A Formal Treatment of Envelope Encryption

Shoichi Hirose[1] and Kazuhiko Minematsu[2,3]

[1] University of Fukui, Fukui, Japan
hrs_shch@u-fukui.ac.jp
[2] NEC, Tokyo, Japan
k-minematsu@nec.com
[3] Yokohama National University, Kanagawa, Japan

Abstract. Envelope encryption is a method to encrypt data with two distinct keys in its basic form. Data is first encrypted with a data-encryption key, and then the data-encryption key is encrypted with a key-encryption key. Despite its deployment in major cloud services, as far as we know, envelope encryption has not received any formal treatment. To address this issue, we first formalize the syntax and security requirements of envelope encryption in the symmetric-key setting. Then, we show that it can be constructed by combining encryptment and authenticated encryption with associated data (AEAD). Encryptment is one-time AEAD satisfying that a small part of a ciphertext works as a commitment to the corresponding secret key, message, and associated data. Finally, we show that the security of the generic construction is reduced to the security of the underlying encryptment and AEAD.

Keywords: Authenticated encryption · Key wrap · Key-committing · Encryptment

1 Introduction

1.1 Background

Envelope encryption involves encrypting data using a symmetric encryption scheme with a secret key known as a data-encryption key. This key is then encrypted with one or more key-encryption keys. Envelope encryption allows a recipient, holding a secret key corresponding to one of the key-encryption keys, to recover the data-encryption key by using the secret key and to decrypt the encrypted data with the data-encryption key. Envelope encryption is deployed in primary cloud services such as Amazon Web Services, Google Cloud Platform, IBM Cloud, and Microsoft Azure.

Envelope encryption usually employs authenticated encryption with associated data (AEAD) [20] to encrypt data. AEAD is an established symmetric-key cryptographic primitive providing privacy and authenticity. Albertini et al. [1] noted that the underlying AEAD must be key-committing. Namely, it should ensure that only the secret key used to create a ciphertext can decrypt it. For non-key-committing AEAD, a ciphertext C may exist which can be decrypted

into distinct data P_1 and P_2 by two secret keys K_1 and K_2, respectively. Then, a recipient receiving C together with K_1 recovers P_1, and another recipient receiving C together with K_2 recovers P_2.

Meanwhile, as far as we know, no formal treatment is given to envelope encryption. It is left open what is *secure* envelope encryption and how we can achieve it.

1.2 Our Contributions

Envelope encryption is effective for multicast from a server to multiple clients, where each client shares a secret key with the server. The server encrypts data with an ephemeral data-encryption key and the ephemeral data-encryption key with the secret keys of recipients among the clients. In some use case, it may be sufficient to guarantee the secrecy of data for a certain period, and the data-encryption key may be disclosed after the period. Even in such a case, ciphertext unforgeability is required. Namely, a malicious adversary should not be able to utilize the disclosed ephemeral data-encryption key and forge a ciphertext.

We first formalize the syntax and security requirements of envelope encryption in the symmetric-key setting. The formalized security requirements are confidentiality, ciphertext integrity, and soundness. The confidentiality and ciphertext integrity are inherited from AEAD. The ciphertext integrity covers the use case for multicast described in the previous paragraph. The soundness captures the notion that all the recipients allowed to decrypt the same encrypted data should obtain the same recovered data. Then, we present a generic construction for envelope encryption combining encryptment [5] and AEAD. Finally, we prove the security of the generic construction in a multi-user setting. The confidentiality is reduced to the confidentiality of encryptment and AEAD. The ciphertext integrity is reduced to the second ciphertext unforgeability of encryptment and the ciphertext integrity of AEAD. The generic construction utilizes the compactly committing characteristic of encryptment for ciphertext integrity. The soundness is reduced to the strong receiver-binding property [5] of encryptment.

1.3 Related Work

Formal treatments of authenticated encryption were initiated by Katz and Yung [12] and Bellare and Namprempre [3]. The first dedicated schemes were presented by Jutla [11]. AEAD was first formalized and investigated by Rogaway [20].

Committing AEAD and its variations were explored by Farshim et al. [7], Len et al. [15], Bellare and Hoang [2], and Chan and Rogaway [4]. For committing AEAD, a whole ciphertext serves as a commitment in general.

Commonly used AEAD schemes are not key-committing. Dodis et al. [5] presented a practical attack showing that AES-GCM [18] is not key-committing. Then, Albertini et al. developed an extensive attack and applied it effectively on AES-GCM, ChaCha20-Poly1305 [17], AES-GCM-SIV [9], and OCB3 [13,14].

Compactly committing AEAD (ccAEAD) was introduced by Grubbs et al. [8], which is useful for message franking [6]. For ccAEAD, a small part of the ciphertext serves as a commitment to the secret key, message, and associated data. Encryptment was introduced by Dodis et al. [5] as a core component of ccAEAD. They showed that encryptment with one call to AEAD or two calls to PRF produces ccAEAD. Hirose and Minematsu [10] showed that encryptment with one call to a tweakable block cipher [16] produces ccAEAD.

1.4 Organization

Section 2 introduces AEAD and encryptment. Section 3 formalizes the syntax and security requirements of envelope encryption. Section 4 presents the generic construction of envelope encryption combining AEAD and encryptment and proves its security. Section 5 gives a concluding remark.

2 Preliminaries

Let $\Sigma := \{0,1\}$ and $\Sigma^* := \bigcup_{i \geq 0} \Sigma^i$. For a set $\mathcal{S}$, let $s \twoheadleftarrow \mathcal{S}$ represent that s is assigned an element chosen uniformly at random from $\mathcal{S}$.

2.1 Authenticated Encryption with Associated Data

Authenticated encryption with associated data (AEAD) [20] is a primitive of symmetric-key cryptography providing privacy and authenticity. Here, AEAD is formalized as deterministic authenticated encryption [21].

Syntax. An AEAD scheme is a tuple of algorithms $\mathsf{AE} := (\mathsf{AEkg}, \mathsf{AEenc}, \mathsf{AEdec})$. It is associated with the following subsets of Σ^*: a key space $\mathcal{K}_{\mathrm{AE}}$, an associated-data space $\mathcal{A}_{\mathrm{AE}}$, a message space $\mathcal{M}_{\mathrm{AE}}$, and a ciphertext space $\mathcal{C}_{\mathrm{AE}}$. It also has a targeted security level, which determines the key length and affects the length of each ciphertext. Let $\mathcal{K}_{\mathrm{AE}} := \Sigma^{\ell_{\mathrm{AE}}}$.

- The key-generation algorithm AEkg simply returns a secret key $K \twoheadleftarrow \mathcal{K}_{\mathrm{AE}}$.
- The encryption algorithm is a function such that $\mathsf{AEenc} : \mathcal{K}_{\mathrm{AE}} \times \mathcal{A}_{\mathrm{AE}} \times \mathcal{M}_{\mathrm{AE}} \to \mathcal{C}_{\mathrm{AE}}$. For any $(K, A, M) \in \mathcal{K}_{\mathrm{AE}} \times \mathcal{A}_{\mathrm{AE}} \times \mathcal{M}_{\mathrm{AE}}$, if $C \leftarrow \mathsf{AEenc}(K, A, M)$, then $|M|$ determines $|C|$, and it is assumed that there exists some function $\mathsf{clen}_{\mathrm{AE}} : \mathbb{N} \to \mathbb{N}$ such that $|C| = \mathsf{clen}_{\mathrm{AE}}(|M|)$.
- The decryption algorithm is a function such that $\mathsf{AEdec} : \mathcal{K}_{\mathrm{AE}} \times \mathcal{A}_{\mathrm{AE}} \times \mathcal{C}_{\mathrm{AE}} \to \mathcal{M}_{\mathrm{AE}} \cup \{\perp\}$, where $\perp \notin \mathcal{M}_{\mathrm{AE}}$.

It is assumed that AE satisfies correctness: For any $(K, A, M) \in \mathcal{K}_{\mathrm{AE}} \times \mathcal{A}_{\mathrm{AE}} \times \mathcal{M}_{\mathrm{AE}}$, $\mathsf{AEdec}(K, A, \mathsf{AEenc}(K, A, M)) = M$. For any $(K, A, C) \in \mathcal{K}_{\mathrm{AE}} \times \mathcal{A}_{\mathrm{AE}} \times \mathcal{C}_{\mathrm{AE}}$, (A, C) is invalid with respect to K if $\mathsf{AEdec}(K, A, C) = \perp$.

Security Requirements. The security requirements of AEAD are confidentiality and ciphertext integrity. Here, they are formalized in a multi-user setting.

Confidentiality. The two games MU-REAL and MU-RAND in Fig. 1 are introduced to formalize the confidentiality as real-or-random indistinguishability. In both of the games, an adversary $\mathbf{A}$ is given oracles **AEnew** and **AEenc**. For a new call, **AEnew** generates a secret key of a new user u. In response to a valid query, **AEenc** returns a ciphertext produced by AEenc in MU-REAL and a uniform random sequence in MU-RAND. $\mathbf{A}$ is not allowed to repeat the same query to **AEenc**. Finally, $\mathbf{A}$ outputs 0 or 1. The advantage of $\mathbf{A}$ for the confidentiality is

$$\mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE}}(\mathbf{A}) := \left|\Pr[\text{MU-REAL}^{\mathbf{A}}_{\mathsf{AE}} = 1] - \Pr[\text{MU-RAND}^{\mathbf{A}}_{\mathsf{AE}} = 1]\right|.$$

$u \leftarrow 0$
$b \leftarrow \mathbf{A}^{\mathbf{AEnew},\mathbf{AEenc}}$
return b

AEnew()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}_{\mathrm{AE}}$

AEenc(i, A, M)
if $i \notin [1, u]$ **then**
  return $\perp$
end if
$C \leftarrow \mathsf{AEenc}(K_i, A, M)$
return C

(a) $\text{MU-REAL}^{\mathbf{A}}_{\mathsf{AE}}$

$u \leftarrow 0$
$b \leftarrow \mathbf{A}^{\mathbf{AEnew},\mathbf{AEenc}}$
return b

AEnew()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}_{\mathrm{AE}}$

AEenc(i, A, M)
if $i \notin [1, u]$ **then**
  return $\perp$
end if
$C \twoheadleftarrow \Sigma^{\mathsf{clen}_{\mathrm{AE}}(|M|)}$
return C

(b) $\text{MU-RAND}^{\mathbf{A}}_{\mathsf{AE}}$

Fig. 1: Games for confidentiality of AEAD

Ciphertext Integrity. The game MU-CTXT in Fig. 2 is introduced to formalize the ciphertext integrity as existential unforgeability. In this game, an adversary $\mathbf{A}$ is given oracles **AEnew**, **AEcorrupt**, **AEenc**, and **AEdec**. The oracles **AEnew** and **AEenc** are equivalent to those of MU-REAL except that **AEenc** stores all of its outputs. **AEcorrupt** returns the secret key of a user specified by $\mathbf{A}$. For a query, **AEdec** runs AEdec and sets *win* `true` if and only if $\mathbf{A}$ succeeds in forgery. The advantage of $\mathbf{A}$ for the ciphertext integrity is

$$\mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{AE}}(\mathbf{A}) := \Pr[\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{AE}} = \mathtt{true}].$$

Remark 1. For confidentiality, $\mathbf{A}$ is not allowed to corrupt users. Otherwise, $\mathbf{A}$ can trivially distinguish MU-REAL and MU-RAND.

2.2 Encryptment

Syntax. Encryptment [5] is formalized as a tuple of algorithms $\mathsf{EC} := (\mathsf{ECkg}, \mathsf{ECenc}, \mathsf{ECdec}, \mathsf{ECver})$. It is associated with the following subsets of Σ^*: a key

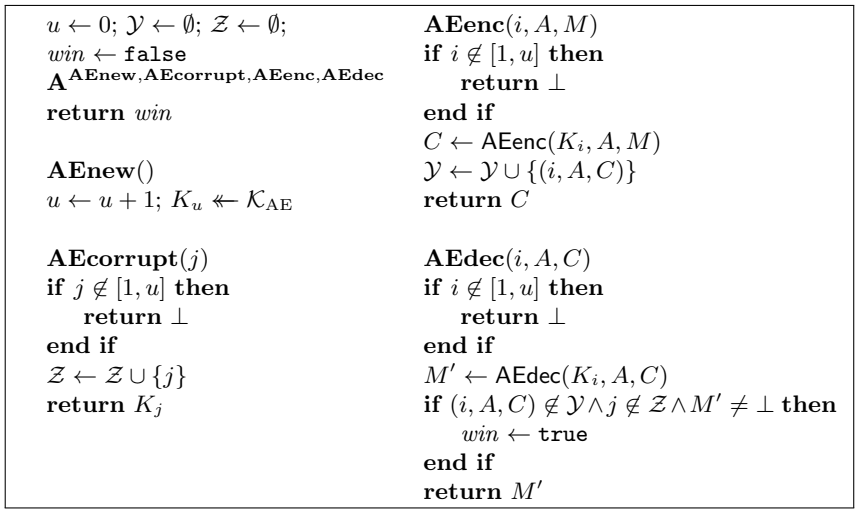


Fig. 2: Game MU-CTXT$^{\mathbf{A}}_{\mathsf{AE}}$ for ciphertext integrity of AEAD

space $\mathcal{K}_{\mathrm{EC}}$, an associated-data space $\mathcal{A}_{\mathrm{EC}}$, a message space $\mathcal{M}_{\mathrm{EC}}$, a ciphertext space $\mathcal{C}_{\mathrm{EC}}$, and a binding-tag space $\mathcal{T}_{\mathrm{EC}}$. It also has a targeted security level, which determines the key length and the binding-tag length. Let $\mathcal{K}_{\mathrm{EC}} := \Sigma^{\ell}$ and $\mathcal{T}_{\mathrm{EC}} := \Sigma^{\tau}$.

- The key-generation algorithm ECkg simply returns a secret key $K \twoheadleftarrow \mathcal{K}_{\mathrm{EC}}$.
- The encryptment algorithm is a function such that $\mathsf{ECenc} : \mathcal{K}_{\mathrm{EC}} \times \mathcal{A}_{\mathrm{EC}} \times \mathcal{M}_{\mathrm{EC}} \to \mathcal{C}_{\mathrm{EC}} \times \mathcal{T}_{\mathrm{EC}}$. For any $(K, A, M) \in \mathcal{K}_{\mathrm{EC}} \times \mathcal{A}_{\mathrm{EC}} \times \mathcal{M}_{\mathrm{EC}}$, if $(C, B) \leftarrow \mathsf{ECenc}(K, A, M)$, then $|M|$ determines $|C|$, and it is assumed that there exists some function $\mathsf{clen}_{\mathrm{EC}} : \mathbb{N} \to \mathbb{N}$ such that $|C| = \mathsf{clen}_{\mathrm{EC}}(|M|)$.
- The decryptment algorithm is a function such that $\mathsf{ECdec} : \mathcal{K}_{\mathrm{EC}} \times \mathcal{A}_{\mathrm{EC}} \times \mathcal{C}_{\mathrm{EC}} \times \mathcal{T}_{\mathrm{EC}} \to \mathcal{M}_{\mathrm{EC}} \cup \{\perp\}$, where $\perp \notin \mathcal{M}_{\mathrm{EC}}$.
- The verification algorithm is a function such that $\mathsf{ECver} : \mathcal{A}_{\mathrm{EC}} \times \mathcal{M}_{\mathrm{EC}} \times \mathcal{K}_{\mathrm{EC}} \times \mathcal{T}_{\mathrm{EC}} \to \Sigma$.

It is assumed that EC satisfies correctness. Namely, for any $(K, A, M) \in \mathcal{K}_{\mathrm{EC}} \times \mathcal{A}_{\mathrm{EC}} \times \mathcal{M}_{\mathrm{EC}}$, if $(C, B) \leftarrow \mathsf{ECenc}(K, A, M)$, then $\mathsf{ECdec}(K, A, C, B) = M$ and $\mathsf{ECver}(A, M, K, B) = 1$. A stronger notion of correctness called strong correctness additionally requires that, for any $(K, A, C, B) \in \mathcal{K}_{\mathrm{EC}} \times \mathcal{A}_{\mathrm{EC}} \times \mathcal{C}_{\mathrm{EC}} \times \mathcal{T}_{\mathrm{EC}}$, if $M \leftarrow \mathsf{ECdec}(K, A, C, B)$, then $\mathsf{ECenc}(K, A, M) = (C, B)$.

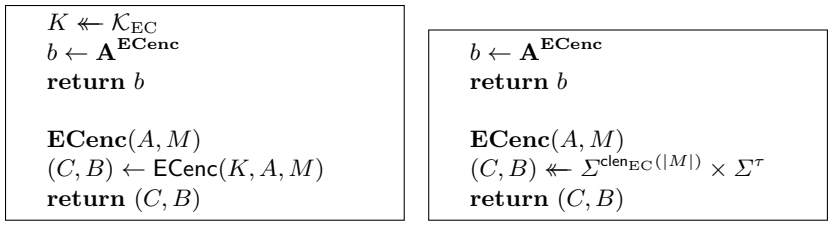
Security Requirements. The security requirements of encryptment are confidentiality, second-ciphertext unforgeability, and binding properties.

Confidentiality. Two games otREAL and otRAND shown in Fig. 3 are introduced to formalize the confidentiality. In both of the games, an adversary $\mathbf{A}$

is allowed to ask a single query to the oracle **ECenc**. The advantage of $\mathbf{A}$ for confidentiality is

$$\mathrm{Adv}^{\text{ot-ror}}_{\mathsf{EC}}(\mathbf{A}) := \left|\Pr[\mathrm{otREAL}^{\mathbf{A}}_{\mathsf{EC}} = 1] - \Pr[\mathrm{otRAND}^{\mathbf{A}}_{\mathsf{EC}} = 1]\right|,$$

where "ot-ror" stands for "one-time real-or-random."



(a) $\mathrm{otREAL}^{\mathbf{A}}_{\mathsf{EC}}$ (b) $\mathrm{otRAND}^{\mathbf{A}}_{\mathsf{EC}}$

Fig. 3: Games for confidentiality of encryptment

Second-Ciphertext Unforgeability. An adversary $\mathbf{A}$ is allowed to ask a single query $(A, M) \in \mathcal{A}_{\mathrm{EC}} \times \mathcal{M}_{\mathrm{EC}}$ to $\mathsf{ECenc}(K, \cdot, \cdot)$ and gets (C, B) and K, where $K \twoheadleftarrow \mathcal{K}_{\mathrm{EC}}$ and $(C, B) \leftarrow \mathsf{ECenc}(K, A, M)$. Finally, $\mathbf{A}$ outputs $(A', C') \in \mathcal{A}_{\mathrm{EC}} \times \mathcal{C}_{\mathrm{EC}}$. The advantage of $\mathbf{A}$ for second-ciphertext unforgeability is

$$\mathrm{Adv}^{\mathrm{scu}}_{\mathsf{EC}}(\mathbf{A}) := \Pr[(A, C) \neq (A', C') \wedge \mathsf{ECdec}(K, A', C', B) \neq \perp].$$

Binding properties. Binding properties are defined in terms of a receiver and a sender. Receiver binding ensures that any malicious receiver cannot blame a non-abusive sender for sending an abusive message. Let $\mathbf{A}$ be an adversary producing a pair of elements in $\mathcal{K}_{\mathrm{EC}} \times \mathcal{A}_{\mathrm{EC}} \times \mathcal{M}_{\mathrm{EC}}$ and a binding tag in $\mathcal{T}_{\mathrm{EC}}$. The advantage of $\mathbf{A}$ for receiver binding is

$$\begin{aligned}\mathrm{Adv}^{\text{r-bind}}_{\mathsf{EC}}(\mathbf{A}) := \Pr[((K, A, M), (K', A', M'), B) \leftarrow \mathbf{A} : (A, M) \neq (A', M') \\ \wedge \mathsf{ECver}(A, M, K, B) = \mathsf{ECver}(A', M', K', B) = 1].\end{aligned}$$

The advantage of $\mathbf{A}$ for strong receiver binding is

$$\begin{aligned}\mathrm{Adv}^{\text{sr-bind}}_{\mathsf{EC}}(\mathbf{A}) := \Pr[((K, A, M), (K', A', M'), B) \leftarrow \mathbf{A} : (K, A, M) \neq (K', A', M') \\ \wedge \mathsf{ECver}(A, M, K, B) = \mathsf{ECver}(A', M', K', B) = 1].\end{aligned}$$

Sender binding ensures that any malicious sender sending an abusive message cannot prevent the receiver from blaming it. The advantage of an adversary $\mathbf{A}$ for sender binding is

$$\begin{aligned}\mathrm{Adv}^{\text{s-bind}}_{\mathsf{EC}}(\mathbf{A}) := \Pr[(K, A, C, B) \leftarrow \mathbf{A}, M \leftarrow \mathsf{ECdec}(K, A, C, B) : \\ M \neq \perp \ \wedge \ \mathsf{ECver}(A, M, K, B) = 0].\end{aligned}$$

3 Envelope Encryption

The syntax and security requirements of envelope encryption are formalized in the symmetric key setting. Thus, it is assumed that symmetric encryption is used for encrypting data-encryption keys as well as data. It is essential that the security requirements are formalized in the multi-user setting since envelope encryption generally assumes that a data-encryption key is encrypted with multiple key-encryption keys.

3.1 Syntax

A tuple of algorithms $\mathsf{EE} := (\mathsf{KEKGen}, \mathsf{DEKGen}, \mathsf{Enc}, \mathsf{Wrap}, \mathsf{Dec})$ specify envelope encryption. It is associated with the following subsets of Σ^*: a key-encryption-key space $\mathcal{K}$, a data-encryption-key space $\mathcal{L}$, an associated-data space $\mathcal{A}$, a message space $\mathcal{M}$, a ciphertext space $\mathcal{C}$, a binding-tag space $\mathcal{T}$, a header space $\mathcal{H}$, and a wrapped-data-encryption-key space $\mathcal{S}$. It also has a targeted security level, which determines the key length, the binding-tag length, and the wrapped-data-encryption-key length.

- The key-encryption-key generation algorithm KEKGen returns a secret key for key encryption chosen uniformly at random from $\mathcal{K}$.
- The data-encryption-key generation algorithm DEKGen returns a secret key for data encryption chosen uniformly at random from $\mathcal{L}$.
- The data-encryption algorithm is a function such that $\mathsf{Enc} : \mathcal{L} \times \mathcal{A} \times \mathcal{M} \to \mathcal{C} \times \mathcal{T}$. For any $(L, A, M) \in \mathcal{L} \times \mathcal{A} \times \mathcal{M}$, if $(C, B) \leftarrow \mathsf{Enc}(L, A, M)$, then $|M|$ determines $|C|$, and it is assumed that there exists a function $\mathsf{clen} : \mathbb{N} \to \mathbb{N}$ such that $|C| = \mathsf{clen}(|M|)$.
- The key-wrap algorithm[1] is a function such that $\mathsf{Wrap} : \mathcal{K} \times \mathcal{T} \times \mathcal{L} \times \mathcal{H} \to \mathcal{S}$.
- The decryption algorithm is a function such that $\mathsf{Dec} : \mathcal{K} \times \mathcal{A} \times \mathcal{C} \times \mathcal{T} \times \mathcal{S} \times \mathcal{H} \to \mathcal{M} \times \mathcal{L} \cup \{\perp\}$, where $\perp \notin \mathcal{M} \times \mathcal{L}$.

It is assumed that EE satisfies correctness: For any $(K, L, A, M, H) \in \mathcal{K} \times \mathcal{L} \times \mathcal{A} \times \mathcal{M} \times \mathcal{H}$, if $(C, B) \leftarrow \mathsf{Enc}(L, A, M)$ and $S \leftarrow \mathsf{Wrap}(K, B, L, H)$, then $\mathsf{Dec}(K, A, C, B, S, H) = M$. For any $(K, A, C, B, S, H) \in \mathcal{K} \times \mathcal{A} \times \mathcal{C} \times \mathcal{T} \times \mathcal{S} \times \mathcal{H}$, (A, C, B, S, H) is invalid with respect to K if $\mathsf{Dec}(K, A, C, B, S, H) = \perp$.

The formalization above specifies the data-encryption algorithm and the key-wrap algorithm separately, and both of them have their associated data, where the latter is called a header. It allows incremental key wrapping. In a cloud application serving encrypted data, for example, a server can allow a new recipient to recover the data by wrapping the data-encryption key with the key-encryption key of the recipient on demand.

3.2 Security Requirements

The security requirements of envelope encryption are confidentiality, ciphertext integrity, and soundness, which are formalized in a multi-user setting.

[1] We do not expect that the key-wrap algorithm specified here is confused with the methods in NIST SP 800-38F [19].

Confidentiality. The two games MU-MO-REAL and MU-MO-RAND in Fig. 4 are introduced to formalize the confidentiality as real-or-random indistinguishability in a multi-opening setting. In both of the games, an adversary $\mathbf{A}$ is given oracles **New**, **Enc**, **Wrap**, **ChalEnc**, and **Dec**. For a new call, **New** generates a secret key-encryption key of a new user u. In response to a query, **Enc** returns a ciphertext produced by Enc. In response to a valid query, **Wrap** returns a wrapped data-encryption key produced by Wrap in MU-MO-REAL and a uniform random sequence sampled from $\mathcal{S}$ in MU-MO-RAND. In response to a query, **Dec** returns a pair of a message and a data-encryption key if the query is a ciphertext produced by **Enc** and **Wrap**. **ChalEnc** returns a ciphertext produced by Enc in MU-MO-REAL and a uniform random sequence in MU-MO-RAND. Finally, $\mathbf{A}$ outputs 0 or 1. The advantage of $\mathbf{A}$ for the confidentiality is

$$\mathrm{Adv}^{\text{mu-mo-ror}}_{\mathsf{EE}}(\mathbf{A}) := \left|\Pr[\text{MU-MO-REAL}^{\mathbf{A}}_{\mathsf{EE}} = 1] - \Pr[\text{MU-MO-RAND}^{\mathbf{A}}_{\mathsf{EE}} = 1]\right|.$$

In this formalization, $\mathbf{A}$ is not allowed to corrupt users. Otherwise, $\mathbf{A}$ can trivially distinguish MU-MO-REAL and MU-MO-RAND.

Ciphertext Integrity. The game MU-MO-CTXT in Fig. 5 is introduced to formalize the ciphertext integrity as existential unforgeability. In this game, an adversary $\mathbf{A}$ is given oracles **New**, **Corrupt**, **Enc**, **Wrap**, and **Dec**. The oracles **New**, **Enc**, and **Wrap** are equivalent to those of MU-MO-REAL. **Corrupt** returns the secret key-encryption key of a user specified by $\mathbf{A}$. For a query, **Dec** runs Dec and sets *win* `true` if and only if $\mathbf{A}$ succeeds in forgery. The advantage of $\mathbf{A}$ for the ciphertext integrity is

$$\mathrm{Adv}^{\text{mu-mo-ctxt}}_{\mathsf{EE}}(\mathbf{A}) := \Pr[\text{MU-MO-CTXT}^{\mathbf{A}}_{\mathsf{EE}} = \mathtt{true}].$$

Soundness. The soundness captures the notion that users should recover the same message from a ciphertext produced by Enc. The advantage of an adversary $\mathbf{A}$ for soundness is

$$\begin{aligned}\mathrm{Adv}^{\text{snd}}_{\mathsf{EE}}(\mathbf{A}) := \Pr[((K, A, C, B, S, H), (K', A', C', B', S', H')) \leftarrow \mathbf{A} :\\ (A, C, B) = (A', C', B') \wedge\\ \mathsf{Dec}(K, A, C, B, S, H) \neq \perp \wedge \mathsf{Dec}(K', A', C', B', S', H') \neq \perp \wedge\\ \mathsf{Dec}(K, A, C, B, S, H) \neq \mathsf{Dec}(K', A', C', B', S', H')].\end{aligned}$$

Since Dec is deterministic, $(A, C, B) = (A', C', B')$ and $\mathsf{Dec}(K, A, C, B, S, H) \neq \mathsf{Dec}(K', A', C', B', S', H')$ imply that $(K, S, H) \neq (K', S', H')$.

4 Generic Construction of Envelope Encryption

$u \leftarrow 0$; $d \leftarrow 0$; $\mathcal{X} \leftarrow \emptyset$; $\mathcal{Y} \leftarrow \emptyset$
$b \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{ChalEnc},\mathbf{Dec}}$
return b

New()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}$

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{Enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return (C_d, B_d)

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \leftarrow \mathsf{Wrap}(K_i, B_j, L_j, H_{i,j})$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, B_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{Enc}(L_d, A, M)$
return (C_d, B_d)

Dec(i, A, C, B, S, H)
if $(i, A, C, B, S, H) \notin \mathcal{Y}$ **then**
  return $\perp$
end if
return $\mathsf{Dec}(K_i, A, C, B, S, H)$

(a) MU-MO-REAL$^{\mathbf{A}}_{\mathsf{EE}}$

$u \leftarrow 0$; $d \leftarrow 0$; $\mathcal{X} \leftarrow \emptyset$; $\mathcal{Y} \leftarrow \emptyset$
$b \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{ChalEnc},\mathbf{Dec}}$
return b

New()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}$

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{Enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return (C_d, B_d)

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{(i,j)} \twoheadleftarrow \mathcal{S}$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, B_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \twoheadleftarrow \Sigma^{\mathsf{clen}(|M|)} \times \mathcal{T}$
return (C_d, B_d)

Dec(i, A, C, B, S, H)
if $(i, A, C, B, S, H) \notin \mathcal{Y}$ **then**
  return $\perp$
end if
return (M_j, L_j) $\triangleright$ $(i, A, C, B, S, H) = (i, A_j, C_j, B_j, S_{i,j}, H_{i,j}) \in \mathcal{Y}$

(b) MU-MO-RAND$^{\mathbf{A}}_{\mathsf{EE}}$

Fig. 4: Games for confidentiality of envelope encryption

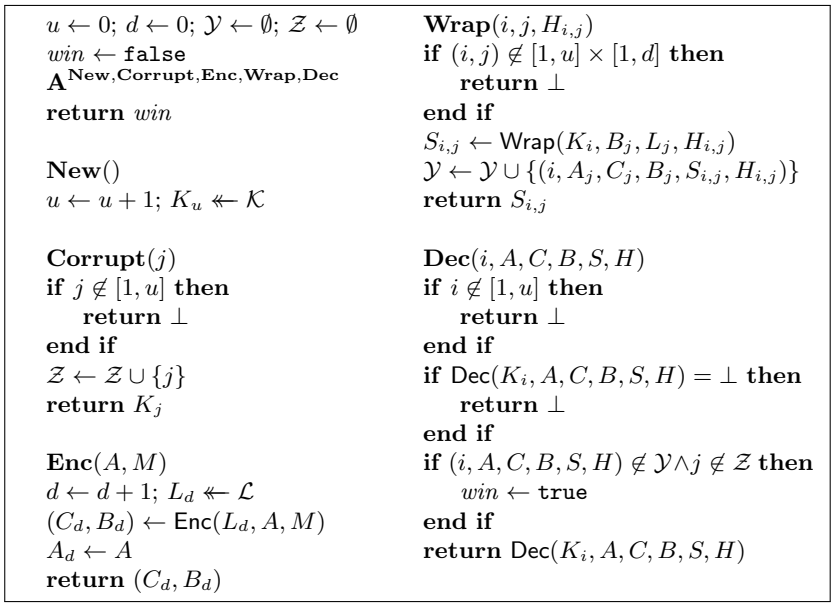


Fig. 5: Game MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{EE}}$ for ciphertext integrity of envelope encryption

4.1 Construction

We present generic construction of envelope encryption $\mathsf{GEE} := (\mathsf{GKEKGen}, \mathsf{GDEKGen}, \mathsf{GEnc}, \mathsf{GWrap}, \mathsf{GDec})$ from encryptment $\mathsf{EC} := (\mathsf{ECkg}, \mathsf{ECenc}, \mathsf{ECdec}, \mathsf{ECver})$ and AEAD $\mathsf{AE} := (\mathsf{AEkg}, \mathsf{AEenc}, \mathsf{AEdec})$. The generic construction follows the observation by Albertini et al. [1] and utilizes encryptment, which is one-time key-committing AEAD, for data encryption. For key wrap, the generic construction utilizes AEAD, which takes a binding tag of encryptment as (a part of) its associated data. This prevents an adversary from forging a ciphertext of envelope encryption by using a disclosed data-encryption key due to, for example, expiration. New data encrypted with a disclosed data-encryption key will be accompanied with a new binding tag.

Key-encryption-key generation AEkg works as GKEKGen. A secret key-encryption key is chosen from $\mathcal{K} := \mathcal{K}_{\mathrm{AE}}$ uniformly at random.

Data-encryption-key generation ECkg works as GDEKGen. A secret data-encryption key is chosen from $\mathcal{L} := \mathcal{K}_{\mathrm{EC}}$ uniformly at random.

Encryption ECenc works as GEnc. For $(L, A, M) \in \mathcal{L} \times \mathcal{A} \times \mathcal{M}$, $(C, B) \leftarrow \mathsf{ECenc}(L, A, M)$, where $\mathcal{A} \subseteq \mathcal{A}_{\mathrm{EC}}$ and $\mathcal{M} \subseteq \mathcal{M}_{\mathrm{EC}}$.

Key Wrap AEenc works as GWrap. For $(K, B, L, H) \in \mathcal{K} \times \mathcal{T} \times \mathcal{L} \times \mathcal{H}$, $S \leftarrow \mathsf{AEenc}(K, (B, H), L)$, where $\mathcal{T} \times \mathcal{H} \subseteq \mathcal{A}_{\mathrm{AE}}$ and $\mathcal{L} \subseteq \mathcal{M}_{\mathrm{AE}}$.

Decryption $\mathsf{GDec}(K, A, C, B, S, H)$ is defined as follows:

1. $L' \leftarrow \mathsf{AEdec}(K, (B, H), S)$.
2. If $L' = \perp$, then return $\perp$. Otherwise, $M' \leftarrow \mathsf{ECdec}(L', A, C, B)$.

3. If $M' = \perp$, then return $\perp$. Otherwise, return (M', L').

Remark 2. For the proposed generic construction, AE can be instantiated by nonce-based AEAD, though AE is formalized as deterministic authenticated encryption in Sect. 2. The discussions on security in the subsequent part of this section apply to such instantiations.

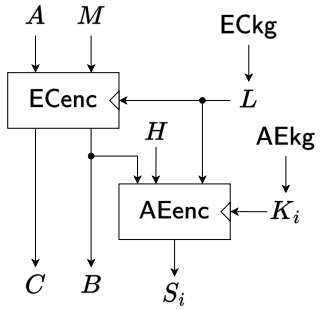


Fig. 6: Diagram of envelope encryption

4.2 Security

Confidentiality. GEE satisfies the confidentiality if the underlying AE and EC satisfy the confidentiality:

Theorem 1. *For any adversary* $\mathbf{A}$ *against* GEE *for confidentiality making* q_{w} *queries to the* **Wrap** *oracle and* q_{c} *queries to the* **ChalEnc** *oracle, there exists some adversaries* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{mu-mo-ror}}_{\mathsf{GEE}}(\mathbf{A}) \leq \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE}}(\dot{\mathbf{A}}) + q_{\mathrm{c}} \cdot \mathrm{Adv}^{\text{ot-ror}}_{\mathsf{EC}}(\ddot{\mathbf{A}}).$$

$\dot{\mathbf{A}}$ *makes at most* q_{w} *queries to the* **AEenc** *oracle. The run time of* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *is at most about that of* $\text{MU-MO-REAL}^{\mathbf{A}}_{\mathsf{GEE}}$.

Proof. For the games $\text{MU-MO-REAL}^{\mathbf{A}}_{\mathsf{GEE}}$ in Fig. 7 and $\text{MU-MO-RAND}^{\mathbf{A}}_{\mathsf{GEE}}$ in Fig. 8,

$$\mathrm{Adv}^{\text{mu-mo-ror}}_{\mathsf{GEE}}(\mathbf{A}) = \left|\Pr[\text{MU-MO-REAL}^{\mathbf{A}}_{\mathsf{GEE}} = 1] - \Pr[\text{MU-MO-RAND}^{\mathbf{A}}_{\mathsf{GEE}} = 1]\right|.$$

The game $\text{MU-MO-ROR-G}^{\mathbf{A}}_{1}$ in Fig. 9 is different from $\text{MU-MO-REAL}^{\mathbf{A}}_{\mathsf{GEE}}$ in that **Dec** selects responses for each query in $\mathcal{Y}$ based on the queries and responses of **Enc** and **Wrap**. This change is minor, and

$$\Pr[\text{MU-MO-ROR-G}^{\mathbf{A}}_{1} = 1] = \Pr[\text{MU-MO-REAL}^{\mathbf{A}}_{\mathsf{GEE}} = 1].$$

The game MU-MO-ROR-$\mathrm{G}_2^{\mathbf{A}}$ in Fig. 10 is different from MU-MO-ROR-$\mathrm{G}_1^{\mathbf{A}}$ in that **Wrap** selects $S_{i,j}$ uniformly at random. Let $\dot{\mathbf{A}}$ be an adversary against AE for confidentiality. $\dot{\mathbf{A}}$ runs MU-MO-ROR-$\mathrm{G}_1^{\mathbf{A}}$ and MU-MO-ROR-$\mathrm{G}_2^{\mathbf{A}}$ in MU-REAL$_{\mathsf{AE}}^{\dot{\mathbf{A}}}$ and MU-RAND$_{\mathsf{AE}}^{\dot{\mathbf{A}}}$, respectively. $\dot{\mathbf{A}}$ runs $\mathbf{A}$ and simulates its oracles. For each call to **New** made by $\mathbf{A}$, $\dot{\mathbf{A}}$ makes a call to **AEnew**. For a query $(i, j) \in [1, u] \times [1, d]$ to **Wrap** made by $\mathbf{A}$, $\dot{\mathbf{A}}$ makes a query $(i, (B_j, H_{i,j}), L_j)$ to **AEenc** and forwards the response to $\mathbf{A}$. Finally, $\dot{\mathbf{A}}$ produces the same output as $\mathbf{A}$. Thus,

$$\mathrm{Adv}_{\mathsf{AE}}^{\text{mu-ror}}(\dot{\mathbf{A}}) = \big|\Pr[\text{MU-MO-ROR-G}_1^{\mathbf{A}} = 1] - \Pr[\text{MU-MO-ROR-G}_2^{\mathbf{A}} = 1]\big|.$$

$\dot{\mathbf{A}}$ makes at most q_{w} queries to **AEenc**, and its run time is at most about that of MU-MO-REAL$_{\mathsf{GEE}}^{\dot{\mathbf{A}}}$.

Now, let us introduce the game MU-MO-HYB$_k^{\mathbf{A}}$ shown in Fig. 11, where $k \in [0, q_{\mathrm{c}}]$. Except **ChalEnc**, it uses the oracles of MU-MO-ROR-$\mathrm{G}_2^{\mathbf{A}}$. Thus, it is equivanet to MU-MO-ROR-$\mathrm{G}_2^{\mathbf{A}}$ and MU-MO-RAND$_{\mathsf{GEE}}^{\mathbf{A}}$ if k equals 0 and q_{c}, respectively. Let $\mathbf{A}'_l$ be an adversary against EC for confidentiality, where $l \in [1, q_{\mathrm{c}}]$. In otREAL$_{\mathsf{EC}}^{\mathbf{A}'_l}$ or otRAND$_{\mathsf{EC}}^{\mathbf{A}'_l}$, $\mathbf{A}'_l$ runs MU-MO-HYB$_{l-1}^{\mathbf{A}}$ except that $\mathbf{A}'_l$ asks the l-th query to **ChalEnc** made by $\mathbf{A}$ to **ECenc**. Finally, $\mathbf{A}'_l$ produces the same output as $\mathbf{A}$. Then, otREAL$_{\mathsf{EC}}^{\mathbf{A}'_l}$ and otRAND$_{\mathsf{EC}}^{\mathbf{A}'_l}$ are equivalent to MU-MO-HYB$_{l-1}^{\mathbf{A}}$ and MU-MO-HYB$_l^{\mathbf{A}}$, respectively. Thus,

$$\begin{aligned}
&\big|\Pr[\text{MU-MO-ROR-G}_2^{\mathbf{A}} = 1] - \Pr[\text{MU-MO-RAND}_{\mathsf{GEE}}^{\mathbf{A}} = 1]\big| \\
&\quad = \big|\Pr[\text{MU-MO-HYB}_0^{\mathbf{A}} = 1] - \Pr[\text{MU-MO-HYB}_{q_{\mathrm{c}}}^{\mathbf{A}} = 1]\big| \\
&\quad \le \sum_{l=1}^{q_{\mathrm{c}}} \big|\Pr[\text{MU-MO-HYB}_{l-1}^{\mathbf{A}} = 1] - \Pr[\text{MU-MO-HYB}_l^{\mathbf{A}} = 1]\big| \\
&\quad \le \sum_{k=1}^{q_{\mathrm{c}}} \big|\Pr[\text{otREAL}_{\mathsf{EC}}^{\mathbf{A}'_l} = 1] - \Pr[\text{otRAND}_{\mathsf{EC}}^{\mathbf{A}'_l} = 1]\big|,
\end{aligned}$$

and the run time of $\mathbf{A}'_l$ is at most about that of MU-MO-REAL$_{\mathsf{GEE}}^{\mathbf{A}}$. Thus, there exists some $\ddot{\mathbf{A}}$ such that

$$\big|\Pr[\text{MU-MO-ROR-G}_2^{\mathbf{A}} = 1] - \Pr[\text{MU-MO-RAND}_{\mathsf{GEE}}^{\mathbf{A}} = 1]\big| \le q_{\mathrm{c}} \cdot \mathrm{Adv}_{\mathsf{EC}}^{\text{ot-ror}}(\ddot{\mathbf{A}}),$$

and the run time of $\ddot{\mathbf{A}}$ is at most about that of MU-MO-REAL$_{\mathsf{GEE}}^{\mathbf{A}}$. □

Ciphertext integrity. GEE satisfies the ciphertext integrity if the underlying AE satisfies the ciphertext integrity and EC satisfies the second-ciphertext unforgeability. To achieve the ciphertext integrity, the generic construction utilizes the binding tag to bind a ciphertext of data with the wrapped data-encryption key instead of relying on the semantics of associated data and headers.

Theorem 2. *For any adversary* $\mathbf{A}$ *against* GEE *for ciphertext integrity making* q_{e} *queries to the* **Enc** *oracle and* q_{w} *queries to the* **Wrap** *oracle, there exists*

$u \leftarrow 0$; $d \leftarrow 0$; $\mathcal{X} \leftarrow \emptyset$; $\mathcal{Y} \leftarrow \emptyset$
$b \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{ChalEnc},\mathbf{Dec}}$
return b

New()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}$

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return (C_d, B_d)

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \leftarrow \mathsf{AEenc}(K_i, (B_j, H_{i,j}), L_j)$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, B_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
return (C_d, B_d)

Dec(i, A, C, B, S, H)
if $(i, A, C, B, S, H) \notin \mathcal{Y}$ **then**
  return $\perp$
end if
return $\mathsf{GDec}(K_i, A, C, B, S, H)$

Fig. 7: MU-MO-REAL$^{\mathbf{A}}_{\mathsf{GEE}}$

$u \leftarrow 0$; $d \leftarrow 0$; $\mathcal{X} \leftarrow \emptyset$; $\mathcal{Y} \leftarrow \emptyset$
$b \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{ChalEnc},\mathbf{Dec}}$
return b

New()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}$

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return (C_d, B_d)

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \twoheadleftarrow \Sigma^{\mathsf{clen}_{\mathrm{AE}}(|L_j|)}$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, B_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \twoheadleftarrow \Sigma^{\mathsf{clen}(|M|)} \times \mathcal{T}$
return (C_d, B_d)

Dec(i, A, C, B, S, H)
if $(i, A, C, B, S, H) \notin \mathcal{Y}$ **then**
  return $\perp$
end if
return (M_j, L_j) $\triangleright$ $(i, A, C, B, S, H) = (i, A_j, C_j, B_j, S_{i,j}, H_{i,j}) \in \mathcal{Y}$

Fig. 8: MU-MO-RAND$^{\mathbf{A}}_{\mathsf{GEE}}$

$u \leftarrow 0$; $d \leftarrow 0$; $\mathcal{X} \leftarrow \emptyset$; $\mathcal{Y} \leftarrow \emptyset$
$b \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{ChalEnc},\mathbf{Dec}}$
return b

New()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}$

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return (C_d, B_d)

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \leftarrow \mathsf{AEenc}(K_i, (B_j, H_{i,j}), L_j)$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, B_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
return (C_d, B_d)

Dec(i, A, C, B, S, H)
if $(i, A, C, B, S, H) \notin \mathcal{Y}$ **then**
  return $\perp$
end if
return $\underline{(M_j, L_j)} \triangleright (i, A, C, B, S, H) =$
$\underline{(i, A_j, C_j, B_j, S_{i,j}, H_{i,j})} \in \mathcal{Y}$

Fig. 9: MU-MO-ROR-G$_1^{\mathbf{A}}$

$u \leftarrow 0$; $d \leftarrow 0$; $\mathcal{X} \leftarrow \emptyset$; $\mathcal{Y} \leftarrow \emptyset$
$b \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{ChalEnc},\mathbf{Dec}}$
return b

New()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}$

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return (C_d, B_d)

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$\underline{S_{i,j} \twoheadleftarrow \Sigma^{\mathsf{clen}_{\mathrm{AE}}(|L_j|)}}$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, B_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
return (C_d, B_d)

Dec(i, A, C, B, S, H)
if $(i, A, C, B, S, H) \notin \mathcal{Y}$ **then**
  return $\perp$
end if
return $(M_j, L_j) \triangleright (i, A, C, B, S, H) =$
$(i, A_j, C_j, B_j, S_{i,j}, H_{i,j}) \in \mathcal{Y}$

Fig. 10: MU-MO-ROR-G$_2^{\mathbf{A}}$

$ctr \leftarrow 0$

ChalEnc(A, M)
$ctr \leftarrow ctr + 1$
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
if $ctr \leq k$ **then**
  $(C_d, B_d) \twoheadleftarrow \Sigma^{\mathsf{clen}_{\mathrm{EC}}(|M|)} \times \mathcal{T}$
else
  $(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
end if
return (C_d, B_d)

Fig. 11: Game MU-MO-HYB$^{\mathbf{A}}_{k}$

some adversaries $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{mu-mo-ctxt}}_{\mathsf{GEE}}(\mathbf{A}) \leq \mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{AE}}(\dot{\mathbf{A}}) + q_{\mathrm{e}} \cdot \mathrm{Adv}^{\mathrm{scu}}_{\mathsf{EC}}(\ddot{\mathbf{A}}).$$

$\dot{\mathbf{A}}$ *makes at most* q_{w} *queries to the* **AEenc** *oracle. The run time of* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *is at most about that of* MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$.

Proof. The game MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$ is given in Fig. 12. Without loss of generality, it is assumed that, whenever MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$ outputs `true`, $\mathbf{A}$ terminates right after a response from **Dec** to its query succeeding in setting *win* `true`.

Suppose that MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$ outputs `true` and $(i^*, A^*, C^*, B^*, S^*, H^*)$ sets *win* `true`. Then, there are two cases for the process of **Dec** on $(i^*, A^*, C^*, B^*, S^*, H^*)$: (1) for every $(i, A, C, B, S, H) \in \mathcal{Y}$, $(i, B, S, H) \neq (i^*, B^*, S^*, H^*)$; (2) there exists some $(i', A', C', B', S', H') \in \mathcal{Y}$ such that $(i', B', S', H') = (i^*, B^*, S^*, H^*)$ and $(A', C') \neq (A^*, C^*)$. Let Win_1 and Win_2 be the events such that MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$ outputs `true` in the first case and the second case, respectively. Then,

$$\mathrm{Adv}^{\text{mu-mo-ctxt}}_{\mathsf{GEE}}(\mathbf{A}) \leq \Pr[\mathrm{Win}_1] + \Pr[\mathrm{Win}_2].$$

For Win_1, let $\mathbf{A}_1$ be an adversary against AE for the ciphertext integrity. In MU-CTXT$^{\mathbf{A}_1}_{\mathsf{AE}}$, $\mathbf{A}_1$ runs MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$. For a call to **New** by $\mathbf{A}$, $\mathbf{A}_1$ makes a call to **AEnew**. For a query to **Corrupt** by $\mathbf{A}$, $\mathbf{A}_1$ asks it to **AEcorrupt** and forwards the reply to $\mathbf{A}$. $\mathbf{A}_1$ simulates **Enc** for $\mathbf{A}$. For a query $(i, j, H_{i,j}) \in [1, u] \times [1, d] \times \mathcal{H}$ to **Wrap** by $\mathbf{A}$, $\mathbf{A}_1$ makes a query $(i, (B_j, H_{i,j}), L_j)$ to **AEenc** and forwards the reply to $\mathbf{A}$. $\mathbf{A}_1$ simulates **Dec** by making use of **AEdec** for AEenc. Then, in MU-CTXT$^{\mathbf{A}_1}_{\mathsf{AE}}$, $(i^*, (B^*, H^*), S^*)$ sets *win* `true`. Thus, $\Pr[\mathrm{Win}_1] \leq \mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{AE}}(\mathbf{A}_1)$. $\mathbf{A}_1$ makes at most q_{w} queries to **AEenc**. The run time of $\mathbf{A}_1$ is at most about that of MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$.

Let us see Win_2. Let $\mathbf{A}_2$ be an adversary against EC for the second ciphertext unforgeability. $\mathbf{A}_2$ first samples $r \in [1, q_{\mathrm{e}}]$ uniformly at random. Then, $\mathbf{A}_2$ runs MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$. $\mathbf{A}_2$ simulates the oracles of $\mathbf{A}$ except that it asks the r-th query (A, M) to **Enc** made by $\mathbf{A}$ to its ECenc oracle and forwards (C, B) among the reply to $\mathbf{A}_2$. $\mathbf{A}_2$ is successful if the r-th query corresponds to

(i', A', C', B', S', H'). Thus, $(1/q_e)\Pr[\mathrm{Win}_2] \leq \mathrm{Adv}^{\mathrm{scu}}_{\mathsf{EC}}(\mathbf{A}_2)$. The run time of $\mathbf{A}_2$ is at most about that of MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$.

There may exist adversaries $\dot{\mathbf{A}}$ and $\ddot{\mathbf{A}}$ obtaining better advantages than $\mathbf{A}_1$ and $\mathbf{A}_2$, respectively, using the same amount of computational resources. □

$u \leftarrow 0$; $d \leftarrow 0$; $\mathcal{Y} \leftarrow \emptyset$; $\mathcal{Z} \leftarrow \emptyset$
$win \leftarrow$ `false`
$\mathbf{A}^{\mathbf{New},\mathbf{Corrupt},\mathbf{Enc},\mathbf{Wrap},\mathbf{Dec}}$
return win

New()
$u \leftarrow u + 1$; $K_u \twoheadleftarrow \mathcal{K}$

Corrupt(j)
if $j \notin [1, u]$ **then**
  return $\perp$
end if
$\mathcal{Z} \leftarrow \mathcal{Z} \cup \{j\}$
return K_j

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$(C_d, B_d) \leftarrow \mathsf{ECenc}(L_d, A, M)$
$A_d \leftarrow A$
return (C_d, B_d)

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \leftarrow \mathsf{AEenc}(K_i, (B_j, H_{i,j}), L_j)$
$\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, B_j, S_{i,j}, H_{i,j})\}$
return $S_{i,j}$

Dec(i, A, C, B, S, H)
if $i \notin [1, u]$ **then**
  return $\perp$
end if
$L' \leftarrow \mathsf{AEdec}(K_i, (B, H), S)$
if $L' = \perp$ **then**
  return $\perp$
end if
$M' \leftarrow \mathsf{ECdec}(L', A, C, B)$
if $M' = \perp$ **then**
  return $\perp$
end if
if $(i, A, C, B, S, H) \notin \mathcal{Y}$ **then**
  $win \leftarrow$ `true`
end if
return (M', L')

Fig. 12: Game MU-MO-CTXT$^{\mathbf{A}}_{\mathsf{GEE}}$

Soundness. GEE satisfies the soundness if the underlying encryptment EC satisfies the strong correctness and the strong receiver binding property:

Theorem 3. *Suppose that* EC *satisfies the strong correctness. Then, for any adversary* $\mathbf{A}$ *against* GEE *for soundness, there exists some adversary* $\dot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\mathrm{snd}}_{\mathsf{GEE}}(\mathbf{A}) \leq \mathrm{Adv}^{\text{sr-bind}}_{\mathsf{EC}}(\dot{\mathbf{A}}).$$

The run time of $\dot{\mathbf{A}}$ *is at most about that of* $\mathbf{A}$.

Proof. $\dot{\mathbf{A}}$ first runs $\mathbf{A}$ and gets a pair (K, A, C, B, S, H) and (K', A', C', B', S', H') produced by $\mathbf{A}$. Suppose that $\mathbf{A}$ succeeds in breaking the soundness of GEE. Then, $(A, C, B) = (A', C', B')$ and $\dot{\mathbf{A}}$ can compute

– $L \leftarrow \mathsf{AEdec}(K, (B, H), S)$, $M \leftarrow \mathsf{ECdec}(L, A, C, B)$ and
– $L' \leftarrow \mathsf{AEdec}(K', (B, H'), S')$, $M' \leftarrow \mathsf{ECdec}(L', A, C, B)$

satisfying $(M, L) \neq (M', L')$. Finally, $\dot{\mathbf{A}}$ outputs $((L, A, M), (L', A, M'), B)$. Since EC satisfies the strong correctness, $\mathsf{ECver}(A, M, L, B) = \mathsf{ECver}(A, M', L', B) = 1$. □

5 Conclusion

We have formalized the syntax and security requirements of envelope encryption. We have proposed a generic construction using encryptment and AEAD and confirmed its security based on the security of the components. Our proposal brings a new application of compactly committing property of authenticated encryption. It presents a purely cryptographic solution for the unforgeability of envelope encryption.

References

1. Albertini, A., Duong, T., Gueron, S., Kölbl, S., Luykx, A., Schmieg, S.: How to abuse and fix authenticated encryption without key commitment. In: Butler, K.R.B., Thomas, K. (eds.) 31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022. pp. 3291–3308. USENIX Association (2022), https://www.usenix.org/conference/usenixsecurity22/presentation/albertini
2. Bellare, M., Hoang, V.T.: Efficient schemes for committing authenticated encryption. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13276, pp. 845–875. Springer (2022). https://doi.org/10.1007/978-3-031-07085-3_29
3. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Okamoto, T. (ed.) Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings. Lecture Notes in Computer Science, vol. 1976, pp. 531–545. Springer (2000). https://doi.org/10.1007/3-540-44448-3_41
4. Chan, J., Rogaway, P.: On committing authenticated-encryption. In: Atluri, V., Pietro, R.D., Jensen, C.D., Meng, W. (eds.) Computer Security - ESORICS 2022 - 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26-30, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13555, pp. 275–294. Springer (2022). https://doi.org/10.1007/978-3-031-17146-8_14
5. Dodis, Y., Grubbs, P., Ristenpart, T., Woodage, J.: Fast message franking: From invisible salamanders to encryptment. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10991, pp. 155–186. Springer (2018). https://doi.org/10.1007/978-3-319-96884-1_6

6. Facebook: Messenger secret conversations. Technical Whitepaper (2016), https://about.fb.com/wp-content/uploads/2016/07/messenger-secret-conversations-technical-whitepaper.pdf
7. Farshim, P., Orlandi, C., Rosie, R.: Security of symmetric primitives under incorrect usage of keys. IACR Transactions on Symmetric Cryptology **2017**(1), 449–473 (2017). https://doi.org/10.13154/tosc.v2017.i1.449-473
8. Grubbs, P., Lu, J., Ristenpart, T.: Message franking via committing authenticated encryption. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III. Lecture Notes in Computer Science, vol. 10403, pp. 66–97. Springer (2017). https://doi.org/10.1007/978-3-319-63697-9_3
9. Gueron, S., Langley, A., Lindell, Y.: AES-GCM-SIV: Specification and analysis. Cryptology ePrint Archive, Paper 2017/168 (2017), https://eprint.iacr.org/2017/168
10. Hirose, S., Minematsu, K.: Compactly committing authenticated encryption using encryptment and tweakable block cipher. Cryptology ePrint Archive, Paper 2022/1670 (2022), https://eprint.iacr.org/2022/1670
11. Jutla, C.S.: Encryption modes with almost free message integrity. In: Pfitzmann, B. (ed.) Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding. Lecture Notes in Computer Science, vol. 2045, pp. 529–544. Springer (2001). https://doi.org/10.1007/3-540-44987-6_32
12. Katz, J., Yung, M.: Unforgeable encryption and chosen ciphertext secure modes of operation. In: Schneier, B. (ed.) Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings. Lecture Notes in Computer Science, vol. 1978, pp. 284–299. Springer (2000). https://doi.org/10.1007/3-540-44706-7_20
13. Krovetz, T., Rogaway, P.: The software performance of authenticated-encryption modes. In: Joux, A. (ed.) Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6733, pp. 306–327. Springer (2011). https://doi.org/10.1007/978-3-642-21702-9_18
14. Krovetz, T., Rogaway, P.: The design and evolution of OCB. Journal of Cryptology **34**(4), 36 (2021). https://doi.org/10.1007/s00145-021-09399-8
15. Len, J., Grubbs, P., Ristenpart, T.: Partitioning oracle attacks. In: Bailey, M., Greenstadt, R. (eds.) 30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021. pp. 195–212. USENIX Association (2021), https://www.usenix.org/conference/usenixsecurity21/presentation/len
16. Liskov, M.D., Rivest, R.L., Wagner, D.A.: Tweakable block ciphers. In: Yung, M. (ed.) Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings. Lecture Notes in Computer Science, vol. 2442, pp. 31–46. Springer (2002). https://doi.org/10.1007/3-540-45708-9_3
17. Nir, Y., Langley, A.: ChaCha20 and Poly1305 for IETF protocols. RFC 8439 (2018). https://doi.org/10.17487/RFC8439
18. NIST Special Publication 800-38D: Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC (2007). https://doi.org/10.6028/NIST.SP.800-38D
19. NIST Special Publication 800-38F: Recommendation for block cipher modes of operation: Methods for key wrapping (2012). https://doi.org/10.6028/NIST.SP.800-38F

20. Rogaway, P.: Authenticated-encryption with associated-data. In: Atluri, V. (ed.) Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS 2002, Washington, DC, USA, November 18-22, 2002. pp. 98–107. ACM (2002). https://doi.org/10.1145/586110.586125
21. Rogaway, P., Shrimpton, T.: A provable-security treatment of the key-wrap problem. In: Vaudenay, S. (ed.) Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings. Lecture Notes in Computer Science, vol. 4004, pp. 373–390. Springer (2006). https://doi.org/10.1007/11761679_23