Flexible Hierarchical Key Assignment Scheme with Time-Based Assured Deletion for Cloud Storage

Ping-Kun Hsu[1], Mu-Ting Lin[1], and Iuon-Chang Lin[1,2(✉)]

[1] Department of Management Information Systems, National Chung Hsing University, 250 Kuo-Kuang Road, Taichung 402, Taiwan
iclin@dragon.nchu.edu.tw
[2] Department of Photonics and Communication Engineering, Asia University, Taichung, Taiwan

Abstract. Everyone now can store their files to the cloud, which makes life more convenience and don't worry about losing the storage device. However, the files store in the cloud is managed by someone may not trustable become a security concern. One of the solutions is encrypting the file and doing assured deletion on it. This paper proposed a time-based assured deletion in a flexible hierarchy structure. Clients will be distributed one derivation key depend on which class and when to over. Then client can use the derivation key to derivate the time-bound secret key to encrypt files. While the predetermined time passing, the secret key will be deleted and the file is unrecoverable. The proposed scheme provides the guarantee of files in the cloud, and also applies to the flexible hierarchy structure which may suitable in the organization.

Keywords: Time-bound · Assured deletion · Cloud storage · Hierarchy key assignment

1 Introduction

In recent years, cloud computing [13, 14] has become an attractive computing infrastructure. One of the attractive reasons is providing the unlimited storage space. For example, the Amazon Simple Storage Service [12] and Dropbox [11] are famous cloud storage providers. Individuals or organizations often require storing their data to avoid data loss, hardware or software failure and inevitable disasters. Instead of buying the storage media to backup these data, individuals or organizations can just outsource their data to the cloud storage provider, which provides storage resources to the data backup.

However, the security concerns of sensitive data become relevant as storing them to the third party. Users are unaware how the cloud service provider mange the information. In addition, it is no idea that whether the employee in the cloud storage provider has accessed or leakage the information. For example, the personal health records make patients can manage their health information in the cloud [15]. The information is highly sensitive. Intentionally or unintended leakage the records may threaten the patient's privacy or right and lead to unexpected consequences.

C.-Y. Chang et al. (Eds.): ICS 2018, CCIS 1013, pp. 599–607, 2019.
https://doi.org/10.1007/978-981-13-9190-3_66

This paper proposed a time-based assured deletion for a flexible hierarchical key management scheme to provide the guarantees of the sensitive data in the cloud. The time-based assured deletion means the secret key has a life cycle, which will be deleted after the predetermined time passes. Then the file encrypted with the time-bound secret key can't recover anymore, it is called assured deletion. So even someone owns the file, without the time-bound secret key, he can't access. Besides, for the organization, this scheme also introduces a more flexible hierarchy to implement. The flexible hierarchy is like the Fig. 1, the user in class C_1 has the authority to access information in C_2 and C_4, but access C_3 is not allowed. The user in class C_2 can access C_3 and C_4, and the user in C_4 can also access C_2. It is not like traditional hierarchy that higher class can access the lower class. This flexible hierarchy is with anti-symmetric and transitive exceptions. The anti-symmetric policy means that C_i could access C_j and C_j can also access C_i, but C_i and C_j are two different classes. For example, the classes C_2 and C_4 are in this case as Fig. 1 show. The transitive exceptions means that C_i can access C_j and C_j can access C_k, but C_i cannot access C_k. For example, the classes C_1, C_2 and C_3 are in this case as Fig. 1 show. This hierarchy structure is more flexible and practical than traditional hierarchy structure in the real world.

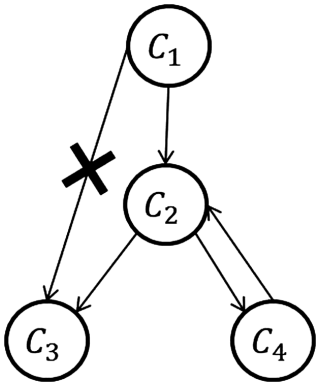

Fig. 1. An example of flexible hierarchy structure with anti-symmetric and transitive exceptions.

The organization of this paper is as follow. In the Sect. 2, related work briefly introduces the method of assured deletion and time-bound hierarchical key assignment. Section 3 introduces the proposed scheme applying on the time-based assured deletion to the flexible hierarchy structure in cloud. There has an example in the Sect. 4 to illustrate the proposed scheme. The security analysis of active and deleted files is provided in the Sect. 5. Finally, the conclusions is presented in the Sect. 6 of this paper.

2 Related Work

This section reviews other related works on assured deletion and time-bound hierarchical key assignment.

2.1 Assured Deletion

A file is requested to be deleted, but someone owns the copy of file and can discover it, making the file in danger. So assured deletion becomes an important guarantee of the file. There are two types of assured deletion. One is policy-based assured deletion [1]. This method means each file will associate with a file access policy. Each policy is with a control key, which is a public-private key pair. The control key is maintained by the key manager, it is a sever which responsible to cryptographic key management. Supposed there has a file associated with a policy, and this file will encrypt with a data key, which is symmetric-key encryption. This data key will further encrypt with the control key corresponding to the policy. When the policy is deleted, the corresponding control key will be revoked from the key manager. Therefore, when the policy with the file no longer exists, the data key and the encrypted file will not be recovered without the control key. It achieves the assured deletion. In the recent work of Tang et al., they extend the assured deletion to cloud backup system with version control [2] and further improved by Tezuka et al. [3]. The system provides the version control backup design to eliminate the storage of redundant data. It also allows the assured deletion that client can know which version or file in the cloud be assured deletion. While other versions or files that share the same data of the deleted one will remain unaffected. In some way, it is a good application to implement assured deletion to the file in the cloud.

The other type of assured deletion is time-based deletion. It is first introduces in [4]. Time-based deletion means the file can be securely deleted and inaccessible after a specific duration. The main idea in [4] is a file encrypted with a data key by the owner, and this data key will further encrypt with a control key by a key manager. The control key is time-limited, meaning it will be removed by the key manager when the predefined time is reached. Without the control key, the data key and the file remain encrypted. Even the cloud storage provider owns the copy of file, it still unrecoverable.

2.2 Time-Bound Hierarchical Key Assignment

To manage access in a hierarchy, the easiest way is the authorized users have the entire successor's security clearance class secret keys. In this way, the key management raised problem in the multilevel security. Akl and Taylor [5] first introduces the concept of super-key, and it also becomes the basis of lots of research [6–8, 10]. They use the top-down method. In their method, the trusted third party will assign each class a prime, secret key and public parameter. If $C_j < C_i$, the user in class C_i can derivate the secret key in C_j with his secret key and public parameters. However, this method needs large amount of storage to store the public parameters. Because the public parameter in class C_j is the product of the public prime of C_i. In 2003, Lin et al. proposed a new key assignment in a more flexible hierarchy [6]. The storage of public parameter is smaller than [5], because it is a single prime in each class.

In 2002, Tzeng [8] proposed a time-bound cryptographic key assignment. It is an extension of [5] with the concept of time. The scheme could apply on secure broadcasting and cryptographic key backup. In the scheme of Bertino et al. [9] proposed, they showed that Tzeng's scheme is applicable to secure broadcasting of XML documents. Besides, their scheme has based on the tamper-resistant to store the

information; it greatly reduces the computation load and storage. In 2005, Yeh [10] also proposed a time-bound hierarchical key assignment, which is based on the RSA cryptographic system. Yeh's scheme is easy to understand and can apply to electronic article subscription system.

This paper is adopted by Lin et al.'s hierarchical key assignment and the tamper-resistant device in Bertino et al.'s scheme. In the Bertino et al.'s scheme, the device is tamper resistant and no one can recover the value or change the clock in it, so it also becomes the basis of this paper. The proposed scheme is aimed to apply time-based assured deletion to protect the file store in the cloud from someone who may discover it even had deleted.

3 The Time-Based Assured Deletion with Flexible Hierarchical Key Management Scheme

This section will introduce the key management with assured deletion in a hierarchy for a more flexible access control policy. In this scheme, the environment is based on the same start time, but the end time is depending on client. Besides, we adopt the tamper-resistant device which can prevent anyone to discover any information in it.

There have three roles in our scheme. One is Central Authority (CA). It is a third party that can be trusted. CA is responsible for generating parameters and issuing tamper-resistant device to clients. The tamper-resistant device includes a derivation key which can derive the secret key to encrypt or decrypt files for a time period. The secret key is based on a symmetric cryptosystem, like DES or AES. Another role is client. After receiving the device that CA gave, the client can do encryption or decryption of files, then save it in the cloud storage. Besides, if the client has authorized, it can access files in other classes. The other is cloud, the storage service provider. It is assumed that there have attackers in the cloud try to discover the file, whether it is deleted or not. The details of this scheme are described as follow:

3.1 Initiation

CA randomly chooses two large primes, p and q. Both of them must keep secret. Then, CA computes the product n, such that $n = p \times q$. CA also chooses another parameter a, which is relatively prime to n and the range is between 2 and $n - 1$.

For every class $\{C_1, C_2, \ldots, C_i\}$, CA chooses a set of distinct primes $\{e_1, e_2, \ldots, e_i\}$. Each e_i is relatively prime to $\phi(n)$, such as $\gcd(\phi(n), e_i) = 1$, and the range is $1 < e_i < \phi(n)$. For every time period, CA also chooses a set of distinct primes $\{g_1, g_2, \ldots, g_z\}$. Assumed the maximum time period is z, and it's an integer. For example, if the time period means one month, so $z = 12$ represents one year. Note that although the system is beginning at 1 and ending at z, it does not mean the system is constraint on this period of time. In addition, each g_z is relative prime to $\phi(n)$, and the range is $1 < g_z < \phi(n)$.

Next, CA will compute $\{d_1, d_2, \ldots, d_i\}$ and $\{h_1, h_2, \ldots, h_z\}$ for client class and time period respectively. Each d_i and h_z is multiplicative inverse of e_i and g_z. For example, $e_i d_i \equiv 1 \, mod \, \phi(n)$.

3.2 Key Assignment

In the key assignment phase, CA creates derivation key for every client in the classes $\{C_1, C_2, \ldots, C_i\}$. The derivation key is as follow:

$$DK_{i,z} = a^{\prod_{C_j \le C_i} d_i \prod_{h_{StartTime} \le h_{EndDate}} h_z} mod\ n \tag{1}$$

where *StartTime* means the start time of the system and *EndDate* means the time client wants to over, both of them assumed to be an integer. $C_j \le C_i$ means the client in the class C_i has authority to access the files in C_j.

Then, CA issues the client a tamper-resistant device, which storing $DK_{i,z}$ and the parameters $\{n, e_j, \ldots, e_i, g_{StartTime}, \ldots, g_{EndDate}\}$. There has a secure clock embedded in this device to keep track of the correct and current time. Because the device is tamper resistant, no one can recover the parameter or change the time of secure clock.

3.3 Key Derivation

The client in class C_i can use the derivation key $DK_{i,z}$ and parameters in the tamper-resistant device to derivate the secret key to encrypt files in its class or access the other classes which has authority. The process of derivation of secret key to encrypt files in client's class is as follow:

$$\begin{aligned} SK_{i,t} &= \left(DK_{i,z}\right)^{\prod_{C_k < C_i, k \ne i} e_k \prod_{g_t \le g_{EndDate}} g_l} mod\ n \\ &= \left(a^{\prod_{C_j \le C_i} d_i \prod_{h_{StartTime} \le h_{EndDate}} h_z}\right)^{\prod_{C_k < C_i, k \ne i} e_k \prod_{g_t \le g_{EndDate}} g_l} mod\ n \\ &= \left(a^{d_i}\right)^{\prod_{h_{StartTime} < h_t} h_l} mod\ n, \end{aligned} \tag{2}$$

where t means the deletion time of this file.

If the client in the class C_i wants to access files in the class C_j, the process of derivation of the secret key of class C_j is as follow:

$$SK_{j,t} = \left(DK_{i,z}\right)^{\prod_{C_k < C_j, k \ne j} e_k \prod_{g_t \le g_{EndDate}} g_l} mod\ n \tag{3}$$

After derivation, it doesn't have to store the secret key. The derivation key $DK_{i,z}$ can derivate it, if the client is in the right authority and timing.

The process of encryption and decryption of files in the Tang et al. scheme [1] needs to interact with key managers many times. If there are many clients in the system, the performance may be inefficient. Their scheme also needs a large amount of storage to store the private key created by every policies. In contrast, the interaction with CA in this scheme only at key assignment phase and when the time passes.

3.4 Assured Deletion

Client in this system can set the deletion time for files to do assure deletion, avoiding someone in the cloud could access even it had deleted. Some parameters in the tamper-resistant will be deleted by CA when time passes t For example, the client in C_i wants to access the file in C_j, so it has to derive the secret key $SK_{j,t}$. However, the parameters $\{g_{StartTime}, \ldots, g_{NowaDay-1}\}$ were deleted cause of the time passes. The process is as follow:

$$\begin{aligned} SK_{j,t} &= \left(DK_{i,z}\right)^{\prod_{C_k < C_j, k \neq j} e_k \prod_{g_{NowaDay} \leq g_{EndDate}} g_l} mod\, n \\ &= \left(a^{\prod_{C_j \leq C_i} d_i \prod_{h_{StartTime} \leq h_{EndDate}} h_z}\right)^{\prod_{C_k < C_j, k \neq j} e_k \prod_{g_{NowaDay} \leq g_{EndDate}} g_l} mod\, n \\ &= \left(a^{d_j}\right)^{\prod_{h_{StartTime} < h_{NowaDay}} h_l} mod\, n \\ &\neq \left(a^{d_j}\right)^{\prod_{h_{StartTime} < h_t} h_l} mod\, n, \end{aligned} \tag{4}$$

where *NowaDay* means the time now. The secret key $SK_{j,t}$ cannot compute correctly, because the deleted parameters. So the file is said assuredly deleted. Noted that this assured deletion doesn't interaction with the cloud.

4 An Example

In the following example, the proposed scheme applies to the structure of an organization with cloud storage provider. The structure is in the Fig. 1. The clients in the class C_1 have highest authority, they can access and derivate the secret key in the class C_2 and C_4. However, they can't access the class C_3, it is restricted by the transitive exceptions. Clients in C_2 can access and derivate secret key in C_3 and C_4. Besides, the clients in C_4 can also access C_2, because the anti-symmetrical. At last, the clients in C_3 have the lowest authority, they can only access the files by C_3 as themselves.

Supposed the start time of the system is $StartTime = 1$ and the maximum time period $z = 12$. CA chooses the parameters $\{e_1, e_2, e_3, e_4\}$ and $\{g_1, g_2, \ldots, g_{12}\}$ for class and time period respectively, and calculate the modular n. There have two clients, one in C_1, and the other in C_2. Assumed the client in C_1 wants its *EndDate* be 6(supposed it be $U_{1,6}$), and the client in C_2 be 5($U_{2,5}$). CA then computes $DK_{1,6} = a^{d_1 d_2 d_4 h_1 h_2 h_3 h_4 h_5 h_6} mod\, n$ and $DK_{2,5} = a^{d_2 d_3 d_4 h_1 h_2 h_3 h_4 h_5} mod\, n$ for $U_{1,6}$ and $U_{2,5}$ respectively. Next, CA issues the tamper-resistant device $\{DK_{1,6}, e_1, e_2, e_4, g_1, g_2, \ldots, g_6, n\}$ and $\{DK_{2,5}, e_2, e_3, e_4, g_1, g_2, \ldots, g_5, n\}$ to them.

If the *NowaDay* is equal to the *StartTime*, $U_{2,5}$ wants to encrypt a file and set assured deletion time is 4. Then it can use the derivation key $DK_{2,5}$ to derive the secret key $SK_{2,4}$ by formula 2 as follow:

$$\begin{aligned} SK_{2,4} &= \left(DK_{2,5}\right)^{e_3 e_4 g_4 g_5} mod\, n \\ &= \left(a^{d_2 d_3 d_4 h_1 h_2 h_3 h_4 h_5}\right)^{e_3 e_4 g_4 g_5} mod\, n \\ &= a^{d_2 h_1 h_2 h_3} mod\, n \end{aligned}$$

Because $DK_{2,5}$ can calculate the secret key $SK_{2,4}$, $U_{2,5}$ doesn't have to store it. When $U_{1,6}$ wants to access the file encrypted with $SK_{2,4}$, it can also use its derivation key by formula 3 as follow:

$$\begin{aligned} SK_{2,4} &= \left(DK_{1,6}\right)^{e_1 e_4 g_4 g_5 g_6} mod\, n \\ &= \left(a^{d_1 d_2 d_4 h_1 h_2 h_3 h_4 h_5 h_6}\right)^{e_1 e_4 g_4 g_5 g_6} mod\, n \\ &= a^{d_2 h_1 h_2 h_3} mod\, n \end{aligned}$$

While time passing and the *NowaDay* become 6, the parameters $\{g_1, g_2, \ldots, g_5\}$ will be deleted in the tamper-resistant device. If $U_{1,6}$ wants to access the file with $SK_{2,4}$, it cannot decrypt this as follow:

$$\begin{aligned} SK_{2,4} &= \left(DK_{1,6}\right)^{e_1 e_4 g_6} mod\, n \\ &= \left(a^{d_1 d_2 d_4 h_1 h_2 h_3 h_4 h_5 h_6}\right)^{e_1 e_4 g_6} mod\, n \\ &= a^{d_2 h_1 h_2 h_3 h_4 h_5} mod\, n \\ &\neq a^{d_2 h_1 h_2 h_3} mod\, n \end{aligned}$$

Without the parameters, the secret key cannot be derivate anymore. Note that the above computations are computing by the tamper-resistant. It can avoid the computation process being revealed. What the client has to do is entering the class and time it wants to encrypt or decrypt. Besides, the device is tamper resistant and had embedded a secure clock to control the correct time, no one could change the time of clock or values.

5 Security Analysis

This section discusses that the attacker cannot recover the file protected by the proposed scheme.

5.1 Active Files

An active files means it is accessible before the deletion time, but it may access by unauthorized users. In this scheme, if class C_i hasn't authority to access class C_j, its derivation key $DK_{i,z}$ will not include a hidden multiplicative inverse d_j. Therefore, the secret key $SK_{j,t}$ is unable to be derived by formula 4. However, if clients are authorized, the derivation key $DK_{i,z}$ will reveal about class C_j for clients to derivate secret key $SK_{j,t}$ with other parameters.

This scheme also resists the common modulus attack. This kind of attack means the message is encrypted with two different exponent values both have the same modulus n and prime to each other. Then the message can be accessed without the private key d_i, this makes RSA cryptosystem insecure. Though all classes in this scheme use the same modulus n and different values of the exponents, it won't discover any data under the common modulus attack. Since no parameters are computed by power of e_i and g_i modular n and the secret key $SK_{i,t} = \left(a^{d_i}\right)^{\prod_{h_{StartTime} < h_l} h_l} mod\, n$ can compute by the tamper-resistant device, it's unnecessary to store. Besides, the parameters e_i and g_i are well protected by the tamper-resistant device, no one can change the value in it.

5.2 Deleted Files

A deleted file means being deleted and encrypted with the secret key $SK_{i,t}$ after the deletion time passing. Since the parameters $\left\{g_{StartTime}, \ldots, g_{NowaDay-1}\right\}$ which are smaller than *NowaDay* will be deleted by the detection of secure clock in the tamper-resistant device. The secret key $SK_{i,t}$ without correct parameters, the result won't be correct as Sect. 3.4 shows.

6 Conclusions

The proposed time-based assured deletion with a flexible hierarchical key assignment scheme is solving the problem of storing sensitive information in the cloud storage, such as personal health records. By the concept of time, the secret key had given a life cycle. After the predetermined time, it will be deleted, making the file unrecoverable. Besides, the hierarchy structure in the scheme has security clearance and closer to the reality, increasing the security and flexible to the organization.

References

1. Tang, Y., Lee, P.C., Lui, C.S., Perlman, R.: Secure overlay cloud storage with access control and assured deletion. IEEE Tran. Dependable Secure Comput **9**(6), 903–916 (2012)
2. Rahumed, A., Chen, C.H., Tang, Y., Lee, P.C., Lui, C.S.: A secure cloud backup system with assured deletion and version control. In: International Conference on Parallel Processing Workshop, pp. 160–167 (2011)
3. Tezuka, S., Uda, R., Okada, K.: ADEC: assured deletion and verifiable version control for cloud storage. In: 26th IEEE International Conference on Advanced Information Networking and Applications, pp. 23–30 (2012)
4. Perlman, R.: File system design with assured delete. In: Proceedings Network and Distributed System Security Symposium (NDSS) (2007)
5. Akl, S.G., Taylor, P.D.: Cryptographic solution to a problem of access control in a hierarchy. ACM Trans. Comput. Syst. **1**(3), 239–248 (1983)
6. Lin, I.C., Hwang, M.S., Chang, C.C.: A new key assignment scheme for enforcing complicated access control policies in hierarchy. Future Gener. Comput. Syst. **19**(4), 457–462 (2003)

7. Wang, S.Y., Laih, C.S.: Merging: an efficient solution for a time-bound hierarchical key assignment scheme. IEEE Tran. Dependable Secure Comput. **3**(1), 91–100 (2006)
8. Tzeng, W.G.: A time-bound cryptographic key assignment scheme for access control in a hierarchy. IEEE Trans. Knowl. Data Eng. **14**(1), 182–188 (2002)
9. Bertino, E., Shang, N., Wagstaff, S.S.: An efficient time-bound hierarchical key management scheme for secure broadcasting. IEEE Trans. Dependable Secure Comput. **5**(2), 65–70 (2008)
10. Yeh, J.H.: An RSA-based time-bound hierarchical key assignment scheme for electronic article subscription. In: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, pp. 285–286 (2005)
11. Dropbox. https://www.dropbox.com
12. Amazon Simple Storage Service (S3). http://aws.amazon.com/s3/
13. Armbrust, M., et al.: Above the clouds: a Berkeley view of cloud computing, Electrical Engineering and Computer Sciences, University of California at Berkeley, Technical report No. UCB/EECS-2009-28 (2009)
14. Dillon, T., Chen, W., Chang, E.: Cloud computing: issues and challenges. In: 24th IEEE International Conference on Advanced Information Networking and Applications, pp. 27–33 (2010)
15. Kaletsch, A., Sunyaev, A.: Privacy engineering: personal health records in cloud computing environments. In: International Conference on Information System (2011)