Secure Outsourced Attribute-Based Signatures

Jin Li[1], Xiaofeng Chen[2], Jingwei Li[3], Chunfu Jia[3], Duncan S. Wong[4], Willy Susilo[5]

[1] School of Computer Science and Educational Software, Guangzhou University, China
[2] State Key Laboratory of Integrated Service Networks (ISN), Xidian University, China
[3] College of Information Technical Science, Nankai University, China
[4] Department of Computer Science, City University of Hong Kong, China
[5] School of Computer Science and Software Engineering, University of Wollongong, Australia

Abstract. Attribute-based signature (ABS) is a useful variant of digital signature, which enables users to sign messages over attributes without revealing any information other than the fact that they have attested to the messages. However, heavy computational cost is required during signing in existing work of ABS, which grows linearly with the size of the predicate formula. As a result, this presents a significant challenge for resource-limited users (such as mobile devices) to perform such heavy computation independently. Aiming at tackling the challenge above, we propose and formalize a new paradigm called OABS, in which the computational overhead at user side is greatly reduced through outsourcing such intensive computation to an untrusted signing-cloud service provider (S-CSP). Furthermore, we apply this novel paradigm to existing ABS to reduce complexity and present two schemes, i) in the first OABS scheme, the number of exponentiations involving in signing is reduced from $O(d)$ to $O(1)$ (nearly three), where d is the upper bound of threshold value defined in the predicate; ii) our second scheme is built on Herranz et al's construction with constant-size signatures. The number of exponentiations in signing is reduced from $O(d^2)$ to $O(d)$ and the communication overhead is $O(1)$. Security analysis demonstrates that both OABS schemes are secure in terms of the unforgeability and attribute-signer privacy definitions specified in the proposed security model. Finally, to allow for high efficiency and flexibility, we discuss extensions of OABS and show how to achieve accountability and outsourced verification as well.

Keywords: Attribute-based signature, outsource, cloud computing

1 Introduction

As a novel cryptographic primitive, attribute-based signature (ABS) enables a party to sign a message with fine-grained access control over identifying information. Specifically, in an ABS system, users obtain their attribute private keys from an attribute authority, with which they can later sign messages for any predicate satisfied by their attributes. A verifier will be convinced of the fact that whether the signer's attributes satisfy the signing predicate while remaining completely ignorant of the identity of signer. ABS is much useful in a wide range of applications including private access control, anonymous credentials, trust negotiations, distributed access control for ad hoc networks, attribute-based messaging, etc (refer to [29] for detailed descriptions of applications).

However, one of the main efficiency drawbacks of ABS is that the time required to sign grows with the complexity of predicate formula. More precisely, the generation of signature requires a large number of module exponentiations, which commonly grows linearly with the size of the predicate formula [26][29][24]. Although the traditional desktop computers should be able to quite easily handle

such task for typical formula size, this presents a significant challenge for users that manage and view private data on mobile devices where processors are often one to two orders of magnitude slower than their desktop counterparts.

The recently emerged next-generation computing paradigm, named cloud computing, provides the feasibility to reduce the computation overhead at user side by outsourcing the computation of signing to a signing-cloud service provider (S-CSP). Though promising as it is, this paradigm also brings forth new challenge when users intend to outsource ABS on untrusted cloud server. Specifically, by the reason that some private information is involved in the outsourced signing operation, it demands a way to prevent the untrusted S-CSP from learning any private information about signer. Moreover, since the cloud service is typically required to be paid in the commercial setting, it also demands a way for signer to guarantee the accountability on cloud service provider once the signature is not generated correctly.

1.1 Our Contribution

In this paper, aiming at reducing the computational overhead at signer side, we propose two efficient outsourced ABS (OABS) schemes denoted by $\mathcal{OABS}$-I and $\mathcal{OABS}$-II. We employ a hybrid private key by introducing a default attribute for all the users in the system. More precisely, an AND gate is involved to bound two sub-components of the user private key: i) the private key component for user's attributes (denoted as outsourcing key OK in this paper) which is to be utilized by S-CSP to compute the outsourced signature; ii) the private key component for the default attribute which is to be utilized by signer to generate a normal ABS signature from the outsurced signature returned from S-CSP.

- Our first scheme $\mathcal{OABS}$-I is built on Li et al's construction [24]. With the help of S-CSP, the number of exponentiations involving in signing is greatly reduced from $O(d)$ to $O(1)$, where d is the upper bound of threshold value in the predicate.
- Our second OABS scheme $\mathcal{OABS}$-II is based on Herranz et al's construction [21]. The main advantage of $\mathcal{OABS}$-II over the previous one is that the signature is much shorter since they have only three group elements. The number of exponentiations for signing a single message is reduced from $O(d^2)$ to $O(d)$ after outsourcing. Furthermore, the communication overhead at user side in outsourcing phase is only $O(1)$.

Finally, to allow for high efficiency and flexibility, we discuss two extensions on OABS including accountability and outsourced verification. On accountability, we embed a normal signature of S-CSP in each outsourced ABS signature to allow for tracing the dishonest actions of S-CSP. On outsourced verification, we utilize the technique [32] to build an efficient outsourced verifying protocol for OABS.

1.2 Related Work

1.2.1 Attribute-based Signature The first formal definition of ABS was presented by Maji et al. [25]. Their construction supports for predicate descripted by monotone span programs, but the security of the scheme is in the generic group model. Li et al. [24] and Shahandashit et al. [29] proposed the ABS schemes that support threshold predicate in standard model. Nevertheless, both of their schemes require $O(|\Omega^*|)$ exponentiations in signing, where Ω^* is the attribute set in the threshold predicate included in the signature. And, the signature length of the above schemes is $O(|\Omega^*|)$, which also grows linear with the size of attributes in the predicate. Escala et al. [14] proposed a revocable ABS for threshold predicate, which shares a similar efficiency with Li et al.'s work [24] in signing. Recently,

Herranz et al. [21] proposed two threshold predicate ABS schemes with constant size signatures. But they are both inefficient. The first scheme not only invoking another similar algorithm Aggregate in [12], but also requires a large number of heavy computations, and the second one involves $O(d^2)$ exponentiations in signing, where d is the upper bound of the threshold value.

Concerning on more expressive predicate, beyond the pioneering work [25], Maji et al. [26] instantiate an ABS for the case that the predicate is described as monotone span program. Though it is proven secure in the standard model but is much less efficient than the original one [25]. Okamoto and Takashima [27] presented the first fully secure ABS scheme in standard model supporting for non-monotone predicates.

We specify that existing work of ABS requires a large number of exponentiations in signing. The complexity commonly grows linearly with the size of the predicate formule in threshold ABS[6]. Such inefficiency becomes even more serious for ABS with more expressive predicate.

1.2.2 Outsourcing Computation The problem that how to securely outsource different kinds of expensive computations has drew considerable attention from theoretical computer science community. Atallah et al. [3] 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 [2] 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 [4] 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 [1] further studied this problem and gave improved protocols based on the so-called weak secret hiding assumption. Recently, Wang et al. [30] presented efficient mechanisms for secure outsourcing of linear programming computation.

Note that though several above schemes have been introduced to securely outsource expensive computations, they are not suitable for reliving the computational overhead of signature scheme at user side. To achieve this goal, the notion of server-aided signature [23][6] has been proposed. Both [23] and [6] aimed to reduce the local computational cost associated with performing exponentiation. Then, they applied such exponentiation outsourcing techniques to realize server-aided batch signature generation. However, such server-aided signature schemes are oriented to accelerating the speed of exponentiation and are not practical for small number of signatures generation. Actually, at a high level, the server-aided signature method can be viewed as a special mediated cryptography. Boneh et al. [8][7] on mediated RSA and Ding et al. [13] on mediated group signatures are another two examples in this area. Mediated cryptographic protocols share the feature of utilizing a partially trusted online server. However, they were proposed to provide efficient revocation, instead of using the traditional revocation list method. There are also some other related work addressing the computation outsourcing in one-time signature by using the hash-chain technique [5][19]. However, these above mentioned techniques cannot be used in ABS to realize OABS efficiently. Furthermore, the notion of accountability on cloud service provider has never been addressed in traditional server-aided signature schemes, which is another critical issue of outsourcing signature in commercial setting.

Another approach might be to leverage recent generic outsourcing technique or delegating computation [11][16][17][15][18] based on fully homomorphic encryption or interactive proofs systems. However, Gentry [17] has shown that even for weak security parameters on "bootstrapping" operation of the ho-

[6] One exception is Herranz et al.'s work [21], but its cost is expensive as well

momorphic operation, it would take at least 30 seconds on a high performance machine. Therefore, even if the privacy of the inputs can be preserved by utilizing these generic techniques, the computational overhead of this technique is still huge and impractical.

Recently, the outsourcing technique has been applied to ABE to reduce computation overhead at user side [20][33]. In [20], Green et al. considered to outsource the decryption of ABE to eliminate the overhead at user side, while an outsourced ABE with outsourced encryption and decryption was presented in [33]. We point out that the outsourcing technique in [20][33] is to blind user's attribute private key by running a number of exponentiations. But such key blinded operation cannot be straightforwardly applied to reduce the computation overhead for ABS. In this paper, we consider a hybrid policy technique that introduces an additional default attribute appended with each user's attribute set. Actually, our technique provides a feasible way to realize the "piecewise key generation" property recently introduced in [28].

1.3 Organization

This paper is organized as follows. In Section 2, we describe some preliminaries. In Section 3, we present the system model and security requirements of OABS. The first proposed scheme and its security and efficiency analysis are presented in Section 4. We propose the second construction with shorter signatures and its security analysis in Section 5. In Section 6, we discuss some extensions of OABS. Finally, we draw conclusion in Section 7.

2 Preliminary

2.1 Bilinear Map

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 $\mathbb{G}_T$;*

2.2 Complexity Assumption

Definition 2 (CDH Problem). *The Computational Diffie-Hellman (CDH) problem is that, for every probabilistic polynomial time algorithm $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $Pr[\mathcal{A}(1^l, g, g^x, g^y) = g^{xy}] \leq \mathsf{negl}(l)$ for all l, where $x, y \in_R \mathbb{Z}_q$, and g is the generator of a group $\mathbb{G}$ of order q, which is a prime of length approximately l.*

We say that (t, ϵ)-CDH assumption holds in $\mathbb{G}$ if there is no adversary $\mathcal{A}$ that runs within time t and solves CDH problem with probability at least ϵ.

Definition 3 (n-DHE Problem [9]). *The n-DHE (Diffie-Hellman Exponent) problem is that, for every probabilistic polynomial time algorithm $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $Pr[\mathcal{A}(1^l, g, g^{\gamma}, g^{\gamma^2}, \ldots, g^{\gamma^n}, g^{\gamma^{n+2}}, \ldots, g^{\gamma^{2n}}) = g^{\gamma^{n+1}}] \leq \mathsf{negl}(l)$ for all l, where $\gamma \in_R \mathbb{Z}_q$, and g is the generator of a group $\mathbb{G}$ of order q, which is a prime of length approximately l.*

We say that (t, ϵ)-n-DHE assumption holds in $\mathbb{G}$ if there is no adversary $\mathcal{A}$ that runs within time t and solves the n-DHE problem with probability at least ϵ.

2.3 Supported Predicate

Furthermore, we state that the OABS constructions in this paper support for predicates Υ consisting of thresholds gates. Specifically, all predicates $\Upsilon_{k,\Omega^*}(\cdot) :\rightarrow \{0,1\}$ for Ω^* with threshold value k from 1 to d are supported, where d is a system parameter and

$$\Upsilon_{k,\Omega^*}(\Omega) = \begin{cases} 1 & |\Omega \cap \Omega^*| \geq k \\ 0 & \text{otherwise} \end{cases}$$

3 Problem Statement

3.1 Definition

There are three entities involved in our OABS system, namely, the attribute authority, users (include signers and verifiers), and S-CSP. Typically, the signers obtain their private keys from attribute authority, with which they are able to sign messages later for any predicate satisfied by the possessed attributes. Verifiers will be convinced of the fact that whether a signature is from one of the users whose attributes satisfy the signing predicate, but remaining completely ignorant of the identity of the signer. Different from the definition of traditional ABS [24][21], an additional entity S-CSP is introduced. Specifically, S-CSP is to finish the outsourced expensive tasks in signing phase and relieve the computational burden at signer side.

An OABS scheme consists of five algorithms, namely, the setup algorithm Setup, the private key generation algorithm KeyGen, the outsourced signing algorithm $\mathsf{Sign}_{\text{out}}$, the signing algorithm Sign, and the verifying algorithm Verify. In more details, the definition of OABS is described as follows.

Definition 4 (Outsourced Attribute-Based Signature). *An outsourced attribute-based signature scheme $\mathcal{OABS}$ consists of five probabilistic polynomial-time algorithms below.*

- $\mathsf{Setup}(\lambda, \mathcal{U}, d)$: *It takes as input – the security parameter λ, attribute universe $\mathcal{U}$ and an auxiliary information d. It outputs the public key PK and the master key MK.*
- $\mathsf{KeyGen}(MK, \Omega)$: *For each user's private key request on attribute set Ω, the private key generation algorithm takes as input – the master key MK and the attribute set Ω. It outputs the user's private key SK and the outsourcing key OK.*
- $\mathsf{Sign}_{\text{out}}(OK, \Omega, \Upsilon)$: *The outsourced signing algorithm takes as input – the outsourcing key OK, the corressponding attribute set Ω and the predicate Υ. It outputs the partial signature σ_{part}.*
- $\mathsf{Sign}(SK, M, \sigma_{\text{part}}, \Upsilon)$: *The signing algorithm takes as input –the private key SK, the message M, the partial signature σ_{part} and the corresponding predicate Υ. It outputs the signature σ of message M with the predicate Υ.*
- $\mathsf{Verify}(M, \sigma, \Upsilon, PK)$: *The verifying algorithm takes as input – a message M, the signature σ, the predicate Υ and public key PK. It outputs 1 if the original signature is deemed valid and 0 otherwise.*

3.2 Security Requirements

A basic ABS scheme must satisfy the usual property of unforgeability, even against a group of colluding users that put their private keys together. In OABS, since a signing aided entity S-CSP is introduced, we require that the forger, even additionally colluding with S-CSP to obtain the outsourcing keys

of any users, still cannot forge a valid signature with any predicate which his/her attributes do not satisfy. In more details, the formal definition of unforgeability is based on the following game involving a challenger $\mathcal{C}$ and a forger $\mathcal{F}$:

Setup. $\mathcal{C}$ chooses a sufficiently large security parameter λ and runs $\mathsf{Setup}(\lambda,\mathcal{U},d)$. It maintains the master key MK and sends the public key PK to $\mathcal{F}$.

Queries. $\mathcal{C}$ initializes an integer $j=0$ and an empty table L. $\mathcal{F}$ is provided the following oracles.

- *Outsourcing key extraction oracle.* Upon receiving an attribute set Ω, $\mathcal{C}$ sets $j=j+1$, runs $\mathsf{KeyGen}(MK,\Omega)$ and returns OK after adding the new entry (j,Ω,SK,OK) into L.
  Note: the *outsourcing key extraction oracle* can be repeatedly queried with the same input.
- *Private key extraction oracle.* Upon receiving an attribute set Ω and an integer i, $\mathcal{C}$ just returns SK if such an entry exists in L. If no such entry exists, the challenger runs $\mathsf{KeyGen}(MK,\Omega)$ and outputs SK after adding the new entry (i,Ω,SK,OK) into L.
- *Signing oracle.* Upon receiving a message M and a predicate Υ, $\mathcal{C}$ returns a signature σ by running $\mathsf{Sign}_{\text{out}}$ and Sign.

Forgery. Finally, $\mathcal{F}$ outputs a pair (M^*,σ^*) with the predicate Υ^*.

We say that $\mathcal{F}$ wins the game if i) σ^* is a valid signature on M^* with Υ^*; ii) for any queried attribute set $\Omega\in U$, $\Upsilon^*(\Omega)\neq 1$; iii) the pair (M^*,Υ^*) has not been submitted to the *signing oracle*. Accordingly, the advantage $Adv^{\text{EUF}}_{\mathcal{OABS},\mathcal{F}}(\lambda)$ of $\mathcal{F}$ is defined as the probability that it wins the game above.

Definition 5 (Unforgeability). *A forger $\mathcal{F}$ (t,q_O,q_K,q_S,ϵ)-breaks an $\mathcal{OABS}$ if $\mathcal{F}$ runs in time at most t, and makes at most q_O outsourcing key extraction queries, q_K private key extraction queries and q_S signing queries while the advantage $Adv^{\text{UF}}_{\mathcal{OABS},\mathcal{F}}$ is at least ϵ. An OABS scheme is (t,q_O,q_K,q_S,ϵ)-unforgeable, if there exists no polynomial forger that can (t,q_O,q_K,q_S,ϵ)-break it.*

Beyond the adaptive predicate security, we also specify that the unforgeability under a weaker adversary model named selective predicate. An OABS scheme $\mathcal{OABS}$ is unforgeable under selective predicate attack, if there exists no polynomial forger that can win in a modified game, where the challenge predicate Υ^* is submitted before Setup.

To guarantee privacy for the signer, we require that the attribute-signer privacy is preserved as in traditional ABS [24][21]. Specifically, it requires that the signature reveals nothing about the identity or attributes of the signer beyond what is explicitly revealed. Following the intuition, we formulize this definition with a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ below.

Setup. $\mathcal{C}$ chooses a sufficiently large security parameter λ and runs $\mathsf{Setup}(\lambda)$. Finally give the pair (PK,MK) to $\mathcal{A}$.

Query. $\mathcal{C}$ provides the *signing oracle*, *private key extraction oracle* and *outsourcing key extraction oracle* as the above game of unforgeability.

Challenge. $\mathcal{A}$ outputs a tuple $(\Upsilon,\Omega_0,\Omega_1,M)$ with the restriction that $\Upsilon(\Omega_0)=\Upsilon(\Omega_1)=1$. $\mathcal{C}$ picks a bit $b\in_R\{0,1\}$ and runs $\mathsf{KeyGen}(MK,\Omega_b)$ to obtain OK_b and SK_b. Furthermore, $\mathcal{C}$ computes the signature σ^* himself by running $\mathsf{Sign}_{\text{out}}$ and Sign with OK_b and SK_b respectively, and sends σ^* back to $\mathcal{A}$.

Guess. $\mathcal{A}$ outputs a bit $b\in\{0,1\}$ and wins if $b=b'$.

The advantage of $\mathcal{A}$ is measured as the probability $Adv^{\text{Priv}}_{\mathcal{OABS},\mathcal{A}}(\lambda)=\mathsf{Pr}[b=b']-\frac{1}{2}$.

Definition 6 (Attribute-signer Privacy). *An outsourced attribute-based signature scheme $\mathcal{OABS}$ satisfies attribute-signer privacy if there exists no polynomially bounded adversary that has non-negligable advantage against challenger in the game above.*

4 The First Scheme $\mathcal{OABS}$-I

4.1 Proposed Construction

We define the Lagrange coefficient $\Delta_{i,S}$ for $i \in \mathbb{Z}_q$ and a set S of elements in $\mathbb{Z}_q$ as $\Delta_{i,S} = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$. Then, the proposed construction $\mathcal{OABS}$-I, which is based on the traditional ABS in [24], is shown as follows.

- $\mathsf{Setup}(\lambda, \mathcal{U}, d)$: First, redefine the attributes in universe $\mathcal{U}$ as elements in $\mathbb{Z}_q$. For simplicity, we can take the first $|\mathcal{U}|$ elements in $\mathbb{Z}_q$ (i.e. $1, 2, \ldots, |\mathcal{U}| \bmod q$) to be the universe. A $(d-1)$-element dummy attribute set $\hat{\Omega}$ and a unique default attribute θ are also defined in $\mathbb{Z}_q$ as well. Next, select a generator $g \in_R \mathbb{G}$ and an integer $x \in_R \mathbb{Z}_q$, and set $g_1 = g^x$. Pick an element $g_2 \in_R \mathbb{G}$ and compute $Z = e(g_1, g_2)$. Two hash functions H_1, H_2 are defined such that $H_1, H_2 : \{0,1\}^* \to \mathbb{G}$. Finally, output the public key $PK = (g, g_1, g_2, Z, H_1, H_2, d)$ and the master key $MK = x$.
- $\mathsf{KeyGen}(MK, \Omega)$: For each user's private key request on attribute set Ω, select $x_1 \in_R \mathbb{Z}_q$ and set $x_2 = x - x_1$. Next, randomly select a $(d-1)$-degree polynomial $q(\cdot)$ such that $q(0) = x_1$ and compute $d_{i0} = g_2^{q(i)} H_1(i)^{r_i}$ and $d_{i1} = g^{r_i}$ for each $i \in \Omega \cup \hat{\Omega}$, where $r_i \in_R \mathbb{Z}_q$. Furthermore, compute $d_{\theta 0} = g_2^{x_2} H_1(\theta)^{r_\theta}$ and $d_{\theta 1} = g^{r_\theta}$ for $r_\theta \in_R \mathbb{Z}_q$. Finally, output the outsourcing key $OK = (\{d_{i0}, d_{i1}\}_{i \in \Omega \cup \hat{\Omega}})$ and the private key $SK = (\{d_{\theta 0}, d_{\theta 1}\}, OK)$.
- $\mathsf{Sign}_{\mathrm{out}}(OK, \Omega, \Upsilon_{k,\Omega^*})$: Upon receiving a signing request from a user with an outsourcing key $OK = (\{d_{i0}, d_{i1}\}_{i \in \Omega \cup \hat{\Omega}})$, S-CSP generates a partial signature with Υ_{k,Ω^*} as follows.
  1) Select an arbitrary k-element subset Ω' with $\Omega' \subseteq \Omega \cap \Omega^*$. Furthermore, select a dummy attribute set $\hat{\Omega}'$ with $\hat{\Omega}' \subseteq \hat{\Omega}$ and $|\hat{\Omega}'| = d - k$.
  2) Pick $n + d - k$ random values s_i for $i \in \Omega^* \cup \Omega'$ where $n = |\Omega^*|$ and compute $\sigma'_0 = \prod_{i \in \Omega' \cup \hat{\Omega}'} d_{i0}^{\Delta_{i,\Omega' \cup \hat{\Omega}'}(0)} \prod_{i \in \Omega^* \cup \hat{\Omega}'} H_1(i)^{s_i}$. Furthermore, compute σ'_i as
$$\sigma'_i = \begin{cases} d_{i1}^{\Delta_{i,\Omega' \cup \hat{\Omega}'}(0)} g^{s_i} & i \in \Omega' \cup \hat{\Omega}' \\ g^{s_i} & i \in \Omega^* \backslash \Omega' \end{cases}$$
  3) Output the partial signature $\sigma_{\mathrm{part}} = (\sigma'_0, \{\sigma'_i\}_{i \in \Omega^* \cup \hat{\Omega}'})$.
- $\mathsf{Sign}(SK, M, \sigma_{\mathrm{part}}, \Upsilon_{k,\Omega^*})$: Upon receiving the partial signature $\sigma_{\mathrm{part}} = (\sigma'_0, \{\sigma'_i\}_{i \in \Omega^* \cup \hat{\Omega}'})$ from S-CSP, the signer with private key $SK = (\{d_{\theta 0}, d_{\theta 1}\}, OK)$ on the attribute set Ω completes the signing algorithm as follows.
  1) Select two values $s, s_\theta \in_R \mathbb{Z}_q$, and compute $\sigma_0 = d_{\theta 0} \cdot H_1(\theta)^{s_\theta} \cdot \sigma'_0 \cdot H_2(M || \Upsilon_{k,\Omega^*})^s$, $\sigma_\theta = d_{\theta 1} g^{s_\theta}$ and $\sigma_\eta = g^s$.
  2) Output the final signature as $\sigma = (\sigma_0, \sigma_\eta, \{\sigma_i\}_{i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\}})$ where $\sigma_i = \sigma'_i$ for $i \in \Omega^* \cup \hat{\Omega}'$.
- $\mathsf{Verify}(M, \sigma, \Upsilon, PK)$: After receiving the signature $\sigma = (\sigma_0, \sigma_\eta, \{\sigma_i\}_{i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\}})$, the verification is presented by checking whether the following equation holds:
$$\frac{e(g, \sigma_0)}{\prod_{i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\}} [e(\sigma_i, H_1(i))] e(\sigma_\eta, H_2(M || \Upsilon_{k,\Omega^*}))} \stackrel{?}{=} Z$$
  If it holds, output 1 and the signature is accepted. Otherwise, output 0 to denote the signature is not valid.

4.2 Security Analysis

The correctness of verification is justified by the following equation:

$$
\begin{aligned}
&\frac{e(g,\sigma_0)}{\prod_{i\in\Omega^*\cup\hat{\Omega}'\cup\{\theta\}}[e(\sigma_i,H_1(i))]e(\sigma_\eta,H_2(M||\Upsilon_{k,\Omega^*}))}\\
&=\frac{e(g,d_{\theta 0})\prod_{i\in\Omega'\cup\hat{\Omega}'}[e(g,d_{i0})^{\Delta_{i,\Omega'\cup\hat{\Omega}'}(0)}]\prod_{i\in\Omega^*\cup\hat{\Omega}'}[e(g,H_1(i))^{s_i}]}{e(d_{\theta 1},H_1(\theta))\prod_{i\in\Omega'\cup\hat{\Omega}'}[e(d_{i1}^{\Delta_{i,\Omega'\cup\hat{\Omega}'}(0)}g^{s_i},H_1(i))]\prod_{i\in\Omega^*\setminus\Omega'}[e(g^{s_i},H_1(i))]}\\
&=e(g,g_2)^{x_2}e(g,g_2)^{\sum_{i\in\Omega'\cup\hat{\Omega}'}q(i)\Delta_{i,\Omega'\cup\hat{\Omega}'}(0)}\\
&=e(g,g_2)^{x_1+x_2}\\
&=Z
\end{aligned}
$$

A critical requirement in delegated computation is to be resistant to cheating by delegatee. Our scheme achieves this by employing the hash function on the concatenation between message and predicate. Therefore, even if S-CSP cheats by modifying signing attributes, the forged signature still cannot pass verification. Formally, we provide the theorem below to show the unforgeability of the proposed scheme.

Theorem 1. *The proposed OABS scheme $\mathcal{OABS}$-I is unforgeable under selective predicate attack in the random oracle model if the CDH assumption holds in $\mathbb{G}$.*

Proof. Please refer to Appendix A.

Beyond the unforgeability, the $\mathcal{OABS}$-I also achieves attribute-signer privacy, which is described below.

Theorem 2. *The proposed OABS scheme $\mathcal{OABS}$-I achieves attribute-signer privacy.*

Proof. Please refer to Appendix B.

4.3 Efficiency Analysis

To be a true OABS, the proposed scheme must satisfy outsourceable requirement. Specifically, the running time of Sign must be less than directly computing the signature itself. In original ABS construction [24], it requires $\frac{3}{2}(|\Omega^*|+d-k)$ single-based exponentiations to generate the signature. However, since multiple exponentiations have been delivered to S-CSP, in $\mathcal{OABS}$-I, the signing algorithm Sign simply requires 3 single-based exponentiations, which is independent of the attribute set Ω^* to be signed.

We also specify that our technique in $\mathcal{OABS}$-I allows S-CSP to perform delegated signing by employing an AND gate at private key for each user. Therefore, to generate an outsourcing key, attribute authority has to compute $|\Omega|+d+1$ exponentiations in $\mathbb{G}$, which is linear with the size of request attribute set Ω. Fortunately, in practical, the generation is allowed to be performed once for all. After obtaining private key and outsourcing key from authority, user is able to (delegated) sign any message with it. Such amortized computation cost of generating the outsourcing key is rather low cost. Moreover, we consider a scenario that user has limited computation and storage ability. In this case, the outsourcing key can be firstly generated by authority and sent to S-CSP. Therefore, user only needs to store a small-sized component $(d_{\theta 0}, d_{\theta 1})$ locally but still maintaining signing capability.

Furthermore, concerning on the additional communication complexity for $\mathcal{OABS}$-I, in outsourced signing phase, the signer has to send a signature generation request to S-CSP and receives $|\Omega|+d-k+2$ elements of $\mathbb{G}$ from it. In general, an element in $\mathbb{G}$ is set to be 160-bit long for 2^{80} security. Thus, the additional data transferred between S-CSP and signer is tens of KBs at most, which can be processed efficiently.

5 The Second Scheme $\mathcal{OABS}$-II with Shorter Signature

5.1 Proposed Construction

Our second OABS scheme $\mathcal{OABS}$-II is based on Herranz et al.'s construction [21]. The main advantage over the previous one is that signatures are much shorter, since they have only three group elements. Actually, in Herranz et al.'s original construction [21], the computational cost of signing algorithm is much heavy for users with limited computational ability. Nevertheless, after utilizing our outsourcing technique, the complexity in signing phase is reduced to constant exponentiations. We provide the $\mathcal{OABS}$-II as follows.

- $\mathsf{Setup}(\lambda, \mathcal{U}, d)$: Similar to the Setup algorithm in $\mathcal{OABS}$-I, define the universe $\mathcal{U}$, the $(d-1)$-element dummy attribute set $\hat{\Omega}$ and the default attribute θ in $\mathbb{Z}_q$. Choose $h, h_i \in_R \mathbb{G}$ for $i = 1, 2, \ldots, 2d+1$ and $u_0, u_1, \cdots, u_w \in_R \mathbb{G}$ as Waters' hash function [31], where w is size limits predefined. For simplicity, the Waters hash function will be denoted by $H : \{0,1\}^* \to \mathbb{G}$ in our construction. Finally, after picking $x, x_0 \in_R \mathbb{Z}_q$, it outputs the public key $PK = (e(g,g)^x, h = g^{x_0}, \{h_i\}_{i=1}^{2d+1}, H)$ and stores the master key $MK = g^x$.
- $\mathsf{KeyGen}(MK, \Omega)$: For each user's private key request on attribute set Ω, select $x_1 \in_R \mathbb{Z}_q$ and define an implicit value $x_2 = x - x_1$. Randomly select an implicit $(d-1)$-degree polynomial $q(z) = x_1 + \gamma_1 z + \cdots + \gamma_{d-1} z^{d-1}$ by choosing random $\gamma_i \in \mathbb{Z}_q$. Next, for each $i \in \Omega \cup \hat{\Omega}$, compute $d_{i0} = g^{q(i)} h^{r_i}, d_{i1} = g^{r_i}$ and $\{f_{ij} = (h_1^{-i^j} h_{j+1})^{r_i}\}_{j=1}^{2d}$ for $r_i \in_R \mathbb{Z}_q$. Then, compute $d_{\theta 0} = g^{x_2} h^{r_\theta}, d_{\theta 1} = g^{r_\theta}$ and $\{f_{\theta j} = (h_1^{-\theta^j} h_{j+1})^{r_\theta}\}_{j=1}^{2d}$ for $r_\theta \in_R \mathbb{Z}_q$. Finally, return the outsourcing key $OK = (\{d_{i0}, d_{i1}, \{f_{ij}\}_{j=1}^{2d}\}_{i \in \Omega \cup \hat{\Omega}})$ and user's private key $SK = (d_{\theta 0}, d_{\theta 1}, \{f_{\theta j}\}_{j=1}^{2d}, OK)$.
- $\mathsf{Sign}_{\mathrm{out}}(OK, \Omega, \Upsilon_{k,\Omega^*})$: Upon receiving a request of generating a partial signature of Υ_{k,Ω^*} with the outsourcing key $OK = (\{d_{i0}, d_{i1}, \{f_{ij}\}_{j=1}^{2d}\}_{i \in \Omega \cup \hat{\Omega}})$, S-CSP proceeds as follows:
  1) Select an arbitrary k-element subset Ω' with $\Omega' \subseteq \Omega \cap \Omega^*$. Furthermore, select a dummy attribute set $\hat{\Omega}'$ with $\hat{\Omega}' \subseteq \hat{\Omega}$ and $|\hat{\Omega}'| = d - k$. Therefore, $\{c_j\}_{j=1}^{2d+1}$ are able to be defined as the coefficients of the polynomial below.

$$p(x) = \prod_{i \in \Omega' \cup \hat{\Omega}' \cup \{\theta\}} (x - i) = \sum_{j=1}^{2d+1} c_j x^{j-1} \tag{1}$$

  where $c_{d+3}, c_{d+4}, \ldots, c_{2d+1}$ are all set to 0.
  2) Pick $r \in_R \mathbb{Z}_q$ and compute

$$\sigma_0' = \prod_{i \in \Omega' \cup \hat{\Omega}'} [d_{i0} \prod_{j=1}^{2d} f_{ij}^{c_{j+1}}]^{\Delta_{i,\Omega' \cup \hat{\Omega}'}(0)} [h \prod_{i=1}^{2d+1} h_i^{c_i}]^r, \sigma_1' = \prod_{i \in \Omega' \cup \hat{\Omega}'} d_{i1}^{\Delta_{i,\Omega' \cup \hat{\Omega}'}(0)} g^r$$

  3) Finally, output the partial signature $\sigma_{\mathrm{part}} = (\sigma_0', \sigma_1', \Omega', \hat{\Omega}')$.

– $\mathsf{Sign}(SK, M, \sigma_{\text{part}}, \Upsilon_{k,\Omega^*})$: Suppose $M||\Upsilon_{k,\Omega^*} \in \{0,1\}^w$, then after receiving the partial signature $\sigma_{\text{part}} = (\sigma_0', \sigma_1', \Omega', \hat{\Omega}')$ from S-CSP, the signer with private key $SK = (d_{\theta 0}, d_{\theta 1}, \{f_{\theta j}\}_{j=1}^{2d}, OK)$ runs the complete signing algorithm as follows:
  1) Similar to the outsourcing signing algorithm, the same coefficients $\{c_j\}_{j=1}^{2d+1}$ is computed from the polynomial $p(z)$ as per (1) with Ω' and $\hat{\Omega}'$.
  1) Pick $s \in_R \mathbb{Z}_q$ and compute $\sigma_0 = \sigma_0'[d_{\theta 0} \prod_{i=1}^{2d} f_{\theta i}^{c_{i+1}}] H(M||\Upsilon_{k,\Omega^*})^s, \sigma_1 = \sigma_1' d_{\theta 1}$ and $\sigma_2 = g^s$.
  2) Finally, output the signature $\sigma = (\sigma_0, \sigma_1, \sigma_2)$.
– $\mathsf{Verify}(\sigma, PK)$: After receiving $\sigma = (\sigma_0, \sigma_1, \sigma_2)$, the verifier computes the coefficients $\{c_j\}_{j=1}^{2d+1}$ from the polynomial $p(z)$ as per (1). Then, the verification is to compute

$$\frac{e(\sigma_0, g)}{e(\sigma_1, h\prod_{i=1}^{2d+1} h_i^{c_i}) e(\sigma_2, H(M||\Upsilon_{k,\Omega^*}))} \stackrel{?}{=} e(g,g)^x$$

If the above equation holds, the signature σ on M with Υ_{k,Ω^*} is valid and accepted; otherwise, it is rejected.

Obviously, in Herranz el al's original construction [21], the signing phase requires $O(d^2)$ (nearly $(\frac{2d+1}{2}+1)(d+1)$) single-based exponentiations, but utilizing our technique, such cost is reduced to $1+\frac{2d+1}{2}$ exponentiations.

5.2 Security Analysis

Before justifying the correctness of the verification, we specify that for each $i \in \Omega' \cup \hat{\Omega}' \cup \{\theta\}$, the equation below

$$\prod_{j=1}^{2d} f_{ij}^{c_{j+1}} = (h_1^{-\sum_{j=1}^{2d} c_{j+1} i^j} \prod_{j=1}^{2d} h_{j+1}^{c_{j+1}})^{r_i} = (\prod_{j=1}^{2d+1} h_j^{c_j})^{r_i}$$

holds based on the fact that $p(i) = 0$. Therefore, the correctness is examined as follows.

$$\begin{aligned}
&\frac{e(\sigma_0, g)}{e(\sigma_1, h\prod_{i=1}^{2d+1} h_i^{c_i}) e(\sigma_2, H(M||\Upsilon_{k,\Omega^*}))} \\
&= \frac{e(g,g)^{x_1+x_2} e(g, h\prod_{j=1}^{2d+1} h_j^{c_j})^{r_\theta + r + \sum_{i\in\Omega'\cup\hat{\Omega}'} r_i \Delta_{i,\Omega'\cup\hat{\Omega}'}(0)} e(g, H(M||\Upsilon_{k,\Omega^*}))^s}{e(g, h\prod_{j=1}^{2d+1} h_j^{c_j})^{\sum_{i\in\Omega'\cup\hat{\Omega}'} r_i \Delta_{i,\Omega'\cup\hat{\Omega}'}(0) + r_\theta + r} e(g, H(M||\Upsilon_{k,\Omega^*}))^s} \\
&= e(g,g)^x
\end{aligned}$$

Theorem 3. *The proposed scheme $\mathcal{OABS}$-II is secure in the sense of selective predicate under the $(2d+1)$-DHE assumption, where d is the upper bound of the threshold value.*

Proof. The parameters assignment is similar to the proof of Theorem 1 in this work except that the $(2d+1)$-DHE assumption is utilized here. Specifically, simulator $\mathcal{S}$ is to compute $g^{\gamma^{2d+2}}$ from $(g, g^\gamma, \ldots, g^{\gamma^{2d+1}}, g^{\gamma^{2d+3}}, \ldots, g^{\gamma^{4d+2}})$ where $\gamma \in_R \mathbb{Z}_q$. Therefore, after obtaining the challenge predicate Υ_{k,Ω^*}, $\mathcal{S}$ randomly picks $\hat{\Omega}' \subseteq \hat{\Omega}$ and computes the polynomial in the same way as equation (1) to obtain the coefficients $\{y_i\}_{i=1}^{2d+1}$. Accordingly, $\mathcal{S}$ is to pick $\alpha, \beta, \alpha_i \in \mathbb{Z}_q$ for $i = 1, 2, \ldots, 2d+1$, and assign parameters as follows: $\{h_i = g^{\gamma^i} g^{\alpha_i}\}_{i=1}^{2d+1}$, $h = g^\alpha g^{-\sum_{i=1}^{2d+1} \gamma^i y_i}$ and $e(g,g)^x = e(g^\gamma, g^{\gamma^{2d+1}})^\beta$. The master key is simulated as $\beta\gamma^{2d+2}$.

Since we build an AND gate on private key as in the first scheme, a random split is to be simulated like to the proof of Theorem 1. The rest of simulation is identical to that of Theorem 3 in [21].

Theorem 4. *The proposed scheme enjoys attribute-signer privacy.*

Proof. By the same reason elaborated in the proof of Theorem 2, we are only able to show the signer privacy when $k = |\Omega^*|$. In this case, the signature is in the form of $\sigma_0^* = \prod_{i\in\Omega'\cup\hat{\Omega}'\cup\{\theta\}} [d_{i0}(\prod_{j=1}^{2d+1} h_j^{c_j})^{r_i}]^{\Delta_{i,\Omega'\cup\hat{\Omega}'}(0)} [h\prod_{i=1}^{2d+1} h_i^{c_i}]^r H(M^*||\Upsilon_{k,\Omega^*})^s, \sigma_1^* = \prod_{i\in\Omega'\cup\hat{\Omega}'\cup\{\theta\}} d_{i1}^{\Delta_{i,\Omega'\cup\hat{\Omega}'}(0)} g^r$ and $\sigma_2^* = g^s$ where we denote $\Delta_{\theta,\Omega'\cup\hat{\Omega}'}(0) = 1$. Obviously, the challenge signature is able to be generated from any attribute set $\Omega \supseteq \Omega^*$ by just selecting individual random values r_i and s. Therefore, attribute-signer privacy is achieved.

6 Extensions

6.1 OABS with S-CSP Accountability

In a practical OABS system, the S-CSP cannot be fully trusted. Thus, how to guarantee the correctness of the result from the S-CSP is critical. The trivial technique is to verify the outsourced signature with the normal verification algorithm in ABS. However, the computational cost of such verification is a large number of bilinear pairings and exponentiations [24][21], which grows linearly with the number of attributes in the predicate. As a result, the computation overhead at signer side will not be reduced at all even after outsourcing. To tackle this challenge, an additional security requirement of accountability is introduced into OABS. Specifically, it requires that any dishonest action and result returned by S-CSP can be detected and traced. Such dishonest behaves could be prevented to a great extent if S-CSP will be punished if detected. The accountable OABS can be utilized to applications such as fine-grained private access control or distributed access control for ad hoc networks etc. When an access request (that is, an OABS) fails, user is allowed to launch another request of access by sending another correct signature. Its definition is provided as follows.

Definition 7 (Accountability). *An outsourced attribute-based signature scheme $\mathcal{OABS}$ satisfies accountability if S-CSP involved in is accountable for its dishonest actions, that is, it can be detected and traced if S-CSP has not generate the outsourcing signatures correctly.*

We show how to achieve S-CSP accountability based on $\mathcal{OABS}$-I which is defined in Section 4.1. Actually, the technique can be easily applied to $\mathcal{OABS}$-II as well. In the accountable OABS scheme, we require that S-CSP must have a key pair for generating signature with ordinary digital signature scheme $\mathcal{SIG} = (\mathsf{KS}, \mathsf{SIG}, \mathsf{VER})$, where $\mathsf{KS}, \mathsf{SIG}$, and VER denote setup algorithm, signing algorithm and verifying algorithm, respectively. The accountable OABS construction is identical to $\mathcal{OABS}$-I in Section 4.1, except that the $\mathsf{Sign}_{\text{out}}$ algorithm[7]. Furthermore, an additional algorithm Trace is also involved to realize the accountability of S-CSP. We present the intuition of the accountable OABS as follows: After complete the outsourcing ABS signature $\sigma' = (\sigma'_0, \{\sigma'_i\}_{i\in\Omega^*\cup\hat{\Omega}'})$ on behalf of user, the S-CSP is required to embed its own signature as a tag (denoted by tag) on each concatenation of outsourcing key OK and the corresponding generated signature σ'. The original signer only needs to verify the ordinary signature signed by S-CSP, instead of verifying the signature σ' of OABS. Then, if the ABS signature fails in verification later, the signer is able to utilize such an "evidence" to trace and confirm whether S-CSP produced the invalid signature or not. We only describe the Trace algorithm for simplicity.

[7] A minor difference exists in KeyGen algorithm as well. Specifically, g^{x_1} is included in OK in the accountable OABS

– Trace: When a verification on ABS signature σ is aborted, the original signer provides σ', OK and tag to an arbitration agency. The arbitration agency firstly checks the correctness of the tag by running the verifying algorithm of $\mathcal{SIG}$. If the verification holds, arbitration agency further checks the correctness of $\sigma' = (\sigma'_0, \{\sigma'_i\}_{i \in \Omega^* \cup \hat{\Omega}'})$ through verifying the equation of

$$\frac{e(g, \sigma'_0)}{\prod_{i \in \Omega^* \cup \hat{\Omega}'}[e(\sigma'_i, H_1(i))])} \stackrel{?}{=} e(g^{x_1}, g)$$

with respect to g^{x_1}. If the above equation does not hold, it further verifies the correctness of OK by running $\mathsf{Sign}_{\text{out}}$ algorithm to get an outsourced signature $\tilde{\sigma}$ and verifying $\tilde{\sigma}$ using the method described in the above equation. If holds, it means that S-CSP misbehaves. Otherwise, the arbitration agency can deduce that the original signer misbehaves.

6.2 OABS with Outsourced Verification

Though the above technique can only guarantee the correctness of outsourced computation with accountability, it cannot check the correctness and detect the misbehaves of S-CSP on spot. To solve this problem, we provide another solution to verify the outsourced signature with low computational cost by introducing another independent entity called verifying-cloud service provider (V-CSP). We also introduce an assumption that the S-CSP and V-CSP will not collude. Actually, such assumption has also appeared in [22][10] to deal with the problem of secure outsourcing computation as well. Accordingly, an outsourced verification protocol, including the transformation algorithm for outsourced verification Transf, the outsourced verifying algorithm $\mathsf{Verify}_{\text{out}}$ and the verifying algorithm Verify, replaces the original verifying algorithm in OABS definition.

– $\mathsf{Transf}(\sigma)$: The transformation algorithm for outsourced verification takes as input – the signature σ. It outputs the transformed signature σ_{trans} and the transformation factor t.
– $\mathsf{Verify}_{\text{out}}(\sigma_{\text{trans}}, M, \Upsilon)$: The outsourced verifying algorithm takes as input – the transformed signature σ_{trans} and the corresponding message M and prediate Υ. It outputs Λ which is used by the verifier to perform verification.
– $\mathsf{Verify}(PK, \Lambda, t)$: The verifying algorithm takes as input – the outsourced verification information Λ and the transformation factor t. It outputs 1 if the original signature is deemed valid and 0 otherwise.

We specify that the outsourced verification will introduce a new notion of soundness into our scheme. Informally, it requires that V-CSP should be able to utilize $\mathsf{Verify}_{\text{out}}$ to convince the verifier that an invalid signature is valid in a negligible probability (refer to [32] for formal definition).

Then, we show how to achieve outsourced verification based on $\mathcal{OABS}$-I. Actually, the technique can be easily extended to $\mathcal{OABS}$-II as well. After receiving an outsourced signature from S-CSP, the signer computes σ and runs the outsourced verification algorithms to verify its correctness, which are described as follows.

– $\mathsf{Transf}(\sigma)$: The signer picks a transformation factor $t \in_R \mathbb{Z}_q$ and computes $\widetilde{\sigma}_0 = g^t \sigma_0$. Finally, it outputs the transformed signature $\sigma_{\text{trans}} = (\widetilde{\sigma}_0, \widetilde{\sigma}_\eta, \{\widetilde{\sigma}_i\}_{i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\}})$ where $\widetilde{\sigma}_\eta = \sigma_\eta$ and $\widetilde{\sigma}_i = \sigma_i$ for $i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\}$.

– $\mathsf{Verify}_{\mathrm{out}}(\sigma_{\mathrm{trans}}, M, \Upsilon_{k,\Omega^*})$: The outsourced verifying algorithm is to compute and return

$$\Lambda = \frac{e(g, \widetilde{\sigma}_0)}{\prod_{i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\}} [e(\widetilde{\sigma}_i, H_1(i))] e(\widetilde{\sigma}_\eta, H_2(M || \Upsilon_{k,\Omega^*}))}$$

– $\mathsf{Verify}(PK, \Lambda, t)$: The verification is presented by checking $\Lambda \stackrel{?}{=} e(g,g)^t Z$. If it holds, output 1 which indicates the signature is indeed from some user with k attributes among Ω^*. Otherwise, output 0.

The proposed OABS scheme with outsourced verification reduces the computation load at signer side through delivering computation to V-CSP but only lefting two exponentiations locally. Because the outsourcing verification method is the same as [32], the security can be also guaranteed based on the assumption that the S-CSP does not collude with the V-CSP. In another word, V-CSP cannot cheat to let an invalid signature pass the verification algorithm because σ_0 is blinded and not available to V-CSP [32].

7 Conclusion

Aiming at eliminating the most computational overhead at signer, we introduce outsourcing computation into ABS and propose two efficient OABS schemes. With the help of C-CSP, our first scheme requires only three exponentiations for signing a single message at signer side. Our second scheme is built on Herranz et al.'s work [21], but reduces the number of exponentiations from $O(d^2)$ to $O(d)$, where d is the upper bound of the threshold value. Furthermore, the communication overhead between the signer and S-CSP is very low which requires only three group elements. We discuss the extensions for OABS including accountability and outsourced verification and show practical solutions as well.

References

1. Atallah, M.J., Frikken, K.B.: Securely outsourcing linear algebra computations. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. pp. 48–59. ASIACCS'10, ACM, New York, NY, USA (2010)
2. Atallah, M.J., Li, J.: Secure outsourcing of sequence comparisons. International Journal of Information Security 4, 277–287 (2005)
3. Atallah, M.J., Pantazopoulos, K., Rice, J.R., Spafford, E.E.: Secure outsourcing of scientific computations. In: Zelkowitz, M.V. (ed.) Trends in Software Engineering, Advances in Computers, vol. 54, pp. 215 – 272. Elsevier (2002)
4. Benjamin, D., Atallah, M.J.: Private and cheating-free outsourcing of algebraic computations. In: Proceedings of the 2008 Sixth Annual Conference on Privacy, Security and Trust. pp. 240–245. PST'08, IEEE Computer Society, Washington, DC, USA (2008)
5. Bicakci, K., Baykal, N.: Saots: A new efficient server assisted signature scheme for pervasive computing. In: Hutter, D., Mller, G., Stephan, W., Ullmann, M. (eds.) Security in Pervasive Computing, Lecture Notes in Computer Science, vol. 2802, pp. 187–200. Springer Berlin / Heidelberg (2004)
6. Bicakci, K., Baykal, N.: Server assisted signatures revisited. In: Okamoto, T. (ed.) Topics in Cryptology – CT-RSA 2004, Lecture Notes in Computer Science, vol. 2964, pp. 1991–1992. Springer Berlin / Heidelberg (2004)
7. Boneh, D., Ding, X., Tsudik, G.: Fine-grained control of security capabilities. ACM Trans. Internet Technol. 4(1), 60–82 (2004)

8. Boneh, D., Ding, X., Tsudik, G., Wong, C.M.: A method for fast revocation of public key certificates and security capabilities. In: USENIX Security Symposium (2001)
9. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup, V. (ed.) Advances in Cryptology – CRYPTO, Lecture Notes in Computer Science, vol. 3621, pp. 258–275. Springer Berlin / Heidelberg (2005)
10. Canetti, R., Riva, B., Rothblum, G.N.: Two 1-round protocols for delegation of computation. Cryptology ePrint Archive, Report 2011/518 (2011)
11. Chung, K.M., Kalai, Y., Liu, F.H., Raz, R.: Memory delegation. In: Rogaway, P. (ed.) Advances in Cryptology – CRYPTO 2011, Lecture Notes in Computer Science, vol. 6841, pp. 151–168. Springer Berlin / Heidelberg (2011)
12. Delerable, C., Pointcheval, D.: Dynamic threshold public-key encryption. In: Wagner, D. (ed.) Advances in Cryptology – CRYPTO 2008, Lecture Notes in Computer Science, vol. 5157, pp. 317–334. Springer Berlin / Heidelberg (2008)
13. Ding, X., Tsudik, G., Xu, S.: Leak-free group signatures with immediate revocation. In: Proceedings of the 24th International Conference on Distributed Computing Systems. pp. 608–615. ICDCS'04, IEEE Computer Society, Washington, DC, USA (2004)
14. Escala, A., Herranz, J., Morillo, P.: Revocable attribute-based signatures with adaptive security in the standard model. In: Nitaj, A., Pointcheval, D. (eds.) Progress in Cryptology – AFRICACRYPT 2011, Lecture Notes in Computer Science, vol. 6737, pp. 224–241. Springer Berlin / Heidelberg (2011)
15. Gennaro, R., Gentry, C., Parno, B.: Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In: Rabin, T. (ed.) Advances in Cryptology – CRYPTO 2010, Lecture Notes in Computer Science, vol. 6223, pp. 465–482. Springer Berlin / Heidelberg (2010)
16. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st annual ACM symposium on Theory of computing. pp. 169–178. STOC'09, ACM, New York, NY, USA (2009)
17. Gentry, C., Halevi, S.: Implementing gentry's fully-homomorphic encryption scheme. In: Paterson, K. (ed.) Advances in Cryptology – EUROCRYPT 2011, Lecture Notes in Computer Science, vol. 6632, pp. 129–148. Springer Berlin / Heidelberg (2011)
18. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: interactive proofs for muggles. In: Proceedings of the 40th annual ACM symposium on Theory of computing. pp. 113–122. STOC '08, ACM, New York, NY, USA (2008)
19. Goyal, V.: More efficient server assisted one time signatures. Cryptology ePrint Archive, Report 2004/135 (2004)
20. Green, M., Hohenberger, S., Waters, B.: Outsourcing the decryption of abe ciphertexts. In: Proceedings of the 20th USENIX conference on Security. pp. 34–34. SEC'11, USENIX Association, Berkeley, CA, USA (2011)
21. Herranz, J., Laguillaumie, F., Libert, B., Rfols, C.: Short attribute-based signatures for threshold predicates. In: Dunkelman, O. (ed.) Topics in Cryptology – CT-RSA 2012, Lecture Notes in Computer Science, vol. 7178, pp. 51–67. Springer Berlin / Heidelberg (2012)
22. Hohenberger, S., Lysyanskaya, A.: How to securely outsource cryptographic computations. In: Proceedings of the Second international conference on Theory of Cryptography. pp. 264–282. TCC'05, Springer-Verlag, Berlin, Heidelberg (2005)
23. Jakobsson, M., Wetzel, S.: Secure server-aided signature generation. In: Kim, K. (ed.) Public Key Cryptography, Lecture Notes in Computer Science, vol. 1992, pp. 383–401. Springer Berlin / Heidelberg (2001)
24. Li, J., Au, M.H., Susilo, W., Xie, D., Ren, K.: Attribute-based signature and its applications. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. pp. 60–69. ASIACCS'10, ACM, New York, NY, USA (2010)
25. Maji, H., Prabhakaran, M., Rosulek, M.: Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. Cryptology ePrint Archive, Report 2008/328 (2008)
26. Maji, H., Prabhakaran, M., Rosulek, M.: Attribute-based signatures. In: Kiayias, A. (ed.) Topics in Cryptology – CT-RSA 2011, Lecture Notes in Computer Science, vol. 6558, pp. 376–392. Springer Berlin / Heidelberg (2011)

27. Okamoto, T., Takashima, K.: Efficient attribute-based signatures for non-monotone predicates in the standard model. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) Public Key Cryptography – PKC 2011, Lecture Notes in Computer Science, vol. 6571, pp. 35–52. Springer Berlin / Heidelberg (2011)
28. Sahai, A., Seyalioglu, H., Waters, B.: Dynamic credentials and ciphertext delegation for attribute-based encryption. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology – CRYPTO 2012, Lecture Notes in Computer Science, vol. 7417, pp. 199–217. Springer Berlin / Heidelberg (2012)
29. Shahandashti, S., Safavi-Naini, R.: Threshold attribute-based signatures and their application to anonymous credential systems. In: Preneel, B. (ed.) Progress in Cryptology – AFRICACRYPT 2009, Lecture Notes in Computer Science, vol. 5580, pp. 198–216. Springer Berlin / Heidelberg (2009)
30. Wang, C., Ren, K., Wang, J.: Secure and practical outsourcing of linear programming in cloud computing. In: IEEE International Conference on Computer Communications (INFOCOM), pp. 820–828. IEEE Computer Society (2011)
31. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005, Lecture Notes in Computer Science, vol. 3494, pp. 114–127. Springer Berlin / Heidelberg (2005)
32. Wu, W., Mu, Y., Susilo, W., Huang, X.: Provably secure server-aided verification signatures. Computers Mathematics with Applications 61(7), 1705–1723 (2011)
33. Zhou, Z., Huang, D.: Efficient and secure data storage operations for mobile cloud computing. Cryptology ePrint Archive, Report 2011/185 (2011)

Appendix A: Proof of Theorem 1

Proof. Assume that a forger $\mathcal{F}$ has a non-negligible advantage ϵ in attacking $\mathcal{OABS}$-I in the sense of selective predicate, we attempt to build a simulator $\mathcal{S}$ that utilizes $\mathcal{F}$ as a sub-algorithm to solve the CDH problem with a non-negligible probability ϵ'.

Suppose the forger $\mathcal{F}$ makes at most $q_{H_1}, q_{H_2}, q_O, q_K$ and q_S queries to hash functions H_1, H_2, outsourcing key extraction oracle, private key extraction oracle and signing oracle respectively. Initially, the simulator $\mathcal{S}$ is given an instance of CDH problem $g, X = g^x$ and $Y = g^y$ where $x, y \in_R \mathbb{Z}_q$, and asked to compute g^{xy}. We proceed the simulation as follows.

Init. $\mathcal{S}$ runs $\mathcal{F}$ and receives a challenge predicate Υ_{k,Ω^*}.

Setup. Denote the dummy attribute set as $\hat{\Omega}$. $\mathcal{S}$ selects a subset $\hat{\Omega}' \subseteq \hat{\Omega}$ with $|\hat{\Omega}'| = d - k$ and publishes $g_1 = X$ and $g_2 = Y$ to $\mathcal{F}$.

Query. $\mathcal{S}$ initializes an integer $j = 0$, an empty table L and an empty set U. $\mathcal{F}$ is allowed to issue queries as follows.

- *H_1-query.* $\mathcal{S}$ maintains a list $\mathcal{L}_1$ to store the answers to the hash oracle H_1. Then, upon receiving the query i, $\mathcal{S}$ performs a check on the list $\mathcal{L}_1$. If an entry for the query is found, the same answer will be returned. Otherwise, $\mathcal{S}$ computes

$$H_1(i) = \begin{cases} g^{\beta_i} & i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\} \\ g_1^{\alpha_i} g^{\beta_i} & i \notin \Omega^* \cup \hat{\Omega}' \cup \{\theta\} \end{cases}$$

  where $\alpha_i, \beta_i \in_R \mathbb{Z}_q$, and returns $H_1(i)$ after adding the tuple $(i, H_1(i))$ into $\mathcal{L}_1$.
- *H_2-query.* $\mathcal{S}$ picks a value $\delta \in_R \{1, 2, \ldots, q_{H_2}\}$ and maintains a list $\mathcal{L}_2$ to store the answers to the hash oracle H_2. Then, upon receiving the j-th time query $M_j || \Upsilon_{k_j,\Omega_j}$ for $1 \leq j \leq q_{H_2}$ and

$1 \leq k_j \leq d$. $\mathcal{S}$ performs a check on the list $\mathcal{L}_2$. If an entry for the query is found, the same answer will be returned. Otherwise, $\mathcal{S}$ computes

$$H_2(M_j||\Upsilon_{k_j,\Omega_j}) = \begin{cases} g_1^{\alpha'_j} g^{\beta'_j} & j \neq \delta \\ g^{\beta'_\delta} & j = \delta \end{cases}$$

where $\alpha'_i, \beta'_i \in_R \mathbb{Z}_q$, and returns $H_2(M_j||\Upsilon_{k_j,\Omega_j})$ after adding the tuple $(M_j||\Upsilon_{k_j,\Omega_j}, H_2(M_j||\Upsilon_{k_j,\Omega_j}))$ into $\mathcal{L}_2$.

– *Outsourcing key query.* Upon receiving an outsourcing key request on attribute set Ω, $\mathcal{S}$ sets $j = j + 1$ and attempts to perform simulation as follows.

- If $|\Omega \cap \Omega^*| < k$, $\mathcal{S}$ picks $x_2 \in_R \mathbb{Z}_q$ and defines three sets Γ, Γ' and S with $\Gamma = (\Omega \cap \Omega^*) \cup \hat{\Omega}', |\Gamma'| = d-1, \Gamma \subseteq \Gamma' \subseteq \Omega \cup \hat{\Omega}'$ and $S = \Gamma' \cup \{0\}$. Then, for $i \in \Gamma'$, $(d_{i0}, d_{i1}) = (g_2^{\tau_i} H_1(i)^{r_i}, g^{r_i})$ where $\tau_i, r_i \in_R \mathbb{Z}_q$. For $i \in (\Omega \cup \hat{\Omega}) \backslash \Gamma'$, let $r_i = -\frac{y\Delta_{i,S}(0)}{\alpha_i} + r'_i$ where $r'_i \in_R \mathbb{Z}_q$ and simulate $(d_{i0}, d_{i1}) = (g_2^{\sum_{j\in\Gamma'} \tau_j \Delta_{j,S}(i) - (x_2 + \frac{\beta_i}{\alpha_i})\Delta_{0,S}(i)} g_1^{\alpha_i r'_i} g^{\beta_i r'_i}, g_2^{-\frac{\Delta_{0,S}(i)}{\alpha_i}} g^{r'_i})$. The intuition behind these assignments is that $q(i) = \sum_{j\in\Gamma'} q(j)\Delta_{j,S}(i) + q(0)\Delta_{0,S}(i)$ and we are implicitly selecting a random $(d-1)$-degree polynomial $q(\cdot)$ by choosing its values for the $d-1$ points randomly as $q(j) = \tau_j$, in addition to having $q(0) = x - x_2$.
- Otherwise (i.e. $|\Omega \cap \Omega^*| \geq k$), $\mathcal{S}$ picks $x_1 \in_R \mathbb{Z}_q$ and randomly selects $(d-1)$-degree polynomial $q(\cdot)$ with $q(0) = x_1$. Then, for each attribute $i \in \Omega \cup \hat{\Omega}$, $(d_{i0}, d_{i1}) = (g_2^{q(i)} H_1(i)^{r_i}, g^{r_i})$ where $r_i \in_R \mathbb{Z}_q$.

Finally, after adding/updating the entry $(j, \Omega, \cdot, OK)$ in L where $OK = (\{d_{i0}, d_{i1}\}_{i\in\Omega\cup\hat{\Omega}})$[8], $\mathcal{S}$ returns OK.

– *Private key query.* Upon receiving an attribute set Ω with $\Upsilon_{k,\Omega^*}(\Omega) \neq 1$ and an integer i, $\mathcal{S}$ firstly sets $U = U \cup \{\Omega\}$. Next, it checks whether the entry (i, Ω, SK, OK) exists in L, if so return SK; otherwise $\mathcal{S}$ picks $x_2 \in_R \mathbb{Z}_q$ and presents the simulation similar to the first case (i.e. $|\Omega \cap \Omega^*| < k$) in *outsourcing key query* to obtain OK, and computes $(d_{\theta 0}, d_{\theta 1}) = (g_2^{x_2} H_1(\theta)^{r_\theta}, g^{r_\theta})$ where $r_\theta \in_R \mathbb{Z}_q$. Finally, after adding/updating the entry (i, Ω, SK, OK) in L where $SK = (\{d_{\theta 0}, d_{\theta 1}\}, OK)$, $\mathcal{S}$ returns SK.

– *Signing query.* Upon receiving the signing request on $(M, \Upsilon_{k',\Omega})$, if $|\Omega \cap \Omega^*| < k$, $\mathcal{S}$ is able to generate the simulated outsourcing key and private key as in the *outsourcing key query* and *private key query* respectively. Then, the signature can be simulated normally. Otherwise (i.e. $|\Omega \cap \Omega^*| \geq k$), if $H_2(M||\Upsilon_{k',\Omega}) = g^{\beta'_\delta}$, the query is aborted. Otherwise, assume $H_2(M||\Upsilon_{k',\Omega}) = g_1^{\alpha'_j} g^{\beta'_j}$. Then, $\mathcal{S}$ randomly selects a k'-element subset $\Omega' \subseteq \Omega$ as well as $(d-k')$-element subset $\hat{\Omega}''$ from $\hat{\Omega}$ and attempts to simulate the signature on M with $\Upsilon_{k',\Omega}$ by letting $s = -\frac{y}{\alpha'_j} + s'$ and outputs the

[8] We note that the value x_2 must be stored by $\mathcal{S}$ to response private/outsourcing key request on Ω ($|\Omega \cap \Omega^*| < k$) later.

simulated the signature as follows:

$$\sigma_0 = H_1(\theta)^{r_\theta + s_\theta} \prod_{i \in \Omega' \cup \hat{\Omega}''} [H_1(i)^{r_i \Delta_{i,\Omega' \cup \hat{\Omega}''}(0)}] \prod_{i \in \Omega \cup \hat{\Omega}''} [H_1(i)^{s_i}] g_1^{\alpha'_j s'} g_2^{-\frac{\beta'_j}{\alpha'_j}} g^{\beta'_j s'}$$

$$\sigma_i = \begin{cases} g^{r_\theta + s_\theta} & i = \theta \\ g^{r_i \Delta_{i,\Omega' \cup \hat{\Omega}'}(0) + s_i} & i \in \Omega' \cup \hat{\Omega}'' \\ g^{s_i} & i \in \Omega \backslash \Omega' \end{cases}$$

$$\sigma_\eta = g_2^{-\frac{1}{\alpha'_j}} g^{s'}$$

where $s', r_i, s_j \in_R \mathbb{Z}_q$ for $i \in \Omega' \cup \hat{\Omega}'' \cup \{\theta\}$ and $j \in \Omega \cup \hat{\Omega}'' \cup \{\theta\}$. Finally, $\mathcal{S}$ returns $(\sigma_0, \sigma_\eta, \{\sigma_i\}_{i \in \Omega \cup \hat{\Omega}'' \cup \{\theta\}})$ back to $\mathcal{A}$. We specify that σ_0 is correctly simulated based on the following observation

$$\begin{aligned} \sigma_0 &= H_1(\theta)^{r_\theta + s_\theta} \prod_{i \in \Omega' \cup \hat{\Omega}''} [H_1(i)^{r_i \Delta_{i,\Omega' \cup \hat{\Omega}''}(0)}] \prod_{i \in \Omega \cup \hat{\Omega}''} [H_1(i)^{s_i}] g_1^{\alpha'_j s'} g_2^{-\frac{\beta'_j}{\alpha'_j}} g^{\beta'_j s'} \\ &= g_2^x H_1(\theta)^{r_\theta + s_\theta} \prod_{i \in \Omega' \cup \hat{\Omega}''} [H_1(i)^{r_i \Delta_{i,\Omega' \cup \hat{\Omega}''}(0)}] \prod_{i \in \Omega \cup \hat{\Omega}''} [H_1(i)^{s_i}] H_2(M || \Upsilon_{k,\Omega})^s \\ &= d_{\theta 0} H_1(\theta)^{s_\theta} \prod_{i \in \Omega' \cup \hat{\Omega}''} [d_{i0}^{\Delta_{i,\Omega' \cup \hat{\Omega}''}(0)}] \prod_{i \in \Omega \cup \hat{\Omega}''} [H_1(i)^{s_i}] H_2(M || \Upsilon_{k,\Omega})^s \end{aligned}$$

Forgery. $\mathcal{F}$ outputs a forged signature σ^* on message M^* with Υ_{k,Ω^*}. Suppose that the associated dummy attribute set is $\hat{\Omega}^*$. If $\hat{\Omega}^* \neq \hat{\Omega}'$ or $H_2(M||\Upsilon_{k,\Omega^*}) \neq g^{\beta'_\delta}$, $\mathcal{S}$ will abort. Otherwise, the signature submitted satisfies the verification which means that

$$\begin{aligned} &\frac{e(g, \sigma_0^*)}{\prod_{i \in \Omega^* \cup \hat{\Omega}^* \cup \{\theta\}} [e(\sigma_i^*, H_1(i))] e(\sigma_\eta^*, H_2(M^* || \Upsilon_{k,\Omega^*}))} \\ = &\frac{e(g, \sigma_0^*)}{\prod_{i \in \Omega^* \cup \hat{\Omega}^* \cup \{\theta\}} [e(\sigma_i^*, g^{\beta_i})] e(\sigma_\eta^*, g^{\beta'_\delta})} \\ = &\frac{e(g, \sigma_0^*)}{\prod_{i \in \Omega^* \cup \hat{\Omega}^* \cup \{\theta\}} [e((\sigma_i^*)^{\beta_i}, g)] e((\sigma_\eta^*)^{\beta'_\delta}, g)} = e(g_1, g_2) \end{aligned}$$

Then, $\mathcal{S}$ can compute $g^{xy} = \frac{\sigma_0^*}{\prod_{i \in \Omega^* \cup \hat{\Omega}^* \cup \{\theta\}} [(\sigma_i^*)^{\beta_i}] (\sigma_\eta^*)^{\beta'_\delta}}$.

In a successful security game, the forged signature on message M^* is submitted with the restriction $H_2(M^*||\Upsilon_{k,\Omega^*}) = g^{\beta'_\delta}$ and $\hat{\Omega}^* = \hat{\Omega}'$. We use t_{EXP} to denote the time cost for single-based exponentiation operation in $\mathbb{G}$ and assume that one multi-based exponentiation which multiplies up to 2 single-based exponentiations takes roughly the same time as a single-based exponentiation. Suppose $\mathcal{F}$ successfully attacks $\mathcal{OABS}$-I with time t, then it can be easily derived that we can build another algorithm solving the CDH problem with time t', where $t' \approx t + (q_{H_1} + q_{H_2} + 2|\overline{\Omega}_O| q_O + 2(|\overline{\Omega}_K| + 1) q_K + \frac{3}{2}(|\overline{\Omega}| + d - \overline{k}) q_S) t_{\text{EXP}}$ [9] , and $|\overline{\Omega}_O|, |\overline{\Omega}_K|$ are the average number of attributes in queried set

[9] In this work, the time cost is always considered by utilizing optimized algorithm to achieve 2 times speed-up in computing multi-exponentiation compared to individual exponentiation.

in *outsourcing key query* and *private key query* respectively, $\overline{\Omega}$ and $\overline{k}$ are the average parameter for the predicate queried in *signing query*. We can also get the probability of solving CDH problem as $\epsilon' = \frac{\epsilon}{q_{H_2}\binom{d-k}{d-1}}$, where $\frac{1}{\binom{d-k}{d-1}}$ is the probability for the correct guess of $(d-k)$-element subset $\hat{\Omega}^*$ from $\hat{\Omega}$.

Appendix B: Proof of Theorem 2

Proof. In order to prove the signer privacy for OABS, we show that a valid signature for the threshold policy Υ_{k,Ω^*}, which was produced using some attribute set Ω_0^* with $\Upsilon_{k,\Omega^*}(\Omega_0^*) = 1$, can also be produced by any other set satisfying such predicate. Furthermore, the signature will not reveal which attribute subset is really used to sign the message because any attribute subset with k elements among the given attributes can generate the signature. Therefore, we only need show the proof of the signer privacy in case $k = n$ where $n = |\Omega^*|$.

Firstly, the challenger runs setup algorithm to publish PK and $MK = x$ to adversary. Adversary is allowed to query *signing oracle*, *private key extraction oracle* and *outsourcing key oracle*. Since the master key is given to adversary, simulation of these oracles is trivial and omitted here. Then, the adversary outputs $(\Upsilon_{k,\Omega^*}, \Omega_0^*, \Omega_1^*, M^*)$ with the restriction $\Omega_0^* \supseteq \Omega^*$ and $\Omega_1^* \supseteq \Omega^*$, and asks the challenger to generate a signature on message M^* with respect to Υ_{k,Ω^*} from either Ω_0^* or Ω_1^*.

The adversary cannot collude with the cloud server to break the signer privacy as defined in the security model. In another word, the challenger will generate the challenge signature by himself. More precisely, the challenger chooses a bit $b \in_R \{0,1\}$ as well as a $(d-k)$-element subset $\hat{\Omega}' \subseteq \hat{\Omega}$, and outputs the challenge signature. Since $\Omega_0^* \cap \Omega^* = \Omega_1^* \cap \Omega^* = \Omega^*$, the challenge signature σ^* is in the form of $(g_2^x \prod_{i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\}} H_1(i)^{r_i \Delta_{i,\Omega^* \cup \hat{\Omega}'}(0)+s_i} H_2(M^* || \upsilon_{k,\Omega^*})^s, g^s, \{g^{r_i \Delta_{i,\Omega^* \cup \hat{\Omega}'}(0)+s_i}\}_{i \in \Omega^* \cup \hat{\Omega}' \cup \{\theta\}})$ where we just let $\Delta_{\theta,\Omega^* \cup \hat{\Omega}'}(0) = 1$ for simplicity. Based on the Lagrange interpolation function, it is obvious that σ^* could be generated from either $SK_{\Omega_0^*}$ or $SK_{\Omega_1^*}$. Therefore, we have proved that the signature generated from the private key $SK_{\Omega_b^*}$ with attribute set Ω_b^*, it could also be generated by $SK_{\Omega_{1-b}^*}$ with Ω_{1-b}^*. Thus the signer privacy for OABS is obtained.