Computer Standards & Interfaces 97 (2026) 104118

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Post-quantum PAKE over lattices revised: Smaug-T.PAKE for mobile devices

Kübra Seyhan [a,*], Sedat Akleylek [b,c], Ahmet Faruk Dursun [a]

[a] Department of Computer Engineering, Ondokuz Mayis University, Faculty of Engineering, Samsun, 55139, Turkiye
[b] Chair of Security and Theoretical Computer Science, University of Tartu, Institute of Computer Science, Tartu, 50090, Estonia
[c] Department of Software Engineering, Istinye University, Faculty of Engineering and Natural Sciences, Istanbul, 34396, Turkiye

ARTICLE INFO

Keywords:
Post-quantum cryptography
Lattice-based cryptography
SMAUG-T
Password-authenticated key exchange

ABSTRACT

In this paper, an efficient post-quantum secure password-authenticated key exchange (PAKE) scheme from a well-structured lattice-based key encapsulation mechanism (KEM) is proposed. The generic KEM to PAKE idea, OCAKE, is modified by considering hybrid module learning with errors (MLWE) + module learning with rounding (MLWR) assumptions to obtain explicit password-based authentication from SMAUG-T.KEM procedures. As a KEM primitive, SMAUG-T.KEM is chosen due to its performance against the National Institute of Standards and Technology (NIST) standard Crystals-Kyber (Kyber) to obtain an efficient and post-quantum secure PAKE scheme. Firstly, the anonymity and fuzziness properties of SMAUG-T.KEM are proven to fit the OCAKE approach in constructing the PAKE version of Smaug.KEM. Then, the post-quantum security of the proposed SMAUG-T.PAKE is analyzed in the universal composability (UC) model based on the hybrid security assumptions and proved properties. The reference C and JAVA codes are written to evaluate whether the targeted efficiency is achieved in different platforms. Based on the central processing unit (CPU) and memory usage, run time, and energy consumption metrics, the proposed solution is compared with current PAKE proposals. The performance results showed that SMAUG-T.PAKE, with two optional encryption modes, Advanced Encryption Standard (AES) or Ascon, presents better performance than the other module-based PAKE solutions from lattices in terms of both reference and mobile results.

1. Introduction

A PAKE protocol provides secure key sharing on an insecure channel by using a pre-shared password as an authentication component [1]. In recent years, PAKE protocols have been preferred in wireless communication, e-passports, and the Internet of Things, where efficiency, portability, simplicity, and independence are essential [2,3]. PAKE usage in resource-limited communication models establishes independent, secure, and portable authenticated communication without extra public key infrastructure, complex components, or central authority. The main trade-off in the PAKE protocols arises between efficiency and security since they use low-entropy pre-shared passwords to derive high-entropy shared keys. The first introduction of PAKEs to the literature was given with encrypted key exchange (EKE) by Bellovin and Merritt in 1992 [4]. After this first attempt, several PAKE solutions were proposed, and standardization attempts were started by covering different design settings, primitives, properties, and usage areas. The recent standardization effort was made by the Internet Engineering Task Force (IETF) in 2020, and Cpace and OPAQUE were announced as standard traditional PAKE protocols [5,6].

The computational hardness of traditional PAKEs was basically captured by following the hardness of discrete logarithm problem (DLP). In the pre-quantum era, the security of PAKEs is maintained as there is no efficient and polynomial-time algorithm to solve DLP if the suitable parameter set is selected. The first appearance of the Shor algorithm [7] changed this situation and started a new challenge for public key cryptography (PKC) in the age of large-scale quantum computers. NIST announced a call to be ready for the post-quantum era by determining PKC standard(s) in 2016. In this period, it was aimed to select the quantum-resistant digital signature and public-key encryption&key-establishment algorithms. The NIST standardization process was finalized in 2024 and 2025, and lattice-based CRYSTALS-KYBER and code-based HQC were determined as the post-quantum KEM standard, while lattice-based CRYSTALS-Dilithium and Falcon, and hash-based SPHINCS+ were determined as digital signature standards [8]. In addition to the international call of the NIST, China, Korea, Ukraine, and Russia also started initiatives to determine their standards for post-quantum cryptography (PQC). Even if there has not been started a standardization process specific to PAKEs, there is ongoing research on the post-quantum secure PAKE design process

* Corresponding author.
E-mail address: kubra.seyhan@bil.omu.edu.tr (K. Seyhan).

https://doi.org/10.1016/j.csi.2025.104118
Received 25 August 2025; Received in revised form 29 October 2025; Accepted 12 December 2025
Available online 15 December 2025

in the literature and industry. In particular, PQC algorithms have higher key and cipher sizes compared to traditional cryptosystems, and they require high bandwidth and system resource usage, making efficient PQC protocol design difficult. As it is known, after standard cryptographic principles are determined, the needs are met by adapting them to different purposes and application areas as a black box in design. So, proposing a post-quantum secure PAKE based on a well-studied algorithm will be one of the appropriate candidates for the future security of PKCs. It is necessary to provide the required security definitions for the combination of different cryptographic principles in PAKE design, to present evidence on ensuring post-quantum security, and to perform performance evaluations on different platforms. Additionally, comparative performance and efficiency analysis with the methods used for PAKE design in the literature and the effect of using different models in cryptographic primitives used as black boxes should be examined to evaluate post-quantum secure PAKE protocols.

1.1. Motivation

PQC has received tremendous interest in industry and academic research due to the increasing work to build a sufficiently powerful quantum computer. To be ready for the post-quantum era, active work is being carried out with various attempts, such as standardization efforts, literature research, and industrial initiatives, specifically regarding the essential key agreement requirement. Among key sharing schemes, PAKEs stand out with their simple operations and strong security based on passwords. While studies on standardization and improvement of traditional PAKE protocols continue, the need to evaluate the post-quantum effect has also emerged. Although standardization efforts have been initiated to determine post-quantum secure standards for basic public key cryptosystems, no attempt has yet been made for specialized primitives like PAKEs. Although there has been increasing interest in developing quantum-safe PAKEs in recent years, little work has yet been done on building PAKEs by integrating standard PQC algorithms. The main motivation for this paper comes from [2,9,10], which identified the design, security, and performance analysis of a post-quantum secure, relatively efficient PAKE derived from standard primitives as an open problem. The cryptographic primitives in constructing the efficient post-quantum secure PAKE solution must be evaluated for their effects on security analysis and performance.

1.2. Contribution

In this paper, we focus on the problem of whether it is possible to construct efficient and post-quantum PAKE using well-defined post-quantum KEM algorithms and additional primitives. To obtain an efficient PAKE, the SMAUG-T [11] algorithm, which is the Korean PQC KEM standard and has been shown to be more efficient than the NIST KEM standards, was used to improve efficiency. With the focus on efficiency, the idea of OCAKE [12] is adopted into the proposed model to capture explicit password-based authentication. The proposed PAKE, SMAUG-T.PAKE, is the first MLWR+MLWE-based PAKE scheme that provides efficiency and post-quantum security. The contributions of this paper to the literature can be summarized as follows.

- The first MLWE+MLWR-based PAKE protocol that provides explicit authentication, anonymity, fuzziness, and efficiency for post-quantum era security is proposed.
- As an extended and modified version of [13], more efficient lattice-based PAKE with reduced encryption/decryption and Smaug.KEM operations are constructed. Unlike [13], MLWE+MLWR-based security definitions are defined in the UC model to capture explicit authentication.
- Firstly, the anonymity and fuzziness properties analysis of SMAUG-T.KEM are defined and provided to build modified MLWR+MLWE-based PAKE construction.
- In order to evaluate the effect of KEM and encryption/decryption operations on PAKE security analysis, MLWE+MLWR-based adaptations are also followed in the UC model.
- Detailed security analysis of the proposed PAKE based on the modified security definitions shows that SMAUG-T.PAKE maintains post-quantum security even with the additional components it includes.
- The efficiency analysis of the proposed PAKE is done by considering different application areas and additional primitive usages. Additionally, detailed performance analyses are provided to demonstrate that more efficient structures can be constructed compared to PAK-based PAKE construction models.
- The constructed SMAUG-T.PAKE is implemented in C for general use-case performance and also in JAVA for mobile application usage. To make a meaningful comparison, NIST standard CRYSTALS-Kyber is also implemented in C, based on the same construction idea.
- The efficiency of SMAUG-T.PAKE is analyzed in terms of CPU usage, memory usage, and energy consumption.
- According to the comparison, the constructed PAKE provides better performance results than other module-based PAKE solutions. Moreover, the proposed SMAUG-T.PAKE gives the best results when using the lightweight cipher Ascon.

1.3. Related work

In the literature, different PAKE design methodologies have been proposed to be ready for the post-quantum era. The presented lattice-based solutions were generally constructed by combining different projective hash functions (PHF), reconciliation structures, password-related computations, etc., according to the selected design idea. The proposed solutions can be divided into three categories: Use projective hash functions to convert an encryption scheme to a PAKE [14–20], build a PAKE from a key exchange (KE) idea by adding password-based authentication [21–29], and convert a KEM to PAKE [12,13,30–34]. In Table 1, a snapshot of lattice-based PAKE literature is given.

It is known that KEM schemes contain some extra encryption/decryption procedures, functionalities, and computations to ensure strong security properties. So, the nature of KEM-based PAKE schemes tends to show poor performance results and strong security features. In the lattice-based PAKE literature, two basic models for converting a KEM to PAKE exist. The first model combines traditional PAKE design [1] and well-structured KEM to add password-based authentication to the KEM idea. It uses KEM procedures as a black box and adds password-related primitives to provide password-based authentication in the KEM. In the second model, four generic KEM to PAKE constructions were provided in the sight of traditional PAKE designs. These generic PAKE models contain extra ideal cipher operations and use KEM procedures to ensure explicit or implicit authentication.

As the main focus of this paper, one-round lattice-based PAKE proposals that were constructed using KE and KEM approaches by adding password-related components will be summarized as follows. Note that to analyze the performance of the proposed PAKE, MLWE/MLWR-based PAKE construction will be used. In the following part, the main construction idea of these PAKEs will be briefly summarized.

In [35], the first lattice-based construction of conventional password-authenticated key exchange (PAK) PAKE [1] was presented in the literature regarding ring learning with errors (RLWE) hardness assumptions. The proposed PAKE provided a password-authenticated shared key generation for the two-party communication model. The security of this scheme was analyzed using the ROM assumptions. In [23], an efficient MLWE-based PAKE scheme, MLWE.PAK.PAKE, was proposed by considering the traditional PAK design idea [1]. MLWE.PAK.PAKE scheme consists of one round and was designed for the two-party communication model. The security analysis against password dictionary attacks was performed on the ROM. In [25],

Table 1
Outlook on lattice-based PAKEs.

PAKE construction model	Literature proposals	Hardness	Security model	Additional primitives
PAKE from KE	[21]	RLWE	ROM	X
	[22]	RLWE	ROM	X
	[23]	MLWE	ROM	X
	[24]	RLWE	ROR	X
	[25]	MLWR	ROM	X
	[26]	RLWE	ROM	X
	[27]	RLWE	ROR	X
	[28]	RLWE	ROM	X
	[29]	MLWE	ROM	X
PAKE from KEM	[30]	MLWE	ROM	X
	[31]	MLWE	UC	Feistel construction
	[32]	MLWE	ROM	Oneway plaintext-checking
	[12]	MLWE	UC	Ideal cipher
	[13]	MLWE + MLWR	UC	Two Cipher
	[33]	Non-uniform LWE	ROM	Ideal Cipher XOR
	[34]	X	ROM	SPHF XOR Encryption
	Ours	MLWE + MLWR	UC	Cipher
PAKE with PHF	[14]	LWE	ROM	SPHF encryption
	[15]	RLWE	UC	Oblivious Pseudorandom function NIZK
	[17]	LWE	Standard model	ASPHF Encryption ECC
	[18]	LWE	ROM	ASPHF
	[20]	LWE+LWR	ROM	SPHF Password hashing scheme
	[19]	LWE	ROM	SPHF NIZK

–ROM: Random Oracle Model –ECC: Error Correction Code –ROR: Real or Random –NIZK: Non-Interactive Zero Knowledge –UC: Universal Composability –SPHF: Smooth PHF –ASPHF: Approximate SPHF.

MLWR-based PAKE was constructed by considering NIST's Saber [36] KE idea and traditional PAKE approach [1]. Even if the design structure of the proposed Saber.PAK.PAKE is the same as MLWE.PAK.PAKE, it presented better performance results due to the efficient structure of the MLWR problem. In [30], the PAKE version of the NIST KEM standard, Kyber, was proposed using KEM functionalities and basic password components. The core construction of Kyber.PAK.PAKE is based on the traditional PAK approach to capture password-authenticated shared key generation. In [29], a MLWE-based three-party PAKE protocol was proposed by following the KE to PAKE design idea. In the security analysis, ROM-based assumptions were followed to provide the formal security.

In recent years, researchers have been working on how to convert a well-structured KEM into a PAKE. The main reason behind this idea is to create PAKE versions of standard KEMs for different applications or usage areas. Different models, [12,31–34], have been proposed in the literature considering the lattice assumptions. In [31], a compact PAKE construction, which requires the underlying KEM's one-wayness and anonymity, and the public key's uniformity, was defined. As a case example, Kyber-based PAKE was proposed and analyzed in the UC model. In [32], a generic PAKE model from a one-way secure against checkable attack secure KEM was introduced. The proposed construction also required that underlying KEM's anonymity and fuzziness. By considering Kyber KEM functions, the security of the proposed idea was analyzed using the ROM assumptions. In [33], KEM's PAKE design utilizes public-key authentication using XOR instead of public-key symmetric encryption. The goal is to create perfect privacy schemes by eliminating the need for ideal passwords. In [34], the definitions for PAKE models based on post-quantum KEM and traditional PAKE assumptions were provided. Details on the security analysis of the proposed hybrid PAKE models in the UC-model were also presented. In [12], Beguinet et al. proposed two different PAKE construction models that include KEM as a black box. According to the provided generic models, the CAKE version presents password-based implicit authentication with strong security proofs based on adaptive corruptions even if the receiver is unsure that he/she can receive the session key. The OCAKE approach captures explicit authentication using a key-confirmation tag without extra encryption. These two constructions require a KEM that provides fuzziness and anonymity properties for the public key and ciphertext, respectively. The security analysis was provided based on Kyber's assumption by considering password authenticated-based ideal functionality in the UC model.

1.4. Outline

In Section 2, the mathematical background is summarized. In Section 3, the constructed SMAUG-T.PAKE is defined, and the correctness analysis is given. In Section 4, the proof of security is detailed under the

Table 2
Notation.

q, p	:	Positive integers modulo, power of 2.
$\mathbb{Z}_q$	:	The quotient ring of integers in modulo q.
$\mathbb{Z}_q[x]$	:	Polynomials with coefficients in $\mathbb{Z}_q$.
$\mathfrak{R}_q = \mathbb{Z}_q[x]/(x^n+1)$	:	Quotient ring.
$\mathfrak{R}_q^{k\times k}$	:	Ring of $k \times k$ matrices on $\mathfrak{R}_q$.
χ	:	Discrete Gaussian distribution with $\sigma = 1.0625$ standard deviation.
S_η	:	SMAUG-T.KEM secret key and error term distribution. For $\eta \in \mathbb{Z}$, S_η denotes the set of polynomials of degree less than n with coefficients in $[-\eta, \eta] \cap \mathbb{Z}$.
^{T}	:	Transpose.
XOF	:	Extendable function. Utilized to generate the seed of A and set as Shake-128.
expandA	:	Sampling function for SMAUG-T.KEM's general public key. expandA$(\cdot)$: $p \in \{0,1\}^{256} \rightarrow A \in \mathfrak{R}_q^{k\times k}$.
hwt_h	:	Sampling function of SMAUG-T.KEM's sparse polynomials with hamming weight h. $hwt_h(\cdot) \rightarrow S_\eta^k$.
hs, hr	:	Non-zero coefficients of sparse polynomials, where $hs = \{140, 150, 145\}$ and $hr = \{132, 147, 140\}$ for 128, 192, 256-bit security.
t	:	Constant rounding component, where $t = 2$.
H	:	$\{0,1\}^{256} \rightarrow \{0,1\}^{256}$. Set as SHA3-256.
G, kdf	:	$\{0,1\}^{256} \times \{0,1\}^{256} \rightarrow \{0,1\}^{256}$. Set as SHA3-512 and Shake-256, respectively.
$\lfloor \cdot \rceil$	:	Rounding operator.
$x \leftarrow_r X$	:	x is selected uniformly random from X distribution.
sk-pk-ssk-ct-pw	:	Secret key-public key-shared secret key-ciphertext-password.
δ	:	Decryption failure probability of SMAUG-T.KEM.
$\|\| \cdot \|\|_\infty$	:	Infinity norm.
negl$(\cdot)$	:	Negligible function
H_1, H_2	:	$\{0,1\}^* \rightarrow \{0,1\}^{256}$. Set as SHA3-256.

Table 3
Algorithms for SMAUG-T.KEM.

SMAUG-T.PKE.KeyGen

seed$\leftarrow \{0,1\}^{256}$
$(p, \tau) \leftarrow$XOF(seed)
$A \leftarrow$expandA$(p) \in \mathfrak{R}_q^{k\times k}$
sk: $s \leftarrow hwt_{hs}(\tau) \in S_\eta^k$
$e \leftarrow \chi(\tau) \in \mathfrak{R}^k$
$b = -A^T s + e \mod q$
pk: $(p, b) \in \{0,1\}^{256} \times \mathfrak{R}_q^k$
Return $((p, b), s)$

SMAUG-T.PKE.Enc

Input: pk: (p, b)
Input: Message $\mu \in \{0,1\}^{256}$
Input: Seed $p' \in \{0,1\}^{256}$
$r \leftarrow hwt_{hr}(p') \in S_\eta^k$
$c_1 = \lfloor \frac{p}{q} Ar \rceil \in \mathfrak{R}_p^k$
$c_2 = \lfloor \frac{p}{q}(b^T r + \frac{q}{t}\mu) \rceil \in \mathfrak{R}_p$
ct: Return $((c_1, c_2))$

SMAUG-T.PKE.Dec

Input: sk: s
Input: ct: $(c_1, c_2) \in \mathfrak{R}_p^k \times \mathfrak{R}_p$
$\mu' = \lfloor \frac{t}{p}(c_2 + c_1^T s) \rceil \in \mathfrak{R}_p$
Message: Return (μ')

SMAUG-T.KEM.KeyGen()

$((p, b), s') =$ SMAUG-T.PKE.KeyGen()
d$\leftarrow \{0,1\}^{256}$
sk: $s = (s', d) \in S_\eta^k \times \{0,1\}^{256}$
Return $((p, b), s)$

SMAUG-T.KEM.Encap($pk = (p, b)$)

$\mu \leftarrow \{0,1\}^{256}$
(c_1, c_2)=SMAUG-T.PKE.Enc$((p, b), \mu; G(\mu, H((p, b))))$
$K = kdf(\mu, H((c_1, c_2))) \in \{0,1\}^{256}$
ssk: Return $((c_1, c_2), K)$

SMAUG-T.KEM.Decap($pk = (p, b), sk = (s', d), ct = (c_1, c_2)$)

$m' =$SMAUG-T.PKE.Dec$(s', (c_1, c_2))$
$(c_1', c_2') =$SMAUG-T.PKE.Enc$((p, b), \mu'; G(\mu', H((p, b))))$
if $(c_1, c_2) \neq (c_1', c_2')$
ssk: Return $K' = kdf(d, H((c_1, c_2))) \in \{0,1\}^{256}$
else
ssk: Return $K' = kdf(\mu, H((c_1, c_2))) \in \{0,1\}^{256}$

UC framework. In Section 5, the implementation results and detailed discussion are presented. Finally, in Section 6, the conclusion and future works are explained.

2. Preliminaries

In this section, we recall the underlying primitives basics and security-related details. The specific symbols and abbreviations are given in Table 2.

2.1. SMAUG-T primitives

In the proposed PAKE, generic OCAKE KEM to PAKE construction [12] is followed to obtain an efficient PAKE version of SMAUG-T.KEM [11], a Korean PQC standardization algorithm. Based on MLWR and MLWE assumptions, SMAUG-T.KEM scheme satisfies indistinguishability under chosen plaintext attacks (IND-CPA) and indistinguishability under adaptive chosen ciphertext attacks (IND-CCA2) security. Due to the module variants of selected problems and the efficiency of MLWR structure in encryption, SMAUG-T.KEM presents reduced public key and ciphertext sizes and running time results.

The main procedures of SMAUG-T.KEM [11] are remembered in Table 3.

As summarized in Table 3, SMAUG-T KeyGen processes follow MLWE assumptions to generate public and secret key pairs.

Definition 1 (*MLWE Problem [37]*)**.** Let $\{q, \eta, k\} \in \mathbb{Z}^+$, general public key $A \leftarrow_r \mathfrak{R}_q^{k\times k}$, secret key $s \leftarrow_r S_\eta^k$, and error term $e \leftarrow_r S_\eta^k$. MLWE distribution is generated with $(A, b = As + e) \in \mathfrak{R}_q^{k\times k} \times \mathfrak{R}_q^{k\times 1}$.

The hardness of MLWE is defined by the advantage (Adv) of adversary (**A**) to solve decisional-MLWE;

$$\mathrm{Adv}_{n,q,k,\eta}^{\mathrm{MLWE}}(\mathbf{A}) = \Big|\Pr[\mathbf{b} = 1 | A \leftarrow_r \mathfrak{R}_q^{k\times k}; b \leftarrow_r \mathfrak{R}_q^k; \mathbf{b} \leftarrow \mathbf{A}(A, b)] - \Pr[\mathbf{b} = 1 | A \leftarrow_r \mathfrak{R}_q^{k\times k}; s \leftarrow S_\eta^k; e \leftarrow S_\eta^k; b \leftarrow As + e; \mathbf{b} \leftarrow \mathbf{A}(A, b = As + e)]\Big| < negl(n)$$

In SMAUG-T's Enc and Dec procedures, given in Table 3 the ciphertexts are generated according to the MLWR assumption to reduce the key sizes.

Definition 2 (*MLWR Problem [36]*)**.** Let $\{p, q, \eta, k\} \in \mathbb{Z}^+$ such that $q \geq p \geq 2$, general public key $A \leftarrow_r \mathfrak{R}_q^{k\times k}$, and secret key $s \leftarrow_r S_\eta^k$. MLWR distribution is defined by $(A, b = \lfloor \frac{p}{q} As \rceil) \in \mathfrak{R}_q^{k\times k} \times \mathfrak{R}_p^{k\times 1}$.

The hardness of MLWR is determined by the Adv of **A** to solve decisional-MLWR:

$$\mathrm{Adv}^{\mathrm{MLWR}}_{n,q,p,k,\eta}(\mathbf{A}) = \Big|\Pr[\mathbf{b}=1 | A \leftarrow_r \mathfrak{R}_q^{k\times k}; b \leftarrow_r \mathfrak{R}_q^k; \mathbf{b} \leftarrow \mathbf{A}(A,b)] - \Pr[\mathbf{b}=1 | A \leftarrow_r \mathfrak{R}_q^{k\times k}; s \leftarrow S_\eta^k; b \leftarrow \lfloor \frac{p}{q} As \rceil \mathbf{b} \leftarrow \mathbf{A}(A, b = \lfloor \frac{p}{q} As \rceil)]\Big| < negl(n)$$

The proposed PAKE mainly aims to obtain an efficient PAKE solution, even if it has complex KEM structures, additional encryption, or primitives. To achieve, the currently proposed generic PAKE construction from a well-structured KEM named as OCAKE was selected.

Definition 3 (*OCAKE Construction [12]*)**.** OCAKE-based PAKE is one of the KEM to PAKE solutions in the literature that was built by considering the traditional one-way encrypted key exchange idea [38]. It stands out with its ability to provide explicit password-based authentication with single encryption. In addition to capture security in the relaxed model with static corruptions, it is a model to build efficient KEM to PAKE designs.

In the OCAKE construction, selected KEM needs to satisfy the fuzziness and anonymity properties that define the randomness of public keys and encapsulation, respectively.

Definition 4 (*Anonymity of a KEM [12,32]*)**.** The anonymity of a KEM scheme is defined by analyzing the randomness of ciphertext distribution. If the ciphertext distribution obtained by the encapsulation function of the KEM scheme is computationally indistinguishable from the uniform ciphertext distribution, it satisfies the anonymity property.

Definition 5 (*Fuzziness of a KEM [12,32]*)**.** Fuzziness is specified as a measure of the properties of the public key distribution. More precisely, if the distribution of public keys cannot be computationally distinguished from the uniform distribution, the KEM is said to be fuzzy.

2.2. Security model

In the literature, there have been used two different models, Bellare–Pointcheval–Rogaway (BPR) [39] and UC [40] to analyze the security of PAKE schemes. In the BPR model, game-based analysis was presented, which includes the adversary's protocol breaking capabilities. With the UC model, there was a simulation-based approach where the security guarantee is more certain. Due to the selected PAKE construction, in this paper, UC model is used to analyze the security of proposed PAKE to present strictly better guarantees. Let us define UC-related security terms [12,40].

- $\prod$-**A**-§-Ŕ-**C**-**S**: Protocol-Adversary-Simulation-Role component-Client-Server
- **F**: Ideal functionality is considered as a honest trusted party that unconditionally answers to queries.
- sid-stt: Session identifier-Status component
- s-sid = (sid, s-sid): Unique sub-session identifier
- $\mathbb{L}$: A List for session record, where $\mathbb{L}$ = (s-sid, $\mathbf{C}_i$, $\mathbf{S}_j$, pw, stt, Ŕ)
- Real world: The execution of the protocol are run between parties in the presence of an **A**.
- Ideal world: Dummy actors and an ideal adversary/simulator interact only with an **F** to determine the output of special function.

The main aim of UC analysis is to imitate a protocol ($\prod$) by utilizing ideal functionality (**F**). If the emulation cannot distinguish the outputs of protocol in the presence of feasible adversary interactions from the outputs of dummy actors and simulator (§) in the interaction of ideal functionality, $\prod$ is considered UC-emulation functionality.

Three different queries describe the ideal functionality of a PAKE protocol [40].

- **new-session:** It allows one of the communicating parties to initiate a connection with the other party utilizing a shared password. The built connection and password of the first party are transcribed by using this query in the functionality.
- **test-pw:** Online dictionary attacks are modeled with this query. When **test-pw** is queried, it also changes the appearance of ideal functionality during the key exchange, as the behavior of the next query is altered based on guessing the correct or incorrect password.
- **new-key:** This query, modeled as an interface, allows the connected parties to be given session keys consistent with their records if the fresh parties utilize the same password.

The explicit authentication of the server is analyzed under the password authentication-based ideal functionality assumptions described in Algorithm 1.

Algorithm 1 Ideal Functionality Definitions of a PAKE that Provide Explicit Server Authentication

procedure SESSION-INITIALIZATION
For (**new-session**, s-sid, Ŕ, pw, $\mathbf{C}_i$, $\mathbf{S}_j$) in $\mathbf{C}_i$;

- (**new-session**, s-sid, Ŕ, $\mathbf{C}_i$, $\mathbf{S}_j$) is sent to **A**.
- If this is the first or second **new-session** query and there is a record (s-sid, $\mathbf{C}_i$, $\mathbf{S}_j$, p̄w, ·, ·) ∈ $\mathbb{L}$, (s-sid, $\mathbf{C}_i$, $\mathbf{S}_j$, pw, **fresh**, Ŕ) is recorded to $\mathbb{L}$.

end procedure
procedure ACTIVE-ATTACK
If a record (s-sid, $\mathbf{C}_i$, $\mathbf{S}_j$, pw, **fresh**, Ŕ) ∈ $\mathbb{L}$ exists when **A** queries (**test-pw**, s-sid, $\mathbf{C}_i$, p̄w), the following reactions are done.

- If pw = p̄w, the record is marked as **compromised**. The answer is labeled as **correct-guess** and returned to **A**.
- Otherwise, the record is marked as **interrupted**. The answer is tagged as **incorrect-guess** and returned to **A**.

end procedure
procedure KEY-GENERATION
When a query (**new-key**, s-sid, $\mathbf{C}_i$, ssk) is received from §, where ssk ∈ {**shared-keys**}.

- In order to be the first **new-key** query for $\mathbf{C}_i$, there must be a record of the form (s-sid, $\mathbf{C}_i$, $\mathbf{S}_j$, pw, **stt**, Ŕ) for any **stt**.
  - If Ŕ=**C**,
    - If **stt=compromised**, or one of the parties $\mathbf{C}_i$ or $\mathbf{S}_j$ is corrupted, and there are two records such as (s-sid, $\mathbf{C}_i$, $\mathbf{S}_j$, pw, **C**) and (s-sid, $\mathbf{S}_j$, $\mathbf{C}_i$, pw, **S**), then (s-sid, ssk) values are sent to $\mathbf{C}_i$.
    - Else If **stt=fresh**,
      - For pw=p̄w, if there is a record (s-sid, $\mathbf{S}_j$, $\mathbf{C}_i$, p̄w, $\mathbf{C}'$) and as s̄s̄k has already been transferred to **fresh** $\mathbf{S}_j$, (s-sid, s̄s̄k) values are sent to $\mathbf{C}_i$.
      - For pw≠ p̄w, an s̄s̄k ←$_r$ {0,1}^{k} is chosen and (s-sid, s̄s̄k) are transferred to $\mathbf{C}_i$.
    - Else If **stt=fresh**,
      - For this s-sid, if there is no **completed** record for $\mathbf{S}_j$, nothing is done.
    - Else If **stt=interrupted**.
      - (s-sid, **error**) values are sent to $\mathbf{C}_i$.
  - If Ŕ=**S**,
    - If **stt=compromised**, or one of the parties $\mathbf{C}_i$ or $\mathbf{S}_j$ is corrupted.
      - (s-sid, ssk) values are sent to $\mathbf{C}_i$.
    - Else If **stt=fresh**, or **stt=interrupted**,
      - An s̄s̄k ←$_r$ {**shared-keys**} is chosen and (s-sid, s̄s̄k) are transferred to $\mathbf{C}_i$.

end procedure
The record is updated as **completed**.

- The role component, Ŕ= {**C**, **S**}, is added to the session-related records kept in $\mathbb{L}$.

- On the same session s-sid, if the client queries **new-key** at a time when the server is not still querying **new-key**, nothing is done.
- The client is cancelled regardless of the status if the parties, $\mathbf{C}_i$ and $\mathbf{S}_j$, do not share the same password.

In the security analysis, it is proved that the proposed PAKE is a UC-emulation in password-based functionality model. When examining the protocol's security, static corruptions are considered in the erasure model during the simulation, where the simulator knows which sides are corrupted. Note that in the theoretical security analysis, password security is not the main concern. In the PAKE protocols, passwords are assumed to be securely shared between the parties by adding some boundaries to the selection and usage of passwords and multi-factor authentication solutions.

3. Proposed SMAUG-T.PAKE

The proposed protocol focused on how a KEM scheme containing performance-efficient components can be converted into an efficient PAKE. For this purpose, SMAUG-T.KEM [11], whose security is defined under MLWE and MLWR assumptions, is chosen as the KEM scheme. To add password-based authentication, the OCAKE model [12], which allows efficient and explicit authentication with single encryption, is followed. This model requires the KEM on which it is based to provide anonymity and fuzziness properties. Before giving the main protocol flow, let us prove the anonymity and fuzziness of SMAUG-T.KEM in Lemma 1.

Lemma 1.

(i) We refer to [11] to check the detailed proofs of indistinguishability. Smaug.KEM on parameters $(n, p, q, k, \eta, hs, hr)$ holds this property if

$$\begin{aligned} Adv^{ind}_{SMAUG\text{-}T}(\mathbf{A}) \leq & Adv^{PRF}_{expandA}(t) + Adv^{MLWE}_{n,q,k,k,\eta,hs}(t) + \\ & Adv^{MLWR}_{n,q,p,k+1,k,\eta,hr}(t) \end{aligned} \tag{1}$$

(ii) To adapt OCAKE model into SMAUG-T.KEM, we proved that SMAUG-T.KEM satisfies the anonymity property in $\mathfrak{R}_p^k \times \mathfrak{R}_p$:

$$\begin{aligned} Adv^{ano}_{SMAUG\text{-}T}(\mathbf{A}) \leq & Adv^{PRF}_{expandA}(t) + Adv^{MLWE}_{n,q,k,k,\eta,hs}(t) + \\ & Adv^{MLWR}_{n,p,q,k+1,k,\eta,hr}(t) \end{aligned} \tag{2}$$

Proof. Let a public key sample of SMAUG-T, $(A^T | b^T)^T \leftarrow_r \mathfrak{R}_q^{(k+1)\times k}$, is given. By rewriting the ciphertext $c = (c_1^T, c_2) \in \mathfrak{R}_p^{k+1}$,

$$c = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \overbrace{\lfloor \frac{p}{q} \cdot \begin{bmatrix} A \\ b^T \end{bmatrix} \cdot r \rceil}^{MLWR} + \frac{p}{t} \cdot \begin{bmatrix} 0 \\ \mu \end{bmatrix} \tag{3}$$

is obtained, where $t|p|q$. As long as the hardness of MLWR is satisfied, the distribution of c is computationally indistinguishable from uniformly random sample in $\mathfrak{R}_p^{k+1}$ since the added $\frac{p}{t} \cdot \begin{bmatrix} 0 \\ \mu \end{bmatrix}$ is also a random vector in $\mathfrak{R}_p^{k+1}$. □

(iii) Like anonymity, we also show that SMAUG-T.KEM provides fuzziness in $\mathfrak{R}_q^k$:

$$Adv^{fuz}_{SMAUG\text{-}T}(\mathbf{A}) \leq Adv^{MLWE}_{n,q,k,k,\eta,hs}(t) \tag{4}$$

Proof. As stated in Table 3, the public key of SMAUG-T.KEM is generated with SMAUG-T.PKE.KeyGen algorithm. Since $pk = (p, b)$, where $b = -A^T s + e \mod q$ and $(p, \tau) \leftarrow XOF(seed)$ is generated by following MLWE assumption, the distribution of pk is computationally indistinguishable from uniformly random sample in $\{0,1\}^{256} \times \mathfrak{R}_q^k$. □

After showing the anonymity and fuzziness of SMAUG-T.KEM, the proposed OCAKE-based SMAUG-T.PAKE scheme is detailed in Fig. 1.

In the constructed PAKE, four sub-processes are performed on the client and server sides. In these processes, SMAUG-T.KEM procedures, given in Table 3, symmetric encryption and decryption, and two different hash functions are used to obtain password-authenticated key sharing with explicit authentication. Note that in the implementation, two symmetric encryption techniques, AES and Ascon, are used to capture the best efficiency. The step-by-step explanation of the proposed PAKE can be summarized as follows.

- Process $\mathbf{C}_0$: The public key and secret key are determined by using SMAUG-T.KEM key generation, defined in Table 3. s-sid$\|pw_{\mathbf{C}}$ concatenation and pk components are provided as the actual inputs of symmetric encryption. Finally, password encrypted pk' is sent to the server.
- Process $\mathbf{S}_0$: Firstly, server decrypts pk' to obtain actual pk by using s-sid and pw_C. Then, the recovered pk is used to obtain ciphertext ct and capsulated key K with the help of SMAUG-T's encapsulation procedure. Ultimately, server generates a key, $\bar{K}$ as an authentication tag and sends $(ct, \bar{K})$ pairs to the client.
- Process $\mathbf{C}_1$: As soon as the client receives the parameters from the server, it generates the key K' by recovering the ciphertext with SMAUG-T's decapsulation procedure. Then, it generates the authentication check component with $\bar{K}' = H_1(\text{s-sid}\|\mathbf{C}\|\mathbf{S}\|pw_{\mathbf{C}}\|pk'\|ct\|K')$. Finally, client computes it is shared key with the help of computed $\{\text{s-sid}, \mathbf{C}, \mathbf{S}, pk', ct, \bar{K}', K'\}$ if $\bar{K}' == \bar{K}$ is hold.
- Process $\mathbf{S}_1$: Server generates its shared key by using $\{\text{s-sid}, \mathbf{C}, \mathbf{S}, pk', ct, \bar{K}, K\}$.

The most fundamental components that affect the correctness of the proposed scheme are $\{K', K\}$ such that $K' = K$. As seen in Fig. 1, these components are generated by using SMAUG-T's encapsulation and decapsulation procedures that were remembered in Table 3. In [11], the decryption failure of SMAUG-T.KEM is defined with $\delta = \Pr[\|r^T \cdot e + s^T \cdot e_1 + e_2\|_\infty < \frac{q}{2t}]$ probability. The analysis showed that if the parameter set is selected by considering this condition, SMAUG-T.KEM and SMAUG-T.PAKE will be run correctly with less than δ probability.

4. Security analysis

In constructing SMAUG-T.PAKE, the explicitly authenticated generic OCAKE model is integrated into the SMAUG-T.KEM algorithm to generate an efficient password-authenticated version. OCAKE structure assumes the adversary can obtain the party's password before the execution. So, the simulator is aware of which party is broken or corrupted. The semantic security is ensured in the UC model with static corruptions if the underlying KEM provides fuzziness, indistinguishability, and anonymity.

The security of the SMAUG-T.PAKE model is analyzed under the password authenticated-based ideal functionality assumptions, defined in Algorithm 1 by making adaptations to MLWE+MLWR assumptions. The advantage of an adversary against OCAKE-based SMAUG-T.PAKE is analyzed in Theorem 1.

Theorem 1. *Let (Enc, Dec) and (H_1, H_2) be ideal ciphers and random oracle pairs, respectively. Let m_E and m_D be the maximum query numbers for encryption (Enc) and decryption (Dec) oracles and m_S as the number of sessions. The semantic security of the proposed SMAUG-T.PAKE in the UC model is specified by Eq. (5), based on the fuzziness, anonymity, and indistinguishability properties of underlying KEM.*

$$\begin{aligned} Adv^{SMAUG\text{-}T.KEM}_{SMAUG\text{-}T.PAKE}(\mathbf{A}) \leq & Adv^{SMAUG\text{-}T.KEM}_{fuzz}(t) \cdot (m_D + m_S) + \\ & Adv^{SMAUG\text{-}T.KEM}_{ano}(t) \cdot m_D + \\ & Adv^{SMAUG\text{-}T.KEM}_{ind}(t) \cdot (m_S + m_D + 1) + \\ & (m_{H_1} + m_{H_2}) \cdot m_S \cdot 2^{-n} + \\ & m_E^2 \cdot 2^{-\kappa} + m_{H_1} \cdot 2^{-n} \end{aligned} \tag{5}$$

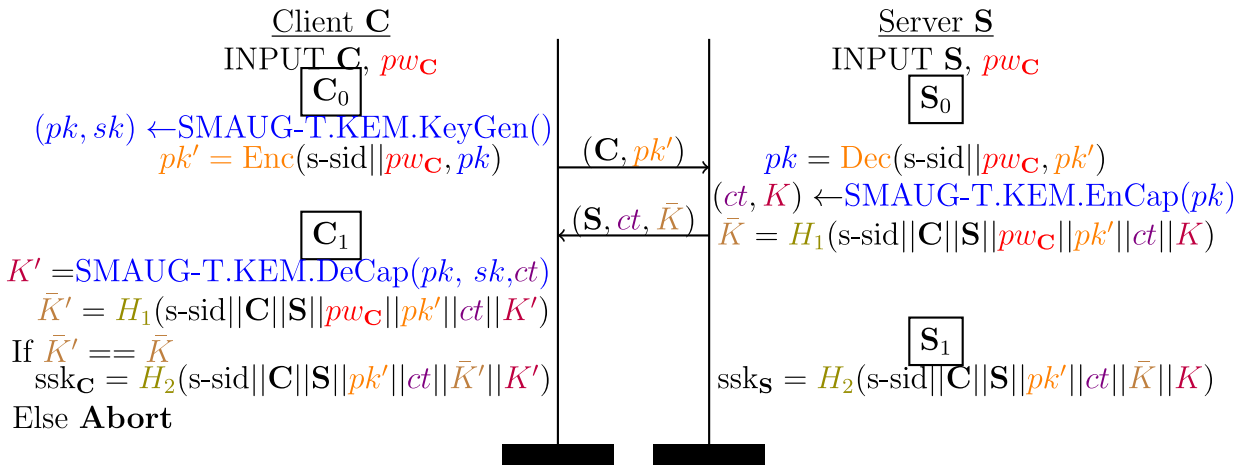

Fig. 1. Proposed OCAKE-based SMAUG-T.PAKE scheme.

Proof. For the sketch of proof, the simulated game sequence Game_i is defined, where $i = \{0, \ldots, 8\}$. The game series starts with real game Game_0 and ends with ideal game Game_8 that uses only its ideal functionality defined in Algorithm 1.

- **Game**$_0$: Game_0 that defines the real-world protocol is identified by considering indistinguishable, anonymous, and fuzzy KEM under the random oracle, erasure, ideal cipher, and static corruption assumptions.
- **Game**$_1$: The ideal cipher and two random oracle simulations are modeled with this game. Let $Pr[\text{Game}_1]$ be the probability of the environment for outputting 1 in the simulated Game_1. Under the assumptions of random oracles H_1 and H_2 and ideal cipher Enc, given in Algorithm 1, the environment can differentiate the real protocol execution from Game_1 when § aborts. The probability of environment in Game_1 is given in Eq. (6).

$$|\Pr[\text{Game}_1] - \Pr[\text{Game}_0]| \leq m_E^2 \cdot 2^{-\kappa-1} + m_{H_1}^2 \cdot 2^{-n-1} \tag{6}$$

- **Game**$_2$: The random secret keys are embedded during the simulation of decryption oracle Dec. So, the probability of environment in Game_2 is associated with fuzziness property of underlying KEM, proved in Lemma 1, since the difference of real and random public key is defined with the this property and given in Eq. (7).

$$|\Pr[\text{Game}_2] - \Pr[\text{Game}_1]| \leq \text{Adv}_{\text{fuzz}}^{\text{SMAUG-T.KEM}}(t) \cdot m_D \tag{7}$$

- **Game**$_3$: In Game_3, the adversary's capacity to estimate $\bar{K}$ without asking the correct query to H_1 is modeled. If this case happens, the § will be cancelled. So, the probability of environment in Game_3 is defined by Eq. (8).

$$|\Pr[\text{Game}_3] - \Pr[\text{Game}_2]| \leq m_S \cdot 2^{-n} \tag{8}$$

- **Game**$_4$: In Game_4, the client's initialization is simulated by utilizing Dec rather than Enc. In the simulation, § selects a random $pk' \leftarrow_r 2^{|E|}$, requests $pk = \text{Dec}(\text{s-sid}||pw_{\mathbf{C}}, pk')$, and forwards pk' to server. These changes do not reveal any difference from Game_3, the probability of environment in Game_4 is defined by Eq. (9).

$$|\Pr[\text{Game}_4] - \Pr[\text{Game}_3]| = 0 \tag{9}$$

- **Game**$_5$: In Game_5, server's answer, $(ct, \bar{K})$, is simulated. In the simulation, § simulates the answer of the honest server when it gets pk'. Since pk' can come either from an honest client or an adversary who corrupts the client, §'s behavior is determined accordingly. Since no difference from the previous game can be achieved in either scenario, the probability of environment in Game_5 is defined by Eq. (10).

$$|\Pr[\text{Game}_5] - \Pr[\text{Game}_4]| = 0 \tag{10}$$

- **Game**$_6$: In Game_6, client's second reaction is simulated. In the simulation, § simulates the answer of honest client when it gets $(ct, \bar{K})$. Since these components can come either from an honest server or from an adversary who corrupts the server, §'s behavior is determined accordingly. In the first case, there is no difference from Game_5 since the honest version is evaluated. In the second scenario, the cancellation situation will occur due to the $\bar{K}' \neq \bar{K}$. So, there will be no difference from Game_5, and the probability of environment in Game_6 is presented in Eq. (11).

$$|\Pr[\text{Game}_6] - \Pr[\text{Game}_5]| = 0 \tag{11}$$

- **Game**$_7$: In Game_7, ciphertext (ct), authentication tag components (K, K'), and shared keys ($\{\text{ssk}_{\mathbf{C}}, \text{ssk}_{\mathbf{S}}\}$) are replaced with random values. Three situations arise.
  - Randomization of ct:
    - By utilizing indistinguishability of the SMAUG-T.KEM, $\text{Adv}_{\text{ind}}^{\text{SMAUG-T.KEM}}(t) \cdot m_D$ is defined the bound in the server's computations since pk comes from decryption Dec is obtained.
    - On the client's part, the indistinguishability of the Smaug.KEM builds the bound without any other components due to the randomization of K'. So, the bound is $\text{Adv}_{\text{ind}}^{\text{SMAUG-T.KEM}}(t)$.
  - Randomization of authentication checks: The environment with random components can fail the game in two ways.
    - Server's side: ct is selected randomly from ciphertext distribution instead of $(ct, K) \leftarrow$ SMAUG-T.KEM.EnCap(pk) computation. Since this case is also associated with anonymity, **A** can distinguish this simulation by querying m_D times decryption Dec to break the anonymity. So, $\text{Adv}_{\text{ano}}^{\text{SMAUG-T.KEM}}(t) \cdot m_D$.
    - Client's side: Since the simulator can try to query H_1 and K was truly random in the previous game, $m_{H_1} \cdot m_S \cdot 2^{-n}$ is obtained.
  - Randomization of shared keys ($\text{ssk}_{\mathbf{C}}, \text{ssk}_{\mathbf{S}} \leftarrow_r \{0,1\}^{\kappa}$): The only way the environment can detect the difference is through H_2 oracle calls. Since K is truly random in the previous game and there are at most m_S changes, $m_{H_2} \cdot m_S \cdot 2^{-n}$ is procured.

Finally, the probability of environment in Game_7 is defined by Eq. (12).

$$\begin{aligned}|\Pr[\text{Game}_7] - \Pr[\text{Game}_6]| \leq & \text{Adv}_{\text{ind}}^{\text{SMAUG-T.KEM}}(t) \cdot (m_D + 1) + \\ & \text{Adv}_{\text{ano}}^{\text{SMAUG-T.KEM}}(t) \cdot m_D + \\ & (m_{H_1} + m_{H_2}) \cdot m_S \cdot 2^{-n}\end{aligned} \tag{12}$$

- **Game_8** : In Game_8, the ideal world is modeled by adding ideal functionality assumptions. According to Algorithm 1, there are two possible fresh session cases.
  - If honest parties use the same password, **stt** = **success** will allow the same session keys to be obtained.
  - When honest parties use the same password, **abort** in ideal functionality is achieved. A random session key is returned due to the **stt** = **fail** situation.

Finally, the total bound of the environment is given in Eq. (13).

$$\begin{aligned}|\Pr[\text{Game}_8] - \Pr[\text{Game}_0]| = & \\ & |\Pr[\text{Game}_8] - \Pr[\text{Game}_7]| + \\ & |\Pr[\text{Game}_7] - \Pr[\text{Game}_6]| + \\ & |\Pr[\text{Game}_6] - \Pr[\text{Game}_5]| + \\ & |\Pr[\text{Game}_5] - \Pr[\text{Game}_4]| + \\ & |\Pr[\text{Game}_4] - \Pr[\text{Game}_3]| + \\ & |\Pr[\text{Game}_3] - \Pr[\text{Game}_2]| + \\ & |\Pr[\text{Game}_2] - \Pr[\text{Game}_1]| + \\ & |\Pr[\text{Game}_1] - \Pr[\text{Game}_0]| \\ \leq & \text{Adv}_{\text{fuzz}}^{\text{SMAUG-T.KEM}}(t) \cdot (m_D + m_S) + \\ & \text{Adv}_{\text{ano}}^{\text{SMAUG-T.KEM}}(t) \cdot m_D + \\ & \text{Adv}_{\text{ind}}^{\text{SMAUG-T.KEM}}(t) \cdot (m_S + m_D + 1) + \\ & (m_{H_1} + m_{H_2}) \cdot m_S \cdot 2^{-n} + \\ & m_E^2 \cdot 2^{-\kappa} + m_{H_1} \cdot 2^{-n} \quad \square\end{aligned} \tag{13}$$

5. Implementation details and discussion

In this section, performance comparison results are presented with reference and mobile implementations to show the effectiveness of the proposed PAKE.

5.1. Performance analysis of the proposed PAKE

For the performance analysis of constructed generic PAKE, we optionally adapted Ascon or AES instead of external encryption to check the different performance options. By using reference C codes of SMAUG-T.KEM and making adaptations to explicit generic PAKE additions, the implementation of SMAUG-T.PAKE is written in C and can be found in [41]. Performance results are obtained on a computer with 32 GB RAM and a hexa-core (6 Core) AMD Ryzen 5 5500 processor running at 3.60 GHz. Processor cycles (median and average) and runtime results are determined by averaging 10,000 runs. For three different security levels, performance results are obtained according to the parameter set in Table 4. The reference SMAUG-T.PAKE implementation results in terms of two-different encryption methods are presented in Table 5.

Table 5 shows that, as expected, lightweight Ascon presents better results for three different security levels when used as the encryption method. Therefore, it is recommended that SMAUG-T.PAKE at Ascon mode can be suitable for resource-constrained devices, while in other applications both AES and Ascon can be used.

We also analyzed how much the KEM selected in the PAKE design model affects the performance. We integrated Kyber instead of the SMAUG-T functions used in the reference SMAUG-T.PAKE and presented the comparative results in Table 6. The codes are written in C and given in [41].

Table 6 implies that in both encryption modes, the SMAUG-T.PAKE scheme gives better results in terms of CPU cycle and runtime than the generic Kyber.PAKE scheme since SMAUG-T.PAKE benefits from the efficient algebraic structure of SMAUG-T.KEM.

We also compared the performance of proposed PAKE with the lattice-based PAKE protocols in the literature and whose accessible codes were found. In order for the comparison to be meaningful, one-stage PAKE protocols were chosen based on the same hard lattice problem. Since the main security of SMAUG-T.PAKE is based on MLWE+MLWR problems, MLWR-based Saber.PAK.PAKE [25], MLWE-based MLWE.PAK.PAKE [23], MLWE-based Kyber.PAK.PAKE [30] are selected to determine the efficiency of PAKE construction models.

The performance results of these schemes are obtained by using the provided C codes, [23,25,30] and run on the same computer. The performances of selected module-based PAKEs with lattice assumptions are evaluated in terms of running times and consumed CPU cycles and are given in Table 7.

Table 7 shows that the proposed SMAUG-T.PAKE shows the best results among other module-based schemes, even if it has additional ideal cipher usage and KEM components. The reason for this is the KEM selection, which consists of efficient arithmetic operations, and the generic model in the PAKE design.

5.2. Case scenario: Reference implementation on mobile environment

The efficient implementation of SMAUG-T.PAKE shows that it can be one of the best options for providing post-quantum secure PAKE for mobile environments. To analyze mobile compatibility, the JAVA codes of SMAUG-T.PAKE is also generated [41]. A computer with 32 GB RAM and a hexa-core AMD Ryzen 5 5500 processor running at 3.60 GHz is used as a server while Samsung Galaxy A51 (8 Cores) with 4x 1.7 GHz ARM Cortex-A53 397 co-processor with 2.3 GHz and 4x 2.3 GHz ARM Cortex-A73 main processor is used as mobile device. The mobile performance results are obtained in terms of running time, memory and CPU usages and presented in Table 8.

To compare the efficiency of these mobile-based one round PAKE schemes, [30] is also examined under the same operating conditions and the results are presented in Table 9 and visualized in Fig. 2. In Fig. 2, mobile device performance metrics such as CPU usage, energy consumption and memory usage are analyzed using the integrated Android Profiler in Android Studio. Each scheme is run in real-time on a mobile device, and the collected performance data is visually recorded. These recordings are obtained through the graphical interface provided by Android Profiler, capturing the execution of each stage over a specified time interval.

Table 9 and Fig. 2 show that for the mobile environment, the AES version of SMAUG-T.PAKE has the best performance. In other application results, it is known that the Ascon module comes to the fore in performance. This result is explained by the fact that AES consists of mobile-optimized generic codes in the JAVA library. Ascon's JAVA codes were not optimized for mobile, resulting in lower performance. As a result, the recommended SMAUG-T.PAKE is the PAKE that provides the best efficient application results for the mobile world.

5.3. Discussion

The main focus of this paper is the design of a new PQC PAKE protocol using a combination of different cryptographic principles, such as KEM and encryption/decryption primitives. It is aimed to evaluate the post-quantum security and efficiency effects of different primitives in the post-quantum secure PAKE design. Contributions are made to the design of the post-quantum secure PAKE protocol, theoretical security analysis, and practical performance results. In order to make a fair

Table 4
Parameter set.

Scheme		Saber.PAK.PAKE [25]			MLWE.PAK.PAKE [23]			Kyber.PAK.PAKE [30]			SMAUG-T.PAKE		
Security level		128	192	256	116	177	239	128	192	256	128	192	256
Module dimension	**k**	2	3	4	2	3	4	2	3	4	2	3	5
Lattice dimension	**n**	256	256	256	256	256	256	256	256	256	256	256	256
Module	**q**	8192	8192	8192	7681	7681	7681	3329	3329	3329	1024	2048	2048
	p	1024	1024	1024	x	x	x	x	x	x	256	256	256
Distribution parameter	η	10	8	6	13	8	6	x	x	x	x	x	x
	η_1	x	x	x	x	x	x	3	2	2	x	x	x
	η_1	x	x	x	x	x	x	2	2	2	x	x	x
Failure rate	δ	2^{-120}	2^{-136}	2^{-165}	$2^{-53.4}$	$2^{-97.4}$	$2^{-131.6}$	2^{-131}	2^{-164}	2^{-174}	2^{-120}	2^{-136}	2^{-167}

Table 5
Performance results of SMAUG-T.PAKE.

Security level		128						192						256					
Cipher option		ASCON			AES			ASCON			AES			ASCON			AES		
Metrics		M	A	ET	M	A	ET	M	A	ET	M	A	ET	M	A	ET	M	A	ET
Phases	$\mathbf{C}_0$	67 571	68 345	17 981	95 975	97 996	26 083	115 703	110 159	31 360	164 879	168 750	45 128	207 971	209 150	58 048	283 031	284 934	78 111
	$\mathbf{S}_0$	70 991	71 489	18 857	117 359	117 824	31 730	110 159	110 602	29 717	199 223	200 136	52 988	218 879	220 093	61 119	350 675	352 776	96 950
	$\mathbf{C}_1$	91 367	91 825	24 507	90 683	90 859	24 234	141 299	142 489	38 547	141 263	141 123	44 128	259 739	261 757	72 709	259 523	261 267	71 542
	$\mathbf{S}_1$	12 491	12 794	2554	12 491	12 484	2468	17 891	18 004	3996	17 891	17 917	3977	26 927	27 118	7533	26 891	27 048	6509
Total **C**		158 938	160 170	42 488	186 658	188 855	50 317	257 002	252 648	69 907	306 142	309 873	89 256	467 710	470 907	130 757	542 554	546 201	149 653
Total **S**		83 482	84 283	21 411	129 850	130 308	34 198	128 050	128 606	33 713	217 114	218 053	56 965	245 806	247 211	68 652	377 566	379 824	103 459
Total		**242 420**	**244 453**	**63 899**	316 508	319 163	84 515	**385 052**	**381 254**	**103 620**	523 256	527 926	146 221	**713 516**	**718 118**	**199 409**	920 120	926 025	253 112

Table 6
Generic Kyber.PAKE vs. Proposed generic SMAUG-T.PAKE.

Security level		*128*						*192*						*256*					
Ideal Cipher		*ASCON*			*AES*			*ASCON*			*AES*			*ASCON*			*AES*		
Metrics		*M*	*A*	*ET*	*M*	*A*	*ET*	*M*	*A*	*ET*	*M*	*A*	*ET*	*M*	*A*	*ET*	*M*	*A*	*ET*
Generic Kyber.PAKE	Total **C**	299 374	300 007	83,336	299 590	300 541	83,484	460 618	466 529	129,592	466 314	467 893	129,961	669 562	670 785	186,329	673 522	675 909	187,752
	Total **S**	180 610	180 929	50,259	189 682	190 498	52,916	268 846	272 226	75,619	285 370	286 199	79,500	384 874	385 520	107,089	405 214	406 273	112,854
	Total	479 984	480 936	133,595	489 272	491 039	136,400	729 464	738 755	205,211	751 684	754 092	209,461	1054 436	1056 305	293,418	1078 736	1082 182	300,606
SMAUG-T.PAKE	Total **C**	158 938	160 170	42,488	186 658	188 855	50,317	257 002	252 648	69,907	306 142	309 873	89,256	467 710	470 907	130,757	542 554	546 201	149,653
	Total **S**	83 482	84 283	21,411	129 850	130 308	34,198	128 050	128 606	33,713	217 114	218 053	56,965	245 806	247 211	68,652	377 566	379 824	103,459
	Total	**242 420**	**244 453**	**63,899**	316 508	319 163	84,515	**385 052**	**381 254**	**103,620**	523 256	527 926	146,221	**713 516**	**718 118**	**199,409**	920 120	926 025	253,112

Table 7
A performance comparison for module-based PAKE schemes.

Security Level	Metrics	Saber.PAK.PAKE [25]			MLWE.PAK.PAKE [23]			Kyber.PAK.PAKE [30]			Generic Kyber.PAKE						SMAUG-T.PAKE					
											AES			*ASCON*			*AES*			*ASCON*		
		C	**S**	*Total*	**C**	**S**	*Total*	**C**	**S**	*Total*	**C**	**S**	*Total*	**C**	**S**	*Total*	**C**	**S**	*Total*	**C**	**S**	*Total*
128	M	186 226	116 314	*302 540*	207 784	217 561	*425 345*	314 566	172 690	*487 256*	299 590	189 682	*489 272*	299 374	180 610	*479 984*	186 658	129 850	*316 508*	158 938	83 482	**242 420**
	A	188 351	117 389	*305 740*	208 697	218 861	*427 558*	315 632	173 564	*489 196*	300 541	190 498	*491 039*	300 007	180 929	*480 936*	188 855	130 308	*319 163*	160 170	84 283	**244 453**
	ET	52,320	32,559	*84,879*	62,549	61,921	*124,470*	87,677	48,213	*135,890*	83,484	52,916	*136,400*	83,336	50,259	*133,595*	50,317	34,198	*84,515*	42,488	21,411	**63,899**
192	M	309 652	184 678	*494 330*	319 104	318 672	*637 776*	498 994	271 654	*770 648*	466 314	285 370	*751 684*	460 618	268 846	*729 464*	306 142	217 114	*523 256*	257 002	128 050	**385 052**
	A	310 952	185 318	*496 270*	319 321	318 349	*637 670*	504 388	273 281	*777 669*	467 893	286 199	*754 092*	466 529	272 226	*738 755*	309 873	218 053	*527 926*	252 648	128 606	**381 254**
	ET	86,376	51,478	*137,854*	84,230	84,927	*169,157*	140,130	75,912	*216,042*	129,961	79,500	*209,461*	129,592	75,619	*205,211*	89,256	56,965	*146,221*	69,907	33,713	**103,620**
256	M	465 478	267 838	*733 316*	449 640	428 235	*877 875*	696 490	373 030	*1069 520*	673 522	405 214	*1078 736*	669 562	384 874	*1054 436*	542 554	377 566	*920 120*	467 710	245 806	**713 516**
	A	469 478	270 625	*740 103*	445 296	422 208	*867 504*	699 656	374 171	*1073 827*	675 909	406 273	*1082 182*	670 785	385 520	*1056 305*	546 201	379 824	*926 025*	470 907	247 211	**718 118**
	ET	130,411	75,174	*205,585*	119,605	117,116	*236,721*	194,349	103,937	*298,286*	187,752	112,854	*300,606*	186,329	107,089	*293,418*	149,653	103,459	*253,112*	130,757	68,652	**199,409**

comparison, PAKE protocols defined on the same algebraic structure, module, were selected even though they contain different design ideas.

To highlight the performance differences, firstly, the properties of selected module-based PAKEs are presented in Table 10. The main characteristics of those schemes that reveal the differences regarding performances appear to be due to the design model, main security, reconciliation idea, and additional primitives such as ideal cipher and KEM usage.

As summarized in Table 10, the PAKE protocol design incorporates different components and design ideas. This leads to different scenarios in both security analysis and performance evaluations. While it would be ideal to compare protocols using the same design idea and methodology, each protocol often differs from the other due to the targeted additional features. For example, the compromise structure, the difficulty problem, and additional principles used lead to different evaluations. In this paper, we contribute to the literature by defining security analysis in the UC model using hybrid hardness assumptions for KEM to PAKE design methods, and by evaluating performance in different encryption modes. We present in-depth security analysis evaluations by updating the defined methods under different security assumptions. We also evaluate usability and applicability by implementing the results in different modes on different platforms. We present theoretical and practical analyses on whether security and

Table 8
Mobile implementation results of SMAUG-T.PAKE.

Security levels	Used encryption	AES						ASCON					
	Metrics	$\mathbf{C}_0$	$\mathbf{S}_0$	$\mathbf{C}_1$	$\mathbf{S}_1$	*Total* **C**	*Total* **S**	$\mathbf{C}_0$	$\mathbf{S}_0$	$\mathbf{C}_1$	$\mathbf{S}_1$	*Total* **C**	*Total* **S**
128	ET	640,772	611,31	491,996	76,133	**1132,768**	**687,443**	994,023	1032,032	527,931	82,921	**1521,954**	**1114,953**
	MU	58,2	47,2	57,4	3,5	**115,6**	**50,7**	337,2	344,4	57,3	3,5	**394,5**	**347,9**
	CPU	8%	6%	8%	2%	**16%**	**8%**	12%	10%	8%	2%	**20%**	**12%**
192	ET	826,574	766,036	710,122	105,035	**1536,696**	**871,071**	1495,424	1426,737	732,831	107,363	**2228,255**	**1534,1**
	MU	93,1	81,5	96,5	4,9	**189,6**	**86,4**	481,8	408,6	96,7	4,9	**578,5**	**413,5**
	CPU	9%	8%	10%	3%	**19%**	**11%**	16%	14%	11%	3%	**27%**	**17%**
256	ET	1111,587	1100,146	1115,541	146,240	**2227,128**	**1246,386**	2326,828	2363,742	1225,591	157,287	**3552,419**	**2521,029**
	MU	185,1	174,5	198,3	7,2	**383,4**	**181,7**	840,1	690,1	198,5	7,2	**1038,6**	**697,3**
	CPU	12%	10%	13%	4%	**25%**	**14%**	25%	21%	12%	4%	**37%**	**25%**

–ET: Elapsed time in microsecond –MU: Memory usage in kilobayt –CPU: CPU usage
–The source codes is given in [41].

Table 9
A comparison for lattice-based PAKE schemes for mobile environment.

Security levels	*Used encryption*	Kyber.PAK.PAKE [30]			SMAUG-T.PAKE					
		x			*AES*			*ASCON*		
	Metrics	Total **C**	Total **S**	*Total*	Total **C**	Total **S**	*Total*	Total **C**	Total **S**	Total
128	*ET*	1645,489	1249,074	2894,563	**1088,904**	**680,411**	**1768,508**	1448,841	1089,582	2538,423
	MU	274,5	90,2	364,7	**113,1**	**45,8**	**158,9**	264,3	89,1	353,4
	CPU	16%	17%	33%	**14%**	**8%**	**22%**	16%	17%	33%
192	*ET*	2015,363	1498,580	3513,943	**1517,179**	**879,821**	**2397,000**	2193,568	1591,124	3784,692
	MU	361,6	135,3	496,9	**186,1**	**84,4**	**270,5**	359,1	134,4	493,5
	CPU	22%	19%	41%	**17%**	**10%**	**27%**	22%	19%	41%
256	*ET*	2825,102	1965,160	4790,262	**2234,221**	**1282,773**	**3516,994**	2825,102	1965,16	4790,262
	MU	477,1	173,8	650,9	**377,9**	**178,6**	**556,5**	477,1	173,8	650,9
	CPU	25%	23%	48%	**22%**	**12%**	**35%**	25%	23%	48%

Table 10
The basic characteristics of module-based PAKEs.

	Saber.PAK.PAKE [25]	MLWE.PAK.PAKE [23]	Kyber.PAK.PAKE [30]	Generic Kyber.PAKE^{+}	Proposed generic Smaug.PAKE
Construction model	Traditional PAK PAKE [1] from well-structured KE	Traditional PAK PAKE [1]	Traditional PAK PAKE [1] from well-structured KEM	Generic PAKE from KEM with the usage of ideal cipher [12]	Generic PAKE from KEM with the usage of ideal cipher [12]
Password usage	Password is added as a component of public key to provide authentication.	Password is added as a component of public key to provide authentication.	Password is added as a component of public key to provide authentication.	The password is used as a parameter to encrypt public key and generate the shared key component to help authentication.	The password is used as a parameter to encrypt public key and generate the shared key component to help authentication.
Main security	MLWR	MLWE	MLWE	MLWE	MLWE+MLWR
Reconciliation	bits	OKCN	Compress-Decompress	Compress-Decompress	Rounding function
Additional Primitives	X	X	KEM	KEM Ideal Cipher	KEM Ideal Cipher
Ideal Cipher	X	X	X	AES ASCON	AES ASCON
Additional requirements	X	X	X	Anonymity fuzziness	Anonymity fuzziness
Number of Hash	4	4	$3+3^{*}$	$2+3^{*}$	$2+3^{*}$
Security model	ROM	ROM	ROM	UC	UC

*: The number of hash functions that were used as a component of KEM.$^{+}$: The implementation of generic Kyber PAKE is written in C to make a comparison.
OKCN: Optimally-balanced key consensus with noise.

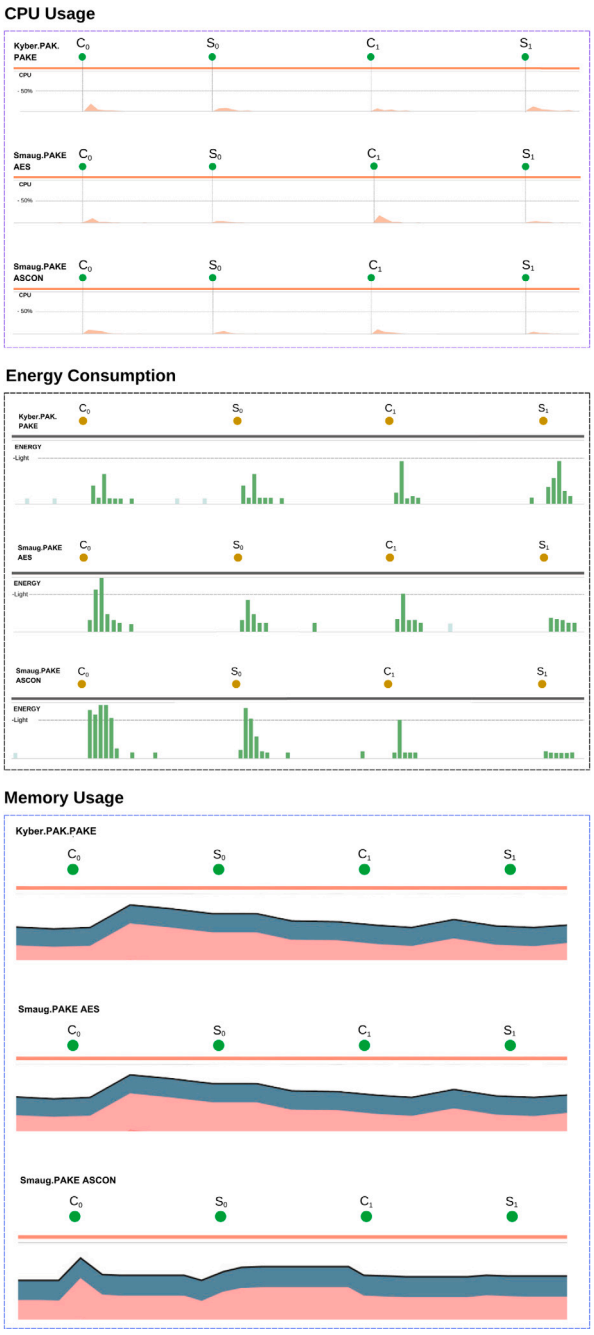

Fig. 2. Energy, CPU, and memory usage comparison diagrams for mobile compatible PAKEs.

performance objectives can be maintained while creating new cryptographic primitives from known efficient ones, with the design of the post-quantum secure PAKE protocol.

A comparison is made with protocols based on the hardness assumption of the MLWE and MLWR problem defined on the module algebraic structure. The performance analyses for five different PAKE protocols, based on the PAK-PAKE [1] and KEM-to-PAKE [12] design ideas, are presented in Table 7. Note that the performance evaluations were conducted only with lattice-based PAKE protocols for which the source code is available, since the source code for other KEM to PAKE frameworks [12,31–34] in the literature is not shared. For fair evaluation, all code was obtained by running on the platforms. The results show that, unlike [23,25,30], SMAUG-T.PAKE offers the best performance in the lightweight encryption mode, even if it includes additional encryption and hash functions.

To demonstrate the efficiency of the proposed model, we also examined the performance of generic PAKE using Kyber.KEM instead of SMAUG-T.KEM. The results in Table 7 show that SMAUG-T is more efficient than Kyber even when converted to PAKE. The security analysis presented in Section 4 demonstrates that the post-quantum security is maintained in the situation of KEM-to-PAKE transformation. The results presented in Table 9 also provide an analysis of the usability of lattice-based PAKE protocols on the mobile platform. These analyses show that the proposed PAKE can provide good performance in the standard algorithm mode even if it includes additional cryptographic primitives. While it is normally expected to provide more efficient results with lightweight algorithms such as Ascon, it has been evaluated that the reason why it provides more efficient results with AES is due to AES's optimizations in the library functions used in the application.

This paper primarily presents evidence for the properties of anonymity and fuzziness associated with the conversion of a cryptographic principle designed as an efficient KEM for the post-quantum world to PAKE. It then analyzes the post-quantum security with hybrid security definitions created in the UC model, where the PAKE transformation retains its post-quantum security. Subsequently, we present usability and applicability analyses, comparing application results with literature solutions on various platforms. In addition to the theoretical and practical focus, efficient hardware-based implementations can also be implemented. For example, hardware-based analyses and implementations for efficient implementations of PQC KEM algorithms have been carried out in studies such as [10,42–45]. Therefore, investigating more efficient versions of the SMAUG-T.PAKE protocol using different architectures and PUF-like hardware-based solutions will be conducted as future work.

6. Conclusion

The design of post-quantum secure key-sharing schemes is one of the challenging issues in the literature. The simple structured authentication idea of PAKE schemes in real-world scenarios has also revealed the need to ensure post-quantum security of PAKE protocols. Therefore, PAKE schemes, which stand out with their strong security and efficiency features, are one of the necessary primitives for the post-quantum security of resource-limited devices. In this paper, we construct an efficient PAKE adaptation from well-structured lattice-based KEM procedures and additional primitives. The constructed SMAUG-T.PAKE, benefits the underlying KEM's efficiency and simple structured KEM to PAKE construction. It is the first MLWE+MLWR-based KEM to PAKE design that provides explicit password-based anonymous and fuzziness authentication. Unlike the lattice-based PAKE protocols in the literature, the security analysis is performed under hybrid assumptions. MLWE+MLWR-based password-authenticated key ideal functionality under the UC model are constructed to analyze the security of the proposed PAKE. The detailed performance analysis with other module-based, KEM-based, and mobile-based PAKE schemes shows that the proposed PAKE provides the best results in terms of consumed CPU cycles and elapsed times, even if it contains additional encryption usage. To the best of our knowledge, the proposed SMAUG-T.PAKE is one of the best candidates for efficient password-based authentication for the post-quantum security of general purposes and mobile usage.

CRediT authorship contribution statement

Kübra Seyhan: Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Conceptualization. **Sedat Akleylek:** Writing – review & editing, Validation, Supervision, Project administration, Methodology. **Ahmet Faruk Dursun:** Writing – review & editing, Software.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Sedat Akleylek reports financial support was provided by Estonian Research Council. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to express their gratitude to the anonymous reviewers for their invaluable suggestions in putting the present study into its final form. Sedat Akleylek was supported by the Estonian Research Council Grant PRG2531 and Estonian Ministry of Defence Grant 2-2/24/541-1.

Data availability

No data was used for the research described in the article.

References

[1] P. MacKenzie, The PAK suite: Protocols for password-authenticated key exchange, Contrib. To IEEE P 1363 (2) (2002).

[2] F. Hao, P.C. van Oorschot, Sok: password-authenticated key exchange–theory, practice, standardization and real-world lessons, in: Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security, 2022, pp. 697–711, http://dx.doi.org/10.1145/3488932.3523256.

[3] J. Jiang, D. Wang, Qpase: Quantum-resistant password-authenticated searchable encryption for cloud storage, IEEE Trans. Inf. Forensics Secur. 19 (2024) 4231–4246.

[4] S.M. Bellovin, M. Merritt, Encrypted key exchange: Password-based protocols secure against dictionary attacks, 1992, http://dx.doi.org/10.7916/D8833ZSK.

[5] F. Hao, Prudent practices in security standardization, IEEE Commun. Stand. Mag. 5 (3) (2021) 40–47, http://dx.doi.org/10.1109/MCOMSTD.121.2100005.

[6] K. Seyhan, S. Akleylek, A comprehensive comparison of lattice-based password authenticated key exchange protocols defined on modules, in: International Conference on Information Technologies and their Applications, Springer, 2024, pp. 91–105, http://dx.doi.org/10.1007/978-3-031-73417-5_8.

[7] P.W. Shor, Algorithms for quantum computation: discrete logarithms and factoring, in: Proceedings 35th Annual Symposium on Foundations of Computer Science, IEEE, 1994, pp. 124–134, http://dx.doi.org/10.1109/SFCS.1994.365700.

[8] National Institute of Standards and Technology (NIST), NIST post-quantum cryptography standardization project, 2025, (Accessed: 24 October 2025) https://csrc.nist.gov/projects/post-quantum-cryptography.

[9] D. Ott, C. Peikert, et al., Identifying research challenges in post quantum cryptography migration and cryptographic agility, 2019, https://arxiv.org/abs/1909.07353.

[10] N. Alnahawi, D. Haas, E. Mauß, A. Wiesmaier, SoK: PQC PAKEs - design, security and performance, 2025, URL https://eprint.iacr.org/2025/119 Cryptology ePrint Archive, Paper 2025/119.

[11] J. Cheon, H. Choe, J. Choi, D. Hong, J. Hong, C. Jung, H. Kang, J. Lee, S. Lim, A. Park, S. Park, J. Seo, H. Seong, J. Shin, SMAUG-T: The Key Exchange Algorithm Based on Module-LWE and Module-LWR, Algorithm specifications, K-PQC Consortium, 2024, (Accessed: 24 October 2025).

[12] H. Beguinet, C. Chevalier, D. Pointcheval, T. Ricosset, M. Rossi, GeT a CAKE: Generic transformations from key encaspulation mechanisms to password authenticated key exchanges, in: International Conference on Applied Cryptography and Network Security, Springer, 2023, pp. 516–538, http://dx.doi.org/10.1007/978-3-031-33491-7_19.

[13] K. Seyhan, S. Akleylek, Smaug kem to smaug-pake: a generic lattice-based password authenticated key exchange, Central European Conference on Cryptology CECC-2024 (2024) 38-41.

[14] Z. Li, D. Wang, Achieving one-round password-based authenticated key exchange over lattices, IEEE Trans. Serv. Comput. 15 (1) (2019) 308–321, http://dx.doi.org/10.1109/TSC.2019.2939836.

[15] Z. Li, H. Zhu, G. Liao, M. Wang, P. Li, P. Gope, QT-PAKE: Secure messaging via quantum-safe threshold PAKE, IEEE Trans. Consum. Electron. (2025).

[16] J. Katz, V. Vaikuntanathan, Smooth projective hashing and password-based authenticated key exchange from lattices, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2009, pp. 636–652, http://dx.doi.org/10.1007/978-3-642-10366-7_37.

[17] J. Zhang, Y. Yu, Two-round PAKE from approximate SPH and instantiations from lattices, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2017, pp. 37–67, http://dx.doi.org/10.1007/978-3-319-70700-6_2.

[18] Z. Zhao, S. Ma, P. Qin, Password authentication key exchange based on key consensus for iot security, Clust. Comput. 26 (1) (2023) 1–12, http://dx.doi.org/10.1007/s10586-022-03665-5.

[19] L. Chen, T. Qu, A. Yin, Quantum-safe multi-server password-based authenticated key exchange protocol, Multimedia Tools Appl. 83 (24) (2024) 65011–65038, http://dx.doi.org/10.1007/s11042-023-17984-1.

[20] Z. Li, D. Wang, E. Morais, Quantum-safe round-optimal password authentication for mobile devices, IEEE Trans. Dependable Secur. Comput. 19 (3) (2020) 1885–1899, http://dx.doi.org/10.1109/TDSC.2020.3040776.

[21] A. Singh, H. Chandra, S. Rana, A robust lattice-based post-quantum three-party key exchange scheme for mobile devices, Concurr. Comput.: Pr. Exp. 37 (6–8) (2025) e70036, http://dx.doi.org/10.1002/cpe.70036.

[22] D. Mishra, K. Pursharthi, M. Singh, A. Mishra, Construction of post quantum secure authenticated key agreement protocol for dew-assisted IoT systems, Int. J. Inf. Secur. 24 (1) (2025) 19, http://dx.doi.org/10.1007/s10207-024-00932-x.

[23] P. Ren, X. Gu, Z. Wang, Efficient module learning with errors-based post-quantum password-authenticated key exchange, IET Inf. Secur. 17 (1) (2023) 3–17, http://dx.doi.org/10.1049/ise2.12094.

[24] R. Ding, C. Cheng, Y. Qin, Further analysis and improvements of a lattice-based anonymous pake scheme, IEEE Syst. J. 16 (3) (2022) 5035–5043, http://dx.doi.org/10.1109/JSYST.2022.3161264.

[25] K. Seyhan, S. Akleylek, A new password-authenticated module learning with rounding-based key exchange protocol: Saber. PAKE, J. Supercomput. 79 (16) (2023) 17859–17896, http://dx.doi.org/10.1007/s11227-023-05251-x.

[26] C. Liu, Z. Zheng, K. Jia, Q. You, Provably secure three-party password-based authenticated key exchange from RLWE, in: International Conference on Information Security Practice and Experience, Springer, 2019, pp. 56–72, http://dx.doi.org/10.1007/978-3-030-34339-2_4.

[27] V. Dabra, A. Bala, S. Kumari, LBA-PAKE: Lattice-based anonymous password authenticated key exchange for mobile devices, IEEE Syst. J. 15 (4) (2020) 5067–5077, http://dx.doi.org/10.1109/JSYST.2020.3023808.

[28] V. Dabra, S. Kumari, A. Bala, S. Yadav, SL3PAKE: simple lattice-based three-party password authenticated key exchange for post-quantum world, J. Inf. Secur. Appl. 84 (2024) 103826, http://dx.doi.org/10.1016/j.jisa.2024.103826.

[29] S. Guo, Y. Song, S. Guo, Y. Yang, S. Song, Three-party password authentication and key exchange protocol based on mlwe, Symmetry 15 (9) (2023) 1750, http://dx.doi.org/10.3390/sym15091750.

[30] K. Seyhan, S. Akleylek, A.F. Dursun, Password authenticated key exchange-based on kyber for mobile devices, PeerJ Comput. Sci. 10 (2024) e1960, http://dx.doi.org/10.7717/peerj-cs.1960.

[31] A. Arriaga, M. Barbosa, S. Jarecki, M. Škrobot, C'est très CHIC: A compact password-authenticated key exchange from lattice-based KEM, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2024, pp. 3–33, http://dx.doi.org/10.1007/978-981-96-0935-2_1.

[32] J. Pan, R. Zeng, A generic construction of tightly secure password-based authenticated key exchange, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2023, pp. 143–175, http://dx.doi.org/10.1007/978-981-99-8742-9_5.

[33] N. Alnahawi, J. Alperin-Sheriff, D. Apon, G.T. Davies, A. Wiesmaier, NICE-PAKE: On the security of KEM-based PAKE constructions without ideal ciphers, 2024, URL https://eprint.iacr.org/2024/1957 Cryptology ePrint Archive, Paper 2024/1957.

[34] J. Vos, S. Jarecki, C.A. Wood, C. Yun, S. Myers, Y. Sierra, A hybrid asymmetric password-authenticated key exchange in the random oracle model, 2025, URL https://eprint.iacr.org/2025/1343 Cryptology ePrint Archive, Paper 2025/1343.

[35] J. Ding, S. Alsayigh, J. Lancrenon, S. Rv, M. Snook, Provably secure password authenticated key exchange based on RLWE for the post-quantum world, in: Cryptographers' Track At the RSA Conference, Springer, 2017, pp. 183–204, http://dx.doi.org/10.1007/978-3-319-52153-4_11.

[36] J.-P. D'Anvers, A. Karmakar, S. Sinha Roy, F. Vercauteren, Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM, in: International Conference on Cryptology in Africa, Springer, 2018, pp. 282–305, http://dx.doi.org/10.1007/978-3-319-89339-6_16.

[37] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J.M. Schanck, P. Schwabe, G. Seiler, D. Stehlé, CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM, in: 2018 IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, 2018, pp. 353–367, http://dx.doi.org/10.1109/EuroSP.2018.00032.

[38] E. Bresson, O. Chevassut, D. Pointcheval, Security proofs for an efficient password-based key exchange, in: Proceedings of the 10th ACM Conference on Computer and Communications Security, 2003, pp. 241–250, http://dx.doi.org/10.1145/948109.948142.

[39] M. Bellare, D. Pointcheval, P. Rogaway, Authenticated key exchange secure against dictionary attacks, in: International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2000, pp. 139–155, http://dx.doi.org/10.1007/3-540-45539-6_11.

[40] R. Canetti, S. Halevi, J. Katz, Y. Lindell, P. MacKenzie, Universally composable password-based key exchange, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2005, pp. 404–421, http://dx.doi.org/10.1007/11426639_24.

[41] A.F. Dursun, Smaug.PAKE for Mobile Devices, 2025, (Accessed: 24 October 2025) https://github.com/afDursun/lattice-based-pakes.

[42] S. Aghapour, K. Ahmadi, M. Anastasova, R. Azarderakhsh, M. Mozaffari Kermani, PUF-dilithium: Design of a PUF-based dilithium architecture benchmarked on ARM processors, ACM Trans. Embed. Comput. Syst. 24 (2) (2025) 1–20, http://dx.doi.org/10.1145/3715328.

[43] K. Ahmadi, S. Aghapour, M.M. Kermani, R. Azarderakhsh, Efficient error detection cryptographic architectures benchmarked on FPGAs for montgomery ladder, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 32 (11) (2024) 2154–2158, http://dx.doi.org/10.1109/TVLSI.2024.3419700.

[44] A. Jati, N. Gupta, A. Chattopadhyay, S.K. Sanadhya, A configurable crystals-kyber hardware implementation with side-channel protection, ACM Trans. Embed. Comput. Syst. 23 (2) (2024) 1–25, http://dx.doi.org/10.1145/3587037.

[45] S. Aghapour, K. Ahmadi, M. Anastasova, M.M. Kermani, R. Azarderakhsh, PUF-kyber: Design of a PUF-based kyber architecture benchmarked on diverse ARM processors, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 43 (12) (2024) 4453–4462, http://dx.doi.org/10.1109/TCAD.2024.3399669.