Password-Based AKEM and HPKE

Axel Durbet[1], Reihaneh Safavi-Naini[1], and Jean-François Biasse[2]

[1] University of Calgary, Calgary, Alberta, Canada
[2] University of South Florida, Tampa, Florida, USA

Abstract. Secure Internet communication requires both confidentiality and mutual authentication. While the Hybrid Public-Key Encryption (HPKE) framework (Alwen *et al.* , Eurocrypt 2021) provides these guarantees, it assumes that users possess (certified) public keys and can securely manage them across devices. In this work, we introduce two primitives: *Password-Authenticated Key Encapsulation Mechanisms* (PAKEM) and *Password-Authenticated Hybrid Encryption* (PAHE). PAKEM enables a client holding only a password to establish an authenticated shared key with a server possessing a (certified) public key, while PAHE additionally enables password-authenticated encryption of arbitrary-length messages in the same setting. For both primitives, we formalize the syntax and define security models capturing privacy (CCA-security of the derived key or encrypted message) and authenticity (resistance against ciphertext forgery). We present a generic construction of PAKEM, prove that it achieves the desired security properties and give an instantiation using NIST post-quantum-standardized cryptographic primitives. We further establish a composition theorem showing that combining a secure PAKEM with a Data Encapsulation Mechanism (DEM) yields a secure PAHE scheme.

Keywords: Password Authentication · KEM · PKE

1 Introduction.

Private communication over the Internet relies on public key cryptography to establish a shared secret key between the communicating parties, and symmetric key cryptography to efficiently encrypt (arbitrary-length) messages. To achieve private communication with provable security, *hybrid encryption* composes a public-key Key Encapsulation Mechanism (KEM), which establishes a fresh shared key between the parties, with a symmetric-key Data Encapsulation Mechanism (DEM) that uses the shared key to encrypt the message. The *KEM/-DEM* paradigm was formalized by Cramer and Shoup [22] and further analyzed by a number of authors [30,31,1]. It is used in widely deployed standards such as S/MIME [42] and OpenPGP [19], and serves as the formal foundation for HPKE [8]. While this paradigm ensures confidentiality of the communication, it does not provide sender authentication or ciphertext integrity (INT-CTXT): a CCA-secure DEM guarantees that an adversary learns nothing about the plaintext from tampered ciphertexts, but does not prevent forging ciphertexts that decrypt to valid messages.

The HPKE Standard and its Applications. The *Hybrid Public-Key Encryption* (HPKE) standard (RFC 9180 [8]) extends the KEM/DEM paradigm to provide support for authentication and integrity, in addition to confidentiality, of the communication. Alwen *et al.* [3] formalized the security of HPKE by introducing *Authenticated KEM (AKEM)* and *Authenticated Public-Key Encryption (APKE)*, and defining their security against two types of adversaries: a weaker *outsider adversary* that does not have access to the private keys of the communicating parties, and a stronger *insider adversary* that has access to the secret key of Alice or Bob. Security against an insider adversary is desirable as, for example, it guarantees that an honest sender whose key is leaked retains security in future communications. The security notions of AKEM against the two types of adversaries are *key indistinguishability*, which requires the derived shared key to be indistinguishable from a uniformly random key, and *key authenticity*, which ensures that a valid shared key can only be established by a legitimate sender and recovered by the intended receiver. APKE is an authenticated encryption system that can be constructed by composing an AKEM and a DEM with appropriate security properties. HPKE has since become a core building block of modern Internet standards, including *Encrypted Client Hello* (ECH) for TLS 1.3 [9], *Oblivious DNS over HTTPS* (ODoH) [38], *Oblivious HTTP* (OHTTP), and the *Messaging Layer Security* (MLS) group messaging protocol.

Limitations of HPKE and the Password Setting and Our Motivation. HPKE's authenticated mode requires both parties to hold long-term KEM key pairs with (certified) public keys. While practical in organizational settings, it is unrealistic to expect individual users to obtain and maintain (certified) long-term public keys. Additionally, the corresponding private keys must be available on every device from which the user wishes to communicate. This requires secure provisioning or synchronization of private keys across devices which would be a significant operational burden. The dominant form of Internet authentication is asymmetric: users authenticate with a password, while servers present a certified public key. This is the setting for almost every online service.

Our goal. We aim to achieve functionality similar to HPKE's authenticated mode when the client authenticates using a password registered with the server, instead of a long-term public-private key pair. Although the HPKE has a Pre-Shared Key (PSK) mode [4] in which user authentication is through a shared secret with the server[3] , the secret has *high entropy* and has the same practical challenges of establishing and securely managing a high-entropy secret across multiple devices, as a long-term key pair. A user password, on the other hand, can be memorized by the user and removes the need for secure storage on the user devices. Passwords, however, are low entropy secrets and password-based protocols are susceptible to offline dictionary attacks either against the protocol transcripts or against user's registered information from a compromised server.

[3] A PSK also enhances post-quantum (PQ) security: if the KEM is broken by a quantum adversary, security is preserved as long as the PSK remains secret.

1.1 Our work.

We introduce *Password-Authenticated Key Encapsulation Mechanism* (PAKEM) and *Password-Authenticated Hybrid Encryption* (PAHE), define their respective security notions in line with AKEM and APKE, and prove composition theorems that relate the security properties of PAHE to those of PAKEM and DEM. PAKEM establishes a shared key between a server holding a KEM key pair and a client who has registered a password with the server. The key derived by PAKEM is tied to the client's password and the server's private key, providing implicit mutual authentication. PAHE is an authenticated hybrid encryption scheme for the same client-server setting, supporting messages of unrestricted length. A secure PAHE can be constructed by composing a PAKEM with a DEM, using the key derived by PAKEM as the DEM's secret key.

Security definitions of PAKEM and PAHE. We consider outsider and insider adversaries and define the following security properties: for PAKEM, key indistinguishability and key authenticity; and for PAHE, message confidentiality and ciphertext authenticity — each defined for both adversary types. We also define *password safety* for PAKEM, requiring that the PAKEM ciphertext and its associated session key do not leak any information about the password, and that verifying a password guess requires online interaction with the server. We define an analogous notion of password safety for PAHE.

Constructions. In Section 4 we give a generic construction of PAKEM (Construction 1) from a CCA-secure KEM, an EUF-CMA-secure digital signature scheme, an IND-CCA-secure symmetric encryption scheme, and a hash function modeled as a random oracle, and prove its correctness, insider key indistinguishability, insider key authenticity, and password safety. We note that insider security implies outsider security, so the construction satisfies all PAKEM security properties. We also give an instantiation of this construction using NIST PQ-standardized cryptographic primitives: ML-KEM [5] (CRYSTALS-Kyber) as the IND-CCA-secure KEM, ML-DSA [6] (CRYSTALS-Dilithium) as the EUF-CMA-secure signature scheme, AES-256-GCM [24] as an IND-CCA-secure symmetric encryption scheme, and SHA3-512 as the hash function. We discuss parameter selection for different NIST security levels, and present numerical results of our proof-of-concept implementation. In Section 6 we give a generic construction of PAHE by composing a PAKEM with a DEM (Construction 2), and prove insider IND-CCA (and so outsider IND-CCA) and outsider authenticity of the composition, provided the PAKEM satisfies the corresponding security notions and the DEM is IND-CCA-secure and INT-CTXT-secure, respectively.[4]

1.2 Related work.

AKEM and HPKE. AKEM and APKE were introduced by Alwen *et al.* [3], who defined multi-user security notions for AKEM and APKE, and proved security of

[4] IND-CCA and INT-CTXT correspond to ciphertext confidentiality and ciphertext integrity, respectively.

the DH-AKEM underlying HPKEAuth. HPKE was subsequently standardized as RFC 9180 [8]. Our security model is an adaptation of the single-user model of [3] to the password setting.

PAPKE. Bradley *et al.* [18] proposed *Password-Authenticated Public-Key Encryption (PAPKE)*, the closest related primitive to PAHE. PAPKE's goal is to provide secure end-to-end encryption, in particular, security against man-in-the-middle attacks, between two entities using a *shared password*. PAPKE's security is modeled using the Universal Composability (UC) framework and does not distinguish between insider and outsider adversaries as PAHE does. A fundamental structural difference between PAPKE and PAHE is that in PAPKE the public key is derived from a shared password and is therefore specific to a pair of users: it cannot serve as a server's public key usable by multiple clients. PAHE, by contrast, uses a single server key pair that is generated independently of any password and supports any number of registered clients, each authenticating with their own password. In PAHE, user enrollment provides the server with registration information (*e.g.* , a digest of the password) for each user; in PAPKE, there is no such enrollment step: the password is embedded directly into key generation and encryption. This distinction leads to a further difference in adversarial power. The security of PAPKE relies on the secrecy of the password *and* the private key, so exposure of either renders the system completely insecure. In PAHE, however, insider adversaries may have access to either the password or the private key, and meaningful security properties are preserved in each case. A stolen password does not compromise the confidentiality of the legitimate user's communications (insider IND-CCA), while the leakage of the sender's secret key does not compromise authenticity (Insider-Auth). In particular, the PAHE generic construction achieves proved insider IND-CCA security, which implies outsider IND-CCA security as well.

KEM. The KEM/DEM framework was introduced by Cramer and Shoup [22] as a modular approach to hybrid encryption. Kurosawa and Desmedt [39] showed that the KEM need not be fully IND-CCA secure for the hybrid to achieve CCA security, and Abe *et al.* [1] introduced the Tag-KEM/DEM framework as a generalization. Necessary and sufficient conditions for secure KEM/DEM composition were analyzed by Herranz *et al.* [31]. KEM security has been further refined: [33] studied hybrid encryption from weakened KEM security notions, and Cremers *et al.* [23] formalized *binding properties* of KEMs, capturing attacks not addressed in earlier definitions. The KEM/DEM framework has been widely adopted, serving as the foundation for standards such as TLS [43] and HPKE [8]. *KEM variants* extend the paradigm to richer settings, including Identity-Based KEM [7,44,14,20,21] and Attribute-Based KEM [27].

PAKE and aPAKE. Passwords have previously been used to provide authentication in Password Authenticated Key Exchange (PAKE) and augmented PAKE (aPAKE) protocols [13,29,45,28,47,35]. In PAKE, communicating parties share the same password, whereas in aPAKE the server stores only a derived value of the password. However, these protocols typically require multiple rounds of interaction and using the resulting session key for encryption generally requires

additional security properties for the composition. In contrast, PAKEM provides implicit password-based client authentication in a single, non-interactive encapsulation step. Moreover, when combined with a data encapsulation mechanism (DEM) satisfying appropriate security properties, PAHE provides authenticated encryption with formally proven security guarantees.

1.3 Outline.

Section 2 presents background information. Section 3 introduces PAKEMs together with their security notions. Section 4 presents a generic PAKEM construction and its security analysis, while Section 5 provides a concrete instantiation based on state-of-the-art cryptographic primitives. Section 6 introduces PAHEs, their associated security notions, and the corresponding composition theorems. Finally, Section 7 concludes the paper.

2 Preliminaries

In this section we recall the background information and introduce the notations used throughout this paper.
Notations. We use the following notations. The set $\{0,1\}^*$ denote the set of bit strings of an unspecified length and $\emptyset$ denote the empty set. We use λ to denote the security parameter, $\mathsf{negl}(\lambda)$ to denote the set of negligible functions and $\mathcal{A}$ a Probabilistic Polynomial-Time (PPT) adversary. The symbol "$\leftarrow$" denotes the assignment of a variable, and "$x \xleftarrow{\$} \mathcal{X}$" denotes x is sampled uniformly at random from the set $\mathcal{X}$. The set of passwords is denoted by $\mathcal{P}$, the set of messages is denoted by $\mathcal{M}$, the set of ciphertexts is denoted by $\mathcal{C}$ and the set of keys is denoted by $\mathcal{K}$. Expressions enclosed in brackets $[\cdot]$ denote the evaluation of a logical expression, evaluating to 1, if the statement is true.
Key Encapsulation Mechanism (KEM). A KEM is a tuple of probabilistic algorithms (KeyGen, Encaps, Decaps) defined as follows:

- **Key Generation**: $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ outputs a public/secret key pair.
- **Encapsulation**: $(c, k) \leftarrow \mathsf{Encaps}(pk)$ outputs a ciphertext c and a key k.
- **Decapsulation**: $k' \leftarrow \mathsf{Decaps}(sk, c)$ recovers the session key or returns $\perp$.

Correctness of KEM requires that for all (pk, sk) and (c, k) output by $\mathsf{Encaps}(pk)$, we have $\mathsf{Decaps}(sk, c) = k$. Informally, a KEM is *IND-CCA secure* if no PPT adversary $\mathcal{A}$ can distinguish an encapsulated key from a random key while given access to a decapsulation oracle. Let $\lambda \in \mathbb{N}$ be a security parameter, $\mathcal{A}$ a PPT adversary and the decapsulation oracle $\mathcal{O}_{\mathrm{Decaps}}$. The experiment [32, Figure 4] $\mathsf{Exp}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}}$ is formally defined in Figure 1.
Data Encapsulation Mechanism (DEM). A DEM is a tuple of (probabilistic) polynomial time algorithms (Gen, Enc, Dec) as follows:

- **Key Generation**: $k \leftarrow \mathsf{Gen}(1^\lambda)$ outputs a secret key k.

– **Encryption**: $c \leftarrow \mathsf{Enc}(k, m)$: takes a message $m \in \mathcal{M}$ and uses the key $k \in \mathcal{K}$, to generate the ciphertext $c \in \mathcal{C}$.
– **Decryption**: $\{m, \perp\} \leftarrow \mathsf{Dec}(k, c)$: takes the ciphertext $c \in \mathcal{C}$ and the key $k \in \mathcal{K}$, and outputs either the message $m \in \mathcal{M}$ or a failure symbol $\perp$.

Correctness requires that for all $k \in \mathcal{K}$ and $m \in \mathcal{M}$, we have $\mathsf{Dec}(k, \mathsf{Enc}(k, m)) = m$. Security of DEM scheme is defined in terms of indistinguishability of the ciphertexts of two messages that are chosen by the adversary. An *IND-CPA secure* [36, Chapter 3] (resp. *IND-CCA* [36, Chapter 3]) secure scheme [10, Figure 4] requires that no PPT adversary $\mathcal{A}$ succeeds in the indistinguishability game with non-negligible probability, when given access to an encryption oracle (*resp.* access to an encryption and decryption oracle). A DEM scheme has *ciphertext integrity* [11, Figure 5] (INT-CTXT) if no PPT adversary $\mathcal{A}$ can produce with non negligible probability a fresh ciphertext that decrypts correctly (the decryption does not return $\perp$). Let $\lambda \in \mathbb{N}$ be a security parameter, $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ a PPT adversary, the encryption oracle $\mathcal{O}_{\mathrm{Enc}}$ and the decryption oracle $\mathcal{O}_{\mathrm{Dec}}$. The experiments $\mathsf{Exp}^{\mathsf{IND\text{-}CCA}}_{\mathsf{Enc}}$ and $\mathsf{Exp}^{\mathsf{INT\text{-}CTXT}}_{\mathsf{Enc}}$ are defined in Figure 2.
Signature [36]. A digital signature scheme is a tuple of polynomial-time algorithms $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ as follows:

– **Key Generation**: $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ generates a pair of keys (pk, sk) with pk the public key (verification key) and sk secret key (signing key).
– **Sign**: $\sigma \leftarrow \mathsf{Sign}(sk, m)$ outputs a signature σ on a message $m \in \mathcal{M}$ using the secret key sk.
– **Verify**: $b \leftarrow \mathsf{Vrfy}(pk, \sigma, m)$ outputs a bit b, with $b = 1$ if the signature σ is valid for message m under public key pk, and $b = 0$ otherwise.

A signature scheme must satisfy correctness, *i.e.* , any honestly generated signature are accepted with an overwhelming probability. A signature scheme is said to be *unforgeable* if it is Existentially Unforgeable under Chosen Message Attacks (EUF-CMA) [12, Section 2.2], *i.e.* , it is computationally infeasible for an adversary to produce a valid signature σ on a fresh message. The experiment $\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{Sig}}$ (EUF-CMA) for a security parameter $\lambda \in \mathbb{N}$, a PPT adversary $\mathcal{A}$, and the signing oracle $\mathcal{O}_{\mathrm{Sign}}$ is defined in Figure 3. We write $\mathsf{KeyGen}(1^\lambda; r)$ to explicitly show the randomness, r of the key generation. This is used in the Fujisaki–Okamoto transformation [32], where the encryption algorithm derives its randomness from the hash of the message.
AKEM and HPKE Security Model. Alwen et al. [3] introduced security notions for AKEM and HPKE, which we adapt to PAKEM and PAHE. Their framework distinguishes between *outsider* adversaries, who interact with encapsulation/encryption and decapsulation/decryption oracles without possessing any secret information, and *insider* adversaries, who additionally know one party's secret (*e.g.* , a password or decapsulation key). The two core security properties *Privacy* and *Authenticity* in both the insider and outsider setting yields four games (Games are given in [3, Listing 3-7]): **Outsider-CCA:** An adversary cannot distinguish the session key from random given the associated ciphertext. **Insider-CCA:** The same guarantee holds even if the adversary additionally knows the sender's

$\underline{\mathsf{Exp}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}}(\lambda)}$:
$(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$; $(c^*, k_0) \leftarrow \mathsf{Encaps}(pk)$
$k_1 \xleftarrow{\$} \mathcal{K}$; $b \xleftarrow{\$} \{0,1\}$; $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Decaps}}}(pk, c^*, k_b)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Decaps}}(c)}$:
If $[c = c^*]$, **Return** $\perp$.
$k \leftarrow \mathsf{Decaps}(sk, c)$
Return k

Fig. 1: IND-CCA Experiment for KEM with a security parameter $\lambda \in \mathbb{N}$, a PPT adversary $\mathcal{A}$ and the decapsulation oracle $\mathcal{O}_{\mathrm{Decaps}}$.

$\underline{\mathsf{Exp}^{\mathsf{IND\text{-}CCA}}_{\mathsf{Enc}}(\lambda)}$:
$k \leftarrow \mathsf{Gen}(1^\lambda)$; $Q \leftarrow \emptyset$
$m_0, m_1, st \leftarrow \mathcal{A}_0^{\mathcal{O}_{\mathrm{Enc}}, \mathcal{O}_{\mathrm{Dec}}}(1^\lambda)$
$b \xleftarrow{\$} \{0,1\}$; $c_b \leftarrow \mathsf{Enc}(k, m_b)$
$b' \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathrm{Enc}}, \mathcal{O}_{\mathrm{Dec}}}(st, 1^\lambda, c_b)$
Return $[b = b']$

$\underline{\mathsf{Exp}^{\mathsf{INT\text{-}CTXT}}_{\mathsf{Enc}}(\lambda)}$:
$k \leftarrow \mathsf{Gen}(1^\lambda)$; $Q \leftarrow \emptyset$
$c^* \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Enc}}}(1^\lambda)$
$m^* \leftarrow \mathsf{Dec}(k, c^*)$
$b \leftarrow [[m^* \neq \perp] \wedge [c^* \notin Q]]$
Return b

$\underline{\mathcal{O}_{\mathrm{Enc}}(m)}$:
$c \leftarrow \mathsf{Enc}(k, m)$
$Q \leftarrow Q \cup \{c\}$
Return c
$\underline{\mathcal{O}_{\mathrm{Dec}}(c)}$:
If $[c = c_b]$: **Return** $\perp$
Return $m \leftarrow \mathsf{Dec}(k, c)$

Fig. 2: IND-CCA and INT-CTXT Experiment for DEM with the decryption oracle $\mathcal{O}_{\mathrm{Dec}}$ and the PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$.

$\underline{\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{Sig}}(\lambda)}$:
$(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$; $Q \leftarrow \emptyset$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Sign}}}(pk)$; $b \leftarrow \mathsf{Vrfy}(pk, m^*, \sigma^*)$
Return $[[b = 1] \wedge [m^* \notin Q]]$

$\underline{\mathcal{O}_{\mathrm{Sign}}(m)}$:
$\sigma \leftarrow \mathsf{Sign}(sk, m)$
$Q \leftarrow Q \cup \{m\}$
Return σ

Fig. 3: The EUF-CMA Exp. for Signature with the signing oracle $\mathcal{O}_{\mathrm{Sign}}$.

secret. **Outsider-Auth:** An outsider cannot forge a ciphertext accepted as originating from a legitimate party. **Insider-Auth:** Even knowing the receiver's secret, the same guarantee holds.

3 Password Authenticated KEM

Password Authenticated Key Encapsulation Mechanism (PAKEM), a KEM that provides user authentication through tying the key to the user's password. PAKEM can be seen as an asymmetric AKEM [3] in which one party (the client) derives authentication from a password rather than from a public key, while the other party (the server) retains a standard authenticated key pair. In this section, we examine how this shift in setting affects the original definitions of AKEM (key indistinguishability and authenticity).

3.1 Security Definitions

PAKEM is a cryptographic primitive that enables two parties to securely derive a shared session key bound to their respective private information (the password on the client side and the secret key on the server side) thereby providing implicit or explicit authentication through the successful derivation of this key.

We formalize security following the security game approach of [3]. A game G is a probabilistic experiment describing the interaction of an adversary $\mathcal{A}$ with an implicit challenger, where the adversary queries prescribed oracles answered by the challenger. Depending on the game, the adversary has access to *encapsulation* and *decapsulation* oracles, which model the encapsulation and decapsulation algorithms run with the appropriate secret inputs. There are also *challenge* oracles whose responses depend on a secret bit sampled at the start of the game. The game G outputs a bit, and A is said to win if the output is 1.

Definition 1 (PAKEM). *A Password Authenticated Key Encapsulation Mechanism (PAKEM) is a tuple of four probabilistic algorithms* (Gen, Enrol, Encaps, Decaps)*, where:*

- ***Key Generation:*** $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$ *is a randomized procedure that takes as input the security parameter* $\lambda \in \mathbb{N}$ *and outputs a public key* pk *and secret key* sk*. The structure of* (pk, sk) *is scheme-specific.*
- ***Enrollment:*** $d \leftarrow \mathsf{Enrol}(pw, pk)$ *is a (randomized) procedure that takes as input a password* pw*, and a public key and outputs the registry information* d *derived from* pw*.*
- ***Encapsulation:*** $(k, c) \leftarrow \mathsf{Encaps}(pk, pw)$ *is a randomized procedure that takes as input the public key* pk *and the password* pw*, and outputs a key* k *and a ciphertext* c*.*
- ***Decapsulation:*** $k \leftarrow \mathsf{Decaps}(sk, d, c)$ *is a deterministic algorithm that, given the secret key* sk*, the registry information* d*, and a ciphertext* c*, outputs the corresponding session key* k*, or a special symbol* $\perp$ *indicating failure (e.g. , incorrect password).*

In AKEM [3], Gen generates a pair of public and secret keys intended for each user. In our setting, Gen derives the server public and secret key while Enrol produces the user related information that will be stored by the server. After those two procedures, the client's secret information is a password pw, the server's secret information is the pair (sk, d), and its corresponding public information is pk. In AKEM, Encaps takes as input a sender's secret key sk and a receiver's public key pk, and returns a ciphertext c and a shared key k. In our construction, since the sender's secret key is replaced by a password, Encaps instead takes the client's password and the server's public key as input, and outputs both a ciphertext and a session key. Similarly, Decaps in AKEM takes as input a receiver's secret key sk, a sender's public key pk, and a ciphertext c, and outputs a shared key k. In PAKEM, the sender's public key is replaced by the registered information d, while all other inputs and outputs remain the same.

Correctness. A PAKEM is said to be correct if the encapsulation and decapsulation procedures agree on the same key whenever both parties behave honestly and have the corresponding secrets. We formalize this notion as follows.

$\underline{\mathsf{Exp}^{\mathsf{COR}}_{\mathsf{PAKEM}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$; $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$; $d \leftarrow \mathsf{Enrol}(pw, pk)$
$(k, c) \leftarrow \mathsf{Encaps}(pk, pw)$; $k' \leftarrow \mathsf{Decaps}(sk, d, c)$
Return $[k \neq k']$

Fig. 4: Correctness Experiment for PAKEM with security parameter $\lambda \in \mathbb{N}$.

Definition 2 (ε-Correctness). *A PAKEM is said to be ε-correct if, for all security parameters $\lambda \in \mathbb{N}$, $\Pr\left[\mathsf{Exp}^{\mathsf{COR}}_{\mathsf{PAKEM}}(\lambda) = 1\right] \leq \varepsilon$, where ε is a negligible function in λ and the probability is taken over the internal randomness of the experiment $\mathsf{Exp}^{\mathsf{COR}}_{\mathsf{PAKEM}}(\lambda)$ defined in Figure 4,.*

In AKEM [3], they expect $\Pr\left[\mathsf{Exp}^{\mathsf{COR}}_{\mathsf{AKEM}}(\lambda) = 1\right] = 0$. While this requirement can be met in pre-quantum settings, achieving perfect correctness is challenging in most post-quantum schemes (*e.g.* , Kyber [5] or Frodo [16]). To account for this, we relax the requirement and consider ε-correctness, where the probability of a decapsulation error is bounded by a negligible function ε in λ.

3.2 Security of PAKEM

In line with AKEM security properties, we aim to achieve two central security properties. The first security notion follows the standard KEM paradigm [31,22] by enforcing that the derived session key must remain indistinguishable from a random key, even when the adversary is allowed a polynomial number of decapsulation queries. This property, known as IND-CCA key security (called privacy in [3]), ensures the confidentiality of the session key against active adversaries. Second, since the derived key also provides explicit authentication of the communicating parties (the client through the password and the server through its registered information and secret key) we require that no adversary can forge a PAKEM ciphertext that would be accepted by the decapsulation algorithm (this property is called authenticity in [3]). It is analogous to ciphertext unforgeability in authenticated encryption [11].

Key Indistinguishability. Key Indistinguishability captures the requirement that PAKEM ciphertexts generated by an honest sender for an honest and uncompromised receiver do not reveal any information about the encapsulated

shared key, *i.e.* , from the adversary's perspective, the key remains indistinguishable from a random key. Akin to AKEM [3], this notion splits into two cases: The first is the outsider case, in which the adversary does not hold any private information. The second is the insider case, in which the adversary knows the password used for encapsulation.

$\underline{\mathsf{Exp}^{(q_e,q_d)\text{-Outsider-CCA}}_{\mathsf{PAKEM}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$; $(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$
$d \leftarrow \mathsf{Enrol}(pw, pk)$; $(c^*, k_0) \leftarrow \mathsf{Encaps}(pk, pw)$
$k_1 \xleftarrow{\$} \mathcal{K}$; $b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Encaps}}, \mathcal{O}_{\mathrm{Decaps}}}(pk, c^*, k_b)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Encaps}}}$:
$(c', k') \leftarrow \mathsf{Encaps}(pk, pw)$
Return (c', k')
$\underline{\mathcal{O}_{\mathrm{Decaps}}(c)}$:
If $[c = c^*]$, **Return** $\perp$.
$k \leftarrow \mathsf{Decaps}(sk, d, c)$
Return k

Fig. 5: (q_e, q_d)-Outsider-CCA Experiment for PAKEM with q_e accesses to the encapsulation oracle $\mathcal{O}_{\mathrm{Encaps}}$ and q_d accesses to the decapsulation oracle $\mathcal{O}_{\mathrm{Decaps}}$.

$\underline{\mathsf{Exp}^{(q_d,q_c)\text{-Insider-CCA}}_{\mathsf{PAKEM}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$
$(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$
$d \leftarrow \mathsf{Enrol}(pw, pk)$
$b \xleftarrow{\$} \{0,1\}$
$Q_E \leftarrow \emptyset$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Decaps}}, \mathcal{O}_{\mathrm{Chal}}}(pk, pw)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Decaps}}(c)}$:
If $\exists k : (c, k) \in Q_E$: **Return** k
$k \leftarrow \mathsf{Decaps}(sk, d, c)$; $Q_E \leftarrow Q_E \cup \{(c, k)\}$
Return k
$\underline{\mathcal{O}_{\mathrm{Chal}}}$:
$(c', k') \xleftarrow{\$} \mathsf{Encaps}(pk, pw)$
If $[b = 1]$, $k' \xleftarrow{\$} \mathcal{K}$
$Q_E \leftarrow Q_E \cup \{(c', k')\}$
Return (c', k')

Fig. 6: (q_d, q_c)-Insider-CCA Experiment for PAKEM with q_d accesses to the decaps. oracle $\mathcal{O}_{\mathrm{Decaps}}$ and q_c accesses to the challenge oracle $\mathcal{O}_{\mathrm{Chal}}$.

Outsider Key Indistinguishability. The experiment proceeds as follows. First, the parameters are generated using Gen and Enrol, which provides the server with its public key, secret key, and enrollment information, and the client with its password[5] . Next, a random bit b is sampled, and a key is encapsulated using Encaps, which takes as input the server's public key and the client's password, producing a ciphertext c^* and a key k. If $b = 0$, the key k is left unchanged; if $b = 1$, it is replaced by a random key. The adversary wins if it can correctly distinguish whether the provided key k corresponds to the real encapsulated

[5] As in BRP [10, Section 2, paragraph Long-lived keys] and many PAKE schemes (*e.g.* , [37, Section 2.1]), the password is assumed to be drawn uniformly at random in a known dictionary denoted by $\mathcal{P}$ during the initialization.

key ($b = 0$) or a random key ($b = 1$), given the server's public key pk and the ciphertext c^*. The adversary has access to both the encapsulation q_e times and the decapsulation oracles q_d times, under the restriction that it is not allowed to query the decapsulation oracle on the challenge ciphertext c^*.
Insider Key Indistinguishability. The experiment is similar to the outsider case but as $\mathcal{A}$ knows the password, we do not give to the adversary an encapsulation oracle as it can perform this operation by itself. However, to prevent trivial wins by querying a challenge ciphertext, all pairs (ciphertext,key) outputted by the challenge oracle are recorded. If $\mathcal{A}$ queries the decapsulation oracle on a challenge ciphertext, the oracle returns the key associated with that ciphertext independently of whether it is the real decapsulated key or a random value (no actual decapsulation is performed in this case).

Definition 3 (Key Indistinguishability). *A PAKEM scheme is said to achieve* mode*-key indistinguishability for a security parameter* λ *if, for the adversarial model* $\mathsf{mode} \in \{\mathsf{Insider}, \mathsf{Outsider}\}$ *and any PPT adversary* $\mathcal{A}$ *making at most* q_e *encapsulation queries,* q_d *decapsulation queries and* q_c *challenge queries, the advantage of* $\mathcal{A}$ *in the experiment* $\mathsf{Exp}^{(q_e,q_d,q_c)\text{-}\mathsf{mode\text{-}CCA}}_{\mathsf{PAKEM}}$ *(defined in Figure 5 and Figure 6) is negligible in* λ*. Formally,*

$$\mathrm{Adv}^{(q_e,q_d,q_c)\text{-}\mathsf{mode\text{-}CCA}}_{\mathcal{A},\mathsf{PAKEM}} = \left| \Pr\left[\mathsf{Exp}^{(q_e,q_d,q_c)\text{-}\mathsf{mode\text{-}CCA}}_{\mathsf{PAKEM}}(\lambda) = 1\right] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda).$$

Authenticity. The authenticity security notion captures the requirement that an adversary should not be able to produce a valid PAKEM ciphertext that would be accepted by an honest receiver as originating from an honest sender. Following the AKEM framework of Alwen *et al.* [3], we formulate authenticity as a distinguishability game. However, we additionally allow the adversary to win immediately whenever it successfully produces a fresh ciphertext that is accepted by the challenge oracle using the decapsulation oracle. Consequently, our notion preserves the indistinguishability-based formulation of [3] while explicitly accounting for ciphertext forgery. As for confidentiality, we distinguish two attack scenarios: outsider authenticity, where the adversary has no knowledge of any secret keys or passwords, and insider authenticity, where the adversary additionally learns the server's secret.
Outsider Key Authenticity. The experiment proceeds as follows. Similar to the Outsider-CCA experiment, the system parameters are generated using Gen and Enrol before the sampling of a random bit b. This bit determines the behavior of the challenge oracle: given a ciphertext choosen by the adversary, the oracle either returns the genuine decapsulation of the ciphertext or a random key. To succeed in this experiment, the adversary must decide whether the challenge oracle outputs a real or random key associated with its chosen ciphertext. The adversary has access to both the encapsulation and decapsulation oracles to assist in this task. However, certain constraints prevent trivial wins. In particular, the ciphertext submitted to the challenge oracle must be *fresh*, meaning it has not been generated by the encapsulation oracle, or already queried to the challenge

oracle. To enforce this, the experiment maintains a list recording all ciphertexts created and tested by the adversary.

Insider Key Authenticity. In the Insider-Auth scenario, the adversary is assumed to have compromised the receiver's (server's) secrets and aims to forge a ciphertext accepted as originating from the legitimate client. As noted in [3], this captures the classical Key Compromise Impersonation (KCI) attack. Although HPKE does not provide KCI security [3], Section 4 shows that our PAKEM construction can achieve it, motivating a formal Insider-Auth experiment.

The experiment proceeds as follows. Similar to the previous experiments, system parameters are generated by both Gen and Enrol before a random bit b is sampled. As in the Outsider-Auth experiment, this bit determines the behavior of the challenge oracle that has not changed. The adversary's goal remains the same; however, since it knows the server's secrets, namely the secret key sk and the registered information d, it can locally perform the decapsulation without relying on an oracle. To prevent a trivial win, the adversary is not allowed to query the challenge oracle on a ciphertext returned by the encapsulation oracle.

Definition 4 (Key Authenticity). *A PAKEM scheme is said to achieve* mode*-key authenticity for a security parameter λ if, for the adversarial model* $\mathsf{mode} \in \{\mathsf{Insider}, \mathsf{Outsider}\}$ *and any PPT adversary $\mathcal{A}$ making at most q_e encapsulation queries, n password guesses or decapsulation queries, and q_c challenge queries, the advantage of $\mathcal{A}$ in the experiment* $\mathsf{Exp}^{(q_e,n,q_c)\text{-}\mathsf{mode\text{-}Auth}}_{\mathsf{PAKEM}}$ *(defined in Figure 7 and Figure 8) is negligible in λ. Formally, with $\mathcal{P}$ denoting the password space:*

$$\mathrm{Adv}^{(q_e,n,q_c)\text{-}\mathsf{mode\text{-}Auth}}_{\mathcal{A},\mathsf{PAKEM}} = \left| \Pr\left[\mathsf{Exp}^{(q_e,n,q_c)\text{-}\mathsf{mode\text{-}Auth}}_{\mathsf{PAKEM}}(\lambda) = 1\right] - \left(\frac{n}{|\mathcal{P}|} + \frac{1}{2}\right) \right| \leq \mathsf{negl}(\lambda).$$

Note that the adversary may attempt to recover the password by using the decapsulation as an oracle. Since the password space is of size $|\mathcal{P}|$ and the adversary is limited to at most n trials during the experiment, the probability of correctly guessing the password is at most $n/|\mathcal{P}|$. Conditioned on failing to recover the password, the adversary gains no additional advantage over random guessing. Hence, in this case, its success probability is at most $1/2$. Therefore, the baseline success probability is bounded by: $\frac{n}{|\mathcal{P}|} + \frac{1}{2}$.

Password Safety. Passwords are low-entropy secrets and therefore vulnerable to offline dictionary attacks. We therefore require that neither the KEM ciphertext nor the associated session key leak any information about the password, so that password verification necessarily requires online interaction with the server. Our notion of password safety is inline with a similar notion introduced in PASTA [2], a password-based threshold authentication system.

Definition 5 (Password Safety). *A PAKEM scheme is said to be password safe if, for a security parameter λ and for any PPT adversary $\mathcal{A}$ making at most q_e encapsulation queries, and q_d decapsulation queries, the advantage of $\mathcal{A}$ in the experiment* $\mathsf{Exp}^{(q_e,q_c)\text{-}\mathsf{PW\text{-}SAFE}}_{\mathsf{PAKEM}}$ *(defined in Figure 9) is negligible in λ. Formally,*

$\underline{\mathsf{Exp}^{(q_e,q_d,q_c)\text{-Outsider-Auth}}_{\mathsf{PAKEM}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$; $(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$
$d \leftarrow \mathsf{Enrol}(pw, pk)$
$Q_E \leftarrow \emptyset$; $b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Encaps}}, \mathcal{O}_{\mathrm{Decaps}}, \mathcal{O}_{\mathrm{Chal}}}(pk)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Decaps}}(c)}$:
$k \leftarrow \mathsf{Decaps}(sk, d, c)$
Return k

$\underline{\mathcal{O}_{\mathrm{Encaps}}}$:
$(c, k) \leftarrow \mathsf{Encaps}(pk, pw)$
$Q_E \leftarrow Q_E \cup \{(c, k)\}$
Return (c, k)

$\underline{\mathcal{O}_{\mathrm{Chal}}(c')}$:
If $[\exists k : (c', k) \in Q_E]$, **Return** $\perp$
$k' \leftarrow \mathsf{Decaps}(sk, d, c')$
If $[k' = \perp]$, **Return** $\perp$
If $[b = 1]$, $k' \xleftarrow{\$} \mathcal{K}$
$Q_E \leftarrow Q_E \cup \{(c', k')\}$
Return k'

Fig. 7: (q_e, q_d, q_c)-Outsider-Auth Experiment for PAKEM.

$\underline{\mathsf{Exp}^{(q_e,q_c)\text{-Insider-Auth}}_{\mathsf{PAKEM}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$
$(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$
$d \leftarrow \mathsf{Enrol}(pw, pk)$
$Q_E \leftarrow \emptyset$
$b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Encaps}}, \mathcal{O}_{\mathrm{Chal}}}(pk, sk, d)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Chal}}(c')}$:
If $\exists k : (c', k) \in Q_E$: **Return** $\perp$
$k' \leftarrow \mathsf{Decaps}(sk, d, c')$
If $[k' = \perp]$: **Return** $\perp$
If $[b = 1]$, $k' \xleftarrow{\$} \mathcal{K}$
$Q_E \leftarrow Q_E \cup \{(c', k')\}$
Return k'

$\underline{\mathcal{O}_{\mathrm{Encaps}}}$:
$(c, k) \leftarrow \mathsf{Encaps}(pk, pw)$; $Q_E \leftarrow Q_E \cup \{(c, k)\}$
Return (c, k)

Fig. 8: (q_e, q_d, q_c)-Insider-Auth Experiment for PAKEM.

with $\mathcal{P}$ denoting the password space:

$$\mathrm{Adv}^{(q_e,q_d)\text{-PW-SAFE}}_{\mathcal{A},\mathsf{PAKEM}} = \left| \Pr\left[\mathsf{Exp}^{(q_e,q_c)\text{-PW-SAFE}}_{\mathsf{PAKEM}}(\lambda) = 1\right] - \frac{q_d}{|\mathcal{P}|} \right| \leq \mathsf{negl}(\lambda).$$

4 Generic Construction of PAKEM

In this section, we present a generic construction of PAKEM by combining a standard KEM with a signature scheme and a DEM. This design captures both the key authenticity and key indistinguishability requirements of a PAKEM against their respective insider adversaries.

4.1 Generic Construction

Technical Overview. We use a CCA-secure KEM, an EUF-CMA digital signature, an IND-CCA secure DEM, and a hash function modeled as a random oracle,

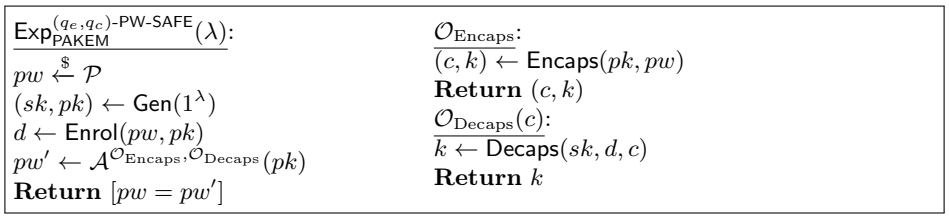

Fig. 9: (q_e, q_d)-PW-SAFE Experiment for PAKEM.

to construct a secure PAKEM (see Figure 1 and Section 4.2). The intuition behind our construction is to derive the signature key pair from the password and use the resulting signing key to authenticate the KEM output, namely the encapsulated key and its associated ciphertext. The resulting PAKEM ciphertext consists of the KEM ciphertext together with the signature encrypted under the encapsulated key. The final PAKEM shared key is derived by hashing the PAKEM ciphertext, the server public key, the signed message, and the signature itself effectively binding the session key to both the sender's password and the server's secret key.

Intuition for Authenticity and Offline Attack Resistance. PAKEM uses the public/secret key pair of the underlying KEM as the receiver's long-term key pair. The sender's password is used during the Enrol algorithm to deterministically derive a public/secret key pair (pk_2, sk_2) for a digital signature scheme. The public key pk_2 is registered with the server, while the sender reconstructs sk_2 from the password whenever encapsulation is performed. Consequently, the signing key does not need to be stored. During encapsulation, the sender first executes the KEM encapsulation to obtain (c_1, k_1) and computes $m = H(c_1 \,\|\, k_1)$. The sender then signs m using sk_2. To prevent offline password attacks, the resulting signature is encrypted under the KEM-derived key k_1 before being included in the ciphertext. The session key is finally derived by hashing all session-specific values, namely the KEM ciphertext, the encrypted signature, the signature verification key, and the shared secret established by the KEM. Since these values depend on both the receiver's long-term secret key and the sender's password-derived signing key, the resulting session key is cryptographically bound to both parties. Upon decapsulation, the receiver recovers the encrypted signature using the KEM key, verifies it with the registered verification key pk_2, and derives the same session key only if the verification succeeds. We stress that (i) the Enrol procedure is assumed to be trusted, and (ii) because sk_2 is deterministically derived from the password, it never needs to be stored by the sender.

Construction 1 (PAKEM from KEM, Signature and DEM) *Let K be a Key Encapsulation Mechanism, S be a signature scheme and E a DEM. Then, a PAKEM scheme can be constructed as shown in Figure 10.*

$\underline{\mathsf{Gen}(\lambda)}$:	$\underline{\mathsf{Enrol}(pk_1, pw)}$:
$(pk_1, sk_1) \leftarrow K.KeyGen(1^\lambda)$	$r \leftarrow H(pw\|\|pk_1)$
Return pk_1, sk_1	$(pk_2, sk_2) \leftarrow S.KeyGen(1^\lambda; r)$
$\underline{\mathsf{Encaps}(pk_1, pw)}$:	**Return** $d = pk_2$
$(c_1, k_1) \leftarrow K.Encaps(pk)$	$\underline{\mathsf{Decaps}(sk_1, pk_2, c = (c_1, c_2))}$:
$m \leftarrow H(c_1\|\|k_1)$	$k_1 \leftarrow K.Decaps(sk_1, c_1)$
$(pk_2, sk_2) \leftarrow S.KeyGen(1^\lambda; H(pw\|\|pk_1))$	$m \leftarrow H(c_1\|\|k_1)$; $\sigma \leftarrow E.Dec(k_1, c_2)$
$\sigma \leftarrow S.Sig(sk_2, m)$; $c_2 \leftarrow E.Enc(k_1, \sigma)$	**If** $S.Vrfy(pk_2, m, \sigma)$:
Return $c = (c_1, c_2), k = H(c_1\|\|c_2\|\|m\|\|\sigma\|\|pk_2)$	**Return** $k = H(c_1\|\|c_2\|\|m\|\|\sigma\|\|pk_2)$
	Else: **Return** $\perp$

Fig. 10: PAKEM from KEM, Signature and DEM

Theorem 1 (Correctness of Construction 1). *Assuming that K is an ε_1-correct KEM, E is a correct DEM and S is an ε_2-correct signature scheme then, the PAKEM given by Construction 1 is $(\varepsilon_1 + \varepsilon_2)$-correct.*

Proof. Let (pk_1, sk_1) be honestly generated by $\mathsf{Gen}(1^\lambda)$, $d = pk_2$ be honestly generated by $\mathsf{Enrol}(pk_1, pw)$ algorithm and let $(c, k) \leftarrow \mathsf{Encaps}(pk_1, pw)$ be the transcript produced by an honest PAKEM encapsulation. Construction 1 yields $c_1, k_1 \leftarrow K.Encaps(pk_1)$, $\sigma \leftarrow S.Sig(sk_2, H(c_1||k_1))$ and $k \leftarrow H(c_1||c_2||m||\sigma||pk_2)$. Note that sk_2 and pk_2 can be recomputed correctly if the password is correct as $S.KeyGen(1^\lambda; r)$ is a deterministic algorithm as the randomness has been fixed. Now consider the decapsulation procedure run by the honest receiver on input (sk_1, pk_2, c). It first computes $k_1' \leftarrow K.Decaps(sk_1, c_1)$. By ε_1-correctness of the KEM K we have $Pr[k_1' = k_1] \geq 1 - \varepsilon_1$. Next the decapsulation runs the signature verification $S.Vrfy(pk_2, H(c_1||k_1), E.Dec(k_1, c_2))$. By the correctness of E, we have $\sigma = E.Dec(k_1, c_2)$. By the ε_2-correctness of the signature scheme S, we get that, whenever $H(c_1||k_1) = H(c_1||k_1')$ (which is the case as H is deterministic function and $k_1 = k_1'$), the verification accepts except with probability at most $\varepsilon_2(\lambda)$. When both events $k_1 = k_1'$ and "the verification accept occurs", the decapsulation computes $k \leftarrow H(c_1||c_2||m||\sigma||pk_2)$ correctly as H is a deterministic function and the result follows. □

4.2 Security

In this section, we discuss the security of Construction 1 with respect to Key Indistinguishability and Key Authenticity.

Key Indistinguishability. We show that the key indistinguishability of Construction 1 follows directly from the IND-CCA security of the underlying KEM. Intuitively, in the Insider-CCA experiment, the adversary is given all secret information held by the client (including the password and the signature keys), but not the server's KEM secret key. Hence, it cannot decapsulate the challenge ciphertext on its own. Because the encapsulated key k_1 produced by the KEM remains

indistinguishable from a random key under IND-CCA security, and since the final PAKEM session key is derived from k_1 through a hash function (modeled as a random oracle) together with adversary/publicly known values, the adversary learns no information about the PAKEM key.

Theorem 2. *Let K be a KEM that is IND-CCA secure, E an IND-CCA secure DEM scheme, S a digital signature scheme*[6] *and H a hash function modeled as a random oracle. Then, Construction 1 is a PAKEM secure against (q_d, q_c)-Insider-CCA adversaries with $(q_d, q_c) \in \mathbb{N}^2_{\geq 1}$.*

Proof. The proof proceeds via a sequence of games. Let $\mathsf{Game}\,0$ denote the original (q_d, q_c)-Insider-CCA experiment for Construction 1, and let $\Pr[G_i(\mathcal{A}) = 1]$ denote the probability that $\mathcal{A}$ wins in Game i.

- **Game 1.** In this game, the challenge oracle is modified so that the key k_1 returned by the KEM is replaced by a uniformly random key. If the adversary $\mathcal{A}$ can distinguish this change, it would break the IND-CCA security of K. Therefore, for any PPT adversary $\mathcal{B}$ against the KEM, we have: $\big|\Pr[G_0(\mathcal{A}) = 1] - \Pr[G_1(\mathcal{A}) = 1]\big| \leq \mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{B},K}$.
- **Game 2.** In this game, we replace the message $m = H(c_1||k_1)$ by a uniformly random value in the message space (*i.e.* , $\{0,1\}^n$ with n the length of the output of the hash function H), under the random oracle assumption. Since H is modeled as a random oracle, this change is indistinguishable to $\mathcal{A}$: $\big|\Pr[G_1(\mathcal{A}) = 1] - \Pr[G_2(\mathcal{A}) = 1]\big| \leq \mathrm{negl}(\lambda)$.
- **Game 3.** In this game, we replace the ciphertext $c_2 = E.Enc(k_1, m)$ by a uniformly random value in the ciphertext space. If the adversary $\mathcal{A}$ can distinguish this change, it would break the IND-CCA security of E. Thus: $\big|\Pr[G_2(\mathcal{A}) = 1] - \Pr[G_3(\mathcal{A}) = 1]\big| \leq \mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{C},E}$.
- **Game 4.** In this game, we replace by a uniformly random value in the key space the output of the hash function $H(c_1||c_2||m||\sigma||pk_2)$ under the random oracle assumption[7] . Since H is modeled as a random oracle, this change is indistinguishable to $\mathcal{A}$: $\big|\Pr[G_3(\mathcal{A}) = 1] - \Pr[G_4(\mathcal{A}) = 1]\big| \leq \mathrm{negl}(\lambda)$.

In Game 4, the ciphertexts and the challenge key are independent. Therefore, the probability that the adversary wins the game is exactly $\frac{1}{2}$ yielding $\Pr[G_i(\mathcal{A}) = 1] = \frac{1}{2}$. Then, we have:

$$\mathrm{Adv}^{(q_d,q_c)\text{-}\mathsf{Insider\text{-}CCA}}_{\mathcal{A},\mathsf{PAKEM}} \leq \mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{B},K} + \mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{C},E} + \mathrm{negl}(\lambda). \qquad \square$$

[6] No assumption is required on the security of the signature scheme. In particular, even if an adversary is able to forge signatures, the construction still remains secure against Outsider-CCA or Insider-CCA adversaries.

[7] The unknown input to the random oracle H includes m, which is the output of a RO and has entropy comparable to the KEM key and ciphertext entropy, as well as pk_2, which has entropy similar to the password. Together, these inputs provide sufficient entropy so that, as long as the key space $\mathcal{K}$ is of reasonable size, the output of H can be considered uniformly distributed over $\mathcal{K}$.

Authenticity. The authenticity of Construction 1 follows from the unforgeability of the underlying signature scheme. Intuitively, any successful forgery in the PAKEM scheme must correspond to forging a valid signature on the message $m = H(c_1||k_1)$ where c_1 is a KEM ciphertext and k_1 is its valid decapsulation. If we allow the adversary to choose m independently of k_1 (thus strictly strengthening the adversary), it must still output a signature σ on m under the public key pk_2 without access to the secret key sk_2 and without having (m, σ) as an output of the encapsulation oracle (that can also be reduce to a signing oracle for the chosen message of the adversary). This is precisely the definition of unforgeability under chosen-message attacks for the signature scheme. Hence, any adversary breaking PAKEM authenticity yields a signature forger.

Theorem 3. *Let K be a KEM, E a DEM, S a digital signature scheme that is EUF-CMA-secure and H a hash function modeled as a random oracle. Then, Construction 1 is a PAKEM secure against (q_d, q_c)-Insider-Auth adversaries with $(q_d, q_c) \in \mathbb{N}^2_{\geq 1}$.*

Proof. The proof proceeds through a sequence of games, starting from the real experiment. Let $\mathsf{Game}\,0$ denote the original (q_d, q_c)-Insider-Auth experiment for Construction 1. We denote by Π the PAKEM defined in Construction 1, and by $\Pr[G_i(\mathcal{A}) = 1]$ denote the probability that $\mathcal{A}$ wins in Game i.

- **Game 1.** In this game, in the Enrol function, we replace r by a random value as H is modeled as a random oracle. However, as pw may have low entropy, the advantage of an adversary to distinguish the real r from a random value is the probability that the adversary guesses the password or distinguishes H from a random oracle. Hence: $\left|\Pr[G_0(\mathcal{A}) = 1] - \Pr[G_1(\mathcal{A}) = 1]\right| \leq \frac{n}{|\mathcal{P}|} + \mathsf{negl}(\lambda)$ with n the number of trials the adversary makes to exhaustively search the password.
- **Game 2.** In Game 2, we observe that $\Pi.\mathsf{Decaps}$ outputs a non-$\perp$ value only if the adversary provides a fresh ciphertext $c^* = (c_1^*, c_2^*)$ such that $S.\mathsf{Vrfy}(pk_2, m^*, \sigma^*) = 1$ under the verification key pk_2 on the message $m^* = H(c_1^* \| k_1^*)$ with $k_1^* = K.\mathsf{Decaps}(sk_1, c_1)$ and $\sigma^* = E.\mathsf{Dec}(k_1^*, c_2)$. We modify the experiment such that the message and the signature used in $\Pi.\mathsf{Encaps}$ are recorded in a list denoted Q_S. We modify $\mathcal{O}_{Chal}$ so that it takes as additional input a message m' and a signature σ' and it verifies that the couple (m', σ') is not in Q_S (and returns $\perp$ if it is). We then modify the experiment so that $\Pi.\mathsf{Decaps}$ takes an additional input composed of m' and σ' and directly runs $S.\mathsf{Vrfy}(pk_2, m', \sigma')$. This modification does not reduce the adversary's success probability, since it only removes intermediate checks. Hence: $\Pr[G_1(\mathcal{A}) = 1] \leq \Pr[G_2(\mathcal{A}) = 1]$.
- **Game 3.** In Game 2, the only way for the adversary to obtain a non-$\perp$ output from $\mathcal{O}_{chal}$ is to produce a signature σ and a message m such that $(m, \sigma) \notin Q_S$ and $\Pi.\mathsf{Decaps}(sk_1, pk_2, c, m, \sigma) \neq \perp$ *i.e.* , $S.\mathsf{Vrfy}(pk_2, m, \sigma) = 1$. In Game 3, we modify the challenge oracle to return $\perp$ on any inputs. An adversary distinguishing Game 2 from Game 3 can be used to produce a pair (m, σ) such that σ is a valid and fresh signature on m under pk_2 and

therefore breaking the EUF-CMA experiment for the signature scheme S. Therefore, $|\Pr[G_2(\mathcal{A})=1] - \Pr[G_3(\mathcal{A})=1]| \leq \mathrm{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A},S}$.

In Game 3, the challenge oracle always outputs $\perp$ thus, the probability that the adversary wins is $\Pr[G_3(\mathcal{A})=1] = 1/2$. Thus, we have:

$$\mathrm{Adv}^{(q_d,q_c)\text{-}\mathsf{Insider\text{-}Auth}}_{\mathcal{A},\mathsf{PAKEM}} \leq \mathrm{Adv}^{\mathsf{EUF-CMA}}_{\mathcal{A},S} + \mathsf{negl}(\lambda). \qquad \square$$

Password Safety. We prove that Construction 1 resists efficient offline exhaustive password-search attacks as a consequence of Theorem 3.

Proposition 1 (Password Safety of Construction 1) *If Construction 1 is* Outsider-Auth *secure, then it is password safe.*

Proof (Sketch). The proof is by reduction to the Outsider-Auth security of PAKEM. Assuming that there exists an adversary $\mathcal{A}$ that wins the password safety experiment with non-negligible probability, we construct an adversary $\mathcal{B}$ that wins in the Outsider-Auth experiment, with non-negligible probability. $\mathcal{B}$ uses $\mathcal{A}$ as a subroutine to obtain a password guess that will be correct with non-negligible probability, and uses that to generate a ciphertext using the encapsulation algorithm. The ciphertext will be valid (for the correct password) with non-negligible probability, and the result follows. $\square$

5 Instantiation of PAKEM

In the following we give an instantiation of PAKEM using NIST PQ-standardized cryptographic primitives, and provide its runtime and bandwidth analysis.
Choice of the Primitives. To maintain proved security of the instantiated PAKEM, following the general construction in Section 4, we will use cryptographic primitives that satisfy the corresponding security requirements according to Theorem2 and Theorem 3. Our instantiation will use the following four algorithms, all satisfying the required security of their corresponding primitive.

- An IND-CCA-secure KEM, instantiated with CRYSTALS-Kyber [17, Section 4, Theorem 3], proven IND-CCA secure in the random-oracle model under the Module-LWE assumption.
- An EUF-CMA-secure signature scheme, instantiated with CRYSTALS-Dilithium [25, Section 4.2], whose security relies on the Module-LWE and Module-SIS assumptions in the random-oracle model.
- An IND-CCA-secure DEM, instantiated with AES-256 in GCM mode [24]. AES-GCM achieves IND-CPA and INT-CTXT security under standard assumptions [34], which together imply IND-CCA security [10]. The nonce (IV) is a 12-byte string, sampled at random, concatenated with the ciphertext [26].
- A hash function modeled as a random oracle, instantiated with SHA3-512 [15].

Parameter Choice. We instantiate Construction 1 following NIST Security Strength Categories [40, Section 4.A.5]. NIST levels 1, 3, and 5 correspond to the classical security of AES-128, AES-192, and AES-256, respectively. For each level, we select a pair of post-quantum primitives providing matching security.
Space Cost. The data transmitted over the channel during an execution corresponds to the size of a PAKEM ciphertext ($\ell_{\mathsf{ct}}^{\mathsf{PAKEM}}$). This ciphertext consists of a Kyber ciphertext ($\ell_{\mathsf{ct}}^{\mathsf{KEM}}$) paired with a Dilithium signature (ℓ_{sig}) encrypted using AES-GCM introducing some padding (ℓ_{pad}), the nonce (12 bytes) and a tag (ℓ_{tag}). The total PAKEM ciphertext size is therefore: $\ell_{\mathsf{ct}}^{\mathsf{PAKEM}} = \ell_{\mathsf{ct}}^{\mathsf{KEM}} + (\ell_{\mathsf{sig}} + \ell_{\mathsf{pad}} + \ell_{\mathsf{tag}} + 12)$. Similarly, server-side storage ($\ell_{\mathsf{store}}^{\mathsf{server}}$) per user consists of the KEM key pair ($\ell_{\mathsf{pk}}^{\mathsf{KEM}}$, $\ell_{\mathsf{sk}}^{\mathsf{KEM}}$) and the signature public key ($\ell_{\mathsf{pk}}^{\mathsf{SIG}}$). The total server-side storage for PAKEM is therefore: $\ell_{\mathsf{store}}^{\mathsf{server}} = \ell_{\mathsf{pk}}^{\mathsf{KEM}} + \ell_{\mathsf{sk}}^{\mathsf{KEM}} + \ell_{\mathsf{pk}}^{\mathsf{SIG}}$. All results are summarized in Tables 1 and 2.
Running-time Evaluation. We evaluate the computational cost of Construction 1 across NIST security levels 1, 3, and 5. We measure the following operations: Setup: key generation and enrollment; Encaps/Sign: encapsulation and signature generation; Decaps/Verify: decapsulation and signature verification. Our implementation is written in C using Liboqs [46] (Kyber and Dilithium) and OpenSSL [41] (SHA3-512 and AES). Experiments were conducted on a Lenovo ThinkPad E16 Gen 3 equipped with an Intel® Core™ Ultra 7 255H processor (16 cores, up to 5.1 GHz) and 32 GiB of DDR5 RAM, running Ubuntu 24.04.3 LTS. All programs were compiled with `-Ofast`, and runtimes were averaged over $10,000$ executions. Table 1 reports the mean runtime for each level, while Table 2 provides a detailed statistical analysis. Across all levels, performance closely matches that of a straightforward KEM-plus-signature composition.

6 Password-Authenticated Hybrid Encryption

As originally introduced by Cramer and Shoup [22], a Hybrid Encryption scheme consists of two components: a KEM and a DEM, enabling the secure encryption of messages of arbitrary length. In this section, we extend this classical paradigm by introducing the notion of PAHE. In PAHE, the KEM component is replaced by a PAKEM to integrate password-based authentication into the hybrid encryption framework. This construction is similar to that in APKE [3], where the KEM was replaced by an AKEM. Here, we replace the AKEM with a PAKEM to achieve PAHE. Thus, our definitions and security notions are natural adaptations of those proposed for APKE in [3].

6.1 Formal Definition

A PAHE is an authenticated public-key encryption scheme for arbitrary-length messages. It allows a sender, possessing a password and the public key of a receiver with whom the password has been registered, to encrypt and authenticate a message. Using the corresponding secret key and enrollment information, the receiver can recover the plaintext and verify its authenticity.

NIST Level	Scheme	Bandwidth (bytes)	Storage (bytes)	Runtime (μs) Setup	Encaps/Sign	Decaps/Verify
Level 1	PAKEM	3,228	3,744	50.84	126.44	70.65
	Kyber-512	780	2,432	12.81	16.21	11.39
	Dilithium2	2,432	1,312	28.92	79.34	29.04
Level 3	PAKEM	4,422	5,536	80.31	193.02	105.20
	Kyber-768	1,100	3,584	17.48	21.85	16.01
	Dilithium3	3,305	1,952	48.81	128.93	47.73
Level 5	PAKEM	6,204	7,328	119.51	244.42	155.30
	Kyber-1024	1,580	4,736	23.99	30.06	22.52
	Dilithium5	4,607	2,592	76.31	156.66	75.29

Table 1: Bandwidth, storage, and runtime (mean over 10,000 iterations) comparison between PAKEM, Kyber and Dilithium.

Level	Runtime (μs) Setup					Encaps					Decaps				
	Min	Median	Mean	Max	SD	Min	Median	Mean	Max	SD	Min	Median	Mean	Max	SD
Level 1	48.33	50.16	50.84	281.149	5.80	82.32	109.90	126.44	482.10	48.29	67.67	69.53	70.65	582.51	10.60
Level 3	78.39	79.66	80.31	322.55	3.87	120.42	170.59	193.02	946.45	78.09	102.00	104.42	105.20	334.60	4.97
Level 5	113.99	117.05	119.51	419.67	13.16	173.52	220.88	244.42	854.82	76.53	148.54	152.38	155.30	642.58	14.27

Table 2: PAKEM statistical runtime analysis in microseconds as a function of the NIST security level on 10,000 iterations.

Definition 6 (PAHE). *A Password-Authenticated Hybrid Encryption (PAHE) scheme is a tuple of four probabilistic algorithms* $\mathsf{PAHE} = (\mathsf{Gen}, \mathsf{Enrol}, \mathsf{Enc}, \mathsf{Dec})$ *defined as follows:*

- ***Generation:*** $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$ *is a randomized algorithm that, on input the security parameter* $\lambda \in \mathbb{N}$ *and outputs a public/secret key pair* (pk, sk).
- ***Enrollment:*** $d \leftarrow \mathsf{Enrol}(pw, pk)$ *is a (randomized) procedure that takes as input a password* pw*, and a public key and outputs the registry information* d *derived from* pw.
- ***Encryption:*** $c \leftarrow \mathsf{AuthEnc}(pk, pw, m)$ *is a probabilistic algorithm that, on input the public key* pk*, a password* pw*, and a message* $m \in \mathcal{M} = \{0,1\}^*$*, outputs a ciphertext* c.
- ***Decryption:*** $m \leftarrow \mathsf{AuthDec}(sk, d, c)$ *is a deterministic algorithm that, on input the secret key* sk*, the registry information* d*, and the ciphertext* c*, returns either the plaintext message* m *or a special symbol* $\perp$ *indicating failure.*

Correctness. A PAHE scheme is said to be correct if, whenever both parties behave honestly, the decryption algorithm outputs the encrypted message with high probability.

Definition 7 (ε-Correctness). *A PAHE is said to be* ε*-correct if, for all security parameters* $\lambda \in \mathbb{N}$, $\Pr\left[\mathsf{Exp}^{\mathsf{COR}}_{\mathsf{PAKEM}}(\lambda) = 1\right] \leq \varepsilon$*, where* ε *is a negligible function in* λ *and the probability is taken over the internal randomness of the experiment* $\mathsf{Exp}^{\mathsf{COR}}_{\mathsf{PAKEM}}(\lambda)$ *defined in Figure 11.*

$\underline{\mathsf{Exp}^{\mathsf{COR}}_{\mathsf{PAKEM}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$; $m \xleftarrow{\$} \mathcal{M}$; $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$; $d \leftarrow \mathsf{Enrol}(pw, pk)$
$c \leftarrow \mathsf{AuthEnc}(pk, pw, m)$; $m' \leftarrow \mathsf{AuthDec}(sk, d, c)$
Return $[m \neq m']$

Fig. 11: Correctness Exp. for PAHE with $\lambda \in \mathbb{N}$ the security parameter.

In both APKE and PAPKE, the schemes are assumed to be perfectly correct. However, as discussed for PAKEM, this requirement is not suitable for post-quantum constructions, which motivates the need for a relaxed definition.

6.2 Security

In this section, we introduce the security properties of PAHE. The modifications from the APKE games to the PAHE games follow the same logic as those between AKEM and PAKEM.

Message Indistinguishability. For PAHE, this notion aims to ensure that ciphertexts sent to an honest, uncompromised receiver do not leak any information about the encrypted message. Following the same reasoning as for AKEM, we define the Outsider-CCA Experiment and the Insider-CCA Experiment. As in the case of AKEM, the Outsider scenario mirrors the classical IND-CCA definition, whereas in the Insider scenario, the adversary is additionally given access to the secret used for encryption.

Definition 8 (Message Indistinguishability). *A PAHE scheme is said to achieve message* mode*-indistinguishability for a security parameter λ if, for the adversarial model* $\mathsf{mode} \in \{\mathsf{Insider}, \mathsf{Outsider}\}$ *and any PPT adversary $\mathcal{A}$ making at most q_e encapsulation queries, q_d decryption queries and q_c challenge queries, the advantage of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{(q_e,q_d,q_c)\text{-}\mathsf{mode}\text{-}\mathsf{CCA}}_{\mathsf{PAHE}}$ (defined in Figure 12 and Figure 13) is negligible in λ. Formally,*

$$\mathrm{Adv}^{(q_e,q_d,q_c)\text{-}\mathsf{mode}\text{-}\mathsf{CCA}}_{\mathcal{A},\mathsf{PAHE}} = \left|\Pr\left[\mathsf{Exp}^{(q_e,q_d,q_c)\text{-}\mathsf{mode}\text{-}\mathsf{CCA}}_{\mathsf{PAHE}}(\lambda) = 1\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda).$$

Ciphertext Authenticity. Akin to APKE [3], we also adapt the authenticity property, which captures the ability of an adversary to produce a valid ciphertext without knowledge of the password, following the same methodology as for PAKEM. In the Outsider-Auth setting, the adversary has no information about either the sender or the receiver, whereas in the Insider-Auth setting, the adversary has compromised the receiver. In both experiments, the goal of the adversary is to convince that the ciphertext sent to the server comes from the user. Hence, we define the Insider and Outsider Experiments.

Definition 9 (Ciphertext Authenticity). *A PAHE scheme is said to achieve* mode*-ciphertext authenticity for a security parameter λ if, for the adversarial*

$\underline{\mathsf{Exp}^{(q_e,q_d)\text{-Outsider-CCA}}_{\mathsf{PAHE}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$; $b \xleftarrow{\$} \{0,1\}$
$(sk, pk) \leftarrow \mathsf{Gen}(\lambda)$; $d \leftarrow \mathsf{Enrol}(pk, pw)$
$(m_1, m_2, st) \leftarrow \mathcal{A}_0^{\mathcal{O}_{\mathrm{Enc}}, \mathcal{O}_{\mathrm{Dec}}}(p_k)$
$c_b \leftarrow \mathsf{AuthEnc}(pk, pw, m_b)$
$b' \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathrm{Enc}}, \mathcal{O}_{\mathrm{Dec}}}(pk, c_b, st)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Enc}}(m)}$:
$c \leftarrow \mathsf{AuthEnc}(pk, pw, m)$
Return c
$\underline{\mathcal{O}_{\mathrm{Dec}}(c')}$:
If $[c' = c_b]$, **Return** $\perp$.
$m' \leftarrow \mathsf{AuthDec}(sk, d, c')$
Return m'

Fig. 12: (q_e, q_d)-Outsider-CCA Experiment for PAHE with security parameter $\lambda \in \mathbb{N}$, $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ a PPT adversary, q_e accesses to the encryption oracle $\mathcal{O}_{\mathrm{Enc}}$ and q_d accesses to the decryption oracle $\mathcal{O}_{\mathrm{Dec}}$.

$\underline{\mathsf{Exp}^{(q_d,q_c)\text{-Insider-CCA}}_{\mathsf{PAHE}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$
$(sk, pk) \leftarrow \mathsf{Gen}(\lambda)$
$d \leftarrow \mathsf{Enrol}(pk, pw)$
$b \xleftarrow{\$} \{0,1\}$
$Q_E \leftarrow \emptyset$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Dec}}, \mathcal{O}_{\mathrm{Chal}}}(pk, pw)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Dec}}(c)}$:
If $[c \in Q_E]$, **Return** $\perp$
$m \leftarrow \mathsf{AuthDec}(sk, d, c)$
Return k
$\underline{\mathcal{O}_{\mathrm{Chal}}(m_0, m_1)}$:
If $[|m_0| \neq |m_1|]$: **Return** $\perp$
$c' \xleftarrow{\$} \mathsf{AuthEnc}(pk, pw, m_b)$
$Q_E \leftarrow Q_E \cup \{c'\}$
Return c'

Fig. 13: (q_d, q_c)-Insider-CCA Experiment for PAHE with security parameter $\lambda \in \mathbb{N}$, $\mathcal{A}$ the PPT adversary, q_d accesses to the decryption oracle $\mathcal{O}_{\mathrm{Dec}}$ and q_c accesses to the challenge oracle $\mathcal{O}_{\mathrm{Chal}}$.

model $\mathsf{mode} \in \{\mathsf{Insider}, \mathsf{Outsider}\}$ *and any PPT adversary* $\mathcal{A}$ *making at most* q_e *encryption queries,* n *password guesses or decapsulation queries, and* q_c *challenge queries, the advantage of* $\mathcal{A}$ *in the experiment* $\mathsf{Exp}^{(q_e,n,q_c)\text{-mode-Auth}}_{\mathsf{PAHE}}$ *(defined in Figure 14 and Figure 15) is negligible in* λ*. Formally, with* $\mathcal{P}$ *denoting the password space:*

$$\mathrm{Adv}^{(q_e,n,q_c)\text{-mode-Auth}}_{\mathcal{A},\mathsf{PAHE}} = \left| \Pr\left[\mathsf{Exp}^{(q_e,n,q_c)\text{-mode-Auth}}_{\mathsf{PAHE}}(\lambda) = 1\right] - \left(\frac{n}{|\mathcal{P}|} + \frac{1}{2}\right) \right| \leq \mathsf{negl}(\lambda).$$

As the adversary in the (q_e, q_c)-Insider-Auth experiment can win the authenticity game by correctly guessing the password, we assume that such an adversary can make at most n guesses per session, yielding a baseline success probability.
Password Safety. Similar to PAKEM, password safety for PAHE requires that the ciphertext leak no information about the password to an outsider adversary, and that any password verification requires online interaction with the server.

Definition 10 (Password Safety). *A PAHE scheme is said to be password safe if, for a security parameter* λ *and for any PPT adversary* $\mathcal{A}$ *making at*

$\underline{\mathsf{Exp}^{(q_e,q_d,q_c)\text{-Outsider-Auth}}_{\mathsf{PAHE}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$
$(sk, pk) \leftarrow \mathsf{Gen}(\lambda)$; $d \leftarrow \mathsf{Enrol}(pk, pw)$
$Q_E \leftarrow \emptyset$; $b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Enc}},\mathcal{O}_{\mathrm{Dec}},\mathcal{O}_{\mathrm{Chal}}}(pk)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Dec}}(c)}$:
$m \leftarrow \mathsf{AuthDec}(sk, d, c)$
Return m

$\underline{\mathcal{O}_{\mathrm{Enc}}(m)}$:
$c \leftarrow \mathsf{AuthEnc}(pk, pw, m)$
$Q_E \leftarrow Q_E \cup \{(m, c)\}$
Return c

$\underline{\mathcal{O}_{\mathrm{Chal}}(c')}$:
If $[\exists m : (c', m) \in Q_E]$, **Return** $\perp$
$m' \leftarrow \mathsf{AuthDec}(sk, d, c')$
If $[m' = \perp]$, **Return** $\perp$
If $[b = 1]$, $m' \xleftarrow{\$} \mathcal{M}$
$Q_E \leftarrow Q_E \cup \{(m', c')\}$
Return m'

Fig. 14: (q_e, q_d, q_c)-Outsider-Auth Experiment for PAHE with security parameter $\lambda \in \mathbb{N}$, $\mathcal{A}$ the PPT adversary, q_e accesses to the enc. oracle $\mathcal{O}_{\mathrm{Enc}}$, q_d accesses to the dec. oracle $\mathcal{O}_{\mathrm{Dec}}$ and q_c accesses to the challenge oracle $\mathcal{O}_{\mathrm{Chal}}$.

$\underline{\mathsf{Exp}^{(q_e,q_d,q_c)\text{-Insider-Auth}}_{\mathsf{PAHE}}(\lambda)}$:
$pw \xleftarrow{\$} \mathcal{P}$
$(sk, pk) \leftarrow \mathsf{Gen}(\lambda)$
$d \leftarrow \mathsf{Enrol}(pk, pw)$
$Q_E \leftarrow \emptyset$; $b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Enc}},\mathcal{O}_{\mathrm{Chal}}}(pk, sk, d)$
Return $[b = b']$

$\underline{\mathcal{O}_{\mathrm{Enc}}(m)}$:
$c \leftarrow \mathsf{AuthEnc}(pk, pw, m)$
$Q_E \leftarrow Q_E \cup \{(m, c)\}$
Return c

$\underline{\mathcal{O}_{\mathrm{Chal}}(c')}$:
If $[\exists m : (c', m) \in Q_E]$:
  Return $\perp$
$m' \leftarrow \mathsf{AuthDec}(sk, d, c')$
If $[m' = \perp]$, **Return** $\perp$
If $[b = 1]$, $m' \xleftarrow{\$} \mathcal{M}$
$Q_E \leftarrow Q_E \cup \{(m', c')\}$
Return m'

Fig. 15: (q_e, q_d, q_c)-Outsider-Auth Experiment for PAHE with security parameter $\lambda \in \mathbb{N}$, $\mathcal{A}$ the PPT adversary, q_e accesses to the enc. oracle $\mathcal{O}_{\mathrm{Enc}}$ and q_c accesses to the challenge oracle $\mathcal{O}_{\mathrm{Chal}}$.

most q_e encryption queries, and q_d decryption queries, the advantage of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{(q_e,q_c)\text{-PW-SAFE}}_{\mathsf{PAHE}}$ (defined in Figure 16) is negligible in λ. Formally, with $\mathcal{P}$ denoting the password space:

$$\mathrm{Adv}^{(q_e,q_d)\text{-PW-SAFE}}_{\mathcal{A},\mathsf{PAHE}} = \left| \Pr\left[\mathsf{Exp}^{(q_e,q_c)\text{-PW-SAFE}}_{\mathsf{PAHE}}(\lambda) = 1\right] - \frac{q_d}{|\mathcal{P}|} \right| \leq \mathsf{negl}(\lambda).$$

6.3 From PAKEM and DEM to PAHE

In this section, we show how a PAHE scheme can be constructed from a PAKEM and a DEM, following the classical hybrid encryption paradigm. This approach mirrors the composition theorems introduced by Herranz *et al.* [31]. We first introduce the construction of PAHE from its PAKEM and DEM components, then prove its correctness, and finally analyze its security properties.

$\underline{\mathsf{Exp}^{(q_e,q_c)\text{-PW-SAFE}}_{\mathsf{PAHE}}(\lambda)}$:	$\underline{\mathcal{O}_{\mathrm{Enc}}(m)}$:
$pw \xleftarrow{\$} \mathcal{P}$	$c \leftarrow \mathsf{AuthEnc}(pk, pw, m)$
$(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$	**Return** c
$d \leftarrow \mathsf{Enrol}(pw, pk)$	
$pw' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{Enc}}, \mathcal{O}_{\mathrm{Dec}}}(pk)$	$\underline{\mathcal{O}_{\mathrm{Dec}}(c)}$:
Return $[pw = pw']$	$m \leftarrow \mathsf{AuthDec}(sk, d, c)$
	Return m

Fig. 16: (q_e, q_d)-PW-SAFE Experiment for PAHE.

Construction 2 (PAHE from PAKEM and DEM) *Let Π be a Password-Authenticated Key Encapsulation Mechanism (PAKEM) and Σ be a DEM. Then, a PAHE scheme can be constructed as shown in Figure 17.*

$\underline{\mathsf{Gen}(\lambda)}$:	$\underline{\mathsf{Enrol}(pk, pw)}$:
$\mathsf{pk}, \mathsf{sk} \leftarrow \Pi.\mathsf{Gen}(\lambda)$	$\mathsf{d} \leftarrow \Pi.\mathsf{Enrol}(\lambda)$
Return pk, sk	**Return** d
$\underline{\mathsf{AuthEnc}(\mathsf{pk}, \mathsf{pw}, m)}$:	$\underline{\mathsf{AuthDec}(\mathsf{sk}, \mathsf{d}, c)}$:
$(c_1, k) \leftarrow \Pi.\mathsf{Encaps}(\mathsf{pk}, \mathsf{pw})$	Parse c as (c_1, c_2)
$c_2 \leftarrow \Sigma.\mathsf{Enc}(k, m)$	$k \leftarrow \Pi.\mathsf{Decaps}(\mathsf{sk}, \mathsf{d}, c_1)$; $m \leftarrow \Sigma.\mathsf{Dec}(k, c_2)$
Return $c = (c_1, c_2)$	**Return** m

Fig. 17: PAHE from PAKEM and DEM.

Theorem 4 (Correctness of Construction 2). *Let Π be a PAKEM that is ε_1-correct, and let Σ be a DEM that is ε_2-correct. Then, the PAHE scheme obtained by Construction 2 is $(\varepsilon_1 + \varepsilon_2)$-correct.*

Proof. Consider the correctness experiment for the composed PAHE scheme. Let $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, $pw \xleftarrow{\$} \mathcal{P}$, $\mathsf{d} \xleftarrow{\$} \mathsf{Enrol}(pk, pw)$ and let $m \leftarrow \mathcal{M}$ be a message sampled from the message space. The encryption algorithm computes $(c_1, k) \leftarrow \Pi.\mathsf{Encaps}(pk, pw)$ and $c_2 \leftarrow \Sigma.\mathsf{Enc}(k, m)$, returning $c = (c_1, c_2)$. Decryption proceeds by computing $k' \leftarrow \Pi.\mathsf{Decaps}(sk, d, c_1)$ and $m' \leftarrow \Sigma.\mathsf{Dec}(k', c_2)$. By the correctness of Π, we have $\Pr[k \neq k'] \leq \varepsilon_1$. By the correctness of Σ, we have $\Pr[m \neq \Sigma.\mathsf{Dec}(k, \Sigma.\mathsf{Enc}(k, m))] \leq \varepsilon_2$. A correctness failure for the composed PAHE occurs if either $k \neq k'$, or $k = k'$ but $\Sigma.\mathsf{Dec}(k, c_2) \neq m$. Thus, the total failure probability is at most $\varepsilon_1 + \varepsilon_2$ and the result follow. □

Message Indistinguishability. We show that if PAKEM is secure against the Insider-CCA and the DEM is IND-CCA then, the PAHE given by Construction 2 is secure against the Insider-CCA.[8]

Theorem 5. *Let Π be a PAKEM scheme that is secure against the (q_d, q_c)-Insider-CCA and Σ be a DEM that is IND-CCA secure. Then, the hybrid encryption scheme describe in Construction 2 yields a PAHE scheme that is secure against (q_d, q_c)-Insider-CCA with $q_d^{\mathsf{PAKEM}} \geq q_d^{\mathsf{PAHE}}$ and $q_c^{\mathsf{PAKEM}} \geq q_c^{\mathsf{PAHE}}$.*

Observe that the session key k used to encrypt the message is produced via a PAKEM encapsulation and remains computationally indistinguishable from random to any adversary. Thus, the first ciphertext component c_1 leaks no information about k, and by the DEM security, the second component c_2 is also indistinguishable from random. Therefore, the ciphertext (c_1, c_2) does not reveal information about the message, even under the Insider-CCA attacks.

Proof. Let $\mathcal{A}$ be a (q_d, q_c)-Insider-CCA adversary against the PAHE scheme of Construction 2. We construct adversaries $\mathcal{B}$ and $\mathcal{C}$ against the underlying PAKEM Π and the DEM Σ, respectively.

- **Game** 0: This is the experiment $\mathsf{Exp}_{\mathsf{PAHE}}^{(q_d,q_c)\text{-Insider-CCA}}$.
- **Game** 1**:** We modify the challenge ciphertext generation. Instead of computing $(c_1, k) \leftarrow \Pi.\mathsf{Encaps}(pk, pw)$, we replace the key by a randomly drawn key. Any distinguisher between Game 0 and Game 1 yields an adversary $\mathcal{B}$ breaking the (q_d, q_c)-Insider-CCA security of Π (with the same number of q_c). Hence: $\left|\Pr[G_0(\mathcal{A}) = 1] - \Pr[G_1(\mathcal{A}) = 1]\right| \leq \mathrm{Adv}_{\mathcal{B},\Pi}^{(q_d,q_c)\text{-Insider-CCA}}$.
- **Game** 2**:** In this game, all challenge ciphertexts are of the form $c^* = (c_1, c_2)$ where $k \xleftarrow{\$} \mathcal{K}$ is uniform and $c_2 \leftarrow \Sigma.\mathsf{Enc}(k, m_b)$, with k independent of everything else. Since c_1 is independent of both k and m_b, it carries no information about the challenge bit. Thus, distinguishing encryptions of m_0 and m_1 reduces to distinguishing encryptions under Σ with a random key. We now bound the difference between Game 1 and Game 2 via a hybrid argument over the q_c challenge queries. Each challenge query randomly draw a key, yielding independent instances of the DEM-IND-CCA experiment. Hence, we have: $\left|\Pr[G_1(\mathcal{A}) = 1] - \Pr[G_2(\mathcal{A}) = 1]\right| \leq q_c \cdot \mathrm{Adv}_{\mathcal{C},\Sigma}^{\mathsf{IND\text{-}CCA}}$.

Note that in Game 2, all challenge encryptions are independent of b, and therefore: $\Pr[G_2(\mathcal{A}) = 1] = 1/2$. Combining the above, we obtain:

$$\mathrm{Adv}_{\mathcal{A},\mathsf{PAHE}}^{(q_d,q_c)\text{-Insider-CCA}} \leq \mathrm{Adv}_{\mathcal{B},\Pi}^{(q_d,q_c)\text{-Insider-CCA}}(\lambda) + q_c \cdot \mathrm{Adv}_{\mathcal{C},\Sigma}^{\mathsf{IND\text{-}CCA}}. \qquad \square$$

Ciphertext Authenticity. We show that if the PAKEM is secure against Outsider-Auth and the DEM is INT-CTXT secure, then the PAHE scheme from Construction 2 is secure against Outsider-Auth.

[8] Using the same proof method, one can show that if PAKEM is secure against the Outsider-CCA then the resulting PAHE is secure against the Outsider-CCA.

Remark that achieving Insider-Auth security with a direct composition of a PAKEM and a DEM is impossible. Indeed, in the Insider setting the adversary is granted access to both the secret key sk of the PAKEM and the auxiliary secret information d. As a consequence, after obtaining a valid ciphertext (c_1, c_2) from the encryption oracle, the adversary can decapsulate c_1 using sk to recover the symmetric key k, and then use the decryption algorithm $\Sigma.\mathsf{Dec}$ together with d to derive valid encryption material for an arbitrary message of its choice. This enables the adversary to compute a modified ciphertext c_2' under the recovered key k, such that (c_1, c_2') is valid and will be accepted by the verification procedure. Therefore, querying the challenge oracle on (c_1, c_2') yields a valid forgery, allowing the adversary to win with probability 1.

Theorem 6. *Let Π be a PAKEM scheme that is secure against the (q_e, q_d, q_c)-Outsider-Auth and Σ be a DEM that is secure against INT-CTXT. Then, the hybrid encryption scheme describe in Construction 2 yields a PAHE scheme that is secure against (q_e, q_d, q_c)-Insider-Auth with $q_e^{\mathsf{PAKEM}} \geq q_e^{\mathsf{PAHE}}$, $q_d^{\mathsf{PAKEM}} \geq q_d^{\mathsf{PAHE}}$ and $q_c^{\mathsf{PAKEM}} \geq q_c^{\mathsf{PAHE}}$.*

Notice that in the PAHE construction, the adversary must produce a forged ciphertext $c = (c_1, c_2)$ that decrypts to a valid message. To succeed, the adversary must ensure that c_1 yields a key k upon decapsulation, and that c_2 is a valid ciphertext under k. However, by the PAKEM authenticity, the adversary cannot generate a c_1 that causes the receiver to derive a key k of its choice, unless it was obtained via a legitimate encapsulation. As a result, the adversary cannot produce a matching c_2 without knowing k, due to the INT-CTXT guaranteed by the DEM. Thus, any forgery attempt will fail with overwhelming probability, ensuring ciphertext authenticity.

Proof. Let $\mathcal{A}$ be a (q_e, q_d, q_c)-Outsider-Auth adversary against the PAHE scheme. We construct adversaries $\mathcal{B}$ against Π and $\mathcal{C}$ against Σ. We first observe that if $\mathcal{A}$ is not able to forge a valid ciphertext that is accepted by $\mathsf{AuthDec}$, then its view is independent of the challenge bit b as every challenge query will outputs $\perp$. In that case, $\mathcal{A}$ can do no better than random guessing, and its success probability is exactly 1/2. Therefore, any non-negligible advantage of $\mathcal{A}$ in the Outsider-Auth experiment must come from producing at least one fresh ciphertext (c_1, c_2) such that $\mathsf{AuthDec}(sk, (c_1, c_2)) \neq \perp$, *i.e.* , from a successful forgery. By definition of $\mathsf{AuthDec}$, this implies that: $k \leftarrow \Pi.\mathsf{Decaps}(sk, c_1) \neq \perp$ and $\Sigma.\mathsf{Dec}(k, c_2) \neq \perp$. We distinguish two cases depending on the origin of the ciphertext components:

- **Case 1:** c_1 is fresh. This means that c_1 was never output by the encryption oracle $\mathcal{O}_{\mathrm{Enc}}$. If (c_1, c_2) is accepted by $\mathsf{AuthDec}$, then we have $k \leftarrow \Pi.\mathsf{Decaps}(sk, c_1) \neq \perp$. Hence, $\mathcal{A}$ has produced a fresh ciphertext c_1 that successfully decapsulates, which directly yields a forgery against the (q_e, q_c)-Insider-Auth security of the PAKEM scheme Π. Therefore, there exists an adversary $\mathcal{B}$ such that: $\Pr[\text{Case 1 occurs}] \leq \mathrm{Adv}^{(q_e,q_d,q_c)\text{-}\mathsf{Outsider\text{-}Auth}}_{\mathcal{B},\Pi}$.
- **Case 2:** c_1 is not fresh but c_2 is fresh. In this case, c_1 was previously returned by the encryption oracle, and thus corresponds to some encapsulated

key k. Since (c_1, c_2) is accepted by $\mathsf{AuthDec}$, we must have: $\Sigma.\mathsf{Dec}(k, c_2) \neq \perp$. Moreover, since c_2 is fresh, it was never output by the encryption oracle under the key k. Hence, $\mathcal{A}$ has produced a valid ciphertext under k that was not previously generated, which constitutes a forgery against the INT-CTXT security of Σ. Therefore, there exists an adversary $\mathcal{C}$ such that: $\Pr[\text{Case 2 occurs}] \leq \mathrm{Adv}^{\mathsf{INT\text{-}CTXT}}_{\mathcal{C},\Sigma}$.

In both cases, a successful forgery by $\mathcal{A}$ implies either a break of the PAKEM authenticity or a break of the DEM ciphertext integrity. Therefore, we obtain:

$$\mathrm{Adv}^{(q_e,q_d,q_c)\text{-}\mathsf{Outsider\text{-}Auth}}_{\mathcal{A},\mathsf{PAHE}} \leq \mathrm{Adv}^{(q_e,q_c)\text{-}\mathsf{Insider\text{-}Auth}}_{\mathcal{B},\Pi} + \mathrm{Adv}^{\mathsf{INT\text{-}CTXT}}_{\mathcal{C},\Sigma}. \qquad \square$$

Using Theorem 6, we can show that PAHE inherits password safety from its Outsider-Auth property.

Proposition 2 (Password Safety of Construction 2) *If Construction 2 satisfies* Outsider-Auth *security, then it is password safe.*

Proof (Sketch). The proof follows the same reduction as for the password safety of PAKEM. If PAHE were not password safe, an adversary could use the guessed password to produce a valid ciphertext, contradicting the Outsider-Auth security of PAHE. $\square$

7 Conclusion

We introduced Password-Authenticated Key Encapsulation Mechanisms (PAKEMs), which enable a user to establish an authenticated shared key with a server, and Password-Authenticated Hybrid Encryption (PAHE), which uses PAKEM to construct authenticated encryption for arbitrary-length messages. We formalized their security requirements and proposed generic constructions, proving that the PAKEM achieves key indistinguishability and authenticity against insider adversaries, while the resulting PAHE provides message indistinguishability against insider adversaries and ciphertext authenticity against outsider adversaries. Our security model and proofs are for two-user security. Extending these results to multi-user security, inline with [3, Theorem 11, 12 and 13 in Appendix] is an interesting open question.

Acknowledgement.

This research was supported in part by Quantum City, affiliated with the University of Calgary, and by an NSERC Alliance Quantum Grant.

References

1. Abe, M., Gennaro, R., Kurosawa, K., Shoup, V.: Tag-kem/dem: A new framework for hybrid encryption and a new analysis of kurosawa-desmedt kem. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 128–146. Springer (2005)

2. Agrawal, S., Miao, P., Mohassel, P., Mukherjee, P.: Pasta: password-based threshold authentication. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 2042–2059 (2018)
3. Alwen, J., Blanchet, B., Hauck, E., Kiltz, E., Lipp, B., Riepel, D.: Analysing the hpke standard. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 87–116. Springer (2021)
4. Alwen, J., Janneck, J., Kiltz, E., Lipp, B.: The pre-shared key modes of hpke. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 329–360. Springer (2023)
5. Avanzi, R., Bai, S., Bos, J., Ding, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: Crystals cryptographic suite for algebraic lattices (2020), `https://pq-crystals.org/kyber/`
6. Avanzi, R., Bai, S., Bos, J., Ding, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: Crystals cryptographic suite for algebraic lattices (2020), `https://pq-crystals.org/dilithium/`
7. Barbosa, M., Chen, L., Cheng, Z., Chimley, M., Dent, A., Farshim, P., Harrison, K., Malone-Lee, J., Smart, N., Vercauteren, F.: Sk- kem: An identity- based kem. Sumission to IEEE P **1363**, 3 (2006)
8. Barnes, R., Bhargavan, K., Lipp, B., Wood, C.: Rfc 9180: Hybrid public key encryption (2022)
9. Barnes, R., Bhargavan, K., Lipp, B., Wood, C.A.: Hybrid public key encryption. Internet Research Task Force (IRTF), RFC **9180** (2022)
10. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 531–545. Springer (2000)
11. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. Journal of cryptology **21**(4), 469–491 (2008)
12. Bellare, M., Rogaway, P.: The exact security of digital signatures-how to sign with rsa and rabin. In: International conference on the theory and applications of cryptographic techniques. pp. 399–416. Springer (1996)
13. Bellovin, S.M., Merritt, M.: Encrypted key exchange: Password-based protocols secure against dictionary attacks (1992)
14. Bentahar, K., Farshim, P., Malone-Lee, J., Smart, N.P.: Generic constructions of identity-based and certificateless kems. Journal of Cryptology **21**, 178–199 (2008)
15. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Keccak. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 313–314. Springer (2013)
16. Bos, J., Costello, C., Ducas, L., Mironov, I., Naehrig, M., Nikolaenko, V., Raghunathan, A., Stebila, D.: Frodo: Take off the ring! practical, quantum-secure key exchange from lwe. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. pp. 1006–1018 (2016)
17. Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-kyber: a cca-secure module-lattice-based kem. In: 2018 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 353–367. IEEE (2018)
18. Bradley, T., Camenisch, J., Jarecki, S., Lehmann, A., Neven, G., Xu, J.: Password-authenticated public-key encryption. In: Applied Cryptography and Network Security: 17th International Conference, ACNS 2019, Bogota, Colombia, June 5–7, 2019, Proceedings 17. pp. 442–462. Springer (2019)

19. Callas, J., Donnerhacke, L., Finney, H., Thayer, R.: Rfc2440: Openpgp message format (1998)
20. Chen, Y., Zhang, Z., Lin, D., Cao, Z.: Cca-secure ib-kem from identity-based extractable hash proof system. The Computer Journal **57**(10), 1537–1556 (2014)
21. Chow, S.S., Liu, J.K., Zhou, J.: Identity-based online/offline key encapsulation and encryption. In: Proceedings of the 6th ACM symposium on information, computer and communications security. pp. 52–60 (2011)
22. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. Cryptology ePrint Archive, Paper 2001/108 (2001), https://eprint.iacr.org/2001/108
23. Cremers, C., Dax, A., Medinger, N.: Keeping up with the kems: Stronger security notions for kems and automated analysis of kem-based protocols. In: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security. pp. 1046–1060 (2024)
24. Daemen, J., Rijmen, V.: Rijndael/aes. In: Encyclopedia of Cryptography and Security, pp. 520–524. Springer (2005)
25. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-dilithium: A lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 238–268 (2018)
26. Dworkin, M.J.: Sp 800-38d. recommendation for block cipher modes of operation: Galois/counter mode (gcm) and gmac (2007), https://csrc.nist.gov/pubs/sp/800/38/d/final
27. Gorantla, M.C., Boyd, C., González Nieto, J.M.: Attribute-based authenticated key exchange. In: Australasian Conference on Information Security and Privacy. pp. 300–317. Springer (2010)
28. Gu, Y., Jarecki, S., Krawczyk, H.: Khape: asymmetric pake from key-hiding key exchange. In: Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part IV 41. pp. 701–730. Springer (2021)
29. Hao, F., Ryan, P.: J-pake: authenticated key exchange without pki. In: Transactions on Computational Science XI: Special Issue on Security in Computing, Part II, pp. 192–206. Springer (2010)
30. Herranz, J., Hofheinz, D., Kiltz, E.: Some (in) sufficient conditions for secure hybrid encryption. Cryptology ePrint Archive (2006)
31. Herranz, J., Hofheinz, D., Kiltz, E.: Some (in) sufficient conditions for secure hybrid encryption. Information and Computation **208**(11), 1243–1257 (2010)
32. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the fujisaki-okamoto transformation. In: Theory of Cryptography Conference. pp. 341–371. Springer (2017)
33. Hofheinz, D., Kiltz, E.: Secure hybrid encryption from weakened key encapsulation. In: Annual International Cryptology Conference. pp. 553–571. Springer (2007)
34. Iwata, T., Ohashi, K., Minematsu, K.: Breaking and repairing gcm security proofs. Lecture Notes in Computer Science **7417** (08 2012). https://doi.org/10.1007/978-3-642-32009-5_3
35. Jarecki, S., Krawczyk, H., Xu, J.: Opaque: an asymmetric pake protocol secure against pre-computation attacks. In: Advances in Cryptology–EUROCRYPT 2018: 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29-May 3, 2018 Proceedings, Part III 37. pp. 456–486. Springer (2018)
36. Katz, J., Lindell, Y.: Introduction to modern cryptography: principles and protocols. Chapman and hall/CRC (2007)

37. Katz, J., Vaikuntanathan, V.: Round-optimal password-based authenticated key exchange. Journal of Cryptology **26**(4), 714–743 (2013)
38. Kinnear, E., McManus, P., Pauly, T., Verma, T., Wood, C.A.: Oblivious dns over https. RFC 9230 (2022)
39. Kurosawa, K., Desmedt, Y.: A new paradigm of hybrid encryption scheme. In: Advances in Cryptology – CRYPTO 2004. Springer (2004)
40. NIST: Submission requirements and evaluation criteria for the post-quantum cryptography standardization process (2016), `https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf`
41. OpenSSL Software Services, Inc., OpenSSL Software Foundation, Inc.: OpenSSL: The open source toolkit for SSL/TLS (2026), `www.openssl.org`
42. Ramsdell, B., Turner, S.: Secure/multipurpose internet mail extensions (s/mime) version 3.2 message specification. Tech. rep. (2010)
43. Rescorla, E.: The transport layer security (tls) protocol version 1.3. Tech. rep. (2018)
44. Sakai, R., Kasahara, M.: Id based cryptosystems with pairing on elliptic curve. Cryptology ePrint Archive (2003)
45. Shin, S., Kobara, K., Imai, H.: Security proof of augpake. Cryptology ePrint Archive (2010)
46. Stebila, D., Mosca, M.: Post-quantum key exchange for the internet and the open quantum safe project. Cryptology ePrint Archive, Paper 2016/1017 (2016), `https://eprint.iacr.org/2016/1017`
47. Wu, T.D., et al.: The secure remote password protocol. In: NDSS. vol. 98, pp. 97–111. Citeseer (1998)