Envelope Encryption in the Symmetric-Key Setting: A Formalization and Generic Constructions

Shoichi Hirose[1] and Kazuhiko Minematsu[2]

[1] University of Fukui, Fukui, Japan
hrs_shch@u-fukui.ac.jp
[2] NEC, Kawasaki, Japan
k-minematsu@nec.com

Abstract. Envelope encryption is a method for encrypting data using two distinct keys. The data are first encrypted using a data-encryption key, and then the data-encryption key is encrypted using a key-encryption key. Although it has been widely adopted in cloud services, to the best of our knowledge, formal treatment is lacking. In this paper, we formalize envelope encryption in the symmetric-key setting by defining its syntax and set of security requirements. Then, we present two generic constructions for envelope encryption. The first construction combines two AEAD schemes, one for data encryption and another for key wrapping, whereas the second leverages encryptment, one-time AEAD with the compactly committing property, for data encryption. We prove that the security of these constructions is reduced to that of the underlying primitives. This study bridges the gap between practical deployment and theoretical foundations and offers a framework for secure envelope encryption.

Keywords: Authenticated encryption · Key wrap · Committing security · Encryptment

1 Introduction

1.1 Background

Envelope encryption encrypts data using a symmetric encryption scheme with a secret key, known as a data-encryption key. Then, the key is encrypted using one or more key-encryption keys. This approach enables a recipient, possessing a secret key that matches one of the key-encryption keys, to recover the data-encryption key and then decrypt the data. Envelope encryption is commonly used for major cloud services [2,3,13,18,23,27,36].

Typically, envelope encryption employs authenticated encryption with associated data (AEAD) [32] to encrypt data. AEAD is a well-established symmetric-key cryptographic technique that guarantees both privacy and authenticity. Albertini et al. [1] highlighted the necessity for AEAD to be key-committing, meaning it should ensure that only the secret key used to generate a ciphertext can

decrypt it. In cases where AEAD is not key-committing, a ciphertext C might be decrypted into different data P_1 and P_2 using two separate secret keys K_1 and K_2, respectively. Consequently, a recipient with C and K_1 would obtain P_1, while another recipient with C and K_2 would obtain P_2. Nonetheless, to the best of our knowledge, there has been no formal examination of envelope encryption in existing literature. It remains unclear what constitutes *secure* envelope encryption and how it can be realized.

Envelope encryption is a highly effective method for multicasting from a server to multiple clients, where each client shares a secret key with the server. The server encrypts the data using an ephemeral data-encryption key, which is subsequently encrypted with the secret keys of the recipient clients. In certain scenarios, it may suffice to ensure data confidentiality for a specified duration, after which the data-encryption key may be disclosed. Nonetheless, even in such instances, ciphertext unforgeability remains imperative. Specifically, a malicious adversary should be unable to exploit the disclosed ephemeral data-encryption key to forge a ciphertext.

1.2 Our Contributions

First, we formalize the syntax and security requirements of envelope encryption in the symmetric-key setting. Formalized security requirements include confidentiality, ciphertext integrity, soundness, unlinkability, and robustness. Confidentiality and ciphertext integrity are inherited from the security notions of AEAD. Ciphertext integrity is relevant to multicast applications described above. Soundness captures the notion that all recipients allowed to decrypt the same ciphertext of data should recover identical data. Unlinkability refers to the notion that, for any wrapped data-encryption key, only the user possessing the corresponding key-encryption key can realize who can recover the data-encryption key and which ciphertext can be decrypted by it. Robustness requires that any wrapped data-encryption key functions exclusively with a single tuple of a key-encryption key and a ciphertext.

Second, we present two generic constructions for envelope encryption. The first construction uses AEAD for both data encryption and key wrapping. The second construction uses encryptment for data encryption and AEAD for key-wrapping. Notably, the latter construction can be perceived as a randomized variant of ccAEAD by Dodis et al. [9], which has a different interface than the original. We then prove the security of generic constructions in the multi-user setting, showing that their security properties are reduced to those of the underlying primitives.

A preliminary version [17] of this paper was presented at ICISC 2024. In this revised version, the syntax has been slightly modified, and the security requirements have been expanded to include unlinkability and robustness. Furthermore, the first generic construction is derived from the modification of the syntax.

1.3 Related Work

The formal exploration of authenticated encryption (AE) was initiated by Katz and Yung [20] and Bellare and Namprempre [5]. The first dedicated schemes were presented by Katz and Yung [20], and Jutla [19]. AEAD was first formalized and investigated by Rogaway [32].

The study of committing AEAD and its variations has been conducted by Farshim et al. [11], Len et al. [24], Bellare and Hoang [4], and Chan and Rogaway [8]. For committing AEAD, the entire ciphertext typically functions as a commitment.

Popular AEAD schemes do not possess the key-committing property. Dodis et al. [9] presented a practical attack revealing that AES-GCM [29] is not key-committing. Subsequently, Albertini et al. developed an extensive attack, successfully applying it to AES-GCM, ChaCha20-Poly1305 [28], AES-GCM-SIV [15], and OCB3 [21,22].

Compactly committing AEAD (ccAEAD), which is useful for message franking [10], was introduced by Grubbs et al. [14]. For ccAEAD, a portion of a ciphertext functions as a commitment to the secret key, message, and associated data. Encryptment was introduced by Dodis et al. [9] as a core component of ccAEAD. They showed that encryptment, utilizing either a single invocation of AEAD or two invocations of a PRF, results in ccAEAD. Additionally, they proposed the encryptment scheme HFC (hash function chaining) and suggested that SpongeWrap [7] works as encryptment. Hirose and Minematsu [16] further established that encryptment with a single invocation of a tweakable block cipher [25] yields ccAEAD.

Madden [26] examined the concept of multi-recipient authenticated Tag-KEM (mATKEM), a form of hybrid encryption designed for secure multicast communication. Additionally, he proposed the application of compactly committing authenticated encryption as a data encapsulation mechanism (DEM). Nevertheless, he did not offer a formal analysis of the approach.

A variety of papers have explored encryption schemes associated with "digital envelope." Some of these works aimed at enhancing secure data sharing within cloud storage environments by integrating symmetric-key and attribute-based encryption schemes [31,34,35]. Others have examined conventional hybrid encryption (e.g. [12]). However, to the best of our knowledge, these works have not provided formal treatment.

1.4 Organization

Section 2 introduces AEAD and encryptment. Section 3 formalizes the syntax and security requirements of envelope encryption. Sections 4 and 5 present two generic constructions of envelope encryption utilizing AEAD and analyze their security. Section 6 provides a concluding remark.

2 Preliminaries

Let $\Sigma := \{0,1\}$ and $\Sigma^* := \bigcup_{i \geq 0} \Sigma^i$. For sequence $x \in \Sigma^*$, $|x|$ is the length of x in bits. For integers j and k such that $j \leq k$, let $[j,k]$ be the set of integers between j and k inclusive. 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) [32] is a primitive of symmetric-key cryptography that provides privacy and authenticity. Here, AEAD is formalized as deterministic authenticated encryption [33].

Syntax. An AEAD scheme is defined as a set of algorithms, denoted as $\mathsf{AE} := (\mathsf{AEkg}, \mathsf{AEenc}, \mathsf{AEdec})$. It is associated with the following subsets of Σ^*: key space $\mathcal{K}_{\mathrm{AE}}$, associated-data space $\mathcal{A}_{\mathrm{AE}}$, message space $\mathcal{M}_{\mathrm{AE}}$, and ciphertext space $\mathcal{C}_{\mathrm{AE}}$. Additionally, it is designed to meet a targeted security level, which determines the key length and the length of each ciphertext.

- The key-generation algorithm AEkg outputs a secret key $K \twoheadleftarrow \mathcal{K}_{\mathrm{AE}}$.
- The encryption algorithm is a function defined as $\mathsf{AEenc} : \mathcal{K}_{\mathrm{AE}} \times \mathcal{A}_{\mathrm{AE}} \times \mathcal{M}_{\mathrm{AE}} \to \mathcal{C}_{\mathrm{AE}}$. 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 $|C|$. It is assumed that there exists a function $\mathsf{clen}_{\mathrm{AE}} : \mathbb{N} \to \mathbb{N}$ such that $|C| = \mathsf{clen}_{\mathrm{AE}}(|M|)$.
- The decryption algorithm is a function defined as $\mathsf{AEdec} : \mathcal{K}_{\mathrm{AE}} \times \mathcal{A}_{\mathrm{AE}} \times \mathcal{C}_{\mathrm{AE}} \to \mathcal{M}_{\mathrm{AE}} \cup \{\perp\}$, where $\perp \notin \mathcal{M}_{\mathrm{AE}}$.

AE is assumed to satisfy correctness, meaning that 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 $\mathsf{AEdec}(K, A, C) = \perp$.

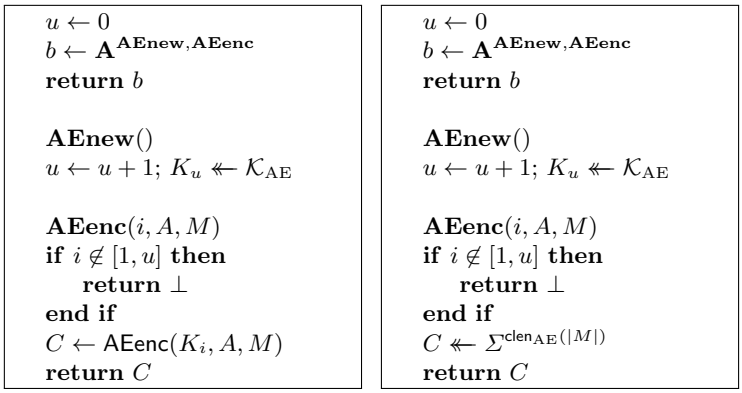
Security Requirements. The conventional security requirements of AEAD include both confidentiality and ciphertext integrity. Their definitions in the single-user setting [33] are naturally extended to those in the multi-user setting, following Bellare and Tackmann [6]. Furthermore, the concept of committing security has gained prominence, particularly in contexts where AEAD schemes are integrated into higher-level protocols.

Confidentiality. The two games MU-REAL and MU-RAND, depicted in Fig. 1, are introduced to formalize confidentiality as real-or-random indistinguishability. In both games, adversary **A** is provided with the oracles **AEnew** and **AEenc**. Upon a new invocation, **AEnew** generates a secret key for a new user. In response to a valid query, **AEenc** returns a ciphertext generated by AEenc in MU-REAL, while it returns a uniform random sequence in MU-RAND. Finally,

A outputs either 0 or 1. The advantage of **A** in terms of confidentiality is defined as follows:

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

It is imperative that **A** does not repeat the same query to **AEenc**. Otherwise, **A** would trivially distinguish between MU-REAL and MU-RAND.



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

Fig. 1: Games for confidentiality of AEAD

Ciphertext Integrity. The game MU-CTXT, depicted in Fig. 2, is introduced to formalize ciphertext integrity as existential unforgeability. In this game, adversary **A** is provided with the oracles **AEnew**, **AEcorrupt**, **AEenc**, and **AEdec**. **AEnew** and **AEenc** are analogous to those in MU-REAL, with the exception that **AEenc** records all its outputs. **AEcorrupt** returns the secret key of a user specified by **A**. For a query, **AEdec** executes AEdec and sets *win* to `true` if and only if **A** successfully forges. The advantage of **A** in terms of ciphertext integrity is defined as

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

Committing Security. Key-committing security requires that a ciphertext acts as a commitment to the key. The advantage of adversary **A** in terms of key-committing security is defined as

$$\begin{aligned}\mathrm{Adv}^{\text{kcmt}}_{\mathsf{AE}}(\mathbf{A}) \coloneqq \Pr[((K, A, M), (K', A', M')) \leftarrow \mathbf{A} : K \neq K' \\ \wedge \mathsf{AEenc}(K, A, M) = \mathsf{AEenc}(K', A', M')].\end{aligned}$$

Context-committing security requires that a ciphertext acts as a commitment to the entire input of the encryption algorithm. The advantage of adversary **A**

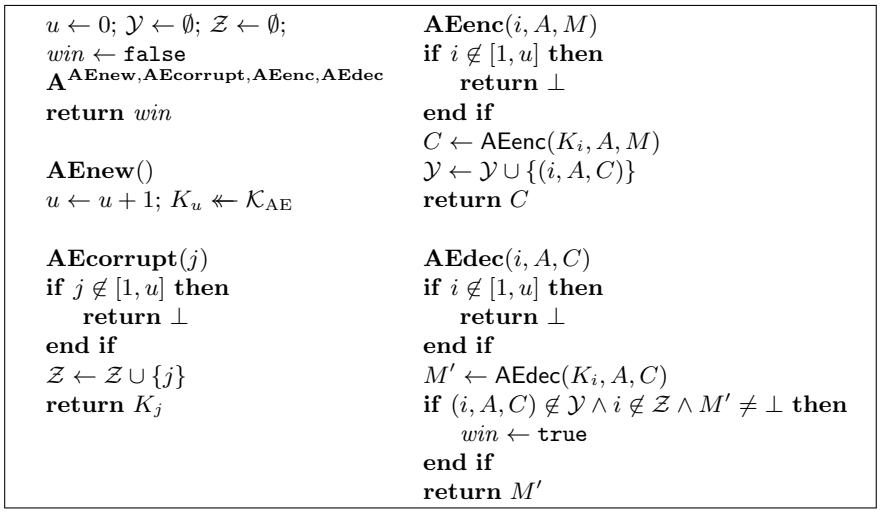


Fig. 2: Game MU-CTXT$^{\mathbf{A}}_{\mathsf{AE}}$ for ciphertext integrity of AEAD

in terms of context-committing security is defined as

$$\mathrm{Adv}^{\mathrm{ccmt}}_{\mathsf{AE}}(\mathbf{A}) \coloneqq \Pr[((K, A, M), (K', A', M')) \leftarrow \mathbf{A} : (K, A, M) \neq (K', A', M') \\ \wedge \mathsf{AEenc}(K, A, M) = \mathsf{AEenc}(K', A', M')].$$

2.2 Encryptment

Syntax. Encryptment [9] is defined as a set of algorithms, denoted as $\mathsf{EC} \coloneqq (\mathsf{ECkg}, \mathsf{ECenc}, \mathsf{ECdec}, \mathsf{ECver})$. It is associated with the following subsets of Σ^*: key space $\mathcal{K}_{\mathrm{EC}}$, associated-data space $\mathcal{A}_{\mathrm{EC}}$, message space $\mathcal{M}_{\mathrm{EC}}$, ciphertext space $\mathcal{C}_{\mathrm{EC}}$, and binding-tag space $\mathcal{T}_{\mathrm{EC}}$. Additionally, it has a specified security level that determines the key length and binding-tag length.

- The key-generation algorithm ECkg outputs a secret key $K \twoheadleftarrow \mathcal{K}_{\mathrm{EC}}$.
- The encryptment algorithm is a function defined as $\mathsf{ECenc} : \mathcal{K}_{\mathrm{EC}} \times \mathcal{A}_{\mathrm{EC}} \times \mathcal{M}_{\mathrm{EC}} \to \mathcal{C}_{\mathrm{EC}} \times \mathcal{T}_{\mathrm{EC}}$. 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 a function $\mathsf{clen}_{\mathrm{EC}} : \mathbb{N} \to \mathbb{N}$ such that $|C| = \mathsf{clen}_{\mathrm{EC}}(|M|)$.
- The decryptment algorithm is a function defined as $\mathsf{ECdec} : \mathcal{K}_{\mathrm{EC}} \times \mathcal{A}_{\mathrm{EC}} \times \mathcal{C}_{\mathrm{EC}} \times \mathcal{T}_{\mathrm{EC}} \to \mathcal{M}_{\mathrm{EC}} \cup \{\perp\}$, where $\perp \notin \mathcal{M}_{\mathrm{EC}}$.
- The verification algorithm is a function such that $\mathsf{ECver} : \mathcal{A}_{\mathrm{EC}} \times \mathcal{M}_{\mathrm{EC}} \times \mathcal{K}_{\mathrm{EC}} \times \mathcal{T}_{\mathrm{EC}} \to \Sigma$.

EC is assumed to satisfy correctness. That is, 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 further 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 $\mathsf{ECenc}(K, A, M) = (C, B)$.

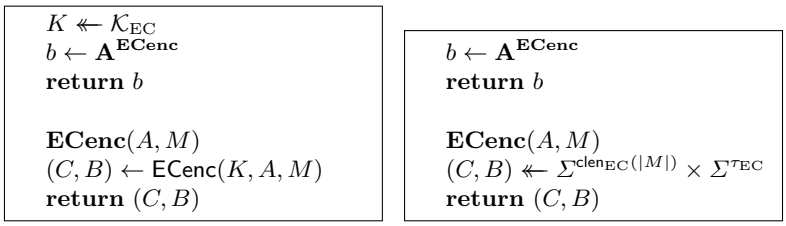
Remark 1. The encryptment scheme HFC [9] and SpongeWrap [7] as encryptment satisfy strong correctness.

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

Confidentiality. Two games, otREAL and otRAND, shown in Fig. 3, are introduced to formalize confidentiality. In both games, adversary $\mathbf{A}$ is allowed to ask a single query to oracle **ECenc**. The advantage of $\mathbf{A}$ in terms of confidentiality is defined as follows:

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

where "ot-ror" signifies "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. 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 receives (C, B) and K, where $K \twoheadleftarrow \mathcal{K}_{\mathrm{EC}}$ and $(C, B) \leftarrow \mathsf{ECenc}(K, A, M)$. Subsequently, $\mathbf{A}$ produces an output $(A', C') \in \mathcal{A}_{\mathrm{EC}} \times \mathcal{C}_{\mathrm{EC}}$. The advantage of $\mathbf{A}$ in terms of second-ciphertext unforgeability is defined as

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

Binding properties. The binding properties are defined in terms of the receiver and the sender. Receiver binding ensures that a malicious receiver cannot unjustly attribute an abusive message to a non-abusive sender. Let $\mathbf{A}$ be an adversary that generates 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 $\mathbf{A}$ in terms of receiver binding is defined as

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

The advantage of $\mathbf{A}$ in terms of strong receiver binding is defined as

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

Sender binding ensures that a malicious sender cannot evade accountability for sending an abusive message. The advantage of adversary $\mathbf{A}$ in terms of sender binding is defined as

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

3 Envelope Encryption

The syntax and security requirements of envelope encryption are formalized in the symmetric-key setting. Consequently, it is presumed that symmetric encryption is employed to secure both data-encryption keys and data. It is crucial to formalize the security requirements in the multi-user setting, as envelope encryption typically involves encrypting a data-encryption key with multiple key-encryption keys.

3.1 Syntax

A set of algorithms, denoted as $\mathsf{EE} \coloneqq (\mathsf{KEKGen}, \mathsf{DEKGen}, \mathsf{Enc}, \mathsf{Wrap}, \mathsf{Dec})$, defines envelope encryption. This is associated with the following subsets of Σ^*: key-encryption-key space $\mathcal{K}$, data-encryption-key space $\mathcal{L}$, associated-data space $\mathcal{A}$, message space $\mathcal{M}$, ciphertext space $\mathcal{C}$, joint space $\mathcal{J}$, header space $\mathcal{H}$, and wrapped-data-encryption-key space $\mathcal{S}$. Additionally, it has a specified security level that determines the key length and wrapped-data-encryption-key length.

- The key-encryption-key generation algorithm KEKGen outputs a secret key for key encryption, chosen uniformly at random from $\mathcal{K}$.
- The data-encryption-key generation algorithm DEKGen outputs a secret key for data encryption, also chosen uniformly at random from $\mathcal{L}$.
- The data-encryption algorithm is a function defined as $\mathsf{Enc} : \mathcal{L} \times \mathcal{A} \times \mathcal{M} \to \mathcal{C}$. For any $(L,A,M) \in \mathcal{L} \times \mathcal{A} \times \mathcal{M}$, if $C \leftarrow \mathsf{Enc}(L,A,M)$, 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 defined as $\mathsf{Wrap} : \mathcal{K} \times \mathcal{L} \times \mathcal{H} \times \mathcal{J} \to \mathcal{S}$, accompanied by the joint function $\mathsf{jnt} : \mathcal{A} \times \mathcal{C} \to \mathcal{J}$.
- The decryption algorithm is a function defined as $\mathsf{Dec} : \mathcal{K} \times \mathcal{A} \times \mathcal{C} \times \mathcal{S} \times \mathcal{H} \to (\mathcal{M} \times \mathcal{L}) \cup \{\perp\}$, where $\perp \notin \mathcal{M} \times \mathcal{L}$.

[1] It should be noted that the key-wrap algorithm specified here does not correspond to the methods in NIST SP 800-38F [30].

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 $C \leftarrow \mathsf{Enc}(L, A, M)$ and $S \leftarrow \mathsf{Wrap}(K, L, H, \mathsf{jnt}(A, C))$, then $\mathsf{Dec}(K, A, C, S, H) = (M, L)$. For any $(K, A, C, S, H) \in \mathcal{K} \times \mathcal{A} \times \mathcal{C} \times \mathcal{S} \times \mathcal{H}$, (A, C, S, H) is considered invalid with respect to K if $\mathsf{Dec}(K, A, C, S, H) = \perp$.

The formalization described above distinctly specifies the data-encryption algorithm and the key-wrap algorithm. This structure facilitates incremental key wrapping. For instance, in a cloud application that serves encrypted data, a server can enable a new recipient to access the data by wrapping the data-encryption key with the recipient's key-encryption key as needed.

The process of data transmission utilizing envelope encryption is illustrated in Fig. 4.

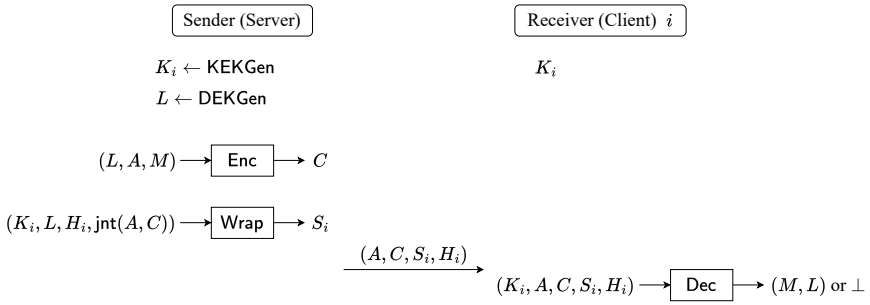


Fig. 4: Data transmission using envelope encryption. The sender shares secret key K_i with receiver i in advance.

3.2 Security Requirements

The security requirements for envelope encryption are confidentiality, ciphertext integrity, soundness, unlinkability, and robustness.

Confidentiality. The two games MU-REAL and MU-RAND, depicted in Fig. 5, are introduced to formalize confidentiality through real-or-random indistinguishability within a multi-opening context. In both games, adversary **A** is provided with the oracles **New**, **Enc**, **Wrap**, **ChalEnc**, and **Dec**. Upon invocation, **New** generates a secret key-encryption key for a new user. In response to a query, **Enc** returns a ciphertext generated by Enc. For a valid query, **Wrap** provides a wrapped data-encryption key produced by Wrap in MU-REAL, and a uniform random sequence sampled from $\mathcal{S}$ in MU-RAND. **ChalEnc** yields a ciphertext generated by Enc in MU-REAL and a uniform random sequence in MU-RAND. In response to a query, **Dec** returns a pair of a message and a data-encryption key only if the query is a ciphertext produced by **Enc** and **Wrap**. Finally, **A**

outputs either 0 or 1. The advantage of $\mathbf{A}$ in terms of confidentiality is defined as follows:

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

In this formalization, $\mathbf{A}$ is prohibited from repeating the same query to **Wrap**. Failure to adhere to the restriction would enable $\mathbf{A}$ to easily distinguish between MU-REAL and MU-RAND.

Ciphertext Integrity. The game MU-CTXT, depicted in Fig. 6, is introduced to formalize ciphertext integrity as existential unforgeability. In this game, adversary $\mathbf{A}$ is provided with the oracles **New**, **Corrupt**, **Enc**, **Wrap**, and **Dec**. The **New**, **Enc**, and **Wrap** oracles are analogous to those in MU-REAL. **Corrupt** returns the secret key-encryption key of a user specified by $\mathbf{A}$. For a query, **Dec** executes Dec and sets *win* to `true` if and only if $\mathbf{A}$ successfully forges. The advantage of $\mathbf{A}$ in terms of ciphertext integrity is defined as

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

Remark 2. In the formalization presented, the joint function is crucial for ciphertext integrity. Suppose that the joint function returns the empty sequence ε. In such a case, once a data-encryption key L is disclosed for a ciphertext (A, C, S, H), where $S = \mathsf{Wrap}(K, L, H, \varepsilon)$ for some key-encryption key K, an adversary can compute $C' = \mathsf{Enc}(L, A', M')$ for any (A', M') and forge (A', C', S, H) without knowledge of K. This constitutes a successful forgery because $\mathsf{Dec}(K, A', C', S, H) = (M', L)$ from correctness.

Soundness. Soundness pertains to the expectation that users should be able to recover the identical message from a ciphertext generated by Enc. The advantage of adversary $\mathbf{A}$ in terms of soundness is defined as

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

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

Unlinkability. Unlinkability captures the notion that, for any wrapped data-encryption key S, only the user who has the corresponding key-encryption key can see

- who is capable of recovering the data-encryption key from S and

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

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

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

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

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

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

(a) MU-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)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{Enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return C_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**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$
$C_d \twoheadleftarrow \Sigma^{\mathsf{clen}(|M|)}$
return C_d

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

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

Fig. 5: Games for confidentiality of envelope encryption

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

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

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

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

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

Dec(i, A, C, S, H)
if $i \notin [1, u]$ **then**
  return $\perp$
end if
if $\mathsf{Dec}(K_i, A, C, S, H) = \perp$ **then**
  return $\perp$
end if
if $(i, A, C, S, H) \notin \mathcal{Y} \wedge i \notin \mathcal{Z}$ **then**
  $win \leftarrow \texttt{true}$
end if
return $\mathsf{Dec}(K_i, A, C, S, H)$

Fig. 6: Game MU-CTXT$^{\mathbf{A}}_{\mathsf{EE}}$ for ciphertext integrity of envelope encryption

– which ciphertext (A, C) can be decrypted using the data-encryption key recovered from S.

The game MU-UNLK$^{\mathbf{A}}_{\mathsf{EE}}$, depicted in Fig. 7, is introduced to formalize unlinkability. In this game, adversary $\mathbf{A}$ is provided with the oracles **New**, **Enc**, **Wrap**, **Dec**, and **ChalWrap**. Upon invocation, **New** generates a secret key-encryption key for a new user. In response to a query, **Enc** returns a ciphertext generated by Enc. In response to a valid query, **Wrap** returns a wrapped data-encryption key produced by Wrap. For a valid query $((i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1}))$, **ChalWrap** returns a ciphertext produced by Wrap for (i_b, j_b, H_{i_b,j_b}), where $b \in \Sigma$ is selected uniformly at random at the beginning of the game. It is important to note that all the tuples $(i, j, H_{i,j})$, (i_0, j_0, H_{i_0,j_0}), and (i_1, j_1, H_{i_1,j_1}) in the queries to **Wrap** and **ChalWrap** must be distinct. In response to a query, **Dec** returns a pair of a message and a data-encryption key only if the query is a ciphertext produced by **Enc** and **Wrap**. Finally, $\mathbf{A}$ outputs either 0 or 1. MU-UNLK$^{\mathbf{A}}_{\mathsf{EE}}$ outputs 1 if the output of $\mathbf{A}$ equals b and 0 otherwise. The advantage of $\mathbf{A}$ in terms of unlinkability is defined as

$$\mathrm{Adv}^{\text{mu-unlk}}_{\mathsf{EE}}(\mathbf{A}) \coloneqq \Pr[\text{MU-UNLK}^{\mathbf{A}}_{\mathsf{EE}} = 1] - 1/2.$$

Robustness. Robustness is intrinsically linked to unlinkability. It requires that any wrapped data-encryption key functions exclusively with a single tuple com-

$u \leftarrow 0$; $d \leftarrow 0$
$\mathcal{Y} \leftarrow \emptyset$; $\mathcal{V} \leftarrow \emptyset$; $\mathcal{W} \leftarrow \emptyset$
$b \twoheadleftarrow \Sigma$
$b' \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{Dec},\mathbf{ChalWrap}}$
return $b \oplus b' \oplus 1$

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

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

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

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

ChalWrap$((i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1}))$
if $(i_0, j_0, H_{i_0,j_0}) = (i_1, j_1, H_{i_1,j_1})$ **then**
  return $\perp$
end if
if $(i_0, j_0) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
if $(i_1, j_1) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
if $(i_0, j_0, H_{i_0,j_0}) \in \mathcal{W} \cup \mathcal{V}$ **then**
  return $\perp$
end if
if $(i_1, j_1, H_{i_1,j_1}) \in \mathcal{W} \cup \mathcal{V}$ **then**
  return $\perp$
end if
$J_{j_b} \leftarrow \mathsf{jnt}(A_{j_b}, C_{j_b})$
$S^* \leftarrow \mathsf{Wrap}(K_{i_b}, L_{j_b}, H_{i_b,j_b}, J_{j_b})$
$\mathcal{V} \leftarrow \mathcal{V} \cup \{(i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1})\}$
return S^*

Fig. 7: Game MU-UNLK$^{\mathbf{A}}_{\mathsf{EE}}$ for unlinkability of envelope encryption

prising a key-encryption key and a ciphertext (K, A, C). The advantage of adversary $\mathbf{A}$ in terms of robustness is defined as

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

4 First Generic Scheme of Envelope Encryption

4.1 Construction

The first generic scheme of envelope encryption, denoted as $\mathsf{GEE1} \coloneqq (\mathsf{G1KEKGen}, \mathsf{G1DEKGen}, \mathsf{G1Enc}, \mathsf{G1Wrap}, \mathsf{G1Dec})$, comprises AEAD schemes $\mathsf{AE1} \coloneqq (\mathsf{AE1kg}, \mathsf{AE1enc}, \mathsf{AE1dec})$ and $\mathsf{AE2} \coloneqq (\mathsf{AE2kg}, \mathsf{AE2enc}, \mathsf{AE2dec})$. AE1 and AE2 are employed for data encryption and key encryption, respectively.

GEE1 is associated with key-encryption-key space $\mathcal{K}$, data-encryption-key space $\mathcal{L}$, associated-data space $\mathcal{A}$, message space $\mathcal{M}$, ciphertext space $\mathcal{C}$, header

space $\mathcal{H}$, and wrapped-data-encryption-key space $\mathcal{S}$. For $d \in \{1, 2\}$, $\mathsf{AE}d$ is associated with key space $\mathcal{K}_d$, associated-data space $\mathcal{A}_d$, message space $\mathcal{M}_d$, and ciphertext space $\mathcal{C}_d$. Let $\mathcal{K}_1 := \Sigma^{\ell_1}$.

Key-encryption-key generation $\mathsf{AE2kg}$ operates as $\mathsf{G1KEKGen}$, selecting a secret key-encryption key from $\mathcal{K} := \mathcal{K}_2$ uniformly at random.

Data-encryption-key generation $\mathsf{AE1kg}$ operates as $\mathsf{G1DEKGen}$, selecting a secret data-encryption key from $\mathcal{L} := \mathcal{K}_1$ uniformly at random.

Encryption $\mathsf{AE1enc}$ acts as $\mathsf{G1Enc}$. For $(L, A, M) \in \mathcal{L} \times \mathcal{A} \times \mathcal{M}$, $C \leftarrow \mathsf{AE1enc}(L, A, M)$, where $\mathcal{A} \subseteq \mathcal{A}_1$, $\mathcal{M} \subseteq \mathcal{M}_1$, and $\mathcal{C} \subseteq \mathcal{C}_1$.

Key Wrap $\mathsf{AE2enc}$ functions as $\mathsf{G1Wrap}$. For $(K, L, H, A, C) \in \mathcal{K} \times \mathcal{L} \times \mathcal{H} \times \mathcal{A} \times \mathcal{C}$, $S \leftarrow \mathsf{AE2enc}(K, (H, (A, C)), L)$, where $\mathcal{H} \times \mathcal{A} \times \mathcal{C} \subseteq \mathcal{A}_2$ and $\mathcal{L} \subseteq \mathcal{M}_2$. The joint function is the identity function.

Decryption $\mathsf{G1Dec}(K, A, C, S, H)$ is defined as follows:

1. $L' \leftarrow \mathsf{AE2dec}(K, (H, (A, C)), S)$.
2. If $L' = \perp$, then return $\perp$. Otherwise, $M' \leftarrow \mathsf{AE1dec}(L', A, C)$.
3. If $M' = \perp$, then return $\perp$. Otherwise, return (M', L').

The encryption and key wrap of $\mathsf{GEE1}$ are illustrated in Fig. 8.

Remark 3. Both $\mathsf{AE1}$ and $\mathsf{AE2}$ can be instantiated by nonce-based AEAD, although AEAD is formalized as deterministic authenticated encryption in Sect. 2. The discussions of security in the subsequent part of this section are applicable to such instantiations.

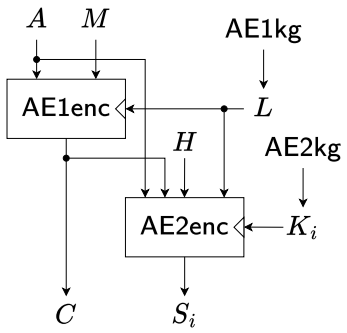


Fig. 8: Encryption and key wrap of $\mathsf{GEE1}$

4.2 Security

Confidentiality. $\mathsf{GEE1}$ satisfies confidentiality if both $\mathsf{AE1}$ and $\mathsf{AE2}$ satisfy confidentiality.

Theorem 1. *For any adversary* $\mathbf{A}$ *against* GEE1 *for confidentiality making* q_{e} *queries to its* **Enc** *oracle,* q_{w} *queries to its* **Wrap** *oracle, and* q_{c} *queries to its* **ChalEnc** *oracle, there exist some adversaries* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{mu-ror}}_{\mathsf{GEE1}}(\mathbf{A}) \le q_{\mathrm{c}} \cdot \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE1}}(\dot{\mathbf{A}}) + \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE2}}(\ddot{\mathbf{A}}) + \frac{(q_{\mathrm{e}}+q_{\mathrm{c}})^2}{2^{\ell_1+1}}.$$

$\dot{\mathbf{A}}$ *makes a single query to its* **AEenc** *oracle.* $\ddot{\mathbf{A}}$ *makes at most* q_{w} *queries to its* **AEenc** *oracle. The run times of* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *are at most approximately that of* $\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE1}}$.

Proof. For games $\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE1}}$ in Fig. 9 and $\text{MU-RAND}^{\mathbf{A}}_{\mathsf{GEE1}}$ in Fig. 10,

$$\mathrm{Adv}^{\text{mu-ror}}_{\mathsf{GEE1}}(\mathbf{A}) = \left|\Pr[\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE1}} = 1] - \Pr[\text{MU-RAND}^{\mathbf{A}}_{\mathsf{GEE1}} = 1]\right|.$$

The game $\text{MU-ROR-G1}^{\mathbf{A}}_1$ in Fig. 11 is different from $\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE1}}$ in that **Dec** selects the responses for each query in $\mathcal{Y}$ based on the queries and responses of **Enc** and **Wrap**. This change is minor, and

$$\Pr[\text{MU-ROR-G1}^{\mathbf{A}}_1 = 1] = \Pr[\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE1}} = 1].$$

The game $\text{MU-ROR-G1}^{\mathbf{A}}_2$ in Fig. 12 is different from $\text{MU-ROR-G1}^{\mathbf{A}}_1$ in that **Wrap** selects $S_{i,j}$ uniformly at random.

Let $\ddot{\mathbf{A}}$ be an adversary against AE2 for confidentiality. $\ddot{\mathbf{A}}$ runs $\mathbf{A}$ and simulates the oracles for $\mathbf{A}$. For each call to **New** made by $\mathbf{A}$, $\ddot{\mathbf{A}}$ makes a call to **AEnew**. For a query $(i, j, H_{i,j})$ such that $(i, j) \in [1, u] \times [1, d]$ to **Wrap** made by $\mathbf{A}$, if $(i, (H_{i,j}, (A_j, C_j)), L_j) \neq (i', (H_{i',j'}, (A_{j'}, C_{j'})), L_{j'})$ for any previous query $(i', j', H_{i',j'})$, then $\ddot{\mathbf{A}}$ makes a query $(i, (H_{i,j}, (A_j, C_j)), L_j)$ to **AEenc** and forwards the response to $\mathbf{A}$. Otherwise, $\ddot{\mathbf{A}}$ repeats the corresponding previous response to $\mathbf{A}$. $\ddot{\mathbf{A}}$ simply simulates **Enc**, **ChalEnc**, and **Dec**. Finally, $\ddot{\mathbf{A}}$ produces the same output as $\mathbf{A}$.

$\ddot{\mathbf{A}}$ runs $\text{MU-ROR-G1}^{\mathbf{A}}_1$ in $\text{MU-REAL}^{\ddot{\mathbf{A}}}_{\mathsf{AE2}}$. Thus, $\Pr[\text{MU-ROR-G1}^{\mathbf{A}}_1 = 1] = \Pr[\text{MU-REAL}^{\ddot{\mathbf{A}}}_{\mathsf{AE2}} = 1]$. In contrast, $\ddot{\mathbf{A}}$ runs $\text{MU-ROR-G1}^{\mathbf{A}}_2$ in $\text{MU-RAND}^{\ddot{\mathbf{A}}}_{\mathsf{AE2}}$ provided that all the data-encryption keys generated by **Enc** and **ChalEnc** are distinct. Thus,

$$\begin{aligned}
&\Pr[\text{MU-ROR-G1}^{\mathbf{A}}_2 = 1] - \frac{(q_{\mathrm{e}}+q_{\mathrm{c}})^2}{2^{\ell_1+1}} \\
&\quad \le \Pr[\text{MU-RAND}^{\ddot{\mathbf{A}}}_{\mathsf{AE2}} = 1] \le \Pr[\text{MU-ROR-G1}^{\mathbf{A}}_2 = 1] + \frac{(q_{\mathrm{e}}+q_{\mathrm{c}})^2}{2^{\ell_1+1}}.
\end{aligned}$$

Thus,

$$\begin{aligned}
&\left|\Pr[\text{MU-ROR-G1}^{\mathbf{A}}_1 = 1] - \Pr[\text{MU-ROR-G1}^{\mathbf{A}}_2 = 1]\right| \\
&\quad \le \left|\Pr[\text{MU-REAL}^{\ddot{\mathbf{A}}}_{\mathsf{AE2}} = 1] - \Pr[\text{MU-RAND}^{\ddot{\mathbf{A}}}_{\mathsf{AE2}} = 1]\right| + \frac{(q_{\mathrm{e}}+q_{\mathrm{c}})^2}{2^{\ell_1+1}} \\
&\quad = \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE2}}(\ddot{\mathbf{A}}) + \frac{(q_{\mathrm{e}}+q_{\mathrm{c}})^2}{2^{\ell_1+1}}.
\end{aligned}$$

$\ddot{\mathbf{A}}$ makes at most q_{w} queries to **AEenc**, and its run time is at most approximately that of $\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE1}}$.

Now, let us introduce the game $\text{MU-HYB1}^{\mathbf{A}}_{k}$ shown in Fig. 13, where $k \in [0, q_{\mathrm{c}}]$. Except **ChalEnc**, it uses the oracles of $\text{MU-ROR-G1}^{\mathbf{A}}_{2}$. Thus, it is equivalent to $\text{MU-ROR-G1}^{\mathbf{A}}_{2}$ and $\text{MU-RAND}^{\mathbf{A}}_{\mathsf{GEE1}}$ if k is equal to 0 and q_{c}, respectively. Let $\mathbf{A}'_l$ be an adversary against $\mathsf{AE1}$ for confidentiality, where $l \in [1, q_{\mathrm{c}}]$. In $\text{MU-REAL}^{\mathbf{A}'_l}_{\mathsf{AE1}}$ or $\text{MU-RAND}^{\mathbf{A}'_l}_{\mathsf{AE1}}$, $\mathbf{A}'_l$ runs $\text{MU-HYB1}^{\mathbf{A}}_{l-1}$ except that, for the l-th query (A, M) to **ChalEnc** made by $\mathbf{A}$, $\mathbf{A}'_l$ makes a call to **AEnew** and asks $(1, A, M)$ to **AEenc**. Finally, $\mathbf{A}'_l$ produces the same output as $\mathbf{A}$. Then, $\text{MU-REAL}^{\mathbf{A}'_l}_{\mathsf{AE1}}$ and $\text{MU-RAND}^{\mathbf{A}'_l}_{\mathsf{AE1}}$ are equivalent to $\text{MU-HYB1}^{\mathbf{A}}_{l-1}$ and $\text{MU-HYB1}^{\mathbf{A}}_{l}$, respectively. Thus,

$$\begin{aligned}
&\left|\Pr[\text{MU-ROR-G1}^{\mathbf{A}}_{2} = 1] - \Pr[\text{MU-RAND}^{\mathbf{A}}_{\mathsf{GEE1}} = 1]\right| \\
&\quad = \left|\Pr[\text{MU-HYB1}^{\mathbf{A}}_{0} = 1] - \Pr[\text{MU-HYB1}^{\mathbf{A}}_{q_{\mathrm{c}}} = 1]\right| \\
&\quad \le \sum_{l=1}^{q_{\mathrm{c}}} \left|\Pr[\text{MU-HYB1}^{\mathbf{A}}_{l-1} = 1] - \Pr[\text{MU-HYB1}^{\mathbf{A}}_{l} = 1]\right| \\
&\quad \le \sum_{l=1}^{q_{\mathrm{c}}} \left|\Pr[\text{MU-REAL}^{\mathbf{A}'_l}_{\mathsf{AE1}} = 1] - \Pr[\text{MU-RAND}^{\mathbf{A}'_l}_{\mathsf{AE1}} = 1]\right|,
\end{aligned}$$

and the run time of $\mathbf{A}'_l$ is at most approximately that of $\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE1}}$. Thus, there exists some $\dot{\mathbf{A}}$ such that

$$\left|\Pr[\text{MU-ROR-G1}^{\mathbf{A}}_{2} = 1] - \Pr[\text{MU-RAND}^{\mathbf{A}}_{\mathsf{GEE1}} = 1]\right| \le q_{\mathrm{c}} \cdot \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE1}}(\dot{\mathbf{A}}),$$

and the run time of $\dot{\mathbf{A}}$ is at most approximately that of $\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE1}}$. □

Ciphertext integrity. $\mathsf{GEE1}$ satisfies ciphertext integrity if $\mathsf{AE2}$ satisfies ciphertext integrity:

Theorem 2. *For any adversary* $\mathbf{A}$ *against* $\mathsf{GEE1}$ *for ciphertext integrity making* q_{w} *queries to its* **Wrap** *oracle and* q_{d} *queries to its* **Dec** *oracle, there exists an adversary* $\dot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{GEE1}}(\mathbf{A}) \le \mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{AE2}}(\dot{\mathbf{A}}).$$

$\dot{\mathbf{A}}$ *makes at most* q_{w} *queries to its* **AEenc** *oracle and* q_{d} *queries to its* **AEdec** *oracle. The run time of* $\dot{\mathbf{A}}$ *is at most approximately that of* $\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{GEE2}}$.

Proof. Game $\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{GEE1}}$ is shown in Fig. 14. Without loss of generality, it is assumed that, whenever $\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{GEE1}}$ outputs `true`, $\mathbf{A}$ terminates immediately after a response from **Dec** to its query succeeding in setting *win* to `true`.

Suppose that $\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{GEE1}}$ outputs `true` and $(i^*, A^*, C^*, S^*, H^*)$ sets *win* to `true`. Let $\mathbf{A}_1$ be an adversary against $\mathsf{AE2}$ for ciphertext integrity. In $\text{MU-CTXT}^{\mathbf{A}_1}_{\mathsf{AE2}}$,

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

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

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return C_d

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \leftarrow \mathsf{AE2enc}(K_i, (H_{i,j}, (A_j, C_j)), L_j)$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
return C_d

Dec(i, A, C, S, H)
if $(i, A, C, S, H) \notin \mathcal{Y}$ **then**
  return $\perp$
end if
return $\mathsf{G1Dec}(K_i, A, C, S, H)$

Fig. 9: MU-REAL$^{\mathbf{A}}_{\mathsf{GEE1}}$

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

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

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return C_d

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \twoheadleftarrow \Sigma^{\mathsf{clen}_{\mathrm{AE2}}(|L_j|)}$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$
$C_d \twoheadleftarrow \Sigma^{\mathsf{clen}_{\mathrm{AE1}}(|M|)}$
return C_d

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

Fig. 10: MU-RAND$^{\mathbf{A}}_{\mathsf{GEE1}}$

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

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

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return C_d

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \leftarrow \mathsf{AE2enc}(K_i, (H_{i,j}, (A_j, C_j)), L_j)$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
return C_d

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

Fig. 11: MU-ROR-G1$_1^{\mathbf{A}}$

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

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

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$; $\mathcal{X} \leftarrow \mathcal{X} \cup \{d\}$
return C_d

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \twoheadleftarrow \Sigma^{\mathsf{clen}_{\mathrm{AE2}}(|L_j|)}$
if $j \in \mathcal{X}$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, S_{i,j}, H_{i,j})\}$
end if
return $S_{i,j}$

ChalEnc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
return C_d

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

Fig. 12: MU-ROR-G1$_2^{\mathbf{A}}$

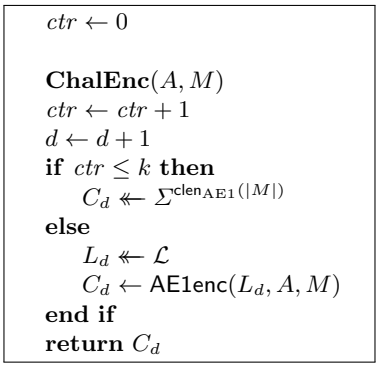


Fig. 13: MU-HYB1$_k^{\mathbf{A}}$, which shares **New**, **Enc**, **Wrap**, and **Dec** with MU-ROR-G1$_2^{\mathbf{A}}$

$\mathbf{A}_1$ runs MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE1}}$. 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 $(i, (H_{i,j}, (A_j, C_j)), L_j)$ to **AEenc** and forwards the reply to $\mathbf{A}$. $\mathbf{A}_1$ simulates **Dec** using **AEdec** for $\mathsf{AE2dec}$. Then, in MU-CTXT$^{\mathbf{A}_1}_{\mathsf{AE2}}$, $(i^*, (H^*, (A^*, C^*)), S^*)$ sets *win* to `true`. Thus, $\mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{GEE1}}(\mathbf{A}) = \mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{AE2}}(\mathbf{A}_1)$. $\mathbf{A}_1$ makes at most q_{w} queries to **AEenc** and at most q_{d} queries to **AEdec**. The run time of $\mathbf{A}_1$ is at most approximately that of MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE1}}$. There may exist adversary $\dot{\mathbf{A}}$ that achieves a better advantage than $\mathbf{A}_1$ using the same amount of computational resources. □

Soundness. $\mathsf{GEE1}$ satisfies soundness if the underlying $\mathsf{AE1}$ satisfies key-committing security:

Theorem 3. *For any adversary* $\mathbf{A}$ *against* $\mathsf{GEE1}$ *for soundness, there exists some adversary* $\dot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\mathrm{snd}}_{\mathsf{GEE1}}(\mathbf{A}) \leq \mathrm{Adv}^{\mathrm{kcmt}}_{\mathsf{AE1}}(\dot{\mathbf{A}}).$$

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

Proof. $\dot{\mathbf{A}}$ first runs $\mathbf{A}$ and obtains a pair (K, A, C, S, H) and (K', A', C', S', H') produced by $\mathbf{A}$. Suppose that $\mathbf{A}$ succeeds in breaking the soundness of $\mathsf{GEE1}$. Then, $(A, C) = (A', C')$, and $\dot{\mathbf{A}}$ can compute

- $L \leftarrow \mathsf{AE2dec}(K, (H, (A, C)), S)$, $M \leftarrow \mathsf{AE1dec}(L, A, C)$ and
- $L' \leftarrow \mathsf{AE2dec}(K', (H', (A', C')), S')$, $M' \leftarrow \mathsf{AE1dec}(L', A', C')$

satisfying $(M, L) \neq (M', L')$. Finally, $\dot{\mathbf{A}}$ outputs $((L, A, M), (L', A, M'))$. Since $\mathsf{AE1dec}$ is deterministic, $M = M'$ if $(L, A, C) = (L', A', C')$. Thus, $L \neq L'$, implying that $\dot{\mathbf{A}}$ succeeds in breaking the key-committing security of $\mathsf{AE1}$. □

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

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

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

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
$A_d \leftarrow A$
return C_d

Wrap($i, j, H_{i,j}$)
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
$S_{i,j} \leftarrow \mathsf{AE2enc}(K_i, (H_{i,j}, (A_j, C_j)), L_j)$
$\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, S_{i,j}, H_{i,j})\}$
return $S_{i,j}$

Dec(i, A, C, S, H)
if $i \notin [1, u]$ **then**
  return $\perp$
end if
$L' \leftarrow \mathsf{AE2dec}(K_i, (H, (A, C)), S)$
if $L' = \perp$ **then**
  return $\perp$
end if
$M' \leftarrow \mathsf{AE1dec}(L', A, C)$
if $M' = \perp$ **then**
  return $\perp$
end if
if $(i, A, C, S, H) \notin \mathcal{Y} \wedge i \notin \mathcal{Z}$ **then**
  $win \leftarrow \texttt{true}$
end if
return (M', L')

Fig. 14: MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE1}}$

Unlinkability. GEE1 satisfies unlinkability if AE2 satisfies confidentiality:

Theorem 4. *For any adversary* $\mathbf{A}$ *against* GEE1 *for unlinkability making* q_{e} *queries to its* **Enc** *oracle,* q_{w} *queries to its* **Wrap** *oracle, and* q_{c} *queries to its* **ChalWrap** *oracle, there exists some adversary* $\dot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{mu-unlk}}_{\mathsf{GEE1}}(\mathbf{A}) \leq \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE2}}(\dot{\mathbf{A}}) + \frac{q_{\mathrm{e}}^2}{2^{\ell_1+1}}.$$

$\dot{\mathbf{A}}$ *makes at most* $(q_{\mathrm{w}} + q_{\mathrm{c}})$ *queries to its* **AEenc** *oracle. The run time of* $\dot{\mathbf{A}}$ *is at most approximately that of* MU-UNLK$^{\mathbf{A}}_{\mathsf{GEE1}}$.

Proof. Game MU-UNLK$^{\mathbf{A}}_{\mathsf{GEE1}}$ is shown in Fig. 15. Game MU-UNLK-G1$^{\mathbf{A}}_1$ in Fig. 16 is different from MU-UNLK$^{\mathbf{A}}_{\mathsf{GEE1}}$ in that **Wrap** and **ChalWrap** select $S_{i,j}$ and S^* uniformly at random, respectively. The function $\mathsf{clen}_{\mathrm{AE2}}$ relates the length of a ciphertext to that of the corresponding message for AE2.

Let $\dot{\mathbf{A}}$ be an adversary against AE2 for confidentiality. $\dot{\mathbf{A}}$ runs $\mathbf{A}$ and simulates the oracles for $\mathbf{A}$. For each call to **New** made by $\mathbf{A}$, $\dot{\mathbf{A}}$ makes a call to **AEnew**. $\dot{\mathbf{A}}$ simulates oracle **Enc** using AE1.

For a query $(i, j, H_{i,j})$ to **Wrap** made by $\mathbf{A}$ such that $(i, j) \in [1, u] \times [1, d]$ and $(i, j, H_{i,j}) \notin \mathcal{W} \cup \mathcal{V}$, if $(i, (H_{i,j}, (A_j, C_j)), L_j)$ is a new query to **AEenc**,

then $\dot{\mathbf{A}}$ asks it to **AEenc** and forwards the response to $\mathbf{A}$. Otherwise, $\dot{\mathbf{A}}$ does not ask it to **AEenc** and repeats the corresponding previous response to $\mathbf{A}$. For a query $((i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1}))$ to **ChalWrap** made by $\mathbf{A}$ such that $(i_0, j_0, H_{i_0,j_0}) \neq (i_1, j_1, H_{i_1,j_1})$, $\{(i_0, j_0), (i_1, j_1)\} \subseteq [1, u] \times [1, d]$, and $\{(i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1})\} \cap (\mathcal{W} \cup \mathcal{V}) = \emptyset$, if $(i_b, (H_{i_b,j_b}, (A_{j_b}, C_{j_b})), L_{j_b})$ is a new query to **AEenc**, then $\dot{\mathbf{A}}$ asks it to **AEenc** and forwards the response to $\mathbf{A}$. Otherwise, $\dot{\mathbf{A}}$ does not ask it to **AEenc** and repeats the corresponding previous response to $\mathbf{A}$. $\dot{\mathbf{A}}$ simply simulates **Dec**. Finally, $\dot{\mathbf{A}}$ outputs $b \oplus b' \oplus 1$.

$\dot{\mathbf{A}}$ runs $\text{MU-UNLK}^{\mathbf{A}}_{\mathsf{GEE1}}$ in $\text{MU-REAL}^{\dot{\mathbf{A}}}_{\mathsf{AE2}}$. In contrast, $\dot{\mathbf{A}}$ runs $\text{MU-UNLK-G1}^{\mathbf{A}}_{1}$ in $\text{MU-RAND}^{\dot{\mathbf{A}}}_{\mathsf{AE2}}$ provided that all the data-encryption keys generated by **Enc** are distinct. Thus,

$$\begin{aligned}
\mathrm{Adv}^{\text{mu-unlk}}_{\mathsf{GEE1}}(\mathbf{A}) &= \Pr[\text{MU-UNLK}^{\mathbf{A}}_{\mathsf{GEE1}} = 1] - 1/2 \\
&= \Pr[\text{MU-UNLK}^{\mathbf{A}}_{\mathsf{GEE1}} = 1] - \Pr[\text{MU-UNLK-G1}^{\mathbf{A}}_{1} = 1] \\
&\leq \Pr[\text{MU-REAL}^{\dot{\mathbf{A}}}_{\mathsf{AE2}} = 1] - \Pr[\text{MU-RAND}^{\dot{\mathbf{A}}}_{\mathsf{AE2}} = 1] + \frac{q_{\mathrm{e}}^2}{2^{\ell_1+1}} \\
&\leq \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE2}}(\dot{\mathbf{A}}) + \frac{q_{\mathrm{e}}^2}{2^{\ell_1+1}}.
\end{aligned}$$

$\dot{\mathbf{A}}$ makes at most $(q_{\mathrm{w}} + q_{\mathrm{c}})$ queries to **AEenc**, and its run time is at most approximately that of $\text{MU-UNLK}^{\mathbf{A}}_{\mathsf{GEE1}}$. □

Robustness. GEE1 satisfies robustness if AE2 satisfies the context-committing property:

Theorem 5. *For any adversary* $\mathbf{A}$ *against* GEE1 *for robustness, there exists some adversary* $\dot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{rbst}}_{\mathsf{GEE1}}(\mathbf{A}) \leq \mathrm{Adv}^{\text{ccmt}}_{\mathsf{AE2}}(\dot{\mathbf{A}}).$$

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

Proof. $\dot{\mathbf{A}}$ first runs $\mathbf{A}$ and obtains a pair (K, A, C, S, H) and (K', A', C', S', H') produced by $\mathbf{A}$. Suppose that $\mathbf{A}$ successfully breaks the robustness of GEE1. Then, $S = S'$, $(K, A, C) \neq (K', A', C')$, and $\dot{\mathbf{A}}$ can compute

- $L \leftarrow \mathsf{AE2dec}(K, (H, (A, C)), S)$, $M \leftarrow \mathsf{AE1dec}(L, A, C)$ and
- $L' \leftarrow \mathsf{AE2dec}(K', (H', (A', C')), S')$, $M' \leftarrow \mathsf{AE1dec}(L', A', C')$.

Finally, $\dot{\mathbf{A}}$ outputs $((K, (H, (A, C)), L), (K', (H', (A', C')), L'))$. $\dot{\mathbf{A}}$ succeeds in breaking the context-committing property of AE2. □

5 Second Generic Scheme of Envelope Encryption

$u \leftarrow 0$; $d \leftarrow 0$
$\mathcal{Y} \leftarrow \emptyset$; $\mathcal{V} \leftarrow \emptyset$; $\mathcal{W} \leftarrow \emptyset$
$b \twoheadleftarrow \Sigma$
$b' \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{Dec},\mathbf{ChalWrap}}$
return $b \oplus b' \oplus 1$

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

Enc(A, M)
$d \leftarrow d + 1$; $L_d \twoheadleftarrow \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$
return C_d

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
if $(i, j, H_{i,j}) \in \mathcal{W} \cup \mathcal{V}$ **then**
  return $\perp$
end if
$S_{i,j} \leftarrow \mathsf{AE2enc}(K_i, (H_{i,j}, (A_j, C_j)), L_j)$
$\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, S_{i,j}, H_{i,j})\}$
$\mathcal{W} \leftarrow \mathcal{W} \cup \{(i, j, H_{i,j})\}$
return $S_{i,j}$

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

ChalWrap$((i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1}))$
if $(i_0, j_0, H_{i_0,j_0}) = (i_1, j_1, H_{i_1,j_1})$ **then**
  return $\perp$
end if
if $(i_0, j_0) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
if $(i_1, j_1) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
if $(i_0, j_0, H_{i_0,j_0}) \in \mathcal{W} \cup \mathcal{V}$ **then**
  return $\perp$
end if
if $(i_1, j_1, H_{i_1,j_1}) \in \mathcal{W} \cup \mathcal{V}$ **then**
  return $\perp$
end if
$S^* \leftarrow \mathsf{AE2enc}(K_{i_b}, (H_{i_b,j_b}, (A_{j_b}, C_{j_b})), L_{j_b})$
$\mathcal{V} \leftarrow \mathcal{V} \cup \{(i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1})\}$
return S^*

Fig. 15: MU-UNLK$^{\mathbf{A}}_{\mathsf{GEE1}}$

5.1 Construction

The second generic scheme of envelope encryption, denoted as $\mathsf{GEE2} \coloneqq (\mathsf{G2KEKGen}, \mathsf{G2DEKGen}, \mathsf{G2Enc}, \mathsf{G2Wrap}, \mathsf{G2Dec})$, comprises encryptment $\mathsf{EC} \coloneqq (\mathsf{ECkg}, \mathsf{ECenc}, \mathsf{ECdec}, \mathsf{ECver})$ and AEAD $\mathsf{AE} \coloneqq (\mathsf{AEkg}, \mathsf{AEenc}, \mathsf{AEdec})$. It follows the observation by Albertini et al. [1] and utilizes encryptment, which is one-time key-committing AEAD, for data encryption.

$\mathsf{GEE2}$ is associated with key-encryption-key space $\mathcal{K}$, data-encryption-key space $\mathcal{L}$, associated-data space $\mathcal{A}$, message space $\mathcal{M}$, ciphertext space $\mathcal{C} \times \mathcal{T}$, header space $\mathcal{H}$, and wrapped-data-encryption-key space $\mathcal{S}$. EC is associated with key space $\mathcal{K}_{\mathrm{EC}} \coloneqq \Sigma^{\ell_{\mathrm{EC}}}$, associated-data space $\mathcal{A}_{\mathrm{EC}}$, message space $\mathcal{M}_{\mathrm{EC}}$, ciphertext space $\mathcal{C}_{\mathrm{EC}}$, and binding-tag space $\mathcal{T}_{\mathrm{EC}}$. AE is associated with key space $\mathcal{K}_{\mathrm{AE}}$, associated-data space $\mathcal{A}_{\mathrm{AE}}$, message space $\mathcal{M}_{\mathrm{AE}}$, and ciphertext space $\mathcal{C}_{\mathrm{AE}}$.

Key-encryption-key generation AEkg operates as $\mathsf{G2KEKGen}$, selecting a secret key-encryption key from $\mathcal{K} \coloneqq \mathcal{K}_{\mathrm{AE}}$ uniformly at random.

Data-encryption-key generation ECkg operates as $\mathsf{G2DEKGen}$, selecting a secret data-encryption key from $\mathcal{L} \coloneqq \mathcal{K}_{\mathrm{EC}}$ uniformly at random.

$u \leftarrow 0$; $d \leftarrow 0$
$\mathcal{Y} \leftarrow \emptyset$; $\mathcal{V} \leftarrow \emptyset$; $\mathcal{W} \leftarrow \emptyset$
$b \xleftarrow{\$} \Sigma$
$b' \leftarrow \mathbf{A}^{\mathbf{New},\mathbf{Enc},\mathbf{Wrap},\mathbf{Dec},\mathbf{ChalWrap}}$
return $b \oplus b' \oplus 1$

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

Enc(A, M)
$d \leftarrow d + 1$; $L_d \xleftarrow{\$} \mathcal{L}$
$C_d \leftarrow \mathsf{AE1enc}(L_d, A, M)$
$(A_d, M_d) \leftarrow (A, M)$
return C_d

Wrap$(i, j, H_{i,j})$
if $(i, j) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
if $(i, j, H_{i,j}) \in \mathcal{W} \cup \mathcal{V}$ **then**
  return $\perp$
end if
$\underline{S_{i,j} \xleftarrow{\$} \Sigma^{\mathsf{clen}_{\mathrm{AE2}}(|L_j|)}}$
$\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(i, A_j, C_j, S_{i,j}, H_{i,j})\}$
$\mathcal{W} \leftarrow \mathcal{W} \cup \{(i, j, H_{i,j})\}$
return $S_{i,j}$

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

ChalWrap$((i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1}))$
if $(i_0, j_0, H_{i_0,j_0}) = (i_1, j_1, H_{i_1,j_1})$ **then**
  return $\perp$
end if
if $(i_0, j_0) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
if $(i_1, j_1) \notin [1, u] \times [1, d]$ **then**
  return $\perp$
end if
if $(i_0, j_0, H_{i_0,j_0}) \in \mathcal{W} \cup \mathcal{V}$ **then**
  return $\perp$
end if
if $(i_1, j_1, H_{i_1,j_1}) \in \mathcal{W} \cup \mathcal{V}$ **then**
  return $\perp$
end if
$\underline{S^* \xleftarrow{\$} \Sigma^{\mathsf{clen}_{\mathrm{AE2}}(|L_{j_b}|)}}$
$\mathcal{V} \leftarrow \mathcal{V} \cup \{(i_0, j_0, H_{i_0,j_0}), (i_1, j_1, H_{i_1,j_1})\}$
return S^*

Fig. 16: MU-UNLK-G1$_1^{\mathbf{A}}$

Encryption ECenc acts as $\mathsf{G2Enc}$. For $(L, A, M) \in \mathcal{L} \times \mathcal{A} \times \mathcal{M}$, $(C, B) \leftarrow \mathsf{ECenc}(L, A, M)$, where $\mathcal{A} \subseteq \mathcal{A}_{\mathrm{EC}}$, $\mathcal{M} \subseteq \mathcal{M}_{\mathrm{EC}}$, $\mathcal{C} \subseteq \mathcal{C}_{\mathrm{EC}}$, and $\mathcal{T} \subseteq \mathcal{T}_{\mathrm{EC}}$.

Key Wrap AEenc functions as $\mathsf{G2Wrap}$. For $(K, L, H, B) \in \mathcal{K} \times \mathcal{L} \times \mathcal{H} \times \mathcal{T}$, $S \leftarrow \mathsf{AEenc}(K, (H, B), L)$, where $\mathcal{H} \times \mathcal{T} \subseteq \mathcal{A}_{\mathrm{AE}}$ and $\mathcal{L} \subseteq \mathcal{M}_{\mathrm{AE}}$. The joint function is $(A, C, B) \mapsto B$.

Decryption $\mathsf{G2Dec}(K, A, C, B, S, H)$ is defined as follows:

1. $L' \leftarrow \mathsf{AEdec}(K, (H, B), S)$.
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').

The encryption and key wrap of $\mathsf{GEE2}$ are illustrated in Fig. 17.

Remark 4. AE can be instantiated by nonce-based AEAD, though AE is formalized as deterministic authenticated encryption in Sect. 2. The discussions of security in the subsequent part of this section are applicable to such instantiations.

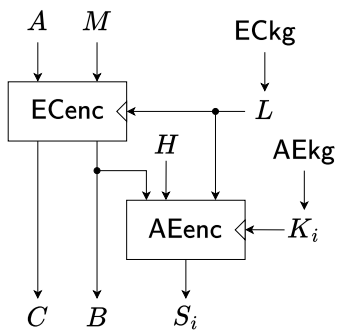


Fig. 17: Encryption and key wrap of GEE2

5.2 Security

Confidentiality. GEE2 satisfies confidentiality if both AE and EC satisfy confidentiality:

Theorem 6. *For any adversary* $\mathbf{A}$ *against* GEE2 *for confidentiality making* q_e *queries to its* **Enc** *oracle,* q_w *queries to its* **Wrap** *oracle, and* q_c *queries to its* **ChalEnc** *oracle, there exist some adversaries* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{mu-ror}}_{\mathsf{GEE2}}(\mathbf{A}) \le q_c \cdot \mathrm{Adv}^{\text{ot-ror}}_{\mathsf{EC}}(\dot{\mathbf{A}}) + \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE}}(\ddot{\mathbf{A}}) + \frac{(q_e + q_c)^2}{2^{\ell_{\mathrm{EC}}+1}}.$$

$\ddot{\mathbf{A}}$ *makes at most* q_w *queries to its* **AEenc** *oracle. The run time of* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *is at most approximately that of* $\text{MU-REAL}^{\mathbf{A}}_{\mathsf{GEE2}}$.

The proof is omitted since it is similar to the proof of Theorem 1.

Ciphertext integrity. GEE2 satisfies ciphertext integrity if AE satisfies ciphertext integrity and EC satisfies second-ciphertext unforgeability:

Theorem 7. *For any adversary* $\mathbf{A}$ *against* GEE2 *for ciphertext integrity making* q_e *queries to its* **Enc** *oracle and* q_w *queries to its* **Wrap** *oracle, there exists some adversaries* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{GEE2}}(\mathbf{A}) \le \mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{AE}}(\dot{\mathbf{A}}) + q_e \cdot \mathrm{Adv}^{\text{scu}}_{\mathsf{EC}}(\ddot{\mathbf{A}}).$$

$\dot{\mathbf{A}}$ *makes at most* q_w *queries to its* **AEenc** *oracle. The run time of* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *is at most approximately that of* $\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{GEE2}}$.

Proof. The game $\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{GEE2}}$ is shown in Fig. 18. Without loss of generality, it is assumed that, whenever $\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{GEE2}}$ outputs `true`, $\mathbf{A}$ terminates immediately after a response from **Dec** to its query succeeding in setting *win* to `true`.

Suppose that $\text{MU-CTXT}^{\mathbf{A}}_{\mathsf{GEE2}}$ outputs `true` and $(i^*, A^*, C^*, B^*, S^*, H^*)$ sets *win* to `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 events such that MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE2}}$ outputs `true` in the first and second cases, respectively. Then,

$$\mathrm{Adv}^{\text{mu-ctxt}}_{\mathsf{GEE2}}(\mathbf{A}) \leq \Pr[\mathrm{Win}_1] + \Pr[\mathrm{Win}_2].$$

For Win_1, let $\mathbf{A}_1$ be an adversary against AE for ciphertext integrity. In MU-CTXT$^{\mathbf{A}_1}_{\mathsf{AE}}$, $\mathbf{A}_1$ runs MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE2}}$. 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* to `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 approximately that of MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE2}}$.

Let us see Win_2. Let $\mathbf{A}_2$ be an adversary against EC for second ciphertext unforgeability. $\mathbf{A}_2$ first samples $r \in [1, q_{\mathrm{e}}]$ uniformly at random. Then, $\mathbf{A}_2$ runs MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE2}}$. $\mathbf{A}_2$ 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 the reply (C, B) to $\mathbf{A}_2$. Then, $\mathbf{A}_2$ is successful if the r-th query corresponds to (i', A', C', B', S', H'). Thus, $(1/q_{\mathrm{e}}) \Pr[\mathrm{Win}_2] \leq \mathrm{Adv}^{\mathrm{scu}}_{\mathsf{EC}}(\mathbf{A}_2)$. The run time of $\mathbf{A}_2$ is at most approximately that of MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE2}}$.

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

Soundness. $\mathsf{GEE2}$ satisfies soundness if EC satisfies strong correctness and the strong receiver binding property:

Theorem 8. *Suppose that* EC *satisfies strong correctness. Then, for any adversary* $\mathbf{A}$ *against* $\mathsf{GEE2}$ *for soundness, there exists some adversary* $\dot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\mathrm{snd}}_{\mathsf{GEE2}}(\mathbf{A}) \leq \mathrm{Adv}^{\text{sr-bind}}_{\mathsf{EC}}(\dot{\mathbf{A}}).$$

The run time of $\dot{\mathbf{A}}$ *is at most approximately 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 $\mathsf{GEE2}$. 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 $((L, A, M), (L', A, M'), B)$. Then, $\mathsf{ECver}(A, M, L, B) = \mathsf{ECver}(A, M', L', B) = 1$ since EC satisfies the strong correctness. □

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

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

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

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 $(i, A, C, B, S, H) \notin \mathcal{Y} \wedge i \notin \mathcal{Z}$ **then**
  $win \leftarrow \texttt{true}$
end if
return (M', L')

Fig. 18: MU-CTXT$^{\mathbf{A}}_{\mathsf{GEE2}}$

Unlinkability. GEE2 satisfies unlinkability if AE satisfies confidentiality:

Theorem 9. *For any adversary* $\mathbf{A}$ *against* GEE2 *for unlinkability making* q_{e} *queries to its* **Enc** *oracle,* q_{w} *queries to its* **Wrap** *oracle, and* q_{c} *queries to its* **ChalWrap** *oracle, there exists some adversary* $\dot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{mu-unlk}}_{\mathsf{GEE2}}(\mathbf{A}) \leq \mathrm{Adv}^{\text{mu-ror}}_{\mathsf{AE}}(\dot{\mathbf{A}}) + \frac{q_{\mathrm{e}}^2}{2^{\ell_{\mathrm{EC}}+1}}.$$

$\dot{\mathbf{A}}$ *makes at most* $(q_{\mathrm{w}} + q_{\mathrm{c}})$ *queries to its* **AEenc** *oracle. The run time of* $\dot{\mathbf{A}}$ *is at most approximately that of* MU-UNLK$^{\mathbf{A}}_{\mathsf{GEE2}}$.

The proof is omitted since it is similar to the proof of Theorem 4.

Robustness. GEE2 satisfies robustness if AE satisfies the context-committing property and EC satisfies the receiver-binding property:

Theorem 10. *Suppose that* EC *satisfies the strong correctness. Then, for any adversary* $\mathbf{A}$ *against* GEE2 *for robustness, there exist some adversaries* $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *such that*

$$\mathrm{Adv}^{\text{rbst}}_{\mathsf{GEE2}}(\mathbf{A}) \leq \mathrm{Adv}^{\text{ccmt}}_{\mathsf{AE}}(\dot{\mathbf{A}}) + \mathrm{Adv}^{\text{r-bind}}_{\mathsf{EC}}(\ddot{\mathbf{A}}).$$

The run times of $\dot{\mathbf{A}}$ *and* $\ddot{\mathbf{A}}$ *are at most approximately that of* $\mathbf{A}$.

Proof. Suppose that adversary $\mathbf{A}$ successfully breaks the robustness of $\mathsf{GEE2}$. Then, for an output $((K, A, C, B, S, H), (K', A', C', B', S', H'))$ of $\mathbf{A}$, $S = S'$, $(K, A, C, B) \neq (K', A', C', B')$, $\mathsf{G2Dec}(K, A, C, B, S, H) = (L, M) \neq \perp$, and $\mathsf{G2Dec}(K', A', C', B', S', H') = (L', M') \neq \perp$, where

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

Since $(K, A, C, B) \neq (K', A', C', B')$,

1. $(K, B) \neq (K', B')$ or
2. $(K, B) = (K', B')$ and $(A, C) \neq (A', C')$.

Suppose that $(K, B) \neq (K', B')$. Let $\dot{\mathbf{A}}$ be an adversary against AE for context-committing security. $\dot{\mathbf{A}}$ first runs $\mathbf{A}$ and gets a pair (K, A, C, B, S, H) and (K', A', C', B', S', H') produced by $\mathbf{A}$. $\dot{\mathbf{A}}$ computes (L, M) and (L', M'). Finally, $\dot{\mathbf{A}}$ outputs $((K, (H, B), L), (K', (H', B'), L'))$. Then, $\dot{\mathbf{A}}$ succeeds in breaking the context-committing security of AE. The run time of $\dot{\mathbf{A}}$ is at most approximately that of $\mathbf{A}$.

Suppose that $(K, B) = (K', B')$ and $(A, C) \neq (A', C')$. Let $\ddot{\mathbf{A}}$ be an adversary against EC for the receiver-binding property. $\ddot{\mathbf{A}}$ first runs $\mathbf{A}$ and gets a pair (K, A, C, B, S, H) and (K', A', C', B', S', H') produced by $\mathbf{A}$. $\ddot{\mathbf{A}}$ computes (L, M) and (L', M'). Finally, $\ddot{\mathbf{A}}$ outputs $((L, A, M), (L', A', M'), B)$. Then, $\ddot{\mathbf{A}}$ succeeds in breaking the receiver-binding property of EC. The run time of $\ddot{\mathbf{A}}$ is at most approximately that of $\mathbf{A}$. □

6 Conclusion

We have provided a formal treatment of envelope encryption in the symmetric key setting. We first defined its syntax and identified five security requirements: confidentiality, ciphertext integrity, soundness, unlinkability, and robustness. We then proposed two generic constructions. The first uses two AEAD schemes: one for encrypting data and the other for wrapping keys. The second employs encryptment for data encryption and AEAD for key wrapping. For both constructions, we proved that their security can be reduced to the security of the underlying primitives. Our formalization and constructions provide a theoretical basis for the secure deployment of envelope encryption, particularly in cloud-based applications and secure multicasting.

Acknowledgements. The first author was supported by JSPS KAKENHI Grant Number JP24K14944.

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 1, 22

2. Alibaba Cloud: Use envelope encryption to encrypt and decrypt local data. https://www.alibabacloud.com/help/en/kms/support/use-envelope-encryption-to-encrypt-and-decrypt-local-data (Accessed on 22/06/2025) 1
3. AWS Key Management Service: AWS KMS concepts. https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html (Accessed on 22/06/2025) 1
4. 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
5. 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 3
6. Bellare, M., Tackmann, B.: The multi-user security of authenticated encryption: AES-GCM in TLS 1.3. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I. Lecture Notes in Computer Science, vol. 9814, pp. 247–276. Springer (2016). https://doi.org/10.1007/978-3-662-53018-4_10 4
7. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Duplexing the sponge: Single-pass authenticated encryption and other applications. In: Miri, A., Vaudenay, S. (eds.) Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers. Lecture Notes in Computer Science, vol. 7118, pp. 320–337. Springer (2011). https://doi.org/10.1007/978-3-642-28496-0_19 3, 7
8. 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 3
9. 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 2, 3, 6, 7
10. Facebook: Messenger secret conversations. Technical Whitepaper (2016), https://about.fb.com/wp-content/uploads/2016/07/messenger-secret-conversations-technical-whitepaper.pdf 3

11. 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 3
12. Ganesan, R., Gobi, M., Vivekanandan, K.: A novel digital envelope approach for a secure e-commerce channel. International Journal of Network Security **11**(3), 121–127 (2010), http://ijns.jalaxy.com.tw/contents/ijns-v11-n3/ijns-2010-v11-n3-p121-127.pdf 3
13. Google Cloud: Envelope encryption. https://cloud.google.com/kms/docs/envelope-encryption (Accessed on 22/06/2025) 1
14. 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 3
15. 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 3
16. Hirose, S., Minematsu, K.: Compactly committing authenticated encryption using encryptment and tweakable block cipher. In: Carlet, C., Mandal, K., Rijmen, V. (eds.) Selected Areas in Cryptography - SAC 2023, 30th International Conference, Fredericton, Canada, August 14–18, 2023, Revised Selected Papers. Lecture Notes in Computer Science, vol. 14201, pp. 233–252. Springer (2023). https://doi.org/10.1007/978-3-031-53368-6_12 3
17. Hirose, S., Minematsu, K.: A formal treatment of envelope encryption. In: Kim, J., Park, J., Lee, W.K. (eds.) Information Security and Cryptology - ICISC 2024 - 27th International Conference, Seoul, South Korea, November 20–22, 2024, Revised Selected Papers. Lecture Notes in Computer Science, vol. 15596, pp. 96–110. Springer (2025). https://doi.org/10.1007/978-981-96-5566-3_6 2
18. IBM Cloud: Protecting your data with envelope encryption. https://cloud.ibm.com/docs/hs-crypto?topic=hs-crypto-envelope-encryption (Accessed on 22/06/2025) 1
19. 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 3
20. 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 3
21. 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 3
22. Krovetz, T., Rogaway, P.: The design and evolution of OCB. J. Cryptol. **34**(4), 36 (2021). https://doi.org/10.1007/s00145-021-09399-8 3
23. Kubernetes: Encrypting confidential data at rest. https://kubernetes.io/docs/tasks/administer-cluster/encrypt-data/ (Accessed on 22/06/2025) 1

24. 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 3
25. 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 3
26. Madden, N.: When a KEM is not enough. https://neilmadden.blog/2021/02/16/when-a-kem-is-not-enough/ (2021) 3
27. Microsoft Azure: Client-side encryption for blobs. https://learn.microsoft.com/en-us/azure/storage/blobs/client-side-encryption (Accessed on 22/06/2025) 1
28. Nir, Y., Langley, A.: ChaCha20 and Poly1305 for IETF protocols. RFC 8439 (2018). https://doi.org/10.17487/RFC8439 3
29. 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 3
30. 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 8
31. Pérez, S., Ramos, J.L.H., Pedone, D., Rotondi, D., Straniero, L., Skarmeta, A.F.: A digital envelope approach using attribute-based encryption for secure data exchange in iot scenarios. In: Global Internet of Things Summit, GIoTS 2017, Geneva, Switzerland, June 6-9, 2017. pp. 1–6. IEEE (2017). https://doi.org/10.1109/GIOTS.2017.8016281 3
32. 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 1, 3, 4
33. 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 4
34. Sosa, V.J.S., Morales-Sandoval, M., Telles-Hurtado, O., Compeán, J.L.G.: Protecting data in the cloud: An assessment of practical digital envelopes from attribute based encryption. In: Bernardino, J., Quix, C., Filipe, J. (eds.) Proceedings of the 6th International Conference on Data Science, Technology and Applications, DATA 2017, Madrid, Spain, July 24-26, 2017. pp. 382–390. SciTePress (2017). https://doi.org/10.5220/0006484603820390 3
35. Xiong, A., Xu, C.: Cloud storage access control scheme of ciphertext algorithm based on digital envelope. Intelligent Automation & Soft Computing **22**(2), 289–294 (2016). https://doi.org/10.1080/10798587.2015.1095488 3
36. Yandex Cloud: Envelope encryption. https://yandex.cloud/en/docs/kms/concepts/envelope (Accessed on 22/06/2025) 1