Secure Data Retrieval for Decentralized Disruption-Tolerant Military Networks

Junbeom Hur and Kyungtae Kang, *Member, IEEE, ACM*

***Abstract*—Mobile nodes in military environments such as a battlefield or a hostile region are likely to suffer from intermittent network connectivity and frequent partitions. Disruption-tolerant network (DTN) technologies are becoming successful solutions that allow wireless devices carried by soldiers to communicate with each other and access the confidential information or command reliably by exploiting external storage nodes. Some of the most challenging issues in this scenario are the enforcement of authorization policies and the policies update for secure data retrieval. Ciphertext-policy attribute-based encryption (CP-ABE) is a promising cryptographic solution to the access control issues. However, the problem of applying CP-ABE in decentralized DTNs introduces several security and privacy challenges with regard to the attribute revocation, key escrow, and coordination of attributes issued from different authorities. In this paper, we propose a secure data retrieval scheme using CP-ABE for decentralized DTNs where multiple key authorities manage their attributes independently. We demonstrate how to apply the proposed mechanism to securely and efficiently manage the confidential data distributed in the disruption-tolerant military network.**

***Index Terms*—Access control, attribute-based encryption (ABE), disruption-tolerant network (DTN), multiauthority, secure data retrieval.**

I. Introduction

In many military network scenarios, connections of wireless devices carried by soldiers may be temporarily disconnected by jamming, environmental factors, and mobility, especially when they operate in hostile environments. Disruption-tolerant network (DTN) technologies are becoming successful solutions that allow nodes to communicate with each other in these extreme networking environments [1]–[3]. Typically, when there is no end-to-end connection between a source and a destination pair, the messages from the source node may need to wait in the intermediate nodes for a substantial amount of time until the connection would be eventually established.

Roy [4] and Chuah [5] introduced storage nodes in DTNs where data is stored or replicated such that only authorized mobile nodes can access the necessary information quickly and efficiently. Many military applications require increased protection of confidential data including access control methods that are cryptographically enforced [6], [7]. In many cases, it is desirable to provide differentiated access services such that data access policies are defined over user attributes or roles, which are managed by the key authorities. For example, in a disruption-tolerant military network, a commander may store a confidential information at a storage node, which should be accessed by members of "Battalion 1" who are participating in "Region 2." In this case, it is a reasonable assumption that multiple key authorities are likely to manage their own dynamic attributes for soldiers in their deployed regions or echelons, which could be frequently changed (e.g., the attribute representing current location of moving soldiers) [4], [8], [9]. We refer to this DTN architecture where multiple authorities issue and manage their own attribute keys independently as a decentralized DTN [10].

The concept of attribute-based encryption (ABE) [11]–[14] is a promising approach that fulfills the requirements for secure data retrieval in DTNs. ABE features a mechanism that enables an access control over encrypted data using access policies and ascribed attributes among private keys and ciphertexts. Especially, ciphertext-policy ABE (CP-ABE) provides a scalable way of encrypting data such that the encryptor defines the attribute set that the decryptor needs to possess in order to decrypt the ciphertext [13]. Thus, different users are allowed to decrypt different pieces of data per the security policy.

However, the problem of applying the ABE to DTNs introduces several security and privacy challenges. Since some users may change their associated attributes at some point (for example, moving their region), or some private keys might be compromised, key revocation (or update) for each attribute is necessary in order to make systems secure. However, this issue is even more difficult, especially in ABE systems, since each attribute is conceivably shared by multiple users (henceforth, we refer to such a collection of users as an attribute group). This implies that revocation of any attribute or any single user in an attribute group would affect the other users in the group. For example, if a user joins or leaves an attribute group, the associated attribute key should be changed and redistributed to all the other members in the same group for backward or forward secrecy. It may result in bottleneck during rekeying procedure, or security degradation due to the windows of vulnerability if the previous attribute key is not updated immediately.

Another challenge is the key escrow problem. In CP-ABE, the key authority generates private keys of users by applying the authority's master secret keys to users' associated set of attributes. Thus, the key authority can decrypt every ciphertext

Manuscript received September 15, 2011; revised February 20, 2012, June 18, 2012; accepted July 23, 2012; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Kasera. Date of publication August 15, 2012; date of current version February 12, 2014. This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012R1A1A1001835). *(Corresponding author: K. Kang)*

J. Hur is with the School of Computer Science and Engineering, Chung-Ang University, Seoul 156-756, Korea (e-mail: jbhur@cau.ac.kr).

K. Kang is with the Department of Computer Science and Engineering, Hanyang University, Ansan 426-791, Korea (e-mail: ktkang@hanyang.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2012.2210729

addressed to specific users by generating their attribute keys. If the key authority is compromised by adversaries when deployed in the hostile environments, this could be a potential threat to the data confidentiality or privacy especially when the data is highly sensitive. The key escrow is an inherent problem even in the multiple-authority systems as long as each key authority has the whole privilege to generate their own attribute keys with their own master secrets. Since such a key generation mechanism based on the single master secret is the basic method for most of the asymmetric encryption systems such as the attribute-based or identity-based encryption protocols, removing escrow in single or multiple-authority CP-ABE is a pivotal open problem.

The last challenge is the coordination of attributes issued from different authorities. When multiple authorities manage and issue attribute keys to users independently with their own master secrets, it is very hard to define fine-grained access policies over attributes issued from different authorities. For example, suppose that attributes "role 1" and "region 1" are managed by the authority A, and "role 2" and "region 2" are managed by the authority B. Then, it is impossible to generate an access policy (("role 1" OR "role 2") AND ("region 1" or "region 2")) in the previous schemes because the OR logic between attributes issued from different authorities cannot be implemented. This is due to the fact that the different authorities generate their own attribute keys using their own independent and individual master secret keys. Therefore, general access policies, such as "n-out-of-m" logic, cannot be expressed in the previous schemes, which is a very practical and commonly required access policy logic.

A. Related Work

ABE comes in two flavors called key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE). In KP-ABE, the encryptor only gets to label a ciphertext with a set of attributes. The key authority chooses a policy for each user that determines which ciphertexts he can decrypt and issues the key to each user by embedding the policy into the user's key. However, the roles of the ciphertexts and keys are reversed in CP-ABE. In CP-ABE, the ciphertext is encrypted with an access policy chosen by an encryptor, but a key is simply created with respect to an attributes set. CP-ABE is more appropriate to DTNs than KP-ABE because it enables encryptors such as a commander to choose an access policy on attributes and to encrypt confidential data under the access structure via encrypting with the corresponding public keys or attributes [4], [7], [15].

1) Attribute Revocation: Bethencourt *et al.* [13] and Boldyreva *et al.* [16] first suggested key revocation mechanisms in CP-ABE and KP-ABE, respectively. Their solutions are to append to each attribute an expiration date (or time) and distribute a new set of keys to valid users after the expiration. The periodic attribute revocable ABE schemes [8], [13], [16], [17] have two main problems.

The first problem is the security degradation in terms of the backward and forward secrecy [18]. It is a considerable scenario that users such as soldiers may change their attributes frequently, e.g., position or location move when considering these as attributes [4], [9]. Then, a user who newly holds the attribute might be able to access the previous data encrypted before he obtains the attribute until the data is reencrypted with the newly updated attribute keys by periodic rekeying (backward secrecy). For example, assume that at time t_i, a ciphertext C is encrypted with a policy that can be decrypted with a set of attributes a_i (embedded in the users keys) for users with a_i. After time t_i, say t_j, a user newly holds the attribute set a_i. Even if the new user should be disallowed to decrypt the ciphertext C for the time instance t_i, he can still decrypt the previous ciphertext C until it is reencrypted with the newly updated attribute keys. On the other hand, a revoked user would still be able to access the encrypted data even if he does not hold the attribute any more until the next expiration time (forward secrecy). For example, when a user is disqualified with the attribute a_i at time t_j, he can still decrypt the ciphertext C of the previous time instance t_i unless the key of the user is expired and the ciphertext is reencrypted with the newly updated key that the user cannot obtain. We call this uncontrolled period of time windows of vulnerability.

The other is the scalability problem. The key authority periodically announces a key update material by unicast at each time-slot so that all of the nonrevoked users can update their keys. This results in the "1-affects-n" problem, which means that the update of a single attribute affects the whole nonrevoked users who share the attribute [19]. This could be a bottleneck for both the key authority and all nonrevoked users.

The immediate key revocation can be done by revoking users using ABE that supports negative clauses [4], [14]. To do so, one just adds conjunctively the AND of negation of revoked user identities (where each is considered as an attribute here). However, this solution still somewhat lacks efficiency performance. This scheme will pose overhead $O(R)$ group elements[1] additively to the size of the ciphertext and $O(\log M)$ multiplicatively to the size of private key over the original CP-ABE scheme of Bethencourt *et al.* [13], where M is the maximum size of revoked attributes set R. Golle *et al.* [20] also proposed a user revocable KP-ABE scheme, but their scheme only works when the number of attributes associated with a ciphertext is exactly half of the universe size.

2) Key Escrow: Most of the existing ABE schemes are constructed on the architecture where a single trusted authority has the power to generate the whole private keys of users with its master secret information [11], [13], [14], [21]–[23]. Thus, the key escrow problem is inherent such that the key authority can decrypt every ciphertext addressed to users in the system by generating their secret keys at any time.

Chase *et al.* [24] presented a distributed KP-ABE scheme that solves the key escrow problem in a multiauthority system. In this approach, all (disjoint) attribute authorities are participating in the key generation protocol in a distributed way such that they cannot pool their data and link multiple attribute sets belonging to the same user. One disadvantage of this fully distributed approach is the performance degradation. Since there is no centralized authority with master secret information, all attribute authorities should communicate with each other in the system to generate a user's secret key. This results in $O(N^2)$ communication overhead on the system setup and the rekeying phases

[1]The group elements mean those in the pairing operation group, not the user group. Since the computation in ABE schemes is done in the pairing operation group $\mathbb{G}_0$, the $O(R)$ group elements in the manuscript means $O(R)$ group elements in the pairing group $\mathbb{G}_0$.

and requires each user to store $O(N^2)$ additional auxiliary key components besides the attributes keys, where N is the number of authorities in the system.

3) Decentralized ABE: Huang *et al.* [9] and Roy *et al.* [4] proposed decentralized CP-ABE schemes in the multiauthority network environment. They achieved a combined access policy over the attributes issued from different authorities by simply encrypting data multiple times. The main disadvantages of this approach are efficiency and expressiveness of access policy. For example, when a commander encrypts a secret mission to soldiers under the policy ("Battalion 1" AND ("Region 2" OR 'Region 3")), it cannot be expressed when each "Region" attribute is managed by different authorities, since simply multiencrypting approaches can by no means express any general "n-out-of-m" logics (e.g., OR, that is 1-out-of-m). For example, let $A_1, \ldots, A_N$ be the key authorities, and $a_1, \ldots, a_N$ be attributes sets they independently manage, respectively. Then, the only access policy expressed with $a_1, \ldots, a_N$ is $(a_1$ AND $\ldots$ AND $a_N)$, which can be achieved by encrypting a message with a_1 by A_1, and then encrypting the resulting ciphertext c_1 with a_2 by A_2 (where c_1 is the ciphertext encrypted under a_1), and then encrypting resulting ciphertext c_2 with a_3 by A_3, and so on, until this multiencryption generates the final ciphertext c_N. Thus, the access logic should be only AND, and they require N iterative encryption operations where N is the number of attribute authorities. Therefore, they are somewhat restricted in terms of expressiveness of the access policy and require $O(N)$ computation and storage costs. Chase [25] and Lewko *et al.* [10] proposed multiauthority KP-ABE and CP-ABE schemes, respectively. However, their schemes also suffer from the key escrow problem like the prior decentralized schemes.

B. Contribution

In this paper, we propose an attribute-based secure data retrieval scheme using CP-ABE for decentralized DTNs. The proposed scheme features the following achievements. First, immediate attribute revocation enhances backward/forward secrecy of confidential data by reducing the windows of vulnerability. Second, encryptors can define a fine-grained access policy using any monotone access structure under attributes issued from any chosen set of authorities. Third, the key escrow problem is resolved by an escrow-free key issuing protocol that exploits the characteristic of the decentralized DTN architecture. The key issuing protocol generates and issues user secret keys by performing a secure two-party computation (2PC) protocol among the key authorities with their own master secrets. The 2PC protocol deters the key authorities from obtaining any master secret information of each other such that none of them could generate the whole set of user keys alone. Thus, users are not required to fully trust the authorities in order to protect their data to be shared. The data confidentiality and privacy can be cryptographically enforced against any curious key authorities or data storage nodes in the proposed scheme.

II. Network Architecture

In this section, we describe the DTN architecture and define the security model.

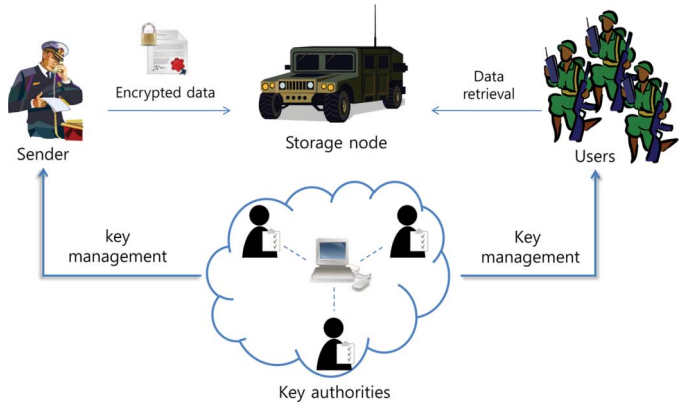


Fig. 1. Architecture of secure data retrieval in a disruption-tolerant military network.

A. System Description and Assumptions

Fig. 1 shows the architecture of the DTN. As shown in Fig. 1, the architecture consists of the following system entities.

1) Key Authorities: They are key generation centers that generate public/secret parameters for CP-ABE. The key authorities consist of a central authority and multiple local authorities. We assume that there are secure and reliable communication channels between a central authority and each local authority during the initial key setup and generation phase. Each local authority manages different attributes and issues corresponding attribute keys to users. They grant differential access rights to individual users based on the users' attributes. The key authorities are assumed to be honest-but-curious. That is, they will honestly execute the assigned tasks in the system, however they would like to learn information of encrypted contents as much as possible.
2) Storage node: This is an entity that stores data from senders and provide corresponding access to users. It may be mobile or static [4], [5]. Similar to the previous schemes, we also assume the storage node to be semitrusted, that is honest-but-curious.
3) Sender: This is an entity who owns confidential messages or data (e.g., a commander) and wishes to store them into the external data storage node for ease of sharing or for reliable delivery to users in the extreme networking environments. A sender is responsible for defining (attribute-based) access policy and enforcing it on its own data by encrypting the data under the policy before storing it to the storage node.
4) User: This is a mobile node who wants to access the data stored at the storage node (e.g., a soldier). If a user possesses a set of attributes satisfying the access policy of the encrypted data defined by the sender, and is not revoked in any of the attributes, then he will be able to decrypt the ciphertext and obtain the data.

Since the key authorities are semi-trusted, they should be deterred from accessing plaintext of the data in the storage node; meanwhile, they should be still able to issue secret keys to users. In order to realize this somewhat contradictory requirement, the central authority and the local authorities engage in the arithmetic 2PC protocol with master secret keys of their own and

issue independent key components to users during the key issuing phase. The 2PC protocol prevents them from knowing each other's master secrets so that none of them can generate the whole set of secret keys of users individually. Thus, we take an assumption that the central authority does not collude with the local authorities (otherwise, they can guess the secret keys of every user by sharing their master secrets).

B. Threat Model and Security Requirements

1) Data confidentiality: Unauthorized users who do not have enough credentials satisfying the access policy should be deterred from accessing the plain data in the storage node. In addition, unauthorized access from the storage node or key authorities should be also prevented.
2) Collusion-resistance: If multiple users collude, they may be able to decrypt a ciphertext by combining their attributes even if each of the users cannot decrypt the ciphertext alone [11]–[13]. For example, suppose there exist a user with attributes {"Battalion 1", "Region 1"} and another user with attributes {"Battalion 2", "Region 2"}. They may succeed in decrypting a ciphertext encrypted under the access policy of ("Battalion 1" AND "Region 2"), even if each of them cannot decrypt it individually. We do not want these colluders to be able to decrypt the secret information by combining their attributes. We also consider collusion attack among curious local authorities to derive users' keys.
3) Backward and forward Secrecy: In the context of ABE, backward secrecy means that any user who comes to hold an attribute (that satisfies the access policy) should be prevented from accessing the plaintext of the previous data exchanged before he holds the attribute. On the other hand, forward secrecy means that any user who drops an attribute should be prevented from accessing the plaintext of the subsequent data exchanged after he drops the attribute, unless the other valid attributes that he is holding satisfy the access policy.

III. Preliminaries and Definition

A. Cryptographic Background

We first provide a formal definition for access structure recapitulating the definitions in [12] and [13]. Then, we will briefly review the necessary facts about the bilinear map and its security assumption.

1) Access Structure: Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,P_2,\ldots,P_n\}}$ is monotone if $\forall B, C$: If $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of nonempty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,P_2,\ldots,P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

In the proposed scheme, the role of the parties is taken by the attributes. Thus, the access structure $\mathbb{A}$ will contain the authorized sets of attributes. From now on, by an access structure, we mean a monotone access structure.

2) Bilinear Pairings: Let $\mathbb{G}_0$ and $\mathbb{G}_1$ be a multiplicative cyclic group of prime order p. Let g be a generator of $\mathbb{G}_0$. A map $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ is said to be *bilinear* if $e(P^a, Q^b) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_0$ and all $a, b \in \mathbb{Z}_p^*$, and *nondegenerate* if $e(g, g) \neq 1$ for the generator g of $\mathbb{G}_0$.

We say that $\mathbb{G}_0$ is a bilinear group if the group operation in $\mathbb{G}_0$ can be computed efficiently and there exists $\mathbb{G}_1$ for which the bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ is efficiently computable.

3) Bilinear Diffie–Hellman Assumption: Using the above notations, the Bilinear Diffie–Hellman (BDH) problem is to compute $e(g, g)^{abc} \in \mathbb{G}_1$ given a generator g of $\mathbb{G}_0$ and elements g^a, g^b, g^c for $a, b, c \in \mathbb{Z}_p^*$. An equivalent formulation of the BDH problem is to compute $e(A, B)^c$ given a generator g of $\mathbb{G}_0$, and elements A, B and g^c in $\mathbb{G}_0$.

An algorithm $\mathcal{A}$ has advantage $\epsilon(\kappa)$ in solving the BDH problem for a bilinear map group $\langle p, \mathbb{G}_0, \mathbb{G}_1, e\rangle$, where κ is the security parameter (the bit length of p), if $\Pr[\mathcal{A}(p, \mathbb{G}_0, \mathbb{G}_1, A, B, g^c) = e(A, B)^c] \geq \epsilon(\kappa)$. If for every polynomial-time algorithm (in the security parameter κ) to solve the BDH problem on $\langle p, \mathbb{G}_0, \mathbb{G}_1, e\rangle$, the advantage $\epsilon(\kappa)$ is a negligible function, then $\langle p, \mathbb{G}_0, \mathbb{G}_1, e\rangle$ is said to satisfy the BDH assumption.

B. Definitions

$x \in_R S$ denotes the operation of picking an element x at random and uniformly from a finite set S. For a probabilistic algorithm $\mathcal{B}, x \xleftarrow{\$} \mathcal{B}$ assigns the output of $\mathcal{B}$ to the variable x. 1^λ denotes a string of λ ones, if $\lambda \in \mathbb{N}$. A function $\epsilon : \mathbb{N} \to \mathbb{R}$ is negligible ($\mathsf{negl}(k)$) if for every constant $c \geq 0$ there exists k_c such that $\epsilon(k) < k^{-c}$ for all $k > k_c$.

Let $\mathcal{U} = \{u_1, \ldots, u_n\}$ be the universe of users. Let CA be the central authority, and $\mathcal{A} = \{A_1, \ldots, A_m\}$ be the universe of local authorities. Let $\mathcal{L} = \{\lambda_1, \ldots, \lambda_p\}$ be the universe of descriptive attributes in the system. Let $A_i(\mathcal{L})$ be the set of attributes managed by A_i (we assume each local authority manages a disjoint set of attributes such that $A_i(\mathcal{L}) \cap A_j(\mathcal{L}) = \phi$ for $i \neq j$). Let $G_i \subset \mathcal{U}$ be a set of users that hold the attribute λ_i, which is referred to as an attribute group.

IV. Proposed Scheme

In this section, we provide a multiauthority CP-ABE scheme for secure data retrieval in decentralized DTNs. Each local authority issues partial personalized and attribute key components to a user by performing secure 2PC protocol with the central authority. Each attribute key of a user can be updated individually and immediately. Thus, the scalability and security can be enhanced in the proposed scheme.

Since the first CP-ABE scheme proposed by Bethencourt *et al.* [13], dozens of CP-ABE schemes have been proposed [7], [21]–[23]. The subsequent CP-ABE schemes are mostly motivated by more rigorous security proof in the standard model. However, most of the schemes failed to achieve the expressiveness of the Bethencourt *et al.*'s scheme, which described an efficient system that was expressive in that it allowed an encryptor to express an access predicate in terms of any monotonic formula over attributes. Therefore, in this section, we develop a variation of the CP-ABE algorithm partially based on (but not limited to) Bethencourt *et al.*'s construction in order to enhance the expressiveness of the access control policy instead of building a new CP-ABE scheme from scratch.

A. Access Tree

1) Description: Let $\mathcal{T}$ be a tree representing an access structure. Each nonleaf node of the tree represents a threshold gate. If num_x is the number of children of a node x and k_x is its threshold value, then $0 \leq k_x \leq \mathrm{num}_x$. Each leaf node x of the tree is described by an attribute and a threshold value $k_x = 1$. λ_x denotes the attribute associated with the leaf node x in the tree. $p(x)$ represents the parent of the node x in the tree. The children of every node are numbered from 1 to num. The function $\mathrm{index}(x)$ returns such a number associated with the node x. The index values are uniquely assigned to nodes in the access structure for a given key in an arbitrary manner.

2) Satisfying an Access Tree: Let $\mathcal{T}_x$ be the subtree of $\mathcal{T}$ rooted at the node x. If a set of attributes γ satisfies the access tree $\mathcal{T}_x$, we denote it as $\mathcal{T}_x(\gamma) = 1$. We compute $\mathcal{T}_x(\gamma)$ recursively as follows. If x is a nonleaf node, evaluate $\mathcal{T}_{x'}(\gamma)$ for all children x' of node x. $\mathcal{T}_x(\gamma)$ returns 1 iff at least k_x children return 1. If x is a leaf node, then $\mathcal{T}_x(\gamma)$ returns 1 iff $\lambda_x \in \gamma$.

B. Scheme Construction

Let $\mathbb{G}_0$ be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}_0$. Let $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ denote the bilinear map. A security parameter, κ, will determine the size of the groups. We will also make use of Lagrange coefficients $\Delta_{i,\Lambda}$ for any $i \in \mathbb{Z}_p^*$ and a set, Λ, of elements in $\mathbb{Z}_p^*$: define $\Delta_{i,\Lambda}(x) = \prod_{j \in \Lambda, j \neq i} \frac{x-j}{i-j}$. We will additionally employ a hash function $H : \{0,1\}^* \rightarrow \mathbb{G}_0$ to associate each attribute with a random group element in $\mathbb{G}_0$, which we will model as a random oracle.

1) System Setup: At the initial system setup phase, the trusted initializer[2] chooses a bilinear group $\mathbb{G}_0$ of prime order p with generator g according to the security parameter. It also chooses hash functions $H : \{0,1\}^* \rightarrow \mathbb{G}_0$ from a family of universal one-way hash functions. The public parameter *param* is given by $(\mathbb{G}_0, g, H)$. For brevity, the public parameter *param* is omitted below.

Central Key Authority: CA chooses a random exponent $\beta \in_R \mathbb{Z}_p^*$. It sets $h = g^\beta$. The master public/private key pair is given by $(\mathrm{PK}_{\mathrm{CA}} = h, \mathrm{MK}_{\mathrm{CA}} = \beta)$.

Local Key Authorities: Each A_i chooses a random exponent $\alpha_i \in_R \mathbb{Z}_p^*$. The master public/private key pair is given by $(\mathrm{PK}_{A_i} = e(g,g)^{\alpha_i}, \mathrm{MK}_{A_i} = \alpha_i)$.

2) Key Generation: In CP-ABE, user secret key components consist of a single personalized key and multiple attribute keys. The personalized key is uniquely determined for each user to prevent collusion attack among users with different attributes. The proposed key generation protocol is composed of the personal key generation followed by the attribute key generation protocols. It exploits arithmetic secure 2PC protocol to eliminate the key escrow problem such that none of the authorities can determine the whole key components of users individually.

Personal Key Generation: The central authority and each local authority are involved in the following protocol. For brevity, the knowledge of proofs are omitted below.

1) When CA authenticates a user u_t, it selects random exponents $\gamma_1, \ldots, \gamma_m \in_R \mathbb{Z}_p^*$ for every local authority $A_1, \ldots, A_m \in \mathcal{A}$; and sets $r_t = \sum_{i=1}^m \gamma_i$. This r_t value is a personalized and unique secret to the user, which should be consistent for any further attribute additions to the user. Then, CA and each A_i engage in a secure 2PC protocol, where CA's private input is (γ_i, β), and A_i's private input is α_i. The secure 2PC protocol returns a private output $x = (\alpha_i + \gamma_i)\beta$ to A_i. This can be done via a general secure 2PC protocol for a simple arithmetic computation [24], [26], [27]. Alternatively, we can do this more efficiently using the construction in [28].
2) A_i randomly picks $\tau \in_R \mathbb{Z}_p^*$. Then, it computes $T = g^{\frac{x}{\tau}} = g^{\frac{(\alpha_i+\gamma_i)\beta}{\tau}}$ and sends it to CA.
3) CA then computes $B = T^{1/\beta^2} = g^{\frac{\alpha_i+\gamma_i}{\tau\beta}}$ and sends it to A_i.
4) A_i outputs a personalized key component $D_i = B^\tau = g^{\frac{(\alpha_i+\gamma_i)}{\beta}}$ and sends it to the user u_t securely.

Then, the user u_t computes its personal key component $D = \prod_{i=1}^m D_i = g^{\frac{(\alpha_1+\cdots+\alpha_m)+r_t}{\beta}}$.

Theorem 1: The above key generation protocol is a secure 2PC protocol for computing $g^{(\alpha_i+\gamma_i)/\beta}$ by A_i, assuming that the underlying arithmetic 2PC and zero knowledge proofs are secure.

Proof: Proof can be found in the Appendix.

Attribute Key Generation: After setting up the personalized key component, each A_i generates attribute keys for a user u_t with a public parameter received from CA as follows.

1) CA first selects a random r', and sends $g^{r_t - r'}$ and $g^{r'}$ to A_i and u_t, respectively.
2) A_i takes a set of attributes $\Lambda_i \subseteq A_i(\mathcal{L})$ as inputs and outputs a set of attribute keys for the user that identifies with that set Λ_i. It chooses random $r_j \in \mathbb{Z}_p^*$ for each attribute $\lambda_j \in \Lambda_i$. Then, it gives the following secret value to the user u_t:

$$\forall \lambda_j \in \Lambda_i \ : \ D_j = g^{r_t - r'} \cdot H(\lambda_j)^{r_j}, D'_j = g^{r_j}.$$

Then, the user computes $g^{r'} \cdot D_j$ for all its attributes key components and finally obtains its whole secret key set as

$$\mathrm{SK}_{u_t} = \Big(D = g^{\frac{(\alpha_1+\cdots+\alpha_m)+r_t}{\beta}}, \quad \forall \lambda_j \in S : D_j = g^{r_t} \cdot H(\lambda_j)^{r_j}, D'_j = g^{r_j} \Big)$$

where $S = \bigcup_{i=1}^m \Lambda_i$.

During the key generation phase using the 2PC protocol, the proposed scheme (especially 2PC protocol) requires $(3m+1)C_0$ messages additively to the key issuing overhead in the previous multiauthority ABE schemes in terms of the communication cost, where m is number of key authorities the user is associated with, and C_0 is the bit size of an element in $\mathbb{G}_0$. However, it is important to note that the 2PC protocol is done only once during the initial key generation phase for each user. Therefore, it is negligible compared to the communication overhead for encryption or key update, which could be much more frequently performed in the DTNs. (The detailed communication cost will be analyzed in Section V-A.)

In terms of the computation cost, each local authority is required to perform two more exponentiation operations. Each user needs to perform $m+1$ multiplication operations for the

[2]To prove security, we assume the parameters for the hash functions are set up by an honest party as in [24], [26], which means the random oracles are not controlled by the adversary in the security proof.

key generation, which incurs negligible computation cost compared to the other pairing or exponentiation operations. (The detailed computation cost will be analyzed in Section V-C.) These costs would be also incurred only for the initial key generation procedures. Therefore, the additional computation overhead for the key generation using the 2PC protocol is acceptable in the system.

3) Data Encryption: When a sender wants to deliver its confidential data M, he defines the tree access structure $\mathcal{T}$ over the universe of attributes $\mathcal{L}$, encrypts the data under $\mathcal{T}$ to enforce attribute-based access control on the data, and stores it into the storage node.

The encryption algorithm chooses a polynomial q_x for each node x in the tree $\mathcal{T}$. These polynomials are chosen in a top-down manner, starting from the root node R.

For each node x in the tree $\mathcal{T}$, the algorithm sets the degree d_x of the polynomial q_x to be one less than the threshold value k_x of that node, that is, $d_x = k_x - 1$. For the root node R, it chooses a random $s \in \mathbb{Z}_p^*$ and sets $q_R(0) = s$. Then, it sets d_R other points of the polynomial q_R randomly to define it completely. For any other node x, it sets $q_x(0) = q_{p(x)}(\mathrm{index}(x))$ and chooses d_x other points randomly to completely define q_x.

Let Y be the set of leaf nodes in the access tree. To encrypt a message $M \in \mathbb{G}_1$ under the tree access structure $\mathcal{T}$, it constructs a ciphertext using public keys of each authority as

$$\mathrm{CT} = (\mathcal{T}, \tilde{C} = Me(g,g)^{(\alpha_1+\cdots+\alpha_m)s}, C = h^s, \\ \forall y \in Y : C_y = g^{q_y(0)}, C'_y = H(\lambda_y)^{q_y(0)}),$$

where $\tilde{C}$ can be computed as $\tilde{C} = M \cdot (\mathrm{PK}_{A_1} \times \cdots \times \mathrm{PK}_{A_m})^s = Me(g,g)^{(\alpha_1+\cdots+\alpha_m)s}$.

After the construction of CT, the sender stores it to the storage node securely. On receiving any data request query from a user, the storage node responds with CT to the user.

It is important to note that the sender can define the access policy under attributes of any chosen set of multiple authorities without any restrictions on the logic expressiveness as opposed to the previous multiauthority schemes [4], [9].

4) Data Decryption: When a user receives the ciphertext CT from the storage node, the user decrypts the ciphertext with its secret key. The algorithm performs in a recursive way. We first define a recursive algorithm $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)$ that takes as inputs a ciphertext CT, a private key SK, which is associated with a set Λ of attributes, and a node x from the tree $\mathcal{T}$. It outputs a group element of $\mathbb{G}$ or $\perp$.

Without loss of generality, we suppose that a user u_t performs the decryption algorithm. If x is a leaf node, then define as follows. If $\lambda_x \in \Lambda$, then

$$\begin{aligned} &\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x) \\ &= \frac{e(D_x, C_x)}{e(D'_x, C'_x)} = \frac{e(g^{r_t} \cdot H(\lambda_x)^{r_x}, g^{q_x(0)})}{e(g^{r_x}, H(\lambda_x)^{q_x(0)})} \\ &= \frac{e(g^{r_t}, g^{q_x(0)}) \cdot e(H(\lambda_x)^{r_x}, g^{q_x(0)})}{e(g^{r_x}, H(\lambda_x)^{q_x(0)})} \\ &= e(g,g)^{r_t q_x(0)}. \end{aligned} \tag{1}$$

If $\lambda_x \notin \Lambda$, we define $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x) = \perp$.

We now consider the recursive case when x is a nonleaf node. The algorithm $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)$ then proceeds as follows. For all nodes z that are children of x, it calls $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, z)$ and stores the output as F_z. Let S_x be an arbitrary k_x-sized set of child nodes z such that $F_z \neq \perp$. If no such set exists, then the node was not satisfied and the function returns $\perp$.

Otherwise, we compute

$$\begin{aligned} F_x &= \prod_{z \in S_x} F_z^{\Delta_{i,S'_x}(0)}, \qquad \text{where} \quad \begin{array}{l} i = \mathrm{index}(z), \\ S'_x = \{\mathrm{index}(z) : z \in S_x\} \end{array} \\ &= \prod_{z \in S_x} (e(g,g)^{r_t \cdot q_z(0)})^{\Delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} (e(g,g)^{r_t \cdot q_{p(z)}(\mathrm{index}(z))})^{\Delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} e(g,g)^{r_t \cdot q_x(i) \cdot \Delta_{i,S'_x}(0)} \\ &= e(g,g)^{r_t \cdot q_x(0)} \end{aligned} \tag{2}$$

and return the result.

The decryption algorithm begins by calling the function on the root node R of the access tree. We observe that $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, R) = e(g,g)^{r_t s}$ if the tree $\mathcal{T}$ is satisfied by Λ for all $\lambda_i \in \Lambda$. When we set $A = \mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, R) = e(g,g)^{r_t s}$, the algorithm decrypts the ciphertext by computing $\tilde{C}/(e(C,D)/A) = M$.

C. Revocation

We observed that it is impossible to revoke specific attribute keys of a user without rekeying the whole set of key components of the user in ABE key structure since the whole key set of a user is bound with the same random value in order to prevent any collusion attack. Therefore, revoking a single attribute in the system requires all users who share the attribute to update all their key components even if the other attributes of them are still valid. This seems very inefficient and may cause severe overhead in terms of the computation and communication cost, especially in large-scaled DTNs.

For example, suppose that a user u_t is qualified with l different attributes. Then, all l attribute keys of the user u_t are generated with the same random number r_t in the ABE key architecture. When an attribute of the user is required to be revoked ($l-1$ other attribute keys of the user are still valid), the other valid $l-1$ keys should be updated with another new r'_t that is different from r_t and delivered to the user. Unless the other $l-1$ keys are updated, the attribute key that is to be revoked could be used as a valid key until their updates since it is still bound with the same r_t. Therefore, in order to revoke a single attribute key of a user, $O(l)$ keys of the user need to be updated. If n users are sharing the attribute, then total $O(nl)$ keys need to be updated in order to revoke just a single attribute in the system.

One promising way to immediately revoke an attribute of specific users is to reencrypt the ciphertext with each attribute group key and selectively distribute the attribute group key to authorized (non-revoked) users who are qualified with the attribute. Before distributing the ciphertext, the storage node receives a set of membership information for each attribute group G that appears in the access tree of CT from the corresponding authorities and reencrypts it as follows.

TABLE I
EXPRESSIVENESS, KEY ESCROW, AND REVOCATION ANALYSIS

Scheme	Authority	Expressiveness	Key Escrow	Revocation
BSW [13]	single	–	yes	periodic attribute revocation
HV [9]	multiple	AND	yes	periodic attribute revocation
RC [4]	multiple	AND	yes	immediate system-level user revocation
Proposed	multiple	any monotone access structure	no	immediate attribute-level user revocation

1) For all $G_y \subset G$, chooses a random $K_{\lambda_y} \in \mathbb{Z}_p^*$. Then, reencrypts CT and generates

$$\mathrm{CT}' = \Big(\mathcal{T}, \tilde{C} = Me(g,g)^{(\alpha_1+\cdots+\alpha_m)s}, C = h^s, \forall y \in Y : C_y = g^{q_y(0)}, C'_y = \left(H(\lambda_y)^{q_y(0)}\right)^{K_{\lambda_y}}\Big).$$

2) Generates a header message $\mathsf{Hdr} = \{\forall y \in Y : \mathsf{Hdr}_y\}$ where each Hdr_y contains the encrypted attribute group keys K_{λ_y}, which could be only decrypted by nonrevoked attribute group members. This can be done by exploiting many previous stateful or stateless group key management schemes [18], [29]–[31]. In this paper, we will adopt the complete subtree method [29], which requires each user to store additional $\log(n)$ key encryption keys (KEKs). The header message would be at most $(n-l)\log\frac{n}{n-l}$ size for each attribute group, where n and l arethe number of all users in the system and that of users in the attribute group, respectively.

On receiving any data request query from a user, the storage node responds with $(\mathsf{Hdr}, \mathrm{CT}')$ to the user. When the user receives them, he first obtains the attribute group keys for all attributes in S that he holds from Hdr. If a user u_t is associated with an attribute λ_j and not revoked from G_j (that is, $u_t \in G_j$), he can decrypt the attribute group key K_{λ_j} from Hdr_j and updates its secret key with it as follows:

$$\mathrm{SK}_{u_t} = \Big(D = g^{\frac{(\alpha_1+\cdots+\alpha_m)+r_t}{\beta}}, \forall \lambda_j \in S : D_j = g^{r_t} \cdot H(\lambda_j)^{r_j}, D'_j = (g^{r_j})^{\frac{1}{K_{\lambda_j}}}\Big).$$

Then, the user can decrypt the ciphertext CT' with its secret key following the above decryption algorithm.

D. Key Update

When a user comes to hold or drop an attribute, the corresponding key should be updated to prevent the user from accessing the previous or subsequent encrypted data for backward or forward secrecy, respectively.

The key update procedure is launched by sending a join or leave request for some attribute group from a user who wants to hold or drop the attribute to the corresponding authority. On receipt of the membership change request for some attribute groups, it notifies the storage node of the event. Without loss of generality, suppose there is any membership change in G_i (e.g., a user comes to hold or drop an attribute λ_i at some time instance). Then, the update procedure progresses as follows.

1) The storage node selects a random $s' \in \mathbb{Z}_p^*$ and a K'_{λ_i}, which is different from the previous attribute group key K_{λ_i}. Then, it reencrypts the ciphertext CT using the public parameters PK as

$$\mathrm{CT}' = \Big(\mathcal{T}, \tilde{C} = Me(g,g)^{(\alpha_1+\cdots+\alpha_m)(s+s')}, C = h^{s+s'}, C_i = g^{q_i(0)+s'}, C'_i = \left(H(\lambda_i)^{q_i(0)+s'}\right)^{K'_{\lambda_i}}, \forall y \in Y \setminus \{i\} : C_y = g^{q_y(0)+s'}, C'_y = \left(H(\lambda_y)^{q_y(0)+s'}\right)^{K_{\lambda_y}}\Big).$$

For the other attribute groups that are not affected by the membership changes, the attribute group keys do not necessarily need to be updated.

2) The storage node generates a new header message Hdr_i with K'_{λ_i} such that a set of attribute group members including a new joining user (for backward secrecy) or excluding a leaving user (for forward secrecy) can decrypt $K'_{\lambda_i} \cdot \forall y \in Y \setminus \{i\}, \mathsf{Hdr}_y$ remains the same.

When a user sends a request query for the data afterward, the storage node responds with the newly updated Hdr and ciphertext CT' encrypted under the updated keys.

It is important to note that even if a user is revoked from some attribute groups, he may still be able to access the data with the other attributes that he holds as long as they satisfy the policy because they would still be effective in the system.

V. ANALYSIS

In this section, we first analyze and compare the efficiency of the proposed scheme to the previous multiauthority CP-ABE schemes in theoretical aspects. Then, the efficiency of the proposed scheme is demonstrated in the network simulation in terms of the communication cost. We also discuss its efficiency when implemented with specific parameters and compare these results to those obtained by the other schemes.

A. Efficiency

Table I shows the authority architecture, logic expressiveness of access structure that can be defined under different disjoint sets of attributes (managed by different authorities), key escrow, and revocation granularity of each CP-ABE scheme. In the proposed scheme, the logic can be very expressive as in the single authority system like BSW [13] such that the access policy can be expressed with any monotone access structure under attributes of any chosen set of authorities; while HV [9] and RC [4] schemes only allow the AND gate among the sets of attributes managed by different authorities. The revocation in the proposed scheme can be done in an immediate way as opposed to BSW. Therefore, attributes of users can be revoked at any time even before the expiration time that might be set

TABLE II
EFFICIENCY ANALYSIS

System	Ciphertext size	Rekeying message	Private key size	Public key size
BSW [13]	$(2t+1)C_0 + C_1 + C_{\mathcal{T}}$	$l(2k+1)C_0$	$(2k+1)C_0$	$C_0 + C_1$
HV [9]	$(2t+m)C_0 + mC_1 + C_{\mathcal{T}}$	$l(2k+1)C_0$	$(2k+m)C_0$	$mC_0 + mC_1$
RC [4]	$(2t+3r+m)C_0 + mC_1 + C_{\mathcal{T}}$	0	$(3k+2m)C_0$	$m(t+4)C_0 + mC_1$
Proposed	$(2t+1)C_0 + C_1 + C_{\mathcal{T}}$	$(n-l)\log\frac{n}{n-l}C_p$	$(2k+1)C_0 + \log n C_k$	$C_0 + mC_1$

C_0: bit size of an element in $\mathbb{G}_0$, C_1: bit size of an element in $\mathbb{G}_1$, C_p: bit size of an element in $\mathbb{Z}_p^*$,
C_k: bit size of a KEK, $C_{\mathcal{T}}$: bit size of an access tree $\mathcal{T}$ in the ciphertext, r: the number of revoked users,
l: the number of users in an attribute group, n: the number of all users in the system,
m: the number of authorities in the system, k: the number of attributes associated with private key of a user,
u: the number of attributes in the system, t: the number of attributes appeared in $\mathcal{T}$.

to the attribute. This enhances security of the stored data by reducing the windows of vulnerability. In addition, the proposed scheme realizes more fine-grained user revocation for each attribute rather than for the whole system as opposed to RC. Thus, even if a user comes to hold or drop any attribute during the service in the proposed scheme, he can still access the data with other attributes that he is holding as long as they satisfy the access policy defined in the ciphertext. The key escrow problem is also resolved in the proposed scheme such that the confidential data would not be revealed to any curious key authorities.

Table II summarizes the efficiency comparison results among CP-ABE schemes. In the comparison, rekeying message size represents the communication cost that the key authority or the storage node needs to send to update nonrevoked users' keys for an attribute. Private key size represents the storage cost required for each user to store attribute keys or KEKs. Public key size represents the size of the system public parameters. In this comparison, the access tree is constructed with attributes of m different authorities except in BSW of which total size is equal to that of the single access tree in BSW. As shown in Table II, the proposed scheme needs rekeying message (Hdr) size of at most $(n-l)\log\frac{n}{n-l}C_p$ to realize user-level access control for each attribute in the system. Although RC does not need to send additional rekeying message for user revocations as opposed to the other schemes, its ciphertext size is linear to the number of revoked users in the system since the user revocation message is included in the ciphertext. The proposed scheme requires a user to store $\log n$ more KEKs than BSW. However, it has an effect on reducing the rekeying message size. The proposed scheme is as efficient as the basic BSW in terms of the ciphertext size while realizing more secure immediate rekeying in multiauthority systems.

B. Simulation

In this simulation, we consider DTN applications using the Internet protected by the attribute-based encryption. Almeroth and Anmar [32] demonstrated the group behavior in the Internet's multicast backbone network (MBone). They showed that the number of users joining a group follows a Poisson distribution with rate $\tilde{\lambda}$, and the membership duration time follows an exponential distribution with a mean duration $1/\mu$. Since each attribute group can be shown as an independent network multicast group where the members of the group share a common attribute, we show the simulation result following this probabilistic behavior distribution [32].

We suppose that user join and leave events are independently and identically distributed in each attribute group following Poisson distribution. The membership duration time for an attribute is assumed to follow an exponential distribution. We set the interarrival time between users as 20 min $(\tilde{\lambda} = 3)$ and the average membership duration time as 20 h $(1/\mu = 20)$. Fig. 2 represents the number of current users and revoked users in an attribute group during 100 h.

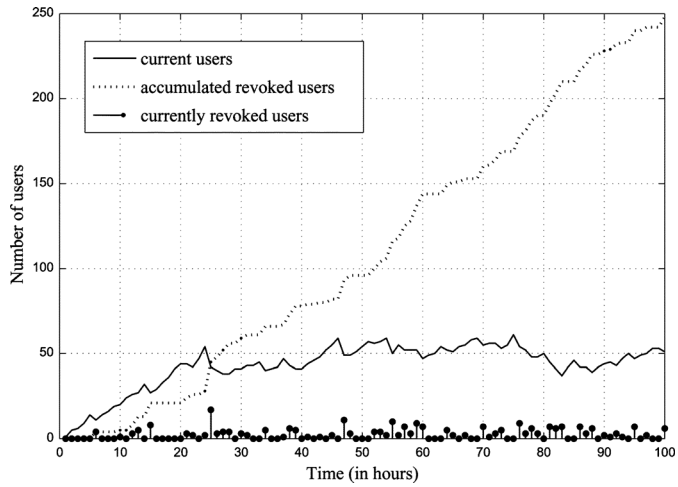


Fig. 2. Number of users in an attribute group.

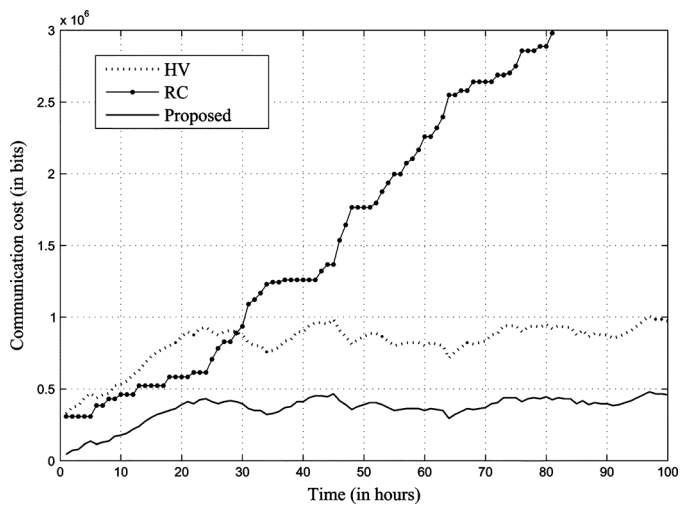


Fig. 3. Communication cost in the multiauthority CP-ABE systems.

Fig. 3 shows the total communication cost that the sender or the storage node needs to send on a membership change in each multiauthority CP-ABE scheme. It includes the ciphertext and rekeying messages for nonrevoked users. It is measured in bits. In this simulation, the total number of users in the network is

TABLE III
COMPARISON OF COMPUTATION COST

		Pairing	Exp. in $\mathbb{G}_0$	Exp. in $\mathbb{G}_1$	Computation (ms)
Time (ms)		2.9	1.0	0.2	
BSW [13]	S		$2t+1$	1	$2t+1.2$
	U	$2k+1$		$\log t$	$5.8k+0.2\log t$ $+2.9$
HV [9]	S		$2t+1$	1	$2t+1.2$
	U	$2k+m$		$m\log(t/m)$	$5.8k+2.9m$ $+0.2m\log(t/m)$
RC [4]	S		$3t+1$	1	$3t+1.2$
	U	$3k+m$		$m\log(t/m)$	$8.7k+2.9m$ $+0.2m\log(t/m)$
Proposed	S		$2t+1$	1	$2t+1.2$
	U	$2k+1$	k	$\log t$	$6.8k+0.2\log t$ $+2.9$

S: sender, U: user

10 000, and the number of attributes in the system is 30. The number of the key authorities is 10, and the average number of attributes associated with a user's key is 10. For a fair comparison with regard to the security perspective, we set the rekeying periods in HV as $1/\tilde{\lambda}$ min. To achieve an 80-bit security level, we set $C_0 = 512, C_p = 160$. $C_{\mathcal{T}}$ is not added to the simulation result because it is common in all multiauthority CP-ABE schemes. As shown in Fig. 3, the communication cost in HV is less than RC in the beginning of the simulation time (until about 30 h). However, as the time elapses, it increases conspicuously because the number of revoked users increases accumulatively. The proposed scheme requires the least communication cost in the network system since the rekeying message in Hdr is comparatively less than the other multiauthority schemes.

C. Implementation

Next, we analyze and measure the computation cost for encrypting (by a sender) and decrypting (by a user) a data. We used a Type-A curve (in the pairing-based cryptography (PBC) library [33]) providing groups in which a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ is defined. Although such curves provide good computational efficiency (especially for pairing computation), the same does not hold from the point of view of the space required to represent group elements. Indeed, each element of $\mathbb{G}_0$ needs 512 bits at an 80-bit security level and 1536 bits when 128-bit of security are chosen.

Table III shows the computational time results. For each operation, we include a benchmark timing. Each cryptographic operation was implemented using the PBC library ver. 0.4.18 [33] on a 3.0-GHz processor PC. The public key parameters were selected to provide 80-bit security level. The implementation uses a 160-bit elliptic curve group based on the supersingular curve $y^2 = x^3 + x$ over a 512-bit finite field. The computational cost is analyzed in terms of the pairing, exponentiation operations in $\mathbb{G}_0$ and $\mathbb{G}_1$. The comparatively negligible hash, symmetric key, and multiplication operations in the group are ignored in the time result. In this analysis, we assume that the access tree in the ciphertext is a complete binary tree.

Computation costs in Table III represent the upper bound of each cost. We can see that the total computation time to encrypt data by a sender in the proposed scheme is the same as BSW, while decryption time by a user requires k exponentiations in $\mathbb{G}_0$ more. These exponentiation operations are to realize the fine-grained key revocation for each attribute group. Therefore, we can observe that there is a tradeoff between computational overhead and granularity of access control, which is closely related to the windows of vulnerability. However, the computation cost for encryption by a sender and decryption by a user are more efficient compared to the other multiauthority schemes.

VI. SECURITY

In this section, we prove the security of our scheme with regard to the security requirements discussed in Section II.

A. Collusion Resistance

In CP-ABE, the secret sharing must be embedded into the ciphertext instead to the private keys of users. Like the previous ABE schemes [11], [13], the private keys (SK) of users are randomized with personalized random values selected by the CA such that they cannot be combined in the proposed scheme. In order to decrypt a ciphertext, the colluding attacker should recover $e(g,g)^{(\alpha_1+\cdots+\alpha_m)s}$. To recover this, the attacker must pair C_y from the ciphertext and D_y from the other colluding users' private keys for an attribute λ_y (we suppose that the attacker does not hold the attribute λ_y). However, this results in the value $e(g,g)^{(\alpha_1+\cdots+\alpha_m)s}$ blinded by some random value, which is uniquely assigned to each user, even if the attribute group keys for the attributes that the user holds are still valid. This value can be blinded out if and only if the user has the enough key components to satisfy the secret sharing scheme embedded in the ciphertext. Another collusion attack scenario is the collusion between revoked users in order to obtain the valid attribute group keys for some attributes that they are not authorized to have (e.g., due to revocation). The attribute group key distribution protocol, which is complete subtree method in the proposed scheme, is secure in terms of the key indistinguishability [29]. Thus, the colluding revoked users can by no means obtain any valid attribute group keys for attributes that they are not authorized to hold. Therefore, the desired value $e(g,g)^{(\alpha_1+\cdots+\alpha_m)s}$ cannot be recovered by collusion attack since the blinding value is randomized from a particular user's private key.

Collusion among the local authorities could determine the personalized key component $D = g^{((\alpha_1+\cdots+\alpha_m)+r_t)/\beta}$ of some user u_t. However, each attribute key component of the user is blinded in the local authorities' view in that they are divided by the secret $g^{r'}$, which is only known to the user and CA. Therefore, the colluding local authorities cannot derive the whole set of secret keys of users.

B. Data Confidentiality

In our trust model, the multiple key authorities are no longer fully trusted as well as the storage node even if they are honest. Therefore, the plain data to be stored should be kept secret from them as well as from unauthorized users.

Data confidentiality on the stored data against unauthorized users can be trivially guaranteed. If the set of attributes of a user cannot satisfy the access tree in the ciphertext, he cannot recover the desired value $e(g,g)^{rs}$ during the decryption process, where r is a random value uniquely assigned to him. On the other hand, when a user is revoked from some attribute groups that satisfy

the access policy, he cannot decrypt the ciphertext either unless the rest of the attributes of him satisfy the access policy. In order to decrypt a node x for an attribute λ_x, the user needs to pair C'_x from the ciphertext and D'_x from its private key. However, this cannot result in the value $e(g,g)^{rq_x(0)}$, which is desired to generate $e(g,g)^{rs}$, since C'_x is blinded by the updated attribute group key that the revoked user from the attribute group can by no means obtain.

Another attack on the stored data can be launched by the storage node and the key authorities. Since they cannot be totally trusted, confidentiality for the stored data against them is another essential security criteria for secure data retrieval in DTNs. The local authorities issue a set of attribute keys for their managing attributes to an authenticated user u, which are blinded by secret information that is distributed to the user from CA. They also issue the user a personalized secret key by performing the secure 2PC protocol with CA. As we discussed in Theorem 1, this key generation protocol discourages each party to obtain each other's master secret key and determine the secret key issued from each other. Therefore, they could not have enough information to determine the whole set of secret key of the user individually.

Even if the storage node manages the attribute group keys, it cannot decrypt any of the nodes in the access tree in the ciphertext. This is because it is only authorized to reencrypt the ciphertext with each attribute group key, but is not allowed to decrypt it (that is, any of the key components of users are not given to the node). Therefore, data confidentiality against the curious key authorities and storage node is also ensured.

C. Backward and Forward Secrecy

When a user comes to hold a set of attributes that satisfy the access policy in the ciphertext at some time instance, the corresponding attribute group keys are updated and delivered to the valid attribute group members securely (including the user). In addition, all of the components encrypted with a secret key s in the ciphertext are reencrypted by the storage node with a random s', and the ciphertext components corresponding to the attributes are also reencrypted with the updated attribute group keys. Even if the user has stored the previous ciphertext exchanged before he obtains the attribute keys and the holding attributes satisfy the access policy, he cannot decrypt the pervious ciphertext. This is because, even if he can succeed in computing $e(g,g)^{r(s+s')}$ from the current ciphertext, it will not help to recover the desired value $e(g,g)^{(\alpha_1+\cdots+\alpha_m)s}$ for the previous ciphertext since it is blinded by a random s'. Therefore, the backward secrecy of the stored data is guaranteed in the proposed scheme.

On the other hand, when a user comes to drop a set of attributes that satisfy the access policy at some time instance, the corresponding attribute group keys are also updated and delivered to the valid attribute group members securely (excluding the user). Then, all of the components encrypted with a secret key s in the ciphertext are reencrypted by the storage node with a random s', and the ciphertext components corresponding to the attributes are also reencrypted with the updated attribute group keys. Then, the user cannot decrypt any nodes corresponding to the attributes after revocation due to the blindness resulted from newly updated attribute group keys. In addition, even if the user has recovered $e(g,g)^{(\alpha_1+\cdots+\alpha_m)s}$ before he was revoked from the attribute groups and stored it, it will not help to decrypt the subsequent ciphertext $e(g,g)^{(\alpha_1+\cdots+\alpha_m)(s+s')}$ reencrypted with a new random s'. Therefore, the forward secrecy of the stored data is guaranteed in the proposed scheme.

VII. Conclusion

DTN technologies are becoming successful solutions in military applications that allow wireless devices to communicate with each other and access the confidential information reliably by exploiting external storage nodes. CP-ABE is a scalable cryptographic solution to the access control and secure data retrieval issues. In this paper, we proposed an efficient and secure data retrieval method using CP-ABE for decentralized DTNs where multiple key authorities manage their attributes independently. The inherent key escrow problem is resolved such that the confidentiality of the stored data is guaranteed even under the hostile environment where key authorities might be compromised or not fully trusted. In addition, the fine-grained key revocation can be done for each attribute group. We demonstrate how to apply the proposed mechanism to securely and efficiently manage the confidential data distributed in the disruption-tolerant military network.

Appendix
Security Proof of Theorem 1

Proof: To show security, we consider the cases of corrupt central authority CA and corrupt local authority A_i.

First, for a corrupt local authority, our simulator $\mathsf{Sim}_{\mathrm{CA}}$ proceeds as follows.

> $\mathsf{Sim}_{\mathrm{CA}}$: First, it will run the arithmetic 2PC simulator for computation of $(\alpha_i+\gamma_i)\beta$. This 2PC will extract α_i from the CA and expect to be provided with $x=(\alpha_i+\gamma_i)\beta \bmod p$. We will choose a random value $x \in_R \mathbb{Z}_p^*$, and give it to the arithmetic 2PC simulator. Note that this is correctly distributed since there is some β such that $x=(\alpha_i+\gamma_i)\beta$ for any x, α_i, γ_i. Next, our simulator will receive T from the adversary, as well as the corresponding zero knowledge proof. We will use the extractor for the proof system to extract τ. Then, it will send α_i to the trusted party and receive $D_i = g^{(\alpha_i+\gamma_i)/\beta}$. Finally, it will compute $B = D_i^{1/\tau}$ and send it to the local authority.

Consider a hybrid simulator $\mathsf{Hyb}_{\mathrm{CA}}$ that takes as input the CA's secrets γ_i and β. It will compute $x=(\alpha_i+\gamma_i)\beta$ using the arithmetic 2PC simulator. When the 2PC simulator provides α_i and asks for output, it will correctly compute $(\alpha_i+\gamma_i)\beta$. Then, it will complete the execution as in the real protocol. This protocol is clearly indistinguishable from the real central authority's protocol by the security of the arithmetic 2PC.

Second, for a corrupt central authority, our simulator Sim_A proceeds as follows.

> Sim_A: First, it will run the arithmetic 2PC simulator for computation of $(\alpha_i+\gamma_i)\beta$. In the process, it will extract γ_i. Next, the simulator will choose random value $T \in_R \mathbb{G}_0$ and send it to CA. It will receive B from CA, and extract β from the corresponding proof. Then, it will send γ_i, β to the trusted party and receive $g^{(\alpha_i+\gamma_i)/\beta}$, which will be A_i's private output.

Consider a hybrid simulator Hyb_A takes as input the A_i's secret α_i. It first runs the arithmetic 2PC simulator for the computation of x with the correct output value according to α_i, and then completes the protocol as the honest local authority would. This is clearly indistinguishable from the real local authority's protocol by the security of the arithmetic 2PC.

Now, assuming that the proof of knowledge scheme is secure, Hyb_A should be indistinguishable from the above simulator Sim_A. This is because the value x, T used by Sim_A will be distributed identically to those in Hyb_A. (Since τ is chosen at random in the real protocol, T will be distributed uniformly over $\mathbb{G}_0$ in the real protocol as in the simulated protocol.) Thus, interaction with our simulation is indistinguishable from interaction with an honest local authority.

Therefore, our construction is a secure 2PC protocol. ■

REFERENCES

[1] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: Routing for vehicle-based disruption tolerant networks," in *Proc. IEEE INFOCOM*, 2006, pp. 1–11.

[2] M. Chuah and P. Yang, "Node density-based adaptive routing scheme for disruption tolerant networks," in *Proc. IEEE MILCOM*, 2006, pp. 1–6.

[3] M. M. B. Tariq, M. Ammar, and E. Zequra, "Mesage ferry route design for sparse ad hoc networks with mobile nodes," in *Proc. ACM MobiHoc*, 2006, pp. 37–48.

[4] S. Roy and M. Chuah, "Secure data retrieval based on ciphertext policy attribute-based encryption (CP-ABE) system for the DTNs," Lehigh CSE Tech. Rep., 2009.

[5] M. Chuah and P. Yang, "Performance evaluation of content-based information retrieval schemes for DTNs," in *Proc. IEEE MILCOM*, 2007, pp. 1–7.

[6] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable secure file sharing on untrusted storage," in *Proc. Conf. File Storage Technol.*, 2003, pp. 29–42.

[7] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Mediated ciphertext-policy attribute-based encryption and its application," in *Proc. WISA*, 2009, LNCS 5932, pp. 309–323.

[8] N. Chen, M. Gerla, D. Huang, and X. Hong, "Secure, selective group broadcast in vehicular networks using dynamic attribute based encryption," in *Proc. Ad Hoc Netw. Workshop*, 2010, pp. 1–8.

[9] D. Huang and M. Verma, "ASPE: Attribute-based secure policy enforcement in vehicular ad hoc networks," *Ad Hoc Netw.*, vol. 7, no. 8, pp. 1526–1535, 2009.

[10] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," Cryptology ePrint Archive: Rep. 2010/351, 2010.

[11] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. Eurocrypt*, 2005, pp. 457–473.

[12] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. ACM Conf. Comput. Commun. Security*, 2006, pp. 89–98.

[13] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Security Privacy*, 2007, pp. 321–334.

[14] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *Proc. ACM Conf. Comput. Commun. Security*, 2007, pp. 195–203.

[15] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proc. ASIACCS*, 2010, pp. 261–270.

[16] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proc. ACM Conf. Comput. Commun. Security*, 2008, pp. 417–426.

[17] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure attribute-based systems," in *Proc. ACM Conf. Comput. Commun. Security*, 2006, pp. 99–112.

[18] S. Rafaeli and D. Hutchison, "A survey of key management for secure group communication," *Comput. Surv.*, vol. 35, no. 3, pp. 309–329, 2003.

[19] S. Mittra, "Iolus: A framework for scalable secure multicasting," in *Proc. ACM SIGCOMM*, 1997, pp. 277–288.

[20] P. Golle, J. Staddon, M. Gagne, and P. Rasmussen, "A content-driven access control system," in *Proc. Symp. Identity Trust Internet*, 2008, pp. 26–35.

[21] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proc. ACM Conf. Comput. Commun. Security*, 2007, pp. 456–465.

[22] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute-based encryption," in *Proc. ICALP*, 2008, pp. 579–591.

[23] X. Liang, Z. Cao, H. Lin, and D. Xing, "Provably secure and efficient bounded ciphertext policy attribute based encryption," in *Proc. ASIACCS*, 2009, pp. 343–352.

[24] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proc. ACM Conf. Comput. Commun. Security*, 2009, pp. 121–130.

[25] M. Chase, "Multi-authority attribute based encryption," in *Proc. TCC*, 2007, LNCS 4329, pp. 515–534.

[26] S. S. M. Chow, "Removing escrow from identity-based encryption," in *Proc. PKC*, 2009, LNCS 5443, pp. 256–276.

[27] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya, "P-signatures and noninteractive anonymous credentials," in *Proc. TCC*, 2008, LNCS 4948, pp. 356–374.

[28] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Hysyanskaya, and H. Shacham, "Randomizable proofs and delegatable anonymous credentials," in *Proc. Crypto*, LNCS 5677, pp. 108–125.

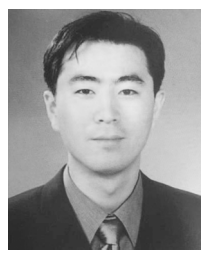
[29] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Proc. CRYPTO*, 2001, LNCS 2139, pp. 41–62.

[30] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," in *Proc. ACM SIGCOMM*, 1998, pp. 68–79.

[31] A. T. Sherman and D. A. McGrew, "Key establishment in large dynamic groups using one-way function trees," *IEEE Trans. Softw. Eng.*, vol. 29, no. 5, pp. 444–458, May 2003.

[32] K. C. Almeroth and M. H. Ammar, "Multicast group behavior in the Internet's multicast backbone (MBone)," *IEEE Commun. Mag.*, vol. 35, no. 6, pp. 124–129, Jun. 1997.

[33] "The Pairing-Based Cryptography Library," Accessed Aug. 2010 [Online]. Available: http://crypto.stanford.edu/pbc/



Junbeom Hur received the B.S. degree from Korea University, Seoul, Korea, in 2001, and the M.S. and Ph.D. degrees from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2005 and 2009, respectively, all in computer science.

He was with the University of Illinois at Urbana–Champaign as a Postdoctoral Researcher from 2009 to 2011. He is currently an Assistant Professor with the School of Computer Science and Engineering, Chung-Ang University, Korea. His research interests include information security, mobile computing security, cyber security, and applied cryptography.

Kyungtae Kang (M'06) received the B.S. degree in computer science and engineering and M.S. and Ph.D. degrees in electrical engineering and computer science from Seoul National University (SNU), Seoul, Korea, in 1999, 2001, and 2007, respectively.

He is an Assistant Professor with the Department of Computer Science and Engineering, Hanyang University, Seoul, Korea. Prior to his tenure at Hanyang University, he spent four years as a Postdoctoral Research Associate with the University of Illinois at Urbana–Champaign (UIUC). His research interests lie primarily in systems, including operating systems, networking, sensor networks, distributed systems, and real-time embedded systems. His most recent research interest is in the interdisciplinary area of cyber-physical systems.

Dr. Kang is a member of the Association for Computing Machinery (ACM).