CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. (2016)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cpe.3915

SPECIAL ISSUE PAPER

Offline/online attribute-based encryption with verifiable outsourced decryption

Zechao Liu[1,2], Zoe L. Jiang[1,3‡], Xuan Wang[1,4,*,†], Xinyi Huang[5,6], S.M. Yiu[7] and Kunihiko Sadakane[8]

[1]*Harbin Institute of Technology, Shenzhen Graduate School, Shenzhen 518055, China*
[2]*Shenzhen Applied Technology Engineering Laboratory for Internet Multimedia Application, Shenzhen, China*
[3]*Guangdong Provincial Key Laboratory of High Performance Computing, Guangzhou, China*
[4]*Public Service Platform of Mobile Internet Application Security Industry, Shenzhen 518055, China*
[5]*Fujian Normal University, Fuzhou 350108, China*
[6]*Nanjing University of Information Science and Technology, Nanjing, Jiangsu 210044, China*
[7]*Department of Computer Science, The University of Hong Kong, Hong Kong, China*
[8]*School of Information Science and Technology, The University of Tokyo, Tokyo, Japan*

SUMMARY

In this big data era, service providers tend to put the data in a third-party cloud system. Social networking websites are typical examples. To protect the security and privacy of the data, data should be stored in encrypted form. This brings forth new challenges: how to allow different users to access only the authorized part of the data without decryption of the data. Attribute-based encryption (ABE) offers fine-grained access control policy over encrypted data such that users can decrypt successfully only if their attributes satisfy the policy. However, one drawback of ABE is that the computational cost grows linearly with the complexity of ciphertext policy or the number of attributes. The situation becomes worse for mobile devices with limited computing resources. To solve this problem, we adopt the offline/online technique combining with the verifiable outsourced computation technique to propose a new ciphertext-policy ABE scheme using bilinear groups in prime order, supporting the offline/online key generation and encryption, as well as the verifiable outsourced decryption. As a result, most computations of key generation and encryption can be executed offline, and the majority of computational workload in decryption can be outsourced to third parties. The scheme is selectively chosen-plaintext attack-secure in the standard model. We also provide the proof of verifiability on outsourced decryption. The simulation results show that our proposed scheme can effectively reduce the computational cost imposed on resource-constrained devices. Copyright © 2016 John Wiley & Sons, Ltd.

Received 30 December 2015; Revised 6 June 2016; Accepted 19 June 2016

KEY WORDS: offline/online ABE; outsourced decryption; verifiable computation; Charm

1. INTRODUCTION

In this big data era, data volume is huge. A typical example is social network. Social networking websites, such as Facebook, MySpace and Friendster, have become established platforms for people to keep in touch and for sharing personal information. The trend is for the service providers to store data in a third-party cloud system. However, social network data contain much sensitive information such as personal photos, gathering information and personal emails, which may be easily leaked or collected by those third-party systems. One method for alleviating the security problem

*Correspondence to: Xuan Wang, Harbin Institute of Technology, Shenzhen Graduate School, Shenzhen 518055, China.
†E-mail: wangxuan@cs.hitsz.edu.cn
‡Co-corresponding author

is to store data in encrypted form. For example, the uploader can encrypt his or her photos and specify the access policy as follows: only people satisfying (FAMILY MEMBERS OR (FEMALE AND CLASSMATES)) can successfully decrypt the photos. Attribute-based encryption (ABE) [1] can fulfill the aforementioned requirement and offers fine-grained access control over encrypted data such that different users can only access the authorized portion of the data in the social network using their private keys if their attributes satisfy the policy.

There are two complementary forms of ABE: key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE). In CP-ABE [2–4], a secret key is associated with user's attributes and access policy is built in ciphertext; while in KP-ABE [1, 5–7], secret key is associated with access policy and attributes are used to describe the encrypted data. Although ABE is suitable for providing fine-grained access control mechanism in social network, at least the following two problems should be resolved to make it practical:

1. **Efficiency**: One of the major drawbacks of existing ABE schemes is that the computational cost grows linearly with the number of attributes specified in the access policy [8]. The situation is worse if ABE is to be executed in resource-constrained devices such as mobile phone or sensors. And in social network applications, users usually use their mobile phones/devices to access the corresponding websites. Therefore, one challenge is how to move the majority of computation to the high-speed outsourcing servers and/or make use of the offline/online concept (see the following for a more detailed description) to cut down the amount of computation to be carried out online.
2. **Verifiable computation**: Outsourced computation is useful only when the returned result can be trusted, which makes verifiable computation a must for such scenarios [9]. The security of ABE with outsourced decryption ensures that an untrusted server will not learn anything about the encrypted data; however, it does not guarantee the correctness of the transformation carried out by the server [10]. Therefore, another challenge is how to guarantee the third parties honestly execute the outsourced computation.

1.1. Our contributions

We propose a new CP-ABE scheme to solve the problems. To tackle the efficiency problem, especially for resource-constrained devices, we propose to perform preprocessing work offline. In an ABE system with many users, the authorized server that generates users' private keys may become a bottleneck, especially when private keys are reissued each time period for revocation purpose [11]. The key-issuing server can perform preprocessing work offline, then the incoming online key requests can be processed quickly. The encryption can also be split into offline and online phases. Messages can be encrypted quickly during the online phase. Using the offline/online technique, the majority of operations are performed offline, and only lightweighted computation is required for the online phase. Besides, we adopt the outsourcing technique to reduce the decryption cost imposed on mobile devices. The outsourced decryption in our scheme has the property of verifiability. Thus, a user can efficiently check if the transformation is carried out correctly by an untrusted server, which also solves the problem of verifiable computation. The main idea of our scheme is described as follows.

Hohenberger *et al.* [11] proposed a concrete CP-ABE scheme with offline/online encryption and gave a brief description on how to design the offline/online key generation. They illustrated their idea in a key encapsulation mechanism (KEM) [12] model. This scheme does not support outsourced decryption, and the decryption result is not verifiable. We follow this scheme as our basic framework. First, we convert it to support outsourced decryption. Second, we adopt the Lin *et al.* [13] idea and further extend it to achieve verifiable outsourced decryption. Thus, our final construction can support offline/online key generation and encryption, as well as verifiable outsourced decryption simultaneously. Experimental results show that our proposed scheme can effectively reduce computational cost imposed on resource-constrained devices.

Note that our scheme is selectively secure in the standard model using bilinear groups in prime order. While full security is a stronger notion of security [14], selective security is still a meaningful notion and can be a reasonable trade-off for performance. Technically speaking, there are

DOI: 10.1002/cpe

several ways to achieve full security, either based on the result from pair encoding [15] or predicate encoding framework [16] or using the dual system group [17] besides dual pairing vector spaces or direct constructions. The security of our scheme is acceptable because the goal is for efficiency. On the other hand, the security will not be significantly lowered, in particular, for social networking applications. The major difference between selective security and full security is that the former requires the attacker to declare the target to attack (in our case, the access structure) before the start of the attack; that is, the system should know the target in advance. For social networking websites, these access structures could be predefined. It is reasonable to assume that the attacker would like to attack the structures within a small subgraph when he or she starts to query/collect information from nearby neighbours in the social network graph. So, selective security already provides certain security guarantee to the system.

1.2. Related work

Attribute-based encryption. The concept of ABE was first proposed by Sahai and Waters [1] in 2005. Then Goyal *et al*. [5] classified it into KP-ABE and CP-ABE and proposed a small universe KP-ABE supporting monotonic access tree. Bethencourt *et al*. [2] proposed the first CP-ABE construction in a generic group model. Cheung *et al*. [3] gave a CP-ABE scheme supporting AND-gates on positive and negative attributes. Ostrovsky *et al*. [6] extended previous ABE schemes to support non-monotonic access structure. Waters [4] proposed CP-ABE schemes in standard model under three different assumptions. Lewko *et al*. [18] proposed the first fully secure CP-ABE schemes in composite order bilinear groups, using Waters' dual system encryption methodology. Hohenberger *et al*. [19] proposed an ABE scheme with fast decryption, where ciphertexts can be decrypted with a constant number of pairing. Besides, a number of variants of ABE schemes have been proposed, such as ABE supporting user revocation [20, 21] and ABE with multi-authorities [22, 23].

Offline/Online Technique. The notion of offline/online technique was first introduced by Even *et al*. [24] for digital signature. In the offline phase, most of the work for signing a message is performed before the message is known, while the online phase requires light computations and hence can be executed efficiently, once the message is presented. Later, Shamir *et al*. [25] proposed a new paradigm called *hash-sign-switch* for designing more efficient offline/online signature schemes. Liu *et al*. [26] proposed an offline/online ring signature scheme, which does not need to know the signers in the offline phase. Baek *et al.* [27] focused on identity-based offline/online signature scheme. Guo *et al*. [28], on the other hand, proposed an identity-based offline/online encryption scheme. Pre-computation was referred as offline phase, and the real encryption was considered as online phase. It was further improved in [29] for efficiency and much shorter ciphertext. There are other offline/online schemes applied to different primitives. For examples, the authors in [30, 31] worked on signcryption; Liu *et al.* [32] proposed an identity-based offline/online key encapsulation mechanism that splits the process of key encapsulation into offline and online phases; and Rouselakis *et al.* [11] proposed an offline/online ABE scheme supporting offline/online encryption and key generation.

Outsourced computation. Recently, Green *et al*. [33] proposed a new paradigm by outsourcing the main decryption computation and proved its security in replayable chosen ciphertext attack (CCA) model. This scheme is similar to the concept of proxy re-encryption [34, 35], which allows a proxy to transform an encryption of message m under Alice's public key into an encryption of the same m under Bob's public key without the proxy learning anything about the encrypted message [10]. One way of outsourcing the decryption is to delegate pairing operations on ABE ciphertexts to a more powerful device. And the pairing delegation was proposed in [36, 37]. Although schemes in [38, 39] considered verifiable computation, they are based on fully homomorphic encryption. Gentry *et al*. [40] has shown that it would take at least 30 s on a high-performance machine, even for weak security parameters on 'bootstrapping' operation of the homomorphic encryption [8, 33]. Lai *et al*. [10] and Lin *et al*. [13] provided practical schemes with verifiable outsourced decryption. Li *et al*. [41] provided the scheme with outsourced encryption. Li *et al*. [8, 42] provided ABE schemes with outsourced key issuing and decryption.

DOI: 10.1002/cpe

1.3. Organization

The rest of this paper is organized as follows. In Section 2, we introduce some background information and cryptographic definitions. Section 3 contains our new models of the proposed CP-ABE scheme. Our concrete construction and the security proof are given in Section 4. Experimental results on the performance of our proposed scheme are shown in Section 5. Finally, we conclude the paper in Section 6.

2. PRELIMINARIES

In this section, we first give background information on bilinear maps. Then we give formal definitions for access structure and linear secret-sharing schemes (LSSS). Finally, we briefly introduce key derivation function and key encapsulation mechanism.

2.1. Bilinear maps

We review bilinear maps described in [4] as follows.
Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}$ and e be a bilinear map, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The bilinear map e has the following properties:

1. Bilinearity: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degeneracy: $e(g, g) \neq 1$.

Also, the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is efficiently computable. The map e is symmetric because $e(g^a, g^b) = e(g^b, g^a) = e(g, g)^{ab}$.

2.2. Access structures

We review the definition of access structure described in [43] as follows.

Definition 1
Access structure [43]: Let $\{P_1, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}$ is monotone for $\forall$ B, C: if $B \in \mathbb{A}$, $B \subseteq C$, then $C \in \mathbb{A}$. An access structure (monotone access structure) is a collection (monotone collection) $\mathbb{A}$ of nonempty subsets of $\{P_1, \ldots, P_n\}$, that is, $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}} \backslash \{\phi\}$. The sets in $\mathbb{A}$ are called authorized sets, and sets not in $\mathbb{A}$ are called unauthorized sets.

We focus on monotone access structure in this paper.

2.3. Linear secret-sharing schemes

We review the definition of LSSS described in [4] as follows:

Definition 2
Linear secret-sharing schemes [4]: A secret-sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if

1. The shares for each party form a vector over $\mathbb{Z}_p$.
2. There exists a matrix M with ℓ rows and n columns called the share-generating matrix for Π. For all $i = 1, \ldots, \ell$, the ith row of M, we let the function ρ defined the party labelling row i as $\rho(i)$. When we consider the column vector $v = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared and $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen, then Mv is the vector of ℓ shares of the secret s according to Π. The share $(Mv)_i$ belongs to party $\rho(i)$.

It is shown in [4] that every LSSS according to the aforementioned definition also has the *linear reconstruction* property, defined as follows: Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, 2, \ldots, \ell\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \omega_i \lambda_i = s$. Furthermore, it is shown in [4] that these constants ω_i can be found in polynomial time in the size of the share-generating matrix M.

DOI: 10.1002/cpe

2.4. Key derivation function

We review the definition of key derivation function (KDF) described in [13] as follows.

Definition 3
A KDF [13] takes as input a secret key DK and a length l, then it outputs a string of l bits. KDF is secure if for any probabilistic polynomial-time adversary $\mathcal{A}$, the following is negligible:

$$|Pr[\mathcal{A}(KDF(DK, l)) = 1] - Pr[\mathcal{A}(R) = 1]|.$$

Notice that DK denotes an initial secret key, l is the output length and R is chosen uniformly at random from $\{0, 1\}^l$.

2.5. Key encapsulation mechanism

Definition 4
Key encapsulation mechanism [12] is an asymmetric encryption technique that generates simultaneously a random key K_S together with its encryption C, referred as encapsulation. Then the key K_S is used for data encryption, while the encapsulation C is used for sharing K_S.

3. NEW MODEL OF PROPOSED CP-ABE SCHEME

In this section, we propose three different kinds of ABE schemes. For each scheme, we also give its security model. For simplicity and convenience, we only describe CP-ABE here.

Definition 5
An offline/online CP-ABKEM scheme consists of the following six algorithms:

$Setup(\lambda, U) \longrightarrow (PK, MSK)$. It takes as input a security parameter λ and the attribute universe U, then outputs public parameters PK and master secret key MSK.

$Offline.KeyGen(MSK) \longrightarrow IK$. It takes as input the master secret key MSK, then outputs an intermediate key IK.

$Online.KeyGen(PK, IK, S) \longrightarrow SK_S$. It takes as input public parameters PK, the intermediate key IK and a set of attributes S. It outputs a private key SK_S associated with the attributes.

$Offline.Encrypt(PK) \longrightarrow IT$. It takes as input public parameters PK, then outputs an intermediate ciphertext IT.

$Online.Encrypt(PK, IT, \mathbb{A}) \longrightarrow (key, CT)$. It takes as input public parameters PK, the intermediate ciphertext IT and an access structure $\mathbb{A}$, then outputs the session key key and ciphertext CT.

$Decrypt(PK, SK_S, CT) \longrightarrow key$. It takes as input public parameters PK, a private key SK_S and a ciphertext CT associated with access structure $\mathbb{A}$. If attribute set S satisfies $\mathbb{A}$, then the algorithm outputs a session key key. Otherwise, it outputs the error message $\perp$.

Note that the aforementioned scheme is the same as the Hohenberger *et al.* scheme [11], except that we replace its key generation algorithm with $Offline.KeyGen$ and $Online.KeyGen$ algorithms.

In the following, we describe the security model for offline/online CP-AB-KEM (OO-CP-ABKEM) scheme. The experiment OO-CP-ABKEM$_{\mathcal{A},\Pi}(\lambda, U)$ proceeds as follows:

- Setup. The challenger $\mathcal{C}$ runs $Setup$ algorithm to obtain public parameters PK and gives it to adversary $\mathcal{A}$.
- Phase 1. The challenger $\mathcal{C}$ initializes an empty table T, an empty set D and an integer counter $j = 0$. The adversary $\mathcal{A}$ adaptively issues queries:
  * Create(S): $\mathcal{C}$ runs $Online.KeyGen(PK, Offline.KeyGen(MSK), S)$ to obtain the private key SK_S, then sets $j = j + 1$ and stores the entry (j, S, SK_S) in table T .
  * Corrupt(i): If the i^{th} entry exists in table T, $\mathcal{C}$ obtains the entry (i, S, SK_S) and sets $D = D \cup \{S\}$. Then $\mathcal{C}$ sends private key SK_S to $\mathcal{A}$. Otherwise, it returns $\perp$.

DOI: 10.1002/cpe

∗ Decrypt(i, CT): If the i^{th} entry exists in table T, $\mathcal{C}$ obtains the entry (i, S, SK_S) and executes the decryption algorithm on input (SK_S, CT). Then $\mathcal{C}$ sends the decryption result to $\mathcal{A}$. Otherwise, it returns $\perp$.

- Challenge. The adversary $\mathcal{A}$ submits an access structure $\mathbb{A}^*$, with restriction that for all $S \in D$, $\mathbb{A}^*$ cannot be satisfied by S. $\mathcal{C}$ first runs $Online.Encrypt(PK, Offline.Encrypt(PK), \mathbb{A}^*)$ to obtain (key^*, CT^*). Then $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$. If $b = 0$, it returns (key^*, CT^*) to $\mathcal{A}$. If $b = 1$, it chooses a random session key R and returns (R, CT^*) to $\mathcal{A}$.
- Phase 2. $\mathcal{A}$ continues to issue queries as in Phase 1, but with the restrictions that
  ∗ $\mathcal{A}$ cannot issue a Corrupt query which attributes set S satisfies the access structure $\mathbb{A}^*$.
  ∗ $\mathcal{A}$ cannot issue a Decrypt query on the challenge ciphertext CT^*.
- Guess. The adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. The experiment returns 1 if and only if $b' = b$.

Definition 6 (Offline/online CP-ABKEM security)
An OO-CP-ABKEM scheme is secure against CCA (CCA-secure) for attribute universe U if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage in this security game is negligible:

$$\left| Pr\left[\text{OO-CP-ABKEM}_{\mathcal{A},\Pi}(\lambda, U) = 1\right] - \frac{1}{2} \right| \leqslant negl(\lambda).$$

CPA security. The OO-CP-ABKEM scheme is secure against chosen-plaintext attacks (CPA-secure) if we remove the Decrypt oracle in both Phases 1 and 2.
Selective security. The OO-CP-ABKEM scheme is $selectively$ secure if we add an Init stage before the Setup stage where the adversary commits to the challenger an access structure $\mathbb{A}^*$ to attack.

Based on the OO-CP-ABKEM scheme, we modify it and let it work in a traditional ABE scheme where plaintext message is encrypted. And we construct an offline/online CP-ABE scheme.

Definition 7
The offline/online CP-ABE (OO-CP-ABE) scheme consists of the following six algorithms:

$Setup(\lambda, U) \longrightarrow (PK, MSK)$. It is the same as $Setup(\lambda, U)$ defined in Definition 5, except that the output of public parameters PK also includes a key derivation function.
$Offline.KeyGen(MSK) \longrightarrow IK$. It is the same as $Offline.KeyGen(MSK)$ defined in Definition 5.
$Online.KeyGen(PK, IK, S) \longrightarrow SK_S$. It is the same as $Online.KeyGen(MSK)$ defined in Definition 5.
$Offline.Encrypt(PK) \longrightarrow IT$. It is the same as $Offline.Encrypt(PK)$ defined in Definition 5.
$Online.Encrypt(PK, IT, M, \mathbb{A}) \longrightarrow CT$. It takes as input public parameters PK, the intermediate ciphertext IT, a message M and an access structure $\mathbb{A}$, then outputs a ciphertext CT.
$Decrypt(PK, SK_S, CT) \longrightarrow M$. It takes as input public parameters PK, a private key SK_S and a ciphertext CT associated with access structure $\mathbb{A}$, then returns a message M.

In the following, we describe the security model for OO-CP-ABE scheme. The experiment OO-CP-ABE$_{\mathcal{A},\Pi}(\lambda, U)$ proceeds as follows:

- Setup. The challenger $\mathcal{C}$ runs $Setup$ algorithm to obtain public parameters PK and gives it to adversary $\mathcal{A}$.
- Phase 1. The challenger $\mathcal{C}$ initializes an empty table T, an empty set D and an integer counter $j = 0$. The adversary $\mathcal{A}$ adaptively issues queries:
  ∗ Create(S): $\mathcal{C}$ runs $Online.KeyGen(PK, Offline.KeyGen(MSK), S)$ to obtain the private key SK_S, then sets $j = j + 1$ and stores the entry (j, S, SK_S) in table T.

DOI: 10.1002/cpe

∗ Corrupt(i): If the i^{th} entry exists in table T, $\mathcal{C}$ obtains the entry (i, S, SK_S) and sets $D = D \cup \{S\}$. Then $\mathcal{C}$ sends private key SK_S to $\mathcal{A}$. Otherwise, it returns $\perp$.

- Challenge. $\mathcal{A}$ submits two messages M_0, M_1 with equal length and an access structure $\mathbb{A}^*$ to the challenger $\mathcal{C}$. For all the private key query in Phase 1, $\mathbb{A}^*$ cannot be satisfied by S, where $S \in D$. $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, then sends to $\mathcal{A}$ the challenge ciphertext CT^* generated from $Online.Encrypt(PK, Offline.Encrypt(PK), M_b, \mathbb{A}^*) \longrightarrow CT^*$.
- Phase 2. $\mathcal{A}$ continues to issue queries as in Phase 1, but with the restrictions that
  ∗ $\mathcal{A}$ cannot issue a Corrupt query which attributes set S satisfies the access structure $\mathbb{A}^*$.
- Guess. The adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. The experiment returns 1 if and only if $b' = b$.

Definition 8 (Offline/online CP-ABE security)
An OO-CP-ABE scheme is CPA-secure for attribute universe U if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage in this security game is negligible:

$$\left| Pr\left[\text{OO-CP-ABE}_{\mathcal{A},\Pi}(\lambda, U) = 1\right] - \frac{1}{2} \right| \leqslant negl(\lambda).$$

Selective security. The OO-CP-ABE scheme is $selectively$ secure if we add an Init stage before the Setup stage where the adversary commits to challenger an access structure $\mathbb{A}^*$ to attack.

Based on the OO-CP-ABE scheme, we construct the final scheme that adds the function of verifiable outsourced decryption.

Definition 9
The offline/online key generation and encryption and verifiable outsourced decryption CP-ABE (OO-VO-CPABE) scheme consists of the following ten algorithms:

$Setup(\lambda, U) \longrightarrow (PK, MSK)$. The setup algorithm is the same as $Setup(\lambda, U)$ defined in Definition 7.

$Offline.KeyGen(MSK) \longrightarrow IK$. The offline key generation algorithm is the same as $Offline.KeyGen(MSK)$ defined in Definition 7.

$Online.KeyGen(PK, IK, S) \longrightarrow SK_S$. The online key generation algorithm is the same as $Online.KeyGen(PK, IK, S)$ defined in Definition 7.

$Offline.Encrypt(PK) \longrightarrow IT$. The offline encryption algorithm is the same as $Offline.Encrypt(PK)$ defined in Definition 7.

$Online.Encrypt(PK, IT, M, \mathbb{A}) \longrightarrow CT$. The online encryption algorithm is the same as $Online.Encrypt(PK, IT, M, \mathbb{A})$ defined in Definition 7.

$Decrypt(PK, SK_S, CT) \longrightarrow M$. The decryption algorithm is the same as $Decrypt(PK, SK_S, CT)$ defined in Definition 7.

$GenTK_{out}(PK, SK_S) \longrightarrow (TK_S, RK_S)$. The transform key generation algorithm takes as input public parameters PK and a private key SK_S. Then it outputs a transform key TK_S and a secret value RK_S used for recovering the plaintext message.

$Transform_{out}(PK, CT, TK_S) \longrightarrow CT'$. The transformation algorithm takes as input public parameters PK, the ciphertext CT and a transform key TK_S . Then it outputs a transformed ciphertext CT'.

$Verify(PK, CT', RK_S) \longrightarrow (b, st)$. The verification algorithm takes as input the public parameters PK, a transformed ciphertext CT' and a secret value RK_S. Then it outputs two-tuples (b, st), where $b \in \{0, 1\}$ and st stores the status value.

$Decrypt_{out}(PK, CT, CT', RK_S) \longrightarrow M$. The decryption algorithm takes as input public parameters PK, a ciphertext CT, a transformed ciphertext CT' and a secret value RK_S. Then it runs $Verify(PK, CT', RK_S)$ to obtain two-tuples (b, st). If $b = 1$, it can output M. If $b = 0$, it outputs an error symbol $\perp$.

DOI: 10.1002/cpe

Correctness. For all $(PK, MSK) \longleftarrow Setup(\lambda, U)$, $IK \longleftarrow Offline.KenGen(MSK)$, $SK_S \longleftarrow Online.KeyGen(PK, IK, S)$, $IT \longleftarrow Offline.Encrypt(PK)$, $CT \longleftarrow Online.Encrypt(PK, IT, M, \mathbb{A})$, $(TK_S, RK_S) \longleftarrow GenTK_{out}(PK, SK_S)$, $CT' \longleftarrow Transform_{out}(PK, CT, TK_S)$, consider two cases:
Case1: if S satisfies $\mathbb{A}$, then both the algorithms $M \longleftarrow Decrypt(PK, SK_S, CT)$ and $M \longleftarrow Decrypt_{out}(PK, CT, CT', RK_S)$ return 1.
Case2: if S does not satisfy $\mathbb{A}$, then both the algorithms $Decrypt(PK, SK_S, CT)$ and $Decrypt_{out}(PK, CT, CT', RK_S)$ output $\perp$.

In the following, we describe the security model for the aforementioned algorithms. Similar to [33], we adopt replayable CCA security. The experiment OO-VO-CPABE$_{\mathcal{A},\Pi}(\lambda, U)$ proceeds as follows:

- Setup. The challenger $\mathcal{C}$ runs $Setup$ algorithm to obtain public parameters PK and gives PK to adversary $\mathcal{A}$.
- Phase 1. $\mathcal{C}$ initializes an empty table T and an empty set D. Then $\mathcal{A}$ can repeatedly issue the following queries:
  $*$ $KeyGen$ query: The challenger $\mathcal{C}$ takes as input attributes set S. Then it runs $Online.KeyGen(PK, Offline.KeyGen(MSK), S)$ to obtain private key SK_S and sets $D = D \cup \{S\}$. Finally, $\mathcal{C}$ sends the private key SK_S to $\mathcal{A}$.
  $*$ $GenTK$ query: The challenger $\mathcal{C}$ takes as input attributes set S, then searches the entry (S, SK_S, TK_S, RK_S) in table T. If the entry exists in table T, then $\mathcal{C}$ returns TK_S to $\mathcal{A}$. Otherwise, $\mathcal{C}$ first runs $KenGen$ to obtain private key SK_S, then runs $(TK_S, RK_S) \longleftarrow GenTK_{out}(PK, SK_S)$ and stores the entry (S, SK_S, TK_S, RK_S) in table T. Finally, $\mathcal{C}$ sends TK_S to $\mathcal{A}$.
  $*$ $Decrypt$ query: The challenger $\mathcal{C}$ takes as input attributes set S and a ciphertext CT, then runs $KenGen$ to obtain private key SK_S and runs $Decrypt(PK, SK_S, CT)$. Finally, $\mathcal{C}$ sends decryption result to $\mathcal{A}$.
  $*$ $Decrypt_{out}$ query: The challenger $\mathcal{C}$ takes as input attributes set S and a pair of ciphertexts (CT, CT'), then searches the entry (S, SK_S, TK_S, RK_S) in table T. If the entry exists in T, then $\mathcal{C}$ runs $M \longleftarrow Decrypt_{out}(PK, CT, CT', RK_S)$ and returns M to $\mathcal{A}$. Otherwise, $\mathcal{C}$ returns $\perp$.
- Challenge. $\mathcal{A}$ submits two messages M_0, M_1 with equal length and an access structure $\mathbb{A}^*$ to challenger $\mathcal{C}$. For all the private key query in Phase 1, $\mathbb{A}^*$ cannot be satisfied by S, where $S \in D$. Then $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$ and sends to $\mathcal{A}$ the challenge ciphertext CT^* generated from: $Online.Encrypt(PK, Offline.Encrypt(PK), M_b, \mathbb{A}^*) \longrightarrow CT^*$.
- Phase 2. $\mathcal{A}$ continues to issue queries as in Phase 1 but with the restrictions that
  $*$ $\mathcal{A}$ cannot issue a $KeyGen$ query which attributes set S satisfies the access structure $\mathbb{A}^*$.
  $*$ $\mathcal{A}$ cannot issue a decryption query on the challenge ciphertext CT^*. Besides, if M_0 or M_1 is the result of $Decrypt$ query or $Decrypt_{out}$ query, $\mathcal{C}$ would return to $\mathcal{A}$ an error symbol $\perp$.
- Guess. The adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. The experiment returns 1 if and only if $b' = b$.

Definition 10 (OO-VO-CPABE security)
The OO-VO-CPABE scheme is replayable CCA-secure for attribute universe U if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage in this security game is negligible:

$$\left| Pr\left[\text{OO-VO-CPABE}_{\mathcal{A},\Pi}(\lambda, U) = 1\right] - \frac{1}{2} \right| \leqslant negl(\lambda).$$

CPA security. The OO-VO-CPABE scheme is CPA-secure if we remove the Decrypt oracle in both Phases 1 and 2.
Selective security. The OO-VO-CPABE scheme is $selectively$ secure if we add an Init stage before the Setup stage where the adversary commits to challenger an access structure $\mathbb{A}^*$ to attack.

DOI: 10.1002/cpe

Verifiability. The verifiability of the OO-VO-CPABE scheme is described by a security game between the challenger $\mathcal{C}$ and adversary $\mathcal{A}$. The game proceeds as follows:

- Setup. The challenger $\mathcal{C}$ runs $Setup$ algorithm to obtain (PK, MSK). Then it sends public parameters PK to the adversary $\mathcal{A}$.
- Phase 1. The challenger $\mathcal{C}$ initializes an empty table T. Then the adversary $\mathcal{A}$ adaptively issues the following queries:
  $*$ $KeyGen$ query: The challenger $\mathcal{C}$ takes as input attributes set S. Then it runs $Online.KeyGen(PK, Offline.KeyGen(MSK), S)$ to obtain private key SK_S and sends SK_S to $\mathcal{A}$.
  $*$ $GenTK$ query: The challenger $\mathcal{C}$ takes as input attributes set S, then searches the entry (S, SK_S, TK_S, RK_S) in table T. If the entry exists in table T, then $\mathcal{C}$ returns TK_S to $\mathcal{A}$. Otherwise, $\mathcal{C}$ first runs $KenGen$ to obtain private key SK_S, then runs $(TK_S, RK_S) \longleftarrow GenTK_{out}(PK, SK_S)$ and stores the entry (S, SK_S, TK_S, RK_S) in table T. Finally, $\mathcal{C}$ sends TK_S to $\mathcal{A}$.
  Without loss of generality, we consider that $\mathcal{A}$ does not issue $GenTK$ query on attributes set S if it has issued $KeyGen$ query on the same attributes set S because $\mathcal{A}$ can generate a transformation key by using the algorithm $GenTK_{out}$ and the private key SK_S.
  $*$ $Decrypt$ query: The challenger $\mathcal{C}$ takes as input attributes set S and a ciphertext CT, then runs $KenGen$ to obtain private key SK_S and runs $Decrypt(PK, SK_S, CT)$. Finally, $\mathcal{C}$ sends decryption result to $\mathcal{A}$.
  $*$ $Decrypt_{out}$ query: The challenger $\mathcal{C}$ takes as input attributes set S and a pair of ciphertexts $(CT, CT^{'})$, then searches the entry (S, SK_S, TK_S, RK_S) in table T. If the entry exists in T, then $\mathcal{C}$ runs $M \longleftarrow Decrypt_{out}(PK, CT, CT^{'}, RK_S)$ and returns M to $\mathcal{A}$. Otherwise, $\mathcal{C}$ returns $\perp$.
- Challenge. The adversary $\mathcal{A}$ submits a message M^* and an access structure $\mathbb{A}^*$ to $\mathcal{C}$. Then $\mathcal{C}$ runs $Online.Encrypt(PK, Offline.Encrypt(PK), M^*, \mathbb{A}^*) \longrightarrow CT^*$ and sends CT^* to $\mathcal{A}$.
- Phase 2. The adversary $\mathcal{A}$ continues to adaptively issue $KeyGen$ query, $GenTK$ query, $Decrypt$ query and $Decrypt_{out}$ query, as in Phase 1.
- Forgery. The adversary $\mathcal{A}$ outputs an attributes set S^* and a transformed ciphertext $CT^{*'}$. Note that $\mathcal{A}$ has issued $GenTK$ query for attributes set S^* and obtains the transformation key TK_{S^*}. The challenger $\mathcal{C}$ stores the entry $(S^*, SK_{S^*}, TK_{S^*}, RK_{S^*})$ in table T. $\mathcal{A}$ wins the game if $Verify(PK, CT^{*'}, RK_{S^*})$ returns 1 and $Decrypt^{'}_{out}(PK, CT^*, CT^{*'}, RK_{S^*}) \notin \{M^*, \perp\}$.

Definition 11 (Verifiability)
The OO-VO-CPABE scheme is verifiable if all probabilistic polynomial-time adversaries have at most a negligible advantage to win the aforementioned game.

4. OUR CONSTRUCTION OF CP-ABE SCHEME

In this section, we first present a concrete CP-AB-KEM scheme with offline/online key generation and encryption, utilizing the Waters *et al.* schemes [11, 44]. This scheme is selectively CPA-secure. Then, we modify the scheme and propose an offline/online CP-ABE scheme, which is also selectively CPA-secure. Finally we construct a third scheme that adds the functionality of verifiable outsourced decryption. It is proved both secure and verifiable.

4.1. Offline/online CP-AB-KEM scheme

Waters *et al.* [11] proposed a concrete CP-ABE scheme with offline/online encryption and gave a brief description on how to design the offline/online key generation. Here, we present a concrete offline/online CP-AB-KEM scheme, supporting offline/online key generation and encryption. We assume a bound N as the maximum number of rows in an LSSS. Notice that LSSS is an access structure that specifies an access policy. The bound can be easily removed using the 'Pooling' technique described in [11].

DOI: 10.1002/cpe

- $Setup(\lambda, U)$. Let e: $\mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_{\mathbb{T}}$, where $\mathbb{G}$ and $\mathbb{G}_{\mathbb{T}}$ are multiplicative cyclic groups of prime order $p \in \Theta(2^{\lambda})$. Choose random generators $g, h, u, v, w \in \mathbb{G}$ and pick a random exponent $\alpha \in \mathbb{Z}_p$. Set the public parameters $PK = (\mathbb{G}, \mathbb{G}_{\mathbb{T}}, p, e, g, h, u, v, w, e(g,g)^{\alpha})$, the master secret key $MSK = (PK, \alpha)$. We regard the attribute universe as a subset of elements in $\mathbb{Z}_p$.
- $Offline.KeyGen(MSK)$. Choose random element $r \in \mathbb{Z}_p$, then compute $K_0 = g^{\alpha} w^r$, $K_1 = g^r$, $K_v = v^{-r}$. Next, for $i = 1$ *to* N, choose random $r_i, x_i \in \mathbb{Z}_p$ and compute $K'_{i,1} = g^{r_i}$, $K'_{i,2} = (u^{x_i} h)^{r_i}$. Let $IK_{main} = \{K_0, K_1, K_v\}$, $IK_{att,i} = \{r_i, x_i, K'_{i,1}, K'_{i,2}\}_{i \in [1,N]}$. Finally, set the intermediate key $IK = (K_0, K_1, K_v, \{r_i, x_i, K'_{i,1}, K'_{i,2}\}_{i \in [1,N]})$.
- $Online.KeyGen(PK, IK, S)$. Assume an attribute set $S = \{A_1, A_2, \ldots, A_k\} \subseteq \mathbb{Z}_p$, where $k \leqslant N$. First, choose one main module $IK_{main} = \{K_0, K_1, K_v\}$ and k attribute modules $IK_{att,i} = \{r_i, x_i, K'_{i,1}, K'_{i,2}\}$, then compute $K_{i,1} = K'_{i,1} = g^{r_i}$, $K_{i,2} = K'_{i,2} \cdot K_v = (u^{x_i} h)^{r_i} \cdot v^{-r}$, $K_{i,3} = r_i \cdot (A_i - x_i)$. The private key is $SK_S = (S, K_0, K_1, \{K_{i,1}, K_{i,2}, K_{i,3}\}_{i \in [1,k]})$.
  Notice that the dominant cost is two multiplications in $\mathbb{Z}_p$ per attribute of S. In addition, any attribute module can be attached to any main module.
- $Offline.Encrypt(PK)$. Choose a random $s \in \mathbb{Z}_p$, then compute $key = e(g,g)^{\alpha s}$, $C_0 = g^s$. Next, for $j = 1$ *to* N, choose random $\lambda'_j, x_j, t_j \in \mathbb{Z}_p$ and compute $C_{j,1} = w^{\lambda'_j} v^{t_j}$, $C_{j,2} = (u^{x_j} h)^{-t_j}$, $C_{j,3} = g^{t_j}$. Let $IT_{main} = \{key, s, C_0\}$, $IT_{att,j} = \{\lambda'_j, x_j, t_j, C_{j,1}, C_{j,2}, C_{j,3}\}_{j \in [1,N]}$. Finally, set the intermediate ciphertext $IT = (key, s, C_0, \{\lambda'_j, x_j, t_j, C_{j,1}, C_{j,2}, C_{j,3}\}_{j \in [1,N]})$.
- $Online.Encrypt(PK, IT, \mathbb{A})$. Let the given access structure be $\mathbb{A} = (\mathcal{M}, \rho)$, where $\mathcal{M}$ is an $l \times n$ and $l \leqslant N$. Choose random $y_2, \ldots, y_n \in \mathbb{Z}_p$ and set the vector $\overrightarrow{y} = (s, y_2, \ldots, y_n)^T$ (T denotes the transpose of the matrix), then compute $\{\lambda_i = M_i \cdot \overrightarrow{y}\}_{i \in [1,l]}$. Next, for $j = 1$ *to* l, compute $C_{j,4} = \lambda_j - \lambda'_j$, $C_{j,5} = t_j \cdot (x_j - \rho(j))$. Finally, set the ciphertext $CT = (\mathbb{A}, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4}, C_{j,5}\}_{j \in [1,l]})$.
  Notice that the dominant cost is one multiplication in $\mathbb{Z}_p$ per row of $\mathcal{M}$.
- $Decrypt(PK, SK_S, CT)$. The decryption algorithm recovers the encapsulated key. It takes as input a ciphertext $CT = (\mathbb{A}, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4}, C_{j,5}\}_{j \in [1,l]})$ and a private key $SK_S = (S, K_0, K_1, \{K_{i,1}, K_{i,2}, K_{i,3}\}_{i \in [1,k]})$. If $S \notin \mathbb{A}$, the algorithm outputs an error message. Otherwise, it sets $I = i : \{\rho(i) \in S\}$ and computes constant $w_i \in \mathbb{Z}_p$ such that $\sum_{i \in I} w_i M_i = (1, 0, \ldots, 0)$. The decryption algorithm then computes

$$key = \frac{e(C_0, K_0)}{e(w^{\sum_{i \in I} C_{i,4} w_i}, K_1) \cdot \prod_{i \in I} (e(C_{i,1}, K_1) \cdot e(C_{i,2} \cdot u^{C_{i,5}}, K_{j,1}) \cdot e(C_{i,3}, u^{K_{j,3}} \cdot K_{j,2}))^{w_i}}$$

Correctness. If the attribute set S satisfies the access structure $\mathbb{A}$, we have that $\sum_{i \in I} w_i \lambda_i = s$. Therefore, key

$$= \frac{e(C_0, K_0)}{e\left(w^{\sum_{i \in I} C_{i,4} w_i}, K_1\right) \cdot \prod_{i \in I} e\left(C_{i,1}, K_1\right)^{w_i} \cdot \prod_{i \in I} e\left(C_{i,2} \cdot u^{C_{i,5}}, K_{j,1}\right)^{w_i} \cdot \prod_{i \in I} e\left(C_{i,3}, u^{K_{j,3}} \cdot K_{j,2}\right)^{w_i}}$$

$$= \frac{e\left(g^s, g^{\alpha} w^r\right)}{e\left(w^{\sum_{i \in I} \left(\lambda_i - \lambda'_i\right) w_i}, g^r\right) \cdot \prod_{i \in I} e\left(w^{\lambda'_i} v^{t_i}, g^r\right)^{w_i} \cdot \prod_{i \in I} e\left((u^{x_i} h)^{-t_i} \cdot u^{t_i (x_i - \rho(i))}, g^{r_j}\right)^{w_i}}$$

$$\cdot \frac{1}{\prod_{i \in I} e\left(g^{t_i}, u^{r_j (A_j - x_j)} \cdot (u^{x_j} h)^{r_j} \cdot v^{-r}\right)^{w_i}}$$

$$= \frac{e(g,g)^{\alpha s} \cdot e(g,w)^{rs}}{e(g,w)^{r \sum_{i \in I} \left(\lambda_i - \lambda'_i\right) w_i} \cdot \prod_{i \in I} \left(e(g,w)^{r \lambda'_i} \cdot e(g,v)^{r t_i}\right)^{w_i} \cdot \prod_{i \in I} (e(g,u)^{-t_i \rho_i r_j} \cdot e(g,h)^{-t_i r_j})^{w_i}}$$

DOI: 10.1002/cpe

$$\cdot \frac{1}{\prod_{i\in I}(e(g,u)^{t_i r_j A_j}\cdot e(g,h)^{t_i r_j}\cdot e(g,v)^{-rt_i})^{w_i}}$$
$$= \frac{e(g,g)^{\alpha s}\cdot e(g,w)^{rs}}{e(g,w)^{r\sum_{i\in I}\lambda_i w_i}}$$
$$= e(g,g)^{\alpha s}$$

Theorem 1
Suppose that Rouselakis and Waters' scheme in [44] is selectively CPA-secure, then our OO-CP-ABKEM scheme is also selectively CPA-secure.

Proof
Suppose that there exists a probabilistic polynomial-time adversary $\mathcal{A}$, who can break our scheme with non-negligible advantage ϵ, then we can build a simulator $\mathcal{B}$ who can break RW scheme in [44] with the same non-negligible advantage ϵ.
Initialization. $\mathcal{B}$ receives a challenge access structure $(\mathcal{M}^*, \rho^*)$ from $\mathcal{A}$ and sends it to the RW challenger $\mathcal{C}$. Let $\mathcal{M}^*$ be an $l \times n$ matrix.
Setup. $\mathcal{B}$ receives public parameters $PK = (\mathbb{G}, \mathbb{G}_{\mathbb{T}}, p, e, g, h, u, v, w, e(g,g)^{\alpha})$ from RW challenger $\mathcal{C}$, then sends PK to $\mathcal{A}$.
Phase 1. $\mathcal{B}$ receives private key query $S^* = (A_1, \ldots A_k)$ from $\mathcal{A}$, then sends it to the RW challenger $\mathcal{C}$ to obtain the private key $SK^*_{RW} = (S^*, K_0, K_1, \{K_{i,1}, K_{i,2}\}_{i\in[1,k]})$. Here, $K_0 = g^{\alpha}w^r$, $K_1 = g^r$, $K_{i,1} = g^{r_i}$, $K_{i,2} = (u^{x_i}h)^{r_i}v^{-r}$. Next, $\mathcal{B}$ selects random blinding values $x_1, \ldots, x_k \in \mathbb{Z}_p$ and computes $K'_{j,1} = K_{j,1} = g^{r_j}$, $K'_{j,2} = K_{j,2}\cdot u^{-x_j} = (u^{A_j}h)^{r_j}v^{-r}u^{-x_j}$, $K'_{j,3} = x_j$. Finally, $\mathcal{B}$ sends the private key $SK^*_{OO} = (S^*, K_0, K_1, \{K'_{j,1}, K'_{j,2}, K'_{j,3}\}_{i\in[1,k]})$ to $\mathcal{A}$.
Challenge. $\mathcal{B}$ chooses two random messages m_0, m_1 with equal length, then sends them to RW challenger $\mathcal{C}$. $\mathcal{C}$ flips a random coin $b \in \{0,1\}$ and encrypts message m_b. Then $\mathcal{C}$ sends $CT^*_{RW} = ((\mathcal{M}^*, \rho^*), C, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j\in[1,l]})$ to $\mathcal{B}$. Here, $C = m_b e(g,g)^{\alpha s}$, $C_0 = g^s$. For each row in $\mathcal{M}^*$, $C_{j,1} = w^{\lambda_j}v^{t_j}$, $C_{j,2} = (u^{\rho^*(j)}h)^{t_j}$, $C_{j,3} = g^{t_j}$. Next, $\mathcal{B}$ chooses random values $z_1, \ldots, z_l, z'_1, \ldots, z'_l \in \mathbb{Z}_p$ and computes $CT^*_{OO} = ((\mathcal{M}^*, \rho^*), C, C_0, \{C'_{j,1}, C'_{j,2}, C'_{j,3}, C'_{j,4}, C'_{j,5}\}_{i\in[1,l]})$. Here, $C'_{j,1} = C_{j,1}\cdot w^{-z_j} = w^{\lambda_j - z_j}v^{t_j}$, $C'_{j,2} = C_{j,2}\cdot u^{-z'_j} = (u^{\rho^*(j)}h)^{t_j}\cdot u^{-z'_j}$, $C'_{j,3} = C_{j,3} = g^{t_j}$, $C'_{j,4} = z_j$, $C'_{j,5} = z'_j$. Then $\mathcal{B}$ guesses which message was encrypted $\tau_b \in \{0,1\}$ and computes $key_{guess} = C/m_{\tau_b}$. Finally, $\mathcal{B}$ sends (key_{guess}, CT^*_{OO}) to $\mathcal{A}$.
Note that if $\tau_b = b$, it means that key_{guess} is the key encapsulated by CT^*_{OO}. If $\tau_b \neq b$, it means that key_{guess} is a random key. Thus, $\mathcal{B}$ has properly simulated the security game for $\mathcal{A}$.
Phase 2. $\mathcal{B}$ proceeds as in Phase 1.
Guess. $\mathcal{A}$ outputs a bit τ_a. If $\tau_a = 0$, it means that $\mathcal{A}$ makes a guess that key_{guess} is the key encapsulated by CT^*_{OO}. If $\tau_a = 1$, it means that $\mathcal{A}$ makes a guess that key_{guess} is a random key. When $\tau_a = 0$, $\mathcal{B}$ outputs τ_b to $\mathcal{C}$. When $\tau_a = 1$, $\mathcal{B}$ outputs $(1-\tau_b)$ to $\mathcal{C}$. Thus, if $\mathcal{A}$ has advantage ϵ to break our scheme, then $\mathcal{B}$ has the same advantage to break the RW scheme. □

4.2. Offline/online CP-ABE scheme

Based on the OO-CP-ABKEM scheme described earlier, we construct an offline/online CP-ABE (OO-CP-ABE) scheme where plaintext message is encrypted. Besides, we make some changes to the system *Setup* and *Online Encryption*, for checking the correctness of decryption. Here, we mainly describe the parts that are different from the aforementioned scheme.

- $Setup(\lambda, U)$. Choose key derivation function KDF with the output length l and choose a collision-resistant hash function $H : \{0,1\}^* \longrightarrow \mathbb{Z}_p$. Then set the public parameters $PK = (\mathbb{G}, \mathbb{G}_{\mathbb{T}}, p, e, g, h, u, v, w, e(g,g)^{\alpha}, KDF, l, H)$, the master secret key $MSK = (PK, \alpha)$.
- $Offline.KeyGen(MSK)$. It is the same as the corresponding algorithm described in the OO-CP-ABKEM scheme. It outputs the intermediate key $IK = (K_0, K_1, K_v, \{r_i, x_i, K'_{i,1}, K'_{i,2}\}_{i\in[1,N]})$.

DOI: 10.1002/cpe

- $Online.KeyGen(PK, IK, S)$. It is the same as the corresponding algorithm described in the OO-CP-ABKEM scheme. It outputs the private key $SK_S = (S, K_0, K_1, \{K_{i,1}, K_{i,2}, K_{i,3}\}_{i\in[1,k]})$.
- $Offline.Encrypt(PK)$. It is the same as the corresponding algorithm described in the OO-CP-ABKEM scheme. It outputs the intermediate ciphertext $IT = (key, s, C_0, \{\lambda^{'}_j, x_j, t_j, C_{j,1}, C_{j,2}, C_{j,3}\}_{j\in[1,N]})$.
- $Online.Encrypt(PK, IT, m, \mathbb{A})$. This algorithm additionally takes as input a message m that is to be encrypted. Same as the OO-CP-ABKEM scheme, this algorithm first outputs $C = (\mathbb{A}, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4}C_{j,5}\}_{j\in[1,l]})$. Next, it chooses random $a \in \mathbb{G}_T$, then computes $\bar{C} = ((m \parallel a) \oplus KDF(key, l))$ and $\hat{C} = u^{H(m)} \cdot v^{H(a)}$. Finally, the ciphertext is $CT = (C, \bar{C}, \hat{C})$.
- $Decrypt(PK, SK_S, CT)$. If $S \notin \mathbb{A}$, the algorithm issues an error message. Otherwise, it first computes

$$key = \frac{e(C_0, K_0)}{e(w^{\sum_{i\in I} C_{i,4} w_i}, K_1) \cdot \prod_{i\in I}(e(C_{i,1}, K_1) \cdot e(C_{i,2} \cdot u^{C_{i,5}}, K_{j,1}) \cdot e(C_{i,3}, u^{K_{j,3}} \cdot K_{j,2}))^{w_i}}$$

  Next, it computes $\bar{C} \oplus KDF(key, l) = (m \parallel a) \oplus KDF(key, l) \oplus KDF(key, l) = (m \parallel a)$. If the algorithm passes the verification $\hat{C} = u^{H(m)} \cdot v^{H(a)}$, then it outputs message m. Otherwise, it outputs the error symbol $\perp$.

Theorem 2
Suppose that the OO-CP-ABKEM scheme is selectively CPA-secure and the key derivation function KDF is secure, then the newly constructed OO-CP-ABE scheme is selectively CPA-secure.

Proof
To prove the theorem, we consider the following four games:

- $Game_0$: The challenger $\mathcal{C}$ generates the ciphertext $CT^* = (C^*, \bar{C}^*, \hat{C}^*)$ by executing the following procedures: $Encrypt(PK, \mathbb{A}) \longrightarrow (key, C^*)$, then chooses random $b \in \{0, 1\}$ and a random message a. Finally computes $\bar{C}^* = ((m_b \parallel a) \oplus KDF(key, l))$, $\hat{C}^* = u^{H(m_b)} \cdot v^{H(a)}$.
- $Game_1$: Same as $Game_0$ except that $\mathcal{C}$ generates $\bar{C}^* = ((m_b \parallel a) \oplus RK)$, where RK is a random session key.
- $Game_2$: Same as $Game_1$ except that $\mathcal{C}$ generates $\bar{C}^* = (0^{|m_b \| a|} \oplus RK)$, which $0^{|m_b \| a|}$ means a random number with the same length of $m_b \parallel a$.
- $Game_3$: Same as $Game_2$ except that $\mathcal{C}$ generates $\hat{C}^* = u^{H(0^{|m_b|})} \cdot v^{H(a)}$, which $0^{|m_b|}$ means a random number with the same length of m_b.

  We consider the computational indistinguishability between $Game_0$ and $Game_1$, $Game_1$ and $Game_2$ and $Game_2$ and $Game_3$, respectively. Because $Game_3$ contains no information about the messages submitted by the adversary $\mathcal{A}$, $\mathcal{A}$ has no advantage in $Game_3$. Thus, $\mathcal{A}$ has no advantage in real security game. The proof is similar to [13]. In order to occupy less space, we omit the detailed proof here.

□

4.3. Offline/online CP-ABE scheme with verifiable outsourced decryption

Based on the OO-CP-ABE scheme described earlier, we present an OO-VO-CPABE scheme. The $Setup$, $Offline.KeyGen$, $Online.KeyGen$, $Offline.Encrypt$, $Online.Encrypt$ and $Decrypt$ algorithms are the same as in the OO-CP-ABE scheme. The remaining algorithms are described as follows:

- $GenTK_{out}(PK, SK_S)$. It takes as input public parameters PK and a private key $SK_S = (S, K_0, K_1, \{K_{i,1}, K_{i,2}, K_{i,3}\}_{i\in[1,k]})$. First, it chooses random number $z \in \mathbb{Z}_p$, then computes $K'_0 = K_0^{1/z}$, $K'_1 = K_1^{1/z}$, $K'_{i,1} = K_{i,1}^{1/z}$, $K'_{i,2} = K_{i,2}^{1/z}$, $K'_{i,3} = K_{i,3}^{1/z}$. Finally, it outputs the transformation key $TK_S = (S, K'_0, K'_1, \{K'_{i,1}, K'_{i,2}, K'_{i,3}\}_{i\in[1,k]})$ and retrieves key $RK_S = z$.

DOI: 10.1002/cpe

- $Transform_{out}(PK, CT, TK_S)$. It takes as input public parameters PK, transformation key TK_S and ciphertext $CT = (C, \bar{C}, \hat{C})$. As described previously, $C = (\mathbb{A} = (\mathcal{M}, \rho), C_0, \{C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4}C_{j,5}\}_{j\in[1,l]})$. If $S \notin \mathbb{A}$, the algorithm issues an error message. Otherwise, it sets $I = i : \{\rho(i) \in S\}$ and computes constant $w_i \in \mathbb{Z}_p$ such that $\sum_{i\in I} w_i M_i = (1, 0, \ldots, 0)$. Then, it computes C':

$$:= \frac{e\left(C_0, K'_0\right)}{e\left(w^{\sum_{i\in I} C_{i,4}w_i}, K'_1\right)\cdot\prod_{i\in I}\left(e\left(C_{i,1}, K'_1\right)\cdot e\left(C_{i,2}\cdot u^{C_{i,5}}, K'_{j,1}\right)\cdot e\left(C_{i,3}, u^{K'_{j,3}}\cdot K'_{j,2}\right)\right)^{w_i}}$$
$$= e(g,g)^{\alpha s/z}$$

  Finally, it outputs $CT' = (C', \bar{C}' = \bar{C}, \hat{C}' = \hat{C})$.

- $Verify(PK, CT', RK_S)$. It first computes $(C')^{RK_S} = (e(g,g)^{\alpha s/z})^z = e(g,g)^{\alpha s} \to key$, then computes $\bar{C}' \oplus KDF(key, l) \to (m \parallel a)$. If $\hat{C}' = u^{H(m)} \cdot v^{H(a)}$, it outputs $b = 1$ and $st = (m, a)$. Otherwise, it outputs $b = 0$.
- $Decrypt_{out}(PK, CT, CT', RK_S)$. Recall that $CT = (C, \bar{C}, \hat{C})$, $CT' = (C', \bar{C}', \hat{C}')$. If $\bar{C} \neq \bar{C}'$ or $\hat{C} \neq \hat{C}'$, it outputs $\perp$. Otherwise, it runs algorithm $Verify(PK, CT', RK_S)$ to obtain the results. If $b = 1$, it outputs message m. If $b = 0$, it outputs $\perp$.

Theorem 3

Suppose that the OO-CP-ABE scheme is selectively CPA-secure, then the newly constructed OO-VO-CPABE scheme is also selectively CPA-secure.

Proof

Suppose that there exists a probabilistic polynomial-time adversary $\mathcal{A}$ who can break our OO-VO-CPABE scheme with non-negligible advantage ϵ, then we can build a simulator $\mathcal{B}$ who can break the OO-CP-ABE scheme in the selective CPA-secure with the same non-negligible advantage ϵ.

Initialization. $\mathcal{B}$ receives a challenge LSSS access structure $\mathbb{A}^* = (\mathcal{M}^*, \rho^*)$ from $\mathcal{A}$ and sends it to the OO-CP-ABE challenger $\mathcal{C}$. Let $\mathcal{M}^*$ be an $l \times n$ matrix.

Setup. $\mathcal{B}$ receives $PK = (\mathbb{G}, \mathbb{G}_{\mathbb{T}}, p, e, g, h, u, v, w, e(g,g)^{\alpha}, KDF, l, H)$ from challenger $\mathcal{C}$, then sends the public parameters PK to $\mathcal{A}$.

Phase 1. $\mathcal{B}$ initializes an empty table T and an empty set D. Then adversary $\mathcal{A}$ can repeatedly issue the following queries:

$*$ $KeyGen$ query: $\mathcal{B}$ takes as input attributes set S, then sends it to $\mathcal{C}$ to obtain the private key SK_S. Next, $\mathcal{B}$ sets $D = D \cup \{S\}$ and sends the private key SK_S to $\mathcal{A}$.

$*$ $GenTK$ query: $\mathcal{B}$ takes as input attributes set S, then searches the entry (S, SK_S, TK_S, RK_S) in table T. If such entry exists, it returns TK_S to $\mathcal{A}$. Otherwise, $\mathcal{B}$ executes private key query to obtain SK_S from $\mathcal{C}$, then $\mathcal{B}$ selects a random value $z \in \mathbb{Z}_p$ and computes $K'_0 = K_0^{1/z}$, $K'_1 = K_1^{1/z}$, $K'_{i,1} = K_{i,1}^{1/z}$, $K'_{i,2} = K_{i,2}^{1/z}$, $K'_{i,3} = K_{i,3}^{1/z}$. Finally, it returns $TK_S = \left(S, K'_0, K'_1, \{K'_{i,1}, K'_{i,2}, K'_{i,3}\}_{i\in[1,k]}\right)$ to $\mathcal{A}$ and stores (S, SK_S, TK_S, z) in table T.

Challenge. $\mathcal{A}$ submits two random messages m_0, m_1 with equal length and an access structure $\mathbb{A}^*$ to $\mathcal{B}$. Then $\mathcal{B}$ sends m_0, m_1, $\mathbb{A}^*$ to $\mathcal{C}$ and receives back the challenge ciphertext CT^*. Finally, $\mathcal{B}$ sends CT^* to $\mathcal{A}$.

Phase 2. Phase 1 is repeated with the restrictions that $\mathcal{A}$ cannot issue a $KeyGen$ query which attributes set S satisfies the challenge access structure $\mathbb{A}^*$.

Guess. The adversary $\mathcal{A}$ outputs a guess b' to $\mathcal{B}$. Then $\mathcal{B}$ outputs b' to $\mathcal{C}$.

If $\mathcal{A}$ can attack our OO-VO-CPABE scheme with non-negligible advantage, then $\mathcal{B}$ can attack the selectively CPA-secure OO-CP-ABE scheme with the same advantage. □

Theorem 4

The OO-VO-CPABE scheme, which has function of outsourced decryption, satisfies the property of verifiability.

DOI: 10.1002/cpe

Proof

Suppose that there exists an adversary $\mathcal{A}$ that can attack the verifiability of our OO-VO-CPABE scheme with non-negligible advantage, then we can build a simulator $\mathcal{B}$ to solve the discrete logarithm assumption in prime order group with the same non-negligible advantage.

Initialization. $\mathcal{B}$ is given $(p, \mathbb{G}, \mathbb{G}_T, e, g, g^x)$.

Setup. $\mathcal{B}$ chooses random values $x', y, z, \alpha \in \mathbb{Z}_p$, key derivation function KDF with the output length l and a hash function $H : \{0,1\}^* \longrightarrow \mathbb{Z}_p$. Then $\mathcal{B}$ sets public parameters $PK = (\mathbb{G}, \mathbb{G}_\mathbb{T}, p, e, g, h = g^{x'}, u = g^x, v = g^y, w = g^z, e(g,g)^\alpha, KDF, l, H)$, the master secret key $MSK = (\alpha)$. Finally, $\mathcal{B}$ sends PK to adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ adaptively issues these queries, including $KeyGen$ query, $GenTK$ query, $Decrypt$ query and $Decrypt_{out}$ query. Then $\mathcal{B}$ can answer these queries properly because $\mathcal{B}$ knows the master secret key.

Challenge. $\mathcal{A}$ submits a message m^* and an access structure $\mathbb{A}^*$ to $\mathcal{B}$. $\mathcal{B}$ encrypts m^* and then sends ciphertext $CT^* = (C^*, \bar{C}^*, \hat{C}^*)$ to $\mathcal{A}$. Here, $C^* = (\mathbb{A}^*, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4}C_{j,5}\}_{j\in[1,l]})$, $\bar{C}^* = ((m^* \parallel a) \oplus KDF(key, l))$, $\hat{C}^* = u^{H(m^*)} \cdot v^{H(a)}$, and $a \in \mathbb{G}_T$ is chosen by $\mathcal{B}$ randomly.

Phase 2. $\mathcal{A}$ continues to adaptively issue queries as in Phase 1.

Forgery. $\mathcal{A}$ outputs a set of attributes S^* and a transformed ciphertext $CT^{*\prime} = (C', \bar{C}', \hat{C}')$, where $\bar{C}' = \bar{C}^*$ and $\hat{C}' = \hat{C}^*$. Note that $\mathcal{A}$ has issued $GenTK$ query for attributes set S^* and obtains the transformation key TK_{S^*}. $\mathcal{B}$ owns the corresponding retrieving key RK_{S^*}.

$\mathcal{B}$ computes $(C')^{RK_{S^*}} = key'$, then computes $\bar{C}' \oplus KDF(key', l) = (m' \parallel a')$. If $\mathcal{A}$ wins the aforementioned game, then $Verify(PK, CT^{*'}, RK_{S^*})$ returns 1 and $Decrypt'_{out}(PK, CT^*, CT^{*'}, RK_{S^*}) \notin \{m^*, \perp\}$. So $\mathcal{B}$ can obtain

$$\begin{aligned} & g^{xH(m^*)+yH(a)} \\ =& u^{H(m^*)}v^{H(a)} = \hat{C}^* = \hat{C}' = u^{H(m')}v^{H(a')} \\ =& g^{xH(m')+yH(a')}, \end{aligned}$$

where $m^* \neq m'$ and m^*, a, m', a', y are known by $\mathcal{B}$. Because H is a collision-resistant hash function, it has overwhelming probability $H(m^*) \neq H(m')$. Then $\mathcal{B}$ outputs $x = \frac{y(H(a')-H(a))}{H(m^*)-H(m')}$ as the solution of the discrete logarithm problem $(p, \mathbb{G}, \mathbb{G}_T, e, g, g^x)$. □

5. PERFORMANCE EVALUATION

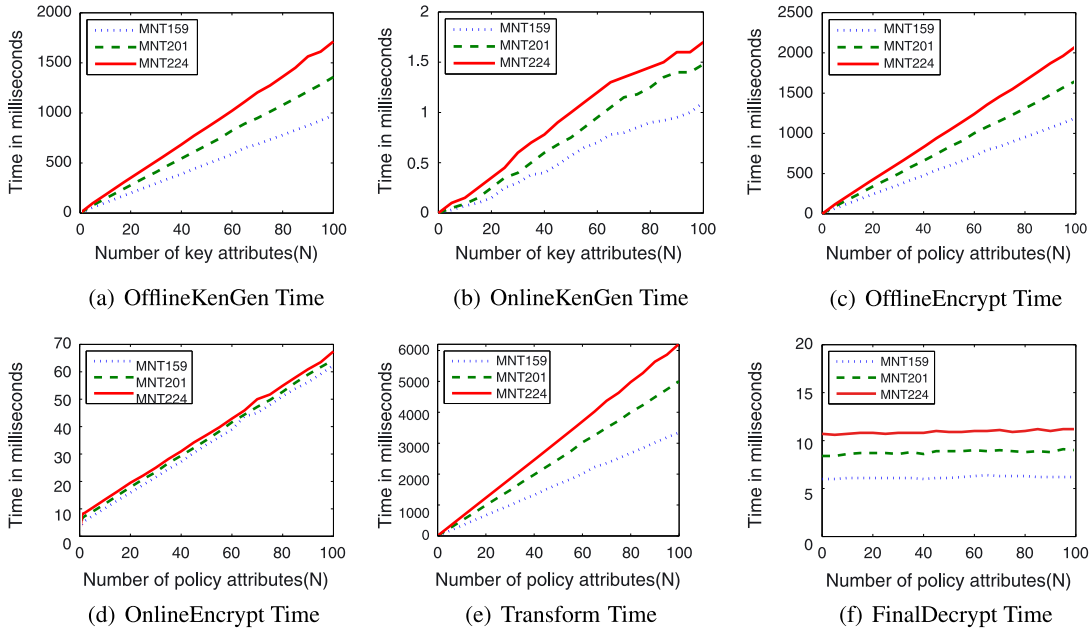
In this section, we evaluate the performance of our CP-ABE scheme, that is, the OO-VO-CPABE scheme.

We implemented our scheme on different elliptic curves provided by Charm [45] from the Stanford Pairing-based Crypto library [46]. Because all Charm routines use formally asymmetric groups [44], we need to make small changes on our scheme of symmetric setting in the implementation. Precisely, an asymmetric pairing e takes elements from $\mathbb{G}_1$ and $\mathbb{G}_2$ as inputs. Our program runs on a virtual machine platform: Vmware@Workstation 10.0.2-build-1744117, a 3.20 GHz Intel Core CPU with 2 GB of RAM running 32-bit Ubuntu 12.04. This virtual machine is installed on a computer: Dell OptiPlex 3020, made in China. The programming language is Python 3.2.

In a CP-ABE scheme, both the encryption and decryption times are affected by the complexity of ciphertext policy. In our experiment, we generate 100 different policies in the form of (A_1 and A_2 and $\ldots$ A_N) as in [10], where each A_i is an attribute and $1 \leqslant N \leqslant 100$. We repeat our experiments 30 times for each ciphertext policy, and we take the average values as our experimental results.

Figure 1 has six subfigures (a)–(f). Each subfigure gives a comparison of execution time of the proposed scheme running on different elliptic curve bilinear groups (MNT159, MNT201, MNT224) supported by Charm. In [47], Rouselakis *et al.* proposed that the approximate security levels of MNT159, MNT201 and MNT224 were 70, 90 and 100 bits, respectively. As expected, the performance results show that our scheme running on MNT224 takes the longest time as the corresponding security level is the highest.

DOI: 10.1002/cpe



(a) OfflineKenGen Time (b) OnlineKenGen Time (c) OfflineEncrypt Time

(d) OnlineEncrypt Time (e) Transform Time (f) FinalDecrypt Time

Figure 1. Performance of our OO-VO-CPABE scheme. (a) OfflineKenGen Time; (b) OnlineKenGen Time; (c) OfflineEncrypt Time; (d) OnlineEncrypt Time; (e) Transform Time; (f) FinalDecrypt Time.

Figure 1(a) and (b) shows that the offline key generation time and online key generation time are linear with key attributes. Figure 1(c) and (d) shows that the offline encryption time and online encryption time are linear with the complexity of the ciphertext's policy (N). Figure 1(e) shows that the outsourced decryption time is also linear with the complexity of the ciphertext's policy (N). As the final decryption step only requires constant amount of computation regardless of the ciphertext policy, the final decryption time depicted in Figure 1(f) is nearly constant.

In Figure 1, we also see that the intensive preprocessing computations could be finished during the offline phase, while only small computational cost is needed during the online phase. Similarly, outsourced decryption could substantially reduce the computational cost required for end devices to recover the plaintext. In our experiment, take MNT224 as an example. When the number of key attributes is 100, the offline key generation time is 1712.6 ms, while the online key generation time is only 1.7 ms. The offline encryption time is 2080.1 ms, while the online encryption time is 67.4 ms. The outsourced decryption time is 6216.6 ms, while the final decryption (the last step to recover the plaintext) time is only 11.2 ms. We conclude that the intensive computing task can be carried out during offline phase or outsourced to a third-party server. Thus, for key generation, encryption and decryption, our scheme can effectively mitigate the computational burden imposed on resource-constrained devices.

6. CONCLUSIONS

In this paper, we propose a concrete offline/online key generation and encryption, as well as verifiable outsourced decryption CP-ABE scheme. The scheme is selectively CPA-secure in the standard model. We also give the proof of verifiability on outsourced decryption. To assess the practicability of our scheme, we provide an implementation and show the result of performance measurements, which indicates that our scheme can effectively mitigate computational cost imposed on resource-constrained devices. Overall, our scheme helps to reduce the cost of ABE to make it practical in applications such as social networking.

ACKNOWLEDGEMENTS

This work is supported in part by the National High Technology Research and Development Program of China (No. 2015AA016008), the National Natural Science Foundation of China (No. 61402136), the Natural

DOI: 10.1002/cpe

Science Foundation of Guangdong Province, China (No. 2014A030313697), the International Exchange and Cooperation Foundation of Shenzhen City, China (No. GJHZ20140422173959303), Shenzhen Applied Technology Engineering Laboratory for Internet Multimedia Application of Shenzhen Development and Reform Commission (No. [2012]720), the Public Service Platform of Mobile Internet Application Security Industry of Shenzhen Development and Reform Commission (No. [2012]900), Guangdong Provincial Key Laboratory of High Performance Computing (No. [2013]82), Shenzhen Engineering Laboratory of Performance Robots at Digital Stage (Shenzhen Development and Reform Commission, Grant No. [2014]1507). Xinyi Huang's work is supported by the National Natural Science Foundation of China (61472083, U1405255), the Program for New Century Excellent Talents in Fujian University (JA14067), Distinguished Young Scholars Fund of Fujian (2016J06013), CICAEET fund and the PAPD fund.

REFERENCES

1. Sahai A, Waters B. Fuzzy identity-based encryption. *Advances in Cryptology (EUROCRYPT'05),* LNCS 2005; **3494**:457–473.
2. Bethencourt J, Sahai A, Waters B. Ciphertext-policy attribute-based encryption. *IEEE Symposium on Security and Privacy (SP'07)*, Oakland, California, USA, 2007; 321–334.
3. Cheung L, Newport C. Provably secure ciphertext policy ABE. *14th ACM Conference on Computer and Communications Security (CCS'07)*, Alexandria, Virginia, USA, 2007; 456–465.
4. Waters B. Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. *Public Key Cryptography (PKC'11),* LNCS 2011; **6571**:53–70.
5. Goyal V, Pandey O, Sahai A, Waters B. Attribute-based encryption for fine-grained access control of encrypted data. *13th ACM Conference on Computer and Communications Security (CCS'06)*, Alexandria, Virginia, USA, 2006; 89–98.
6. Ostrovsky R, Sahai A, Waters B. Attribute-based encryption with non-monotonic access structures. *14th ACM Conference on Computer and Communications Security (CCS'07)*, Alexandria, Virginia, USA, 2007; 195–203.
7. Okamoto T, Takashima K. Fully secure functional encryption with general relations from the decisional linear assumption. *Advances in Cryptology (CRYPTO'10),* LNCS 2010; **6223**:191–208.
8. Li J, Chen X, Li J, Jia C, Ma J, Lou W. Fine-grained access control system based on outsourced attribute-based encryption. *Computer Security (ESORICS'13),* LNCS 2013; **8134**:592–609.
9. Parno B, Raykova M, Vaikuntanathan V. How to delegate and verify in public: verifiable computation from attribute-based encryption. *(TCC'12),* LNCS 2012; **7194**:422–439.
10. Lai J, Deng R, Guan C, Weng J. Attribute-based encryption with verifiable outsourced decryption. *IEEE Transactions on Information Forensics and Security* 2013; **8**(8):1343–1354.
11. Hohenberger S, Waters B. Online/offline attribute-based encryption. *Public Key Cryptography (PKC'14),* LNCS 2014; **8383**:293–310.
12. Kurosawa K, Phong LT. Kurosawa–desmedt key encapsulation mechanism, revisited. *AFRICACRYPT'14,* LNCS Springer, Heidelberg, 2014; **8469**:51-68.
13. Lin S, Zhang R, Ma H, Wang M. Revisiting attribute-based encryption with verifiable outsourced decryption. *IEEE Transactions on Information Forensics and Security* 2015; **10**(10):2119–2130.
14. Lewko A, Waters B. New proof methods for attribute-based encryption: achieving full security through selective techniques. *CRYPTO'12,* LNCS 2012; **7417**:180–198.
15. Attrapadung N. Dual system encryption via doubly selective security: framework, fully secure functional encryption for regular languages, and more. *Advances in Cryptology (EUROCRYPT'14),* LNCS 2014; **8441**:557–577.
16. Chen J, Gay R, Wee H. Improved dual system ABE in prime-order groups via predicate encodings. *Advances in Cryptology (EUROCRYPT'15),* LNCS 2015; **9057**:595–624.
17. Chen J, Wee H. Fully,(almost) tightly secure IBE and dual system groups. *Advances in Cryptology (CRYPTO'13),* LNCS 2013; **8043**:435–460.
18. Lewko A, Okamoto T, Sahai A, Takashima K, Waters B. Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. *Advances in Cryptology (EUROCRYPT'10),* LNCS 2010; **6110**:72–91.
19. Hohenberger S, Waters B. Attribute-based encryption with fast decryption. *Public Key Cryptography (PKC'13),* LNCS 2013; **7778**:162–179.
20. Attrapadung N, Imai H. Attribute-based encryption supporting direct/indirect revocation modes. *Cryptography and Coding,* LNCS 2009; **5921**:278–300.
21. Chen Y, Jiang Z, Yiu SM, Liu J, Au M, Wang X. Fully secure ciphertext-policy attribute based encryption with security mediator. *ICICS'14,* LNCS 2015; **8958**:274–289.
22. Chase M, Chow SM. Improving privacy and security in multi-authority attribute-based encryption. *16th ACM Conference on Computer and Communications Security (CCS'09)*, Chicago, Illinois, USA, 2009; 121–130.
23. Lewko A, Waters B. Decentralizing attribute-based encryption. *Advances in Cryptology (EUROCRYPT'11),* LNCS 2011; **6632**:568–588.
24. Even S, Goldreich O, Micali S. On-line/off-line digital signatures. *Advances in Cryptology (CRYPTO'89),*LNCS 1990; **435**:263–275.

DOI: 10.1002/cpe

25. Shamir A, Tauman Y. Improved online/offline signature schemes. *Advances in Cryptology (CRYPTO'01),* LNCS 2001; **2139**:355–367.
26. Liu JK, Au MH, Susilo W, Zhou J. Online/offline ring signature scheme. *ICICS'09,* LNCS 2009; **5927**:80–90.
27. Liu JK, Baek J, Zhou J, Yang Y, Wong JW. Efficient online/offline identity-based signature for wireless sensor network. *International Journal of Information Security* 2010; **9**(4):287–296.
28. Guo F, Mu Y, Chen Z. Identity-based online/offline encryption. *(FC'08),* LNCS 2008; **5143**:247–261.
29. Liu JK, Zhou J. An efficient identity-based online/offline encryption scheme. *ACNS'09,* LNCS 2009; **5536**:156–167.
30. Sun D, Mu Y, Susilo W. A generic construction of identity-based online/offline signcryption. *ISPA'08*, Sydney, NSW, Australia, 2008; 707–712.
31. Liu JK, Baek J, Zhou J. Online/offline identity-based signcryption revisited. *Inscrypt'10,* LNCS 2010; **6584**:36–51.
32. Chow SS, Liu JK, Zhou J. Identity-based online/offline key encapsulation and encryption. *ASIACCS'11*, Hong Kong, China, 2011; 52–60.
33. Green M, Hohenberger S, Waters B. Outsourcing the decryption of ABE ciphertexts. *Proceedings of the 20th USENIX Conference on Security (SEC'11)*, San Francisco, CA, USA, 2011; 34.
34. Ateniese G, Fu K, Green M, Hohenberger S. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on information and system security (TISSEC)* 2006; **9**:1–30.
35. Blaze M, Bleumer G, Strauss M. Divertible protocols cryptography and atomic proxy cryptography. *Advances in Cryptology (EUROCRYPT'98),* LNCS 1998; **1403**:127–144.
36. Tsang PP, Chow SS, Smith SW. Batch pairing delegation. *Advances in Information and Computer Security,* LNCS 2007; **4752**:74–90.
37. Chevallier-Mames B, Coron JS, McCullagh N, Naccache D, Scott M. Secure delegation of elliptic-curve pairing. *Smart Card Research and Advanced Application,* LNCS 2010; **6035**:24–35.
38. Chung KM, Kalai Y, Vadhan S. Improved delegation of computation using fully homomorphic encryption. *Advances in Cryptology,* LNCS 2010; **6223**:483–501.
39. Gennaro R, Gentry C, Parno B. Non-interactive verifiable computing: outsourcing computation to untrusted workers. *Advances in Cryptology,* LNCS 2010; **6223**:465–482.
40. Gentry C, Halevi S. Implementing Gentry's fully-homomorphic encryption scheme. *Advances in Cryptology (EUROCRYPT'11),* LNCS 2011; **6035**:129–148.
41. Li J, Jia C, Li J, Chen X. Outsourcing encryption of attribute-based encryption with MapReduce. *ICICS'12,* LNCS 2012; **7618**:191–201.
42. Li J, Huang X, Li J, Chen X, Xiang Y. Securely outsourcing attribute-based encryption with check ability. IEEE Transactions on Parallel and Distributed Systems. *LNCS* 2013; **25**(8):2201–2210.
43. Beimel A. Secure schemes for secret sharing and key distribution. *Ph.D. dissertation, Israel Inst. of Technology*, Technion City, Haifa, Israel, 1996.
44. Rouselakis Y, Waters B. Practical constructions and new proof methods for large universe attribute-based encryption. *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security (CCS'13)*, Berlin, Germany, 2013; 463–474.
45. Charm. (Available from: http://www.charm-crypto.com) [Accessed on 5 Dec 2015].
46. Lynn B. *The Stanford pairing based crypto library*. (Available from: http://crypto.stanford.edu/pbc) [Accessed on 20 Nov 2015].
47. Rouselakis Y, Waters B. Efficient statically-secure large-universe multi-authority attribute-based encryption. *IACR Cryptology ePrint Archive* 2015; **2015**:16.

DOI: 10.1002/cpe