International Journal of Security and Its Applications
Vol.9, No.11 (2015), pp.23-34
http://dx.doi.org/10.14257/ijsia.2015.9.11.03

Secured Data Sharing based on Information Centric Trust in the Internet of Vehicles

Eun-Kyu Lee and Ki Young Lee*

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

Abstract

Attribute-Based Encryption provides an encrypted access control mechanism. A user applies and obtains a private key that is associated with various attributes of the applicant. A secret message is encrypted with an access control policy tree representing a logical combination of different attributes. Only the users whose attributes satisfy the policy tree can decrypt the message. There are many scenarios that require dynamic attributes. Whenever a dynamic attribute updates, a new key is regenerated and issued even if there are 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. To resolve the efficiency problem, this paper introduces 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. We show that the new concept improves the efficiency of the encryption scheme and helps us develop a new trust management method.

Keywords*: Security, Encryption, Attribute based encryption, Access control, Trust management, Internet of Things*

1. Introduction

Assume a scenario where a technical manager at a software company wants to announce some important information to her group, whose members include three development subgroups: three development engineers, five test engineers, two interns, and one program manager. They work in different offices, and each of them has different executive levels. The technical manager has different messages to different subgroups: to development subgroup, she needs to send out an internal deadline for current development of an undergoing project; to test subgroup, she needs to make a schedule for them to test one unreleased new product; to intern subgroup, she needs to assign to them some assistant tasks; to the program manager, she needs to set up a meeting with her to discuss next season product. Moreover, in each subgroup, the technical manager may need to contact some specific members for the projects for which only they are responsible. In addition, within this group, some development engineers may also want to contact some specific test engineers for their own testing plans. Some test engineers may also contact interns for more assistance.

The scenario is an example of one-to-many multicast group communication whose popularity has grown significantly with the increasing demand of group-based applications, such as online forums, pay per view channels, multimedia conferences, and various information dissemination services, such as news, weather, or share prices updates. In some scenarios, such as the technical manager example, selective secure group communication is required, in which a group message is encrypted under some secure

* Ki Young Lee is the corresponding author.

ISSN: 1738-9976 IJSIA

policy and sent to every member, but only qualified members can decrypt and read the message. The qualification of a user is determined by a user-specific key, which is created by an authority unit of the group and gets distributed to the user. Hence, secure selective data dissemination involves message encryption, user key generation, and key distribution, and, thus, it requires the following elements: an encryption scheme, management of key material, and security policies. Those elements enable group communication to well deploy secure selective data dissemination, which not only works in a conventional local network, but also performs well in a mobile environment, where confidentiality, integrity, authenticity, and data security are highly concerned.

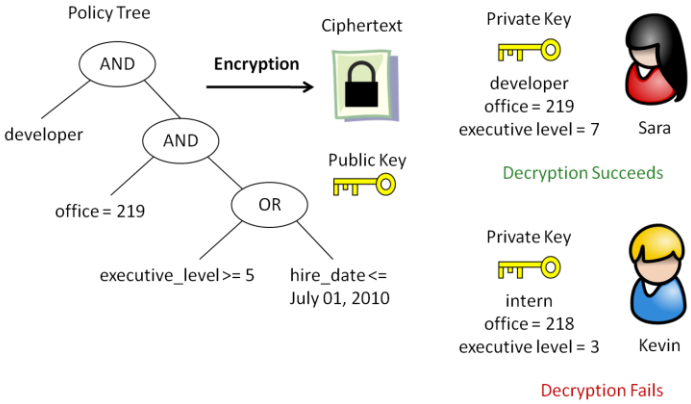

Figure 1. Policy Tree and Attribute-Associated Private Keys in CP-ABE

From previous description, we know that the qualification of a user to decrypt and read an encrypted message is determined by both the secure policy under which the message is encrypted and the properties of the user. Recent Attribute-Based Encryption (ABE) scheme makes it possible to dynamically reassign user-specific keys when requirements and conditions change [2, 3]. Here we use the manager example to explain the concept of ABE: say the internal deadline can read by only the development engineers who work in office 219; moreover, they should either be at executive level 5 or higher, or be hired no later than July 01, 2010. Each of the properties mentioned is an attribute; all of them define a security policy for the group message, which contains the internal deadline. Then, along with a public key created by the system, the secret message is encrypted under the secure policy, which is represented by an access control policy tree, consisting of the attributes logically combined as follows: "developer AND office = 219 AND (executive_level >= 5 OR hire_date <= July 01, 2010)", as shown in Figure 1. In this example, the technical manager is the encryptor, who encrypts the internal deadline to be a selective secret group message. Any group member is a potential decryptor. In order to read the message, they should apply to an authority (*i.e.* Key Master) for their own private key, which is associated with various properties (*i.e.* attributes) of the applicant, such as title (developer, tester, intern, *etc.*), office location, hire date, or executive level. Only the members whose attributes satisfy the policy tree can decrypt the message and read it.

The rest of the paper is organized as follows: Section 2 gives the essential background on ABE and its variants. It also identifies potential inefficiencies of the current ABE systems. Section 3 introduces the concept and design of a *dynamic* attribute-based encryption, DABE, a scheme where attributes can be added or modified upon expiration. Section 4 describes the implementation of DABE. Section 5 lists possible scenarios, the corresponding experiments and results. Section 6 concludes this paper.

2. Preliminary

2.1. Attribute-Based Encryption (ABE)

An Attribute Based Encryption (ABE) is a new mechanism for encrypted access control [2, 3]. There exists an authority to handle key requests from group members, create private keys based on their own attributes, and distribute the resultant keys to applicants. The authority creates a public key and makes it available to the entire group in advance. Each member can be either an encryptor or a decryptor; in either case, the public key is needed, whereas in decryption, the decryptor also needs his/her own private key to decrypt the encrypted message. There are two variants of ABE: Key-Policy Attribute-Based Encryption (KP-ABE) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [1], which will be described separately in the followings.

2.1.1. Key-policy Attribute-based Encryption

Sahai *et al.*, introduces KP-ABE as a public key cryptography primitive for one-to-many communication communications [2]. In KP-ABE, there exists a universe of attributes U. Each attribute in U constructs one component of public key. The message is associated with a set of attributes R, each of which, therefore, corresponds to one specific component in the public key. The encryptor associates R to the message by encrypting it with the corresponding public key components. Each group member is assigned an access tree structure, *i.e.*, a policy tree, where interior nodes are logical gates and leaf nodes are associated with the member's own attributes. When a group member requests the private key, the key would reflect the assigned access tree so that the member could decrypt the message if and only if the set of attributes R satisfies the access tree in the member's private key.

2.1.2. Ciphertext-policy Attribute-based Encryption

A CP-ABE is illustrated in Figure 1. The manager example is based on CP-ABE, which, to some extent, could be seen as an extension of conventional PKI to groups, but removes the overhead resulting from too many stored public keys in PKI. In CP-ABE, similarly, the authority generates one public key for the group and private keys for all group members. However, unlike KP-ABE, there is no universe set of attributes, and, thus, the public key is not associated with any attribute. As described in the manager example, each group member has his/her own set of attributes, which are associated with the resultant private key. The message is encrypted by an encryptor under an access policy. Only the members whose attributes associated with their private keys satisfy the policy tree can decrypt the ciphertext.

Compared to KP-ABE, which involves a large amount of attribute duplicates both in private keys and ciphertext, CP-ABE is more efficient both in time and space. Moreover, in general, CP-ABE is easier and more straightforward to be understood and applied. Thus, in this paper, we choose CP-ABE as the underneath ABE system, and we hope to invent new features to build an improved attribute-based encryption scheme on top of conventional CP-ABE.

2.2. Motivation: Dynamic Attribute and Information Centric Trust

In current CP-ABE, attributes are assumed to be static, *i.e.*, their values never change, such as hire dates of the engineers in our manager example. However, it is very possible that the value of an attribute gets updated over time. Recall the manager example: the executive level of an engineer may change as time goes, depending on his/her performance and contribution; the office where an engineer works may also change from time to time. In this way, we can see that there exist two types of attributes: a time-

unrelated attribute and a time-related attribute. The value of a time-unrelated attribute, we name it *static attribute*, does not change since its initial creation, whereas the value of time-related attribute, named *dynamic attribute*, changes as time goes or events happen. 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 dynamic attributes associated within a private key by simply reconstructing the entire private key, which causes replacement of both dynamic attributes and static attributes. As we know, the value of static 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 dynamic attributes and keep the rest time-unrelated attributes, when a new private key is requested. We call this research challenge a *partial update of a dynamic attribute*.

While the separation of dynamic attributes from static attributes still remains the technical challenge, it enables a few advanced security systems. One example is in a trust management, and this paper introduces a concept of *Information-Centric Trust (ICT)* that provide an adaptive and proactive trust management solution in multi-coalition, mobile scenarios such as vehicular communications among connected cars. Attributes in ICT identify a group of entities (*e.g.*, taxis associated with a company, police cars in a city), a type of events (*e.g.*, accidents, congestions), or the property of events (location-based services, road traffic updates). They can be further classified as dynamic attributes and static attributes, depending on whether the attributes change frequently or not. We note that the notion of ICT is not new, but have been studied in literatures.

Hong *et al.*, [4] introduced the concept of Situation Aware Trust that established a Data-Centric Trust [8] by using the CP-ABE scheme. In these theoretical researches, vehicles fulfill a set of attributes form a policy group. A policy group is specified by the information source and is organized automatically without relying on a trust party to manage the group. For instance, a company A's taxi driver can broadcast a message encrypted with policy tree "(company A) AND (Westwood Blvd.) AND (10-11am)" to tell his colleagues that conventioneers will be waiting for pick up at Westwood Blvd. hotels. The message may be broadcast to a very large area of the city. However, only company A's taxicabs that have been *certified* to be near Westwood Blvd. in the time window 10-11am can decrypt the message. Thus competitive advantage is ensured. At the same time, company A's taxi drivers likely to work the morning shift in remote areas will not be notified and will not waste time and gas unnecessarily. The broadcast group boundary is not clearly defined yet, but whoever satisfies the policy tree can join the group. This feature allows users in ICT set up trust proactively.

3. Fundamentals of Dynamic Attribute Based Encryption

As an enabler of the attribute-level operation, this paper introduces the concept of Dynamic Attribute-Based Encryption (DABE) that tends to solve the inefficiency of CP-ABE. When updating a private key, only expired (time-related) dynamic attributes are replaced by their latest value at user side. Figure 2 shows the procedure of the DABE system. However, the authors only discuss the possibility of using dynamic attributes while remaining its design and realization untouched. This section scrutinizes DABE from algorithmic perspective and designs a brand-new encryption system.

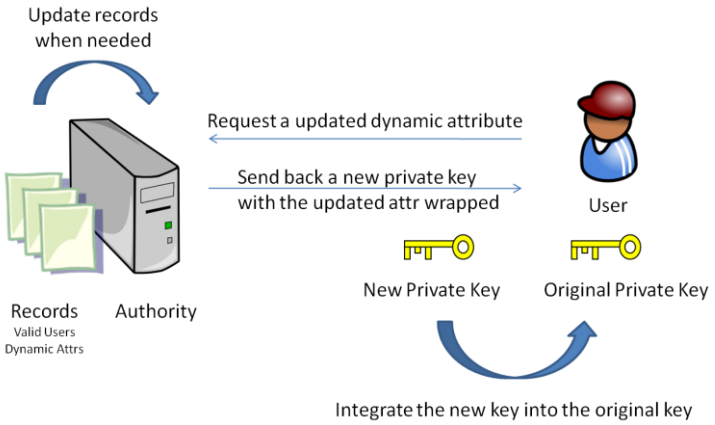

Figure 2. Illustration of Dynamic Attribute-Based Encryption (DABE)

3.1. DABE: Concept and Concerns

DABE is to update expired dynamic attributes in an existing private key without modifying the rest of attributes associated with the key, when a user requests to renew the key. This concept brings up three concerns: key recipient recognition, valid expiration time of dynamic attributes, and proper format of dynamic attributes.

3.1.1. Key Recipient Recognition: The authority in CP-ABE computes a private key (secret key or SK) with the following formula:

$$SK = (D = g^{(\alpha + r)/\beta}, \forall j \in S: D_j = g^r \cdot H(j)^{rj}, D_j' = g^{rj}),$$

where S is the set of attributes owned by key applicant. A private key SK consists of $|S| + 1$ components: D, D_1, D_2, ... $D_{|S|}$. D corresponds to the key recipient and each D_j component corresponds to an attribute in S. A random number is assigned to each component, *i.e.* r, r_1, r_2, ... $r_{|S|}$, and is used to identify the corresponding component. Thus, each r_j identifies one attribute, whereas r identifies the key recipient.

In CP-ABE, the random numbers are hidden from the key recipient, so that she cannot launch a collusion attack. Each time a group member requests a new private key, the authority in CP-ABE not only generates the entire private key by reconstructing each D_j with the corresponding attribute, but it also assigns a new random number, say, r', to construct D in the new key. When a new private key issued to a recipient, a new private key replaces the old one. Since a user has one key at a time, the new key replaces the old one, and the new random number r' uniquely identifies the key recipient. Therefore, the authority is not required to memorize the original identity (*i.e.* the random number r) of the recipient.

In DABE, on the other hand, the authority performs an attribute-level operation. That is, the authority sends out the updated version of requested dynamic attributes to a key recipient to update the D_j's related dynamic attributes. In this case, the key recipient already has her own private key at hand and keeps receiving updated dynamic attributes, which will be integrated into the original private key. Thus, the identity of each group member is fixed in DABE. Furthermore, the authority has to memorize the identity of each user, so that when a user requests for updated dynamic attributes, the authority can correctly recognize the user. In DABE, the authority is required to store the identity of each group member. How to efficiently store and retrieve identity information is therefore a concern.

3.1.2. Valid expiration time: The main difference between a static attribute and a dynamic attribute is that the former holds a fixed value, but the latter needs to update its value. A dynamic attribute is a unique feature in DABE, and, therefore, when to update the attributes becomes a new concern. A unique fading function is used for each dynamic attribute to address the concern. The fading function determines the expiration time of the corresponding dynamic attribute, under the assumption that every dynamic attribute periodically expires and gets updated. However, the assumption does not hold in the real world, for there is no guarantee that a dynamic attribute would periodically expire; rather, it would be much more reasonable to have dynamic attributes expire and updated when needed. Thus, the next concern is about how to reasonably and correctly determine valid expiration time of a dynamic attribute.

3.1.3. Proper format of dynamic attributes: In DABE, there may exist 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. A proper format not only enhances computation efficiency but also clarifies many ambiguities of DABE. In short, how to properly define a format for dynamic attributes is the last concern that this paper considers.

3.2. DABE Design

As an improvement, DABE is built on top of CP-ABE. Thus, DABE design involves modifications to conventional CP-ABE systems. Those modifications can be put into three categories: dynamic attributes, records on authority, and key management.

3.2.1. Dynamic attributes: As mentioned in Section 3.1.3, 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.

Another improvement is that we change the way of assigning expiration time to dynamic attributes. A unique fading function for each dynamic attribute is proposed so that each dynamic attribute has a different period, upon which a new expiration time is assigned. For example, dynamic attribute A expires every 5 minutes, dynamic attribute B expires every 1 hour, and dynamic attribute C expires every 1 week. However, in real world, such fading functions do not exist. As described in Section 3.1.2, the assumption that all dynamic attributes get updated periodically does not hold; it is much more reasonable to update dynamic attributes when needed. Moreover, an attribute-specific

fading function is too abstract to be implemented, not to mention that it is proposed on an unreasonable basis. Thus, we change the way of assigning expiration time by enabling the authority to update the expiration time of each dynamic attribute manually when needed, under the assumption that the authority knows when to update. We believe it is more reasonable in real-time situation, making it possible to implement expiration of dynamic attribute.

3.2.2. Records on authority: As mentioned in Section 3.1.1, in DABE, the authority has to store the identity of each user, *i.e.* user-specific random number *r*. Under the assumption that the authority has enough space to store *r*, efficient retrieval of data becomes the most important concern. In our design, we choose hash table as the data structure to store user identity. The structure is showed in Figure 3. As we can see, in user-identity hash table, each record consists of a key-value pair, where key is the name of a user's private key, and the value is the user's identity. With this information, the authority can easily recognize the user from its request, and correctly send out the required updated dynamic attribute to the user, as shown in Figure 2.

In addition, the authority also needs to store the list of available dynamic attributes along with their expiration time. Each record is again a key-value pair; the key is of the form <attribute name>@<attribute value>, and the value is expiration time, as shown in Figure 4. With these dynamic attributes stored, the authority can easily retrieve the latest expiration and attribute value from the dynamic-attribute hash table when a user requests the update.

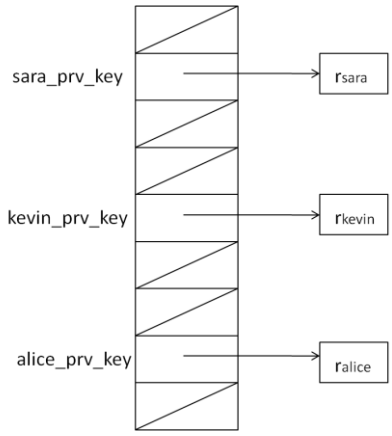

Figure 3. User-Identity Hash Table on Authority

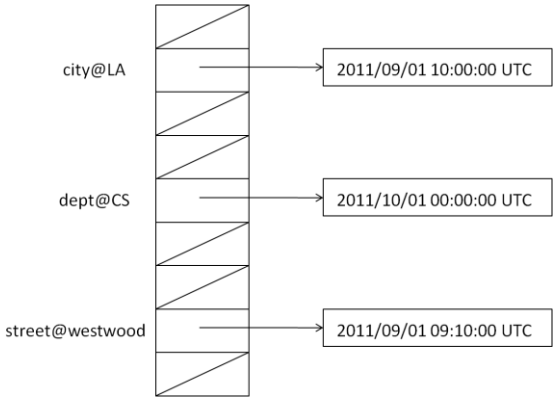

Figure 4. Dynamic-Attribute Hash Table on Authority

3.2.3. Key management: As shown in Figure 2, when a user requests the latest update of a dynamic attribute, rather than send out the attribute to the user, the authority creates a new private key with only that requested dynamic attribute wrapped, and sends out the new key to the user. When the user receives the new key, he/she replaces the old version of the requested dynamic attribute in the original private key by the latest update wrapped in the new key. After that, the user keeps the original private key, which has been updated, and throws away the new key. In our design, we call this process *key integration by replacement*.

The modifications mentioned in previous subsections are based on three assumptions. First, we assume that it is a user's responsibility to actively request an update of some dynamic attribute from the authority. Second, we assume that an encryptor checks with the authority to obtain the knowledge of available dynamic attributes before encrypting a secret message. Third, we assume that a user requests the update of a dynamic attribute only if this dynamic attribute has existed in the original private key. In other words, a user

does not request the update of a dynamic attribute if that dynamic attribute is not in the original private key.

4. DABE Implementation

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. Our construction of functions KeyGen, KeyUpdate, AttrUpdate, and KeyIntegrate are described in following subsections.

4.1. KeyGen

As in CP-ABE, the authority processes key requests from users, and creates private keys for them, based on their own attributes. However, with dynamic attributes, the implementation of *KeyGen* is different. The authority takes as arguments a set of attributes along with the public key and the master key created in Setup, and creates a private key for the key applicant. The set of attributes are those owned by the applicant. Thus, for each of the attributes provided, the authority will check whether it is a dynamic attribute or a static attribute. If the current attribute is static, the authority directly constructs the private key with the current attribute. Otherwise, it is a dynamic attribute, for which the authority would check if it has been added to the dynamic-attribute hash table. If so, then the authority would retrieve the latest update of the attribute to construct the private key; if not, then the authority would add this dynamic attribute into dynamic-attribute hash table, assign a default expiration time (*e.g.* 1 month), and use this information to construct the private key. Construction of the private key continues until all the provided attributes are processed and added as part of the key. After that, the authority stores the identity of the key applicant into the user-identity hash table, and sends the resultant private key through a secure channel to the applicant. The procedure is illustrated in Figure 5.

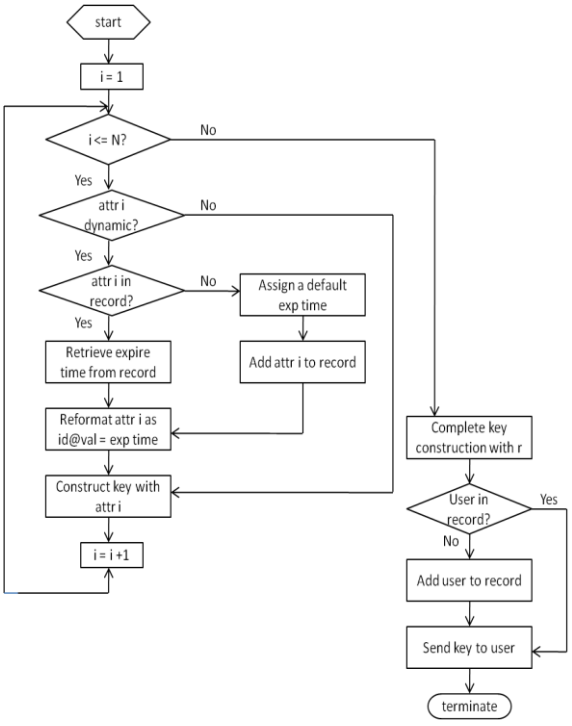

Figure 5. Flowchart of KeyGen in DABE

4.2. AttrUpdate

Using the *AttrUpdate* function, the authority is able to update the dynamic attributes stored. The authority takes the name of a dynamic attribute, its value, and an expiration time as arguments. It checks if the dynamic attribute already exists in the dynamic-attribute hash table. If so, the authority would update the existing record by replacing the old expiration time with the input expiration time. If not, the authority would store the dynamic attribute and the input expiration time as a new record in dynamic-attribute hash table. The procedure is illustrated in Figure 6.

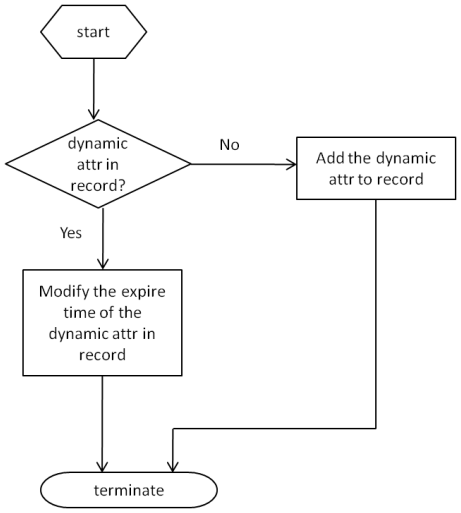

Figure 6. Flowchart of *AttrUpdate* in DABE

4.3. KeyUpdate

Using the *KeyUpdate* function, the authority is able to send out the update of a dynamic attribute to a user upon his/her request. The user and the requested dynamic attribute are specified in argument list. The authority first checks if the user identity has been stored in user-identity hash table, which means that the user must be a group member and obtain a private key before requesting any attribute update. If not, the authority will refuse the request. If so, then the authority further checks if this requested dynamic attribute exists in dynamic-attribute hash table. If not, the authority will insert a new record consisting of the attribute name, value, and a default expiration time into dynamic-attribute hash table, and wrap it in a new private key. If the attribute does exist, the authority retrieves the latest update from the hash table and creates a new key with this information wrapped. Finally, the authority sends out the update of the requested dynamic attribute in the form of a private key over a secure channel to the user. The procedure is illustrated in Figure 7.

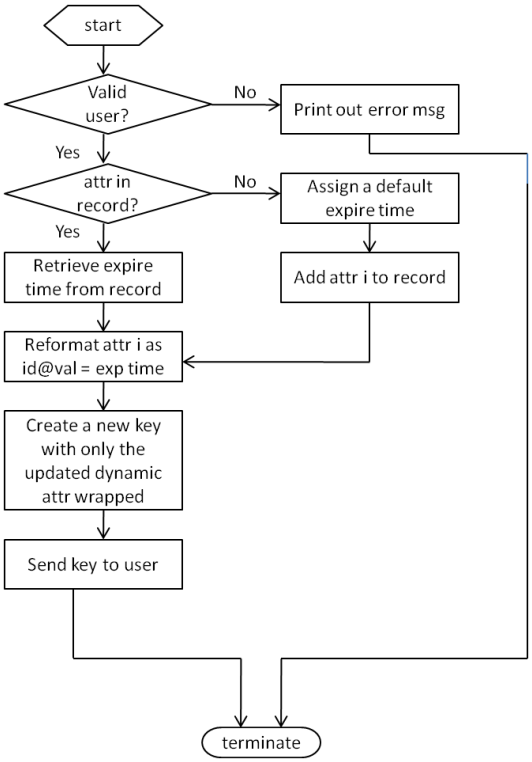

Figure 7. Flowchart of *KeyUpdate*

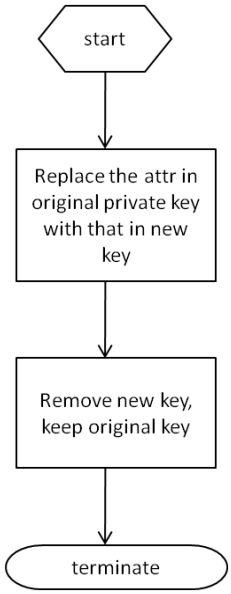

Figure 8. Flowchart of *KeyIntegrate*

4.3. KeyIntegrate

When a user receives a new private key that contains the update of a dynamic attribute he/she requests, the user needs to integrate the new key and his/her original key. The strategy to integrate two keys is replacement, by which the user replaces the expired dynamic attribute in the original key by the updated version in the new key. After that, the user keeps the updated original private key and throws away the new key. The procedure is illustrated in Figure 8.

5. Experimentation

In this section, we evaluate the performance of DABE by measuring the elapsed time when a user requests a new key from the authority. We experiment with different parameters on our metrics to demonstrate the effectiveness of DABE.

5.1. Preliminary

We consider an Internet of Vehicles (IoV) scenario, which uses moving cars as nodes in a network to create a mobile network [7]. The IoV turns every participating car into a wireless router or node, allowing cars approximately 100 to 300 meters of each other to connect and, in turn, create a network with a wide range. In this scenario, it would be good to adopt secure selective one-to-many group communication in the IoV, which needs ABE to handle encryption of group messages as well as the generation and distribution of group keys. Since users in this scenario are moving cars, attributes related to location are inherently dynamic, such as current street, current city, current zip code, or current state. Other dynamic attributes might be membership of AAA, membership of UHaul, *etc*. Among those dynamic attributes, by nature, location-related attributes should update more frequently than membership-related ones. We assume that the authority unit is powerful enough to determine the expiration time of each dynamic attribute. A moving car can also have static attributes, such as plate number, driver's license number, *etc.*

During our experiment, we measure the time it takes for a user to request a new key from the authority, both in CP-ABE and DABE. The elapsed time consists of three parts: the time the authority takes to create a new key, the transmission time over a secure channel, and the time the user takes to own a new key. Our parameters include the number of all associated attributes in a private key, and the number of associated dynamic attributes in the same private key. The notations are listed in Table 1.

Table 1. Notations Used in Key Computation

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

Note the extreme case where all the dynamic attributes get updated. In this case, the time to renew a key in DABE will be 2.365**ND* seconds, which is almost equal to the time to renew a key in CP-ABE. This phenomenon shows that the time it takes to embed dynamic attributes in a private key dominates the creation of the entire private key. In other words, the advantage of DABE exists in the situation where only some of the dynamic attributes get updated. However, even in the extreme case where all the dynamic attributes are updated, DABE does not lose – rather, it performs as well as the conventional CP-ABE.

6. Conclusion

Our work reinterprets the concept of DABE and constructs the system with our innovate design. Among all the variants of attribute-based encryption (ABE) systems, we choose CP-ABE as the basis ABE system, on top of which DABE is built. In this new system, attributes are classified into two categories: static and dynamic. The categorization of attributes improves efficiency of the generation and renewal of private key. In order to adequately adopt dynamic attributes, the authority keeps record of valid user identity and available dynamic attribute along with their expiration times. The authority also has the power to update the dynamic attributes and set their latest expiration time. When a user requests the update of some dynamic attribute, the authority will process this request by sending out the update wrapped in a new private key. And then, on the client side, the user will be able to integrate his/her original key and the new key. In the IoV scenario, our experimentation result clearly shows that the total time it takes to update a private key in DABE is much less than in conventional CP-ABE. The effectiveness of DABE is demonstrated.

The future work includes new dynamic attribute request, updates of multiple dynamic attributes in a new private key, and more efficient encryption scheme. In our current construction, we assume that a user requests an update of some dynamic attribute only if the attribute has existed in his/her original private key. However, it is possible for a user to request the latest update of a new dynamic attribute that is not yet in the original private key. Besides, in our current construction, a new private key is created with only one dynamic attribute's latest update wrapped. Although, with this construction, DABE performs much better than CP-ABE in most cases, it would be even much more efficient for a user to request updates of multiple dynamic attributes at once, with them all embedded in a new private key. At last, we want to further improve the encryption functionality in conventional CP-ABE to make it more suitable and efficient in DABE.

References

[1] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption", "Proceedings of IEEE Symposium on Security and Privacy" (Oakland), **(2007)**.
[2] A. Sahai and B. Waters, "Fuzzy Identity Based Encryption", "Springer Advances in Cryptology – Eurocrypt", vol. 3494, pp. 457-473, **(2005)**.
[3] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute Based Encryption for Fine-Grained Access Control of Encrypted Data", "Proceedings of ACM conference on Computer and Communications Security" (ACM CCS), **(2006).**
[4] X. Hong, D. Huang, M. Gerla, and Z. Cao, "SAT: Building New Trust Architecture for Vehicular Networks", "Proceedings of ACM International Workshop on Mobility in the Evolving Internet Architecture " (MobiArch), Seattle, USA, August 22, **(2008)**.
[5] Z. Zhou and D. Huang, "An Optimal Key Distribution Scheme for Secure Multicast Group Communication. Proceedings of IEEE Infocom", **(2010).**
[6] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure Attribute-Based Systems. Proceedings of ACM Conference on Computer and Communications Security ", (ACM CCS), **(2006**).
[7] E. Lee, E.-K. Lee , S. Oh, and M. Gerla, "Vehicular Cloud Networking: Architecture and Design Principles", "IEEE Communications Magazine", vol. 52, no. 2, **(2014).**
[8] M. Raya, P. Papadimitratos, V. Gligor, and J. Hubaux. Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks. In Proceedings of IEEE Infocom, **(2008).**

Authors

Eun-Kyu Lee, Dr. Lee is an assistant professor in the Department of Information and Telecomm. Engineering of Incheon National University (INU). Before joining INU, he had worked for Symantec Research Labs (Los Angeles, USA) and Electronics and Telecommunications Research Institute (Daejeon, Korea) for several years. He has been involved in many research projects in the fields of smart grid, smart building, electric vehicle network, location-based services, and Telematics. He received M.S. and Ph.D. in Computer Science from the University of California, Los Angeles in and 2014, respectively. His research interest includes Internet of Things, cyber-physical systems, wireless networks, cybersecurity and privacy, middleware, and pervasive mobile computing.

Ki Young Lee, Dr. Lee received the B.S. and M.Eng. Degrees in Electrical Engineering from Yonsei University, Seoul, Korea in 1982 and 1984, respectively. And he received M.S. (1987) from the University of Colorado, Boulder and the Ph.D. (1993) from the University of Alabama in Electrical & Computer Engineering. Since 1994, he has been a professor in the Department of Information and Telecommunication Engineering at Incheon National University. His research interests include Internet traffic control and protocols, user authentication protocols, and security mechanism of IOT environment.

.

