Advanced Science and Technology Letters
Vol.99 (ITCS 2015), pp.27-30
http://dx.doi.org/10.14257/astl.2015.99.07

Selective Data Dissemination using Attribute Based Encryption

Eun-Kyu Lee and Ki Young Lee[1]

Dept. of Information and Telecommunication Engineering,
Incheon National University, Incheon, Korea
{eklee, kylee}@inu.ac.kr

Abstract. Ciphertext-Policy Attribute-Based Encryption provides an encrypted access control mechanism for broadcasting messages. A secret message is encrypted with an access control policy tree that contains the logical combination of different attributes. Each qualified user can apply and obtain a private key, which is associated with the various attributes of the applicant. Only the users whose attributes satisfy the policy tree can decrypt the message. However, in some scenarios, dynamic attributes are required and hence new keys must be regenerated and issued whenever a dynamic attribute updates, even if there might be hundreds of attributes involved in the key. This is not efficient since the cost of generating new private key is proportional to the number of attributes associated with that private key. In this paper we introduce the concept of Dynamic Attribute Based Encryption, a key revocation mechanism, which is necessary to prevent a user from keeping a private key with expired attributes.

Keywords: security, encryption, attribute based encryption, group comm.

1 Introduction

A technical manager at a software company wants to announce some important information to her group, whose members include three development subgroups: five test engineers, two interns, and one program manager. The technical manager has different messages to different subgroups – say, she needs to send out an internal deadline for current development of an undergoing project to development subgroup. Moreover, within this group, a development engineer may also want to contact some specific test engineers for their own testing plans. In this scenario of selective secure group communication, a group message is encrypted under some secure policy and sent to every members, but only qualified members can decrypt and read the message. Attribute-Based Encryption (ABE) [2] has been applied to the communications as an enabling technique that implements an encrypted access control.

In Ciphertext Policy (CP)-ABE [1], for example, a user's private key is associated with an arbitrary number of attributes. One attribute corresponds to one property.

[1] Ki Young Lee is the corresponding author.

ISSN: 2287-1233 ASTL

Properties such as name, ages and employers are different from person to person, so users have different attributes associated with their private keys. The publisher uses a policy tree, i.e., the logical combination of various attributes, and the public key to encrypt a message. Only clients with those attributes associated with their private keys satisfy the policy tree can decrypt the message.

In current CP-ABE, attributes are assumed to be static, i.e. their values never change. However, it is very possible that the value of an attribute gets updated over time. In the manager example, the executive level of an engineer may change as the employees promote and the office room may also change from time to time. Therefore, we see two types of attributes: time-unrelated attributes, whose value never updates, and time-related attributes, whose value gets updated as time goes. Based on this observation, the assumption of CP-ABE regarding attributes is problematic: an attribute not only represents a property of a member, but it also reflects the current state of that property, which in some cases are time-related. Current CP-ABE systems update time-related attributes associated within a private key by simply reconstructing the entire private key, which causes replacement of both time-related attributes and time-unrelated attributes. As we know, the value of time-unrelated attributes never change; thus the replacement of such attributes is unnecessary. If there are one hundred attributes associated with a private key, but only one of them is time-related, reconstructing the entire private key could be a huge waste of resource and bandwidth, resulting unnecessary communication overhead. It would be much more efficient if we could only update those time-related attributes and keep the rest time-unrelated attributes, when a new private key is requested.

2 Proposal: Partial Key Revocation with Dynamic Attributes

To save CPU resources, bandwidth and time, we avoid updating those attributes that stay unchanged. To achieve this, this paper proposes the concept of *attribute fading function*, making attributes independent and dynamic. With fading function, an attribute associated with a private key has its own expiration time. When an underlying property changes, the user requests a new attribute from the authority to represent his new property and the out-of-date attribute expires after a certain period of time. By this mean, a user can update partial attributes, rather than all of them, in one update – named *partial, dynamic key revocation*. When updating a private key, only expired time-related attributes are replaced by their latest value at user side. The time-related attribute is named dynamic attribute, while the time-unrelated attribute is static attribute. Fig. 2 shows the procedure of the proposed Dynamic Attribute-Based Encryption (DABE) system.

In DABE, there are many confusions regarding adequate usage of dynamic attributes, which include, but not limited to, concerns about multiple values of a specific dynamic attribute coexisting in the same private key, or the value of one dynamic attributes wrongly colliding the value of another dynamic attributes. Those concerns can screw up the underneath ABE. In order to leverage dynamic attributes in DABE, it is crucial to define a data format for dynamic attributes.

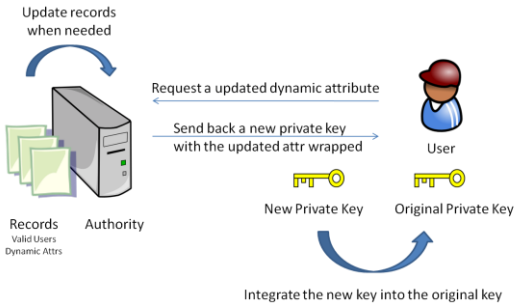

Fig. 1. The proposed Dynamic Attribute-Based Encryption scheme takes a fading function and performs dynamic key revocation.

A good format of dynamic attributes can clarify ambiguities, resulting in more efficient computation. We observe that, among all the input attributes, dynamic attributes appear in the form of "<attribute name> = <attribute value>". We reformat dynamic attributes so that they have the following form: "<attribute name>@<attribute value> = <expiration time>", making expiration time as part of a dynamic attribute. With this format, there are two values associated with a dynamic attribute: its attribute value, and its expiration time. This format, therefore, gives the authority more flexibility in creating private keys. Furthermore, with our new format, the concern that the value of one dynamic attribute accidentally equals to that of another dynamic attribute no longer exists. A dynamic attribute is represented only by its value without a specific name, which definitely give rise to the value collision problem between two different dynamic attributes, allowing users to decrypt the message they are not supposed to decrypt and, therefore, screw up ABE systems. However, with our new format, each dynamic attribute is specified by both its name and its value. For example, two dynamic attributes "city = Los Angeles" and "street = Los Angeles" will never be confused even if they are in the same private key, for their representations, with our new format, would be "city@Los Angeles" and "street@Los Angeles", respectively.

3 Implementation and Evaluation

DABE inherits all the functionalities of CP-ABE [1], which include Setup, Encrypt, Decrypt, but with one improved function KeyGen, and three new functions: AttrUpdate, KeyUpdate, and KeyIntegrate. The authority executes KeyGen, AttrUpdate, and KeyUpdate, whereas KeyIntegrate is run by the user.

We implement the proposed scheme on a testbed of Lenovo R60 with Ubuntu 11.04 and run experiments with scenarios having a user update one of dynamic attributes in his key per request. Then, we evaluate the performance of DABE by measuring the elapsed time when a user requests a new key from the authority. The box below lists notations used in our evaluation.

NA: Number of Attributes associated with a key ND: Number of Dynamic attributes associated with a key	TA: Time Authority takes to create a new key TC: Transmission time over secure 2Mbps channel TU: Time User takes to integrate keys

When a dynamic attributes expires, the authority of DABE sends out the latest update of that dynamic attribute to the user. In DABE, it takes around 2.25 seconds to create a new private key with the latest update of a requested dynamic attribute wrapped. That is, TA = 2.25s. The size of the one-attribute-wrapped private key is around 23767 bytes. We assume the secure channel over which the key is sent has speed 2Mbps, so the transmission time would roughly be (23767*8) / (2*10^6) = 0.095 seconds, i.e. TC = 0.095s. Lastly, it takes 0.02 seconds for the user to integrate the new key into the original key, i.e. TU = 0.02s. Thus, in total, it takes TA+TC+TU = 2.365s for a user to renew his/her private key, under the assumption that only one associated dynamic attribute gets updated. If there are N dynamic attributes needing updating, it would take 2.365*N seconds for the user to renew the private key.

Table 1. Profile of key creation.

	TA (s)	*Size (byte)*	*TC (s)*	*Ttotal (s)*
NA=10, ND=1	2.650	26048	0.104	2.754
NA=10, ND=2	4.821	49028	0.197	5.018
NA=10, ND=5	11.481	118896	0.475	11.956
NA=10, ND=8	18.203	188534	0.754	18.957
NA=20, ND=2	5.082	51900	0.208	5.290
NA=20, ND=4	9.495	98323	0.393	9.888
NA=20, ND=10	22.795	238005	0.952	23.747
NA=20, ND=2	36.278	377249	1.510	37.788

On the other hand, in CP-ABE, whenever an attribute needs updating, the authority has to recreate the whole private key for the user. The time it takes to create a private key depends on how many attributes associated with the key, i.e. the value of NA; in particular, it depends on how many dynamic attributes associated, i.e. the value of ND, as shown in Table 2. With the same value of NA, the bigger the value of ND is, the longer it takes to create a private key, and the larger its size would be. In CP-ABE, TU is always zero, because users do not integrate keys; they only accept an entire new private key when some dynamic attribute gets updated. The total time it takes to renew the key when a dynamic attribute gets updated is, as we can see from the table, is much longer than that in DABE, i.e. 2.365s. It thus demonstrates the effectiveness of DABE.

References

1. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-Policy Attribute-Based Encryption. In: IEEE Symposium on Security and Privacy (Oakland) (2007)
2. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute Based Encryption for Fine-Grained Access Conrol of Encrypted Data. In: ACM conference on Computer and Communications Security (2006)

