JAIST Repository
https://dspace.jaist.ac.jp/

| | |
|---|---|
| Title | Dynamic Attribute-based Signcryption without Random Oracles |
| Author(s) | Emura, Keita; Miyaji, Atsuko; Rahman, Mohammad Shahriar |
| Citation | International Journal of Applied Cryptography (IJACT), 2(3): 199-211 |
| Issue Date | 2012 |
| Type | Journal Article |
| Text version | author |
| URL | http://hdl.handle.net/10119/10715 |
| Rights | Copyright (C) 2012 Inderscience. Keita Emura, Atsuko Miyaji, and Mohammad Shahriar Rahman, International Journal of Applied Cryptography (IJACT), 2(3), 2012, 199-211. http://dx.doi.org/10.1504/IJACT.2012.045589 |
| Description | |

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

# Dynamic Attribute-based Signcryption without Random Oracles

**Keita Emura***
Center for Highly Dependable Embedded Systems Technology, Japan Advanced Institute of Science and Technology, Japan
E-mail: k-emura@jaist.ac.jp
∗ Corresponding author

**Atsuko Miyaji and Mohammad Shahriar Rahman**
School of Information Science, Japan Advanced Institute of Science and Technology, Japan
E-mail:{miyaji, mohammad}@jaist.ac.jp

**Abstract:** In SCN2010, Gagné, Narayan, and Safavi-Naini proposed attribute-based signcryption (ABSC) with threshold structure. As in ciphertext-policy attribute-based encryption (CP-ABE), an encryptor can specify the access structure of decryptors, and as in attribute-based signature (ABS), each decryptor can verify the encryptor's attributes. On the contrary to the access structure of decryptors, the access structure of the encryptor needs to be fixed in the setup phase. In this paper, we propose ABSC with dynamic property, where access structures of encryptor can be updated flexibly without re-issuing secret keys of users. We call this primitive dynamic ABSC (DABSC). Our DABSC scheme is secure in the standard model under the decision bilinear Diffie-Hellman assumption and the computational Diffie-Hellman assumption.

**Keywords:** Attribute-Based Signcryption; Dynamic Property; Authenticated Fine-Grained Storage Systems

**Reference** This paper should be made as follows: Emura, K., Miyaji, A., and Mohammad, R.S. (2012) 'Dynamic Attribute-based Signcryption without Random Oracles', Int. J. Applied Cryptography, Vol**, No*, pp**–**.

**Biographical notes:** Keita Emura is a postdoctral researcher at Center for Highly Dependable Embedded Systems Technology, Japan Advanced Institute of Science and Technology (JAIST). His research interests include cryptography and information security. He is a member of the Institute of Electronics, Information and Communication Engineers (IEICE).

Atsuko Miyaji is a professor at JAIST. Her research interests include the application of projective varieties theory to cryptography and information security. She is a member of the International Association for Cryptologic Research (IACR), the IEICE, the Information Processing Society of Japan (IPSJ), and the Mathematical Society of Japan.

Mohammad Shahriar Rahman is a Ph.D student at JAIST. His research interests include privacy-preserving data mining, game theory, and information security. He is a student member of the IEICE and the IACR.

## 1 Introduction

### 1.1 Attribute-Based Cryptography

Beyond identity-based cryptosystems (such as identity-based encryption [7]), recently, many cryptographic schemes with user's attributes have been proposed. We summarize attribute-based cryptography as follows:

**Attribute-Based Encryption (ABE)**: Perhaps this is the most famous and important topic in attribute-based cryptography. ABE is an encryption scheme, where users with some attributes can decrypt the ciphertext associ-

Copyright © 200x Inderscience Enterprises Ltd.

ated with these attributes. In ABE schemes, an encryptor can indicate many decryptors by assigning common attributes of these decryptors such as gender, age, affiliation, and so on. There are three kinds of ABE, key-policy ABE (KP-ABE), ciphertext-policy ABE (CP-ABE), and dual-policy ABE. KP-ABE [2, 20, 40] are schemes such that each private key is associated with an access structure, and therefore a key generation authority (KGA) decides the access policy of the corresponding decryption key. CP-ABE [5, 12, 16, 19, 22, 29, 39, 45] are schemes such that each ciphertext is associated with an access structure, and therefore an encryptor can decide the access policy of the corresponding ciphertext. Dual-Policy ABE [1] is a conjunctively combined scheme between KP-ABE and CP-ABE. A significant generic conversion for transforming a chosen-plaintext (CPA) secure ABE to a chosen-ciphertext (CCA) secure ABE has been proposed by Yamada et al. [46]. It is remarkable that this transformation can be applied for both CP-ABE and KP-ABE, whereas the Goyal et al. [20] one is for KP-ABE only.

**Attribute-Based Signature (ABS)**: ABS [27, 28, 34, 43] assures the verifier that a signer has endorsed the message with a set of attributes. A signature does not leak which attributes were used to generate it, that satisfies a predicate, except the assigned attributes (attribute-signer privacy). One exception is a ABS scheme with weak signer-attribute privacy [43], where some attributes are revealed from a signature. This property is achieved by using an interactive verification protocol. Maji et al. [34] proposed an ABS scheme that supports a powerful set of predicates (including AND, OR, and threshold gate) to apply monotone span program [23], although this scheme is proved in the generic group model. Note that, very recently, Maji et al. also proposed an ABS scheme with such powerful set of predicates [35] which is based on the Boneh-Boyen signature [6], the Waters signature [44], and the Groth-Sahai proof system [21]. Other provably secure ABS schemes have been proposed in [27, 28, 43] with threshold structures.

**Attribute-Based Group Signature (ABGS)**: ABGS [15, 24, 25] is a kind of group signature [10], where a user with a set of attributes can prove anonymously whether he/she has these attributes or not. Anonymity stands for a verifier cannot identify who the actual signer is from group members. As a difference from ABS, there is a opening manager (as in group signatures) who can identify the actual signer (anonymity revocation), and a verifier can "explicitly" verify whether a user has these attributes or not. By applying this explicit attribute verification, anonymous survey for the collection of attribute statistics is proposed [15].

In these ABGS schemes, an access tree [20] is applied to express these relationships. In Khader's schemes [24, 25], if an access structure has to be changed, then a user has to be re-issued with all attribute certificates. The first dynamic ABGS has been proposed in [15], where relationships among attributes can be changed after setup phase. Due to the dynamic property, re-issuing the attribute certificate previously issued for each user is not necessary. The dynamic property is achieved by applying a *bottom-up approach* construction (we introduce it in Section 4). Unfortunately, previous ABGS schemes are inefficient compared with ABS, as these ABGS schemes apply signatures converted by Fiat-Shamir heuristic from zero-knowledge proofs of knowledge (so, these ABGS are secure in the random oracle model).

**Attribute-Based Signcryption (ABSC)**: Recently, Gagné, Narayan, and Safavi-Naini proposed ABSC with threshold structure [18] to achieve Cost(ABS & CP-ABE) $<$ Cost(ABS)+Cost(CP-ABE)[1]. That is, their ABSC scheme is efficient compared with the encrypt-then-sign paradigm. As in CP-ABE, an encryptor can specify the access structure of decryptors, and as in ABS, each decryptor can verify the encryptor's attributes. The Gagné et al. ABSC scheme is secure under the hashed modified bilinear Diffie-Hellman assumption and the modified CDH assumption without random oracles. Note that the Gagné et al. definition does not consider the signer-attribute privacy. Through this fact, a decryptor can verify the encryptor's attribute *explicitly* as in ABGS. This "explicit attribute verifiability" property is preferable for the following encrypted storage system usage.

### 1.2 Encrypted Storage System

Encrypted storage system is a well-known application of CP-ABE. The benefit point of CP-ABE (compared with the simple encryption method) is described as follows. If a data is encrypted by using an encryption key, then the total number of encryption and decryption keys increase. If plural data are encrypted by using one encryption key, then a fine-grained access control is not achieved. To indicate the set of common attributes of decryptors (such as affiliation, post, and so on), CP-ABE schemes can achieve a fine-grained access control without increasing the number of keys.

On the contrary to the decryptor's attributes, there is no way to verify the set of attributes of encryptor if CP-ABE is applied only. To check the source of storage files, attributes of encryptor is important information. By applying the Gagné et al. ABSC, both CP-ABE and ABS properties can be handled for encrypted storage system usage, simultaneously. So, a decryptor can check the encryptor's attribute explicitly. However, the threshold structure (which is supported by the Gagné et al. ABSC) is not suitable for encrypted storage system usage, although it is useful for fault tolerance usage. In addition, the access structure of the encryptor is specified only once, and it cannot be changed. More precisely, the threshold value is decided in the key generation phase, and it cannot be updated without re-issuing the new key.

[1]This expression is from the title of Zheng's paper [47]. There are several signcryption schemes [13, 26, 31, 32, 36, 37, 38, 41] (resp. identity-based signcryption schemes [8, 11, 42]) in the literature.

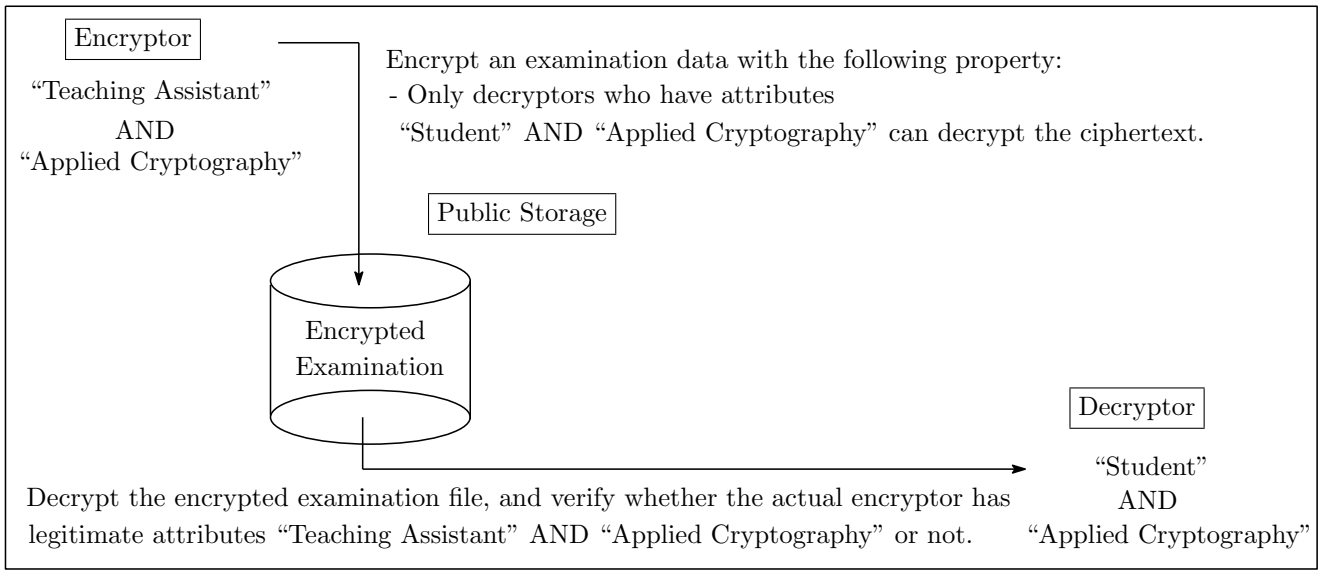



**Fig. 1: Authenticated Fine-Grained Storage Systems**

### 1.3 Our Contribution

In this paper[2], we propose ABSC with dynamic property, where access structures of encryptor can be changed without re-issuing secret keys of users. We call this primitive Dynamic ABSC (DABSC). Our DABSC is based on the Cheung-Newport CP-ABE scheme [12] and the Li et al. ABS [27], and applies the bottom-up approach construction [15] to implement the dynamic property of the signature (i.e, for proving the encryptor's attributes) part. Required complexity assumptions in our proposal are standard ones, i.e., the decisional bilinear Diffie-Hellman (DBDH) assumption and the computational Diffie-Hellman (CDH) assumption. It is particularly worth noting that the bottom-up approach construction itself does not require the random oracle, although the eventual dynamic ABGS requires random oracles. So, our DABSC scheme is secure in the standard model. Our DABSC scheme supports the tree structure of the access structure of encryptor (ABS part), and the AND-gates with wildcard expression of the access structure of decryptor (CP-ABE part).

As an application of DABSC, we consider authenticated fine-grained storage systems. For example, let a teaching assistant of a lecture "Applied Cryptography" would like to store an encrypted examination data for students (who take Applied Cryptography) only. In addition, students would like to check whether a stored file was made by a teaching assistant of Applied Cryptography. Then an encryptor makes a ciphertext part associated with attributes of a decryptor (Student $\wedge$ Applied Cryptography), and also makes a signature part associated with attributes of the encryptor (Teaching Assistant $\wedge$ Applied Cryptography). We illustrate it in Fig1.

The dynamic property is suitable for the following example. We assume that the encryptor (who is a teaching assistant of Applied Cryptography) becomes a teaching assistant of a lecture "Discrete Mathematics", and the encryptor has obtained the secret key for attributes Teaching Assistant and Applied Cryptography. If the dynamic property is not handled, then KGA needs to re-issue the secret key of both Applied Cryptography and Teaching Assistant for handling the updated access structure of encryptor. It is quite inefficient and impractical (See Table 1).

**Table 1. Computational complexity of changing predicate**

| | KGA | User |
|---|---|---|
| Non-dynamic scheme | $O(N \cdot n_e)$ | $O(n_e)$ |
| Dynamic scheme | $O(n_e)$ | None |

$N$ : the number of users
$n_e$ : the maximum number of attributes having each user

Under the dynamic property, KGA only has to issue the secret key of Discrete Mathematics, and no computation by the encryptor is required. While the above example describes the case of small number of predicates, we believe that the dynamic property gives us a very efficient and practical solution when the number of predicates grows large. Under the dynamic property, even if the current predicate is updated, users do not have to be involved in the updating procedure. This is the most benefit point of our proposal.

## 2 Preliminaries

In this section, we define bilinear groups and complexity assumptions. Through the remaining paper, $x \xleftarrow{\$} S$ means

[2] A preliminary version of this paper appears in the 16th Australasian Conference on Information Security and Privacy, ACISP 2011 [17]. This is the full paper.

that $x$ is chosen uniformly from a set $S$. $y \leftarrow A(x)$ means that $y$ is an output of an algorithm $A$ under an input $x$. In addition, we denote *State* as the state information transmitted by the adversary to himself across stages of the attack in experiments.

**Definition 1** (Bilinear Groups)**.** *Bilinear groups and a bilinear map are defined as follows:*

1. *$\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of prime order $p$.*
2. *$g$ is a generator of $\mathbb{G}$.*
3. *$e$ is an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties.*
   - *Bilinearity : for all $u, u', v, v' \in \mathbb{G}$, $e(uu', v) = e(u, v)e(u', v)$ and $e(u, vv') = e(u, v)e(u, v')$.*
   - *Non-degeneracy : $e(g, g) \neq 1_{\mathbb{G}_T}$ ($1_{\mathbb{G}_T}$ is the $\mathbb{G}_T$'s unit).*

**Definition 2** (The CDH assumption)**.** *The computational Diffie-Hellman (CDH) problem in $\mathbb{G}$ is defined as follows: given a $(g, g^a, g^b) \in \mathbb{G}^3$ as input, where $a, b \in \mathbb{Z}_p^*$, which outputs a value $g^{ab}$. We say that the CDH assumption holds in $\mathbb{G}$ if for all probabilistic polynomial-time (PPT) algorithm, an advantage $\epsilon$, where $\Pr[\mathcal{A}(g, g^a, g^b) = g^{ab}] \geq \epsilon$, is negligible.*

**Definition 3** (The DBDH assumption)**.** *The decisional bilinear Diffie-Hellman (DBDH) problem in $\mathbb{G}$ is a problem, for input of a tuple $(g, g^a, g^b, g^c, Z) \in \mathbb{G}^4 \times \mathbb{G}_T$ to decide $Z = e(g, g)^{abc}$ or not. We say that the DBDH assumption holds in $\mathbb{G}_1$ if for all PPT algorithm, an advantage $\epsilon$, where $Adv_{DBDH}(\mathcal{A}) := |\Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 0] - \Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g, g)^z) = 0]| \geq \epsilon$ and $e(g, g)^z \in \mathbb{G}_T \setminus \{e(g, g)^{abc}\}$, is negligible.*

### 2.1 Strongly Existentially Unforgeable (sUF) One-Time Signatures

We apply the Cheung-Newport CP-ABE scheme [12], which needs the CHK transformation [9] to satisfy CCA security by sUF one-time signature (e.g., [4]). So, here we define sUF one-time signature as follows. An sUF one-time signature consists of three algorithms, $\mathsf{Sig.KeyGen}$, $\mathsf{Sign}$, and $\mathsf{Verify}$. $\mathsf{Sig.KeyGen}$ is a probabilistic algorithm which outputs a signing/verification key pair $(K_s, K_v)$. $\mathsf{Sign}$ is a probabilistic algorithm which outputs a signature $\sigma$ from $K_s$, and a message $M \in \mathcal{M}_{Sig}$, where $\mathcal{M}_{Sig}$ is the message space of a signature scheme. $\mathsf{Verify}$ is a deterministic algorithm which outputs a bit from $\sigma$, $K_v$ and $M$. "$\mathsf{Verify}$ outputs 1" stands for that $\sigma$ is a valid signature of $M$, and 0, otherwise. The security experiment of sUF one-time signature under an adaptive chosen message attack (one-time sUF-CMA) is defined as follows:

**Definition 4** (one-time sUF-CMA)**.** *We say that a signature scheme is one-time sUF-CMA secure if the advantage $Adv_{\mathcal{A}}^{\text{one-time sUF-CMA}}(1^k)$ is negligible for any PPT adversary $\mathcal{A}$.*

$$
\begin{aligned}
&Adv_{\mathcal{A}}{}^{\text{one-time sUF-CMA}}(1^k) \\
&= \Pr\big[(K_s, K_v) \leftarrow \mathsf{Sig.KeyGen}(1^k); \\
&\qquad (M, State) \leftarrow \mathcal{A}(K_v); \sigma \leftarrow \mathsf{Sign}(K_s, M); \\
&\qquad (M^*, \sigma^*) \leftarrow \mathcal{A}(K_v, \sigma, State); \\
&\qquad (M^*, \sigma^*) \neq (M, \sigma); \mathsf{Verify}(K_v, \sigma^*, M^*) = 1\big]
\end{aligned}
$$

Intuitively, an sUF one-time signature scheme is secure when no adversary $\mathcal{A}$ can issue a pair $(M^*, \sigma^*)$ even if $\mathcal{A}$ has already obtained a signature $\sigma \neq \sigma^*$ of the signed message $M^*$.

## 3 Definitions of DABSC

### 3.1 System Operations of DABSC

In the following, values are subscripted by $e$ for encryptors, and values are subscripted by $d$ for decryptors. Let $\mathbb{A}_e = (att_1, att_2, \ldots, att_{n_e})$ be the universe of possible attributes of encryptors, $\mathbb{A}_d = (att_1, att_2, \ldots, att_{n_d})$ be the universe of possible attributes of decryptors, and $\Upsilon_e$ (resp. $\Upsilon_d$) be a claim-predicate over $\mathbb{A}_e$ (resp. $\mathbb{A}_d$) of encryptors (resp. decryptors). We say that an attribute set $\Gamma_e \subseteq \mathbb{A}_e$ (resp. $\Gamma_d \subseteq \mathbb{A}_d$) satisfies a claim-predicate $\Upsilon_e$ (resp. $\Upsilon_d$) if $\Upsilon_e(\Gamma_e) = 1$ (resp. $\Upsilon_d(\Gamma_d) = 1$). In our proposed scheme, $\Upsilon_e$ is represented by tree structures (i.e., access trees) and $\Upsilon_d$ is represented by AND-gates with wildcard expression. In the following definition, an encryptor can select an access structure of decryptor $\Upsilon_d$ for each signcryption ciphertext (which follows the ciphertext-policy property of ABE). On the contrary, an access structure of encryptor $\Upsilon_e$ is publicly opened. This means that a legitimate encryptor who has attributes satisfying $\Upsilon_e$ can make a signcryption ciphertext.

Next, we modify the definitions of the Gagné et al. ABSC [18] to handle the dynamic property.

**Definition 5** (Dynamic Attribute-Based Signcryption (DABSC))**.**

$\mathsf{Setup}$**:** *This algorithm takes as inputs a security parameter $k \in \mathbb{N}$, and returns public parameters params and a master key msk.*

$\mathsf{sExtract}$**:** *This algorithm takes as inputs params, msk, and a set of attributes of an encryptor $\Gamma_e \subseteq \mathbb{A}_e$, and returns signing keys $\{sk_{e,i}\}_{att_i \in \Gamma_e}$.*

$\mathsf{uExtract}$**:** *This algorithm takes as inputs params, msk, and a set of attributes of an decryptor $\Gamma_d \subseteq \mathbb{A}_d$, and returns decryption keys $\{sk_{d,i}\}_{att_i \in \Gamma_d}$.*

$\mathsf{BuildPredicate}$**:** *This algorithm takes as inputs params, msk, and the $\ell$-th access tree $T_\ell$, and returns the public value of $\ell$-th access tree $\Upsilon_e^\ell$.*

$\mathsf{Signcrypt}$**:** *This algorithm takes as inputs params, $\Upsilon_e^\ell$, $\{sk_{e,i}\}_{att_i \in \Gamma_e}$, where $\Upsilon_e^\ell(\Gamma_e) = 1$, an access structure*

**Table 2. DABSC Experiments**

S-IND-DABSC-CCA2

$$Adv_{\mathcal{A}}^{\text{S-IND-DABSC-CCA2}}(k) =$$
$$|\Pr\big[(\Upsilon_d^*, T_0, State) \leftarrow \mathcal{A}(k);\ (params, msk) \leftarrow \mathsf{Setup}(1^k);$$
$$\text{Set } \mathcal{O} := \{\mathsf{sExtract}(params, msk, \cdot), \mathsf{uExtract}(params, msk, \cdot), \mathsf{Unsigncrypt}(params, \cdot, \cdot),$$
$$\mathsf{BuildPredicate}(params, msk, \cdot)\};$$
$$(M_0^*, M_1^*, \Gamma_e^*, State) \leftarrow \mathcal{A}^{\mathcal{O}}(params, State);\ b \xleftarrow{\$} \{0,1\};\ C^* \leftarrow \mathsf{Signcrypt}(params, \Upsilon_e^\ell, \{sk_{e,i}\}_{att_i \in \Gamma_e^*}, \Upsilon_d^*, M_b^*);$$
$$b' \leftarrow \mathcal{A}^{\mathcal{O}}(C^*, State);\ b = b'\big] - \tfrac{1}{2}|$$

S-EUF-DABSC-CMA

$$Adv_{\mathcal{A}}^{\text{S-EUF-DABSC-CMA}}(k) =$$
$$\Pr\big[(T_e^*, T_0, State) \leftarrow \mathcal{A}(k);\ (params, msk) \leftarrow \mathsf{Setup}(1^k);$$
$$\text{Set } \mathcal{O} := \{\mathsf{sExtract}(params, msk, \cdot), \mathsf{uExtract}(params, msk, \cdot), \mathsf{BuildPredicate}(params, msk, \cdot),$$
$$\mathsf{Signcrypt}(params, \cdot, \cdot, \cdot)\};$$
$$(C^*, \Gamma_d^*) \leftarrow \mathcal{A}^{\mathcal{O}}(params, State);\ \mathsf{Unsigncrypt}(params, \Upsilon_e^*, \{sk_{d,i}\}_{att_i \in \Gamma_d^*}, C^*) = M^* \neq \perp;$$
$$\text{(For } \Gamma_e \text{ where } \Upsilon_e^*(\Gamma_e) = 1,\ \mathcal{A} \text{ did not query either } (M, \Gamma_e, \Upsilon_d^*) \text{ to the } \mathsf{Signcrypt} \text{ oracle}$$
$$\text{or } \Gamma_e \text{ to the } \mathsf{sExtract} \text{ oracle, where } \Upsilon_e^*(\Gamma_e) = 1) \vee (\Upsilon_e^*(\Gamma_e^*) \neq 1)\big]$$

$\Upsilon_d$, *and a plaintext* $M$, *and returns a ciphertext* $C$ *on* $M$. *We assume that* $\Gamma_e$ *and* $\Upsilon_d$ *are included into* $C$.

Unsigncrypt: *This algorithm takes as inputs params,* $\Upsilon_e^\ell$, $\{sk_{d,i}\}_{att_i \in \Gamma_d}$, *where* $\Upsilon_d(\Gamma_d) = 1$, *and* $C$, *and verifies whether the encryptor's attributes satisfy* $\Upsilon_e^\ell$ *or not, along with* $\Gamma_e$ *and* $\Upsilon_e^\ell$. *If not, then output* $\perp$, *and* $M$ *otherwise.*

The above algorithms follow the correctness requirement: for all $(params, msk) \leftarrow \mathsf{Setup}(1^k)$, $\{sk_{e,i}\}_{att_i \in \Gamma_e} \leftarrow \mathsf{sExtract}(params, msk, \Gamma_e)$, $\{sk_{d,i}\}_{att_i \in \Gamma_d} \leftarrow \mathsf{uExtract}(params, msk, \Gamma_d)$, $\Upsilon_e^\ell \leftarrow \mathsf{BuildPredicate}(params, msk, T_\ell)$, and $C \leftarrow \mathsf{Signcrypt}(params, \Upsilon_e^\ell, \{sk_{e,i}\}_{att_i \in \Gamma_e}, \Upsilon_d, M)$ with $\Upsilon_e^\ell(\Gamma_e) = 1$, $M \leftarrow \mathsf{Unsigncrypt}(params, \Upsilon_e^\ell, \{sk_{d,i}\}_{att_i \in \Gamma_d}, C)$ holds when $\Upsilon_d(\Gamma_d) = 1$.

### 3.2 Security Requirements

Here, we define indistinguishability against adaptive chosen-ciphertext attack property under selective attribute model (S-IND-DABSC-CCA2) and existential unforgeability against chosen-message attack in the selective attribute model (S-EUF-DABSC-CMA). S-IND-DABSC-CCA2 guarantees that no PPT adversary $\mathcal{A}$ (which is essentially the same as the CCA adversary of CP-ABE [12]) can guess whether the actual plaintext is $M_0^*$ or $M_1^*$, namely, no plaintext information is revealed from the ciphertext. Note that S-IND-DABSC-CCA2 captures collusion resistance (i.e., $\mathcal{A}$ is allowed to issue $\Gamma_d$ and $\Gamma_d'$ to the $\mathsf{uExtract}$ oracle such that $\Upsilon_d^*(\Gamma_d) \neq 1$, $\Upsilon_d^*(\Gamma_d') \neq 1$, $\Gamma_d \cup \Gamma_d' = \Gamma_d^*$, and $\Upsilon_d^*(\Gamma_d^*) = 1$) as in the conventional CP-ABE definition.

**Definition 6** (S-IND-DABSC-CCA2)**.** *A DABSC scheme is said to be S-IND-DABSC-CCA2 secure if the advantage* $Adv_{\mathcal{A}}^{\text{S-IND-DABSC-CCA2}}(k)$ *is negligible for any PPT adversary* $\mathcal{A}$ *in the S-IND-DABSC-CCA2 experiment (defined in Table 2). The notations are defined as follows:*

- $\mathsf{sExtract}$ *is the oracle for any* $\Gamma_e$, *where it returns* $\{sk_{e,i}\}_{att_i \in \Gamma_e}$.
- $\mathsf{uExtract}$ *is the oracle for any* $\Gamma_d$ *with* $\Upsilon_d^*(\Gamma_d) \neq 1$, *where it returns* $\{sk_{d,i}\}_{att_i \in \Gamma_d}$.
- $\mathsf{Unsigncrypt}$ *is the unsigncryption oracle, where for input of* $(C, \Gamma_d)$ *with the restriction* $(C^*, \Gamma_d)$ *such that* $\Upsilon_d^*(\Gamma_d) = 1$. *It returns the result of* $\mathsf{Unsigncrypt}(params, \Upsilon_e^i, \{sk_{d,i}\}_{att_i \in \Gamma_d}, C)$, *where* $\Upsilon_e^i$ *is the current predicate when* $\mathcal{A}$ *issues the unsigncryption query.*
- $\mathsf{BuildPredicate}$ *is the oracle for input of an access tree* $T$, *it returns the corresponding public predicate* $\Upsilon_e$.
  - $\mathcal{A}$ *can explicitly control* $\Upsilon_e^i$ *via the* $\mathsf{BuildPredicate}$ *oracle.*
  - *Note that we require* $\Upsilon_e^\ell(\Gamma_e^*) = 1$, *where* $\Upsilon_e^\ell$ *is the public predicate in the challenge phase. In addition,* $T$ *(and the initial access tree* $T_0$ *also) must follow the condition that leaves of trees appear in* $\mathbb{A}_e$.

Next, we define S-EUF-DABSC-CMA. In the definition of S-EUF-DABSC-CMA, we consider two types of adversaries. S-EUF-DABSC-CMA guarantees that no (type 1) adversary $\mathcal{A}$ can make a forged ciphertext which is correctly decrypted (i.e., the $\mathsf{Unsigncrypt}$ algorithm outputs $M \neq \perp$) even though $\mathcal{A}$ did not issue either $\Gamma_e$ to the $\mathsf{sExtract}$ oracle such that $\Upsilon_e^*(\Gamma_e) = 1$ or $(M, \Gamma_e, \Upsilon_d^*)$ to the $\mathsf{Signcrypt}$ oracle such that $\Upsilon_e^*(\Gamma_e) = 1$. Moreover, no (type 2) $\mathcal{A}$ (who can obtain all $\{sk_{e,i}\}_{att_i \in \Gamma_e}$) can make a

forged ciphertext which is correctly decrypted even though $\Upsilon_e^*(\Gamma_e) \neq 1$. Type 1 adversary (which is the same as the unforgeability adversary of ABS [27]) captures collusion resistance (i.e., $\mathcal{A}$ is allowed to issue $\Gamma_e$ and $\Gamma'_e$ to the sExtract oracle such that $\Upsilon_e^*(\Gamma_e) \neq 1$, $\Upsilon_e^*(\Gamma'_e) \neq 1$, $\Gamma_e \cup \Gamma'_e = \Gamma_e^*$, and $\Upsilon_e^*(\Gamma_e^*) = 1$). Type 2 adversary captures that the Unsigncrypt algorithm does not accept the ciphertext made by $\Gamma_e$ such that $\Upsilon_e^*(\Gamma_e) \neq 1$ with overwhelming probability.

**Definition 7** (S-EUF-DABSC-CMA)**.** *A DABSC scheme is said to be S-EUF-DABSC-CMA secure if the advantage $Adv_{\mathcal{A}}^{S\text{-}EUF\text{-}DABSC\text{-}CMA}(k)$ is negligible for any PPT adversary $\mathcal{A}$ in the S-EUF-DABSC-CMA experiment (defined in Table 2). The notations are defined as follows:*

- sExtract *is the oracle for any $\Gamma_e$, where it returns $\{sk_{e,i}\}_{att_i \in \Gamma_e}$.*
- uExtract *is the oracle for any $\Gamma_d$, where it returns $\{sk_{d,i}\}_{att_i \in \Gamma_d}$.*
- Signcrypt *is the signcryption oracle, where for input of $(M, \Gamma_e, \Upsilon_d)$ it returns the result of* Signcrypt$(params, \Upsilon_e^i, \{sk_{e,i}\}_{att_i \in \Gamma_e}, \Upsilon_d)$*, where $\Upsilon_e^i$ is the current predicate when $\mathcal{A}$ issues the signcryption query.*
- BuildPredicate *is the oracle for input of an access tree $T$, it returns the corresponding public predicate $\Upsilon_e$.*
  - *$\mathcal{A}$ can arbitrary choose the encryptor's access structure via the* BuildPredicate *oracle. Note that, let $\Upsilon_e^* \leftarrow$* BuildPredicate$(params, msk, T_e^*)$ *be the predicate when $\mathcal{A}$ outputs the forged ciphertext.*
  - *$T$ (and the initial access tree $T_0$ also) must follow the condition that leaves of trees appear in $\mathbb{A}_e$.*

## 4 Bottom-up Approach Construction

In this section, we introduce the bottom-up approach construction [15], since we use the following algorithms in the proposed DABSC in a black-box manner. Unlike the top-down approach construction[3], a secret value of the root node is computed in the last of all. Let $T$ be an access tree, where threshold gates are defined on each interior node of the tree, and the leaves are associated with attributes. The set of leaf attributes is a subset of $\mathbb{A}_e = (att_1, att_2, \ldots, att_{n_e})$. Let $\ell_x$ be the number of children of node $x$, and $k_x$ $(0 < k_x \leq \ell_x)$ be the threshold value on the threshold gate of node $x$. When $k_x = 1$ (resp. $k_x = \ell_x$), the threshold gate is called OR gate (resp. AND gate).

[3]This methodology is usually applied in the conventional tree-based cryptosystem. That is, first a secret value of the root node (say $s_T$) is chosen, and next a polynomial $q_{root}(x)$ of degree $(k_{root} - 1)$ is defined such that $q_{root}(0) = s_T$, and a secret value of a child node is set $q_{root}(index(child))$. If a predicate (i.e., the structure of the access tree) is changed, then these procedures must be executed again.

Briefly, the main idea of the bottom-up approach construction is described as follows. For a node $x$, $(\ell_x - k_x)$ dummy nodes are additionally defined, and the threshold value is changed from $k_x$ to $\ell_x$. That is, all threshold gates become AND gates. $k_x$ children (or more) can compute the secret value of their parent node since they can gather $k_x + (\ell_x - k_x) = \ell_x$ values by making up for the lack of $(\ell_x - k_x)$ values from dummy nodes. The secret value of root can be computed by executing this procedure recursively. Dummy nodes can absorb the transformation of the tree structure, and therefore the (actual) secret values assigned with leaves do not have to be updated along with the changing predicate. This methodology is related to share selectable secret sharing [14], where no unauthorized set can obtain information of the secret even if shares are selectable as arbitrary values which are independent of the secret.

**Protocol 1** (Bottom-up Approach Construction [15])**.** *Let $index(\cdot)$ be the function which returns the index of the node, and $p$ be a prime number which is set as a group order in our DABSC. We assume that all nodes (including dummy ones and leaves) are assigned unique index numbers.*

AddDummyNode$(T)$ *: This algorithm adds dummy nodes to the access tree as follows. The algorithm takes as input an access tree $T$, and returns the extended access tree $T^{ext}$ with dummy nodes on $T$ as follows.*

1. *For an interior node $x$, $(\ell_x - k_x)$ dummy nodes are added to $x$'s children, and its threshold value changed from $k_x$ to $\ell_x$. Let $D_T$ be a set of dummy nodes.*
2. *The resulting tree, called $T^{ext}$, is output (we assume that $T^{ext}$ includes $D_T$).*

*Next algorithm, called* AssignedValue*, assigns a set of secret values $S = \{s_j \in \mathbb{Z}_p\}_{att_j \in \mathbb{A}_d}$ for nodes on $T^{ext}$. In our DBSC scheme presented in Section 5, the secret values $S$ are assigned just for the leaves with attributes, and these values will not change when the access tree is changed.*

AssignedValue$(p, S, T^{ext})$ *: This algorithm takes as input $p$, $S$ and $T^{ext}$ and returns a secret value $s_x \in \mathbb{Z}_p$ for each node $x$ of $T^{ext}$. Let $\{child\}_x$ be the set of $x$'s children, except the dummy ones, and $\{d\}_x$ be the set of $x$'s dummy nodes. For an interior node $x$ of $T^{ext}$, a polynomial $q_x$ of degree $(\ell_x - 1)$ is assigned as follows.*

1. *For $att_j \in \{child\}_x$, let $q_x$ be a polynomial of degree at most $(\ell_x - 1)$ which passes though $(index(att_j), s_j)$ for all $s_j \in S$ $(j = 1, 2, \ldots, \ell_x)$.*
2. *For a dummy node $d_j \in \{d\}_x$, the secret value $s_{d_j} := q_x(index(d_j))$ $(j = 1, 2, \ldots, \ell_x - k_x)$ is assigned.*
3. *For $x$, $s_x := q_x(0)$ is assigned.*

*Repeat the above procedure up to the root node, $s_T := q_{root}(0)$ is computed, and is the secret value of the original access tree $T$. Output $(\{s_{d_j}\}_{d_j \in D_T}, s_T)$.*

MakeSimplifiedTree$(\Gamma_e, T^{ext})$ *: This algorithm takes as input the set of attributes $\subseteq \mathbb{A}_e$ (which satisfies the access tree $T$) and $T^{ext}$, and outputs the simplified access tree $T^{\Gamma_e}$ and the set of the product of Lagrange coefficients $\Delta_{\Gamma_e}$.*

1. *The set of redundant attributes $\{att_j\}_{att_j \in \mathbb{A}_e \setminus \Gamma_e}$ are deleted from leaves of $T^{ext}$. In addition, an interior node $x$ that has children less than the threshold value $\ell_x$ is deleted from $T^{ext}$ along with $x$'s descendants. Let $D^{\Gamma_e}$ be the set of dummy nodes and $T^{\Gamma_e}$ be the access tree after these clearances.*
2. *For all nodes $x$ of $T^{\Gamma_e}$ except the root node, $L_x$ is defined as follows:*
   (a) *Define the depth 2 subtree of $T^{\Gamma_e}$ with $x$ as a leaf node. Let $c_x$ be the set of indices of leaves.*
   (b) *Compute $L_x := \prod_{k \in c_x \setminus \{index(x)\}} \frac{-k}{index(x)-k}$.*
3. *Let $leaf \in \{att_j \in \Gamma_e\} \cup \{d_j \in D^{\Gamma_e}\}$ be a leaf node of $T^{\Gamma_e}$. For $leaf$, $\Delta_{leaf}$ is defined as $\Delta_{leaf} := \prod_{node \in Path_{leaf} \setminus root} L_{node}$, where $Path_{leaf} := \{leaf, parent_1, \ldots, parent_{n_{leaf}} = root\}$ is the set of nodes that appears in the path from $leaf$ to the root node. Then, the following equation holds.*

$$\sum_{att_j \in \Gamma_e} \Delta_{att_j} s_j + \sum_{d_j \in D^{\Gamma_e}} \Delta_{d_j} s_{d_j} = s_T$$

4. *Output $T^{\Gamma_e}$, and $\Delta_{\Gamma_e} = (\{\Delta_{att_j}\}_{att_j \in \Gamma_e}, \{\Delta_{d_j}\}_{d_j \in D^{\Gamma_e}})$.*

For the sake of clarity, we introduce the example of the bottom-up approach construction in the Appendix C.

## 5 Proposed DABSC Scheme

In this section, we show our DABSC scheme. The basic idea of our construction is as follows:

- The verification key of the underlying one-time signature $K_v$ is set as the signed message of the signature part (the Li et al. ABS [27]).
- $K_v$ is also applied for achieving the CCA security of the encryption part (the Cheung-Newport CP-ABE scheme [12]).

That is, $K_v$ is applied for combining the ABS part and the CP-ABE part. Since the Li et al. ABS supports the (polynomial-based) threshold structure, we can apply the bottom-up approach construction on the signature part for implementing the dynamic property. In addition, we apply the classical shared randomness methodology, where the encryption part and the signature part share the same randomness ($s$ and $\hat{C} = g^s$ in the actual construction) to reduce the computation costs. These approaches allow us to construct DABSC scheme without random oracles.

**Protocol 2** (The proposed DABSC scheme)**.** *Let $m$ be the length of $K_v$, and $K_{v,i}$ be the $i$-th bit of $K_v$.*

Setup$(1^k)$**:** *Choose a prime number $p$, a bilinear group $(\mathbb{G}, \mathbb{G}_T)$ with order $p$, generators $g, g_1, g_2, h_1, \ldots, h_{n_e}$, $X', X_1, \ldots, X_m \xleftarrow{\$} \mathbb{G}$, and $y, t_1, \ldots, t_{3n_d}, u_1, \ldots, u_{2m}$, $s_1, \ldots, s_{n_e} \xleftarrow{\$} \mathbb{Z}_p$, and compute $Y = e(g,g)^y$, $T_i = g^{t_i}$ $(i = 1, 2, \ldots, 3n_d)$, and $U_i = g^{u_i}$ $(i = 1, 2, \ldots, 2m)$. Output $params = (g, g_1, g_2, Y, \{T_i\}_{i=1}^{3n_d}, \{h_i\}_{i=1}^{n_e}, \{U_i\}_{i=1}^{2m}, X', \{X_i\}_{i=1}^{m})$ and $msk = (y, s_1, \ldots, s_{n_e}, t_1, \ldots, t_{3n_d})$.*

*The public elements $T_i$, $T_{n_d+i}$, and $T_{2n_d+i}$ correspond to the three types of occurrences of $att_i \in \mathbb{A}_d$: positive, negative, and don't care. $U_i$ (resp. $U_{m+i}$) expresses the $i$-th bit of $K_v$ is 0 (resp. 1).*

sExtract$(params, msk, \Gamma_e)$**:** *For $att_i \in \Gamma_e$, choose $v_i \xleftarrow{\$} \mathbb{Z}_p$, compute $d_{i0} = g_2^{s_i}(g_1 h_i)^{v_i}$ and $d_{i1} = g^{v_i}$, and output signing keys $\{sk_{e,i} = (d_{i0}, d_{i1})\}_{att_i \in \Gamma_e}$.*

uExtract$(params, msk, \Gamma_d)$**:** *For all $i = 1, 2, \ldots, n_d$, choose $r_i \xleftarrow{\$} \mathbb{Z}_p$. For all $i = 1, 2, \ldots, m$, choose $\omega_i \xleftarrow{\$} \mathbb{Z}_p$. Set $r := \sum_{i=1}^{n_d} r_i + \sum_{i=1}^{m} \omega_i$ and compute $\hat{D} = g^{y-r}$. Every $att_i \notin \Gamma_d$ is implicitly considered a negative attribute. For $att_i \in \Gamma_d$, compute $D_i = g^{\frac{r_i}{t_i}}$. For $att_i \notin \Gamma_d$ (i.e., for $\neg att_i$), compute $D_i = g^{\frac{r_i}{t_{n_d+i}}}$. For all $i = 1, 2, \ldots, n_d$, compute $F_i = g^{\frac{r_i}{t_{2n_d+i}}}$. For all $i = 1, 2, \ldots, m$, compute $G_{i0} = g^{\frac{\omega_i}{u_i}}$ and $G_{i1} = g^{\frac{\omega_i}{u_{m+i}}}$. Output decryption keys $sk_d = (\{(D_i, F_i)\}_{att_i \in \Gamma_d}, \{G_{i0}, G_{i1}\}_{i=1}^{m}, \hat{D})$. We denote $\{sk_{d,i}\}_{att_i \in \Gamma_d} = (\{(D_i, F_i)\}_{att_i \in \Gamma_d}, \{G_{i0}, G_{i1}\}_{i=1}^{m}, \hat{D})$.*

BuildPredicate$(params, msk, T_\ell)$**:** *Run $T_\ell^{ext} \leftarrow$ AddDummyNode$(T_\ell)$ and $(\{s_{d_j}\}_{d_j \in D_{T_\ell}}, s_{T_\ell}) \leftarrow$ AssignedValue$(p, S, T_\ell^{ext})$, where $S = \{s_i\}_{i=1}^{n_e}$ (which is contained in msk). For all $d_j \in D_{T_\ell}$, choose $h'_j \xleftarrow{\$} \mathbb{G}$ and $v_j \xleftarrow{\$} \mathbb{Z}_p$, and compute $d_{j0} = g_2^{s_{d_j}}(g_1 h'_j)^{v_j}$ and $d_{j1} = g^{v_j}$. Compute $Z_\ell = e(g_2^{s_{T_\ell}}, g)$, and output $\Upsilon_e^\ell = (T_\ell, T_\ell^{ext}, \{(h'_j, d_{j0}, d_{j1})\}_{d_j \in D_{T_\ell}}, Z_\ell)$.*

Signcrypt$(params, \Upsilon_e^\ell, \{sk_{e,i}\}_{att_i \in \Gamma_e}, \Upsilon_d, M)$**:** *Parse $\Upsilon_d = \bigwedge_{i \in I} \underline{i}$. If $\Upsilon_e^\ell(\Gamma_e) \neq 1$, then output $\perp$. Otherwise, run $(T_\ell^{\Gamma_e}, \Delta_{\Gamma_e} = (\{\Delta_{att_j}\}_{att_j \in \Gamma_e}, \{\Delta_{d_j}\}_{d_j \in D^{\Gamma_e}})) \leftarrow$ MakeSimplifiedTree$(\Gamma_e, T_\ell^{ext})$ and $(K_s, K_v) \leftarrow$ Sig.KeyGen$(1^k)$. Parse $K_v = (K_{v,1}, \ldots, K_{v,m})$. Choose $s \xleftarrow{\$} \mathbb{Z}_p$, $r'_i \xleftarrow{\$} \mathbb{Z}_p$ for $i = 1, 2, \ldots, |\Gamma_e|$ and $r''_i \xleftarrow{\$} \mathbb{Z}_p$ for $i = 1, 2, \ldots, |D^{\Gamma_e}|$. Compute*

**Table 3. The Signcrypt Algorithm**

$$\tilde{C} = M \cdot Y^s,\ \hat{C} = g^s,\ C_i = \begin{cases} T_i^s & (\underline{i} = att_i) \\ T_{n_d+i}^s & (\underline{i} = \neg att_i) \\ T_{2n_d+i}^s & (i \in [1, n_d] \setminus I) \end{cases},\ E_i = \begin{cases} U_i^s & (K_{v,i} = 0) \\ U_{m+i}^s & (K_{v,i} = 1) \end{cases},$$

$\sigma_0 = \big(\prod_{att_i \in \Gamma_e} d_{i0}^{\Delta_{att_i}} (g_1 h_i)^{r'_i}\big)\big(\prod_{att_i \in D^{\Gamma_e}} d_{i0}^{\Delta_{d_i}} (g_1 h'_i)^{r''_i}\big)\big(X' \prod_{i=1}^m X_i^{K_{v,i}}\big)^s$,
$\sigma_i = d_{i1}^{\Delta_{att_i}} g^{r'_i}\ \ (i = 1, 2, \ldots, |\Gamma_e|),\ \sigma'_i = d_{i1}^{\Delta_{d_i}} g^{r''_i}\ \ (i = 1, 2, \ldots, |D^{\Gamma_e}|)$, and
$\Sigma \leftarrow \mathsf{Sign}(K_s, \langle \Gamma_e, \Upsilon_d, \tilde{C}, \hat{C}, \{C_i\}_{i=1}^{n_d}, \{E_i\}_{i=1}^m, \sigma_0, \{\sigma_i\}_{att_i \in \Gamma_e}, \{\sigma'_i\}_{d_i \in D^{\Gamma_e}} \rangle)$.

$C = (\Gamma_e, \Upsilon_d, \tilde{C}, \hat{C}, \{C_i\}_{i=1}^{n_d}, \{E_i\}_{i=1}^m, \sigma_0, \{\sigma_i\}_{att_i \in \Gamma_e}, \{\sigma'_i\}_{d_i \in D^{\Gamma_e}}, \Sigma, K_v)$

**Table 4. The Unsigncrypt Algorithm**

Check $1 \stackrel{?}{=} \mathsf{Verify}(K_v, \Sigma, \langle \Gamma_e, \Upsilon_d, \tilde{C}, \hat{C}, \{C_i\}_{i=1}^{n_d}, \{E_i\}_{i=1}^m, \sigma_0, \{\sigma_i\}_{att_i \in \Gamma_e}, \{\sigma'_i\}_{d_i \in D^{\Gamma_e}} \rangle)$
If not, output $\perp$. Otherwise, compute

$$Z_e := \frac{e(\sigma_0, g)}{\prod_{att_i \in \Gamma_e} e(g_1 h_i, \sigma_i) \prod_{d_i \in D^{\Gamma_e}} e(g_1 h'_i, \sigma'_i) e(X' \prod_{i=1}^m X_i^{K_{v,i}}, \hat{C})} \text{ and check } Z_e \stackrel{?}{=} Z_\ell.$$

If not, output $\perp$. Otherwise, output

$$\frac{\tilde{C}}{e(\hat{C}, \hat{D}) \prod_{att_i \in I} e(C_i, D_i) \prod_{att_i \notin I} e(C_i, F_i) \prod_{i \in \{i | K_{v,i} = 0\}} e(E_i, G_{i0}) \prod_{i \in \{i | K_{v,i} = 1\}} e(E_i, G_{i1})} = M$$

$C = (\Gamma_e, \Upsilon_d, \tilde{C}, \hat{C}, \{C_i\}_{i=1}^{n_d}, \{E_i\}_{i=1}^m, \sigma_0, \{\sigma_i\}_{att_i \in \Gamma_e}, \{\sigma'_i\}_{d_i \in D^{\Gamma_e}}, \Sigma, K_v)$ *according to Table 3, and output* $C$.

$\mathsf{Unsigncrypt}(params, \Upsilon_e^\ell, \{sk_{d,i}\}_{att_i \in \Gamma_d}, C)$**:** *Parse* $\Upsilon_d = \bigwedge_{i \in I} \underline{i}$. *Thes algorithm works according to Table 4. Note that the verification phase (i.e.,* $Z_e \stackrel{?}{=} Z_\ell$*) achieves public verifiability [3, 33] (i.e., the decryptor's private key is no longer needed in signature part verification).*

We prove the correctness in the Appendix A.

## 6 Security Analysis

Here we state the theorems describing the security of our DABSC scheme.

**Theorem 1.** *Our DABSC scheme is S-IND-DABSC-CCA2 secure under the DBDH assumption, and the underlying one-time signature is sUF.*

**Theorem 2.** *Our DABSC scheme is S-EUF-DABSC-CMA secure under the CDH assumption and the DBDH assumption.*

Essentially, the proof of Theorem 1 is analogous to the proof of the Cheung-Newport CP-ABE scheme [12] since the secret values of the encryption part and the secret values of the signature part are independent. The signature part of our DABSC scheme is essentially the Waters identity-based signature [44] (or rather, the Li et al. threshold signature [30]). That is, intuitively we can apply the Waters hash technique to prove the unforgeability of the proposed scheme, but it is not clear how to handle the dynamic property using the Waters hash technique alone. So, we give the proof of Theorem 2 in the Appendix B.

## 7 Conclusion

In this paper, we investigate the new concept called DABSC, where access structures of encryptor can be updated flexibly without re-issuing secret keys of users, and propose the formal models of DABSC and the concrete DABSC scheme. It is notable that required complexity assumptions in our proposal are standard ones, i.e., the DBDH assumption and the CDH assumption. Moreover, we do not rely on the random oracles to prove the security. As an application of DABSC, we have introduced authenticated fine-grained storage systems.

## REFERENCES


[1] Nuttapong Attrapadung and Hideki Imai. Dual-policy attribute based encryption: Simultaneous access control with ciphertext and key policies. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 93(1):116–125, 2010.

[2] Nuttapong Attrapadung, Benoît Libert, and Elie de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *Public Key Cryptography*, pages 90–108, 2011.

[3] Feng Bao and Robert H. Deng. A signcryption scheme with signature directly verifiable by public key. In *Public Key Cryptography*, pages 55–59, 1998.

[4] Mihir Bellare and Sarah Shoup. Two-tier signatures, strongly unforgeable signatures, and fiat-shamir without random oracles. In *Public Key Cryptography*, pages 201–216, 2007.

[5] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334, 2007.

[6] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the sdh assumption in bilinear groups. *J. Cryptology*, 21(2):149–177, 2008.

[7] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.

[8] Xavier Boyen. Multipurpose identity-based signcryption (a swiss army knife for identity-based cryptography). In *CRYPTO*, pages 383–399, 2003.

[9] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT*, pages 207–222, 2004.

[10] David Chaum and Eugène van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265, 1991.

[11] Liqun Chen and John Malone-Lee. Improved identity-based signcryption. In *Public Key Cryptography*, pages 362–379, 2005.

[12] Ling Cheung and Calvin C. Newport. Provably secure ciphertext policy ABE. In *ACM Conference on Computer and Communications Security*, pages 456–465, 2007.

[13] Alexander W. Dent, Marc Fischlin, Mark Manulis, Martijn Stam, and Dominique Schröder. Confidential signatures and deterministic signcryption. In *Public Key Cryptography*, pages 462–479, 2010.

[14] Keita Emura, Atsuko Miyaji, Akito Nomura, Mohammad Shahriar Rahman, and Masakazu Soshi. Ideal secret sharing schemes with share selectability. In *ICICS*, pages 143–157, 2011.

[15] Keita Emura, Atsuko Miyaji, and Kazumasa Omote. A dynamic attribute-based group signature scheme and its application in an anonymous survey for the collection of attribute statistics. *Journal of Information Processing*, 17:216–231, 2009.

[16] Keita Emura, Atsuko Miyaji, Kazumasa Omote, Akito Nomura, and Masakazu Soshi. A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. *IJACT*, 2(1):46–59, 2010.

[17] Keita Emura, Atsuko Miyaji, and Mohammad Shahriar Rahman. Toward dynamic attribute-based signcryption (poster). In *ACISP*, pages 439–443, 2011.

[18] Martin Gagné, Shivaramakrishnan Narayan, and Reihaneh Safavi-Naini. Threshold attribute-based signcryption. In *SCN*, pages 154–171, 2010.

[19] Vipul Goyal, Abhishek Jain, Omkant Pandey, and Amit Sahai. Bounded ciphertext policy attribute based encryption. In *ICALP (2)*, pages 579–591, 2008.

[20] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98, 2006.

[21] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, pages 415–432, 2008.

[22] Javier Herranz, Fabien Laguillaumie, and Carla Ràfols. Constant size ciphertexts in threshold attribute-based encryption. In *Public Key Cryptography*, pages 19–34, 2010.

[23] Mauricio Karchmer and Avi Wigderson. On span programs. In *Structure in Complexity Theory Conference*, pages 102–111, 1993.

[24] Dalia Khader. Attribute based group signature with revocation. Cryptology ePrint Archive, Report 2007/241, 2007. `http://eprint.iacr.org/`.

[25] Dalia Khader. Attribute based group signatures. Cryptology ePrint Archive, Report 2007/159, 2007. `http://eprint.iacr.org/`.

[26] Chung Ki Li, Guomin Yang, Duncan S. Wong, Xiaotie Deng, and Sherman S. M. Chow. An efficient signcryption scheme with key privacy. In *EuroPKI*, pages 78–93, 2007.

[27] Jin Li, Man Ho Au, Willy Susilo, Dongqing Xie, and Kui Ren. Attribute-based signature and its applications. In *ASIACCS*, pages 13–16, 2010.

[28] Jin Li and Kwangjo Kim. Hidden attribute-based signatures without anonymity revocation. *International Journal of Information Sciences*, 180(9):1681–1689, 2010.

[29] Jin Li, Kui Ren, Bo Zhu, and Zhiguo Wan. Privacy-aware attribute-based encryption with user accountability. In *ISC*, pages 347–362, 2009.

[30] Jin Li, Tsz Hon Yuen, and Kwangjo Kim. Practical threshold signatures without random oracles. In *ProvSec*, pages 198–207, 2007.

[31] Benoît Libert and Jean-Jacques Quisquater. Efficient signcryption with key privacy from gap diffie-hellman groups. In *Public Key Cryptography*, pages 187–200, 2004.

[32] Benoît Libert and Jean-Jacques Quisquater. Improved signcryption from q-diffie-hellman problems. In *SCN*, pages 220–234, 2004.

[33] Changshe Ma. Efficient short signcryption scheme with public verifiability. In *Inscrypt*, pages 118–129, 2006.

[34] Hemanta Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. Cryptology ePrint Archive, Report 2008/328, 2008. `http://eprint.iacr.org/`.

[35] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In *CT-RSA*, pages 376–392, 2011.

[36] John Malone-Lee. Signcryption with non-interactive non-repudiation. *Des. Codes Cryptography*, 37(1):81–109, 2005.

[37] John Malone-Lee and Wenbo Mao. Two birds one stone: Signcryption using RSA. In *CT-RSA*, pages 211–225, 2003.

[38] Takahiro Matsuda, Kanta Matsuura, and Jacob C. N. Schuldt. Efficient constructions of signcryption schemes and signcryption composability. In *INDOCRYPT*, pages 321–342, 2009.

[39] Takashi Nishide, Kazuki Yoneyama, and Kazuo Ohta. Attribute-based encryption with partially hidden ciphertext policies. *IEICE Transactions*, 92-A(1):22–32, 2009.

[40] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.

[41] S. Sharmila Deva Selvi, S. Sree Vivek, and C. Pandu Rangan. Breaking and fixing of an identity based multi-signcryption scheme. In *ProvSec*, pages 61–75, 2009.

[42] S. Sharmila Deva Selvi, S. Sree Vivek, Rahul Srinivasan, and Chandrasekaran Pandu Rangan. An efficient identity-based signcryption scheme for multiple receivers. In *IWSEC*, pages 71–88, 2009.

[43] Siamak Fayyaz Shahandashti and Reihaneh Safavi-Naini. Threshold attribute-based signatures and their application to anonymous credential systems. In *AFRICACRYPT*, pages 198–216, 2009.

[44] Brent Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, pages 114–127, 2005.

[45] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. Cryptology ePrint Archive, Report 2008/290, 2008.

[46] Shota Yamada, Nuttapong Attrapadung, Goichiro Hanaoka, and Noboru Kunihiro. Generic constructions for chosen-ciphertext secure attribute based encryption. In *Public Key Cryptography*, pages 71–89, 2011.

[47] Yuliang Zheng. Digital signcryption or how to achieve Cost(signature & encryption) << Cost(signature) + Cost(encryption). In *CRYPTO*, pages 165–179, 1997.

## Appendix.A

Correctness is shown from the equations presented in Table 5.

## Appendix.B

Here, we show the proof of Theorem 2.

*Proof.* Let $\mathcal{A}$ be adversary who breaks the S-EUF-DABSC-CMA security of our DABSC scheme. Then, we construct an algorithm $\mathcal{B}$ that breaks the CDH problem as follows. Let $(g, g^a, g^b)$ be the CDH instance. First, $\mathcal{A}$ sends $T_e^*$ to $\mathcal{B}$. Let $\mathbb{A}^*$ be the set of attributes which appear in leaves of $T_e^*$. $\mathcal{B}$ chooses $y, t_1, \ldots, t_{3n_d}, u_1, \ldots, u_{2m}, s'_1, \ldots, s'_{n_e}, z_1, \ldots, z_{n_e} \xleftarrow{\$} \mathbb{Z}_p$, sets $g_1 = g^a$, $g_2 = g^b$, and set $h_i$ and $s_i$ along with whether $att_i \in \mathbb{A}^*$ or not such that

$$h_i = \begin{cases} g^{z_i} & (att_i \in \mathbb{A}_e \setminus \mathbb{A}^*) \\ g^{z_i}/g_1 & (att_i \in \mathbb{A}^*) \end{cases}$$

$$\text{and } s_i = \begin{cases} as'_i & (att_i \in \mathbb{A}_e \setminus \mathbb{A}^*) \\ s'_i & (att_i \in \mathbb{A}^*) \end{cases}$$

**Table 5. Correctness**

$\dagger := e(\hat{C}, \hat{D}) = e(g,g)^{s(y-r)}$, $\ddagger := \prod_{att_i \in I} e(C_i, D_i) \prod_{att_i \notin I} e(C_i, F_i) = e(g,g)^{s \sum_{i \in I} r_i}$,
$\diamond := \prod_{i \in \{i | K_{v,i}=0\}} e(E_i, G_{i0}) \prod_{i \in \{i | K_{v,i}=1\}} e(E_i, G_{i1}) = e(g,g)^{s \sum_{i=1}^{m} \omega_i}$, and so

$\dagger \times \ddagger \times \diamond = e(g,g)^{s(y-r)} e(g,g)^{rs} = e(g,g)^{sy}$ holds

From the above derivations, $\tilde{C}/e(g,g)^{sy} = M$ holds. In addition,

$$\begin{aligned}
e(\sigma_0, g) &= e(\big(\prod_{att_i \in \Gamma_e} d_{i0}^{\Delta_{att_i}} (g_1 h_i)^{r'_i}\big)\big(\prod_{d_i \in D^{\Gamma_e}} d_{i0}^{\Delta_{d_i}} (g_1 h'_i)^{r''_i}\big), g) e((X' \prod_{i=1}^{m} X_i^{K_{v,i}})^s, g) \\
&= e(g_2^{\sum_{att_i \in \Gamma_e} \Delta_{att_i} s_i + \sum_{d_i \in D^{\Gamma_e}} \Delta_{d_i} s_{d_i}}, g) e(\prod_{att_i \in \Gamma_e} (g_1 h_i)^{v_i \Delta_{att_i} + r'_i}, g) e(\prod_{d_i \in D^{\Gamma_e}} (g_1 h'_i)^{v_i \Delta_{d_i} + r''_i}, g) e(X' \prod_{i=1}^{m} X_i^{K_{v,i}}, g^s) \\
&= e(g_2^{s_{T_\ell}}, g) (\prod_{att_i \in \Gamma_e} e((g_1 h_i)^{v_i \Delta_{att_i} + r'_i}, g)) (\prod_{d_i \in D^{\Gamma_e}} e((g_1 h'_i)^{v_i \Delta_{d_i} + r''_i}, g)) e(X' \prod_{i=1}^{m} X_i^{K_{v,i}}, \hat{C}) \\
&= e(g'_\ell, g) (\prod_{att_i \in \Gamma_e} e(g_1 h_i, g^{v_i \Delta_{att_i} + r'_i})) (\prod_{d_i \in D^{\Gamma_e}} e(g_1 h'_i, g^{v_i \Delta_{d_i} + r''_i})) e(X' \prod_{i=1}^{m} X_i^{K_{v,i}}, \hat{C}) \\
&= e(g'_\ell, g) (\prod_{att_i \in \Gamma_e} e(g_1 h_i, \sigma_i)) (\prod_{d_i \in D^{\Gamma_e}} e(g_1 h'_i, \sigma'_i)) e(X' \prod_{i=1}^{m} X_i^{K_{v,i}}, \hat{C}), \text{ and so}
\end{aligned}$$

$Z_e = e(g'_\ell, g) = Z_\ell$ holds.

Let $m' = 4q_s$, where $q_s$ is the number of the uExtract queries. $\mathcal{B}$ chooses $m$-length vector $\overrightarrow{x} = (x_i)$, where the elements of $\overrightarrow{x}$ are chosen uniformly at random from the integers between 0 and $m'-1$, $x'$, $y'$, and the elements of $m$-length vector $\overrightarrow{y}$ are also chosen uniformly at random from the integers between 0 and $m'-1$. $\mathcal{B}$ also chooses an integer $k$ uniformly at random from the integers between 1 and $m$. For $K_v \in \{0,1\}^m$, we define three functions as follows.

$$\begin{aligned}
F(K_v) &= (p - m'k) + x' + \sum_{i=1}^{m} x_i^{K_{v,i}} \\
J(K_v) &= y' + \sum_{i=1}^{m} y_i^{K_{v,i}} \\
K(K_v) &= \begin{cases} 0 & (\text{If } x' + \sum_{i=1}^{m} x_i^{K_{v,i}} \equiv 0 \pmod{m'}) \\ 1 & (\text{Otherwise}) \end{cases}
\end{aligned}$$

In addition, $\mathcal{B}$ computes $Y = e(g,g)^y$, $T_i = g^{t_i}$ $(i = 1, 2, \ldots, 3n_d)$, $X' = g_1^{p-km+x'} g^{y'}$, and $X_i = g_1^{x_i} g^{y_i}$ $(i = 1, 2, \ldots, m)$. $\mathcal{B}$ gives $params = (g, g_1, g_2, Y, \{T_i\}_{i=1}^{3n_d}, \{h_i\}_{i=1}^{n_e}, \{U_i\}_{i=1}^{2m}, X', \{X_i\}_{i=1}^{m})$ to $\mathcal{A}$. Since $\mathcal{B}$ knows $(y, \{t_i\}_{i=1}^{3n_d}, \{u_i\}_{i=1}^{2m})$, $\mathcal{B}$ can answer any uExtract queires issued by $\mathcal{A}$.

For a sExtract query $\Gamma_e$, $\mathcal{B}$ returns the legitimate keys $\{sk_{e,i}\}_{att_i \in \Gamma_e}$ as follows:

**The case** $att_i \in \Gamma_e \setminus \mathbb{A}^*$ (i.e., $s_i = a s'_i$)

1. Choose $\tilde{v}_i \xleftarrow{\$} \mathbb{Z}_p$, and set $v_i := -b s'_i + \tilde{v}_i$.
2. Compute $d_{i0} = (g^a)^{\tilde{v}_i} (g^b)^{-x_i s'_i} \cdot g^{x_i \tilde{v}_i}$ and $d_{i1} = (g^b)^{-s'_i} \cdot g^{\tilde{v}_i}$. These are legitimate keys since the following equations hold.

$$\begin{aligned}
(g^a)^{\tilde{v}_i} (g^b)^{-z_i s'_i} \cdot g^{z_i \tilde{v}_i} &= (g^a)^{\tilde{v}_i} h_i^{-b s'_i + \tilde{v}_i} \\
&= g^{ab s'_i} g^{-ab s'_i} (g^a)^{\tilde{v}_i} (h_i)^{-b s'_i + \tilde{v}_i} \\
&= g^{ab s'_i} (g^a)^{-b s'_i + \tilde{v}_i} (h_i)^{-b s'_i + \tilde{v}_i} \\
&= g^{ab s'_i} (g^a h_i)^{-b s'_i + \tilde{v}_i} \\
&= g_2^{s_i} (g_1 h_i)^{v_i}, \text{ and} \\
(g^b)^{-s'_i} \cdot g^{\tilde{v}_i} &= g^{v_i}
\end{aligned}$$

**The case** $att_i \in \Gamma_e \cap \mathbb{A}^*$ (i.e., $s_i = s'_i$)

- Choose $v_i \xleftarrow{\$} \mathbb{Z}_p$, and compute $d_{i0} = g_2^{s'_i} (g_1 h_i)^{v_i}$ and $d_{i1} = g^{v_i}$.

For a BuildPredicate query $T_\ell$, $s_{T_\ell}$ can be represented such that $s_{T_\ell} = \alpha_\ell + a\beta_\ell$ for some $\alpha_\ell, \beta_\ell \in \mathbb{Z}_p$ where $\alpha_\ell$ and $\beta_\ell$ can be represented by $s'_i$ and Lagrange coefficients. That is, $\mathcal{B}$ can compute $\alpha_\ell$ and $\beta_\ell$. So, $Z_\ell$ can be computed by $\mathcal{B}$ such that $Z_\ell = e(g^b, g^{\alpha_\ell} (g^a)^{\beta_\ell}) = e(g_2, g^{s_{T_\ell}}) = e(g_2^{s_{T_\ell}}, g)$. Note that $(d_{j0}, d_{j1})$ can be computed by using the same simulation of the sExtract oracle. We denote $z'_i$ for defining $h'_i$ such that

$$h'_i = \begin{cases} g^{z'_i} & (d_i \notin D^{\mathbb{A}^*}) \\ g^{z'_i}/g_1 & (d_i \in D^{\mathbb{A}^*}) \end{cases}$$

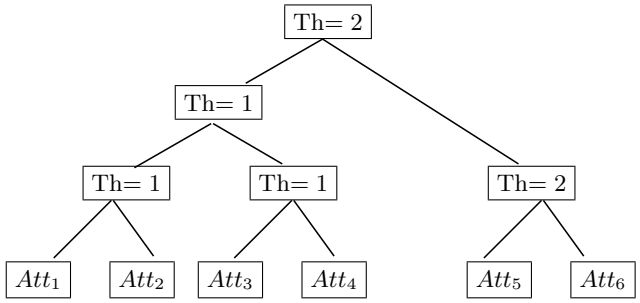
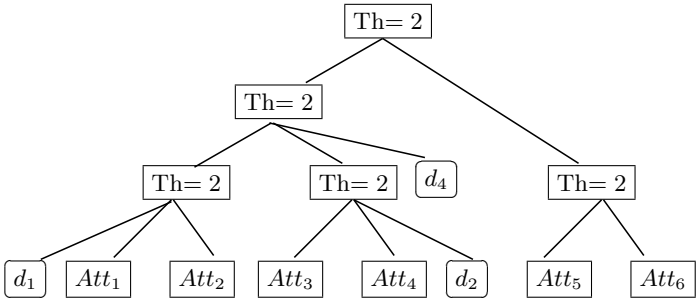
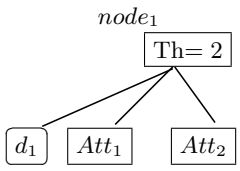
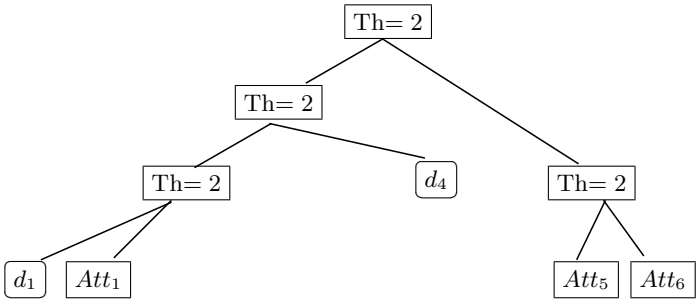
For a Signcrypt query $(M, \Gamma_e, \Upsilon_d)$, since $\mathcal{B}$ knows all values for computing a ciphertext, $\mathcal{B}$ can answer the query.

Finally, $\mathcal{A}$ outputs $C^* = (\Gamma_e^*, \Upsilon_d^*, \tilde{C}^*, \hat{C}^*, \{C_i^*\}_{i=1}^{n_d}, \{E_i^*\}_{i=1}^{m}, \sigma_0^*, \{\sigma_i\}_{att_i \in \Gamma_e^*}, \{\sigma'_i\}_{d_i \in D^{\Gamma_e^*}}, \Sigma^*, K_v^*)$. If $K(K_v^*) = 1$, then $\mathcal{B}$ aborts.

**Type 1 Adversary** : Since $s_i := s'_i$ for all $att_i \in \mathbb{A}^*$, $s_{T_e^*}$ is represented as follows.

$$s_{T^*} = \alpha^* + a\beta^* \text{ where } \alpha^* \in \mathbb{Z}_p \text{ and } \beta^* = 0$$

**Table 6. Access Trees**

| Access Tree $T$ | Extended Access Tree $T^{ext}$ |
|---|---|
|  |  |
| **Assignment of secret values on $T^{ext}$** | **Simplified Access Tree $T^{\Gamma_e}$ ($\Gamma_e = \{att_1, att_5, att_6\}$)** |
|  $s_1$ and $s_2$ are pre-determined. $f_{node_1}(x) := \frac{x-index(att_1)}{index(att_2)-index(att_1)} s_1 + \frac{x-index(att_2)}{index(att_1)-index(att_2)} s_2$ $s_{node_1} := f_{node_1}(0)$ $s_{d_1} := f_{node_1}(index(d_1))$ |  |

So, we need to modify $s_{T^*}$ for embedding the CDH instance. Let $s'_{T^*} := as_{T^*}$ be the modified root secret value of $T^*$. Note that the value related $s_{T^*}$ including $\Upsilon^*_e$ is $Z^* := e(g_2^{s_{T^*}}, g)$ only. In addition, $Z^*_{\text{modify}} := e(g_2^{s'_{T^*}}, g) = e((g^b)^{s_{T^*}}, g^a)$ and $Z^*$ is computationally indistinguishable under the DBDH assumption. So, $\mathcal{B}$ publishes $Z^*_{\text{modify}}$ instead of $Z^*$. Note that type 1 $\mathcal{A}$ cannot issue $\Gamma_e$ to the sExtract oracle such that $\Upsilon^*_e(\Gamma_e) = 1$, and therefore $\mathcal{A}$ cannot use $\{sk_{e,i}\}_{att_i \in \Gamma_e}$ (with the condition $\Upsilon^*_e(\Gamma_e) = 1$) to distinguish $Z^*_{\text{modify}}$ and $Z^*$. From $C^*$, $\mathcal{B}$ can compute $g^{ab}$ such that:

$$\begin{aligned} & \Big(\frac{\sigma^*_0}{(\hat{C}^*)^{J(K^*_v)} \prod_{att_i \in \Gamma^*_e} \sigma_i^{z_i} \prod_{d_i \in D^{\Gamma^*_e}} \sigma_i'^{z'_i}}\Big)^{1/\alpha^*} \\ = \ & (g_2^{s'_{T^*_e}})^{1/\alpha^*} \\ = \ & g_2^a \\ = \ & g^{ab} \end{aligned}$$

Note that the probability $\alpha^* = 0$ is $1/p$ and is negligible.

**Type 2 Adversary** : Since $\Upsilon^*_e(\Gamma^*_e) \neq 1$, there exists $att_{i^*} \in \mathbb{A}_e \setminus \mathbb{A}^*$ such that $att_{i^*} \in \{att | att \in \Gamma_e \text{ s.t. } \Upsilon^*_e(\Gamma_e) = 1\}$. That is, since $s_{i^*} := as'_{i^*}$, $s_{T^*_e}$ is represented as follows.

$$s_{T^*} = \alpha^* + a\beta^* \text{ where } \alpha^*, \beta^* \in \mathbb{Z}_p \text{ and } \beta^* \neq 0$$

From $C^*$, $\mathcal{B}$ can compute $g^{ab}$ such that:

$$\begin{aligned} & \Big(\big(\frac{\sigma^*_0}{(\hat{C}^*)^{J(K^*_v)} \prod_{att_i \in \Gamma^*_e} \sigma_i^{z_i} \prod_{d_i \in D^{\Gamma^*_e}} \sigma_i'^{z'_i}}\big)/g^{\alpha^*}\Big)^{1/\beta^*} \\ = \ & (g_2^{s_{T^*_e}}/g^{\alpha^*})^{1/\beta^*} \\ = \ & g_2^a \\ = \ & g^{ab} \end{aligned}$$

□

## Appendix.C

Here, we introduce the example of the bottom-up construction in Table 6. In this example, $s_1, s_2, \ldots, s_6 \in S$ are predetermined, and are assigned for the leaves with attributes $Att_1, Att_2, \ldots, Att_6$. According to the tree structure, the secret values assigned with the dummy nodes (e.g., $s_{d_1}$) or interior nodes (e.g., $s_{node_1}$) are modified. But, $\{s_i\}_{i=1}^6$ will not change when the access tree is changed.