Mathematical Foundations of Computing
©American Institute of Mathematical Sciences
Volume **1**, Number **1**, February **2018**
doi:10.3934/mfc.2018005
pp. **77–100**

CP_ABSC: AN ATTRIBUTE-BASED SIGNCRYPTION SCHEME TO SECURE MULTICAST COMMUNICATIONS IN SMART GRIDS

Chunqiang Hu[1,2], Jiguo Yu[3], Xiuzhen Cheng[4], Zhi Tian[5]
Kemal Akkaya[6] and Limin Sun[7]

[1] Key Laboratory of Dependable Service Computing in Cyber Physical Society
Ministry of Education, Chongqing University, China
[2] School of Software Engineering, Chongqing University, China
[3] School of Information Science and Engineering, Qufu Normal University, China
[4] Department of Computer Science, The George Washington University, USA
[5] Electrical & Computer Engineering Department, George Mason University, USA
[6] Electrical & Computer Engineering Department, Florida International University, USA
[7] Beijing Key Laboratory of IOT Information Security Technology
Institute of Information Engineering
Chinese Academy of Sciences (CAS), China

(Communicated by Zhipeng Cai)

Abstract. In this paper, we present a signcryption scheme called CP_ABSC based on Ciphertext-Policy Attribute Based Encryption (CP_ABE) [7] to secure the multicast communications in smart grids that require access control, data encryption, and authentication to ensure message integrity and confidentiality. CP_ABSC provides algorithms for key management, signcryption, and designcryption. It can be used to signcrypt a message based on the access rights specified by the message itself. A user can designcrypt a ciphertext if and only if it possesses the attributes required by the access structure of the data. Thus CP_ABSC effectively defines a multicast group based on the access rights of the data specified by the data itself, which differs significantly from the traditional Internet based multicast where the destination group is predetermined and must be known by the data source. CP_ABSC provides collusion attack resistance, message authentication, forgery prevention, and confidentiality. It can be easily applied in smart grids to secure the instructions/commands broadcast from a utility company to multiple smart meters (push-based multicast) and the data retrieved from a smart meter to multiple destinations (pull-based multicast). Compared to CP_ABE, CP_ABSC combines encryption with signature at a lower computational cost for signcryption and a slightly higher cost in designcryption for signature verification. We also consider the adoption of attribute-based signature (ABS), and conclude that CP_ABSC has a much lower computational cost than ABS.

2010 *Mathematics Subject Classification.* Primary: 58F15, 58F17; Secondary: 53C35.
Key words and phrases. Smart meters, smart grids, secure communication mechanisms, access control, bilinear maps, signcryption.
The preliminary version of this paper appears in [16].
This research was partially supported by the National Natural Science Foundation of China under grants 61702062, 61373027 and 61472418, and the National Science Foundation of the US under grants CCF-1442642, IIS-1343976, CNS-1318872, and CNS-1550313.
* Corresponding author: Chunqiang Hu.

1. **Introduction.** Recently, smart grids have gained tremendous attention among researchers and engineers as they bring new features such as renewable-based generation, demand-response, wide area protection, and smart metering into the power grids [23]. Despite these attractive features, smart grids face many challenges, especially in cybersecurity and privacy [1]. For example, integrated Supervisory Control and Data Acquisition (SCADA)/Energy Management systems [5] were reported to suffer significant security breaches [25, 45, 51]. In this paper, we focus on the problem of securing multicast communications in smart grids, and consider the following three security properties [3, 12, 17, 39, 47]:

- *Integrity:* An adversary may modify or destruct the messages (e.g., content, timing, sequence order, etc.) between the power company and the data sources before they are transmitted, or may manipulate the message contents being transmitted. This attack raises significant safety concerns, which may lead to wrong decisions on power management.
- *Confidentiality:* Smart meter readings and instructions from utility companies or power grid control units should be protected to prevent eavesdropping attacks during transmissions as the fine-grained smart meter data can release sensitive information (e.g., presence or absence of human beings) of a household, resulting in privacy leakage and safety concerns [12], and the exposed instructions may disclose sensitive information such as the supply and demand of a micro-grid, based on which potential attacks may lead to the dysfunction of the whole smart grid architecture [4, 43].
- *Authentication and access control.* In smart grid information systems, access control is the key to ensuring the legal access of sensitive data and authentication can guarantee the legitimacy of the data and data source. For examples, unauthorized accesses to the smart meter data can reveal the details of household activities, and an unauthenticated software update from a spoofed service provider could make the attacked smart device function disorderly.

This paper addresses the challenges of securing multicast communications among smart meters and service providers in a smart grid. There are two types of multicasts under our consideration: *push-based* multicast for command distribution to multiple smart meters from a service provider and *pull-based* multicast to support asynchronous downloads of the smart meter data from a data repository. A common feature of these two types of multicasts compared to traditional Internet-based multicasts is their lack of unique identities to define the destination multicast group: the recipients of the command/data actually are specified by a set of attributes. For example, an instruction from a refrigerator manufacturer can specify that Model X in Washington DC manufactured in the year 2000 needs a software update. The set of attributes, namely, the model "X", the location "Washington DC", and the time "year 2000", clearly defines the group of refrigerators whose software should be updated, which implies that the manufacturer does not need to know about the details of each refrigerator being sold. This is quite convenient as requiring each refrigerator that has been sold to be registered with its manufacturer is not feasible in practice.

Existing multicast schemes such as TELSA, Biba, HORS, and OTS [8, 9, 11, 19, 23, 31, 33, 34, 36, 44] are all push-based. They mainly focus on authentication, largely ignoring access control and confidentiality. On the other hand, pull-based multicast in which each recipient retrieves the data at its own will when needed, is as desirable

in smart grids. For example, multiple service providers may need to retrieve the fine-grained electricity usage data of a household for different purposes at different times; thus the smart meter should store its data at a repository for future (asynchronous) downloads. This poses significant security and privacy concerns because the access of the data in a repository is completely out of the control of the smart meter that generates the data but it should be the data source's decision regarding who can access the data – a service provider in California may not need the utility data of a house in Washington DC.

In this paper, we propose a signcryption scheme termed CP_ABSC based on Ciphertext-Policy Attribute-based Encryption (CP_ABE) [7] to support both pull-based and push-based secure multicasts in smart grids. CP_ABSC combines signature and encryption, and provides a new mechanism for data encryption, access control, and authentication. The basic idea of CP_ABSC is to signcrypt the message/data based on its access policy (represented by an access tree and specified by the data (data source) itself) and designcrypt the corresponding ciphertext with a secret key computed from a set of attributes. The access tree defines the access rights of the data based on the attributes and is carried by the ciphertext. Only when the recipient possessing the set of attributes that satisfy the access policy carried by the ciphertext can successfully decrypt and authenticate the original message/data. Because a multicast group is uniquely defined by the data itself via the access policy, secure multicasts are effectively achieved.

The contribution of this paper can be summarized as follows:

- We develop a novel scheme called Ciphertext-policy Attribute Based Signcryption (CP_ABSC). The proposed scheme ensures security and privacy of the message/data in smart grids (e.g., commands and utility data) by performing signature and encryption in one operation.
- We prove the correctness of CP_ABSC and analyze its efficiency and feasibility. In particular, we discuss the security strength of CP_ABSC under four major attack scenarios: collusion, message authentication, forgery, and confidentiality. We also conduct a quantitative performance analysis, and our results indicate that CP_ABSC is computationally efficient and feasible.
- We demonstrate how to apply the proposed signcryption scheme to secure different data communications in smart grids. Particularly, we develop a protocol to secure the instructions sent from utility companies to smart meters (push-based multicast); we also develop a procedure for the smart meter data to be securely stored and asynchronously accessed (downloaded) by different service providers based on CP_ABSC (pull-based multicast).
- We consider the alternative of applying attribute based signature (ABS) to achieve anonymous communications and analyze its performance in terms of computational cost. Our results indicate that adopting ABS incurs much higher cost compared to CP_ABSC.

The remainder of this paper is structured as follows: In Section 2, we summarize the most related work. Section 3 introduces preliminaries and the system model. Section 4 presents our signcryption scheme CP_ABSC, proves its correctness, and analyzes its security strength and computational performance. In Section 5, we compare and contrast CP_ABSC with the two most related ABE schemes: CP_ABE and KP_ABE. Section 6 illustrates how to use CP_ABSC to secure multicast communications in smart grids. In Section 7, we present how to adopt attribute-based

signature to achieve anonymous communications and analyzes its performance in terms of computational cost. Finally, the paper is concluded with Section8.

2. **Related work.** Smart grids are heterogeneous networks composed of different architectures. Security problems could involve various components of a smart grid. Liu *et al.* [24] investigated cyber security and privacy in smart grids, pointing out further study areas to enhance the security level of the grids. Sun *et al.* [49] claimed that Ethernet Passive Optical Networks (EPON) would be a promising solution for smart grid broadband access networks, and proposed a secure communication protocol for EPON by using identity-based cryptography, which generates a public key from an arbitrary data string, and binds the corresponding private key with the information. This work lacks a performance analysis in terms of package overhead and scalability. Metke and Ekl [29] claimed that wireless smart grids can be more secure with existing standards such as 802.16e (Mobile WiMax); but they did not analyze the feasibility of their proposed scheme in smart grids.

Securing smart grid multicast communications require *access control*, *data encryption*, and *authentication*. Nevertheless, existing research [10,22,23,32,48] either does not address all requirements mentioned above or is too computationally inefficient. In [22], the authors proposed a privacy preservation scheme for smart grid multicast communications that combines both centralized and contributory group key schemes [46] [20] to protect the privacy of smart grid multicast messages. However this scheme does not consider message authentication. Li and Cao considered multicast authentication in smart grids via one time signature [23], which ignores data confidentiality and considers only message authentication. Nicanfar *et al.* [32] proposed an authentication and key management scheme for smart grid unicast and multicast communications, but this scheme does not consider data confidentiality. These schemes share the following common feature: the destination group needs to be predetermined and known by the data source. In our study, we assume that the multicast group is defined by an access policy specified by the data source based on an attribute set, which implies that all recipient satisfying the access policy is a legitimate receiver of the message.

As the design of CP_ABSC is motivated by CP_ABE [7], we summarize the research most related to Identity-Based Encryption (IBE) [41] and Attribute-Based Encryption (ABE) [38] here. There exist two different and complementary notions of ABE: Key-Policy ABE (KP_ABE) [15] and Ciphertext-Policy ABE (CP_ABE) [7]. The key features of KP_ABE and CP_ABE are summarized in Section 5, where we compare them with our proposed scheme CP_ABSC. A new construction of CP_ABE, named Constant-sized CP_ABE (denoted as CCP_ABE), was presented in [50], which reduces the ciphertext length to a constant size for an AND gate access policy with any given number of attributes at the cost of long secret keys and complicated access structures. A scheme that employs IBE to provide a zero-configuration encryption and authentication solution for end-to-end secure communications was proposed in [42]. The concept of IBE was utilized by [26] to construct a signature and later verify the signature. KP_ABE was adopted by [13] to broadcast a single encrypted message to a specific group of users. The Lewko-Waters ABE scheme [21], which was based on Linear Secret Sharing to construct the access policy, was used by [37] to ensure access control but the data source does not have control over its data. Note that the schemes mentioned above can not ensure message integrity and confidentiality. A signcryption scheme based on KP_ABE was proposed in [14],

which does not meet the requirements of many practical applications as the data source can not intelligently decide who should or should not have access to its data.

In this paper, we present a signcryption scheme termed Ciphertext-Policy Attribute-Based SignCryption (CP_ABSC) to provide the security services required by the multicast communications for our system model proposed in Section 3.2, where the multicast destinations are defined by an access policy carried by the ciphertext. CP_ABSC is applied to secure the push-based multicasts of instructions/commands from a service provider to multiple smart meters. It is also employed to signcrypt the smart grid data stored in the data repositories and designcrypt the ciphertext retrieved by multiple verified service providers when needed (asynchronous pull-based multicast). Compared to CP_ABE, CP_ABSC provides both encryption and signature without significantly increasing the computational cost (actually only the computational cost of designcryption is slightly increased compared to CP_ABE due to signature verification in CP_ABSC). CP_ABSC has strong security strength in terms of collusion resistance, message authentication, forgery prevention, and confidentiality.

3. **Preliminaries and system model.** In this section, we present the required preliminary knowledge and our system model.

3.1. **Preliminaries.** We now introduce the preliminary knowledge employed by the cryptographic algorithms of CP_ABSC.

3.1.1. *Bilinear mapping and the bilinear Diffie-Hellman problem.* Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_3$ be three bilinear groups of prime order p, and let g_1 be a generator of $\mathbb{G}_1$ and g_2 be a generator of $\mathbb{G}_2$. Our proposed scheme makes use of a bilinear mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ with the following properties:

1. *Bilinear:* A mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ is bilinear if and only if for $\forall P \in \mathbb{G}_1, \forall Q \in \mathbb{G}_2$, and $\forall a, b \in \mathbb{Z}_p$, $e(P^a, Q^b) = e(P, Q)^{ab}$ holds. Here $\mathbb{Z}_p = \{0, 1, \ldots, p-1\}$ is a Galois field of order p.
2. *Non-degeneracy:* The generator g_1 and g_2 satisfies $e(g_1, g_2) \neq 1$.
3. *Computability:* There is an efficient algorithm to compute $e(P, Q)$ for $\forall P, Q \in \mathbb{G}_1 \times \mathbb{G}_2$.

With a bilinear mapping, one can get the following **Bilinear Diffie-Hellman problem (BDH)**: Given three groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_3$ of the same prime order p. Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ be a bilinear mapping and g_1, g_2 be respectively the generator of $\mathbb{G}_1$ and $\mathbb{G}_2$. The objective of BDH is to compute $e(g_1, g_2)^{abc}$, where $a, b, c \in \mathbb{Z}_p$, from the given $(g_1, g_1^a, g_1^c, g_2, g_2^a, g_2^c)$.

Note that the hardness [38] of the decision version of BDH, i.e., the decisional bilinear Diffie-Hellman problem (DBDH), forms the basis for the security of our scheme CP_ABSC.

3.1.2. *Secret sharing.* Another important cryptographic primitive used by CP_ABSC is secret sharing [18, 40]. In the context of a *dealer* sharing a secret with n *participants* $u_1, \ldots, u_n$, a participant learns the secret if and only if it can cooperate with at least $t-1$ other participants (on sharing what they learn from the dealer), where $t \leq n$ is a pre-determined parameter. The secret to be shared by the dealer is $s \in \mathbb{Z}_p$, where $p > n$. Before secret sharing, each participant u_i holds a secret key $k_i \in \mathbb{Z}_p$, which is only known by u_i and the dealer.

The dealer follows a two-step process. First, it constructs a polynomial function $f(z)$ of degree $t-1$, i.e.,

$$f(z) = s + \sum_{j=1}^{t-1} a_j z^j, \tag{1}$$

by randomly choosing $t-1$ i.i.d. coefficients (the a_j's) from $\mathbb{Z}_p$. Note that all (additive and multiplication) operations used in (1) and throughout the rest of the paper are modular arithmetic (defined over $\mathbb{Z}_p$) as opposed to real arithmetic. Also note that s forms the constant component of $f(z)$, i.e., $s = f(0)$. Then, in the second step, the dealer transmits to each u_i a secret share s_i computed from k_i (see (2)), the secret key known only by u_i and the dealer.

$$s_i = f(k_i), \tag{2}$$

We now show how t or more users can cooperate to recover s by sharing the secret shares received from the dealer. Without loss of generality, let $u_1, \ldots, u_t$ be the cooperating users. These t users can reconstruct the secret $s = f(0)$ from $s_1 = f(k_1), \ldots, s_t = f(k_t)$ by computing

$$s = f(0) = \sum_{j=1}^{t} \left(s_j \prod_{i \in [1,t], i \neq j} \frac{0 - k_i}{k_i - k_j} \right). \tag{3}$$

Note that the cumulative product in (3) is essentially a Lagrange coefficient. The correctness of (3) can be easily verified based on the definition of $f(z)$.

3.2. **System model.** We consider a smart grid communication system depicted in Figure 1. There are four major entities in this system: KeyGeneration Center (KGC), Smart Meter, Data Repository, and Service Provider. Below, we summarize the major functions of each entity.

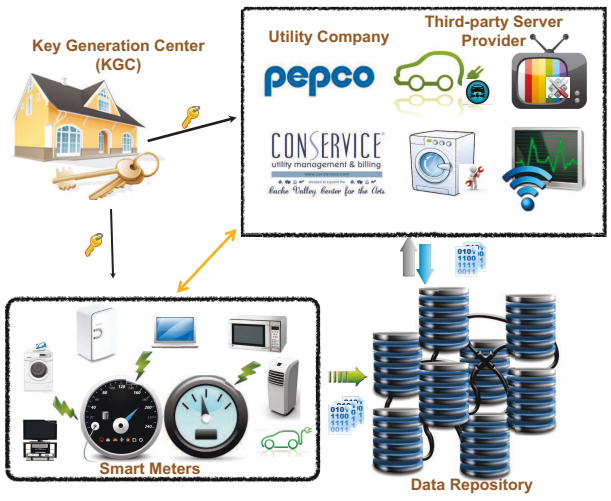

FIGURE 1. A communication architecture in smart grid systems.

3.2.1. ***Key Generation Center (KGC).*** The KGC generates and distributes keys for smart meters and service providers.

3.2.2. ***Smart meter.*** Smart metering plays a pivotal rule in any smart grid system. A smart meter acts as a gateway between the internal and the external entities and protects user privacy by hiding individual components from the electric utility companies. It is the smart meter instead of a service provider that directly controls individual household devices. When an electric utility company requests to reduce the overall power consumption, the smart meter can determine which device to shut down. It can also limit the electric usage of certain devices according to the customer's priority settings.

We assume that a smart meter has sufficient computational capacity to signcrypt the data produced by household devices and designcrypt the messages sent by service providers. When a service provider wants to collect the power consumption information of its customers' devices, it needs to communicate with a data repository and retrieve the (signcrypted) data.

3.2.3. ***Service provider.*** Service Providers refer to Utility Companies (UC) and Third-Party Service Providers (TPSP). A UC can send consumption related instructions and emergency/error notifications to smart meters and collect sub-hourly power usage reports. It can also interact with smart meters in regulating power consumption levels. For example, to reduce power loads during peak hours, a UC can instruct smart meters to limit their usage. It can then be up to the smart meters to regulate their household devices. This approach hides individual devices from the UC and protects user privacy. A TPSP could be a device manufacture that may need to upgrade the software of a device.

3.2.4. ***Data repository.*** Data repositories store the encrypted power usage data. A data repository can have multiple data storage centers, which can provide efficient search mechanisms and help organize the data in the databases. The data stored in a repository can be accessed by service providers and other smart grid entities that may need the data for demand and supply control.

This system model involves two types of multicast communications: the multicast of instructions/commands from a service provider to all the smart meters (push-based multicast), and the retrieval of the smart meter data from a repository by multiple service providers (pull-based multicast), with the destinations in both types of multicasts being defined by a set of attributes instead of unique IDs as in traditional networks. The keys computed by KGC ensure that the smart meter data is securely stored in a repository and retrieved only by verified users when needed, and that the instructions indeed come from verified service providers.

4. CP_ABSC: A Ciphertext-Policy Attribute Based Signcryption scheme.

4.1. **Access control policy – the access tree.** Our main idea is to design an attribute-based signcryption scheme that views an identity as a set of attributes, and enforces a lower bound on the number of common attributes between a user's identity and its access rights specified by the sensitive data. We use an access tree structure proposed by [7], which is illustrated in Fig. 2, to control the data consumer's access to the encrypted data. In Fig. 2, each non-leaf node x is associated with two parameters, num_x and k_x, where num_x is the number of child nodes of node x, and $k_x \in [1, num_x]$ is its threshold value indicating that node x performs the OR operation over all subsets of k_x child nodes of x, with each subset supporting an AND operation; each leaf node x is described by an attribute and a threshold value $k_x = 1$. We also associate an index with each node x in T, denoted by $index(x)$.

Since a tree with $|S|$ number of attributes can have at most $2|S|-1$ nodes, we can assign a unique number in $\{1, 2, \cdots, 2|S|-1\}$ to each node in the tree based on pre-order tree traversal. Other tree traversal techniques such as in-order or post-order can also be applied. Let $parent(x)$ be the parent node of x in T.

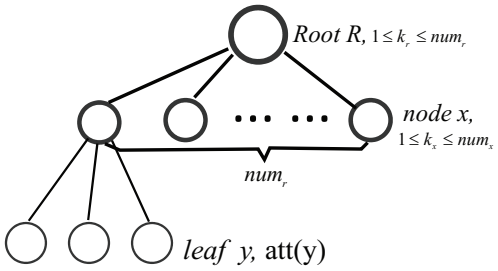

FIGURE 2. An access control tree structure.

For example, a user may specify the following access structure for a data item: (Third-Party Service Provider AND (Arlington, VA OR Washington, DC)), which indicates that only the third-party service providers in Arlington, VA or Washington, DC have the access to this data. Thus a user located in Washington DC with a set of attributes {Third-Party Service Provider, Washington DC, Air-Conditioner} has the right to access the data mentioned above. The corresponding access control tree for this example is illustrated in Fig. 3.

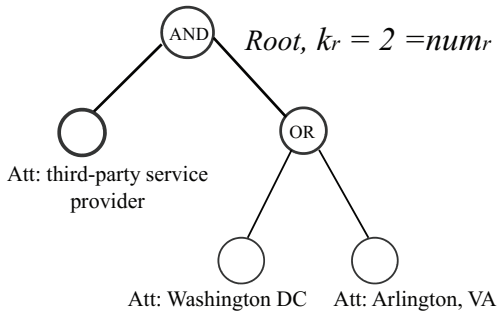

FIGURE 3. An example access control structure in Smart Grid.

4.2. **CP_ABSC: Ciphertext-Policy Attribute Based Signcryption.** In this subsection, we propose CP_ABSC, a Ciphertext-Policy Attribute Based SignCryption scheme. CP_ABSC consists of four primitive algorithms. **Algorithm** 1 is executed by KGC to provide system initialization. It generates and distributes the public parameters of the system to all the involved entities.

Algorithm 2 is executed by KGC to generate three keys for an attribute set S: a designcryption key SK, a signing key K_{sign}, and a verification key K_{ver}. The designcryption key is the private key of the user possessing S. More specifically, for each attribute set S, KGC: i) generates a private key SK, which is adopted by the owner of S to designcrypt ciphertexts; ii) produces a signing key K_{sign}, which is utilized by the owner of S to sign its ciphertext messages; and iii) constructs and

Algorithm 1 System Initialization [7]

1: Select a prime p, the generators g_1 and g_2 of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and a bilinear mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$.
2: Choose two random exponents $\alpha, \beta \in \mathbb{Z}_p$.
3: Select a hash function $H_1 : \{0,1\}^* \to \mathbb{Z}_p$. This function H_1 is viewed as a random oracle.
4: Publish the public parameters given by

$$PK = (p, \mathbb{G}_1, \mathbb{G}_2, H_1, g_1, g_2, h = g_1^{\beta}, t = e(g_1, g_2)^{\alpha}) \tag{4}$$

5: Compute the master key $MSK = (\beta, g_2^{\alpha})$.

publishes a verification key K_{ver}, which is used by others to verify the signatures signed by the owner of S.

Algorithm 3 details the signcryption procedure, which is the core of the proposed CP_ABSC. This algorithm is mainly performed by the senders (data sources) to signcrypt their data before transmitting to the data repositories or the receivers. In a typical application, a sender encrypts a message/data M whose access control is specified by an access tree T, and signs the message by the signing key computed from **Algorithm** 2. **Algorithm** 3 is designed to provide confidentiality, integrity, authentication, and non-repudiation, for the purpose of ensuring the security and privacy of the data sources.

Algorithm 4 implements verification and decryption, which should be executed by the receivers to get the plaintext data based on their attributes since they receive only encrypted data from either a sender or a data repository. Note that **Algorithm** 4 calls a function *DecryptNode* described in **Algorithm** 5, which was originally proposed by [7]. Here we include *DecryptNode* for completeness and to help the users without the knowledge of CP_ABE to understand CP_ABSC.

Algorithm 2 KeyGeneration (*MSK*, S)

Inputs: The master key *MSK* and a set of attributes S.

1: Select random numbers $r_{en}, r_{sn} \in \mathbb{Z}_p$
2: Compute the secret key component $D_{en} = g_2^{\frac{(\alpha + r_{en})}{\beta}}$ and the signing key $K_{sign} = g_2^{\frac{(\alpha + r_{sn})}{\beta}}$.
3: **for** each attribute $j \in S$ **do**
4: Select a random number $r_j \in \mathbb{Z}_p$
5: Compute the secret key components

$$D_j = g_2^{r_{en}} \cdot g_2^{(H_1(j) \cdot r_j)} \text{ and } D'_j = g_2^{r_j}$$

6: **end for**
7: The secret key SK is:

$$SK = (D_{en}, \forall j \in S : D_j, D'_j). \tag{5}$$

8: Compute the verification key: $K_{ver} = g_2^{r_{sn}}$
9: Send SK and K_{sign} to the owner of the attribute set S; publish K_{ver}.

4.3. **The correctness of CP_ABSC.** In this subsection, we show that CP_ABSC is indeed feasible and correct. We claim that **Algorithm** 4 can correctly decrypt

Algorithm 3 SignCryption*(PK, M, T, K_{sign})*

Inputs: The public parameter PK; plaintext message M; the tree T rooted at node R specifying the access control of message M; and the signing key K_s.

1: Choose a polynomial q_x and sets its degree $d_x = k_x - 1$ for each node x in the tree T.
2: Choose a random number $s \in \mathbb{Z}_p$ and sets $q_R(0) = s$;
3: Choose d_R random numbers from $\mathbb{Z}_p$ to completely define the polynomial q_R.
4: **for** any other node x in T **do**
5: Set $q_x(0) = q_{parent(x)}(index(x))$.
6: Select d_x random numbers from $\mathbb{Z}_p$ to completely define q_x.
7: **end for**
8: Let Y be the set of leaf nodes in T. The ciphertext CT is constructed based on the access tree T as follows:

$$\begin{aligned} CT &= (T, \tilde{C} = M \oplus t^s, C = h^s, \\ &\quad \forall y \in Y : C_y = g_1^{q_y(0)}, \\ &\quad\quad C'_y = g_1^{(H_1(att(y)) \cdot q_y(0))}) \end{aligned} \tag{6}$$

9: Choose a random $\zeta \in \mathbb{Z}_p$; compute $\delta = e(C, g_2)^{\zeta}$, $\pi = H_1(\delta | M)$, and $\psi = g_2^{\zeta} \cdot (K_{sign})^{\pi}$.
10: Output the message:

$$CT_{sign} = (T, \tilde{C}, C, \forall y \in Y : C_y, C'_y; W = g_1^s, \pi, \psi)$$

Algorithm 4 DeSignCryption (CT_{sign}, SK, S)

Inputs: The ciphertext $CT = (T, \tilde{C}, C, \forall y \in Y : C_y, C'_y)$, W, π, ψ; the private key SK for designcryption; and the set of possessed attributes S.

1: $A = DecryptNode(CT, SK, R)$
2: **if** $A \neq \perp$ **then**
3: $\tilde{A} = e(C, D_{en})/A$
4: **end if**
5: Compute

$$\delta' = \frac{e(C, \psi)}{(e(W, K_{ver}) \cdot \tilde{A})^{\pi}} \tag{7}$$

6: **if** $H_1(\delta' | M') = \pi$ **then**
7: return $M = M'$
8: **end if**
9: Return $\perp$

the ciphertext if the designcryptor satisfies the access policy, and can verify whether the received message has been forged or falsified, and whether the received message is indeed sent by the sender or not.

First, from the decryption procedure, we have

$$\begin{aligned} M' &= \tilde{C} \oplus \tilde{A} \\ &= \tilde{C} \oplus (\frac{e(C, D)}{A}) \\ &= \tilde{C} \oplus (\frac{e(h^s, g_2^{(\alpha + r_{en})/\beta})}{e(g_1, g_2)^{r_{en} s}}) \\ &= M \oplus e(g_1, g_2)^{\alpha s} \oplus (\frac{e(g_1^{\beta s}, g_2^{\alpha + r_{en}/\beta})}{e(g_1, g_2)^{r_{en} s}}) \end{aligned}$$

Algorithm 5 Function *DecryptNode* (CT, SK, x)

Inputs: A ciphertext $CT = (T, \tilde{C}, C, \forall y \in Y : C_y, C'_y)$; the secret key SK, which is associated with a set of attributes S, the node x from T.

if x is a leaf node of T **then**
  Let $i = att(x)$
  if $i \in S$ **then**

$$\text{Return } F_x = \frac{e(C_i, D_i)}{e(C'_i, D'_i)} = e(g_1, g_2)^{r_{en} q_x(0)} \tag{8}$$

  else Return $\perp$
  end if
else
  for Each child node z of x **do**
    $F_z = DecryptNode(CT, SK, z)$
  end for
end if
Let S_x be an arbitrary k_x-sized set of child nodes of x such that $F_z \neq \perp$ for $\forall z \in S_x$.
if S_x exists **then**
  for Each node $z \in S_x$ **do**
    $i_z = index(z)$
    $S'_z = \{index(z) \mid\mid z \in S_x\}$
    $\triangle_{i_z, S'_z}(y) = \prod_{j \in S'_z, j \neq i_z} \frac{y-j}{i_z - j}$
  end for
  Return

$$\begin{aligned} F_x &= \prod_{z \in S_x} F_z^{\triangle_{i_z, S'_z}(0)} \\ &= \prod_{z \in S_x} (e(g_1, g_2)^{r_{en} \cdot q_z(0)})^{\triangle_{i_z, S'_z}(0)} \\ &= \prod_{z \in S_x} e(g_1, g_2)^{r_{en} \cdot q_x(i_z) \cdot \triangle_{i_z, S'_z}(0)} \\ &= e(g_1, g_2)^{r_{en} \cdot q_x(0)} \end{aligned}$$

  .
else
  Return $F_x = \perp$
end if

$$\begin{aligned} &= M \oplus e(g_1, g_2)^{\alpha s} \oplus \left(\frac{e(g_1, g_2)^{\beta s \cdot (\alpha + r_{en})/\beta}}{e(g_1, g_2)^{r_{en} s}}\right) \\ &= M \oplus e(g_1, g_2)^{\alpha s} \oplus \left(\frac{e(g_1, g_2)^{(\alpha s + r_{en} s)}}{e(g_1, g_2)^{r_{en} s}}\right) \\ &= M \oplus e(g_1, g_2)^{\alpha s} \oplus e(g_1, g_2)^{\alpha s} \\ &= M. \end{aligned}$$

which indicates that **Algorithm** 4 can correctly decrypt the ciphertext if the designcryptor satisfies the access policy.

Second, the receiver verifies whether the message M' has been forged or falsified, and whether the received message is indeed sent by the claimed sender or not. The designcryptor computes δ' by:

$$\begin{aligned}
\delta' &= \frac{e(C,\psi)}{(e(W,K_{ver})\cdot\tilde{A})^{\pi}} \\
&= \frac{e(g_1^{\beta s}, g_2^{\zeta}\times g_2^{\frac{(\alpha+r_{sn})}{\beta}\pi})}{(e(g_1^{s}, g_2^{r_{sn}})\cdot e(g_1,g_2)^{\alpha s})^{\pi}} \\
&= e(g_1,g_2)^{\beta s(\zeta+\frac{(\alpha+r_{sn})}{\beta}\pi)-sr_{sn}\pi-\alpha s\pi} \\
&= e(g_1,g_2)^{\beta s\zeta+s(\alpha+r_{sn})\pi-sr_{sn}\pi-\alpha s\pi} \\
&= e(g_1,g_2)^{\beta s\zeta} = e(C,g_2)^{\zeta} \\
&= \delta.
\end{aligned}$$

If $H_1(\delta'|M') = \pi$, M' is valid, i.e., $M = M'$, and the message is indeed sent by the sender; otherwise, M' is invalid.

4.4. **Security analysis.** In this subsection, we analyze the security strength of the proposed scheme CP_ABSC by examining how it can counter a few major attacks.

4.4.1. *Collusion attack resistance.* In CP_ABSC, the set of attributes composes the identity. In order to allow different users to access the same cipertext, the scheme provides an access tree structure for each signcrypted data item, and requires only a subset of the attributes for designcryption. Since the secret key computation involves a unique random number for each attribute in the access policy, our scheme can defend against collusion attacks.

For example, assume that neither user U_1 nor user U_2 possesses a sufficient number of attributes to successfully designcrypt the ciphertext CT_{sign} alone but the combined attribute set has sufficient number of attributes for the designcryption. Then U_1 and U_2 may collude by combining their attributes in any way. However, they are not able to combine their secret keys (the SKs) to get a secret key for the combined set of attributes according to **Algorithm** 2 because the KGC generates different random values r_{en} for U_1 and U_2. Thus they could not designcrypt the message, indicating that the proposed scheme is secure against collusion attacks.

4.4.2. *Message authentication.* Assume that a user U wants to get a message M from the data repository. Before the data is stored in the data repository, the sender has signcrypted it with **Algorithm** 3. When U plans to obtain the data from the data repository, it needs its private key $SK = (D = g_2^{\frac{(\alpha+r_{en})}{\beta}}, \forall j \in S : D_j = g_2^{r_{en}}\cdot g_2^{(H_1(j)\cdot r_j)}, D'_j = g_2^{r_j})$, which are computed by **Algorithm** 2. Meanwhile, U obtains the sender's verification key from KGC. It designcrypts the ciphertext to get the message M' by **Algorithm** 4: if $H_1(\delta'|M') = \pi$ is established, the decrypted message M is valid; otherwise, it is discarded.

4.4.3. *Unforgeability.* An adversary who wishes to forge the signcryption of a legal user must possess the user's private key. However, the adversary cannot infer the private key K_{sign} and the root node of the access tree s because r_s and s are chosen randomly. On the other hand, the adversary cannot create a new, valid ciphertext from other user's ciphertexts. Even if the adversary changes the ciphertext of the message, the receiver can still verify that the ciphertext is illegal by **Algorithm** 4. If the adversary colludes with other users to forge the ciphertext, it cannot succeed according to the above security analysis on defending collusion attacks. Thus we claim that our proposed scheme is unforgeable under chosen message attacks.

4.4.4. *Confidentiality.* Decryption requires the knowledge of $e(g_1, g_2)^{\alpha s}$. The decryption procedure is the same as that of CP_ABE [7] as they use the same DecyptNode function (**Algorithm** 5), and thus CP_ABSC has the same security strength as that of the CP_ABE. The designcryption requires the knowledge of $\delta = e(C, g_2)^{\zeta}$. For a passive adversary, the available information is CT_{sign}. Only W in the signature reveals s, but it is difficult to get s from W since it is difficult to compute the discrete logarithm. Even if the adversary constructs the bilinear mapping e via C and the public parameter g_2 to obtain $e(C, g_2)$, it can not get the ζ, which is randomly chosen by the signcryptor. The adversary may want to get ζ from ψ, but it has to get K_{sign} first. Nevertheless, even though the K_{sign} is compromised, the adversary still can't get ζ from ψ due to the difficulty of computing the discrete logarithm. Given the discussion above and the fact that CP_ABE is proven secure under chosen-ciphertext attacks, our scheme is secure under chosen-ciphertext attacks too.

4.4.5. *Key revocation.* For attribute based encryption, key revocation is a very difficult issue. The core challenge lies in that a data source does not know its receiver's certificate before encrypting its data; thus it can not check whether or not the receiver has been revoked. Moreover, the fact that multiple receivers may match the same decryption policy in ABE makes key revocation even harder. The popular solution is to add a timestamp signaling when the attributes used in encryption should expire [35]. However, this method has the following drawback: since the data source and KGC may agree on an exact expiration time for an attribute required by the data but the attribute at the user side may have a much longer expiration time, receivers are forced to connect to the KGC and maintain a large amount of private keys as one key for each time period is needed if the data source is able to specify a policy designating a revocation time on a fine-grained scale. To overcome this problem, the following approach can be adopted: the KGC can distribute a single key with an expiration time $eTime$ to the users rather than a separate key for each time period before $eTime$. When a data source encrypts a message with an expiration time $Time$, its receiver with a key that expires in $eTime$ can decrypt only when $eTime \geq Time$ and the rest of the policy matches the user's attributes. By this way, different data may have different expiration times and there is no need to have close coordination between the data source and KGC.

4.5. **Performance analysis.** In this subsection, we present a quantitative performance study. Our main concern is efficiency in terms of computational cost.

TABLE 1. The Computational Cost of Different Functions and Operations between CP_ABE and our scheme

	CP_ABE [7]	CP_ABSC
Key Generation	$n\mathbb{G}_1 + (n+2)\mathbb{G}_2 + nH_{\mathbb{G}_2}$	$(2n+5)\mathbb{G}_2$
Encryption	$(k+1)\mathbb{G}_1 + k\mathbb{G}_2 + 1\mathbb{G}_3 + kH_{\mathbb{G}_2}$	$2((k+1)\mathbb{G}_1 + \mathbb{G}_2 + \mathbb{G}_3) + 2$ (pairings)
Decryption	$(2k'+1)$ (pairings)	$1\mathbb{G}_3 + (2k'+3)$ (pairings)

Notes: $\mathbb{G}_1$ in the table means an exponentiation operation in $\mathbb{G}_1$ group; $\mathbb{G}_2$ and $\mathbb{G}_3$ are defined similarly. $H_{\mathbb{G}_1}$ means hashing an attribute string or a message into an element in $\mathbb{G}_1$; $H_{\mathbb{G}_2}$ is defined similarly.

Our scheme CP_ABSC does not incur a high computational cost in KeyGeneration, SignCryption, and DeSignCryption compared to CP_ABE. Table 1 reports the amount of operations performed by CP_ABE and CP_ABSC. The notations are

explained as follows: n is the number of attributes a user holds, k is the number of leaf nodes in the access tree T, k' is the number of attributes a user possesses, $\mathbb{G}_1$ (respectively $\mathbb{G}_2$ and $\mathbb{G}_3$) denotes an exponent operation in $\mathbb{G}_1$ group (respectively $\mathbb{G}_2$ group and $\mathbb{G}_3$ group), $H_{\mathbb{G}_1}$ (respectively $H_{\mathbb{G}_2}$) means hashing an attribute or message into an element in $\mathbb{G}_1$ (respectively $\mathbb{G}_2$).

Starting with KeyGeneration. As described in **Algorithm** 2, there is $2n+5$ exponent operations in $\mathbb{G}_2$, which includes 5 exponent operations of $\{g_2^{r_{en}}, g_2^{\beta}, g_2^{r_{sn}}, D_{en}, K_{sign}\}$, and $2n$ exponent operations of $\{D_j, D'_j\}$. In CP_ABE [7], the total operations is $n\mathbb{G}_1 + (n+2)\mathbb{G}_2 + nH_{\mathbb{G}_2}$.

Moving next to the SignCryption procedure in **Algorithm** 3. There are $2k+2$ exponent operations in group $\mathbb{G}_1$ and 2 exponent operations in group $\mathbb{G}_2$. Additionally, there are 2 map operations and 2 pairing. The combined overhead is thus $(2k+2)\mathbb{G}_1 + 2\mathbb{G}_2 + 2\mathbb{G}_3 + 2$ (pairings). Similarly, in CP_ABE, the total operations is $(k+1)\mathbb{G}_1 + k\mathbb{G}_2 + 1\mathbb{G}_3 + kH_{\mathbb{G}_2}$.

The DeSignCryption in **Algorithm** 4 involves $(2k'+3)$ operations (pairings). In CP_ABE, there are $(2k'+1)$ operations (pairings).

We run the experiment with Ubuntu 12.04 running as a VM on a MACBook Air with one 1.8GHz core and 1GB memory. The implementation uses a Python library called Charm-crypto [2], which is a framework used to prototype advanced cryptosystems such as IBE and IBS (Identity-Based Signature). The core mathematical functions behind Charm are from the Stanford Pairing-Based Cryptography (PBC) library [27], which is an open source C library that performs mathematical operations underlying pairing-based cryptosystems.

We execute the implementation under both symmetric (SS512) and asymmetric groups (MNT159 and MNT159.S), both with 80 bits of security, to compare CP_ABE and CP_ABSC. In SS512, the map is $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are the same group. In MNT159, the map is $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, where $\mathbb{G}_2$ and $\mathbb{G}_3$ are extension groups of $\mathbb{G}_1$. The elements in $\mathbb{G}_2$ and $\mathbb{G}_3$ are longer than those in $\mathbb{G}_1$. The longer the element, the larger the computational cost in exponential operations. In MNT159.S, we swapped the $\mathbb{G}_1$ and $\mathbb{G}_2$ group so that most of the key generation operations are in $\mathbb{G}_1$ instead of $\mathbb{G}_2$.

Table 2 lists the run time of each operation/function in SS512 and MNT159. One can see that some operations are more efficient in SS512 than in MNT159 while others are the opposite. For example, the operations $H_{\mathbb{G}_1}$ and $\mathbb{G}_1$ have less run time in MNT159 than in SS512 but the operations of $\mathbb{G}_2$ and $H_{\mathbb{G}_2}$ have less runtime in SS512 than in MNT159.

We also compare the efficiency and computational cost between CP_ABSC and CP_ABE for KeyGeneration, SignCryption/Encryption, and DeSignCryption/Decryption, and report the results in Figures 4-6. Figure 4 demonstrates the run times of KeyGeneration. We observe that MNT159.S has the best performance since we swapped $\mathbb{G}_1$ and $\mathbb{G}_2$ and most of the operations are in $\mathbb{G}_1$ after the swap. Figure 5 reports the encryption run times. We notice that the run time in CP_ABE and that in our scheme CP_ABSC is almost linear with respect to the number of leaf nodes in the access policy. The polynomial operation at the leaf nodes does not significantly contribute to the run time. Comparing the run time between CP_ABE encryption and CP_ABSC signcryption, one can see that our scheme CP_ABSC costs less time than CP_ABE because we don't need to compute $H_{\mathbb{G}_2}$. Figure 6 illustrates the run times of decryption. Our scheme CP_ABSC is slightly higher than that of CP_ABE due to the fact that we add the signature verification process. However, because

the computational cost of CP_ABE is more expensive as the number of attributes increases, the cost of signature verification is relatively nonsignificant in practice.

Considering the three processes of KeyGeneration, SignCryption, and DeSignCryption, MNT159.S has considerably better performance than MNT159. We recommend executing the schemes in asymmetric groups and swapping $\mathbb{G}_1$ and $\mathbb{G}_2$ to gain a better performance.

In summary, the run time is predictable for key generation and encryption in our scheme and is correlated with the number of attributes. Comparing the run times of key generation, encryption, and decryption between CP_ABE and our scheme CP_ABSC, the run times of our scheme is a little higher than CP_ABE for some cases. However, considering that our scheme combines encryption and signature, CP_ABSC is feasible and more desirable than the encryption-only CP_ABE.

TABLE 2. The Computational Cost of Different Operations in Charm Library

Group	$\mathbb{G}_1$	$\mathbb{G}_2$	$\mathbb{G}_3$	(pairings)	$H_{\mathbb{G}_1}$	$H_{\mathbb{G}_2}$
SS512	3.73	3.70	0.48	3.92	8.34	8.39
MNT159	1.12	9.84	2.62	8.42	0.10	34.82

Notes: Time is in ms. The result in this table is the average of 1000 runs.

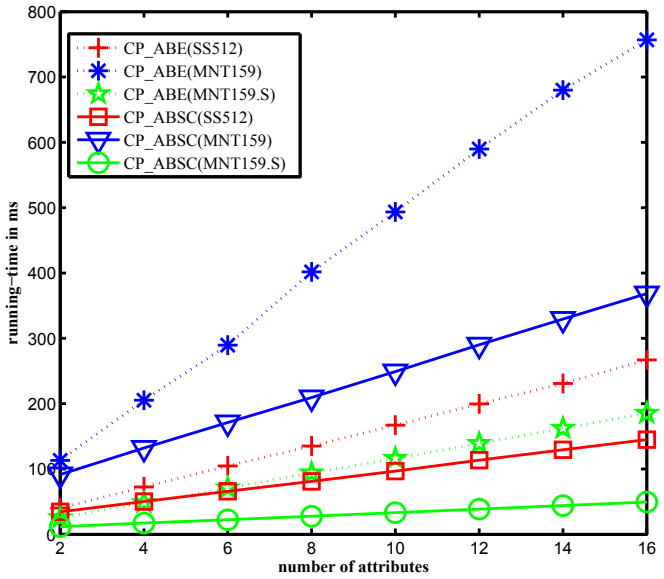

FIGURE 4. Key generation time.

5. **KP_ABE, CP_ABE, and CP_ABSC.** In Key-Policy ABE (KP_ABE) [15], which is viewed as an extension of the Fuzzy IBE (FIBE) [38], encryption is completely determined by the full set of descriptive attributes (no access policy is needed) possessed by the data source while the decryption key is computed by KGC from an access policy defined by the KGC. In order to decrypt a ciphertext,

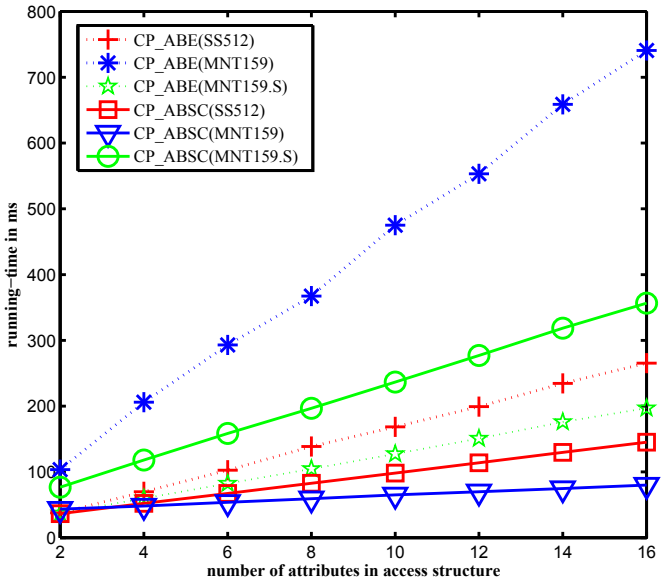

FIGURE 5. Encryption time.

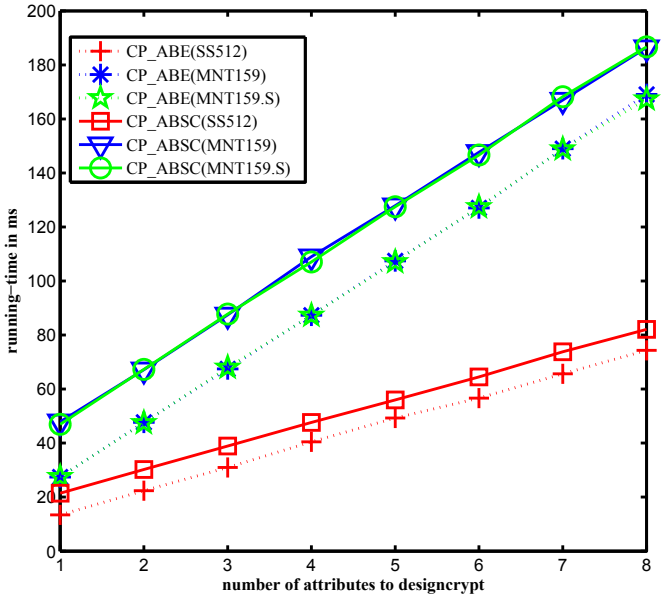

FIGURE 6. Decryption time.

a user must go to KGC to get a decryption key, which is computed from both the access policy and the attributes of the user In CP_ABE [7], encryption is completely determined by an access policy defined from the set of attributes possessed by the data source, and the ciphertext carries the access policy; the decryption key is computed by KGC and is associated with a user possessing a certain set of descriptive

TABLE 3. Comparison between CP_ABE and CP_ABSC

The scheme	System Initial.	KeyGeneration	Encryption	Decryption
CP_ABE [7]	symmetric groups	private key	encryption	decryption
CP_ABSC	asymmetric groups	(sign+verify) key	signcrypt.	decrypt.&verify.

attributes. In other words, KGC helps a user compute a decryption key based on the user's attributes, and the decryption key is not related to any access policy. A user can decrypt a ciphertext if and only if its attributes satisfy the access policy carried by the ciphertext, and the ciphertext can be correctly deciphered by multiple users with different attribute sets. In key generation, the users' private keys are associated with the access policy in KP_ABE; however, the users' private keys are associated only with the attributes in CP_ABE and CP_ABSC. In encryption, the ciphertext is associated with only the attributes in KP_ABE; however, in CP_ABE and CP_ABSC the ciphertext is associated with both the access policy and the attributes. In CP_ABE and CP_ABSC, a data source is able to intelligently decide who should or should not have access to its data as it is the decision maker of the access policy but the access policy is determined by KGC in KP_ABE.

In the following we compare CP_ABSC and CP_ABE [7] to identify their differences as CP_ABSC is more closely related to CP_ABE. The characteristics of CP_ABSC and CP_ABE are summarized in Table 3.

5.0.1. *System initialization.* The system initialization procedure is utilized to construct the groups and the corresponding group generators, and the bilinear mapping. The difference between CP_ABSC and CP_ABE in this step lies in that the former employs asymmetric groups while the latter utilizes symmetric groups.

5.0.2. *Key generation.* The KeyGeneration algorithm in our scheme CP_ABSC is different from the corresponding algorithm in [7] from the following two aspects: i) since we are designing a signcryption scheme, we need to compute a signing key (which is kept secret to the owner of the attributes), a verification key (which will be public), and a designcryption key (which is kept secret to the owner of the attributes for designcryption), while CP_ABE only needs one key for decryption; and ii) due to the fact that CP_ABSC utilizes asymmetric groups, its key generation is more computationally efficient than the one proposed in [7] according to our comparison study in Section 4.5.

5.0.3. *Encryption (Signcryption).* The SignCryption algorithm combines signature and encryption, while the one in [7] performs only encryption. The computational cost of our SignCryption algorithm is less than that of encryption+signature, and is also less than that of the encryption algorithm in [7], according to our analysis in Section 4.5, attributed to the adoption of asymmetric groups.

5.0.4. *Decryption (Designcryption).* The DeSignCryption in CP_ABSC includes decryption and verification, while the decrypt algorithm in [7] performs only decryption. The computational cost of DeSignCryption is only slightly higher than that of the decyption algorithm in [7], according to our analysis in Section 4.5.

Note that our scheme has a performance boost compared to Encryption+Signature (also see Section 7). In practice, it's more straightforward to do signcryption in just one operation.

6. **Applications of CP_ABSC in smart grids.** In this section, we illustrate how to use CP_ABSC to secure the communications of a smart grid system. Initially, KGC computes the public parameters PK according to **Algorithm** 1, and posts PK to all active entities (smart meters and service providers) in the system. Each entity also needs to register with KGC to get the corresponding keys computed from **Algorithm** 2. For example, a utility company needs a private key SK for designcryption based on its access attributes, a signing key K_{sign} to sign its commands, and a verification key K_{ver} for others to verify its signature.

6.1. **Send instructions to smart meters.** When a service provider wants to send instructions or commands to one or more smart meters, the service provider constructs an access structure T that describes the set of smart meters satisfying certain access policy defined by the AND/OR relationship of a set of attributes. It then signcrypts an instruction I with a timestamp ts. The timestamp can be the current time or the current time with an expiration time. Generally speaking, the timestamp can help the receivers decide whether or not instruction I is valid and resist replay attacks. The following procedure implements a scenario where a service provider broadcasts I to smart meters.

1. The service provider broadcasts the following signcrypted instruction to the smart meters according to **Algorithm** 3:

$$Service\ provider \rightarrow Smart\ meters : SignCryption(I||ts, T, K_{sign}).$$

2. When a smart meter receives the signcrypted instruction, it designcrypts and verifies the message according to **Algorithm** 4. If the verification is passed, the smart meter executes the instruction. Note that a smart meter can designcrypt an instruction if and only if its possessed attributes satisfy the access policy carried by the ciphertext.

Note that a feedback response could be sent back if the instruction is successfully received by a smart meter but this step is optional as many smart meters may receive the same instruction, as in push-based multicast communications.

6.2. **Retrieving data.** In order to protect the power usage data, a smart meter signcrypts the data of its household devices using **Algorithm** 3 based on the access policy specified by the data, and then sends the signcrypted data CT_{sign} to a data repository. When a service provider possessing an attribute set S wants to get the data for a particular household device, it contacts the data repository and gets the signcrypted data CT_{sign}. The following procedure illustrates how a service provider possessing a certain attribute set S (correspondingly, obtaining the private key SK and signing key K_{sign} from KGC) obtains sensitive data from the data repository.

1. A smart meter signcrypts its reading M with a timestamp ts, $M||ts$, based on **Algorithm** 3 and then sends CT_{sign} to the data repository. This step can be performed whenever a new data item is generated.

$$Smart\ meter \rightarrow Data\ repository : CT_{sign}.$$

2. When a service provider holding an attribute set S needs to access the smart meter data, it contacts the data repository to obtain the signcrypted data CT_{sign}:

$$Data\ repository \rightarrow Service\ provider : CT_{sign}.$$

3. Upon receiving the signcrypted data CT_{sign}, the service provider designcrypts CT_{sign} with its privacy key SK and verifies the message with K_{ver} of the data source according to **Algorithm** 4: it first recovers the plaintext M' based on its private key SK and then computes δ'; if $H_1(\delta'|M') = \pi$, which demonstrates the successful designcryption of the data, the service provider accepts M'; otherwise, the message is dropped.

7. An alternative: Adopting Attribute-Based Signature.

In this section, we briefly introduce how to adopt attribute-based signature (ABS) instead of the signature mechanism proposed for CP_ABSC to achieve anonymous communications. Particularly, we implement Maji's attribute-based signature scheme [28] and analyze its efficiency and computational cost.

7.1. Anonymity with ABS.

In smart grids, utility companies need their customers' energy consumption data for billing purpose; certain TPSPs need to collect electricity usage records to monitor device status and detect potential problems; others need energy consumption data to conduct market research. In these cases, the data sources should be informed of the usage of their data and be ensured that their data are anonymized before being used to protect privacy [30].

Attribute-Based Signature (ABS) provides a strong guaranty that the signature was produced by a single party whose attributes satisfy a possibly complicated claim. ABS does not attest to the identity of the individual who has endorsed a message. The signature reveals nothing about the signer's identity nor even the attributes of the signer beyond what is explicitly revealed by the claim being made. This successfully solves the problem of data anonymity. With such signatures, marketing companies can only know that the data does come from the desired group of customers with certain attributes while the customers' identities are fully protected.

We briefly present how to integrate Maji's ABS scheme with our basic idea of ciphertext-policy attribute-based encryption in this section. Maji's ABS [28] consists of four algorithms: TSetup (), AttrGen(), Sign(), and Verify(). TSetup() is utilized to set up the system (defined by the set of attributes) and to generate a key pair (public key and private key) for the system; AttrGen() is used to generate a signing key based on the attributes by KGC; Sign() is used to create a signature for a message; and Verify() is used to verify the signature. Moreover, in Maji's ABS, the access structure is transformed into an access matrix via a monotone boolean function, in which an attribute is mapped into a row of matrix. The detailed construction of this ABS can be found in [28]. Note that Maji's ABS scheme is secure under a generic model, which is not as strong as the standard model. However, we choose Maji's ABS here as an example for simplicity as other ABS schemes can be adopted similarly.

The integration of Maji's ABS is simple. In **Algorithm** 1 we need to add the setup of the ABS scheme; then we add the key generation component for the secret key of ABS in **Algorithm** 2. To achieve anonymity, we replace Line 9 of **Algorithm** 3 with the ABS Sign() procedure that signs a message by the attributes and employ the verification procedure of ABS in **Algorithm** 4. As a result, in a typical communication scenario, a sender encrypts the payload of the message with CP_ABE and signs the payload with ABS, and the receiver verifies the signature based on the verification algorithm of the Maji's scheme and then decrypts the data.

In short, the procedure outlined above proposes a combination of CP_ABE and ABS (denoted by CP_ABE+ABS in sequel) to achieve anonymity. The confidentiality of the data is protected by CP_ABE.

7.2. **Performance analysis on CP_ABE+ABS.** We have implemented Maji's ABS scheme with the Charm library. In this subsection, we analyze the performance of CP_ABE+ABS.

- In a typical scenario, a smart meter first registers in KGC and obtains its keys based on its attributes from the KGC. One exponential operation is needed in registration and one is needed to calculate a key for each attribute.
- We follow the verification procedure proposed in [28]. The computational cost of pairing in the verification of CP_ABE+ABS is $\ell + 4$ while it is $\ell \cdot t + 2$ if CP_ABE and ABS are implemented independently, where ℓ and t are respectively the numbers of rows and columns in the access matrix.
- The implementation also works under asymmetric groups such as the MNT curve.

Table 4 summarizes the number of operations for TSetup (), AttrGen(), Sign(), and Verify() of the implemented Maji's scheme, where ℓ_r is the number of required attributes to sign a message.We notice that the computational cost of Maji's ABS is higher than that of CP_ABSC; therefore CP_ABE+ABS obviously has much higher computational cost than CP_ABSC.

TABLE 4. Number of operations in the Maji's ABS scheme

TSetup ()	$1\mathbb{G}_1$ /user
AttrGen()	$1\mathbb{G}_1$ / attribute
Sign()	$2\mathbb{G}_1$+$3(\ell_r)\mathbb{G}_1$+$2(\ell - \ell_r)\mathbb{G}_1$ + $2(\ell \cdot t)\mathbb{G}_2$
Verify()	$1\mathbb{G}_1$+$2(\ell \cdot t + t)\mathbb{G}_2$+$(\ell + 4)$(pairings)

TABLE 5. Key generation per attribute of the Maji's ABS scheme

SS512	MNT159	MNT159.S	BN.S
3.67 ms	9.72 ms	1.13 ms	2.30 ms

According to Table 4, AttrGen() could be the bottleneck of a smart grid system, because a user needs to contact KGC for each attribute, and each secret key involves more than one attribute. The computational cost of TSetup() is far less than that of AttrGen(). Table 5 reports the key generation time per attribute of the Maji's ABS scheme. In this table, SS512 is a symmetric group with 80 bits of security, and MNT159 is an asymmetric group, also with 80 bits of security. Here MNT159.S means that we swap the $\mathbb{G}_1$ and $\mathbb{G}_2$ group. Note that we employ the BN.S curves [6] in our implementation since they have advantages in $\mathbb{G}_1$ and $\mathbb{G}_2$ Exponential operations.

The Maji's ABS scheme has a large number of Exponential operations in $\mathbb{G}_1$ and $\mathbb{G}_2$. The computational cost in Sign() and Verify() is higher than that of Dec() and Enc() in CP_ABE, as we have expected. Particularly, the computational cost increases with $\ell \cdot t$ and ℓ_r in Sign(). According to the access structure, the number of required attributes to sign a message is $\ell_r = \ell/2$. From Fig. 7, one can observe that both MNT159.S and BN.S have better performance in AttrGen(); and the BN.S

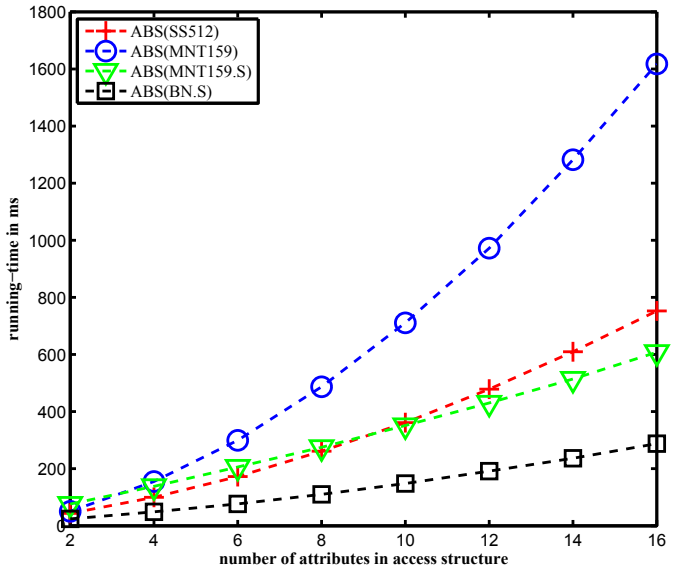

FIGURE 7. ABS signature running-time.

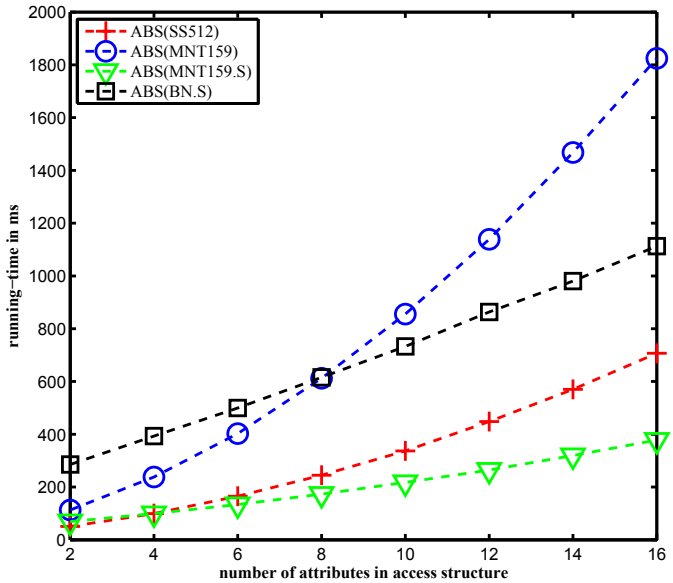

FIGURE 8. ABS verification running-time.

in Sign() has the lowest cost since the Exponential operations in $\mathbb{G}_1$ and $\mathbb{G}_2$ are less expensive than other schemes. From Figure 8, we notice that MNT159.S has better performance in verification. If one considers the performance of Sign() as the highest priority, the BN.S curve should be used and $\mathbb{G}_1$ should be swapped with $\mathbb{G}_2$. However, since the sender only needs to generate one signature for each message but verification might happen more than once, MNT159.S might be a better choice.

In terms of size, the signature of the BN.S curve is shorter when compared with that of the MNT curve in $\mathbb{G}_1$. The elements are twice as longer in $\mathbb{G}_2$ than those in

$\mathbb{G}_1$ in the BN.S curve while the elements are 3 times longer in $\mathbb{G}_2$ than those in $\mathbb{G}_1$ in the MNT curve. The computational resources on smart meters are limited, and thus the BN.S curves would be a better choice. However, if the efficiency of verify() has a higher priority, MNT159.S should be used. For example, in the scenario when data consumers need to collect anonymous data from a group of users that satisfy an access structure, the verification needs to be performed for each user and MNT159.S could reduce the computational cost.

7.3. **CP_ABSC v.s. CP_ABE+ABS.** Our CP_ABSC is a scheme that naturally combines CP_ABE and signature. According to the discussions mentioned above, CP_ABSC has less computational cost than CP_ABE+ABS. However, if we want to achieve anonymity in certain circumstances, we can use the combination of CP_ABE and ABS. The computational cost of ABS has been discussed in the previous subsection and one can see that it is much higher than the traditional signature due to the nature of the ABS scheme. For instance, if the number of attributes involved in the access structure is 10, and a signer needs to process 5 of them, it takes about 150 ms to sign when the BN.S curve with swapped $\mathbb{G}_1$ and $\mathbb{G}_2$ is used. In this case, the verification time is over 700 ms. If we add the run time of CP_ABE, it becomes even worse. Nevertheless, given the fact that ABS serves a special purpose of providing anonymity, the extra computational overhead can be justified.

8. **Conclusion and future work.** In this paper, we present a signcryption scheme CP_ABSC that naturally combines CP_ABE and signature, and analyze its properties. Our security analysis demonstrates that CP_ABSC can successfully ensure the data communication security in smart grids. The performance analysis further shows that CP_ABSC is efficient in terms of computational cost. We also consider the implementation of CP_ABE and the Maji's ABS scheme to obtain CP_ABE+ABS, and our performance analysis indicates that CP_ABE+ABS has a higher computational cost compared to CP_ABSC; nevertheless, CP_ABE+ABS can achieve anonymous communications.

Our future research lies in the following two directions: i) designing more efficient signcryption schemes with less computation and storage requirements, which could be better suitable for practical situations in smart grids; ii) considering dynamical schemes that can dynamically add attributes to meet the requirements of the smart grid as time goes.

REFERENCES

[1] Guidelines for smart grid cyber security (vol. 1 to 3), http://csrc.nist.gov/publications/PubsNISTIRs.html, 2010.

[2] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green and A. D. Rubin, Charm: A framework for rapidly prototyping cryptosystems, *Journal of Cryptographic Engineering*, **3** (2013), 111–128.

[3] A. Alrawais, A. Alhothaily, J. Yu, C. Hu and X. Cheng, Secureguard: A certificate validation system in public key infrastructure, *to appear in IEEE Transactions on Vehicular Technology*.

[4] Z. A. Baig and A.-R. Amoudi, An analysis of smart grid attacks and countermeasures, *Journal of Communications*, **8** (2013), 473–479.

[5] G. Baker and A. Berg, Supervisory control and data acquisition (scada) systems, *The Critical Infrastructure Protection Report*, **1** (2002), 5–6.

[6] P. S. Barreto and M. Naehrig, Pairing-friendly elliptic curves of prime order, in *Selected areas in cryptography*, Springer, 2006, 319–331.

[7] J. Bethencourt, A. Sahai and B. Waters, Ciphertext-policy attribute-based encryption, in *Security and Privacy, 2007. SP'07. IEEE Symposium on*, IEEE, 2007, 321–334.
[8] Z. Cai, Z.-Z. Chen and G. Lin, A 3.4713-approximation algorithm for the capacitated multicast tree routing problem, *Theoretical Computer Science*, **410** (2009), 5415–5424.
[9] Z. Cai, R. Goebel and G. Lin, Size-constrained tree partitioning: Approximating the multicast k-tree routing problem, *Theoretical Computer Science*, **412** (2011), 240–245.
[10] Z. Cai, Z. He, X. Guan and Y. Li, Collective data-sanitization for preventing sensitive information inference attacks in social networks, *IEEE Transactions on Dependable and Secure Computing*, **PP** (2016), 1–1.
[11] Z. Cai, G. Lin and G. Xue, Improved approximation algorithms for the capacitated multicast routing problem, Computing and combinatorics, *Lecture Notes in Comput. Sci.,* Springer, Berlin, **3595** (2005), 136–145.
[12] Z. Erkin, J. R. Troncoso-Pastoriza, R. L. Lagendijk and F. Perez-Gonzalez, Privacy-preserving data aggregation in smart metering systems: An overview, *Signal Processing Magazine, IEEE*, **30** (2013), 75–86.
[13] Z. Fadlullah, N. Kato, R. Lu, X. Shen and Y. Nozaki, Toward secure targeted broadcast in smart grid, *Communications Magazine, IEEE*, **50** (2012), 150–156.
[14] M. Gagné, S. Narayan and R. Safavi-Naini, Threshold attribute-based signcryption, in *Security and Cryptography for Networks*, Springer, 2010, 154–171.
[15] V. Goyal, O. Pandey, A. Sahai and B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in *Proceedings of the 13th ACM conference on Computer and communications security*, ACM, 2006, 89–98.
[16] C. Hu, X. Cheng, Z. Tian, J. Yu, K. Akkaya and L. Sun, An attribute-based signcryption scheme to secure attribute-defined multicast communications, in *International Conference on Security and Privacy in Communication Systems*, Springer, 2015, 418–437.
[17] C. Hu, W. Li, X. Cheng, J. Yu, S. Wang and R. Bie, A secure and verifiable access control scheme for big data storage in clouds, *IEEE Transactions on Big Data*.
[18] C. Hu, X. Liao and X. Cheng, Verifiable multi-secret sharing based on lrsr sequences, *Theoretical Computer Science*, **445** (2012), 52–62.
[19] M. Kgwadi and T. Kunz, Securing RDS broadcast messages for smart grid applications, *Proceeding: IWCMC '10 Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*, (2011), 1177–1181.
[20] Y. Kim, A. Perrig and G. Tsudik, Simple and fault-tolerant key agreement for dynamic collaborative groups, in *Proceedings of the 7th ACM conference on Computer and communications security*, ACM, 2000, 235–244.
[21] A. Lewko and B. Waters, Decentralizing attribute-based encryption, *Advances in Cryptology–EUROCRYPT 2011*, **6632** (2011), 568–588.
[22] D. Li, Z. Aung, S. Sampalli, J. Williams and A. Sanchez, Privacy preservation scheme for multicast communications in smart buildings of the smart grid, *Smart Grid and Renewable Energy*, **4** (2013), Article ID:33928,12 pages.
[23] Q. Li and G. Cao, Multicast authentication in the smart grid with one-time signature, *Smart Grid, IEEE Transactions on*, **2** (2011), 686–696.
[24] J. Liu, Y. Xiao, S. Li, W. Liang and C. Chen, Cyber security and privacy issues in smart grids, *Communications Surveys & Tutorials, IEEE*, **14** (2012), 981–997.
[25] Y. Liu, P. Ning and M. Reiter, False data injection attacks against state estimation in electric power grids, *Proceeding: CCS '09 Proceedings of the 16th ACM conference on Computer and communications security,* (2009), 21–32.
[26] R. Lu, X. Liang, X. Li, X. Lin, X. Shen et al., Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications, *IEEE Trans. on Parallel and Distributed Systems*.
[27] B. Lynn, *On the Implementation of Pairing-Based Cryptosystems*, PhD thesis, Stanford University, 2007.
[28] H. K. Maji, M. Prabhakaran and M. Rosulek, Attribute-based signatures, in *Topics in Cryptology–CT-RSA 2011*, Springer, 2011, 376–392.
[29] A. Metke and R. Ekl, Security technology for smart grid networks, *Smart Grid, IEEE Transactions on*, **1** (2010), 99–107.
[30] A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet and D. Irwin, Private memoirs of a smart meter, in *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*, ACM, 2010, 61–66.

[31] W. Neumann, Horse: An extension of an r-time signature scheme with fast signing and verification, in *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on*, IEEE, **1** (2004), 129–134.
[32] H. Nicanfar, P. Jokar and V. C. Leung, Smart grid authentication and key management for unicast and multicast communications, in *Innovative Smart Grid Technologies Asia (ISGT), 2011 IEEE PES*, IEEE, 2011, 1–8.
[33] A. Perrig, The biba one-time signature and broadcast authentication protocol, in *Proceedings of the 8th ACM conference on Computer and Communications Security*, ACM, 2001, 28–37.
[34] A. Perrig, R. Canetti, J. Tygar and D. Song, The tesla broadcast authentication protocol, *CryptoBytes*, **5** (2002), 2–13.
[35] M. Pirretti, P. Traynor, P. McDaniel and B. Waters, Secure attribute-based systems, in *Proceedings of the 13th ACM conference on Computer and communications security*, ACM, 2006, 99–112.
[36] L. Reyzin and N. Reyzin, Better than biba: Short one-time signatures with fast signing and verifying, in *Information Security and Privacy*, Springer, 2002, 144–153.
[37] S. Ruj, A. Nayak and I. Stojmenovic, A security architecture for data aggregation and access control in smart grids, arXiv:1111.2619.
[38] A. Sahai and B. Waters, Fuzzy identity-based encryption, *Advances in Cryptology–EUROCRYPT 2005*, **3494** (2005), 457–473.
[39] N. Saputro, K. Akkaya and S. Uludag, A survey of routing protocols for smart grid communications, *Computer Networks*, **56** (2012), 2742–2771.
[40] A. Shamir, How to share a secret, *Communications of the ACM*, **22** (1979), 612–613.
[41] A. Shamir, Identity-based cryptosystems and signature schemes, in *Advances in cryptology*, Springer, **196** (1985), 47–53.
[42] H. So, S. Kwok, E. Lam and K. Lui, Zero-configuration identity-based signcryption scheme for smart grid, in *IEEE International Conference on Smart Grid Communications*, IEEE, 2010, 321–326.
[43] C. Valli, A. Woodward, C. Carpene, P. Hannay and M. Brand, Eavesdropping on the smart grid, in *Australian Digital Forensics Conference*, 2012, 54–60.
[44] Q. Wang, H. Khurana, Y. Huang and K. Nahrstedt, Time valid one-time signature for time-critical multicast data authentication, in *INFOCOM 2009, IEEE*, IEEE, 2009, 1233–1241.
[45] W. WANG and Z. LU, Cyber security in the smart grid: Survey and challenges, *Computer networks*, **57** (2013), 1344–1371.
[46] C. K. Wong, M. Gouda and S. S. Lam, Secure group communications using key graphs, *SIGCOMM '98 Proceedings of the ACM SIGCOMM '98 conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, (1998), 68–79.
[47] K. Xing, C. Hu, J. Yu, X. Cheng and F. Zhang, Mutual privacy preserving k-means clustering in social participatory sensing, *IEEE Transactions on Industrial Informatics*, **13** (2017), 2066–2076.
[48] L. Zhang, Z. Cai and X. Wang, Fakemask: a novel privacy preserving approach for smartphones, *IEEE Transactions on Network and Service Management*, **13** (2016), 335–348.
[49] S. Zhongwei, H. Sitian, M. Yaning and S. Fengjie, Security mechanism for smart distribution grid using ethernet passive optical network, in *2010 2nd International Conference on Advanced Computer Control*, **3** (2010), 246–250.
[50] Z. Zhou and D. Huang, On efficient ciphertext-policy attribute based encryption and broadcast encryption, in *Proceedings of the 17th ACM conference on Computer and communications security*, ACM, 2010, 753–755.
[51] C. Zimmer and F. Mueller, Fault tolerant network routing through software overlays for intelligent power grids, in *Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on*, IEEE, 2010, 542–549.

Received September 2017; revised November 2017.

E-mail address: chu@cqu.edu.cn
E-mail address: jiguoyu@sina.com
E-mail address: cheng@gwu.edu
E-mail address: ztian1@gmu.edu
E-mail address: kakkaya@fiu.edu
E-mail address: sunlimin@iie.ac.cn