Secure Outsourced Attribute-based Encryption

Jin Li, Jingwei Li, Xiaofeng Chen, Chunfu Jia and Duncan S. Wong

***Abstract*—Attribute-Based Encryption (ABE) is a promising cryptographic primitive which significantly enhances the versatility of access control mechanisms. Due to the high expressiveness of ABE policies, the computational complexities of ABE key-issuing (by Attribute Authorities (AAs)) and decryption (by eligible users) are getting prohibitively high. Despite that the existing Outsourced ABE solutions are able to offload some intensive computing tasks to a third party, for example, a cloud, so to relieve the local burden of eligible users during decryption, the high computational complexity of the key-issuing at the AAs has yet to be addressed, while an ABE system will continue to grow with more users being included, and with the user revocation being considered in practice which will trigger more key (re-)issuing.**

Aiming at tackling the challenges above, for the first time, we propose a Secure Outsourced ABE system, which not only supports secure outsourced decryption, but also provides secure outsourced key-issuing. Unlike the current outsourced ABE systems, our new method offloads all access policy and attribute related operations in the key-issuing process or decryption to a Key Generation Service Provider (KGSP) and a Decryption Service Provider (DSP), respectively, leaving only a constant number of simple operations for the AAs and eligible users to perform locally. Furthermore, we show that both outsourcing processes (to KGSP and to DSP) are secure, namely, the KGSP and the DSP would not be able to recover the keys or decrypt the ciphertexts, respectively.

In addition, we consider the scenario that a KGSP or DSP may be dishonest and could maliciously generate some incorrect returning values rather than following the outsourced operations. Therefore, in this paper, we also propose another ABE construction which allows the AAs and eligible users to check the correctness of outsourced operations in an efficient way. The security of the construction is analyzed under a recently formalized model called Refereed Delegation of Computation (RDoC).

***Index Terms*—Attribute-based encryption, access control, outsourcing computation, cloud computing, checkability**

I. Introduction

As a novel public key primitive, Attribute-Based Encryption (ABE) [1] has attracted much attention in the research community. In ABE system, users' private keys and ciphertexts are labeled with sets of descriptive attributes and access policies respectively, and a particular key can decrypt a particular ciphertext only if associated attributes and policy are matched. Until now, there are two kinds of ABE having been proposed: Key-Policy Attribute-Based Encryption (KP-ABE) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE). In KP-ABE, the access policy is assigned in private key, whereas, in CP-ABE, it is specified in ciphertext.

Recently, as the development of cloud computing, users' concerns about data security are the main obstacles that impedes cloud computing from widely adopted. These concerns are originated from the fact that sensitive data resides in public cloud, which is maintained and operated by untrusted Cloud Service Provider (CSP). ABE provides a secure way that allows data owner to share outsourced data on untrusted storage server instead of trusted server with specified group of users. This advantage makes the methodology appealing in cloud storage that requires secure access control for a large number of users belonging to different organizations.

Nevertheless, one of the main efficiency drawbacks of ABE is that the computational cost during decryption phase grows with the complexity of the access formula. Thus, before widely deployed, there is an increasing need to improve the efficiency of ABE. To address this problem, outsourced ABE, which provides a way to outsource intensive computing task during decryption to CSP without revealing data or private keys, was introduced [2][3]. It has a wide range of applications. For example, in the mobile cloud computing consisting of mobile devices or sensors as information collection nodes, user terminal (e.g. mobile device) has limited computation ability to independently finish basic encryption or decryption to protect sensitive data residing in public cloud. Outsourced ABE allows user to perform heavy decryption through "borrowing" the computation resources from CSP. Therefore, with this paradigm, the computation/storage intensive tasks can be performed even by resource-constrained users.

Beyond the heavy decryption outsourced, we observe that the attribute authority has to deal with a lot of heavy computation in a scalable system. More precisely, the attribute authority has to issue private keys to all users, but yet generation of private key typically requires large modular exponentiation computation, which grows linearly with the complexity of the predicate formula. When a large number of users call for their private keys, it may overload attribute authority. Moreover, key management mechanism key revocation in particular, is necessary in a secure and scalable ABE system. In most of existing ABE schemes, the revocation of any single private key requires key-update at attribute authority for the remaining unrevoked keys which

Jin Li is with the School of Computer Science, Guangzhou University, China, e-mail: lijin@gzhu.edu.cn.

Jingwei Li and Chunfu Jia are with the College of Information Technical Science, Nankai University, China, e-mail: lijw@mail.nankai.edu.cn, cfjia@nankai.edu.cn.

Xiaofeng Chen is with the State Key Laboratory of Integrated Service Networks (ISN), Xidian University, Xi'an, China, e-mail: xfchen@xidian.edu.cn.

Duncan S. Wong is with the Department of Computer Science, City University of Hong Kong, China, e-mail: duncan@cityu.edu.hk.

share common attributes with the one to be revoked. All of these heavy tasks centralized at authority side would make it become the efficiency bottleneck in the access control system.

A. Contribution

Aiming at eliminating the most overhead computation at both attribute authority and user sides, we propose an outsourced ABE scheme not only supporting outsourced decryption but also enabling delegating key generation. In this construction, we introduce a trival policy controlled by a default attribute and use an AND gate connecting the trival policy and user's policy. During key-issuing, attribute authority can outsource computation through delegating the task of generating partial private key for user's policy to a Key Generation Service Provider (KGSP) to reduce local overhead. Moreover, the outsourced decryption is realized by utilizing the idea of key blinding. More precisely, user can send the blinded private key to a Decryption Service Provider (DSP) to perform partial decryption and do the complete decryption at local. Following our technique, constant efficiency is achieved at both attribute authority and user sides.

In addition, we observe that when experiencing commercial cloud computing services, the CSPs may be selfish in order to save its computation or bandwidth, which may cause results returned incorrectly. In order to deal with this problem, we consider to realize checkability on results returned from both KGSP and DSP, and provide a security and functionality enhanced construction, which is provable secure under the recent formulized Refereed Delegation of Computation (RDoC) model. Out technique is to make a secret sharing on the outsourcing key for KGSP and let k parallel KGSPs utilize their individual share to generate partial private keys. After that an additional key combination phase is performed at authority side to avoid malicious collaboration between at most $k-1$ KGSPs and users. Moreover, we use the idea of "ringer" [4] and appending redundancy to fight against the dishonest actions of KGSPs and DSP. As far as we know, this is the first time considering the checkability of outsourced ABE.

B. Related Work

The notion of ABE, which was introduced as fuzzy identity-based encryption in [1], was firstly dealt with by Goyal et al. [5]. Two different and complementary notions of ABE were defined in [5]: KP-ABE and CP-ABE. A construction of KP-ABE was provided in the same paper [5], while the first CP-APE construction supporting tree-based structure in generic group model is presented by Bethencourt et al. [6]. Accordingly, several constructions supporting for any kinds of access structures were provided [7][8][9]. Concerning revocation of ABE, a delegatable revocation is proposed in [10] to achieve scalable and fine-grained access control.

To reduce the load at local, it always desires to deliver expensive computational tasks outside. Actually, the problem that how to securely outsource different kinds of expensive computations has drew considerable attention from theoretical computer science community. Atallah et al. [11] presented a framework for secure outsourcing of scientific computations such as matrix multiplication and quadrature. Nevertheless, the solution used the disguise technique and thus leaded to leakage of private information. Atallah and Li [12] investigated the problem of computing the edit distance between two sequences and presented an efficient protocol to securely outsource sequence comparison with two servers. Furthermore, Benjamin and Atallah [13] addressed the problem of secure outsourcing for widely applicable linear algebraic computations. Nevertheless, the proposed protocols required the expensive operations of homomorphic encryption. Atallah and Frikken [14] further studied this problem and gave improved protocols based on the so-called weak secret hiding assumption. Recently, Wang et al. [15] presented efficient mechanisms for secure outsourcing of linear programming computation.

We note that though several schemes have been introduced to securely outsource kinds of expensive computations, they are not suitable for reliving ABE computational overhead of exponentiation at user side. To achieve this goal, the traditional approach is to utilize server-aided techniques [16][17][18]. However, previous work are oriented to accelerating the speed of exponentiation using untrusted servers. Directly utilizing these techniques in ABE will not work efficiently. Another approach might be to leverage recent general outsourcing technique or delegating computation [19][20][21][22][23] based on fully homomorphic encryption or interactive proof system. However, Gentry [23] has shown that even for weak security parameters on "bootstrapping" operation of the homomorphic encryption, it would take at least 30 seconds on a high performance machine. Therefore, even if the privacy of the input and output can be preserved by utilizing these general techniques, the computational overhead is still huge and impractical.

Another two related work similar to us are [3] and [2]. In [2], a novel paradigm for outsourcing the decryption of ABE is provided while in [3] the authors presented the Privacy Preserving CP-ABE (PP-CP-ABE) scheme which allows to securely outsource both decryption and encryption to third party service providers. Compared with our work, the two lack of the consideration on the eliminating the overhead computation at attribute authority. Additionally, we consider a security and functionality enhanced construction enalbing checkability on returned results from CSPs.

C. Organization

This paper is organized as follows. In Section II we describe some preliminaries. In Section III, we present the system model and security definition. The proposed construction and its security and efficiency analysis are presented in Section IV. In Section V, we consider a both security and functionality enhanced construction under RDoC model. Finally, we draw conclusion in Section VI.

Acronym	Description
AA	attribute authority
KGSP	key generation service provider
DSP	decryption service provider
SSP	storage service provider

TABLE I
NOTATIONS USED IN THIS PAPER

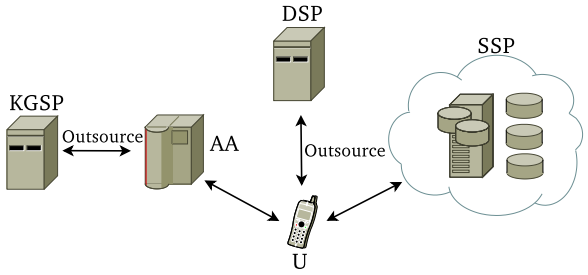


Fig. 1. System Model for Outsourced ABE Scheme

II. PRELIMINARY

In this section, we define the notations used in this paper and review some cryptographic background.

A. Notations

The notations used in this paper are listed in TABLE I.

B. Cryptographic Background

In this paper, we use the bilinear pairings on elliptic curves. We now give a brief review on the property of pairing and the candidate hard problem that will be used.

Definition 1 (Bilinear Map): Let $\mathbb{G}, \mathbb{G}_T$ be cyclic groups of prime order q, writing the group action multiplicatively. g is a generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a map with the following properties:

- *Bilinearity*: $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ for all $g_1, g_2 \in \mathbb{G}$, and $a, b \in_R \mathbb{Z}_q$;
- *Non-degeneracy*: There exists $g_1, g_2 \in \mathbb{G}$ such that $e(g_1, g_2) \neq 1$, in other words, the map does not send all pairs in $\mathbb{G} \times \mathbb{G}$ to the identity in $\mathbb{G}_T$;
- *Computability*: There is an efficient algorithm to compute $e(g_1, g_2)$ for all $g_1, g_2 \in \mathbb{G}$.

Definition 2 (DBDH Problem): The Decision Bilinear Diffie-Hellman (DBDH) problem is that, given g, g^x, g^y, $g^z \in \mathbb{G}$ for unknown values $x, y, z \in_R \mathbb{Z}_q$, and $T \in_R \mathbb{G}_T$, to decide if $T = e(g, g)^{xyz}$.

We say that the (t, ϵ)-DBDH assumption holds in $\mathbb{G}$ if no t-time algorithm has probability at least $\frac{1}{2} + \epsilon$ in solving the DBDH problem for non-negligible ϵ.

III. SYSTEM MODEL AND SECURITY DEFINITION

A. System Model

We present the system model for outsourced ABE scheme in Fig. 1. Compared with the model for typical ABE, a KGSP and a DSP are additionally involved.

- KGSP is to perform aided key-issuing computation to relieve AA load in a scale system when a large number of users make requests on private key generation and key-update.
- DSP is to finish delegated expensive operations to overcome the disadvantage that the decryption phase in typical ABE requires a large number of overload operations at U.

Following the custom in [2], we denote $(I_{\text{enc}}, I_{\text{key}})$ as the input to encryption and key generation. In CP-ABE scheme, $(I_{\text{enc}}, I_{\text{key}}) = (\mathbb{A}, \omega)$ while that is $(\omega, \mathbb{A})$ in KP-ABE, where ω and $\mathbb{A}$ are attribute set and access structure, respectively. Then, based on the proposed system model, we provide algorithm definitions as follows.

- $\mathsf{Setup}(\lambda)$: The setup algorithm takes as input – a security parameter λ. It outputs a public key PK and a master key MK.
- $\mathsf{KeyGen}_{\text{init}}(I_{\text{key}}, MK)$: For each user's private key request, the initialization algorithm for delegated key generation takes as input – an access policy (or attribute set) I_{key} and the master key MK. It outputs the key pair $(OK_{\text{KGSP}}, OK_{\text{AA}})$.
- $\mathsf{KeyGen}_{\text{out}}(I_{\text{key}}, OK_{\text{KGSP}})$: The delegated key generation algorithm takes as input – the access structure (or attribute set) I_{key} and the key OK_{KGSP} for KGSP. It outputs a partial transformation key TK_{KGSP}.
- $\mathsf{KeyGen}_{\text{in}}(I_{\text{key}}, OK_{\text{AA}})$: The inside key generation algorithm takes as input – the access structure (or attribute set) I_{key} and the key OK_{AA} for attribute authority. It outputs another partial transformation key TK_{AA}.
- $\mathsf{KeyBlind}(TK)$: The transformation key blinding algorithm takes as input – the transformation key $TK = (TK_{\text{KGSP}}, TK_{\text{AA}})$. It outputs a private key SK and a blinded transformation key $\widetilde{TK}$.
- $\mathsf{Encrypt}(M, I_{\text{enc}})$: The encryption algorithm takes as input – a message M and an attribute set (or access structure) I_{enc} to be encrypted with. It outputs the ciphertext CT.
- $\mathsf{Decrypt}_{\text{out}}(CT, \widetilde{TK})$: The delegated decryption algorithm takes as input – a ciphertext CT which was assumed to be encrypted under the attribute set (or access structure) I_{enc} and the blinded transformation key $\widetilde{TK}$ for access structure (or attribute set) I_{key}. It outputs the partially decrypted ciphertext CT_{part} if $\gamma(I_{\text{key}}, I_{\text{enc}}) = 1$, otherwise outputs $\perp$, where $\gamma(\cdot, \cdot)$ is a predicate predefined.
- $\mathsf{Decrypt}(CT_{\text{part}}, SK)$: The decryption algorithm takes as input – the partially decrypted ciphertext CT_{part} and the private key SK. It outputs the original message M.

B. Security Definition

In this work, we assume that all the entities except AA are "honest-but-curious". More precisely, they will follow our proposed protocol but try to find out as much private information as possible based on their possessions. The

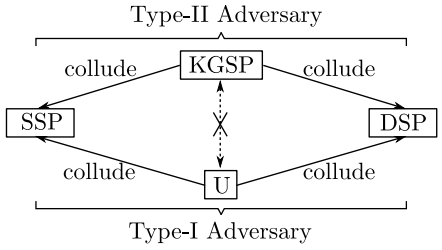


Fig. 2. Adversary Model for Outsourced ABE Scheme

adversary model described in Fig. 2 is considered. More precisely, since KGSP and U respectively owns the knowledge of OK_{KGSP} for KGSP and user's private key, they are considered as active attackers which are allowed to collude with DSP and SSP to launch harmful attack separately. Following this consideration, two types of adversaries are categorized.

- Type-I adversary defined as a group of curious users colluding with SSP and DSP, is able to potentially access private keys for all the corrupted users, all the ciphertext stored at SSP, all the blinded transformation keys stored at DSP, etc, and aims to decrypt ciphertext intended for users not in the group.
- Type-II adversary defined as KGSP colluding with SSP and DSP, is able to potentially access all the keys for KGSP, all the ciphertexts stored at SSP, all the blinded transformation keys stored at DSP, etc, and aims to decrypt any ciphertext.

Having this intuition, we follow the Replayable Chosen Ciphertext Attack (RCCA) security in [24][2] to define RCCA security for our setting in Fig. 3. We point out that the security game for type-I adversary is similar to that in [2] which shares the same delegated decryption paradigm with ours.

Denote $\mathcal{A}_i$ as type-i adversary for $i = \mathrm{I}, \mathrm{II}$. Their advantages in attacking the outsourced ABE scheme $\mathcal{E}$ is measured by the probability $Adv^{\mathrm{RCCA}}_{\mathcal{E},\mathcal{A}_i}(\lambda) = \left|\mathsf{Pr}[b_i = b_i'] - \frac{1}{2}\right|$ respectively.

Definition 3 (RCCA Security): An outsourced CP-ABE or KP-ABE scheme with delegated key generation and decryption is secure against replayable chosen-ciphertext attack if all polynomial time adversaries have at most a negligible advantage in the RCCA security game for both type-I and type-II adversaries.

Finally, beyond the RCCA security, we also specify that, i) An ABE scheme with delegated key generation and decryption is CPA-secure (or secure against chosen-plaintext attack) if no polynomial time adversary has non-negiligible advantage in modified games for type-I and type-II adversaries respectively, in which the $\mathcal{O}_M(\cdot)$ in both phase 1 and phase 2 is removed; ii) An ABE scheme with delegated key generation and decryption is secure in selective model if no polynomial time adversary has non-negiligible advantage in modified games for type-I and type-II adversaries respectively, in which the I^*_{enc} submission is advanced to an additional init stage before setup.

IV. Proposed Construction

A. Access Structure

Definition 4 (Access Structure): Let $\{P_1, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ *and* $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (or monotone access structure) is a collection (or monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$. The sets in $\mathbb{A}$ are called authorized sets.

Furthermore, we could define the predicate $\gamma(\cdot, \cdot)$ as follows.

$$\gamma(\omega, \mathbb{A}) = \begin{cases} 1 \text{ if } \omega \in \mathbb{A} \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

In this paper, the role of the party is taken by the attributes. Thus, the access structure $\mathbb{A}$ will contain the authorized sets of attributes. Specifically, our construction supports for access structure described as $\mathbb{A} = \{\omega \subseteq \mathcal{U} : |\omega \cap \omega^*| \geq d\}$ where $\mathcal{U}$ is the attribute universe, ω and ω^* are attribute sets and d is a predefined threshold value.

For simplicity, we will take user's attribute set to input to key generation instead of his access structure which is different from our definition in section III.A. We note that such substitution is trival since user is easy to compute his access structure with the individual attribute set. Furthermore, we deliver the decision for access control to $\gamma(\cdot, \cdot)$ and redefine such predicate as follows.

$$\gamma_d(\omega, \omega^*) = \begin{cases} 1 \text{ if } |\omega \cap \omega^*| \geq d \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

B. Intuition for Proposed Construction

The challenge for constructing outsourced ABE scheme is the realization of delegated key generation and decryption.

- To outsource private key generation, we utilize a hybrid key policy $\mathsf{Policy} = \mathsf{Policy}_{\mathrm{KGSP}} \wedge \mathsf{Policy}_{\mathrm{AA}}$ in proposed construction, where $\wedge$ is an AND gate connecting two sub-policies $\mathsf{Policy}_{\mathrm{KGSP}}$ and $\mathsf{Policy}_{\mathrm{AA}}$. $\mathsf{Policy}_{\mathrm{KGSP}}$ is for the request attribute set which will be performed at KGSP while $\mathsf{Policy}_{\mathrm{AA}}$ is a trival policy controlled by AA. The reason that we say it is trival is that a single default attribute θ is appended with each request attribute set, which has no effect on the global access control policy. Using this trick, we are allowed to randomly generate an outsourcing key (which is OK_{KGSP} in our construction) to delegate partial key generation operation to KGSP without master or private key leakage.
- To outsource decryption, we make use of the idea in [3] by choosing a random "blinding factor" (which is t in our construction) to produce blinded transformation key which is able to be sent to DSP to perform decryption partially instead of private key itself. This skill allows us to delegate partial decryption operation to DSP without private key or original message leakage.

RCCA Security Game for Type-I Adversary

Setup. Challenger runs $\mathsf{Setup}(\lambda)$ and gives PK to adversary.

Phase 1. Challenger initializes an integer $j = 0$, an empty table T and an empty set D to provide adversary with the oracles below.

- $\mathcal{O}_{\widetilde{TK}}(I_{\rm key})$: The challenger sets $j = j + 1$ and runs key generation (including key blinding) completely for $I_{\rm key}$ to obtain $SK, TK, OK_{\rm KGSP}$ and $\widetilde{TK}$. After storing the entry $(j, I_{\rm key}, SK, OK_{\rm KGSP})$ in T, return $\widetilde{TK}$.
- $\mathcal{O}_{SK}(i)$: The challenger checks whether the entry $(i, I_{\rm key}, SK, \cdot)$ exists in T. If so, set $D = D \cup \{I_{\rm key}\}$ and return SK, otherwise return $\perp$.
- $\mathcal{O}_M(i, CT)$: Suppose the ciphertext CT is encrypted under $I_{\rm enc}$. The challenger checks whether $(i, I_{\rm key}, SK, \cdot)$ exists in T and $\gamma(I_{\rm key}, I_{\rm enc}) = 1$. If so, perform decryption completely on CT and return original message M; otherwise return $\perp$.

Challenge. Adversary submits two messages M_0 and M_1 as well as $I^*_{\rm enc}$ satisfying $\gamma(I_{\rm key}, I^*_{\rm enc}) \neq 1$ for all $I_{\rm key} \in D$. Challenger picks $b_{\rm I} \in_R \{0, 1\}$, encrypts $M_{b_{\rm I}}$ under $I^*_{\rm enc}$ and returns the resulting ciphertext CT^*.

Phase 2. Phase 1 is repeated with the restrictions: i) Adversary cannot issue query $\mathcal{O}_{SK}(i)$ resulting in a value $I_{\rm key}$ with $\gamma(I_{\rm key}, I^*_{\rm enc}) = 1$. ii) Adversary cannot issue query $\mathcal{O}_M(\cdot)$ resulting M_0 or M_1.

Guess. Adversary outputs a guess $b'_{\rm I}$ of $b_{\rm I}$.

RCCA Security Game for Type-II Adversary

Setup. Challenger runs $\mathsf{Setup}(\lambda)$ and gives PK to adversary.

Phase 1. Challenger initializes an integer $j = 0$, an empty table T and an empty set D to provide adversary the oracles below.

- $\mathcal{O}_{\widetilde{TK}}(I_{\rm key})$: It is identical to $\mathcal{O}_{\widetilde{TK}}(I_{\rm key})$ for type-I adversary.
- $\mathcal{O}_{OK_{\rm KGSP}}(i)$: The challenger checks whether the entry $(i, I_{\rm key}, SK, OK_{\rm KGSP})$ exists in T. If so return $(OK_{\rm KGSP}, TK)$, otherwise return $\perp$.
- $\mathcal{O}_M(i, CT)$: It is identical to $\mathcal{O}_M(i, CT)$ for type-I adversary.

Challenge. Adversary submits two messages M_0 and M_1 as well as $I^*_{\rm enc}$. Challenger picks $b_{\rm II} \in_R \{0, 1\}$, encrypts $M_{b_{\rm II}}$ under $I^*_{\rm enc}$ and returns the resulting ciphertext CT^*.

Phase 2. Phase 1 is repeated with the restriction that adversary cannot issue query to $\mathcal{O}_M(\cdot)$ resulting M_0 or M_1.

Guess. Adversary outputs a guess $b'_{\rm II}$ of $b_{\rm II}$.

Fig. 3. Security Games for Type-I and Type-II Adversaries

C. Construction

Before providing our construction, we define the Lagrange coefficient $\Delta_{i,S}$ for $i \in \mathbb{Z}_q$ and a set S of elements in $\mathbb{Z}_q$: $\Delta_{i,S} = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$. Our scheme is based on ABE in [1] which shares the same access formula. The message space for our construction is $\mathbb{G}_T$. Actually, using a map-to-point function, we can extend it to support for message space consisting of $\{0, 1\}^*$. The construction in detail is shown as follows.

- $\mathsf{Setup}(\lambda)$: First, define the attributes in universe $\mathcal{U}$ as elements in $\mathbb{Z}_q$. For simplicity, let $n = |\mathcal{U}|$ and we can take the first n elements in $\mathbb{Z}_q$ (i.e. $1, 2, \ldots, n \bmod q$) to be the universe. Next, select a generator $g \in_R \mathbb{G}$ and an integer $x \in_R \mathbb{Z}_q$, and set $g_1 = g^x$. Then, pick elements $g_2, h, h_1, \ldots, h_n \in_R \mathbb{G}$. Finally, output the public key $PK = (g, g_1, g_2, h, h_1, \ldots, h_n)$ and the master key $MK = x$.
- $\mathsf{KeyGen}_{\rm init}(\omega, MK)$: For each user's private key request on ω, select $x_1 \in_R \mathbb{Z}_q$ and set $x_2 = x - x_1 \bmod q$. Finally output $OK_{\rm KGSP} = x_1$ as the outsourcing key for KGSP and $OK_{\rm AA} = x_2$ for attribute authority itself.
- $\mathsf{KeyGen}_{\rm out}(\omega, OK_{\rm KGSP})$: Randomly select a $d-1$ degree polynomial $q(\cdot)$ such that $q(0) = x_1$. Then, for each $i \in \omega$, choose $r_i \in_R \mathbb{Z}_q$, and compute $d_{i0} = g_2^{q(i)} \cdot (g_1 h_i)^{r_i}$ and $d_{i1} = g^{r_i}$. Finally, output $TK_{\rm KGSP} = (\{d_{i0}, d_{i1}\}_{i \in \omega})$.
- $\mathsf{KeyGen}_{\rm in}(\omega, OK_{\rm AA})$: Select $r_\theta \in_R \mathbb{Z}_q$ and compute $d_{\theta 0} = g_2^{x_2} \cdot (g_1 h)^{r_\theta}$ and $d_{\theta 1} = g^{r_\theta}$. Finally, output $TK_{\rm AA} = (d_{\theta 0}, d_{\theta 1})$.
- $\mathsf{KeyBlind}(TK = (TK_{\rm KGSP}, TK_{\rm AA}))$: Select $t \in_R \mathbb{Z}_q$, and compute $\widetilde{TK} = (\{d^t_{i0}, d^t_{i1}\}_{i \in \omega \cup \{\theta\}})$. Finally, output $SK = (t, TK)$ and $\widetilde{TK}$.
- $\mathsf{Encrypt}(M, \omega')$: Firstly, select a random number $s \in_R \mathbb{Z}_q$. Then, compute $C_0 = M \cdot e(g_1, g_2)^s$, $C_1 = g^s$, $E_\theta = (g_1 h)^s$ and $E_i = (g_1 h_i)^s$ for $i \in \omega'$. Finally, publish the ciphertext as $CT = (\omega' \cup \{\theta\}, C_0, C_1, \{E_i\}_{i \in \omega' \cup \{\theta\}})$.
- $\mathsf{Decrypt}_{\rm out}(CT, \widetilde{TK})$: Suppose that a ciphertext CT is encrypted under an attribute set ω' and we have a blinded transformation key $\widetilde{TK}$ for attribute set ω, which satisfies the restriction that $\gamma_d(\omega, \omega') = 1$. Then, outsourced decryption proceeds as follows. Firstly, an arbitrary d-element subset set $S \subseteq \omega \cap \omega'$ is selected. Then, the partially decrypted ciphertext is computed as follows.

$$
\begin{aligned}
CT_{\text{part}} &= \frac{e(C_1, d_{\theta 0}^t)\prod_{i\in S} e(C_1, d_{i0}^t)^{\Delta_{i,S}(0)}}{e(d_{\theta 1}^t, E_\theta)\prod_{i\in S} e(d_{i1}^t, E_i)^{\Delta_{i,S}(0)}} \\
&= e(g,g_2)^{stx_2} e(g,g_2)^{st\sum_{i\in S} q(i)\Delta_{i,S}(0)} \\
&= e(g,g_2)^{stx_2} e(g,g_2)^{stx_1} \\
&= e(g_1,g_2)^{st} \qquad (3)
\end{aligned}
$$

- $\mathsf{Decrypt}(CT_{\text{part}}, SK)$: Completely decrypt the ciphertext as follows.

$$
\begin{aligned}
\frac{C_0}{(CT_{\text{part}})^{\frac{1}{t}}} &= \frac{M\cdot e(g_1,g_2)^s}{[e(g_1,g_2)^{st}]^{\frac{1}{t}}} \\
&= \frac{M\cdot e(g_1,g_2)^s}{e(g_1,g_2)^s} = M \qquad (4)
\end{aligned}
$$

D. Efficiency Analysis

We compare our scheme with the original ABE [1] and the state-of-the-art [2][3] in TABLE II. We use EXP to denote a multi-based exponentiation operation in $\mathbb{G}$ and P the pairing operation. We assume one multi-based exponentiation multiplies up to 2 single-based exponentiations and takes roughly the same time as single-based exponentiations. ω and d denotes the attribute set and threshold value respectively.

To the best of our knowledge, the outsourced key generation in ABE has not been considered before and our scheme is the first construction achieving this property. Following our terminology, the number of exponentiations in the group $\mathbb{G}$ for AA is reduced to two, while in other ABE schemes [1][2][3], it is linear with the number of attributes in the request set (i.e. $2|\omega|$). Actually in our construction, the exponentiation computation is delivered to KGSP and requester. More precisely, after obtaining the transformation key from AA, the requester must spend $|\omega|+1$ exponentiations on generating private key and blinded transformation key. We think it is reasonable that AA delivers some computations back to users because such modification would reduce AA load and make it respond to a large number of users' requests in time.

In decryption, a trick similar to [2][3] is used in our scheme and the three schemes achieve the identical efficiency: all the pairing operations are delivered to DSP and the computational cost of decryption for user is constant, only one exponentiation operation. Whereas the original ABE scheme [1] requires $2d$ pairing as well as $2d$ exponentiation operations for a single decryption, where d is the threshold value.

Concerning on the communication complexity in our scheme, user has to send a private key request to AA and receive $2|\omega|+2$ elements in $\mathbb{G}$. Furthermore, he is able to send blinded transformation key (as well as $2|\omega|+2$ elements in $\mathbb{G}$) to DSP to perform partially decryption in future. In general, an element in $\mathbb{G}$ is set to be 160-bit long for 2^{80} security. The data transferred among the cloud service providers, AA and user is tens of KBs at most, which can be processed efficiently.

E. Security Analysis

Theorem 1: The outsourced ABE scheme is indistinguishable secure against chosen-plaintext attack in selective model under DBDH assumption.

Proof: For the purpose of proving the security of our scheme, we should show that it is secure againt both type-I and type-II adversaries. It is noted that the main difference between type-I and type-II adversary is the oracles provided. Specifically, type-I adversary is provided with $\mathcal{O}_{SK}(\cdot)$ while challenger gives $\mathcal{O}_{OK_{\text{KGSP}}}(\cdot)$ to type-II adversary. Thus, we will utilize a bit b to denote this difference (i.e., if $b=0$, adversary is provided with $\mathcal{O}_{SK}(\cdot)$; and otherwise with $\mathcal{O}_{OK_{\text{KGSP}}}(\cdot)$) and attempt to provide a single simulation against both of type-I and type-II adversaries.

Assume that an adversary $\mathcal{A}$ has advantage ϵ in attacking the proposed outsourced ABE scheme in the sense of CPA security for selective model, we will build a simulator $\mathcal{S}$ that uses $\mathcal{A}$ as a sub-algorithm to solve the DBDH problem with a non-negligible probability. Therefore, in this analysis our main task is to provide a correct simulation of the real game between a challenger and an adversary $\mathcal{A}$.

Suppose the challenger flips a fair binary coin μ outside of $\mathcal{S}_{\text{I}}$'s view. If $\mu=0$, $\mathcal{S}_{\text{I}}$ is given $(X=g^x, Y=g^y, Z=g^z, T=e(g,g)^{xyz})$; otherwise, $(X=g^x, Y=g^y, Z=g^z, T=e(g,g)^v)$ for random $x,y,z,v\in\mathbb{Z}_q$. $\mathcal{S}_{\text{I}}$ is asked to output a value μ' as the guess for μ. We provide the simulation as follows.

Init. The simulator $\mathcal{S}$ runs $\mathcal{A}$ and receives the challenge attribute set ω^*.

Setup. Simulator $\mathcal{S}$ assigns the public key as follows. It sets $g_1=X, g_2=Y$ and $h=g_1^{-1}g^{-\alpha}$ where $\alpha\in_R\mathbb{Z}_q$. For $i\in\omega^*$, it randomly selects $\alpha_i\in_R\mathbb{Z}_q$ and sets $h_i=g_1^{-1}g^{\alpha_i}$. For $i\notin\omega^*$, it randomly selects $\alpha_i\in_R\mathbb{Z}_q$ and sets $h_i=g^{\alpha_i}$. Finally, $\mathcal{S}$ sends the public key $PK=(g,g_1,g_2,h,h_1,\ldots,h_n)$ to $\mathcal{A}$ where n is the number of attributes in universe.

Phase 1. $\mathcal{S}$ initializes an integer $j=0$ and an empty table T to provide $\mathcal{A}$ several oracles as follows.

- $\mathcal{O}_{\widetilde{TK}}(\omega)$: $\mathcal{S}$ sets $j=j+1$ and answers A's query in one of the following ways.
  - If $b=0$, $\mathcal{A}$ is a type-I adversary. In this case, $\mathcal{S}$ selects $x_2\in_R\mathbb{Z}_q$ and attempts to simulate $TK_{\text{KGSP}}=(\{d_{i0},d_{i1}\}_{i\in\omega})$ as follows. Define three sets Γ,Γ' and S with $\Gamma=\omega\cap\omega^*, |\Gamma'|=d-1, \Gamma\subseteq\Gamma'\subseteq\omega$ and $S=\Gamma'\cup\{0\}$. Then, for each $i\in\Gamma'$, compute $(d_{i0},d_{i1})=(g_2^{\tau_i}(g_1h_i)^{r_i}, g^{r_i})$ where $\tau_i, r_i\in_R\mathbb{Z}_q$. For each attribute $i\in\omega-\Gamma'$, the simulator set $r_i=-y\Delta_{0,S}(i)+r_i'$ and obtains $d_{i0}=g_2^{\sum_{j\in\Gamma'}\Delta_{j,S}(i)\tau_j-(x_2+\alpha_i)\Delta_{0,S}(i)}(g_1h_i)^{r_i'}$ and $d_{i1}=g_2^{-\Delta_{0,S}(i)}g^{r_i'}$), where $r_i'\in_R\mathbb{Z}_q$. The intuition behind these assignments is that $q(i)=\sum_{j\in\Gamma'}\Delta_{j,S}(i)q(j)+\Delta_{0,S}(i)q(0)$ and we are implicitly choosing a random $d-1$ degree polynomial $q(\cdot)$ by choosing its value for the $d-1$ points randomly as $q(i)=\tau_i$ for $i\in\Gamma'$, in

Schemes	Key generation (AA)	Key generation (KGSP)	Decryption (U)	Decryption (DSP)
original ABE [1]	$2\|\omega\|$EXP	–	$2d$P+$2d$EXP	–
outsourced ABE in [2]	$2\|\omega\|$EXP	–	EXP	$2d$P+$2d$EXP
outsourced ABE in [3]	$2\|\omega\|$EXP	–	EXP	$2d$P+$2d$EXP
outsourced ABE in this paper	2EXP	$2\|\omega\|$EXP	EXP	$2d$P+$2d$EXP

The symbol '–' denotes that this property is not considered in the corresponding scheme.

TABLE II
EFFICIENCY COMPARISON

addition to having $q(0) = x - x_2$. Next, $\mathcal{S}$ runs $TK_{\mathrm{AA}} = \mathsf{KeyGen}_{\mathrm{in}}(\omega, x_2)$ and $\mathsf{KeyBlind}(TK = (TK_{\mathrm{KGSP}}, TK_{\mathrm{AA}}))$ to obtain SK and $\widetilde{TK}$. After adding the entry $(j, \omega, SK, \cdot)$ into T, return $\widetilde{TK}$.

– Otherwise, $\mathcal{A}$ is type-II adversary. $\mathcal{S}$ selects $x_1 \in_R \mathbb{Z}_q$ and computes $TK_{\mathrm{KGSP}} = \mathsf{KeyGen}_{\mathrm{out}}(\omega, x_1)$. Meanwhile, it is to simulate $(d_{\theta 0}, d_{\theta 1}) = (g_2^{x-x_1}(g_1 h)^{r_\theta}, g^{r_\theta})$ by setting $r_\theta = \frac{xy}{\alpha}$ and computing $(d_{\theta 0}, d_{\theta 1}) = (g_2^{-x_1}, g^{\frac{xy}{\alpha}})$. Next, pick $t = \frac{1}{y}$ and obtain $\widetilde{TK} = \{d_{i0}^t, d_{i1}^t\}_{i \in \omega \cup \{\theta\}}$. After adding the entry $(j, \omega, SK = (t, TK), x_1)$ into T, return $\widetilde{TK}$.

- $\mathcal{O}_{SK}$: This oracle is only provided with type-I adversary (i.e. $b = 0$). Upon receiving i, if there is an entry $(i, \omega, SK, \cdot)$ with $|\omega \cap \omega^*| \geq d$ in T, $\mathcal{S}$ returns SK.
- $\mathcal{O}_{OK_{\mathrm{KGSP}}}(i)$: This oracle is only provided with type-II adversary (i.e. $b = 1$). Upon receiving i, if there is an entry (i, ω, SK, x_1) in T, $\mathcal{S}$ returns x_1.

Challenge. The adversary $\mathcal{A}$ will submit two challenge messages M_0 and M_1 to $\mathcal{S}$. The simulator flips a fair binary coin ν and returns an encryption of M_ν. The ciphertext is simulated as $CT^* = (\omega^* \cup \{\theta\}, M_\nu T, g^z, g^{-z\alpha}, \{g^{z\alpha_i}\}_{i \in \omega^*})$. We note that: i) If $\mu = 0$, then $T = e(g,g)^{xyz}$. If we let $s = z$, then we have $C_0 = M_\nu T = M_\nu e(g,g)^{xyz} = M_\nu e(g_1, g_2)^z, C_1 = g^z, E_\theta = g^{-z\alpha} = (g_1 g_1^{-1} g^{-\alpha})^z = (g_1 h)^z$ and $E_i = g^{z\alpha_i} = (g_1 g_1^{-1} g^{\alpha_i})^z = (g_1 h_i)^z$ for $i \in \omega^*$. Therefore, the ciphertext is random encryption of the message M_ν under the attribute set ω^*. ii) Otherwise, if $\mu = 1$, then $T = e(g,g)^v$. We then have $C_0 = M_\nu e(g,g)^v$. Since v is random, C_0 will be a random element in $\mathbb{G}_T$ from $\mathcal{A}$'s view and the encrypted message contains no information about M_ν.

Phase 2. Phase 1 is repeated with the restriction that $\mathcal{A}$ cannot issue private key query on ω in the case $b = 0$, where $|\omega \cap \omega^*| \geq d$.

Guess. $\mathcal{A}$ will submit a guess ν' of ν. If $\nu' = \nu$ the simulator $\mathcal{S}$ will output $\mu' = 0$ to indicate that it was given a DBDH-tuple otherwise it will output $\mu' = 1$ to indicate it was given a random 4-tuple.

In the case where $\mu = 1$, $\mathcal{A}$ has no information about ν. Therefore, we have $\mathsf{Pr}[\nu \neq \nu' | \mu = 1] = \frac{1}{2}$. Since $\mathcal{S}$ guesses $\mu' = 1$ when $\nu \neq \nu'$, we have $\mathsf{Pr}[\mu' = \mu | \mu = 1] = \frac{1}{2}$.

If $\mu = 0$, then the adversary sees an encryption of M_ν. It has an advantage ϵ in this situation by definition. Therefore, we have $\mathsf{Pr}[\nu = \nu' | \mu = 0] = \frac{1}{2} + \epsilon$. Since $\mathcal{S}$ guesses $\mu' = 0$ when $\nu = \nu'$, we have $\mathsf{Pr}[\mu' = \mu | \mu = 0] = \frac{1}{2} + \epsilon$.

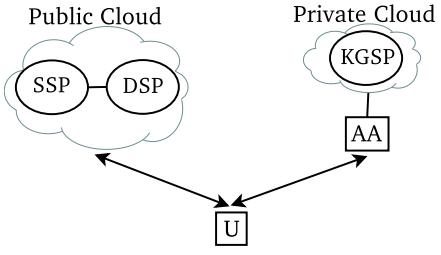


Fig. 4. Application in Hybrid Cloud Setting

The overall advantage of $\mathcal{S}$ in the DBDH game is $\frac{1}{2}\mathsf{Pr}[\mu' = \mu | \mu = 0] + \frac{1}{2}\mathsf{Pr}[\mu' = \mu | \mu = 1] - \frac{1}{2} = \frac{1}{2}(\frac{1}{2} + \epsilon) + \frac{1}{2}\frac{1}{2} - \frac{1}{2} = \frac{1}{2}\epsilon$. ■

Finally, we note that though our construction is CPA-secure, it is allowed to be extended to the stronger RCCA-security guarantee by using simulation-sound NIZK proofs [25]. Alternatively, if we are willing to use random oracle, then we can use standard techniques such as the Fujisaki-Okamoto transformation [26].

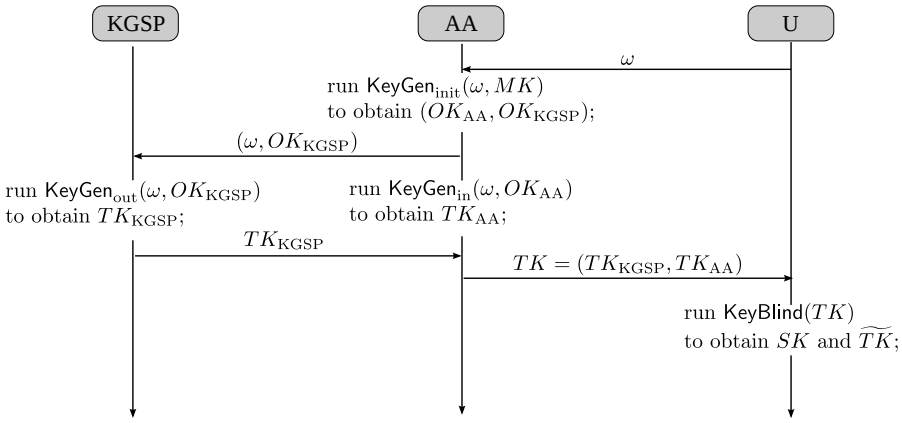
F. Practical Consideration

As illustrated in Fig. 4, we can consider to utilize our construction in hybrid clouds. More precisely, KGSP is maintained as a private cloud with high trust to deal with sensitive information, but leaving SSP and DSP as public cloud to provide public storage and computation service respectively. Actually, this type of hybrid setting has become more and more attractive as many organizations are moving to the public cloud due to its benefit of highly available and scalable resources but still want to store and process the critical data in the private cloud.

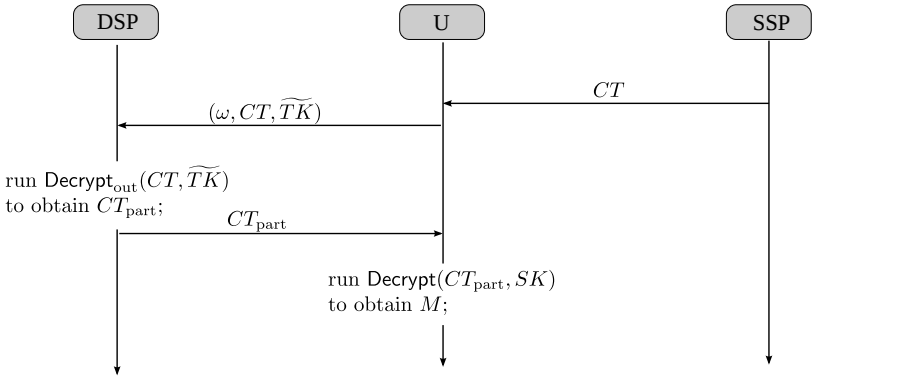
As shown in Fig. 5, we provide the working process of proposed construction for outsourced key generation and decryption.

In the outsourced key generation shown in Fig. 5(a), AA and KGSP are allowed to parallelly perform computation to produce partial transformation key for customized and default attributes. Specifically, after producing the key pair $(OK_{\mathrm{KGSP}}, OK_{\mathrm{AA}})$ with $\mathsf{KeyGen}_{\mathrm{init}}(\omega, MK)$, two individual phase can be executed simultaneously. i) At KGSP, OK_{KGSP} is involved to generate partial transformation key for customized attribute set ω. ii) At AA, OK_{AA} is involved to generate the other partial transformation key for default attribute θ. The parallel computation is benefit for improving efficiency in key generation for ABE system.

In the outsourced decryption shown in Fig. 5(b), user firstly fetches ciphertext from SSP and computes the inter-



(a) Outsourced Key Generation



(b) Outsourced Decryption

Fig. 5. Outsourced Key Generation and Decryption

section subset S locally. Therefore, only a partial ciphertext, blinded transformation key and intersection subset need to be delivered to DSP to perform partial decryption. Alternatively, it allows another scenario, in which after key generation user directly sends his attribute set ω and corresponding blinded transformation key $\widetilde{TK}$ to DSP to be stored. In this case, the DSP performs a role as proxy, who can automatically retrieve ciphertexts that user is interested in and forward to him partially decrypted one. The DSP could be the user's mail server, or the same entity along with SSP in cloud environment.

V. Another Construction with Checkability

We observe that in the commercial cloud computing, for saving computation or bandwidththe, CSPs may be selfish to execute only a fraction of delegated operation and return result incorrectly. The dishonest action of CSPs may cause users obtain incorrect keys or messages. We also point out that our first construction provides provable secrecy in the sense that KGSP is maintained as a private cloud with high trust. By this, we mean that an untrusted KGSP is able to collaborate with user to fake private key to enhance his "power". More precisely, Suppose a user and the KGSP collude together. They are able to obtain $OK_{\mathrm{KGSP}} = x_1$ and $\{d_{\theta 0}, d_{\theta 1}\}$ corresponding to this user. With this possession, they can generate TK'_{KGSP} for target attribute set ω' and joint it with $\{d_{\theta 0}, d_{\theta 1}\}$ to obtain the faked TK' . Using this one, ciphertext satisfied by ω' can be decrypted.

Therefore, in this section, aiming at providing checkability as well as reducing the trust on KGSP, we propose another construction under the widely used RDoC model.

A. Outsourced ABE in Refereed Delegation of Computation Model

RDoC model originates from the model of refereed games in [27], and is later formalized in [18][28]. In RDoC model, the client is able to interact with multiple servers and it has a right output as long as there exists one server that follows the proposed protocol. One of the most advantages of RDoC over traditional model with single server is that the security risk on the single server is reduced to multiple

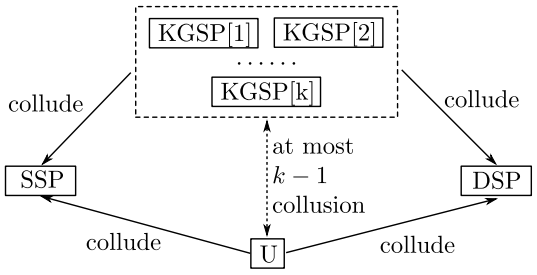


Fig. 6. Adversary Model for Outsourced ABE under RDoC

servers involved in. As the result of both the practicality and utility, RDoC model recently has been widely utilized in the literature of outsourced computation [18][28][29][30][14].

To reduce the trust on KGSP, we will consider the outsourced ABE system in RDoC model, in which k KGSPs cooperatively work together to provide AA with the key generation service ($k-1$ are malicious at most). In this case, as the adversary model illustrated in Fig. 6, an additional collusion between user and at most $k-1$ malicious KGSPs is allowed. Then, the two types adversaries defined in Section III.B are semi-merged together and able to obtain $OK_{\mathrm{KGSP}[i]}$ for malicious $\mathrm{KGSP}[i]$, private keys for all the corrupted users, all the blinded transformation keys stored at DSP and so on, where $i = 1, 2, \ldots, k-1$.

Having this intuition above, we can redefine the security definition of our setting for RDoC model. The challenger will maintain the table T, set D and integer j to provide the adversary with three type of oracles (if RCCA is considered $\mathcal{O}_M(\cdot)$ should be added).

- $\mathcal{O}_{OK_{\mathrm{KGSP}}}(I_{\mathrm{key}}, b)$: Challenger sets $j = j+1$ and runs key generation (including key blinding) completely for I_{key} to obtain $SK, \{OK_{\mathrm{KGSP}[i]}\}_{i=1}^{k}$ and $\widetilde{TK}$. After adding the entry $(j, I_{\mathrm{key}}, SK, \{OK_{\mathrm{KGSP}[i]}\}_{i=1}^{k}, \widetilde{TK})$ into T, return $\{OK_{\mathrm{KGSP}[i]}\}_{i=1, i\neq b}^{k}$.
- $\mathcal{O}_{\widetilde{TK}}(i)$: The challenger checks whether the entry $(i, I_{\mathrm{key}}, SK, \{OK_{\mathrm{KGSP}[j]}\}_{j=1}^{k}, \widetilde{TK})$ exists in T, if so return $\widetilde{TK}$; otherwise return $\perp$.
- $\mathcal{O}_{SK}(i)$: The challenger checks whether the entry $(i, I_{\mathrm{key}}, SK, \{OK_{\mathrm{KGSP}[j]}\}_{j=1}^{k}, \widetilde{TK})$ exists in T, if so set $D = D \cup \{I_{\mathrm{key}}\}$ and return SK, otherwise return $\perp$.

B. Intuition for Proposed Construction

For simplicity, we only consider and provide the second construction with two KGSPs. The key challenge for our second construction exists in two folds.

- One is how to prevent from the collusion between the user and the malicious KGSP. Our solution is to intelligently extend the hybrid policy trick in the first construction. Specifically, in addition to building an AND gate between $\mathcal{P}_{\mathrm{AA}}$ and $\mathcal{P}_{\mathrm{KGSP}}$, we introduce a $(2,2)$-secret sharing on $\mathcal{P}_{\mathrm{KGSP}}$ and make each KGSP only know its own share $OK_{\mathrm{KGSP}[i]}$ for $i = 1, 2$. In this sense, even if user collude with a KGSP and obtain $\{OK_{\mathrm{KGSP}[i]}\}$ for $i = 1$ or 2, he cannot recover the secret (which is actually x_1 in our construction) to serve the devil.
- The other is how to detect the dishonest action from KGSPs and DSP beyond collusion. To fight against it,
  - we extend the idea of "ringer" [4] to our setting to convince that KGSPs do indeed perform all the computations that were outsourced to them. More precisely, AA generates a random value $(d-1)$-degree polynomial $q_{\mathrm{RG}}(\cdot)$ and sends it along with $q_{\mathrm{KGSP}[i]}(\cdot)$ in a random order to $\mathrm{KGSP}[i]$. Each KGSP generates partial transformation key using both $q_{\mathrm{RG}}(\cdot)$ and $q_{\mathrm{KGSP}[i]}(\cdot)$, and AA detects the dishonest action by checking all the partial transformation key computed from $q_{\mathrm{RG}}(\cdot)$ (to make sure that all the honest KGSPs will obtain the same result from OK_{RG} in a honest computation, the random values $\{r_i\}$ for $q_{\mathrm{RG}}(\cdot)$ should be selected by AA in advance).
  - In addition, we detect the dishonest action of a malicious DSP by adding redundancy. Specifically, we can require that all the users in the system agree on a redundancy 0^k (i.e., a k-length 0 bit string) and append it with original message in each encryption. Then, after performing complete decryption to obtain the plaintext, the user can detect the dishonest action of DSP by checking the redundancy.

C. The Construction under RDoC Model

We provide our second construction with two KGSPs as follows.

- $\mathsf{Setup}(\lambda)$: It is similar to the same algorithm in our previous construction but an integer k should be agreed in public key. Specifically, the $PK = \{g, g_1, g_2, h, h_1, \ldots, h_n, k\}$ and $MK = x$ are output.
- $\mathsf{KeyGen}_{\mathrm{init}}(\omega, MK)$: For each user's private key request on ω, AA picks $x_{11}, x_{12} \in_R \mathbb{Z}_q$ and sets $OK_{\mathrm{KGSP}[1]} = x_{11}, OK_{\mathrm{KGSP}[2]} = x_{12}$ and $OK_{\mathrm{AA}} = x_2 = x - x_{11} - x_{12} \mod q$. Next, select $(d-1)$-degree random polynomials $q_{\mathrm{KGSP}[1]}(\cdot)$ and $q_{\mathrm{KGSP}[2]}(\cdot)$ with the restrictions: i) let ω' be any $(d-1)$-element subset of ω, $q_{\mathrm{KGSP}[1]}(i) = q_{\mathrm{KGSP}[2]}(i)$ for each $i \in \omega'$; ii) $q_{\mathrm{KGSP}[1]}(0) = x_{11}$; iii) $q_{\mathrm{KGSP}[2]}(0) = x_{12}$. Thirdly, to enable convincing the dishonest action of KGSPs later, select another random polynomial $q_{\mathrm{RG}}(\cdot)$. Furthermore, for each $i \in \omega$, pick $r_{\mathrm{KGSP}[1],i}, r_{\mathrm{KGSP}[2],i}, r_{\mathrm{RG},i} \in_R \mathbb{Z}_q$ with the restriction that $r_{\mathrm{KGSP}[1],j} = r_{\mathrm{KGSP}[2],j}$ where $j \in \omega'$. Finally, AA sends $(S[1]_{\mathrm{REAL}}, S_{\mathrm{RG}})$ and $(S[2]_{\mathrm{REAL}}, S_{\mathrm{RG}})$ to KGSP[1] and KGSP[2] respectively, where the pair $S[j]_{\mathrm{REAL}} = (q_{\mathrm{KGSP}[j]}(\cdot), \{r_{\mathrm{KGSP}[j],i}\}_{i\in\omega})$ and $S_{\mathrm{RG}} = (q_{\mathrm{RG}}(\cdot), \{r_{\mathrm{RG},i}\}_{i\in\omega})$ for $j = 1, 2$. We emphasize that in the both communications $S[\cdot]_{\mathrm{REAL}}$ and S_{RG} should be sent in random orders to avoid KGSPs knowing which one is really to be computed for partial transformation key.

- $\mathsf{KeyGen}_{\mathrm{out}}(S[j]_{\mathrm{REAL}}, S_{\mathrm{RG}})$: $\mathrm{KGSP}[j]$ generates partial transformation key for both $q_{\mathrm{KGSP}[j]}(\cdot)$ and $q_{\mathrm{RG}}(\cdot)$. More precisely, $\mathrm{KGSP}[j]$ computes

$$TK_{\mathrm{KGSP}[j]} = (\{d[j]_{i0}, d[j]_{i1}\}_{i\in\omega})$$

where $d[j]_{i0} = g_2^{q_{\mathrm{KGSP}[j](i)}}(g_1h_i)^{r_{\mathrm{KGSP}[j],i}}, d[j]_{i1} = g^{r_{\mathrm{KGSP}[j],i}}$ and

$$TK_{\mathrm{RG}_j} = (\{d[\mathrm{RG}_j]_{i0}, d[\mathrm{RG}_j]_{i1}\})$$

where $d[\mathrm{RG}_j]_{i0} = g_2^{q_{\mathrm{RG}}(i)}(g_1h_i)^{r_{\mathrm{RG},i}}, d[\mathrm{RG}_j]_{i1} = g^{r_{\mathrm{RG},i}}$, and sends $(TK_{\mathrm{KGSP}[j]}, TK_{\mathrm{RG}_j})$ to AA in its receiving order.
- $\mathsf{KeyGen}_{\mathrm{in}}(\omega, OK_{\mathrm{AA}})$: Similar to the same algorithm described in our basic construction, AA selects $r_\theta \in_R \mathbb{Z}_q$and computes $d_{\theta 0} = g_2^{x_2}\cdot(g_1h)^{r_\theta}$ and $d_{\theta 1} = g^{r_\theta}$. Finally output $TK_{\mathrm{AA}} = (\{d_{\theta 0}, d_{\theta 1}\})$.
- $\mathsf{KeyCheck}(TK_{\mathrm{KGSP}[1]}, TK_{\mathrm{RG}_1}, TK_{\mathrm{KGSP}[2]}, TK_{\mathrm{RG}_2})$: AA checks that both KGSPs produce the correct outputs, i.e., $d[1]_{j0} = d[2]_{j0}, d[1]_{j1} = d[2]_{j1}$ for all $j \in \omega'$ and $d[\mathrm{RG}_1]_{i0} = d[\mathrm{RG}_2]_{i0}, d[\mathrm{RG}_1]_{i1} = d[\mathrm{RG}_2]_{i1}$ for all $i \in \omega$. After that, continue to combine the partial transformation key together by computing $d_{i0} = d[1]_{i0}\cdot d[2]_{i0}$ and $d_{i1} = d[1]_{i1}\cdot d[2]_{i1}$ for all $i \in \omega$. Finally output the complete transformation key $TK = (\{d_{i0}, d_{i1}\}_{i\in\omega\cup\{\theta\}})$.
- $\mathsf{KeyBlind}(TK)$: It is identical to the same algorithm in our previous construction and outputs $SK = (t, TK)$ and $\widetilde{TK}$.
- $\mathsf{Encrypt}(M, \omega')$: User firstly appends the message M to be encrypted with a redundancy 0^k to obtain $M_T = M||0^k$ where $||$ is the concatenation of string. Then, the rest is identical to the same algorithm in the first construction but to encrypt M_T. Finally, output $CT = (\omega'\cup\{\theta\}, M_T e(g_1, g_2)^s, g^s, \{g_1h_i\}_{i\in\omega'}, g_1h)$.
- $\mathsf{Decrypt}_{\mathrm{out}}(CT, \widetilde{TK})$: It is identical to the same algorithm in previous construction and outputs $CT_{\mathrm{part}} = e(g_1, g_2)^{st}$.
- $\mathsf{Decrypt}(CT_{\mathrm{part}}, SK)$: It is identical to the same algorithm in previous construction except that the dishonest action of DSP should be detected through checking redundancy. Specifically, by executing the decryption algorithm in previous construction, M_T is obtained. The user continues to check whether a redundancy 0^k is appended with M_T. If so (i.e., $M_T = M||0^k$), M is obtained through truncation; otherwise, a dishonest action of DSP is detected.

D. Analysis

Our second construction has almost the same efficiency with the first one. Specifically, in key-issuing, though another key combination operation is required at attribute authority side, it costs mutiplications for $|\omega|$ times, which is negligible using the modern devices.

Then, we provide the security analysis below.

Theorem 2: The second construction is secure against chosen-plaintext attack in the sense of the security definition modified in Section V.A under DBDH assumption.

Proof: Similar to theorem 1, we will build a simulator $\mathcal{S}$ as follows.

Init. $\mathcal{S}$ runs $\mathcal{A}$ and receives the challenge set ω^*.

Setup. $\mathcal{S}$ sets $g_1 = X, g_2 = Y$ and $h = g_1^{-1}g^{-\alpha}$ for $\alpha \in_R \mathbb{Z}_q$. For $i \in \omega^*$, $h_i = g_1^{-1}g^{\alpha_i}$ where $\alpha_i \in_R \mathbb{Z}_q$. For $i \notin \omega^*$, $h_i = g^{\alpha_i}$ where $\alpha_i \in_R \mathbb{Z}_q$. Finally, $\mathcal{S}$ sends the public key $PK = (g, g_1, g_2, h, h_1, \ldots, h_n)$ to $\mathcal{A}$ where n is the number of attributes in universe.

Phase 1. $\mathcal{S}$ initializes an integer $j = 0$ and an empty table T to provide $\mathcal{A}$ oracles as follows.

- $\mathcal{O}_{OK_{\mathrm{KGSP}}}(\omega, b)$: $\mathcal{S}$ firstly selects $x_2, x_{1b} \in_R \mathbb{Z}_q$, and randomly picks a $(d-1)$-degree polynomial $q_{\mathrm{KGSP}[b]}(\cdot)$ with $q_{\mathrm{KGSP}[b]}(0) = x_{1b}$. $\mathcal{S}$ continues to compute the partial transformation keys $TK_{\mathrm{KGSP}[b]} = (\{g_2^{q_{\mathrm{KGSP}[b](i)}}(g_1h_i)^{r_{\mathrm{KGSP}[b],i}}, g^{r_{\mathrm{KGSP}[b],i}}\}_{i\in\omega})$ for KGSP[b] and $TK_{\mathrm{AA}} = (g_2^{x_2}(g_1h)^{r_\theta}, g^{r_\theta})$ for AA, where $r_{\mathrm{KGSP}[b],i}, r_\theta \in_R \mathbb{Z}_q$ for each $i \in \omega$. Furthermore, define three sets Γ, Γ' and S with $\Gamma = \omega\cap\omega^*, |\Gamma'| = d-1, \Gamma \subseteq \Gamma' \subseteq \omega$ and $S = \Gamma'\cup\{0\}$. $\mathcal{S}$ sets $r_{\mathrm{KGSP}[1-b],i} = -y\Delta_{j,S}(0) + r'_i$ where $r'_i \in_R \mathbb{Z}_q$, and simulates $TK_{\mathrm{KGSP}[1-b]} = (\{d[1-b]_{i0}, d[1-b]_{i1}\}_{i\in\omega})$:
  1) For each attribute $i \in \Gamma'$, set $d[1-b]_{i0} = g_2^{\tau_i}(g_1h_i)^{r_{\mathrm{KGSP}[1-b],i}}$ and $d[1-b]_{i1} = g^{r_{\mathrm{KGSP}[1-b],i}}$, where $\tau_i = q_{\mathrm{KGSP}[b]}(i)$ and $r_{\mathrm{KGSP}[1-b],i} = r_{\mathrm{KGSP}[b],i}$.
  2) For each attribute $i \in \omega - \Gamma'$, set $d[1-b]_{i0} = g_2^{\sum_{j\in\Gamma'}\Delta_{j,S}(i)\tau_j - (x_2+x_{1b}+\alpha_i)\Delta_{j,S}(0)}\cdot(g_1h_i)^{r'_i}$ and $d[1-b]_{i1} = g^{-y\Delta_{j,S}(0)+r'_i}$.
  $\mathcal{S}$ furthermore computes $TK = (\{d_{i0}, d_{i1}\}_{i\in\omega\cup\{\theta\}})$ where $d_{i0} = d[b]_{i0}\cdot d[1-b]_{i0}$ and $d_{i1} = d[b]_{i1}\cdot d[1-b]_{i1}$ for $i \in \omega$. After setting $j = j+1$ and adding the entry $(j, \omega, \cdot, x_{1b}, x_{1(2-b)}, \cdot)$ into T, return x_{1b}.
- $\mathcal{O}_{\widetilde{TK}}(i)$: The challenger checks whether the entry $(i, \omega, \cdot, OK_{\mathrm{KGSP}[1]}, OK_{\mathrm{KGSP}[2]}, \cdot)$ exists in T, if so pick $t = \frac{t'}{y}$ where $t' \in_R \mathbb{Z}_q$ and compute $\widetilde{TK} = \{d_{i0}^t, d_{i1}^t\}_{i\in\omega\cup\{\theta\}}$. After complete this entry, return $\widetilde{TK}$. If no such entry exists, return $\perp$.
- $\mathcal{O}_{SK}(i)$: If there exists an entry in the form of $(i, \omega, \cdot, OK_{\mathrm{KGSP}[1]}, OK_{\mathrm{KGSP}[2]}, \cdot)$ in T satisfying $\gamma_d(\omega, \omega^*) = 0$, it is identical to $\mathcal{O}_{\widetilde{TK}}(i)$ to simulate the key blinding procedure but returns SK. Otherwise, return $\perp$.

Challenge. $\mathcal{A}$ will submit two challenge messages M_0 and M_1 to $\mathcal{S}$. The simulator flips a random coin ν and returns an encryption of M_ν. The ciphertext is simulated as $CT^* = (\omega^*\cup\{\theta\}, M_\nu T, g^z, g^{-z\alpha}, \{g^{z\alpha_i}\}_{i\in\omega^*})$.

Phase 2. Phase 1 is repeated with the restriction that $\mathcal{A}$ cannot issue query on ω to $\mathcal{O}_{SK}(\cdot)$ where $\gamma_d(\omega, \omega^*) = 1$.

Guess. $\mathcal{A}$ will submit a guess ν' of ν. If $\nu' = \nu$ the simulator $\mathcal{S}$ will output $\mu' = 0$ to indicate that it was given a DBDH-tuple otherwise it will output $\mu' = 1$ to indicate it was given a random 4-tuple.

Similar to the proof of theorem 1, the overall advantage of $\mathcal{S}$ in the DBDH game is $\frac{1}{2}\mathsf{Pr}[\mu' = \mu|\mu = 0] + \frac{1}{2}\mathsf{Pr}[\mu' = \mu|\mu = 1] - \frac{1}{2} = \frac{1}{2}\epsilon$. ■

Scheme	Advanced construction	Construction in [3]	Construction in [2]
outsourced key generation	$\surd$	$\times$	$\times$
outsourced decryption	$\surd$	$\surd$	$\surd$
checkability	$\surd$	$\times$	$\times$

TABLE III
FUNCTIONALITY COMPARISON

E. Functionality Comparison

Beyond outsourced key generation and decryption, the checkability on KGSP is supported in our second construction. Specifically, since $\mathrm{KGSP}[1]$ (or $\mathrm{KGSP}[2]$) cannot distinguish the outsourced private key generation from the two outsourced tasks. If $\mathrm{KGSP}[1]$ ($\mathrm{KGSP}[2]$) fails during any execution of $\mathsf{KeyGen}_{\mathrm{out}}(\cdot)$, it will be detected with probability $\frac{d-1+|\omega|}{2|\omega|}$ which is not less than $\frac{1}{2}$. In addition, through appending redundancy, the dishonest action of DSP can be easily detected in our construction.

We provide a functionality comparison between our second construction and other outsourced ABE in TABLE III.

VI. CONCLUSION

In this paper, for the first time, we provide an outsourced ABE scheme simultaneously supporting outsourced key-issuing and decryption. With the aid of KGSP and DSP, our scheme achieves constant efficiency at both authority and user sides. In addition, we provide a trust-reduced construction with two KGSPs which is secure under recently formulized RDoC model. Unlike the state-of-the-art outsourced ABE, checkability is supported by this construction.

REFERENCES

[1] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology - EUROCRYPT 2005*, ser. Lecture Notes in Computer Science, R. Cramer, Ed. Springer Berlin / Heidelberg, 2005, vol. 3494, pp. 457–473.

[2] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proceedings of the 20th USENIX conference on Security*, ser. SEC'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 34–34.

[3] Z. Zhou and D. Huang, "Efficient and secure data storage operations for mobile cloud computing," Cryptology ePrint Archive, Report 2011/185, 2011.

[4] P. Golle and I. Mironov, "Uncheatable distributed computations," in *Proceedings of the 2001 Conference on Topics in Cryptology: The Cryptographer's Track at RSA*, ser. CT-RSA 2001. London, UK, UK: Springer-Verlag, 2001, pp. 425–440.

[5] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 89–98.

[6] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE Symposium on Security and Privacy 2007*, may 2007, pp. 321–334.

[7] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proceedings of the 14th ACM conference on Computer and communications security*, ser. CCS '07, 2007, pp. 456–465.

[8] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *Applied Cryptography and Network Security*, ser. Lecture Notes in Computer Science, S. Bellovin, R. Gennaro, A. Keromytis, and M. Yung, Eds. Springer Berlin / Heidelberg, 2008, vol. 5037, pp. 111–129.

[9] J. Li, K. Ren, B. Zhu, and Z. Wan, "Privacy-aware attribute-based encryption with user accountability," in *Information Security*, ser. Lecture Notes in Computer Science, P. Samarati, M. Yung, F. Martinelli, and C. Ardagna, Eds. Springer Berlin / Heidelberg, 2009, vol. 5735, pp. 347–362.

[10] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Proceedings of the 29th conference on Information communications*, ser. INFOCOM'10. Piscataway, NJ, USA: IEEE Press, 2010, pp. 534–542.

[11] M. J. Atallah, K. Pantazopoulos, J. R. Rice, and E. E. Spafford, "Secure outsourcing of scientific computations," in *Trends in Software Engineering*, ser. Advances in Computers, M. V. Zelkowitz, Ed. Elsevier, 2002, vol. 54, pp. 215 – 272.

[12] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons," *International Journal of Information Security*, vol. 4, pp. 277–287, 2005.

[13] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in *Proceedings of the 2008 Sixth Annual Conference on Privacy, Security and Trust*, ser. PST '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 240–245.

[14] M. J. Atallah and K. B. Frikken, "Securely outsourcing linear algebra computations," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS '10. New York, NY, USA: ACM, 2010, pp. 48–59.

[15] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2011, pp. 820–828.

[16] K. Bicakci and N. Baykal, "Server assisted signatures revisited," in *Topics in Cryptology - CT-RSA 2004*, ser. Lecture Notes in Computer Science, T. Okamoto, Ed. Springer Berlin / Heidelberg, 2004, vol. 2964, pp. 1991–1992.

[17] M. Jakobsson and S. Wetzel, "Secure server-aided signature generation," in *Public Key Cryptography*, 2001, pp. 383–401.

[18] S. Hohenberger and A. Lysyanskaya, "How to securely outsource cryptographic computations," in *Theory of Cryptography*, ser. Lecture Notes in Computer Science, J. Kilian, Ed. Springer Berlin / Heidelberg, 2005, vol. 3378, pp. 264–282.

[19] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum, "Delegating computation: interactive proofs for muggles," in *Proceedings of the 40th annual ACM symposium on Theory of computing*, ser. STOC '08. New York, NY, USA: ACM, 2008, pp. 113–122.

[20] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st annual ACM symposium on Theory of computing*, ser. STOC '09. New York, NY, USA: ACM, 2009, pp. 169–178.

[21] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Advances in Cryptology - CRYPTO 2010*, ser. Lecture Notes in Computer Science, T. Rabin, Ed. Springer Berlin / Heidelberg, 2010, vol. 6223, pp. 465–482.

[22] K.-M. Chung, Y. Kalai, F.-H. Liu, and R. Raz, "Memory delegation," in *Advances in Cryptology - CRYPTO 2011*, ser. Lecture Notes in Computer Science, P. Rogaway, Ed. Springer Berlin / Heidelberg, 2011, vol. 6841, pp. 151–168.

[23] C. Gentry and S. Halevi, "Implementing gentry's fully-homomorphic encryption scheme," in *Advances in Cryptology - EUROCRYPT 2011*, ser. Lecture Notes in Computer Science, K. Paterson, Ed. Springer Berlin / Heidelberg, 2011, vol. 6632, pp. 129–148.

[24] R. Canetti, H. Krawczyk, and J. Nielsen, "Relaxing chosen-ciphertext security," in *Advances in Cryptology - CRYPTO 2003*, ser. Lecture Notes in Computer Science, D. Boneh, Ed. Springer Berlin / Heidelberg, 2003, vol. 2729, pp. 565–582.

[25] A. Sahai, "Non-malleable non-interactive zero knowledge and adap-

tive chosen-ciphertext security," in *1999. 40th Annual Symposium on Foundations of Computer Science*, 1999, pp. 543–553.

[26] E. Fujisaki and T. Okamoto, "Secure integration of asymmetric and symmetric encryption schemes," in *Proceeding of the 19th Annual International Cryptology Conference on Advances in Cryptology*, ser. Lecture Notes in Computer Science, M. Wiener, Ed. Springer Berlin / Heidelberg, 1999, vol. 1666, pp. 537–554.

[27] U. Feige and J. Kilian, "Making games short (extended abstract)," in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, ser. STOC '97. New York, NY, USA: ACM, 1997, pp. 506–516.

[28] R. Canetti, B. Riva, and G. Rothblum, "Two protocols for delegation of computation," in *Information Theoretic Security*, ser. Lecture Notes in Computer Science, A. Smith, Ed. Springer Berlin / Heidelberg, 2012, vol. 7412, pp. 37–61.

[29] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," in *Computer Security ESORICS 2012*, ser. Lecture Notes in Computer Science, S. Foresti, M. Yung, and F. Martinelli, Eds. Springer Berlin / Heidelberg, 2012, vol. 7459, pp. 541–556.

[30] R. Canetti, B. Riva, and G. N. Rothblum, "Practical delegation of computation using multiple servers," in *Proceedings of the 18th ACM conference on Computer and communications security*, ser. CCS '11. New York, NY, USA: ACM, 2011, pp. 445–454.