PAPER Special Section on Information and Communication System Security

Tag-KEM/DEM Framework for Public-Key Encryption with Non-Interactive Opening*

Yusuke SAKAI[†a)], **Takahiro MATSUDA**[†], ***Nonmembers***, ***and*** **Goichiro HANAOKA**[†], ***Member***

SUMMARY In a large-scale information-sharing platform, such as a cloud storage, it is often required to not only securely protect sensitive information but also recover it in a reliable manner. Public-key encryption with non-interactive opening (PKENO) is considered as a suitable cryptographic tool for this requirement. This primitive is an extension of public-key encryption which enables a receiver to provide a non-interactive proof which confirms that a given ciphertext is decrypted to some public plaintext. In this paper, we present a Tag-KEM/DEM framework for PKENO. In particular, we define a new cryptographic primitive called a Tag-KEM with non-interactive opening (Tag-KEMNO), and prove the KEM/DEM composition theorem for this primitives, which ensures a key encapsulation mechanism (KEM) and a data encapsulation mechanism (DEM) can be, under certain conditions, combined to form a secure PKENO scheme. This theorem provides a secure way of combining a Tag-KEMNO scheme with a DEM scheme to construct a secure PKENO scheme. Using this framework, we explain the essence of existing constructions of PKENO. Furthermore, we present four constructions of Tag-KEMNO, which yields four PKENO constructions. These PKENO constructions coincide with the existing constructions, thereby we explain the essence of these existing constructions. In addition, our Tag-KEMNO framework enables us to expand the plaintext space of a PKENO scheme. Some of the previous PKENO schemes are only able to encrypt a plaintext of restricted length, and there has been no known way to expand this restricted plaintext space to the space of arbitrary-length plaintexts. Using our framework, we can obtain a PKENO scheme with the unbounded-length plaintext space by modifying and adapting such a PKENO scheme with a bounded-length plaintext space.

key words: *Public-key encryption with non-interactive opening, Tag-KEM*

1. Introduction

Recent emergence of large-scale information-sharing platforms, such as cloud storage, demands for balancing needs of data privacy and utility. Especially, it is often required to not only securely protect sensitive information but also recover it in a reliable manner. Let us consider a situation that several users upload data to a cloud storage server in an encrypted form to share the data among these users. To retrieve data, a user requests an access to that data, possibly uploaded by another user, to the cloud server. When the request is approved, the cloud decrypts the data and sends back the plaintext. However, in this case, we have to assume that cloud server is highly trusted so that it will not replace the correct decryption result with wrong one. Unfortunately, in conventional encryption schemes, it is not easy to determine whether the "decryption result" which is returned from the cloud server is really a correct one.

Public-key encryption with non-interactive opening (PKENO) [2], [3] is an extension of public-key encryption, in which the receiver can produce a non-interactive *proof* for any given ciphertext. This proof can be verified publicly to confirm that the given ciphertext is decrypted to some public plaintext, without compromising confidentiality of any other ciphertexts. By using PKENO, for a given pair of a ciphertext and a plaintext, anybody can determine whether the plaintext is the decryption result of the ciphertext or not. Hence, PKENO is considered as a suitable cryptographic tool for solving the above problem.

PKENO was originally introduced by Damgård and Thorbek [2], [3], for designing secure multiparty computation, in particular for claiming that a sender of an encrypted message deviates from the protocol by sending fake information in an encrypted form. Secure multiparty computation is often used as the central tool for yielding data privacy and utility over a cloud environment, and therefore, PKENO is also important as one of the essential building blocks for it.

There is a subtlety in designing a PKENO scheme, which is evidenced by the fact that one of the constructions by Damgård et al. was broken and repaired by Galindo [4]. The subtlety partly stems from the (implicit) use of a KEM/DEM construction, and partly from the fact that the proof includes the session key encapsulated in the KEM part. The attack is possible because the adversary can replace the DEM part of the target ciphertext with an arbitrary DEM ciphertext, and the adversary can ask the proof *even for this modified ciphertext.* This proof includes the session key of the KEM part, which is the same as that of the target ciphertext, and hence it compromises the confidentiality of the target ciphertext.

This subtlety was overcome by the repair due to Galindo [4] and the following schemes due to Lai et al. [5] and Galindo et al. [6]. All these schemes overcome the subtlety by somehow *binding the DEM part to the KEM part in a publicly verifiable way*, and making any modification break this binding. For such a modified ciphertext, the receiver no longer needs to produce a proof, but only needs to claim that the binding is broken, which is publicly verifiable.

Manuscript received October 28, 2017.
Manuscript revised May 11, 2018.
Manuscript publicized August 22, 2018.

[†]The authors are with , National Institute of Advanced Industrial Science and Technology, Tokyo, 135–0064 Japan.

*A preliminary version of this paper appeared at The International Symposium on Information Theory and Its Applications (ISITA 2016) [1]. This full version includes three additional instantiations and a performance comparison among these instantiations and existing schemes.

a) E-mail: yusuke.sakai@aist.go.jp

DOI: 10.1587/transinf.2017ICP0003

Although there are secure PKENO constructions, their design principle is relatively ad hoc, and there is no unified paradigm for designing a secure PKENO scheme. In addition, as Galindo's attack suggests, to encrypt a session key using some KEM (with the non-interactive opening feature) and to encrypt the plaintext using the session key, are insufficient for constructing a secure PKENO scheme. The existing secure PKENO schemes somehow circumvent this subtlety, but it is not clearly understood how it is accomplished.

1.1 Our Contribution

For a better understanding of the existing secure PKENO schemes, we formalize the above idea behind them by adopting the Tag-KEM/DEM framework by Abe et al. [7]. More specifically, we formalize the notion of Tag-KEM with Non-interactive Opening (Tag-KEMNO) and prove a KEM/DEM composition theorem. Furthermore, we can obtain several existing schemes [4]–[6] as instantiations of our frameworks, which explains how the existing schemes circumvent the aforementioned subtlety.

Tag-KEMNO (and Tag-KEM) have an ability to bind some message, called a tag, to a ciphertext. Furthermore, the security definition of Tag-KEMNO ensures that a ciphertext for some tag can be securely opened under *a different tag*, without compromising the session key encapsulated under the former tag. In our construction, this mechanism is used to bind the DEM part to the KEM part. More specifically, the DEM ciphertext is used as the tag. This way, even if the DEM part is replaced with another ciphertext, the KEM part can be securely opened under the new tag (i.e., the new DEM ciphertext).

As a by-product, our Tag-KEMNO/DEM framework enables us to expand the plaintext space of PKENO schemes in a generic way. Note that existing constructions do not support such arbitrary-length plaintexts, with exceptions of Galindo's scheme [4] and Dachman-Soled et al.'s scheme [8]. Any other schemes are only able to encrypt a single group element. A straightforward adoption of the KEM/DEM construction to encrypt long messages is potentially dangerous as illustrated by Galindo's attack. In contrast, our Tag-KEMNO/DEM framework enables us to securely expand the plaintext space, without suffering from attacks like Galindo's.

One exception not falling into our framework is the generic construction by Dachman-Soled et al. [8]. Their scheme can be instantiated in two ways. Unfortunately, these two schemes do not directly fall into our framework. To extend our framework to be able to explain these schemes is an interesting future direction.

1.2 Paper Organization

In Sect. 2 we introduce several cryptographic primitives that will be used throughout the paper. In Sect. 3 we introduce our new primitive named Tag-KEMNO, which is an extension of the Tag-KEM primitive with an additional ability to prove the correctness of decapsulation. In Sect. 4 we prove our main theorem that shows that when we combine a Tag-KEMNO scheme with some symmetric-key primitive, we can obtain a secure PKENO scheme. In Sect. 5 we present several instantiations of our Tag-KEMNO primitive, which are obtained by extracting the KEM part of known PKENO schemes. In Sect. 6 we discuss some implications of our main theorem, the first of which is an explanation of some known schemes through our Tag-KEMNO primitive, while the second is a generic way to expand the plaintext space of a PKENO scheme. Finally, Sect. 7 concludes the paper.

2. Preliminary

Here we give the definitions of cryptographic primitives used in this paper.

2.1 Public-Key Encryption with Non-Interactive Opening

A PKENO scheme consists of the following five algorithms [3].

- $\mathsf{PKg}(1^\lambda) \to (ek, dk)$. The key generation algorithm takes as an input a security parameter 1^λ and outputs a pair (ek, dk) of an encryption key and a decryption key.
- $\mathsf{PEnc}(ek, M) \to C$. The encryption algorithm takes as inputs an encryption key ek and a plaintext M, and outputs a ciphertext C.
- $\mathsf{PDec}(dk, C) \to m/\perp$. The decryption algorithm takes as inputs a decryption key dk and a ciphertext C, and outputs a plaintext m or the rejection symbol $\perp$.
- $\mathsf{PProve}(dk, C)$. The proof algorithm takes as inputs a decryption key dk and a ciphertext C, and outputs a proof π.
- $\mathsf{PVerify}(ek, C, M, \pi)$. The verification algorithm takes as inputs an encryption key ek, a ciphertext C, a plaintext M, and a proof π, and outputs a bit 1 or 0 indicating the validity of the proof.

For correctness, we require the following: For any $\lambda \in \mathbb{N}$ and any key pair $(ek, dk) \leftarrow \mathsf{PKg}(1^\lambda)$,

1. for any plaintext M, it holds that $\mathsf{PDec}(dk, \mathsf{PEnc}(ek, M)) = M$,
2. for any (either valid and invalid) ciphertext C, it holds that $\mathsf{PVerify}(ek, C, \mathsf{PDec}(dk, C), \mathsf{PProve}(dk, C)) = 1$.

We then give the definition of the confidentiality requirement [3].

Definition 1. A PKENO scheme is indistinguishable against chosen ciphertext and proof attacks if for any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage $|\Pr[\mathbf{Exp}^{\text{PKENO-CCPA}}(\lambda) = 1] - 1/2|$ is negligible in λ, where the experiment $\mathbf{Exp}^{\text{PKENO-CCPA}}(\lambda)$ is defined in Fig. 1, and in the `guess` phase, the adversary is not allowed to submit C^* to either PDec or PProve.

$\mathbf{Exp}^{\text{PKENO-CCPA}}(\lambda)$:
$b \leftarrow \{0,1\}$
$(ek, dk) \leftarrow \mathsf{PKg}(1^\lambda)$
$(M_0, M_1, state) \leftarrow \mathcal{A}^{\mathsf{PDec}(dk,\cdot),\mathsf{PProve}(dk,\cdot)}(\mathtt{find}, ek)$
$C^* \leftarrow \mathsf{PEnc}(ek, M_b)$
$b' \leftarrow A^{\mathsf{PDec}(dk,\cdot),\mathsf{PProve}(dk,\cdot)}(\mathtt{guess}, state, C^*)$
if $b = b'$ return 1 else return 0

$\mathbf{Exp}^{\text{PKENO-commit}}(\lambda)$:
$(ek, dk) \leftarrow \mathsf{PKg}(1^\lambda)$
$(C, M, \pi, M', \pi') \leftarrow \mathcal{A}(ek, dk)$
return 1 if the following holds:
$\mathsf{PVerify}(ek, C, M, \pi) = 1$,
$\mathsf{PVerify}(ek, C, M', \pi') = 1$,
$M \neq M'$
otherwise return 0

$\mathbf{Exp}^{\text{DEM}}(\lambda)$:
$d \leftarrow \{0,1\}$
$K \leftarrow \mathcal{K}$
$(M_0, M_1, state) \leftarrow \mathcal{A}(\mathtt{find}, 1^\lambda)$
$\chi^* \leftarrow \mathsf{DEnc}(K, M_d)$
$d' \leftarrow \mathcal{A}(\mathtt{guess}, state, \chi^*)$
if $d = d'$ return 1 else return 0

$\mathbf{Exp}^{\text{Tag-KEMNO-CCPA}}(\lambda)$:
$d \leftarrow \{0,1\}$
$(ek, dk) \leftarrow \mathsf{KKg}(1^\lambda)$
$(\omega, K_0) \leftarrow \mathsf{KKey}(ek)$
$K_1 \leftarrow \mathcal{K}$
$(t^*, state) \leftarrow \mathcal{A}^{\mathsf{KDecap}(dk,\cdot,\cdot),\mathsf{KProve}(dk,\cdot,\cdot)}(\mathtt{find}, ek, K_d)$
$\psi^* \leftarrow \mathsf{KEncap}(ek, \omega, t^*)$
$d' \leftarrow \mathcal{A}^{\mathsf{KDecap}(dk,\cdot,\cdot),\mathsf{KProve}(dk,\cdot,\cdot)}(\mathtt{guess}, state, \psi^*)$
if $d = d'$ return 1 else return 0

$\mathbf{Exp}^{\text{Tag-KEMNO-commit}}(\lambda)$:
$(ek, dk) \leftarrow \mathsf{KKg}(1^\lambda)$
$(t, C, K, \theta, K', \theta') \leftarrow \mathcal{A}(ek, dk)$
return 1 if the following holds:
$\mathsf{KVerify}(ek, t, C, K, \theta) = 1$,
$\mathsf{KVerify}(ek, t, C, K', \theta') = 1$,
$K \neq K'$.
otherwise return 0

Fig. 1 The experiments for the definitions of security.

In the experiment $\mathbf{Exp}^{\text{PKENO-CCPA}}(\lambda)$, the special inputs `find` and `guess` are used for indicating in which phase the adversary is running. The same holds for $\mathbf{Exp}^{\text{DEM}}(\lambda)$ and $\mathbf{Exp}^{\text{Tag-KEMNO-CCPA}}(\lambda)$.

We give the definition of soundness of proofs (called the committing property), which was defined by Galindo et al. [6]. In words this property requires that even a malicious receiver cannot lie about the decryption result of a ciphertext.

Definition 2. A PKENO scheme is committing if for any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage $\Pr[\mathbf{Exp}^{\text{PKENO-commit}}(\lambda) = 1]$ is negligible in λ, where the experiment $\mathbf{Exp}^{\text{PKENO-commit}}(\lambda)$ is defined in Fig. 1.

2.2 Data Encapsulation Mechanism

A data encapsulation mechanism (DEM) scheme is defined by the following two algorithms [7].

- $\mathsf{DEnc}(K, M) \to \chi$. The encryption algorithm takes as inputs a session key K and a plaintext M, and outputs a ciphertext χ.
- $\mathsf{DDec}(K, \chi) \to M$. The decryption algorithm takes as inputs a session key K and a ciphertext χ, and outputs a plaintext M.

For correctness, we require the following: For any session key K and any plaintext M, it holds that $\mathsf{DDec}(K, \mathsf{DEnc}(K, M)) = M$.

We require a DEM scheme to satisfy passive security [7].

Definition 3. A DEM scheme is secure against passive attacks if for any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage $|\Pr[\mathbf{Exp}^{\text{DEM}}(\lambda) = 1] - 1/2|$ is negligible in λ, where $\mathbf{Exp}^{\text{DEM}}(\lambda)$ is defined in Fig. 1, and $\mathcal{K}$ is the key space of the DEM scheme.

2.3 Groups with and without a Bilinear Map

In our instantiation, we make use of bilinear groups. Let $\mathcal{G}$ be a probabilistic polynomial-time algorithm which takes as an input a security parameter 1^λ and outputs a tuple $(p, \mathbb{G}, \mathbb{G}_T, e, g)$ where p is a prime, $\mathbb{G}$ and $\mathbb{G}_T$ are multiplicative groups of order p, $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a non-degenerate bilinear map, and g is a generator of $\mathbb{G}$. We say that the decisional bilinear Diffie-Hellman (DBDH) assumption holds if for any probabilistic polynomial-time algorithm $\mathcal{A}$,

$$\begin{aligned}|\Pr[gk = (p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda); \alpha, \beta, \gamma \leftarrow \mathbb{Z}_p :\\ \mathcal{A}(gk, g^\alpha, g^\beta, g^\gamma, e(g,g)^{\alpha\beta\gamma}) = 1]\\ - \Pr[gk = (p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda); \alpha, \beta, \gamma, \tau \leftarrow \mathbb{Z}_p :\\ \mathcal{A}(gk, g^\alpha, g^\beta, g^\gamma, e(g,g)^\tau) = 1]|\end{aligned}$$

is negligible. We say that the decisional linear (DLIN) assumption holds if for any probabilistic polynomial-time algorithm $\mathcal{A}$,

$$\begin{aligned}|\Pr[gk = (p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda); u, v \leftarrow \mathbb{G} \setminus \{1\};\\ \alpha, \beta \leftarrow \mathbb{Z}_p : \mathcal{A}(gk, u, v, u^\alpha, v^\beta, g^{\alpha+\beta}) = 1]\\ - \Pr[gk = (p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda); u, v, \leftarrow \mathbb{G} \setminus \{1\};\\ \alpha, \beta, \gamma \leftarrow \mathbb{Z}_p : \mathcal{A}(gk, u, v, u^\alpha, v^\beta, g^\gamma) = 1]|\end{aligned}$$

is negligible.

In addition, we also employ a group without a bilinear map. Let $\mathcal{G}_{\mathrm{ddh}}$ be a probabilistic polynomial-time algorithm which takes as an input a security parameter 1^λ and outputs a tuple $(p, \mathbb{G}, g)$ where p is a prime, $\mathbb{G}$ is a multiplicative group of order p, and g is a generator of $\mathbb{G}$. We say that the decisional Diffie-Hellman (DDH) assumption holds if for any probabilistic polynomial-time algorithm $\mathcal{A}$,

$$\begin{aligned}|\Pr[&gk = (p, \mathbb{G}) \leftarrow \mathcal{G}_{\mathrm{ddh}}(1^\lambda);\\ &\alpha, \beta \leftarrow \mathbb{Z}_p : \mathcal{A}(gk, g^\alpha, g^\beta, g^{\alpha\beta}) = 1]\\ -\Pr[&gk = (p, \mathbb{G}) \leftarrow \mathcal{G}_{\mathrm{ddh}}(1^\lambda);\\ &\alpha, \beta, \gamma \leftarrow \mathbb{Z}_p : \mathcal{A}(gk, g^\alpha, g^\beta, g^\gamma) = 1]|\end{aligned}$$

is negligible.

2.4 Collision-Resistant Hash Function

A hash function family is defined as a pair $(\mathcal{H}, \mathsf{Hash})$ of algorithms: the hash key generation algorithm $\mathcal{H}$ takes as an input a security parameter 1^λ, and outputs a hash key hk; the deterministic hashing algorithm Hash takes as inputs a hash key hk and a message $M \in \{0,1\}^*$, and outputs a hash value h. A hash function family is collision resistant if for any probabilistic polynomial-time algorithm $\mathcal{A}$, the advantage $\Pr[hk \leftarrow \mathcal{H}(1^\lambda); (M, M') \leftarrow \mathcal{A}(hk) : M \neq M' \wedge \mathsf{Hash}(hk, M) = \mathsf{Hash}(hk, M')]$ is negligible in λ.

2.5 One-Time Signatures

A signature scheme is defined as a tuple $(\mathsf{SKg}, \mathsf{SSign}, \mathsf{SVerify})$ of three algorithms: the key generation algorithm SKg takes as an input a security parameter 1^λ, and outputs a pair (vk, sk) of a verification key and a signing key; the signing algorithm SSign takes as inputs a signing key sk and a message M, and outputs a signature σ; the verification algorithm $\mathsf{SVerify}$ takes as inputs a verification key vk, a message M, and a signature σ, and outputs a bit 1 or 0. For correctness, we require the following: for any $\lambda \in \mathbb{N}$, any $(vk, sk) \leftarrow \mathsf{SKg}(1^\lambda)$, and any message M, it holds that $\mathsf{SVerify}(vk, M, \mathsf{SSign}(sk, M)) = 1$. A signature scheme $(\mathsf{SKg}, \mathsf{SSign}, \mathsf{SVerify})$ is said to be a strongly unforgeable one-time signature scheme, if for any probabilistic polynomial-time adversary $\mathcal{A}$, it holds that $\Pr[(vk, sk) \leftarrow \mathsf{SKg}(1^\lambda); (M, state) \leftarrow \mathcal{A}(vk); \sigma \leftarrow \mathsf{SSign}(sk, M); (m^*, \sigma^*) \leftarrow \mathcal{A}(state, \sigma) : (m, \sigma) \neq (m^*, \sigma^*) \wedge \mathsf{SVerify}(vk, m^*, \sigma^*) = 1]$ is negligible in λ.

3. Tag-KEM with Non-Interactive Opening

Now we define a new cryptographic primitive named *Tag-KEM with Non-interactive Opening (Tag-KEMNO)*. A Tag-KEMNO scheme consists of the following six algorithms.

- $\mathsf{KKg}(1^\lambda) \to (ek, dk)$. The key generation algorithm takes as an input a security parameter 1^λ and outputs a pair (ek, dk) of an encapsulation key and a decapsulation key.
- $\mathsf{KKey}(ek) \to (\omega, K)$. The session key generation algorithm takes as an input an encapsulation key ek and outputs a pair of state information ω and a session key K.
- $\mathsf{KEncap}(ek, \omega, t) \to \psi$. The encapsulation algorithm takes as inputs an encapsulation key ek, state information ω, and a tag $t \in \{0,1\}^*$, and outputs a ciphertext ψ.
- $\mathsf{KDecap}(dk, t, \psi) \to K/\perp$. The decapsulation algorithm takes as inputs a decapsulation key dk, a tag t, and a ciphertext C, and outputs a session key K or the rejection symbol $\perp$.
- $\mathsf{KProve}(dk, t, \psi) \to \theta$. The proof algorithm takes as inputs a decapsulation key dk, a tag t, and a ciphertext ψ, and outputs a proof θ.
- $\mathsf{KVerify}(ek, t, \psi, K, \theta) \to 1/0$. The verification algorithm takes as inputs an encapsulation key ek, a tag t, a ciphertext ψ, a session key K, and a proof θ, and outputs 1 or 0 indicating the validity of the proof.

For correctness, we require the following: For any $\lambda \in \mathbb{N}$, any key pair $(ek, dk) \leftarrow \mathsf{KKg}(1^\lambda)$, and any tag $t \in \{0,1\}^*$,

- for any state information and session key $(\omega, K) \leftarrow \mathsf{KKey}(ek)$, it holds that $\mathsf{KDecap}(dk, t, \mathsf{KEncap}(ek, \omega, t)) = K$
- for any (either valid and invalid) ciphertext ψ it holds that $\mathsf{KVerify}(ek, t, \psi, \mathsf{KDecap}(dk, t, \psi), \mathsf{KProve}(dk, t, \psi)) = 1$.

We then give definitions of confidentiality and soundness of the proof.

Definition 4. A Tag-KEMNO scheme is indistinguishable against chosen ciphertext and proof attacks if for any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage $|\Pr[\mathbf{Exp}^{\text{Tag-KEMNO-CCPA}}(\lambda) = 1] - 1/2|$ is negligible in λ, where the experiment $\mathbf{Exp}^{\text{Tag-KEMNO-CCPA}}(\lambda)$ is defined in Fig. 1, and in the `guess` phase, the adversary is not allowed to submit (t^*, ψ^*) to either KDecap or KProve, and $\mathcal{K}$ is the session key space associated with the scheme.

Definition 5. A Tag-KEMNO scheme is committing if for any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage $\Pr[\mathbf{Exp}^{\text{Tag-KEMNO-commit}}(\lambda) = 1]$ is negligible in λ, where the experiment $\mathbf{Exp}^{\text{Tag-KEMNO-commit}}$ is defined in Fig. 1.

We emphasize that it is crucial for our construction that the game only forbids an adversary to issue the challenge pair (t^*, ψ^*) as a decapsulation query. In particular, an adversary is *not* forbidden to issue the challenge ciphertext ψ^* with a tag t if $t \neq t^*$. It is essential for resisting Galindo's attack [6] to make the underlying Tag-KEMNO scheme secure against adversaries making this type of query.

4. The Tag-KEMNO/DEM Composition Theorem

In this section we present our generic construction of

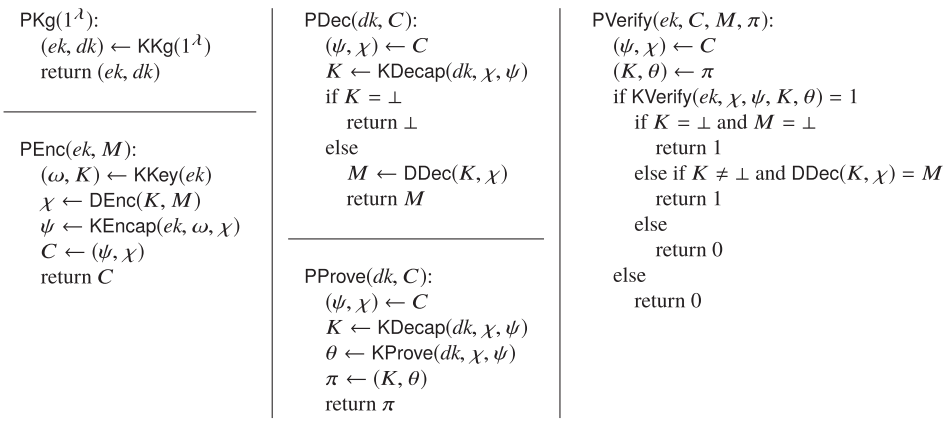

Fig. 2 The proposed construction of PKENO.

PKENO from Tag-KEMNO and DEM. Let (KKg, KKey, KEncap, KDecap, KProve, KVerify) be a Tag-KEMNO scheme and (DEnc, DDec) be a DEM scheme. Then we construct a PKENO scheme as in Fig. 2.

We explain the idea of the construction. As we mentioned in the introduction, the main idea is to bind a DEM ciphertext to the KEM ciphertext in a publicly verifiable manner. This idea is implemented by the process $\psi \leftarrow \mathsf{KEncap}(ek, \omega, \chi)$ where χ is the DEM ciphertext used as the tag. The proof for a ciphertext (ψ, χ) will be generated by revealing the decapsulation of ψ under the tag χ.

The intuition of the security of the scheme is as follows. Denoting the challenge ciphertext by (ψ^*, χ^*), let us consider two types of queries. The first type of queries is those (ψ, χ) satisfying $\psi \neq \psi^*$, while the second type is those satisfying $\psi = \psi^*$ but $\chi \neq \chi^*$. The first type is relatively easily dealt with, as the underlying Tag-KEMNO scheme is chosen-ciphertext secure. Precisely, due to the chosen-ciphertext security, revealing the decryption result of a ciphertext ψ ($\neq \psi^*$) to an adversary does not affect the security of ψ^*. Then let us consider the latter case. As described in the indistinguishability game for a Tag-KEMNO scheme, the Tag-KEMNO part ψ^* of the challenge is still secure even when an adversary learns the decapsulation of ψ^* itself *under a tag χ different from χ^**. This restriction exactly matches the restriction in the indistinguishability game of PKENO. Namely, in the Tag-KEM game an adversary is allowed to query ψ^* with a tag $\chi \neq \chi^*$, while in the PKENO game an adversary is allowed to query (ψ^*, χ) when $\chi \neq \chi^*$. Our scheme can be proven secure assuming the Tag-KEMNO scheme is secure.

Theorem 6. *The proposed construction is indistinguishable against chosen ciphertext and proof attacks, if the underlying Tag-KEMNO scheme is indistinguishable against chosen ciphertext and proof attacks and the underlying DEM scheme is secure against passive attacks.*

Proof. Let $\mathcal{A}$ be an adversary against the PKENO scheme. For the proof of security, we consider the following sequence of games.

- **Game 0.** This game is identical to $\mathbf{Exp}^{\text{PKENO-CCPA}}(\lambda)$.
- **Game 1.** In this game the challenge ciphertext is modified to use a fresh session key K' instead of using the session key K encapsulated in ψ^*. More precisely, the experiment executes the following procedure to generate the challenge ciphertext: $(\omega, K) \leftarrow \mathsf{KKey}(ek)$; $K' \leftarrow \mathcal{K}$; $\chi^* \leftarrow \mathsf{DEnc}(K', M_b)$; $\psi^* \leftarrow \mathsf{KEncap}(ek, \omega, \chi^*)$; then return (ψ^*, χ^*) to the adversary.

For Game i, we denote by succ_i the event that $b = b'$ holds. Then we have that $\Pr[\mathbf{Exp}^{\text{PKENO-CCPA}}(\lambda)] = \Pr[\mathsf{succ}_0]$ and $|\Pr[\mathsf{succ}_0] - 1/2| \leq |\Pr[\mathsf{succ}_0] - \Pr[\mathsf{succ}_1]| + |\Pr[\mathsf{succ}_1] - 1/2|$. In the following we bound these two terms.

Lemma 7. *There is a probabilistic polynomial-time adversary $\mathcal{B}_{\text{DEM}}$ that attacks the underlying Tag-KEMNO scheme in the sense of indistinguishability against chosen ciphertext and proof attacks, and whose advantage is ε_{KEM} such that $|\Pr[\mathsf{succ}_0] - \Pr[\mathsf{succ}_1]| = 2\varepsilon_{\text{DEM}}$.*

Proof of Lemma 7. We construct an adversary $\mathcal{B}_{\text{KEM}}$ that attacks the indistinguishability of the underlying Tag-KEMNO scheme. The description of $\mathcal{B}_{\text{KEM}}$ is as follows:

- $\mathcal{B}_{\text{KEM}}(\mathtt{find}, ek, K^*)$**.** $\mathcal{B}_{\text{KEM}}$ first chooses a random bit $b \leftarrow \{0, 1\}$ and runs $\mathcal{A}(\mathtt{find}, ek)$. For a decryption query (ψ, χ), $\mathcal{B}_{\text{KEM}}$ issues the decapsulation query (χ, ψ) to its decapsulation oracle, and receives K. If $K = \perp$, $\mathcal{B}_{\text{KEM}}$ returns $\perp$ to $\mathcal{A}$, and otherwise it runs $M \leftarrow \mathsf{DDec}(K, \chi)$ and returns M to $\mathcal{A}$. For a proof query (ψ, χ), $\mathcal{B}_{\text{KEM}}$ issues the decapsulation query (χ, ψ) to obtain K and issues the proof query (χ, ψ) to obtain θ. Then $\mathcal{B}_{\text{KEM}}$ returns $\pi = (K, \theta)$ to $\mathcal{A}$. When $\mathcal{A}$ outputs $(M_0, M_1, state_{\mathcal{A}})$ and terminates, $\mathcal{B}_{\text{KEM}}$ computes $\chi^* \leftarrow \mathsf{DDec}(K^*, M_b)$. Finally $\mathcal{B}_{\text{KEM}}$ sets $state_{\mathcal{B}_{\text{KEM}}}$ to be the entire view of $\mathcal{B}_{\text{KEM}}$ itself and terminates with output $(\chi^*, state_{\mathcal{B}_{\text{KEM}}})$.
- $\mathcal{B}_{\text{KEM}}(\mathtt{guess}, state_{\mathcal{B}_{\text{KEM}}}, \psi^*)$**.** $\mathcal{B}_{\text{KEM}}$ runs $\mathcal{A}(\mathtt{guess}, state_{\mathcal{A}}, (\psi^*, \chi^*))$. For decryption queries and proof queries, $\mathcal{B}_{\text{KEM}}$ replies as in the `find` phase. In this

step, $\mathcal{B}_{\mathrm{KEM}}$ cannot obtain the decapsulation and proof of (χ^*, ψ^*) from its oracles. However, this does not cause failure of the simulation, because $\mathcal{A}$ is also not allowed to submit (χ^*, ψ^*) as a query. When $\mathcal{A}$ outputs a bit b' and terminates, $\mathcal{B}_{\mathrm{KEM}}$ sets $d' \leftarrow 0$ if $b = b'$ and sets $d' \leftarrow 1$ otherwise. Then $\mathcal{B}_{\mathrm{KEM}}$ outputs d'.

If $\mathcal{B}_{\mathrm{KEM}}$ receives the real key as K^* (namely, $d = 0$ where d is the random bit chosen by the experiment $\mathbf{Exp}^{\text{Tag-KEMNO-CCPA}}(\lambda)$), $\mathcal{B}_{\mathrm{KEM}}$ perfectly simulates Game 0. Furthermore, if $\mathcal{B}_{\mathrm{KEM}}$ receives a random key as K^* ($d = 1$), $\mathcal{B}_{\mathrm{KEM}}$ perfectly simulates Game 1. Hence we have that

$$\begin{aligned}
&|\Pr[\mathsf{succ}_0] - \Pr[\mathsf{succ}_1]| \\
&= |\Pr[b = b' | d = 0] - \Pr[b = b' | d = 1]| \\
&= |\Pr[b = b' | d = 0] + \Pr[b \neq b' | d = 1] - 1| \\
&= \left| \frac{\Pr[b = b' \wedge d = 0]}{\Pr[d = 0]} + \frac{\Pr[b \neq b' \wedge d = 1]}{\Pr[d = 1]} - 1 \right| \\
&= 2 \left| \Pr[b = b' \wedge d = 0] + \Pr[b \neq b' \wedge d = 1] - \frac{1}{2} \right| \\
&= 2 \left| \Pr[d' = 0 \wedge d = 0] + \Pr[d' = 1 \wedge d = 1] - \frac{1}{2} \right| \\
&= 2 \left| \Pr[d = d'] - \frac{1}{2} \right| = 2\varepsilon_{\mathrm{KEM}},
\end{aligned}$$

where the fourth equality uses $\Pr[d = 0] = \Pr[d = 1] = 1/2$, and the fifth equality comes from the construction of $\mathcal{B}_{\mathrm{KEM}}$ that outputs $d' = 0$ if $b = b'$ and outputs $d' = 1$ if $b \neq b'$. This completes the proof of Lemma 7. □

Lemma 8. *There is a probabilistic polynomial-time adversary $\mathcal{B}_{\mathrm{DEM}}$ that attacks the underlying DEM scheme in the sense of security against passive attacks, whose advantage is $\varepsilon_{\mathrm{DEM}} = |\Pr[\mathsf{succ}_1] - 1/2|$.*

Proof of Lemma 8. We construct an adversary $\mathcal{B}_{\mathrm{DEM}}$ which attacks security against passive attacks of the underlying DEM scheme. The description of $\mathcal{B}_{\mathrm{DEM}}$ is as follows.

- $\mathcal{B}_{\mathrm{DEM}}(\mathtt{find}, 1^\lambda)$**.** $\mathcal{B}_{\mathrm{DEM}}$ runs $(ek, dk) \leftarrow \mathsf{KKg}(1^\lambda)$. Then $\mathcal{B}_{\mathrm{DEM}}$ runs $\mathcal{A}(\mathtt{find}, ek)$. For a decryption query (ψ, χ), $\mathcal{B}_{\mathrm{DEM}}$ runs $K \leftarrow \mathsf{KDecap}(dk, \chi, \psi)$ to obtain K and if $K = \perp$, $\mathcal{B}_{\mathrm{DEM}}$ returns $\perp$ to $\mathcal{A}$. If $K \neq \perp$, $\mathcal{B}_{\mathrm{DEM}}$ runs $M \leftarrow \mathsf{DDec}(K, \chi)$ and returns M to $\mathcal{A}$. For a proof query (ψ, χ), $\mathcal{B}_{\mathrm{DEM}}$ runs $K \leftarrow \mathsf{KDecap}(dk, \chi, \psi)$ and $\theta \leftarrow \mathsf{KProve}(dk, \chi, \psi)$ and returns $\pi = (K, \theta)$ to $\mathcal{A}$. When $\mathcal{A}$ outputs $(M_0, M_1, state_{\mathcal{A}})$ and terminates, $\mathcal{B}_{\mathrm{DEM}}$ outputs $(M_0, M_1, state_{\mathcal{B}_{\mathrm{DEM}}})$ where $state_{\mathcal{B}_{\mathrm{DEM}}}$ is the entire view of $\mathcal{B}_{\mathrm{DEM}}$, and terminates.
- $\mathcal{B}_{\mathrm{DEM}}(\mathtt{guess}, state_{\mathcal{B}_{\mathrm{DEM}}}, \chi^*)$**.** $\mathcal{B}_{\mathrm{DEM}}$ generates the challenge ciphertext by running $(\omega, K) \leftarrow \mathsf{KKey}(ek)$ and $\psi^* \leftarrow \mathsf{KEncap}(ek, \omega, \chi^*)$. Then $\mathcal{B}_{\mathrm{DEM}}$ runs $\mathcal{A}(\mathtt{guess}, state_{\mathcal{A}}, (\psi^*, \chi^*))$. Decryption queries and proof queries are replied as in the previous phase. Finally when $\mathcal{A}$ outputs b' and terminates, $\mathcal{B}_{\mathrm{DEM}}$ sets $d' \leftarrow b'$ and outputs d'.

$\mathcal{B}_{\mathrm{DEM}}$ perfectly simulates Game 1, and in particular the bit d that the experiment $\mathbf{Exp}^{\mathrm{DEM}}(\lambda)$ generates corresponds to the bit b that the simulated Game 1 generates. Furthermore, since the output d' of $\mathcal{B}_{\mathrm{DEM}}$ is equal to b', we have $|\Pr[\mathsf{succ}_1] - 1/2| = |\Pr[b = b'] - 1/2| = |\Pr[d = d'] - 1/2| = \varepsilon_{\mathrm{DEM}}$. This completes the proof of Lemma 8. □

Finally, combining the lemmas, we have that

$$\left| \Pr[\mathbf{Exp}^{\text{PKENO-CCPA}}(\lambda) = 1] - \frac{1}{2} \right| \leq 2\varepsilon_{\mathrm{KEM}} + \varepsilon_{\mathrm{DEM}}.$$

From the assumption that the underlying schemes are secure, we have that the above is negligible, which completes the entire proof of Theorem 6. □

Theorem 9. *The proposed construction is committing if the underlying Tag-KEMNO is committing and the decryption algorithm of the underlying DEM scheme is deterministic.*

Proof. Let us consider an adversary $\mathcal{A}$ against the committing property. We then show that there exists a reduction $\mathcal{B}$ such that if $\mathcal{A}$ successfully breaks the committing property of the PKENO scheme, $\mathcal{B}$ successfully breaks the committing property of the underlying Tag-KEMNO scheme.

The description of $\mathcal{B}$ is as follows: Given an encapsulation key ek and the decapsulation key dk, $\mathcal{B}$ runs $\mathcal{A}(1^\lambda, ek, dk)$ and obtains (C, M, π, M', π'); then $\mathcal{B}$ parses C to (ψ, χ), π to (K, θ), and π' to (K', θ'); finally $\mathcal{B}$ outputs $(\chi, \psi, K, \theta, K', \theta')$.

We then analyze this $\mathcal{B}$. Let (C, M, π, M', π') be an output of $\mathcal{A}$ and let $C = (\psi, \chi)$, $\pi = (K, \theta)$, and $\pi' = (K', \theta)$. We then argue that assuming this output satisfies the winning condition of $\mathcal{A}$, the output $(\chi, \psi, K, \theta, K', \theta')$ satisfies the winning condition of $\mathcal{B}$. More specifically, assuming $\mathsf{PVerify}(ek, C, M, \pi) = 1$, $\mathsf{PVerify}(ek, C, M', \pi') = 1$, and $M \neq M'$, we argue that $\mathsf{KVerify}(ek, \chi, \psi, K, \theta) = 1$, $\mathsf{KVerify}(ek, \chi, \psi, K', \theta') = 1$, and $K \neq K'$. Due to the construction of $\mathsf{PVerify}$, $\mathsf{PVerify}(ek, C, M, \pi) = \mathsf{PVerify}(ek, C, M', \pi') = 1$ implies that $\mathsf{KVerify}(ek, \chi, \psi, K, \theta) = 1$ and $\mathsf{KVerify}(ek, \chi, \psi, K', \theta') = 1$.

In order to show that $K \neq K'$, for contradiction we assume that $K = K'$. We have two cases: (1) $K = K' = \perp$; (2) $K = K' \neq \perp$. For the case (1), due to the construction of $\mathsf{PVerify}$, we have that $M = M' = \perp$, which contradicts to the assumption that $M \neq M'$. For the case (2), due to the construction of $\mathsf{PVerify}$, we have that $M = \mathsf{DDec}(K, \chi)$ and $M' = \mathsf{DDec}(K', \chi)$. Because $K = K'$ and DDec is deterministic, we have that $M = M'$, which is again a contradiction. In any case we have a contradiction, thus we have that $K \neq K'$.

Therefore, for any case, whenever $\mathcal{A}$ satisfies the winning condition of its experiment, $\mathcal{B}$ satisfies its own winning condition. □

5. Instantiations

In this section we show four instantiations of our framework to demonstrate that our framework explains existing

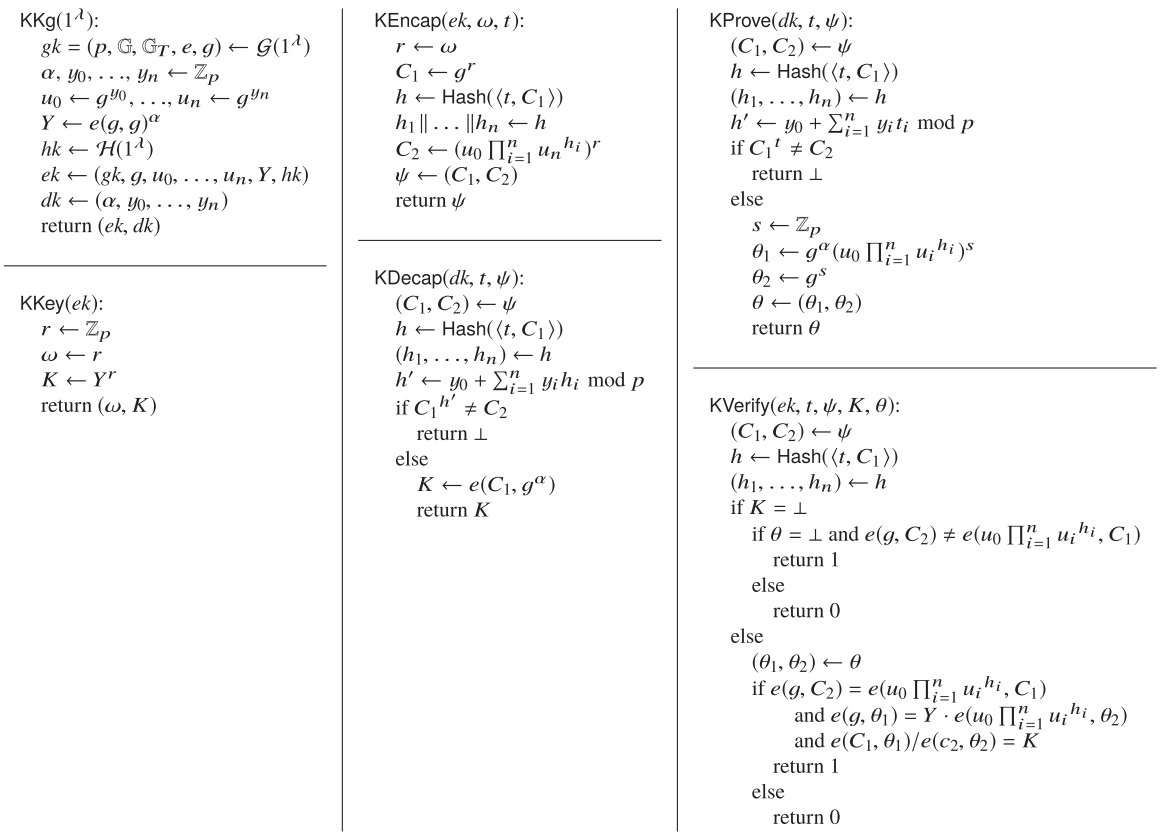

Fig. 3 An instantiation of Tag-KEMNO from the DBDH assumption based on Galindo's scheme [4]. In this scheme we denote by n the length of the output of the hash function Hash.

constructions of PKENO. The instantiations are shown in Figs. 3–6, each of which shows an instantiation of Tag-KEMNO from various number-theoretic assumptions. Those instantiations are respectively obtained by modifying and adapting the PKENO scheme by Galindo [4], the PKENO scheme by Lai et al. [5], and two schemes by Galindo et al. [6].

The first instantiation (Fig. 3) is based on the DBDH assumption and the collision-resistant hash function family. The indistinguishability is proven from the DBDH assumption and the collision resistance of the hash function family, while the committing property is proven unconditionally.

The second instantiation (Fig. 4) is also based on the DBDH assumption and the collision-resistant hash function family. The indistinguishability is proven from the DBDH assumption and the collision resistance of hash function family, while the committing property is proven unconditionally.

The third instantiation (Fig. 5) is based on the DDH assumption and the random oracle model. The indistinguishability is proven in the random oracle model from the DDH assumption, while the committing property is proven unconditionally in the random oracle model. We notice that both of the indistinguishability and the committing property are proven in the random oracle model.

The last instantiation (Fig. 6) is based on the DLIN assumption and a strongly unforgeable one-time signature scheme. The indistinguishability of this scheme is proven from the DLIN assumption and the strong unforgeability of the one-time signature scheme, while the committing property is proven unconditionally.

We omit the security proofs of these instantiations because the proofs are easily obtained by modifying the proofs of the original schemes.

These four schemes have their own merits and demerits. Among these schemes, the first scheme based on Galindo's scheme (Fig. 3) has the shortest ciphertext length. Two drawbacks of this scheme are a long encryption key and a large reduction loss. These drawbacks are overcome by the second scheme based on Lai et al.'s PKENO scheme (Fig. 4). This scheme has a shorter encryption key (a constant number of group elements) and a tighter reduction to the DBDH assumption, at the cost of slightly longer ciphertexts. The third construction based on Galindo et al.'s DDH-based scheme achieves the shortest public key size among these schemes. In addition, the construction does not make use of any pairing operations. These benefits are achieved at the cost of depending on the random oracle model. The last scheme, based on Galindo et al.'s DLIN-based scheme, is structure-preserving [9] (in the sense that the ciphertext does not contain $\mathbb{G}_T$-elements) when combined with a one-time pad over the group $\mathbb{G}$. Thus it is useful for building

KKg(1^λ):
$gk = (p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda)$
$\alpha, x, y, z \leftarrow \mathbb{Z}_p$
$g_1 \leftarrow g^\alpha$
$u \leftarrow g^x$
$v \leftarrow g^y$
$d \leftarrow g^z$
$g_2 \leftarrow \mathbb{G}$
$Z \leftarrow e(g_1, g_2)$
$hk \leftarrow \mathcal{H}(1^\lambda)$
$ek \leftarrow (gk, g, hk, Z, u, v, d)$
$dk \leftarrow ({g_2}^\alpha, x, y, z)$
return (ek, dk)

KKey(ek):
$s \leftarrow \mathbb{Z}_p$
$\omega \leftarrow s$
$K \leftarrow Z^s$
return (ω, K)

KEncap(ek, ω, t):
$s \leftarrow \omega$
$C_1 \leftarrow g^s$
$h \leftarrow \mathsf{Hash}(\langle t, C_1 \rangle)$
$r \leftarrow \mathbb{Z}_p$
$C_2 \leftarrow (u^h v^r d)^s$
$\psi \leftarrow (C_1, C_2, r)$
return ψ

KDecap(dk, t, ψ):
$({g_2}^\alpha, x, y, z) \leftarrow dk$
$(C_1, C_2, r) \leftarrow \psi$
$h \leftarrow \mathsf{Hash}(\langle t, C_1 \rangle)$
if ${C_1}^{hx+ry+z} \neq C_2$
return $\perp$
else
return $e(C_1, {g_2}^\alpha)$

KProve(dk, t, ψ):
$({g_2}^\alpha, x, y, z) \leftarrow dk$
$(C_1, C_2, r) \leftarrow \psi$
$h \leftarrow \mathsf{Hash}(\langle t, C_1 \rangle)$
if ${C_1}^{hx+ry+z} \neq C_2$
return $\perp$
else
$\gamma \leftarrow \mathbb{Z}_p$
$\theta_1 \leftarrow {g_2}^\alpha (u^t v^r d)^\gamma$
$\theta_2 \leftarrow g^\gamma$
$\theta \leftarrow (\theta_1, \theta_2)$
return θ

KVerify(ek, t, ψ, K, θ):
$(C_1, C_2, r) \leftarrow \psi$
$h \leftarrow \mathsf{Hash}(\langle t, C_1 \rangle)$
if $K = \perp$
if $\theta = \perp$ and $e(C_1, u^h v^r d) \neq e(g, C_2)$
return 1
else
return 0
else
$(\theta_1, \theta_2) \leftarrow \theta$
$h \leftarrow \mathsf{Hash}(\langle t, C_1 \rangle)$
if $e(C_1, u^h v^r d) = e(g, C_2)$
and $e(g, \theta_1) = Z \cdot e(u^h v^r d, \theta_2)$
and $e(C_1, \theta_1)/e(C_2, \theta_2) = K$
return 1
else
return 0

Fig. 4 An instantiation of Tag-KEMNO from the DBDH assumption based on Lai et al.'s scheme [5].

KKg(1^λ):
$(p, \mathbb{G}, g) \leftarrow \mathcal{G}_{\mathrm{ddh}}(1^\lambda)$
$x \leftarrow \mathbb{Z}_p$
$g, \bar{g} \leftarrow \mathbb{G}$
choose hash functions H_1, H_2
$ek \leftarrow (p, \mathbb{G}, g, \bar{g}, h)$
$dk \leftarrow x$
return (ek, dk)

KKey(ek):
$r \leftarrow \mathbb{Z}_p$
$\omega \leftarrow r$
$K \leftarrow h^r$
return (ω, K)

KEncap(ek, ω, t):
$r \leftarrow \omega$
$s \leftarrow \mathbb{Z}_p$
$C \leftarrow g^r$
$R \leftarrow g^s$
$\bar{C} \leftarrow \bar{g}^r$
$\bar{R} \leftarrow \bar{g}^s$
$e \leftarrow H_1(\langle t, C, R, \bar{C}, \bar{R} \rangle)$
$f \leftarrow s + er$
$\psi \leftarrow (C, \bar{C}, e, f)$

KDecap(dk, t, ψ):
$(C, \bar{C}, e, f) \leftarrow \psi$
if $c = H_1(\langle t, C, g^f / C^e, \bar{C}, \bar{g}^f / \bar{C}^e \rangle)$
$K \leftarrow {C_1}^x$
return K
else
return $\perp$

KProve(dk, t, ψ):
$(C, \bar{C}, e, f) \leftarrow \psi$
if $c = H_1(\langle t, C, g^f / C^e, \bar{C}, \bar{g}^f / \bar{C}^e \rangle)$
$s' \leftarrow \mathbb{Z}_p$
$R' \leftarrow g^{s'}$
$R'' \leftarrow C^{s'}$
$e' \leftarrow H_2(\langle C, K, R', R'' \rangle)$
$f' \leftarrow s' + e'x$
$\theta \leftarrow (e', f')$
else
$\theta \leftarrow \perp$

KVerify(ek, t, ψ, K, θ):
$(C, \bar{C}, e, f) \leftarrow \psi$
if $K = \perp$
if $\theta = \perp$ and $e \neq H_1(\langle t, C, g^f / C^e, \bar{C}, \bar{g}^f / \bar{C}^e \rangle)$
return 1
else
return 0
else
$(e', f') \leftarrow \theta$
if $e = H_1(\langle t, C, g^f / C^e, \bar{C}, \bar{g}^f / \bar{C}^e \rangle)$
and $e' = H_2(\langle C, K, g^{f'} / h^{e'}, C^{f'} / K^{e'} \rangle)$
return 1
else
return 0

Fig. 5 An instantiation from the DDH assumption based on Galindo et al.'s scheme [6].

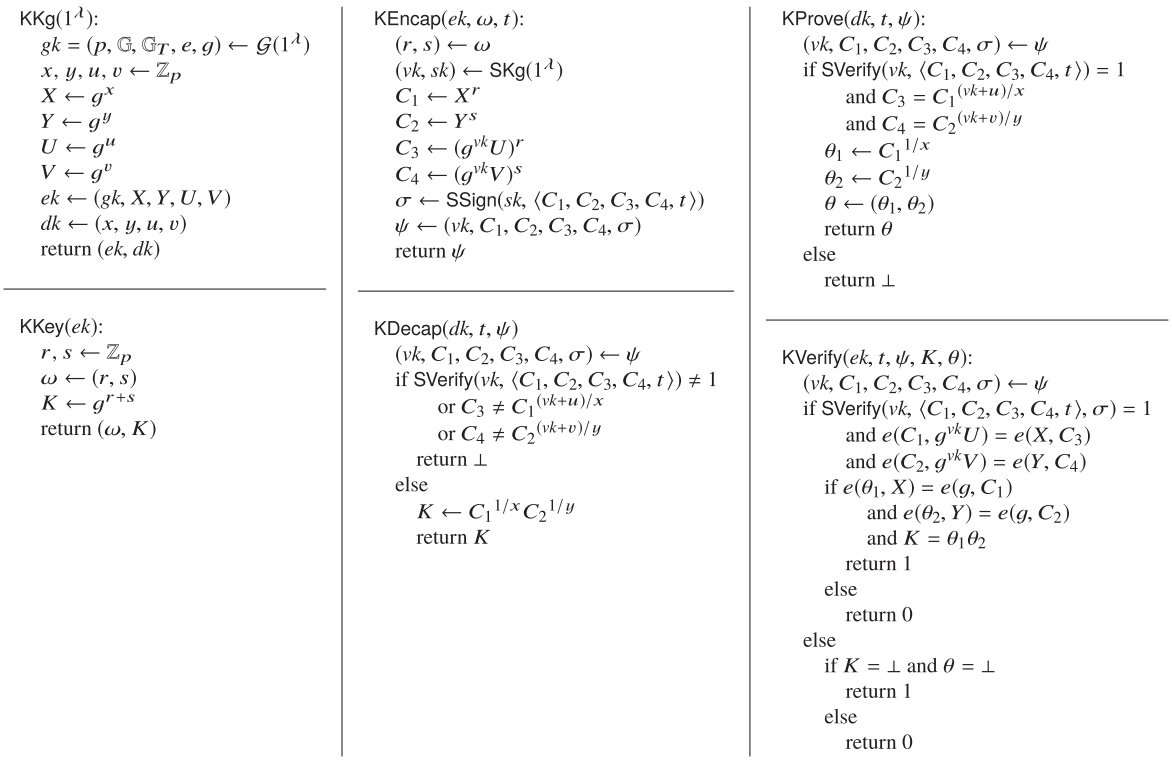

Fig. 6 An instantiation from the DLIN assumption based on Galindo et al.'s scheme [6].

Table 1 Comparison among PKENO schemes.

	Encryption key	Decryption key	Ciphertext	Proof	Plaintext	Assumption
Galindo	$(n+2)\ell_{\mathbb{G}} + \ell_{\mathbb{G}_T} + \ell_{hk}$	$(n+2)\ell_{\mathbb{Z}_p}$	$\ell_{\mathrm{DEM}} + 2\ell_{\mathbb{G}}$	$2\ell_{\mathbb{G}}$	$\{0,1\}^*$	DBDH, CR, DEM
Lai et al.	$4\ell_{\mathbb{G}} + \ell_{\mathbb{G}_T} + \ell_{hk}$	$\ell_{\mathbb{G}} + 3\ell_{\mathbb{Z}_p}$	$\ell_{\mathbb{G}_T} + 2\ell_{\mathbb{G}} + \ell_{\mathbb{Z}_p}$	$2\ell_{\mathbb{G}}$	$\mathbb{G}_T$	DBDH, CR
Galindo et al. (1)	$3\ell_{\mathbb{G}_{\mathrm{ddh}}}$	$\ell_{\mathbb{Z}_p}$	$\ell_{H_1} + 2\ell_{\mathbb{G}_{\mathrm{ddh}}} + 2\ell_{\mathbb{Z}_p}$	$\ell_{\mathbb{G}_{\mathrm{ddh}}} + 2\ell_{\mathbb{Z}_p}$	$\{0,1\}^{\ell_{H_1}}$	DDH, RO
Galindo et al. (2)	$5\ell_{\mathbb{G}}$	$4\ell_{\mathbb{Z}_p}$	$\ell_{vk} + 5\ell_{\mathbb{G}} + \ell_{\sigma}$	$2\ell_{\mathbb{G}}$	$\mathbb{G}$	DLIN, One-time sig.
Ours (Fig. 3)	$(n+2)\ell_{\mathbb{G}} + \ell_{\mathbb{G}_T} + \ell_{hk}$	$(n+2)\ell_{\mathbb{Z}_p}$	$\ell_{\mathrm{DEM}} + 2\ell_{\mathbb{G}}$	$2\ell_{\mathbb{G}} + \ell_{\mathbb{G}_T}$	$\{0,1\}^*$	DBDH, CR, DEM
Ours (Fig. 4)	$4\ell_{\mathbb{G}} + \ell_{\mathbb{G}_T} + \ell_{hk}$	$\ell_{\mathbb{G}} + 3\ell_{\mathbb{Z}_p}$	$\ell_{\mathrm{DEM}} + 2\ell_{\mathbb{G}} + \ell_{\mathbb{Z}_p}$	$2\ell_{\mathbb{G}} + \ell_{\mathbb{G}_T}$	$\{0,1\}^*$	DBDH, CR, DEM
Ours (Fig. 5)	$3\ell_{\mathbb{G}_{\mathrm{ddh}}}$	$\ell_{\mathbb{Z}_p}$	$\ell_{\mathrm{DEM}} + 2\ell_{\mathbb{G}_{\mathrm{ddh}}} + 2\ell_{\mathbb{Z}_p}$	$2\ell_{\mathbb{Z}_p} + \ell_{\mathbb{G}_{\mathrm{ddh}}}$	$\{0,1\}^*$	DDH, RO, DEM
Ours (Fig. 6)	$5\ell_{\mathbb{G}}$	$4\ell_{\mathbb{Z}_p}$	$\ell_{\mathrm{DEM}} + \ell_{vk} + 4\ell_{\mathbb{G}} + \ell_{\sigma}$	$3\ell_{\mathbb{G}}$	$\{0,1\}^*$	DLIN, One-time sig., DEM

n: The length of an output of the collision-resistant hash function.
$\ell_{\mathbb{G}}, \ell_{\mathbb{G}_T}$: The lengths of a $\mathbb{G}$ element and a $\mathbb{G}_T$ element, where there is a bilinear map $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$
ℓ_{hk}: The length of hashing key
$\ell_{\mathbb{Z}_p}$: The length of a $\mathbb{Z}_p$ element
ℓ_{DEM}: The length of a ciphertext of the DEM scheme
$\ell_{\mathbb{G}_{\mathrm{ddh}}}$: The length of a $\mathbb{G}_{\mathrm{ddh}}$ element, where the decisional Diffie-Hellman assumption holds
ℓ_{H_1}: The length of an output of the hash function H_1
ℓ_{vk}, ℓ_{σ}: The lengths of a verification key and a signature of the one-time signature scheme
DBDH: The decisional bilinear Diffie-Hellman assumption
CR: Collision-resistant hash function family
DEM.: Passively secure DEM scheme
RO: The random oracle model
DLIN: Decisional linear assumption
One-time Sig.: One-time signatures

various cryptographic primitives on top of it, which is discussed by Galindo et al. [6] and Sakai et al. [10].

6. Discussion

6.1 Explaining Existing Schemes

When instantiating our Tag-KEM/DEM framework by combining the construction in Fig. 3 with an appropriate DEM scheme, we can obtain Galindo's PKENO construction. Similarly, we can obtain Lai et al.'s PKENO scheme and Galindo et al.'s two PKENO constructions, by combining our Tag-KEMNO schemes with an appropriate DEM scheme. These constructions explain the idea behind the existing schemes, in particular, how they prevent Galindo's attack. We remind the reader that in Galindo's attack the

adversary replaces the DEM part of the challenge ciphertext and queries this modified ciphertext to the proof oracle. In this way the adversary obtains the encapsulated session key behind the challenge ciphertext, and breaks the confidentiality of the challenge ciphertext. In our instantiations this malleability of the ciphertext is prevented by binding the DEM part to the KEM part. Namely, this scheme uses the DEM ciphertext as the tag of the KEM ciphertext to resist Galindo's attack.

6.2 Expanding the Plaintext Space

If we want to expand the plaintext space to support arbitrary-length plaintexts, we can combine our Tag-KEMNO scheme with a DEM which supports arbitrary-length plaintexts. We also note that in this way we obtain a *generic* method for obtaining a PKENO scheme which can encrypt arbitrary-length plaintexts by following our Tag-KEMNO/DEM framework. We summarize the four existing PKENO schemes and our instantiations combined with a DEM scheme with arbitrary-length plaintexts in Table 1. The table shows that our instantiations, which supports arbitrary-length plaintexts, inherit its own merit from the original schemes while expanding the supported plaintext space.

7. Conclusion

In this paper, we formalized an extension of the Tag-KEM framework for PKENO schemes. It includes the formalization of a new cryptographic primitive named Tag-KEMNO, and the secure composition theorem of Tag-KEMNO and a DEM scheme to form a secure PKENO scheme. We then showed that four instantiations of our Tag-KEMNO primitives, based on some known constructions of PKENO. Finally we discussed that our Tag-KEMNO framework explains previous constructions and also is used to expand the plaintext space of a PKENO scheme. As remarked in the introduction, our framework does not capture the two important schemes by Dachman-Soled et al. [8]. To extend the framework to capture these two schemes is an important future direction.

Using our schemes, for instance, we can securely reduce the trust put on a cloud server in various situations, for example, an encrypted file-sharing system mentioned in Sect. 1. Furthermore, the existing schemes can only encrypt a short and fixed-length plaintext, while our instantiations can encrypt a long, arbitrary-length, plaintext. Such a long plaintext is mandatory for cloud applications, as they are particularly useful when the shared files have large sizes. These attractive features enable more flexible and reliable information sharing in a cloud environment.

Acknowledgments

A part of this work is supported by JST CREST grant number JPMJCR1688.

References

[1] Y. Sakai, T. Matsuda, and G. Hanaoka, "Tag-KEM/DEM framework for public-key encryption with non-interactive opening," Proc. 2016 International Symposium on Information Theory and Its Application, pp.231–235, IEEE, 2016.
[2] I. Damgård and R. Thorbek, "Non-interactive proofs for integer multiplication," EUROCRYPT 2007, ed. M. Naor, LNCS, vol.4515, pp.412–429, Springer Berlin Heidelberg, 2007.
[3] I. Damgård, D. Hofheinz, E. Kiltz, and R. Thorbek, "Public-key encryption with non-interactive opening," CT-RSA 2008, LNCS, vol.4964, pp.239–255, Springer, 2008.
[4] D. Galindo, "Breaking and repairing Damgård et al. public key encryption scheme with non-interactive opening," CT-RSA 2009, LNCS, vol.5473, pp.389–398, Springer, 2009.
[5] J. Lai, R.H. Deng, S. Liu, and W. Kou, "Efficient CCA-secure PKE from identity-based techniques," CT-RSA 2010, LNCS, vol.5985, pp.132–147, Springer, 2010.
[6] D. Galindo, B. Libert, M. Fischlin, G. Fuchsbauer, A. Lehmann, M. Manulis, and D. Schröder, "Public-key encryption with non-interactive opening: New constructions and stronger definitions," AFRICACRYPT 2010, LNCS, vol.6055, pp.333–350, Springer, 2010.
[7] M. Abe, R. Gennaro, and K. Kurosawa, "Tag-KEM/DEM: A new framework for hybrid encryption," Journal of Cryptology, vol.21, no.1, pp.97–130, 2008.
[8] D. Dachman-Soled, G. Fuchsbauer, P. Mohassel, and A. O'Neill, "Enhanced chosen-ciphertext security and applications," PKC 2014, LNCS, vol.8383, pp.329–344, Springer, 2014.
[9] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo, "Structure-preserving signatures and commitments to group elements," Journal of Cryptology, vol.29, no.2, pp.363–421, 2016.
[10] Y. Sakai, J.C.N. Schuldt, K. Emura, G. Hanaoka, and K. Ohta, "On the security of dynamic group signatures: Preventing signature hijacking," PKC 2012, eds. M. Fischlin, J.A. Buchmann, and M. Manulis, LNCS, vol.7293, pp.715–732, Springer, 2012.

Yusuke Sakai received his B.E., M.E., and Ph.D. degrees from the University of Electro-Communications, Tokyo, Japan, in 2009, 2011, and 2014, respectively. From 2012 to 2014, and from 2014 to 2017, he had been a research fellow of Japan Society for the Promotion of Science (JSPS). In 2017, he joined National Institute of Advanced Industrial Science and Technology (AIST), Japan. He is presently engaged in research on cryptography and information security, particularly designing secure and practical public-key cryptosystems. He received SCIS Paper Prize from IEICE in 2011 and the Best Student Award in IWSEC 2010.

Takahiro Matsuda received his bachelors, masters, and Ph.D. degrees in Information and Communication Engineering from the University of Tokyo in 2006, 2008, and 2011, respectively. From 2009 to 2011 and from 2011 to 2013, he had been a Research Fellow of Japan Society for the Promotion of Science (JSPS). From 2011, he has been with the National Institute of Advanced Industrial Science and Technology (AIST), Japan. His research interests are in the areas of public key cryptography and theory of cryptography.

Goichiro Hanaoka graduated from the Department of Engineering, the University of Tokyo in 1997. He received the Ph.D. degree from the University of Tokyo in 2002. He joined AIST in 2005. Currently, he is Leader of the Advanced Cryptosystems Research Group, Information Technology Research Institute, AIST. He engages in the R&D for encryption and information security technologies including the efficient design and security evaluation of public key cryptosystems. He has received numerous awards including the DoCoMo Mobile Science Award (2016), Mobile Communication Fund; the Wilkes Award (2007), British Computer Society; Best Paper Award (2008), The Institute of Electronics, Information and Communication Engineers (IEICE); and Innovative Paper Awards (2012, 2014), Symposium on Cryptography & Information Security (SCIS), IEICE.