Mesopotamian journal of Cybersecurity
Vol.5,No.3, **pp**. 1272–1291
DOI: https://doi.org/10.58496/MJCS/2025/066; ISSN: 2958-6542
https://mesopotamian.press/journals/index.php/cybersecurity

Research Article

A Multi-Factor Quantum-Resistant and Privacy-Preserving Authentication Protocol for Decentralized Systems

Ahmed Kh. Zamil[1,*], Ammar D. Jasim[2]

[1] *Department of Information and Communications Engineering, College of Information Engineering, Al-Nahrain University, Baghdad, Iraq.*

[2] *Department of Computer Network Engineering, College of Information Engineering, Al-Nahrain University, Baghdad, Iraq.*

ARTICLE INFO

Article History

Received 10 Jul 2025
Revised 06 Oct 2025
Accepted 14 Nov 2025
Published 14 Dec 2025

Keywords

Quantum-resistant authentication
Privacy-preserving protocols
Lattice-based cryptography
Multi-factor authentication
Decentralized authentication

ABSTRACT

This paper presents a Quantum-Resistant and Privacy-Preserving Authentication Protocol (PPAP) designed for decentralized systems. PPAP integrates lattice-based cryptographic primitives, particularly Module-Lattice Key Encapsulation Mechanisms (ML-KEM), addressing both quantum computing threats and stringent privacy requirements. The protocol introduces advanced identity masking techniques and a multi-factor authentication layer employing Cheon-Kim-Kim-Song (CKKS) homomorphic encryption, enabling secure and privacy-aware authentication that supports approximate knowledge-based factors such as biometrics and handwritten signatures. Rigorous formal analysis, including cryptographic proofs and automated verification using the Tamarin Prover, validates PPAP's resilience against classical and quantum adversaries, as well as common security threats such as replay attacks, impersonation, and credential linkage. Performance evaluations demonstrate the practical feasibility of PPAP, highlighting its computational efficiency and minimized communication overhead, suitable for diverse scenarios ranging from Internet of Things (IoT) and mobile environments to high-security infrastructure. Furthermore, the Time-Aware Predictive Access Model (TAPM) significantly optimizes authentication lookup complexity, demonstrating substantial empirical improvement. Future work will focus on further performance enhancements, integration of additional biometric modalities, and deployment in practical decentralized applications.

1. INTRODUCTION

Secure authentication remains a cornerstone of modern digital communications. Traditional authentication protocols, such as Kerberos and Transport Layer Security (TLS) handshakes, rely heavily on static credentials and centralized trust anchors like certificate authorities, which pose a single point of failure [1] . While effective in their era, these designs increasingly face two critical limitations: escalating privacy concerns and the growing threat posed by quantum computing.

The first challenge is privacy erosion. In many protocols, identity data, such as usernames or certificates, is transmitted in plaintext or can be inferred through side channels. This contradicts modern data protection principles, including data minimization, pseudonymization, and unlinkability, all of which are central to regulatory frameworks like the General Data Protection Regulation (GDPR). Users should be able to prove legitimacy without unnecessarily revealing or linking their identities [2].

Simultaneously, the cryptographic foundations of existing protocols face an existential threat from quantum computers. Algorithms such as Shor's [3] and Grover's [4] can break widely deployed public-key systems. To mitigate this, the cryptographic community, through initiatives like National Institute of Standards and Technology (NIST) Post-Quantum Cryptography Standardization, has advocated the adoption of lattice-based primitives like Cryptographic Suite for Algebraic Lattices (CRYSTALS-Kyber), which are resistant to both classical and quantum attacks [5].

In response to these challenges, this paper introduces the Quantum-Resistant and Privacy-Preserving Authentication Protocol (PPAP), an authentication framework designed for decentralized systems. PPAP leverages post-quantum cryptographic primitives, specifically the Module-Lattice Key Encapsulation Mechanism (ML-KEM) [6], to ensure resilience against both classical and quantum adversaries. The protocol further incorporates advanced identity masking techniques using keyed hash functions and session-specific nonces, thereby achieving unlinkable pseudonymity and protecting sensitive identity information throughout the authentication process. To establish trust while preserving user

*Corresponding author. Email: ahmed.zamil@nahrainuniv.edu.iq

privacy, PPAP introduces a multi-factor authentication layer based on the Cheon-Kim-Kim-Song (CKKS) homomorphic encryption scheme [7]. This enables privacy-preserving evaluation of approximate knowledge-based factors such as biometric traits or handwritten signatures, ensuring that sensitive inputs are never exposed in plaintext. Additionally, the protocol supports multiple authentication modes with single and dual identity mutual authentication, enabling flexible deployment across a range of distributed environments.

PPAP is formally analyzed using cryptographic proofs and verified through the Tamarin Prover [8], demonstrating resistance to impersonation, replay, and linkage attacks under both classical and quantum threat models. Extensive evaluation confirms the protocol's practical viability, achieving efficient authentication performance and supporting scalable, privacy-compliant operation. By addressing the key limitations in existing approaches, PPAP represents a robust step toward next-generation authentication systems in decentralized and privacy-sensitive domains.

1.1. Research Contribution

In this paper, we present the following primary contributions:

1. We design a peer-to-peer multi-factor authentication protocol that eliminates reliance on centralized authorities, enabling fine-grained trust in distributed environments. The protocol achieves strong credential unlinkability through identity masking and domain-specific registration nonces, preventing cross-domain correlation of credentials.
2. We integrate quantum-resistant and privacy-preserving cryptography, combining ML-KEM for post-quantum secure key encapsulation with CKKS homomorphic encryption to support privacy-preserving approximate matching of behavioral or biometric factors.
3. We propose a novel Unified Key Derivation (UKD) mechanism that deterministically generates both ML-KEM and CKKS key material from the same credential-bound seed, thereby simplifying key management and ensuring unlinkability.
4. We introduce a novel Time-Aware Predictive Access Model (TAPM) that leverages temporal access patterns to significantly optimize authentication lookup efficiency in decentralized systems, validated using real-world access data to demonstrate its practical effectiveness.
5. We conduct comprehensive experimental validation on heterogeneous platforms, including Intel x86-64 and Advanced RISC Machine (ARM) based processors, demonstrating the feasibility of our proposed protocol in both high-performance and resource-constrained environments. These results not only confirm the practicality of the proposed protocol but also provide insights into architectural trade-offs relevant for real-world deployments.

The remainder of this paper is structured as follows: Section 2 provides background information and reviews related work on privacy-preserving and post-quantum authentication protocols, highlighting existing limitations and establishing the context for PPAP. Section 3 presents preliminary definitions, cryptographic assumptions, and the adversarial model employed. Section 4 introduces the detailed system model, design goals, and the underlying architecture of the PPAP protocol. Section 5 describes the core authentication procedures, including multiple flexible authentication modes supported by PPAP, protocol operation in decentralized systems, the proposed multi-factor trust layer emphasizing the privacy-preserving approximate matching authentication factors embedded in a unified authentication process, and our proposed optimization for identity unmasking. Section 6 provides a rigorous security analysis, including formal verification results using the Tamarin Prover and provable security proofs based on established cryptographic assumptions. Section 7 discusses implementation details and evaluates the protocol's performance, assessing factors such as computational efficiency, communication overhead, and practical feasibility. Finally, Section 8 concludes the paper and outlines directions for future research.

2. BACKGROUND AND RELATED WORK

2.1 Background

Authentication protocols serve as the foundational layer of digital trust, enabling secure identification and interaction between entities in distributed systems. Conventional schemes such as TLS, OAuth2, and OpenID Connect often rely on centralized identity providers and classical public-key cryptography [9] [10] [11]. These protocols are increasingly insufficient in the face of modern security and privacy requirements. Two principal challenges have emerged: the threat of large-scale quantum computers to current cryptographic schemes, and the pressing need for privacy-preserving, decentralized identity authentication mechanisms.

Classical asymmetric algorithms like RSA and Elliptic Curve Digital Signature Algorithm (ECDSA) are vulnerable to quantum algorithms such as Shor's and Grover's, prompting the cryptographic community to advance post-quantum cryptography (PQC) through lattice-based primitives including Key Encapsulation Mechanisms (KEMs) and digital signature algorithms (DSAs) [12]. Concurrently, privacy regulations like the GDPR demand strict protection of personal

identifiers and advocate for principles such as data minimization, pseudonymization, and unlinkability [13]. These dual concerns necessitate novel authentication designs that are both quantum-safe and privacy-preserving.
The proposed PPAP protocol addresses these demands through a unique architecture that integrates lattice-based KEMs, dynamic identity masking, and a privacy-preserving multi-factor trust layer. To contextualize its novelty, we examine existing post-quantum and privacy-focused authentication schemes and highlight their limitations.

2.2 Related Work

Recent literature has proposed a variety of quantum-safe and privacy-preserving authentication mechanisms, yet several limitations persist that hinder their suitability for fully decentralized environments. Riva-Cambrin et al. [14] introduced a token-based system employing lattice-based KEMs and digital signatures, offering forward secrecy in constrained settings but relying on centralized authorities for token issuance. Fraile et al . [15] proposed a hybrid Elliptic-curve Diffie–Hellman (ECDH) and post-quantum KEM scheme enhancing TLS, but it remains certificate-dependent and lacks privacy guarantees. Yao et al. [16] extended Security Protocol and Data Model (SPDM) with post-quantum primitives for device authentication; however, their approach is inherently centralized, and hardware focused. Samandari and Gritti [17] adapted post-quantum KEMs to the Message Queuing Telemetry Transport (MQTT) protocol but required static credential storage without anonymization. Arjona et al. [18] leveraged homomorphic encryption over Classic McEliece for biometric authentication, yet their design exposed hashed identifiers and lacked unlinkability. Basu and Islam [19] introduced an One-Time Password (OTP) based quantum-resilient protocol with partial anonymity but relied on synchronized databases. Jiang et al. [20] presented a threshold password protocol under lattice assumptions, improving brute-force resistance but without credential hiding. Lu and Li [21] proposed a fog-based lightweight authentication for microgrids that included biometric support, yet required pre-installed identities and omitted identity masking. Damir et al. [22] presented a Ring Learning With Errors (RLWE) based scheme for post-5G networks, supporting mutual authentication and forward secrecy, yet it lacked anonymity and decentralized deployment support. Compared to these works, PPAP achieves a unique synthesis of quantum resistance, privacy preservation, unlinkability, and decentralization, addressing the critical limitations found in prior proposals.

3. PRELIMINARIES

Let λ be the security parameter. All probabilistic polynomial-time (PPT) algorithms are parameterized by λ.
Definition 1: A Key Encapsulation Mechanism (KEM) comprises three probabilistic polynomial-time algorithms defined as follows.

- *The key generation algorithm* $(dk, ek) \leftarrow \text{Gen}(\lambda)$ takes a security parameter λ used as a seed for randomness as input and outputs a pair of keys: an encapsulation key ek and a decapsulation key dk.
- *The encapsulation algorithm* $c \leftarrow \text{Encaps}(ek, s)$ receives the encapsulation key ek and a secret message s, and returns a ciphertext c.
- *The decapsulation algorithm* $s \leftarrow \text{Decaps}(dk, c)$ accepts the decapsulation key dk and ciphertext c, and outputs the original secret message s. A KEM is said to be correct if, for all key pairs (dk, ek) generated by Gen and any ciphertext c produced by $\text{Encaps}(ek, s)$, it holds that $\text{Decaps}(dk, c) = s$.

All scheme-specific parameters (e.g. lattice dimension for ML-KEM) are fixed as system constants and not passed explicitly to the algorithms.
Definition 2: A function negl(λ) is called negligible if, for every polynomial $p(\lambda)$, there exists a positive integer l such that for all $\lambda > l$, it holds that $\text{negl}(\lambda) < 1/p(\lambda)$. Negligible functions decrease faster than the inverse of any polynomial, thus becoming insignificant as the security parameter λ grows.
Definition 3: A Key Encapsulation Mechanism (KEM) is said to be indistinguishable under chosen-ciphertext attacks (IND-CCA secure) if, for every probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the adversary's advantage in the following game is negligible in the security parameter λ. Initially, the challenger generates a key pair $(ek, dk) \leftarrow \text{Gen}(1^{\lambda})$ and provides the adversary with the encapsulation key ek and oracle access to the decapsulation oracle $\vartheta_{Decaps}(.)$. The adversary may query this oracle on any ciphertext c, receiving the corresponding plaintext, except for the challenge ciphertext defined subsequently. At some point, the adversary outputs a challenge request. The challenger then selects a random secret s^* from the secret space, encapsulates it as $c^* \leftarrow \text{Encaps}(ek, s^*)$, and sends c^* back to the adversary. The adversary continues oracle queries, excluding the challenge ciphertext c^*, and ultimately outputs a guess b' attempting to distinguish s^* from a random string. The adversary's advantage is defined as:

$$\text{Adv}_{\mathcal{A}}^{\text{IND-CCA}}(\lambda) = \left| \Pr[b' = 1] - \frac{1}{2} \right| \quad (1)$$

The KEM is IND-CCA secure if $\text{Adv}_{\mathcal{A}}^{\text{IND-CCA}}(\lambda)$ is negligible in λ.
Definition 4: A function Random() is defined as a stateless random oracle that, upon each invocation, outputs a value $r \in \{0, 1\}^n$ selected uniformly at random, where n is the specified output length. Each output is sampled independently from a

uniform distribution over all n-bit strings. The function does not take any input, and successive outputs are independent of one another. For any polynomial number of calls, all outputs are independent and identically distributed, satisfying the property that for any $i \neq j$, the probability of collision $\Pr[r_i=r_j]=2^{-n}$

Definition 5: In the random oracle model, a hash function $HF:\{0,1\}^* \rightarrow \{0,1\}^n$ is modeled as an ideal function that behaves as follows: for each unique input $x \in \{0,1\}^*$, the output $HF(x)$ is selected uniformly at random from $\{0,1\}^n$ and is fixed for all future queries to the same input. If the same input x is queried again, the oracle returns the previously assigned value $HF(x)$, ensuring consistency. Outputs for distinct inputs are independent and computationally unpredictable to any probabilistic polynomial-time adversary without querying the oracle. The probability of a collision among q distinct queries is bounded above by

$$\text{coll}(n) \leq \frac{q^2}{q^{n+1}} \tag{2}$$

Which is negligible as long as $q \ll 2^{n/2}$ according to the birthday bound.

Definition 6: A Key Derivation Function (KDF) is a deterministic polynomial-time algorithm that derives a cryptographic session key from one or more input secrets. Given a set of input values $s_1, s_2, \ldots, s_n \in \{0,1\}^*$, and a desired output length ℓ, the KDF computes:

$$k = \text{KDF}(s_1 \parallel s_2 \parallel \ldots \parallel s_n) \tag{3}$$

Where $\parallel$ denotes concatenation, and $k \in \{0,1\}^\ell$ is the resulting session key. A secure KDF must satisfy the following properties:

- *Determinism*: identical inputs always yield the same output.
- *Collision Resistance*: it is computationally infeasible to find two different input tuples that produce the same output.
- *Key Indistinguishability*: the output key k is computationally indistinguishable from a random string of length ℓ to any efficient adversary without knowledge of all input secrets.

Definition 7: The CKKS scheme is a homomorphic encryption (HE) system that supports approximate arithmetic over encrypted data. It consists of the following polynomial-time algorithms.

- *The key generation algorithm* $(sk, pk, rk, gk) \leftarrow \text{HGen}(\lambda)$ takes a security parameter λ as input, and outputs a key set (sk, pk, rk, gk), where sk is the secret key, pk is the public key, rk is the relinearization key, and gk is the set of rotation keys.
- *The encryption algorithm* $c \leftarrow \text{Encrypt}(pk, m)$ takes the public key pk and a plaintext vector m (typically consisting of complex numbers) and produces a ciphertext c.
- *The decryption algorithm* $m' \leftarrow \text{Decrypt}(sk, c)$ uses the secret key sk to recover an approximate plaintext m' from c, where $m' \approx m$. The scheme also supports homomorphic addition and subtraction; given two ciphertexts c_1 and c_2, the algorithms $\text{Add}(c_1, c_2)$ and $\text{Subtract}(c_1, c_2)$ produce ciphertexts corresponding approximately to m_1+m_2 and m_1-m_2, respectively.

The CKKS scheme is correct if the decryption of homomorphically evaluated ciphertexts yields a result that approximates the true result with negligible error under proper parameter settings. All scheme-specific parameters (e.g., polynomial modulus degree and ciphertext modulus) are assumed to be fixed system constants and are not passed explicitly to the algorithms.

4. SYSTEM MODEL

In the PPAP system model, each user or device is represented as a peer entity capable of independently initiating or responding to authentication requests. The model assumes several cryptographic capabilities. First, users derive hidden keys deterministically from credentials (such as passwords) and domain-specific registration nonces using memory-hard functions. Second, ephemeral and deterministic public keys are generated using the ML-KEM scheme, which provides post-quantum key encapsulation. Third, identity attributes, such as usernames, are masked using keyed hash functions and randomly generated nonces, ensuring unlinkability and privacy. Fourth, privacy-preserving operations are enabled through CKKS-based homomorphic encryption, supporting secure computation on encrypted biometric or behavioral data. Finally, all protocol messages are transmitted over reliable transport protocols such as Transmission Control Protocol (TCP) [23] or Quick UDP Internet Connections (QUIC) [24], which ensure in-order delivery and retransmission of lost packets, thereby providing dependable communication channels between peers.

4.1. Threat Model and Assumptions

PPAP assumes a Dolev-Yao (DY) adversary model [25], where the attacker has full control over the communication channel and can intercept, modify, delay, replay, or inject messages. However, the underlying cryptographic primitives such as post-quantum key encapsulation mechanisms (KEMs), Secure Hash Algorithm 3 (SHA-3) hashing [26], and CKKS homomorphic

encryption is assumed to be quantum-resistant under their respective security models. Specifically, KEMs are assumed to achieve indistinguishability under adaptive chosen-ciphertext attacks (IND-CCA2) based on the hardness of the RLWE [27]. While SHA-3 and CKKS do not provide IND-CCA2 security, they are considered quantum-resistant against known attacks under Grover's model [4] and RLWE hardness, respectively. For session key security analysis, we additionally adopt the Canetti–Krawczyk (CK) adversary model [28], which extends Dolev-Yao to capture key indistinguishability, forward secrecy, and key compromise scenarios.

4.2. Design Goals

The PPAP protocol is designed to meet several critical goals for secure authentication in decentralized and privacy-sensitive environments. First, it supports decoupled mutual authentication between peers, where each party maintains independent credentials for each trust relationship. This model eliminates reliance on centralized identity providers or global directories and enables fine-grained trust control. Second, the protocol provides a fully decentralized identity infrastructure, avoiding the need for certificate authorities, public key repositories, or shared password databases. Third, PPAP ensures post-quantum cryptographic resilience by employing primitives such as ML-KEM, Argon2 [29], and SHA-3, each selected for their security under known quantum attack models. Fourth, privacy is rigorously preserved by masking identities with session-specific nonces, using domain-bound credentials, and ensuring unlinkability across sessions and domains. Fifth, the design guarantees credentials unlinkability even when the same user credential is reused across different domains by incorporating dynamic registration nonces. Sixth, it defends against replay and impersonation attacks using ephemeral KEM keys and nonce-based challenge-response mechanisms. Seventh, forward secrecy is maintained by deriving each session key from short-lived secrets that are erased after use. Finally, PPAP integrates multi-factor authentication mechanisms including approximate matching of encrypted biometric or behavioral traits without exposing raw user data, achieving layered and privacy-preserving trust guarantees.

5. THE PROPOSED PROTOCOL

PPAP consists of two main phases: a registration phase (one-time setup, per domain) and an authentication phase (per-session, mutual authentication). This separation ensures that identity credentials are never reused across domains, and that authentication remains unlinkable, decentralized, and post-quantum secure.
In our protocol description, we use Alice and Bob to represent two entities engaging in mutual authentication, with Alice typically initiating the authentication process and Bob responding.

5.1. Registration Phase

Prior to any authentication, a user must perform a one-time registration on each node they wish to interact with. The purpose of this phase is to derive a domain-specific and unlinkable key pair.
Credential-Derived Key Registration: ML-KEM keypairs are deterministically generated from credentials. The encapsulation key is shared securely with the registerer along with a registration nonce.

1) Alice sends a registration request including Alice's identity u_a and a random nonce r_a to Bob.
   Alice → Bob: u_a, r_a
2) Bob replies with a random nonce r_b.
   Bob → Alice: r_b
3) Both parties compute the registration nonce η_a:
   $\eta_a \leftarrow \mathrm{HF}(r_a \,\|\, u_a \,\|\, r_b)$
4) Alice derives a 64-bit seed using the Argon2 key derivation function with Alice's identity u_a, credential pw_a, and the registration nonce η_a. This seed is then used to generate a deterministic key pair:
$$seed_a \leftarrow \mathrm{Argon2}(u_a \,\|\, pw_a \,\|\, \eta_a) \quad (4)$$
$$(dk_a, ek_a) \leftarrow \mathrm{Gen}(seed_a) \quad (5)$$
5) Alice sends the encapsulation key ek_a to Bob.
   Alice → Bob: ek_a.
6) Bob stores u_a, ek_a, and η_a in the database.

The SGen function summarizes the key pair generation defined in (4) and (5):
$$(dk, ek) \leftarrow \mathrm{SGen}(u, pw, \eta) := \mathrm{Gen}(\mathrm{Argon2}(u \,\|\, pw \,\|\, \eta) \quad (6)$$
Argon2 is a cryptographic password hashing algorithm, winner of the 2015 Password Hashing Competition, provides robust resistance against brute-force and side-channel attacks [30]. The use of dynamic registration nonce prevents dictionary attacks and ensures that identical passwords yield different and unlinkable key pairs. Furthermore, even a malicious node cannot use the derived key to authenticate the user elsewhere due to nonce mismatch.

5.2. Authentication Phase

The authentication phase enables mutual authentication between two peers using ephemeral KEMs, identity masking, and encapsulated secrets. This concludes a full mutual authentication exchange that relies solely on ephemeral and derived KEM keys without signatures, certificates, or plaintext identities.

Identity Masking is achieved by hashing the concatenation of user identity u with a masking nonce μ to conceal identifiers from external observers, ensuring that usernames or account identifiers are never transmitted in plaintext.

1) Alice connects to Bob and receives the masking nonce μ_b.
   Bob → Alice: μ_b
2) Alice calculates the masked identity z_a:

$$z_a \leftarrow \text{Mask}(u_a, \mu_b) := \text{HF}(u_a \parallel \mu_b) \tag{7}$$

3) Bob looks up z_a in the database to find u_a, ek_a, and η_a if available:

$$(u_a, ek_a, \eta_a) \leftarrow \text{Lookup}(z_a) \tag{8}$$

The *Lookup* function can be implemented in various ways, if μ_b was generated randomly per-session, the function must compute $\text{HF}(u_x \parallel \mu_b)$ for all user entries in the database to find a match. Alternatively, using a per-domain μ_b provides optimal performance but reduced security.

Perfect Forward Secrecy (PFS) guarantees that the compromise of long-term encapsulation keys does not endanger the confidentiality of past session keys, as all session keys are derived from ephemeral key material discarded after use. The proposed protocol achieves PFS by utilizing ephemeral key pairs that are generated for each session, along with an ephemeral session secret s_e, and discarded once the handshake ends. These ephemeral secrets are later combined with secrets derived from credentials to prevent man-in-the-middle (MITM) attacks.

$$(dk_e, ek_e) \leftarrow \text{EGen} := \text{Gen}(\text{Random}()) \tag{9}$$

Authentication Modes: PPAP supports multiple authentication modes, allowing flexibility based on the authentication methods of initiating and receiving nodes. The masking nonces (μ_a, μ_b) are considered random or predefined for these interactions. The diverse authentication modes adaptability allows various deployment scenarios, from traditional client-server interactions to fully decentralized peer-to-peer mutual authentication, which is crucial for broad applicability.

Single-Identity Authentication: The initiating node (e.g., Alice) presents an identity and proves possession of the corresponding credentials, while the receiving node (e.g., Bob) does not disclose its own identity. The initiating node nevertheless verifies the legitimacy of the receiving node by ensuring it holds the correct encapsulation key and can validate the credentials. The steps are illustrated in Figure 1.

1) Alice generates ephemeral session keys ek_e and dk_e as in (9).
   $(ek_e, dk_e) \leftarrow \text{EGen}()$
2) Alice sends the ephemeral encapsulation key to Bob.
   Alice → Bob: ek_e
3) Bob samples a random secret s_e and encapsulates it under ek_e received from Alice.
   $s_e \leftarrow \text{Random}()$
   $c_e \leftarrow \text{Encaps}(ek_e, s_e)$
4) Bob sends the encapsulated ephemeral secret c_e and identity masking nonce μ_b to Alice.
   Bob → Alice: c_e, μ_b
5) Alice recovers s_e and masks Alice's identity u_a using μ_b as in (7).
   $s_e \leftarrow \text{Decaps}(dk_e, c_e)$
   $z_a \leftarrow \text{Mask}(u_a, \mu_b)$
6) Alice sends the masked identity z_a to Bob.
   Alice → Bob: z_a
7) Bob calls the Lookup function as in (8) to retrieve Alice's record (u_a, ek_a, η_a) then samples a random secret s_b and encapsulates it under ek_a.
   $(u_a, ek_a, \eta_a) \leftarrow \text{Lookup}(z_a)$
   $s_b \leftarrow \text{Random}()$
   $c_b \leftarrow \text{Encaps}(ek_a, s_b)$
8) Bob sends the encapsulated secret c_b to Alice along with η_a.
   Bob → Alice: c_b, η_a
9) Alice decapsulates c_b with dk_a to recover s_b. This can be achieved through two modes:
   - **Persistent Mode**: Alice uses a cached dk_a stored on the device from a previous session associated with the same identity u_a to decapsulate Bob's secret s_b.
     $s_b \leftarrow \text{Decaps}(dk_a, c_b)$

- **Interactive Mode**: Alice generates the (dk_a, ek_a) keypair as in (6), then uses dk_a to decapsulate Bob's secret.
  $(dk_a, ek_a) \leftarrow \text{SGen}(u_a, pw_a, \eta_a)$
  $s_b \leftarrow \text{Decaps}(dk_a, c_b)$

10) Both parties compute the session key k_s using the key derivation function defined in (3).
$k_s \leftarrow \text{KDF}(s_e \| s_b \| \mu_b \| \eta_a \| u_a)$

The session key is uniquely bound to the ephemeral secret s_e, the peer-specific secret s_b, the domain nonce μ_b, the registration nonce η_a, and the user identifier u_a. The inclusion of these values guarantees freshness, forward secrecy under the security of the underlying ML-KEM, and unlinkability across domains. Consequently, PPAP achieves implicit key agreement in addition to authentication, enabling the derived session key k_s to be used for establishing a secure communication channel between peers.

The single-identity procedure supports multiple directional configurations depending on whether the initiator or responder regenerates credentials or relies on cached keys. Specifically, it supports four modes of operation the *Interactive → Anonymous*, *Anonymous → Interactive*, *Persistent → Anonymous*, and *Anonymous → Persistent*.

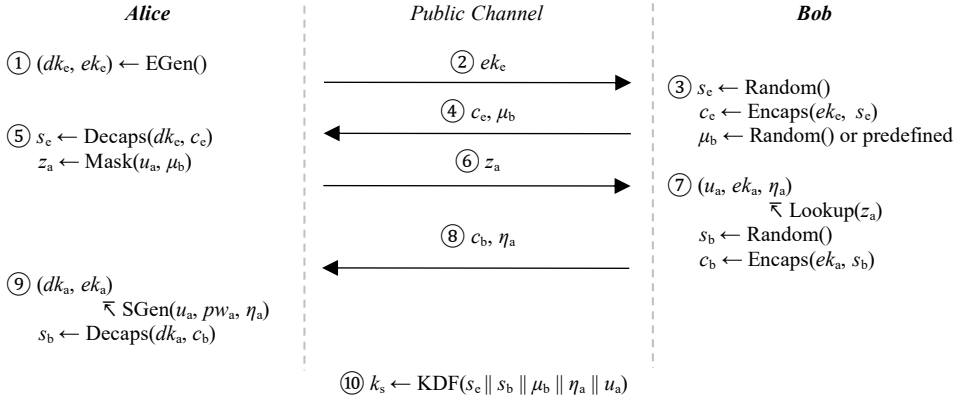

Fig. 1. Single-identity authentication (Interactive → Anonymous)

Dual-Identity Authentication: Both nodes present their identities and prove possession of the corresponding credentials. Each node verifies the legitimacy of the other by confirming the correct use of the associated encapsulation and decapsulation keys. This mode provides full bidirectional assurance of identity. The process is shown in Figure 2:

1) Alice generates ek_e and dk_e as in (9).
$(\text{ek}_e, \text{dk}_e) \leftarrow \text{EGen}()$
2) Alice sends ek_e and the masking nonce μ_a to Bob.
Alice → Bob: ek_e, μ_a
3) Bob samples a random ephemeral secret s_e, encapsulates it using the ephemeral encapsulation key ek_e received from Alice and masks Bob's identity u_b with Alice's masking nonce μ_a.
$c_e \leftarrow \text{Encaps}(ek_e, s_e)$
$z_b \leftarrow \text{Mask}(u_b, \mu_a)$
4) Bob sends the encapsulated ephemeral secret c_e to Alice along with Bob's masking nonce μ_b and Bob's masked identity z_b.
Bob → Alice: c_e, μ_b, z_b
5) Alice decapsulates c_e using dk_e to recover s_e. Alice masks Alice's identity u_a using (7), samples a secret s_a and encapsulates it with Bob's encapsulation key ek_b. Bob's ek_b is obtained via lookup on z_a as in (8).
$s_e \leftarrow \text{Decaps}(dk_e, c_e)$
$(u_b, ek_b, \eta_b) \leftarrow \text{Lookup}(z_b)$
$z_a \leftarrow \text{Mask}(u_a, \mu_b)$
$s_a \leftarrow$ Random()
$c_a \leftarrow \text{Encaps}(ek_b, s_a)$
6) Alice sends z_a, c_a, and Bob's registration nonce η_b to Bob.
Alice → Bob: z_a, c_a, η_b
7) Bob either uses dk_b from persistent storage or deterministically generates it. Bob decapsulates c_a using dk_b to find s_a then invokes the Lookup function to obtain Alice's record, which includes Alice's identity u_a, encapsulation key ek_a, and registration nonce η_a. Bob generates a random secret s_b and encapsulates it under ek_a.
$(dk_b, ek_b) \leftarrow \text{SGen}(u_b, pw_b, \eta_b)$

$s_a \leftarrow \text{Decaps}(dk_b, c_a)$
$(u_a, ek_a, \eta_a) \leftarrow \text{Lookup}(z_a)$
$s_b \leftarrow \text{Random}()$
$c_b \leftarrow \text{Encaps}(ek_a, s_b)$

8) Bob sends the encapsulated secret c_b and η_a to Alice.
Bob → Alice: c_b, η_a

9) Alice either generates dk_a or retrieves it from persistent storage and then uses it to decapsulate s_b from c_b.
$(dk_a, ek_a) \leftarrow \text{SGen}(u_a, pw_a, \eta_a)$
$s_b \leftarrow \text{Decaps}(dk_a, c_b)$

10) Finally, both parties compute the session key k_s by invoking KDF on $s_e \| s_a \| s_b \| \mu_a \| \mu_b \| \eta_a \| \eta_b \| u_a \| u_b$.
$k_s \leftarrow \text{KDF}(s_e \| s_a \| s_b \| \mu_a \| \mu_b \| \eta_a \| \eta_b \| u_a \| u_b)$

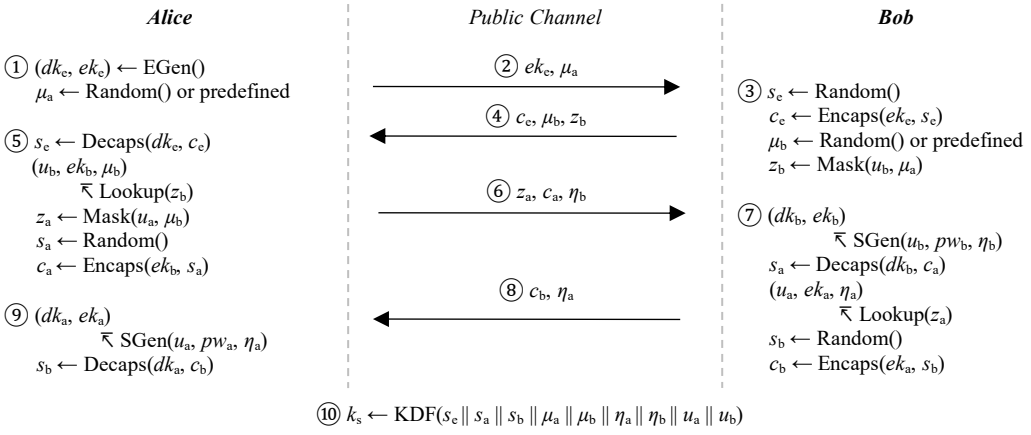

Fig. 2. Dual-identity authentication (Interactive → Interactive)

5.3. Protocol Operation in Decentralized Systems

In PPAP, each node maintains two perspectives of authentication data:

1) Hosted Accounts table, which are credential records for other participants stored locally. When a peer initiates authentication, the request is validated against the corresponding Hosted Account table entry.
2) The Self-Credential table, representing the node's own cryptographic identity used when connecting to other nodes; each entry may include a cached decapsulation key (*dk*).

As illustrated in Figure 3, Alice, Bob, Carol, and Dave operate within a decentralized setting without reliance on central authority. Each node stores Hosted Accounts for peers to be authenticated, while relying on its Self-Credential to prove identity to others. If the decapsulation key is not available, the node prompts the local operator to provide a credential. The input is used to deterministically regenerate the pair (*ek*, *dk*). For example, when Carol on node C authenticates with node D, node C initiates a prompt for Carol to input credentials because no *dk* is cached in the Self-Credential table of node C, on the other hand Alice on node A can authenticate on node C through the cached *dk* entry.

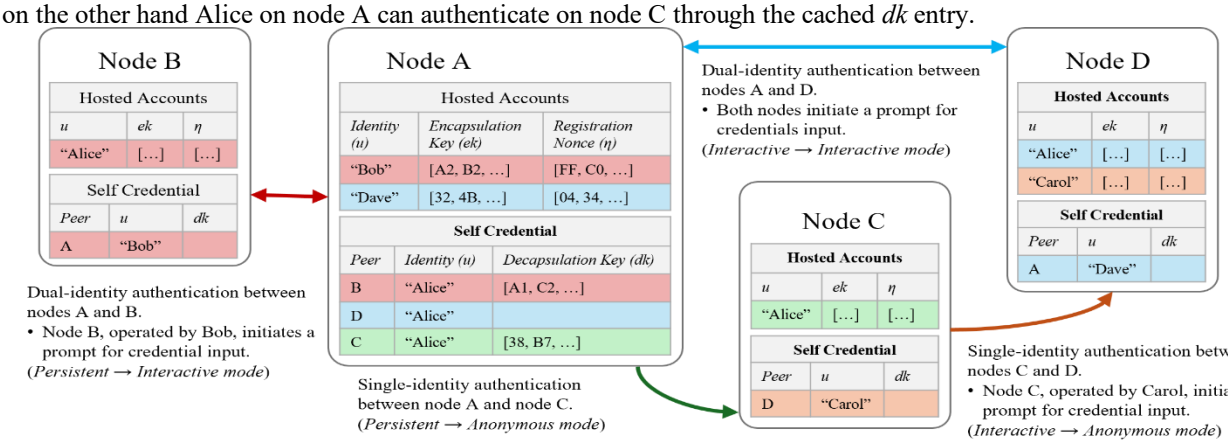

Fig. 3. Decentralized authentication in PPAP. Nodes maintain Hosted Accounts for peers and a Self-Credential for their own identity. Arrows indicate either dual-identity authentication (both sides provide identities) or single-identity authentication (only one side proves identity).

5.4. Multi-Factor Trust Layer

To enhance authentication assurance while preserving user privacy, PPAP introduces a multi-factor authentication using homomorphic encryption. This layer is embedded within the authentication phase and designed to operate under strict privacy constraints: no raw data, biometric template, or plaintext knowledge are exposed during registration or authentication.

The trust layer is designed for approximate knowledge-based factors that do not require absolute precision and may be employed with authentication data that meets a predetermined acceptance threshold, such as behavioral analysis, handwritten signatures, or biometric characteristics. It employs CKKS to ensure the preservation of privacy. The choice of CKKS addresses the critical challenge of privacy in fuzzy matching scenarios, enabling computation on encrypted data without compromising sensitive biometric templates [31]. Biometric authentication typically involves comparing a live biometric sample with a stored template to determine a match. This process traditionally requires either the raw biometric data or its extracted features to be accessible in plaintext, often on a server, which poses significant privacy risks if the server is compromised [32]. HE offers a solution by allowing computations to be performed directly on encrypted data, meaning the node never sees the sensitive plaintext. The CKKS scheme is particularly well-suited for this application because it supports approximate arithmetic operations, which are ideal for calculating distances or similarity scores inherent in biometric matching algorithms.

In this mode Bob verifies Alice's answer and the possession of the secret decryption key. The registration and authentication processes are shown in Figure 4 (a) and (b) respectively.

Registration steps:

1) Bob sends a set of questions Q and random nonce r_b.
   Bob → Alice: r_b
2) Alice selects $\theta_a \in Q$ and generates a random nonce r_b to compute the registration nonce.
   $$\eta_a \leftarrow HF(r_b \,||\, u_a \,||\, r_a) \qquad (10)$$
   Alice generates CKKS homomorphic keys sk_a, pk_a, gk_a and rk_a deterministically using u_a, pw_a and η_a.
   $(sk_a, pk_a, gk_a, rk_a) \leftarrow HGen(Argon2(u_a \,||\, pw_a \,||\, \eta_a))$
   Alice encrypts the initial answer iw_a into an initial ciphered answer icw_a
   $icw_a \leftarrow encrypt(sk_a, iw_a)$
3) Alice sends icw_a to Bob along with θ_a, r_a, pk_a, gk_a, and rk_a.
   Alice → Bob: icw_a, θ_a, r_a, pk_a, gk_a, rk_a
4) Bob derives η_a using (10) and saves icw_a, θ_a, η_a, pk_a, rk_a, and gk_a in the database.

Authentication steps:

1) Bob sends η_a and θ_a to Alice.
   Bob → Alice: η_a, θ_a
2) Alice generates the HE keys if not persistent in the node and encrypts the answer w_a into cw_a.
   $(sk_a, pk_a, rk_a, gk_a) \leftarrow HGen(Argon2(u_a \,||\, pw_a \,||\, \eta_a))$
   $cw_a \leftarrow Encrypt(pk_a, w_a)$.
3) Alice sends cw_a to Bob.
   Alice → Bob: cw_a
4) Bob computes the distance d between the encrypted answer cw_a and the initial ciphered answer icw_a.
   $d \leftarrow Subtract(cw_a, icw_a)$
   Bob adds a random noise e to the computed distance, which will be used to challenge Alice's possession of the answer and the secret key used for decryption.
   $e \leftarrow Random()$
   $d^* \leftarrow Add(d, e)$.
5) Bob sends d^* to Alice.
   Bob → Alice: d^*
6) Alice decrypts d^* to recover the challenge $\hat{e}$.
   $\hat{e} \leftarrow Decrypt(sk_a, d^*)$
7) Alice sends $\hat{e}$ to Bob.
   Alice → Bob: $\hat{e}$
8) Bob rejects Alice if the comparison between $\hat{e}$ and e is above a specific threshold τ.
   $$Reject(\hat{e}, e) := \begin{cases} 1. & if\ |\hat{e} - e| > \tau \\ 0. & otherwise \end{cases}$$

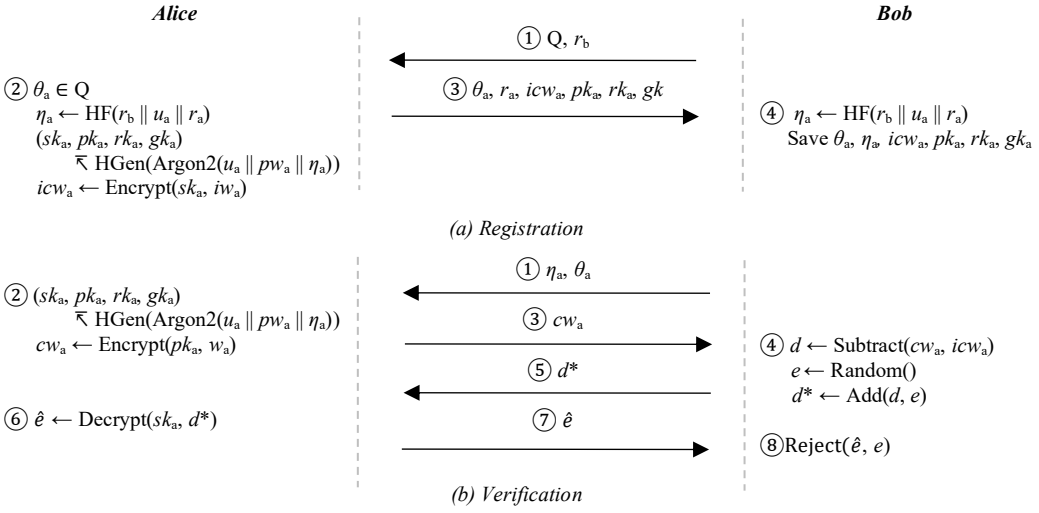

Fig. 4. Approximate Matching Authentication

Subtract, *Add*, and *Reject* operations introduced in the protocol serve as a general framework, allowing for different set of sophisticated underlying algorithms to be implemented depending on the specific biometric feature and template used in approximate matching. The authentication process can be repeated for multiple fingerprint checks or signature verifications if needed each with its specific implementation of these functions.

5.5. Unified Key Derivation Authentication

Unified Key Derivation (UKD) is a design principle in PPAP where all cryptographic keys, including those for KEM and CKKS operations, are deterministically generated from a single source, the user's identity, credentials and registration nonce. This approach simplifies the user experience by eliminating the need to manage multiple distinct keys, as all necessary keys are derived consistently and automatically.

The SGen function is extended to support CKKS keypair generation using a combination of the user's identity, credential, and registration nonce. It is redefined as follows:

$$(dk, ek, pk, sk) \leftarrow \text{UnifiedSGen}(u, pw, \eta) := (\text{Gen}(seed), \text{HGen}(seed)), \text{ where } seed = \text{Argon2}(u \| pw \| \eta)$$

The Lookup function is extended to additionally return the initial ciphered answer *icw* that was previously stored in the verifier's database for comparison:

$$(u, ek, \eta, icw) \leftarrow \text{UnifiedLookup}(z)$$

Figure 5 illustrates the Unified Key Derivation Authentication mode, which integrates ML-KEM–based mutual authentication with a CKKS-based trust layer.

The initial phase (Steps 1–8) follows ML-KEM encapsulation and decapsulation: Alice and Bob exchange ephemeral keys, mask their identities with nonces, and retrieve each other's credential records through lookup. This ensures resistance to quantum adversaries while preserving anonymity across domains.

In the second phase (Steps 9–15), the CKKS encryption layer provides a lightweight trust anchor: Alice encrypts a credential-derived factor, Bob applies subtraction and noise addition to form a challenge, and Alice decrypts and returns the result. This homomorphic exchange verifies possession of valid credentials without revealing them in plaintext.

Finally, in Step 16, both parties derive the shared session key k_s by applying the key derivation function (KDF) to the combined ML-KEM secrets, masking nonces, registration nonces, and identities. This unifies the KEM authentication and CKKS trust layer into a single post-quantum multi-factor authentication protocol.

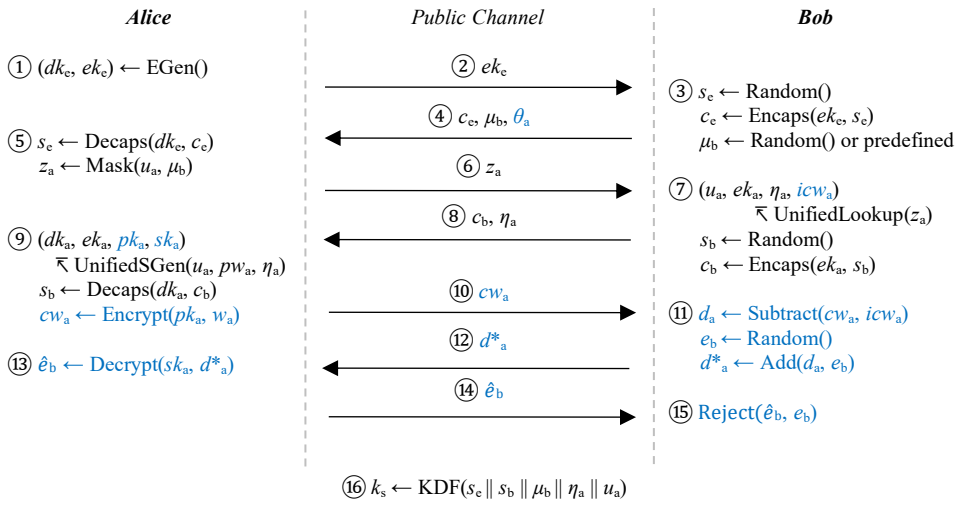

Fig. 5. Unified Key Derivation Authentication. New operations introduced beyond single-identity authentication are highlighted in blue.

5.6. Registration Nonce Unlinkability

To ensure strong unlinkability and user privacy, it is essential that the registration nonce η used in credential-derived key registration is never static across many sessions. If a fixed registration nonce is reused repeatedly, it becomes a persistent identifier that can be observed and correlated by passive or active adversaries. This compromises the unlinkability property, as an external observer or even a compromised verifier could track the user's interactions across different sessions, devices, or services by simply matching the repeated η or the deterministic keys derived from it. In a decentralized or privacy-sensitive environment, such persistent identifiers undermine anonymity guarantees and expose users to profiling and surveillance.

To mitigate this risk, the system must support either the generation of multiple unique registration nonces or an update mechanism to refresh the nonce after each successful authentication. By incorporating ephemeral randomness and verifier-generated nonces into the computation of each η, the resulting key pairs and encapsulation keys remain unlinkable, even if the underlying user credentials remain the same. This approach ensures that each registration or authentication session produces distinct cryptographic material, reinforcing privacy and preventing long-term identity correlation.

5.7. Time-Aware Predictive Access Model for Credential Lookup

To reduce the average number of hash computations during credential lookup of randomly generated μ, we propose a Time-Aware Predictive Access Model (TAPM) that exploits the inherent skew in user authentication behavior. Empirical studies have shown that user access frequencies in large-scale systems often follow a Zipfian distribution [33], where a small subset of users accounts for a disproportionately large fraction of login activity [34]. TAPM leverages this by combining a recency-weighted frequency score with cyclic temporal access patterns (e.g., rush hours) to prioritize likely candidates. This allows the system to concentrate hash comparisons on a smaller subset of active users, thereby improving lookup efficiency under non-uniform access distributions. TAPM raises the probability that an authenticating user appears near the head of the probe list. Node b generates a fresh random mask μ_b and sends it to node a. Node a replies with $z_a = \mathrm{HF}(u_a \,||\, \mu_b)$. The task of node b is to recover u_x such that:

$$z_x \leftarrow \mathrm{HF}(u_x \,||\, \mu_b) \stackrel{?}{=} z_a$$

A naïve search over all registered identities incurs $O(N)$ hash computations. TAPM reduces average cost by maintaining a small, time-aware candidate list.

Model Formulation: Each user u is characterized by a set of past access timestamps $T_u = \{t_{u,1}, t_{u,2}, \ldots\}$ and a time-of-day frequency histogram $H_u[h]$ for hour $h \in \{0, \ldots, 23\}$, where $H_u[h]$ is the count of logins user u made during hour h (aggregated over suitable past interval). We define decayed popularity score:

$$F_u = \sum_{t \in T_u} exp(-\lambda(t_{now} - t)) \quad (11)$$

Where $\lambda = \ln 2 / t_{½}$ and $t_{½}$ is the decay half-life. Recent logins contribute more heavily. Next, the time-of-day affinity at current hour h_{now} is defined as:

$$A_u(h_{now}) = \frac{H_u[h_{now}]}{max_h H_u[h]} \quad (12)$$

These two quantities from (11) and (12) are blended into a single predictive priority in log-space:

$$P_u = exp((1 - \gamma)\, ln\, F_u + \gamma\, A_u(h_{now})) \quad (13)$$

Where $\gamma \in [0,1]$ governs the trade-off between overall decay popularity score and time-of-day affinity.
Candidate Selection: Let UsersList denote the complete set of all registered users in the system, with cardinality $N = |\text{UsersList}|$. Each user u is assigned a predictive priority score P_u based on recency and temporal access patterns as defined in (13). Let PredList denote the ordered subset containing the top L users with the highest predictive priority P_u.
Upon receiving the masked identity z_a, node b executes a two-stage lookup process shown in Algorithm 1. In the hot-path probing phase, node b iterates over users in PredList. computing $z_x \leftarrow \text{HF}(u_x \| \mu_b)$ for each candidate and comparing it with z_a If a match is found, the corresponding identity u_x is returned immediately. Otherwise, the search continues with the cold-path probing phase, which checks all remaining users in (UsersList \ PredList).

Algorithm 1: TAPM Lookup
Inputs: UsersList, PredList, z_a, μ_b
Output: u_a
// Hot-path probing
for each u_x in PredList do
  $z_x \leftarrow \text{HF}(u_x \| \mu_b)$
  if ($z_x = z_a$) then return u_x
end for
// Cold-path probing
for each u_x in (UsersList \ PredList) do
  $z_x \leftarrow \text{HF}(u_x \| \mu_b)$
  if ($z_x = z_a$) then return u_x
end for
return ⊥ // No match found

This two-stage design reduces the number of hash evaluations per authentication request. The expected lookup cost is expressed as:

$$C_{avg} = P_{hit}\, L + (1 - P_{hit})\, N$$

When $P_{hit} \gg L/N$, the average complexity approaches $O(L)$, providing significant performance improvement over the naïve $O(N)$ full-scan baseline.
Parameter Selection and Deployment Considerations: To effectively balance security and performance in deployments, several system parameters require careful tuning. The decay half-life parameter $t_{½}$ should reflect how rapidly user access patterns change, typically ranging between one to seven days. The blend weight γ determines the emphasis between recent activity and time-of-day affinity; a lower value favors overall recency, while a higher value emphasizes temporal login consistency. The candidate list size L should be configured to represent a fraction of the user base, commonly the top 5-10% most active users, to maintain high prediction accuracy while reducing lookup overhead. For practical deployment, system implementers should periodically update user frequency scores and time histograms (e.g., hourly and daily) to adapt to evolving usage patterns while minimizing computational cost.

6. SECURITY ANALYSIS

This section presents the formal security properties provided by PPAP and evaluates its resilience against adversaries under both classical and quantum threat models. The analysis assumes a Dolev–Yao adversary capable of full network control and considers cryptographic primitives to be ideal, except where explicitly challenged by quantum capabilities.

6.1. Provable Security

Theorem 1 (Session Key Security): Let $\mathcal{A}$ be any probabilistic polynomial-time adversary operating in the Dolev–Yao and CK-adversary models, as described in the threat model. Assume that: The key encapsulation mechanism (KEM) is IND-CCA secure (Definition 3), hash functions and random oracles (Definition 5) used for session-specific or nonce-based values have outputs of length n bits and the number of protocol sessions, users, or random oracle queries is much less than $2^{n/2}$ (i.e., well below the birthday bound). Then, the advantage of $\mathcal{A}$ in distinguishing the session key from random or in causing an honest party to accept without a matching session is at most:

$$\text{Adv}_{\mathcal{A}}^{\text{SK-SEC}}(\lambda) \leq \text{Adv}_{\mathcal{A}}^{\text{IND-CCA}}(\lambda) + \text{negl}(\lambda) + \text{coll}(n)$$

Where:

- $\text{Adv}_{\mathcal{A}}^{\text{SK-SEC}}(\lambda)$ is the adversary's advantage in the session key experiment,
- $\text{Adv}_{\mathcal{A}}^{\text{IND-CCA}}(\lambda)$is the advantage against KEM as in (1).
- $\text{negl}(\lambda)$ is a negligible function in the security parameter (Definition 2).
- $\text{coll}(n)$ is the probability of a collision due to the birthday paradox shown in (2).

Proof: We use a standard game-hopping technique to bound the adversary's advantage.

Game 0: $\mathcal{A}$ interacts with honest parties executing the actual protocol. The session key is computed as:

$$k_s = \text{KDF}(\text{HF}(u_a \,||\, \mu_b) \,||\, \text{HF}(u_b \,||\, \mu_a) \,||\, s_e)$$

Let $\Pr[Succ_0]$ denote the probability that $\mathcal{A}$ succeeds in Game 0.

Game 1: Replace the KEM shared secret s_e with uniformly random value s_e^*. By the IND-CCA security of the KEM (Definition 3), if $\mathcal{A}$ can distinguish this game from the real protocol, it can break the IND-CCA security of the KEM.

Thus,

$$|\Pr[\text{Succ}_0] - \Pr[\text{Succ}_1]| \leq \text{Adv}_{\mathcal{A}}^{\text{IND-CCA}}(\lambda)$$

Game 2: In this game, hash functions and KDFs used with nonces or unpredictable session-specific values are modeled as random oracles with n-bit outputs (Definition 6).

Replace $HF(u_a \,||\, \mu_b)$ and $HF(u_b \,||\, \mu_a)$ with independent uniform random values. By the random oracle model:

$$|\Pr[\text{Succ}_1] - \Pr[\text{Succ}_2]| \leq \text{negl}(\lambda) + \text{coll}(n)$$

Game 3: In this game, the session key (k_s) for the test session is replaced by an independent, uniformly random value (since the adversary cannot distinguish it from the real one except with negligible probability due to the above games).

Thus,

$$\Pr[\text{Succ}_2] \leq ½$$

Bounding the total advantage by the triangle inequality, the total advantage of the adversary is:

$$\text{Adv}_{\mathcal{A}}^{\text{SK-SEC}}(\lambda) = |\Pr\left[\text{Succ}_0 - \frac{1}{2}\right]| \leq \text{Adv}_{\mathcal{A}}^{\text{IND-CCA}}(\lambda) + \text{negl}(\lambda) + \text{coll}(n)$$

Where:

- $\text{Adv}_{\mathcal{A}}^{\text{IND-CCA}}(\lambda)$ is the maximal advantage in breaking the KEM.
- $\text{negl}(\lambda)$ is negligible in the security parameter.
- $\text{coll}(n) \approx q^2/2^{(n+1)}$ is the probability of a birthday collision in n-bit outputs.

Provided n is at least 128 (e.g., $n = 256$), and q (the total number of random oracle or hash queries) is far less than $2^{n/2}$, the birthday collision probability $\text{coll}(n)$ is negligible. Therefore, the PPAP protocol achieves session key security and mutual authentication as claimed.

6.2. Automatic Security Verification Using Tamarin Prover

To rigorously evaluate the security properties of the proposed protocol, we employed the Tamarin Prover, an advanced tool for the automated formal verification of security protocols [8]. Tamarin facilitates the modeling and analysis of cryptographic protocols within the symbolic Dolev-Yao adversary model, allowing for the verification of both trace properties (such as authentication, secrecy, and integrity) and observational equivalence properties (such as anonymity and unlinkability).

The protocol was modeled in Tamarin using its high-level, rule-based specification language (Figure 6). Each protocol role, message flow, and cryptographic operations were represented as a series of rewrite rules that describe the evolution of the protocol state. Key cryptographic primitives, including key encapsulation mechanisms (KEM), hash functions, and key derivation functions (KDF), were encoded as abstract operations, following standard assumptions about their security. Adversarial capabilities, such as interception, replay, and message injection, were automatically accounted for by the Tamarin execution environment.

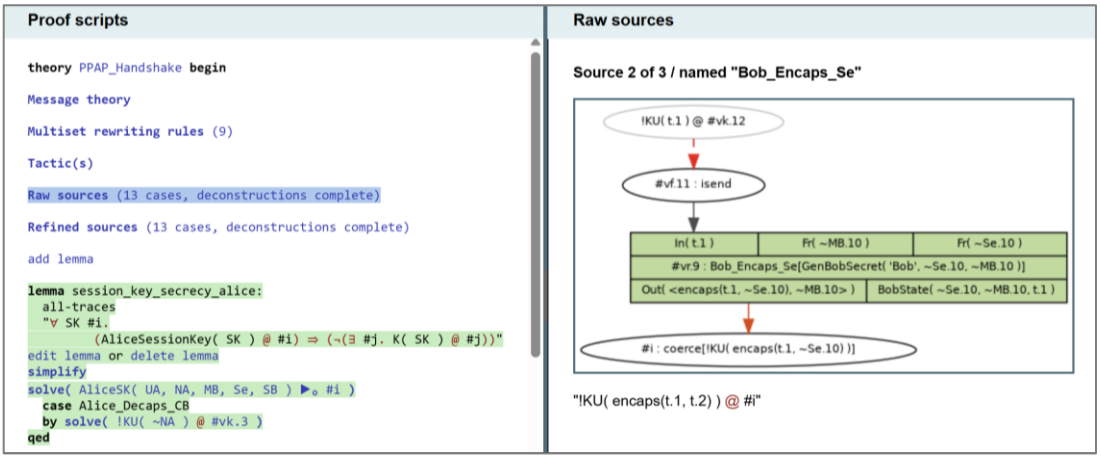

Fig. 6. PPAP in Tamarin Prover

6.3. Further Security Analysis

Mutual Authentication: PPAP ensures mutual authentication by requiring both parties to verify challenges tied to each other's credential-derived keys. Each participant must correctly perform a KEM encapsulation or decapsulation with the peer's key, proving possession of valid credentials.
Forward Secrecy: Forward secrecy is preserved by ephemeral key pairs generated independently in each session. Session keys are derived from a combination of ephemeral secrets exchanged via KEM encapsulations, ensuring that the compromise of long-term user credentials does not reveal any previously negotiated session keys.
Resistance to Replay and Impersonation Attacks: PPAP employs per-session randomness in the form of masking nonces and ephemeral keys. All identity-related information is dynamically derived using these nonces, ensuring that intercepted messages cannot be reused in future sessions. Moreover, successful decapsulation and masking validation require possession of hidden keys derived from user-specific credentials and registration nonces.
Anonymity and Unlinkability: PPAP protects user privacy by avoiding the transmission of plaintext identifiers. User identities are hashed and masked using fresh nonces, resulting in unlinkable pseudonyms. Since the identity masking mechanism is session-specific, adversaries cannot correlate authentication attempts across sessions. Additionally, the lack of centralized authority eliminates global tracking vectors.
Credential Unlinkability Across Domains: In PPAP, KEM keypair are deterministically derived from the combination of the username, password, and a domain-specific registration nonce which is negotiated by both parties at registration phase. This construction guarantees that the same password used across different domains results in entirely distinct credentials, thereby preventing correlation or credential reuse by adversaries or malicious servers.
Post-Quantum Resilience: All cryptographic operations in PPAP are constructed from primitives that are widely believed to remain secure against quantum adversaries. The protocol employs ML-KEM and CKKS schemes, both of which derive their security from hard lattice problems, rendering them resistant to attacks based on Shor's algorithm. Additionally, PPAP utilizes SHA-3 and Argon2 functions to resist large-scale brute-force attacks. The combined use of these primitives limits the advantage of quantum parallelism to the generic quadratic speed-up offered by Grover's algorithm, thereby ensuring robust resilience in post-quantum environments.
Privacy-Preserving Multi-Factor Authentication: PPAP supports multi-factor authentication without compromising user privacy. Approximate knowledge-based authentication is implemented using the CKKS approximate homomorphic encryption scheme, enabling encrypted similarity evaluation without disclosing feature vectors or matching thresholds. These mechanisms are designed to comply with modern data protection requirements such as those imposed by the GDPR.
Based on the above discussion, the comparison on security among [14], [15], [16], [17], [18], [19], [20], [21], [22] and PPAP is described in Table I. The comparison shows our proposed protocol covers more security features.

TABLE I. SECURITY FEATURES COMPARISON

Feature ↓ \ Protocol →	PPAP	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]
Mutual Authentication	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Forward Secrecy	✓	✓	✓	✓	✓	×	×	✓	✓	✓
User Anonymity & Unlinkability	✓	×	×	×	×	✓	×	✓	✓	×
Replay & Impersonation Resistance	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Knowledge/Biometric Support	✓	×	×	×	×	✓	✓	✓	×	×
Decentralized (No Central Authority)	✓	×	✓	×	×	×	×	×	✓	×
Credential Unlinkability	✓	×	×	×	×	✓	×	×	×	×
Privacy-Preserving	✓	×	×	×	✓	✓	×	✓	✓	×
✓ Supported, × Not Supported										

7. PERFORMANCE EVALUATION

To assess the practicality and performance of the PPAP protocol, we evaluated its latency and communication overhead across multiple handshake modes, cryptographic configurations, and homomorphic encryption parameters. The benchmarking platforms were selected to represent devices equipped with human-machine interfaces (HMI) capable of credential input, ranging from a high-performance Intel Core i7-14700K workstation [35] to a Qualcomm Snapdragon X Elite X1E-80-100 laptop [36], and a Snapdragon 8 Elite SM8750-AB smartphone [37]. This setup reflects not only conventional desktop and mobile environments but also extends to industrial control panels, smart-home panels, in-vehicle

information systems, and similar Internet of Things (IoT) with HMI-driven applications [38], all implemented using the .NET 8.0 framework [39].

7.1. Authentication Modes Performance

For secure password-based key generation, the implementation employs the Argon2id algorithm from the Sodium library [40] via a .NET interop wrapper and Bouncy Castle library for C# [41] to implement the ML-KEM. Each security case was configured as in Table II. Tables III and IV summarize the communication cost (in bytes) and latency (in milliseconds), respectively, across all authentication modes.

TABLE II. SECURITY PARAMETER CONFIGURATION

Case	ML-KEM	Argon2id Memory	Argon2id Iterations	HF/KDF	Security Level
1	1024	64 MB	4	SHA3-256	Highest
2	768	49 MB	3	SHA3-256	Medium
3	512	32 MB	2	SHA3-256	Lightweight

TABLE III. COMMUNICATION COST (BYTES)

Authentication Mode	*Case 1*	*Case 2*	*Case 3*
Anonymous → Interactive	4,868	3,524	2,416
Anonymous → Persistent	4,768	3,424	2,400
Interactive → Anonymous	4,868	3,524	2,416
Interactive → Interactive	6,600	4,776	3,264
Interactive → Persistent	6,500	4,676	3,248
Persistent → Anonymous	4,768	3,424	2,400
Persistent → Interactive	6,500	4,676	3,248
Persistent → Persistent	6,400	4,576	3,232

TABLE IV. LATENCY BENCHMARK (IN MILLISECONDS)

	Intel 14700K			Snapdragon X1E-80-100			Snapdragon SM8750-AB		
	Case 1	*Case 2*	*Case 3*	*Case 1*	*Case 2*	*Case 3*	*Case 1*	*Case 2*	*Case 3*
Anonymous → Interactive	62.93	33.92	15.43	113.88	64.36	28.52	153.72	69.83	40.12
Anonymous → Persistent	0.27	0.17	0.11	0.28	0.19	0.12	0.42	0.25	0.18
Interactive → Anonymous	59.95	33.99	15.69	111.87	64.21	28.12	120.88	69.71	40.95
Interactive → Interactive	121.41	68.79	30.76	228.36	130.03	56.74	209.49	139.81	72.45
Interactive → Persistent	60.23	34.21	15.91	113.32	64.83	28.48	110.3	80.66	33.88
Persistent → Anonymous	0.27	0.17	0.11	0.28	0.18	0.12	0.47	0.29	0.18
Persistent → Interactive	61.16	36.54	15.42	113.21	62.99	28.1	138.54	73.58	34.98
Persistent → Persistent	0.38	0.26	0.17	0.42	0.28	0.18	0.65	0.38	0.28

The benchmark results demonstrate that the proposed PPAP protocol achieves practical performance across heterogeneous platforms equipped with human–machine interfaces (HMI), including desktop workstations, laptops, and smartphones. Communication cost scales proportionally with the chosen security configuration, where high-assurance Case 1 incurs the largest overhead, while lightweight Case 3 reduces both latency and message size by nearly 50%. Latency measurements further indicate that high-performance processors, such as the Intel Core i7-14700K, consistently outperform mobile-class Snapdragon chipsets, yet all platforms remain within acceptable limits for real-time authentication. Notably, persistent authentication modes yield sub-millisecond response times and minimal bandwidth consumption, highlighting their suitability for resource-constrained HMI-driven environments. These results confirm the adaptability of PPAP to diverse deployment scenarios while maintaining a balanced trade-off between security strength and operational efficiency.

7.3. Identity Masking Performance

We compute masking operation time consumption on our test machines as shown in Figure 7, where the x-axis represents the degree of parallelism and y-axis is the time cost in milliseconds per 1,000,000 masking operations (SHA3-256), ranging from 1 to each machine maximum number of cores per central processing unit (CPU).

Under a workload of 1,000,000 SHA3 hashes executed with full parallelism, the Intel Core i7-14700K consistently achieved the lowest average time across all configurations, outperforming both Snapdragon X1E-80-100 and Snapdragon SM8750-AB. This superiority stems from the Intel Core i7-14700K's higher core/thread budget (8P+12E, 28 threads), higher sustained clocks with desktop-class power/cooling headroom, and efficient vectorized code paths [35], yielding superior throughput for parallel hashing.

We assess the lookup performance of our TAPM against a naïve baseline (full scan over a fixed, random permutation of the user list) taking 9702 users, 13 days and 18 million access logs from the Los Alamos National Laboratory User-Computer Authentication Associations in Time dataset [42]. We selected L = 970 and processed 100 requests of authentication per hour of real authentication data from the dataset taken from another period of 13 days. Figure 8 shows the probability of the user being in the candidate list (P_{hit}) for $\gamma = 0.24$, $\gamma = 0.5$ and $\gamma = 0.75$. In all cases we find a significant gain in performance in comparison to naïve scan.

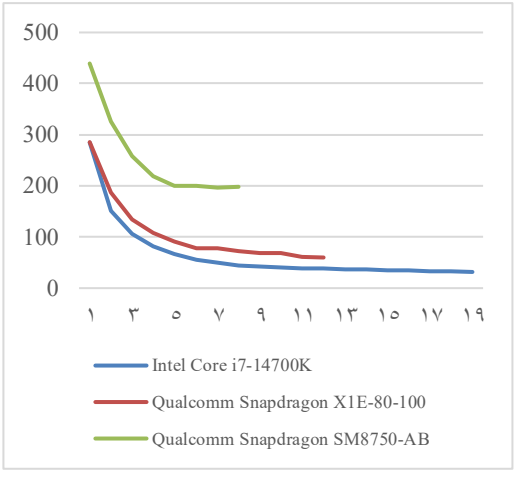

Fig. 7. Effect of parallelism on masking cost (ms)

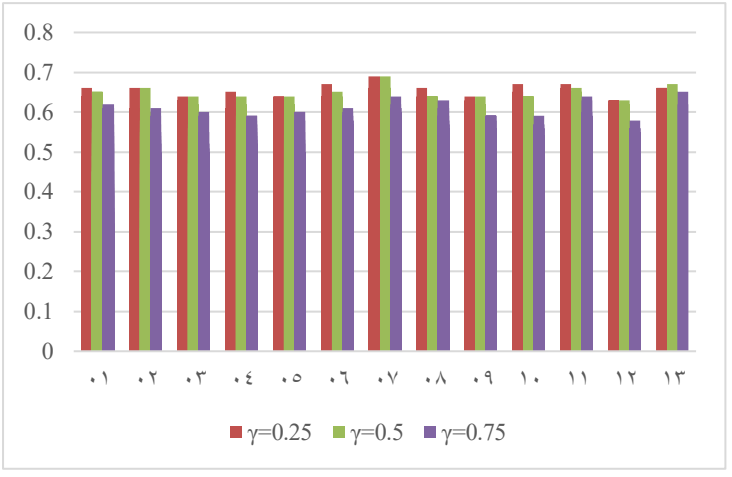

Fig. 8. Hourly probability of user presence in the predicted candidate list (P_{hit})

7.4. Trust Layer Performance

To evaluate the overhead of homomorphic encryption, we implemented approximate biometric matching using the SEAL library [43] (version 4.1.2, CKKS scheme). The reported latency values represent the total round-trip time for a complete authentication conversation between the two participating parties, including SEAL context initialization on both sides. Communication cost and end-to-end latency were measured across multiple polynomial degrees and modulus-chain configurations on Microsoft Windows 11 (Build 26200) and Linux 6.17, as summarized in Table V.

All benchmarks were performed over 1,000 iterations to obtain stable average values. These results demonstrate that the protocol's core cryptographic operations are efficient and suitable for deployment on modern client devices. Furthermore, parameter configurability allows tuning for optimal security-performance trade-offs according to specific application requirements.

TABLE V. APPROXIMATE MATCHING AUTHENTICATION BENCHMARK

Poly Mod Degree	Coeff Modulus Bit Sizes	Depth	Scale	Security	Latency (ms)					Communication Cost (bytes)
					Core i7 14700K		*X1E-80-100*		*SM8750-AB*	
					Linux	*Windows*	*Linux*	*Windows*	*Linux*	
4096	[40, 20, 40]	1–2	2^{20}	< 128-bit	9.5	57.12	10.58	119.94	10.78	161,787
8192	[60, 40, 40, 60]	3–4	2^{40}	128-bit	20.81	177.41	18.62	352.55	22.21	668,526
8192	[60, 30 × 3, 60]	4–5	2^{30}	128-bit	26.63	257.33	31.99	514.62	36.27	705,764
16384	[60, 40 × 4, 60]	5–6	2^{40}	128-bit	57.77	694.24	67.05	1415.55	79.97	210,7215

16384	[60, 50 × 3, 60]	4–5	2^{50}	128-bit	39.68	501.60	54.50	1046.97	61.08	1,938,280
32768	[60, 50 × 8, 60]	8–10	2^{50}	256-bit	265.16	3433.19	312.9	7036.48	351.11	8,588,497

All processors evaluated in Table V, Intel Core i7-14700K, Snapdragon X Elite, and Snapdragon SM8750-AB, benefit from modern vector instruction sets, Advanced Vector Extensions 2 (AVX2) on Intel and Advanced SIMD Extension (NEON) on ARM [44], which accelerate the polynomial arithmetic, Number Theoretic Transform (NTT), and modular operations that dominate homomorphic workloads. The Intel Core i7 leverages wide AVX2 execution paths and high clock speeds to handle large parameter sets efficiently [45], while the ARM-based Snapdragon platforms utilize highly optimized NEON pipelines and balanced core clusters to deliver competitive performance even under stricter power constraints [46] that are well-suited for resource-constrained HMI-driven environments.

Our measurements show that Linux achieves lower latency during the initialization phase due to more efficient compiler optimizations, memory allocation, and threading behavior. This difference is limited to initialization; once the SEAL context is constructed, Windows achieves similar performance for encryption, decryption, and evaluation.

7.5. Trade-off Summary

The evaluation results demonstrate the flexibility and adaptability of the PPAP protocol across different deployment scenarios. High-security environments, such as critical infrastructure systems, are best served by employing the highest security parameters (Case 1) and more robust CKKS homomorphic encryption settings, ensuring maximum protection against sophisticated adversaries at the expense of higher latency and computational costs. Conversely, for resource-constrained environments like IoT devices or mobile applications, lightweight configurations (Case 3) provide substantial efficiency gains, achieving low-latency authentication (under 16 ms) and reduced communication overhead (2-3 KB). This tunability allows PPAP to effectively balance security requirements and performance constraints tailored to diverse practical contexts.

The benchmark results indicate that security configuration in ML-KEM should be selected based on application context. Case 1 offers the strongest protection but introduces the highest communication cost and latency, making it more suitable for critical infrastructures where assurance outweighs performance. Case 2 provides a balanced middle ground, while Case 3 achieves near-instantaneous response times and reduced message sizes, making it highly practical for everyday authentication in HMI-driven environments. Persistent modes consistently deliver sub-millisecond performance across all devices and are recommended where frequent reconnections are expected.

For SHA3 parallel hashing, the Intel Core i7-14700K is the recommended platform when maximum throughput is required, such as in large-scale credential derivation, integrity checking, or audit logging pipelines. Snapdragon platforms, while adequate for moderate workloads, are better suited when energy efficiency or mobility is a priority rather than absolute hashing speed. Thus, deployment choices should distinguish between centralized back-end servers, where throughput is paramount, and edge devices, where moderate performance suffices.

With respect to CKKS, the most practical recommendation is to employ moderate parameter sets (e.g., polynomial modulus degree 4096 or 8192) to balance security and real-time responsiveness. These configurations enable sub-second latencies and manageable communication costs, ensuring viability for interactive HMI applications. Higher parameter sets (e.g., 16384 or 32768) may be reserved for specialized cases requiring greater depth or precision but should be avoided in latency-sensitive scenarios. Snapdragon platforms are well suited for interactive workloads at moderate parameter sets, while Intel platforms remain valuable when large-scale or mixed workloads demand broader system capacity.

In summary, the results highlight clear configuration guidelines: employ high-security ML-KEM cases only where justified, use persistent modes whenever possible to optimize responsiveness, rely on Intel workstations for bulk parallel hashing, and select moderate CKKS parameters on Snapdragon-based devices for privacy-preserving computation in real-time HMI environments.

8. CONCLUSION

This paper introduced a quantum-resistant and privacy-preserving authentication protocol (PPAP) tailored explicitly for decentralized systems. PPAP leverages ML-KEM to effectively address security threats posed by quantum computing, while integrating advanced identity masking techniques to uphold stringent privacy requirements. By incorporating a multi-factor trust layer based on CKKS homomorphic encryption, PPAP uniquely enables secure and privacy-preserving authentication, supporting approximate knowledge factors such as biometrics or handwritten signatures. Rigorous formal analysis, complemented by automated verification using the Tamarin Prover, confirmed the robustness of PPAP against both classical and quantum adversaries, demonstrating resilience against standard security threats including replay, impersonation, and linkage attacks. Empirical evaluation highlights the protocol's efficiency, achieving practical computational and

communication overhead suitable for diverse deployment scenarios, from resource-constrained environments (IoT/mobile) to high-security infrastructures.

The Time-Aware Predictive Access Model (TAPM) proposed significantly reduces lookup complexity by effectively leveraging user access patterns, experimentally demonstrating a substantial improvement ($P_{hit} \approx 0.6$ greater than $L/N \approx 0.1$), thus validating its practical efficiency in real-world authentication scenarios.

The Unified Key Derivation (UKD) mechanism proposed in this work represents an important step toward simplifying key management in decentralized systems. In future work, we plan to extend UKD to support additional cryptographic primitives, evaluate its interoperability with emerging post-quantum standards, and investigate its role in enabling secure credential recovery and cross-domain identity portability. In particular, we plan to investigate the incorporation of privacy-preserving handwritten signature verification using homomorphic encryption and noise-tolerant feature matching. Furthermore, we aim to explore the feasibility of continuous authentication based on individual cardiac rhythms, such as electrocardiogram (ECG) [47] or photoplethysmography (PPG) signals, acquired from wearable devices like smartwatches and smart rings [48].

Finally, we aim to integrate and evaluate the PPAP protocol within the Esiur open-source distributed runtime framework [49] as part of a broader effort to enable secure decentralized Artificial Intelligence (AI) inference orchestration enabling execution of Large Language Models (LLMs) and other neural workloads across heterogeneous platforms. This integration will substantiate the practicality, adaptability, and robustness of PPAP in realistic distributed computing environments.

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

There is no funding for this research.

Acknowledgment

None.

References

[1] B. Jayaraman, H. Li, and D. Evans, "Decentralized certificate authorities," *arXiv preprint arXiv:1706.03370*, 2017.

[2] C. J. Bennett, "The European General Data Protection Regulation: An instrument for the globalization of privacy standards?," *Information Polity*, vol. 23, no. 2, pp. 239–246, 2018.

[3] T. Monz *et al.*, "Realization of a scalable Shor algorithm," *Science*, vol. 351, no. 6277, pp. 1068–1070, 2016.

[4] L. K. Grover, "A fast quantum mechanical algorithm for database search," presented at the Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, 1996, pp. 212–219.

[5] J. Bos *et al.*, "CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM," presented at the 2018 IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, 2018, pp. 353–367.

[6] National Institute of Standards and Technology (US), "Module-lattice-based key-encapsulation mechanism standard," National Institute of Standards and Technology (U.S.), Washington, D.C., NIST FIPS 203, Aug. 2024. doi: 10.6028/NIST.FIPS.203.

[7] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," presented at the Advances in cryptology–ASIACRYPT 2017: 23rd international conference on the theory and applications of cryptology and information security, Hong kong, China, December 3-7, 2017, proceedings, part i 23, Springer, 2017, pp. 409–437.

[8] S. Meier, B. Schmidt, C. Cremers, and D. Basin, "The TAMARIN prover for the symbolic analysis of security protocols," presented at the Computer Aided Verification: 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings 25, Springer, 2013, pp. 696–701.

[9] E. Rescorla, "The transport layer security (TLS) protocol version 1.3," 2070–1721, 2018.

[10] D. Hardt, "The OAuth 2.0 authorization framework," 2070–1721, 2012.

[11] N. Sakimura, J. Bradley, M. Jones, B. De Medeiros, and C. Mortimore, "OpenID Connect Core 1.0 incorporating errata set 1," *The OpenID Foundation, specification*, vol. 335, 2014.

[12] H. Nejatollahi, N. Dutt, S. Ray, F. Regazzoni, I. Banerjee, and R. Cammarota, "Post-quantum lattice-based cryptography implementations: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 6, pp. 1–41, 2019.

[13] P. Regulation, "Regulation (EU) 2016/679 of the European Parliament and of the Council," *Regulation (eu)*, vol. 679, p. 2016, 2016.

[14] H. A. Riva-Cambrin, R. Singh, S. Lama, and G. R. Sutherland, "Extensible Post Quantum Cryptography Based Authentication," *arXiv preprint arXiv:2505.16112*, 2025.

[15] L. P. Fraile *et al.*, "Enabling Quantum-Resistant EDHOC: Design and Performance Evaluation," *IEEE Access*, 2025.

[16] J. Yao, K. Matusiewicz, and V. Zimmer, "Post quantum design in SPDM for device authentication and key establishment," *Cryptography*, vol. 6, no. 4, p. 48, 2022.
[17] J. Samandari and C. Gritti, "Post-quantum authentication in the MQTT protocol," *Journal of cybersecurity and privacy*, vol. 3, no. 3, pp. 416–434, 2023.
[18] R. Arjona, P. López-González, R. Román, and I. Baturone, "Post-quantum biometric authentication based on homomorphic encryption and classic McEliece," *Applied Sciences*, vol. 13, no. 2, p. 757, 2023.
[19] S. Basu and S. H. Islam, "Quantum-attack-resilience OTP-based multi-factor mutual authentication and session key agreement scheme for mobile users," *Computers and Electrical Engineering*, vol. 119, p. 109495, 2024.
[20] J. Jiang, D. Wang, G. Zhang, and Z. Chen, "Quantum-resistant password-based threshold single-sign-on authentication with updatable server private key," presented at the European Symposium on Research in Computer Security, Springer, 2022, pp. 295–316.
[21] S. Lu and X. Li, "Quantum-resistant lightweight authentication and key agreement protocol for fog-based microgrids," *IEEE Access*, vol. 9, pp. 27588–27600, 2021.
[22] M. T. Damir, T. Meskanen, S. Ramezanian, and V. Niemi, "A beyond-5G authentication and key agreement protocol," presented at the International Conference on Network and System Security, Springer, 2022, pp. 249–264.
[23] F. Qian, A. Gerber, Z. M. Mao, S. Sen, O. Spatscheck, and W. Willinger, "TCP revisited: a fresh look at TCP in the wild," presented at the Proceedings of the 9th ACM SIGCOMM conference on Internet measurement, 2009, pp. 76–89.
[24] A. Langley *et al.*, "The quic transport protocol: Design and internet-scale deployment," presented at the Proceedings of the conference of the ACM special interest group on data communication, 2017, pp. 183–196.
[25] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on information theory*, vol. 29, no. 2, pp. 198–208, 1983.
[26] National Institute of Standards and Technology (US), "SHA-3 standard : permutation-based hash and extendable-output functions," National Institute of Standards and Technology, Washington, D.C., 2015. doi: 10.6028/nist.fips.202.
[27] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe, "Post-quantum key {Exchange—A} new hope," presented at the 25th USENIX Security Symposium (USENIX Security 16), 2016, pp. 327–343.
[28] R. Canetti and H. Krawczyk, "Analysis of key-exchange protocols and their use for building secure channels," presented at the International conference on the theory and applications of cryptographic techniques, Springer, 2001, pp. 453–474.
[29] A. Biryukov, D. Dinu, and D. Khovratovich, "Argon2: new generation of memory-hard functions for password hashing and other applications," presented at the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, 2016, pp. 292–302.
[30] "Password Hashing Competition." Accessed: July 09, 2025. [Online]. Available: https://www.password-hashing.net/
[31] B. Li and D. Micciancio, "On the security of homomorphic encryption on approximate numbers," presented at the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2021, pp. 648–677.
[32] A. K. Jain, A. Ross, and U. Uludag, "Biometric template security: Challenges and solutions," presented at the 2005 13th European signal processing conference, IEEE, 2005, pp. 1–4.
[33] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," presented at the IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320), IEEE, 1999, pp. 126–134.
[34] S. Sen and J. Wang, "Analyzing peer-to-peer traffic across large networks," presented at the Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment, 2002, pp. 137–150.
[35] "Intel® Core™ i7 processor 14700K (33M Cache, up to 5.60 GHz) - Product Specifications," Intel. Accessed: Sept. 28, 2025. [Online]. Available: https://www.intel.com/content/www/us/en/products/sku/236783/intel-core-i7-processor-14700k-33m-cache-up-to-5-60-ghz/specifications.html
[36] "Snapdragon X Elite." Accessed: Sept. 29, 2025. [Online]. Available: https://www.qualcomm.com/products/mobile/snapdragon/laptops-and-tablets/snapdragon-x-elite
[37] "Snapdragon 8 Elite Mobile Platform." Accessed: Sept. 29, 2025. [Online]. Available: https://www.qualcomm.com/products/mobile/snapdragon/smartphones/snapdragon-8-series-mobile-platforms/snapdragon-8-elite-mobile-platform
[38] "Advantech Introduces SOM-6820 : A New Era of Power Efficiency and Edge Intelligence with the Qualcomm Snapdragon® X-Elite Series Processor." Accessed: Oct. 02, 2025. [Online]. Available:

https://www.advantech.com/emt/resources/news/advantech-introduces-som-6820--a-new-era-of-power-efficiency-and-edge-intelligence-with-the-qualcomm-snapdragon®-x-elite-series-processor
[39] G. Seth, "Announcing .NET 8," .NET Blog. Accessed: July 09, 2025. [Online]. Available: https://devblogs.microsoft.com/dotnet/announcing-dotnet-8/
[40] "Introduction | libsodium." Accessed: July 09, 2025. [Online]. Available: https://doc.libsodium.org
[41] "Bouncy Castle open-source cryptographic APIs," Bouncycastle. Accessed: July 09, 2025. [Online]. Available: https://www.bouncycastle.org/
[42] A. Kent, "Anonymized User-Computer Authentication Associations in Time." Los Alamos National Laboratory (LANL), Los Alamos, NM (United States), p. 1 file, 2014. doi: 10.11578/1160076.
[43] H. Chen, K. Laine, and R. Player, "Simple encrypted arithmetic library-SEAL v2. 1," presented at the Financial Cryptography and Data Security: FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers 21, Springer, 2017, pp. 3–18.
[44] G. Williams and P. Kanapathipillai, "Qualcomm Oryon CPU in Snapdragon X Elite: Micro-Architecture and Design," *IEEE Micro*, vol. 45, no. 3, pp. 8–14, June 2025, doi: 10.1109/MM.2025.3568807.
[45] F. Boemer, S. Kim, G. Seifu, F. DM de Souza, and V. Gopal, "Intel HEXL: accelerating homomorphic encryption with Intel AVX512-IFMA52," presented at the Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography, 2021, pp. 57–62.
[46] J. Deng *et al.*, "Heterogeneous Computing Platform for Power-Performance Efficient On-Device AI," in *2024 IEEE International Electron Devices Meeting (IEDM)*, Dec. 2024, pp. 1–4. doi: 10.1109/IEDM50854.2024.10873428.
[47] D. Meltzer and D. Luengo, "ECG-Based Biometric Recognition: A Survey of Methods and Databases," *Sensors*, vol. 25, no. 6, p. 1864, Mar. 2025, doi: 10.3390/s25061864.
[48] R. Donida Labati, V. Piuri, F. Rundo, and F. Scotti, "Photoplethysmographic biometrics: A comprehensive survey," *Pattern Recognition Letters*, vol. 156, pp. 119–125, Apr. 2022, doi: 10.1016/j.patrec.2022.03.006.
[49] A. Zamil, "Esiur - Distributed Resource Framework," Esiur. Accessed: July 10, 2025. [Online]. Available: https://www.esiur.com