in \{0,1\}^w$ である。上式で d_i は整数とみなされている。また，ls は n, w に応じて決まる整数であり，$16 - ls$ が w の倍数であり，かつ，$\mathsf{Checksum}(S)$ の 2 進数表記の長さが常に $16 - ls$ bits 以下となるよう定められる。

メッセージ M に対する署名アルゴリズムを以下に示す。

1. アルゴリズムのパラメータセットに応じて $type, n, p, w$ の値を定める。
2. $C \in \{0,1\}^{8n}$ を無作為に選択し，$Q := H(I \| [q]_4 \| [\mathsf{8181}]_2 \| C \| M)$，$c := \mathsf{Checksum}(Q)$ とする。
3. $Q \| c \in \{0,1\}^{wp}$ をそれぞれ長さ w bits のブロック $V_0, V_1, \ldots, V_{p-1}$ に分割する。
4. $0 \le i \le p-1$ について，$\sigma_i := H^{V_i}_{I,q,i}(x_i; 0)$ とする。ここで，V_i は整数とみなされる。

メッセージ M に対する署名は $\sigma := [type]_4 \| C \| \sigma_0 \| \sigma_1 \| \cdots \| \sigma_{p-1}$ である。

■検証アルゴリズム　鍵生成と署名のアルゴリズムより容易に導出される。詳細は [18, 33] を参照のこと。

7.3.1.2　LMS

LMS は LM-OTS と同じハッシュ関数を用いることが推奨されている。LMS のマークル木の各節点には番号が付される。根の番号は 1 であり，番号 ν の節点の左の子と右の子の番号はそれぞれ $2\nu, 2\nu + 1$ である。

h はマークル木の高さを表す。m はマークル木の各節点に対応する系列の byte 長を表し，これはハッシュ関数の出力長である。h, m の値は LMS の typecode によって定められる。

■鍵生成アルゴリズム

1. $I \in \{0,1\}^{128}$ を一様ランダムに取る。
2. $0 \le q \le 2^h - 1$ について，LM-OTS の公開鍵と秘密鍵の組 (pk^q, sk^q) を生成する。
3. マークル木の番号 r の節点に対応する系列 $T[r]$ は以下のように定義される。

$$T[r] := \begin{cases} H(I \| [r]_4 \| [\mathsf{8282}]_2 \| pk^{r-2^h}) & 2^h \le r \le 2^{h+1} - 1 \text{ のとき} \\ H(I \| [r]_4 \| [\mathsf{8383}]_2 \| T[2r] \| T[2r+1]) & 1 \le r \le 2^h - 1 \text{ のとき} \end{cases}$$

公開鍵は $[type]_4 \| [otstype]_4 \| I \| T[1]$ である。秘密鍵は $sk^0, sk^1, \ldots, sk^{2^h-1}$ である。ここで $type, otstype$ はそれぞれ LMS, LMS-OTS の typecode を表す。

■署名アルゴリズム　以下では $0 \le q \le 2^h - 1$ である。最初の署名では $q = 0$ とし，1 回の署名ごとに q の値を 1 だけ増やすことにより，LM-OTS の各秘密鍵を複数回使用しないようにしなければならない。

メッセージ M に対する署名アルゴリズムを以下に示す。

1. 秘密鍵 sk^q を用いて LM-OTS による M の署名 σ を計算する。
2. $0 \leq i \leq h-1$ について，$p_i := T[\lfloor (q+2^h)/2^i \rfloor \oplus 1]$ とする。

M の署名は $[q]_4 \| \sigma \| [type]_4 \| p_0 \| p_1 \| \cdots \| p_{h-1}$ である。$p_0, p_1, \ldots, p_{h-1}$ は LM-OTS の公開鍵 pk^q の認証パスである。

■検証アルゴリズム　鍵生成と署名のアルゴリズムより容易に導出される。詳細は [18, 33] を参照のこと。

7.3.1.3　HSS

HSS は 7.2.3 節で述べられたような LMS 木の階層構造による署名方式である。階層数 L は $1 \leq L \leq 8$ を満たす。

HSS は stateful な署名方式なので，メッセージの署名の際に最下層の LMS 木の秘密鍵が使い尽くされたとき，そのメッセージの署名で使用された L 個の LMS 木のうち，第 l 層から下のすべての LMS 木の秘密鍵が使い尽くされた最大の l を求める。$l = L-1$ のときは，新たな署名を作成しない。$l \leq L-2$ のとき，$0 \leq i \leq l$ について，秘密鍵が使い尽くされた第 i 層の LMS 木を破棄し，それぞれに替わる新たな LMS 木を使用して新しいメッセージへの署名を行う。

HSS の鍵生成，署名，検証の各アルゴリズムについての詳細は [18, 33] を参照のこと。

7.3.1.4　パラメータの設定と安全性

LMS の選択文書攻撃に対する存在偽造不能性（EUF-CMA）については，Katz [28] や Fluhrer [18] によりランダムオラクルモデルを仮定して示されており，また，Eaton [17] により量子ランダムオラクルモデルを仮定して示されている。なお，IRTF RFC 8554 [33] には，ハッシュ関数は第二原像攻撃に対する安全性を満たさなければならないと記されている。

NIST SP 800-208 [12] では，ハッシュ関数として SHA-256, SHA-256/192, SHAKE256/256, SHAKE256/192 を使用することが認可されている。ここで，SHA-256/192 は SHA-256 の出力の上位 192 bits を出力とするハッシュ関数である。SHAKE256/256, SHAKE256/192 はそれぞれ，出力長を 256 bits，192 bits とする SHAKE256 である。NIST SP 800-208 [12] と IRTF RFC 8554 [33] の両方に掲載されている SHA-256 を用いる場合の LM-OTS, LMS のパラメータセットの値の一覧をそれぞれ表 7.3, 7.4 に示す。

表 7.3: LM-OTS のパラメータセットと署名長（単位は Byte）

名称	n	w	p	ls	署名長
LMOTS_SHA256_N32_W1	32	1	265	7	8,516
LMOTS_SHA256_N32_W2	32	2	133	6	4,292
LMOTS_SHA256_N32_W4	32	4	67	4	2,180
LMOTS_SHA256_N32_W8	32	8	34	0	1,124

表 7.4: LMS のパラメータセット

名称	m	h
LMS_SHA256_M32_H5	32	5
LMS_SHA256_M32_H10	32	10
LMS_SHA256_M32_H15	32	15
LMS_SHA256_M32_H20	32	20
LMS_SHA256_M32_H25	32	25

7.3.2　XMSS: eXtended Merkle Signature Scheme

XMSS は [10, 24] で提案された方式の改良版 [25] に基づく署名方式であり，WOTS$^+$ と呼ばれる Winternitz one-time signature に基づく 1 回署名方式 [22] を用いる[*1]。

XMSS では三つの鍵付きハッシュ関数 $\mathsf{F}, \mathsf{H}, \mathsf{H}_{\mathrm{msg}}$ と擬似ランダム関数 R が用いられる。いずれも出力の byte 長は等しく，これを n とする。F の入力は byte 長 n の鍵と byte 長 n の系列である。H の入力は byte 長 n の鍵と byte

[*1] 7.3.3 節の SLH-DSA で用いられる 1 回署名方式とマークル木を用いた署名方式もそれぞれ WOTS$^+$，XMSS と呼ばれるが，アルゴリズムには相違点が存在する。

長 $2n$ の系列である。$\mathsf{H}_{\mathrm{msg}}$ の入力は byte 長 $3n$ の鍵と任意 byte 長の系列である。R の入力は byte 長 n の鍵と byte 長 32 の系列である。これらの関数は SHA-2 [38] または SHA-3 [39] を用いて定義される。例えば，$n = 32$ のとき，SHA-256 を用いて以下のように定義される。

$$\begin{aligned}\mathsf{F}(k, x) &:= \text{SHA-256}([0]_{32} \| k \| x)\\ \mathsf{H}(k, x) &:= \text{SHA-256}([1]_{32} \| k \| x)\\ \mathsf{H}_{\mathrm{msg}}(k, x) &:= \text{SHA-256}([2]_{32} \| k \| x)\\ \mathsf{R}(k, x) &:= \text{SHA-256}([3]_{32} \| k \| x)\end{aligned}$$

XMSS では，ハッシュ関数の呼び出しをランダム化するために，それぞれのハッシュ関数の呼び出しで，鍵とビットマスクが用いられる。これらは擬似ランダム関数を用いて生成され，入力として byte 系列の seed と長さ 32 Bytes のアドレス ADRS が与えられる。アドレスは 3 種あり，それぞれ OTS ハッシュアドレス，L 木アドレス，ハッシュ木アドレスと呼ばれる。それらの構造を図 7.3 に示す。

layer address	(4 Bytes)
tree address	(8 Bytes)
type = 0	(4 Bytes)
OTS address	(4 Bytes)
chain address	(4 Bytes)
hash address	(4 Bytes)
keyAndMask	(4 Bytes)

(a) OTS ハッシュアドレス

layer address	(4 Bytes)
tree address	(8 Bytes)
type = 1	(4 Bytes)
L-tree address	(4 Bytes)
tree height	(4 Bytes)
tree index	(4 Bytes)
keyAndMask	(4 Bytes)

(b) L 木アドレス

layer address	(4 Bytes)
tree address	(8 Bytes)
type = 2	(4 Bytes)
Padding = 0	(4 Bytes)
tree height	(4 Bytes)
tree index	(4 Bytes)
keyAndMask	(4 Bytes)

(c) ハッシュ木アドレス

図 7.3: アドレスの構造

7.3.2.1 WOTS$^+$

$w \in \{4, 16\}$ は Winternitz パラメータと呼ばれる。$\ell := \ell_1 + \ell_2$ は公開鍵，秘密鍵，署名を構成する byte 長 n の要素の個数を表す。ここで，

$$\ell_1 := \lceil 8n / \log_2 w \rceil, \quad \ell_2 := \lfloor \log_2(\ell_1(w-1)) / \log_2 w \rfloor + 1$$

である。

■チェイニング関数 チェイニング関数 chain の入力は，長さ n Bytes の系列 X，スタートインデクス i，ステップ数 s，長さ 32 Bytes のアドレス ADRS，長さ n Bytes のシード seed であり，以下のように定義される。

$$\mathsf{chain}(X, i, s, \mathsf{seed}, \mathrm{ADRS}) := \begin{cases} X & s = 0 \text{ のとき} \\ \texttt{NULL} & i + s \geq w \text{ のとき} \\ \mathsf{F}(Key, \mathsf{chain}(X, i, s-1, \mathsf{seed}, \mathrm{ADRS}) \oplus BM) & \text{それ以外のとき} \end{cases}$$

ここで，

$$Key := \mathsf{R}(\mathsf{seed}, \mathrm{ADRS}' \| [i+s-1]_4 \| [0]_4), \quad BM := \mathsf{R}(\mathsf{seed}, \mathrm{ADRS}' \| [i+s-1]_4 \| [1]_4)$$

である。なお，ADRS$'$ は ADRS の上位 24 Bytes であり，例えば，ADRS$' \| [i+s-1]_4 \| [0]_4$ は図 7.3a の ADRS の hash address, keyAndMask の値をそれぞれ，$[i+s-1]_4, [0]_4$ とすることを表している。

■鍵生成アルゴリズム　入力は ADRS, seed である。

1. $0 \le i \le \ell - 1$ について，$sk_i \in \{0,1\}^{8n}$ を一様ランダムに取る。
2. $0 \le i \le \ell - 1$ について，ADRS の chain address の値を $[i]_4$ とし，

$$pk_i := \mathsf{chain}(sk_i, 0, w-1, \mathsf{seed}, \mathrm{ADRS})$$

とする。この計算を図 7.4 に示す。この図で

$$Key_j := \mathsf{R}(\mathsf{seed}, \mathrm{ADRS}'\|[j]_4\|[0]_4), \quad BM_j := \mathsf{R}(\mathsf{seed}, \mathrm{ADRS}'\|[j]_4\|[1]_4)$$

である。

公開鍵は $pk := (pk_0, pk_1, \ldots, pk_{\ell-1})$ である。秘密鍵は $sk := (sk_0, sk_1, \ldots, sk_{\ell-1})$ である。

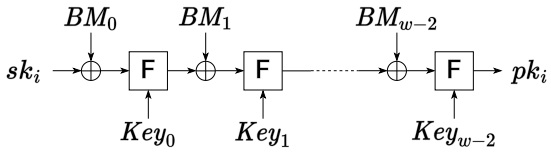


図 7.4: pk_i の計算

■署名アルゴリズム　入力は byte 長 n のメッセージ M，秘密鍵 sk，アドレス ADRS，シード seed である。

1. M をそれぞれ長さ $\log_2 w$ bits の ℓ_1 個のブロックに分割し，先頭から順に $M_0, M_1, \ldots, M_{\ell_1-1}$ とする。これらを整数とみなすと，$0 \le i \le \ell_1 - 1$ について，$M_i \in \{0, 1, \ldots, w-1\}$ である。
2. $C := \sum_{i=0}^{\ell_1-1} (w - 1 - M_i)$ とする。
3. $C \cdot 2^{8-(\ell_2 \log_2 w \bmod 8)}$ を長さ $\lceil (\ell_2 \log_2 w)/8 \rceil$ Bytes の系列とみなし，それぞれ長さ $\log_2 w$ bits の ℓ_2 個のブロックに分割し，先頭から順に $M_{\ell_1}, M_{\ell_1+1}, \ldots, M_{\ell-1}$ とする。
4. $0 \le i \le \ell - 1$ について，ADRS の chain address の値を i とし，

$$sig_i := \mathsf{chain}(sk_i, 0, M_i, \mathsf{seed}, \mathrm{ADRS})$$

とする。

メッセージ M に対する署名は $sig_0, sig_1, \ldots, sig_{\ell-1}$ である。

■検証アルゴリズム　鍵生成と署名のアルゴリズムより容易に導出される。詳細は [21] を参照のこと。

7.3.2.2　XMSS

XMSS はマークル木を用いた署名方式であり，公開鍵と秘密鍵の各組は完全二分木に対応付けられる。

XMSS のハッシュ木の構成のために，ランダム化ハッシュ関数 RH が導入されている。RH の入力は長さ n Bytes の $LEFT, RIGHT$，長さ n Bytes のシード seed，長さ 32 Bytes のアドレス ADRS であり，以下のように定義される。

$$\mathsf{RH}(LEFT, RIGHT, \mathsf{seed}, \mathrm{ADRS}) := \mathsf{H}(Key, (LEFT \oplus BM_0)\|(RIGHT \oplus BM_1))$$

ここで，

$$Key := \mathsf{R}(\mathsf{seed}, \mathrm{ADRS}'\|[0]_4), \quad BM_0 := \mathsf{R}(\mathsf{seed}, \mathrm{ADRS}'\|[1]_4), \quad BM_1 := \mathsf{R}(\mathsf{seed}, \mathrm{ADRS}'\|[2]_4)$$

である。なお，ADRS′ は ADRS の上位 28 Bytes であり，例えば，ADRS′$\|[0]_4$ は ADRS の図 7.3 の keyAndMask の値を $[0]_4$ とすることを表している。

秘密鍵の生成には [10] に示されているような擬似ランダム鍵生成法を用いることが許容されているが，その安全性は少なくとも XMSS の安全性と同等でなければならない。

■鍵生成アルゴリズム　鍵生成アルゴリズムではマークル木が構成され，その各葉には WOTS$^+$ の公開鍵が対応する。WOTS$^+$ の公開鍵に対して L 木と呼ばれるハッシュ木が構成され，その木の根のハッシュ値が XMSS のマークル木の葉に割り当てられる。L 木の高さ $j(\geq 0)$ の左から $i(\geq 0)$ 番目の節点を $\mathrm{Node}_{i,j}$ と表記する。L 木は以下にしたがって構成される。入力は WOTS$^+$ の公開鍵 $pk := (pk_0, pk_1, \ldots, pk_{\ell-1})$，L 木アドレス ADRS，シード seed である。

1. $0 \leq i \leq \ell - 1$ について，$\mathrm{Node}_{i,0} := pk_i$ とする。
2. $j \geq 0$ について，根が得られるまで以下にしたがって $\mathrm{Node}_{i,j+1}$ を計算する。なお，値の定義された $\mathrm{Node}_{i,j}$ の個数を ℓ' とする。
   (a) $0 \leq i < \lfloor \ell'/2 \rfloor$ について，$\mathrm{Node}_{i,j+1} := \mathsf{RH}(\mathrm{Node}_{2i,j}, \mathrm{Node}_{2i+1,j}, \mathsf{seed}, \mathrm{ADRS})$ とする。ここで，ADRS の tree height を $[j]_4$，tree index を $[i]_4$ とする。さらに，ℓ' が奇数のとき，$\mathrm{Node}_{\lfloor \ell'/2 \rfloor, j+1} := \mathrm{Node}_{\ell'-1,j}$ とする。
   (b) $j \leftarrow j + 1$ とする。

鍵生成アルゴリズムで構成されるマークル木の高さを h とすると，このマークル木には 2^h 個の葉が存在する。このマークル木に対応する 2^h 個の WOTS$^+$ の公開鍵，それらの L 木，さらに，このマークル木の計算に用いられる OTS ハッシュアドレス，L 木アドレス，ハッシュ木アドレスの layer address，tree address はすべて，それぞれ $[0]_4$，$[0]_8$ である。左から $k\,(\geq 0)$ 番目の葉に対応する OTS ハッシュアドレスの OTS address，L 木アドレスの L-tree address は $[k]_4$ である。

鍵生成アルゴリズムで構成されるマークル木の葉は対応する L 木の根である。葉以外の節点は L 木の節点と同じ方法で計算される。なお，このマークル木は完全二分木なので，上述の L 木の計算手続きで，ℓ' は常に偶数となる。

秘密鍵は，2^h 個の WOTS$^+$ の秘密鍵，次の署名に使用される WOTS$^+$ の秘密鍵に対応するマークル木の葉の番号 idx，署名されるメッセージのハッシュの計算に使用される SK_{PRF}，マークル木の根 $root$，seed である。公開鍵は，マークル木の根，seed である。ここで，SK_{PRF} と seed はこの鍵生成アルゴリズムで無作為に選択される長さ n Bytes の系列である。また，公開鍵には識別子 OID が付される。

■署名アルゴリズム　メッセージ M の署名は，署名に使用される WOTS$^+$ の秘密鍵の番号 idx，M のダイジェストの計算に使用される乱数 r，WOTS$^+$ による署名，マークル木の idx 番目の葉の認証パスからなる。

1. M のダイジェストを $M' := \mathsf{H}_{\mathrm{msg}}(r \| root \| [idx]_n, M)$ とする。ここで，$r := \mathsf{R}(SK_{\mathrm{PRF}}, [idx]_4)$ である。
2. WOTS$^+$ の idx 番目の秘密鍵を用いて M' に署名し，マークル木の idx 番目の葉の認証パスを計算する。

WOTS$^+$ の同じ秘密鍵が 2 回以上使用されないよう，idx は $idx \leftarrow idx + 1$ により更新される。

■検証アルゴリズム　鍵生成と署名のアルゴリズムより容易に導出される。詳細は [21] を参照のこと。

7.3.2.3　XMSS^{MT}

XMSS^{MT} は，7.2.3 節のマークル木の階層構造による署名方式に相当する。XMSS^{MT} 木はハイパー木と呼ばれ，d 層の XMSS 木からなる。ここで，XMSS 木は 7.3.2.2 節の鍵生成アルゴリズムで生成される L 木とマークル木からなる木を表す。第 $(d-1)$ 層と第 0 層はそれぞれ，XMSS^{MT} 木の根と葉に相当する。すべての XMSS 木の高さは等しく，Winternitz パラメータもすべて同じ値が用いられる。第 x 層の左から y 番目の XMSS 木の構成で使用される

OTS ハッシュアドレス，L 木アドレス，ハッシュ木アドレスの layer address と tree address は，それぞれ $[x]_4$，$[y]_4$ である。

XMSS^{MT} の鍵生成，署名，検証の各アルゴリズムについての詳細は [21] を参照のこと。

7.3.2.4 パラメータの設定と安全性

Kampanakis と Fluhrer [26] により，LMS と XMSS の比較が論じられている。

Hülsing [25] らは，XMSS について安全性証明を与え，EUF-CMA 安全性を満たすことを鍵付きハッシュ関数 $\mathsf{F}, \mathsf{H}, \mathsf{H}_{\mathrm{msg}}$ と擬似ランダム関数 R の以下の安全性に帰着している。

- F が以下の性質を満たすこと
  - multi-function, multi-target second preimage resistance (MM-SPR)
  - すべての出力が 2 個以上の原像を持つこと
- H が MM-SPR を満たすこと
- $\mathsf{H}_{\mathrm{msg}}$ が multi-target extended target collision resistance (M-ETCR) を満たすこと
- R が擬似ランダム関数 (PRF) であること

ここで，MM-SPR, M-ETCR は，$\mathsf{F}, \mathsf{H}, \mathsf{H}_{\mathrm{msg}}$ の構成に用いられるハッシュ関数の第二原像攻撃に対する安全性に基づく性質である。一方，PRF は，秘密鍵入力を有するハッシュ関数が擬似ランダム関数であることを要求する。さらに，R による鍵とビットマスクの生成については，ハッシュ関数がランダムオラクルであることが仮定される。

IRTF RFC 8391 [21] では，上述の XMSS の安全性に関する結果に基づいて，$n = 32, 64$ のとき，それぞれ，256 bit 安全性，512 bit 安全性が提供されると記されている。また，量子計算機を用いた攻撃に対してはそれぞれ，128 bit 安全性，256 bit 安全性が提供されると記されている。

IRTF RFC 8391 [21] では，ハッシュ関数として SHA-256 を用いることが要求されているが，オプションとして SHAKE128/256, SHA-512, SHAKE256/512 を用いることが記されている。一方, NIST SP 800-208 では, SHA-256, SHA-256/192, SHAKE256/256, SHAKE256/192 を用いることが認可されている。NIST SP 800-208 [12] と IRTF RFC 8391 [21] の両方に掲載されている SHA-256 を用いる場合の WOTS$^+$, XMSS, XMSS^{MT} のパラメータセットの値の一覧をそれぞれ表 7.5, 7.6, 7.7 に示す。

表 7.5: WOTS$^+$ のパラメータセット

名称	n	w	ℓ
WOTSP-SHA2_256	32	16	67

表 7.6: XMSS のパラメータセットと署名長（単位は Byte）

名称	n	w	ℓ	h	署名長
XMSS-SHA2_10_256	32	16	67	10	2,500
XMSS-SHA2_16_256	32	16	67	16	2,692
XMSS-SHA2_20_256	32	16	67	20	2,820

7.3.3 SLH-DSA

SLH-DSA [40] は 7.2.3 節のマークル木の階層構造による署名方式に基づく stateless な署名方式である。SLH-DSA で用いられる 1 回署名方式とマークル木を用いた署名方式はそれぞれ WOTS$^+$ (Winternitz One-Time Signature Plus scheme)，XMSS (eXtended Merkle Signature Scheme) と呼ばれる[*2]。また，XMSS で構成されるマークル木は XMSS 木と呼ばれる。SLH-DSA が 7.2.3 節で述べられた方式と異なる点は，FORS (Forest of Random Subsets)

[*2] これらの名称は 7.3.2 節の XMSS の対応する署名方式の名称と同一であるが，アルゴリズムには相違点が存在する。

表 7.7: XMSS^{MT} のパラメータセットと署名長（単位は Byte）

名称	n	w	ℓ	h	d	署名長
XMSSMT-SHA2_20/2_256	32	16	67	20	2	4,963
XMSSMT-SHA2_20/4_256	32	16	67	20	4	9,251
XMSSMT-SHA2_40/2_256	32	16	67	40	2	5,605
XMSSMT-SHA2_40/4_256	32	16	67	40	4	9,893
XMSSMT-SHA2_40/8_256	32	16	67	40	8	18,469
XMSSMT-SHA2_60/3_256	32	16	67	60	3	8,392
XMSSMT-SHA2_60/6_256	32	16	67	60	6	14,824
XMSSMT-SHA2_60/12_256	32	16	67	60	12	27,688

と呼ばれるハッシュ関数に基づく数回 (few-time) 署名方式が導入されている点である。数回署名方式は，一組の公開鍵と秘密鍵の組を用いて，複数個のメッセージに署名できる。SLH-DSA では，メッセージは FORS を用いて署名され，FORS の公開鍵が hypertree と呼ばれる XMSS 木の階層構造による署名方式を用いて署名される。SLH-DSA は，数回署名方式を導入して署名可能な回数を増加させることにより，stateless であることを達成している。なお，WOTS$^+$，XMSS，hypertree，FORS は SLH-DSA の構成要素として使用されるのみであり，それぞれの単独での使用は許容されていない。

SLH-DSA の公開鍵は長さ n Bytes の 2 つの系列 **PK**.root と **PK**.seed である。**PK**.root は hypertree の最上層の XMSS 木の根である。**PK**.seed は無作為に選択される。SLH-DSA の秘密鍵は n Bytes の 2 つの系列 **SK**.seed と **SK**.prf であり，いずれも無作為に選択される。なお，NIST FIPS 205 [40] では，**PK**.seed, **SK**.seed, **SK**.prf の生成に SP 800-90A [4]，SP 800-90B [44]，SP 800-90C [5] で規定されているランダム bit 生成器を使用することが求められている。WOTS$^+$ と FORS のすべての秘密鍵は，**SK**.seed を用いて擬似ランダム関数により生成される。**SK**.prf は，メッセージダイジェストの計算に使用される乱数系列の生成に使用される。

SLH-DSA の署名では，メッセージダイジェストはこれらの乱数系列を用いたランダム化されたハッシュ関数により生成され，そのメッセージダイジェストの一部を用いてメッセージの署名に用いる FORS の公開鍵と秘密鍵の組が選択される。

SLH-DSA では以下の関数が用いられる。

- $\mathbf{PRF}_{msg} : \mathbb{B}^n \times \mathbb{B}^n \times \mathbb{B}^* \to \mathbb{B}^n$ はメッセージダイジェストの計算に使用される乱数系列を生成する擬似ランダム関数である。
- $\mathbf{H}_{msg} : \mathbb{B}^n \times \mathbb{B}^n \times \mathbb{B}^n \times \mathbb{B}^* \to \mathbb{B}^m$ はメッセージダイジェストを計算するハッシュ関数である。
- $\mathbf{PRF} : \mathbb{B}^n \times \mathbb{B}^n \times \mathbb{B}^{32} \to \mathbb{B}^n$ は WOTS$^+$，FORS の秘密鍵を生成する擬似ランダム関数である。
- $\mathbf{T}_\ell : \mathbb{B}^n \times \mathbb{B}^{32} \times \mathbb{B}^{\ell n} \to \mathbb{B}^n$ は WOTS$^+$，XMSS 木，FORS で用いられるハッシュ関数である。

さらに，$\mathbf{T}_1, \mathbf{T}_2$ について，$\mathbf{F} := \mathbf{T}_1$, $\mathbf{H} := \mathbf{T}_2$ の表記が用いられる。

SLH-DSA では，図 7.5 に示す 7 種のアドレスが用いられる。どのアドレスも長さは 32 Bytes である。各アドレスの layer address と tree address は XMSS 木の階層構造で，ハッシュ関数がどの XMSS 木で用いられるかを表す。これに基づき，FORS 木アドレス，FORS 木根圧縮アドレス，FORS 鍵生成アドレスの layer address の値はすべて 0 と定められている。

layer address	(4 Bytes)
tree address	(12 Bytes)
type = 0	(4 Bytes)
key pair address	(4 Bytes)
chain address	(4 Bytes)
hash address	(4 Bytes)

(a) WOTS$^+$ ハッシュアドレス

layer address	(4 Bytes)
tree address	(12 Bytes)
type = 1	(4 Bytes)
key pair address	(4 Bytes)
0	(4 Bytes)
0	(4 Bytes)

(b) WOTS$^+$ 公開鍵圧縮アドレス

layer address	(4 Bytes)
tree address	(12 Bytes)
type = 2	(4 Bytes)
0	(4 Bytes)
tree height	(4 Bytes)
tree index	(4 Bytes)

(c) ハッシュ木アドレス

layer address = 0	(4 Bytes)
tree address	(12 Bytes)
type = 3	(4 Bytes)
key pair address	(4 Bytes)
tree height	(4 Bytes)
tree index	(4 Bytes)

(d) FORS 木アドレス

layer address = 0	(4 Bytes)
tree address	(12 Bytes)
type = 4	(4 Bytes)
key pair address	(4 Bytes)
0	(4 Bytes)
0	(4 Bytes)

(e) FORS 木根圧縮アドレス

layer address	(4 Bytes)
tree address	(12 Bytes)
type = 5	(4 Bytes)
key pair address	(4 Bytes)
chain address	(4 Bytes)
0	(4 Bytes)

(f) WOTS$^+$ 鍵生成アドレス

layer address = 0	(4 Bytes)
tree address	(12 Bytes)
type = 6	(4 Bytes)
key pair address	(4 Bytes)
0	(4 Bytes)
tree index	(4 Bytes)

(g) FORS 鍵生成アドレス

図 7.5: アドレスの構造

7.3.3.1 WOTS$^+$

WOTS$^+$ は Winternitz one-time signature に基づく 1 回署名方式である。WOTS$^+$ は 2 つのパラメータ n と lg_w を用いる。n はセキュリティパラメータであり，署名されるメッセージ，公開鍵，秘密鍵，署名を構成する系列の byte 長である。lg_w は Winternitz パラメータと呼ばれる正整数 w について $lg_w := \log_2 w$ と定義される。WOTS$^+$ では $lg_w = 4$ と定められており，$w = 16$ である。

WOTS$^+$ の公開鍵，秘密鍵，署名を構成する系列の個数は $len := len_1 + len_2$ で表される。ここで，

$$len_1 := \lceil 8n/lg_w \rceil, \quad len_2 := \lfloor \log_2(len_1(w-1))/lg_w \rfloor + 1$$

である。$lg_w = 4$ なので，$len_1 = 2n$，$len_2 = 3$，$len = 2n + 3$ である。

■チェイニング関数 チェイニング関数 chain の入力は，長さ n Bytes の系列 X，スタートインデクス i，ステップ数 s，**PK**.seed，WOTS$^+$ ハッシュアドレス **ADRS** であり，以下のように定義される。

1. $tmp \leftarrow X$ とする。
2. $i \leq j \leq i + s - 1$ について，**ADRS** の hash address を j とし，$tmp \leftarrow \mathbf{F}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, tmp)$ とする。
3. tmp を返す。

■鍵生成アルゴリズム　入力は **SK**.seed, **PK**.seed, WOTS$^+$ ハッシュアドレス **ADRS** である。なお，**ADRS** の chain address, hash address の値はいずれも 0 である。

1. $0 \leq i \leq len - 1$ について，**ADRS** の chain address の値を i とし，

$$sk_i \leftarrow \mathbf{PRF}(\mathbf{PK}.\text{seed}, \mathbf{SK}.\text{seed}, \text{skADRS}) \qquad pk_i \leftarrow \mathsf{chain}(sk_i, 0, w-1, \mathbf{PK}.\text{seed}, \mathbf{ADRS})$$

   とする。なお，skADRS は WOTS$^+$ 鍵生成アドレスであり，layer address, tree address, key pair address, chain addresss については **ADRS** と同じ値が用いられる。
2. $pk \leftarrow \mathbf{T}_{len}(\mathbf{PK}.\text{seed}, \text{wotspkADRS}, pk_0 \| \cdots \| pk_{len-1})$ とする。ここで，wotspkADRS は WOTS$^+$ 公開鍵圧縮アドレスであり，layer address, tree address, key pair address については **ADRS** と同じ値が用いられる。

公開鍵は pk である。秘密鍵は $sk := (sk_0, sk_1, \ldots, sk_{len-1})$ である。

■署名アルゴリズム　入力は byte 長 n のメッセージ M，**SK**.seed，**PK**.seed，WOTS$^+$ ハッシュアドレス **ADRS** である。**ADRS** の layer address, tree address, key pair address で指定される WOTS$^+$ の秘密鍵を用いて署名が生成される。なお，**ADRS** の chain address, hash address の値はいずれも 0 である。

1. M をそれぞれ長さ lg_w bits の len_1 個のブロックに分割し，先頭から順に $msg_0, msg_1, \ldots, msg_{len_1-1}$ とする。これらを整数とみなすと，$0 \leq i \leq len_1 - 1$ について，$msg_i \in \{0, 1, \ldots, w-1\}$ である。
2. $csum \leftarrow \sum_{i=0}^{len_1-1} (w - 1 - msg_i)$ とする。
3. $csum \cdot 2^{(8-(len_2 \cdot lg_w \bmod 8)) \bmod 8}$ を長さ $\lceil (len_2 \cdot lg_w)/8 \rceil$ Bytes の系列とみなし，それぞれ長さ lg_w bits の len_2 個のブロックに分割し，先頭から順に $msg_{len_1}, msg_{len_1+1}, \ldots, msg_{len-1}$ とする。
4. $0 \leq i \leq len - 1$ について，**ADRS** の chain address の値を i とし，

$$sk_i \leftarrow \mathbf{PRF}(\mathbf{PK}.\text{seed}, \mathbf{SK}.\text{seed}, \text{skADRS}) \qquad sig_i \leftarrow \mathsf{chain}(sk_i, 0, msg_i, \mathbf{PK}.\text{seed}, \mathbf{ADRS})$$

   とする。なお，skADRS は WOTS$^+$ 鍵生成アドレスであり，layer address, tree address, key pair address, chain addresss については **ADRS** と同じ値が用いられる。

メッセージ M に対する署名は $sig_0, sig_1, \ldots, sig_{len-1}$ である。

■検証アルゴリズム　SLH-DSA では WOTS$^+$ が単独で使用されることが想定されていないため，検証アルゴリズムは明示されておらず，Winternitz one-time signature の検証に必須の，メッセージと署名の組から対応する公開鍵の候補を計算するアルゴリズムが示されている。なお，このアルゴリズムは，鍵生成と署名のアルゴリズムより容易に導出される。詳細は NIST FIPS 205 [40] を参照のこと。

7.3.3.2　XMSS

XMSS はマークル木を用いた署名方式であり，WOTS$^+$ を用いて構成される。

■鍵生成アルゴリズム　XMSS では，WOTS$^+$ の公開鍵を各葉にもつ高さ h' のマークル木（XMSS 木）を構成することにより，公開鍵が生成される。XMSS 木の高さ $z\,(\geq 0)$ の左から $i\,(\geq 0)$ 番目の節点を $node_{i,j}$ と表記する。入力は **SK**.seed, **PK**.seed, **ADRS** である。

1. $0 \leq i \leq 2^{h'} - 1$ について，$node_{i,0} \leftarrow pk_i$ とする。ここで，pk_i は **SK**.seed, **PK**.seed を用いて計算される WOTS$^+$ の公開鍵である。なお，pk_i の計算に用いられるアドレスの layer address と tree address の値は **ADRS** のそれらと等しく，key pair address の値は $[i]_4$ である。

2. $1 \leq z \leq h'$ について，それぞれ，$0 \leq i \leq 2^{h'-z} - 1$ について，

$$node_{i,z} \leftarrow \mathbf{H}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, node_{2i,z-1} \| node_{2i+1,z-1})$$

とする。ここで，**ADRS** はハッシュ木アドレスであり，tree height の値は $[z]_4$，tree index の値は $[i]_4$ である。

公開鍵は XMSS 木の根 $node_{0,h'}$ であり，秘密鍵は $2^{h'}$ 個の WOTS$^+$ の秘密鍵である。

なお，XMSS の単独での使用が想定されていないことから，NIST FIPS 205 [40] では，鍵生成アルゴリズムは明示されておらず，XMSS 木の各節点を計算する再帰的アルゴリズムが示されている。

■署名アルゴリズム　SLH-DSA では，XMSS で署名されるメッセージは XMSS の公開鍵あるいは FORS の公開鍵のみである。入力は $M, \mathbf{SK}.\text{seed}, idx, \mathbf{PK}.\text{seed}, \mathbf{ADRS}$ である。M は長さ n Bytes のメッセージ，idx は M の署名に使用される WOTS$^+$ の鍵の key pair address である。

WOTS$^+$ の idx 番目の秘密鍵を用いて M に署名し，XMSS 木の idx 番目の葉の認証パスを計算する。

■検証アルゴリズム　NIST FIPS 205 [40] では，検証に必須の，メッセージと署名の組から対応する公開鍵の候補を計算するアルゴリズムが示されている。このアルゴリズムは鍵生成と署名のアルゴリズムより容易に導出されるので，詳細は NIST FIPS 205 [40] を参照のこと。

7.3.3.3　Hypertree

SLH-DSA では，hypertree と呼ばれる XMSS 木の階層構造が用いられる。hypertree は d 層の XMSS 木からなり，すべての XMSS 木の高さは等しい。第 $(d-1)$ 層と第 0 層はそれぞれ hypertree の根と葉に相当する。第 x 層の左から y 番目の XMSS 木の構成で使用される WOTS$^+$ ハッシュアドレス，WOTS$^+$ 公開鍵圧縮アドレス，WOTS$^+$ 鍵生成アドレス，ハッシュ木アドレスの layer address と tree address はそれぞれ $[x]_4$，$[y]_{12}$ である。

hypertree の公開鍵は第 $(d-1)$ 層の XMSS の公開鍵である。hypertree の署名，検証の各アルゴリズムについての詳細は NIST FIPS 205 [40] を参照のこと。

7.3.3.4　FORS

FORS は，数回署名方式 HORS [43] に基づく HORST [8] の改良版である。FORS は $k, t := 2^a$ をパラメータとし，長さ ka bits の系列に署名を行う。

■鍵生成アルゴリズム　入力は **SK**.seed, **PK**.seed, FORS 木アドレス **ADRS** である。なお，**ADRS** の layer address, tree address, key pair address は，生成された FORS の公開鍵の署名に用いられる WOTS$^+$ の鍵の生成に用いられるアドレスのそれらの値と等しい。

1. $0 \leq i \leq kt - 1$ について，

$$sk_i \leftarrow \mathbf{PRF}(\mathbf{PK}.\text{seed}, \mathbf{SK}.\text{seed}, \text{skADRS}) \qquad node_{i,0} \leftarrow \mathbf{F}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, sk_i)$$

とする。ここで，skADRS は FORS 鍵生成アドレスであり，layer address, tree address, key pair address については **ADRS** と同じ値が用いられ，tree index の値は $[i]_4$ である。また，**ADRS** の tree index の値は $[i]_4$ である。

2. $1 \leq z \leq a$ について，それぞれ，$0 \leq i \leq k \cdot 2^{a-z} - 1$ について，

$$node_{i,z} \leftarrow \mathbf{H}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, node_{2i,z-1} \| node_{2i+1,z-1})$$

とする。ここで，**ADRS** の tree height の値は $[z]_4$，tree index の値は $[i]_4$ である。

3. $pk \leftarrow \mathbf{T}_k(\mathbf{PK}.\mathrm{seed}, \mathrm{forspkADRS}, node_{0,a} \| \cdots \| node_{k-1,a})$ とする。ここで，forspkADRS は FORS 木根圧縮アドレスであり，layer address, tree address, key pair address については **ADRS** と同じ値が用いられる。

このアルゴリズムにより，$node_{0,a}, node_{1,a}, \ldots, node_{k-1,a}$ を根とする k 個のマークル木が構成されている。公開鍵は pk である。秘密鍵は $sk_0, sk_1, \ldots, sk_{kt-1}$ である。

■署名アルゴリズム 長さ $k \cdot a$ bits のメッセージダイジェスト md をそれぞれ長さ a bits の k 個のブロック $md_0, md_1, \ldots, md_{k-1}$ に分割する。すなわち，$md = md_0 \| md_1 \| \cdots \| md_{k-1}$ である。さらに，md_i を 2 進数表記の非負整数とみなす。md の署名は $sk_{0 \cdot t + md_0}, sk_{1 \cdot t + md_1}, \ldots, sk_{(k-1)t + md_{k-1}}$ と，$0 \leq i \leq k-1$ について，$node_{i,a}$ を根とするマークル木の $node_{i \cdot t + md_i, 0}$ の認証パスである。詳細は NIST FIPS 205 [40] を参照のこと。

■検証アルゴリズム NIST FIPS 205 [40] では，検証に必須の，メッセージと署名の組から対応する公開鍵の候補を計算するアルゴリズムが示されている。詳細は NIST FIPS 205 [40] を参照のこと。

7.3.3.5 SLH-DSA

前節までの構成要素を用いて SLH-DSA の署名が構成される。SLH-DSA のパラメータは以下のとおりである。

- セキュリティパラメータ n（単位は Byte）
- hypertree のパラメータ $h, d, h'(= h/d)$
- FORS のパラメータ a, k
- WOTS$^+$ のパラメータ lg_w
- メッセージダイジェストの byte 長 $m = \lceil (h - h')/8 \rceil + \lceil h'/8 \rceil + \lceil (k \cdot a)/8 \rceil$

■鍵生成アルゴリズム $\mathbf{SK}.\mathrm{seed}, \mathbf{SK}.\mathrm{prf} \in \mathbb{B}^n$ はいずれも無作為に選択される。$\mathbf{PK}.\mathrm{seed} \in \mathbb{B}^n$ は無作為に選択される。$\mathbf{PK}.\mathrm{root} \in \mathbb{B}^n$ は hypertree の第 $(d-1)$ 層の XMSS 木の根である。秘密鍵は $\mathbf{SK}.\mathrm{seed}, \mathbf{SK}.\mathrm{prf}, \mathbf{PK}.\mathrm{seed}, \mathbf{PK}.\mathrm{root}$ である。公開鍵は $\mathbf{PK}.\mathrm{seed}, \mathbf{PK}.\mathrm{root}$ である。したがって，秘密鍵，公開鍵のサイズはそれぞれ，$4n$ Bytes，$2n$ Bytes である。

■署名アルゴリズム メッセージ M の署名は以下のように生成される。

1. $R := \mathbf{PRF}_{msg}(\mathbf{SK}.\mathrm{prf}, opt\_rand, M)$ とする。ここで，opt_rand を $\mathbb{B}^n$ の乱数とすることがデフォルトとされており，特にサイドチャネル攻撃が懸念される場合については強く推奨されているが，乱数生成器が利用可能でない場合は $opt\_rand = \mathbf{PK}.\mathrm{seed}$ とすることが許容されている。
2. $digest := \mathbf{H}_{msg}(R, \mathbf{PK}.\mathrm{seed}, \mathbf{PK}.\mathrm{root}, M)$ とする。$digest$ の最初の $\lceil (k \cdot a)/8 \rceil$ Bytes，次の $\lceil (h - h')/8 \rceil$ Bytes，その次の $\lceil h'/8 \rceil$ Bytes をそれぞれ md，整数 idx_{tree} の 2 進数表記，整数 idx_{leaf} の 2 進数表記とする。
3. hypertree の第 0 層の左から idx_{tree} 番目の XMSS 木の左から idx_{leaf} 番目の葉に対応する FORS の鍵を用いて md の先頭 $k \cdot a$ bits に対する署名を生成する。
4. 上の署名で用いられた FORS の公開鍵への hypertree による署名を生成する。

M の署名は R，md への FORS による署名，md への署名の検証に用いられる FORS の公開鍵への hypertree による署名からなる。したがって，署名のサイズは $(1 + k(a+1) + h + d \cdot len)n$ Bytes である。

SLH-DSA では，署名アルゴリズムに与えられるメッセージ M を署名対象の内容から生成する二つの方法が定められている。これらは pure 版，pre-hash 版と呼ばれている。署名アルゴリズムに対して，pure 版ではコンテクストと署名対象の内容とが与えられ，pre-hash 版ではコンテクストと署名対象の内容のハッシュ値とが与えられる。なお，コンテクストは長さが高々 255 Bytes の系列であり，デフォルトでは空列である。詳細については NIST FIPS 205 [40]

を参照のこと。

■検証アルゴリズム　署名アルゴリズムより容易に導出されるので，詳細については NIST FIPS 205 [40] を参照のこと。

7.3.3.6　パラメータの設定と安全性

SLH-DSA については，表 7.8 の 12 個のパラメータセットが示されている。この表の最左欄の名称は，使用されるハッシュ関数とセキュリティパラメータ n の bit 長を単位とした値を示している。さらに，s と f はそれぞれ，署名サイズ，計算時間が小さくなるよう定められたパラメータセットであることを示している。また，安全性レベルは NIST PQC 標準化プロジェクトの Call for Proposals に記された安全性強度のカテゴリである。これらのパラメータセットは，一組の公開鍵と秘密鍵により高々 2^{64} 個のメッセージが署名される場合の選択文書攻撃に対する存在偽造不能性を考慮して定められている。

表 7.8: SLH-DSA のパラメータセット。公開鍵長，署名長の単位は Byte である。

名称	n	h	d	h'	a	k	lg_w	m	安全性レベル	公開鍵長	署名長
SLH-DSA-SHA2-128s SLH-DSA-SHAKE-128s	16	63	7	9	12	14	4	30	レベル 1	32	7,856
SLH-DSA-SHA2-128f SLH-DSA-SHAKE-128f	16	66	22	3	6	33	4	34	レベル 1	32	17,088
SLH-DSA-SHA2-192s SLH-DSA-SHAKE-192s	24	63	7	9	14	17	4	39	レベル 3	48	16,224
SLH-DSA-SHA2-192f SLH-DSA-SHAKE-192f	24	66	22	3	8	33	4	42	レベル 3	48	35,664
SLH-DSA-SHA2-256s SLH-DSA-SHAKE-256s	32	64	8	8	14	22	4	47	レベル 5	64	29,792
SLH-DSA-SHA2-256f SLH-DSA-SHAKE-256f	32	68	17	4	9	35	4	49	レベル 5	64	49,856

Hülsing と Kudinov [23] は，SPHINCS$^+$ が EUF-CMA 安全性を満たすことをハッシュ関数 $\mathbf{T}_\ell$，$\mathbf{H}_{msg}$，擬似ランダム関数 $\mathbf{PRF}, \mathbf{PRF}_{msg}$ の以下の安全性に帰着している。

- $\mathbf{T}_\ell$ が以下の性質を満たすこと
  - single-function, multi-target collision resistance (SM-TCR)
  - single-function, multi-target preimage resistance (SM-PRE)
  - single-function, multi-target decisional second preimage resistance (SM-DSPR)
  - single-function, multi-target undetectability (SM-UD)
- $\mathbf{H}_{msg}$ が interleaved target subset resilience (ITSR) を満たすこと
- $\mathbf{PRF}, \mathbf{PRF}_{msg}$ が擬似ランダム関数 (PRF) であること

ここで，SM-TCR, SM-DSPR, ITSR は，$\mathbf{T}_\ell, \mathbf{H}_{msg}$ の構成に用いられるハッシュ関数の第二原像攻撃に対する安全性に基づく性質であり，SM-PRE は原像攻撃に対する安全性に基づく性質である。一方，SM-UD, PRF は，秘密鍵入力を有するハッシュ関数が擬似ランダム関数であることを要求する。

Barbosa ら [3] は，SPHINCS$^+$ で用いられている XMSS について，コンピュータで検証された安全性証明を与えて

いる。

さらに，NIST FIPS 205 [40] には，SLH-DSA の実装をサイドチャネル攻撃 [27] や故障攻撃 [1, 11, 19, 45] から保護するための注意が払われなければならないことが記されている。

7.3.3.7 ハッシュ関数の実現法

SLH-DSA の関数はすべて，SHAKE256，SHA-2 のうちのいずれかを用いて構成される。これらの構成は，SPHINCS$^+$ で simple な実現と呼ばれる構成であり，7.2.4 節で述べられたビットマスクは用いられていない。

SHAKE256 を用いた構成は以下のとおりである。

$$\begin{aligned}
\mathbf{H}_{msg}(R, \mathbf{PK}.\text{seed}, \mathbf{PK}.\text{root}, M) &:= \text{SHAKE256}(R \| \mathbf{PK}.\text{seed} \| \mathbf{PK}.\text{root} \| M, 8m) \\
\mathbf{PRF}(\mathbf{PK}.\text{seed}, \mathbf{SK}.\text{seed}, \mathbf{ADRS}) &:= \text{SHAKE256}(\mathbf{PK}.\text{seed} \| \mathbf{ADRS} \| \mathbf{SK}.\text{seed}, 8n) \\
\mathbf{PRF}_{msg}(\mathbf{SK}.\text{prf}, \mathit{opt_rand}, M) &:= \text{SHAKE256}(\mathbf{SK}.\text{prf} \| \mathit{opt_rand} \| M, 8n) \\
\mathbf{F}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_1) &:= \text{SHAKE256}(\mathbf{PK}.\text{seed} \| \mathbf{ADRS} \| M_1, 8n) \\
\mathbf{H}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_2) &:= \text{SHAKE256}(\mathbf{PK}.\text{seed} \| \mathbf{ADRS} \| M_2, 8n) \\
\mathbf{T}_\ell(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_\ell) &:= \text{SHAKE256}(\mathbf{PK}.\text{seed} \| \mathbf{ADRS} \| M_\ell, 8n)
\end{aligned}$$

安全性レベル 1 に対する SHA-2 を用いた構成は以下のとおりである。

$$\begin{aligned}
\mathbf{H}_{msg}(R, \mathbf{PK}.\text{seed}, \mathbf{PK}.\text{root}, M) &:= \text{MGF1-SHA-256}(R \| \mathbf{PK}.\text{seed} \| \text{SHA-256}(R \| \mathbf{PK}.\text{seed} \| \mathbf{PK}.\text{root} \| M, m)) \\
\mathbf{PRF}(\mathbf{PK}.\text{seed}, \mathbf{SK}.\text{seed}, \mathbf{ADRS}) &:= \mathsf{Trunc}_n(\text{SHA-256}(\mathbf{PK}.\text{seed} \| [0]_{64-n} \| \mathbf{ADRS}^c \| \mathbf{SK}.\text{seed})) \\
\mathbf{PRF}_{msg}(\mathbf{SK}.\text{prf}, \mathit{opt_rand}, M) &:= \mathsf{Trunc}_n(\text{HMAC-SHA-256}(\mathbf{SK}.\text{prf} \| \mathit{opt_rand} \| M)) \\
\mathbf{F}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_1) &:= \mathsf{Trunc}_n(\text{SHA-256}(\mathbf{PK}.\text{seed} \| [0]_{64-n} \| \mathbf{ADRS}^c \| M_1)) \\
\mathbf{H}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_2) &:= \mathsf{Trunc}_n(\text{SHA-256}(\mathbf{PK}.\text{seed} \| [0]_{64-n} \| \mathbf{ADRS}^c \| M_2)) \\
\mathbf{T}_\ell(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_\ell) &:= \mathsf{Trunc}_n(\text{SHA-256}(\mathbf{PK}.\text{seed} \| [0]_{64-n} \| \mathbf{ADRS}^c \| M_\ell))
\end{aligned}$$

安全性レベル 3, 5 に対する SHA-2 を用いた構成は以下のとおりである。

$$\begin{aligned}
\mathbf{H}_{msg}(R, \mathbf{PK}.\text{seed}, \mathbf{PK}.\text{root}, M) &:= \text{MGF1-SHA-512}(R \| \mathbf{PK}.\text{seed} \| \text{SHA-512}(R \| \mathbf{PK}.\text{seed} \| \mathbf{PK}.\text{root} \| M, m)) \\
\mathbf{PRF}(\mathbf{PK}.\text{seed}, \mathbf{SK}.\text{seed}, \mathbf{ADRS}) &:= \mathsf{Trunc}_n(\text{SHA-256}(\mathbf{PK}.\text{seed} \| [0]_{64-n} \| \mathbf{ADRS}^c \| \mathbf{SK}.\text{seed})) \\
\mathbf{PRF}_{msg}(\mathbf{SK}.\text{prf}, \mathit{opt_rand}, M) &:= \mathsf{Trunc}_n(\text{HMAC-SHA-512}(\mathbf{SK}.\text{prf} \| \mathit{opt_rand} \| M)) \\
\mathbf{F}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_1) &:= \mathsf{Trunc}_n(\text{SHA-256}(\mathbf{PK}.\text{seed} \| [0]_{64-n} \| \mathbf{ADRS}^c \| M_1)) \\
\mathbf{H}(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_2) &:= \mathsf{Trunc}_n(\text{SHA-512}(\mathbf{PK}.\text{seed} \| [0]_{128-n} \| \mathbf{ADRS}^c \| M_2)) \\
\mathbf{T}_\ell(\mathbf{PK}.\text{seed}, \mathbf{ADRS}, M_\ell) &:= \mathsf{Trunc}_n(\text{SHA-512}(\mathbf{PK}.\text{seed} \| [0]_{128-n} \| \mathbf{ADRS}^c \| M_\ell))
\end{aligned}$$

ここで，MGF1-SHA-256，MGF1-SHA-512 は RFC 8017 [37] の Appendix B.2.1 に記載されている MGF1 であり，HMAC-SHA-256，HMAC-SHA-512 は FIPS 198-1 [41] の HMAC である。また，$\mathsf{Trunc}_l(x)$ は byte 系列 x の左端から l Bytes を出力する関数である。さらに，$\mathbf{ADRS}^c$ は $\mathbf{ADRS}$ の layer address，tree address，type をそれぞれ 1 Byte，8 Bytes，1 Byte に短縮した長さ 22 Bytes のアドレスである。

SPHINCS$^+$ では，当初，SHA-2 を用いた実現で SHA-256 のみが用いられていたが，SHA-256 を用いた実現では安全性のレベル 5 が達成できないことを示す攻撃 [42] が示されたことから，SPHINCS$^+$ v.3.1 では，安全性レベル 3, 5 について，$\mathbf{H}_{msg}$, $\mathbf{PRF}_{msg}$, $\mathbf{H}$, $\mathbf{T}_\ell$ が SHA-512 を用いて実現されることとなり，SLH-DSA でもそれに従っている。

7.4 ハッシュ関数に基づく署名技術に関するまとめ

本章では，ハッシュ関数に基づく署名技術として，Lighton-Micali hash-based signatures, XMSS, SLH-DSA を取り上げた。これらはいずれも 7.2 節で述べた代表的なハッシュ関数に基づく署名方式に基づく構造を有する。Lighton-

Micali hash-based signatures [33] と XMSS [21] は NIST の推奨アルゴリズムであり [12]，SLH-DSA は NIST PQC 標準化プロジェクトで選出された SPHINCS$^+$[2] に基づく標準アルゴリズムである。

ハッシュ関数に基づく署名技術の安全性はハッシュ関数の第二原像攻撃に対する安全性に依存しているが，XMSS, SLH-DSA については，秘密鍵入力を有するハッシュ関数が擬似ランダム関数であることにも依存する。さらに，ビットマスクの生成についてはハッシュ関数がランダムオラクルであることが仮定される。また，偽造攻撃の計算量は，ハッシュ関数がランダムオラクルであることを仮定して見積もられている。

ハッシュ関数に基づく署名技術については，stateful であること，すなわち，各メッセージの署名に用いられる 1 回署名の秘密鍵を 2 回以上使用することのないよう管理しなければならないことが問題であった。Lighton-Micali hash-based signatures と XMSS はいずれもハッシュ関数に基づく stateful な署名方式であり，それらを推奨アルゴリズムとする NIST SP 800-208 [12] には，ハッシュ関数に基づく stateful な署名方式は一般的な使用には適するものでなく，近い将来に実装が必要であり，その実装が長期間の使用を予定されており，かつ，使用開始後に他の署名方式への移行が実用的でないような応用での使用が意図されていると述べられている。

SLH-DSA は XMSS の設計で得られた知見に基づいて設計されており，HSS, XMSS^{MT} と同様の構造を有するが，各メッセージの署名に一つの秘密鍵で数回署名可能な FORS を用いることによって署名可能な回数を増加させることにより，stateless であることを達成している。

第 7 章の参照文献

[1] D. Amiet, L. Leuenberger, A. Curiger, P. Zbinden. FPGA-based SPHINCS$^+$ Implementations: Mind the Glitch. DSD. IEEE, 2020, pp. 229–237.

[2] J.-P. Aumasson et al. SPHINCS$^+$ Submission to the NIST post-quantum project, v.3.1. `https://sphincs.org/data/sphincs+-r3.1-specification.pdf`. 2022-06. (2024-03-08 閲覧).

[3] M. Barbosa, F. Dupressoir, B. Grégoire, A. Hülsing, M. Meijers, P.-Y. Strub. Machine-Checked Security for XMSS as in RFC 8391 and SPHINCS$^+$. CRYPTO (5). Vol. 14085. Lecture Notes in Computer Science. Springer, 2023, pp. 421–454.

[4] E. Barker, J. Kelsey. Recommendation for Random Number Generation Using Deterministic Random Bit Generators. NIST SP 800-90A Rev. 1, `https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf`. 2015-06.

[5] E. Barker, J. Kelsey, K. McKay, A. Roginsky, M. S. Turan. Recommendation for Random Bit Generator (RBG) Constructions. NIST SP 800-90C (4th public draft), `https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90C.4pd.pdf`. 2024-07. (2025-02-17 閲覧).

[6] D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, P. Schwabe. The SPHINCS$^+$ Signature Framework. CCS. ACM, 2019, pp. 2129–2146.

[7] D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, P. Schwabe. The SPHINCS+ Signature Framework. Cryptology ePrint Archive, Paper 2019/1086. 2019. `https://eprint.iacr.org/2019/1086`.

[8] D. J. Bernstein et al. SPHINCS: Practical Stateless Hash-Based Signatures. EUROCRYPT (1). Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 368–397.

[9] G. Bertoni, J. Daemen, M. Peeters, G. Van Assche. Sponge functions. ECRYPT Hash Workshop. 2007.

[10] J. Buchmann, E. Dahmen, A. Hülsing. XMSS – A Practical Forward Secure Signature Scheme Based on Minimal Security Assumptions. PQCrypto. Vol. 7071. Lecture Notes in Computer Science. Springer, 2011, pp. 117–129.

[11] L. Castelnovi, A. Martinelli, T. Prest. Grafting Trees: A Fault Attack Against the SPHINCS Framework. PQCrypto. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 165–184.

[12] D. Cooper, D. Apon, Q. Dang, M. Davidson, M. Dworkin, C. Miller. Recommendation for Stateful Hash-Based Signature Schemes. NIST SP 800-208, `https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-208.pdf`. 2020-10.

[13] E. Dahmen, K. Okeya, T. Takagi, C. Vuillaume. Digital Signatures Out of Second-Preimage Resistant Hash Functions. PQCrypto. Vol. 5299. Lecture Notes in Computer Science. Springer, 2008, pp. 109–123.

[14] I. Damgård. A Design Principle for Hash Functions. CRYPTO. Vol. 435. Lecture Notes in Computer Science. Springer, 1989, pp. 416–427.

[15] W. Diffie, M. E. Hellman. New directions in cryptography. IEEE Trans. Inf. Theory. Vol. 22, Num. 6 (1976), pp. 644–654.

[16] C. Dods, N. P. Smart, M. Stam. Hash Based Digital Signature Schemes. IMACC. Vol. 3796. Lecture Notes in Computer Science. Springer, 2005, pp. 96–115.

[17] E. Eaton. Leighton-Micali Hash-Based Signatures in the Quantum Random-Oracle Model. SAC. Vol. 10719. Lecture Notes in Computer Science. Springer, 2017, pp. 263–280.

[18] S. Fluhrer. Further Analysis of a Proposed Hash-Based Signature Standard. Cryptology ePrint Archive, Paper 2017/553. 2017. `https://eprint.iacr.org/2017/553`.

[19] A. Genêt. On Protecting SPHINCS+ Against Fault Attacks. IACR Trans. Cryptogr. Hardw. Embed. Syst. Vol. 2023, Num. 2 (2023), pp. 80–114.

[20] L. K. Grover. A fast quantum mechanical algorithm for database search. STOC. ACM, 1996, pp. 212–219.

[21] A. Huelsing, D. Butin, S.-L. Gazdag, J. Rijneveld, A. Mohaisen. XMSS: eXtended Merkle Signature Scheme. RFC 8391, `https://www.rfc-editor.org/info/rfc8391`. 2018-05.

[22] A. Hülsing. W-OTS$^+$ – Shorter Signatures for Hash-Based Signature Schemes. AFRICACRYPT. Vol. 7918. Lecture Notes in Computer Science. Springer, 2013, pp. 173–188.

[23] A. Hülsing, M. A. Kudinov. Recovering the Tight Security Proof of SPHINCS$^+$. ASIACRYPT (4). Vol. 13794. Lecture Notes in Computer Science. Springer, 2022, pp. 3–33.

[24] A. Hülsing, L. Rausch, J. Buchmann. Optimal Parameters for XMSS MT. CD-ARES Workshops. Vol. 8128. Lecture Notes in Computer Science. Springer, 2013, pp. 194–208.

[25] A. Hülsing, J. Rijneveld, F. Song. Mitigating Multi-target Attacks in Hash-Based Signatures. Public Key Cryptography (1). Vol. 9614. Lecture Notes in Computer Science. Springer, 2016, pp. 387–416.

[26] P. Kampanakis, S. Fluhrer. LMS vs XMSS: Comparion of two Hash-Based Signature Standards. Cryptology ePrint Archive, Paper 2017/349. 2017. `https://eprint.iacr.org/2017/349`.

[27] M. J. Kannwischer, A. Genêt, D. Butin, J. Krämer, J. Buchmann. Differential Power Analysis of XMSS and SPHINCS. COSADE. Vol. 10815. Lecture Notes in Computer Science. Springer, 2018, pp. 168–188.

[28] J. Katz. Analysis of a Proposed Hash-Based Signature Standard. SSR. Vol. 10074. Lecture Notes in Computer Science. Springer, 2016, pp. 261–273.

[29] J. Kelsey, B. Schneier. Second Preimages on n-Bit Hash Functions for Much Less than 2^n Work. EUROCRYPT. Vol. 3494. Lecture Notes in Computer Science. Springer, 2005, pp. 474–490.

[30] P. Lafrance, A. Menezes. On the security of the WOTS-PRF signature scheme. Adv. Math. Commun. Vol. 13, Num. 1 (2019), pp. 185–193.

[31] L. Lamport. Constructing digital signatures from a one-way function. SRI International Technical Report, CSL-98. 1979-10.

[32] F. T. Leighton, S. Micali. Large provably fast and secure digital signature schemes based on secure hash functions. 1995-07. US Patent 5,432,852.

[33] D. McGrew, M. Curcio, S. Fluhrer. Leighton-Micali Hash-Based Signatures. RFC 8554, `https://www.rfc-editor.org/info/rfc8554`. 2019-04.

[34] R. Merkle. Secrecy, Authentication, and Public Key Systems. PhD thesis. Stanford University, 1979. `https://www.merkle.com/papers/Thesis1979.pdf`.

[35] R. C. Merkle. A Certified Digital Signature. CRYPTO. Vol. 435. Lecture Notes in Computer Science. Springer, 1989, pp. 218–238.

[36] R. C. Merkle. One Way Hash Functions and DES. CRYPTO. Vol. 435. Lecture Notes in Computer Science. Springer, 1989, pp. 428–446.

[37] K. Moriarty, B. Kaliski, J. Jonsson, A. Rusch. PKCS #1: RSA Cryptography Specifications Version 2.2. RFC 8017, https://www.rfc-editor.org/info/rfc8017. 2016-11.

[38] NIST. Secure Hash Standard (SHS). NIST FIPS 180-4, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf. 2015-08.

[39] NIST. SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. NIST FIPS 202, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf. 2015-08.

[40] NIST. Stateless Hash-Based Digital Signature Standard. NIST FIPS 205, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.205.pdf. 2024-08.

[41] NIST. The Keyed-Hash Message Authentication Code (HMAC). NIST FIPS 198-1, https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.198-1.pdf. 2008-07.

[42] R. A. Perlner, J. Kelsey, D. A. Cooper. Breaking Category Five SPHINCS$^+$ with SHA-256. PQCrypto. Vol. 13512. Lecture Notes in Computer Science. Springer, 2022, pp. 501–522.

[43] L. Reyzin, N. Reyzin. Better than BiBa: Short One-Time Signatures with Fast Signing and Verifying. ACISP. Vol. 2384. Lecture Notes in Computer Science. Springer, 2002, pp. 144–153.

[44] M. S. Turan, E. Barker, J. Kelsey, K. McKay, M. Baish, M. Boyle. Recommendation for the Entropy Sources Used for Random Bit Generation. NIST SP 800-90B, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf. 2018-01.

[45] A. Wagner, V. Wesselkamp, F. Oberhansl, M. Schink, E. Strieder. Faulting Winternitz One-Time Signatures to Forge LMS, XMSS, or SPHINCS$^+$ Signatures. PQCrypto. Vol. 14154. Lecture Notes in Computer Science. Springer, 2023, pp. 658–687.

耐量子計算機暗号の研究動向調査報告書
[CRYPTREC TR-2001-2024]
不許複製　禁無断転載
発行日：2025 年 3 月 31 日（第 1 版）
発行者
・〒184-8795
東京都小金井市貫井北町四丁目２番１号
国立研究開発法人情報通信研究機構
（サイバーセキュリティ研究所　セキュリティ基盤研究室）
NATIONAL INSTITUTE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY
4-2-1 NUKUI-KITAMACHI, KOGANEI
TOKYO, 184-8795 JAPAN
・〒113-6591
東京都文京区本駒込二丁目２８番８号
独立行政法人情報処理推進機構
（セキュリティセンター 技術評価部 暗号グループ）
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN
2-28-8 HONKOMAGOME, BUNKYO-KU
TOKYO, 113-6591 JAPAN