多項式NTRU問題に基づく準同型暗号方式の拡張

稲邉 二千翔[1,a)]　若杉 飛鳥[1]

概要：完全準同型暗号方式とは，暗号文を復号することなく，平文上の任意の回路を暗号文上で評価できる暗号方式である．多くの完全準同型暗号方式は，LWE (Learning With Errors) 問題や RLWE (Ring LWE) 問題に基づいて構成されている．一方，NTRU 問題を安全性の根拠とする完全準同型暗号方式も知られており，Lin らは，多項式 NTRU 問題を用いた方式を近年提案した．本方式によって，一度の暗号文演算で複数の平文を処理可能な SIMD（Single Instruction, Multiple Data）演算を実現できる．しかし，本方式は，共通鍵による方式であり，更に平文が 1 ビットに限られている．本稿では，Lin らによる方式を，公開鍵を用いた整数平文の完全準同型暗号方式へ拡張し，提案方式を構成するベース暗号方式が IND-CPA 安全性を満たすことを示す．更に，提案方式が 128-bit security を満たし，主要なアルゴリズムの復号失敗確率が 2^{-128} 未満となるように演算鍵サイズを最適化するパラメータセットを提示する．最後に，提案方式の実装検証を行い，Bootstrapping による SIMD 演算が正しく動作することを確認する．

キーワード：完全準同型暗号, NTRU 問題

Extension of Homomorphic Encryption Schemes Based on Polynomial NTRU Problems

Nichika Inabe[1,a)]　Asuka Wakasugi[1]

Abstract: Fully homomorphic encryption (FHE) is an encryption scheme that can evaluate any circuit on plaintexts over ciphertexts without decryption. Most FHE schemes are constructed based on the LWE (Learning With Errors) or RLWE (Ring LWE) problems. Meanwhile, FHE schemes whose security relies on the NTRU problem are also known, and Lin et al. recently proposed a scheme based on the polynomial NTRU problem. Their scheme realizes SIMD (Single Instruction, Multiple Data) operations that process multiple plaintexts in a single homomorphic operation. However, their scheme is secret-key and its plaintext is limited to a single bit. In this paper, we extend Lin et al.'s scheme to a public-key FHE scheme for integer plaintexts, and show that the base encryption scheme underlying the proposed scheme satisfies IND-CPA security. Furthermore, we present a parameter set that optimizes the evaluation-key size so that the proposed scheme satisfies 128-bit security and the decryption-failure probability of its main algorithms is below 2^{-128}. Finally, we implement the proposed scheme and confirm that its SIMD operations via bootstrapping work correctly.

Keywords: Fully Homomorphic Encryption，NTRU Problem

1. 背景

準同型暗号は，暗号文を復号することなく，平文上の演算に対応する演算を暗号文上で実行できる暗号方式である．中でも完全準同型暗号（FHE）は，任意の回路を暗号文上で評価できる準同型暗号であり，Gentry [16] により初めてその構成が示された．その後，BFV/BGV 方式 [8], [15]，FHEW/TFHE 方式 [10], [12]，CKKS 方式 [9] など，様々な方式が知られている．これらの方式は，いずれも LWE（Learning With Errors）問題 [23] や RLWE（Ring LWE）問題 [22] を安全性の根拠としている．BFV/BGV/CKKS

[1] EAGLYS 株式会社
EAGLYS Inc.
[a)] nichika.inabe@eaglys.co.jp

方式では，複数の平文値を一つの暗号文の複数のスロットに格納し，各スロットに同一の演算を並列に適用する SIMD（Single Instruction, Multiple Data）演算が可能である．これらの方式は，OpenFHE [1] や Lattigo [14] などのライブラリに実装されている．一方，FHEW/TFHE 方式では，単一の平文値が暗号文に格納されており，TFHE-rs [24] などのライブラリに実装されている．

また，NTRU 問題 [17] を安全性の根拠とする FHE 方式も知られており，LTV 方式 [21] や YASHE 方式 [5]，FINAL 方式 [4]，Lin らの方式 [20] などが該当する．

1.1 FHE の基本的な概念

暗号方式は，一般に，鍵生成アルゴリズム KeyGen，暗号化アルゴリズム Enc，および復号アルゴリズム Dec によって記述される．暗号化の対象となる平文としては，方式に応じて，1 ビット値や整数値などが用いられる．FHE 方式は，これらに加えて，暗号文を復号することなく，任意の回路を暗号文上で評価するアルゴリズム群を備える．本稿では，暗号文加算 Add や暗号文乗算 Mul などを総称して Eval と表記する．また，暗号方式は，鍵の構成に基づき，暗号化と復号に共通の秘密鍵を用いる共通鍵暗号方式と，暗号化に公開鍵を用い，復号に対応する秘密鍵を用いる公開鍵暗号方式とに大別される．

LWE 問題，RLWE 問題，NTRU 問題を安全性の根拠とする FHE 方式は，暗号文内にノイズが含まれており，このノイズの大きさは，暗号文上での演算を重ねるたびに増大する．平文 m の暗号文 $\mathrm{Enc}(m)$ に対して，そのノイズの大きさが閾値である正整数 τ 以上の場合，つまり，$|\mathrm{Dec}(\mathrm{Enc}(m)) - m| \geq \tau$ が成り立つ場合を復号失敗と呼ぶ．Bootstrapping は，復号処理を暗号文上で準同型的に評価する処理であり，その処理では演算鍵と呼ばれる補助情報が用いられる．

1.2 先行研究

Bootstrapping は暗号文を再び演算可能な状態にする処理であるが，Bootstrapping と同時に，暗号文に対応する平文に任意の関数を適用する手法も提案されている．代表例として，CKKS 方式における FBS（Functional Bootstrapping） [3] や，TFHE 方式における FBS [6] や PBS（Programmable Bootstrapping） [11] が挙げられる．

また，Bootstrapping に用いる演算鍵はサイズが大きく，その削減を目的とした研究も行われている．TFHE 方式の Bootstrapping に基づいて構成された FINAL 方式 [4] は，演算鍵サイズの削減を目的として提案された FHE 方式であり，TFHE 方式と比較して演算鍵のサイズを約 45% 削減している．

さらに，TFHE 方式や FINAL 方式に対する様々な拡張が提案されている．TFHE 方式については，共通鍵型から公開鍵型への拡張 [18] や，1 ビット平文から整数平文への拡張 [7] が提案されている．TFHE 方式や FINAL 方式を拡張し，SIMD 演算を実現する FHE 方式 [19], [20] も提案されている．

表 1 各方式におけるベース暗号文と演算用暗号文

方式	ベース暗号文	演算用暗号文
TFHE 方式	LWE 暗号文	RLWE 暗号文
FINAL 方式	LWE 暗号文	NTRU 暗号文
Lin らの方式	NTRU 暗号文	NTRU 暗号文

1.3 対象方式

本研究では，TFHE 方式，FINAL 方式，および Lin らにより提案された FHE 方式を対象とする．これらの方式では，KeyGen および Enc に加えて，Bootstrapping に用いる暗号文を生成するための鍵生成アルゴリズムおよび暗号化アルゴリズムがあり，これらのアルゴリズムをそれぞれ演算用暗号化鍵生成，演算用暗号化と呼ぶ．本稿では，KeyGen および Enc を，それぞれベース鍵生成およびベース暗号化と呼び，それぞれのアルゴリズムの出力をベース鍵およびベース暗号文と呼ぶ．同様に，演算用暗号化鍵生成および演算用暗号化の出力を，それぞれ演算用暗号化鍵および演算用暗号文と呼ぶ．また，ベース鍵生成，ベース暗号化，および復号からなる暗号方式をベース暗号方式と呼ぶ．ここで，演算用暗号化鍵と演算鍵は異なる対象であることに注意されたい．これらの方式では演算鍵として，BSK（Bootstrapping Key）および KSK（Key Switching Key）が用いられ，それぞれ BSK は複数の演算用暗号文，KSK は複数のベース暗号文から構成される．

本稿では，LWE 問題，RLWE 問題，および NTRU 問題に基づいて構成された暗号化方式によって生成される暗号文を，それぞれ LWE 暗号文，RLWE 暗号文，および NTRU 暗号文と呼ぶ．TFHE/FINAL/Lin らの方式にて，どのような暗号文が用いられているかを表 1 に示す．

1.4 目的

本研究の目的は，TFHE 方式と比較して演算鍵全体のサイズが小さく，SIMD 演算に対応し，暗号化に公開鍵を用い，かつ整数平文を扱う FHE 方式を構成することである．

前節で示したように，演算鍵の一つである BSK は複数の演算用暗号文から構成される．演算用暗号文として，TFHE 方式では RLWE 暗号文を，FINAL 方式では NTRU 暗号文を用いる．一般に，RLWE 暗号文は二つ以上の多項式からなる組で表現されるのに対し，NTRU 暗号文は単一の多項式で表現される．FINAL 方式は，この暗号文構造の違いを利用することにより，TFHE 方式と比較して BSK のサイズを削減し，演算鍵全体のサイズを縮小している．

FINAL 方式は，ベース暗号文として LWE 暗号文を用

いるため，一つのベース暗号文に一つの平文を格納する．一方，Lin らの方式は，ベース暗号文として NTRU 暗号文を用い，複数の平文を多項式の各係数に格納する．この構成により，Lin らの方式は，一つの暗号文に格納された複数の平文に対して SIMD 演算を実現している．

このように，Lin らの方式は，TFHE 方式と比較して演算鍵全体のサイズを削減でき，SIMD 演算にも対応する．しかし，同方式は共通鍵型で提案されており，扱える平文も 1 ビット値に限られる．そこで，本研究では，Lin らの方式を公開鍵型で整数平文を扱う FHE 方式へ拡張する．

1.5 貢献

本研究の貢献は以下の通りである．まず，Lin らの方式を拡張して，公開鍵型で整数平文を扱う FHE 方式を構成する．また，提案方式を構成するベース暗号方式が IND-CPA 安全性を満たすことを証明する．続いて，提案方式が 128-bit security を満たし，増大したノイズを持つ暗号文の復号失敗確率が 2^{-128} 未満となるよう，演算鍵のサイズに最適化したパラメータを与える．さらに，bit security・復号失敗確率・平文のビット数などを揃えた上で，公開鍵を用いた整数平文として実装されているライブラリ TFHE-rs と提案方式で演算鍵サイズの比較を行う．最後に，提案方式を実装し，Bootstrapping を用いた SIMD 演算が正しく動作することを確認する．

1.6 構成

本稿の構成は以下の通りである．第 2 章では，提案方式の基盤となる NTRU 問題や RLWE 問題などについて述べる．第 3 章では，Lin らの方式に対して，公開鍵型で整数平文を扱う FHE 方式へ拡張し，ベース暗号方式の IND-CPA 安全性を示す．第 4 章では，提案方式のパラメータセットが満たすべき制約を述べ，演算鍵サイズに最適化したパラメータを導出し，TFHE-rs との演算鍵サイズの比較，および，実装による動作確認を行う．第 5 章で結論を述べる．

2. 前提知識

2.1 表記

本稿では，実数，整数，自然数，およびビットの集合を，それぞれ $\mathbb{R}, \mathbb{Z}, \mathbb{N}$，および $\mathbb{B} := \{0,1\}$ と表す．2 冪である正整数 n と整数 $q \geq 2$ に対し，$\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$ とする．また，$\mathbb{Z}_q$ の各元を，区間 $[-q/2, q/2) \cap \mathbb{Z}$ に属する唯一の整数代表元と同一視する．さらに，剰余環 $R_q := \mathbb{Z}_q[X]/(X^n+1)$ を定める．分布 χ に従って x を標本抽出することを $x \leftarrow \chi$ と表す．$\mathbb{Z}_q$ 上の分布 χ_q に従って多項式 $a \in R_q$ の各係数を独立に標本抽出することを $a \leftarrow \chi_q^n$ と表記する．さらに，χ_σ を $\mathbb{Z}_3$ 上の分布，$D_{\sigma'}$ を $\mathbb{Z}$ 上の平均 0 の離散ガウス分布とし，σ および σ' はそれぞれの分布の標準偏差を表す．実数 x に対し，最近接整数への丸めを $\lfloor x \rceil$，床関数を $\lfloor x \rfloor$ と表す．ただし，x の少数部分がちょうど 1/2 と一致する場合には，x に最も近い二つの整数の一方をそれぞれ確率 1/2 で選択する．したがって，この場合には $\mathbb{E}\big[\lfloor x \rceil - x\big] = 0$ が成り立つ．多項式 a の各係数の絶対値の最大値である無限ノルムを $\|a\|_\infty$ と表す．

2.2 決定版 NTRU 問題と決定版 RLWE 問題

二つの分布族に対する識別器および計算量的識別不可能性の定義については付録 A.1 を参照されたい．

$\lambda \in \mathbb{N}$ に対して，パラメータ族 $\{P_\lambda = (n_\lambda, q_\lambda, \sigma_\lambda)\}_\lambda$ とおく．また，$R_\lambda = R_{q_\lambda}, \chi_\lambda = \chi_{\sigma_\lambda}^{n_\lambda}$ とする．$\lambda \in \mathbb{N}$ に対して，分布 NTRU_λ を次のように定める．R_λ 上で可逆になるように $f \leftarrow \chi_\lambda$ を標本抽出し，また，独立に $g \leftarrow \chi_\lambda$ を標本抽出する．$h = g \cdot f^{-1} \pmod{R_\lambda}$ とするとき，NTRU_λ は，このように生成された h の分布である．また，$U_\lambda^{\mathrm{NTRU}}$ を R_λ 上の一様分布とする．決定版 NTRU 問題とは，NTRU_λ に従って標本抽出された元と $U_\lambda^{\mathrm{NTRU}}$ に従って標本抽出された元とを識別する問題であり，これらの分布族が計算量的に識別不可能であるという仮定を NTRU 仮定と呼ぶ．

$\lambda \in \mathbb{N}$ に対して，決定版 NTRU 問題と同じ R_λ, χ_λ を用いる．さらにパラメータ σ'_λ に対して，分布 $D_\lambda = D_{\sigma'_\lambda}^{n_\lambda}$ を定める．サンプル数を $k = k(\lambda)$ とするとき，分布 RLWE_λ を次のように定める．秘密鍵 $s \leftarrow \chi_\lambda$ を標本抽出し，各 $i = 1, \ldots, k$ について，R_λ 上の一様分布から a_i を，ノイズ $e_i \leftarrow D_\lambda$ を独立に標本抽出する．このとき RLWE_λ は，$\{(a_i,\ a_i \cdot s + e_i)\}_{i=1}^k$ の分布である．また，R_λ 上の一様分布から独立に標本抽出した a_i, u_i からなる $\{(a_i,\ u_i)\}_{i=1}^k$ の分布を $U_\lambda^{\mathrm{RLWE}}$ とする．決定版 RLWE 問題とは，RLWE_λ に従って標本抽出された組と $U_\lambda^{\mathrm{RLWE}}$ に従って標本抽出された組を識別する問題である．本稿では，サンプル数 k が λ に関する多項式で上から抑えられるとき，分布族 $\{\mathrm{RLWE}_\lambda\}_\lambda$ と $\{U_\lambda^{\mathrm{RLWE}}\}_\lambda$ が計算量的に識別不可能であるという仮定を RLWE 仮定と呼ぶ．

3. 提案方式

本章では，Lin らの方式 [20] を公開鍵型かつ整数平文へと拡張した，FHE 方式を提案する．Lin らの方式における暗号化やガジェット分解，Key Switching，Bootstrapping については付録 B を参照されたい．また，提案方式におけるベース暗号方式の IND-CPA 安全性証明の概説を与える．

3.1 提案方式におけるパラメータと主要なアルゴリズム

3.1.1 パラメータ

提案方式では，セキュリティパラメータ λ に応じて各種の方式パラメータが設定される．本節では，これらの方式パラメータを，ベース暗号文に関するパラメータ，演算用暗号文に関するパラメータ，および Key Switching に関するパ

Algorithm 1 IPKBase.KeyGen$(n, q, \sigma, \omega, \ell_{\mathrm{ks}})$

Input: $n,\ q,\ \sigma,\ \omega,\ \ell_{\mathrm{ks}}$
Output: $f_q \in \mathbb{Z}_3[X]/(X^n+1)$
Output: $(h_{\mathrm{pk}}, h'_{\mathrm{pk}}) \in R_q \times R_q$
Output: $h_{\mathrm{ks}} \in R_q$
$f_q \leftarrow \chi_\sigma^n$
while f_q is not invertible in R_q **do**
    $f_q \leftarrow \chi_\sigma^n$
$g_{\mathrm{pk}} \leftarrow \chi_\sigma^n\ ,\ g'_{\mathrm{pk}} \leftarrow \chi_\sigma^n$
$\Delta_{\omega,q} \leftarrow \frac{q}{2^{\omega+1}}$
$h_{\mathrm{pk}} \leftarrow (g_{\mathrm{pk}} + \Delta_{\omega,q}) \cdot f_q^{-1} \pmod{R_q}$
$h'_{\mathrm{pk}} \leftarrow g'_{\mathrm{pk}} \cdot f_q^{-1} \pmod{R_q}$
for $t := 0$ **to** $\ell_{\mathrm{ks}} - 1$ **do**
    $g_t \leftarrow \chi_\sigma^n$
    $h''_t \leftarrow (g_t + q/B_{\mathrm{ks}}^{t+1}) \cdot f_q^{-1} \pmod{R_q}$
$h_{\mathrm{ks}} \leftarrow \{h''_t : 0 \le t < \ell_{\mathrm{ks}}\}$
return $f_q, (h_{\mathrm{pk}}, h'_{\mathrm{pk}}), h_{\mathrm{ks}}$

ラメータの三群に分ける．ベース暗号文に関するパラメータは，多項式の次数 $n \in \mathbb{Z}_{>0}$，モジュラス $q \in \mathbb{Z}_{>0}$，ベース秘密鍵分布，およびノイズ分布の標準偏差 $\sigma, \sigma' \in \mathbb{R}_{>0}$，平文ビット ω である．演算用暗号文に関するパラメータは，多項式の次数 $N \in \mathbb{Z}_{>0}$，モジュラス $Q \in \mathbb{Z}_{>0}$，演算用暗号化秘密鍵分布，およびノイズ分布の標準偏差 $\varsigma \in \mathbb{R}_{>0}$，ガジェット分解パラメータ $\ell_{\mathrm{bsk}}, B_{\mathrm{bsk}} \in \mathbb{Z}_{>0}$ である．Key Switching に関するパラメータは，Key Switching アルゴリズムのためのパラメータ $\ell_{\mathrm{ks}}, B_{\mathrm{ks}} \in \mathbb{Z}_{>0}$ である．なお，本稿では，q, Q, n, N は 2 の冪乗に固定する．

3.1.2 ベース暗号化

ベース鍵生成 IPKBase.KeyGen$(n, q, \sigma, \omega, \ell_{\mathbf{ks}})$: ベース鍵生成アルゴリズムは，パラメータ $n, q, \sigma, \omega, \ell_{\mathrm{ks}}$ を受け取り，ベース秘密鍵 $f_q \in \mathbb{Z}_3[X]/(X^n+1)$ とベース公開鍵 $(h_{\mathrm{pk}}, h'_{\mathrm{pk}})$，そして KSK 用公開鍵 h_{ks} を生成する．この KSK 用公開鍵は，Key Switching 鍵を生成するために用いられる．このアルゴリズムの疑似コードを Algorithm 1 に示す．

ベース暗号化 IPKBase.Enc$(h_{\mathbf{pk}}, h'_{\mathbf{pk}}, m)$: ベース暗号化アルゴリズムは, 平文 $m \in \mathbb{Z}_{2^\omega}[X]/(X^n+1)$ に対し, ベース公開鍵 $h_{\mathrm{pk}}, h'_{\mathrm{pk}}$ を用いてベース暗号文 $c \in R_q$ を生成する．ベース暗号文 c は，乱数 $s_{\mathrm{pk}} \leftarrow \chi_\sigma^n$ とノイズ $e_{\mathrm{pk}} \leftarrow D_{\sigma'}^n$ を用いて，$c := h'_{\mathrm{pk}} \cdot s_{\mathrm{pk}} + h_{\mathrm{pk}} \cdot m + e_{\mathrm{pk}} \pmod{R_q}$ と計算される．

復号 IPKBase.Dec(f_q, c): 復号アルゴリズムは，ベース暗号文 c に対してベース秘密鍵 f_q を用いることで平文 m を生成する．具体的には，スケーリング係数 $\Delta_{\omega,q} := q/2^{\omega+1}$ に対して，IPKBase.Dec$(f_q, c) := \lfloor \varphi_{f_q}(c)/\Delta_{\omega,q} \rceil$ を計算し，$\|\varphi_{f_q}(c)/\Delta_{\omega,q} - m\|_\infty < 1/2$ を満たすとき，m が得られる．

3.1.3 演算用暗号化と演算鍵生成

演算用暗号文は 2 種類あり，これらは EncS 暗号化アルゴリズム と EncV 暗号化アルゴリズムによって生成され，

Algorithm 2 IPKKSKGen$(h_{\mathrm{ks}}, h'_{\mathrm{pk}}, f_Q)$

Input: $h_{\mathrm{ks}} = \{h''_t : 0 \le t < \ell_{\mathrm{ks}}\},\ h'_{\mathrm{pk}} \in R_q$
Input: $f_Q \in R_Q$
Output: KSK $\in (R_q)^{N \times \ell_{\mathrm{ks}}}$
$\sum_{r=0}^{N-1} s_r X^r \leftarrow f_Q$
for $r := 0$ **to** $N-1$ **do**
    for $t := 0$ **to** $\ell_{\mathrm{ks}} - 1$ **do**
        $s'_{r,t} \leftarrow \chi_\sigma^n\ ,\ e_{r,t} \leftarrow D_{\sigma'}^n$
        $\mathrm{KSK}_{r,t} \leftarrow h'_{\mathrm{pk}} \cdot s'_{r,t} + h''_t \cdot s_r + e_{r,t} \pmod{R_q}$
KSK $\leftarrow \{\mathrm{KSK}_{r,t} : 0 \le r < N,\ 0 \le t < \ell_{\mathrm{ks}}\}$
return KSK

それぞれの暗号文は EncS 暗号文，EncV 暗号文と呼ばれる．演算用鍵生成，EncV 暗号化及び Bootstrapping 鍵生成は，Lin らの方式と同じアルゴリズムを用いる．

EncS 暗号化 IPKOpe.EncS(f_Q, u):EncS 暗号化アルゴリズムは，$u \in \mathbb{Z}_{2^\omega}[X]/(X^N+1)$ に対し，演算用暗号化鍵 f_Q を用いて EncS 暗号文 $c_u \in R_Q := \mathbb{Z}_Q[X]/(X^N+1)$ を生成する．具体的には，EncS 暗号文は，スケーリング係数 $\Delta_{\omega,Q} := Q/2^{\omega+1}$ とノイズ $g_u \leftarrow \chi_\varsigma^N$ に対して，$c_u := g_u \cdot f_Q^{-1} + \Delta_{\omega,Q} \cdot u \pmod{R_Q}$ と定義される．

Bootstrapping 鍵生成 IPKBSKGen(f_q, f_Q):
Bootstrapping 鍵生成アルゴリズムは，ベース秘密鍵 f_q と演算用暗号化鍵 f_Q を受け取って，EncV 暗号文の対 $(\mathrm{bsk}_{i,0}, \mathrm{bsk}_{i,1})$ の集合 BSK $:= \left\{(\mathrm{bsk}_{i,0}, \mathrm{bsk}_{i,1})\right\}_{i=0}^{n-1} \in (R_Q)^{n \times 2 \times \ell_{\mathrm{bsk}}}$ を生成する．

Key Switching 鍵生成 IPKKSKGen$(h_{\mathbf{ks}}, h'_{\mathbf{pk}}, f_Q)$:Key Switching 鍵生成アルゴリズムは，$h_{\mathrm{ks}}, h'_{\mathrm{pk}}, f_Q$ を受け取って，ベース暗号文の集合を生成する．このアルゴリズムの疑似コードを，Algorithm 2 に示す．

3.1.4 Programmable Bootstrapping

External Product, CMux, Modulus Switching 及び Key Switching は，Lin らの方式と同じアルゴリズムを用いる．

Native PBS IPKNativePBS$(c, \mathbf{BSK}, \mathbf{KSK}, \{v_j\})$:
Native PBS は，平文 $m := \sum_{j=0}^{n-1} m_j X^j$ に対するベース暗号文 c，Bootstrapping 鍵 BSK，Key Switching 鍵 KSK，および任意の関数の集合 $\{f_j : \mathbb{Z}_{2^\omega} \to \mathbb{Z}_{2^\omega}\}_{j=0}^{n-1}$ を用いて定義されるルックアップテーブルの集合 $\{v_j\}_{j=0}^{n-1}$ を受け取り，平文 $\sum_{j=0}^{n-1} f_j(m_j) X^j$ に対するベース暗号文 c_{bst} を返す．ここで，各ルックアップテーブルは $v_j := \sum_{r=0}^{N-1} F_j(\lfloor r \cdot 2^\omega / N \rceil \bmod 2^\omega)\, \Delta_{\omega,Q}\, X^r$ であり，F_j は関数 f_j と関数 $\mathrm{sgn}(x) := 1$ if $x \ge 0$, -1 otherwise を用いて $F_j(x) := \mathrm{sgn}(x)\, f_j(x)$ と定義される．このアルゴリズムの疑似コードを Algorithm 3 に示す．Lin らの Native Bootstrapping との主な違いは，ルックアップテーブルの構成である．Native PBS では，5 行目で各ルックアップテーブル v_j をアキュムレータの初期値とすることで，ベース暗号文のノイズ削減と同時に平文の各係数 m_j に関数 f_j をそれぞれ適用している．

Algorithm 3 IPKNativePBS($c, \mathrm{BSK}, \mathrm{KSK}, \{v_j\}$)

Input: $c \in R_q$, BSK, KSK, $\{v_j\}_{j=0}^{n-1}$
Output: $c_{\mathrm{bst}} \in R_q$

$c' \leftarrow \mathrm{MS}(c, q, 2N)$
$c'' \leftarrow c'(X^{-1}) \pmod{\mathbb{Z}_{2N}[X]/(X^n+1)}$
for $j := 0$ **to** $n-1$ **do**
  $\tilde{c} \leftarrow c'' \cdot X^j \pmod{\mathbb{Z}_{2N}[X]/(X^n+1)}$
  $\mathrm{ACC}_j \leftarrow v_j$
  $\sum_{i=0}^{n-1} \tilde{c}_i X^i \leftarrow \tilde{c}$
  for $i := 0$ **to** $n-1$ **do**
    $\mathrm{ACC}_j \leftarrow \mathrm{ExtProd}(\mathrm{ACC}_j, \mathrm{CMux}(-\tilde{c}_i, \mathrm{bsk}_{i,0}, \mathrm{bsk}_{i,1}))$
  $\mathrm{ACC}'_j \leftarrow \mathrm{MS}(\mathrm{ACC}_j, Q, q)$
$\mathrm{ACC} \leftarrow (\mathrm{ACC}'_0, \cdots, \mathrm{ACC}'_{n-1})$
$c_{\mathrm{bst}} \leftarrow \mathrm{KS}(\mathrm{ACC}, \mathrm{KSK})$
return c_{bst}

3.2 ベース暗号方式の IND-CPA 安全性

本節では，提案方式を構成するベース暗号方式 Π_{base} が，Multi-sample NTRU 仮定および RLWE 仮定のもとで IND-CPA 安全であることの概説を与える．公開鍵暗号方式の IND-CPA 安全性の定義については付録 A.2 を参照されたい．

まず，本節で用いる Multi-sample NTRU 仮定を述べる．$\lambda \in \mathbb{N}$ に対して，決定版 NTRU 問題と同じ R_λ, χ_λ を用いる．R_λ 上で可逆になるように $f_q \leftarrow \chi_\lambda$ を，独立に $g_{\mathrm{pk}}, g'_{\mathrm{pk}} \leftarrow \chi_\lambda$ を標本抽出し，$h_{\mathrm{pk}} = (g_{\mathrm{pk}} + \Delta_{\omega,q_\lambda}) f_q^{-1}$，$h'_{\mathrm{pk}} = g'_{\mathrm{pk}} f_q^{-1} \pmod{R_\lambda}$ とする．このとき，組 $(h_{\mathrm{pk}}, h'_{\mathrm{pk}})$ の分布を mNTRU_λ とし，h_{pk} と独立な R_λ 上の一様元 a との組 (h_{pk}, a) の分布を $U_\lambda^{\mathrm{mNTRU}}$ とする．Multi-sample NTRU 問題とは，mNTRU_λ と $U_\lambda^{\mathrm{mNTRU}}$ に従って標本抽出された組を識別する問題であり，これらの分布族が計算量的に識別不可能であるという仮定を Multi-sample NTRU 仮定と呼ぶ．以下，攻撃者 $\mathcal{A}$ と挑戦者の間で 3 つのゲーム $\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2$ を考え，各ゲーム $\mathbf{G}_i$ における $\mathcal{A}$ の優位性を $\mathrm{Adv}_i(\lambda)$ と定義する．

$\mathbf{G}_0$: 挑戦者はベース鍵生成である Algorithm 1 を実行してベース公開鍵 $(h_{\mathrm{pk}}, h'_{\mathrm{pk}})$ を $\mathcal{A}$ に与え，ベース秘密鍵 f_q を秘匿する．$\mathcal{A}$ が選んだ同じ長さの 2 つの平文 m_0, m_1 に対し，挑戦者はチャレンジビット $b \in \mathbb{B}$ を一様に選び，チャレンジ暗号文 $c = h'_{\mathrm{pk}} \cdot s_{\mathrm{pk}} + h_{\mathrm{pk}} \cdot m_b + e_{\mathrm{pk}} \pmod{R_q}$ を $\mathcal{A}$ に返す．このゲームは IND-CPA ゲームに一致するため，$\mathrm{Adv}_0(\lambda) = \mathrm{Adv}^{\text{IND-CPA}}_{\Pi_{\mathrm{base}},\mathcal{A}}(\lambda)$ が成り立つ．

$\mathbf{G}_1$:$\mathbf{G}_0$ において，ベース公開鍵の h'_{pk} を h_{pk} と独立な R_q 上の一様分布からの元 a' に置き換えたゲームである．このとき，チャレンジ暗号文は $c = a' \cdot s_{\mathrm{pk}} + h_{\mathrm{pk}} \cdot m_b + e_{\mathrm{pk}}$ となる．$\mathbf{G}_0$ と $\mathbf{G}_1$ は $\mathcal{A}$ に与えるベース公開鍵の分布のみが異なり，$\mathbf{G}_0$ と $\mathbf{G}_1$ それぞれのベース公開鍵の分布は Multi-sample NTRU 仮定のもとで識別不可能である．したがって，$\mathbf{G}_0$ と $\mathbf{G}_1$ の $\mathcal{A}$ の優位性の差 $|\mathrm{Adv}_0(\lambda) - \mathrm{Adv}_1(\lambda)|$ は無視できる．

$\mathbf{G}_2$:$\mathbf{G}_1$ において，チャレンジ暗号文中の $a' \cdot s_{\mathrm{pk}} + e_{\mathrm{pk}}$ を R_q 上の一様分布からの元 u に置き換えたゲームである．このとき，チャレンジ暗号文は $c = u + h_{\mathrm{pk}} \cdot m_b$ となる．ここで，a' は R_q 上の一様分布の元であることから，組 $(a', a' \cdot s_{\mathrm{pk}} + e_{\mathrm{pk}})$ は RLWE 問題のサンプルとみなせる．RLWE 仮定より $(a', a' \cdot s_{\mathrm{pk}} + e_{\mathrm{pk}})$ の分布と (a', u) の分布は識別不可能である．したがって，$\mathbf{G}_1$ と $\mathbf{G}_2$ の $\mathcal{A}$ の優位性の差 $|\mathrm{Adv}_1(\lambda) - \mathrm{Adv}_2(\lambda)|$ は無視できる．

$\mathbf{G}_2$ では，u は h_{pk} およびチャレンジビット b と独立な，R_q 上の一様分布からの元であるため，$c = u + h_{\mathrm{pk}} \cdot m_b$ もまた b に依存しない一様分布からの元となり，b の情報を持たない．よって $\mathrm{Adv}_2(\lambda) = 0$ である．三角不等式より，$\mathrm{Adv}^{\text{IND-CPA}}_{\Pi_{\mathrm{base}},\mathcal{A}}(\lambda) \le |\mathrm{Adv}_0(\lambda) - \mathrm{Adv}_1(\lambda)| + |\mathrm{Adv}_1(\lambda) - \mathrm{Adv}_2(\lambda)| + \mathrm{Adv}_2(\lambda)$ が成り立つ．右辺は Multi-sample NTRU 仮定および RLWE 仮定のもとで無視できるため，$\mathrm{Adv}^{\text{IND-CPA}}_{\Pi_{\mathrm{base}},\mathcal{A}}(\lambda)$ も無視できる．したがって，ベース暗号方式 Π_{base} は IND-CPA 安全である．

4. 提案方式におけるパラメータの導出と実装

本章では，提案方式の bit security および復号失敗確率に関するパラメータの制約を示し，これらの制約を満たす範囲で演算鍵サイズを最適化するパラメータ値を導出する．さらに，TFHE-rs との演算鍵サイズを比較し，実装により Native PBS による SIMD 演算の動作を確認する．

4.1 パラメータが満たすべき制約とパラメータ値の導出

提案方式では，複数のパラメータが用いられ，これらの選択は方式の安全性，復号失敗確率，および演算鍵サイズに影響する．本稿では，提案方式のパラメータを次の二つの制約のもとで定める．第一に，提案方式が 128-bit security を満たす必要がある．第二に，提案方式の主要アルゴリズムにおける復号失敗確率が 2^{-128} 未満であることを要求する．ここで復号失敗確率とは，ベース暗号文のノイズ e が復号の成功条件 $\|e\|_\infty < \Delta_{\omega,q}/2$ を満たさない確率である．これらの制約を満たすパラメータの組は一般に複数存在するため，本稿で設定する探索範囲において，演算鍵サイズが可能な限り小さいものを選択する．以下では，4.1.1 節で bit security に関する制約を，4.1.2 節で復号失敗確率に関する制約をそれぞれ示し，4.1.3 節でこれらの制約のもとで演算鍵サイズを最適化するパラメータを導出する．

4.1.1 Bit Security

暗号方式に対する既知の最良の攻撃に要する計算量が 2^λ 以上であると見積もられるとき，その暗号方式は λ-bit security を満たすという．本稿では，128-bit security を目標とする．3.2 節では，NTRU 仮定のもとでベース公開鍵が，RLWE 仮定のもとでベース暗号文が，それぞれ一様分布から生成された元と計算量的に識別不可能であること

表 2 128-bit security を満たすモジュラスの上限

次数 (n, N)	1024	2048	4096	8192 以上
モジュラス (q, Q) の上限	2^{19}	2^{30}	2^{45}	$> 2^{64}$

から，ベース暗号方式の IND-CPA 安全性を示した．よって，ベース暗号方式が，与えられたパラメータセットの下で 128-bit security を満たすためには，対応するインスタンスから誘導される NTRU 問題および RLWE 問題のそれぞれについて，既知の最良の解読に基づいて見積もられる計算コストが 2^{128} 以上であることが必要条件である．

ベース公開鍵および BSK に対応する NTRU 問題の解読計算コストは，NTRU Fatigue [13] を用いて見積もり，ベース暗号文に対応する RLWE 問題の解読計算コストは，Lattice Estimator [2] を用いて見積もる[*1]．なお，本稿では，NTRU インスタンスや RLWE インスタンスの秘密鍵分布の各係数が，$-1, 0, 1$ をそれぞれ確率 $1/4, 1/2, 1/4$ で独立にとるものとし，この係数分布を ternary 分布と呼ぶ．NTRU 問題や RLWE 問題では，次数と秘密鍵分布，ノイズ分布を固定したとき，モジュラスが大きいほど問題の解読計算コストは減少するため，128-bit security を満たすようなモジュラスとして取れる値には上限がある．一方で，モジュラスを大きくとるほど，復号に失敗しないようなノイズの上限 $\Delta_{\omega,q}/2$ が大きくなり，ガジェット分解の桁数 $\ell_{\mathrm{bsk}}, \ell_{\mathrm{ks}}$ をより少ない値で取ることができ，演算鍵サイズの削減につながる．したがって，次数と秘密鍵分布，ノイズ分布を固定したときに，128-bit security を満たすようなモジュラスの上限を調べ，以降ではモジュラスの値を上限で固定する．

本節では，次数を 1024 から 131072 までの 2 冪とし，ベース公開鍵に対応する NTRU 問題，ベース暗号文に対応する RLWE 問題，および BSK に対応する NTRU 問題について，解読計算コストが 2^{128} 以上となるモジュラスの上限を求める．q の上限は，ベース公開鍵およびベース暗号文から得られた上限のうち小さい方とし，Q の上限は BSK から得られた上限とする．NTRU Fatigue および Lattice Estimator を用いて計算コストを見積った結果，n に対する q の上限と N に対する Q の上限は一致した．よって，以降では，(n, q) と (N, Q) を区別せず，各次数に対するモジュラスの上限を表 2 で示す．なお，次数が 8192 以上のときはモジュラスの上限が常に 2^{64} を超えた．実装の制約上，モジュラスを高々 2^{64} で表現するため，次数が 8192 以上の際のモジュラスは 2^{64} で固定する．

4.1.2 主要アルゴリズムの復号失敗確率

本節では，Algorithm 3 の 1 行目で実行される Modulus Switching および Algorithm 3 で定義される Native PBS の出力におけるノイズ e の分布を解析し，それぞれの出力暗号文の復号失敗確率の上界が bit security に合わせて 2^{-128} 未満となるように設定する．

Modulus Switching および Native PBS の出力におけるノイズ多項式 e の各係数は，ternary 分布に従うノイズやベース秘密鍵，平均 0，標準偏差 σ' の離散ガウス分布に従うノイズ，演算用暗号化鍵，ガジェット分解の出力である多項式の各係数，ガジェット分解の打ち切り誤差，および丸め込み誤差の積からなる項の和として表される．各項は，少なくとも一つの平均 0 のノイズを因子として含むため，ノイズ分布を互いに独立と仮定すると，e の各係数の分布の平均は 0 となる．そこで，e の各係数の分布を，平均 0，分散 σ_{out}^2 の正規分布 $\mathcal{N}(0, \sigma_{\mathrm{out}}^2)$ で近似する．正規分布 $\mathcal{N}(0, \sigma^2)$ に従う確率変数 X が区間 $[-t, t]$ に収まらない確率は，相補誤差関数 $\mathrm{erfc}(x) = \dfrac{2}{\sqrt{\pi}} \displaystyle\int_x^\infty e^{-u^2}\, du$ を用いて $\Pr[|X| \geq t] = \mathrm{erfc}\left(\dfrac{t}{\sqrt{2}\,\sigma}\right)$ と表される．したがって，e の 1 つの係数に対して，その絶対値が閾値 $\Delta_{\omega,q}/2$ 以上となり復号失敗する確率 P_{fail} は，e の各係数の分布の標準偏差 σ_{out} を用いて $P_{\mathrm{fail}}(\sigma_{\mathrm{out}}) := \mathrm{erfc}\left(\dfrac{\Delta_{\omega,q}/2}{\sqrt{2}\,\sigma_{\mathrm{out}}}\right)$ と定義される．提案方式は n 個の平文を多項式の各係数に格納するため，少なくとも 1 つの係数が復号に失敗する確率は，$P_{\mathrm{fail_union}}(\sigma_{\mathrm{out}}) \leq n \cdot P_{\mathrm{fail}}(\sigma_{\mathrm{out}})$ で上から抑えられる．したがって，Modulus Switching および Native PBS の出力のノイズ分布の分散を導出し，その平方根として得られる標準偏差を $P_{\mathrm{fail_union}}(\sigma_{\mathrm{out}})$ に代入し，出力が 2^{-128} 未満となれば，Native PBS によって出力されるベース暗号文の復号失敗確率は 2^{-128} 未満であることが分かる．

Modulus Switching によるノイズの分散は，

$$\mathrm{Var}(e_{\mathrm{MS}}) \leq \left(\tfrac{2N}{q}\right)^2 \left[\, n\,\sigma^4 + 2^{2\omega-2}\, n\,\sigma^2 + n\,\sigma'^2\,\sigma^2 \,\right] + \tfrac{n}{12}\sigma^2$$

で与えられる．また，Native PBS によるノイズの分散は，

$$\sigma_{\mathrm{ks}}^2 \leq \sigma_{\mathrm{ms}}^2 + N\ell_{\mathrm{ks}}\, n \left(\mathrm{Var}(G_{\mathrm{ks}}^{-1}) + \mathbb{E}[G_{\mathrm{ks}}^{-1}]^2\right) \mathrm{Var}(\bar{g}) + N\,\mathrm{Var}(\varepsilon_{\mathrm{ks}})\,\mathbb{E}[f_Q^2]$$

で与えられる．ここで，$\mathbb{E}[f_Q^2] = \mathrm{Var}(f_Q) + \mathbb{E}[f_Q]^2$ は演算用秘密鍵の二次モーメントを表す．これらの導出は，Appendix C を参照されたい．

4.1.3 演算鍵のサイズ最適化とパラメータ値の導出

提案方式の演算鍵は, BSK と KSK から構成され, 演算鍵全体のサイズは，これらの和とする．本実装では，多項式の各係数を 32 ビットまたは 64 ビットの整数として格納する．モジュラスのビット長 m（$m = \log_2 q$ または $m = \log_2 Q$）に対して, 1 係数当たりの格納ビット数 $w(m)$ を, $m \leq 32$ のとき $w(m) = 32$，$32 < m \leq 64$ のとき $w(m) = 64$ と定め

[*1] 安全性の見積もりには NTRU Fatigue estimator（`https://github.com/WvanWoerden/NTRUFatigue`, commit `02971b3`, 2026-03-11）および Lattice Estimator（`https://github.com/malb/lattice-estimator`, commit `352ddaf`, 2025-09-17）を用いた．

表 3 各 ω に対する最適なパラメータと演算鍵サイズ

ω	n	N	q	Q	σ^2, ς^2	σ'	k	$(B_{\mathrm{bsk}}, \ell_{\mathrm{bsk}})$	$(B_{\mathrm{ks}}, \ell_{\mathrm{ks}})$	BSK	KSK	合計
4	2048	4096	2^{30}	2^{45}	1/2	1	1	$(2^{13}, 2)$	$(2^3, 7)$	268.44 MB	234.88 MB	503.32 MB
6	4096	32768	2^{45}	2^{64}	1/2	1	1	$(2^{31}, 1)$	$(2^{13}, 2)$	2.15 GB	2.15 GB	4.29 GB
8	4096	131072	2^{45}	2^{64}	1/2	1	1	$(2^{36}, 1)$	$(2^8, 4)$	8.59 GB	17.18 GB	25.77 GB

る．BSK は，各 $i = 0, \ldots, n-1$ に対する EncV 暗号文の対 $(\mathrm{bsk}_{i,0}, \mathrm{bsk}_{i,1})$ から構成され，各 EncV 暗号文は N 個の係数を持つ ℓ_{bsk} 個の多項式からなる．また，KSK は，n 個の係数を持つ $N\ell_{\mathrm{ks}}$ 個の多項式からなる．よって，それぞれの鍵サイズは，$\mathrm{Size(BSK)} = 2\,n\,\ell_{\mathrm{bsk}}\,N\,w(\log_2 Q)/8$ [byte]，$\mathrm{Size(KSK)} = n\,\ell_{\mathrm{ks}}\,N\,w(\log_2 q)/8$ [byte] で与えられる．4.1.1 節に示した安全性に関する条件および 4.1.2 節に示した復号失敗確率に関する条件を満たすパラメータの組を探索する．そのうち，演算鍵サイズに最適化されたパラメータを選択する．選定したパラメータと，それらに対応する演算鍵のサイズを表 3 に示す．

4.2 演算鍵サイズの比較

整数型平文の FHE 方式として実装されている TFHE-rs に対して，bit security，復号失敗確率，扱える平文の大きさなどを揃えた上で，演算鍵のサイズを比較する．

表 3 に記載されているパラメータを用いた提案方式の演算鍵サイズと，公開鍵型パラメータ `V1_6_PARAM_MESSAGE_`μ`_CARRY_`μ`_COMPACT_PK_PBS_KS_GAUSSIAN_2M128`（$\mu = 2, 3, 4$）による TFHE-rs の演算鍵サイズの比較を表 4 に示す．なお，$\mu = 2, 3, 4$ はそれぞれ $\omega = 4, 6, 8$ と対応している．結果として，いずれの ω においても提案方式の演算鍵サイズが TFHE-rs より大きい．提案方式の演算鍵サイズが大きくなった要因は，安全性を担保するために多項式次数を大きく取る必要があったことや，公開鍵型への拡張に伴い，Algorithm 2 の 5 行目において Key Switching のノイズの値が増大したことにある．復号失敗確率の制約下に抑えるために提案方式ではより大きなパラメータを要する．

そこで，公開鍵化に伴う Key Switching のノイズの増大を除いた場合で評価するため，提案方式を公開鍵型から共通鍵型に変更し，再度 TFHE-rs との演算鍵サイズの比較を行う．Lin らの方式を整数平文に拡張した FHE 方式に対して，4.1.1 節に示した安全性に関する条件および 4.1.2 節に示した復号失敗確率に関する条件を満たすような，演算鍵サイズに最適化されたパラメータを導出する．Lin らの方式を整数平文に拡張した FHE 方式の鍵サイズと，共通鍵型パラメータ `V1_6_PARAM_MESSAGE_`μ`_CARRY_`μ`_PBS_KS_GAUSSIAN_2M128`（$\mu = 2, 3, 4$）による TFHE-rs の鍵サイズの比較を表 5 に示す．結果として，$\omega = 6, 8$ において，提案方式の演算鍵サイズが TFHE-rs の約半分だった．これは，共通鍵型に変更したことで，Key Switching のノイズが小さくなり，結果として，KSK サイズが大幅に減少したことに起因する．一方で $\omega = 4$ では TFHE-rs よりも提案方式の方が演算鍵サイズが大きい．これは，多項式次数など BSK のサイズに関わるパラメータに変化がなく BSK のサイズが大きいままであったためである．

4.3 実装検証

提案方式を実装し，$\omega = 4$ のとき表 3 のパラメータのもとで，$\mathbb{Z}_{2^\omega}$ から一様ランダムに選んだ $n = 2048$ 個の値を係数とするような平文を用いて，Native PBS を実行した．結果として，全係数の復号結果が入力平文に対する期待値と一致し，SIMD 演算が正しく動作することを確認した．

5. 結論

本稿では，多項式 NTRU 問題に基づいた Lin らの FHE 方式を，公開鍵型かつ整数平文を扱う FHE 方式へ拡張し，Native PBS を用いた SIMD 演算が正しく動作することを実装により確認した．提案方式を構成するベース暗号方式が Multi-sample NTRU 仮定および RLWE 仮定のもとで IND-CPA 安全性を満たすことを示した．また，提案方式が 128-bit security と復号失敗確率 2^{-128} 未満を満たし，さらに，演算鍵サイズを最適化するようなパラメータを導出した．演算鍵サイズを諸条件を揃えて TFHE-rs と比較した結果，公開鍵型ではいずれの平文ビットでも提案方式が TFHE-rs を上回ったが，共通鍵型に変更すると $\omega = 6, 8$ では TFHE-rs の約半分に収まった．

今後の課題は．ベース暗号方式の IND-CPA 安全性の証明で用いた Multi-sample NTRU 仮定の妥当性の検討である．Multi-sample NTRU 仮定と標準的な NTRU 仮定との帰着関係を明らかにすることで，提案方式の安全性を検証する．また，演算鍵サイズのさらなる削減に取り組む．モジュラスを大きく取ることができれば，復号失敗確率の条件から，ガジェット分解の桁数や次数を減らすことができ，演算鍵サイズの削減につながる．一方で，現状では，多項式の次数や秘密鍵分布を固定したときに，NTRU 問題の制約によりモジュラスを大きく取ることができない．これらの課題への方針として，ベース暗号方式が基づく問題を，RLWE 問題へ変更することが挙げられる．

参考文献

[1] Al Badawi, A. et al.: OpenFHE: Open-Source Fully Homomorphic Encryption Library, *Proceedings of the 10th Workshop on Encrypted Computing & Applied Homo-*

表 4 公開鍵型の FHE 方式における BSK・KSK サイズの比較

	提案方式（公開鍵型）			TFHE-rs		
ω	BSK	KSK	サイズ和	BSK	KSK	サイズ和
4	268.44 MB	234.88 MB	503.32 MB	64.0 MB	80.1 MB	144.1 MB
6	2.15 GB	2.15 GB	4.30 GB	2.00 GB	256.1 MB	2.26 GB
8	8.59 GB	17.18 GB	25.77 GB	12.0 GB	2.00 GB	14.0 GB

表 5 共通鍵型の FHE 方式における BSK・KSK サイズの比較

	提案方式（共通鍵型）			TFHE-rs		
ω	BSK	KSK	サイズ和	BSK	KSK	サイズ和
4	268.44 MB	67.11 MB	335.55 MB	64.1 MB	96.2 MB	160.3 MB
6	536.87 MB	536.87 MB	1.07 GB	1.16 GB	1.02 GB	2.18 GB
8	2.15 GB	5.37 GB	7.52 GB	7.97 GB	5.98 GB	13.95 GB

morphic Cryptography (WAHC 2022), pp. 53–63 (2022).

[2] Albrecht, M. R., Player, R. and Scott, S.: On the Concrete Hardness of Learning with Errors, *Journal of Mathematical Cryptology*, Vol. 9, No. 3, pp. 169–203 (2015).

[3] Alexandru, A., Kim, A. and Polyakov, Y.: General Functional Bootstrapping using CKKS, Cryptology ePrint Archive, Paper 2024/1623 (2024).

[4] Bonte, C., Iliashenko, I., Park, J., Pereira, H. V. L. and Smart, N. P.: FINAL: Faster FHE Instantiated with NTRU and LWE, *Advances in Cryptology – ASIACRYPT 2022*, LNCS, Vol. 13792, pp. 188–215 (2022).

[5] Bos, J. W., Lauter, K., Loftus, J. and Naehrig, M.: Improved Security for a Ring-Based Fully Homomorphic Encryption Scheme, *Cryptography and Coding (IMACC 2013)*, LNCS, Vol. 8308, pp. 45–64 (2013).

[6] Boura, C., Gama, N., Georgieva, M. and Jetchev, D.: Simulating Homomorphic Evaluation of Deep Learning Predictions, *Cyber Security Cryptography and Machine Learning (CSCML 2019)*, LNCS, Vol. 11527, pp. 212–230 (2019).

[7] Bourse, F., Minelli, M., Minihold, M. and Paillier, P.: Fast Homomorphic Evaluation of Deep Discretized Neural Networks, *Advances in Cryptology – CRYPTO 2018*, LNCS, Vol. 10993, pp. 483–512 (2018).

[8] Brakerski, Z., Gentry, C. and Vaikuntanathan, V.: (Leveled) Fully Homomorphic Encryption without Bootstrapping, *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS 2012)*, pp. 309–325 (2012).

[9] Cheon, J. H., Kim, A., Kim, M. and Song, Y.: Homomorphic Encryption for Arithmetic of Approximate Numbers, *Advances in Cryptology – ASIACRYPT 2017*, LNCS, Vol. 10624, pp. 409–437 (2017).

[10] Chillotti, I., Gama, N., Georgieva, M. and Izabachène, M.: TFHE: Fast Fully Homomorphic Encryption over the Torus, *Journal of Cryptology*, Vol. 33, No. 1, pp. 34–91 (2020).

[11] Chillotti, I., Joye, M. and Paillier, P.: Programmable Bootstrapping Enables Efficient Homomorphic Inference of Deep Neural Networks, *Cyber Security Cryptography and Machine Learning (CSCML 2021)*, LNCS, Vol. 12716, pp. 1–19 (2021).

[12] Ducas, L. and Micciancio, D.: FHEW: Bootstrapping Homomorphic Encryption in Less Than a Second, *Advances in Cryptology – EUROCRYPT 2015*, LNCS, Vol. 9056, pp. 617–640 (2015).

[13] Ducas, L. and van Woerden, W.: NTRU Fatigue: How Stretched is Overstretched?, *Advances in Cryptology – ASIACRYPT 2021*, LNCS, Vol. 13093, pp. 3–32 (2021).

[14] EPFL-LDS and Tune Insight SA: Lattigo v6, Online: `https://github.com/tuneinsight/lattigo` (2024).

[15] Fan, J. and Vercauteren, F.: Somewhat Practical Fully Homomorphic Encryption, Cryptology ePrint Archive, Paper 2012/144 (2012).

[16] Gentry, C.: Fully Homomorphic Encryption Using Ideal Lattices, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC 2009)*, pp. 169–178 (2009).

[17] Hoffstein, J., Pipher, J. and Silverman, J. H.: NTRU: A Ring-Based Public Key Cryptosystem, *Algorithmic Number Theory (ANTS-III)*, LNCS, Vol. 1423, pp. 267–288 (1998).

[18] Joye, M.: TFHE Public-Key Encryption Revisited, *Topics in Cryptology – CT-RSA 2024*, LNCS, Vol. 14643, pp. 277–291 (2024).

[19] Li, Z. et al.: BatchBoot: Fast Batched Bootstrapping for TFHE Scheme and Practical Applications, *Proceedings of the 35th USENIX Security Symposium (USENIX Security 2026)* (2026).

[20] Lin, W.-T. and Wu, J.-L.: From Matrix to Polynomial NTRU FHE: Enabling Amortized Bootstrapping via Sparse Keys, Cryptology ePrint Archive, Paper 2026/068, version 20260502:062413 (2026).

[21] López-Alt, A., Tromer, E. and Vaikuntanathan, V.: On-the-Fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption, *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC 2012)*, pp. 1219–1234 (2012).

[22] Lyubashevsky, V., Peikert, C. and Regev, O.: On Ideal Lattices and Learning with Errors over Rings, *Advances in Cryptology – EUROCRYPT 2010*, LNCS, Vol. 6110, pp. 1–23 (2010).

[23] Regev, O.: On Lattices, Learning with Errors, Random Linear Codes, and Cryptography, *Journal of the ACM*, Vol. 56, No. 6, p. 34 (2009).

[24] Zama: TFHE-rs: A Pure Rust Implementation of the TFHE Scheme for Boolean and Integer Arithmetics Over Encrypted Data, Online: `https://github.com/zama-ai/tfhe-rs` (2022).

付　　録

A　IND-CPA 安全性

A.1　分布族に対する識別器と計算量的識別不可能性

二つの分布族に対する識別器および計算量的識別不可能性を定義する．分布族を $X = \{X_\lambda\}_{\lambda\in\mathbb{N}}$，$Y = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ とするとき，識別器 $\mathcal{D}$ とは，任意の λ に対して，1^λ と X_λ または Y_λ から標本抽出された値を受け取り，ビットを出力する確率的多項式時間アルゴリズムである．$x \leftarrow X_\lambda$ および $y \leftarrow Y_\lambda$ に対する識別器 $\mathcal{D}$ の識別優位性を $\mathrm{Adv}_{\mathcal{D},X,Y}(\lambda) := \left|\Pr[\mathcal{D}(1^\lambda, x) = 1] - \Pr[\mathcal{D}(1^\lambda, y) = 1]\right|$ と定義する．任意の識別器 $\mathcal{D}$ に対して，$\mathrm{Adv}_{\mathcal{D},X,Y}$ が λ について無視できるとき，分布族 X と Y は計算量的に識別不可能であるという．ここで，関数 $\psi : \mathbb{N} \to [0,1]$ が無視できるとは，任意の定数 $c > 0$ に対してある $\lambda_0 \in \mathbb{N}$ が存在し，すべての $\lambda \geq \lambda_0$ について $\psi(\lambda) < \lambda^{-c}$ が成り立つことをいう．

A.2　公開鍵暗号方式の IND-CPA 安全性

公開鍵暗号方式 Π_{pk} の機密性を，選択平文攻撃に対する識別不可能性（indistinguishability under chosen-plaintext attack; IND-CPA）によって定義する．セキュリティパラメータを λ とし，攻撃者 $\mathcal{A}$ と挑戦者との間で，以下の安全性ゲームを実行する．まず，挑戦者はベース公開鍵 pk とベース秘密鍵 sk を生成する．挑戦者は pk を $\mathcal{A}$ に与え，sk を秘匿する．次に，$\mathcal{A}$ は同じ長さの二つの平文 m_0, m_1 を選択し，挑戦者に送る．挑戦者はビット $b \in \mathbb{B}$ を一様ランダムに選択し，m_b をベース暗号化することで，挑戦暗号文 $\mathsf{ct}^\star \leftarrow \mathsf{BaseEnc}(\mathsf{pk}, m_b)$ を生成し，$\mathcal{A}$ に返す．最後に，$\mathcal{A}$ は $\mathsf{ct}^\star$ から b の推定値 $b' \in \mathbb{B}$ を出力する．このとき，$\mathcal{A}$ の優位性を $\mathrm{Adv}^{\text{IND-CPA}}_{\Pi_{\mathrm{pk}},\mathcal{A}}(\lambda) := \left|\Pr[b' = b] - \frac{1}{2}\right|$ と定義する．任意の確率的多項式時間攻撃者 $\mathcal{A}$ に対して，$\mathrm{Adv}^{\text{IND-CPA}}_{\Pi_{\mathrm{pk}},\mathcal{A}}$ が λ について無視できるとき，ベース暗号方式 Π_{pk} は IND-CPA 安全であるという．

B　Lin らの方式

Lin らの方式 [20] における暗号化や Bootstrapping について述べる．ただし，Lin らの方式内の Amortized Bootstrapping は扱わない．

まず，Lin らの方式内で用いられるガジェット分解を定義する．B と ℓ を正整数とする．多項式 $c \in R_q$ に対して，ガジェット分解 $G^{-1}(c)$ とは，c を ℓ 個の多項式 $(d_1, \ldots, d_\ell) \in R_q^\ell$ へ近似的に分解する操作であり，各 d_i は $\|d_i\|_\infty < B/2$ を満たす．また，ガジェット分解における打ち切り誤差 ε とは，$\varepsilon := c - \sum_{i=1}^{\ell} d_i \cdot (q/B^i)$ のことであり，$\|\varepsilon\|_\infty \leq q/(2B^\ell)$ を満たす．

Algorithm 4 Base.KeyGen(n, q, σ)

Input: $n,\ q,\ \sigma$
Output: $f_q \in \mathbb{Z}_3[X]/(X^n + 1)$
1: $f_q \leftarrow \chi_\sigma^n$
2: **while** f_q is not invertible in R_q **do**
3: 　　$f_q \leftarrow \chi_\sigma^n$
4: **return** f_q

B.1　ベース暗号化

ベース鍵生成 Base.KeyGen(n, q, σ): ベース鍵生成アルゴリズムは，パラメータ n, q, σ を受け取り，R_q 上で可逆なベース秘密鍵 $f_q \in \mathbb{Z}_3[X]/(X^n + 1)$ を生成する．このアルゴリズムの疑似コードを Algorithm 4 に示す．1 行目では，ベース秘密鍵の分布 χ_σ^n に対して，$f_q \leftarrow \chi_\sigma^n$ のサンプリングが行われる．2,3 行目では，f_q が R_q 上で可逆でない場合，f_q の再サンプリングを繰り返す．

ベース暗号化 Base.Enc(f_q, m): ベース暗号化アルゴリズムは，平文 $m \in \mathbb{B}[X]/(X^n + 1)$ に対し，ベース秘密鍵 f_q を用いてベース暗号文 $c \in R_q$ を生成する．ベース暗号文 c は，スケーリング係数 $\Delta_q := \lfloor q/4 \rfloor$ とノイズ $g \leftarrow \chi_\sigma^n$ を用いて，$c := (g + \Delta_q \cdot m) \cdot f_q^{-1} \pmod{R_q}$ と計算される．

復号 Base.Dec(f_q, c): 復号アルゴリズムは，ベース暗号文 c に対してベース秘密鍵 f_q を用いることで平文 m を生成する．具体的には，位相関数を $\varphi_{f_q}(c) := c \cdot f_q$ と定義すると，復号アルゴリズムは $\mathrm{Base.Dec}(f_q, c) := \lfloor \varphi_{f_q}(c)/\Delta_q \rceil$ と表せる．ここで，$\|\varphi_{f_q}(c)/\Delta_q - m\|_\infty < 1/2$ を満たすとき，m が得られる．

B.2　演算用暗号化と演算鍵生成

演算用暗号文は 2 種類あり，これらは，EncS 暗号化アルゴリズムと EncV 暗号化アルゴリズムによって生成され，それぞれの暗号文は EncS 暗号文，EncV 暗号文と呼ばれる．また，演算鍵も BSK と KSK の 2 種類があり，これらの演算鍵は BSKGen アルゴリズムと KSKGen アルゴリズムによって，それぞれ生成される．演算用暗号化鍵生成，EncS 暗号化，EncV 暗号化，及び Bootstrapping 鍵生成の詳細は，[20] を参照されたい．

演算用暗号化鍵生成 Ope.KeyGen(N, Q, ς): 演算用暗号化鍵生成アルゴリズムは，パラメータ N, Q, ς を受け取り，R_Q 上で可逆な演算用暗号化鍵 $f_Q \in R_Q$ を生成する．

EncS 暗号化 Ope.EncS(f_Q, u):EncS 暗号化アルゴリズムは，$u \in \mathbb{Z}_3[X]/(X^N + 1)$ に対し，演算用暗号化鍵 f_Q を用いて EncS 暗号文 $c_u \in R_Q$ を生成する．具体的には，EncS 暗号文は，スケーリング係数 $\Delta_Q := \lfloor Q/4 \rfloor$ とノイズ $g_u \leftarrow \chi_\varsigma^N$ に対して，$c_u := g_u \cdot f_Q^{-1} + \Delta_Q \cdot u \pmod{R_Q}$ と定義される．

EncV 暗号化 Ope.EncV(f_Q, v):EncV 暗号化アルゴリ

Algorithm 5 KSKGen(f_q, f_Q)

Input: $f_q \in \mathbb{Z}_3[X]/(X^n+1)$, $f_Q \in R_Q$
Output: KSK $\in (R_q)^{N\times \ell_{\mathrm{ks}}}$
1: $\sum_{r=0}^{N-1} s_r X^r \leftarrow f_Q$
2: **for** $r := 0$ **to** $N-1$ **do**
3: **for** $t := 0$ **to** $\ell_{\mathrm{ks}}-1$ **do**
4: $g_{r,t} \leftarrow \chi_\sigma^n$
5: $\mathrm{KSK}_{r,t} \leftarrow (g_{r,t} + (q/B_{\mathrm{ks}}^{t+1}) \cdot s_r) \cdot f_q^{-1} \pmod{R_q}$
6: KSK $\leftarrow \{\mathrm{KSK}_{r,t} : 0 \le r < N,\ 0 \le t < \ell_{\mathrm{ks}}\}$
7: **return** KSK

Algorithm 6 NativeBS$(c, \mathrm{BSK}, \mathrm{KSK}, v)$

Input: $c \in R_q$, BSK, KSK, v
Output: $c_{\mathrm{bst}} \in R_q$
1: $c' \leftarrow \mathrm{MS}(c, q, 2N)$
2: $c'' \leftarrow c'(X^{-1}) \pmod{\mathbb{Z}_{2N}[X]/(X^n+1)}$
3: **for** $j := 0$ **to** $n-1$ **do**
4: $\tilde{c} \leftarrow c'' \cdot X^j \pmod{\mathbb{Z}_{2N}[X]/(X^n+1)}$
5: $\mathrm{ACC}_j \leftarrow v$
6: $\sum_{i=0}^{n-1} \tilde{c}_i X^i \leftarrow \tilde{c}$
7: **for** $i := 0$ **to** $n-1$ **do**
8: $\mathrm{ACC}_j \leftarrow \mathrm{ExtProd}(\mathrm{ACC}_j, \mathrm{CMux}(-\tilde{c}_i, \mathrm{bsk}_{i,0}, \mathrm{bsk}_{i,1}))$
9: $\mathrm{ACC}'_j \leftarrow \mathrm{MS}(\mathrm{ACC}_j, Q, q)$
10: $\mathrm{ACC} \leftarrow (\mathrm{ACC}'_0, \cdots, \mathrm{ACC}'_{n-1})$
11: $c_{\mathrm{bst}} \leftarrow \mathrm{KS}(\mathrm{ACC}, \mathrm{KSK})$
12: **return** c_{bst}

ズムは，パラメータ $\ell_{\mathrm{bsk}}, B_{\mathrm{bsk}}$ と $v \in \{\pm b \cdot X^k : b \in \{0,1\},\ k \in \mathbb{Z}_{\ge 0}\}$ に対し，演算用暗号化鍵 f_Q を用いて EncV 暗号文 $c_v \in (R_Q)^{\ell_{\mathrm{bsk}}}$ を生成する．

Bootstrapping 鍵生成 BSKGen(f_q, f_Q):Bootstrapping 鍵生成アルゴリズムは，ベース秘密鍵 f_q と演算用暗号化鍵 f_Q を受け取って，EncV 暗号文の対 $(\mathrm{bsk}_{i,0}, \mathrm{bsk}_{i,1})$ の集合 $\mathrm{BSK} := \left\{(\mathrm{bsk}_{i,0}, \mathrm{bsk}_{i,1})\right\}_{i=0}^{n-1} \in (R_Q)^{n\times 2\times \ell_{\mathrm{bsk}}}$ を生成する．

Key Switching 鍵生成 KSKGen(f_q, f_Q):Key Switching 鍵生成アルゴリズムは，f_q, f_Q を受け取って，ベース暗号文の集合を生成する．このアルゴリズムの疑似コードを Algorithm 5 に示す．1 行目で，f_Q の各係数を s_r と定め，4 行目でノイズ $g_{r,t} \leftarrow \chi_\sigma^n$ が定まる．5 行目で，s_r に対し，ベース秘密鍵 f_q を用いてベース暗号文 $\mathrm{KSK}_{r,t}$ が生成される．これを $N \times \ell_{\mathrm{ks}}$ 回繰り返すことで，$\mathrm{KSK} \in (R_q)^{N\times \ell_{\mathrm{ks}}}$ が導出される．

B.3 Bootstrapping

External Product，CMux 及び Key Switching の詳細は [20] を参照されたい．

External Product ExtProd(c_u, c_v):External Product は，EncS 暗号文 $c_u = \mathrm{Ope.EncS}(f_Q, u)$ と EncV 暗号文 $c_v = \mathrm{Ope.EncV}(f_Q, v)$ を受け取って，EncS 暗号文 $\mathrm{Ope.EncS}(f_Q, u \cdot v)$ を返す．

CMux CMux$(a_i, \mathbf{bsk}_{i,0}, \mathbf{bsk}_{i,1})$:$i \in \{0, 1, \ldots, n-1\}$ に対して，CMux は Bootstrapping 鍵 BSK の i 番目の要素 $(\mathrm{bsk}_{i,0}, \mathrm{bsk}_{i,1})$ と整数 $a_i \in \mathbb{Z}$ を受け取り，ベース秘密鍵 $f_q := \sum_{i=0}^{n-1} f_i X^i$ に対して定義される EncV 暗号文 $\mathrm{Ope.EncV}(f_Q, X^{a_i \cdot f_i})$ を返す．

Modulus Switching MS(c, a, b):Modulus Switching は，正の整数 a, b に対して，ベース暗号文 $c \in \mathbb{Z}_a[X]/(X^n+1)$ をベース暗号文 $c' \in \mathbb{Z}_b[X]/(X^n+1)$ へ変換する，もしくは EncS 暗号文 $c \in \mathbb{Z}_a[X]/(X^N+1)$ を EncS 暗号文 $c' \in \mathbb{Z}_b[X]/(X^N+1)$ へ変換するアルゴリズムである．具体的には，c の各係数 c_j に対して $c'_j := \left\lfloor \frac{b}{a} \cdot c_j \right\rceil \pmod b$ を行う．

Key Switching KS$(\mathbf{ACC}, \mathbf{KSK})$:Key Switching は，演算用暗号化鍵 f_Q に対する $\mathbb{Z}_q[X]/(X^N+1)$ 上の EncS 暗号文の集合 $\mathrm{ACC} = \{\mathrm{Ope.EncS}(f_Q, u_j(X))\}_{j=0}^{n-1}$ と Key Switching 鍵 KSK を受け取り，各 $u_j(X)$ の定数項 $u_{j,0}$ とベース秘密鍵 f_q に対するベース暗号文 $\mathrm{Base.Enc}(f_q, \sum_{j=0}^{n-1} u_{j,0} X^j)$ を返す．

Native Bootstrapping NativeBS$(c, \mathbf{BSK}, \mathbf{KSK}, v)$:
Native Bootstrapping は，平文 $m := \sum_{j=0}^{n-1} m_j X^j$ に対するベース暗号文 $c = \mathrm{Base.Enc}(f_q, m)$，Bootstrapping 鍵 BSK，Key Switching 鍵 KSK，ルックアップテーブル $v := \Delta_Q \cdot X^{N/4} \sum_{r=0}^{N/2} X^r$ を受け取り，ノイズが削減されたベース暗号文 $\mathrm{Base.Enc}(f_q, m)$ を返す．このアルゴリズムの疑似コードを Algorithm 6 に示す．1 行目で，入力のベース暗号文 $c \in R_q$ が，Modulus Switching によりベース暗号文 $c' \in \mathbb{Z}_{2N}[X]/(X^n+1)$ に変換される．2〜8 行目で，$\varphi_{f_q}(c')$ の j 番目の係数 $(\varphi_{f_q}(c'))_j$ とルックアップテーブル v に対して，EncS 暗号文 $\mathrm{ACC}_j := \mathrm{Ope.EncS}\left(f_Q, v \cdot X^{-(\varphi_{f_q}(c'))_j}/\Delta_Q\right)$ が生成される．この際，$v \cdot X^{-(\varphi_{f_q}(c'))_j}/\Delta_Q$ の定数項は m_j となる．9〜11 行目で，Modulus Switching と Key Switching によってベース暗号文 $c_{\mathrm{bst}} := \mathrm{Base.Enc}(f_q, m)$ が得られる．なお，[20] の NativeBS は，ブール演算を経てスケーリングが $\lfloor q/2 \rfloor$ となったベース暗号文を入力とする．一方，本稿では整数平文への拡張を目的としている関係上，ベース暗号文のスケーリングを一貫して $\Delta_q = \lfloor q/4 \rfloor$ で固定する．したがって，Bootstrapping に入力されるベース暗号文のスケーリングも $\lfloor q/4 \rfloor$ となり，整合するようにルックアップテーブル v を構成する．

C 主要なアルゴリズムにおけるノイズの分散の導出

4.1.2 節で用いた Modulus Switching および Native PBS の出力におけるノイズの分散の導出を与える．演算用秘密

鍵の二次モーメントを $\mathbb{E}[f_Q^2] = \mathrm{Var}(f_Q) + \mathbb{E}[f_Q]^2$ と表す．f_Q は定数項とそれ以外で分布が異なるが，本稿では，簡単のため全ての係数が定数項と分布であり，同じ二次モーメントをもつものとして一律に $\mathbb{E}[f_Q^2]$ を評価する．また，$\lfloor \cdot \rceil$ による丸め込み誤差は，区間 $[-1/2, 1/2]$ 上の連続一様分布に従うと仮定し，丸め込み誤差の平均は 0，分散は $1/12$ となる．

C.1 Modulus Switching のノイズ分散

Algorithm 3 の 1 行目で実行される Modulus Switching 直後のベース暗号文に含まれるノイズ e_{MS} の分散は，

$$\mathrm{Var}(e_{\mathrm{MS}}) \le \left(\tfrac{2N}{q}\right)^2 \left[n\,\sigma^4 + 2^{2\omega-2}\, n\,\sigma^2 + n\,\sigma'^2\,\sigma^2 \right] + \tfrac{n}{12}\sigma^2$$

で与えられる．第 1 項の括弧内はベース暗号文が持つノイズの分散で，第 2 項は丸め込み誤差にベース秘密鍵 f_q を掛けて生じたノイズの分散である．ここで，係数 $2^{2\omega-2}$ は，ベース暗号文に含まれる平文 $m \in \mathbb{Z}_{2^\omega}[X]/(X^n+1)$ に由来し，本稿では m の各係数を取りうる最大値 $2^{\omega-1}$ で見積もっている．Modulus Switching の出力であるベース暗号文が復号失敗する確率は，モジュラス q に換算した標準偏差 $\sqrt{\mathrm{Var}(e_{\mathrm{MS_modq}})} = (q/2N)\sqrt{\mathrm{Var}(e_{\mathrm{MS}})}$ を，P_{fail} の式に代入することで評価する．

C.2 Native PBS のノイズ分散

Algorithm 3 の Native PBS は，ベース暗号文に対して，8 行目で n 回の External Product，9 行目で EncS 暗号文に対する Modulus Switching，そして，11 行目で Key Switching を順に適用し，ベース暗号文を出力する．まず n 回の External Product 直後の EncS 暗号文に含まれるノイズの分散は，

$$\begin{aligned}\sigma^2_{\mathrm{ACC}} \le\ & nN\ell_{\mathrm{bsk}}\,\alpha\left(\mathrm{Var}(G^{-1}_{\mathrm{bsk}}) + \mathbb{E}[G^{-1}_{\mathrm{bsk}}]^2\right)\varsigma^2 \\ & + \varsigma^2 + nN\,\mathrm{Var}(\varepsilon_{\mathrm{bsk}})\,\mathbb{E}[f_Q^2]\end{aligned}$$

となる．ここで α は秘密鍵分布に依存する定数であり，ternary 分布では $\alpha = 4$ である．$\mathrm{Var}(G^{-1}_{\mathrm{bsk}})$ と $\mathbb{E}[G^{-1}_{\mathrm{bsk}}]^2$ は，パラメータ $B_{\mathrm{bsk}}, \ell_{\mathrm{bsk}}$ に対するガジェット分解 G^{-1} が出力する各多項式の係数の分散と平均の二乗である．付録 B 記載のガジェット分解より，この係数は $\{-B_{\mathrm{bsk}}/2, \ldots, B_{\mathrm{bsk}}/2-1\}$ 上のほぼ一様分布に従い平均 $-1/2$ をもつため，$\mathrm{Var}(G^{-1}_{\mathrm{bsk}}) = (B^2_{\mathrm{bsk}}-1)/12$，$\mathbb{E}[G^{-1}_{\mathrm{bsk}}]^2 = 1/4$ である．$\varepsilon_{\mathrm{bsk}}$ はガジェット分解の打ち切り誤差であり，$M_{\mathrm{bsk}} := Q/B^{\ell_{\mathrm{bsk}}}_{\mathrm{bsk}}$ と定義すると，平均 0，分散 $\mathrm{Var}(\varepsilon_{\mathrm{bsk}}) = M^2_{\mathrm{bsk}}/12 + 1/6$ と書ける．次に，Modulus Switching 直後の EncS 暗号文に含まれるノイズの分散を σ^2_{ms} とすると，

$$\sigma^2_{\mathrm{ms}} = \left(\tfrac{q}{Q}\right)^2 \sigma^2_{\mathrm{ACC}} + \tfrac{N}{12}\,\mathbb{E}[f_Q^2]$$

となる．最後に，Key Switching 直後のベース暗号文に含まれるノイズの分散を σ^2_{ks} とすると，

$$\begin{aligned}\sigma^2_{\mathrm{ks}} \le\ & \sigma^2_{\mathrm{ms}} + N\ell_{\mathrm{ks}}\,n\left(\mathrm{Var}(G^{-1}_{\mathrm{ks}}) + \mathbb{E}[G^{-1}_{\mathrm{ks}}]^2\right)\mathrm{Var}(\bar{g}) \\ & + N\,\mathrm{Var}(\varepsilon_{\mathrm{ks}})\,\mathbb{E}[f_Q^2]\end{aligned}$$

と書ける．ここで G^{-1}_{ks} と $\varepsilon_{\mathrm{ks}}$ は，$M_{\mathrm{ks}} := q/B^{\ell_{\mathrm{ks}}}_{\mathrm{ks}}$ に対して，$\mathrm{Var}(G^{-1}_{\mathrm{ks}}) = (B^2_{\mathrm{ks}}-1)/12$，$\mathbb{E}[G^{-1}_{\mathrm{ks}}]^2 = 1/4$，$\mathrm{Var}(\varepsilon_{\mathrm{ks}}) = M^2_{\mathrm{ks}}/12 + 1/6$ である．また $\bar{g}$ は KSK に含まれるノイズであり，その分散は $\mathrm{Var}(\bar{g}) = n\,\sigma^4 + \sigma^2\,\mathbb{E}[f_Q^2] + n\,\sigma'^2\,\sigma^2$，平均は 0 である．Native PBS が出力するベース暗号文に含まれるノイズ e_{PBS} の分散 $\mathrm{Var}(e_{\mathrm{PBS}})$ に対して，等号 $\mathrm{Var}(e_{\mathrm{PBS}}) = \sigma^2_{\mathrm{ks}}$ が成り立つ．このノイズの分散 $\mathrm{Var}(e_{\mathrm{PBS}})$ の平方根として得られる標準偏差を $P_{\mathrm{fail_union}}$ の式に代入し，出力が 2^{-128} 未満となるパラメータを選ぶ．