Abstract

Multi-receiver encryption allows a sender to choose a set of authorized receivers and send them a message securely and efficiently. Only one ciphertext corresponding to the message is generated regardless of the number of receivers. Thus it is practical and useful for video conferencing systems, pay-per-view channels, distance education, and so forth. In 2010, for further protecting receivers’ privacy, anonymous multi-receiver identity-based (ID-based) encryption was first discussed, and from then on, many works on the topic have been presented so far. To deal with the key escrow problem inherited from ID-based encryption (IBE), Islam et al. proposed the first anonymous multi-receiver certificateless encryption (AMRCLE) in 2014. In 2015, Hung et al. proposed a novel AMRCLE to improve the efficiency. However, we found that their security proofs are flawed, i.e., the simulation cannot be successfully performed. In this paper, we present a novel AMRCLE scheme with CCA security in confidentiality and anonymity against both Type I and Type II adversaries. Moreover, the identity of the sender of a ciphertext can be authenticated by the receiver after a successful decryption. To the best of our knowledge, the proposed scheme is the first CCA secure AMRCLE scheme, and furthermore, we also pioneer in achieving sender authentication in AMRCLE.

Index Terms
anonymity; multi-receiver encryption; chosen-ciphertext attacks; certificateless encryption

I. Introduction

Multi-receiver encryption is a practical and useful methodology for a sender to compute and transmit only one ciphertext corresponding to a message for multiple receivers. By using such encryption schemes, the communication cost is greatly decreased. Thus it is popular among some advanced services such as video conferencing, pay-per-view TV, and distance education. An important issue in such services is the authentication of the sender, that is, the source and legality of the digital products should be guaranteed. Many researchers focused on this topic and have proposed interesting results [1], [8], [26].

In some situations, such as ordering sensitive TV programs, anonymity of receivers might be required. In such environment, a customer may expect that her/his identity is not revealed when communication is proceeded. Motivated from this requirement, Fan et al. [9] first introduced the concept of anonymous multi-receiver ID-based encryption (AMRIBE) in 2010. Also, a multi-receiver ID-based encryption scheme using Lagrange interpolating polynomials is proposed in [9]. From then on many related results have been proposed [4], [5], [6], [7], [11], [13], [16], [18], [19], [20], [22], [23], [24], [25], [27], [28], [29], [30], [31], [32], [33]. Very recently, Fan et al.[10] proposed a novel type of AMRIBE with sender authentication called anonymous multi-receiver identity-based authenticated encryption. Their work is formally proved to be CCA security both in confidentiality and anonymity. Some analyses to the previous works have also been proposed in [10].

However, those schemes are constructed in ID-based cryptosystem [3], which means the key escrow problem exists in these schemes. Since the private keys of users’ are generated by a third party, key generation center (KGC), it can decrypt all the ciphertexts and reveal the identity of any receiver. Once the KGC is compromised, it would cause great damage to the system. It seems that the use of AMRIBE might be limited to small and closed groups with a fully trusted third authority. Therefore, in view of the aforementioned reasons, Islam et al. [14] proposed an anonymous multi-receiver certificateless encryption (AMRCLE) scheme in 2014. However, we found that their security proofs are flawed. In their security proofs, the simulator cannot successfully create the challenge ciphertext, and thus, it fails in the simulation. There is another AMRCLE has been proposed by Hung et al.[12] in 2015. They claimed that their scheme has better efficiency compared with Islam et al.’s scheme. Nevertheless, we found the their security proofs is unable to cover all possible attackers since some specific restriction is made to the attackers.

In this manuscript, the first AMRCLE scheme with CCA security is presented. Our work is based on the result proposed in [10]. The scheme of [10] is constructed on ID-Based setting, and thus the key escrow problem exists, where KGC can decrypt all the ciphertexts and get the identity of any cipherctxt receiver. Our work adopts the concept of certificateless cryptosystems instead of ID-Based setting, which makes it possible for the proposed scheme to be secure even against a malicious KGC. We provide formal proofs to demonstrate the proposed scheme as being CCA secure against both Type I and Type II attackers in the random oracle model [2]. Furthermore, it achieves sender authentication, which makes it possible for a receiver to confirm
the sender of a ciphertext after a successful decryption, such that it also is the first anonymous multi-receiver certificateless authenticated encryption (AMRCLAE) scheme in the literature.

II. Preliminaries

In this section, we define anonymous multi-receiver certificateless encryption and review some hard problems and assumption.

Definition 1. The definition of AMRCLAE was given by Islam et al. in [14]. In this paper, we define anonymous multi-receiver certificateless authenticated encryption (AMRCLAE). An AMRCLAE scheme consists of the following algorithms:

- **Setup** takes as input a security parameter \( l \). It returns a master secret key \( msk \) and system parameters \( \text{params} \).
- **PartialKeyExtract** takes as input the master secret key \( msk \) and a user’s identity \( ID_i \in \{0,1\}^* \), then returns the partial private key \( D_{i} \) of the user.
- **SetSecretValue** takes as input a user’s identity \( ID_i \) and outputs the secret value \( x_i \) of the user.
- **SetPublicKey** takes as input \( \text{params} \), a user’s identity \( ID_i \) and user’s partial private key \( D_{i} \). It returns the private key \( S_{i} \) of the user.
- **Encrypt** takes as input a message \( m \), a sender’s private key \( S_{s} \), and an identity list \( (ID_1,ID_2,\ldots,ID_n) \) and returns a ciphertext \( C \). We write \( C = \text{Encrypt}(\text{params},S_{s},(ID_1,ID_2,\ldots,ID_n),m) \).
- **Decrypt** takes as input a ciphertext \( C \) and the private key \( S_{i} \) of a user with identity \( ID_i \) and returns a message \( m \). We write \( m = \text{Decrypt}(\text{params},C,S_{i}) \).

Let \( G_1 \) and \( G_2 \) be two cyclic groups of prime order \( q \), \( P \) be a generator of \( G_1 \), and \( e : G_1 \times G_1 \to G_2 \) be a bilinear mapping.

Definition 2 (The Bilinear Diffie-Hellman (BDH) Problem). Given \( (P,aP,bP,cP) \) for some random \( a,b,c \in \mathbb{Z}_q^* \), compute \( e(P,P)^{abc} \).

Definition 3 (The Decisional Bilinear Diffie-Hellman (DBDH) Problem). Given \( (P,aP,bP,cP,Z) \) for some random \( a,b,c \in \mathbb{Z}_q^* \) and \( Z \in_R \{e(P,P)^{abc},Y \in_R G_2 \{e(P,P)^{abc}\} \} \), decide if \( Z = e(P,P)^{abc} \).

Definition 4 (The DBDH Assumption [3]). Define that an algorithm \( A \) with output \( \beta \in \{0,1\} \) has advantage \( \epsilon \) in solving the \( DBDH \) problem if

\[
\Pr[A(P,aP,bP,cP,e(P,P)^{abc}) = 1] - \Pr[A(P,aP,bP,cP,Z) = 1] \geq \epsilon
\]

where \( a,b,c \in_R \mathbb{Z}_q^* \) and \( Z \in_R \{e(P,P)^{abc},Y \in_R G_2 \{e(P,P)^{abc}\} \} \). We say that the \( DBDH \) assumption holds if no polynomial-time algorithm has non-negligible advantage in solving the \( DBDH \) problem.

Definition 5 (The Modified Decisional Bilinear Diffie-Hellman (M-DBDH) Problem [10]). Given \( (P,aP,bP,cP,e(P,P)^{b^r,c},Z) \) for some random \( a,b,c \in \mathbb{Z}_q^* \) and \( Z \in_R \{e(P,P)^{abc},Y \in_R G_2 \{e(P,P)^{abc}\} \} \), decide if \( Z = e(P,P)^{abc} \). Define that an algorithm \( A \) with output \( \beta \in \{0,1\} \) has advantage \( \epsilon \) in solving the \( M-DBDH \) problem if

\[
\Pr[A(P,aP,bP,cP,e(P,P)^{b^r,c},e(P,P)^{abc}) = 1] - \Pr[A(P,aP,bP,cP,e(P,P)^{b^r,c},Z) = 1] \geq \epsilon
\]

where \( a,b,c \in_R \mathbb{Z}_q^* \) and \( Z \in_R \{e(P,P)^{abc},Y \in_R G_2 \{e(P,P)^{abc}\} \} \).

Definition 6 (The M-DBDH Assumption [10]). We say that the \( M-DBDH \) assumption holds if no polynomial-time algorithm has non-negligible advantage in solving the \( M-DBDH \) problem.

III. Related Works

In 2014, Islam et al. proposed the first AMRCLAE scheme in order to deal with the key escrow problem [14]. They also claimed that their scheme is provably secure under the CDH assumption. In 2015, Hung et al. proposed a new AMRCLAE scheme with security proofs and better efficiency compared to [14]. In this section, we briefly review Islam et al.’s scheme and Hung et al.’s scheme.
A. Islam et al.’s scheme

1) Scheme description:

- **Setup**
  PKG performs the following operations:
  1) Generate a group $G$ with prime order $p$ and generator $P$.
  2) Choose an integer $s \in \mathbb{Z}_p^*$ randomly as the master secret key, and set $P_0 = sP$.
  3) Choose four cryptographic one-way hash functions, $H_0, H_1, H_2, H_3 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$.
  4) Select secure symmetric encryption/decryption function $E_{sk}(\cdot)/D_{sk}(\cdot)$, where $x$ is the symmetric key.
  5) Publish the system parameters $params = \{G, p, P, P_0, E_{sk}(\cdot)/D_{sk}(\cdot), H_0, H_1, H_2, H_3\}$ and keep the master key $s$ secret.

- **SetSecretValue**
  A user $i$ with identity $ID_i$ chooses $x_i \in \mathbb{Z}_p^*$ as his secret key and $P_i = x_iP$ as the corresponding public key.

- **PartialKeyExtract**
  User $i$ sends the tuple $(ID_i, P_i)$ to PKG. Then the PKG executes the following:
  1) Choose a number $t_i \in \mathbb{Z}_p^*$ and calculate $T_i = t_iP$.
  2) Calculate $l_i = H_0(ID_i, T_i, P_i)$ and $d_i = (t_i + s l_i) \mod p$.
  3) Send $(d_i, T_i)$ to user $ID_i$ through secure channel.

- **SetPrivateKey**
  $ID_i$ keeps $sk_i = (d_i, x_i)$ as his full private key.

- **SetPublicKey**
  $ID_i$ keeps $pk_i = (P_i, T_i)$ as his full public key.

- **Encrypt**
  A sender produces the ciphertext of a message by performing the following steps:
  1) Choose a message $m$ and select a set of $t$ receivers, whose identities are $\{ID_1, ..., ID_t\}$.
  2) Choose $r \in \mathbb{Z}_p^*$ and compute $\xi = rP$.
  3) For $i = 1$ to $t$, compute $l_i = H_0(ID_i, T_i, P_i), u_i = r(T_i + l_iP_0 + P_i)$, and $a_i = H_1(\xi, u_i)$.
  4) Select a $\theta \in \mathbb{Z}_p^*$ and compute $\psi(x) = \prod_{i=1}^{t}(x - a_i) + \theta = \sum_{i=0}^{t-1} c_i x^i \mod p$.
  5) Compute $\beta = H_2(\xi, \theta, \delta) = E_{\beta}(m)$ and $\gamma = H_3(c_0, c_1, ..., c_t-1, m, \theta, \xi, \delta)$.
  6) Set the ciphertext $(c_0, c_1, ..., c_t-1, \xi, \delta, \gamma)$.

- **Decrypt**
  After receiving the ciphertext $(c_0, c_1, ..., c_t-1, \xi, \delta, \gamma)$, a selected receiver, say $ID_i$, can decrypt the ciphertext as follows:
  1) Compute $u_i = (d_i + x_i)\xi$ and $a_i = H_1(\xi, u_i)$.
  2) Compute $\theta = \sum_{i=0}^{t-1} c_i (a_i)^i + (a_i)^t$.
  3) Compute $\beta = H_2(\xi, \theta, \delta) = D_{\beta}(\delta)$, and $\gamma' = H_3(c_0, c_1, ..., c_t-1, m, \theta, \xi, \delta)$.
  4) Accept $m$ if $\gamma = \gamma'$.

2) Discussion on the simulation of the CCA game for confidentiality against Type II adversary: In the Setup phase, the simulator $S$ sets $P_0 = aP$. When the adversary $A$ queries for $ID_i$’s partial private key, $S$ first chooses $d_i, l_i \in \mathbb{Z}_p^*$, and computes $T_i = d_iP - l_iP_0$, and sets $H_0(ID_i, T_i, P_i) = l_i$, where $P_i$ is the public key of $ID_i$. And then $S$ sets $d_i$ as the partial private key of $ID_i$. In the Challenge phase, $S$ needs to compute $u_j = b(T_j + P_j)$ for the target user $ID_j$, where $P_j = x_jP$. However, $b(T_j + P_j) = b(d_jP - l_jP_0 + x_jP) = (d_j + x_j)(bP) - l_j(abP)$. Since computing $abP$ is hard, $S$ cannot successfully generate the challenge ciphertext and fails the simulation. Same flaws also happen in the other proofs. Even if we use the DDH assumption as the underlying assumption, one can break the assumption through a bilinear mapping function.

B. Hung et al.’s scheme

1) Scheme description:

- **Setup**
  PKG performs the following operations:
  1) Generate two cyclic group $G_1, G_2$ with prime order $q$, bilinear mapping function $e : G_1 \times G_1 \rightarrow G_2$, and a generator $P$ of $G_1$.
  2) Choose an integer $s \in \mathbb{Z}_q^*$ randomly as the master secret key, and set $P_{pub} = sP$.
  3) Choose six cryptographic one-way hash functions, $H_0 : \{0,1\}^* \rightarrow G_1, H_1 : G_2 \times G_1 \rightarrow \{0,1\}^w, H_2, H_3, H_4 : \{0,1\}^w \rightarrow \{0,1\}^w$, and $H_5 : \{0,1\}^* \times G_1 \rightarrow \mathbb{Z}_q^*$ for some integer $w$.
  4) Select secure symmetric encryption/decryption function $E_{sk}(\cdot)/D_{sk}(\cdot)$, where $sk$ is the symmetric key.
  5) Publish the system parameters $params = \{G_1, G_2, e, q, P, P_{pub}, E_{sk}(\cdot), D_{sk}(\cdot), H_0, H_1, H_2, H_3, H_4, H_5\}$ and keep the master key $s$ secret.
![Image of a page from a document]

- **PartialKeyExtract**
  A user sends the his identity $ID$ to PKG. Then the PKG computes the partial private key $DID = s \cdot QID = sH_0(ID)$.

- **SetSecretValue**
  The user with identity $ID$ chooses $xid \in \mathbb{Z}_q^*$ as his secret value.

- **SetPublicKey**
  The user with identity $ID$ compute his public key as $PID = xid \cdot P$.

- **SetPrivateKey**
  The user with identity $ID$ keeps $SID = (DID, xid)$ as his private key.

- **Multiencryp**
  A sender produces the ciphertext of a message by performing the following steps:
  1) Choose a message $m$ and select a set of $t$ receivers with public key $(ID_1, PID_1), \ldots, (ID_t, PID_t)$.
  2) Choose $r \in \mathbb{Z}_q^*$ and compute $U = rP$ and $F_i = rPID_i$ for $i = 1, \ldots, t$.
  3) For $i = 1$ to $t$, compute $QID_i = H_0(ID_i), K_i = e(P_{pub}, QID_i)^r$, and $T_i = H_1(K_i, F_i)$.
  4) Pick an ephemeral value $\sigma \in \{0, 1\}^w$ at random and compute $C_i = H_2(T_i) \parallel (H_3(T_i) \oplus \sigma)$ for $i = 1, \ldots, t$.
  5) Compute $V = E_{H_3(\sigma)}(m)$ and $\Lambda = H_5(m, \sigma, C_1, C_2, \ldots, C_t, V, U)$.
  6) Set the ciphertext $CT = (C_0, C_1, \ldots, C_t, V, U, \Lambda)$.

- **Decrypt**
  After receiving the ciphertext $CT = (C_0, C_1, \ldots, C_t, V, U, \Lambda)$, a selected receiver, say $ID$, with full private key $(DID, xid)$, can decrypt the ciphertext as follows.
  1) Compute $K = e(U, DID), F = xid \cdot U, T = H_1(K, F)$, and $H_2(T)$.
  2) Use $H_2(T)$ to find its associated $C_i$ by the relation $C_i = H_2(T) \parallel W$, where $W$ denotes the remaining strings by removing $H_2(T)$ from $C_i$.
  3) Compute $\sigma' = W \oplus H_3(T)$ and $m' = D_{H_3(\sigma')}(V)$.
  4) Accept $m$ if $\Lambda = H_5(m', \sigma', C_1, \ldots, C_t, V, U)$.

2) **Discussion on the simulation of the CCA game for confidentiality against Type I adversary:** In the proof of Theorem 1, the confidentiality against Type I adversary, the authors made an assumption that if the adversary $A$ wins the game, then must have queried to the oracle $H_i$ with some specific inputs $(K, F)$, such that $BDDH(P, QID_i, P_{pub}, U^*, K) = 1$. With the specific inputs from $A$, the challenger is able to solve the Gap-BDH problem by computing $e(P, P)^{abc} = K^{u_1 \cdot l}$. Since $U^*$ is set to be $eP$ in the challenge phase, if $A$ is able to compute $K$, it implies $A$ is able to computes the symmetry key $sk$ used to encrypt $m_{\beta}$, and thus $A$ wins the game. However, the proof only aims at the attackers who are capable of getting the key, $sk$, before winning the CCA game. The authors have not considered the attackers who can win the game without getting the key. As a result, their proof does not cover all possible attackers. Besides, the same problem exists in the proof for Theorem 2, Theorem 3, Theorem 4, too.

IV. An Anonymous Multi-Receiver Certificataless Authenticated Encryption Scheme

In this section, we propose an AMRCLAE scheme with provable CCA security in both confidentiality and anonymity against Type I and Type II attackers. The notations are shown in Table I.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G_1$</td>
<td>a cyclic additive group of prime order $q$</td>
</tr>
<tr>
<td>$G_2$</td>
<td>a cyclic multiplicative group of prime order $q$</td>
</tr>
<tr>
<td>$e$</td>
<td>a bilinear mapping; $e : G_1 \times G_1 \rightarrow G_2$</td>
</tr>
<tr>
<td>$P$</td>
<td>a generator of $G_1$</td>
</tr>
<tr>
<td>KGC</td>
<td>the key generation center</td>
</tr>
<tr>
<td>$P_{pub}$</td>
<td>the public key of KGC</td>
</tr>
<tr>
<td>$M$</td>
<td>a message</td>
</tr>
<tr>
<td>$ID_i$</td>
<td>the identity of user $i$</td>
</tr>
<tr>
<td>$Q_i$</td>
<td>the hashed value of $ID_i$</td>
</tr>
<tr>
<td>$d_i$</td>
<td>the partial private key of $ID_i$</td>
</tr>
<tr>
<td>$P_i$</td>
<td>the public key of $ID_i$</td>
</tr>
<tr>
<td>$S_i$</td>
<td>the private key of $ID_i$</td>
</tr>
</tbody>
</table>

The proposed AMRCLAE scheme is described as follows.

- **Setup**
  KGC performs the following operations:
  1) Choose an integer $\alpha \in \mathbb{Z}_q^*$ randomly as the master secret key, and set $P_{pub} = \alpha P$. 

---

**TABLE I**

The Notations
2) Choose three cryptographic one-way hash functions, $H : \{0,1\}^* \rightarrow G_1$, $H_1 : G_2 \times G_2 \rightarrow \mathbb{Z}_q^*$, and $H_2 : G_2 \times \mathbb{Z}_q^* \rightarrow \mathbb{Z}_q^*$.
3) Compute $\Omega = e(P, P)$.
4) Publish the system parameters $\text{params} = \{G_1, G_2, e, q, P, P_{pub}, H, H_1, H_2, \Omega\}$ and keep the master key $\alpha$ secret.

- **PartialKeyExtract**
  
  When user $i$ joins the system, KGC will compute $Q_i = H(ID_i)$ and the partial private key $d_i = \alpha Q_i$ of the user, and then KGC will send $d_i$ to user $i$ in a secure manner.

- **SetSecretValue**

  A user with identity $ID_i$ randomly chooses $x_i \in \mathbb{Z}_q^*$ as his secret value.

- **SetPrivateKey**

  A user with identity $ID_i$ sets his private key $S_i = (d_i, x_i)$.

- **SetPublicKey**

  A user with identity $ID_i$ computes $P_i = x_i P$ as his public key.

- **Encrypt**

  A sender, whose identity is $ID_s$, produces the ciphertext of a message by performing the following steps:

  1) Choose a message $M \in G_2$ and select a set of $t$ receivers, whose identities are $\{ID_1, \ldots, ID_t\}$.
  2) Choose $k \in \mathbb{Z}_q^*$ at random and compute $r = H_2(M, k)$.
  3) For $i = 1$ to $t$, compute $Q_i = H(ID_i)$ and $v_i = H_1(e(rQ_i, d_s), e(rQ_i, x_s P))$, where $P_i$ is the public key of the receiver with identity $ID_i$, and $(d_s, x_s)$ is the private key of the sender.
  4) Compute $f(x) = \sum_{i=0}^{t-1} c_i x^i + x^t \mod q$.
  5) Compute $U = r P, U_i = r P_i, V = r Q_s$, and $W = M \cdot \Omega^K$, where $P_s$ is the public key of the sender.
  6) Set the ciphertext $C = (c_0, c_1, \ldots, c_{t-1}, U, U_1, V, W, ID_s)$.

- **Decrypt**

  After receiving the ciphertext $C = (c_0, c_1, \ldots, c_{t-1}, U, U_1, V, W, ID_s)$, a selected receiver, say $ID_i$, can decrypt $C$ as follows.

  1) Compute $v_i' = H_1(e(V, d_i), e(x_i Q_i, U_1))$.
  2) Compute $k' = f(v_i') = \sum_{j=0}^{t-1} c_j (v_i')^j + (v_i')^t \mod q$.
  3) Compute $M' = W/\Omega^{k'}$.
  4) Accept $M'$ if $U = H_2(M', k') P$. Otherwise, output 1. To authenticate the sender, the decryptor can verify if $e(U, H(ID_s)) = e(V, P)$ and $e(U, P_s) = e(U_1, P)$.

The correctness of encryption and decryption is demonstrated as follows.

\[
\begin{align*}
v_i' &= H_1(e(V, d_i), e(x_i Q_i, U_1)) \\
    &= H_1(e(rQ_i, \alpha Q_i), e(x_i Q_i, r x_s P)) \\
    &= H_1(e(rQ_i, \alpha Q_i), e(rQ_i, x_s x_i P)) \\
    &= H_1(e(rQ_i, d_s), e(rQ_i, x_s P)) \\
    &= v_i,
\end{align*}
\]

and

\[
k' = f(v_i') = f(v_i) = k.
\]

Thus, the selected receiver $ID_i$ can successfully recover the message by computing $M' = W/\Omega^{k'} = W/\Omega^K = M$, and he will accept the message due to $U = H_2(M, k)P = H_2(M', k') P$. After successfully recovering the message, it follows that $e(V, d_i) = e(rH(ID') \cdot \alpha H(ID_i)) = e(rH(ID_i), d')$ for some identity $ID'$. The receiver can be convinced that the ciphertext is encrypted with the private key of some valid user $ID'$, where $V = rH(ID')$. If $e(U, H(ID_s)) = e(V, P)$, $V = rH(ID_s)$, which implies $ID' = ID_s$. Similarly, $e(x_i Q_i, U_1) = e(rQ_i, x' P)$ for some user’s secret value $x'$ such that $U_1 = r(x' P)$. Once $e(U, P_s) = e(U_1, P), U_1 = r P_s = r(x_s P)$, which implies $x' = x_s$. Thus, the sender is authenticated.

V. Security Models and Proofs

In this section, we will define the security models and the security notions for AMRCLEAE with sender authentication. The security notions are "Indistinguishability of encryptions under selective multi-ID, chosen-ciphertext attacks" (IND-sMID-CCA) and "Anonymous indistinguishability of encryptions under selective multi-ID, chosen-ciphertext attacks" (Anon-sMID-CCA). There are two types of adversary in the definition of our security model. A Type I adversary is able to replace the public key of any user, but not able to access the master secret key $msk$. A Type II adversary has the ability to access the master secret key $msk$ but is not allowed to replace the public keys of users. The Type II adversary models security against a malicious KGC and the Type I adversary models security against a malicious user. We will prove that our proposed scheme is CCA secure in confidentiality and anonymity against Type I and Type II adversaries.
A. Confidentiality

Definition 7 (The IND-sMID-CCA Game I). Let \( A \) be a polynomial-time Type I attacker. \( A \) interacts with a simulator \( S \) in the following game.

Initialization. \( A \) chooses a set of identities \( ID^* = \{ID_1^*, ID_2^*, ..., ID_t^*\} \) and sends \( ID^* \) to \( S \).
Setup. \( S \) runs the Setup algorithm to generate \( \text{params} \) and \( \text{msk} \). \( S \) then sends \( \text{params} \) and \( \text{msk} \) to \( A \).

Phase 1. \( A \) issues the following queries.
- Hash query: \( S \) operates hash functions on the inputs given by \( A \) and returns the hashed values.
- PartialKeyExtract\((ID)\): \( A \) sends an identity \( ID \) to \( S \) and \( S \) returns the partial private key of \( ID \) where \( \text{PartialKeyExtract}(ID) \) cannot be queried if \( ID \in ID^* \).
- SecretValue\((ID)\): \( A \) sends an identity \( ID \) to \( S \) and \( S \) returns the secret value \( x_i \) of \( ID \).
- PublicKey\((ID)\): \( A \) sends an identity \( ID \) to \( S \) and \( S \) returns the public key \( P_i \) of \( ID \).
- KeyReplacement\((ID, P'_i, x'_i)\): \( A \) sends an identity \( ID \), the public key, and the secret value to \( S \). \( S \) then replaces the public key and the secret value of \( ID \) with \( P'_i \) and \( x'_i \), respectively.
- Encrypt\((ID_s, ID_1^*, ..., ID_a^*, M)\): \( A \) sends a sender’s identity \( ID_s \), a receiver set \( \{ID_1^*, ..., ID_a^*\} \), and a message \( M \) to \( S \). \( S \) returns a ciphertext \( C \) to \( A \).
- Decrypt\((C, ID)\): \( A \) sends an identity \( ID \) and a ciphertext \( C \) to \( S \) and \( S \) returns the result of the decryption.

Challenge. \( A \) submits a sender’s identity \( ID_s \) and \( (M_0, M_1) \) to \( S \) where \( M_0, M_1 \) are two distinct messages of the same length and \( ID_s \notin ID^* \). \( S \) then randomly chooses \( \beta \in \{0, 1\} \) and generates \( C^* = \text{Encrypt}(ID_s, ID_1^*, ..., ID_a^*, M_\beta) \). Finally, \( S \) sends \( C^* \) to \( A \).

Phase 2. \( A \) issues the queries defined in Phase 1 except for issuing Decrypt queries with \( C = C^* \) and \( ID_i \in ID^* \).

Guess. Finally, \( A \) outputs \( \beta' \in \{0, 1\} \) and wins the game if \( \beta' = \beta \).

The advantage of \( A \) winning the game is defined as

\[
\text{Adv}_{\text{IND-sMID-CCA-I}}^A = \left| Pr[\beta' = \beta] - \frac{1}{2} \right|.
\]

An AMRCLAE scheme is said to be IND-sMID-CCA secure against Type I adversary if there exists no polynomial-time Type I attacker that can win the IND-sMID-CCA game I with non-negligible advantage.

Definition 8 (The IND-sMID-CCA Game II). Let \( A \) be a polynomial-time Type II attacker. \( A \) interacts with a simulator \( S \) in the following game.

Initialization. \( A \) chooses a set of identities \( ID^* = \{ID_1^*, ID_2^*, ..., ID_t^*\} \) and sends \( ID^* \) to \( S \).
Setup. \( S \) runs the Setup algorithm to generate \( \text{params} \) and \( \text{msk} \). \( S \) then sends \( \text{params} \) and \( \text{msk} \) to \( A \).

Phase 1. \( A \) issues the following queries.
- Hash query: \( S \) operates hash functions on the inputs given by \( A \) and returns the hashed values.
- SecretValue\((ID)\): \( A \) sends an identity \( ID \) to \( S \) and \( S \) returns the secret value \( x_i \) of \( ID \) where \( \text{SecretValue}(ID) \) cannot be queried if \( ID \in ID^* \).
- PublicKey\((ID)\): \( A \) sends an identity \( ID \) to \( S \) and \( S \) returns the public key \( P_i \) of \( ID \).
- KeyReplacement\((ID, P'_i, x'_i)\): \( A \) sends an identity \( ID \), the public key, and the secret value to \( S \). \( S \) then replaces the public key and the secret value of \( ID \) with \( P'_i \) and \( x'_i \), respectively.
- Encrypt\((ID_s, ID_1^*, ..., ID_a^*, M)\): \( A \) sends a sender’s identity \( ID_s \), a receiver set \( \{ID_1^*, ..., ID_a^*\} \), and a message \( M \) to \( S \). \( S \) returns a ciphertext \( C \) to \( A \).
- Decrypt\((C, ID)\): \( A \) sends an identity \( ID \) and a ciphertext \( C \) to \( S \) and \( S \) returns the result of the decryption.

Challenge. \( A \) submits a sender’s identity \( ID_s \) and \( (M_0, M_1) \) to \( S \) where \( M_0, M_1 \) are two distinct messages of the same length and \( ID_s \notin ID^* \). \( S \) then randomly chooses \( \beta \in \{0, 1\} \) and generates \( C^* = \text{Encrypt}(ID_s, ID_1^*, ..., ID_a^*, M_\beta) \). Finally, \( S \) sends \( C^* \) to \( A \).

Phase 2. \( A \) issues the queries defined in Phase 1 except for issuing Decrypt queries with \( C = C^* \) and \( ID_i \in ID^* \).

Guess. Finally, \( A \) outputs \( \beta' \in \{0, 1\} \) and wins the game if \( \beta' = \beta \).

The advantage of \( A \) winning the game is defined as

\[
\text{Adv}_{\text{IND-sMID-CCA-II}}^A = \left| Pr[\beta' = \beta] - \frac{1}{2} \right|.
\]

It is not necessary to simulate \( \text{PartialKeyExtract} \) oracle of \( A \) since \( A \) has \( \text{msk} \). An AMRCLAE scheme is said to be IND-sMID-CCA secure against Type II adversary if there exists no polynomial-time Type II attacker that can win the IND-sMID-CCA game II with non-negligible advantage.
The main idea of the proof is proof by contradiction. Assume that the proposed scheme is not IND-sMID-CCA secure against Type I adversary, i.e., there exists a polynomial-time Type I adversary \( \mathcal{A} \) that wins the IND-sMID-CCA game I with non-negligible advantage. Then we will construct a polynomial-time algorithm \( S \) that has non-negligible advantage in solving the M-DBDH problem.

First, \( S \) is given \( < q, G_1, G_2, e, P, aP, bP, cP, e(P, P)^{bP}, Z > \) which is an instance of the M-DBDH problem. \( S \) simulates the game for \( \mathcal{A} \) as follows:

Initialization. \( S \) outputs a target identity set \( ID^* = \{ ID_1^*, \ldots, ID_t^* \} \).

Setup. \( S \) sets \( P_{pub} = cP \), computes \( \Omega = e(P, P) \), and outputs \( \{ G_1, G_2, e, q, P, P_{pub}, H, H_1, H_2, \Omega \} \) as the public parameters where \( H, H_1, \) and \( H_2 \) are three random oracles controlled by \( S \).

Phase 1. \( S \) maintains \( H \)-list, \( H_1 \)-list, and \( H_2 \)-list to store the results of querying \( H, H_1, \) and \( H_2, \) respectively. In this phase \( \mathcal{A} \) can issue the following queries:

- \( H \)-query:
  This oracle takes an identity \( ID_j \in \{0,1\}^* \) as input. If there exists a record \((ID_j, Q_j, q_j)\) in \( H \)-list, return \( Q_j \). Otherwise, do the following:
  1) Randomly select \( q_j \in \mathbb{Z}_q^* \).
  2) If \( ID_j \in ID^* \), compute \( Q_j = q_j(bP) \); else \( Q_j = q_jP \).
  3) Return \( Q_j \) and add \((ID_j, Q_j, q_j)\) into \( H \)-list.

- \( H_1 \)-query:
  This oracle takes \((X_j, Y_j)\) as input, where \( X_j, Y_j \in G_2 \). If there exists a record \((X_j, Y_j, v_j)\) in \( H_1 \)-list, return \( v_j \). Otherwise, do the following:
  1) Randomly choose \( v_j \in \mathbb{Z}_q^* \).
  2) Add \((X_j, Y_j, v_j)\) to \( H_1 \)-list.
  3) Return \( v_j \).

- \( H_2 \)-query:
  This oracle takes \( M_j \in G_2 \) and an integer \( k_j \in \mathbb{Z}_q^* \) as input. If there exists a record \((M_j, k_j, r_j, U_j)\) in \( H_2 \)-list, return \( r_j \). Otherwise, do the following:
  1) Randomly choose \( r_j \in \mathbb{Z}_q^* \) and compute \( U_j = r_jP \).
  2) Add \((M_j, k_j, r_j, U_j)\) to \( H_2 \)-list.
  3) Return \( r_j \).

- PartialKeyExtract:
  This oracle takes an identity \( ID_j \) as input. Call \( H(ID_j) \) and retrieve \( q_j \) form \( H \)-list. Then, \( S \) does the following:
  – If \( ID_j \in ID^* \), return “reject”.
  – Otherwise, compute \( d_j = q_j(eP) \) and return \( d_j \).

- PublicKey:
  This oracle takes an identity \( ID_j \) as input. If there exists \((ID_j, P_j, x_j)\) in \( pk \)-list, return \( P_j \). Otherwise, do the following:
  1) Choose \( x_j \in \mathbb{Z}_q^* \).
  2) Compute \( P_j = x_jP \).
  3) Add \((ID_j, P_j, x_j)\) to \( pk \)-list.
  4) Return \( P_j \).

- SecretValue:
  This oracle takes an identity \( ID_j \) as input. If there exists \((ID_j, P_j, x_j)\) in \( pk \)-list, return \( x_j \). Otherwise, do the following:
  1) Choose \( x_j \in \mathbb{Z}_q^* \).
  2) Compute \( P_j = x_jP \).
  3) Add \((ID_j, P_j, x_j)\) to \( pk \)-list.
  4) Return \( x_j \).

- KeyReplacement:
  \( \mathcal{A} \) may issue this query with input \((ID_j, P'_j, x'_j)\). \( S \) then replaces the record \((ID_j, P_j, x_j)\) in \( pk \)-list with \((ID_j, P'_j, x'_j)\).

- Encrypt:
  This oracle takes \( u + 1 \) identities \((ID_s, ID_1, \ldots, ID_u)\) and a message \( M \) as input. Upon receiving an Encrypt query, \( S \) does the following:
1) Choose \( k, r \in \mathbb{Z}_q^* \) at random and set \( H_2(M, k) = r \).

2) For \( i = 1 \) to \( u \),
   - if \( ID_s \notin ID^* \), compute \( v_i = H_1(e(Q_1, d_s)^r, e(Q_1, P_s)^{r\cdot x_i}) \), where \((d_s, x_i)\) is the private key of the sender \( ID_s \);
   - if \( ID_s \in ID^* \) and \( ID_i \notin ID^* \), compute \( v_i = H_1(e(d_i, Q_s)^r, e(Q_s, P_s)^{r\cdot x_i}) \), where \((d_i, x_i)\) is the private key of the receiver \( ID_i \) and \( P_s \) is the public key of \( ID_s \);
   - if \( ID_s, ID_i \in ID^* \), set \( T = e(P, P)^{b_i\cdot c} \); compute \( v_i = H_1(T^{r\cdot q \cdot \beta}, e(Q_i, P_i)^{r\cdot x_i}) \).

3) Compute \( f(x) = \prod_{i=1}^{u}(x - v_i) + k = \sum_{i=0}^{u-1} c_i x^i + x^u \mod q \).

4) Compute \( U = rP, U_1 = rP_s, V = rQ_s, \) and \( W = M \cdot \Omega^k \).

5) Set the ciphertext \( C = (c_0, c_1, \ldots, c_{u-1}, U, U_1, V, W, ID_s) \) and return \( C \).

- Decrypt:

  This oracle takes an identity \( ID_i \) and a ciphertext \( C \) as input. Upon receiving a Decrypt query, denoted by Decrypt\((C, ID_j)\) where \( C = (c_0, \ldots, c_{u-1}, U, U_1, V, W, ID_s) \), \( S \) does the following:

1) Search \( H_2 \)-list to get \((M_i, k_i, r_i, U_i)\) with \( U_i = U \). If not found, return "reject".

2) Search \( H \)-list to get \((ID_s, Q_s, q_s)\) with \( e(U, Q_s) = e(P, V) \). If not found, return "reject".

3) This step can be separated into three cases:
   - if \( ID_s \notin ID^* \), compute \( v_j = H_1(e(Q_j, d_s)^r, e(Q_j, P_s)^{r\cdot x_j}) \);
   - if \( ID_s \in ID^* \) and \( ID_j \notin ID^* \), compute \( v_j = H_1(e(d_j, Q_s)^r, e(Q_j, P_s)^{r\cdot x_j}) \);
   - if \( ID_s, ID_j \in ID^* \), set \( T = e(P, P)^{b_j\cdot c} \) and compute \( v_j = H_1(T^{r\cdot q \cdot \beta}, e(Q_j, P_j)^{r\cdot x_j}) \).

4) Compute \( k = c_0 + c_1 v_j + \cdots + c_{u-1} v_j^{u-1} + v_j^u \mod q \).

5) Check whether \( k_i = k \) and \( M_i = W/\Omega^k \) or not. If not, return "reject". Otherwise, return \( M_i \).

**Challenge.** \( A \) sends \((M_0, M_1)\) and \( ID_s \) to \( S \) where \( M_0, M_1 \in G_2 \) are two distinct messages with the same length and \( ID_s \notin ID^* \). \( S \) performs the following operations:

1) Choose \( \beta \in \{0, 1\} \) randomly.

2) For \( i = 1 \) to \( t \), call \( H(ID_i) \) and retrieve \( q_i^* \) from \( H \)-list.

3) Call \( H(ID_s) \) and retrieve \( q_s \) from \( H \)-list.

4) Search \((P_i^*, x_i^*)\) for \( i = 1, \ldots, t \) and \((P_s, x_s)\) in the \( pk \)-list.

5) Choose \( k \in \mathbb{Z}_q^* \) and set \( U^* = aP, U_1^* = x_s(aP), \) and \( V^* = q_s(aP) \).

6) For \( i = 1 \) to \( t \), compute \( v_i = H_1(Z^{q_i^*}, e(q_i^*(bP), x_s x_i^*(aP))) \).

7) Compute \( f(x) = \prod_{i=1}^{t}(x - v_i) + k = \sum_{i=0}^{t-1} c_i x^i + x^t \mod q \) and \( W^* = M_\beta \cdot \Omega^k \).

8) Set the ciphertext \( C^* = (c_0, c_1, \ldots, c_{t-1}, U^*, U_1^*, V^*, W^*, ID_s) \) and send \( C^* \) to \( A \).

**Phase 2.** \( A \) makes queries as those in **Phase 1**. However, if \( A \) issues a Decrypt query with input \( C = C^* \) and \( ID_i \in ID^* \), \( S \) will return "reject".

**Guess.** Finally, \( A \) outputs \( \beta' \in \{0, 1\} \). If \( \beta' = \beta \), then \( S \) outputs 1. Otherwise, \( S \) randomly chooses \( \beta \in \{0, 1\} \) and outputs \( \beta \).

If \( Z = e(P, P)^{abc} \), then \( Z^{q_i^*} = e(P, P)^{abc q_i^*} = e(q_i^*(bP), q_s(cP)) = e(Q_s^*, d_s)^a \) for \( i = 1 \) to \( t \). Therefore, \( C^* \) is a correct ciphertext. Otherwise, \( Z \) is an element randomly chosen in \( G_2 \). As the construction above, \( S \) correctly simulates the IND-sMID-CCA game I. If \( A \) wins the IND-sMID-CCA game with non-negligible advantage at least \( \epsilon \), \( |Pr[\beta' = \beta] - \frac{1}{2}| \geq \epsilon \) under a correct simulation of the game, i.e., \( |Pr[\lambda(A) = \beta'] - \frac{1}{2}| \geq \epsilon \), where \( \lambda(A) \) is a correct AMRCLAE scheme. Thus, we have that

\[
Pr[S(P, aP, bP, cP, e(P, P)^{b\cdot c}, e(P, P)^{abc}) = 1] = Pr[\lambda(A) = \beta] + \frac{1}{2}(1 - Pr[\lambda(A) = \beta])
\]

\[
= \frac{1}{2} Pr[\lambda(A) = \beta] + \frac{1}{2}
\]

and

\[
Pr[S(P, aP, bP, cP, e(P, P)^{b\cdot c}, Z) = 1]
\]

\[
= \frac{1}{2} Pr[S(P, aP, bP, cP, e(P, P)^{b\cdot c}, e(P, P)^{abc}) = 1] + \frac{1}{2} Pr[S(P, aP, bP, cP, e(P, P)^{b\cdot c}, X \in R G_2 \{\{e(P, P)^{abc}\}) = 1]
\]

\[
= \frac{1}{2} \left( \frac{1}{2} Pr[\lambda(A) = \beta] + \frac{1}{2} \right) + \frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)
\]

\[
= \frac{5}{8}.
\]
We can obtain
\[
Pr\{S(P, aP, bP, cP, e(P, P)^{b^2c}, e(P, P)^{abc}) = 1\}
- Pr\{S(P, aP, bP, cP, e(P, P)^{b^2c}, Z) = 1\}
= \frac{1}{4} Pr\{A(\Omega) = \beta\} - \frac{1}{8} \geq \frac{1}{4} \geq \epsilon.
\]

Therefore, \(S\) solves the M-DBDH problem with non-negligible advantage \(\frac{\epsilon}{4}\) within polynomial time. \(\square\)

Theorem 1 ensures the CCA security of confidentiality against Type I adversary. In confidentiality, there do not exist inside attackers (selected receivers) because every selected receiver can decrypt the ciphertext.

**Theorem 2.** (Confidentiality) The proposed AMRCLAE scheme is IND-sMID-CCA secure against Type II adversary in the random oracle model if the M-DBDH assumption holds.

**Proof.** First, \(S\) is given \(<q,G_1,G_2,e,P,aP,bP,cP, e(P, P)^{b^2c}, Z\>\). \(S\) simulates the game for an adversary \(A\) as follows:

**Initialization.** \(A\) outputs a target identity set \(ID^* = \{ID^*_1,\ldots, ID^*_t\}\).

**Setup.** \(S\) sets \(P_{pub} = \alpha P\), where \(\alpha \in \mathbb{Z}^*_q\), and sends \(\alpha\) to \(A\). The rest of the step is the same as that of the proof of Theorem 1

**Phase 1.** \(S\) maintains \(H\)-list, \(H_1\)-list, and \(H_2\)-list and \(A\) can issue the following queries:
- \(H\)-query:
  This oracle is identical to that of the proof of Theorem 1 if we replace “\(q_j(bP)\)” by “\(q_j(cP)\)”.
- \(H_1\)-query:
  This oracle is identical to that of the proof of Theorem 1.
- \(H_2\)-query:
  This oracle is identical to that of the proof of Theorem 1.
- **PublicKey:**
  This oracle is identical to that of the proof of Theorem 1 except step 2) shown below.
- **SecretValue:**
  This oracle takes an identity \(ID_j\) as input. If \(ID_j \in ID^*\), \(S\) returns “reject”. If \((ID_j, P_j, x_j)\) is in \(pk\)-list, return \(x_j\). Otherwise, \(S\) simulates this oracle as that of the proof of Theorem 1.
- **KeyReplacement:**
  \(A\) may issue this query with input \((ID_j, P'_j, x'_j)\). If \(ID_j \in ID^*\), \(S\) return “reject”. \(S\) then replaces the record \(ID_j, P_j, x_j\) in \(pk\)-list with \((ID_j, P'_j, x'_j)\).
- **Encrypt:**
  This oracle is the same as that of the proof of Theorem 1 except the following.
  - if \(ID_s, ID_t \in ID^*\), set \(T = e(P, P)^{b^2c}\) and compute \(v_i = H_1(e(Q_i, d_s)^r, T^{rx, x_i q_i})\).
- **Decrypt:**
  This oracle is the same as that of the proof of Theorem 1 except the following.
  - if \(ID_s, ID_t \in ID^*\), set \(T = e(P, P)^{b^2c}\) and compute \(v_j = H_1(e(Q_j, d_s)^{r_i}, T^{rx, x_j q_i})\).

**Challenge.** This step is the same as that of the proof of Theorem 1 except the following.
6) For \(i = 1\) to \(t\), compute \(v_i = H_1(e(Q'_i, \alpha q_s(aP)), Z^{r_i x, x_i})\).

**Phase 2 and Guess.** The two steps are identical to those of the proof of Theorem 1.

If \(Z = e(P, P)^{abc}\), then \(C^*\) is a correct ciphertext and \(S\) correctly simulates the IND-sMID-CCA game II. If \(A\) wins the IND-sMID-CCA game II with non-negligible advantage \(\epsilon\), \(Pr\{A(\Omega) = \beta' = \beta\} - \frac{1}{2} \geq \epsilon\), where \(\Omega\) is a correct AMRCLAE scheme. Thus, we have that
\[
Pr\{S(P, aP, bP, cP, e(P, P)^{b^2c}, e(P, P)^{abc}) = 1\}
- Pr\{S(P, aP, bP, cP, e(P, P)^{b^2c}, Z) = 1\}
= \frac{1}{4} Pr\{A(\Omega) = \beta\} - \frac{1}{8} \geq \frac{1}{4} \geq \epsilon.
\]

Thus, \(S\) solves the M-DBDH problem with non-negligible advantage \(\frac{\epsilon}{4}\) within polynomial time. \(\square\)

Theorem 2 ensures the CCA security of confidentiality against Type II adversary.
B. Anonymity

Definition 9 (The Anon-sMID-CCA Game I). Let \( A \) be a polynomial-time Type I attacker. \( A \) interacts with a simulator \( S \) in the following game.

Initialization. \( A \) chooses two identities \( ID^* = \{ID^*_0, ID^*_1\} \) and sends \( ID^* \) to \( S \).
Set up and Phase 1. These two phases are the same as those of Definition 7.
Challenge. \( A \) submits a sender’s identity \( ID_s \), a message \( M \), and a set of identities \( \{ID_2, ID_3, \ldots, ID_t\} \) to \( S \), with restrictions that \( ID_s \notin ID^* \) and \( \text{PartialKeyExtract}(ID_s) \) has not been queried before. \( S \) then randomly chooses \( \beta \in \{0,1\} \) and generates \( C^* = \text{Encrypt}(ID_s, ID^*_\beta, ID_2, \ldots, ID_t, M) \). Finally, \( S \) sends \( C^* \) to \( A \).
Phase 2. \( A \) issues the queries defined in Phase 1 except for issuing \( \text{Decrypt}(C^*, ID^*_0) \), \( \text{Decrypt}(C^*, ID^*_1) \), and \( \text{PartialKeyExtract}(ID_s) \).
Guess. Finally, \( A \) outputs \( \beta' \in \{0,1\} \) and wins the game if \( \beta' = \beta \).

The advantage of \( A \) winning the game is defined as

\[
\text{Adv}^{\text{Anon-sMID-CCA-I}}(A) = \left| \Pr[\beta' = \beta] - \frac{1}{2} \right|
\]

An AMRCLAE scheme is said to be Anon-sMID-CCA secure against Type I adversary if there exists no polynomial-time Type I attacker that can win the Anon-sMID-CCA game I with non-negligible advantage.

Definition 10 (The Anon-sMID-CCA Game II). Let \( A \) be a polynomial-time Type II attacker. \( A \) interacts with a simulator \( S \) in the following game.

Initialization. \( A \) chooses two identities \( ID^* = \{ID^*_0, ID^*_1\} \) and sends \( ID^* \) to \( S \).
Set up and Phase 1. These two phases are the same as those of Definition 8.
Challenge. \( A \) submits a sender’s identity \( ID_s \), a message \( M \), and a set of identities \( \{ID_2, ID_3, \ldots, ID_t\} \) to \( S \), with restrictions that \( ID_s \notin ID^* \) and \( \text{SecretValue}(ID_s) \) has not been queried before. \( S \) then randomly chooses \( \beta \in \{0,1\} \) and generates \( C^* = \text{Encrypt}(ID_s, ID^*_\beta, ID_2, \ldots, ID_t, M) \). Finally, \( S \) sends \( C^* \) to \( A \).
Phase 2. \( A \) issues the queries defined in Phase 1 except for issuing \( \text{Decrypt}(C^*, ID^*_0) \), \( \text{Decrypt}(C^*, ID^*_1) \), and \( \text{SecretValue}(ID_s) \).
Guess. Finally, \( A \) outputs \( \beta' \in \{0,1\} \) and wins the game if \( \beta' = \beta \).

The advantage of \( A \) winning the game is defined as

\[
\text{Adv}^{\text{Anon-sMID-CCA-II}}(A) = \left| \Pr[\beta' = \beta] - \frac{1}{2} \right|
\]

It is not necessary to simulate \( \text{PartialKeyExtract} \) oracle of \( A \) since \( A \) has \( \text{msk} \). An AMRCLAE scheme is said to be Anon-sMID-CCA secure against Type II adversary if there exists no polynomial-time Type II attacker that can win the Anon-sMID-CCA game II with non-negligible advantage.

Note that there are some restrictions about the choice of \( ID_s \) in the challenge phase. In the Anon-sMID-CCA Game I and the Anon-sMID-CCA Game II, we have not modelled the sender as an attacker. The anonymity will be meaningless if the sender is the attacker since the receivers are chosen by him.

Theorem 3. (Anonymity) The proposed AMRCLAE scheme is Anon-sMID-CCA secure against Type I adversary in the random oracle model if the M-DDBH assumption holds.

Proof. \( S \) is given \( < q, G_1, G_2, c, P, aP, bP, cP, e(P,P)^b, c, Z > \) and then simulates the game for \( A \) as follows:

Initialization. \( A \) outputs a target identity set \( ID^* = \{ID^*_0, ID^*_1\} \).

Setup and Phase 1. These two phases are the same as those in the proof of Theorem 1.

Challenge. \( A \) sends a message \( M, t - 1 \) receivers’ identities \( \{ID_2, ID_3, \ldots, ID_t\} \), and a sender’s identity \( ID_s \) to \( S \) with restrictions that \( ID_s \notin ID^* \) and \( \text{PartialKeyExtract}(ID_s) \) has not been queried. \( S \) does the following:

1) Choose \( \beta \in \{0,1\} \) randomly.
2) For \( i = 2 \) to \( t \), call \( H(ID_i) \) and retrieve \( q_i \) from \( \text{H-list} \).
3) Call \( H(ID^*_\beta) \) and retrieve \( q_{\beta}^* \) from \( \text{H-list} \).
4) Call \( H(ID_s) \) and retrieve \( q_s \) from \( \text{H-list} \).
5) Get \( P_i \) and \( x_i \) for \( i = 2, \ldots, t \) from the \( \text{pk-list} \).
6) Get \( P_{\beta}, x_{\beta}, P_s, \) and \( x_s \) from the \( \text{pk-list} \).
7) Choose \( k \in \mathbb{Z}_q^* \) and set \( U^* = aP, U^*_1 = x_1(aP), \) and \( V^* = q_s(aP) \).
8) For $i = 2$ to $t$, compute $v_i = H_i(e(q_iU^*, q_s(cP)), e(q_iU^*, x_siP))$.
9) Compute $v_β = H_1(Z^βq_β, e(q_β^*(bP), x_βx_β^*(aP)))$.
10) Compute $f(x) = (x - v_β) \prod_{i=2}^{t} (x - v_i) + k = \sum_{i=0}^{t-1} c_i x^i + x^t \mod q$ and $W^* = M \cdot Ω^k$.
11) Set the ciphertext $C^* = (c_0, c_1, \ldots, c_{t-1}, U^1, U^2, V^*, W^*, ID_s)$ and send $C^*$ to $A$.

Phase 2. $A$ makes queries as those in Phase 1. However, if $A$ issues $\text{Decrypt}(C^*, ID_s \in ID^*)$ or $\text{PartialKeyExtract}(ID_s)$, $S$ will return "reject".

**Guess.** $A$ outputs $β' \in \{0, 1\}$. If $β' = β$, then $S$ outputs 1. Otherwise, $S$ outputs a random bit $\bar{β}$.

If $Z = e(P, P)^{abc}$, then $Z^βq_β = e(P, P)^{abcq_βq_β} = e(q_β^*(bP), q_s(cP))^a = e(Q_β^*, d_β)^a$ for $β \in \{0, 1\}$ and $C^*$ is a correct ciphertext. If $A$ wins the game with non-negligible advantage $ε$, $|Pr[A(Ω) = β' = β] - 1/2| ≥ ε$, where $Ω$ is a correct AMRCLAE scheme. Thus, we have that

\[
\begin{align*}
= \left| \frac{1}{2} Pr[A(Ω) = β] + \frac{1}{2} \right| - \left| \frac{1}{2} Pr[A(Ω) = β] - \frac{5}{8} \right| \\
= \frac{1}{4} Pr[A(Ω) = β] - \frac{1}{8} = \frac{1}{4} Pr[A(Ω) = β] - \frac{1}{2} ≥ \frac{ε}{4}.
\end{align*}
\]

Therefore, $S$ solves the M-DBDH problem with non-negligible advantage $ε/4$ within polynomial time.

Theorem 3 guarantees the CCA security of anonymity against Type I adversary.

**Theorem 4.** (Anonymity) The proposed AMRCLAE scheme is Anon-sMID-CCA secure against Type II adversary in the random oracle model if the M-DBDH assumption holds.

**Proof.** $S$ is given $q, G_1, G_2, ε, P, aP, bP, cP, e(P, P)^{bc}, Z >$ and then simulates the game for an adversary $A$ as follows:

**Initialization.** $A$ outputs a target identity set $ID^* = \{ID_0^*, ID_1^*\}$.

**Setup and Phase 1.** These two phases are the same as those in the proof of Theorem 2.

**Challenge.** $A$ sends a message $M$, $t - 1$ receivers’ identities $\{ID_2, ID_3, \ldots, ID_t\}$, and a sender’s identity $ID_s$ to $S$, with restrictions that $ID_s \notin ID^*$ and $\text{SecretValue}(ID_s)$ has not been queried before. $S$ does the same works as those of the proof of Theorem 3 except the following.
8) For $i = 2$ to $t$, compute $v_i = H_i(e(q_iU^*, αq_sP), e(q_iU^*, x_siP))$.
9) Compute $v_β = H_1(e(q_β^*(cP), αq_s(aP)), Z^βx_βx_β^*)$.

Phase 2. $A$ makes queries as those in Phase 1. However, if $A$ issues $\text{Decrypt}(C^*, ID_s \in ID^*)$ or $\text{SecretValue}(ID_s)$, $S$ will return "reject".

**Guess.** $A$ outputs $β' \in \{0, 1\}$. If $β' = β$, then $S$ outputs 1. Otherwise, $S$ outputs a random bit $\bar{β}$.

If $Z = e(P, P)^{abc}$, $S$ correctly simulates the Anon-sMID-CCA game II. If $A$ wins the game with non-negligible advantage $ε$, $|Pr[A(Ω) = β' = β] - 1/2| ≥ ε$, where $Ω$ is a correct AMRCLAE scheme. Thus, we have that

\[
\begin{align*}
\end{align*}
\]

Therefore, $S$ solves the M-DBDH problem with non-negligible advantage $ε/4$ within polynomial time.

Theorem 4 guarantees the CCA security of anonymity against Type II adversary.

**C. Sender Authentication**

**Definition 11** (The Sender Authentication Game I). Let $A$ be a polynomial-time Type I attacker. $A$ interacts with a simulator $S$ in the following game.

**Initialization.** $A$ chooses a target sender identity $ID_s^*$ and target receiver identity $ID_R^*$. Then $A$ sends $ID^* = \{ID_s^*, ID_R^*\}$ to $S$. 

Setup and Phase 1. These two phases are the same as those of Definition 7.

Forge. Finally, \( A \) outputs a ciphertext \( C^* \) with restrictions that the sender is \( ID_s^* \), \( ID_R^* \) is one of the receivers, and \( C^* \) was not outputted by querying the Encrypt oracle. \( A \) wins the game if \( C^* \) is a correct ciphertext.

The advantage of \( A \) winning the game is defined as
\[
\text{Adv}^{SA-I}(A) = \Pr[\text{Decrypt}(C, ID_R^*) \neq 1].
\]

An AMRCLAE scheme is said to satisfy sender authentication against Type I adversary if there exists no polynomial-time Type I attacker that can win the Sender Authentication game I with non-negligible advantage.

**Definition 12** (The Sender Authentication Game II). Let \( A \) be a polynomial-time Type II attacker. \( A \) interacts with a simulator \( S \) in the following game.

**Initialization.** \( A \) chooses a target sender identity \( ID_s^* \) and target receiver identity \( ID_R^* \). Then \( A \) sends \( ID^* = \{ ID_s^*, ID_R^* \} \) to \( S \).

**Setup and Phase 1.** These two phases are the same as those of Definition 8.

Forge. Finally, \( A \) outputs a ciphertext \( C^* \) with restrictions that the sender is \( ID_s^* \), \( ID_R^* \) is one of the receivers, and \( C^* \) was not outputted by querying the Encrypt oracle. \( A \) wins the game if \( C^* \) is a correct ciphertext.

The advantage of \( A \) winning the game is defined as
\[
\text{Adv}^{SA-II}(A) = \Pr[\text{Decrypt}(C, ID_R^*) \neq 1].
\]

An AMRCLAE scheme is said to satisfy sender authentication against Type II adversary if there exists no polynomial-time Type II attacker that can win the Sender Authentication game II with non-negligible advantage.

**Theorem 5.** (Sender Authentication) The proposed AMRCLAE scheme satisfies sender authentication against Type I adversary in the random oracle model if the 1-wDBDHI assumption holds.

**Proof.** Assume that there exists a polynomial-time Type I adversary \( A \) that wins the Sender Authentication game I with non-negligible advantage. Then we will construct a polynomial-time algorithm \( S \) that has non-negligible advantage in solving the 1-wDBDHI problem.

First, \( S \) is given \( < q, G_1, G_2, e, P, bP, cP, Z > \) which is an instance of the 1-wDBDHI problem. \( S \) simulates the game for \( A \) as follows:

**Initialization.** \( A \) outputs an identity set \( ID^* = \{ ID_s^*, ID_R^* \} \).

**Setup and Phase 1.** If we set \( T = Z \), then these two phases will be the same as those in the proof of Theorem 1.

Forge. Finally, \( A \) outputs \( C^* = (c_0, c_1, \ldots, c_{t-1}, U^*, U_1^*, \ldots, U_t^*, ID_s^*) \), where \( C^* \) was not outputted by querying the Encrypt oracle. Then \( S \) performs the followings.

1. Search \( H_2 \)-list to get \( (M_i, k_i, r_i, U_i) \) with \( U_i = U^* \).
2. Compute \( v^* = H_1(Z^{q_{i, r_i}} v, e(H(ID_R^*), x_R^{U_i})) \).
3. Compute \( k^* = c_0 + c_1 v^* + \cdots + c_{t-1} (v^*)^{t-i} \mod q \).
4. Verify if \( r_i H(ID_s^*) = V^*, k_i = k^*, M_i = W/\Omega^{k_i}, \) and \( e(U^*, H(ID_s^*)) = e(V^*, P) \). If not, \( S \) outputs 0. Otherwise, \( S \) outputs 1.

In the Encrypt oracle, if \( T = Z = e(P, P)^{b^*c} \), then \( T^{s_{r_i q_i}} = e(P, P)^{b^*c r_i q_i} = e(q_i, bP) \cdot q_s(bP)^{r_i} = e(Q_s, d_s)^{r_i} \). Similarly, in the Decrypt oracle, if \( T = Z = e(P, P)^{b^*c} \), then \( T^{s_{r_i q_i}} = e(P, P)^{b^*c r_i q_i} = e(q_i, bP) \cdot q_s(bP)^{r_i} = e(Q_s, d_s)^{r_i} \). As the construction above, \( S \) correctly simulates the game if \( Z = e(P, P)^{b^*c} \). Assume that \( A \) wins the game with non-negligible advantage at least \( \epsilon \) under a correct simulation. To analyse the advantage of solving the 1-wDBDHI problem, we define the following events.

1. The game has been correctly simulated.
2. \( A \) wins the game.

Then we have that
\[
\Pr[S(P, bP, cP, e(P, P)^{b^*c}) = 1] = \Pr[E_1 \land E_2]
\]
\[
\Pr[E_1] \Pr[E_2 | E_1] \geq 1 \cdot \epsilon = \epsilon
\]
and
\[
\left| \Pr[S(P, bP, cP, e(P, P)^{b^*c}) = 1] - \Pr[S(P, bP, cP, Z) = 1] \right| = \frac{1}{2} \Pr[S(P, bP, cP, e(P, P)^{b^*c}) = 1] \geq \frac{\epsilon}{2}.
\]
Therefore, $S$ solves the 1-wDBDHI problem with non-negligible advantage $\frac{\epsilon}{2}$ within polynomial time.

**Theorem 6. (Sender Authentication)** The proposed AMRCLE scheme satisfies sender authentication against Type II adversary in the random oracle model if the 1-wDBDHI assumption holds.

**Proof.** $S$ is given $< q, G_1, G_2, e, P, bP, cP, Z >$ and then simulates the game for an adversary $A$ as follows:

1. **Initialization.** $A$ outputs an identity set $ID^* = \{ ID^*_s, ID^*_R \}$.

2. **Setup and Phase 1.** If we set $T = Z$, these two phases will be the same as those in the proof of Theorem 2.

   1) Search $H_2$-list to get $(M_i, k_i, r_i, U_i)$ with $U_i = U^*$.
   2) Compute $v^* = H_1(e(V^*, \alpha H(ID^*_R)), Z^*q_h^{x^*_s x^*_h})$.
   3) Compute $k^* = c_0 + c_1 v^* + \cdots + c_{t-1} (v^*)^{t-1} + (v^*)^t \mod q$.
   4) Verify if $r_i P^* = U^*_i, k_i = k^*, M_i = W/\Omega^{k^*}$, and $e(U^*_i, P^*_s) = e(U^*_s, P)$. If not, $S$ outputs 0. Otherwise, $S$ outputs 1.

   $S$ correctly simulates the game if $Z = e(P, P)^{v^*c}$. Assume that $A$ wins the game with non-negligible advantage at least $\epsilon$ under a correct simulation. Then we have that

\[
\left| Pr[S(P, bP, cP, e(P, P)^{v^*c}) = 1] - Pr[S(P, bP, cP, Z) = 1] \right| \geq \frac{\epsilon}{2}.
\]

That is, $S$ solves the 1-wDBDHI problem with non-negligible advantage $\frac{\epsilon}{2}$ within polynomial time. 

Theorem 5 and Theorem 6 guarantees that the proposed scheme satisfies Sender Authentication. In other words, even if an adversary compromises with any $t-1$ of $t$ receivers, the adversary cannot impersonate the sender to generate a correct ciphertext for the $t$ receivers.

VI. Comparisons

In this section, we compare the proposed scheme with [10] and the existing AMRCLEs [14], [12] in performance and security. According to [15], [17], [21], and [34], we can obtain that $T_p \approx 5T$, $T_n \approx 29T_m$, $T_e \approx 240T_m$, $T_h \approx 23T_m$, and $T_a \approx 0.12T_m$ shown in Table II, which summarizes the computation costs of encryption/decryption and the ciphertext length for multiple receivers.

The security comparison is shown in Table III. There are two kinds of Type I adversaries in the security model of anonymity, who are insiders and outsiders. An insider is a selected receiver, while an outsider is not. An insider can get information more than an outsider since an insider can decrypt the ciphertext. The proposed scheme owns CCA security against both Type I and Type II attackers simultaneously in not only confidentiality but also anonymity. Especially, it also is the first AMRCLE scheme that achieves sender authentication. All the properties of our scheme have been formally proved in Section V.

VII. Conclusion

This paper has presented the first anonymous multi-receiver certificateless authenticated encryption scheme. The proposed scheme achieves provable CCA security in both confidentiality and anonymity against Type I and Type II attackers, and it also achieves sender authentication. The security of our scheme is guaranteed based on the M-DBDH assumption and the 1-wDBDHI assumption under the random oracle model. An open problem in this field is to find a secure anonymous multi-receiver certificateless authenticated encryption scheme under the standard model, i.e., without random oracles.

Acknowledgment

This work was partially supported by the Ministry of Science and Technology of the Taiwan under grants MOST 105-2923-E-110-001-MY3, MOST 105-2221-E-110-053-MY2, and "Aim for the Top University Plan" of the National Sun Yat-sen University and Ministry of Education, Taiwan, R.O.C.
TABLE II

<table>
<thead>
<tr>
<th>Encryption cost</th>
<th>Decryption cost</th>
<th>Ciphertext Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2t + 2)T_h + (2t + 2)T_s + 2tT_m + T_{poly} ≈ (104t + 104)T_m + T_{poly}</td>
<td>3T_h + tT_m + T_s ≈ (t + 98)T_m</td>
<td>(t + 1)</td>
</tr>
<tr>
<td>(4t + 2)T_h + (2t + 1)T_s + T_p ≈ (1350t + 75)T_m</td>
<td>5T_h + T_s + T_p ≈ (1344)T_m</td>
<td></td>
</tr>
<tr>
<td>(2t + 1)T_h + 4T_s + tT_m + T_{poly} ≈ (1240t + 139)T_m + T_{poly}</td>
<td>2T_h + T_T_m + T_p + T_s ≈ (t + 1275)T_m</td>
<td>t</td>
</tr>
<tr>
<td>Ours (t + 17)h, 7 (3f + 3)T_s + 2tT_m + T_{poly} ≈ (2510t + 139)T_m + T_{poly}</td>
<td>2T_h + 2T_s + T_T_m + 2T_{poly} ≈ (t + 2504)T_m</td>
<td>t</td>
</tr>
</tbody>
</table>

*|T_p*: the cost of a pairing operation

*|T_h*: the cost of a hash operation

*|T_m*: the cost of a modular multiplication in Z_q

*|T_s*: the cost of a modular exponentiation in Z_q

*|T_s*: the cost of a scalar multiplication in an additive group or an exponentiation in a multiplicative group

*|T_m*: the cost of an addition in an additive group or a multiplication in a multiplicative group

*|T_{poly}*: the cost of constructing polynomial

*|t*: the cost of using Chinese Remainder Theorem

*||ID|*: the bit-length of an identity

*|q*: a big prime

*|u*: the bit-length of an element in an additive group

*|w*: the bit-length of an element in a multiplicative group

*|s*: the bit-length of a symmetric encryption key

TABLE III

<table>
<thead>
<tr>
<th>Confidentiality</th>
<th>Anonymity</th>
<th>Security Model</th>
<th>Sender Authentication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outsider KGC</td>
<td>Outsider Insider KGC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[14] CCA CCA</td>
<td>CCA CCA CCA ROM No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[12] CCA CCA</td>
<td>CCA CCA CCA ROM No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[10] CCA CCA</td>
<td>CCA CCA CCA ROM Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours CCA CCA</td>
<td>CCA CCA CCA ROM Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*There exist some problems in their security proofs, shown in Section III-A and Section III-B.

References


Yi-Feng Tseng was born in Kaohsiung, Taiwan. He received the MS degree in computer science and engineering from National Sun Yat-sen University, Taiwan, in 2014, and now is a Ph.D. candidate in the same department. His research interests include cloud computing and security, network and communication security, information security, cryptographic protocols, and applied cryptography.

Chun-I Fan received the M.S. degree in computer science and information engineering from the National Chiao Tung University, Hsinchu, Taiwan, in 1993, and the Ph.D. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 1998. From 1999 to 2003, he was an Associate Researcher and a Project Leader with Telecommunication Laboratories, Chungwa Telecom Company, Ltd., Taoyuan, Taiwan. In 2003, he joined the faculty of the Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, Taiwan, where he has been a Full Professor since 2010. His current research interests include applied cryptology, cryptographic protocols, and information and communication security. Prof. Fan is the Deputy Chairman of the Chinese Cryptology and Information Security Association, and the Chief Executive Officer (CEO) of “Aim for the Top University 2012 Plan” Office at National Sun Yat-sen University. He was the recipient of the Best Student Paper Awards from the National Conference on Information Security in 1998, the Dragon Ph.D. Thesis Award from Acer Foundation, the Best Ph.D. Thesis Award from the Institute of Information and Computing Machinery in 1999, and the Engineering Professors Award from Chinese Institute of Engineers - Kaohsiung Chapter in 2016. Prof. Fan is also an Outstanding Faculty in Academic Research in National Sun Yat-sen University.