IEEE Open Journal of the Computer Society

Received 5 July 2022; accepted 8 August 2022. Date of publication 11 August 2022;
date of current version 24 August 2022. The review of this paper was arranged by Senior Editor Jiankun Hu.

Digital Object Identifier 10.1109/OJCS.2022.3198073

Leakage-Resilient Certificate-Based Authenticated Key Exchange Protocol

TUNG-TSO TSAI [1], SEN-SHAN HUANG [2], YUH-MIN TSENG [2], YUN-HSIN CHUANG [2], AND YING-HAO HUNG [3]

[1]Department of Computer Science and Engineering, National Ocean University, Keelung 202, Taiwan
[2]Department of Mathematics, National Changhua University of Education, Changhua 500, Taiwan
[3]Department of Mathematics, National Experimental High School at Hsinchu Science Park, Hsinchu 300, Taiwan

CORRESPONDING AUTHOR: YUH-MIN TSENG (e-mail: ymtseng@cc.ncue.edu.tw)

The work was supported by the Ministry of Science and Technology, Taiwan under Grants MOST110-2221-E-018-006-MY2, MOST110-2221-E-018-007-MY2, and MOST110-2222-E-019-001-MY2.

ABSTRACT Certificate-based public key cryptography (CB-PKC) removes the problem of certificate management in traditional public key systems and avoids the key escrow problem in identity-based public key systems. In the past, many authenticated key exchange (AKE) protocols based on CB-PKC systems, called CB-AKE, were proposed to be applied to secure communications between two remote participants. However, these existing CB-AKE protocols become insecure since attackers could compute and obtain the whole secret key from some partial leaked information of the secret key by side channel attacks. In this paper, our goal is to propose the *first* CB-AKE protocol with the property to resist side channel attacks, called leakage-resilient CB-AKE (LR-CB-AKE). The proposed LR-CB-AKE protocol is formally proven to be secure in the generic bilinear group (GBG) model under the discrete logarithm (DL) and computational Diffie-Hellman (CDH) assumptions.

INDEX TERMS Authenticated key exchange, certificate-based cryptography, generic bilinear group, leakage-resilience.

I. INTRODUCTION

Certificate-based public key cryptography (CB-PKC), proposed by Gentry [1], removes the problem of certificate management in traditional public key systems and avoids the key escrow problem in identity-based public key systems. CB-PKC has two roles: certificate authority (CA) and users. Each user generates a user secret key and a partial public key, and transmits the partial public key to the CA. After receiving the partial public key of the user, the CA creates a certificate and the other partial public key for the user. Therefore, in the CB-PKC system, the public keys of each user are respectively generated by herself/himself and the CA, and her/his full private keys comprise the user secret key and the certificate.

The first authenticated key exchange (AKE) protocol [2] based on CB-PKC systems, call CB-AKE protocol, was proposed to be applied to secure communications between two remote participants. A common session key (CSK) is established by two remote participants on an open network (insecure network). They may employ the CSK to encrypt data and transmit the encrypted data to the other participant to ensure the confidentiality of the data. Until now, several CB-AKE protocols [3], [4], [5], [6] have been proposed. However, none of these protocols can resist side channel attacks [7], [8]. These protocols are insecure since attackers could compute and obtain the whole secret key from some partial leaked information of the secret key under side channel attacks. To the best of our knowledge, there is no CB-AKE protocol with the ability to resist side channel attacks. Here, we will propose the first CB-AKE protocol that can resist such attacks, called leakage-resilient CB-AKE (LR-CB-AKE) protocol.

A. RELATED WORK

Traditional public key cryptography (PKC) has an inborn problem, namely, certificate management of a public-key infrastructure (PKI). Since a user's public key in PKC systems is an arbitrary number that has no meaning, a certificate is needed to connect the user's public key with her/his identity information. Therefore, PKC needs the PKI to manage

each user's certificate. An intuitive idea that a user's identity (ID) can be regarded as her/his public key was proposed by Shamir [9]. With this idea, Boneh and Franklin [10] proposed the first practical ID-based encryption scheme which includes two roles: private key generator (PKG) and users. Each user's public key is her/his own identity while her/his private key is made by the PKG. Obviously, there is a key escrow problem in the sense that the PKG knows the private key of each user. To eliminate the problems of both certificate management and key escrow at the same time, Gentry [1] proposed the certificate-based cryptography (PKC) concept, in which the private key of each user is divided into two parts: one is selected by the user, and the other is the certificate generated by the CA.

AKE protocols [11], [12], [13], [14] can be used to establish a common session key (CSK) of two remote users for secure communication on open networks. Based on ID-based PKC (ID-PKC) systems, the first ID-AKE protocol was proposed by Smart [15]. However, the protocol has a drawback, namely, no forward secrecy property. A new ID-AKE protocol that possesses forward secrecy and has better efficiency was proposed by Shim [16]. Subsequently, several ID-AKE protocols [17], [18], [19] were proposed to improve efficiency and security. To solve the key escrow problem, the first CB-AKE protocol was proposed by Wang and Cao [2]. However, Lim et al. [3] proved that Wang and Cao's protocol was insecure when ephemeral secret keys were leaked. Moreover, Lim et al. [3] proposed a new CB-AKE protocol to improve the security. Latterly, many studies on CB-AKE protocols were published in the literatures [4], [5], [6].

Indeed, none of those existing CB-AKE protocols can resist side channel attacks [7], [8]. An attacker can obtain partial leaked information of the private key by such attacks. Once this attack is repeated, the attacker could calculate the full private key. To resist such attacks, many cryptographic researchers have put the leakage-resilient (LR) property on cryptographic protocols. Two LR-AKE protocols [20], [21] based on traditional PKC systems were proposed, but these two protocols become insecure when ephemeral secret keys are compromised. In order to improve security or efficiency, various LR-AKE protocols have been published in the literatures [22], [23]. Although these protocols can meet the leakage-resilient property, there is a common disadvantage that the total leaked information (bits) of the secret key are limited (bounded) during each session of the system life cycle. In order to achieve unbounded total leaked bits, an unbounded LR-AKE protocol was proposed by Alawatugoda et al. [24]. However, the efficiency of the unbounded LR-AKE protocol is not good enough because the employed key update technique [25] is time-consuming. The multiplicative blinding method [26], [27] is employed to construct a new unbounded LR-AKE protocol [28] to improve the efficiency of the key update processes.

To remove the use of certificates and retain the leakage-resilient property, Elashry et al. [29] proposed the first LR-ID-AKE protocol. Unfortunately, both leakage and impersonation attacks, will occur in Elashry et al.'s protocol, which was pointed out by Hatri et al. [30]. In fact, the limitation of total leaked bits is also an important issue to be studied. A secure LR-ID-AKE protocol with bounded total leaked bits was proposed by Ruan et al. [31]. Afterwards, Wu et al. [32] proposed an unbounded LR-ID-AKE protocol in the sense that attackers can obtain some information of the secret key during each session of the system life cycle while the total leaked bits are unlimited.

TABLE 1. Comparisons Between the Existing Protocols and Our LR-CB-AKE Protocol

Protocols	Public key setting	Avoiding key escrow issue	Resisting side-channel attacks	Restriction of leaked information
[31]	ID-based	No	Yes	Bounded
[32]	ID-based	No	Yes	Unbounded
[5]	Certificate-based	Yes	No	-
[6]	Certificate-based	Yes	No	-
Ours	Certificate-based	Yes	Yes	Unbounded

B. MOTIVATION

Up to now, the related cryptography resisting side-channel attacks [7], [8] is still a recently significant research topic. One limitation of the existing CB-AKE protocols [3], [4], [5], [6] is that the private keys cannot be partially disclosed to adversaries, namely, these private keys must be completely hidden from adversaries. As a result, these protocols could suffer from side-channel attacks and become insecure. Table 1 shows the comparisons between the LR-ID-AKE protocols [31], [32], the CB-AKE protocols [5], [6] and our LR-CB-AKE protocol in terms of public key setting, avoiding key escrow issue, resisting side-channel attacks and the restriction of leaked information. Our goal is to propose the first LR-CB-AKE protocol that can avoid key escrow issue, resist side-channel attacks and possess the property with unbounded leaked information.

C. CONTRIBUTIONS AND ORGANIZATIONS

Although unbounded LR-ID-AKE protocols avoid certificate management and possess the security against leakage attacks, these protocols inherit the key escrow problem. As mentioned earlier, the existing CB-AKE protocols can remove the problems of key escrow and certificate management. However, none of them can provide the security against leakage attacks of secret key. Therefore, we will propose the first leakage-resilient CB-AKE (LR-CB-AKE) protocol. We will achieve several contributions as mentioned below.

- We formulate a new framework and security model for LR-CB-AKE protocol.
- Based on the new framework, a concrete LR-CB-AKE protocol is proposed.
- Under the new security model, the proposed LR-CB-AKE protocol is formally proven to be secure.
- As compared with the previous LR-ID-AKE and CB-AKE protocols, our LR-CB-AKE protocol not only

withstands side channel attacks, but also eliminates the key escrow problem.

The rest of the article is as follows. Section II gives some preliminaries. The framework and security notions for LR-CB-AKE protocol are defined in Section III. A concrete LR-CB-AKE protocol is presented in Section IV. Section V demonstrates the security of the LR-CB-AKE protocol. We compare the performance with several existing LR-ID-AKE and CB-AKE protocols in Section VI. A conclusion is given in Section VII.

II. PRELIMINARIES

A. BILINEAR GROUPS

Assume that G_1 and G_2 are multiplicative cyclic groups of the same order p for a large prime p. A map $\hat{e} : G_1 \times G_1 \rightarrow G_2$ is a bilinear map that has the following three properties.

- Bilinearity: for all $X, Y \in G_1$ and $x, y \in Z_p^*$, we have $\hat{e}(X^x, Y^y) = \hat{e}(X, Y)^{xy}$.
- Non-degeneracy: $\hat{e}(g, g) \neq 1$, where g and $\hat{e}(g, g)$ are generators in G_1 and G_2, respectively.
- Computability: the bilinear map can be efficiently computed by an algorithm.

Here, the equation $\hat{e}(X^x, X^y) = \hat{e}(X, X)^{xy} = \hat{e}(X^y, X^x)$ holds since $\hat{e}$ is symmetric. For more details of bilinear groups, one can refer to [10], [33].

B. GENERIC BILINEAR GROUP (GBG) MODEL

In order to provide security proofs for cryptographic mechanisms, Boneh et al. [34] defined the generic bilinear group (GBG) model based on the generic group (GG) model [35]. Injective mapping functions will be employed to encode group elements to bit-strings in the GBG model. Since we have two groups G_1 and G_2 as defined earlier, two injective mapping functions $IMF_1 : Z_p^* \rightarrow \Psi_{G_1}$ and $IMF_2 : Z_p^* \rightarrow \Psi_{G_2}$ will respectively be selected to perform the encoding process, where Ψ_{G_1} and Ψ_{G_2} are, respectively, the encoded bit-string sets of G_1 and G_2. After the encoding process, the elements in the two groups will be represented in the form of bit-strings. We denote $|\Psi_{G_1}|$ and $|\Psi_{G_2}|$ as the numbers of two sets Ψ_{G_1} and Ψ_{G_2}, respectively. Here, the two sets are disjoint and $|\Psi_{G_1}| = |\Psi_{G_2}| = p$.

Next, to express the multiplications of G_1 and G_2 and the computation of $\hat{e}$ in the GBG model, we define three group operations as follows:

- $GOP_1(IMF_1(r), IMF_1(s)) \rightarrow IMF_1(r + s \bmod p)$.
- $GOP_2(IMF_2(r), IMF_2(s)) \rightarrow IMF_2(r + s \bmod p)$.
- $GOP_p(IMF_1(r), IMF_1(s)) \rightarrow IMF_2(r \cdot s \bmod p)$.

Here, $g = IMF_1(1)$ and $\hat{e}(g, g) = IMF_2(1)$. In addition, the relevant implementation has been completed. Readers can refer to the literature [36].

C. COMPLEXITY ASSUMPTIONS

Two well-known difficult problems are the discrete logarithm (DL) and the computational Diffie-Hellman (CDH) problems which are used to define two associated complexity assumptions as follows.

Definition 1 (DL assumption): By the DL problem, g and g^a in G_1 are given but $a \in Z_p^*$ is unknown. Assume that there is a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ who wants to calculate a. The advantage of calculating the correct value can be defined as $Adv_{\mathcal{A}} = \Pr[\mathcal{A}(g, g^a) = a]$.

Definition 2 (CDH assumption): By the CDH problem, g, g^a and g^b in G_1 are given but a and b in Z_p^* are unknown. Assume that there is a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ who wants to calculate g^{ab}. The advantage of calculating the correct value can be defined as $Adv_{\mathcal{A}} = \Pr[\mathcal{A}(g, g^a, g^b) = g^{ab}]$.

D. ENTROPY

Since a leakage-resilient scheme/protocol allows some information of the secret key to be leaked, the entropy concept can be hired to measure the security of the system after some information of the secret key was leaked. We define RV and CRV as two finite random variables and then state two types of min-entropies as follows.

1. The min-entropy of RV is
$$H_\infty(RV) = -\log_2 \left(\max_{rv} \Pr[RV = rv] \right)$$
2. The average conditional min-entropy of RV is $\widetilde{H}_\infty(RV|CRV) = -\log_2(E_{crv \leftarrow CRV}[\max_{rv} \Pr[RV = rv|CRV = crv]])$.

In a computation round, two types of participated secret keys can occur: one is a single secret key, and the other is multiple secret keys. For the leakage of a single secret key, we can use Lemma 1 [37] below to measure the security of the system. On the other hand, for the leakage of multiple secret keys, Lemma 2 are employed to measure the security of the system [26].

Lemma 1: Assume that a random variable (can be viewed as a single secret key involved in an algorithm) is K and its maximal leaked information length is λ. Let $f : K \rightarrow \{0, 1\}^\lambda$ be a leakage function, we obtain $\widetilde{H}_\infty(K|f(K)) \geqq H_\infty(K) - \lambda$.

Lemma 2: Assume that multiple random variables (can be viewed as multiple secret keys involved in an algorithm) are $K_1, K_2, \ldots, K_n$ and a highest d-degree polynomial related to these variables is $F \in Z_p[K_1, K_2, \ldots, K_n]$. Let $PD_1, PD_2, \ldots, PD_n$ be probability distributions on Z_p such that $H_\infty(PD_i) \geqq \log p - \lambda$ and $0 \leqq \lambda \leqq \log p$, for $i = 1, 2, \ldots, n$. When $k_i \overset{PD_i}{\leftarrow} Z_p$, for $i = 1, 2, \ldots, n$, are independent, we have $\Pr[F(K_1 = k_1, K_2 = k_2, \ldots, K_n = k_n) = 0] \leqq (d/p)2^\lambda$.

According to the inequality $\Pr[F(K_1 = k_1, K_2 = k_2, \ldots, K_n = k_n) = 0] \leqq (d/p)2^\lambda$ in Lemma 2, we can obtain the following result.

Corollary 1: If $k_i \overset{PD_i}{\leftarrow} Z_p$, for $i = 1, 2, \ldots, n$, are independent, then $\Pr[F(K_1 = k_1, K_2 = k_2, \ldots, K_n = k_n) = 0]$ is negligible if $\lambda < (1 - \epsilon) \log p$.

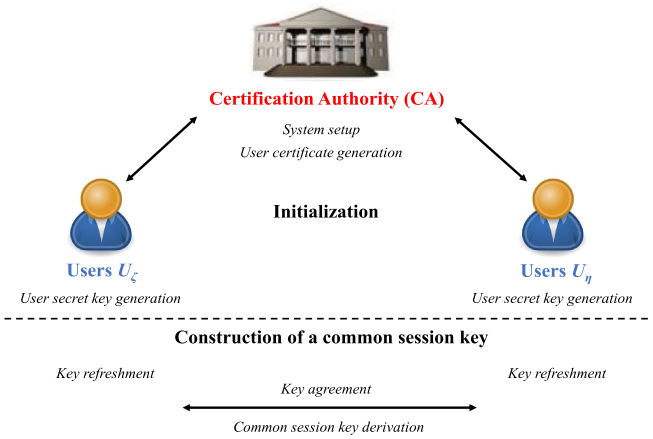

FIGURE 1. The framework of LR-CB-AKE protocol.

TABLE 2. Notations

Notations	Meaning
SK	the system secret key
SK_{pair_0}	the initial system secret key pair
ID_ζ	an identity of the user U_ζ
USK_ζ	the user secret key of the user U_ζ
$USK_{\zeta_{pair_0}}$	the initial user secret key pair of the user U_ζ
FPK_ζ	the first public key of the user U_ζ
(UCA_ζ, UCB_ζ)	two user certificates of the user U_ζ
$(UCA_{\zeta_{pair_0}}, UCB_{\zeta_{pair_0}})$	two initial user certificate pairs of the user U_ζ
SPK_ζ	the second public key of the user U_ζ
ESK_ζ	an ephemeral secret key of the user U_ζ
X	an ephemeral public key of the user U_ζ
ESK_η	an ephemeral secret key of the user U_η
Y	an ephemeral public key of the user U_η
CSK	a common session key of the users U_ζ and U_η

- *User certificate generation:* This algorithm is run by the CA who gains the user's two user certificates UCA_ζ and UCB_ζ and the corresponding second public key SPK_ζ after inputting the user's identity ID_ζ and first public key FPK_ζ in the i-th session. Meanwhile, the CA must use $(SK_{i-1,1}, SK_{i-1,2})$ to update the current system secret key pair $SK_{pair_i} = (SK_{i,1}, SK_{i,2})$. In addition, the CA sends UCA_ζ, UCB_ζ and SPK_ζ to the user so that the user can compute two initial user certificate pairs $UCA_{\zeta_{pair_0}} = (UCA_{\zeta,0,1}, UCA_{\zeta,0,2})$ and $UCB_{\zeta_{pair_0}} = (UCB_{\zeta,0,1}, UCB_{\zeta,0,2})$, and the complete public key (FPK_ζ, SPK_ζ).

– Construction of a common session key:

- *Key refreshment:* For the k-th session, this algorithm is run by the user with identity ID_ζ who gains the refreshed user secret key pair $USK_{\zeta_{pair_k}} = (USK_{\zeta,k,1}, USK_{\zeta,k,2})$ and certificate pairs $UCA_{\zeta_{pair_k}} = (UCA_{\zeta,k,1}, UCA_{\zeta,k,2})$ and $UCB_{\zeta_{pair_k}} = (UCB_{\zeta,k,1}, UCB_{\zeta,k,2})$ after inputting $USK_{\zeta_{pair_{k-1}}} = (USK_{\zeta,k-1,1}, USK_{\zeta,k-1,2}), UCA_{\zeta_{pair_{k-1}}} = (UCA_{\zeta,k-1,1}, UCA_{\zeta,k-1,2})$ and $UCB_{\zeta_{pair_{k-1}}} = (UCB_{\zeta,k-1,1}, UCB_{\zeta,k-1,2})$.
- *Key agreement:* Two users U_ζ and U_η with identities ID_ζ and ID_η select ephemeral secret keys $ESK_\zeta = x$ and $ESK_\eta = y \in Z_p^*$, and compute $X = g^x$ and $Y = g^y$, respectively. U_ζ sends X and her/his two public keys FPK_ζ and SPK_ζ to U_η while U_η sends Y and her/his two public keys FPK_η and SPK_η to U_ζ. Then, U_ζ and U_η respectively compute session keys.
- *Common session key derivation:* This algorithm is run by U_ζ and U_η respectively to gain a common session key CSK after inputting the session keys.

III. FRAMEWORK AND SECURITY NOTIONS

A. FRAMEWORK

The framework of LR-CB-AKE protocol includes two roles and six algorithms, as shown in the Fig. 1. One role is the CA who executes *System setup* and *User certificate generation* algorithms, and the other is the user who performs the remaining four algorithms, namely, *User secret key generation*, *Key refreshment*, *Key agreement* and *Common session key derivation*. For convenience, some notations used in these algorithms are summarized in Table 2. Next, we define the six algorithms as follows.

– Initialization:

- *System setup:* This algorithm is run by the CA who gains the system secret key SK after inputting a security parameter. Then, the CA uses SK to produce the initial system secret key pair $SK_{pair_0} = (SK_{0,1}, SK_{0,2})$ and public parameters PP.
- *User secret key generation:* This algorithm is run by a user with identity ID_ζ who gains her/his user secret key USK_ζ, initial user secret key pair $USK_{\zeta_{pair_0}} = (USK_{\zeta,0,1}, USK_{\zeta,0,2})$ and the first public key FPK_ζ.

B. SECURITY NOTIONS

In the past, the system secret key SK and users' CSK of CB-AKE protocols were not allowed to be leaked since the systems did not have the leakage-resilient properties. Instead, a LR-CB-AKE protocol allows adversaries to obtain information of SK and CSK. We employ four leakage functions $f_{\mathrm{UCG},i}$, $h_{\mathrm{UCG},i}$, $f_{\mathrm{CSK},\zeta,k}$, $h_{\mathrm{CSK},\zeta,k}$ to describe how adversaries can obtain some information of SK and CSK. The first two leakage functions $f_{\mathrm{UCG},i}$ and $h_{\mathrm{UCG},i}$ respectively take $SK_{i,1}$ and $SK_{i,2}$ as input in the i-th invocation in *User certificate generation* algorithm, and output $\Lambda_{f_{\mathrm{UCG},i}}$ and $\Lambda_{h_{\mathrm{UCG},i}}$ as the length of the leaked information. The last two leakage functions $f_{\mathrm{CSK},\zeta,k}$ and $h_{\mathrm{CSK},\zeta,k}$ respectively take $(USK_{\zeta,k,1}, UCA_{\zeta,k,1}, ESK_\zeta)$ and $(USK_{\zeta,k,2}, UCA_{\zeta,k,2})$ as input in the k-th session of the user with identity ID_ζ in session key construction phase, and output $\Lambda_{f_{\mathrm{CSK},\zeta,k}}$ and $\Lambda_{h_{\mathrm{CSK},\zeta,k}}$ as the length of the leaked information. Notice that the maximum length of each output is only λ bits.

As the CB-AKE protocols [5], [6] proposed in the past, there are also two types of adversaries in our security model.

Type 1 adversary is an external attacker (not a system member) who has the ability to replace the public key of any user. Type 2 adversary is the malicious CA in the system who has the system secret key, but cannot replace the public key of any user. Next, we adopt the properties of leakage resilient [31], [32] and the security models of CB-AKEs [5], [6] to define a new security model for LR-CB-AKE. Indeed, the new security model is an extension of the extended Canetti-Krawczyk (eCK) model proposed by Lamacchia et al. [11]. The eCK model allows that adversaries can compromise either short-term secret key (ephemeral secret key) or long-term secret key (user secret key and certificate) of a participator. The new security model not only retains the properties of the eCK model, but also allows adversaries to obtain the leaked information of these secret keys by adding two leak queries, namely, *user certificate generation leak* query and *send leak* query. Assume that there exists a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ who attempts to break a LR-CB-AKE protocol. $\mathcal{A}$ will play the following security game with a challenger $\mathcal{B}$ to obtain the probability of breaking the LR-CB-AKE protocol.

– *Setup:* The challenger $\mathcal{B}$ performs the *System setup* algorithm to generate the system secret key and public parameters. Then, $\mathcal{B}$ sends the public parameters to $\mathcal{A}$, and gives the system secret key to $\mathcal{A}$ if $\mathcal{A}$ is a Type 2 adversary.

– *Query:* $\mathcal{A}$ can adaptively issue $\mathcal{B}$ the following queries. Here $\prod_{\zeta}^{k}$ denotes an oracle with an identity ID_{ζ} in the k-th session.

- *Send* $(\prod_{\zeta}^{k}, m)$: Upon inputting the oracle $\prod_{\zeta}^{k}$ and message m, the challenger $\mathcal{B}$ gives $\mathcal{A}$ the corresponding response of $\prod_{\zeta}^{k}$ and m.
- *Reveal* $(\prod_{\zeta}^{k})$: Upon inputting the oracle $\prod_{\zeta}^{k}$, the challenger $\mathcal{B}$ gives $\mathcal{A}$ a common session key which is held by $\prod_{\zeta}^{k}$.
- *User secret key generation* (ID_{ζ}): Upon inputting a user's identity ID_{ζ}, the challenger $\mathcal{B}$ gives $\mathcal{A}$ the associated initial user secret key pair $USK_{\zeta_{pair_0}}$ and the user's first public key FPK_{ζ}.
- *User certificate generation* $(ID_{\zeta}, FPK_{\zeta})$: Upon inputting a user's identity ID_{ζ} and the user's first public key FPK_{ζ}, the challenger $\mathcal{B}$ gives $\mathcal{A}$ the associated user certificates UCA_{ζ} and UCB_{ζ} and the corresponding second user public key SPK_{ζ}. This query is only issued by Type 1 adversary.
- *Replace public key* $(ID_{\zeta}, FPK_{\zeta}, {}' SPK'_{\zeta})$: Upon inputting a user's identity ID_{ζ} and the user public keys $(FPK_{\zeta}, {}' SPK'_{\zeta})$, the challenger $\mathcal{B}$ sets the user ID_{ζ}'s new public keys as $(FPK_{\zeta}, {}' SPK'_{\zeta})$. This query is only issued by Type 1 adversary.
- *Ephemeral secret key reveal* $(\prod_{\zeta}^{k})$: Upon inputting the oracle $\prod_{\zeta}^{k}$, the challenger $\mathcal{B}$ gives $\mathcal{A}$ an ephemeral secret key ESK which is held by $\prod_{\zeta}^{k}$.
- *User certificate generation leak* $(ID_{\zeta}, FPK_{\zeta}, f_{\mathrm{UCG},i}, h_{\mathrm{UCG},i})$: Upon inputting a user's ID_{ζ}, FPK_{ζ} and two leakage functions $(f_{\mathrm{UCG},i}, h_{\mathrm{UCG},i})$, the challenger $\mathcal{B}$ gives $\mathcal{A}$ leakage information $\Lambda_{f_{\mathrm{UCG},i}}$ and $\Lambda_{h_{\mathrm{UCG},i}}$ of the system secret key pair $SK_{i,1}$ and $SK_{i,2}$, respectively.
- *Send leak* $(\prod_{\zeta}^{k}, f_{\mathrm{CSK},\zeta,k}, h_{\mathrm{CSK},\zeta,k})$: Upon inputting the oracle $\prod_{\zeta}^{k}$ and two leakage functions $(f_{\mathrm{CSK},\zeta,k}, h_{\mathrm{CSK},\zeta,k})$, the challenger $\mathcal{B}$ gives $\mathcal{A}$ the leakage information $\Lambda_{f_{\mathrm{CSK},\zeta,k}}$ and $\Lambda_{h_{\mathrm{CSK},\zeta,k}}$ of $(USK_{\zeta,k,1}, UCA_{\zeta,k,1}, ESK_{\zeta})$ and $(USK_{\zeta,k,2}, UCA_{\zeta,k,2})$, respectively.
- *Test* $(\prod_{\zeta}^{k})$: Upon inputting the oracle $\prod_{\zeta}^{k}$, the challenger $\mathcal{B}$ gives $\mathcal{A}$ a session key according to the result of a random $coin \in \{0, 1\}$. If $coin = 1$, the session key is held by $\prod_{\zeta}^{k}$; otherwise, the common session key is a random value from session key space.

When the adversary $\mathcal{A}$ receives the common session key, $\mathcal{A}$ responds a guess $coin'$. If $coin' = coin$, the adversary $\mathcal{A}$ wins the security game. Finally, two security properties of the security game are defined as below.

Definition 3 (Partnership): Assume that there exist $\prod_{\zeta}^{k}$ and $\prod_{\eta}^{l}$ that state the user with identity ID_{ζ}'s k-th session and the user with identity ID_{η}'s l-th session, respectively. When $\prod_{\zeta}^{k}$ and $\prod_{\eta}^{l}$ authenticate each other and generate a common session key, we say that they have the partnership property.

Definition 4 (Freshness): We say that a common session key established by two oracles $\prod_{\zeta}^{k}$ and $\prod_{\eta}^{l}$ has freshness property, if the following three conditions in the *Query phase* are true.

I. The query to *Reveal* $(\prod_{\zeta}^{k})$ and *Reveal* $(\prod_{\eta}^{l})$ cannot occur.

II. At least one of the queries to *User secret key generation* (ID_{ζ}), *User certificate generation* $(ID_{\zeta}, FPK_{\zeta})$ and *Ephemeral secret key reveal* $(\prod_{\zeta}^{k})$ cannot occur.

III. At least one of the queries to *User secret key generation* (ID_{η}), *User certificate generation* (ID_{η}, FPK_{η}) and *Ephemeral secret key reveal* $(\prod_{\eta}^{l})$ cannot occur.

IV. LR-CB-AKE PROTOCOL

Our concrete protocol includes two phases, namely, the initialization and construction of a common session key. In the initialization phase, there are three algorithms, namely, *System setup*, *User secret key generation* and *User certificate generation*. In the construction of a common session key, three algorithms, namely, *Key refreshment*, *Key agreement* and *Common session key derivation*.

– Initialization:

• *System setup:* To generate the system public parameters PP and the initial system secret key pair SK_{pair_0} for LR-CB-AKE, the CA first takes as input a security parameter 1^{κ} and then performs the following tasks.

(1) Generate two multiplicative cyclic groups G_1 and G_2 of a large prime order p. Construct an admissible bilinear map $\hat{e} : G_1 \times G_1 \rightarrow G_2$. Assume that g is a generator of G_1.

(2) Pick a value $s \in Z_p^*$ in random, and compute the system secret key $SK = g^s$ and the system public key $PK = \hat{e}(g^s, g)$.
(3) Randomly choose a value $\gamma \in Z_p^*$ and set the initial system secret key pair $SK_{pair_0} = (SK_{0,1}, SK_{0,2}) = (g^\gamma, SK \cdot g^{-\gamma})$.
(4) Choose two random values $t, v \in Z_p^*$ and compute $T = g^t$ and $V = g^v$.
(5) Pick a hash function $H : G_2 \times G_1 \times G_1 \to G_1$ and set the system public parameters $PP = \{p, G_1, G_2, g, \hat{e}, PK, T, V, H\}$.

• *User secret key generation:* In the i-th session, to generate the initial user secret key pair $USK_{\zeta_{pair_0}}$ and the user's first public key FPK_ζ, the user U_ζ with identity ID_ζ performs the following tasks.

(1) Pick a random value $u_\zeta \in Z_p^*$ and compute the user secret key $USK_\zeta = h^{u_\zeta}$, where $h = H(PK||T||V)$.
(2) Randomly choose a value $c_i \in Z_p^*$ and set the initial user secret key pair $USK_{\zeta_{pair_0}} = (USK_{\zeta,0,1}, USK_{\zeta,0,2}) = (g^{c_i}, USK_\zeta \cdot g^{-c_i})$.
(3) Use the value u_ζ in (1) to set the user's first public key $FPK_\zeta = g^{u_\zeta}$.

By the similar way, for the user ID_η, we can generate the initial user secret key pair $USK_{\eta_{pair_0}} = (USK_{\eta,0,1}, USK_{\eta,0,2}) = (g^{c_j}, USK_\eta \cdot g^{-c_j})$ in the j-th session and the user's first public key $FPK_\eta = g^{u_\eta}$.

• *User certificate generation:* In the i-th session, when receiving an identity ID_ζ of user U_ζ and the associated first public key FPK_ζ, the CA is responsible for generating the user's two user certificates UCA_ζ and UCB_ζ, and the user's second public key SPK_ζ as follows.

(1) Randomly choose a value $\alpha_i \in Z_p^*$ and update the system secret key pair $SK_{pair_i} = (SK_{i,1}, SK_{i,2}) = (SK_{i-1,1} \cdot g^{\alpha_i}, SK_{i-1,2} \cdot g^{-\alpha_i})$, where $(SK_{i-1,1}, SK_{i-1,2})$ is the current system secret key pair.
(2) Set $b_\zeta = ID_\zeta||FPK_\zeta$ and randomly pick a value $\beta_\zeta \in Z_p^*$. Two user certificates are produced as follows.
✓ $UCA_\zeta = SK_{i,2} \cdot UCAT_\zeta$, where $UCAT_\zeta = SK_{i,1} \cdot (T \cdot V^{b_\zeta})^{\beta_\zeta}$,
✓ $UCB_\zeta = h^{\beta_\zeta}$, where $h = H(PK||T||V)$.
(3) Use the value β_ζ in (2) to compute the user's second public key $SPK_\zeta = g^{\beta_\zeta}$.

The CA sends the two user certificates UCA_ζ and UCB_ζ, and the second public key SPK_ζ to the user. Notice that the user certificate UCA_ζ contains the system secret key SK due to $UCA_\zeta = SK_{i,2} \cdot UCAT_\zeta = SK_{i,2} \cdot SK_{i,1} \cdot (T \cdot V^{b_\zeta})^{\beta_\zeta} = SK \cdot (T \cdot V^{b_\zeta})^{\beta_\zeta}$. Afterwards, two initial user certificate pairs $UCA_{\zeta_{pair_0}}$ and $UCB_{\zeta_{pair_0}}$ are computed by the user as below.

✓ $UCA_{\zeta_{pair_0}} = (UCA_{\zeta,0,1}, UCA_{\zeta,0,2}) = (g^{d_i}, UCA_\zeta \cdot g^{-d_i})$,
✓ $UCB_{\zeta_{pair_0}} = (UCB_{\zeta,0,1}, UCB_{\zeta,0,2}) = (h^{d_i}, UCB_\zeta \cdot h^{-d_i})$,

where $d_i \in Z_p^*$ and $h = H(PK||T||V)$.

By a similar way, for the user U_η, we can generate the two initial user certificate pairs $UCA_{\eta_{pair_0}} = (g^{d_j}, UCA_\eta \cdot g^{-d_j})$ and $UCB_{\eta_{pair_0}} = (h^{d_j}, UCB_\eta \cdot h^{-d_j})$ and the second public key $SPK_\eta = g^{\beta_\eta}$ in the j-th session.

- Construction of a common session key:

• *Key refreshment:* To refresh the user certificate pairs, two users U_ζ and U_η respectively perform the following two tasks.

(1) The user U_ζ computes $h = H(PK||T||V)$, picks two random values $m_k, n_k \in Z_p^*$ in the k-th session, and computes
✓ $USK_{\zeta_{pair_k}} = (USK_{\zeta,k,1}, USK_{\zeta,k,2}) = (USK_{\zeta,k-1,1} \cdot g^{m_k}, USK_{\zeta,k-1,2} \cdot g^{-m_k})$,
✓ $UCA_{\zeta_{pair_k}} = (UCA_{\zeta,k,1}, UCA_{\zeta,k,2}) = (UCA_{\zeta,k-1,1} \cdot g^{n_k}, UCA_{\zeta,k-1,2} \cdot g^{-n_k})$,
✓ $UCB_{\zeta_{pair_k}} = (UCB_{\zeta,k,1}, UCB_{\zeta,k,2}) = (UCB_{\zeta,k-1,1} \cdot h^{n_k}, UCB_{\zeta,k-1,2} \cdot h^{-n_k})$.
(2) The user U_η computes $h = H(PK||T||V)$, picks two random values $m_l, n_l \in Z_p^*$ in the l-th session, and computes
✓ $USK_{\eta_{pair_l}} = (USK_{\eta,l,1}, USK_{\eta,l,2}) = (USK_{\eta,l-1,1} \cdot g^{m_l}, USK_{\eta,l-1,2} \cdot g^{-m_l})$,
✓ $UCA_{\eta_{pair_l}} = (UCA_{\eta,l,1}, UCA_{\eta,l,2}) = (UCA_{\eta,l-1,1} \cdot g^{n_l}, UCA_{\eta,l-1,2} \cdot g^{-n_l})$,
✓ $UCB_{\eta_{pair_l}} = (UCB_{\eta,l,1}, UCB_{\eta,l,2}) = (UCB_{\eta,l-1,1} \cdot h^{n_l}, UCB_{\eta,l-1,2} \cdot h^{-n_l})$.

• *Key agreement:* Let U_ζ and U_η be two participants. U_ζ and U_η select ephemeral secret keys $ESK_\zeta = x$ and $ESK_\eta = y \in Z_p^*$, and compute $X = g^x$ and $Y = g^y$, respectively. U_ζ sends X and her/his two public keys FPK_ζ and SPK_ζ to U_η while U_η sends Y and her/his two public keys FPK_η and SPK_η to U_ζ. Then, U_ζ and U_η respectively perform the following two tasks.

(1) U_ζ computes the follows:
✓ $K_{\zeta,k,1} = Y^x$.
✓ $K_{\zeta,k,2} = \hat{e}(h^x, FPK_\eta)$, where $h = H(PK||T||V)$.
✓ $K_{\zeta,k,3} = PK^x \cdot \hat{e}(SPK_\eta, T \cdot V^{b_\eta})^x$, where $b_\eta = ID_\eta||FPK_\eta$.
✓ $K_{\zeta,k,4} = KT_{\zeta,k,4} \cdot \hat{e}(Y, USK_{\zeta,k,2})$, where $KT_{\zeta,k,4} = \hat{e}(Y, USK_{\zeta,k,1})$.
✓ $K_{\zeta,k,5} = KT_{\zeta,k,5} \cdot \hat{e}(FPK_\eta, USK_{\zeta,k,2})$, where $KT_{\zeta,k,5} = \hat{e}(FPK_\eta, USK_{\zeta,k,1})$.
✓ $K_{\zeta,k,6} = KT_{\zeta,k,6} \cdot \hat{e}(SPK_\eta, USK_{\zeta,k,2})$, where $KT_{\zeta,k,6} = \hat{e}(SPK_\eta, USK_{\zeta,k,1})$.
✓ $K_{\zeta,k,7} = KT_{\zeta,k,7} \cdot \hat{e}(Y, UCA_{\zeta,k,2})$, where $KT_{\zeta,k,7} = \hat{e}(Y, UCA_{\zeta,k,1})$.
✓ $K_{\zeta,k,8} = KT_{\zeta,k,8} \cdot \hat{e}(FPK_\eta, UCB_{\zeta,k,2})$, where $KT_{\zeta,k,8} = \hat{e}(FPK_\eta, UCB_{\zeta,k,1})$.
✓ $K_{\zeta,k,9} = KT_{\zeta,k,9} \cdot \hat{e}(SPK_\eta, UCB_{\zeta,k,2})$, where $KT_{\zeta,k,9} = \hat{e}(SPK_\eta, UCB_{\zeta,k,1})$.
(2) U_η computes the follows:
✓ $K_{\eta,l,1} = X^y$.
✓ $K_{\eta,l,2} = KT_{\eta,l,2} \cdot \hat{e}(X, USK_{\eta,l,2})$, where $KT_{\eta,l,2} = \hat{e}(X, USK_{\eta,l,1})$.
✓ $K_{\eta,l,3} = KT_{\eta,l,3} \cdot \hat{e}(X, UCA_{\eta,l,2})$, where $KT_{\eta,l,3} = \hat{e}(X, UCA_{\eta,l,1})$.
✓ $K_{\eta,l,4} = \hat{e}(h^y, FPK_\zeta)$, where $h = H(PK||T||V)$.

$$K_{\zeta,k,1} = Y^x = g^{yx} = X^y = K_{\eta,l,1}$$

$$K_{\zeta,k,2} = \hat{e}(h^x, FPK_\eta) = \hat{e}(h^x, g^{u_\eta}) = \hat{e}(g^x, h^{u_\eta}) = \hat{e}(X, USK_{\eta,l}) = \hat{e}(X, USK_{\eta,l,1}) \cdot \hat{e}(X, USK_{\eta,l,2}) = KT_{\eta,l,2} \cdot \hat{e}(X, USK_{\eta,l,2}) = K_{\eta,l,2}$$

$$K_{\zeta,k,3} = PK^x \cdot \hat{e}(SPK_\eta, T \cdot V^{b_\eta})^x = \hat{e}(g, SK)^x \cdot \hat{e}(g^{\beta_\eta}, (T \cdot V^{b_\eta})^x) = \hat{e}(g^x, SK) \cdot \hat{e}(g^x, (T \cdot V^{b_\eta})^{\beta_\eta}) = \hat{e}(X, SK \cdot (T \cdot V^{b_\eta})^{\beta_\eta}) = \hat{e}(X, UCA_{\eta,l,1}) \cdot \hat{e}(X, UCA_{\eta,l,2}) = KT_{\eta,l,3} \cdot \hat{e}(X, UCA_{\eta,l,2}) = K_{\eta,l,3}$$

$$K_{\zeta,k,4} = KT_{\zeta,k,4} \cdot \hat{e}(Y, USK_{\zeta,k,2}) = \hat{e}(Y, USK_{\zeta,k,1}) \cdot \hat{e}(Y, USK_{\zeta,k,2}) = \hat{e}(Y, USK_{\zeta,k}) = \hat{e}(g^y, h^{u_\zeta}) = \hat{e}(h^y, g^{u_\zeta}) = \hat{e}(h^y, FPK_\zeta) = K_{\eta,l,4}$$

$$K_{\zeta,k,5} = KT_{\zeta,k,5} \cdot \hat{e}(FPK_\eta, USK_{\zeta,k,2}) = \hat{e}(FPK_\eta, USK_{\zeta,k,1}) \cdot \hat{e}(FPK_\eta, USK_{\zeta,k,2}) = \hat{e}(FPK_\eta, USK_{\zeta,k}) = \hat{e}(g^{u_\eta}, h^{u_\zeta}) = \hat{e}(g^{u_\zeta}, h^{u_\eta}) = \hat{e}(g^{u_\zeta}, USK_{\eta,l,1}) \cdot \hat{e}(g^{u_\zeta}, USK_{\eta,l,2}) = KT_{\eta,l,5} \cdot \hat{e}(FPK_\zeta, USK_{\eta,l,2}) = K_{\eta,l,5}$$

$$K_{\zeta,k,6} = KT_{\zeta,k,6} \cdot \hat{e}(SPK_\eta, USK_{\zeta,k,2}) = \hat{e}(SPK_\eta, USK_{\zeta,k,1}) \cdot \hat{e}(SPK_\eta, USK_{\zeta,k,2}) = \hat{e}(SPK_\eta, USK_{\zeta,k}) = \hat{e}(g^{\beta_\eta}, h^{u_\zeta}) = \hat{e}(g^{u_\zeta}, h^{\beta_\eta}) = \hat{e}(FPK_\zeta, UCB_{\eta,l}) = \hat{e}(FPK_\zeta, UCB_{\eta,l,1}) \cdot \hat{e}(FPK_\zeta, UCB_{\eta,l,2}) = KT_{\eta,l,6} \cdot \hat{e}(FPK_\zeta, UCB_{\eta,l,2}) = K_{\eta,l,6}$$

$$K_{\zeta,k,7} = KT_{\zeta,k,7} \cdot \hat{e}(Y, UCA_{\zeta,k,2}) = \hat{e}(Y, UCA_{\zeta,k,1}) \cdot \hat{e}(Y, UCA_{\zeta,k,2}) = \hat{e}(Y, UCA_{\zeta,k}) = \hat{e}(g^y, SK \cdot (T \cdot V^{b_\zeta})^{\beta_\zeta}) = \hat{e}(g^y, SK) \cdot \hat{e}(g^y, (T \cdot V^{b_\zeta})^{\beta_\zeta}) = \hat{e}(g, SK)^y \cdot \hat{e}(g^{\beta_\zeta}, (T \cdot V^{b_\zeta})^y) = PK^y \cdot \hat{e}(SPK_\zeta, T \cdot V^{b_\zeta})^y = K_{\eta,l,7}$$

$$K_{\zeta,k,8} = KT_{\zeta,k,8} \cdot \hat{e}(FPK_\eta, UCB_{\zeta,k,2}) = \hat{e}(FPK_\eta, UCB_{\zeta,k,1}) \cdot \hat{e}(FPK_\eta, UCB_{\zeta,k,2}) = \hat{e}(FPK_\eta, UCB_{\zeta,k}) = \hat{e}(g^{u_\eta}, h^{\beta_\zeta}) = \hat{e}(g^{\beta_\zeta}, h^{u_\eta}) = \hat{e}(SPK_\zeta, USK_{\eta,l}) = \hat{e}(SPK_\zeta, USK_{\eta,l,1}) \cdot \hat{e}(SPK_\zeta, USK_{\eta,l,2}) = KT_{\eta,l,8} \cdot \hat{e}(SPK_\zeta, USK_{\eta,l,2}) = K_{\eta,l,8}$$

$$K_{\zeta,k,9} = KT_{\zeta,k,9} \cdot \hat{e}(SPK_\eta, UCB_{\zeta,k,2}) = \hat{e}(SPK_\eta, UCB_{\zeta,k,1}) \cdot \hat{e}(SPK_\eta, UCB_{\zeta,k,2}) = \hat{e}(SPK_\eta, UCB_{\zeta,k}) = \hat{e}(g^{\beta_\eta}, SK^{\beta_\zeta}) = \hat{e}(g^{\beta_\zeta}, SK^{\beta_\eta}) = \hat{e}(SPK_\zeta, UCB_{\eta,l}) = \hat{e}(SPK_\zeta, UCB_{\eta,l,1}) \cdot \hat{e}(SPK_\zeta, UCB_{\eta,l,2}) = KT_{\zeta,l,9} \cdot \hat{e}(SPK_\zeta, UCB_{\eta,l,2}) = K_{\eta,l,9}$$

FIGURE 2. Nine equalities.

✓ $K_{\eta,l,5} = KT_{\eta,l,5} \cdot \hat{e}(FPK_\zeta, USK_{\eta,l,2})$, where $KT_{\eta,l,5} = \hat{e}(FPK_\zeta, USK_{\eta,l,1})$.
✓ $K_{\eta,l,6} = KT_{\eta,l,6} \cdot \hat{e}(FPK_\zeta, UCB_{\eta,l,2})$, where $KT_{\eta,l,6} = \hat{e}(FPK_\zeta, UCB_{\eta,l,1})$.
✓ $K_{\eta,l,7} = PK^y \cdot \hat{e}(SPK_\zeta, T \cdot V^{b_\zeta})^y$, where $b_\zeta = ID_\zeta || FPK_\zeta$.
✓ $K_{\eta,l,8} = KT_{\eta,l,8} \cdot \hat{e}(SPK_\zeta, USK_{\eta,l,2})$, where $KT_{\eta,l,8} = \hat{e}(SPK_\zeta, USK_{\eta,l,1})$.
✓ $K_{\eta,l,9} = KT_{\eta,l,9} \cdot \hat{e}(SPK_\zeta, UCB_{\eta,l,2})$, where $KT_{\eta,l,9} = \hat{e}(SPK_\zeta, UCB_{\eta,l,1})$.

• *Common session key derivation:* The common session key can be established by U_ζ and U_η as presented below. Also, Fig. 2 shows that the common session key $CSK_{\zeta,k}$ is equal to the common session key $CSK_{\eta,l}$.

- (1) U_ζ compute $CSK_{\zeta,k} = K_{\zeta,k,1} \oplus K_{\zeta,k,2} \oplus K_{\zeta,k,3} \oplus K_{\zeta,k,4} \oplus K_{\zeta,k,5} \oplus K_{\zeta,k,6} \oplus K_{i,k,7} \oplus K_{\zeta,k,8} \oplus K_{\zeta,k,9}$.
- (2) U_η compute $CSK_{\eta,l} = K_{\eta,l,1} \oplus K_{\eta,l,2} \oplus K_{\eta,l,3} \oplus K_{\eta,l,4} \oplus K_{\eta,l,5} \oplus K_{\eta,l,6} \oplus K_{i,l,7} \oplus K_{\eta,l,8} \oplus K_{\eta,l,9}$.

V. SECURITY ANALYSIS

One theorem and two lemmas are given in this section. The proof of the theorem employs the the lemmas to prove that the proposed LR-CB-AKE protocol is secure in the GBG model under the DL and CDH assumptions.

Theorem 1: In the GBG model, the proposed LR-CB-AKE protocol is secure in the security game if the DL and CDH assumptions hold.

Proof: Assume that U_ζ and U_η are two participants in the proposed LR-CB-AKE protocol, and they possess a partnership. We denote $\prod_\zeta^k$ as an oracle with the participant U_ζ in the k-th session, and $\prod_\eta^l$ as another oracle with the participant U_η in the l-th session. Note that $\prod_\zeta^k$ and $\prod_\eta^l$ are two oracles in the partnership session. With these two oracles, a session key can be established. The session key, which can be respectively calculated by $\prod_\zeta^k$ and $\prod_\eta^l$, is composed of user secret key, user certificates and ephemeral secret keys. As mentioned in Section III-B, there exists an adversary $\mathcal{A}$ who wants to guess the correct session key to win the security game. During the security games, the adversary $\mathcal{A}$ can issue the *User secret key generation query*, *User certificate generation query* and *Ephemeral key reveal query* to obtain the user secret key, user certificates and ephemeral secret keys, respectively. According to the definition of freshness, there are nine circumstances as discussed below.

✓*Circumstance 1:* Neither the ESK of $\prod_\zeta^k$ nor $\prod_\eta^l$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain USK and (UCA, UCB) of $\prod_\zeta^k$ or $\prod_\eta^l$.

✓*Circumstance 2:* Neither the USK of $\prod_\zeta^k$ nor $\prod_\eta^l$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain ESK and (UCA, UCB) of $\prod_\zeta^k$ or $\prod_\eta^l$.

✓*Circumstance 3:* Neither the (UCA, UCB) of $\prod_\zeta^k$ nor $\prod_\eta^l$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain ESK and USK of $\prod_\zeta^k$ or $\prod_\eta^l$.

✓Circumstance 4: Neither the ESK of $\prod_{\zeta}^{k}$ nor the USK of $\prod_{\eta}^{l}$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain other keys of $\prod_{\zeta}^{k}$ or $\prod_{\eta}^{l}$.

✓Circumstance 5: Neither the ESK of $\prod_{\zeta}^{k}$ nor the (UCA, UCB) of $\prod_{\eta}^{l}$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain other keys of $\prod_{\zeta}^{k}$ or $\prod_{\eta}^{l}$.

✓Circumstance 6: Neither the USK of $\prod_{\zeta}^{k}$ nor the ESK of $\prod_{\eta}^{l}$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain other keys of $\prod_{\zeta}^{k}$ or $\prod_{\eta}^{l}$.

✓Circumstance 7: Neither the USK of $\prod_{\zeta}^{k}$ nor the (UCA, UCB) of $\prod_{\eta}^{l}$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain other keys of $\prod_{\zeta}^{k}$ or $\prod_{\eta}^{l}$.

✓Circumstance 8: Neither the (UCA, UCB) of $\prod_{\zeta}^{k}$ nor the ESK of $\prod_{\eta}^{l}$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain other keys of $\prod_{\zeta}^{k}$ or $\prod_{\eta}^{l}$.

✓Circumstance 9: Neither the (UCA, UCB) of $\prod_{\zeta}^{k}$ nor the USK of $\prod_{\eta}^{l}$ can be obtained by $\mathcal{A}$, but $\mathcal{A}$ is able to gain other keys of $\prod_{\zeta}^{k}$ or $\prod_{\eta}^{l}$.

For the above circumstances, we use two Lemmas 3 and 4 to provide the security analysis. Based on the two lemmas, the proposed LR-CB-AKE protocol is secure in the security game.

Lemma 3: Under Circumstance 1, the proposed LR-CB-AKE protocol is secure in the GBG model if the CDH assumption holds.

Proof: We know that Circumstance 1 allows $\mathcal{A}$ to gain USK and (UCA, UCB) of $\prod_{\zeta}^{k}$ or $\prod_{\eta}^{l}$. Therefore, by these obtained keys, $K_{\zeta,k,i}$ $(= K_{\eta,l,i})$, for $i = 2, 3, \ldots, 9$, can be computed. However, neither the ephemeral secret key x of $\prod_{\zeta}^{k}$ nor he ephemeral secret key y of $\prod_{\eta}^{l}$ can be obtained by $\mathcal{A}$ in Circumstance 1. $\mathcal{A}$ cannot obtain g^{xy} $(= K_{\zeta,k,1} = K_{\eta,l,1})$ by the two given values $X = g^{x}$ and $Y = g^{y}$ due to the CDH assumption. Since the composition of the session key requires all the nine keys $K_{\zeta,k,i}$ $(i = 1, 2, \ldots, 9)$, $\mathcal{A}$ cannot calculate the session key due to the lack of $K_{\zeta,k,1}$. Although $\mathcal{A}$ is restricted from gaining $K_{\zeta,k,1}$, some leaked information of $ESK = x$ or y from the *Send leak query* can be obtained by $\mathcal{A}$. However, the leaked information obtained in each session is independent since x and y are randomly reselected in each new session. Therefore, the leaked information doesn't help $\mathcal{A}$ to calculate $K_{\zeta,k,1}$ or $K_{\eta,l,1}$. Under the CDH assumption, $\mathcal{A}$'s probability of winning the security game can be ignored.

Lemma 4: Under Circumstances 2 to 9, the proposed LR-CB-AKE protocol is secure in the GBG model if the DL assumption holds.

Proof: The GBG model provides security analysis of secret key leakage, and it converts each element in the group into a different bit-string. As mentioned in Section II-B, to express the multiplications of G_1 and G_2 and the computation of $\hat{e}$ in the GBG model, we have three group operations GOP_1, GOP_2 and GOP_p via an algorithm $\mathcal{B}$. In the security game, an adversary $\mathcal{A}$ can query about these three group operations. The algorithm $\mathcal{B}$, who attempts to solve the DL problem (assumption), plays the role of the challenger and interacts with the adversary $\mathcal{A}$ in the following security game.

– *Setup phase:* The challenger $\mathcal{B}$ performs the *System setup* algorithm to obtain the system secret key SK and public parameters PP of the LR-CB-AKE protocol. The public parameters PP are set as $\{p, G_1, G_2, g, \hat{e}, PK, T, V, H\}$, where p, G_1 and G_2 are defined as in Section II, and g, PK, T, V and H are encoded as the associated bit-strings. Then, $\mathcal{B}$ sends the public parameters to $\mathcal{A}$, and gives the system secret key to $\mathcal{A}$ if $\mathcal{A}$ is a Type 2 adversary. In order to record $\mathcal{A}$'s queries, including inputs and outputs, $\mathcal{B}$ prepares five lists $L_1, L_2, L_{USK}, L_{UC}$ and L_S as follows. Notice that all the five lists contain the polynomial representations, since we employ the Lemma 2 to complete the security analysis.

- L_1 and L_2 are used to record the polynomial representation of elements of G_1 and G_2, and the corresponding bit-strings of elements of G_1 and G_2 after transformation, respectively.

✓L_1 records pairs in the form of $(\mathbb{P}G_{1,m,n,r}, \mathbb{B}G_{1,m,n,r})$, where $\mathbb{P}G_{1,m,n,r}$ is the polynomial representation of an element of G_1 and $\mathbb{B}G_{1,m,n,r}$ is the corresponding bit-string with m type of query, n-th query and r-th element. Meanwhile six pairs $(\mathbb{P}SK, \mathbb{B}G_{1,S,0,1})$, $(\mathbb{P}g, \mathbb{B}G_{1,S,0,2})$, $(\mathbb{P}T, \mathbb{B}G_{1,S,0,3})$, $(\mathbb{P}V, \mathbb{B}G_{1,S,0,4})$, $(\mathbb{P}H, \mathbb{B}G_{1,S,0,5})$ and $(\mathbb{P}h, \mathbb{B}G_{1,S,0,6})$ are added in L_1 by $\mathcal{B}$.

✓L_2 records pairs in the form of $(\mathbb{P}G_{2,m,n,r}, \mathbb{B}G_{2,m,n,r})$, where $\mathbb{P}G_{2,m,n,r}$ is the polynomial representation of an element of G_2 and $\mathbb{B}G_{2,m,n,r}$ is the corresponding bit-string with m type of query, n-th query and r-th element. One pair $(\mathbb{P}PK, \mathbb{B}G_{2,S,0,1})$ is added in L_2 by $\mathcal{B}$, where $\mathbb{P}PK = \mathbb{P}g \cdot \mathbb{P}SK$.

The following two transformations TF-1 and TF-2 are employed to assist $\mathcal{B}$ in answering $\mathcal{A}$'s queries about L_1/L_2.

I. TF-1: When $\mathcal{A}$'s query is $\mathbb{P}G_{1,m,n,r}/\mathbb{P}G_{2,m,n,r}$, $\mathcal{B}$ uses TF-1 to search L_1/L_2. If it is found, the corresponding $\mathbb{B}G_{1,m,n,r}/\mathbb{B}G_{2,m,n,r}$ will be returned; otherwise, a bit-string will be randomly selected as $\mathbb{B}G_{1,m,n,r}/\mathbb{B}G_{2,m,n,r}$ to be returned. In addition, $\mathcal{B}$ adds $(\mathbb{P}G_{1,m,n,r}, \mathbb{B}G_{1,m,n,r})/(\mathbb{P}G_{2,m,n,r}, \mathbb{B}G_{2,m,n,r})$ into L_1/L_2.

I. TF-2: When $\mathcal{A}$'s query is $\mathbb{B}G_{1,m,n,r}/\mathbb{B}G_{2,m,n,r}$, $\mathcal{B}$ uses TF-2 to search L_1/L_2. If it is found, the corresponding $\mathbb{P}G_{1,m,n,r}/\mathbb{P}G_{2,m,n,r}$ will be returned; otherwise, $\mathcal{B}$ returns $\perp$.

•L_{USK} records $(ID_\zeta, \mathbb{P}USK_\zeta, \mathbb{P}FPK_\zeta)$. Here, ID_ζ is the user U_ζ's identity. $\mathbb{P}USK_\zeta$ and $\mathbb{P}FPK_\zeta$ are, respectively, the multivariate polynomials of the user secret key USK_ζ and the user's first public key FPK_ζ.

•L_{UC} records $(ID_\zeta, \mathbb{P}FPK_\zeta, \mathbb{P}UCA_\zeta, \mathbb{P}UCB_\zeta, \mathbb{P}SPK_\zeta)$. Here, ID_ζ is a user U_ζ's identity. $\mathbb{P}FPK_\zeta, \mathbb{P}UCA_\zeta, \mathbb{P}UCB_\zeta$ and $\mathbb{P}SPK_\zeta$ are multivariate polynomials of the user's first

public key FPK_ζ, two user certificates UCA_ζ, UCB_ζ, and the user's second public key SPK_ζ, respectively.

- L_S records ($\prod_\zeta^k$, $\mathbb{B}PN_{\zeta,k}$, $\mathbb{B}PNFPK_{\zeta,k}$, $\mathbb{B}PNSPK_{\zeta,k}$, $ESK_{\zeta,k}$, $CSK_{\zeta,k}$, $\mathbb{B}TM_{\zeta,k}$, $\mathbb{P}TM_{\zeta,k}$, $\mathbb{B}PNTM_{\zeta,k}$, $\mathbb{P}PNTM_{\zeta,k}$). Here, $\prod_\zeta^k$ is an oracle with the participant U_ζ in the k-th session. The remaining items are the communication details of $\prod_\zeta^k$ and defined as follows.

✓ $\mathbb{B}PN_{\zeta,k}$: the identity of U_ζ's partner in the k-th session.
✓ $\mathbb{B}PNFPK_{\zeta,k}$: the first public key of U_ζ's partner in the k-th session.
✓ $\mathbb{B}PNSPK_{\zeta,k}$: the second public key of U_ζ's partner in the k-th session.
✓ $ESK_{\zeta,k}$: ESK of U_ζ in the k-th session.
✓ $CSK_{\zeta,k}$: CSK of U_ζ in the k-th session.
✓ $\mathbb{B}TM_{\zeta,k}$: the transmitted message of $\prod_\zeta^k$ with the representation of a bit-string.
✓ $\mathbb{P}TM_{\zeta,k}$: the transmitted message of $\prod_\zeta^k$ with the representation of a multivariate polynomial.
✓ $\mathbb{B}PNTM_{\zeta,k}$: the transmitted message of U_ζ's partner in the k-th session with the representation of a bit-string.
✓ $\mathbb{P}PNTM_{\zeta,k}$: the transmitted message of U_ζ's partner in the k-th session with the representation of the multivariate polynomial.

– *Query phase:* This phase allows $\mathcal{A}$ to make different queries as follows.

- *GOP_1 query* ($\mathbb{B}G_{1,Q,r,1}$, $\mathbb{B}G_{1,Q,r,2}$, $calc$): When receiving this query in the r-th query, $\mathcal{B}$ returns $\mathbb{B}G_{1,Q,r,3}$ by performing the follows.

✓ Transform ($\mathbb{B}G_{1,Q,r,1}$, $\mathbb{B}G_{1,Q,r,2}$) into ($\mathbb{P}G_{1,Q,r,1}$, $\mathbb{P}G_{1,Q,r,2}$) by using TF-2.
✓ Compute $\mathbb{P}G_{1,Q,r,3}$ according to $\mathbb{P}G_{1,Q,r,1}$, $\mathbb{P}G_{1,Q,r,2}$ and $calc$. If $calc$ = "multiplication," set $\mathbb{P}G_{1,Q,r,3} = \mathbb{P}G_{1,Q,r,1} + \mathbb{P}G_{1,Q,r,2}$. If $calc$ = "division," set $\mathbb{P}G_{1,Q,r,3} = \mathbb{P}G_{1,Q,r,1} - \mathbb{P}G_{1,Q,r,2}$.
✓ Transform $\mathbb{P}G_{1,Q,r,3}$ into $\mathbb{B}G_{1,Q,r,3}$ by using TF-1.

- *GOP_2 query* ($\mathbb{B}G_{2,Q,r,1}$, $\mathbb{B}G_{2,Q,r,2}$, $calc$): When receiving this query in the r-th query, $\mathcal{B}$ returns $\mathbb{B}G_{2,Q,r,3}$ by performing the follows.

✓ Transform ($\mathbb{B}G_{2,Q,r,1}$, $\mathbb{B}G_{2,Q,r,2}$) into ($\mathbb{P}G_{2,Q,r,1}$, $\mathbb{P}G_{2,Q,r,2}$) by using TF-2.
✓ Compute $\mathbb{P}G_{2,Q,r,3}$ according to $\mathbb{P}G_{2,Q,r,1}$, $\mathbb{P}G_{2,Q,r,2}$ and $calc$. If $calc$ = "multiplication," set $\mathbb{P}G_{2,Q,r,3} = \mathbb{P}G_{2,Q,r,1} + \mathbb{P}G_{2,Q,r,2}$. If $calc$ = "division," set $\mathbb{P}G_{2,Q,r,3} = \mathbb{P}G_{2,Q,r,1} - \mathbb{P}G_{2,Q,r,2}$.
✓ Transform $\mathbb{P}G_{2,Q,r,3}$ into $\mathbb{B}G_{2,Q,r,3}$ by using TF-1.

- *GOP_p query* ($\mathbb{B}G_{1,P,r,1}$, $\mathbb{B}G_{1,P,r,2}$): When receiving this query in the r-th query, $\mathcal{B}$ returns $\mathbb{B}G_{1,P,r,3}$ by performing the follows..

✓ Transform ($\mathbb{B}G_{1,P,r,1}$, $\mathbb{B}G_{1,P,r,2}$) into ($\mathbb{P}G_{1,P,r,1}$, $\mathbb{P}G_{1,P,r,2}$) by using TF-2.
✓ Compute $\mathbb{P}G_{1,P,r,3} = \mathbb{P}G_{1,P,r,1} \cdot \mathbb{B}G_{1,P,r,2}$ according to $\mathbb{P}G_{1,P,r,1}$ and $\mathbb{B}G_{1,P,r,2}$.
✓ Transform $\mathbb{P}G_{1,P,r,3}$ into $\mathbb{B}G_{1,P,r,3}$ by using TF-1.

- *User secret key generation query* (ID_ζ): $\mathcal{B}$ looks for ID_ζ in L_{USK}, and transforms ($\mathbb{P}USK_\zeta$, $\mathbb{P}FPK_\zeta$) into ($\mathbb{B}USK_\zeta$, $\mathbb{B}FPK_\zeta$) by using TF-1 if ID_ζ exists in L_{USK}. Then, $\mathcal{B}$ sends ($\mathbb{B}USK_\zeta$, $\mathbb{B}FPK_\zeta$) to $\mathcal{A}$. If ID_ζ does not exist in L_{USK}, $\mathcal{B}$ proceeds the following steps.

✓ Set the polynomial $\mathbb{P}USK_\zeta = \mathbb{P}RG_{USK,i,1}$, where $\mathbb{P}RG_{USK,i,1}$ is a new variate in G_1.
✓ Set the polynomial $\mathbb{P}FPK_\zeta = \mathbb{P}SG_{USK,i,1}$, where $\mathbb{P}SG_{USK,i,1}$ is a new variate in G_1.
✓ Add ($\mathbb{P}USK_\zeta$, $\mathbb{P}FPK_\zeta$) in L_{USK}.
✓ Transform ($\mathbb{P}USK_\zeta$, $\mathbb{P}FPK_\zeta$) into ($\mathbb{B}USK_\zeta$, $\mathbb{B}FPK_\zeta$) by using TF-1.
✓ Return ($\mathbb{B}USK_\zeta$, $\mathbb{B}FPK_\zeta$) to $\mathcal{A}$.

- *User certificate generation* (ID_ζ, FPK_ζ): $\mathcal{B}$ looks for ID_ζ and FPK_ζ in L_{UC}, and transforms ($\mathbb{P}UCA_\zeta$, $\mathbb{P}UCB_\zeta$, $\mathbb{P}SPK_\zeta$) into ($\mathbb{B}UCA_\zeta$, $\mathbb{B}UCB_\zeta$, $\mathbb{B}SPK_\zeta$) by using TF-1 if ID_ζ exists in L_{UC}. Then, $\mathcal{B}$ sends ($\mathbb{B}UCA_\zeta$, $\mathbb{B}UCB_\zeta$, $\mathbb{B}SPK_\zeta$) to $\mathcal{A}$. If ID_ζ does not exist in L_{UC}, $\mathcal{B}$ proceeds the following steps.

✓ Set the polynomial $\mathbb{P}UCA_\zeta = \mathbb{P}SK + \mathbb{P}RG_{UC,i,1} \cdot (\mathbb{P}T + \mathbb{P}RID \cdot \mathbb{P}V)$, where $\mathbb{P}RG_{UC,i,1}$ is a new variate in G_1 and $\mathbb{P}RID = ID_\zeta || FPK_\zeta$.
✓ Set the polynomial $\mathbb{P}UCB_\zeta = \mathbb{P}SG_{UC,i,1}$, where $\mathbb{P}SG_{UC,i,1}$ is a new variate in G_1.
✓ Set the polynomial $\mathbb{P}SPK_\zeta = \mathbb{P}TG_{UC,i,1}$, where $\mathbb{P}TG_{UC,i,1}$ is a new variate in G_1.
✓ Add ($\mathbb{P}UCA_\zeta$, $\mathbb{P}UCB_\zeta$, $\mathbb{P}SPK_\zeta$) in L_{UC}.
✓ Transform ($\mathbb{P}UCA_\zeta$, $\mathbb{P}UCB_\zeta$, $\mathbb{P}SPK_\zeta$) into ($\mathbb{B}UCA_\zeta$, $\mathbb{B}UCB_\zeta$, $\mathbb{B}SPK_\zeta$) by using TF-1.
✓ Return ($\mathbb{B}UCA_\zeta$, $\mathbb{B}UCB_\zeta$, $\mathbb{B}SPK_\zeta$) to $\mathcal{A}$.

- *User certificate generation leak query* (ID_ζ, FPK_ζ, $f_{\mathrm{UCG},i}$, $h_{\mathrm{UCG},i}$): $\mathcal{B}$ takes ID_ζ, FPK_ζ, $f_{\mathrm{UCG},i}$ and $h_{\mathrm{UCG},i}$ as input, and returns $\Lambda_{f_{\mathrm{UCG},i}} = f_{\mathrm{UCG},i}(SK_{i,1})$ and $\Lambda_{h_{\mathrm{UCG},i}} = h_{\mathrm{UCG},i}(SK_{i,2})$.
- *Replace public key* (ID_ζ, $\mathbb{B}FPK'_\zeta$, $\mathbb{B}SPK'_\zeta$): $\mathcal{B}$ first respectively transforms $\mathbb{B}FPK'_\zeta$ and $\mathbb{B}SPK'_\zeta$ into $\mathbb{P}FPK'_\zeta$ and $\mathbb{P}SPK'_\zeta$ by using TF-2, and then uses $\mathbb{P}FPK'_\zeta$ and $\mathbb{P}SPK'_\zeta$ to update the lists L_{USK} and L_{UC}.
- *Ephemeral-secret-corrupt* ($\prod_\zeta^k$): $\mathcal{B}$ looks for $\prod_\zeta^k$ in L_S, and returns $ESK_{\zeta,k}$ if $\prod_\zeta^k$ exists in L_S. Otherwise, $\mathcal{B}$ returns "false".
- *Send* ($\prod_\zeta^k$, $\mathbb{B}PNTM_{\zeta,k}$, $\mathbb{B}PN_{\zeta,k}$, $\mathbb{B}PNFPK_{\zeta,k}$, $\mathbb{B}PNSPK_{\zeta,k}$): $\mathcal{B}$ looks for ($\prod_\zeta^k$, $\mathbb{B}TRM_{\zeta,k}$, $\mathbb{B}PN_{\zeta,k}$, $\mathbb{B}PNFPK_{\zeta,k}$, $\mathbb{B}PNSPK_{\zeta,k}$) in L_S, and returns $\mathbb{B}TM_{\zeta,k}$ if ($\prod_\zeta^k$, $\mathbb{B}TRM_{\zeta,k}$, $\mathbb{B}PN_{\zeta,k}$, $\mathbb{B}PNFPK_{\zeta,k}$, $\mathbb{B}PNSPK_{\zeta,k}$) exists in L_S. If this record does not exist in L_S, $\mathcal{B}$ does the following steps.

✓ Transform $\mathbb{B}PNTM_{\zeta,k}$ into $\mathbb{P}PNTM_{\zeta,k}$ by using TF-2.

TABLE 3. Time Required for Bilinear Pairing and Exponentiation Operations

Operations	T_{pair}	T_{exp}
Computational cost	7.8351 ms	0.4746 ms

✓ Set $\mathbb{P}TM_{\zeta,k} = ESK_{\zeta,k} \cdot \mathbb{P}g$, where $ESK_{\zeta,k} \in Z_p^*$ is an ephemeral secret key chosen in random.
✓ Transform $\mathbb{P}TM_{\zeta,k}$ into $\mathbb{B}TM_{\zeta,k}$ by using TF-1, and send it to $\mathcal{A}$.
✓ Add ($\prod_\zeta^k$, $\mathbb{B}PN_{\zeta,k}$, $\mathbb{B}PNFPK_{\zeta,k}$, $\mathbb{B}PNSPK_{\zeta,k}$, $ESK_{\zeta,k}$, -, $\mathbb{B}TM_{\zeta,k}$, $\mathbb{P}TM_{\zeta,k}$, $\mathbb{B}PNTM_{\zeta,k}$, $\mathbb{P}PNTM_{\zeta,k}$) in L_S.

- *Send leak query* ($\prod_\zeta^k$, $f_{\mathrm{CSK},\zeta,k}$, $h_{\mathrm{CSK},\zeta,k}$): $\mathcal{B}$ takes $\prod_\zeta^k$, $f_{\mathrm{CSK},\zeta,k}$ and $h_{\mathrm{CSK},\zeta,k}$ as input, and returns $\Lambda_{f_{\mathrm{CSK},\zeta,k}} = f_{\mathrm{CSK},\zeta,k}(USK_{\zeta,k,1}, UCA_{\zeta,k,1}, UCB_{\zeta,k,1}, ESK_{\zeta,k})$ and $\Lambda_{h_{\mathrm{CSK},\zeta,k}} = h_{\mathrm{CSK},\zeta,k}(USK_{\zeta,k,2}, UCA_{\zeta,k,2}, UCB_{\zeta,k,2}, K_{\zeta,k,1}, K_{\zeta,k,2}, \ldots, K_{\zeta,k,9})$.
- *Reveal query* ($\prod_\zeta^k$): $\mathcal{B}$ takes $\prod_\zeta^k$ as input, and returns the session key by performing the follows.

✓ Use $\prod_\zeta^k$ to find $\mathbb{B}PN_{\zeta,k}$ in L_S.
✓ Transform $\mathbb{B}PN_{\zeta,k}$ into $\mathbb{P}PN_{\zeta,k}$ by using TF-2.
✓ Use $\mathbb{B}PN_{\zeta,k}$ to find partner's public keys $\mathbb{P}PNFPK_\zeta$ in L_{USK} and $\mathbb{P}PNSPK_\zeta$ in L_{UC}.
✓ Obtain the corresponding user secret key $\mathbb{P}USK_\zeta$ in L_{USK} and user certificates ($\mathbb{P}UCA_\zeta$, $\mathbb{P}UCB_\zeta$) in L_{UC} by using the identity ID_ζ of $\prod_\zeta^k$.
✓ Set $ESK_{\zeta,k} = RESK_{\zeta,k}$, where $RESK_{\zeta,k}$ a new variable.
✓ Compute $K_{\zeta,k,1}, K_{\zeta,k,2}, \ldots, K_{\zeta,k,9}$ as follows.
◇ $K_{\zeta,k,1} = ESK_{\zeta,k} \cdot \mathbb{P}TRM_{\zeta,k}$.
◇ $K_{\zeta,k,2} = ESK_{\zeta,k} \cdot \mathbb{P}h \cdot \mathbb{P}PNFPK_\zeta$.
◇ $K_{\zeta,k,3} = ESK_{\zeta,k} \cdot (\mathbb{P}PK + \mathbb{P}PNSPK_\zeta \cdot (\mathbb{P}T + \mathbb{P}PN_{\zeta,k} \cdot \mathbb{P}V))$.
◇ $K_{\zeta,k,4} = \mathbb{P}TRM_{\zeta,k} \cdot \mathbb{P}USK_\zeta$.
◇ $K_{\zeta,k,5} = \mathbb{P}PNFPK_\zeta \cdot \mathbb{P}USK_\zeta$.
◇ $K_{\zeta,k,6} = \mathbb{P}PNSPK_\zeta \cdot \mathbb{P}USK_\zeta$.
◇ $K_{\zeta,k,7} = \mathbb{P}TRM_{\zeta,k} \cdot \mathbb{P}UCA_\zeta$.
◇ $K_{\zeta,k,8} = \mathbb{P}PNFPK_\zeta \cdot \mathbb{P}UCB_\zeta$.
◇ $K_{\zeta,k,9} = \mathbb{P}PNSPK_\zeta \cdot \mathbb{P}UCB_\zeta$.
✓ Add $K_{\zeta,k,1}$ in L_1 and $K_{\zeta,k,2}$, $K_{\zeta,k,3}, \ldots, K_{\zeta,k,9}$ in L_2. Transform $(K_{\zeta,k,1}, K_{\zeta,k,2}, \ldots, K_{\zeta,k,9})$ to $(\mathbb{B}K_{\zeta,k,1}, \mathbb{B}K_{\zeta,k,2}, \ldots, \mathbb{B}K_{\zeta,k,9})$ by using TF-1.
✓ Set the session key $CSK_{\zeta,k} = \mathbb{B}K_{\zeta,k,1} \oplus \mathbb{B}K_{\zeta,k,2} \oplus \ldots \oplus \mathbb{B}K_{\zeta,k,9}$ as the session key.

- *Test query* ($\prod_\zeta^k$): $\mathcal{B}$ takes $\prod_\zeta^k$ as input, and returns "false" if $\prod_\zeta^k$ does not exist in L_S. Otherwise, $\mathcal{B}$ can obtain $CSK_{\zeta,k}$ from the *Reveal query* ($\prod_\zeta^k$). Then, $\mathcal{B}$ gives $\mathcal{A}$ a session key according to the result of a random $coin \in \{0, 1\}$. If $coin = 1$, the session key is $CSK_{\zeta,k}$; otherwise, the session key is a random value from session key space.

To analyze $\mathcal{A}$'s advantage of winning the security game, the number of elements and each polynomial's degree in L_1 and L_2 must be calculated.
– Calculate the number of elements in L_1 and L_2.

- The *Setup phase* adds 6 and 1 elements in L_1 and L_2, respectively.
- At most 3 elements accede to L_1 or L_2 in each GOP_1, GOP_2 or GOP_p *query*.
- At most 2 elements accede to L_1 or L_2 in each *Send query*.
- At most 10 elements accede to L_1 or L_2 in each *Reveal query*, *User secret key generation query* and *User certificate generation query*.

Assume that $|L_1|$ and $|L_2|$ denote respectively as the numbers of elements in L_1 and L_2. Then, we obtain $|L_1| + |L_2| \leq 6 + 3q_{GOP} + 2q_S + 10q_R + 10q_{USK} + 10q_{UC} \leq 10q$, where q_{GOP} is the total requested number of GOP_1, GOP_2 and GOP_p queries, and q_S, q_R, q_{USK}, and q_{UC}, respectively, are the number of *Send*, *Reveal*, *User secret key generation*, and *User certificate generation queries*.
– Calculate the degree of each polynomial in L_1 and L_2.

- In L_1, according to the following description, each polynomial has degree at most 3.

✓ $\mathbb{P}SK$, $\mathbb{P}g$, $\mathbb{P}T$, $\mathbb{P}V$, $\mathbb{P}H$ and $\mathbb{P}h$ with degree 1 accede to L_1 in the *Setup phase*.
✓ $\mathbb{P}USK_\zeta$ and $\mathbb{P}FPK_\zeta$ with degree 1 appear in the *User secret key generation query*.
✓ $\mathbb{P}UCA_\zeta$, $\mathbb{P}UCB_\zeta$ and $\mathbb{P}SPK_\zeta$ with degrees 3, 1 and 1, respectively, appear in the *User certificate generation query*.
✓ $K_{\zeta,k,1}$ with degree 2 appears in the *Reveal query*.
✓ $\mathbb{P}G_{Q,r,3}$ with degree 3 appears in the GOP_1 *query*.

- In L_2, according to the following description, each polynomial has degree at most 6.

✓ $\mathbb{P}PK$ with degree 2 accede to L_2 in the Setup phase.
✓ $\mathbb{P}G_{2,Q,r,3}$ has the maximal degree of $\mathbb{P}G_{2,Q,r,1}$ and $\mathbb{P}G_{2,Q,r,2}$ in the GOP_2 *query*.
✓ $\mathbb{P}G_{2,P,r,1}$ with degree 6 appears in the GOP_p *query*.
✓ $K_{\zeta,k,4}$, $K_{\zeta,k,5}, \ldots, K_{\zeta,k,9}$ have degree 2 while $K_{\zeta,k,2}$ and $K_{\zeta,k,3}$ have degrees 3 and 4 in the *Reveal query*.

Next, we split into two cases to analyze $\mathcal{A}$'s advantage of winning the security game.
–*Case 1:* Assume that $\mathcal{A}$ wins the security game without issuing the *User certificate generation leak query* or *Send leak query*. Then, at least one of the following two events will happen.

- $Event_1$ states that $\mathcal{A}$ finds a collision of two distinct polynomials in L_1 or L_2. Let $\mathbb{P}G_{1,i}$ and $\mathbb{P}G_{1,j}$ be any two distinct polynomials in L_1, and n be the number of all variates in L_1. $\mathcal{A}$ computes $\mathbb{P}G_{1,C}(x_1, x_2, \ldots, x_n) = \mathbb{P}G_{1,i} - \mathbb{P}G_{1,j}$, where $x_i \in Z_p^*$ for $i = 1, 2, \ldots, n$. If $\mathbb{P}G_{1,i} = \mathbb{P}G_{1,j}$, the collision occurs, i.e. $\mathbb{P}G_{1,C}(x_1, x_2, \ldots, x_n) = 0$. According to Lemma 2, the probability of $\mathbb{P}G_{1,C}(x_1, x_2, \ldots, x_n) = 0$ is at most $3/p$ since the polynomial in L_1 has degree at most 3. Hence, the collision probability is $(3/p)\binom{|L_1|}{2}$ since there are $\binom{|L_1|}{2}$ ways to select two distinct polynomials $\mathbb{P}G_{1,i}$ and $\mathbb{P}G_{1,j}$ in L_1, By similar analysis, the probability of the collision in L_2 is $(6/p)\binom{|L_2|}{2}$. The probability of $Event_1$ is

$$\begin{aligned}\Pr[Event_1] &\leqq (3/p)\binom{|L_1|}{2} + (6/p)\binom{|L_2|}{2} \\ &\leqq (6/p)(|L_1| + |L_2|)^2 \\ &\leqq 600q^2/p, \text{ since } |L_1| + |L_2| \leqq 10q.\end{aligned}$$

TABLE 4. Comparison of Our LR-CB-AKE With Existing LR-ID-AKE and CB-AKE

	Wu *et al.*'s LR-ID-AKE [32]	Lu *et al.*'s CB-AKE [6]	Our LR-CB-AKE
System setup cost	$T_{pair} + 5T_{exp}$ (10.2081 ms)	T_{exp} (0.4746 ms)	$T_{pair} + T_{exp}$ (8.3097 ms)
User secret key extract cost	$11T_{exp}$ (5.2206 ms)	$2T_{exp}$ (0.9492 ms)	$10T_{exp}$ (4.746 ms)
Session key construction cost	$3T_{pair} + 11T_{exp}$ (28.7259 ms)	$T_{pair} + 8T_{exp}$ (11.6319 ms)	$14T_{pair} + 11T_{exp}$ (114.912 ms)
Avoiding key escrow issues	No	Yes	**Yes**
Resisting side-channel attacks	Yes	No	**Yes**
Restriction of leaked information	Unbounded	-	**Unbounded**

• $Event_2$ states that $\mathcal{A}$ cannot find a collision of two distinct polynomials in L_1 and L_2. The probability of $Event_2$ is defined as $\Pr[Event_2] \leqq 1/2$, since the probability in the *Test query* is 1/2.

We respectively denote $\Pr_{\mathcal{A}-C1}$ and $Adv_{\mathcal{A}-C1}$ as the probability and the advantage of winning the security game in the Case 1. Then, we have

$$\Pr_{\mathcal{A}-C1} \leqq \Pr[Event_1] + \Pr[Event_2] \leqq 600q^2/p + 1/2,$$

$$Adv_{\mathcal{A}-C1} \leqq |600q^2/p + 1/2 - 1/2| = 600q^2/p = O(q^2/p).$$

Hence, under the situation of $q = poly(\log p)$, $Adv_{\mathcal{A}-C1}$ is negligible.

– *Case 2:* Assume that $\mathcal{A}$ wins the security game with issuing *User certificate generation leak query* and *Send leak query*.

• $(UCA_{\zeta,k,1}, UCA_{\zeta,k,2})$: Although the leaked information of $UCA_{\zeta,k,1}$ and $UCA_{\zeta,k,2}$ can be obtained in each session, the leaked information is mutually independent due to the property $UCA_{\zeta} = UCA_{\zeta,0,1} \cdot UCA_{\zeta,0,2} = UCA_{\zeta,1,1} \cdot UCA_{\zeta,1,2} = \cdots = UCA_{\zeta,k,1}$, $UCA_{\zeta,k,2}$ and the refreshing technique. The leaked information of $UCA_{\zeta,k,1}$ and $UCA_{\zeta,k,2}$ is bounded with λ bits.

• $(UCB_{\zeta,k,1}, UCB_{\zeta,k,2})$: By similar analysis, the leaked information of $UCB_{\zeta,k,1}$ and $UCB_{\zeta,k,2}$ is at most λ bits.

• $(K_{\zeta,k,1}, K_{\zeta,k,2}, K_{\zeta,k,3}, K_{\zeta,k,4}, K_{\zeta,k,5}, K_{\zeta,k,6}, K_{\zeta,k,7}, K_{\zeta,k,8}, K_{\zeta,k,9})$: Undoubtedly, a session key $CSK_{\zeta,k}$ can be calculated by using $K_{\zeta,k,1}, K_{\zeta,k,2}, \ldots, K_{\zeta,k,9}$, while the leaked information of the session key $CSK_{\zeta,k}$ is at most λ bits. It is worth noting that the leaked information in each session is different due to the freshness property mentioned in Definition 2.

Next, we split into three events to analyze $\mathcal{A}$'s advantage of winning the security game in Case 2.

(1) $Event_{SK}$ denotes the event that $\mathcal{A}$ can gain SK by using $\Lambda_{f_{\mathrm{UCG},i}}$ and $\Lambda_{h_{\mathrm{UCG},i}}$. Assume that $\overline{Event_{SK}}$ is the complement event of $Event_{SK}$.

(2) $Event_{UC}$ denotes the event that $\mathcal{A}$ can gain UCA_{ζ} and UCB_{ζ} by using $\Lambda_{f_{\mathrm{CSK},\zeta,k}}$ and $\Lambda_{h_{\mathrm{CSK},\zeta,k}}$. Assume that $\overline{Event_{UC}}$ is the complement event of $Event_{UC}$.

(3) $Event_{coin}$ denotes the event that $\mathcal{A}$'s guess $coin'$ is correct in the *Test phase*.

We denote $\Pr[\mathcal{A}]$ as the probability of winning the security game in the Case 2. Then, we have

$$\begin{aligned}\Pr[\mathcal{A}] &= \Pr[Event_{coin}] \\ &= \Pr[Event_{coin} \wedge (Event_{SK} \vee Event_{UC})] \\ &\quad + \Pr[Event_{coin} \wedge (\overline{Event_{SK}} \wedge \overline{Event_{UC}})] \\ &\leqq \Pr[Event_{SK} \vee Event_{UC}] \\ &\quad + \Pr[Event_{coin} \wedge (\overline{Event_{SK}} \wedge \overline{Event_{UC}})].\end{aligned}$$

Here, we can obtain $\Pr[Event_{coin} \wedge (\overline{Event_{SK}} \wedge \overline{Event_{UC}})] = 1/2$ since $\mathcal{A}$'s average probability of guessing under the condition $\overline{Event_{SK}} \wedge \overline{Event_{UC}}$ is only 1/2. Thus, we have

$$\Pr[\mathcal{A}] \leq \Pr[Event_{SK} \vee Event_{UC}] + 1/2.$$

We denote $Adv_{\mathcal{A}}$ as the advantage of winning the security game in the Case 2. Then, we have

$$Adv_{\mathcal{A}} \leq |\Pr[\mathcal{A}] - 1/2| = \Pr[Event_{SK} \vee Event_{UC}].$$

In Case 1, the advantage of winning the security game is $Adv_{\mathcal{A}-C1} \leq 600q^2/p = O(q^2/p)$. Since the leaked information of USK or ESK is at most 2λ bits, we have

$$Adv_{\mathcal{A}} \leq Adv_{\mathcal{A}-C1} \cdot 2^{2\lambda} \leq O\left((q^2/p) \cdot 2^{2\lambda}\right).$$

Hence, under the situation of $\lambda < \log p - \omega(\log\log p)$ and Corollary 1, $Adv_{\mathcal{A}}$ is negligible.

VI. PERFORMANCE ANALYSIS AND COMPARISONS

We compare the performance and properties between our LR-CB-AKE protocol, the LR-ID-AKE protocol [32] and the CB-AKE protocol [6]. For the performance analysis, two notations are defined to benchmark the computational cost of system setup, user secret key extract or session key construction.

- • T_{pair}: the time required for a bilinear pairing operation $\hat{e} : G_1 \times G_1 \rightarrow G_2$.
- • T_{exp}: the time required for an exponentiation operation in G_1 or G_2.

According to the simulation results performed in [38], we have $T_{pair} = 7.8351$ ms and $T_{exp} = 0.4746$ ms, as shown in Table 3. This result is obtained by the Intel Core i7-8550 U CPU 1.80 Ghz processor and using a finite field F_p, G_1 and G_2 as the input parameters for simulation. Here, p is a prime number with 256 bits, and G_1 and G_2 are groups that has 224 bits prime order over the finite field F_p.

Table 4 shows the comparisons of our LR-CB-AKE with the existing LR-ID-AKE [32] and CB-AKE [6] in terms of computational cost and security properties. For the computation cost, it is obvious that the CB-AKE [6] is the best. However, the CB-AKE cannot withstand side-channel attacks. When the system secret key is leaked, the adversary could break the system and obtain the full system secret key. On

the other hand, although the LR-ID-AKE [32] can withstand side-channel attacks, there is inborn problem, namely, key escrow problem. So, the PKG holds each user's private key and can perform signature or decryption procedures all by itself. Our LR-CB-AKE can not only withstand side-channel attacks, but also eliminate the key escrow problem.

VII. CONCLUSION

In this article, we proposed the *first* LR-CB-AKE protocol, which resists side channel attacks. We defined the framework of LR-CB-AKE protocols, and considered the leakage resilient properties and the security models of the existing CB-AKEs protocols to give a new security model for LR-CB-AKE protocols. The proposed protocol was formally proven to be secure in the GBG model under the CDH and DL assumptions. As compared with the previous LR-ID-AKE and CB-AKE protocols, our LR-CB-AKE protocol not only withstands side-channel attacks, but also eliminates the key escrow problem.

REFERENCES

[1] C. Gentry, "Certificate-based encryption and the certificate revocation problem," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, 2003, vol. 2656, pp. 272–293.

[2] S. Wang and Z. Cao, "Escrow-free certificate-based authenticated key agreement protocol from pairings," *Wuhan Univ. J. Natural Sci.*, vol. 12, no. 1, pp. 63–66, Jan. 2007.

[3] M. Lim, S. Lee, and H. Lee, "An improved variant of Wang-Cao's certificated-based authenticated key agreement protocol," in *Proc. 4th Int. Conf. Netw. Comput. Adv. Inf. Manage.*, 2008, pp. 198–201.

[4] G. Lippold, C. Boyd, and J. Nieto, "Strongly secure certificateless key agreement," in *Proc. Int. Conf. Pairing-Based Cryptography*, 2009, vol. 5671, pp. 206–230.

[5] Y. Lu, Q. Zhang, J. Li, and J. Shen, "An efficient certificate-based authenticated key agreement protocol without bilinear pairing," *Inf. Technol. Control*, vol. 46, no. 3, pp. 345–359, Sep. 2017.

[6] Y. Lu, Q. Zhang, and J. Li, "A certificate-based AKA protocol secure against public key replacement attacks," *Int. Arab J. Inf. Technol.*, vol. 16, no. 4, pp. 754–765, Jul. 2019.

[7] T. Kubota, K. Yoshida, M. Shiozaki, and T. Fujino, "Deep learning side-channel attack against hardware implementations of AES," *Microprocessors Microsyst.*, vol. 87, Nov. 2021, Art. no. 103383.

[8] K. Ngo, E. Dubrova, Q. Guo, and T. Johansson, "A side-channel attack on a masked ind-CCA secure saber kem implementation," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2021, no. 4, pp. 676–707, 2021.

[9] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. Workshop Theory Appl. Cryptographic Techn.*, 1984, vol. 196, pp. 47–53.

[10] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Proc. Annu. Int. Cryptology Conf.*, 2001, vol. 2139, pp. 213–229.

[11] B. LaMacchia, K. Lauter, and A. Mityagin, "Stronger security of authenticated key exchange," in *Proc. Int. Conf. Provable Secur.*, 2007, vol. 4784, pp. 1–16.

[12] T.-Y. Wu, Z. Lee, M. S. Obaidat, S. Kumari, and C.-M. Chen, "An authenticated key exchange protocol for multi-server architecture in 5G networks," *IEEE Access*, vol. 8, pp. 28096–28018, 2020.

[13] T.-Y. Wu, T. Wang, Y.-Q. Lee, W. Zheng, S. Kumari, and S. Kumar, "Improved authenticated key agreement scheme for fog-driven IoT healthcare system," *Secur. Commun. Netw.*, vol. 2021, Jan. 2021, Art. no. 6658041.

[14] P. Gope and B. Sikdar, "A privacy-aware reconfigurable authenticated key exchange scheme for secure communication in smart grids," *IEEE Trans. Smart Grid*, vol. 12, no. 6, pp. 5335–5348, Nov. 2021.

[15] N. P. Smart, "Identity-based authenticated key agreement protocol based on Weil pairing," *Electron. Lett.*, vol. 38, no. 13, pp. 630–632, 2002.

[16] K. Shim, "Efficient ID-based authenticated key agreement protocol based on weil pairing," *Electron. Lett.*, vol. 39, no. 8, pp. 653–654, 2003.

[17] L. Chen, Z. Cheng, and N. P. Smart, "Identity-based key agreement protocols from pairings," *Int. J. Inf. Secur.*, vol. 6, no. 4, pp. 213–241, Jan. 2007.

[18] D. S. Gupta, S. H. Islam, M. S. Obaidat, P. Vijayakumar, N. Kumar, and Y. Park, "A provably secure and lightweight identity-based two-party authenticated key agreement protocol for IIoT environments," *IEEE Syst. J.*, vol. 15, no. 2, pp. 1732–1741, Jun. 2021.

[19] Y. Li, Q. Cheng, and W. Shi, "Security analysis of a lightweight identity-based two-party authenticated key agreement protocol for IIoT environments," *Secur. Commun. Netw.*, vol. 2021, Feb. 2021, Art. no. 5573886.

[20] J. Alwen, Y. Dodis, and D. Wichs, "Leakage-resilient public-key cryptography in the bounded-retrieval model," in *Proc. Annu. Int. Cryptology Conf.*, 2009, vol. 5677, pp. 36–54.

[21] Y. Dodis, K. Haralambiev, A. Lopez-Alt, and D. Wichs, "Efficient public key cryptography in the presence of key leakage," in *Proc. Int. Conf. Theory Appl. Cryptology Inf. Secur.*, 2010, vol. 6477, pp. 613–631.

[22] J. Alawatugoda, D. Stebila, and C. Boyd, "Modelling after-the-fact leakage for key exchange," in *Proc. 9th ACM Symp. Inf., Comput., Commun. Secur.*, 2014, pp. 207–216.

[23] R. Chen, Y. Mu, G. Yang, W. Susilo, and F. Guo, "Strong authenticated key exchange with auxiliary inputs," in *Proc. Des., Codes Cryptography*, 2017, vol. 85, pp. 145–173.

[24] J. Alawatugoda, D. Stebila, and C. Boyd, "Continuous after-the-fact leakage-resilient eCK-secure key exchange," in *Proc. IMA Int. Conf. Cryptography Coding*, 2015, vol. 9496, pp. 277–294.

[25] S. Dziembowski and S. Faust, "Leakage-resilient cryptography from the inner-product extractor," in *Proc. Int. Conf. Theory Appl. Cryptology Inf. Secur.*, 2011, vol. 7073, pp. 702–721.

[26] D. Galindo and S. Virek, "A practical leakage-resilient signature scheme in the generic group model," in *Proc. Int. Conf. Sel. Areas Cryptography*, 2012, vol. 7707, pp. 50–65.

[27] J.-D. Wu, Y.-M. Tseng, and S.-S. Huang, "Leakage-resilient ID-based signature scheme in the generic bilinear group model," *Secur. Commun. Netw.*, vol. 9, no. 17, pp. 3987–4001, Nov. 2016.

[28] J.-D. Wu, Y.-M. Tseng, and S.-S. Huang, "Efficient leakage-resilient authenticated key agreement protocol in the continual leakage eck model," *IEEE Access*, vol. 6, pp. 17130–17142, 2018.

[29] I. Elashry, Y. Mu, and W. Susilo, "A resilient identity-based authenticated key exchange protocol," *Secur. Commun. Netw.*, vol. 8, no. 13, pp. 2279–2290, Sep. 2015.

[30] Y. Hatri, A. Otmani, and K. Guenda, "Cryptanalysis of an identity-based authenticated key exchange protocol," *Int. J. Commun. Syst.*, vol. 31, no. 3, Feb. 2018, Art. no. e3477.

[31] O. Ruan, Y. Zhang, M. Zhang, J. Zhou, and L. Harn, "After-the-fact leakage-resilient identity-based authenticated key exchange," *IEEE Syst. J.*, vol. 12, no. 2, pp. 2017–2026, Jun. 2018.

[32] J.-D. Wu, Y.-M. Tseng, and S.-S. Huang, "An identity-based authenticated key exchange protocol resilient to continuous key leakage," *IEEE Syst. J.*, vol. 13, no. 4, pp. 3968–3979, Dec. 2019.

[33] M. Scott, "On the efficient implementation of pairing-based protocols," in *Proc. IMA Int. Conf. Cryptography Coding*, 2011, vol. 7089, pp. 296–308.

[34] D. Boneh, X. Boyen, and E. J. Goh, "Hierarchical identity-based encryption with constant size ciphertext," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2005, vol. 3494, pp. 440–456.

[35] V. Shoup, "Lower bounds for discrete logarithms and related problems," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, 1997, vol. 1233, pp. 256–266.

[36] D. Galindo, J. Großschädl, Z. Liu, P. K. Vadnala, and S. Vivek, "Implementation of a leakage-resilient ElGamal key encapsulation mechanism," *J. Cryptographic Eng.*, vol. 6, no. 3, pp. 229–238, 2016.

[37] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," *SIAM J. Comput.*, vol. 38, no. 1, pp. 97–139, Jan. 2008.

[38] Y. Li, Q. Cheng, X. Liu, and X. Li, "A secure anonymous identity-based scheme in new authentication architecture for mobile edge computing," *IEEE Syst. J.*, vol. 15, no. 1, pp. 935–946, Mar. 2021.