cryptography

MDPI

Article

A Searchable Encryption Scheme Based on CRYSTALS-Dilithium

Minghui Zheng [1,2], Anqi Xiao [1,*], Shicheng Huang [1] and Deju Kong [1,2]

[1] School of Intelligent Science and Engineering, Hubei Minzu University, 39 Xueyuan Road, Enshi 445000, China; mhzheng3@163.com (M.Z.); 15072601235@163.com (S.H.); kong126@outlook.com (D.K.)
[2] School of Cyberspace Security, Sichuan University, Chengdu 610065, China
* Correspondence: xaq404300657@163.com

Abstract

With the advancement in quantum computing technology, the number theory-based hard problems underlying traditional searchable encryption algorithms are now vulnerable to efficient quantum attacks. To address this challenge, this paper proposes Dilithium-PAEKS (Dilithium-Public Authenticated Encryption with Keyword Search), a searchable encryption scheme based on the post-quantum cryptographic algorithm CRYSTALS-Dilithium. By transforming the verification relationship of digital signatures into a matching relationship between trapdoors and ciphertexts, the scheme not only meets the functional requirements of searchable encryption but also demonstrates quantum resistance. The implementation enhances algorithm efficiency through keyword-based signatures and dynamic matching testing mechanisms. The security of the scheme is defined by the MLWE and MSIS hard problems, with proofs of keyword ciphertext indistinguishability and trapdoor indistinguishability under the random oracle model. Additionally, the scheme provides strong resistance against both outside and insider keyword guessing attacks through sender–receiver binding mechanisms and trapdoor indistinguishability properties. Experimental results show that, compared to the post-quantum schemes CP-Absel and LB-FSSE, the proposed scheme demonstrates superior overall computational efficiency while maintaining stronger quantum resistance than the traditional scheme SM9-PAEKS.

Keywords: searchable encryption; CRYSTALS-Dilithium; lattice-based cryptography; random oracle

1. Introduction

Searchable encryption technology allows users to perform keyword searches directly on encrypted data without the need for decryption. It ensures data privacy while enabling efficient retrieval of ciphertext, and is widely used in privacy-sensitive scenarios such as cloud storage and medical data sharing. However, with the continuous development of quantum computers and the advent of quantum algorithms such as Shor's [1] and Grover's [2], the number theory-based hard problems underlying traditional searchable encryption algorithms are now vulnerable to efficient quantum attacks [3]. This makes it imperative to develop searchable encryption schemes that can resist quantum threats while maintaining practical efficiency.

Existing searchable encryption schemes can be broadly categorized into classical constructions and post-quantum constructions. Among classical approaches, Song et al. [4] first proposed the concept of Symmetric Searchable Encryption (SSE) in 2000. Boneh et al. [5]

Academic Editor: Josef Pieprzyk

Received: 12 February 2026
Revised: 23 March 2026
Accepted: 25 March 2026
Published: 27 March 2026

subsequently pioneered Public Key Encryption with Keyword Search (PEKS) in 2004 by integrating searchable encryption with public key cryptography, but the scheme suffered from dependence on secure channels and vulnerability to keyword guessing attacks. To address these issues, Xu et al. [6] and Li et al. [7] proposed schemes based on composite order bilinear groups, and Deng et al. [8] further improved trapdoor indistinguishability under the Decisional Bilinear Diffie–Hellman (DBDH) assumption. Nevertheless, bilinear pairing-based schemes generally incur high computational overhead. To improve efficiency, Cui et al. [9] proposed a multi-keyword searchable encryption scheme based on Elliptic Curve Cryptography (ECC), and Zhang et al. [10] and Pu et al. [11] proposed schemes based on SM9, further enriching the application of domestic cryptographic algorithms in searchable encryption. However, all the above schemes rely on traditional number theory hard problems and are therefore vulnerable to quantum attacks.

In the post-quantum domain, lattice-based cryptography has been introduced into searchable encryption. Zhang et al. [12] proposed an adaptive hierarchical searchable encryption scheme based on error learning, suitable for dynamic encrypted databases and cloud healthcare systems, but carrying the risk of keyword leakage. Liu et al. [13] proposed a traceable and revocable lattice-based attribute encryption scheme based on the CLWE problem, though it incurs high cloud storage costs. Yu et al. [14] combined attribute signing and searchable encryption for the first time; the scheme demonstrates good quantum resistance, but keyword encryption is time-consuming due to attribute signing. Varri et al. [15] proposed a lattice-based Ciphertext-Policy Attribute-based Searchable Encryption scheme (CP-ABSEL) under the LWE hardness assumption, achieving fine-grained access control and multi-keyword search, but its implementation may incur significant computational overhead due to operations like short basis generation. Islam et al. [16] designed an efficient forward-secure lattice-based searchable encryption scheme leveraging blockchain technology, which demonstrates resilience against insider keyword guessing attacks (IKGAs), but the integration of blockchain introduces additional overheads, including transaction latency and gas fees. In summary, existing post-quantum searchable encryption schemes still face trade-offs among computational efficiency, storage overhead, and comprehensive security guarantees.

To address the above limitations, this paper proposes Dilithium-PAEKS (Dilithium-Public Authenticated Encryption with Keyword Search), a searchable encryption scheme based on the NIST-standardized post-quantum digital signature algorithm CRYSTALS-Dilithium [17,18]. The main contributions of this paper are as follows: (1) We propose a novel PAEKS construction based on CRYSTALS-Dilithium, whose security is formally reducible to two lattice-based hard problems: modular learning with error (MLWE) [19] and modular short integer solution (MSIS) [20], thereby providing demonstrable quantum resistance. (2) The scheme introduces keyword-bound signatures and dynamic matching test mechanisms to achieve efficient ciphertext retrieval while ensuring security, and we provide formal proofs of keyword ciphertext indistinguishability (IND-CKA) and trapdoor indistinguishability (TIK) under the random oracle model. (3) Through sender–receiver binding mechanisms and trapdoor indistinguishability properties, the scheme provides provable resistance against both outside keyword guessing attacks (OKGAs) and the more challenging insider keyword guessing attacks (IKGAs), ensuring security even when the cloud server is semi-trusted or malicious.

2. Design of Dilithium-PAEKS Scheme

This section introduces the parameter, basic concepts, the security assumptions, and the detailed design of the Dilithium-PAEKS scheme, and analyzes the correctness of the design.The detailed descriptions of the parameters used in this paper are listed in Table 1.

Table 1. List of parameters used in this paper.

Parameters and Functions	Meaning
λ	security parameter, default value
R_q	mod q polynomial ring $\mathbb{Z}_q[x]/(x^g+1)$, $q = 2^{23} - 2^{13} + 1 = 8380417$
ℓ, κ, k	Vector dimension parameter, $k = 4$, $\ell = 4$
$\gamma 1, \gamma 2, \beta, d, \omega$	reject sampling and compression parameters, $\gamma_1 = 2^{17}, \gamma_2 = \lfloor(\gamma_1 - \beta)/2\rfloor$, $2^d = 8192$ ($d = 13$)
$\|\cdot\|_\infty$	Maximum absolute value of polynomial coefficients
CRH	collision-resistant hash function
H	Challenge generation function : $\{0,1\}^* \to B_{60}$
$ExpandMask$	Determines the random value for generating the signature scheme
$ExpandA$	$\{0,1\}^{256} \to R_q^{k\times\ell}$ Expand the matrix and output it as an NTT field representation
$Power2Round_q$	Separate high and low bits of data
$Decompose_q$	Different methods of high–low separation
$MakeHint_q$	Show hint
$UseHint_q$	restoring the separated higher-order bit
$HighBits_q$	Extract the first part of the value in the higher-order bit
$LowBits_q$	Extract the second part of the value in the higher-order bit
pk_s, sk_s	Sender's public key and private key pair
pk_r, sk_r	Receiver's public key and private key pair
C_w	Keyword ciphertext, consisting of (z, c, w_1)
T_w	Keyword trapdoor, consisting of $(z', c', \mu w)$
z, z'	Vectors used in ciphertext and trapdoor respectively, typically derived from random commitments in the signature process
c, c'	Challenge values used in ciphertext and trapdoor respectively, gener-ated by a hash function
w_1	Part of the ciphertext, possibly obtained by extracting low bits from the intermediate variable w
μw, $\mu w'$	Binding hash value ensuring keyword consistency, computed as $\mu w = CRH(tr\|\|w\|\|\rho_r) = CRH(tr\|\|w\|\|\rho_s)$
A_s	Sender's public key matrix (in Dilithium, the public key includes matrix A and vector t)
$t_1(S)$	High bits of the sender's public key, derived from the decomposition of $t = As_1 + s_2$
$s_1(S)$, $s_2(S)$	Two components of the sender's private key, namely the main secret vector and the noise term

2.1. Basic Concepts and Formal Definitions

This section formally defines the core primitives of searchable encryption and the key security properties involved in the scheme, and clarifies the design boundary of the proposed Dilithium-PAEKS, laying a theoretical foundation for subsequent scheme design and security proof.

Definition 1 (**Symmetric Searchable Encryption** (SSE)). *A cryptographic primitive that supports keyword search on symmetrically encrypted ciphertext without decryption. It realizes ciphertext retrieval under a symmetric cryptosystem, with core requirements of ciphertext retrievability and data privacy, but is only applicable to peer-to-peer data sharing scenarios.*

Definition 2 (**Public Key Encryption with Keyword Search** (PEKS)). *A public-key extension of SSE, which uses the receiver's public key for encryption and private key for trapdoor generation, solving ciphertext search in asymmetric cryptosystems. Its core defects are reliance on secure*

channels for trapdoor transmission and vulnerability to keyword guessing attacks (KGA), and it lacks sender identity authentication.

Definition 3 (**Public Authenticated Encryption with Keyword Search** (PAEKS)). *An enhanced primitive of PEKS with the full name Public Authenticated Encryption with Keyword Search, which integrates sender identity authentication into the PEKS framework. It solves the defects of secure channel dependence and weak anti-KGA ability of PEKS, and supports keyword search on public key ciphertext in open channels.*

The proposed Dilithium-PAEKS is a lattice-based PAEKS scheme constructed on CRYSTALS-Dilithium, with the core boundary of post-quantum security + no secure channel + sender–receiver dual authentication + keyword search on ciphertext, adapting to post-quantum privacy protection scenarios such as cloud storage.

2.2. Detailed Design of the Scheme

Dilithium-PAEKS consists of five polynomial-time algorithms: (1) the initialization algorithm takes system parameters as input and generate public parameters; (2) the receiver key generation algorithm produces receiver public–private key pairs from specified inputs; (3) the sender key generation algorithm creates sender public–private key pairs based on given inputs; (4) the keyword encryption algorithm generates keyword trapdoors using keywords, sender public keys, receiver private keys, and system parameters as inputs; (5) the matching test algorithm verifies keyword consistency between keyword ciphertexts, keyword trapdoors, and system parameters. The detailed computational procedures for each algorithm are outlined below.

(1) Initialization algorithm $Setup(\lambda \rightarrow params)$

Step 1: The system inputs the security parameter λ and publishes the set of public system parameters $params = \{\lambda, n, q, \ell, \kappa, k, \gamma_1, \gamma_2, \beta, d, \omega, CRH, H\}$, $CRH : \{0,1\}^* \rightarrow \{0,1\}^{384}$.

(2) Key generation for the recipient $KeyGen_R((params \rightarrow (sk_R, pk_R))$

Step 1: Generate a key vector by uniformly sampling the matrix $(s_1, s_2) \leftarrow S_\eta^\ell \times S_\eta^\kappa$.

Step 2: Generate a matrix using an expansion function $A_R \in R_q^{k \times \ell} := ExpendA(\rho_R)$.

Step 3: A_R is the matrix generated by the function $ExpendA$, S_1 and S_2 are the minimal vectors randomly sampled from the set $S_q^\ell \times S_\eta^\kappa$, the public key d is indistinguishable from a random vector, and thus $d := A_R s_1 + s_2$ is an instance of MLWE.

Step 4: Compress the public key using a high–low bit separation function $(t_1, t_0) := Power2Round_q(t, d)$.

Step 5: Generate the recipient's identity identifier (tr_R) using a hash function $CRH(\rho_R || t_1)$.

Step 6: Return (sk_R, pk_R), $sk_R := (\rho_R, K_R, tr_R, s_1, s_2, t_0)$, $pk_R := (\rho_R, t_1)$.

(3) Sender key generation $KeyGen_S((params \rightarrow (sk_S, pk_S))$

The process is the same as the recipient key generation, $KeyGen_R$ return (sk_S, pk_S), $sk_S = (\rho_S, K_S, tr_S, s_1^{(S)}, s_2^{(S)}, t_0^{(S)})$, $pk_S := (\rho_S, t_1^{(S)})$.

(4) Keyword Encryption $PAEKS((w, sk_S, pk_R, params) \rightarrow C_w)$

This algorithm takes system parameters $(params)$, keywords $w \in \{0,1\}^*$, the sender's private key $sk_S := (\rho_S, K_S, tr_S, s_1^{(S)}, s_2^{(S)}, t_0^{(S)})$, and the receiver's public key $pk_R := (\rho_R, t_1)$ as inputs, and outputs the keyword ciphertext. $C_w = (z, c, w_1)$, where $z \in R_q^\ell, c \in B_{60}, w_1 \in R_q^k$.

Step 1: Calculate $\mu_w = CRH(tr_S \parallel w \parallel \rho_R) \in \{0,1\}^{384}$.

Step 2: Generate the mask vector $y = ExpandMask_q(K_S \parallel \mu_w) \in S^{\ell}{}_{\gamma_1 - 1}$.

Step 3: Generate the expanded matrix $A_S = ExpandA(\rho_S)$.
Step 4: Calculate the intermediate vector $w = A_S\, y \in {R_q}^k$.
Step 5: Calculate the high-order bits $w_1 = HighBits_q(w, 2_{\gamma_2}) \in {R_q}^k$.
Step 6: Calculate $c = H(\mu_w \parallel w_1) \in B_{60}$.
Step 7: Calculate the response vector $z = y + c \cdot s_1^{(S)} \in {R_q}^\ell$.
Step 8: Verify whether the following formula holds. If it does, output the result $C_w = (z, c, w_1)$.

$$\parallel z \parallel_\infty < \gamma_1 - \beta \tag{1}$$

(5) Keyword trapdoor generation $Trapdoor((w\prime, sk_R, pk_S, params) \rightarrow T_{w\prime})$

The algorithm takes the system parameters $(params)$, keywords $w \in \{0,1\}^*$, the sender's public key $pk_S := (\rho_S, t_1^{(S)})$, and the receiver's private key $sk_R := (\rho_R, K_R, tr_R, s_1^{(R)}, s_2^{(R)}, t_0^{(R)})$ as input, and outputs the keyword trapdoor $T_{w'}$.
Step 1: Calculate $\mu_{w'} = CRH(tr_R \parallel w \parallel \rho_S) \in \{0,1\}^{384}$.
Step 2: Generate the mask vector $y' = ExpandMask_q(K_R \parallel \mu_{w'})$.
Step 3: Calculate $A_S = ExpandA(\rho_S)$.
Step 4: Calculate $w' = A_S y'$.
Step 5: Calculate $w_1' = HighBits_q(w', 2_{\gamma_2}) \in {R_q}^k$.
Step 6: Calculate $c' = H(\mu_{w'} \parallel w_1')$.
Step 7: Calculate $z' = y' + c' \cdot s_1$.
Step 8: Verify if the following formula holds true. If so, output $T_{w'} = (z', c', \mu_{w'})$.

$$\parallel z' \parallel_\infty < \gamma_1 - \beta \tag{2}$$

(6) Match test $Test(C_w, T_{w\prime}, params) \rightarrow \{0,1\}$

This algorithm takes as input the keyword ciphertext, keyword trapdoor, and system parameters $(params)$, and performs a matching test to verify whether the keywords contained in them are identical. The specific algorithm is as follows:
Step 1: Calculate $A_S \leftarrow ExpandA(\rho_S)$.
Step 2: Generate the mask vector $w^* \leftarrow A_S z - c \cdot t_1^{(S)} \cdot 2^d$.
Step 3: Calculate $A_S = ExpandA(\rho_S)$
Step 4: Calculate $w_1^* \leftarrow UseHint_q(0, w^*, 2\gamma_2)$.
Step 5: Check if the following condition holds. If not, return 0; otherwise, proceed to the next step.

$$w^*{}_1 \neq w_1 \; or \; \parallel z \parallel_\infty \geq \gamma_1 - \beta \tag{3}$$

Step 6: Calculate $w'^* = A_S z' - c' \cdot t_1^{(S)} \cdot 2^d$.
Step 7: Calculate $w_1'^* = UseHint_q(0, w'^*, 2\gamma_2)$.
Step 8: Compute the challenge $c^* = H(\mu_{w'} || w_1'^*)$.
Step 9: Check if the following condition holds. If true, proceed to the next step; otherwise, return 0.

$$c^* \neq c' \; or \; ||z'||_\infty \geq \gamma_1 - \beta \tag{4}$$

Step 10: Calculate $c^{**} = H(\mu_{w'} || w_1)$.
Step 11: Check if the following condition holds. Return 1 if true, otherwise, return 0.

$$c^{**} = c \tag{5}$$

2.3. Correctness Analysis of the Dilithium-PAEKS Scheme

If the receiver generates a valid keyword trapdoor, the keyword ciphertext is valid, and the keywords in both match, the matching test algorithm will pass. Let the sender's

public–private key pair be (pk_S, sk_S), the receiver's public–private key pair be (pk_R, sk_R), the keyword ciphertext be $C_w = (z, c, w_1)$, and the keyword trapdoor be $T_w = (z', c', \mu_w)$. The correctness verification of the scheme consists of sender signature verification, receiver signature verification, and keyword matching verification. The specific verification process is as follows.

Sender signature verification:

$$w^* = A_S z - c \cdot t_1^{(S)} \cdot 2^d = A_S(y + c \cdot s_1^{(S)}) - c \cdot (A_S s_1^{(S)} + s_2^{(S)})_1 \cdot 2^d = w - c \cdot s_2^{(S)} + O(\gamma_2) \quad (6)$$

then there is

$$w_1^* = UseHint_q(0, w^*, 2\gamma_2) = w_1 \quad (7)$$

Recipient signature verification:

$$w'^* = A_S z' - c' \cdot t_1^{(S)} \cdot 2^d = A_S(y' + c' \cdot s_1) - c' \cdot (A_S s_1^{(S)} + s_2^{(S)})_1 \cdot 2^d \approx w' \quad (8)$$

then there is

$$c^* = H(\mu_w \parallel w_1'^*) = c' \quad (9)$$

Keyword match verification:

$$c^{**} = H(\mu_w \parallel w_1) = H(\mu_w \parallel w_1) = c \quad (10)$$

Only when $w = w'$, $\mu_w = \mu_{w\prime}$,

$$c^{**} = c, \quad (11)$$

then there is

$$Test \rightarrow 1. \quad (12)$$

Analysis shows that signature verification ensures the validity of C_w and T_w; bound hash $\mu_w = CRH(tr_S \parallel w \parallel \rho_R) = CRH(tr_R \parallel w \parallel \rho_S)$ ensures keyword consistency; when $w = w'$, $H(\mu_w \parallel w_1) = c$ holds. In conclusion, Dilithium-PAEKS satisfies the correctness requirements of the PAEKS scheme.

3. Security Proof of the Scheme

This section first presents the security assumptions of the scheme and proves its security based on these assumptions.

3.1. Core Security Property Definitions

Before the security proof, this section defines the basic security requirements and advanced security properties of the PAEKS scheme, all based on the random oracle model, with the adversary defined as a PPT algorithm.

Definition 4 (Correctness). *For the Dilithium-PAEKS scheme, given valid system parameters params, legal sender–receiver key pairs (pk_s, sk_s) and (pk_R, sk_R), a keyword ciphertext CT_W generated by the legal encryption and trapdoor T_W generated by the legal trapdoor algorithm, the matching test algorithm satisfies $Test(params, CT_w, T_w) = 1$. That is, the valid ciphertext and trapdoor of the same keyword can pass the matching test with certainty.*

Definition 5 (Completeness). *For any keyword set W, the Dilithium-PAEKS scheme can correctly generate the corresponding ciphertext set $\{CT_w\}w \in W$ and trapdoor set $\{T_w\}w \in W$, and for any $w \in W$, the matching test result is 1; for any non-matching keyword pair, the result is 0. The scheme can completely distinguish the matching and non-matching relationships between keywords, ciphertexts and trapdoors.*

Definition 6 (**Soundness**). *For any PPT adversary, it is computationally infeasible to forge a ciphertext CT_w^* or a trapdoor T_w^* such that $Test(params, CT_w, T_w^*) = 1$ or $Test(params, CT_w^*, T_w) = 1$ holds for an unauthenticated sender or an unmatched keyword w^*.*

Definition 7 (**IND-CKA** (Indistinguishable under Chosen-Keyword Attack)). *The Dilithium-PAEKS scheme is IND-CKA secure if for any PPT adversary A, the advantage $Adv_A^{IND-CKA} = \left| Pr[Aguessescorrectly] - \frac{1}{2} \right|$ is negligible. The adversary has the ability of adaptive chosen-keyword query, and can query the ciphertext of any keyword except the challenge keyword from the challenger; the security goal is that the adversary cannot distinguish the ciphertexts of two different challenge keywords with a probability significantly higher than 1/2.*

Definition 8 (**TIK** (Trapdoor Indistinguishability)). *The Dilithium-PAEKS scheme has TIK security if for any PPT adversary A, the advantage $Adv_A^{TIK} = \left| Pr[Aguessescorrectly] - \frac{1}{2} \right|$ is negligible. For two arbitrary and distinct keywords W_0, W_1, the adversary cannot distinguish the trapdoor T_{w0} and T_{w1} generated by the legal trapdoor algorithm with a probability significantly higher than 1/2, thus avoiding keyword information leakage through trapdoor analysis.*

3.2. Security Assumptions

Definition 9 (**MLWE hypothesis**). *Given a matrix and a vector, for any probabilistic polynomial time (PPT) adversary, the probability of distinguishing them from a uniformly random sample is negligible, i.e., $A = {R_q}^{k \times \ell}$, $t = As_1 + s_2$, $s_1 \leftarrow S_q^{\ell}, s_2 \leftarrow S_\eta^{\kappa}$, $\mathcal{A}$, (A, t).*

$$\Pr\left[\begin{array}{c} A \leftarrow {R_q}^{k \times \ell}, \\ s_1 \leftarrow {S_q}^{\ell}, \\ s_2 \leftarrow {S_\eta}^{\kappa}, \\ t := As_1 + s_2, \\ (A_0, t_0) := (A, t), \\ (A_1, t_1) \leftarrow Uniform, \\ \mathcal{A}(A_b, t_b) = b \end{array}\right] - \frac{1}{2} \leq negl(\lambda) \tag{13}$$

Definition 10 (**MSIS hypothesis**). *Given a matrix, find a non-zero vector that satisfies the condition. For any PPT adversary, the probability of successfully solving this problem is, i.e., $A \leftarrow R_q^{k \times \ell}, (z_1, z_2), Az_1 + z_2 = 0, \| (z_1, z_2) \|_\infty \leq \beta, \mathcal{A}$.*

$$Pr\left[\begin{array}{c} A \leftarrow {R_q}^{k \times \ell}, \\ (z_1, z_2) \leftarrow \mathcal{A}(A), \\ Az_1 + z_2 = 0, \\ \| (z_1, z_2) \|_\infty \leq \beta, \\ (z_1, z_2) \neq 0 \end{array}\right] \leq negl(\lambda) \tag{14}$$

3.3. Security Proof

This section provides a security analysis and proof of Dilithium-PAEKS. By proving Theorem 1, the scheme is shown to satisfy keyword ciphertext indistinguishability and keyword trapdoor indistinguishability.

Theorem 1. *If the MLWE and MSIS security assumptions are valid and CRH is a collision-resistant hash function, the Dilithium-PAEKS scheme satisfies ciphertext indistinguishability and trapdoor*

privacy. Theorem 1 is derived from two lemmas: Lemma 1 proves the ciphertext indistinguishability of the scheme, and Lemma 2 demonstrates the trapdoor indistinguishability.

Lemma 1. *If the MLWE security assumption holds, Dilithium-PAEKS is IND-CKA secure.*

Proof. Suppose a PPT adversary $\mathcal{A}$ can break the IND-CKA security of the scheme with a non-negligible advantage after making qH hash queries. Then, the challenger $\mathcal{C}$ can solve the MLWE problem with a non-negligible advantage through an interaction. Input an instance of the MLWE problem (A, t) where $t = As_1 + s_2$ with $(s_1 \leftarrow S_\eta{}^\ell, s_2 \leftarrow S_\eta{}^\kappa)$. The specific interaction process between $\mathcal{C}$ and $\mathcal{A}$ is as follows.

(1) Initialization phase

First, $\mathcal{C}$ randomly select $k \in [1, qH]$. Then $\mathcal{C}$ generates the sender's key pair by choosing random vectors $s_1(s), s_2(s)$ and computing the sender's public key $pk_S = (\rho_S, t_1^{(S)})$ where $t_{\mathbf{1}}(s) = Power2Round_q\ (A_s s_{\mathbf{1}}(s) + s_{\mathbf{2}}(s))$. The receiver's public key is set as $pk_R = (\rho_R, Power2Round_q(t))$ using the MLWE instance. $\mathcal{C}$ then sets the system public parameter list $params = (\lambda, n, q, \ell, \kappa, k, \gamma_1, \gamma_2, \beta, d, \omega, CRH, H)$, and returns $(params, pk_S, pk_R)$ to $\mathcal{A}$.

(2) Hash query phase

$\mathcal{A}$ The system adaptively makes two types of inquiries:

① CRH query: $\mathcal{C}$ maintains the list $L_1 =< w_i, \mu_i >$. If w_i exists in the list, return μ_i; otherwise, update L_1 and return μ_i according to Equation (15):

$$\mu_i = \left\{ \begin{array}{l} CRH(tr_S \parallel w_i \parallel \rho_R) i \neq k \\ Random \in \{0,1\}^{384} i = k \end{array} \right\} \tag{15}$$

② H query: $\mathcal{C}$ maintains the list $L_2 =< \mu_i, w_1, i, c_i >$. If (μ_i, w_1, i) exists, return c_i; otherwise, update L_2 and return c_i according to formula (16):

$$c_i = \left\{ \begin{array}{l} Uniform(B_{60}) i \neq k \\ H(\mu_i \parallel w_1, i) i = k \end{array} \right\} \tag{16}$$

(3) Inquiry stage

① Keyword ciphertext query:

For a query on W_i with $i \neq k$ $\mathcal{C}$ uses the sender's private key $s_1(s)$ generated in the initialization phase to compute $y \leftarrow ExpandMask(K_S \parallel \mu_i)$, $w_1 \leftarrow HighBits_q(A_S y, 2\gamma_2)$, $z \leftarrow y + c_i s_1{}^{(S)}$, returns $C_w = (z, c_i, w_1)$ to $\mathcal{A}$.

② Keyword trapdoor query $\mathcal{O}_{Trapdoor}(w_i')$:

For a query on W_i with $i \neq k\mathcal{C}$ computes (using the same sender's private key $s_1(s)$, $\mu_{w_i\prime} \leftarrow CRH(tr_R \parallel w_i\prime \parallel \rho_S)$, $y' \leftarrow ExpandMask(K_R \parallel \mu_{w_i\prime})$, $w_1' \leftarrow HighBits_q(A_S y', 2\gamma_2)$, $z' \leftarrow y' + c's1$, $Tw_i' = (z', c', \mu_{w_i\prime})$, returns $T_{w_i\prime}$ to $\mathcal{A}$.

(4) Challenging Phase

① $\mathcal{A}$ submits challenge keywords w_0^*, w_1^*.

② $\mathcal{C}$ randomly selects $b \leftarrow \{0,1\}$. If $w_b^* \neq w_k$, terminate the simulation; otherwise, generate the challenge ciphertext using the sender's private key $s_1(s)$: $y^* \leftarrow Uniform(S^\ell_{\gamma_1 - 1})$, $w_1^* \leftarrow HighBits_q(t, 2\gamma_2)$, $c^* \leftarrow Uniform(B_{60})$, $z^* \leftarrow y^* + c^* s_1{}^{(S)}$, $C_{w_b^*} = (z^*, c^*, w_1^*)$; return $C_{w_b^*}$ to $\mathcal{A}$.

(5) Output stage

① $\mathcal{A}$ outputs guess b'.

② $\mathcal{C}$ constructs an MLWE discriminator: if $b' = b$ the output is an MLWE instance, otherwise, the output is a uniform random. The probability of success is $Adv_{\mathcal{C}}^{MLWE} \geq \frac{1}{qH}(Adv_{\mathcal{A}}^{IND-CKA} - negl(\lambda))$, which contradicts the hypothesis. □

Lemma 2. *If the MSIS security assumption holds, then Dilithium-PAEKS is TIK secure.*

Proof. Assume there exists a PPT adversary B that can break TIK security. In that case, the challenger $\mathcal{D}$ can solve the MSIS problem. Input an MSIS instance $A \in R_q^{k \times \ell}$, and find a non-zero vector (z_1, z_2) that satisfies $Az_1 + z_2 = 0$ and $\| (z_1, z_2) \|_\infty \leq \beta$. The interaction process is as follows.

(1) Initialization phase

① $\mathcal{C}$ sets system parameters *params*, randomly selects $k \in [1, qT]$, where qT is the trapdoor inquiry count.

② Set sender matrix $A_S = A$.

③ Generate receiver key pair (sk_R, pk_R).

④ Return $(params, pk_S, pk_R)$ to $\mathcal{B}$.

(2) Hash query phase

① CRH query: $\mathcal{D}$ maintains the list $L_1 =< w_i, \mu_i >$. If the item exists, return it; otherwise, update it and return according to Equation (17):

$$\mu_i = \left\{ \begin{array}{l} CRH(tr_S \parallel w_i \parallel \rho_R) i \neq k \\ Random \in \{0,1\}^{384} i = k \end{array} \right\} \tag{17}$$

② H query: Maintain the list $L_2 =< \mu_i, w_1, i, c_i >$. If (μ_i, w_1, i) exists, return c_i; otherwise, update and return according to Equation (18):

$$c_i = \left\{ \begin{array}{l} Uniform(B_{60}) i \neq k \\ H(\mu_i \parallel w_1, i) i = k \end{array} \right\} \tag{18}$$

(3) Inquiry stage

① Keyword ciphertext query phase $\mathcal{O}_{PAEKS}(w_i)$:

If $i = k$, $\mathcal{D}$ terminates (probability $\frac{1}{qH}$); otherwise compute $y \leftarrow ExpandMask(K_S \parallel \mu_i)$, $w_1 \leftarrow HighBits_q(A_S y, 2\gamma_2)$, $z \leftarrow y + c_i s_1^{(S)}$, return $C_w = (z, c_i, w_1)$ to $\mathcal{B}$.

② Keyword trap question $\mathcal{O}_{Trapdoor}(w_i')$:

$\mathcal{D}$ maintains the lists L_{CRH}, L_H to handle hash queries and computes $z = y + c \cdot s_1^{(R)}$ with the real private key, returning $T_w = (z, c, \mu_w)$.

(4) Forgery stage

$\mathcal{B}$ outputs a forged trapdoor $T_w^* = (\mathbf{z}^*, c^*, \mu_w^*)$, where $\| z^* \|_\infty < \gamma_1 - \beta$ and $w^* \notin \mathcal{O}_T$ inquiry.

(5) MSIS Reduction and Solution

① $\mathcal{B}$ first checks the list L_H for the corresponding item $\mu_w{}^*$, such that $c^* = H(\mu_w{}^* \parallel w_1{}^*)$.

② Compute $w^* = A_S z^* - c^* \cdot t_1^{(S)} \cdot 2^d$, $w_1^* = UseHint_q(0, w^*, 2\gamma_2)$.

③ From H-consistency, ${w_1}^* = w_1$, the error term e^* satisfies $UseHint_q\left|\left|\mathbf{e}^*\right.\right\|_\infty \leq \gamma_2$ and Equation (19):

$$\underbrace{A_S(z^* - y^*)}_{z_1} + \underbrace{\left(c^* \cdot s2^{(S)} - e^*\right)}_{z_2} = 0 \tag{19}$$

④ then outputs the solution to the MSIS problem, where $\mathcal{B}$, (z_1, z_2), $c = H(w, pk\_S, pk\_R, w_1)$, $z_2 = c^* s2^{(S)} - e^*$. $\mathcal{B}$ probability of success contradicts the hypothesis $\Pr[B\ solve\ \text{MSIS}] \geq -negl(\lambda)$, and the theorem is proved. □

3.4. Resistance to Keyword Guessing Attacks

Keyword guessing attacks are a critical threat to searchable encryption schemes. We analyze two attack scenarios:

(1) Outside Keyword Guessing Attacks (OKGA)

In this scenario, an external adversary who intercepts a ciphertext attempts to guess the keyword. Our IND-CKA proof (Lemma 1) demonstrates that the keyword is information-theoretically hidden within the MLWE-hard instance, making offline guessing computationally infeasible.

(2) Insider Keyword Guessing Attacks (IKGAs)

More critically, malicious servers possessing valid trapdoors might attempt to perform offline dictionary attacks on intercepted ciphertexts. Traditional PAEKS schemes are vulnerable because trapdoors can be tested against ciphertexts without sender participation. Dilithium-PAEKS prevents IKGA through:

a. Sender–Receiver Binding: Each ciphertext embeds both the sender's private key s_S and the receiver's public key TW through the hash computation $c = H(w, pk\_S, pk\_R, w_1)$. Without knowing $c = H(w, pk\_S, pk\_R, w_1)$, even a malicious server holding a valid trapdoor TW cannot forge a matching ciphertext for keyword verification.
b. Trapdoor Indistinguishability: As proven in Lemma 2, trapdoors leak no information about keywords beyond what is revealed during legitimate matching operations. The MSIS-hardness ensures that even computational adversaries cannot extract keyword information from trapdoor analysis.

Formal security game for IKGA resistance can be defined as an extension of the IND-CKA game where the adversary is additionally given access to a trapdoor generation oracle. The proof follows a similar reduction to Lemma 1 with additional trapdoor query handling.

4. Experimental Analysis and Comparison

This section evaluates the performance of the Dilithium-PAEKS scheme on a hardware platform featuring an Intel® Core™ i5-12400F processor (Intel Corporation, Santa Clara, CA, USA), 16 GB of RAM, and Windows 11 operating system. The testing was conducted using PyCharm 2023.3.2 with the MIRACL cryptographic core library.

This experiment compared Dilithium-PAEKS with three representative cryptographic schemes: the SM9-PAEKS scheme based on the SM9 national cryptographic algorithm from reference [7], the scheme referred to as CP-AbSEL in reference [15], and the post-quantum cryptographic scheme LB-FSSE from reference [16]. Under a fixed-keyword condition, each of the three core components—keyword encryption, trapdoor generation, and matching test—was executed independently 1000 times for every scheme. The arithmetic mean of the execution times was calculated to minimize measurement errors.

As illustrated in Figures 1–3, in terms of keyword encryption, trapdoor generation, and matching test algorithms, the Dilithium-PAEKS scheme exhibits lower efficiency compared

to SM9-PAEKS. Although SM9-PAEKS demonstrates higher operational efficiency, it is vulnerable to quantum attacks.

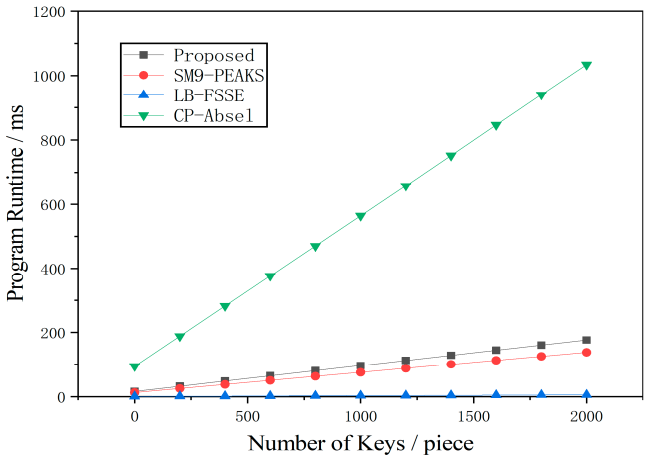

Figure 1. Comparison of time consumption of four keyword encryption algorithms.

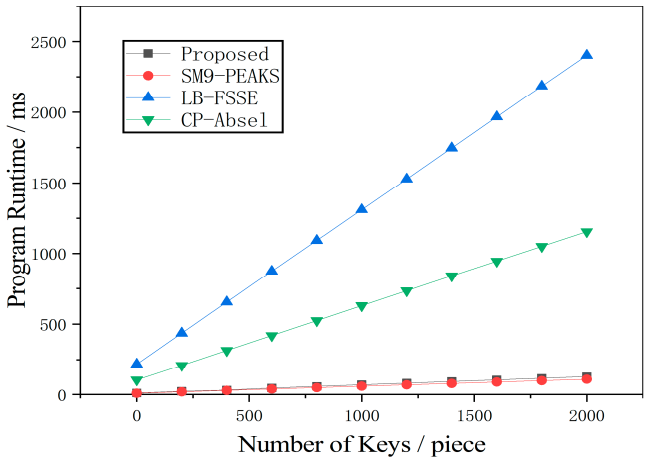

Figure 2. Comparison of time consumption of four schemes for keyword trapdoor generation algorithm.

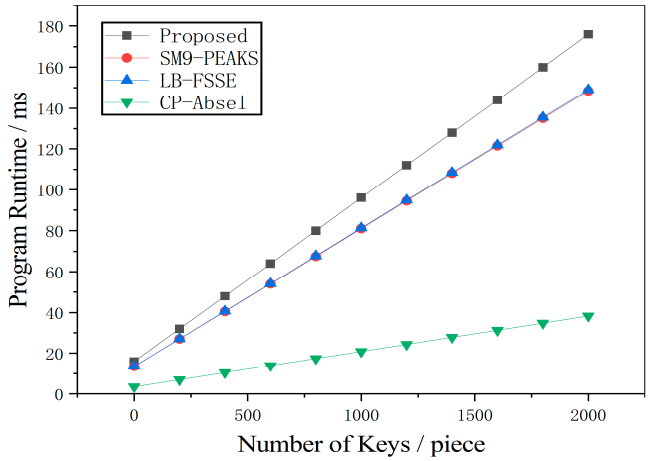

Figure 3. Comparison of time consumption of matching test algorithm for four schemes.

As shown in Figures 1–3, when the number of keywords is 200, the execution time of Dilithium-PAEKS in the keyword encryption algorithm is 25.14 ms, which is higher than that of SM9-PAEKS (12.56 ms) and LB-FSSE (0.5 ms), but lower than that of CP-Absel (93.37 ms). In the trapdoor generation algorithm, the execution time of Dilithium-PAEKS is comparable to that of SM9-PAEKS, and its operational efficiency surpasses both LB-FSSE

and CP-Absel. In the matching test algorithm, however, the execution time of Dilithium-PAEKS exceeds that of the other three schemes.

Figure 4 shows that the total execution time of Dilithium-PAEKS (54.76 ms) is higher than that of SM9-PAEKS (38.89 ms) but significantly lower than that of CP-Absel (202.48 ms) and LB-FSSE (232.45 ms). It is worth noting that although LB-FSSE achieves extremely fast keyword encryption (0.5 ms), both schemes are fundamentally built upon lattice-based cryptography (specifically the LWE problem), where the inherent computational overhead of high-dimensional matrix operations and noise sampling exceeds that of traditional cryptosystems. Building on this foundation, CP-ABSEL incorporates a ciphertext-policy attribute-based encryption mechanism, which forces the keyword encryption and trapdoor generation processes to additionally handle complex access control structures, further exacerbating the computational burden. Meanwhile, the scheme by Islam et al., in its pursuit of forward security and integration with blockchain technology, must not only bear the inherent computational costs of lattice-based cryptography but also endure the additional overhead of state update mechanisms and systemic delays caused by blockchain transaction latency and ledger verification. It is precisely this combination of the "high overhead of underlying lattice primitives" and the "composite burden of high-level enhanced functionalities" that results in the lower overall efficiency of these two schemes in key algorithmic phases compared to the more streamlined design of Dilithium-PAEKS.

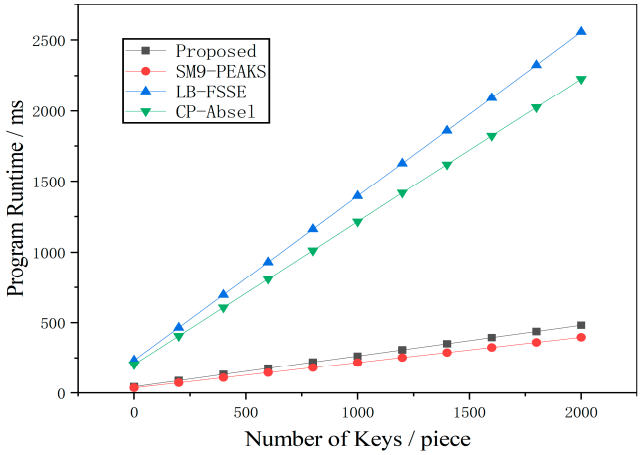

Figure 4. Comparison of total time for four solutions.

In summary, Dilithium-PAEKS demonstrates advantages over existing searchable encryption algorithms in terms of both security and operational efficiency.

5. Conclusions

To address quantum security threats in traditional searchable encryption schemes, this study is the first to apply the post-quantum standard CRYSTALS-Dilithium algorithm to public-key searchable encryption, proposing the Dilithium-PAEKS framework. By implementing lattice-based key generation, keyword-bound signatures, and dynamic matching mechanisms, the scheme achieves both high efficiency and strong security. Theoretical analysis confirms its indistinguishability between ciphertexts and trapdoors under the MLWE/MSIS assumption. Crucially, the scheme demonstrates robust resistance to both outside and insider keyword guessing attacks, addressing a major vulnerability in existing searchable encryption solutions through innovative Sender–Receiver binding and trapdoor privacy mechanisms. The experimental results show superior overall computational efficiency compared to representative post-quantum searchable encryption schemes such as CP-Absel and LB-FSSE while avoiding the high computational overhead associated

with bilinear pairings and blockchain integration. Future work will focus on supporting multi-keyword retrieval, optimizing dynamic database scenarios, and developing bandwidth-efficient designs to further advance the practical adoption of post-quantum searchable encryption technology.

Author Contributions: Conceptualization, M.Z. and A.X.; methodology, A.X.; software, M.Z.; validation, D.K.; formal analysis, A.X.; investigation, S.H.; resources, M.Z.; data curation, D.K.; writing—original draft preparation, A.X.; writing—review and editing, M.Z.; visualization, S.H.; project administration, M.Z.; funding acquisition, M.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lanyon, B.P.; Weinhold, T.J.; Langford, N.K.; Barbieri, M.; James, D.F.; Gilchrist, A.; White, A.G. Experimental demonstration of a compiled version of Shor's algorithm with quantum entanglement. *Phys. Rev. Lett.* **2007**, *99*, 250505. [CrossRef] [PubMed]
2. Long, G.L. Grover algorithm with zero theoretical failure rate. *Phys. Rev. A* **2001**, *64*, 022307. [CrossRef]
3. Fernandez-Carames, T.M.; Fraga-Lamas, P. Towards Post-Quantum Blockchain: A Review on Blockchain Cryptography Resistant to Quantum Computing Attacks. *IEEE Access* **2020**, *8*, 21091–21116. [CrossRef]
4. Song, D.X.; Wagner, D.; Perrig, A. Practical techniques for searches on encrypted data. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy (S&P 2000), Berkeley, CA, USA, 14–17 May 2000*; IEEE: Piscataway, NJ, USA, 2000; pp. 44–55.
5. Boneh, D.; Di Crescenzo, G.; Ostrovsky, R.; Persiano, G. Public key encryption with keyword search. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2004), Interlaken, Switzerland, 2–6 May 2004*; Springer: Berlin, Germany, 2004; pp. 506–522.
6. Xu, L.; Xu, C.G.; Yu, X.L. Secure and Efficient Data Retrieval Scheme Using Searchable Encryption in Cloud. *J. Cryptolog. Res.* **2016**, *3*, 330–339. [CrossRef]
7. Li, S.Q.; Yang, B.; Wang, T.; Zhou, Y.W. Efficient Public Key Encryption with Keyword Search Without Using Secure Channel. *J. Cryptolog. Res.* **2019**, *6*, 283–292. [CrossRef]
8. Deng, Z.H.; Wang, S.H.; Wang, P. Analysis and Improvement of Searchable Encryption Scheme Based on Composite-Order Bilinear Pair. *Comput. Eng.* **2020**, *46*, 123–128+135. [CrossRef]
9. Cui, R.R.; Zhang, Y.S.; Wei, Y. Multiple Keywords Searchable Encryption Scheme Based on Elliptic Curve. *J. Jinan Univ.* **2019**, *33*, 353–360. [CrossRef]
10. Zhang, C.; Peng, C.G.; Ding, H.F.; Xu, D.Q. Searchable Encryption Scheme Based on China State Cryptography Standard SM9. *Comput. Eng.* **2022**, *48*, 159–167. [CrossRef]
11. Pu, L.; Lin, C.; Wu, W.; Gu, J.; He, D. Public-key Authenticated Encryption Scheme with Keyword Search from Chinese Cryptographic SM9. *J. Softw.* **2025**, *36*, 4271–4284. [CrossRef]
12. Zhang, E.; Hou, Y.Y.; Li, G.L.; Li, H.M.; Li, Y. Adaptive hierarchical searchable encryption scheme based on learning with errors. *Comput. Appl.* **2020**, *40*, 148–156.
13. Liu, Y.; Wang, L.C.; Zhou, Y.B. TTRC-ABE: A Traceable and Revocable Grid-Based Attribute Encryption Scheme Based on the CLWE Problem. *J. Electron. Inf. Technol.* **2025**, *47*, 1911–1926. [CrossRef]
14. Yu, H.; Bai, X. Identity-based searchable attribute signcryption in lattice for a blockchain-based medical system. *Front. Inf. Technol. Electron. Eng.* **2024**, *25*, 461–472. [CrossRef]
15. Varri, U.S.; Pasupuleti, S.K.; Kadambari, K.V. CP-ABSEL: Ciphertext-policy attribute-based searchable encryption from lattice in cloud storage. *J. Cloud Comp.* **2021**, *10*, 1290–1302. [CrossRef]
16. Islam, S.H.; Mishra, N.; Biswas, S.; Keswani, B.; Zeadally, S. An efficient and forward-secure lattice-based searchable encryption scheme for the Big-data era. *Comput. Electr. Eng.* **2021**, *96*, 107533. [CrossRef]
17. Ducas, L.; Kiltz, E.; Lepoint, T.; Lyubashevsky, V.; Schwabe, P.; Seiler, G.; Stehlé, D. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2018**, *2018*, 238–268. [CrossRef]
18. National Institute of Standards and Technology (NIST). *FIPS 204: Module-Lattice-Based Digital Signature Standard*; U.S. Department of Commerce: Gaithersburg, MD, USA, 2024.

19. Bos, J.; Ducas, L.; Kiltz, E.; Lepoint, T.; Lyubashevsky, V.; Schanck, J.M.; Schwabe, P.; Seiler, G.; Stehlé, D. CRYSTALS–Kyber: A CCA-Secure Module-Lattice-Based KEM. In *Proceedings of the IEEE European Symposium on Security and Privacy (EuroS&P 2018), London, UK, 24–26 April 2018*; IEEE: Piscataway, NJ, USA, 2018; pp. 353–367.
20. Lyubashevsky, V.; Seiler, G. Short, Invertible Elements in Partially Splitting Cyclotomic Rings and Applications to Lattice-Based Zero-Knowledge Proofs. In *Proceedings of the Advances in Cryptology–EUROCRYPT 2018, Tel Aviv, Israel, 29 April–3 May 2018*; Springer: Cham, Switzerland, 2018; pp. 204–224.