調査レポート

耐量子暗号アルゴリズム ML-DSA の解説

情報通信研究部
社会基盤技術チーム　シニアコンサルタント
玉垣勇樹

量子コンピュータが実用化されると，RSA や ECDSA など従来の公開鍵暗号は解読可能になり，通信の安全性が失われるおそれがある．本稿では，National Institute of Standards and Technology（NIST）が進める耐量子暗号標準化プロジェクトの最新動向を俯瞰し，特に 2024 年に FIPS 204 として標準化された格子ベース署名方式である ML-DSA を中心に解説する．具体的には，ML-DSA の Fiat-Shamir with Aborts 型ゼロ知識識別プロトコルの流れを，Schnorr 署名との対比を通じて説明する．最後に，同一メッセージに対する署名の偽造困難性を SelfTargetMSIS 問題の求解困難性へ帰着させることで，署名の偽造困難性を導く安全性証明の要点を解説する．

1　はじめに

量子計算は，従来のコンピュータでは難しい多変数の複雑な相互作用を効率よく処理できる可能性を持つ．この能力は社会に恩恵をもたらす一方，広く利用されてきた一部の暗号の解読を招くおそれがある．この脅威に備える新しい暗号技術は「耐量子暗号」，または「Post-Quantum Cryptography（PQC）」と呼ばれる．本稿では，NIST が進める PQC 標準化プロジェクトを概観し，2024 年に FIPS 204 として標準化された Module-Lattice-Based Digital Signature Algorithm（ML-DSA）を中心に解説する．

2　PQC に移行すべき暗号

暗号技術は，鍵の有無や使い方により分類できる．ひとつは暗号化と復号に同一の鍵を用いる対称鍵暗号（共通鍵暗号）であり，ブロック暗号の AES などがある．鍵を用いない暗号プリミティブとして暗号ハッシュ関数（SHA-256 など）も重要な技術である．もうひとつは暗号化と復号に異なる鍵を用いる非対称鍵暗号（公開鍵暗号）であり，RSA 暗号やデジタル署名の ECDSA などが代表的である．

対称鍵暗号や暗号ハッシュ関数は，複雑な変換を繰り返し適用することにより安全性を獲得する．これらの技術に対する量子計算を用いた汎用的な攻撃として，Grover の探索アルゴリズムが知られる．これは，鍵や元のメッセージを見つけるための総当たり探索を高速化するが，必要な試行回数をN回から$\sqrt{N}$回に削減する程度にとどまる．したがって，対称鍵暗号は鍵長を，暗号ハッシュ関数は出力長を十分に長くすることで量子計算への耐性を確保できる．

非対称鍵暗号は数学的計算困難性に安全性を委ねている．現在普及している RSA や ECDSA は，それぞれ合成数$\boldsymbol{N}$の素因数分解や離散対数問題に基づいている．入力のビット長を$\boldsymbol{n = log_2 N}$とすると，Shor のアルゴリズムにより量子計算機上で概ね$\boldsymbol{O(n^3)}$の計算量で解けることが知られており，将来的にこれらの暗号方式は危殆化するおそれがある．したがって，非対称鍵暗号は解読される前に計画的に PQC へ移行する必要がある．

3　NIST による PQC 標準化

NIST では，2016 年から PQC の標準化プロジェクトが開始され，2025 年 6 月時点で 5 つの暗号アルゴリズムが標準化対象として採択された [1-5]．PQC は Key Encapsulation Mechanism（KEM）と呼ばれる秘密鍵を安全に配送する仕組みとデジタル署名に大別される．**表 1** および**表 2** にそれぞれで採択された暗号を一覧する．デジタル署名に関して，署名サイズの小型化や安全性の異なる選択肢を増やす目的で追加評価 [6]が進められている．

表 1　NIST による KEM の標準化アルゴリズム

ベースアルゴリズム名	優先度	暗号種類	標準化後アルゴリズム名	標準規格名
CRYSTALS-KYBER	Primary	格子ベース暗号	ML-KEM	FIPS 203
HQC	Backup	符号ベース暗号	2027 年中に標準化予定	

表 2　NIST によるデジタル署名の標準化アルゴリズム，*2025 年 6 月時点で未公表

ベースアルゴリズム名	優先度	暗号種類	標準化後アルゴリズム名	標準規格名
CRYSTALS-Dilithium	Primary	格子ベース暗号	ML-DSA	FIPS 204
SPHINCS+	Backup	ハッシュベース暗号	SLH-DSA	FIPS 205
FALCON	Backup	格子ベース暗号	FN-DSA(仮称)*	FIPS 206 (仮称)*

4　ML-DSA の概要

本節では，FIPS 204 として標準化されたデジタル署名方式，ML-DSA を解説する．ML-DSA の安全性は，格子ベース暗号における二つの計算問題，Module Learning With Errors（MLWE）問題と Module Short Integer Solution（MSIS）問題の困難性を根拠としている．具体的には，秘密鍵の復元困難性は MLWE 問題に，署名の偽造困難性は MSIS 問題に，それぞれ基づく．この署名方式を理解するため，本節ではまず，その原型と言える Schnorr 署名を手がかりに，ゼロ知識識別プロトコルを説明する．そのうえで ML-DSA アルゴリズム全体の概要を示し，最後に署名偽造耐性の数学的基盤である MSIS 問題と ML-DSA の関係を解説する．

以降では，n, k, l, K, Lは自然数，$\mathbb{Z}_q$は要素集合$\{0, 1, \dots, q-1\}$からなる，素数qを法とする整数環とする．$\mathbb{Z}_q^n$は$\mathbb{Z}_q$の要素のn組全体とする．$R_q = \mathbb{Z}_q[X]/(X^n + 1)$は多項式剰余環とする．記号「‖」はバイト列もしくはビット列の連結を，$\|\cdot\|_\infty \le \gamma$という制約は，各要素の絶対値が正の整数$\gamma$以下であることを表す．

4.1 ゼロ知識識別プロトコル概論

ゼロ知識証明とは，証明者が検証者に対して，ある主張（例えば「自分が秘密鍵を知っていること」）を暴露することなく真偽だけを納得させる手法である．特に，本人認証の主張を証明するための対話的な手順を，ゼロ知識識別プロトコルと呼ぶ．ML-DSA では Schnorr 署名と同系のゼロ知識識別プロトコルを用いており，署名の生成から検証までの一連の流れも Schnorr 署名と類似している．以下では，(1) 対話型のゼロ知識識別プロトコル，(2) それを非対話化した署名生成・検証の順に概要を示す．

対話型ゼロ知識識別プロトコルの流れの模式図を**図 1** に示す．

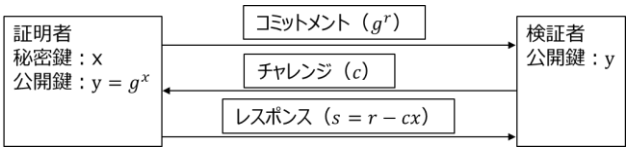

図 1　Schnorr 署名の対話型プロトコル

証明者は秘密鍵$x \in \mathbb{Z}_q$と公開鍵$y = g^x \in \mathbb{G}$（$\mathbb{G}$:位数qの巡回群，g:既知の生成元)を，検証者は公開鍵yを有しているとする．下記のステップで検証を行う．

1. コミットメント
   証明者はランダムな整数$r \in \mathbb{Z}_q$を生成してg^rを検証者に送る．
2. チャレンジ
   検証者はランダムな整数$c \in \mathbb{Z}_q$を生成して証明者に送る．
3. レスポンス
   証明者は$s \in \mathbb{Z}_q = r - cx$を検証者に送る．
4. 検証
   検証者は$g^s \cdot y^c$とg^rが一致することを確認する．

チャレンジが検証者側でランダムに生成されるため，レスポンスから秘密鍵に関する情報は統計的に得られない．正当な証明者が手順どおりに実行すれば式(1)のように$g^s \cdot y^c$とg^rは一致する．

$$g^s \cdot y^c = g^s \cdot (g^x)^c = g^{(s+xc)} = g^{(r-cx+xc)} = g^r \qquad (1)$$

次に，対話型ゼロ知識識別プロトコルを Fiat-Shamir（FS）変換により非対話化した流れを**図 2** に示す.

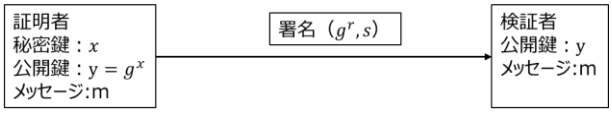

図 2　Schnorr 署名の非対話型プロトコル

署名生成および署名検証の手順は以下である.

署名生成

1. コミットメント
   ランダムな整数$r \in \mathbb{Z}_q$を生成し，g^rを計算する.
2. チャレンジ
   g^rとメッセージ$m \in \{0,1\}^*$（任意のビット列）を連結し，ハッシュ関数$H()$でハッシュ化したものを$c \in \mathbb{Z}_q = H(g^r \parallel m)$とする.
3. レスポンス
   $s = r - cx$として，(g^r, s)を検証者に送る.

署名検証

1. メッセージと署名からチャレンジc'を再計算する.
2. $g^s \cdot y^{c'}$とg^rが一致することを確認する.

4.2 ML-DSA のプロトコル

FS 変換は「コミットメント→ハッシュ関数によるチャレンジ生成→レスポンス」の手順で対話型証明を非対話化する．一方，Fiat-Shamir with Aborts（FSA）は，レスポンスがあらかじめ定めた「短さ」の範囲を逸脱した場合に試行を打ち切り（Abort），コミットメント生成からやり直す処理を追加することで，応答分布を秘密鍵から統計的に切り離し，より厳密なゼロ知識性を保証する．両者の差は Abort の有無と，それによる安全性（統計的ゼロ知識）の強化にある．ML-DSA におけるゼロ知識識別プロトコルの模式図を**図 3** に示す．ここでは FS 変換ではなく，FSA 型のプロトコルである点に注意されたい.

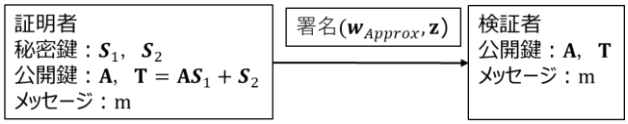

図 3 ML-DSA の Fiat–Shamir with Aborts 型プロトコル

証明者は小さい係数で構成された秘密鍵$\boldsymbol{S}_1 \in \mathbb{Z}_q^{L\times n}$，$\boldsymbol{S}_2 \in \mathbb{Z}_q^{K\times n}$と公開鍵として一様ランダムな行列$\boldsymbol{A} \in \mathbb{Z}_q^{K\times L}$および$\boldsymbol{T} \in \mathbb{Z}_q^{K\times n} = \boldsymbol{A}\boldsymbol{S}_1 + \boldsymbol{S}_2$を，検証者は$(\boldsymbol{A}, \boldsymbol{T})$を知っている状態とする．署名生成および署名検証の大まかな手順は以下である.

署名生成

1. コミットメント
   小さい係数で構成された乱数ベクトル$\boldsymbol{y} \in \mathbb{Z}_q^L$及び$\boldsymbol{y_2} \in \mathbb{Z}_q^K$を生成し，$\boldsymbol{w}_{Approx} \in \mathbb{Z}_q^K = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{y_2}$を計算する.
2. チャレンジ
   メッセージと$\boldsymbol{w}_{Approx}$から小さい係数で構成された$\boldsymbol{c} \in \mathbb{Z}_q^n$を計算する.
3. レスポンス
   $\boldsymbol{z} \in \mathbb{Z}_q^L = \boldsymbol{y} + \boldsymbol{S}_1\boldsymbol{c}$を計算する.
4. Abort 判定
   「短さ」を測る閾値γを設定し，$\|\boldsymbol{z}\|_\infty \le \gamma$が成立しなければ最初からやり直す. 成立した場合，$(\boldsymbol{w}_{Approx}, \boldsymbol{z})$を検証者に送る.

署名検証

1. メッセージと署名からチャレンジ$\boldsymbol{c'}$を再計算する.
2. $\boldsymbol{A}\boldsymbol{z} - \boldsymbol{T}\boldsymbol{c'} \approx \boldsymbol{w}_{Approx}$が成立することを確認する.

$\boldsymbol{w}_{Approx}$全体を署名とすることはサイズが大きくなり現実的ではない．鍵長や署名サイズの削減と処理速度の向上を目的として，モジュール構造付き行列を用いるなどの工夫がなされる. 以降では，$\boldsymbol{A} \in R_q^{k\times l}$，$\boldsymbol{s}_1 \in R_q^L$，$\boldsymbol{s}_2 \in R_q^k$，$\boldsymbol{y} \in R_q^l$，$c \in R_q$，$l = L/n$，$k = K/n$に置きかえる．公開鍵$\boldsymbol{t} \in R_q^k = \boldsymbol{A}\boldsymbol{s}_1 + \boldsymbol{s}_2$であるが，各係数について$\boldsymbol{t} = \boldsymbol{t}_1 \cdot \boldsymbol{2^d} + \boldsymbol{t}_0$のように上位 d ビット$\boldsymbol{t}_1$と下位ビット$\boldsymbol{t}_0$に分解する. 決定論的な手続きにより$\boldsymbol{A}$を再生成する乱数シード$\rho$を$\boldsymbol{A}$そのものの代わりに公開する．したがって公開鍵は$(\rho, \boldsymbol{t}_1)$として格納し，公開鍵の鍵長を削減する．ML-DSA の実際のゼロ知識識別プロトコルの模式図を**図 4** に示す.

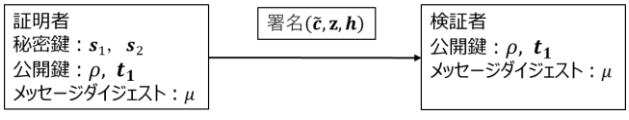

図 4　ML-DSA の実際の署名生成・検証プロトコル

署名生成と署名検証は概ね次の手順で行う．なお，証明者側で行ういずれかの Abort 判定に失敗した場合は，乱数ベクトル$\boldsymbol{y}$の生成からやり直す．

署名生成

1. コミットメント
   $\|\boldsymbol{y}\|_\infty \le \gamma_1 - 1$を満たす乱数ベクトル$\boldsymbol{y}$を生成する．$\gamma_1$は 2 のべき乗の定数である．$\boldsymbol{w} \in R_q^k = \boldsymbol{Ay}$を計算し，上位ビット$\boldsymbol{w_1} = \mathrm{HighBits}(\boldsymbol{w})$を得る．HighBits(·)は，各係数を格子幅$2\gamma_2$の格子点（$\gamma_2$は整数）へ丸めた結果を返す．これは，各係数を$2\gamma_2$で割り，商を返す操作に似ている．
2. チャレンジ
   メッセージmと公開鍵のハッシュ値などからメッセージダイジェストμを作成し，$\tilde{c} = H(\mu \| w1Encode(\boldsymbol{w_1}))$を求める．$w1Encode(\cdot)$は多項式ベクトル$\boldsymbol{w_1}$をハッシュ関数で扱えるよう，一意のバイト列に変換する公開関数である．$\tilde{c}$を乱数シードとして，決定論的な手続きによりチャレンジ多項式$c \in R_q$（各係数は$\{-1, 0, 1\}$，ハミング重みτ）を作成する．ハミング重みは，多項式に含まれる非ゼロ係数の個数である．
3. レスポンス
   $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{s_1}c$を計算する．
4. Abort 判定①（レスポンスの範囲チェック）
   $\|\boldsymbol{z}\|_\infty < \gamma_1 - \beta$を満たすか検証する．$\beta$は安全マージンとして機能する自然数である．
5. Abort 判定②（丸め誤差チェック）
   $\boldsymbol{r_0} = \mathrm{LowBits}(\boldsymbol{w} - c\boldsymbol{s_2})$を計算し，$\|\boldsymbol{r_0}\|_\infty < \gamma_2 - \beta$を満たすか検証する．LowBits(·)は，各係数を格子幅が$2\gamma_2$の格子に射影した際の「格子点からのずれ」（丸め誤差）を返す．
6. Abort 判定③（ヒント生成チェック）
   証明者は，検証者が式(2)より$\boldsymbol{w'} = \boldsymbol{w} - c\boldsymbol{s_2} + c\boldsymbol{t_0}$から$\boldsymbol{w_1}$を再構築するヒント$\boldsymbol{h} \in R_2^k$を用意する．ヒント$\boldsymbol{h}$は$\boldsymbol{w} - c\boldsymbol{s_2} + c\boldsymbol{t_0} (= \boldsymbol{Az} - c\boldsymbol{t_1} \cdot 2^{\mathrm{d}})$と$\boldsymbol{w} - c\boldsymbol{s_2} (= \boldsymbol{Az} - c\boldsymbol{t})$の各係数が，格子幅が$2\gamma_2$の格子に丸めた際に異なる格子点になる場合は 1，同じ格子点になる場合は 0 のビット列である．$\|c\boldsymbol{t_0}\|_\infty < \gamma_2$かつ$\boldsymbol{h}$のハミング重みは自然数$\omega$より小さいかを検証する．
7. 署名$(\tilde{c}, \boldsymbol{z}, \boldsymbol{h})$を検証者に送る．

署名検証

1. チャレンジ$\tilde{c}$からチャレンジcを作成する．次に，$\boldsymbol{w'} = \boldsymbol{Az} - c\boldsymbol{t_1} \cdot 2^{\mathrm{d}}$を計算し，ヒント$\boldsymbol{h}$と$\boldsymbol{w'}$から$\boldsymbol{w_1'} = UseHint(\boldsymbol{h}, \boldsymbol{w'})$を計算する．$UseHint(\cdot)$は公開鍵の一部（$\boldsymbol{t_0}$：$\boldsymbol{t}$の下位ビット）が省略されていることによる丸め誤差を，ヒント$\boldsymbol{h}$によって補正し，証明者が計算した元のコミットメント$\boldsymbol{w_1}$と同じ値を検証者側で再現する関数である．
2. $\tilde{c'} = H(\mu \| w1Encode(\boldsymbol{w_1'}))$を求め，$\tilde{c'} = \tilde{c}$となることを確認する．正当な署名であれば，検証者がこの手順で計算した$\tilde{c'}$が，証明者が内部的に計算した$\tilde{c}$と一致するようにパラメータが設計されている．詳細は FIPS 204[7]を参照頂きたい．

$$
\begin{aligned}
\boldsymbol{w'} &= \boldsymbol{Az} - c\boldsymbol{t_1} \cdot 2^{\mathrm{d}} \\
&= \boldsymbol{A}(\boldsymbol{y} + \boldsymbol{s_1}c) - c\boldsymbol{t_1} \cdot 2^{\mathrm{d}} \\
&= \boldsymbol{Ay} + c\boldsymbol{As_1} - c\boldsymbol{t_1} \cdot 2^{\mathrm{d}} \\
&= \boldsymbol{Ay} + c(\boldsymbol{t} - \boldsymbol{s_2}) - c\boldsymbol{t_1} \cdot 2^{\mathrm{d}} \\
&= \boldsymbol{Ay} - c\boldsymbol{s_2} + c(\boldsymbol{t} - \boldsymbol{t_1} \cdot 2^{\mathrm{d}}) \\
&= \boldsymbol{Ay} - c\boldsymbol{s_2} + c\boldsymbol{t_0} \\
&= \boldsymbol{w} - c\boldsymbol{s_2} + c\boldsymbol{t_0}
\end{aligned}
\qquad (2)
$$

4.3 MSIS 問題

署名の偽造困難性の基盤となる MSIS 問題について紹介する．一様ランダムな行列$\boldsymbol{A} \in R_q^{k \times l}$および単位行列$\boldsymbol{I} \in R_q^{k \times k}$に対して式(3)が成り立つ非ゼロのベクトル$\boldsymbol{x} \in R_q^{k+l}$を求める問題である．

$$
\begin{aligned}
[\boldsymbol{I} \,|\boldsymbol{A}] \cdot \boldsymbol{x} &= 0 \\
\|\boldsymbol{x}\|_\infty &\le \gamma
\end{aligned}
\qquad (3)
$$

γは行列$\boldsymbol{A}$の要素と比較して十分小さい整数である．$\|\boldsymbol{x}\|_\infty \le \gamma$の制約がなければ，連立一次方程式の問題に帰着するが，この制約により解空間は格子点のうち，原点近傍に限定され，効率的に解くアルゴリズムは知られていない．

4.4 MSIS 問題と ML-DSA の署名偽造困難性

ML-DSA の署名偽造困難性が MSIS 問題の困難性に基づくことを解説する．本稿では，同一メッセージに対する署名偽造に限定して直観的な説明を行うが，実際の安全性証明 [8]は攻撃者が取り得る一般的な偽造シナリオを含めて安全性が保証されている．

攻撃者が同一のメッセージに対してレスポンスが異なる有効な署名を作成できたと仮定する．具体的には署名 1: $(\tilde{c}, \boldsymbol{z_1}, \boldsymbol{h_1})$と，署名 2: $(\tilde{c}, \boldsymbol{z_2}, \boldsymbol{h_2})$を作成できたとする．両方の署名が検証に通るならば，両署名は同一のチャレンジcをもつことを意味する．

署名検証プロセスでは，署名と公開鍵からコミットメントの上位ビット$\boldsymbol{w_1'}$を再計算する．2 つの署名がどちらも有効であるならば，再計算される$\boldsymbol{w_1'}$は同一でなければならない．つまり，$\boldsymbol{Az_1 - ct_1 \cdot 2^d}$と$\boldsymbol{Az_2 - ct_1 \cdot 2^d}$はどちらも元のコミットメントをヒントを用いて正しく復元できる程度に近いベクトルでなければならない．したがってこれら 2 つのベクトルの差分は，下位ビットにしか非ゼロ成分を持たない「短い」(係数の絶対値が小さい) ベクトルになる．これを$\boldsymbol{v}$とおくと，式(4)が成り立つ．

$$(\boldsymbol{Az_1} - \boldsymbol{ct_1} \cdot 2^d) - (\boldsymbol{Az_2} - \boldsymbol{ct_1} \cdot 2^d) = \boldsymbol{A}(\boldsymbol{z_1} - \boldsymbol{z_2}) = \boldsymbol{v} \quad (4)$$

$\boldsymbol{z_1}$と$\boldsymbol{z_2}$は署名生成時の Abort 判定を通過した「短い」ベクトルであり，その差$\boldsymbol{u} = (\boldsymbol{z_2} - \boldsymbol{z_1})$も「短い」ベクトルである．式(4)から$\boldsymbol{v} = \boldsymbol{A}(\boldsymbol{z_1} - \boldsymbol{z_2}) = -\boldsymbol{Au}$が得られる．すなわち「短い」ベクトル$\boldsymbol{u}$に対して$-\boldsymbol{Au}$が「短い」ベクトル$\boldsymbol{v}$となる．項を移項して$\boldsymbol{Iv} + \boldsymbol{Au} = \boldsymbol{0}$となり，これは$[\boldsymbol{I} \mid \boldsymbol{A}]$に対する MSIS 問題となる．より正確には SelfTargetMSIS 問題である．SelfTarget-MSIS 問題は，ハッシュ関数を含む署名方式特有の安全性を表す問題であり，この解は標準的な MSIS 問題の解を与える．攻撃者が同一メッセージに対して 2 つの有効な署名を生成できれば，その時点で SelfTargetMSIS 問題（ひいては MSIS 問題）を解けたことになる．したがって，MSIS 問題が計算困難である限り，ML-DSA に対する署名偽造も困難である．

5 むすび

本稿では，標準化動向を概観したうえで，2024 年に標準化されたデジタル署名方式である ML-DSA の仕組みを可能な限り平易にまとめた．暗号アルゴリズムの完全移行には一般に 5 年から 15 年を要するとされており，量子コンピュータの実用化の到来時期が不確実であることを勘案しても，PQC への計画的移行は早急に着手すべき課題である．移行自体は暗号理論を深く理解していなくても進められるが，基本原理を把握しておくことは製品選定や障害対応に役立つ．本稿が FIPS 204 をはじめとする NIST ドキュメントに触れる際の導入資料となれば幸いである．

最後に，社会インフラとしての信頼性が不可欠な金融領域において，PQC への移行は極めて重要な経営課題である．当社では，PQC 技術の導入を積極的に推進し，社内システムのセキュリティ強化とお客さまの安全性向上に日々取り組んでいる．そこで培った知見やノウハウを広く共有することで，金融業界全体の円滑な移行を支援し，ひいては社会全体の情報セキュリティ水準向上に寄与する所存である．

引 用 文 献

1) National Institute of Standards and Technology, Post-Quantum Cryptography, https://csrc.nist.gov/projects/post-quantum-cryptography, (参照 2025-06-30)
2) National Institute of Standards and Technology, 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, (参照 2025-06-30)
3) National Institute of Standards and Technology, NIST Selects HQC as Fifth Algorithm for Post-Quantum Encryption, https://www.nist.gov/news-events/news/2025/03/nist-selects-hqc-fifth-algorithm-post-quantum-encryption, (参照 2025-06-30)
4) National Institute of Standards and Technology, PQC Standardization Process: Announcing Four Candidates to be Standardized, Plus Fourth Round Candidates, https://www.nist.gov/news-events/news/2022/07/pqc-standardization-process-announcing-four-candidates-be-standardized-plus, (参照 2025-06-30)
5) Alagic, G., et al.: Status Report on the Fourth Round of the NIST Post-Quantum Cryptography Standardization Process, *National Institute of Standards and Technology: Gaithersburg, MD, USA* (2025).
6) National Institute of Standards and Technology, Post-Quantum Cryptography: Additional Digital Signature Schemes, https://csrc.nist.gov/projects/pqc-dig-sig, (参照 2025-06-30)
7) National Institute of Standards and Technology (2024) Module-Lattice-Based Digital Signature Standard. (Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) NIST FIPS 204.https://doi.org/10.6028/NIST.FIPS.204
8) Ducas, L., et al. : CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2018): 238-268.