Fully Secure Searchable Encryption from PRFs, Pairings, and Lattices

Hirotomo Shinoki*, Hisayoshi Sato, and Masayuki Yoshino

Hitachi, Ltd., Japan

November 5, 2024

Abstract

Searchable encryption is a cryptographic primitive that allows us to perform searches on encrypted data. Searchable encryption schemes require that ciphertexts do not leak information about keywords. However, most of the existing schemes do not achieve the security notion that trapdoors do not leak information. Shen et al. (TCC 2009) proposed a security notion called full security, which includes both ciphertext privacy and trapdoor privacy, but there are few fully secure constructions. Full security is defined for the secret key settings since it is known that public key schemes cannot achieve the trapdoor privacy in principle.

In this paper, we construct a query-bounded fully secure scheme from pseudorandom functions. In addition, we propose three types of efficient (unbounded) fully secure schemes. One of them is based on bilinear groups, and the others are besed on lattices. We then analyze the existing constructions. First, we simplify the Cheng et al. scheme (Information Sciences 2023) and prove its security. This scheme had not been proved to be secure. Second, we show that the Li-Boyen pairing-based scheme (IACR CiC 2024) does not achieve the trapdoor privacy, not as claimed.

1 Introduction

Searchable encryption is a cryptosystem that allows keyword search on encrypted data. There are two types of searchable encryption: Searchable Symmetric Encryption (SSE) [33] and Public key Encryption with Keyword Search (PEKS) [7]. The secret key is used for encryption in SSE, while the public key is used in PEKS. To search for a keyword w, one generates a trapdoor T_w using the secret key. Then, by running the test algorithm, anyone can check whether the given ciphertext is associated with w.

Searchable encryption requires that ciphertexts leak no information on the keywords. However, most of the existing schemes do not satisfy the security notion that trapdoors leak no information. With respect to the public key setting, it is known that PEKS cannot achieve trapdoor privacy in principle. In fact, Byun et al. [10] showed that PEKS is vulnerable to the Keyword Guessing Attack (KGA) and the associated keyword is leaked from the trapdoor when the keyword space is small. In KGA, the attacker tries to find which keyword corresponds to the given trapdoor. The attacker can choose a keyword, encrypt it, and check whether the chosen keyword corresponds to the trapdoor by running the test algorithm. If the number of candidate keywords is small, the attacker can easily find the correct keyword. SSE can achieve the trapdoor privacy, but many SSE schemes use deterministic trapdoor generation algorithms. In these cases, anyone can know whether the given two trapdoors are associated with the same keyword.

In 2009, Shen et al. [32] proposed the security notion called predicate privacy for Predicate Encryption (PE) in the secret key setting. Predicate-private PE for equality is equivalent to

*hirotomo.shinoki.sw@hitachi.com

trapdoor-private SSE. They proposed an inner product predicate encryption with selective predicate privacy using the composite order bilinear groups. They also formulated the notion of full security, which implies both ciphertext privacy and trapdoor privacy. These security notions can be generalized to Functional Encryption (FE), and there has been a lot of research on function-private FE schemes [2,3,4,6,9,13,20,22,34]. However, these schemes have different functionalities from SSE or have complex structure to support rich functionalities. Until recently, there has been no research on the construction of simple schemes that support the equality predicate.

In 2024, Li and Boyen [21] proposed two efficient trapdoor-private SSE schemes. These schemes are based on pairings and lattices respectively. They also proposed a generic construction of Public key Authenticated Encryption with Keyword Search (PAEKS) from trapdoor-private SSE and Non-Interactive Key Exchange (NIKE). PAEKS is a variant primitive of PEKS, which was introduced by Huang and Li [19] in 2017 to prevent KGA. As mentioned above, PEKS is vulnerable to KGA in principle. In PAEKS, the data sender has its own secret key to authenticate the ciphertext. Since the test algorithm works correctly only if the ciphertext is authenticated, PAEKS has the potential to prevent KGA. Many PAEKS schemes have been proposed [11, 12, 16, 23, 28, 29, 30], but most PAEKS schemes achieve only limited trapdoor privacy. Besides the Li-Boyen constructions, there are two PAEKS with (unlimited) trapdoor privacy [11, 12]. These schemes imply trapdoor-private SSE.

1.1 Our Contributions

In this paper, we propose four types of fully secure SSE.

The first construction uses a pseudorandom function. This construction achieves the bounded version of full security. In the bounded full security setting, the upper bound of the number of queries must be determined before the key generation phase. Although this setting is a weaker variant, the security can be proved without any additional assumption. The other schemes we propose achieve (unbounded) full security by introducing additional assumptions.

The second construction uses a pseudorandom function and bilinear groups. This construction is based on the Uniform Matrix Decisional Diffie-Hellman assumption, which we write as $\mathcal{U}_k$-MDDH. This assumption is parameterized by an integer $k \geq 1$. Increasing k makes the assumption weaker, but makes the scheme less efficient. Note that the $\mathcal{U}_k$-MDDH assumption is weaker than the k-Lin assumption.

The third construction uses a pseudorandom function and lattices. This construction is based on the (Ring) Learning with Errors assumption. In addition, it is quite simple compared to the existing schemes. In fact, it does not involve complicated lattice algorithms such as the preimage sampling.

The forth construction is also based on a pseudorandom function and lattices. By introducing the NTRU assumption, this scheme is more efficient than the third construction.

The comparison of SSE with trapdoor privacy is summarized in Table 1. CM22 and CQFM23 are PAEKS with trapdoor privacy. It has been claimed that the security of CQFM23 follows from the Computational Oracle Diffie-Hellman (CODH) assumption in [12], but Li and Boyen [21] pointed out that there is an error in the security analysis. It was also shown that CODH is insufficient and that at least the Decisional Bilinear Diffie-Hellman (DBDH) assumption is required. Thus, no security proof has been given for CQFM23. Note that it seems difficult to prove the security directly from the DBDH assumption. This is because the bilinear map only appears in the search algorithm and does not appear in the security analysis. In this paper, we simplify CQFM23 and show that its security is based on the Decisional Linear (DLin) assumption. This scheme can be seen as a variant of our $\mathcal{U}_2$-MDDH-based construction and can be extended to k-Lin-based construction. We also show that CQFM23 can be generalized to the asymmetric bilinear group setting. In this case, the full security is based on the bilateral k-Lin (bil-k-Lin) assumption. We also analyze the existing constructions by Li and Boyen [21]. In particular, their security proof for the pairing-based scheme (LB24-P) is incorrect, and the trapdoor privacy can be broken. The Li-Boyen lattice-based construction framework can be used to construct LWE-based scheme, RLWE-based scheme, and NTRU-based scheme. We write them as LB24-L, LB24-R, and LB24-N respectively.

Ours1 is the first SSE from symmetric primitives that achieves (at least) bounded trapdoor privacy. Ours2 is more efficient than the existing pairing-based constructions when $k = 1$. For lattice-based constructions, Ours3 is much more efficient compared to CM22 and LB24-L. Similarly, Ours3 (Ring) and Ours4 are more efficient than LB24-R and LB24-N respectively. The existing lattice-based constructions use the lattice trapdoor generation algorithm [5,14,26] or the preimage sampling algorithm [18]. On the other hand, our lattice-based constructions do not use such complicated algorithms and consist only of simple computations.

Finally, we note that ciphertext privacy and trapdoor privacy have been considered in the previous research, but full security has not. Full security implies "ciphertext privacy and trapdoor privacy," but not vice versa. In this paper, we prove that our constructions achieve full security instead of proving ciphertext privacy and trapdoor privacy.

Table 1: Comparison of SSE with Trapdoor Privacy

	Ciphertext	Trapdoor	Assumptions	Remarks
CM22 [11]	$\Omega(\kappa_{\mathsf{ct}} n \log^2 q)$	$\Omega(\kappa_{\mathsf{td}} n \log^2 q)$	LWE	
CQFM23 [12]	$4\|G\|$	$4\|G\|$	**PRF, DLin**	Modified
LB24-P [21]	$2\|G_1\| + \|G_T\|$	$2\|G_2\|$	(PRF, DBDH)	**Broken**
LB24-L [21]	$\Omega(\kappa_{\mathsf{ct}} n \log^2 q)$	$\Omega(\kappa_{\mathsf{td}} n \log^2 q)$	PRF, LWE	
LB24-R [21]	$\Omega(N \log^2 q)$	$\Omega(N \log^2 q)$	PRF, RLWE	
LB24-N [21]	$\approx \frac{3}{2} N \log q$	$\approx \frac{3}{2} N \log q$	PRF, RLWE, NTRU	
Ours1	$Q_{\mathsf{ct}} + Q_{\mathsf{td}} \cdot \|\mathbb{F}\|$	$Q_{\mathsf{td}} + Q_{\mathsf{ct}} \cdot \|\mathbb{F}\|$	PRF	Q-bounded
Ours2	$2k\|G_1\|$	$2k\|G_2\|$	PRF, U_k-MDDH	
Ours3	$\kappa_{\mathsf{ct}} \cdot 3n \log q$	$\kappa_{\mathsf{td}} \cdot 3n \log q$	PRF, LWE	
Ours3 (Ring)	$3N \log q$	$3N \log q$	PRF, RLWE	
Ours4	$N \log q$	$N \log q$	PRF, NTRU	
Gen. CQFM23	$2k\|G_1\|$	$2k\|G_2\|$	PRF, bil-k-Lin	

For a finite set S, $|S|$ denotes the bit size of a random element in S. CQFM23 uses a symmetric bilinear map from $G \times G$. LB24-P, Ours2, and Generalized CQFM23 use a bilinear map from $G_1 \times G_2$ to G_T. Q_{ct} and Q_{td} denote upper bounds of the number of ciphertext queries and trapdoor queries respectively. For simplicity, it is assumed that Q_{ct} and Q_{td} are large enough in Ours1.

1.2 Paper Organization

In Section 2, we summarize the basic definitions used in this paper. In Section 3, 4, and 5, we propose the PRF-based bounded construction (Ours1), the pairing-based construction (Ours2), and the lattice-based construction (Ours3 and Ours4) respectively. In Section 6, we analyze the Cheng et al. scheme (CQFM23) and show that its security can be reduced to the (bil-)DLin assumption. In Section 7, we analyze the Li-Boyen schemes and give an attack for the pairing-based scheme.

2 Preliminaries

The basic notations used in this paper are summarized here. "Probabilistic Polynomial-Time" is abbreviated to "PPT". For a finite set S, $x \xleftarrow{\$} S$ means sampling x uniformly at random from S, and $\mathrm{Unif}(S)$ denotes the uniform distribution over S. We say that a function $f : \mathbb{N} \to \mathbb{R}$ is negligible and write $f(n) = \mathsf{negl}(n)$ if for any positive integer k there exists an integer n_k such that $|f(n)| < n^{-k}$ for any $n > n_k$. We say that a probability $p(n)$ is overwhelming if $1 - p(n)$ is negligible. Let $\mathbb{Z}_q$ denote the quotient ring $\mathbb{Z}/q\mathbb{Z}$. Let $\log x$ denote $\log_2 x$. We use the following notations for matrix concatenation:

$$[A|B] = \begin{pmatrix} A & B \end{pmatrix}, \quad [A;C] = \begin{pmatrix} A \\ C \end{pmatrix}.$$

2.1 Pseudorandom Function (PRF)

Let $\{\mathcal{K}_\lambda\}_{\lambda\in\mathbb{N}}$, $\{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$, and $\{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ be families of finite sets. A family of functions $F = \{F_\lambda : \mathcal{K}_\lambda \times \mathcal{X}_\lambda \to \mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ is said to be pseudorandom if for any PPT adversary $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{A},F}(\lambda) := \frac{1}{2}\left|\Pr[\mathsf{K} \xleftarrow{\$} \mathcal{K}_\lambda : 1 \leftarrow \mathcal{A}^{F_\lambda(\mathsf{K},\cdot)}(1^\lambda)] - \Pr[f \xleftarrow{\$} \mathsf{Func}[\mathcal{X}_\lambda, \mathcal{Y}_\lambda] : 1 \leftarrow \mathcal{A}^{f(\cdot)}(1^\lambda)]\right|$$

is negligible in λ. Here, $\mathsf{Func}[A,B]$ denotes the set of all functions from A to B. In this paper, we often omit the description of λ and write $F_\lambda(\mathsf{K},x)$ as $F(\mathsf{K},x)$ or $F_{\mathsf{K}}(x)$.

2.2 Bilinear Groups

The bilinear group generator $\mathcal{G}$ is a PPT algorithm that outputs $\mathbb{G} := (p, G_1, G_2, G_T, g_1, g_2, e) \leftarrow \mathcal{G}(1^\lambda)$. Here, p is a prime of $\Theta(\lambda)$ bits, G_1, G_2, G_T are cyclic groups of order p, g_1 is a generator of G_1, g_2 is a generator of G_2, and $e : G_1 \times G_2 \to G_T$ is a non-degenerate bilinear map. For matrices $A = (a_{i,j}) \in \mathbb{Z}_p^{m\times n}$, we write $[A]_1 := (g_1^{a_{i,j}}) \in G_1^{m\times n}$, $[A]_2 = (g_2^{a_{i,j}}) \in G_2^{m\times n}$, and $[A]_T = (e(g_1,g_2)^{a_{i,j}}) \in G_T^{m\times n}$. Also, for matrices A and B, we write $e([A]_1,[B]_2) := [AB]_T$. Note that $[AB]_T$ is efficiently computable from $([A]_1,[B]_2)$.

Definition 1 (Matrix Diffie-Hellman Assumption)**.** Let $\ell > k$ and $\mathcal{D}_{\ell,k}$ be a distribution on $\mathbb{Z}_p^{\ell\times k}$. We say that the $\mathcal{D}_{\ell,k}$-Matrix Decisional Diffie-Hellman ($\mathcal{D}_{\ell,k}$-MDDH) assumption on G_i $(i = 1,2)$ holds if $(\mathbb{G}, [A]_i, [Ar]_i)$ is computationally indistinguishable from $(\mathbb{G}, [A]_i, [u]_i)$ where $A \leftarrow \mathcal{D}_{\ell,k}$, $r \xleftarrow{\$} \mathbb{Z}_p^k$, and $u \xleftarrow{\$} \mathbb{Z}_p^\ell$. We just say that the $\mathcal{D}_{\ell,k}$-MDDH assumption holds if it holds for both G_1 and G_2.

We say that the bilateral $\mathcal{D}_{\ell,k}$-Matrix Decisional Diffie-Hellman (bil-$\mathcal{D}_{\ell,k}$-MDDH) assumption holds if $(\mathbb{G}, [A]_1, [Ar]_1, [A]_2, [Ar]_2)$ is computationally indistinguishable from $(\mathbb{G}, [A]_1, [u]_1, [A]_2, [u]_2)$.

In this paper, we mainly consider the case that $\ell = k+1$ and $\mathcal{D}_{\ell,k}$ is uniform. We call this case the U_k-MDDH assumption. It is known that U_k-MDDH is weaker than any $\mathcal{D}_{k+1,k}$-MDDH. In particular, the k-Lin assumption implies the U_k-MDDH assumption. We use U_k-MDDH in the following form.

Lemma 1. For $i = 1,2$ and $r \in \mathbb{Z}_p^k$, let $\mathcal{O}^r_{G_i}$ be an oracle that samples $a \xleftarrow{\$} \mathbb{Z}_p^k$ and outputs $([a]_i, [a^\top r]_i)$. Let $\mathcal{O}_{G_i}$ be an oracle that samples $a \xleftarrow{\$} \mathbb{Z}_p^k$, $u \xleftarrow{\$} \mathbb{Z}_p$ and outputs $([a]_i, [u]_i)$. If the U_k-MDDH assumption on G_i holds, for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}^{U_k\text{-}\mathsf{MDDH}}_{i,\mathcal{A}}(\lambda) := \frac{1}{2}\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^r_{G_i}}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_{G_i}}(1^\lambda)]\right|$$

is negligible for a randomly chosen secret $r \xleftarrow{\$} \mathbb{Z}_p^k$.

Proof. Using a U_k-MDDH instance $([A]_i, [s]_i)$, the oracles in this lemma can be simulated in the following way: one samples $t \xleftarrow{\$} \mathbb{Z}_p^{k+1}$ and outputs $([t^\top A]_i, [t^\top s]_i)$. Since the bit length of p is $\Theta(\lambda)$, $A \in \mathbb{Z}_p^{(k+1)\times k}$ is rank k with an overwhelming probability. Thus, it simulates $\mathcal{O}^r_{G_i}$ when $s = Ar$. Similarly, when s is random, $[A|s] \in \mathbb{Z}_p^{(k+1)\times(k+1)}$ is full-rank with an overwhelming probability. Thus, it simulates $\mathcal{O}_{G_i}$ in this case. □

For $m \geq 2$, m-instance version of U_k-MDDH is defined by replacing $r \xleftarrow{\$} \mathbb{Z}_p^k$ with $r \xleftarrow{\$} \mathbb{Z}_p^{k\times m}$ and $u \xleftarrow{\$} \mathbb{Z}_p$ with $u \xleftarrow{\$} \mathbb{Z}_p^m$ in the setting of Lemma 1. In this case, we write the advantage as $\mathsf{Adv}^{U_k\text{-}\mathsf{MDDH}_m}_{i,\mathcal{A}}(\lambda)$. Then, the following tight reduction holds.

Lemma 2 ([17])**.** For any PPT adversary $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}$ such that

$$\mathsf{Adv}_{i,\mathcal{A}}^{U_k\text{-}\mathsf{MDDH}_m}(\lambda) \leq \mathsf{Adv}_{i,\mathcal{B}}^{U_k\text{-}\mathsf{MDDH}}(\lambda) + \frac{1}{2(p-1)}.$$

2.3 Lattices

We introduce the assumptions on lattices used in this paper.

Definition 2 (LWE Assumption)**.** For each positive integer λ, let $n(\lambda)$ be a positive integer, $q(\lambda) \geq 3$ be an integer, and $\chi(\lambda)$ be an error distribution on $\mathbb{Z}_q$.

(Normal) LWE [31] For $s \in \mathbb{Z}_q^n$, let $\mathcal{O}_s$ be the oracle that samples $a \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, $x \leftarrow \chi$ and outputs $(a, a^\top s + x)$. Let $\mathcal{O}_\$$ be the oracle that outputs $(a,b) \overset{\$}{\leftarrow} \mathbb{Z}_q^n \times \mathbb{Z}_q$. We say that the (n,q,χ)-LWE assumption holds if for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{LWE}_{(n,q,\chi)}}(\lambda) := \frac{1}{2}\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_s}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_\$}(1^\lambda)]\right|$$

is negligible for a randomly chosen secret $s \overset{\$}{\leftarrow} \mathbb{Z}_q^n$. If $\mathcal{A}$ makes at most m queries, we write the advantage as $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{LWE}_{n,m,q,\chi}}(\lambda)$. We use similar notations for the LWE-variant assumptions and the NTRU assumption defined below.

Non-uniform LWE [8] Let η be a distribution on $\mathbb{Z}_q^n$. For $s \in \mathbb{Z}_q^n$, let $\mathcal{O}_s^\eta$ be the oracle that samples $a \leftarrow \eta$, $x \leftarrow \chi$ and outputs $(a, a^\top s + x)$. Let $\mathcal{O}_\$^\eta$ be the oracle that outputs $(a,b) \leftarrow \eta \times \mathrm{Unif}(\mathbb{Z}_q)$. We say that the (n,q,χ,η)-NLWE assumption holds if for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{NLWE}_{(n,q,\chi,\eta)}}(\lambda) := \frac{1}{2}\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_s^\eta}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_\$^\eta}(1^\lambda)]\right|$$

is negligible for a randomly chosen secret $s \overset{\$}{\leftarrow} \mathbb{Z}_q^n$.

Boneh et al. [8] proved that the (k,q,χ,η)-NLWE problem is as hard as the (n,q,χ)-LWE problem if η satisfies the property called coset sampleability.

Definition 3 (Coset Sampleability [8])**.** A distribution η on $\mathbb{Z}_q^k$ is called n-coset sampleable if there exist two PPT algorithms MatrixGen and SamplePre such that:

- $\mathsf{MatrixGen}(1^\lambda)$ outputs $M \in \mathbb{Z}_q^{n\times k}$ and auxiliary data T.
- $\mathsf{SamplePre}(z \in \mathbb{Z}_q^n, M, T)$ outputs $y \in \mathbb{Z}_q^k$ such that $My = z$. In addition, if z is sampled uniformly at random, the distribution of y is η.

Lemma 3 ([8])**.** If η is n-coset sampleable, the (k,q,χ,η)-NLWE problem is at least as hard as the (n,q,χ)-LWE problem. Namely, for any PPT adversary $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}$ such that

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{NLWE}_{k,m,q,\chi,\eta}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{n,m,q,\chi}}(\lambda).$$

Next, we review the notion of Ring-LWE (RLWE) assumption.

Definition 4 (RLWE Assumption)**.** Let $N(\lambda)$ be a power of 2, $q(\lambda)$ be a positive integer, and $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^N+1)$. Let $\chi(\lambda)$ be an error distribution on $\mathcal{R}_q$. For $s \in \mathcal{R}_q$, let $\mathcal{O}_s$ be the oracle that samples $a \overset{\$}{\leftarrow} \mathcal{R}_q$, $x \leftarrow \chi$ and outputs $(a, as + x)$. Let $\mathcal{O}_\$$ be the oracle that outputs $(a,b) \overset{\$}{\leftarrow} \mathcal{R}_q^2$. We say that the (N,q,χ)-RLWE assumption holds if for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{RLWE}_{(N,q,\chi)}}(\lambda) := \frac{1}{2}\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_s}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_\$}(1^\lambda)]\right|$$

is negligible for a randomly chosen secret $s \overset{\$}{\leftarrow} \mathcal{R}_q$.

Finally, we introduce the NTRU assumption.

Definition 5 (NTRU assumption)**.** Let $N(\lambda)$ and $q(\lambda)$ be positive integers, and $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^N + 1)$. Let $\chi(\lambda)$ be a distribution on $\mathcal{R}_q$ and $\chi^\times$ be the conditional distribution of χ on $\mathcal{R}_q^\times$, where $\mathcal{R}_q^\times$ is the multiplicative group of $\mathcal{R}_q$. For $f \in \mathcal{R}_q^\times$, let $\mathcal{O}_f$ be the oracle that samples $g \leftarrow \chi$ and outputs $f^{-1}g$. Let $\mathcal{O}_\$$ be the oracle that outputs $h \overset{\$}{\leftarrow} \mathcal{R}_q$. We say that the (N, q, χ)-NTRU assumption holds if for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{NTRU}_{(N,q,\chi)}}(\lambda) := \frac{1}{2}\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_f}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_\$}(1^\lambda)]\right|$$

is negligible for $f \leftarrow \chi^\times$.

2.4 Searchable Symmetric Encryption

The following definition of Searchable Symmetric Encryption (SSE) is a special case of the symmetric predicate-only encryption proposed by Shen et al [32]. It is called as Symmetric-Key Equality-Predicate Encryption (EPE) in [21].

- $\mathsf{KeyGen}(1^\lambda)$: Given a security parameter λ, it outputs a public parameter pp and a secret key K.
- $\mathsf{Enc}(\mathsf{pp}, \mathsf{K}, w)$: Given a public parameter pp, a secret key K, and a keyword w, it outputs a ciphertext C.
- $\mathsf{Trapdoor}(\mathsf{pp}, \mathsf{K}, w)$: Given a public parameter pp, a secret key K, and a keyword w, it outputs a trapdoor T.
- $\mathsf{Test}(\mathsf{pp}, T, C)$: Given a public parameter pp, a trapdoor T, and a ciphertext C, it outputs a bit $b \in \{0, 1\}$.

For simplicity, we often omit the description of pp from the input of algorithms.

We say that the SSE scheme is correct if $\mathsf{Test}(T, C)$ almost always outputs 1 when T and C have been generated from the same keyword. We say that the SSE scheme is consistent if $\mathsf{Test}(T, C)$ almost always outputs 0 when T and C have been generated from different keywords. Our formulations below are based on the definitions for PEKS [1].

Definition 6 (Correctness)**.** We say that an SSE scheme is correct if the minimum value of

$$\Pr[(\mathsf{pp}, \mathsf{K}) \leftarrow \mathsf{KeyGen}(1^\lambda); C \leftarrow \mathsf{Enc}(\mathsf{K}, w); T \leftarrow \mathsf{Trapdoor}(\mathsf{K}, w) : \mathsf{Test}(T, C) = 1]$$

with respect to w is overwhelming in λ.

Definition 7 (Consistency)**.** We say that an SSE scheme is (computationally) consistent if for any PPT algorithm $\mathcal{A}$,

$$\begin{aligned}\Pr[(\mathsf{pp}, \mathsf{K}) \leftarrow \mathsf{KeyGen}(1^\lambda); (w, w') \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp}); C \leftarrow \mathsf{Enc}(\mathsf{K}, w);\\ T \leftarrow \mathsf{Trapdoor}(\mathsf{K}, w') : \mathsf{Test}(T, C) = 0 \lor w = w']\end{aligned}$$

is overwhelming in λ.

We introduce the notion of full security for SSE. As weaker properties, the definitions of ciphertext privacy and trapdoor privacy are given in Appendix B.

Definition 8 (Full security [32])**.** We say that an SSE scheme Σ is fully secure if for any PPT algorithm $\mathcal{A}$, the advantage

$$\begin{aligned}\mathsf{Adv}_{\mathcal{A},\Sigma}^{\mathsf{Full}}(\lambda) := \Bigg| &\Pr[(\mathsf{pp}, \mathsf{K}) \leftarrow \mathsf{KeyGen}(1^\lambda); b \overset{\$}{\leftarrow} \{0, 1\};\\ &b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{ct}}^b(\mathsf{K},\cdot,\cdot), \mathcal{O}_{\mathsf{td}}^b(\mathsf{K},\cdot,\cdot)}(1^\lambda, \mathsf{pp}) : b = b'] - \frac{1}{2}\Bigg|\end{aligned}$$

is negligible in λ. Here, $\mathcal{O}_{\mathsf{ct}}^b(\mathsf{K}, \cdot, \cdot)$ outputs $C \leftarrow \mathsf{Enc}(\mathsf{K}, w_b)$ given (w_0, w_1) as input. $\mathcal{O}_{\mathsf{td}}^b(\mathsf{K}, \cdot, \cdot)$ outputs $T \leftarrow \mathsf{Trapdoor}(\mathsf{K}, w_b)$ given (w_0, w_1) as input. The restriction is that for any input (x_0, x_1) to $\mathcal{O}_{\mathsf{ct}}^b$ and any input (y_0, y_1) to $\mathcal{O}_{\mathsf{td}}^b$, $(x_0, x_1) = (y_0, y_1)$ or "$x_0 \neq y_0 \land x_1 \neq y_1$" holds.

Let $Q_{\mathsf{ct}}(\lambda)$ (resp. $Q_{\mathsf{td}}(\lambda)$) be an upper bound polynomial of the number of ciphertext (resp. trapdoor) queries. In the "bounded version" of full security, Q_{td} and Q_{ct} are determined before KeyGen phase. We write $\mathsf{KeyGen}(1^\lambda, Q_{\mathsf{td}}, Q_{\mathsf{ct}})$ instead of $\mathsf{KeyGen}(1^\lambda)$ when considering bounded full security. We write the advantage of the bounded full security game as $\mathsf{Adv}^{\mathsf{BFull}}_{\mathcal{A},\Sigma}(\lambda)$.

3 Bounded fully secure SSE from PRF

In this section, we propose an SSE construction from PRF, which we write as $\mathsf{SSE1}$. In addition, we show that $\mathsf{SSE1}$ satisfies correctness, consistency, and bounded full security.

3.1 Construction

Let $\mathbb{F}$ be a finite field of order $q(\lambda) = 2^{\omega(\log \lambda)}$. Let $\mathcal{W}$ denote the keyword space. The construction of $\mathsf{SSE1}$ is as follows:

- $\mathsf{KeyGen}(1^\lambda, Q_{\mathsf{td}}, Q_{\mathsf{ct}})$:
  1. Sets $d_1 \geq Q_{\mathsf{td}}$ and $d_2 \geq Q_{\mathsf{ct}}$.
  2. Chooses a pseudorandom function $F : \mathcal{K} \times \mathcal{W} \to \mathbb{F}^{d_1 \times d_2}$.
  3. Chooses a distribution η_i on $\mathbb{F}^{d_i}$ for $i = 1, 2$. For each η_i, it is required that d_i independent samples from η_i become linearly independent with an overwhelming probability.
  4. Samples $\mathsf{K} \xleftarrow{\$} \mathcal{K}$.
  5. Outputs the public parameter F, η_1, η_2 and the secret key K.
- $\mathsf{Enc}(\mathsf{K}, w)$: Samples $x \leftarrow \eta_2$ and outputs $(c_1, c_2) := (x, F_{\mathsf{K}}(w)x)$.
- $\mathsf{Trapdoor}(\mathsf{K}, w)$: Samples $y \leftarrow \eta_1$ and outputs $(t_1, t_2) := (y, F_{\mathsf{K}}(w)^\top y)$.
- $\mathsf{Test}((t_1, t_2), (c_1, c_2))$: Outputs 1 if $c_1^\top t_2 = c_2^\top t_1$, otherwise outputs 0.

$\eta_i = \mathrm{Unif}(\mathbb{F}^{d_i})$ satisfies the above condition since $q = 2^{\omega(\log \lambda)}$. Instead, different distributions may be used for more efficient constructions. For example, when q is prime and $\eta_i = \mathrm{Unif}(\{0,1\}^{d_i})$, the probability that d_i samples are linearly independent is $1 - O(1/p) - 2^{-\Omega(d_i)}$ [25, 27]. By setting $d_i = \omega(\log \lambda)$, this example satisfies the condition.

3.2 Properties

We show that $\mathsf{SSE1}$ satisfies correctness, consistency, and bounded full security.

Correctness easily follows from the definition. Consistency follows from the pseudorandomness of F.

Theorem 1. $\mathsf{SSE1}$ is consistent provided that F is pseudorandom.

Proof. Suppose that the consistency adversary outputs (w, w') where $w \neq w'$. For $(c_1, c_2) \leftarrow \mathsf{Enc}(\mathsf{K}, w)$ and $(t_1, t_2) \leftarrow \mathsf{Trapdoor}(\mathsf{K}, w')$,

$$c_1^\top t_2 - c_2^\top t_1 = x^\top (F_{\mathsf{K}}(w') - F_{\mathsf{K}}(w))^\top y$$

holds. By the pseudorandomness of F, it is sufficient to show that

$$\Pr\left[x \leftarrow \eta_2, y \leftarrow \eta_1; M \xleftarrow{\$} \mathbb{F}^{d_1 \times d_2} : x^\top M^\top y = 0\right]$$

is negligible. Since $x, y \neq 0$ with an overwhelming probability, the probability that $x^\top M^\top y = 0$ is negligible. □

Bounded full security also follows from the pseudorandomness of F. We prove the following theorem in the next subsection.

Theorem 2. $\mathsf{SSE1}$ is bounded fully secure provided that F is pseudorandom. Namely, for any PPT adversary $\mathcal{A}$ that makes a bounded number of queries, there exists a PPT algorithm $\mathcal{B}$ such that

$$\mathsf{Adv}^{\mathsf{BFull}}_{\mathcal{A},\mathsf{SSE1}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B},F}(\lambda) + \mathsf{negl}(\lambda).$$

3.3 Proof of Theorem 2

Let $\mathcal{O}^b_{\mathsf{K}}$ denote the pair of the oracles $(\mathcal{O}^b_{\mathsf{ct}}(\mathsf{K},\cdot,\cdot), \mathcal{O}^b_{\mathsf{td}}(\mathsf{K},\cdot,\cdot))$. Consider the following pair of stateful oracles $\mathcal{O}^{\mathsf{M}} = (\mathcal{O}^{\mathsf{M}}_{\mathsf{ct}}(\cdot,\cdot), \mathcal{O}^{\mathsf{M}}_{\mathsf{td}}(\cdot,\cdot))$.

1. $\mathsf{List} = \emptyset$ at the beginning.
2. $\mathcal{O}^{\mathsf{M}}_{\mathsf{ct}}(w_0,w_1)$ samples $x \leftarrow \eta_2$. If $((w_0,w_1),M) \in \mathsf{List}$ for some $M \in \mathbb{F}^{d_1\times d_2}$, it returns (x, Mx). Otherwise, it samples $M \xleftarrow{\$} \mathbb{F}^{d_1\times d_2}$ and returns (x, Mx). Then, $((w_0,w_1),M)$ is appended to List.
3. $\mathcal{O}^{\mathsf{M}}_{\mathsf{td}}(w_0,w_1)$ samples $y \leftarrow \eta_1$. If $((w_0,w_1),M) \in \mathsf{List}$ for some $M \in \mathbb{F}^{d_1\times d_2}$, it returns $(y, M^\top y)$. Otherwise, it samples $M \xleftarrow{\$} \mathbb{F}^{d_1\times d_2}$ and returns $(y, M^\top y)$. Then, $((w_0,w_1),M)$ is appended to List.

We prove that any PPT adversary cannot distinguish $\mathcal{O}^0_{\mathsf{K}}$ (or equivalently $\mathcal{O}^1_{\mathsf{K}}$) with $\mathcal{O}^{\mathsf{M}}$ with a non-negligible advantage provided that F is pseudorandom.

Now, we define $\mathcal{O}^0$ by replacing the pseudorandom function in $\mathcal{O}^0_{\mathsf{K}}$ with the randomly chosen function. Let $\mathsf{FList} = \emptyset$ at the beginning. When the oracles compute $F_{\mathsf{K}}(w)$ for the first time, it chooses $M \xleftarrow{\$} \mathbb{F}^{d_1\times d_2}$ and appends (w,M) to FList. After that, the oracles use M instead of $F_{\mathsf{K}}(w)$. Then, for any PPT adversary $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}$ such that

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0_{\mathsf{K}}}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0}(1^\lambda)]\big| \le 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B},F}(\lambda).$$

Here, the description of pp is omitted from the input.

Next, we define the intermediate oracles between $\mathcal{O}^0$ and $\mathcal{O}^{\mathsf{M}}$. $\mathcal{O}^{0,i}$ $(i = 0,1,\ldots,Q)$ sets $\mathsf{List} = \emptyset$ and $\mathsf{FList} = \emptyset$ at the beginning. Then, $\mathcal{O}^{0,i}_{\mathsf{ct}}(\cdot,\cdot)$ behaves as follows:

1. It samples $x \leftarrow \eta_2$.
2. If $((w_0,w_1),M) \in \mathsf{List}$ for some $M \in \mathbb{F}^{d_1\times d_2}$, it returns (x, Mx).
3. If $|\mathsf{List}| < i$, it samples $M \xleftarrow{\$} \mathbb{F}^{d_1\times d_2}$ and returns (x, Mx). Then, $((w_0,w_1),M)$ is appended to List.
4. If $(w_0,M) \in \mathsf{FList}$ for some $M \in \mathbb{F}^{d_1\times d_2}$, it returns (x, Mx).
5. Otherwise, it samples $M \xleftarrow{\$} \mathbb{F}^{d_1\times d_2}$ and returns (x, Mx). Then, (w_0,M) is appended to FList.

$\mathcal{O}^{0,i}_{\mathsf{td}}(\cdot,\cdot)$ are defined in the same way. They sample $y \leftarrow \eta_1$ and return $(y, M^\top y)$. We have $\mathcal{O}^0 = \mathcal{O}^{0,0}$ and $\mathcal{O}^{\mathsf{M}} = \mathcal{O}^{0,Q}$.

We prove that $\mathcal{O}^{0,i}$ and $\mathcal{O}^{0,i+1}$ are computationally indistinguishable. Regarding $\mathcal{O}^{0,i}$, let Q_{i+1} denote the query in which the first element is appended to FList. In the case of $\mathcal{O}^{0,i+1}$, this corresponds to the query in which the $(i+1)$-th element is appended to List. The difference between these oracles is the behaviour when receiving Q_{i+1}. Let $(w_{0,i+1}, w_{1,i+1})$ be the pair of keywords corresponding to Q_{i+1}.

Let E denote the event that $(w_{0,i+1}, w_{1,i+1})$ has not been queried both to the ciphertext oracle and the trapdoor oracle throughout the game. In the case of $\neg E$, queries of the form $(w_{0,i+1},\cdot)$ except $(w_{0,i+1}, w_{1,i+1})$ are not allowed. Thus, seen from the outside, $\mathcal{O}^{0,i}$ and $\mathcal{O}^{0,i+1}$ behave in the same way. It follows that for any PPT adversary $\mathcal{A}$,

$$\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge \neg E] = \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}}(1^\lambda) \wedge \neg E].$$

E can be divided into two cases: E_1 and E_2. E_1 denotes the case that $(w_{0,i+1}, w_{1,i+1})$ has been queried only to the ciphertext oracle. E_2 denotes the case that $(w_{0,i+1}, w_{1,i+1})$ has been queried only to the trapdoor oracle.

By symmetry, we consider the case of E_1. Then, trapdoor queries of the form $(w_{0,i+1},\cdot)$ are not allowed. Let $\mathcal{O}^{0,i+1/3}$ denote the oracle constructed by modifying $\mathcal{O}^{0,i}$ in the following way:

For Q_{i+1} or ciphertext queries of the form $\mathsf{input} = (w_{0,i+1},\cdot)$ after Q_{i+1},

1. If $(\mathsf{input}, M) \in \mathsf{List}$ for some $M \in \mathbb{F}^{d_1 \times d_2}$, it returns (x, Mx) where $x \leftarrow \eta_2$.
2. Otherwise, it returns $(x, u) \leftarrow \eta_2 \times \mathrm{Unif}(\mathbb{F}^{d_1})$.

In addition, let $\mathcal{O}^{0,2/3}$ be the oracle defined by applying the above modification only to Q_{i+1}.

The responses to the ciphertext queries on $(w_{0,i+1}, \cdot)$ that are not in List are summarized in Table 2. Q^+_{i+1} denotes the next such ciphertext query of Q_{i+1}. $\mathsf{List}(\cdot)$ denotes M such that $(\cdot, M) \in \mathsf{List}$, and $\mathsf{FList}(\cdot)$ denotes M such that $(\cdot, M) \in \mathsf{FList}$. "Random" means that the oracle returns a sample from $\eta_2 \times \mathrm{Unif}(\mathbb{F}^{d_1})$.

Table 2: Responses to ciphertext queries on $(w_{0,i+1}, \cdot)$ that are not in List

	Q_{i+1}	Q^+_{i+1}	After Q^+_{i+1}
$\mathcal{O}^i$	uses $\mathsf{FList}(w_{0,i+1}) \xleftarrow{\$} \mathbb{F}^{d_1 \times d_2}$	uses $\mathsf{FList}(w_{0,i+1})$	uses $\mathsf{FList}(w_{0,i+1})$
$\mathcal{O}^{i+1/3}$	random	random	random
$\mathcal{O}^{i+2/3}$	random	uses $\mathsf{FList}(w_{0,i+1}) \xleftarrow{\$} \mathbb{F}^{d_1 \times d_2}$	uses $\mathsf{FList}(w_{0,i+1})$
$\mathcal{O}^{i+1}$	uses $\mathsf{List}(\mathsf{input}) \xleftarrow{\$} \mathbb{F}^{d_1 \times d_2}$	uses $\mathsf{FList}(w_{0,i+1}) \xleftarrow{\$} \mathbb{F}^{d_1 \times d_2}$	uses $\mathsf{FList}(w_{0,i+1})$

In the case of $\mathcal{O}^i$, let $x_1, x_2, \ldots, x_k$ ($k \le Q_{\mathsf{ct}}$) be the set of x sampled in such queries. Since $x_1, x_2, \ldots, x_k$ are linearly independent with an overwhelming probability,

$$([x_1|x_2|\cdots|x_k], \mathsf{FList}(w_{0,i+1}) \cdot [x_1|x_2|\cdots|x_k])$$

is statistically close to the distribution $\eta_2^k \times \mathrm{Unif}(\mathbb{F}^{d_1 \times k})$. Since $\mathsf{FList}(w_{0,i+1})$ is independent of the other view of $\mathcal{A}$, we have

$$|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1/3}}(1^\lambda) \wedge E_1]| = \mathsf{negl}(\lambda).$$

In the same way, we have

$$|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+(t-1)/3}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+t/3}}(1^\lambda) \wedge E_1]| = \mathsf{negl}(\lambda)$$

for each $t = 2, 3$. Since this property holds also for E_2, we have

$$|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}}(1^\lambda)]| = \mathsf{negl}(\lambda).$$

Therefore,

$$\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0_{\mathsf{K}}}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{M}}}(1^\lambda)]\right| \le 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B},F}(\lambda) + \mathsf{negl}(\lambda)$$

and this completes the proof.

4 Pairing-based SSE with Full Security

In this section, we propose an SSE construction based on pairings, which we write as $\mathsf{SSE2}$. In addition, we show that this construction satisfies correctness, consistency, and full security. Full security follows from the $\mathcal{U}_k$-MDDH assumption.

4.1 Construction

Let $\mathcal{G}$ be a bilinear group generation algorithm and k be a positive integer. For simplicity of description, we assume that the group order p can be regarded as a deterministic function of the security parameter λ. Let $\mathcal{W}$ denote the keyword space and $F : \mathcal{K} \times \mathcal{W} \to \mathbb{Z}_p^{k \times k}$ be a pseudorandom function. The construction of $\mathsf{SSE2}$ is as follows:

- $\mathsf{KeyGen}(1^\lambda)$:

1. Runs $\mathbb{G} := (p, G_1, G_2, G_T, g_1, g_2, e) \leftarrow \mathcal{G}(1^\lambda)$.
2. Samples $\mathsf{K} \xleftarrow{\$} \mathcal{K}$.
3. Outputs the public parameter $\mathbb{G}$ and the secret key K.

- $\mathsf{Enc}(\mathsf{K}, w)$: Samples $x \xleftarrow{\$} \mathbb{Z}_p^k$ and outputs $(c_1, c_2) := ([x]_1, [F_\mathsf{K}(w)x]_1)$.
- $\mathsf{Trapdoor}(\mathsf{K}, w)$: Samples $y \xleftarrow{\$} \mathbb{Z}_p^k$ and outputs $(t_1, t_2) := ([y]_2, [F_\mathsf{K}(w)^\top y]_2)$.
- $\mathsf{Test}((t_1, t_2), (c_1, c_2))$: Outputs 1 if $e(c_1, t_2) = e(c_2, t_1)$, otherwise outputs 0.

k determines the underlying assumption of $\mathsf{SSE2}$. The case of $k = 1$ is the most efficient. The underlying assumption U_1-MDDH is the strongest, but still it is weaker than the SXDH assumption.

4.2 Properties

It is easy to see that $\mathsf{SSE2}$ satisfies correctness. In fact, if a ciphertext (c_1, c_2) and a trapdoor (t_1, t_2) are associated with the same keyword w,

$$e(c_1, t_2) = \left[x^\top F_\mathsf{K}(w)^\top y\right]_T = e(c_2, t_1).$$

Consistency follows from the pseudorandomness of F.

Theorem 3. $\mathsf{SSE2}$ is consistent provided that F is pseudorandom.

Proof. Suppose that the consistency adversary outputs (w, w') where $w \neq w'$. For $(c_1, c_2) \leftarrow \mathsf{Enc}(\mathsf{K}, w)$ and $(t_1, t_2) \leftarrow \mathsf{Trapdoor}(\mathsf{K}, w')$,

$$e(c_1, t_2)/e(c_2, t_1) = \left[x^\top (F_\mathsf{K}(w') - F_\mathsf{K}(w))^\top y\right]_T$$

holds. By the pseudorandomness of F, it is sufficient to show that

$$\Pr\left[x, y \xleftarrow{\$} \mathbb{Z}_p^k; M \xleftarrow{\$} \mathbb{Z}_p^{k\times k} : x^\top M y = 0\right]$$

is negligible. Since $x, y \neq 0$ with an overwhelming probability, the probability that $x^\top M y = 0$ is negligible. □

Full security follows from the pseudorandomness of F and the U_k-MDDH assumption. We prove the following theorem in the next subsection.

Theorem 4. $\mathsf{SSE2}$ is fully secure provided that F is pseudorandom and the U_k-MDDH assumption holds. Namely, for any PPT adversary $\mathcal{A}$ that makes at most Q queries, there exist PPT algorithms $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ such that

$$\mathsf{Adv}^{\mathsf{Full}}_{\mathcal{A},\mathsf{SSE2}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B}_1,F}(\lambda) + 6Q\left(\mathsf{Adv}^{U_k\text{-}\mathsf{MDDH}}_{1,\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{U_k\text{-}\mathsf{MDDH}}_{2,\mathcal{B}_3}(\lambda)\right) + \mathsf{negl}(\lambda).$$

4.3 Proof of Theorem 4

The proof flow is the same as that of Theorem 2. Let Q be an upper bound of the number of queries. We define the oracles $\mathcal{O}^0_\mathsf{K}$, $\mathcal{O}^{0,t/3}(t = 0, 1, \ldots, 3Q)$ and the events E, E_1, E_2 in the same way.

First, there exists a PPT algorithm $\mathcal{B}_1$ such that

$$\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0_\mathsf{K}}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0}(1^\lambda)]\right| \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B}_1,F}(\lambda).$$

Then, we evaluate the computational difference between $\mathcal{O}^{0,i}$ and $\mathcal{O}^{0,i+1}$ for $i = 0, 1, \ldots, Q-1$. We have

$$\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge \neg E] = \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}}(1^\lambda) \wedge \neg E],$$

so we consider the case of E_1 by symmetry. By treating $\mathsf{FList}(w_{0,i+1})$ as the k-instance MDDH secret, it follows that there exists a PPT algorithm $\mathcal{B}_{2,1}$ such that

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1/3}}(1^\lambda) \wedge E_1]\big| \leq 2\mathsf{Adv}_{1,\mathcal{B}_{2,1}}^{U_k\text{-}\mathsf{MDDH}_k}(\lambda).$$

Similarly, there exist PPT algorithms $\mathcal{B}_{2,t}$ $(t = 2, 3)$ such that

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+(t-1)/3}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+t/3}}(1^\lambda) \wedge E_1]\big| \leq 2\mathsf{Adv}_{1,\mathcal{B}_{2,t}}^{U_k\text{-}\mathsf{MDDH}_k}(\lambda).$$

Thus, for some PPT algorithm $\mathcal{B}_2$,

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}}(1^\lambda) \wedge E_1]\big| \leq 6\mathsf{Adv}_{1,\mathcal{B}_2}^{U_k\text{-}\mathsf{MDDH}_k}(\lambda)$$

holds. By symmetry, there exists a PPT algorithm $\mathcal{B}_3$ such that

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge E_2] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}}(1^\lambda) \wedge E_2]\big| \leq 6\mathsf{Adv}_{2,\mathcal{B}_3}^{U_k\text{-}\mathsf{MDDH}_k}(\lambda).$$

By Lemma 2, the inequality in Theorem 4 holds.

5 Lattice-based SSE with Full Security

In this section, we propose SSE constructions with full security based on lattices. First, we describe an LWE-based construction, which we write as SSE3-L. Then, we show that the RLWE-based scheme SSE3-R can be constructed in the same way. Finally, we propose an NTRU-based construction SSE4, which is more efficient than the (R)LWE-based schemes.

5.1 Revisiting Coset Sampleability

For constructing SSE, we use an n-coset sampleable distribution η on $\mathbb{Z}_q^k$ that outputs short vectors. Boneh et al. [8] proposed two examples of such distributions.

In the first example, η is the uniform distribution on $\{0,1\}^k$ where $k = n\lceil \log q \rceil$. q is restricted to a power of 2 if q is polynomially bounded. In the second example, η is a discrete Gaussian distribution with the deviation σ where $k = \Omega(n \log q)$ and $\sigma = \Omega(\sqrt{n \log q})$.

In both of these cases, $k \geq n \log q$ holds. We propose a generalized construction of the first example. For an integer $d(\lambda) \geq 2$, we define n-coset sampleable distributions η_d with the dimension $k = dn$ as follows. Let $q_0 = \lceil q^{1/d} \rceil$.

- $\mathsf{MatrixGen}(1^\lambda)$: It outputs

$$M = \left[q_0^{d-1} I_n | q_0^{d-2} I_n | \cdots | I_n\right] \in \mathbb{Z}_q^{n \times k}$$

  and auxiliary data $T = \emptyset$.
- $\mathsf{SamplePre}(z \in \mathbb{Z}_q^n, M, T)$: It treats z as $z \in (-q/2, q/2]^n$ and sets $y_{d-1} = \lfloor z/q_0^{d-1} \rceil$, $z_{d-1} = z - q_0^{d-1} y_{d-1}$. Then, it inductively sets $y_i = \lfloor z_{i+1}/q_0^i \rceil$ and $z_i = z_{i+1} - q_0^i y_i$ for $i = d-2, \ldots, 0$. Finally, it outputs $[y_{d-1}; \cdots ; y_0] \in [-q_0/2, q_0/2]^k$.

Next, we try to apply the non-uniform setting to the RLWE assumption. However, the non-uniform RLWE assumption does not hold if η outputs sufficiently short vectors. This is because if $(a_1, a_1 s + x_1)$ and $(a_2, a_2 s + x_2)$ are non-uniform RLWE instances, $a_1(a_2 s + x_2) - a_2(a_1 s + x_1)$ is relatively short.

To construct non-uniform assumptions that are as weak as RLWE, we introduce the Non-uniform Module-LWE (NMLWE) assumption.

Definition 9. Let $k(\lambda)$ be a positive integer and $\eta(\lambda)$ be a distribution on $\mathcal{R}_q^k$. Let $\chi(\lambda)$ be an error distribution on $\mathcal{R}_q$. For $s \in \mathcal{R}_q^k$, let $\mathcal{O}_s^\eta$ be the oracle that samples $a \leftarrow \eta$, $x \leftarrow \chi$ and outputs $(a, a^\top s + x)$. Let $\mathcal{O}^\eta_\$$ be the oracle that outputs $(a, b) \leftarrow \eta \times \mathrm{Unif}(\mathcal{R}_q)$. We say that the (N, q, k, χ, η)-NMLWE assumption holds if for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{NMLWE}_{(N,q,k,\chi,\eta)}}(\lambda) := \frac{1}{2}\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^\eta_s}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^\eta_\$}(1^\lambda)]\right|$$

is negligible for a randomly chosen secret $s \overset{\$}{\leftarrow} \mathcal{R}_q$.

As in the case of NLWE, the NMLWE problem is as hard as the standard MLWE problem when η achieves some properties. For simplicity of description, we only deal with the problems that are as hard as RLWE. We define $\mathcal{R}_q$-coset sampleability as follows.

Definition 10. A distribution η on $\mathcal{R}_q^k$ is called $\mathcal{R}_q$-coset sampleable if there exist two PPT algorithms $\mathsf{MatrixGen}$ and $\mathsf{SamplePre}$ such that:

- $\mathsf{MatrixGen}(1^\lambda)$ outputs $v \in \mathcal{R}_q^k$ and auxiliary data T.
- $\mathsf{SamplePre}(z \in \mathcal{R}_q, v, T)$ outputs $y \in \mathcal{R}_q^k$ such that $v^\top y = z$. In addition, if z is sampled uniformly at random, the distribution of y is η.

In a similar way to the case of n-coset sampleability, for an integer $k(\lambda) \geq 2$, we can construct $\mathcal{R}_q$-sampleable distributions $\eta_k^{\mathcal{R}}$. Let $q_0 = \lceil q^{1/k} \rceil$.

- $\mathsf{MatrixGen}(1^\lambda)$: It outputs

$$v = (q_0^{k-1}, q_0^{k-2}, \ldots, 1)^\top \in \mathcal{R}_q^k$$

  and auxiliary data $T = \emptyset$.
- $\mathsf{SamplePre}(z \in \mathcal{R}_q, v, T)$: It treats coefficients of z as elements in $(-q/2, q/2]$, and sets $y_{k-1} = \lfloor z/q_0^{k-1} \rceil$, $z_{k-1} = z - q_0^{k-1} y_{k-1}$. Then, it inductively defines $y_i = \lfloor z_{i+1}/q_0^i \rceil$ and $z_i = z_{i+1} - q_0^i y_i$ for $i = k-2, \ldots, 0$. Finally, it outputs $(y_{k-1}, \ldots, y_0)^\top \in \mathcal{R}_q^k$ where each coefficient is in $[-q_0/2, q_0/2]$.

For $\mathcal{R}_q$-coset sampleable distributions, the similar reduction to Lemma 3 holds. The following lemma can be proved in the same way as Lemma 3.

Lemma 4. If η is $\mathcal{R}_q$-coset sampleable, the (N, q, k, χ, η)-NMLWE problem is at least as hard as the (N, q, χ)-RLWE problem. Namely, for any PPT adversary $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}$ such that

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{NMLWE}_{N,m,q,k,\chi,\eta}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{RLWE}_{N,m,q,\chi}}(\lambda).$$

Proof. Let $\mathcal{A}$ be any PPT solver of NMLWE. We construct an RLWE solver $\mathcal{B}$ using $\mathcal{A}$. First, $\mathcal{B}$ runs $(v, T) \leftarrow \mathsf{MatrixGen}(1^\lambda)$ and samples $r \overset{\$}{\leftarrow} \mathcal{R}_q^k$. Suppose that $\mathcal{B}$ has received an RLWE instance $(a, b) \in \mathcal{R}_q^2$. $\mathcal{B}$ runs $a' \leftarrow \mathsf{SamplePre}(a, v, T)$ and sets $b' = b + a'^\top r$. Finally, $\mathcal{B}$ sends (a', b') to $\mathcal{A}$ as an NMLWE instance.

If $b = as + x$ where s is the secret and $x \leftarrow \chi$,

$$b' = (a'^\top v)s + a'^\top r + x = a'^\top (sv + r) + x.$$

Note that $sv + r$ is fixed for all instances. If (a, b) is uniformly random, (a', b) is distributed as $\eta \times \mathrm{Unif}(\mathcal{R}_q)$. Thus, (a', b') is a proper NMLWE instance. □

5.2 LWE-based Construction

Let $m(\lambda), q(\lambda), \kappa_{\mathsf{td}}(\lambda), \kappa_{\mathsf{ct}}(\lambda)$ be positive integers such that $\kappa_{\mathsf{td}}\kappa_{\mathsf{ct}} = \omega(\log \lambda)$. Let $\chi(\lambda)$ be an error distribution on $\mathbb{Z}_q$ and $\eta(\lambda)$ be a distribution that outputs short vectors. Let $\mathcal{W}$ denote the keyword space and $F : \mathcal{K} \times \mathcal{W} \to \mathbb{Z}_q^{m \times m}$ be a pseudorandom function. The construction of $\mathsf{SSE3\text{-}L}$ is as follows:

- $\mathsf{KeyGen}(1^\lambda)$:

1. Samples $\mathsf{K} \overset{\$}{\leftarrow} \mathcal{K}$.
2. Outputs the secret key K.

- $\mathsf{Enc}(\mathsf{K}, w)$:
  1. Samples $c_j \leftarrow \eta$ and $x_j \leftarrow \chi^m$ for $j = 1, \dots, \kappa_{\mathsf{ct}}$.
  2. Computes $d_j = F_{\mathsf{K}}(w)c_j + x_j$ for each j.
  3. Outputs $\{c_j, d_j\}_{j=1}^{\kappa_{\mathsf{ct}}}$.
- $\mathsf{Trapdoor}(\mathsf{K}, w)$:
  1. Samples $t_i \leftarrow \eta$ and $y_i \leftarrow \chi^m$ for $i = 1, \dots, \kappa_{\mathsf{td}}$.
  2. Computes $u_i = F_{\mathsf{K}}(w)^\top t_i + y_i$ for each i.
  3. Outputs $\{t_i, u_i\}_{i=1}^{\kappa_{\mathsf{td}}}$.
- $\mathsf{Test}(\{t_i, u_i\}_{i=1}^{\kappa_{\mathsf{td}}}, \{c_j, d_j\}_{j=1}^{\kappa_{\mathsf{ct}}})$:
  1. Computes $\alpha_{i,j} = c_j^\top u_i - d_j^\top t_i \in \mathbb{Z}_q$ for each $(i, j) \in [1, \kappa_{\mathsf{td}}] \times [1, \kappa_{\mathsf{ct}}]$.
  2. Treats $\alpha_{i,j}$ as an integer in $(-q/2, q/2]$. Outputs 1 if $-\lfloor q/4 \rfloor < \alpha_{i,j} < \lfloor q/4 \rfloor$ for every (i, j), otherwise outputs 0.

5.3 Properties

First, we consider correctness of $\mathsf{SSE3\text{-}L}$. Suppose that $\{t_i, u_i\}_{i=1}^{\kappa_{\mathsf{td}}}$ and $\{c_j, d_j\}_{j=1}^{\kappa_{\mathsf{ct}}}$ are generated from the same keyword. Then, for any (i, j), $|\alpha_{i,j}| = |c_j^\top y_i - x_j^\top t_i|$ holds. This value is relatively small, so this scheme can achieve correctness in appropriate parameter settings.

For example, consider the case that $\eta = \eta_2$ and $m = 2n$. In order that the worst-case to average-case reduction works, let χ be the discrete Gaussian with the deviation $\sigma = \Theta(\sqrt{n})$. Correctness holds if

$$q = \sqrt{2n} q_0 \sigma \cdot \omega(\sqrt{\log n}).$$

Thus, when $q = \omega(n^2 \log n)$, our scheme is correct. As another example, when $\eta = \eta_{\lceil \log q \rceil}$, smaller $q = \omega(n \log n)$ can be used. However, since m is large in this example, η_2 seems to be more attractive in terms of efficiency.

Consistency follows from the pseudorandomness of F and the NLWE assumption.

Theorem 5. $\mathsf{SSE3\text{-}L}$ is consistent provided that F is pseudorandom and the (m, q, χ, η)-NLWE assumption holds.

Proof. By the pseudorandomness of F and the (m, q, χ, η)-NLWE assumption, it is sufficient to show that

$$\begin{aligned}\Pr[c_j \leftarrow \eta, d_j \overset{\$}{\leftarrow} \mathbb{Z}_q^m \text{ for } j = 1, \dots, \kappa_{\mathsf{ct}};\\ t_i \leftarrow \eta, u_i \overset{\$}{\leftarrow} \mathbb{Z}_q^m \text{ for } i = 1, \dots, \kappa_{\mathsf{td}} :\\ 1 \leftarrow \mathsf{Test}(\{t_i, u_i\}_{i=1}^{\kappa_{\mathsf{td}}}, \{c_j, d_j\}_{j=1}^{\kappa_{\mathsf{ct}}})]\end{aligned}$$

is negligible. Let $z_{i,j} \leftarrow \chi$ for each $(i, j) \in [1, \kappa_{\mathsf{td}}] \times [1, \kappa_{\mathsf{ct}}]$. Then, by the (m, q, χ, η)-NLWE assumption and the hybrid argument, $\{c_j^\top u_i + z_{i,j}\}_{i,j}$ is computationally indistinguishable from random. Moreover, since d_j, u_i are independent of $c_j, u_i, z_{i,j}$, $\{c_j^\top u_i - d_j^\top t_i + z_{i,j}\}_{i,j}$ is also computationally indistinguishable from random. Suppose that the output of χ is in $[-E, E]$ with an overwhelming probability. If Test outputs 1, $|c_j^\top u_i - d_j^\top t_i + z_{i,j}| < q/4 + E$ holds for any (i, j). Thus, this probability is bounded above by $(1/2 + 2E/q)^{\kappa_{\mathsf{td}}\kappa_{\mathsf{ct}}} + \mathsf{negl}(\lambda)$. When κ_{td} and κ_{ct} are large enough that $\kappa_{\mathsf{td}}\kappa_{\mathsf{ct}} = \omega(\log \lambda)$, $\mathsf{SSE3\text{-}L}$ is consistent. □

Full security also follows from the pseudorandomness of F and the NLWE assumption. We prove the following theorem in the next subsection.

Theorem 6. SSE3-L is fully secure provided that F is pseudorandom and the (m, q, χ, η)-NLWE assumption holds. Namely, for any PPT adversary $\mathcal{A}$ that makes at most Q queries, there exist PPT algorithms $\mathcal{B}_1, \mathcal{B}_2$ such that

$$\mathsf{Adv}^{\mathsf{Full}}_{\mathcal{A},\mathsf{SSE3\text{-}L}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B}_1,F}(\lambda) + 12mQ \cdot \mathsf{Adv}^{\mathsf{NLWE}_{m,\kappa Q,q,\chi,\eta}}_{\mathcal{B}_2}(\lambda)$$

where $\kappa := \max\{\kappa_{\mathsf{td}}, \kappa_{\mathsf{ct}}\}$.

5.4 Proof of Theorem 6

The proof flow is the same as that of Theorem 2. Let Q be an upper bound of the number of queries. We define the oracles $\mathcal{O}^0_{\mathsf{K}}$, $\mathcal{O}^{0,t/3}(t = 0, 1, \ldots, 3Q)$ and the events E, E_1, E_2 in the same way.

First, there exists a PPT algorithm $\mathcal{B}_1$ such that

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0_{\mathsf{K}}}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0}(1^\lambda)]\big| \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B}_1,F}(\lambda).$$

Then, we evaluate the computational difference between $\mathcal{O}^{0,i}$ and $\mathcal{O}^{0,i+1}$ for $i = 0, 1, \ldots, Q-1$. We have

$$\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge \neg E] = \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}}(1^\lambda) \wedge \neg E],$$

so we consider the case of E_1 by symmetry. By treating $\mathsf{FList}(w_{0,i+1})$ as the m-instance NLWE secret and using the hybrid argument, it follows that there exists a PPT algorithm $\mathcal{B}_{2,1}$ such that

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1/3}}(1^\lambda) \wedge E_1]\big| \leq 2m\mathsf{Adv}^{\mathsf{NLWE}_{m,\kappa_{\mathsf{ct}}Q,q,\chi,\eta}}_{\mathcal{B}_{2,1}}(\lambda).$$

Similarly, there exist PPT algorithms $\mathcal{B}_{2,t}$ $(t = 2, 3)$ such that

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+(t-1)/3}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+t/3}}(1^\lambda) \wedge E_1]\big| \leq 2m\mathsf{Adv}^{\mathsf{NLWE}_{m,\kappa_{\mathsf{ct}}Q,q,\chi,\eta}}_{\mathcal{B}_{2,t}}(\lambda).$$

Thus, for some PPT algorithm $\mathcal{B}_2$,

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}}(1^\lambda) \wedge E_1]\big| \leq 6m\mathsf{Adv}^{\mathsf{NLWE}_{m,\kappa_{\mathsf{ct}}Q,q,\chi,\eta}}_{\mathcal{B}_2}(\lambda)$$

holds. By symmetry, there exists a PPT algorithm $\mathcal{B}'_2$ such that

$$\big|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}}(1^\lambda) \wedge E_2] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}}(1^\lambda) \wedge E_2]\big| \leq 6m\mathsf{Adv}^{\mathsf{NLWE}_{m,\kappa_{\mathsf{td}}Q,q,\chi,\eta}}_{\mathcal{B}'_2}(\lambda).$$

Therefore, the inequality in Theorem 6 holds.

5.5 RLWE-based Construction

We can construct the RLWE-based variant SSE3-R of SSE3-L. Let $k(\lambda)$ be a positive integer (e.g. $k = 2$). Let $\chi(\lambda)$ be an error distribution on $\mathcal{R}_q$ and $\eta(\lambda)$ be a distribution on $\mathcal{R}_q^k$ that outputs short vectors. Let $\mathcal{W}$ denote the keyword space and $F : \mathcal{K} \times \mathcal{W} \to \mathcal{R}_q^{k \times k}$ be a pseudorandom function. The construction of SSE3-R is as follows:

- $\mathsf{KeyGen}(1^\lambda)$:
  1. Samples $\mathsf{K} \xleftarrow{\$} \mathcal{K}$.
  2. Outputs the secret key K.
- $\mathsf{Enc}(\mathsf{K}, w)$:
  1. Samples $c \leftarrow \eta$ and $x \leftarrow \chi^k$.
  2. Computes $d = F_{\mathsf{K}}(w)c + x$.
  3. Outputs (c, d).

- $\mathsf{Trapdoor}(\mathsf{K}, w)$:
  1. Samples $t \leftarrow \eta$ and $y \leftarrow \chi^k$.
  2. Computes $u = F_{\mathsf{K}}(w)^\top t + y$.
  3. Outputs (t, u).
- $\mathsf{Test}((t, u), (c, d))$:
  1. Computes $\alpha = c^\top u - d^\top t \in \mathcal{R}_q$.
  2. Treats the coefficients of α as integers in $(-q/2, q/2]$. Outputs 1 if they are in $(-\lfloor q/4 \rfloor, \lfloor q/4 \rfloor)$ for $\Theta(\lambda)$ coefficients in a fixed position, otherwise outputs 0. Note that $N > \lambda$ holds for practical RLWE parameters.

Correctness, consistency, and full security can be proved in the same way as $\mathsf{SSE3\text{-}L}$.

If (c, d) and (t, u) are generated from the same keyword, $c^\top u - d^\top t = c^\top y - x^\top t$ has relatively small coefficients. For example, when $\eta = \eta_2^{\mathcal{R}}$ and χ is the discrete Gaussian distribution with the deviation σ, correctness holds if

$$q = \sqrt{2N} q_0 \sigma \cdot \omega(\sqrt{\log N}).$$

Consistency follows from the pseudorandomness of F and the (N, q, k, χ, η)-NMLWE assumption. The proof is almost the same as that of Theorem 5. By the NMLWE assumption, $c^\top u - d^\top t + z$ is computationally indistinguishable from random where $(c, d), (t, u) \leftarrow \eta \times \mathrm{Unif}(\mathcal{R}_q)$ and $z \leftarrow \chi$.

Full security follows also from the pseudorandomness of F and the (N, q, k, χ, η)-NMLWE assumption. For any PPT adversary $\mathcal{A}$ of $\mathsf{SSE3\text{-}R}$ that makes at most Q queries, there exist PPT algorithms $\mathcal{B}_1, \mathcal{B}_2$ such that

$$\mathsf{Adv}^{\mathsf{Full}}_{\mathcal{A},\mathsf{SSE3\text{-}R}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B}_1,F}(\lambda) + 12kQ \cdot \mathsf{Adv}^{\mathsf{NMLWE}_{N,Q,q,k,\chi,\eta}}_{\mathcal{B}_2}(\lambda).$$

5.6 NTRU-based Construction

We construct the NTRU-based scheme $\mathsf{SSE4}$. Let $\chi(\lambda)$ be a distribution on $\mathcal{R}_q$ that outputs polnomials with small coefficients. Let $\mathcal{W}$ denote the keyword space and $F : \mathcal{K} \times \mathcal{W} \to \mathcal{R}_q^\times$ be a pseudorandom function. The construction of $\mathsf{SSE4}$ is as follows:

- $\mathsf{KeyGen}(1^\lambda)$:
  1. Samples $\mathsf{K} \xleftarrow{\$} \mathcal{K}$.
  2. Outputs the secret key K.
- $\mathsf{Enc}(\mathsf{K}, w)$:
  1. Samples $a \leftarrow \chi$.
  2. Outputs $c = a \cdot F_{\mathsf{K}}(w)$.
- $\mathsf{Trapdoor}(\mathsf{K}, w)$:
  1. Samples $b \leftarrow \chi$.
  2. Outputs $t = b \cdot F_{\mathsf{K}}(w)^{-1}$.
- $\mathsf{Test}(t, c)$:
  1. Computes $\alpha = c \cdot t \in \mathcal{R}_q$.
  2. Treats the coefficients of α as integers in $(-q/2, q/2]$. Outputs 1 if they are in $(-\lfloor q/4 \rfloor, \lfloor q/4 \rfloor)$ for $\Theta(\lambda)$ coefficients in a fixed position, otherwise outputs 0.

If c and t are generated from the same keyword, $ct = ab$ has relatively small coefficients since $a, b \leftarrow \chi$. For example, when χ is distributed on $\{-1, 0, 1\}$, the negative false does not occur if we set $q > 4N$.

If c and t are generated from different keywords, $(c,t) \in \mathcal{R}_q \times \mathcal{R}_q$ is indistinguishable from uniform by the NTRU assumption. The RLWE assumption ensures that ct is relatively close to uniform, as in the case of $\mathsf{SSE3\text{-}R}$. A drawback of using the RLWE assumption is that N is restricted to be a power of 2. One way to use flexible N is sampling g from $\chi^\times$ in $\mathsf{Trapdoor}$. If $|\mathcal{R}_q^\times|/|\mathcal{R}_q|$ is not negligible, $t \in \mathcal{R}_q^\times$ is indistinguishable from uniform by the NTRU assumption. In this case, $ct \in \mathcal{R}_q$ is indistinguishable from uniform, and consistency follows.

Full security follows from the pseudorandomness of F and the (N,q,χ)-NTRU assumption. For any PPT adversary $\mathcal{A}$ of $\mathsf{SSE4}$ that makes at most Q queries, there exist PPT algorithms $\mathcal{B}_1, \mathcal{B}_2$ such that

$$\mathsf{Adv}^{\mathsf{Full}}_{\mathcal{A},\mathsf{SSE4}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B}_1,F}(\lambda) + 12Q \cdot \mathsf{Adv}^{\mathsf{NTRU}_{N,Q,q,\chi}}_{\mathcal{B}_2}(\lambda).$$

The proof is the same as that of the other constructions, but note that NTRU instances can be converted to "the ciphertext/trapdoor form" in the proof. Let h be an NTRU instance and $h' = h * r$ where $r \xleftarrow{\$} \mathcal{R}_q^\times$. If $h = f^{-1}g$, $h' = (f^{-1}r)g$ where $f^{-1}r$ is uniformly random in $\mathcal{R}_q^\times$. If h is uniformly random, h' is also uniformly random.

6 Simplifying and Verifying Cheng et al. Scheme

In this section, we analyze the Cheng et al. PAEKS scheme [12]. This scheme is essentially the combination of NIKE and SSE, so we focus on the SSE part. It has been unknown whether the SSE part is secure since Li and Boyen [21] pointed out that the original proof is incomplete. We simplify and generalize this scheme. Then, we show that full security can be proved from the bilateral MDDH assumption.

6.1 Simplification of Cheng et al. PAEKS Scheme

First, we describe the algorithms of the Cheng et al. PAEKS [12] scheme. The syntax of PAEKS is given in Appendix A.

- $\mathsf{Setup}(1^\lambda)$:
  1. Runs the symmetric bilinear group generation algorithm $\mathbb{G} = (p, G, G_T, g, e) \leftarrow \mathcal{G}(1^\lambda)$.
  2. Chooses hash functions $H : \{0,1\}^* \to G$ and $\hat{H} : G \to \{0,1\}^\ell$.
  3. Samples $d \xleftarrow{\$} \mathbb{Z}_p$ and sets $h = g^d$.
  4. Outputs the public parameter $\mathsf{pp} = (\mathbb{G}, h, H, \hat{H})$.
- $\mathsf{KeyGen_S}(\mathsf{pp})$: Samples $y \xleftarrow{\$} \mathbb{Z}_p$ and outputs $(\mathsf{pk_S}, \mathsf{sk_S}) = (g^y, y)$.
- $\mathsf{KeyGen_R}(\mathsf{pp})$: Samples $x \xleftarrow{\$} \mathbb{Z}_p$ and outputs $(\mathsf{pk_R}, \mathsf{sk_R}) = (g^x, x)$.
- $\mathsf{Enc}(\mathsf{pk_R}, \mathsf{pk_S}, \mathsf{sk_S}, w)$: Samples $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$ and outputs the ciphertext (C_1, C_2, C_3, C_4) where

$$C_1 = H(\hat{H}(g^{xy}), w, \mathsf{pk_R}, \mathsf{pk_S})^{r_1} \cdot h^{yr_2},\ C_2 = g^{xr_1},\ C_3 = h^{r_2},\ C_4 = g^{r_1}.$$

- $\mathsf{Trapdoor}(\mathsf{pk_R}, \mathsf{pk_S}, \mathsf{sk_R}, w)$: Samples $s_1, s_2 \xleftarrow{\$} \mathbb{Z}_p$ and outputs the trapdoor (T_1, T_2, T_3, T_4) where

$$T_1 = H(\hat{H}(g^{xy}), w, \mathsf{pk_R}, \mathsf{pk_S})^{s_1} \cdot h^{xs_2},\ T_2 = g^{ys_1},\ T_3 = h^{s_2},\ T_4 = g^{s_1}.$$

- $\mathsf{Test}((T_1, T_2, T_3, T_4), (C_1, C_2, C_3, C_4))$: Outputs 1 if $e(C_1, T_4) \cdot e(C_2, T_3) = e(T_1, C_4) \cdot e(T_2, C_3)$, otherwise outputs 0.

In this scheme, H and $\hat{H}$ are used to reduce the security to the Computational Oracle Diffie-Hellman assumption. However, since this reduction is not correct as stated in [21], we replace this part with a pseudorandom function in order to remove the random oracle. Moreover, it is not necessary to use h since the distribution of (h^{yr_2}, h^{r_2}) (resp. (h^{xs_2}, h^{s_2})) is identical to that of (g^{yr_2}, g^{r_2}) (resp. (g^{xs_2}, g^{s_2})).

Now, we propose a simplified version of this scheme, which we write as CSSE. We describe an SSE scheme since SSE can be easily converted to PAEKS by using NIKE to compute the shared secret key [21].

Let $F : \mathcal{K} \times \mathcal{W} \to \mathbb{Z}_p$ be a pseudorandom function. For generality, we use the bilinear map $e : G_1 \times G_2 \to G_T$ where $G_1 = G_2$ does not necessarily hold.

- $\mathsf{KeyGen}(1^\lambda)$:
  1. Runs the bilinear group generation algorithm $\mathbb{G} = (p, G_1, G_2, G_T, g_1, g_2, e) \leftarrow \mathcal{G}(1^\lambda)$.
  2. Samples $x, y \xleftarrow{\$} \mathbb{Z}_p$.
  3. Samples $\mathsf{K} \xleftarrow{\$} \mathcal{K}$.
  4. Outputs the public parameter $\mathbb{G}$ and the secret key $\mathsf{K}' = (\mathsf{K}, x, y)$.
- $\mathsf{Enc}(\mathsf{K}', w)$: Samples $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$ and outputs a ciphertext (C_1, C_2, C_3, C_4) where

$$C_1 = g_1^{F_{\mathsf{K}}(w) r_1 + y r_2},\ C_2 = g_1^{x r_1},\ C_3 = g_1^{r_2},\ C_4 = g_1^{r_1}.$$

- $\mathsf{Trapdoor}(\mathsf{K}', w)$: Samples $s_1, s_2 \xleftarrow{\$} \mathbb{Z}_p$ and outputs a trapdoor (T_1, T_2, T_3, T_4) where

$$T_1 = g_2^{F_{\mathsf{K}}(w) s_1 + x s_2},\ T_2 = g_2^{y s_1},\ T_3 = g_2^{s_2},\ T_4 = g_2^{s_1}.$$

- $\mathsf{Test}((T_1, T_2, T_3, T_4), (C_1, C_2, C_3, C_4))$: Outputs 1 if $e(C_1, T_4) \cdot e(C_2, T_3) = e(C_4, T_1) \cdot e(C_3, T_2)$, otherwise outputs 0.

Note that instead of setting (x, y) as a part of the secret key, we can set $(g_1^x, g_1^y, g_2^x, g_2^y)$ as a part of the public parameter.

Let $G_{\mathsf{K}'} : \mathcal{W} \to \mathbb{Z}_p^{2\times 2}$ denote the function defined by

$$G_{\mathsf{K}'}(w) := \begin{pmatrix} F_{\mathsf{K}}(w) & y \\ x & 0 \end{pmatrix}.$$

Using $G_{\mathsf{K}'}$, ciphertexts and the trapdoors can be written in the similar form to $\mathsf{SSE2}$ based on the U_2-MDDH assumption.

- Ciphertext: $([r]_1, [G_{\mathsf{K}'}(w) r]_1)$ where $r \xleftarrow{\$} \mathbb{Z}_p^2$.
- Trapdoor: $([s]_2, [G_{\mathsf{K}'}(w)^\top s]_2)$ where $s \xleftarrow{\$} \mathbb{Z}_p^2$.

Compared to U_2-MDDH-based $\mathsf{SSE2}$, the ciphertext size and the trapdoor size is the same. The advantage is the improved computational cost due to the smaller range of F. The disadvantage is that the security is reduced to a stronger assumption. We describe the details in the next subsection.

6.2 Security Analysis

Correctness and consistency can be easily proved in the same way as $\mathsf{SSE2}$. We analyze the security of CSSE. In the (bil-)U_2-MDDH assumption, the matrix distribution is $\mathrm{Unif}(\mathbb{Z}_p^{3\times 2})$. In the (bil-)DLin assumption, the matrix distribution is

$$A = \begin{pmatrix} a_1 & 0 \\ 0 & a_2 \\ 1 & 1 \end{pmatrix} \text{ where } a_1, a_2 \xleftarrow{\$} \mathbb{Z}_p\,.$$

Full security of CSSE is reduced to the bil-MDDH assumption such that the matrix distribution is

$$\begin{pmatrix} b_1 & b_2 \\ b_3 & b_4 \\ 0 & 1 \end{pmatrix} \text{ where } b_1, b_2, b_3, b_4 \xleftarrow{\$} \mathbb{Z}_p,$$

which we write as V_2 in this paper. We show that the (bil-)DLin assumption implies the (bil-)V_2-MDDH assumption using the MDDH reduction method [17]. Let L, R be the matrices defined as

$$L = \begin{pmatrix} 1 & 0 & l_1 \\ 0 & 1 & l_2 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{where } l_1, l_2 \xleftarrow{\$} \mathbb{Z}_p, \quad R = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}.$$

For a bil-DLin instance $([A]_1, [z]_1, [A]_2, [z]_2)$, the distribution of LAR is V_2 since

$$LAR = \begin{pmatrix} a_1 & l_1 \\ -a_2 & a_2 + l_2 \\ 0 & 1 \end{pmatrix}.$$

By the invertibility of R, $([LAR]_1, [LAz]_1, [LAR]_2, [LAz]_2)$ is a bil-V_2-MDDH instance. Thus, the (bil-)DLin assumption implies the (bil-)V_2-MDDH assumption.

We give the security proof in the next subsection.

Theorem 7. The modified Cheng et al. SSE scheme CSSE is fully secure provided that F is pseudorandom and the bil-V_2-MDDH assumption holds.

This scheme can be generalized to the constructions based on the bilateral k-Lin assumption by setting $F : \mathcal{K} \times \mathcal{W} \to \mathbb{Z}_p^{(k-1)\times(k-1)}$, $x, y \xleftarrow{\$} \mathbb{Z}_p^k$, and

$$G_{\mathsf{K}'}(w) := \begin{pmatrix} F_{\mathsf{K}}(w) & y \\ x^\top & 0 \end{pmatrix}.$$

6.3 Proof of Theorem 7

Similarly to U_k-MDDH, the following lemma holds. The proof is the same as that of Lemma 1.

Lemma 5. For $r \in \mathbb{Z}_p^2$, let $\mathcal{O}^r$ be an oracle that samples $a \xleftarrow{\$} \mathbb{Z}_p^2$ and outputs $([a]_1, [a^\top r]_1, [a]_2, [a^\top r]_2)$. Let $\mathcal{O}$ be an oracle that samples $a \xleftarrow{\$} \mathbb{Z}_p^2$, $u \xleftarrow{\$} \mathbb{Z}_p$ and outputs $([a]_1, [u]_1, [a]_2, [u]_2)$. If the bil-$V_2$-MDDH assumption holds, for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{bil}\text{-}V_2\text{-}\mathsf{MDDH}}(\lambda) := \frac{1}{2}\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^r}(1^\lambda, [y]_1, [y]_2)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, [y]_1, [y]_2)]\right|$$

is negligible for a randomly chosen secret $r = (x, y)^\top \xleftarrow{\$} \mathbb{Z}_p^2$.

Proof. Using the bil-V_2-MDDH instance $([A]_1, [s]_1, [A]_2, [s]_2)$, the view of $\mathcal{A}$ in this lemma can be simulated. $[y]_1$ and $[y]_2$ are set as the third components of $[s]_1$ and $[s]_2$ respectively. The oracle simulator samples $t \xleftarrow{\$} \mathbb{Z}_p^3$ and outputs $([t^\top A]_1, [t^\top s]_1, [t^\top A]_2, [t^\top s]_2)$. Since the bit length of p is $\Theta(\lambda)$ and two rows of A are uniformly random, $A \in \mathbb{Z}_p^{3\times 2}$ is rank 2 with an overwhelming probability. Thus, it simulates $\mathcal{O}^r$ when $s = Ar$. When s is random, $[A|s] \in \mathbb{Z}_p^{3\times 3}$ is full-rank with an overwhelming probability since two .rows of $[A|s]$ are uniformly random. Thus, it simulates $\mathcal{O}$ in this case. □

Let Q be an upper bound of the number of queries. We define the oracles $\mathcal{O}^0_{\mathsf{K}'}$, $\mathcal{O}^{0,t/3}_{x,y}(t = 0, 1, \ldots, 3Q)$ and the events E, E_1, E_2 in the same way as the proof of Theorem 2. Note that the secret key K' contains x, y in CSSE, so the oracles depend on x, y.

First, there exists a PPT algorithm $\mathcal{B}_1$ such that

$$\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0_{\mathsf{K}'}}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^0_{x,y}}(1^\lambda)]\right| \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B}_1,F}(\lambda).$$

In addition, we have

$$\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}_{x,y}}(1^\lambda) \wedge \neg E] = \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}_{x,y}}(1^\lambda) \wedge \neg E].$$

In the case of E_1, consider treating $(\mathsf{FList}(w_{0,i+1}), y)$ as the secret and constructing a bil-V_2-MDDH solver using the oracle distinguisher of $\mathcal{O}^{0,i}_{x,y}$ and $\mathcal{O}^{0,i+1/3}_{x,y}$. The solver samples $x \xleftarrow{\$} \mathbb{Z}_p$. Since the solver knows $(x, [y]_1, [y]_2)$, it can correctly respond to queries from the distinguisher. Note that in this case, "random" in Table 2 means that the oracle returns $([(r_1, r_2)^\top]_1, [(r_3, r_1 x)^\top]_1)$ where $r_1, r_2, r_3 \xleftarrow{\$} \mathbb{Z}_p$. Thus, it follows that there exists a PPT algorithm $\mathcal{B}_{2,1}$ such that

$$\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}_{x,y}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1/3}_{x,y}}(1^\lambda) \wedge E_1]\right| \leq 2\mathsf{Adv}^{\mathsf{bil}\text{-}V_2\text{-}\mathsf{MDDH}}_{\mathcal{B}_{2,1}}(\lambda).$$

Similarly, there exist PPT algorithms $\mathcal{B}_{2,t}$ $(t = 2, 3)$ such that

$$\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+(t-1)/3}_{x,y}}(1^\lambda) \wedge E_1] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+t/3}_{x,y}}(1^\lambda) \wedge E_1]\right| \leq 2\mathsf{Adv}^{\mathsf{bil}\text{-}V_2\text{-}\mathsf{MDDH}}_{\mathcal{B}_{2,t}}(\lambda).$$

Thus, for some PPT algorithm $\mathcal{B}_2$,

$$\left|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i}_{x,y}}(1^\lambda) \wedge E] - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}^{0,i+1}_{x,y}}(1^\lambda) \wedge E]\right| \leq 12\mathsf{Adv}^{\mathsf{bil}\text{-}V_2\text{-}\mathsf{MDDH}}_{\mathcal{B}_2}(\lambda).$$

holds. Therefore,

$$\mathsf{Adv}^{\mathsf{Full}}_{\mathcal{A},\mathsf{CSSE}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{B}_1,F}(\lambda) + 12Q\mathsf{Adv}^{\mathsf{bil}\text{-}V_2\text{-}\mathsf{MDDH}}_{\mathcal{B}_2}(\lambda)$$

holds and this completes the proof.

7 Comments on Li-Boyen Schemes

In this section, we show that the Li-Boyen pairing-based SSE [21] is not trapdoor-private. Then, we give some comments on the Li-Boyen lattice-based SSE.

7.1 Attack on Li-Boyen Pairing-based Scheme

First, we describe the algorithms of the Li-Boyen pairing-based scheme [21]. Let $F : \mathcal{K} \times \mathcal{W} \to \mathbb{Z}_p$ be a pseudorandom function.

- $\mathsf{KeyGen}(1^\lambda)$:
  1. Runs $\mathbb{G} := (p, G_1, G_2, G_T, g_1, g_2, e) \leftarrow \mathcal{G}(1^\lambda)$.
  2. Samples $z \xleftarrow{\$} \mathbb{Z}_p$ and sets $u_1 = g_1^z$, $u_2 = g_2^z$.
  3. Samples $h_2 \xleftarrow{\$} G_2$.
  4. Samples $\mathsf{K} \xleftarrow{\$} \mathcal{K}$.
  5. Outputs the public parameter $(\mathbb{G}, u_1, u_2, h_2)$ and the secret key K.
- $\mathsf{Enc}(\mathsf{K}, w)$: Samples $s \xleftarrow{\$} \mathbb{Z}_p$ and outputs $(c_1, c_2, c_3) := \left(e(g_1, h_2)^{F_{\mathsf{K}}(w)\cdot s}, u_1^s, g_1^s\right)$.
- $\mathsf{Trapdoor}(\mathsf{K}, w)$: Samples $r \xleftarrow{\$} \mathbb{Z}_p$ and outputs $(t_1, t_2) := \left(h_2^{F_{\mathsf{K}}(w)} u_2^r, g_2^r\right)$.
- $\mathsf{Test}((t_1, t_2), (c_1, c_2, c_3))$: Outputs 1 if $e(c_3, t_1) = c_1 \cdot e(c_2, t_2)$, otherwise outputs 0.

They claimed that its ciphertext privacy can be reduced to the DBDH assumption, and its trapdoor privacy can be reduced to the DDH assumption on G_1 (See Appendix B for the definitions of ciphertext/trapdoor privacy). However, their proof for trapdoor privacy uses an incorrect DDH definition, and trapdoor privacy can be broken in the following way.

Theorem 8. The Li-Boyen pairing-based SSE does not satisfy trapdoor privacy.

Proof. The adversary chooses arbitrary challenge keywords w_0^*, w_1^* ($w_0^* \neq w_1^*$) and receives a challenge trapdoor $(t_1^*, t_2^*) = \left(h_2^{F_{\mathsf{K}}(w_b^*)} u_2^{r^*}, g_2^{r^*}\right)$ where $b \xleftarrow{\$} \{0,1\}$. In addition, the adversary receives a trapdoor $(t_1, t_2) = \left(h_2^{F_{\mathsf{K}}(w_0^*)} u_2^{r}, g_2^{r}\right)$ of w_0^* and a ciphertext $(c_1, c_2, c_3) = \left(e(g_1, h_2)^{F_{\mathsf{K}}(w)\cdot s}, u_1^s, g_1^s\right)$ of an arbitrary keyword w ($w \notin \{w_0^*, w_1^*\}$). Then, we have $e(c_3, t_1^*)/e(c_2, t_2^*) = e(g_1, h_2)^{F_{\mathsf{K}}(w_b^*)\cdot s}$ and $e(c_3, t_1)/e(c_2, t_2) = e(g_1, h_2)^{F_{\mathsf{K}}(w_0^*)\cdot s}$. By checking whether they are equal, the adversary can guess whether $b = 0$ with an overwhelming probability. □

In this attack, (u_1, g_1) can be used instead of (c_2, c_3). However, we used (c_2, c_3) to indicate that keeping (u_1, g_1) as secret in this scheme does not affect the proof.

7.2 On Li-Boyen Lattice-based Scheme

Li and Boyen have proposed a trapdoor-private SSE framework from GPV sampler [18]. They gave a concrete instantiation based on RLWE. Note that they set q as prime, but the security proof uses a kind of regularity lemma for $q = 3^k$ [15]. In the case that q is prime, the regularity lemma by Lyubashevsky et al. [24] can be used (with a little modification in the construction).

Since the algorithms of the LWE-based instantiation and the NTRU-based instantiation are not concretely described in [21], we give some additional explanations about Table1 here. For the LWE-based instantiation, one needs to generate multiple pairs of the vectors like our construction to ensure consistency. We used κ_{ct} and κ_{td} to indicate that in Table1. For the NTRU-based instantiation, we give the concrete description in Appendix C. It is claimed in [21] that ciphertexts and trapdoors are in $\mathcal{R}_q^3$, but Appendix C shows that they are actually in $\mathcal{R}_q^2$.

8 Conclusion

In this paper, we proposed four types of fully secure SSE schemes: the bounded construction from PRFs, the pairing-based construction, the (R)LWE-based construction, and the NTRU-based construction. They are more efficient than the existing SSE schemes with trapdoor privacy. We note that our SSE schemes can be converted to efficient PAEKS schemes with full CI security and full TI security by using the generic construction [21]. In addition, we gave a security proof for the Cheng et al. scheme [12], the original proof for which was shown to be incorrect. We also proved that the Li-Boyen pairing-based scheme [21] does not achieve the trapdoor privacy property that they claim it does.

References

[1] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi, "Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions," CRYPTO 2005, pages 205-222, 2005.

[2] Michel Abdalla, Dario Catalano, Dario Fiore, Romain Gay, and Bogdan Ursu, "Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings," CRYPTO 2018, pages 597-627, 2018.

[3] Shashank Agrawal, Shweta Agrawal, Saikrishna Badrinarayanan, Abishek Kumarasubramanian, Manoj Prabhakaran, and Amit Sahai, " On the practical security of inner product functional encryption," PKC 2015, pages 777-798, 2015.

[4] Shweta Agrawal and Shota Yamada, “CP-ABE for circuits (and more) in the symmetric key setting,” TCC 2020, pages 117-148, 2020.

[5] Joël Alwen and Chris Peikert, “Generating shorter bases for hard random lattices,” Theory of Computing Systems 48, pages 535-553, 2011.

[6] Allison Bishop, Abhishek Jain, and Lucas Kowalczyk, “Function-hiding inner product encryption,” ASIACRYPT 2015, pages 470-491, 2015.

[7] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano, “Public key encryption with keyword search,” EUROCRYPT 2004, pages 506-522, 2004.

[8] Dan Boneh, Kevin Lewi, Hart Montgomery, and Ananth Raghunathan, “Key homomorphic PRFs and their applications,” CRYPTO 2013, pages 410-428, 2013.

[9] Zvika Brakerski and Gil Segev, “Function-private functional encryption in the private-key setting,” Journal of Cryptology 31, pages 202-225, 2018.

[10] Jin Wook Byun, Hyun Suk Rhee, Hyun-A Park, and Dong Hoon Lee, “Off-line keyword guessing attacks on recent keyword search schemes over encrypted data,” SDM 2006, pages 75-83, 2006.

[11] Leixiao Cheng and Fei Meng, “Public key authenticated encryption with keyword search from LWE,” ESORICS 2022, pages 303-324, 2022.

[12] Leixiao Cheng, Jing Qin, Feng Feng, Fei Meng, “Security-enhanced public-key authenticated searchable encryption,” Information Sciences 647:119454, 2023.

[13] Pratish Datta, Ratna Dutta, and Sourav Mukhopadhyay, “Functional encryption for inner product with full function privacy,” PKC 2016, pages 164-195, 2016.

[14] Léo Ducas, Vadim Lyubashevsky, and Thomas Prest, “Efficient identity-based encryption over NTRU lattices,” ASIACRYPT 2014, pages 22-41, 2014.

[15] Léo Ducas and Daniele Micciancio, “Improved short lattice signatures in the standard model,” CRYPTO 2014, pages 335-352, 2014.

[16] Keita Emura, “Generic construction of public-key authenticated encryption with keyword search revisited: stronger security and efficient construction,” APKC 2022, pages 39-49, 2022.

[17] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar, “An algebraic framework for Diffie–Hellman assumptions,” Journal of Cryptology 30, pages 242-288, 2017.

[18] Craig Gentry, Chris Peikert, Vinod Vaikuntanathan, “Trapdoors for hard lattices and new cryptographic constructions,” ACM STOC 2008, pages 197-206, 2008.

[19] Qiong Huang and Hongbo Li, “An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks,” Information Sciences 403, pages 1-14, 2017.

[20] Sam Kim, Kevin Lewi, Avradip Mandal, Hart Montgomery, Arnab Roy, and David J. Wu, “Function-hiding inner product encryption is practical,” SCN 2018, pages 544-562, 2018.

[21] Qinyi Li and Xavier Boyen, “Public-key authenticated encryption with keyword search made easy,” IACR Communications in Cryptology, vol. 1, no. 2, 2024.

[22] Huijia Lin, “Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs,” CRYPTO 2017, pages 599-629, 2017.

[23] Zi-Yuan Liu, Yi-Fan Tseng, Raylin Tso, Masahiro Mambo, Yu-Chi Chen, “Public-key authenticated encryption with keyword search: Cryptanalysis, enhanced security, and quantum-resistant instantiation,” ASIACCS 2022, pages 423-436, 2022.

[24] Vadim Lyubashevsky, Chris Peikert, and Oded Regev, “A toolkit for ring-LWE cryptography,” EUROCRYPT 2013, pages 35-54, 2013.

[25] Kenneth Maples, “Singularity of random matrices over finite fields,” arXiv preprint arXiv:1012.2372, 2010.

[26] Daniele Micciancio and Chris Peikert, “Trapdoors for lattices: Simpler, tighter, faster, smaller,” EUROCRYPT 2012, pages 700-718, 2012.

[27] Hoi H. Nguyen and Elliot Paquette, “Surjectivity of near-square random matrices,” Combinatorics, Probability and Computing 29.2, pages 267-292, 2020.

[28] Mahnaz Noroozi and Ziba Eslami, “Public key authenticated encryption with keyword search: revisited,” IET Information Security 13(4), pages 336-342, 2019.

[29] Baodong Qin, Yu Chen, Qiong Huang, Ximeng Liu, Dong Zheng, “Public-key authenticated encryption with keyword search revisited: Security model and constructions,” Information Sciences 516, pages 515-528, 2020.

[30] Baodong Qin, Hui Cui, Xiaokun Zheng, and Dong Zheng, “Improved security model for public-key authenticated encryption with keyword search,” ProvSec 2021, pages 19-38, 2021.

[31] Oded Regev, “On lattices, learning with errors, random ear codes, and cryptography,” Journal of the ACM 56(6), pages 1-40, 2009.

[32] Emily Shen, Elaine Shi, and Brent Waters, “Predicate privacy in encryption systems,” TCC 2009, pages 457-473, 2009.

[33] Dawn Xiaodong Song, David Wagner, and Adrian Perrig, “Practical techniques for searches on encrypted data,” IEEE S&P 2000, pages 44-55, 2000.

[34] Junichi Tomida, Masayuki Abe, and Tatsuaki Okamoto, “Efficient functional encryption for inner-product values with full-hiding security,” ISC 2016, pages 408-425, 2016.

A Public key Authenticated Encryption with Keyword Search (PAEKS)

We describe the syntax of Public key Authenticated Encryption with Keyword Search (PAEKS). PAEKS consists of the following six PPT algorithms.

- $\mathsf{Setup}(1^\lambda)$: Given a security parameter λ, it outputs a public parameter pp.
- $\mathsf{KeyGen}_\mathsf{S}(\mathsf{pp})$: Given a public parameter pp, it outputs a sender's public key pk_S and a sender's secret key sk_S.
- $\mathsf{KeyGen}_\mathsf{R}(\mathsf{pp})$: Given a public parameter pp, it outputs a receiver's public key pk_R and a receiver's secret key sk_R.
- $\mathsf{Enc}(\mathsf{pp}, \mathsf{pk}_\mathsf{R}, \mathsf{pk}_\mathsf{S}, \mathsf{sk}_\mathsf{S}, w)$: Given a public parameter pp, a receiver's public key pk_R, a sender's public key pk_S, a sender's secret key sk_S, and a keyword w, it outputs a ciphertext C.
- $\mathsf{Trapdoor}(\mathsf{pp}, \mathsf{pk}_\mathsf{R}, \mathsf{pk}_\mathsf{S}, \mathsf{sk}_\mathsf{R}, w)$: Given a public parameter pp, a receiver's public key pk_R, a sender's public key pk_S, a receiver's secret key sk_R, and a keyword w, it outputs a trapdoor T.
- $\mathsf{Test}(\mathsf{pp}, T, C)$: Given a public parameter pp, a trapdoor T, and a ciphertext C, it outputs a bit $b \in \{0, 1\}$.

In the generic construction of PAEKS by Li and Boyen [21], Enc and $\mathsf{Trapdoor}$ compute the same secret value K using NIKE. Then, ciphertexts and trapdoors are generated by running $\mathsf{SSE.Enc}(\mathsf{K}, w)$ and $\mathsf{SSE.Trapdoor}(\mathsf{K}, w)$ respectively.

B Ciphertext/Trapdoor Privacy of SSE

We describe the definitions of ciphertext privacy and trapdoor privacy. Note that full security implies both of them.

Let $\mathcal{O}_{\mathsf{ct}}(\mathsf{K}, \cdot)$ be the oracle that outputs $C \leftarrow \mathsf{Enc}(\mathsf{K}, w)$ given a keyword w as input. Let $\mathcal{O}_{\mathsf{td}}(\mathsf{K}, \cdot)$ be the oracle that outputs $T \leftarrow \mathsf{Trapdoor}(\mathsf{K}, w)$ given a keyword w as input.

Definition 11 (Ciphertext Privacy)**.** We say that an SSE scheme is ciphertext-private if for any PPT algorithm $\mathcal{A}$, the advantage

$$\Big| \Pr[(\mathsf{pp}, \mathsf{K}) \leftarrow \mathsf{KeyGen}(1^\lambda); (w_0^*, w_1^*, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{ct}}(\mathsf{K},\cdot), \mathcal{O}_{\mathsf{td}}(\mathsf{K},\cdot)}(\mathsf{find}, \mathsf{pp});$$
$$b \xleftarrow{\$} \{0,1\}; C^* \leftarrow \mathsf{Enc}(\mathsf{K}, w_b^*); b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{ct}}(\mathsf{K},\cdot), \mathcal{O}_{\mathsf{td}}(\mathsf{K},\cdot)}(\mathsf{guess}, \mathsf{st}, C^*) : b = b'] - \frac{1}{2} \Big|$$

is negligible in λ. The challenge keyword w_0^*, w_1^* cannot be input to $\mathcal{O}_{\mathsf{td}}(\mathsf{K}, \cdot)$.

Definition 12 (Trapdoor Privacy)**.** We say that an SSE scheme is trapdoor-private if for any PPT algorithm $\mathcal{A}$, the advantage

$$\Big| \Pr[(\mathsf{pp}, \mathsf{K}) \leftarrow \mathsf{KeyGen}(1^\lambda); (w_0^*, w_1^*, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{ct}}(\mathsf{K},\cdot), \mathcal{O}_{\mathsf{td}}(\mathsf{K},\cdot)}(\mathsf{find}, \mathsf{pp});$$
$$b \xleftarrow{\$} \{0,1\}; T^* \leftarrow \mathsf{Trapdoor}(\mathsf{K}, w_b^*); b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{ct}}(\mathsf{K},\cdot), \mathcal{O}_{\mathsf{td}}(\mathsf{K},\cdot)}(\mathsf{guess}, \mathsf{st}, T^*) : b = b'] - \frac{1}{2} \Big|$$

is negligible in λ. The challenge keyword w_0^*, w_1^* cannot be input to $\mathcal{O}_{\mathsf{ct}}(\mathsf{K}, \cdot)$.

C Description of Li-Boyen NTRU-based Scheme

We describe the concrete construction of the Li-Boyen NTRU-based scheme. Let $\mathsf{DLP.KeyGen}$ denote the key generation algorithm of Ducas et al. IBE (DLP-IBE) [14] and $\mathsf{SampleD}$ denote the preimage sampler used in DLP-IBE. Let $\chi(\lambda)$ be an error distribution on $\mathcal{R}_q$, $\sigma(\lambda), \beta(\lambda)$ be positive real parameters, and $F : \mathcal{K} \times (\mathcal{W} \times \{0,1\}) \rightarrow \{0,1\}^\ell$ be a pseudorandom function.

- $\mathsf{KeyGen}(1^\lambda)$:
  1. Samples $u \xleftarrow{\$} \mathcal{R}_q$.
  2. Samples $\mathsf{K} \xleftarrow{\$} \mathcal{K}$.
  3. Outputs the public parameter u and the secret key K.
- $\mathsf{Enc}(\mathsf{K}, w)$:
  1. Computes $\mathsf{rnd}_0 = F_{\mathsf{K}}(w, 0), \mathsf{rnd}_1 = F_{\mathsf{K}}(w, 1)$.
  2. Runs $(h, B) \leftarrow \mathsf{DLP.KeyGen}(1^\lambda; \mathsf{rnd}_0)$.
  3. Runs $(h', B') \leftarrow \mathsf{DLP.KeyGen}(1^\lambda; \mathsf{rnd}_1)$.
  4. Samples $s, x, y \leftarrow \chi$.
  5. Runs $(d_1, d_2) \leftarrow \mathsf{SampleD}(B', us + x, \sigma)$.
     (Then $d_1 + h'd_2 = us + x$ holds.)
  6. Computes $c \leftarrow hs + y$.
  7. Outputs the ciphertext $(c, d_2) \in \mathcal{R}_q^2$.
- $\mathsf{Trapdoor}(\mathsf{K}, w)$:
  1. Computes $\mathsf{rnd}_0 = F_{\mathsf{K}}(w, 0), \mathsf{rnd}_1 = F_{\mathsf{K}}(w, 1)$.
  2. Runs $(h, B) \leftarrow \mathsf{DLP.KeyGen}(1^\lambda; \mathsf{rnd}_0)$.
  3. Runs $(h', B') \leftarrow \mathsf{DLP.KeyGen}(1^\lambda; \mathsf{rnd}_1)$.

4. Samples $s', x', y' \leftarrow \chi$.
5. Runs $(u_1, u_2) \leftarrow \mathsf{SampleD}(B, us' + x', \sigma)$.
   (Then $u_1 + hu_2 = us' + x'$ holds.)
6. Computes $t \leftarrow h's' + y'$.
7. Outputs the trapdoor $(t, u_2) \in \mathcal{R}_q^2$.

- $\mathsf{Test}((t, u_2), (c, d_2))$: Outputs 1 if $\|td_2 - u_2c\| \leq \beta$, otherwise outputs 0.