This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

Outsourced Decentralized Multi-authority Attribute Based Signature and Its Application in IoT

Jiameng Sun, Ye Su, Jing Qin, Jiankun Hu , *Senior Member, IEEE,* and Jixin Ma.

***Abstract*—IoT (Internet of things) devices often collect data and store the data in the cloud for sharing and further processing; This collection, sharing, and processing will inevitably encounter secure access and authentication issues. Attribute based signature (ABS), which utilizes the signer's attributes to generate private keys, plays a competent role in data authentication and identity privacy preservation. In ABS, there are multiple authorities that issue different private keys for signers based on their various attributes, and a central authority is usually established to manage all these attribute authorities. However, one security concern is that if the central authority is compromised, the whole system will be broken. In this paper, we present an outsourced decentralized multi-authority attribute based signature (ODMA-ABS) scheme. The proposed ODMA-ABS achieves attribute privacy and stronger authority-corruption resistance than existing multi-authority attribute based signature schemes can achieve. In addition, the overhead to generate a signature is further reduced by outsourcing expensive computation to a signing cloud server. We present extensive security analysis and experimental simulation of the proposed scheme. We also propose an access control scheme that is based on ODMA-ABS.**

***Index Terms*—attribute based signature, anonymous authentication, outsourcing computation, access control.**

I. Introduction

ATTRIBUTE based signature (ABS) is a primitive that derives from identity based signature [1]. ABS enables a signer to endorse a piece of message using a set of attributes instead of his/her unique identity, promoting the harmony between message endorsement and identity privacy preservation. In an ABS, the user uses his attribute set to query the attribute authority for private keys corresponding to certain attributes. Since an individual may have various kinds of attributes, e.g. gender, profession, address, etc., there are usually multiple attribute authorities that handle requests of different kinds of attributes. This multi-authority setting helps amortize the computational overhead in a single authority case and more importantly, enhances the security since one or some of the authorities' compromising or corruption may not affect the others.

However, a multi-authority setting leads to difficulty regarding how to generate a common secret for signature's generation given that authorities may not trust or communicate with each other. A feasible means is to establish a central authority that holds the common secret and allocates different parts of this secret to each attribute authority to help with private keys' generation. However this means returns us to the disadvantages in a single authority scenario because once the central authority is compromised, the whole system will no longer be secure. Therefore, a key point in a multi-authority attribute based signature is decentralization. In this paper, we take use of the idea that is introduced in decentralized multi-authority attribute based encryption schemes [2]. The core idea is to simulate the functionality of a central authority through interactions among different attribute authorities.

Efficiency is another key point for the practical applications of the ABS. Actually, the multi-authority setting does not reduce overhead for the signer, instead, it may even increase the overhead in certain ways because the signer must interact with multiple authorities to obtain the secret keys associated with his/her whole attributes. This situation may even worsen when the signer wants to perform the message endorsing action on portable devices that lack sufficient computational power, such as the IoT devices. Fortunately, this predicament can be effectively improved if we notice that generating a signature is essentially using private keys to execute a computation on the message because now, with the assistance of Cloud computing, the signer can choose to outsource a heavy portion of the computation that must be done in the signing phase to a cloud server to reduce his/her local cost. In this paradigm, since the cloud server is commercial and may not be fully trusted, the challenge is to assure the correctness of a server's output as well as the security of the signature protocol (e.g. unforgeability). Specifically, given an attribute based signature protocol, the signer cannot simply send all of his/her private keys to the cloud server to generate the signature. This issue is obvious because, if so, the server will be able to forge the signer's signature on any other messages thereafter. To maintain security, one feasible means is to encode the private keys using fully homomorphic encryption; however, this will inherently bring huge computation overhead. In this paper, we divide the private keys in two parts and let the signer outsource the "big" part of them while maintaining the "small" part as a secret. The cloud server uses the "big" part of the private keys to generate a partial signature and return it to the signer. The signer can combine the other part that is generated from the

Jiameng Sun is with School of Mathematics, Shandong University, Jinan, 250100, China. E-mail: (sunjiameng1991@163.com)

Ye Su is with School of Mathematics, Shandong University, Jinan, 250100, China. E-mail: (sy0422@163.com)

Jing Qin (corresponding author) is with School of Mathematics, Shandong University, Jinan, 250100, China, and State Key Laboratory of Information Security Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China. E-mail:(qinjing@sdu.edu.cn)

Jiankun Hu is with School of Engineering and Information Technology, University of New South Wales Defence Force Academy, Canberra, Australia. E-mail:(J.Hu@adfa.edu.au)

Jixin Ma is with Centre for Computer and Computational Science at School of Computing and Mathematical Sciences, University of Greenwich, London, UK. E-mail:(j.ma@greenwich.ac.uk)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

"small" private key part with the partial signature to obtain the final signature. Note that, without the part of the signature that is generated from the "small" private key part, the partial signature is not a valid signature. In addition, it is necessary that the partial signature is able to be verified by the signer to maintain correctness. Fortunately, this verification can be derived from the verification of the original ABS protocol; this enables us to construct the secure and efficient ABS protocol.

A. Applications

As an important cryptographic primitive to maintain anonymity, (multi-authority) attribute based signature has various applications in the IoT scenario. Here, we introduce two common applications, anonymous data certification and access control. There are also other applications of ABS, such as fulfilling security requirement in attribute based messaging system [3], etc..

1) Anonymous data certification: In the IoT paradigm, smart devices that are equipped with sensors are connected to the cloud. The sensors continuously collect data information from the surroundings and then upload these data to the cloud for sharing or further processing. For example, in the vehicular ad hoc networks (VANETs), sensors on the vehicle collect the surrounding road condition of their location and can share it with others in the network. In the smart home system, the sensors embedded in the house collect information of the user's house, such as temperature and light intensity, and upload the data to the cloud, after which evaluations on the data are performed to decide the optimal deployment for this house. Usually, the data that is collected by the sensors should be certificated before uploading to convince the cloud that the data are from devices that belong to a legitimate user. To do so, it is feasible to utilize the user's identity to generate a signature for the data, that is, applying an identity based signature scheme on the data. However, this means will inherently expose the user's identity. Notice that the exposure of the identity may lead to huge security concerns. For example, in a VANET, the road information that is uploaded by the user implicitly implies the location of the user him/herself, and an adversary can easily obtain it then infer where the user is when this uploaded information is certified by his/her certain identity. Additionally, in the smart home system, if the uploaded information is linked to a certain identity, an adversary can observe and obtain useful information about the user's house such as when it is empty. The above concerns are caused because the linkage between data and an identity exists. However, it is currently sufficient to certify that the data originates from a legitimate user if we can prove that the data owner has some required characters (or attributes). Attribute based signature is able to realize this functionality and therefore preserves the identity privacy.

2) Access control: Another application of attribute based signature is to provide an access control policy for IoT devices. Traditional access control schemes usually grant access by establishing a relationship between devices and individuals. However, with the increase of the amount and variety of IoT devices, it is inconvenient to define the identities that can access certain devices, especially when the device is not personal. In fact, the access to such devices is essentially defined by a number of characters (attributes) instead of specific identities. furthermore, if the user accesses a public device by verifying his/her identity, certain private information may be leaked by monitoring the device. For example his/her location information may be learned by tracking the location of the device. Since multi-authority attribute based signature implicitly provides a means to prove the legitimacy of an anonymous identity, it can be utilized to construct a flexible and privacy preserving access control scheme that is based on attributes.

B. Our Contribution

In this paper, we propose a novel model called outsourced decentralized multi-authority attribute based signature (ODMA-ABS). This model captures both the security requirements of an ABS protocol and the efficiency requirement of an outsourcing computation protocol. We present the specific construction of an ODMA-ABS protocol and provide the corresponding analysis of each property, such as correctness, unforgeability, attribute privacy, efficiency, outsourcing security and privacy. We also conduct an experimental simulation to show the performance of ODMA-ABS. In addition, we propose an attribute based access control scheme, ABAC, as a specific application of ODMA-ABS and discuss the key update of ABAC.

This paper is an extension of its corresponding conference version [6]. In the revised version, the adversarial model of ODMA-ABS is enhanced from semi-honest to malicious. We propose a novel ODMA-ABS that is secure under this new model, analyze the outsourcing security and privacy, and conduct an experimental simulation to show practical performance. The proposed novel ABAC achieves nontransferability and is efficient for the user by introducing an aiding server. In addition, some of the content is extended. For example, we specifically explain the ideas behind our construction and the unforgeability proof. We formalize the definition of attribute privacy and provide rigorous proof instead of simply intuition, and more figures and tables are added for better understanding. All of the above are not shown in the conference version.

C. Related Works

1) Attribute based signature: There are a number of protocols that maintain anonymous authentication, such as group signature [4] and ring signature [5]. However, these protocols all suffer from certain disadvantages. Group signature, which requires a "manager" to address the identities of the group, may suffer from identity exposure when the "manager" is compromised. Ring signature, which requires the signer to collect all the public keys of other members in the ring, may not be convenient in many practical scenarios. Considering thiss situation, attribute based signature may play a much better role in practice. ABS was first defined and constructed in 2008 by Maji et al. [3]. The researchers presented a scheme supporting the predicate described by a monotone span program. Later, a (k, n)-threshold predicate construction was

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

proposed by Shahandashti and Safavi-Naini [7]. In 2010, Li et al. [8] proposed another construction that achieves better efficiency than that in [7].Futhermore, the researchers also proposed a construction for the multi-authority scenario. There are other works that focus on MA-ABS [9], [10], [11]. In 2013, Okamoto and Takashima [12] proposed the first decentralized multi-authority attribute-based signature (DMA-ABS) scheme in the random oracle model, which removes the trusted central authority of the MA-ABS scheme. Although the researchers' scheme was as efficient as the former MA-ABS, the sizes of the private keys and signature remain very large.

2) Outsourcing computation: Outsourcing computation was first proposed by Hohenberger and Lysyanskaya [13] and formally defined by Gennaro et al. [14]. Gennaro et al. proposed a means to use Gentry's [15] fully homomorphic encryption (FHE) together with Yao's [16] 2-party computation to securely outsource any circuit evaluation. Since operating FHE is very expensive, the scheme is not very practical. Later other works [17], [18], [19] were proposed to improve the performance in using FHE, but the performance is still not ideal.

To avoid using FHE, certain works have been done to seek other means to securely outsource specific kinds of computation. Regarding attribute based cryptography, Green et al. [20] proposed a scheme to outsource the decryption phase of attribute based encryption (ABE) scheme, and Zhou et al. [21] proposed a scheme to outsource the encryption and decryption of ABE. Recently, Cai et al. [22] proposed an Electronic health record system, which supports offline encryption and outsourced decryption in mobile cloud computing. In 2014, Chen et al. [23] proposed an outsourced attribute based signature (OABS) scheme. The researchers utilize the blinding technique to make attribute private keys piecewise; thus, it will preserve attribute privacy in the outsourcing paradigm. However the scheme is constructed in a single authority model. In 2017, Ren et al. [24] proposed another OABS scheme that achieves correctness verification of the outsourced signing operation; however, it remains a single authority proposal. In Table I, we present a comparison in certain properties among different ABS schemes, including the decentralized multi-authority ABS[12], the outsourced ABS [23], the outsourced ABS with verification of outsourcing operation [24] and our ODMA-ABS.

D. Paper Organization

The remaining part of the paper is organized as follows. Necessary preliminaries are provided for the proposed schemes in section II. The model of decentralized multi-authority attribute based signature (DMA-ABS) is defined in section III. The proposed outsourced decentralized multi-authority attribute based signature scheme is presented and analyzed in section IV. An attribute based access control scheme is proposed and discussed in section V. Section VI concludes the paper.

II. Preliminaries

In this section, we first introduce the notations used in this paper. Then we provide the definitions of pseudorandom function (PRF), Bilinear Map, threshold predicate for multi set and Lagrange coefficient. We also provide the computational assumptions that our protocol are based on, the co-CDH assumption and XDH assumption.

A. Notations

The main notations that are used in this paper are shown and explained in Table II.

TABLE II
NOTATIONS IN THIS PAPER

Notations	Description
$A_{k,u}$	attribute subset that belongs to user u and is managed by attribute authority k
Υ_{n,C^*}	(n, C^*)-threshold predicate
$G \in_R \mathbb{G}$	randomly select an element G in group $\mathbb{G}$
$\hat{e}(g_1, g_2)$	bilinear operation of group element g_1 and g_2
H	Hash function
σ	signature

B. Pseudorandom Function

Our construction utilizes the pseudorandom function (PRF) [25]. Informally speaking, a PRF takes as input a random value in the domain and outputs a value that is computational indistinguishable from a truly random value in the range. We give the specific definition of the pseudorandom function family below.

Definition 1 (Pseudorandom function family)**.** *Let* $\mathcal{F} = \{f_s | s \in \{0,1\}^\lambda\}_{\lambda \in \mathbb{N}}$. *Then* $\mathcal{F}$ *is called a family of* $(X(\lambda), Y(\lambda))$ *pseudorandom function if the following holds,*

- $\forall \lambda \in \mathbb{N}, \forall s \in \{0,1\}^\lambda, f_s : \{0,1\}^{X(\lambda)} \rightarrow \{0,1\}^{Y(\lambda)}$*;*
- $\forall \lambda \in \mathbb{N}, \forall s \in \{0,1\}^\lambda, f_s$ *can be computed in polynomial time;*
- *Pseudorandomness: for any probabilistic polynomial time algorithm* $\mathcal{A}$*, there holds*

$$|Pr[\mathcal{A}^{f_s}(1^\lambda) = 1 | s \xleftarrow{U} \{0,1\}^\lambda] - Pr[\mathcal{A}^{g}(1^\lambda) = 1 | g \xleftarrow{U} \mathcal{R}(X(\lambda), Y(\lambda))]| \leq negl(\lambda),$$

where $\mathcal{R}(X(\lambda), Y(\lambda))$ *is the set of all possible functions* $g : \{0,1\}^{X(\lambda)} \rightarrow \{0,1\}^{Y(\lambda)}$.

TABLE I
COMPARISON AMONG DIFFERENT ATTRIBUTE BASED SIGNATURE SCHEMES

	DMA-ABS[12]	OABS[23]	OABS-V[24]	ODMA-ABS
Authority(ies) involved	multiple(N)	single	single	multiple(N)
Decentralization	yes	$\perp$	$\perp$	yes
Corrupted authority(ies) tolerance	weak	no	no	strong($N-1$)
Outsourcing verifiability	$\perp$	no	yes	yes

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

C. Bilinear Map

Our construction also utilizes the bilinear map. The bilinear map is a powerful tool in non-interactive authentication and has been widely applied in both signature and outsourcing computation schemes [26], [27].

Definition 2 (Bilinear map)**.** *Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be finite cyclic multiplicative groups of prime order p, and g_1, g_2 be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. A map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is called a bilinear map if it satisfies the following properties:*

- *Bilinearity: it holds that*

$$\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab} \tag{1}$$

  for all $a, b \in \mathbb{Z}_p$.
- *Non-degeneracy: There exist $G_1 \in \mathbb{G}_1, G_2 \in \mathbb{G}_2$ such that $\hat{e}(G_1, G_2) \neq 1$.*
- *Computability: There exists an efficient algorithm to compute $\hat{e}(G_1, G_2)$ for any $G_1 \in \mathbb{G}_1, G_2 \in \mathbb{G}_2$.*

We say that the bilinear group is symmetric if there exists an efficiently computable isomorphism ϕ from $\mathbb{G}_1$ to $\mathbb{G}_2$ and an efficiently computable isomorphism ϕ' from $\mathbb{G}_2$ to $\mathbb{G}_1$. Similarly, we say that the bilinear group is asymmetric if such ϕ' does not exist. (ϕ from $\mathbb{G}_1$ to $\mathbb{G}_2$ may or may not exists.) Our construction is based on asymmetric group where such ϕ exists.

D. Threshold Predicate

Our construction applies to threshold predicate scenario. We modify the definition of threshold presicate to make it suitable for multi-set scenario.

Definition 3 (Threshold predicate)**.** *Let $C^* = \bigcup_{k\in\{1,\cdots,N\}} C_k^*$ be a union of attribute sets, with each C_k^* of size m_k. And let $n = \sum_{k\in\{1,\cdots,N\}} n_k$ be an integer, where each n_k is an integer in $[0, m_k]$. A threshold predicate for multi-set is a monotone boolean function that is defined as follows.*

$$\Upsilon_{n,C^*}(A) = \begin{cases} 1, & |A \cap C_k^*| \geq n_k, \text{ for } k \in \{1,\cdots,N\} \\ 0, & \text{otherwise} \end{cases}$$

We say that an attribute set A satisfies a predicate Υ if $\Upsilon(A) = 1$.

E. Lagrange Coefficient

Our construction utilizes the Lagrange interpolation.

Definition 4 (Lagrange coefficient)**.** *Let $p(\cdot)$ be a $d-1$ degree polynomial, and $p(1), \cdots, p(d)$ be the corresponding values in d points. Write $S = \{1, \cdots, d\}$. We define the Lagrange coefficient in the computation $p(x)$ as*

$$\Delta_{i,S}(x) = \prod_{j\in S, j\neq i} \frac{x-j}{i-j}$$

Then the polynomial $p(x)$ can be represented as

$$p(x) = \sum_{i=1}^{d} p(i)\Delta_{i,S}(x) \tag{2}$$

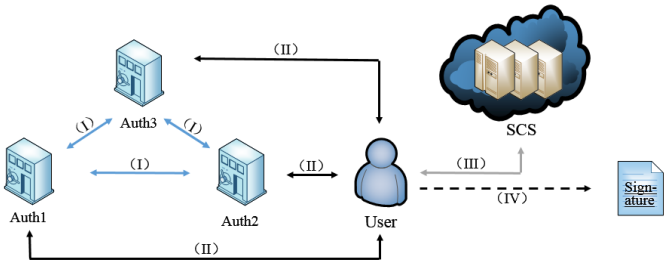


Fig. 1. System Architecture: "Auth1, Auth2, Auth3" refer to different attribute authorities, "User" refers to the signer who would like to generate the signature, "SCS" refers to the signing cloud server that generates partial signature for the user. The signature is verified by a verifier, who is not indicated in the figure.

F. Computational Assumption

Our construction is based on standard assumptions, the co-CDH assumption and the XDH assumption.

Definition 5 (co-CDH assumption)**.** *Let $\mathcal{G}_B$ be a bilinear group generator that takes as input a parameter λ and outputs a description of a bilinear group $(p, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e})$, where p is the order, g_1, g_2 the generators of $\mathbb{G}_1, \mathbb{G}_2$ respectively, and $\hat{e}$ the bilinear map from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$. We say that the co-CDH assumption (t, ϵ) holds if for any probabilistic polynomial t-time algorithm $\mathcal{A}$ there holds*

$$|Pr[\mathcal{A}(p, g_1, g_2, g_1^a, g_2^b) = g_2^{ab}]| \leq \epsilon,$$

where $a, b \in_R \mathbb{Z}_p$.

Definition 6 (XDH assumption)**.** *Let $(p, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, \phi)$ be an asymmetric bilinear tuple where ϕ is a one way map from $\mathbb{G}_1$ to $\mathbb{G}_2$. We say that the XDH assumption (t, ϵ) holds in $\mathbb{G}_2$ if for any probabilistic polynomial t-time algorithm $\mathcal{A}$ there holds*

$$|Pr[\mathcal{A}(p, \hat{e}, \phi, g_1, g_2, g_2^a, g_2^b, g_2^{ab}) = 1] - Pr[\mathcal{A}(p, \hat{e}, \phi, g_1, g_2, g_2^a, g_2^b, g_2^c) = 1| \leq \epsilon,$$

where $a, b, c \in_R \mathbb{Z}_p$.

Notice that the XDH assumption does not hold in symmetric bilinear groups. This is obvious since that one can map g_2^a onto the element in $\mathbb{G}_1$ and uses bilinear map $\hat{e}$ to distinguish the XDH tuple. The same thing also holds for group $\mathbb{G}_1$ in asymmetric setting since ϕ exists.

III. Modeling ODMA-ABS

In this section, we present some relative definitions of the ODMA-ABS. Fig.1 shows the system architecture of ODMA-ABS. We define each algorithm that is corresponding to each phase in Fig.1.

A. Definitions

Definition 7 (Outsourced decentralized multi-authority attribute based signature)**.** *An outsourced decentralized multi-authority attribute based signature protocol π is defined via the following five algorithms.*

(I) $(PP, MSK) \leftarrow$ ***Setup***(λ, N)*: The randomized system setup algorithm takes as input a secure parameter λ, the*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

number of attribute authorities N, and outputs the public parameter PP and the master secret key MSK for each attribute authority. This is done by interactions among the multiple authorities.

(II) $(OK, SK) \leftarrow$ ***AKeyGen***(PP, MSK, A_u)*: The randomized key generation algorithm takes as input the public parameter PP, the master secret key MSK, the attribute set A_u of user u and outputs the outsourcing key OK and the private key SK. Here $A_u = \bigcup_{k\in\{1,\cdots,N\}} A_{k,u}$, where $A_{k,u}$ represents the attribute subset that is certified by the kth attribute authority. This procedure is done independently by interaction between the user and each attribute authority.*

(III) $(\sigma_{part}, W) \leftarrow$ ***Sign***$_{out}(OK, A_u, \Upsilon_{n,C^*})$*: The randomized outsourced signing algorithm takes as input the outsourcing key OK of all attribute authorities, the user's attribute set A_u, a predicate Υ_{n_k,C_k^*} where $C^* = \bigcup_{k\in\{1,\cdots,N\}} C_k^*$ and $n = \sum_{k\in\{1,\cdots,N\}} n_k$. It outputs the partial signature σ_{part} and the corresponding witness W. This algorithm is done by the signing cloud server (SCS).*

(IV) $(\sigma, \perp) \leftarrow$ ***Sign***$(SK, m, \sigma_{part}, W, \Upsilon_{n,C^*})$*: The randomized signing algorithm takes as input the private key SK, the message to-be-signed m, the partial signature σ_{part}, the witness W, the predicate Υ_{n,C^*} and outputs the formal signature σ if the partial signature is verified and accepted, or a symbol $\perp$ if the partial signature is rejected. This is done by the user.*

(V) $b \leftarrow$ ***Verify***$(PP, \sigma, m, \Upsilon_{n,C^*})$*: The deterministic verification algorithm takes as input the public parameter PP, the signature σ with signed message m and predicate Υ_{n,C^*} and outputs 1 if it the signature is valid and 0 otherwise. This is done by the verifier.*

Definition 8 (Correctness)**.** *An ODMA-ABS protocol is correct if for all $(PP, MSK) \xleftarrow{R}$ **Setup**(λ, N), all attribute sets $\{A_k\}$, all $(OK, SK) \xleftarrow{R}$ **AKeyGen**(PP, MSK, A_u), all predicate Υ_{n,C^*}, all $\sigma_{part} \xleftarrow{R}$ **Sign**$_{out}(OK, A_u, \Upsilon_{n,C^*})$, all messages m, and all $\sigma \xleftarrow{R}$ **Sign**$(SK, m, \sigma_{part}, \Upsilon_{n,C^*})$, the verification algorithm outputs 1 with probability $1-\epsilon$ where ϵ is negligible.*

An ODMA-ABS protocol is required to satisfy the existential unforgeability property. Consider the unforgeability experiment $Exp^{euf}_{\mathcal{F},\text{ODMA-ABS}}$ between a challenger $\mathcal{C}$ and an adversary $\mathcal{F}$ in a selective predicate model.

- **Initial** $\mathcal{C}$ receives a corrupted authorities set $\mathbb{K}_c \subseteq \{1,\cdots,N\}$ and the challenge predicate Υ_{n,C^*}, where $C^* = \bigcup_{k\notin\mathbb{K}_c} C_k^*$ and $n = \sum_{k\notin\mathbb{K}_c} n_k$.
- **Setup** $\mathcal{C}$ runs the **Setup**(λ, N) algorithm. It sends PP to $\mathcal{F}$ and keeps MSK.
- **Query** $\mathcal{F}$ is allowed to perform a series of the following queries.
  - *Private key query*: $\mathcal{F}$ can query $\mathcal{C}$ for SK with an identity u.
  - *Outsourcing key query*: $\mathcal{F}$ can query $\mathcal{C}$ for OK with attribute union set $A = \bigcup_{k\notin\mathbb{K}_c} A_k$.
  - *Signing query*: $\mathcal{F}$ can query $\mathcal{C}$ for signature with message m and predicate Υ.
- **Forgery** $\mathcal{F}$ outputs a tuple $(m^*, \sigma^*, \Upsilon_{n,C^*})$.

We say that $\mathcal{F}$ wins if i) $(m^*, \sigma^*, \Upsilon_{n,C^*})$ passes the verification algorithm; ii) $\Upsilon_{n,C^*}(A) \neq 1$ for any A that has been submitted to the *Outsourcing key query* oracle; iii) (m^*, Υ_{n,C^*}) has not been submitted to the *Signing query* oracle.

Definition 9 (Unforgeability)**.** *An ODMA-ABS scheme is $(t, \epsilon, n, q_P, q_O, q_S)$-unforgeable if no probabilistic polynomial t time adversary $\mathcal{F}$ can win the unforgeability experiment over ϵ when at most n attribute authorities are corrupted, with at most q_P, q_O, and q_S times of private key query, outsourcing key query and signing query respectively.*

An ODMA-ABS protocol is supposed to achieve attribute privacy for the signer. Informally speaking, attribute privacy requires that the signature reveals nothing about user's identity or attribute set except for what is revealed explicitly. This holds even for the situation that the adversary gets access to the user's attribute private keys. Former definition is based on the following experiment $Exp^{apr}_{\mathcal{F},\text{ODMA-ABS}}$ between a challenger $\mathcal{C}$ and an adversary $\mathcal{F}$.

- **Setup** $\mathcal{C}$ runs the **Setup**(λ, N) algorithm. It sends (PP, MSK) to $\mathcal{F}$.
- **Query** $\mathcal{F}$ is provided with access to *private key oracle*, *outsourcing key oracle* and *signing oracle* as that in $Exp^{\text{euf}}_{\mathcal{F},\text{ODMA-ABS}}$.
- **Challenge** $\mathcal{F}$ outputs a tuple $(A_{u0}, A_{u1}, \Upsilon_{n,C^*}, m)$, where $\Upsilon_{n,C^*}(A_{ui}) = 1$ for $(i = 0, 1)$. $\mathcal{F}$ then sends the tuple to $\mathcal{C}$. $\mathcal{C}$ picks a random bit $b \in \{1, 2\}$ and runs **AKeyGen** algorithm with input (PP, MSK, A_{ub}). Consequently, $\mathcal{C}$ gets OK_b and SK_b. It then runs algorithms **Sign**$_{\text{out}}$ and **Sign** with input $(OK_b, A_{ub}, \Upsilon_{n,C^*})$ and $(SK, m, \sigma_{\text{part}}, \Upsilon_{n,C^*})$ respectively to get a challenging signature σ^*, which is then sent back to $\mathcal{F}$.
- **Guess** Upon receiving σ^*, $\mathcal{F}$ outputs a bit $b' \in \{0, 1\}$ and wins the game if $b' = b$.

Definition 10 (Attribute privacy)**.** *An ODMA-ABS protocol achieves attribute private for the signer if no probabilistic polynomial time adversary $\mathcal{F}$ can win the above game with an advantage probability no less than $\frac{1}{2} + negl(\lambda)$, where $negl()$ is a negligible function.*

As an outsourcing computation protocol, the ODMA-ABS is also required to satisfy the efficiency property.

Definition 11 (Efficiency)**.** *An ODMA-ABS scheme is efficient if the total computation cost on the signer side is less than the cost to generate the signature all by the user.*

IV. The Proposed ODMA-ABS Protocol

A. Anonymous Key Issue Protocol

Our construction takes use of an anonymous key issue (AKI) protocol proposed in [28]. We would like to give a brief introduction first.

The AKI is an interactive protocol executed between a user and an attribute authority. In the AKI, a user U and an attribute authority A have access to some commonly known values, and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

keep some secret values. They jointly compute a value through the following steps.

- **Step 1**: Set up the public parameters including a group $\mathbb{G}$ of order p, two group elements $g, h \in \mathbb{G}$ and a hash function that maps a string to an element in $\mathbb{Z}_p$.
- **Step 2**: A general 2-party computation protocol is executed on input (u, a_1) from U and β from A, where u is the hash value of the U's identity GID, and $(a_1, \beta) \in_R \mathbb{G}$. Such 2-party computation can be realized via the protocol in [29]. As a result, A obtains $x := (\beta + u)a_1 \mod p$.
- **Step 3**: A computes $X_1 := g^{\tau/x}$, $X_2 := h^{\alpha\tau}$ and sends X_1, X_2 to U, where $\alpha, \tau \in_R \mathbb{Z}_p$.
- **Step 4**: On receiving X_1, X_2, U picks $a_2 \in_R \mathbb{Z}_p$ and computes $Y := (X_1^{a_1} X_2)^{a_2}$. Then Y is sent to A.
- **Step 5**: On receiving Y, A picks a $\gamma \in_R \mathbb{Z}_p$ and computes $Z := Y^{\gamma/\tau}$. Then Z is sent to U.
- **Step 6**: finally, on receiving Z, U computes $Z^{1/a_2} = (h^{\alpha} g^{1/(\beta+u)})^{\gamma}$.

The correctness of the above protocol is easy to verify by the following equation.

$$\begin{aligned} Z^{1/a_2} &= Y^{\gamma/(\tau/a_2)} = (X_1^{a_1\gamma/\tau} X_2^{\gamma/\tau}) \\ &= (g^{a_1\gamma/x} h^{\alpha\gamma}) = (h^{\alpha} g^{1/(\beta+u)})^{\gamma} \end{aligned} \tag{3}$$

The above protocol is secure under DDH assumption assuming that the underlying 2-party computation is secure. The proof can be found in [28].

Now we are about to utilize the AKI protocol to construct the signature protocol.

B. ODMA-ABS Construction

The construction is based on the signature scheme of Li et at.'s [8]. In [8], a multi-authority ABS is proposed with a central authority established to allocate differen parts of a shared secret to each authority. By utilizing the AKI protocol above, the function of the central authority can be simulated by interactions among the multiple authorities. However, notice that the scheme in [8] is based on the symmetric bilinear group where XDH assumption does not hold and thus the AKI can not be used directly. So we should first change the underlying bilinear group to an asymmetric one and reconstruct a corresponding multi-authority attribute based signature protocol. After that we are able to use the AKI protocol to remove the central authority. In addition, it is observed that the resulting attribute signing keys for the user cannot reconstruct a secret without the master secret key, hence they can be outsourced to a signing cloud server (SCS) to generate a partial signature and the user can then combine this partial signature with the other part that is generated from the master secret key to fulfill the formal signature, which will greatly reduce the computational overhead of the user. However, since the SCS is commercial and cannot be fully trusted, it is necessary for the user to check whether the SCS has performed the partial signature computation correctly, and this check is better to operate on the spot. We address this problem by requiring the SCS to generate an auxiliary witness for the user to verify the validity of the partial signature. The verification is passed only when the partial signature is correctly generated. Additionally, to reduce the computational overhead of the user, we let the SCS do most of the computations required in the verification and output the corresponding results as the witness.

We will present the details of each algorithm. Let the Lagrange coefficients be as that in definition 4. Then the proposed ODMA-ABS scheme is shown by the following five algorithms as follows.

- **Setup**(λ, N): On input the security parameter λ, and a common reference string (CRS), the N attribute authorities generate an admissible asymmetric bilinear group denoted by $e = (p, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}(\cdot,\cdot), \phi(\cdot))$ and two collision resistant Hash functions (CRHF) $H : \{0,1\}^* \to \mathbb{Z}_p$ and $H' : \{0,1\}^* \to \mathbb{G}_2$. The former maps user's global identity GID to an element in $\mathbb{Z}_p$ (denoted by u, i.e. $H(GID) = u$) and the latter maps the message to be signed to an element in $\mathbb{G}_2$. Select an admissible $G_2 \in_R \mathbb{G}_2$ among the authorities. Next, for each attribute authority $k \in \{1, \cdots, N\}$, redefine the attributes in universe $\{\mathcal{U}_k\}_{k\in\{1,\cdots,N\}}$ as elements in $\mathbb{Z}_p$, and define a d_k-element default attribute set C_k. Choose a CRHF $H_k : \{0,1\}^* \to \mathbb{G}_2$, which maps each attribute i to the element in $\mathbb{G}_2$. Select $x_k, v_k \in_R \mathbb{Z}_p$ and compute $y_k = G_2^{x_k}, Y_k = g_1^{v_k}, Z_k = \hat{e}(Y_k, G_2)$.
  Next the authorities pairwise engage a two party key exchange protocol such that authority k and j share a unique seed $s_{kj} \in \mathbb{Z}_p$, which is only known to them two but not to any other authority $i \notin \{k, j\}$. Specially, define $s_{kj} = s_{jk}$. Then the pseudorandom function between authority k and j for user u is defined as

$$\text{PRF}_{kj}(u) = G_2^{x_k x_j/(s_{kj}+u)}, u \in \mathbb{Z}_p \tag{4}$$

  Finally, the public parameter

$$PP = (\{y_k, Y_k, Z_k, H_k, d_k\}_{k\in\{1,\cdots,N\}}, H, H', e, G_2),$$

  and the master secret key

$$MSK = (\{x_k, v_k, \{s_{kj}\}_{j\in\{1,\cdots,N\}\setminus\{k\}}\}_{k\in\{1,\cdots,N\}}).$$

- **AKeyGen**(PP, MSK, A_u): To generate the key on attribute set $A_u = \bigcup_{k\in\{1,\cdots,N\}} A_{k,u}$, user u interacts with authority k that works as follows.
  1) For $j \in \{1, \cdots, N\}\setminus\{k\}$, let

$$g = y_j^{x_k}, h = G_2, \alpha = \delta_{kj} R_{kj}, \beta = s_{kj}, \gamma = \delta_{kj},$$

     where R_{kj} is selected randomly from $\mathbb{Z}_p$ by k, $\delta_{kj} = 1$ if $k > j$ and $\delta_{kj} = -1$ otherwise. Then $N - 1$ times of anonymous key issuing protocols between u and k are executed independently. As a consequence, u gets

$$D_{kj} = G_2^{R_{kj}} \text{PRF}_{kj}(u), k > j$$
$$D_{kj} = G_2^{R_{kj}} / \text{PRF}_{kj}(u), k < j$$

  2) Authority k randomly selects a $d_k - 1$ degree polynomial $p_k(\cdot)$ such that

$$p_k(0) = v_k - \sum_{j\in\{1,\cdots,N\}\setminus\{k\}} R_{kj} \tag{5}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

and for each attribute $i \in A_{k,u} \cup C_k$ and each $r_{k,i}$ randomly selected from $\mathbb{Z}_p$, k computes

$$d_{k,i0} = G_2^{p_k(i)} H_k(i)^{r_{k,i}}, d_{k,i1} = g_1^{r_{k,i}}.$$

3) Authority k outputs the user's outsourcing key

$$OK_k = (\{d_{k,i0}, d_{k,i1}\}_{i \in A_{k,u} \cup C_k}),$$

and the user's private key

$$SK_k = (\{D_{kj}\}_{j \in \{1,\cdots,N\} \setminus \{k\}}).$$

Finally, the user's outsourcing keys and private keys are

$$OK = \{OK_k\}_{k \in \{1,\cdots,N\}}, SK = \{SK_k\}_{k \in \{1,\cdots,N\}}$$

- **Sign**$_{\text{out}}(OK, A_u, \Upsilon_{n,C^*})$: For an outsourced signing query from user u, with $OK = \{OK_k\}_{k \in \{1,\cdots,N\}}$, $A_u = \bigcup_{k \in \{1,\cdots,N\}} A_{k,u}$ and predicate $\Upsilon_{n,C^*}(\cdot)$, where $C^* = \bigcup_{k \in \{1,\cdots,N\}} C_k^*$, $n = \sum_{k \in \{1,\cdots,N\}} n_k$, $|C_k^*| = m_k$ and n_k could be 0, the SCS works as follows.
  1) For each $k \in \{1, \cdots, N\}$, select a random n_k-element attribute subset $A'_k \subseteq A_{k,u} \cap C_k^*$, and a $(d_k - n_k)$-element default attribute subset $C'_k \subseteq C_k$. Define $S_k = A'_k \cup C'_k$, and then $|S_k| = d_k$. The explanation of these different kinds of attribute sets managed by authority k is listed together in Table III for better understanding.

TABLE III
EXPLANATION OF DIFFERENT SETS MANAGED BY AUTHORITY k

$\mathcal{U}_k$	the universe set
C_k	the d_k-element default set chosen by authority k
$A_{k,u}$	attribute subset that belongs to user u
C_k^*	the m_k-element candidate attribute set in the predicate
A'_k	$A'_k \subseteq A_{k,u} \cap C_k^*$, $\lvert A'_k \rvert = n_k$
C'_k	$C'_k \subseteq C_k$, $\lvert C'_k \rvert = d_k - n_k$
S_k	$S_k = A'_k \cup C'_k$, $\lvert S_k \rvert = d_k$

  2) Select $m_k + d_k - n_k$ random values $s_{k,1}, \cdots, s_{k,m_k+d_k-n_k}$ from $\mathbb{Z}_p$, and compute the following items

$$\sigma'_0 = \prod_{1 \le k \le N} (\prod_{i \in S_k} d_{k,i0}^{\Delta_{i,S_k}(0)} \prod_{i \in C_k^* \cup C'_k} H_k(i)^{s_{k,i}})$$

$$\sigma'_{k,i} = \begin{cases} d_{k,i1}^{\Delta_{i,S_k}(0)} g_1^{s_{k,i}}, & i \in S_k \\ g_1^{s_{k,i}}, & i \in C_k^* \backslash A'_k \end{cases}$$

  3) Compute the following terms in addition.

$$W_1 = \hat{e}(g_1, \sigma'_0), W_2 = \prod_{1 \le k \le N} [\prod_{i \in C_k^* \cup C'_k} Z_k \hat{e}(\sigma'_{k,i}, H_k(i))]$$

Finally, output the partial signature together with the corresponding witness

$$\sigma_{\text{part}} = (\sigma'_0, \{\{\sigma'_{k,i}\}_{i \in C_k^* \cup C'_k}\}_{k \in \{1,\cdots,N\}}), W = (W_1, W_2).$$

- **Sign**$(SK, m, \sigma_{\text{part}}, W, \Upsilon_{n,C^*})$: To generate the formal signature after receiving σ_{part} from SCS and owning $(SK, m, \Upsilon_{n,C^*})$, u works as follows.
  1) Compute

$$D_u = \prod_{(k,j) \in \{1,\cdots,N\} \times \{1,\cdots,N\} \setminus \{k\}} D_{k,j}$$

  2) Check if the following equation holds.

$$\hat{e}(g_1, D_u) \cdot W_1 \stackrel{?}{=} W_2 \quad (6)$$

  3) If the above equation holds, then select a random value s from $\mathbb{Z}_p$ and compute

$$\sigma_0 = D_u \cdot \sigma'_0 \cdot H'(m \| \Upsilon_{n,C^*})^s$$
$$\sigma_s = g_1^s, \qquad \sigma_{k,i} = \sigma'_{k,i}$$

Finally, output the final signature as

$$\sigma = (\sigma_0, \sigma_s, \{\{\sigma_{k,i}\}_{i \in C_k^* \cup C'_k}\}_{k \in \{1,\cdots,N\}})$$

- **Verify**$(PP, \sigma, m, \Upsilon_{n,C^*})$: On input the signature σ, the message m with predicate Υ_{n,C^*} and let $v = H'(m \| \Upsilon_{n,C^*})$, the verification is processed by checking the following equation

$$\hat{e}(g_1, \sigma_0) \stackrel{?}{=} \prod_{1 \le k \le N} [\prod_{i \in C_k^* \cup C'_k} Z_k \hat{e}(\sigma_{k,i}, H_k(i))] \cdot \hat{e}(\sigma_s, v) \quad (7)$$

Output 1 and accept the signature if the above equation holds; otherwise 0 and reject the signature.

Remark: The verification of equation 6 is an extension compared with the scheme in the conference version. Note that by multiplying $\hat{e}(g_1, v)$ on each side of equation 6 we get equation 7. Thus the correctness of this verification procedure is guaranteed. Also, since W_1, W_2 are generated from the partial signature and public parameter, thus they will not affect the security of the protocol. In addition, the verification of partial signature only requires one pairing operation and one multiplication operation. So this does not affect the efficiency of the whole scheme.

C. Security Analysis

We present analysis of correctness, unforgeability, attribute privacy, efficiency respectively, and outsourcing security and privacy.

1) Correctness: Correctness is shown in the following theorem.

Theorem 1. *The proposed ODMA-ABS scheme is correct.*

Proof. Let $D_u = \prod_{(k,j) \in \{1,\cdots,N\} \times \{1,\cdots,N\} \setminus \{k\}} D_{k,j}$, $S_k = A'_k \cup c'_k$ and $v = H'(m \| \Upsilon_{n,C^*})$. The correctness of our proposed scheme is proved by the following equation.

$$\begin{aligned} \hat{e}(g_1, \sigma_0) &= \hat{e}(g_1, D_u \cdot \prod_k (\prod_{i \in S_k} d_{k,i0}^{\Delta_{i,S_k}(0)} \prod_{i \in C_k^* \cup C'_k} H_k(i)^{s_{k,i}}) v^s) \\ &= \hat{e}(g_1, g_2^{\sum_k v_k} \prod_k (\prod_{i \in S_k} H_k(i)^{r_{k,i} \Delta_{i,S_k}(0)} \prod_{i \in C_k^* \cup C'_k} H_k(i)^{s_{k,i}}) v^s) \\ &= \hat{e}(g_1, g_2^{\sum_k v_k}) \cdot \prod_k (\prod_{i \in C_k^* \cup C'_k} \hat{e}(\sigma_{k,i}, H_k(i))) \cdot \hat{e}(g_1, v^s) \\ &= \prod_{1 \le k \le N} [\prod_{i \in C_k^* \cup C'_k} Z_k \hat{e}(\sigma_{k,i}, H_k(i))] \cdot \hat{e}(\sigma_s, v) \end{aligned} \quad (8)$$

□

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

2) Unforgeability: The ODMA-ABS protocol achieves unforgeability. We have the following theorem.

Theorem 2. *The proposed ODMA-ABS protocol is existentially unforgeable under co-CDH assumption.*

Proof. The intuition behind the reduction to co-CDH problem is as follows: According to the unforgeability definition, given a co-CDH tuple (g_1^a, g_2^b), We expect to construct a simulator that solves the co-CDH problem by utilizing an adversary who is able to forge the ODMA-ABS as a subroutine. The simulator needs to simulate the response of the attribute authority when receiving a query from the adversary and after that gives a co-CDH solution from the signature that the adversary forges. Note that in the multi-authority setting, the adversary is allowed to corrupt a number of attribute authorities. Here we prove the case when $N-1$ attribute authorities are corrupted. Since the corrupted authorities are all set before setup stage, the simulator only needs to simulate the honest authority to answer the query of the adversary. The specific procedure is presented below.

Let $\mathcal{F}$ be an adversary that has a non-negligible advantage ϵ in breaking the unforgeability of the proposed ODMA-ABS protocol. And suppose that $\mathcal{F}$ makes at most $q_{H_k}, q_{H'}, q_P, q_O$ and q_S times of queries to the hash functions H_k (of authority k), H', the private key generation oracle, the outsourcing key generation oracle and the signing oracle respectively. Given a co-CDH tuple $(g_1^a, g_2^b, \hat{e})$ and to compute g_2^{ab}, the simulator $\mathcal{S}$ executes the simulation as follows.

- **Initial** $\mathcal{S}$ receives a corrupted authorities set $\{1, \cdots, N\}\backslash\{k\}$ and a challenge predicate Υ_{n_k, C_k^*}. Here the predicate is restricted for only authority k since other authorities are corrupted and can generate the corresponding signature for any predicate.
- **Setup** Let the default attribute set denoted by C_k for authority k. $\mathcal{S}$ selects a $d_k - n_k$ subset $C_k' \subseteq C_k$ and publishes $G_1 = g_1^a, G_2 = g_2^b$ to $\mathcal{F}$. k picks $x_k \in_R \mathbb{Z}_p$ and interacts with other $k-1$ authorities and shares the secret seeds s_{kj}. Publish $y_k = G_2^{x_k}$ 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_k-query*. $\mathcal{S}$ maintains a list $\mathcal{L}_1$ to store the answers to the hash oracle H_k. On receiving a query i, $\mathcal{C}$ first checks the list $\mathcal{L}_1$ and returns the corresponding answer if the same value has been queried. Otherwise, $\mathcal{S}$ simulates as follows.
    1) If $i \in C_k^* \cup C_k'$, it chooses a $\beta_{k,i} \in \mathbb{Z}_p$ and answers $H_k(i) = g_2^{\beta_{k,i}}$.
    2) If $i \notin C_k^* \cup C_k'$,it chooses $\alpha_{k,i}, \beta_{k,i} \in \mathbb{Z}_p$ and answers $H_k(i) = {G_2}^{\alpha_{k,i}} g_2^{\beta_{k,i}}$

    After returning $H_k(i)$, $\mathcal{S}$ adds the tuple $(i, H_k(i))$ onto the list $\mathcal{L}_1$.
  - *H'-query*. $\mathcal{S}$ picks a value $\delta \in_R \{1, \cdots, q_{H'}\}$ and maintains a list $\mathcal{L}_2$ to store the answers to the hash oracle H'. On receiving the l-th query $m_l \parallel \Upsilon_{n_l, C_l^*}$ for $1 \le l \le q_2$ and $1 \le n_{l,k} \le d_k$, where $n_l = \sum_{k \notin \mathbb{K}_c} n_{l,k}$, $\mathcal{S}$ first checks the list $\mathcal{L}_2$ and returns the corresponding answer if the same value has been queried. Otherwise, $\mathcal{S}$ simulates as follows.
    1) If $l = \delta$, it chooses a $\beta'_{k,\delta} \in \mathbb{Z}_p$ and answers $H'(m_l \parallel \Upsilon_{n_l, C_l^*}) = g_2^{\beta'_{k,\delta}}$.
    2) If $l \neq \delta$,it chooses $\alpha_{k,l}, \beta_{k,'l} \in \mathbb{Z}_p$ and answers $H'(m_l \parallel \Upsilon_{n_l, C_l^*}) = {G_2}^{\alpha'_{k,l}} g_2^{\beta'_{k,l}}$

    After returning $H'(m_l \parallel \Upsilon_{n_l, C_l^*})$, $\mathcal{S}$ adds the tuple $(m_l \parallel \Upsilon_{n_l, C_l^*}, H'(m_l \parallel \Upsilon_{n_l, C_l^*}))$ onto the list $\mathcal{L}_2$.
  - *Private key query*. On receiving a secret key query of identity u, $\mathcal{S}$ first sets $U = U \cup \{u\}$ and maintains a list $\mathcal{L}_3$. Then it checks if (u, SK) exists in L. If so, return SK. Otherwise, $\mathcal{S}$ performs the same operation as that in ODMA-ABS protocol. This is because the secret key set $\{x_k\}$ is not invalid. As a consequence, $\mathcal{F}$ receives $D_{kj} = G_2^{R_{kj}} \text{PRF}_{kj}(u)$ for $k > j$ and $D_{kj} = G_2^{R_{kj}} / \text{PRF}_{kj}(u)$ for $k < j$, where R_{kj} is randomly selected from $\mathbb{Z}_p$ by $\mathcal{C}$. Finally, after returning SK, $\mathcal{S}$ adds the tuple (u, SK) onto the list $\mathcal{L}_3$.
  - *Outsourcing key query*. On receiving an outsourcing key query of attribute set A_k, $\mathcal{S}$ sets $j = j + 1$ and maintains a list $\mathcal{L}_4$. Then $\mathcal{S}$ simulates as follows.
    1) If $|A_k \cap C_k^*| < n_k$, $\mathcal{S}$ chooses three sets Γ, Γ', S such that $\Gamma = (A_k \cap C_k^*) \cup C_k'$, $|\Gamma'| = d_k - 1$and $\Gamma \subseteq \Gamma' \subseteq A_k \cup C_k'$, $S = \Gamma' \cup \{0\}$. Here we assume $|A_k| > n_k$. Then for $i \in \Gamma'$, $\mathcal{S}$ chooses $\tau_{k,i}, r_k, i \in_R \mathbb{Z}_p$ and simulates
    $$\begin{cases} d_{k,i0} = \phi(G_1)^{\tau_{k,i}} H_k(i)^{r_{k,i}} \\ d_{k,i1} = g_1^{r_{k,i}} \end{cases}$$
    For $i \in (A_k \cup C_k)\backslash\Gamma'$, let $R_k = \sum_{j \in \{1,\cdots,N\}\backslash\{k\}} R_{kj}$, $r_{k,i} = r'_{k,i} - \frac{a\Delta_{i,S}(0)}{\alpha_{k,i}}$ where $r'_k, i \in_R \mathbb{Z}_p$ and simulate
    $$\begin{cases} d_{k,i0} = \phi(G_1)^{\sum_{j \in \Gamma'} \tau_{k,j}\Delta_{j,S}(i) - (R_k + \frac{\beta_{k,i}}{\alpha_{k,i}})\Delta_{0,S}(i)} \cdot \\ \qquad {G_2}^{\alpha_{k,i} r'_{k,i}} g_2^{\beta_{k,i} r'_{k,i}} \\ d_{k,i1} = {G_1}^{-\frac{\Delta_{0,S}(i)}{\alpha_{k,i}}} g_2^{r'_{k,i}} \end{cases}.$$
    2) If $|A_k \cap C_k^*| \ge n_k$, $\mathcal{C}$ chooses a_1 and a random polynomial $p_k(\cdot)$ with $p_k(0) = a_1$ and simulates OK_k for $i \in A_k \cup C_k$ as
    $$(d_{k,i0}, d_{k,i1}) = (\phi(G_1)^{p_k(\cdot)} H_k(i)^{r_{k,i}}, g_1^{r_{k,i}})$$
    where $r_{k,i} \in_R \mathbb{Z}_p$.
  - *Signing Query*. On receiving a signing query on (m, Υ_{n', C^*}), $\mathcal{S}$ first simulates the outsourced signing phase for corrupted authorities $j \in \{1, \cdots, N\}\backslash\{k\}$. Since all keys corresponding to corrupted authorities are well prepared, the simulation of $\sigma_{j,0}$ and $\sigma_{j,i}$ executes just the same as that in the outsourced signing phase of ODMA-ABS protocol, where $\sigma_{j,0}$ represents the factors to construct σ'_0 except the part generated from authority k's OK. To simulate the part of k's with query (m, Υ_{n', C^*}), $\mathcal{C}$ checks whether $|A_k \cap C_k^*| < n_k$. If so, $\mathcal{S}$ executes the above private key query and outsourcing key query once again and generates a signature with other

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

$k-1$ part normally as that in the protocol. Otherwise, i.e. $|A_k \cap C_k^*| \geq n_k$, if $H'(m \parallel \Upsilon_{n,C^*}) = g_2^{\beta'_{k,\delta}}$, the simulation is aborted. Otherwise, assume that $H'(m \parallel \Upsilon_{n,C^*}) = G_2^{\alpha'_{k,l}} g_2^{\beta'_{k,l}}$, then $\mathcal{S}$ selects a n'_k-element subset $\hat{C}^* \subseteq C_k^*$ and a $(d_k - n'_k)$-element subset $C''_k \subseteq C_k$. Let $s_{k,i} = s'_{k,i} - \frac{a}{\alpha'_{k,l}}$, $S_k = \hat{C}^* \cup C''_k$ and the signature is simulated as follows:

$$\begin{aligned}
\sigma_0 = & \prod_{j\in\{1,\cdots,N\}\setminus\{k\}} \sigma_{j,0} \prod_{i\in S_k} [H_k(i)^{r_{k,i}\Delta_{i,S}(0)}] \\
& \prod_{i\in C_k^*\cup C''_k} [H_k(i)^{s_{k,i}}] G_2^{\alpha'_{k,l}s'_{k,l}} \phi(G_1)^{-\frac{\beta'_{k,l}}{\alpha'_{k,l}}} g_2^{\beta'_{k,l}s'_{k,l}} \\
\sigma_{k,i} = & \begin{cases} g_1^{r_{k,i}\Delta_{i,S_k}(0)+s_{k,i}}, & i \in S_k \\ g_1^{s_{k,i}}, & i \in C_k^*\setminus\hat{C}^* \end{cases} \\
\sigma_s = & G_1^{-\frac{1}{\alpha'_{k,l}}} g_1^{s'}
\end{aligned}$$

where $s', r_{k,i}, s_{k,t} \in_R \mathbb{Z}_p$ for $i \in \hat{C}^* \cup C''_k$, $t \in C^* \cup C''_k$ and $\{\sigma_{j,i}\}_{j\in\{1,\cdots,N\}\setminus\{k\}}$ remain the same as that in ODMA-ABS protocol. Finally, $\mathcal{S}$ returns simulated signature $(\sigma_0, \sigma_s, \{\{\sigma_{j,i}\}_{i\in C_j^*\cup C''_j}\}_{j\in\{1,\cdots,N\}})$ to $\mathcal{F}$. Here we claim that σ_0 is correctly simulated based on the following equation

$$\begin{aligned}
\sigma_0 = & \prod_{j\in\{1,\cdots,N\}\setminus\{k\}} \sigma_{j,0} \prod_{i\in S_k} [H_k(i)^{r_{k,i}\Delta_{i,S_k}(0)}] \\
& \prod_{j\in C_k^*\cup C''_k} [H_k(i)^{s_{k,i}}] G_2^{\alpha'_{k,l}s'_{k,l}} \phi(G_1)^{-\frac{\beta'_{k,l}}{\alpha'_{k,l}}} g_2^{\beta'_{k,l}s'_{k,l}} \\
= & \phi(G_1)^b \prod_{j\in\{1,\cdots,N\}\setminus\{k\}} \sigma_{j,0} \prod_{i\in S_k} [H_k(i)^{r_{k,i}\Delta_{i,S_k}(0)}] \\
& \prod_{i\in C_k^*\cup C''_k} [H_k(i)^{s_{k,i}}] H'(m \parallel \Upsilon_{n',C^*})^s \\
= & \phi(G_1)^{R_k} \prod_{j\in\{1,\cdots,N\}\setminus\{k\}} \sigma_{j,0} \prod_{i\in S_k} [d_{k,i0}^{\Delta_{i,S_k}(0)}] \\
& \prod_{i\in C_k^*\cup C''_k} [H_k(i)^{s_{k,i}}] H'(m \parallel \Upsilon_{n',C^*})^s \\
= & D_u \prod_{1\leq j\leq N} [\prod_{i\in S_j} d_{j,i0}^{\Delta_{i,S_j}(0)} \prod_{i\in C_j^*\cup C''_j} H_j(i)^{s_{j,i}}] \\
& H'(m \parallel \Upsilon_{n',C^*})^s
\end{aligned} \tag{9}$$

- **Forgery** After performing the query phase, $\mathcal{F}$ outputs a forged signature σ^* on message m^* with predicate Υ_{n,C^*}. If the associated default attribute set is not C'_k or $H'(m \parallel \Upsilon_{n',C^*}) \neq g_2^{\beta'_{k,\delta}}$, $\mathcal{S}$ will abort. Otherwise the forged signature passes the verification algorithm which means,

$$\begin{aligned}
\hat{e}(g_1, \sigma_0^*) = & \prod_{1\leq j\leq N} [\prod_{i\in C_j^*\cup C'_j} Y_j \hat{e}(\sigma_{j,i}^*, H_j(i))] \\
& \hat{e}(\sigma_s^*, H'(m^* \parallel \Upsilon_{n',C^*})) \\
= & \prod_{1\leq j\leq N} [\prod_{i\in C_j^*\cup C'_j} \hat{e}(\sigma_{j,i}^*, H_j(i))] \\
& \prod_{1\leq j\leq N} Z_j \hat{e}(\sigma_s^*, H'(m^* \parallel \Upsilon_{n',C^*})) \\
= & \prod_{1\leq j\leq N} [\prod_{i\in C_j^*\cup C'_j} \hat{e}(\sigma_{j,i}^*, g_2^{\beta_{j,i}})] \\
& \hat{e}(G_1, G_2) \prod_{j\in\{1,\cdots,N\}\setminus\{k\}} Z_j \hat{e}(\sigma_s^*, g_2^{\beta'_{k,\delta}}))
\end{aligned} \tag{10}$$

Then $\mathcal{S}$ can compute g_2^{ab} as,

$$g_2^{ab} = \frac{\sigma_0^*}{\prod_{j\in\{1,\cdots,N\}\setminus\{k\}} G_2^{v_j} \prod_{i\in C_k^*\cup C'_k} (\phi(\sigma_{k,i}^*)^{\beta_{k,i}}) \phi(\sigma_s{}^*)^{\beta'_{k,\delta}}} \tag{11}$$

Assume that $\mathcal{F}$ breaks the unforgeability of the proposed ODMA-ABS protocol in t time with probability ϵ. Then we can build a simulator to solve the co-CDH problem in t' time with probability ϵ', where $t' \approx (\sum_k q_{H_k} + q_{H'} + N(N-1)q_P + 2|\overline{A_O}|q_O + \frac{3}{2}\sum_k(|\overline{C_k^*}| + d_k - \overline{n_k})q_S)t_e$ and $\epsilon' = \frac{\epsilon}{q_{H'}\binom{d_k-n_k}{d_k-1}}$. Here $\overline{A_O}$ represents the average number of attributes in queried set in *outsourcing key query* respectively. $\overline{C_k^*}$ and $\overline{n_k}$ represent the average values of the predicate parameters in *signing query*. $\frac{1}{q_{h'}}$ and $\frac{1}{\binom{d_k-n_k}{d_k-1}}$ represent the probabilities of which the cases "the associated default attribute set is C'_k and $H'(m, \Upsilon_{n'_k,C_k^*}) = g_2^{\beta'_{k,\delta}}$" happen respectively. What is more, t_e represents the time to perform a single-based exponentiation operation in $\mathbb{G}_1$ or $\mathbb{G}_2$, and we assume without loss of reasonability that one multi-based exponentiation which multiples up to 2 single-based exponentiation takes roughly the same as a single-based exponentiation [23]. □

3) Attribute privacy: The ODMA-ABS also achieves attribute privacy.

Theorem 3. *The proposed ODMA-ABS protocol is attribute private.*

Proof. The proof of the theorem is simple. In fact, the core of our construction is that any of the values of d_k attribute points can reconstruct the polynomial by Lagrange interpolation. And we simply take n_k attributes from the user's attribute set (also belong to the predicate attribute set) and $d_k - n_k$ attributes from the default attribute set. Thus, it is obvious that for attribute sets A_{u0} and A_{u1} that both satisfy the predicate, they both catch such d_k elements that together with some $d_k - n_k$ elements from the default attribute set can retrieve the value of the polynomial at point 0 and furthermore generate a valid signature. This indicates that the valid signature do not reveal which attribute set it takes use of. Assume without loss of generality that $n_k = m_k$, we present the formal proof below.

- **Setup** The challenger $\mathcal{C}$ chooses a security parameter λ and runs setup algorithm in the ODMA-ABS scheme to generate

  $$PP = (\{y_k, Y_k, Z_k, H_k, d_k\}_{k\in\{1,\cdots,N\}}, H, H', \hat{e}, G_2),$$
  $$MSK = (\{x_k, v_k, \{s_{kj}\}_{j\in\{1,\cdots,N\}\setminus\{k\}}\}_{k\in\{1,\cdots,N\}}).$$

  It then publishes theme to adversary $\mathcal{F}$.
- **Query** $\mathcal{F}$ is provided with access to *private key oracle*, *outsourcing key oracle* and *signing oracle*. Since the master secret key is given to adversary. The simulation of the oracle response is obvious. We omit it here.
- **Challenge** After querying phase, $\mathcal{F}$ outputs a tuple $(A_{u0}, A_{u1}, \Upsilon_{n,C^*}, m)$, where $\Upsilon_{n,C^*}(A_{ui}) = 1$ for $(i = 0, 1)$, i.e. $C^* \subset A_{ui}$ for $i = 0, 1$. The tuple is then sent to $\mathcal{C}$. $\mathcal{C}$ randomly picks a bit $b \in \{0, 1\}$ to determine which attribute set is used to generate the signature. After that $\mathcal{C}$ chooses a $(d_k - n_k)$-element default attribute

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

subset $C'_k \subseteq C_k$ for each $k \in \{1, \cdots, N\}$ and runs the Sign_{out} and Sign algorithms accordingly. Consequently, the challenging signature σ^* is generated and sent to $\mathcal{F}$.

- **Guess** $\mathcal{F}$ outputs a bit b' as a guess to which attribute is used to generate the challenging signature.

Note that in the **Challenge** phase, since $C^* \subset A_{ui}$ for $i = 0, 1$, we have $A'_k \subseteq A_{k,ui} \cap C^*_k = C^*_k$ for $i = 0, 1$.With the assumption that $n_k = m_k$ and the fact that $|A'_k| = n_k$, we have $A'_k = C^*_k$. Thus after running the Sign_{out} and Sign algorithms, the challenging signature σ^* is of the following form, by letting $A'_k = C^*_k$.

$$\sigma^* = (\sigma_0, \sigma_s, \{\{\sigma_{k,i}\}_{i \in C^*_k \cup C'_k}\}_{k \in \{1, \cdots, N\}}),$$

where

$$\sigma_0 = \prod_{1 \le k \le N} G_2^{v_k} \Big(\prod_{i \in C^*_k \cup C'_k} H_k(i)^{r_{k,i} \Delta_{i,S_k}(0) + s_{k,i}} \Big) \cdot H'(m \| \Upsilon_{n,C^*})^s, \tag{12}$$

and

$$\sigma_{k,i} = g_1^{r_{k,i} \Delta_{i,S_k}(0) + s_{k,i}}, \text{ for } i \in C^*_k \cup C'_k, k \in \{1, \cdots, N\}. \tag{13}$$

$\sigma_s = g_1^s$ remains the same as that in the ODMA-ABS scheme.

Based on the Lagrange interpolation function, it is easy to see that the challenging signature σ^* can be generated by secret keys produced from either attribute set A_{u0} or A_{u1}. In another word, any signature generated by secret keys produced from A_{u0} can also be generated by secret keys produced from A_{u1}. Thus given a valid signature and without extra information, the adversary is not able to distinguish which set of A_{ui} ($i \in \{0, 1\}$) is used to generate it with a probability better than $\frac{1}{2} + negl(\lambda)$, which means the attribute privacy for signer holds. □

4) Efficiency: The ODMA-ABS protocol achieves the efficiency requirement of an outsourcing computation scheme.

Theorem 4. *The proposed ODMA-ABS protocol is efficient.*

Proof. It is easy to see that without SCS, the extra computations that the signer needs to operate are the same as that in the **Sign_{out}** and the computation that the sigher saves is just the checking of equation 6. It is obvious that the extra computations are far more expensive than the saved one. This indicates that the efficiency is satisfied. More details of the simulated experiment will be illustrated to show the efficiency in section IV-D. □

5) Outsourcing security and privacy: As an outsourcing computation protocol, our ODMA-ABS also achieves the property of outsourcing security. In an outosurcing protocol, security implies that an untrusted worker (i.e. cloud server) is unable to make the user accept an incorrect result that it outputs with non-negligible probability. And for ODMA-ABS, this means that an untrusted server, if not honestly executing the **Sign_{out}** algorithm, is unable to output a partial signature that passes the user's verification (i.e. equation 6) with non-negligible probability. To prove that outsourcing security holds for ODMA-ABS, we notice that the verification of outsourcing procedure (i.e. equation 6) is equivalent to the verification of final signature (i.e. equation 7). So we would like to reduce the outsourcing security to ODMA-ABS unforgeabilility. We claim that an adversarial cloud server, who has the outsourcing key that is related to singer's attribute set A_u, is not able to forge a valid signature. Consider the following two cases.

(i) The cloud server would like to forge a signature for some predicate Υ that satisfies $\Upsilon(A_u) = 0$. This is the same scenario as that in the ungorgeability experiment, with the difference that the adversary is unable to access the *private key oracle*. According to Theorem 2, the adversary cannot forge a valid signature, even if it is able to access the *private key oracle*.

(ii) The cloud server would like to forge a signature for the predicate $\hat{\Upsilon}$ that satisfies $\hat{\Upsilon}(A_u) = 1$. Notice that this is no longer identical to the unforgeability experiment. To prove the unforgeability in this situation, we consider a multi-authority ABS protocol. In the new MA-ABS, a central authority (indicated by authority 0) holds a secret key SK that is identical to the SK in ODMA-ABS. This SK is associated with the attribute that is unique for each user. If we mark this attribute as i_u then the user's attribute set will become $A'_u = A_u \cup i_u$. Also, the central authority allocates different master secret keys to each attribute authority k ($k \in \{1, \cdots, N\}$). The authority k uses its unique master key to generate attribute private key that is identical to OK_k in ODMA-ABS. Other algorithms in MA-ABS remains the same as that in ODMA-ABS. Such MA-ABS is equivalent to the one proposed in [8]. If we consider the same adversarial behavior in this MA-ABS, it will be forging a valid signature for the predicate $\hat{\Upsilon}$ that satisfies $\hat{\Upsilon}(A_u) = 0$. With a similar method in [8] and the unforgeability proof of ODMA-ABS, it is easy to prove the unforgeability of the corresponding MA-ABS protocol.

Above all, the security of the outsourcing procedure is proved.

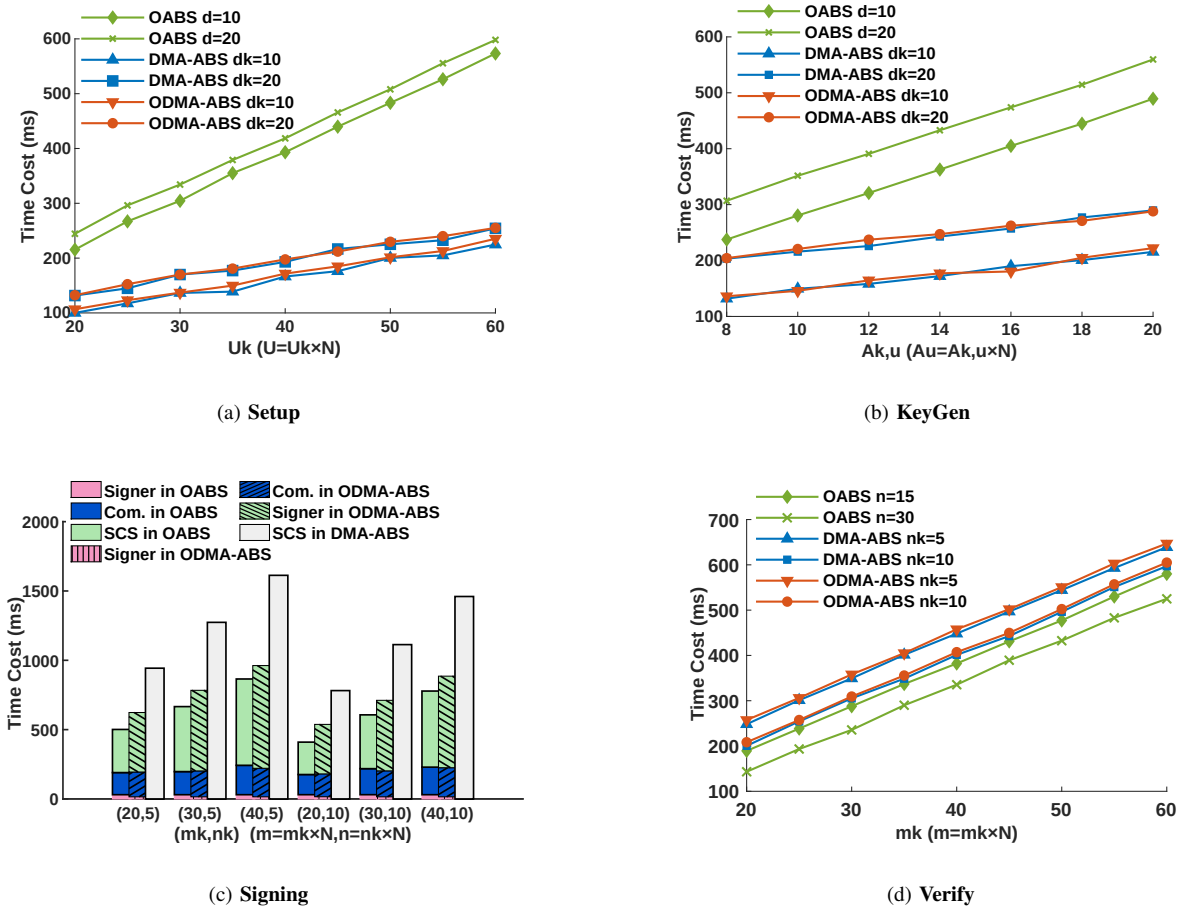
As to privacy, the cloud server gets the outsourcing keys of the user. However, these outsourcing keys include nothing that is relative to the user's identity u. u is embedded in SK that is kept secret by the user. So the privacy of the user's identity is preserved.

D. Performance

To evaluate the performance of the proposed scheme, we present theoretical analysis of storage complexity and experimental simulation of computation and communication overhead.

1) Theoretical analysis of storage complexity: For the signer, the storage overhead involves private keys SK and outsourcing keys OK that are returned by attribute authorities, partial signature σ_{part} that is returned by the signing cloud server, and the final signature σ. To be specific, let $|G|$ represent the length of the element in the group used in ODMA-ABS protocol, i.e. $|G| = poly(\lambda)$. We show a comparison between ODMA-ABS and chen et al.'s OABS [23] with respect to key size and signature size in Table IV. In the table, $i_{k,u} = |A_{k,u}|$, d_k, n_k, m_k refer to the same as that in the ODMA-ABS protocol, and i_u, d, n, m refer to the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing



(a) **Setup**

(b) **KeyGen**

(c) **Signing**

(d) **Verify**

Fig. 2. Efficiency Comparison for Setup, KeyGen, Signing and Verify among OABS, ODMA-ABS and DMA-ABS

TABLE IV
COMPARISON WITH RESPECT TO KEY SIZE AND SIGNATURE SIZE

	ODMA-ABS	OABS[23]
SK size	$\|G\|$	$2\|G\|$
OK size	$2\sum_{k=1}^{N}(i_{k,u}+d_k)\|G\|$	$2(i_u+d-1)\|G\|$
σ_{part} size	$(1+\sum_{k=1}^{N}(m_k+d_k-n_k)N)\|G\|$	$(1+m+d-n)\|G\|$
σ size	$(2+\sum_{k=1}^{N}(m_k+d_k-n_k)N)\|G\|$	$(3+m+d-n)\|G\|$

TABLE V
CLOUD CONFIGURATION

physical id	: 0
cpu cores	: 14
processor	: 8
cpu model	: Intel(R) Xeon(R) Gold 6132 CPU @ 2.60GHz
memory	: 8 GB

corresponding $i_{k,u}, d_k, n_k, m_k$ in our multi-authority setting. From the table, it is easy to tell that the storage complexities of ODMA-ABS and OABS are both $O(poly(\lambda))$. Also, for the attributes of the same signer, we have $i_u = \sum_{k=1}^{N} i_{k,u}$. In practice, it is acceptable to have the values of n_k and m_k satisfying $n = \sum_{k=1}^{N} n_k$ and $m = \sum_{k=1}^{N} m_k$. So if we set proper values for d_k, we can get an ODMA-ABS that is comparable with OABS [23] in storage overhead.

2) Experimental Results for ODMA-ABS: In this part, we analyze the results of the experimental simulations for each algorithm in the proposed ODMA-ABS scheme. Notice that there are three entities involved in the experiment, the attribute authorities, the signer, and the signing cloud server. The attribute authorities are simulated on a Linux machine with Intel (R) Core (TM) i7-6550U CPU processor running at 2.50 GHz and 4 GB RAM; The signer is simulated on a virtual machine with Intel (R) Core (TM) i7-8550U CPU processor running at 1.80 GHz and 512 MB RAM; And the signing cloud server (SCS) is deployed in the public cloud provided by our institution with the following configuration shown in Table V. The bandwidth between user and cloud server is 200 Mbps. The **Verify** algorithm is simulated on a Linux machine with Intel (R) Core (TM) i7-6550U CPU processor running at 2.50 GHz and 4 GB RAM. When implementing each algorithm in the proposed protocol, C programming language with the Pairing Based Cryptography Library-0.5.14 (PBC Library-0.5.14) is used.

In the experiment, We implement three schemes, the OABS [23], the original DMA-ABS and the proposed ODMA-ABS. The original DMA-ABS is constructed simply in the following ways. (i) The **Setup**, **KeyGen** and **Verify** algorithms remain unchanged compared with ODMA-ABS. The same $\mathbf{Sign}_{\mathbf{out}}$ algorithm is performed by the signer instead of being outsourced to the SCS. (ii) The computation of terms W_1, W_2 and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

the verification of outsourcing results are no longer needed by either the SCS or the user. By comparing the original DMA-ABS with ODMA-ABS, we show a computation overhead reduction by means of outsourcing technology. And by comparing OABS [23] with ODMA-ABS, we show that the computation and communication overhead of the two schemes are comparable, while ODMA-ABS achieves better properties in security and privacy preservation. Now we give more details about the experiments of each algorithm. We fix the number of attribute authorities in DMA-ABS and ODMA-ABS $N = 3$.

The **Setup** algorithms are performed by the attribute authority(ies). In OABS [23], there is only one authority and we just need to simulate the deployment in this authority. When it comes to DMA-ABS and ODMA-ABS, there are multiple authorities. And each authority not only executes its own operation but also interacts with other ones. We claim that the internal operation of each authority can be executed in parallel. In the **Setup** phases of all the three schemes, the variables that affect the time cost are d (d_k) and $|\mathcal{U}|$ ($|\mathcal{U}_k|$), i.e. the orders of the default attribute set and the attribute universe respectively. It is reasonable to have $|\mathcal{U}| = \sum_k |\mathcal{U}_k|$ because multiple authorities just classify and manage different kinds of attributes, while not making the whole attributes universe increase. But it is improper to have $d = \sum_k d_k$ because the value of d_k is also associated with the security of the protocol and should not be much smallers than d. In this experiment, we let $d_k = d$. This may result in a little more storage burden for ODMA-ABS compared with OABS [23]. We present a time cost comparison of **Setup** algorithms among three schemes in Fig 2(a) by choosing $d_k = 10, 20$ respectively, and $|\mathcal{U}_k|$ varying from 20 to 60. The result shows that even when we set $d = d_k$, which makes $\sum_k d_k > d$, our ODMA-ABS still achieves a better performance than OABS. This is because in ODMA-ABS the deployment procedures inside all authorities are executed in parallel, and the interactions among authorities are much less time-consuming than the deployment procedures. Fig 2(a) also shows that the time costs of DMA-ABS and ODMA-ABS are almost the same. This is because the two **Setup** algorithms in DMA-ABS and ODMA-ABS remain the same.

In the **KeyGen** phase, the variables that affect the time cost become d (d_k) and $|A_u|$ ($|A_{k,u}|$). In the experiment, we also choose $d_k = d = 10, 20$ respectively, and $|A_{k,u}|$ varying from 8 to 20. Notice that $|A_u| = \sum_k |A_{k,u}|$ since the user's attribute set remains identical in different schemes. The result of time cost is shown in Fig 2(b). From the figure we can see that our ODMA-ABS achieves a better performance than OABS in **KeyGen** time. This is because in ODMA-ABS the attribute authorities can generate user's attribute in parallel. Also, the **KeyGen** algorithms in DMA-ABS and ODMA-ABS are still idnetical, which results in little difference of time cost between ODMA-ABS and DMA-ABS.

In the **Signing** phase, the user needs to interact with the signing cloud server (SCS). The total time cost in the whole procedure involves communication part and computation part. The variables that affect the communication cost are d (d_k) and $|A_u|$ ($|A_{k,u}|$) as is shown in Table IV. And the variables that affect the computation cost become n (n_k) and m (m_k), i.e. the threshold values and the orders of the attribute set in the predicate. In this experiment, we assume that the certain user has obtained the keys from the attribute authority(ies) and would like to generate a signature with respect to the predicate Υ_{n,C^*}, where $|C^*| = m$ and $n = \sum_{k=1}^{N} n_k$, $m = \sum_{k=1}^{N} m_k$. Under such circumstances the values of d (d_k) and $|A_u|$ ($|A_{k,u}|$) are fixed and the values of n (n_k) and m (m_k) are changeable. The simulation of the algorithms in three different schemes are conducted in several cases for $d = d_k = 10$, $|A_{k,u}| = 10$, $n_k = 5, 10$, $m_k = 20, 30, 40$ respectively. The result is shown in Fig 2(c), where "com." is short for "communication". Comparing the performance of ODMA-ABS with DMA-ABS, it is obvious to see that the total time cost to generate the signature is reduced by utilizing outsourcing technique, and the reduction is much more remarkable with respect to the user's overhead. This is consistent with our analysis in section IV-C4. Comparing the performance of ODMA-ABS with OABS, we can see that the performance on the signer side in ODMA-ABS is better than that in OABS. This is because in ODMA-ABS the signer just needs to multiply the secret key by σ_{part}, while the signer in OABS has to deal with another private key pair associated with the default attribute, which is much more time-consuming. However, due to the fact that $d < \sum_k d_k$, the communication and SCS's computation cost in ODMA-ABS are more than that in OABS. Actually, when we choose a smaller value for d_k to have $d = \sum_k d_k$, the performance of communication and SCS's computation cost in both schemes can become comparable, while the advantage of ODMA-ABS in signer's overhead still maintains. We consider it as a trade-off between efficiency and security.

Finally, in the **Verify** phase, the variables that affect the time cost are d (d_k),n (n_k) and m (m_k). More specifically, by observing equation 7 we can deduce that the time cost of **Verify** algorithm is proportional to the value $m + d - n$ ($m_k + d_k - n_k$). Similar with the assumption in the **Signing** phase, we fix $d_k = d = 10$ and varying the values of n_k and m_k with $n_k = 5, 10$, m_k from 20 to 60. The result is shown in Fig 2(d). We can see that the performance of ODMA-ABS and DMA-ABS are almost identical since the two **Verify** algorithms in both schemes remain the same. And the performance of OABS is better than the two decentralizing schemes. This is still because of the values we choose for d and d_k.

V. An Access Control Scheme Based on ODMA-ABS

In this section, we discuss the applications of ODMA-ABS that can be used in the IoT scenario. As introduced in previous section, (multi-authority) attribute based signature can provide anonymous certification of data and access control for IoT devices. To realize access control for IoT devices, we design an attribute based access control (ABAC) scheme that is derived from the idea of the proposed ODMA-ABS.

A. Construction of the Attribute Based Access Control scheme

There are usually three kinds of entities in the attribute based access control system: the server that stores resource,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2019.2902380, IEEE Transactions on Cloud Computing

the user who would like to access certain resource in the server and the attribute authorities that issue private keys for the user. We will design the ABAC scheme in decentralized multi-authority setting. In ABAC, the user can request attribute private keys from each attribute authority. Then the user generates corresponding proofs to show that his/her attributes satisfy certain predicate and sends them to the server for verification. One concern in this process is that the user may share some of his/her attribute private keys with others to let them be able to have the same access to certain resource. To address this, we introduce a unique secret key that is related to the user. The user needs to generate a proof of this secret key together with his/her attributes to prove his/her legitimate identity. This proof generation is easy to conduct in the asymmetric bilinear group. In addition, the user's computational overhead to generate the proof of the legitimate identity can be further reduced by outsourcing. Since the outsourcing technique is similar with that in the ODMA-ABS, it will be omitted in the following construction. Our ABAC scheme involves three phases that work as follows.

- **Setup**: On input the security parameter λ, and a common reference string (CRS), the N attribute authorities generate an admissible asymmetric bilinear group denoted by $e = (p, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}(\cdot,\cdot), \phi(\cdot))$ and two collision resistant Hash functions (CRHF) $H : \{0,1\}^* \to \mathbb{Z}_p$, $H' : \{0,1\}^* \to \mathbb{G}_2$. The hash functions are similar with that in the ODMA-ABS scheme. Select an admissible $G_2 \in_R \mathbb{G}_2$ among the authorities. For each attribute authority $k \in \{1, \cdots, N\}$, redefine the attributes in universe $\{\mathcal{U}_k\}_{k\in\{1,\cdots,N\}}$ as elements in $\mathbb{Z}_p$, and define a d_k-element default attribute set C_k. Choose a CRHF $H_k : \{0,1\}^* \to \mathbb{G}_2$ that maps each attribute i to the element in $\mathbb{G}_2$. Select $x_k, v_k \in_R \mathbb{Z}_p$ and compute $y_k = G_2^{x_k}, Y_k = g_1^{v_k}, Z_k = \hat{e}(Y_k, G_2)$. Next the authorities pairwise engage a two party key exchange protocol as that in ODMA-ABS and consequently authority k and j share a unique seed $s_{kj} \in \mathbb{Z}_p$ that satisfies $s_{kj} = s_{jk}$. Finally, output the public parameter and the master secret key as

$$PP = (\{y_k, Y_k, Z_k, H_k, d_k\}_{k\in\{1,\cdots,N\}}, H, H', e, G_2),$$
$$MSK = (\{x_k, v_k, \{s_{kj}\}_{j\in\{1,\cdots,N\}\backslash\{k\}}\}_{k\in\{1,\cdots,N\}}).$$

- **User Grant**: Assume that the user u has the public key $pk_u = G_2^s$ where s is the secret key and his attribute set $A_u = \bigcup_{k\in\{1,\cdots,N\}} A_{k,u}$, user u interacts with authority k that works as follows.
  1) For $j \in \{1, \cdots, N\}\backslash\{k\}$, run the AKI protocol between j and u as that in ODMA-ABS. As a consequence, u gets
  $$D_{kj} = G_2^{R_{kj}}\text{PRF}_{kj}(u), k > j$$
  $$D_{kj} = G_2^{R_{kj}}/\text{PRF}_{kj}(u), k < j$$
  2) k randomly selects a $d_k - 1$ degree polynomial $p_k(\cdot)$ such that
  $$p_k(0) = v_k - \sum_{j\in\{1,\cdots,N\}\backslash\{k\}} R_{kj} \quad (14)$$
  and for each $i \in A_{k,u} \cup C_k$ and each $r_{k,i}$ randomly selected from $\mathbb{Z}_p$, computes
  $$d_{k,i0} = G_2^{p_k(i)}(pk_u H_k(i))^{r_{k,i}}, d_{k,i1} = g_1^{r_{k,i}}.$$
  3) k outputs the private key tuple $SK_K =$
  $$(\{d_{k,i0}, d_{k,i1}\}_{i\in A_{k,u}\cup C_k}, \{D_{kj}\}_{j\in\{1,\cdots,N\}\backslash\{k\}}).$$
  Finally, output the user's private keys as
  $$SK = \{SK_k\}_{k\in\{1,\cdots,N\}}.$$

- **File Access**: Suppose that the user has obtained the private keys and would like to access some resource with the policy that is defined by predicate $\Upsilon_{n,C^*}(\cdot)$ (similar with that in ODMA-ABS scheme), the user takes the following steps.
  1) For each $k \in \{1, \cdots, N\}$, select a random n_k-element attribute subset $A'_k \subseteq A_{k,u} \cap C^*_k$, and a $(d_k - n_k)$-element default attribute subset $C'_k \subseteq C_k$. Define $S_k = A'_k \cup C'_k$, and then $|S_k| = d_k$.
  2) Select $m_k + d - n_k$ random values $s_{k,1}, \cdots, s_{k,m_k+d-n_k}$ from $\mathbb{Z}_p$, and compute the following items
  $$\sigma'_0 = \prod_{1\le k\le N}(\prod_{i\in S_k} d_{k,i0}^{\Delta_{i,S_k}(0)} \prod_{i\in C^*_k\cup C'_k}(pk_u H_k(i))^{s_{k,i}})$$
  $$\sigma'_{k,i} = \begin{cases} d_{k,i1}^{\Delta_{i,S_k}(0)} g_1^{s_{k,i}}, & i \in S_k \\ g_1^{s_{k,i}}, & i \in C^*_k \backslash A'_k \end{cases}$$
  3) Compute
  $$D_u = \prod_{(k,j)\in\{1,\cdots,N\}\times\{1,\cdots,N\}\backslash\{k\}} D_{k,j},$$
  and then compute
  $$\sigma_0 = D_u \cdot \sigma'_0, \sigma_{k,i} = \sigma'^{s}_{k,i}.$$
  4) The user then sends the tuple $(\sigma_0, \sigma'_{k,i}, \sigma_{k,i})$. The server checks if the following equation holds.
  $$\hat{e}(g_1, \sigma_0) \stackrel{?}{=} \prod_{1\le k\le N}[\prod_{i\in C^*_k\cup C'_k} Z_k \hat{e}(\sigma'_{k,i}, H_k(i))\hat{e}(\sigma_{k,i}, G_2)] \quad (15)$$
  If the above equation holds, then the user will be allowed to access the resource that is stored in the server. Otherwise, the access request will be rejected.

B. attribute key update

In this part, we discuss the update of attribute private keys. In practical scenario, the attribute set of a user may not be constant all the time. Sometimes, certain attribute of the user will be revoked from the user's attribute set or there will be some new attributes inserted in the user's attribute set. It is required that the attribute private keys are effectively updated when the attribute set of the user is changed. Here "effectively" means that the update shall only happen to the keys that are associated with the inserted/revoked attributes. In this paper, we just briefly introduce how to realize effective update for

attribute private keys, but leave the technical details in the future work. To do so, we let each attribute authority build a set of all users who owns certain attribute and allocate a unique attribute group key for the set. The group key and the set are shared with the resource storage server. The storage server then choose a key encoding key (KEK) to encode the group key. When a user propose an access request to the storage server, it returns the encoded group key to the user. The user can obtain the KEK by his current attribute set and then get the group key. And the authentication tuple that the user generates is required to contain the correct attribute group key. When certain attribute is changed, the attribute authority not only generates a new private key for user but also change the corresponding attribute group key and the user set for the storage server. The server also updates the corresponding KEK. When a request is proposed by the user, the updated group key and KEK will be sent to him/her. And the outdated attribute private keys can no longer be used since it cannot generate the correct authentication tuple with either updated group key or outdated group key. We will present detailed algorithms for attribute key update in the future work. Moreover, to make the access control scheme more robust, we also consider two-factor authentication. There are some good techniques that can be utilized to realize two-factor authentication [30], [31]. We will discuss more in the future work.

VI. Conclusion

We have presented an outsourced decentralized multi-authority attribute based signature scheme. The ODMA-ABS achieves attribute privacy and is able to detect malicious behaviors of the signing cloud server. Furthermore, by outsourcing heavy computation overhead to a signing cloud server, the scheme also achieves strong performance in efficiency. We have proved the security properties and compared certain schemes to show the advantages of our scheme. An access control scheme based on the proposed ODMA-ABS is also presented as a specific application of ODMA-ABS.

Acknowledgment

This work is supported by the National Nature Science Foundation of China under Grant No.: 61772311 and 61272091.

The scientific calculations in this paper have been done on the HPC Cloud Platform of Shandong University.

References

[1] A. Shamir. Identity-based cryptosystems and signature schemes. Workshop on the Theory and Application of Cryptographic Techniques. Springer Berlin Heidelberg, 1984: 47-53.

[2] M. Chase, S. S. Chow. Improving privacy and security in multi-authority attribute-based encryption. Proceedings of the 16th ACM conference on Computer and communications security, pp. 121-130. ACM, 2009.

[3] H. Maji, M. Prabhakaran, M. Rosulek. Attribute based signatures: Achieving attribute privacy and collusion-resistance. 2008. http://eprint.iacr.org/2008/328.

[4] D. Khader. Attribute Based Group Signatures. IACR Cryptology ePrint Archive 2007 (2007): 159.

[5] R. L. Rivest, A. Shamir, Y. Tauman. How to leak a secret. International Conference on the Theory and Application of Cryptology and Information Security, pp. 552-565. Springer, Berlin, Heidelberg, 2001.

[6] J. Sun, J. Qin, and J. Ma. Securely Outsourcing Decentralized Multi-authority Attribute Based Signature. Cyberspace Safety and Security: 9th International Symposium, CSS 2017, Xian China, October 23C25, 2017, Proceedings. Vol. 10581. Springer, 2017.

[7] S. F. Shahandashti and R. Safavi-Naini. Threshold attribute-based signatures and their application to anonymous credential systems. AFRICACRYPT'09, 2009, 198-216.

[8] J. Li, M. H. Au, W. Susilo, D. Xie, K. Ren. Attribute-based signature and its applications. ACM Symposium on Information, Computer and Communications Security. ACM, 2010:60-69.

[9] H. Maji, M. Prabhakaran, M. Rosulek. Attribute-based signatures. International Conference on Topics in Cryptology: Ct-Rsa. Springer-Verlag, 2011:376-392.

[10] D. Cao, B. Zhao, X. Wang, J. Su, G. Ji. Multi-authority Attribute-Based Signature. Third International Conference on Intelligent NETWORKING and Collaborative Systems. IEEE Computer Society, 2011:668-672.

[11] Y. Chen, J. Chen, G. Yang. Provable Secure Multi-Authority Attribute Based Signatures. Journal of Convergence Information Technology, 2013, 8(2):545-553.

[12] T. Okamoto, K. Takashima. Decentralized Attribute-Based Signatures. Lecture Notes in Computer Science, 2013, 7778:125-142.

[13] S. Hohenberger, A. Lysyanskaya. How to securely outsource cryptographic computations. Theory of Cryptography. Springer Berlin Heidelberg, 2005: 264-282.

[14] R. Gennaro, C. Gentry, B. Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers.Advances in Cryptology-CCRYPTO 2010. Springer Berlin Heidelberg, 2010: 465-482.

[15] C. Gentry. Fully homomorphic encryption using ideal lattices.STOC.2009: 169-178.

[16] A. C. Yao. Protocols for secure computations. FOCS. 1982, 82: 160-164.

[17] C. Gentry. Computing arbitrary functions of encrypted data. Communications of the ACM, 2010, 53(3): 97-105.

[18] D. Fiore, R. Gennaro, V. Pastro. Efficiently Verifiable Computation on Encrypted Data.ACM Sigsac Conference on Computer and Communications Security. ACM, 2014:844-855.

[19] J. Lai, R. H. Deng, H. Pang, J. Weng. Verifiable Computation on Outsourced Encrypted Data. Computer Security - ESORICS 2014. Springer International Publishing, 2014:273-291.

[20] M. Green, S. Hohenberger, B. Waters. Outsourcing the decryption of ABE ciphertexts. Usenix Conference on Security. USENIX Association, 2011:34-34.

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

[22] Z. Cai, H. Yan, P. Li, Z. Huang, C. Gao. Towards secure and flexible EHR sharing in mobile health cloud under static assumptions. Cluster Computing 20(3): 2415-2422, 2017.

[23] X. Chen, J. Li, X. Huang, J. Li, Y. Xiang, and D. S. Wong. Secure outsourced attribute-based signatures. IEEE Transactions on Parallel and Distributed Systems 25, no. 12 (2014): 3285-3294.

[24] Y. Ren, T. Jiang. Verifiable outsourced attribute-based signature scheme. Multimedia Tools & Applications, 2017:1-11.

[25] O. Goldreich, S. Goldwasser, S. Micali. How to construct Randolli functions. Foundations of Computer Science, 1984. 25th Annual Symposium on. IEEE, 1984: 464-479.

[26] D. Boneh, I. Mironov, V. Shoup. A secure signature scheme from bilinear maps. Proc. of CT-RSA 2003, 98-110.

[27] S. Benabbas, R. Gennaro, Y. Vahlis. Verifiable delegation of computation over large datasets. Proc. of CRYPTO 2011, 111-131.

[28] S. M. Chow. New privacy-preserving architectures for identity-/attribute-based encryption. Doctoral dissertation, Courant Institute of Mathematical Sciences New York, 2010.

[29] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, H. Shacham. Randomizable Proofs and Delegatable Anonymous Credentials. International Cryptology Conference on Advances in Cryptology. Springer-Verlag, 2009:108-125.

[30] D. He, D. Wang. Robust biometrics-based authentication scheme for multiserver environment. IEEE Systems Journal 9, no. 3 (2015): 816-823.

[31] D. Wang, P. Wang. Two birds with one stone: Two-factor authentication with security beyond conventional bound. IEEE transactions on dependable and secure computing (2018).