Information Sciences 275 (2014) 370–384

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Ciphertext-policy hierarchical attribute-based encryption with short ciphertexts

Hua Deng [a,b], Qianhong Wu [b,*], Bo Qin [c], Josep Domingo-Ferrer [d], Lei Zhang [e], Jianwei Liu [b], Wenchang Shi [c]

[a] *School of Computer, Wuhan University, Wuhan, China*
[b] *School of Electronics and Information Engineering, Beihang University, Beijing, China*
[c] *School of Information, Renmin University of China, Beijing, China*
[d] *Dept. of Computer Engineering and Mathematics, Universitat Rovira i Virgili, Catalonia, Spain*
[e] *Software Engineering Institute, East China Normal University, Shanghai, China*

ARTICLE INFO

Article history:
Received 27 August 2013
Received in revised form 4 January 2014
Accepted 8 January 2014
Available online 1 February 2014

Keywords:
Cloud storage
Access control
Attribute-based encryption
Ciphertext-policy attribute-based encryption

ABSTRACT

Attribute-based encryption (ABE) systems allow encrypting to uncertain receivers by means of an access policy specifying the attributes that the intended receivers should possess. ABE promises to deliver fine-grained access control of encrypted data. However, when data are encrypted using an ABE scheme, key management is difficult if there is a large number of users from various backgrounds. In this paper, we elaborate on ABE and propose a new versatile cryptosystem referred to as ciphertext-policy hierarchical ABE (CP-HABE). In a CP-HABE scheme, the attributes are organized in a matrix and the users having higher-level attributes can delegate their access rights to the users at a lower level. These features enable a CP-HABE system to host a large number of users from different organizations by delegating keys, e.g., enabling efficient data sharing among hierarchically organized large groups. We construct a CP-HABE scheme with short ciphertexts. The scheme is proven secure in the standard model under non-interactive assumptions.

1. Introduction

Attribute-based encryption (ABE) [26] is a cryptographic primitive which provides fine-grained access control over the outsourced ciphertexts (quite relevant in cloud environments). It allows encrypting to uncertain decryptors by means of an access policy specifying what attributes the intended decryptors should possess. ABE can be classified into two categories of key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE). In KP-ABE, each ciphertext is labeled by the encryptor with a set of descriptive attributes, and access policies over these attributes are ascribed to users' secret keys. The encryptor only needs to know the public attributes of the potential decryptors. CP-ABE is similar to KP-ABE, except that the access policy is labelled with each ciphertext and a secret key is associated with a user's attributes. The encryptor can associate the ciphertext with an access policy specifying the attributes that the authorized decryptors should have.

Although ABE systems have interesting features, they could be elaborated further to cater for more complicated applications. To see this point, let us consider the following scenario.

[*] Corresponding author. Address: School of Electronics and Information Engineering, Beihang University, Xueyuan Road 37, Haidian District, 100191 Beijing, China. Tel.: +86 10 8233 9469.

E-mail address: qhwu@xidian.edu.cn (Q. Wu).

http://dx.doi.org/10.1016/j.ins.2014.01.035
0020-0255/© 2014 Elsevier Inc. All rights reserved.

Motivating scenario: Company A may allow its employees to outsource encrypted data to a server maintained by a third party, e.g., a cloud service provider. The outsourced data are only accessible to the employees of company A and its peer companies. The employees of the companies may have attributes such as job ranks (e.g., general manager, department manager, project leader, and engineer) together with other attributes with hierarchical features, e.g., working years. Company A signs contracts with its peer companies to share the encrypted data if the employees of the peer companies possess the required attributes.

If employing the existing ABE schemes, company A needs to validate the attributes of the employees of its peer companies and distribute secret keys for all matching employees. This implies heavy communication, computation and management overheads. Moreover, the meaningful attributes of the employees of the peer companies should be kept private to prevent leaking sensitive information about the peer companies' internal organization and most valuable employees. Indeed, what needed is a kind of delegation of access rights, which can enable the peer companies to generate secret keys for their own employees.

There already are some ABE schemes which can provide *limited* delegation of access rights. The KP-ABE schemes in [10,19] allow users to delegate decryption keys for more *restrictive* access structures. That is, the access structure of the delegated key must be more restrictive than that of the original key so as to accommodate new attributes. As for CP-ABE, Bethencourt et al. [3], Goyal et al. [9] and Waters [34] presented CP-ABE schemes supporting delegation, but all of these schemes only allow delegating keys for attribute sets that are subsets of the original attribute sets. However, in the motivating scenario, an employee should be allowed to delegate to his/her subordinates without the constraint that the subordinates' attribute sets have to be subsets of the delegating employee's attribute set.

The difficulty of providing delegation for the motivating scenario lies in the deployment of secret sharing scheme. In most CP-ABE schemes [3,9,10,16,23,34], a secret sharing scheme is employed to realize an access policy associated with a ciphertext. In these realizations, each attribute belonging to the access policy obtains a share of the secret. This requires that the same attribute included in the attribute set of the secret key possess a separate key component so that the share can contribute to reconstructing the secret key to be used in decryption. But a delegator's secret key is generated by a key generator; hence, without knowing the secret key of the key generator, the delegator cannot generate a key component for a new attribute. This is the main shortcoming that prevents applying the above mentioned CP-ABE schemes to the motivating scenario.

1.1. Our contributions

In this paper we present a new cryptographic primitive referred to as hierarchical attribute-based encryption (HABE). The main goal of the HABE primitive is to provide an appropriate delegation mechanism for the motivating application scenario, as well as the flexible encryption of ABE. With HABE, company A can enable fine-grained access control over its outsourced data; the employees of the peer companies signing contracts with company A can access the data if they possess the required attributes. In particular, the peer companies can generate secret keys for their own employees: the employees with higher job rank (e.g., manager) can delegate secret keys to those with lower rank (e.g., engineer). Hence, at company A, HABE greatly reduces the communication and management overhead; at the peer companies, HABE also protects the sensitive information about the internal organization and the most valuable employees.

Our work is motivated by the observation that many attributes are hierarchical in the real world. This fact allows us to arrange the attribute universe in HABE in a matrix. The attributes higher in the hierarchy (e.g. manager) sit in upper levels of the matrix while attributes lower in the hierarchy (e.g. engineer) sit in lower levels of the matrix. This arrangement naturally leads to the notion of *hierarchical attributes* or *attribute vectors*, which can be formed by sampling attributes from an upper level to a lower level. By using the attribute vectors, we can achieve the required delegation mechanism that allows new attributes to be added into the original attribute set without requiring the delegator to know the secret key of the key generator. Our delegation mechanism is similar to that of hierarchical identity-based encryption (HIBE, recalled in Section 1.2) but, unlike HIBE (which allows delegation to any identity), our HABE ensures that only attributes valid and consistent with the attribute matrix can be delegated to.

We model ciphertext-policy HABE (CP-HABE), that can be applied to the motivating scenario. In CP-HABE, ciphertexts are generated with access policies specifying the attribute vectors that the potential decryptors should possess. Importantly, a user at a higher level can delegate a secret key to a user at a lower level without the constraint that the latter's attribute set must be a subset of the former's. We then define the full security of CP-HABE. In full security, the attacker is allowed to obtain the public parameters, create attribute vectors, specify access policies, and query for the keys corresponding to the sets of attribute vectors. Then the attacker outputs two challenge messages and a challenge set of attribute vectors. One of the two messages is chosen to generate a ciphertext associated with the challenge policy. Even for such a polynomial-time attacker, it is not possible to distinguish which message was used to generate the ciphertext, provided that the attacker did not query for the key associated with the attribute vector set that satisfies the challenge access policy.

We construct a CP-HABE scheme by employing a linear secret sharing scheme (LSSS) to achieve suitable expressiveness of access structures. One might attempt a straightforward construction of CP-HABE from HIBE by treating each attribute vector as an identity vector in the underlying HIBE. Unfortunately, such a straightforward construction is vulnerable to collusion attacks in which a coalition of users manages to decrypt ciphertexts intended to none of them, provided that the union of the attribute vectors of the colluding users meets the access policy. This kind of attacks must be prevented in practice. We address this problem by randomizing the secret keys assigned to each user, and, by using the well-established dual