Secure Outsourced Attribute-based Signatures

Xiaofeng Chen, Jin Li, Xinyi Huang, Jingwei Li, Yang Xiang, and Duncan S. Wong

Abstract—Attribute-based signature (ABS) 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-constrained devices (such as mobile devices or RFID tags) to perform such heavy computations independently. Aiming at tackling the challenge above, we first propose and formalize a new paradigm called Outsourced ABS, i.e., OABS, in which the computational overhead at user side is greatly reduced through outsourcing intensive computations to an untrusted cloud service provider (S-CSP). Furthermore, we apply this novel paradigm to existing ABS schemes to reduce the complexity. As a result, we present two concrete OABS 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 as well.

Index Terms—Outsource-secure algorithm, Cloud computing, Attribute-based signature.

1 INTRODUCTION

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.

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. 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.

Recently, cloud computing is getting widespread attentions in the scientific community. This new computing paradigm enables convenient and on-demand network access to a centralized pool of configurable computing resources that can be rapidly deployed with great efficiency and minimal management overhead. Cloud computing has plenty of benefits for the real-world applications such as on-demand self-service, ubiquitous network access, location-independent resource pooling, rapid resource elasticity, usage-based pricing, and outsourcing etc. In the outsourcing computation paradigm, the users with resource-constraint devices can outsource the heavy computation workloads into the cloud server and enjoy the unlimited computing resources in a pay-per-use manner. Though promising as it is, this paradigm also brings forth some new challenges when users intend to outsource ABS on an untrusted cloud server. Specifically, since some private information is involved in the outsourcing signing operation, it demands a way to prevent the untrusted S-CSP from learning any private information of the signer.
Moreover, since the cloud service is typically required to be paid in the commercial setting, it also demands a way for the 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 \( O_{ABS-I} \) and \( O_{ABS-II} \). 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 outsourced signature returned from S-CSP. The security of the both schemes can be guaranteed based on the observation that outsourcing key is restricted by the user. In this way, S-CSP can only sign the specified message on behalf of user.

- Our first scheme \( O_{ABS-I} \) is built on Li et al.’s construction [27]. 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 \( O_{ABS-II} \) is based on Herranz et al.’s construction [25]. The main advantage of \( O_{ABS-II} \) over the previous one is that the signature is much shorter since it has 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 [38] 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. [31]. Their construction supports for predicate described by monotone span programs, but the security of the scheme is in the generic group model. Li et al. [27] and Shahandashit et al. [35] proposed the ABS schemes that support threshold predicate in standard model. Nevertheless, both of their schemes require \( O(|\Omega|^*) \) exponentiations in signing, where \( |\Omega|^* \) 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. [17] proposed a revocable ABS for threshold predicate, which shares a similar efficiency with Li et al.’s work [27] in signing. Recently, Herranz et al. [25] proposed two threshold predicate ABS schemes with constant size signatures. But they are both inefficient. The first scheme not only invokes another similar algorithm Aggregate in [15], 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 [31], Maji et al. [30] 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 [31]. Okamoto and Takashima [33] 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 formula in threshold ABS [31]. 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. [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 led 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 [3] further studied this problem and gave improved protocols based on the so-called weak secret hiding assumption. Recently, Wang et al. [37] presented efficient mechanisms for secure outsourcing of linear programming computation.

In the cryptographic community, Chaum and Pedersen [11] firstly introduced the notion of wallets with

1. One exception is Herranz et al.’s work [25]. However, its computational cost is still expensive as well.
observers, a piece of secure hardware installed on the client’s computer to perform some expensive computations. Hohenberger and Lysyanskaya [24] proposed the first outsource-secure algorithm for modular exponentiations based on the approaches of pre-computation [9], [32] and server-aided computation [8], [10]. Recently, Chen et al. [13] proposed more efficient outsource-secure algorithms for (simultaneously) modular exponentiation in the two untrusted program model.

The notion of server-aided signature [10],[26],[28],[29] has been proposed in order to reduce the local computational cost associated with performing exponentiation. 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. [6],[7] on mediated RSA and Ding et al. [16] 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 [23]. 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 [14],[18],[19],[20],[21] based on fully homomorphic encryption or interactive proofs systems. However, Gentry [19] has shown that even for weak security parameters on “bootstrapping” operation of the homomorphic 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 [22],[31]. In [22], 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 [41]. We point out that the outsourcing technique in [22],[41] 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 [34].

1.3 Organization

This paper is organized as follows. In Section 2, we present the system model and security requirements of OABS. The first proposed scheme and its security and efficiency analysis are presented in Section 3. We propose the second construction with shorter signatures and its security analysis in Section 4. In section 5, we provide a thorough experimental evaluation of the proposed OABS schemes. In Section 6, we discuss an extension of OABS. Finally, we draw conclusion in Section 7.

2 Security Model and Definitions

2.1 Formal 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 [25],[27], 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.

Definition 1 (Outsourced Attribute-Based Signature): An outsourced attribute-based signature scheme OABS consists of five probabilistic polynomial-time algorithms below.

- Setup(λ,U,d): It takes as input – a security parameter λ, an attribute universe U and an auxiliary information d. It outputs the public key PK and the master key MK. The master key MK must be kept secret.

- KeyGen(MK,Ω): The key generation algorithm is run by the attribute authority. 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. The private key SK is sent to the requested user via a secure channel, while the outsourcing key OK is sent to S-CSP via the public channel.

- Signout(OK,Ω,T): The outsourced signing algorithm, which is run by the S-CSP, takes as input – the outsourcing key OK, the corresponding
attribute set $\Omega$ and the predicate $\Upsilon$. It outputs the partial signature $\sigma_{\text{part}}$.

- **Sign($SK, M, \sigma_{\text{part}}, \Upsilon$):** The signing algorithm, which is run by the signer, takes as input – the private key $SK$, the message $M$, the partial signature $\sigma_{\text{part}}$ generated by the S-CSP and the corresponding predicate $\Upsilon$. It outputs the signature $\sigma$ of message $M$ with the predicate $\Upsilon$.

- **Verify($M, \sigma, \Upsilon, PK$):** The verifying algorithm takes as input – a message $M$, the signature $\sigma$, the predicate $\Upsilon$ and public key $PK$. It outputs 1 if the original signature is deemed valid and 0 otherwise.

Furthermore, we state that the OABS constructions in this paper support for predicates $\Upsilon$ consisting of thresholds gates. Specifically, all predicates $C$ based on the following game involving a challenger $A$.

### Queries.

- **Outsourcing key extraction oracle.** Upon receiving an attribute set $\Omega$, $C$ sets $j = j + 1$, runs $\text{KeyGen}(MK, \Omega)$ and returns $OK$ after adding the new entry $(j, \Omega, SK, OK)$ into $L$.

### 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 “honest-but-curious” 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 $C$ and a forger $\mathcal{F}$:

#### Setup

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

#### Queries

- **Outsourcing key extraction oracle.** Upon receiving an attribute set $\Omega$ and an integer $i$, $C$ just returns $SK$ if such an entry exists in $L$. If no such entry exists, the challenger runs $\text{KeyGen}(MK, \Omega)$ and outputs $SK$ after adding the new entry $(i, \Omega, SK, OK)$ into $L$.
- **Signing oracle.** Upon receiving a message $M$ and a predicate $\Upsilon$, $C$ returns a signature $\sigma$ by running $\text{Sign_{out}}$ and $\text{Sign}$.

### Forgery

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

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

#### Definition 2 (Unforgeability)

A forger $\mathcal{F}$ $(t, q_O, q_K, q_S, \epsilon)$-breaks an 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 $\text{Adv}_{\text{OABS}, \mathcal{F}}$ is at least $\epsilon$. An OABS scheme is $(t, q_O, q_K, q_S, \epsilon)$-unforgeable, if there exists no polynomial forger that can $(t, q_O, q_K, q_S, \epsilon)$-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 $\Upsilon^*$ is submitted before Setup.

To guarantee privacy for the signer, we require that the attribute-signer privacy is preserved as in traditional ABS [25][27]. 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 formalize this definition with a game between a challenger $C$ and an adversary $A$ below.

#### Setup

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

#### Query

$C$ provides the signing oracle, private key extraction oracle and outsourcing key extraction oracle as the above game of unforgeability.

#### Challenge

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

#### Guess

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

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

#### Definition 3 (Attribute-signer Privacy)

An outsourced attribute-based signature scheme $\mathcal{OABS}$ satisfies attribute-signer privacy if for any polynomially bounded adversary, it wins in the game above with a negligible advantage against the challenger, i.e., $\text{Adv}_{\text{Priv}}(\lambda) \leq \text{negl}(\lambda)$. 

1045-9219 (c) 2013 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
3 THE FIRST CONSTRUCTION OABS-I

3.1 The Proposed Scheme

Let $G, G_T$ be two cyclic multiplicative groups of prime order $q$. Let $e: G \times G \rightarrow G_T$ be a bilinear pairing. Let $\Delta_{i,s} = \prod_{j \in S \setminus \{i\}} \frac{1}{x_j}$ be the Lagrange coefficient for $i \in \mathbb{Z}_q$ and a set $S$ of elements in $\mathbb{Z}_q$.

The proposed scheme $OABS-I$, which is based on the traditional ABS in [27], is shown as follows.

- **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., $1, 2, \ldots, |\mathcal{U}| \mod q$) to be the universe. A $(d - 1)$-element dummy attribute set $\Omega$ and a unique default attribute $\theta$ are also defined in $\mathbb{Z}_q$ as well. Next, select a generator $g \in_R G$ and an integer $x \in_R \mathbb{Z}_q$, and set $g_1 = g^x$. Pick an element $g_2 \in_R G$ and compute $Z = e(g_1, g_2)$. Two hash functions $H_1, H_2$ are defined so that $H_1, H_2: \{0, 1\}^* \rightarrow G$. Finally, output the public key $PK = (g, g_1, g_2, Z, H_1, H_2, d)$ and the master key $MK = x$.

- **KeyGen($MK, \Omega$):** For each user's private key request on attribute set $\Omega$, 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 $d_{00} = g_2^{q(0)}H_1(i)^{r_1}$ and $d_{11} = g^{r_1}$ for each $i \in \Omega \cup \Omega'$. Where $r_1 \in_R \mathbb{Z}_q$. Furthermore, compute $d_{00} = g^{q(0)}H_1(i)^{r_0}$ and $d_{01} = g^{r_0}$ for $r_0 \in_R \mathbb{Z}_q$. Finally, output the outsourcing key $OK = \{(d_{00}, d_{11})\}_{i \in \Omega \cup \Omega'}$ and the private key $SK = \{(d_{00}, d_{01})\}$.

- **Signout($OK, \Omega, Y_{k, \Omega'}$):** Upon receiving a signing request from a user with an outsourcing key $OK = \{(d_{00}, d_{11})\}_{i \in \Omega \cup \Omega'}$, S-CSP generates a partial signature with $Y_{k, \Omega'}$ as follows:
  1. Select an arbitrary $k$-element subset $\Omega'$ with $\Omega' \subseteq \Omega \cap \Omega'$. Furthermore, select a dummy attribute set $\Omega''$ with $\Omega'' \subseteq \Omega$ and $|\Omega''| = d - k$.
  2. Pick $n + d - k$ random values $s_i$ for $i \in \Omega' \cup \Omega'$ where $n = |\Omega'|$ and compute $\sigma_0' = \prod_{i \in \Omega' \cup \Omega'} d_{i1}^{\Delta_i, \Omega'' \cup \Omega'}(g^{s_i})$. Furthermore, compute $\sigma_1' = \prod_{i \in \Omega' \cup \Omega'} d_{i1}^{\Delta_i, \Omega'' \cup \Omega'}(g^{s_i})$.
  3. Output the partial signature $\sigma_{\text{part}} = (\sigma_0', (\sigma_1')_{i \in \Omega' \cup \Omega'})$.

- **Sign($SK, M, \sigma_{\text{part}}, Y_{k, \Omega'}$):** Upon receiving the partial signature $\sigma_{\text{part}} = (\sigma_0', (\sigma_1')_{i \in \Omega' \cup \Omega'})$ from S-CSP, the signer with private key $SK = \{(d_{00}, d_{01})\}$ on the attribute set $\Omega$ completes the signing algorithm as follows:
  1. Select two values $s, s_{\theta} \in_R \mathbb{Z}_q, \mathbb{Z}_q$, and compute $\sigma_0 = d_{00} \cdot H_1(i)^{\theta} \cdot \sigma_0' \cdot H_2(M|Y_{k, \Omega'}^*)$, $\sigma_\theta = d_{01}g^{s_\theta}$ and $\eta = g^\gamma$.

2) Output the final signature as $\sigma = (\sigma_0, \sigma_\theta, (\sigma_1')_{i \in \Omega^* \cup \Omega'^*})$, where $\sigma_i = \sigma_i'$ for $i \in \Omega^* \cup \Omega'^*$.

- **Verify($M, \sigma, Y, PK$):** After receiving the signature $\sigma = (\sigma_0, \sigma_\theta, (\sigma_1')_{i \in \Omega^* \cup \Omega'^*})$, the verification is presented by checking whether the following equation holds:

$$\prod_{i \in \Omega^* \cup \Omega'^*} e(\sigma_i, H_1(i))e(\sigma_\theta, H_2(M|Y_{k, \Omega'}^*)) = Z$$

If and only if it holds, output 1 and accept the signature.

3.2 Security Analysis

The correctness is trivial by the following equation:

$$e(g, g_0)\prod_{i \in \Omega^* \cup \Omega'^*} e(\sigma_i, H_1(i))e(\sigma_\theta, H_2(M|Y_{k, \Omega'}^*)) = e(g, g_2)^{x_1x_2}$$

$$Z$$

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. In the following, we first introduce some problems in $G$.

Definition 4 (CDH Problem): The Computational Diffie-Hellman (CDH) problem is that, for every probabilistic polynomial time algorithm $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that $\mathcal{A}(A^1, g, g^a, g^b) = g^{ab}$ for all $l$, where $A, g \in \mathbb{Z}_q$, and $g$ is the generator of a group $G$ of order $q$, which is a prime of length approximately $l$.

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

Definition 5 (n-DHE Problem): The n-DHE (Diffie-Hellman Exponent) problem [8] is that, for every probabilistic polynomial time algorithm $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that $\mathcal{A}(A^1, g, g^a, g^{a^2}, \ldots, g^{a^n}, g^{a^{n+1}}, \ldots, g^{a^{2^n}}) = g^{\gamma(n+1)}$ for all $l$, where $\gamma \in \mathbb{Z}_q$, and $g$ is the generator of a group $G$ of order $q$, which is a prime of length approximately $l$.

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

The following theorems show that our proposed scheme satisfies the desired security properties:

Theorem 3.1: The proposed OABS scheme $OABS-I$ is unforgeable under selective predicate attack in the
random oracle model if the CDH assumption holds in \( \mathbb{G} \).

Proof: Please refer to the proof in Appendix A. \( \Box \)

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

Theorem 3.2: The proposed OABS scheme \( \text{OABS-I} \) achieves attribute-signer privacy.

Proof: Please refer to the proof in Appendix B. \( \Box \)

### 3.3 Efficiency Analysis

In this section, we present the efficiency analysis of the proposed scheme. Actually, the running time of Sign is much less than that of directly computing the signature itself. In original ABS construction \([27]\), it requires \( \frac{3}{4}(|\Omega| + d - k) \) single-based exponentiations to generate the signature. However, since multiple exponentiations have been delivered to S-CSP, in \( \text{OABS-I} \), the signing algorithm Sign simply requires 3 single-based exponentiations, which is independent of the attribute set \( \Omega^* \) to be signed.

We also specify that our technique in \( \text{OABS-I} \) 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 \( \Omega \). 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 \((d_{00}, d_{01})\) locally but still maintaining signing capability.

Furthermore, concerning on the communication complexity for \( \text{OABS-I} \), 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 2^90 security. Thus, the additional data transferred between S-CSP and signer is tens of KBs at most, which can be processed efficiently.

### 4 THE SECOND CONSTRUCTION \( \text{OABS-II} \)

#### 4.1 The Proposed Scheme

Our second OABS scheme \( \text{OABS-II} \) is based on Herranz et al.’s construction \([25]\). The main advantage over the first one is that signature length is much shorter (the signature only consists of three group elements). Actually, in Herranz et al.’s original construction \([25]\), 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 \( \text{OABS-II} \) as follows.

- **Setup(\( \lambda, U, d \)):** Similar to the Setup algorithm in \( \text{OABS-I} \), define the universe \( U \), the \((d - 1)\)-element dummy attribute set \( \Omega \) and the default attribute \( \theta \) in \( \mathbb{Z}_q \). Choose \( h, h_i \in \mathbb{G} \) for \( i = 1, 2, \ldots, d + 1 \) and \( u_0, u_1, \cdots, u_w \in \mathbb{G} \) as Waters’ hash function \([30]\), where \( w \) is size limits predefined. For simplicity, the Waters hash function will be denoted by \( H : \{0,1\}^* \rightarrow \mathbb{G} \) in our construction. Finally, after picking \( x, x_0 \in \mathbb{Z}_q^* \), it outputs the public key \( PK = (e(g, g)^x, g = g^{x_0}, \{h_i\}_{i=1}^{2d+1}, H) \) and stores the master key \( MK = x \).
- **KeyGen(\( MK, \Omega \)):** For each user’s private key request on attribute set \( \Omega \), select \( x_1 \in \mathbb{Z}_q^* \) 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 \( i \in \Omega \cup \Omega \), compute \( d_{i0} = g^{\gamma(i)h_i}, d_{i1} = g^{x^2} \) and \( \{f_{ij}\}_{j=1}^{2d} = (h_1^{-\gamma(i)h_{i+1}})^{x^2} \) for \( r_{ij} \in \mathbb{Z}_q \). Finally, return the outsourcing key \( OK = \{(d_{00}, d_{01}, \{f_{ij}\}_{j=1}^{2d})\}_{i \in \Omega \cup \Omega} \) and user’s private key \( SK = (d_{00}, d_{01}, \{f_{ij}\}_{j=1}^{2d}, \text{OK}) \).
- **Sign out(\( OK, \Omega, Y_k, \Omega^* \)):** Upon receiving a request of generating a partial signature of \( Y_k, \Omega^* \) with the outsourcing key \( OK = \{(d_{00}, d_{01}, \{f_{ij}\}_{j=1}^{2d})\}_{i \in \Omega \cup \Omega} \), S-CSP proceeds as follows:
  1. Select an arbitrary \( k \)-element subset \( \Omega' \) with \( \Omega' \subseteq \Omega \cap \Omega^* \). Furthermore, select a dummy attribute set \( \Omega' \) with \( \Omega' \subseteq \Omega \) and \(|\Omega'| = d - k \). Therefore, \( \{c_{ij}\}_{j=1}^{2d+1} \) are able to be defined as the coefficients of the polynomial below:

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

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

\[
\sigma_0' = \prod_{i \in \Omega \cup \Omega'} [d_{00} \prod_{j=1}^{2d} f_{ij}^{c_{ij}+1}] h_1^{c_{ij}+1} \prod_{i=1}^{2d+1} h_i^{c_{ij}+1} \quad (2)
\]

and

\[
\sigma_1' = \prod_{i \in \Omega \cup \Omega'} \gamma_i^{c_{ij}+1} \quad (3)
\]

- **Sign(\( SK, M, \sigma_{\text{part}}, Y_k, \Omega^* \)):** Suppose \( M || Y_k, \Omega^* \in \{0,1\}^* \), then after receiving the partial signature \( \sigma_{\text{part}} = (\sigma_0', \sigma_1', \Omega', \Omega') \) from S-CSP, the signer with
private key $SK = (d_{00}, d_{01}, \{f_{0j}\}_{j=1}^{2d+1}, OK)$ runs the complete signing algorithm as follows:

1) Similar to the outsourcing signing algorithm, the same polynomials $\{c_i\}_{i=1}^{2d+1}$ is computed from the polynomial $p(z)$ as per (1) with $\Omega'$ and $\Omega$.

2) Pick $s \in R \subseteq Z_q$ and compute $\sigma_0 = \sigma_0^2 d_{00} \prod_{i=1}^{2d+1} f_{0i}^{c_{i+1}} |H(M)||Y_{k,\Omega'})^s$, $\sigma_1 = \sigma_1^d d_{01}$ and $\sigma_2 = g^r$.

3) Finally, output the signature $\sigma = (\sigma_0, \sigma_1, \sigma_2)$.

\text{Verify}(\sigma, PK) : After receiving $\sigma = (\sigma_0, \sigma_1, \sigma_2)$, the verifier computes the coefficients $\{c_i\}_{i=1}^{2d+1}$ from the polynomial $p(z)$ as per (1). Then, the verification is to compute

$$e(\sigma_0, g) e(\sigma_1, h \prod_{i=1}^{2d+1} h_i^{c_{i+1}}) e(\sigma_2, H(M||Y_{k,\Omega})) \equiv e(g, g)^x$$

If the above equation holds, the signature $\sigma$ on $M$ with $Y_{k,\Omega'}$ is valid and accepted; otherwise, it is rejected.

Obviously, in Herranz el al’s original construction [25], the signing phase requires $O(d^2)$ single-based exponentiations, but utilizing our technique, such cost is reduced to $O(d)$ exponentiations.

4.2 Security Analysis

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

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

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

$$e(\sigma_0, g) e(\sigma_1, h \prod_{i=1}^{2d+1} h_i^{c_{i+1}}) e(\sigma_2, H(M||Y_{k,\Omega})) = e(g, g)^x$$

**Theorem 4.1:** The proposed scheme $O_{ABS}$-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:** Please refer to the proof in Appendix C.

**Theorem 4.2:** The proposed scheme enjoys attribute-signer privacy.

**Proof:** Please refer to the proof in Appendix D.

5 Performance Evaluation

In this section, we provide a thorough experimental evaluation of the proposed outsourcing algorithms and cryptographic schemes. Our experiment is simulated on a LINUX machine with Intel Core 2 processors running at 2.40 GHz and 2G memory. Throughout this experiment, to precisely evaluate the computation complexity at both signer and S-CSP sides, we make this LINUX machine simulate the both entities. In this case, it is easy to distinguish the overhead at the delegator (i.e., the signer) and worker (i.e., the S-CSP) from time cost.

5.1 Experimental Results for $O_{ABS}$-I

In Fig. 1(a), we show the efficiency comparison for the setup phase between $O_{ABS}$-I and $ABS$-I. In this evaluation, we fix the threshold $d = 8$ and 16 respectively, and vary the number of attributes in the universe from 60 to 100. Theoretically, the proposed scheme $O_{ABS}$-I requires a little more time than the original one $ABS$-I. More specifically, compared with the original scheme $ABS$-I, our construction requires an additional initialization for a default attribute $\theta$ for facilitating outsourcing. However, we have to point out that the additional time is negligible in the case of a large universe. Though the figure, we can deduce that the schemes $O_{ABS}$-I and $ABS$-I have the same computational complexity in the Setup phase (note that the two curves almost overlap with each other).

Fig. 1(b) illustrates the efficiency comparison in the KeyGen phase between two schemes. It is not surprising to see that $O_{ABS}$-I also requires (very) little more time than $ABS$-I. This is because the attribute authority has to additionally issue private keys for the new introduced default attribute $\theta$, involving nearly two exponentiations. Similarly, the additional time is negligible in the case of a large universe. Therefore, the two schemes have almost the same computational complexity in the KeyGen phase.

The performance of signing phase in both $O_{ABS}$-I and $ABS$-I is shown in Fig. 1(c). Note that in this experiment, we assume that signer has been assigned with the attribute private key for all the attributes in the universe $\mathcal{U}$ of size 100, and randomly pick $|\Omega'|$ attributes from $\mathcal{U}$ to prove that he/she has at least $d$ attributes of $\Omega'$ (i.e., the signing predicate is $Y_{d,\Omega'}$), where $d$ is the threshold value. The efficiency in Fig. 1(c) is evaluated in several cases for $|\Omega'| = 20, 40, 60$ and $d = 8, 16$, respectively. It is clear that $O_{ABS}$-I is overall slightly expensive regarding to computation and communication complexity because we consider the issue of secure outsourcing. Specifically, compared with $ABS$-I, our scheme supports private delegation, which sacrifices slight efficiency (e.g. some computations involving the default attribute $\theta$) for preserving the privacy of signer and his/her signing key. However, concerning on the local computation performed by the signer, our scheme $O_{ABS}$-I achieves much nearly constant performance compared with the linear increasing efficiency of $ABS$-I. This advantage allows our scheme to be applied for the resource-constraint devices to complete the task of ABS.

**TABLE 1** shows the time cost for the verifying phase in $O_{ABS}$-I and $ABS$-I (values in parentheses are the
time cost for \( ABS-I \), while the other is for \( OABS-I \). Actually, our scheme \( OABS \) demands additional verifying operations (nearly one pairing operation) for the default attribute \( \theta \). This makes the verification of our scheme is slightly slower than that of the original scheme. Moreover, we observe the efficiency in both \( OABS-I \) and \( ABS-I \) is independent with the threshold value \( d \), because we generate the experimental signature for verification in the same manner with the signing phase, which only embeds the signer’s attributes and will not involve the dummy attributes.

5.2 Experimental Results for \( OABS-II \)

As with the same assumption in the evaluation of \( OABS-I \), we also evaluate the efficiency of \( OABS-II \) and compare it with that of original scheme \( ABS-II \). Fig. 2(a) shows the setup phase of \( OABS-II \) and \( ABS-II \). Actually, both schemes share almost the same computational complexity in this phase, leading to the similar efficiency presentation. But unlike the same phase in \( OABS-I \) and \( ABS-I \), the efficiency in setup phase of \( OABS-II \) and \( ABS-II \) is mainly depended on the threshold value \( d \) because the setup algorithm does not need to initialize a group element for each attribute in the universe. As a result, the setup in \( OABS-II \) is more efficient than that in \( OABS-I \) for small \( d \).

The efficiency curve in Fig. 2(b) for \( OABS-II \) and \( ABS-II \) shows a similar shape to that in the previous experiment for \( OABS-I \) and \( ABS-I \). But unlike the first scheme that only needs two exponentiations in order to generate private key component for a single attribute, the attribute authority in the second scheme has to compute \( (2d + 2) \) elements in \( G \) for each attribute, leading to the intensive complexity in this phase.

Similarly, in the signing phase (the efficiency is shown in Fig. 2(c)), \( OABS-II \) is slower than \( ABS-II \) in overall performance, but has a much more efficient in local performance. Furthermore, we have to point out that unlike a linear efficiency (linear with both \( \Omega \) and \( d \)) in the first scheme, the time cost of signing in original scheme \( ABS-II \) is linear with \( d^2 \) (i.e., \( O(d^2) \)). Due to our outsourcing technique, the local computation in \( OABS-II \) is now reduced to \( O(d) \).

The experimental results of the verifying phase for our second scheme are shown in TABLE 2 (values in parentheses are the time cost for \( ABS-II \), while the other is for \( OABS-II \)). It is clear that \( OABS-II \) and \( ABS-II \) share an identical efficiency in this phase, which is just depended on the threshold \( d \) in predicate (note that the predicate used is \( \tau_{d,\Omega^*} \) as with the assumption in the evaluation of the first scheme).

6 OABS WITH S-CSP ACCOUNTABILITY

In this Section, we show how to extend OABS with the additional property of 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 [25][27], 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 behaviors
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 6 (Accountability): An outsourced ABS scheme 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 OABS-I which is defined in Section 4.1. Actually, the technique can be easily applied to 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 $SIG = (KS, SIG, VER)$, where KS, SIG, and VER denote setup algorithm, signing algorithm and verifying algorithm, respectively. The accountable OABS construction is identical to OABS-I in Section 4.1, except that the Sign$_{out}$ algorithm is also involved to realize the accountability of S-CSP. We present the intuition of the accountable OABS as follows: After complete the outsourcing key $OK$ and the corresponding generated signature $\sigma'$, the original signer only needs to verify the ordinary signature signed by S-CSP, instead of verifying the signature $\sigma'$ of OABS. Then, if the

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.

- Trace: When a verification on ABS signature $\sigma$ is aborted, the original signer provides $\sigma', OK$ and tag to an arbitration agency. The arbitration agency firstly checks the correctness of the tag by running the verifying algorithm of $SIG$. If the verification holds, arbitration agency further checks the correctness of $\sigma' = (\sigma'_0, \{\sigma'_i\}_{i \in \Omega'} , \hat{\Omega}')$ through verifying the equation of $e(g, \sigma'_0) \prod_{i \in \Omega'} [e(\sigma'_i, H_i(i))] = e(g^{\tau'}, g)$ with respect to $g^{\tau'}$. If the above equation does not hold, it further verifies the correctness of $OK$ by running Sign$_{out}$ 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.

### 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 [23], 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 show practical solutions as well.

Acknowledgement

We are grateful to the anonymous referees for their invaluable suggestions. This work is supported by the National Natural Science Foundation of China (Nos. 61272455, 61202450 and 61100224), Ph.D. Programs Foundation of Ministry of Education of China (No. 20123503120001), Distinguished Young Scholars Fund of Department of Education, Fujian Province (JA13062), the Fundamental Research Funds for the Central Universities (K50511010001 and JY10000901034), and China 111 Project (No. B08038).

References

Xiaofeng Chen received his B.S. and M.S. on Mathematics in Northwest University, China. He got his Ph.D degree in Cryptography from Xidian University at 2003. Currently, he works at Xidian University as a professor. His research interests include applied cryptography and cloud computing security. He has published over 100 research papers in refereed international conferences and journals. His work has been cited more than 1000 times at Google Scholar. He has served as the program/general chair or program committee member in over 30 international conferences.

Yang Xiang received his PhD in Computer Science from Deakin University, Australia. He is currently a full professor at School of Information Technology, Deakin University. He is the Director of the Network Security and Computing Lab (NSCLab). His research interests include network and system security, distributed systems, and networking. In particular, he is currently leading his team developing active defense systems against large-scale distributed network attacks. He is the Chief Investigator of several projects in network and system security, funded by the Australian Research Council (ARC). He has published more than 130 research papers in many international journals and conferences, such as IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Information Security and Forensics, and IEEE Journal on Selected Areas in Communications. Two of his papers were selected as the featured articles in the April 2009 and the July 2013 issues of IEEE Transactions on Parallel and Distributed Systems. He has published two books, Software Similarity and Classification (Springer) and Dynamic and Advanced Data Mining for Progressing Technological Development (IGI-Global). He has served as the Program/General Chair for many international conferences such as ICASP 12/11, IEEE/IFIP EUC 11, IEEE TrustCom 13/11, IEEE HPCC 10/09, IEEE ICPADS 08, NSS 11/10/09/08/07. He has been the PC member for more than 60 international conferences in distributed systems, networking, and security. He serves as the Associate Editor of IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, Security and Communication Networks (Wiley), and the Editor of Journal of Network and Computer Applications. He is the Coordinator, Asia for IEEE Computer Society Technical Committee on Distributed Processing (TCDP). He is a Senior Member of the IEEE.

Jin Li received his B.S. (2002) in Mathematics from Southwest University. He got his Ph.D degree in information security from Sun Yat-sen University at 2007. Currently, he works at Guangzhou University as a professor. He has been selected as one of science and technology new star in Guangdong province. His research interests include Applied Cryptography and Security in Cloud Computing. He has published over 50 research papers in refereed international conferences and journals and has served as the program chair or program committee member in many international conferences.

Xinyi Huang received his Ph.D degree from the School of Computer Science and Software Engineering, University of Wollongong, Australia, in 2009. He is currently a Professor at the Fujian Provincial Key Laboratory of Network Security and Cryptology, School of Mathematics and Computer Science, Fujian Normal University, China. His research interests include cryptography and information security. He has published over 60 research papers in refereed international conferences and journals. His work has been cited more than 1000 times at Google Scholar. He is in the Editorial Board of International Journal of Information Security (IJIIS, Springer) and has served as the program/general chair or program committee member in over 40 international conferences.

Yang Xiang received his PhD in Computer Science from Deakin University, Australia. He is currently a full professor at School of Information Technology, Deakin University. He is the Director of the Network Security and Computing Lab (NSCLab). His research interests include network and system security, distributed systems, and networking. In particular, he is currently leading his team developing active defense systems against large-scale distributed network attacks. He is the Chief Investigator of several projects in network and system security, funded by the Australian Research Council (ARC). He has published more than 130 research papers in many international journals and conferences, such as IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Information Security and Forensics, and IEEE Journal on Selected Areas in Communications. Two of his papers were selected as the featured articles in the April 2009 and the July 2013 issues of IEEE Transactions on Parallel and Distributed Systems. He has published two books, Software Similarity and Classification (Springer) and Dynamic and Advanced Data Mining for Progressing Technological Development (IGI-Global). He has served as the Program/General Chair for many international conferences such as ICASP 12/11, IEEE/IFIP EUC 11, IEEE TrustCom 13/11, IEEE HPCC 10/09, IEEE ICPADS 08, NSS 11/10/09/08/07. He has been the PC member for more than 60 international conferences in distributed systems, networking, and security. He serves as the Associate Editor of IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, Security and Communication Networks (Wiley), and the Editor of Journal of Network and Computer Applications. He is the Coordinator, Asia for IEEE Computer Society Technical Committee on Distributed Processing (TCDP). He is a Senior Member of the IEEE.

Duncan Wong received his B.Eng. degree in Electrical and Electronic Engineering with first class honors from the University of Hong Kong in 1994, M.Phil. degree in Information Engineering from the Chinese University of Hong Kong in 1998 and Ph.D. degree in Computer Science from Northeastern University, Boston, MA, USA in 2002. After graduation, he has been a visiting assistant professor at the Chinese University of Hong Kong for one year before joining City University of Hong Kong in September 2003. He is now an associate professor in the Department of Computer Science.