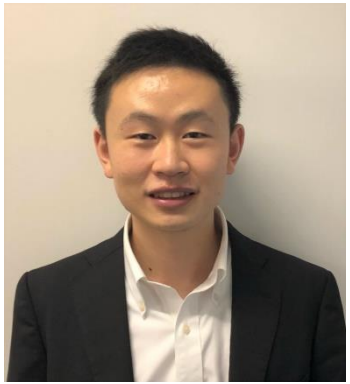
信頼される安心を、社会へ。
SECOM

耐量子計算機暗号のHSMへの実装方法とその Cryptographic boundary構成方法についての考察



セコム株式会社　IS研究所
肖　俊廷　　伊藤　忠彦

NISTのPQCプロジェクト

2016.2
NISTが耐量子暗号（PQC）の標準化計画を発表した

2017.11
Round 1が始まった（候補者：69件）

2019.1
Round 2が始まった（候補者：26件）

2020.5
実用化コメントの募集[1]が始まった

2020.7
Round 3が始まった（候補者：7＋8件）

2022/2024
標準のドラフト版

[1] https://csrc.nist.gov/publications/detail/white-paper/2020/05/26/getting-ready-for-post-quantum-cryptography/draft

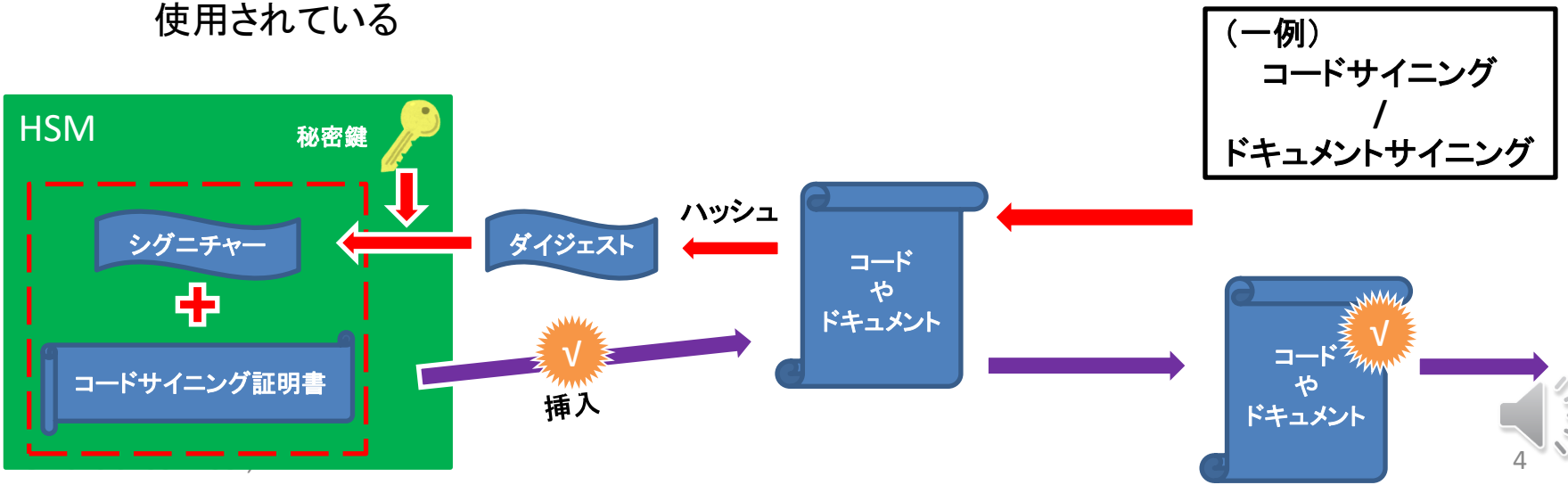
Hardware security module

- Hardware security module (HSM)
  - 物理的に保護されたセキュリティモジュール
  - 多くは特定の認定基準を満たす（e.g. FIPS 140-2やCommon Criteria）
  - 暗号、復号化、認証などの暗号化操作を実行する信頼できる環境
  - 公開鍵基盤（PKI）やインターネットバンキングのインフラの一部などの重要なインフラで使用されている

ProtectServer External 2 HSM
https://cpl.thalesgroup.com/sites/default/files/content/product_briefs/field_document/2020-09/Thales_ProtectServer_Network_HSM2_PB.pdf

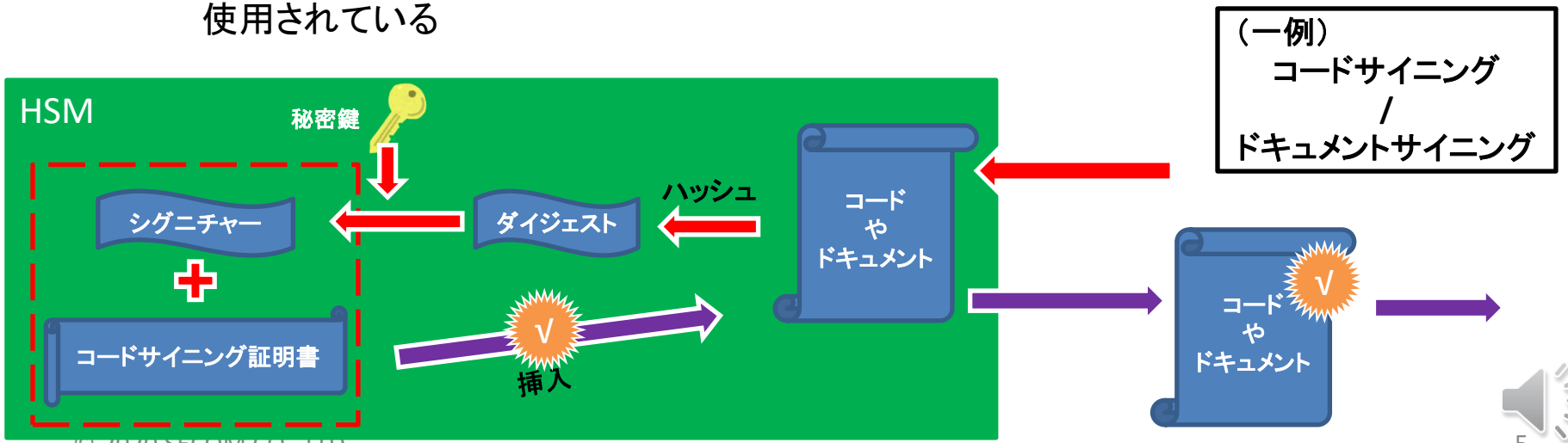
Hardware security module

- Hardware security module (HSM)
  - 物理的に保護されたセキュリティモジュール
  - 特定の認定基準を満たす（e.g. FIPS 140-2やCommon Criteria）
  - 暗号、復号化、認証などの暗号化操作を実行する信頼できる環境
  - 公開鍵基盤（PKI）やインターネットバンキングのインフラの一部などの重要なインフラで使用されている



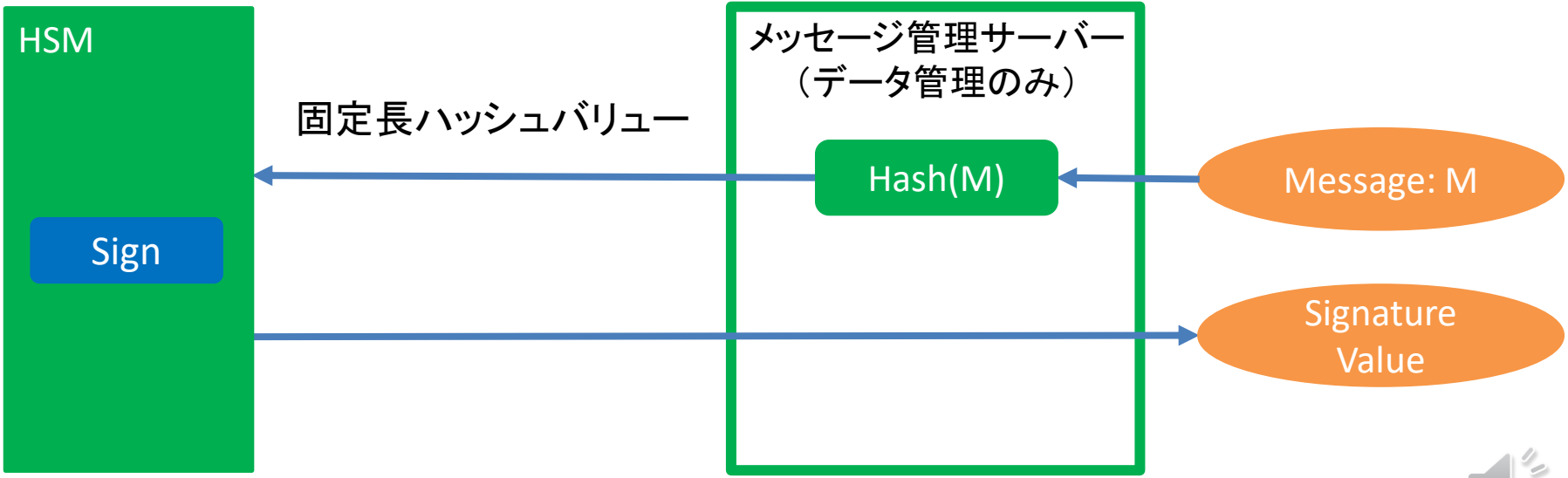
Hardware security module

- Hardware security module (HSM)
  - 物理的に保護されたセキュリティモジュール
  - 特定の認定基準を満たす（e.g. FIPS 140-2やCommon Criteria）
  - 暗号、復号化、認証などの暗号化操作を実行する信頼できる環境
  - 公開鍵基盤（PKI）やインターネットバンキングのインフラの一部などの重要なインフラで使用されている



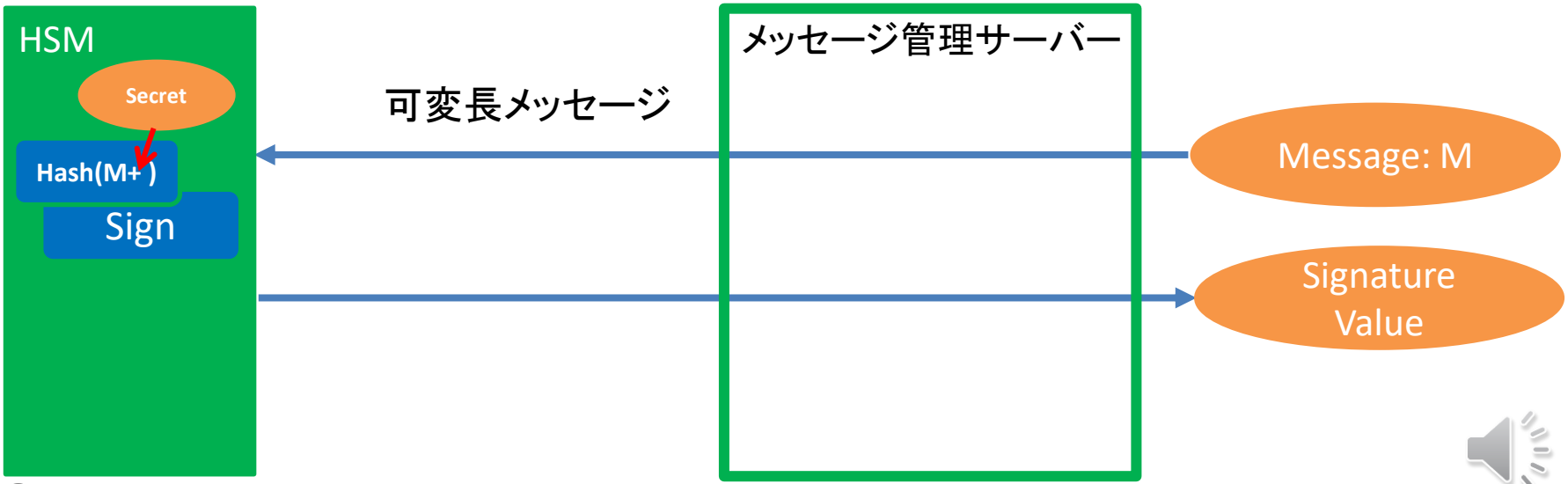
HSMとハッシュ関数

- 一般的に、平文のハッシュをメッセージ管理サーバーで計算し、その後、HSMへ送る。署名生成はHSM内で行う。



HSMとハッシュ関数

- 平文と秘密情報を合わせてハッシュの入力としなければいけない暗号も存在する。その場合、署名操作のみでなく、秘密情報の生成もHSMで行うことが望ましい。



本研究の成果

- 大きなデータについての署名の検討

現状のNIST等の他の検討は、小さなデータサイズだけを想定しているように思える

大きなデータも小さなデータを、違う仕組みで処理しないといけない（相互運用性やAPI統一の問題）

- 大きなデータも小さなデータも、同じ仕組みで処理するためには、どんな仕組みが必要かという検討も行った

研究のアプローチ

- 耐量子暗号デジタル署名アルゴリズムはどのようにハッシュ計算をしているかを確認すること

効率性の最適化が考えられる

- 耐量子暗号デジタル署名アルゴリズムの各部分がどの物理境界内で行われるかを識別すること

安全性と効率性を考慮したセキュリティメカニズムの設計が考えられる

耐量子暗号に適した物理境界のアーキテクチャに移行するコストが考えられる

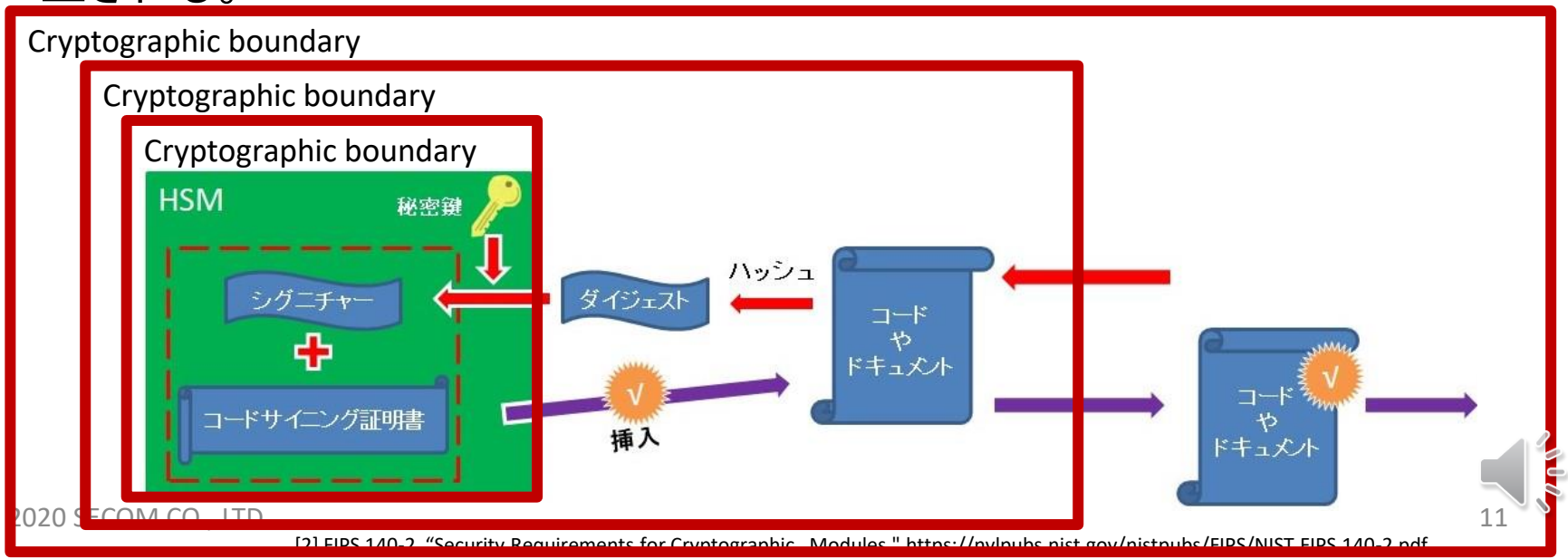
Cryptographic Boundary

- サイバーシステム内で使用される暗号化モジュールに対して、cryptographic boundary[2]は、当該暗号モジュールを構成するすべてのソフトウェア、ハードウェア、およびファームウェアを含む物理境界により確立される。

[2] FIPS 140-2. “Security Requirements for Cryptographic Modules." https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-2.pdf.

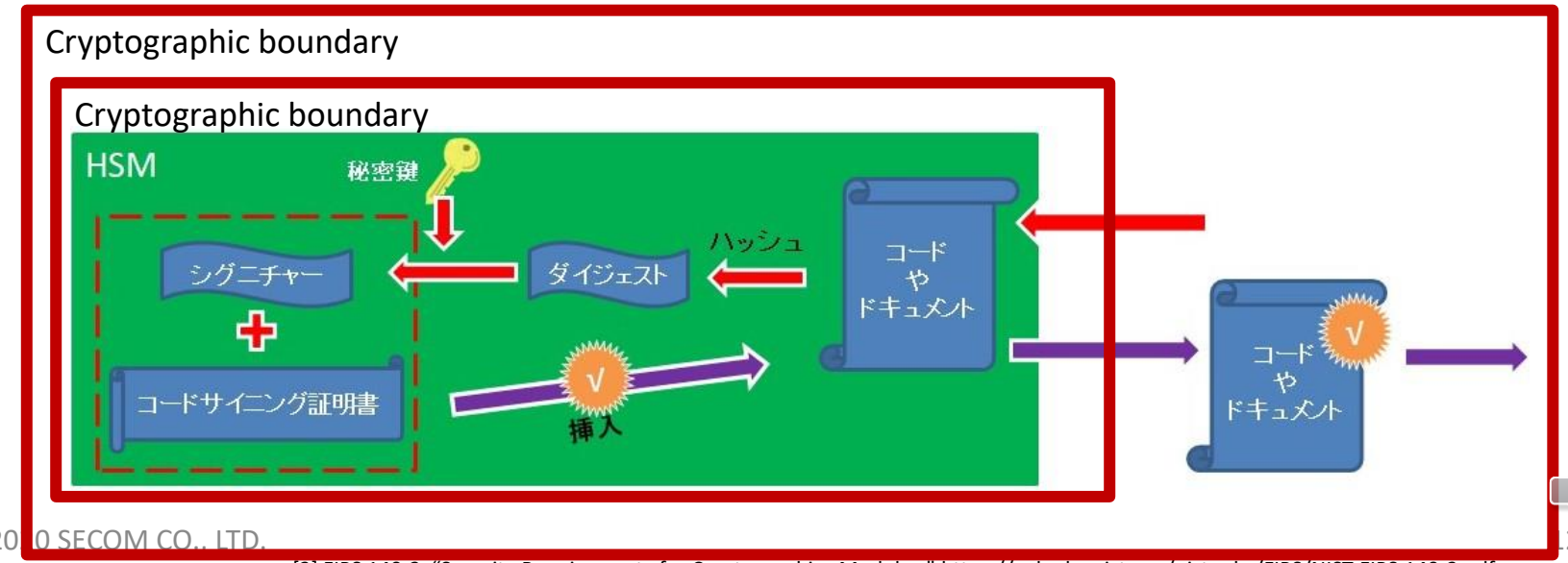
Cryptographic Boundary

- サイバーシステム内で使用される暗号化モジュールに対して、cryptographic boundary[2]は、当該暗号モジュールを構成するすべてのソフトウェア、ハードウェア、およびファームウェアを含む物理境界により確立される。



Cryptographic Boundary

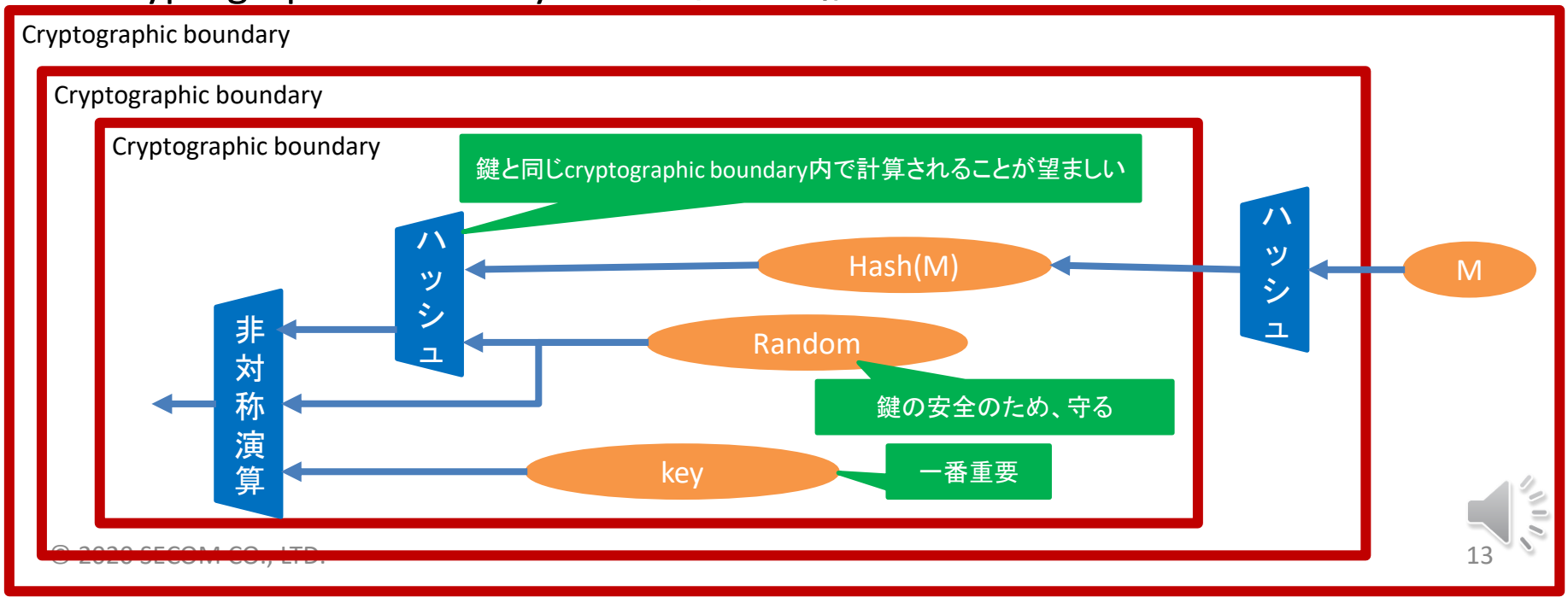
- サイバーシステム内で使用される暗号化モジュールに対して、cryptographic boundary[2]は、当該暗号モジュールを構成するすべてのソフトウェア、ハードウェア、およびファームウェアを含む物理境界により確立される。



[2] FIPS 140-2. “Security Requirements for Cryptographic Modules." https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-2.pdf.

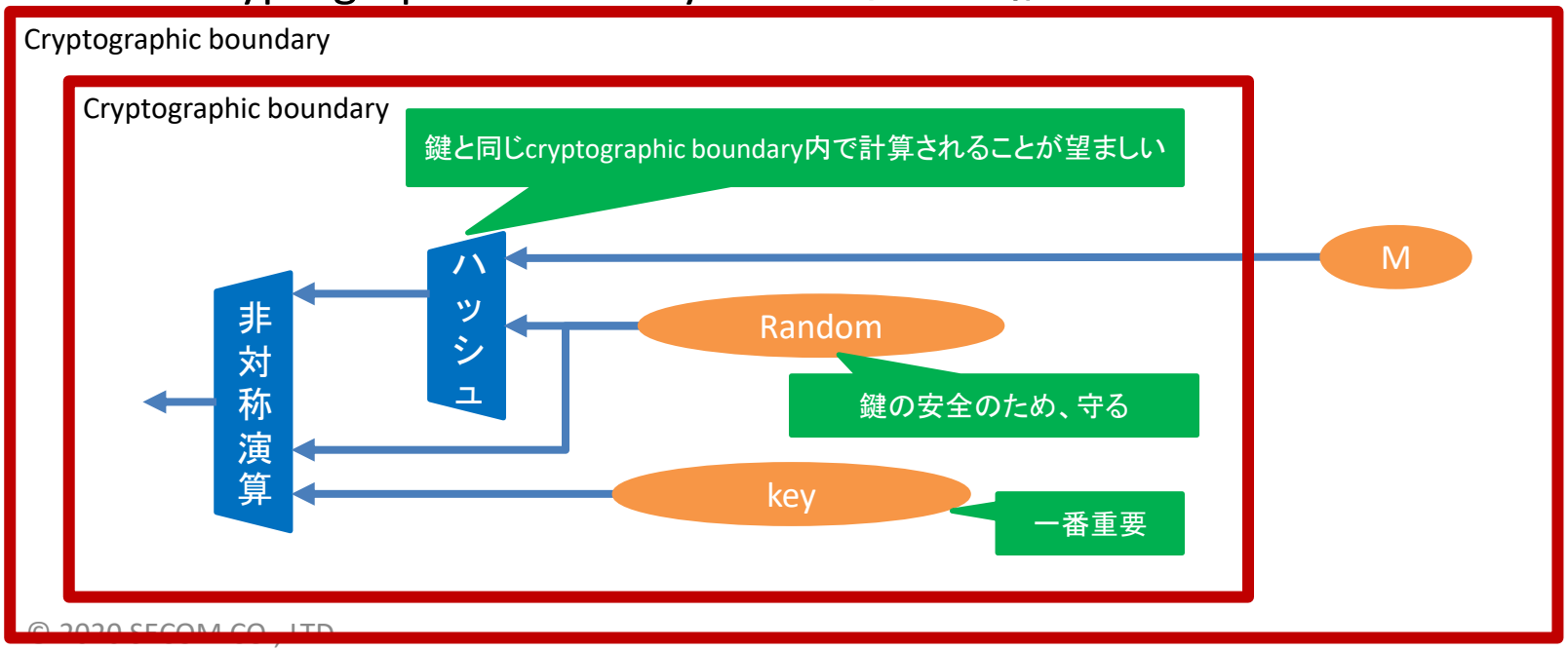
署名スキームに対する Cryptographic Boundaryの設計

- 秘密情報の生成、及び署名生成が、平文に対するハッシュと異なるcryptographic boundary内で行われる構成



署名スキームに対する Cryptographic Boundaryの設計

- 秘密情報の生成、秘密情報と平文のハッシュ計算、及び署名生成が単一のcryptographic boundary内で行われる構成



格子ベースの署名アルゴリズム

- 格子暗号（lattice-based cryptography）
  - 格子問題の困難性に基づき、耐量子計算機の一種である
  - 多くはSHA-3ファミリーのハッシュ関数を利用する
- Round 2候補者としての格子ベース署名アルゴリズムを選んだ

<table>
<tr><td colspan="3">Round 2候補者</td></tr>
<tr><td colspan="2">Round 3候補者（Finalistsとして）</td><td rowspan="2">qTESLA[5]</td></tr>
<tr><td>FALCON[3]</td><td>CRYSTAL-DILITHIUM[4]</td></tr>
</table>

[3] P. A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang. "Falcon: Fast-fourier lattice-based compact signatures over ntru." submission to the nist's post-quantum cryptography standardization process.(2018), 2018.
[4] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehle. "CRYSTALS-Dilithium: A lattice-based digital signature scheme." *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238-268, 2018. https://tches.iacr.org/index.php/TCHES/article/view/839.
[5] N. Bindel, S. Akeylek, E. Alkim, P. S. Barreto, J. Buchmann, E. Eaton, G. Gutoski, J. Kramer, P. Longa, H. Polat, J. E. Richardini, and G. Zanon. qtesla. submission to the nist's post-quantum cryptography standardization process.(2018), 2018.

FALCONに対する Cryptographic Boundary の構成

- アルゴリズムによるCryptographic Boundaryへの制約

Algorithm 1: Signature Generation of FALCON

Input : Private key **sk**; Message **m**; A bound β.

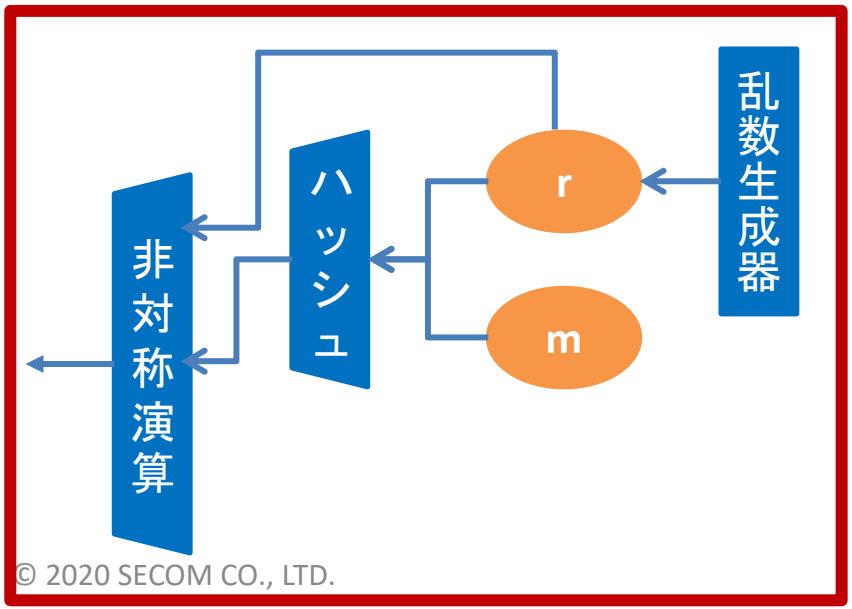
Output: The signature *sig* = (**r, s**).

1. $\mathbf{r} \in \{0,1\}^{320}$ uniformly
2. $\mathbf{c} = \mathrm{HashToPoint}(\mathbf{r}||\mathbf{m})$
3. $\mathbf{t} = (\mathrm{FFT}(\mathbf{c}), \mathrm{FFT}(0)) \cdot \widehat{\mathbf{B}}^{-1}$
4. **do**
5. $\quad \mathbf{z} = \mathrm{ffSampling}_n(\mathbf{t}, \mathbf{T})$
6. $\quad \mathbf{s} = (\mathbf{t} - \mathbf{z})\widehat{\mathbf{B}}$
7. **while** $\left|\left|\mathbf{s}\right|\right| > \beta$
8. $(\mathbf{s}_1, \mathbf{s}_2) = invFFT(\mathbf{s})$
9. $\mathbf{s} = \mathrm{Compress}(\mathbf{s}_2)$
10. **return** *sig* = (**r, s**)

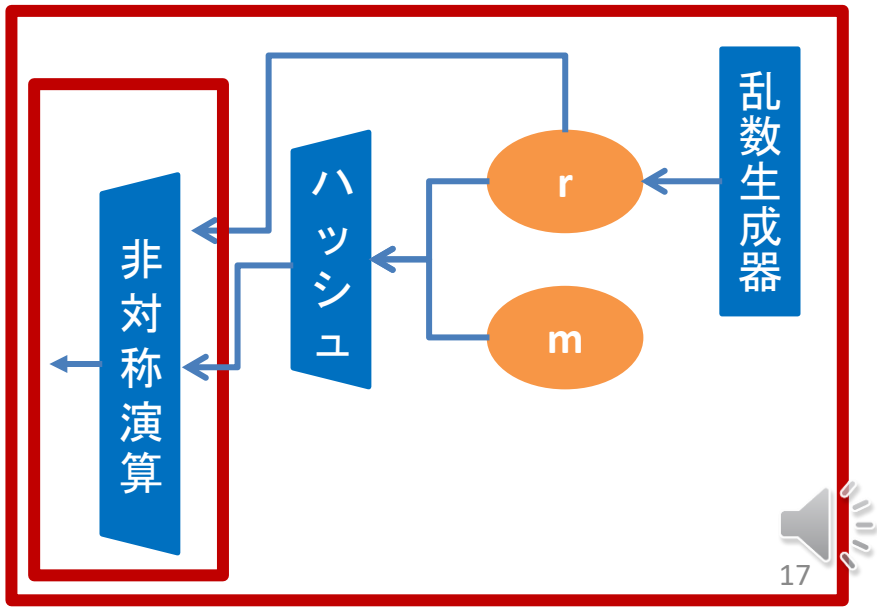
FALCONに対する Cryptographic Boundary の構成

- Cryptographic Boundaryの構成例

例1：ハッシュと非対称演算が同一の cryptographic boundary内で行う



例2：ハッシュと非対称演算が異なるの cryptographic boundary内で行う



CRYSTALS-DILITHIUMに対するCryptographic Boundary の構成

- アルゴリズムによるCryptographic Boundaryへの制約

Algorithm 2: Signature Generation of CRYSTALS-DILITHIUM

Input : $\mathbf{sk} = (\boldsymbol{\rho}, \mathbf{K}, \mathbf{tr}, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$; Message **m**.

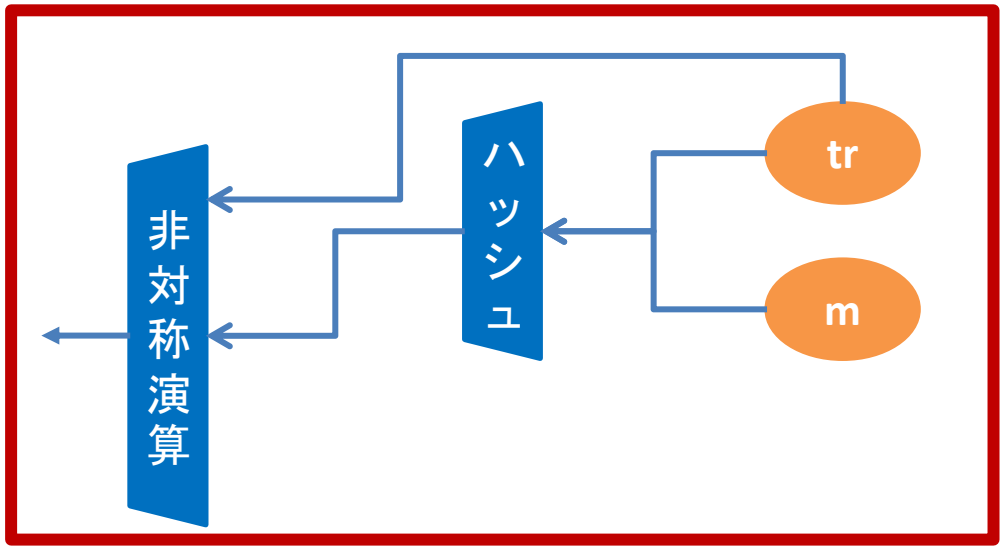
Output: The signature $\boldsymbol{\sigma} = (\mathbf{z}, \mathbf{h}, c)$.

1. $\mathbf{A} \in R_q^{kxl} = \mathrm{ExpandA}(\boldsymbol{\rho})$
2. $\boldsymbol{\mu} \in \{0,1\}^{384} = \mathrm{CRH}(\mathbf{tr}||\mathbf{m})$
3. $\mathrm{k} = 0, (\mathbf{z}, \mathbf{h}) = \perp$
4. $\boldsymbol{\rho}' \in \{0,1\}^{384} = CRH(\mathbf{K}||\boldsymbol{\mu})$
5. **while** $(\mathbf{z}, \mathbf{h}) = \perp$ **do**
6. $\quad \boldsymbol{y} \in S_{\gamma_1 - 1}^{l} = ExpandMask(\boldsymbol{\rho}', k)$
7. $\quad \boldsymbol{w} = \mathbf{A}\boldsymbol{y}$
8. $\quad \boldsymbol{w_1} = \mathrm{HighBits_q}(\boldsymbol{w}, 2\gamma_2)$
9. $\quad \boldsymbol{c} \in B_{60} = \mathrm{H}(\boldsymbol{\mu}, \boldsymbol{w_1})$
10. $\quad \boldsymbol{z} = \boldsymbol{y} + \boldsymbol{cs_1}$
11. $\quad (\boldsymbol{r_1}, \boldsymbol{r_0}) = \mathrm{Decompose_q}(\boldsymbol{w} - \boldsymbol{cs_2}, 2\gamma_2)$
12. $\quad$ **if** $||z||_\infty \geq \gamma_1 - \beta$ **or** $||\boldsymbol{r_0}||_\infty \geq \gamma_2 - \beta$ **or** $\boldsymbol{r_1} \neq \boldsymbol{w_1}$ **then**
13. $\quad\quad (\mathbf{z}, \mathbf{h}) = \perp$
14. $\quad$ **end**
15. $\quad$ **else**
16. $\quad\quad \boldsymbol{h} = \mathrm{MakeHint_q}(-\boldsymbol{ct_0}, \boldsymbol{w} - \boldsymbol{cs_2} + \boldsymbol{ct_0}, 2\gamma_2)$
17. $\quad\quad$ **if** $||\boldsymbol{ct_0}||_\infty \geq \gamma_2$ **or** the # of 1's in $\boldsymbol{h}$ is greater than ω **then**
18. $\quad\quad\quad (\mathbf{z}, \mathbf{h}) = \perp$
19. $\quad\quad$ **end**
20. $\quad$ **end**
21. $\quad k = k + 1$
22. **end**
23. **return** $\boldsymbol{\sigma} = (\mathbf{z}, \mathbf{h}, c)$

CRYSTALS-DILITHIUMに対する Cryptographic Boundary の構成

- Cryptographic Boundaryの構成例

例1：ハッシュと非対称演算が同一の
cryptographic boundary内で行う



qTESLAに対する Cryptographic Boundary の構成

- アルゴリズムによるCryptographic Boundaryへの制約

Algorithm 3: Signature Generation of qTESLA

Input : $\mathbf{sk} = (\boldsymbol{s}, \boldsymbol{e}_1, \dots, \boldsymbol{e}_k, \mathbf{seed}_a, \mathbf{seed}_y, \mathbf{g})$; Message $\mathbf{m}$.

Output: The signature $sig = (\mathbf{z}, \mathbf{c}')$.

1. counter $= 1$
2. $\boldsymbol{r} \in \{0,1\}^k$
3. $\mathbf{rand} = \mathrm{PRF}_2(\mathbf{seed}_y, \boldsymbol{r}, \mathbf{m})$
4. $\boldsymbol{y} = \mathrm{ySampler}(\mathbf{rand}, \mathrm{counter})$
5. $\boldsymbol{a}_1, \dots, \boldsymbol{a}_k \leftarrow \mathrm{GenA}(\mathbf{seed}_a)$
6. **for** $i = 1, \dots, k$ **do**
7. $\quad \boldsymbol{v}_i = \boldsymbol{a}_i \boldsymbol{y} \bmod^{\pm} q$
8. **end**
9. $\boldsymbol{c}' = \mathrm{H}(\boldsymbol{v}_1, \dots, \boldsymbol{v}_k, G(\boldsymbol{m}), \mathbf{g})$
10. $\boldsymbol{c} = \{pos_{list}, sign_{list}\} \leftarrow \mathrm{End}(\boldsymbol{c}')$

From Ver. 2.8 (11/08/2019)

2. $\boldsymbol{r} \in \{0,1\}^k$
3. $\mathbf{rand} = \mathrm{PRF}_2(\mathbf{seed}_y, \boldsymbol{r}, G(\mathbf{m}))$

** $G(\,): \{0,1\}^* \rightarrow \{0,1\}^{512}$

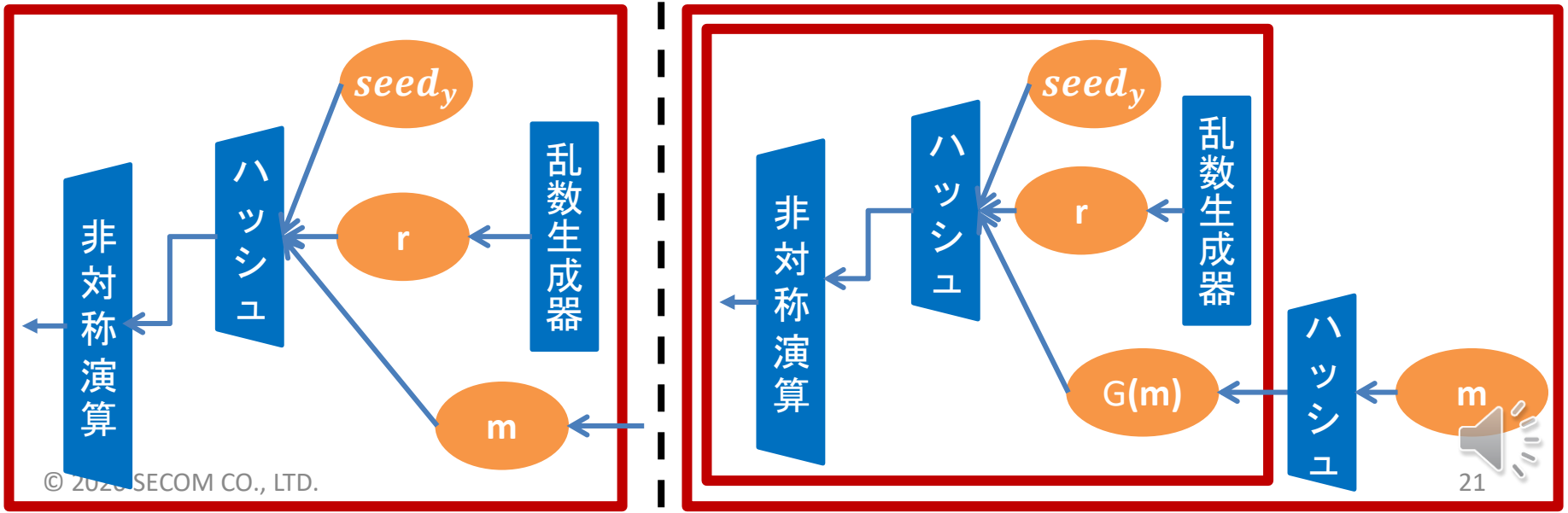
11. $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{sc}$
12. **if** $\boldsymbol{z} \notin R_{[B-S]}$ **then**
13. $\quad$ counter $=$ counter $+ 1$
14. $\quad$ Restart as step 4
15. **end**
16. **for** $i = 1, \dots, k$ **do**
17. $\quad \boldsymbol{w}_i = \boldsymbol{v}_i - \boldsymbol{e}_i \boldsymbol{c} \bmod^{\pm} q$
18. $\quad$ **if** $||[\boldsymbol{w}_i]_L||_\infty \geq 2^{d-1} - E \vee ||\boldsymbol{w}_i||_\infty \geq \lfloor q/2 \rfloor - E$ **then**
19. $\quad\quad$ counter $=$ counter $+ 1$
20. $\quad\quad$ Restart as step 4
21. $\quad$ **end**
22. **end**
23. **return** $sig = (\mathbf{z}, \mathbf{c}')$

qTESLAに対する Cryptographic Boundary の構成

- Cryptographic Boundaryの構成例

例1：ハッシュと非対称演算が同一の cryptographic boundary内で行う

例2：ハッシュと非対称演算が異なるの cryptographic boundary内で行う



実験

- 実験目的
  - HSM内部での格子ベース署名に関する操作の効率性を分析する
- HSMの仕様
  - タレス社製HSMであるProtectServer External 2 (PSE-2)を利用

- 実験内容
  - 格子ベース署名アルゴリズムのFALCON, CRYSTAL-DILITHIUM, qTESLAを選択
  - 異なるCryptographic Boundary構成で、HSM内で実行したハッシュ計算を計測する
  - 二種類のCryptographic boundary構造
    - タイプA：ハッシュと非対称演算が同一のcryptographic boundary内で行う
    - タイプB：ハッシュと非対称演算が異なるのcryptographic boundary内で行う
  - 実行時間はミリ秒（ms）で記載、メッセージのサイズは、キロバイト（K）またはメガバイト（M)で記載

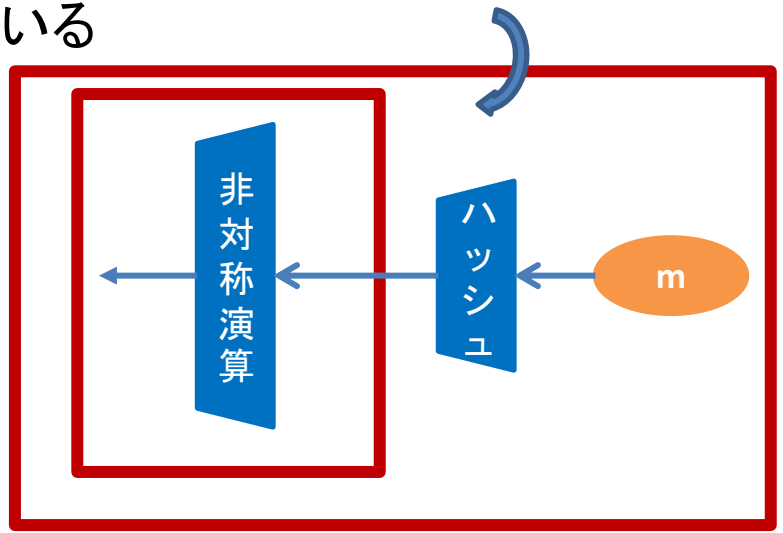
実験結果

スキーム	Boundaryタイプ	実行時間 (ms)				
		1K	10K	100k	1M	10M
FALCON	A	33.36	38.08	142.26	1240.59	11667.19
	B	0				
DILITHIUM	A	34.67	45.79	156.19	1351.40	12727.83
qTESLA	A (before Ver. 2.8)	34.78	44.78	138.05	1196.26	11810.52
	B (from Ver. 2.8)	30.84	38.25	38.63	38.63	31.42

1. FALCON： タイプBを採用する場合、HSM内でハッシュ計算がなく、リソースの負担なし
2. DILITHIUM： HSM内の処理について、ハッシュ計算はボトルネック
3. qTESLA： Ver. 2.8からタイプBが使え、HSM内でハッシュ計算についてのリソース負担が軽い

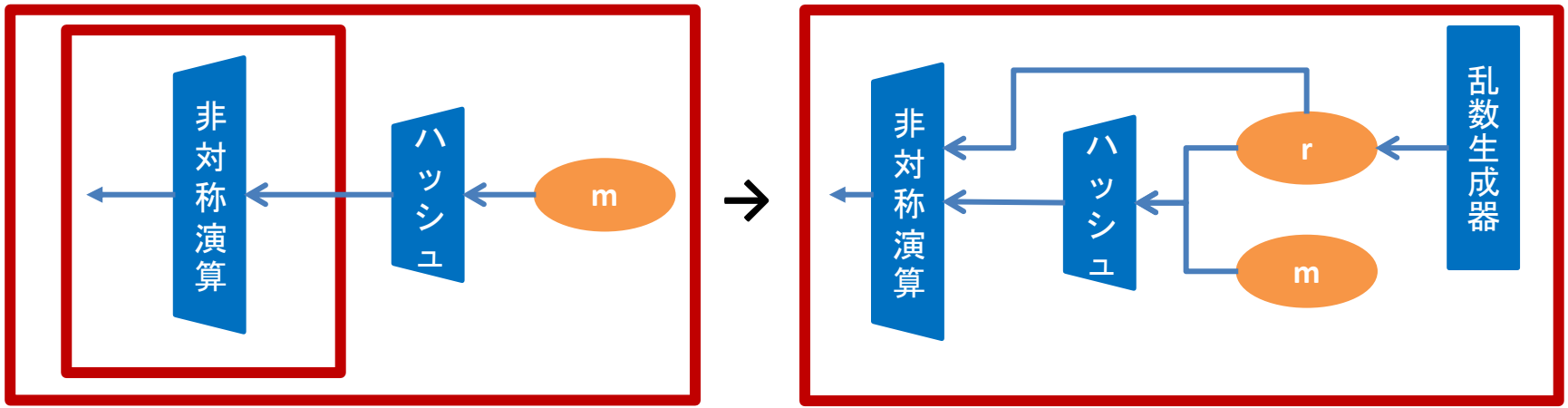
各Cryptographic Boundaryを採用した場合の移行コスト

- 現状にはHSMを利用したRSAやECDSAの署名システムは、タイプBのcryptographic boundary構造を採用している
- 格子ベース署名に移行する場合
  - ① タイプB（RSA or ECDSA）→ タイプA（lattice）
  - ② タイプB（RSA or ECDSA）→ タイプB（lattice）
- 移行コストを考えるのが必要



各Cryptographic Boundaryを採用した場合の移行コスト

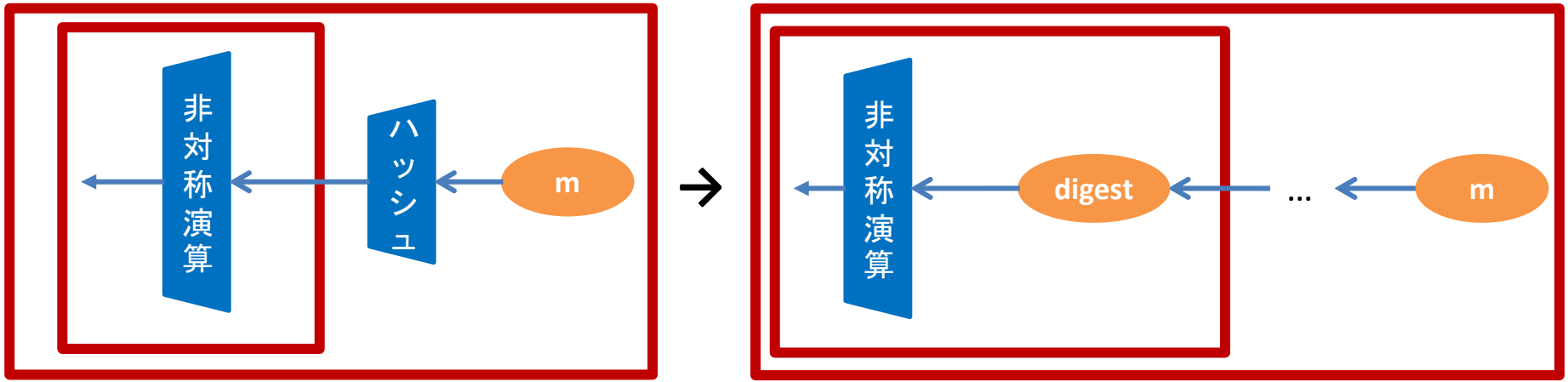
① タイプB（RSA or ECDSA） → タイプA（Lattice-based）



1. もっと複雑なアクセスコントロールメカニズム
2. コンピューティングリソースの需要増加

各Cryptographic Boundaryを採用した場合の移行コスト

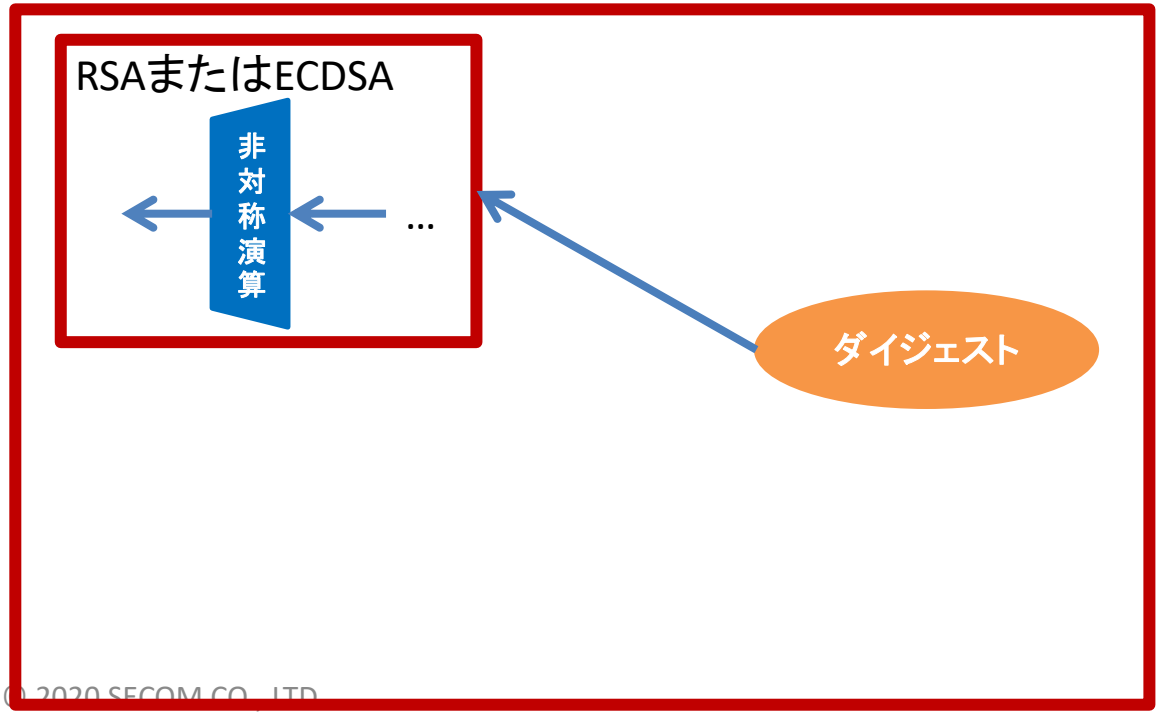
② タイプB（RSA or ECDSA） → タイプB（Lattice-based）



1. アーキテクチャの差異が小さい
2. 構成の変更が限定的

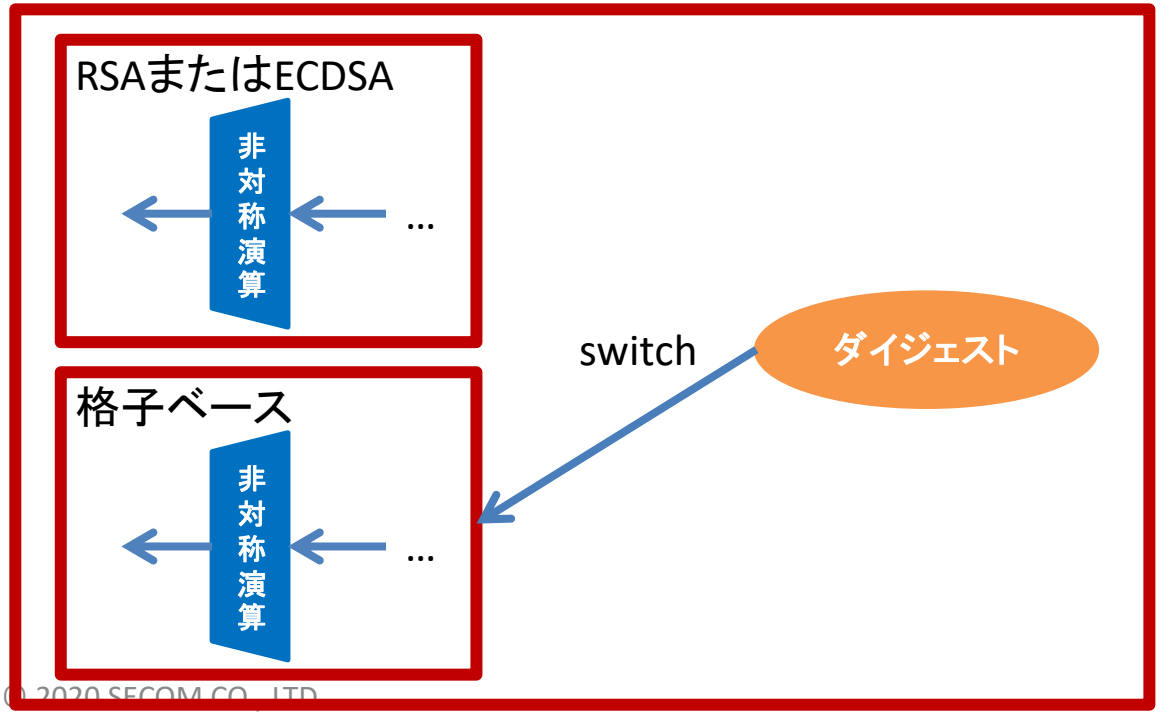
各Cryptographic Boundaryを採用した場合の移行コスト

② タイプB（RSA or ECDSA） → タイプB（Lattice-based）



各Cryptographic Boundaryを採用した場合の移行コスト

② タイプB（RSA or ECDSA） → タイプB（Lattice-based）



結論

- 三つの格子ベース署名アルゴリズム（FALCON, DILITHIUM, qTESLA）のハッシュ処理方式を確認した
- 確認した結果によって、HSM上で使うときのそれぞれの署名アルゴリズムに適したcryptographic boundary構成を検討した
- 検討したcryptographic boundary構成に基づき、HSM内で実行したハッシュ計算の計測により、性能を評価した
- クラシックなデジタル署名アルゴリズムから格子ベース署名アルゴリズムへの移行コストを分析した