A New Key Assignment Scheme for Enforcing Complicated Access Control Policies in Hierarchy *

Iuon-Chang Lin[‡] Min-Shiang Hwang[†] Chin-Chen Chang[‡]

Department of Information Management[†]
Chaoyang University of Technology
168, Gifeng E. Rd., Wufeng,
Taichung County, Taiwan 413, R.O.C.
Email: mshwang@mail.cyut.edu.tw
Fax: 886-4-3742337

Department of Computer Science and Information Engineering[‡]
National Chung Cheng University
160, San-Hsing, Min-Hsiung, Chaiyi 621, Taiwan, R. O. C.

May 31, 2002

*This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no.: NSC90-2213-E-324-005.

†Corresponding author: Prof. Min-Shiang Hwang

A New Key Assignment Scheme for Enforcing Complicated Access Control Policies in Hierarchy

Abstract

In a traditional key assignment scheme, an access control policy is used to solve the access control problem in a hierarchy. A higher security class can access lower security classes, but the opposite is not allowed. However, in some cases, this can be troublesome because of the lack of flexibility. In this paper, we shall propose a secure key assignment scheme which can be performed not only in a hierarchy but also in more complicated policies with anti-symmetrical and transitive exceptions.

Keywords: Access control, cryptography, data security, key assignment, multilevel security

1 Introduction

In the past decade, many key assignment schemes have been proposed in the literature to control access in a hierarchy [1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12]. In a hierarchic access control policy, all users are allocated into a number of disjoint sets of security classes $C_1, C_2, \cdots, C_m$. According to the partially ordered hierarchy, a user in security class C_j can derive the secret keys of the users in any security class C_i that is in the same security level as C_j or lower, but the opposite is not allowed. In other words, the users in C_j can access the information held by the users in C_i. The relation can be expressed as $C_i \leq C_j$.

In real-life situations, many organizations are in partially ordered hierarchies. However, the hierarchy structure is not suitable for all the organization in the societies. For example, there is a more flexible organization structure in Figure 1. A user in the top level user class C_1 possesses the authority to access information items of classes C_2 and C_4, but access to the information items of C_3 is not allowed; a user in C_2 can access information items of C_3 and C_4; a user in C_4 can access information items of C_2. It is difficult to meet the relationships by using the traditional key assignment scheme in hierarchy.

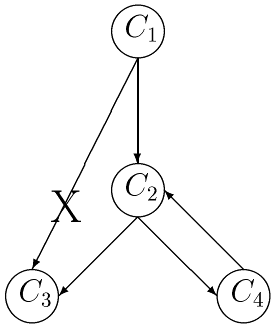

Figure 1: An example access control policy in a hierarchy structure with explicit anti-symmetric and transitive exceptions

Recently, Yeh, Chow and Newman proposed a more flexible key assignment scheme [14] (named, the YCN scheme) for enforcing access control policy in a user matrix model. The user matrix model cannot only enforce the access control policies in the user hierarchy model but also enforce the two extension policies: *transitive exceptions* and *anti-symmetrical arrangements*. The key assignment scheme in the hierarchy structure with an explicit transitive exception policy goes that C_i can access C_j and C_j can access C_k, but C_i cannot access C_k. For example, C_1 can access C_2 and C_2 can access C_3, but C_1 cannot access C_3 as Figure 1 shows.

The key assignment scheme in the hierarchy structure with an anti-symmetrical policy is that C_i can access C_j and C_j can access C_i, but C_i and C_j are two different user classes. For example, C_2 can access C_4 and C_4 can access C_2, but C_2 and C_4 are two different user classes as Figure 1 shows. Therefore, the YCN scheme is more flexible than the schemes in the user hierarchy model in solving the access control problem. The scheme has opened a brand new research area for key assignment in a hierarchy. However, the YCN scheme is not secure. Hwang presented counter-evidence to point out the YCN scheme is not secure [9]. In some cases, several user classes in YCN scheme can collaborate to derive the derivation and encryption keys. To amend the problem in security, we shall propose a secure key assignment scheme which can also enforce the complicate access control policies. Besides, our proposed scheme does not require large amount of storage for storing the public parameters.

The organization of the paper is as follows. In the next section, we shall briefly review related work on key assignment schemes in a user hierarchy. Following the review, we shall propose our new key assignment scheme for enforcing more flexible access control policies in Section 3. In Section 4, we shall give a simple example to illustrate our scheme. The security of our proposed scheme and the required storage and computations are analyzed thereafter. Finally, our conclusion will be presented in the last section of this paper.

2 Related Work

In this section, we shall introduce some related works that are key assignment schemes in a hierarchy. One of the simplest methods to control access in a hierarchy is to make only the authorized users hold all the users' secret keys of the successor security classes. In this simple method, each user must hold and manage a set of subordinate keys. Such an arrangement raises the key management problem of multilevel security [1].

To solve this problem, one of the workable ways is to use a super-key instead of many subordinate keys. The concept of the super-key was first introduced by Akl and Taylor in [1]. Since then, many schemes have proposed their schemes for solving the problem based on the concept of the super-key. Akl and Taylor designed the key assignment scheme using the top-down approach. In their scheme, a central authority assigns to each user class a prime, a secret key, and a public parameter. If C_j has a security clearance higher than C_i, the users in C_j can easily derive the secret key of C_i with their own secret key and the public parameters of C_i and C_j. Thus, the scheme can solve the key management problem. However, the values of public parameters are very large. Since the public parameter of the user class C_i in the Akl-Taylor scheme is the product of the primes of C_j which is not a descendant of C_i, the scheme requires a large amount of storage to store the public parameters.

In 1985, Mackinnon et al. [11] proposed an algorithm to reduce the values of public parameters in Akl and Taylor's scheme. The method is called canonical assignment. However, the scheme also requires a large amount of storage to store the public parameters [2, 12]. Moreover, the optimal canonical algorithm is difficult to find.

In 1990, Harn and Lin proposed a cryptography-based hierarchy scheme [3]. This scheme is similar to the Akl-Taylor scheme, but the Harn-Lin scheme used a bottom-up approach instead of the top-down approach employed in the Akl-Taylor scheme. Moreover, the security of the Harn-Lin scheme is based on the difficulty of factoring a large number. The main advantage of this scheme is that the size of the storage space is much smaller.

Recently, many related schemes have been proposed [2, 4, 5, 6, 7, 12]. These schemes are quite advanced in storage complexity, computational complexity, and the efficiency of changing the user classes. However, they are not flexible enough to suit more complicated policies, such as policies with transitive exceptions and anti-symmetrical arrangements. In [14], Yeh, Chow, and Newman proposed a key assignment scheme in a user matrix model, which is more flexible than the schemes in a user hierarchy. In the YCN scheme, it is impossible for illegal users to derive the derivation and encryption keys. However, Hwang shown that several user classes can collaborate to derive the derivation and encryption keys in some cases under YCN scheme [9]. In this paper, we shall propose a secure key assignment scheme to amend the problem in security. Our proposed scheme is also more flexible.

3 A New Key Assignment Scheme

In this section, we shall propose a new key assignment scheme for access control in a more flexible hierarchy structure. We design the scheme using the bottom-up approach. The security of our scheme is based on the difficulty of factoring a product of two large primes.

There is a Central Authority (CA) in our scheme. The responsibility of this CA is to generate and distribute keys. Initially, CA assigns each user class two keys: a secret key and a derivation key. The secret key is used to encipher and decipher documents in a symmetric cryptosystem such as DES, IDEAL, or AES (Advanced Encryption Standard) [13]. The derivation key is used to derive the secret key of other user classes which are allowed to access. The details of the new key assignment scheme are described as follows.

Step 1. CA randomly chooses two large primes: p and q. Both of p and q need to be kept secret. Next, CA calculates n such that $n = p \times q$, where n is public.

Step 2. CA chooses another parameter, g, which is relatively prime to n and in the range between 2 and $n-1$.

Step 3. CA chooses a set of distinct primes $\{e_1, e_2, \cdots, e_m\}$ for all user classes $\{C_1, C_2, \cdots, C_m\}$, where e_i has to relatively prime to $\phi(n)$, i.e. $\gcd(\phi(n), e_i)=$ 1 and $1 < e_i < \phi(n)$. Then, CA publishes the parameters $\{e_1, e_2, \cdots, e_m\}$ and n.

Step 4. CA calculates $\{d_1, d_2, \cdots d_m\}$, where each d_i is the multiplicative inverse of e_i, i.e. $e_i \times d_i \equiv 1 \bmod \phi(n)$, where $\phi(n)$ denotes the Euler's totient function of n.

Step 5. CA generates the derivation keys $\{DK_1, DK_2, \cdots, DK_m\}$ and the secret keys $\{SK_1, SK_2, \cdots, SK_m\}$ for all user classes $\{C_1, C_2, \cdots, C_m\}$ as follows.

$$\begin{aligned} DK_i &= g^{\prod_{C_j < C_i}(d_j)} \bmod n, & (1) \\ SK_i &= g^{d_i} \bmod n. & (2) \end{aligned}$$

$C_j < C_i$ means that the user class C_i possesses the authority to access the information items of C_m. Next, CA delivers SK_i and DK_i to each user in the user class C_i through a secure channel. Each user has to keep SK_i and DK_i secret.

Step 6. If the user classes keep the relation $C_j < C_i$, an user in class C_i can derive the secret key of class C_j with the derivation key DK_i as follows:

$$\begin{aligned} SK_j &= DK_i^{\prod_{C_k < C_i, k \neq j}(e_k)} \bmod n \\ &= (g^{\prod_{C_k < C_i}(d_k)})^{\prod_{C_k < C_i, k \neq j}(e_k)} \bmod n \\ &= g^{d_j} \bmod n. & (3) \end{aligned}$$

The public parameters in the Akl-Tayor scheme [1] are the products of the primes associated with non-authority classes. If there are many user classes in the system, the values of public parameters will very large. Therefore, their scheme requires a large amount of storage to store the public parameters. In contrast, the public parameter e_i of user class C_i is a single prime in our proposed scheme. Thus, our scheme requires only small storage to store the public parameters.

4 An Example

In the following example, we apply the proposed scheme to the structure of an organization in Figure 1. The users in C_1 possess the greatest authority; they can derive the secret keys of the uses in C_2 and C_4, but they cannot derive the secret key of the users in C_3 because it is restricted by the transitive exceptions policy. The users in C_2 have the authority to derive the secret keys of the users in C_3 and C_4. Furthermore, the anti-symmetrical policy allows that the users in C_4 can derive the secret key of the uses in C_2. Finally, the users in C_3 have the least authority; they can only access information held by the uses in the same class as themselves.

Initially, CA chooses the public parameters e_1, e_2, e_3, e_4 for all user classes C_1, C_2, C_3, C_4 and calculates the public modular n. According to Equation (1), CA can calculate the derivation keys $DK_1 = g^{d_2 \times d_4} \bmod n$, $DK_2 = g^{d_3 \times d_4} \bmod n$, $DK_3 = null$, and $DK_4 = g^{d_2} \bmod n$ for the users in C_1, C_2, C_3, and C_4 respectively. Obviously, the secret keys $SK_1 = g^{d_1} \bmod n$, $SK_2 = g^{d_2} \bmod n$, $SK_3 = g^{d_3} \bmod n$, and $SK_4 = g^{d_4} \bmod n$ for the users in C_1, C_2, C_3, and C_4 can also be calculated by CA using Equation (2).

Using the derivation keys and the public parameters, the users in C_1 can derive the secret keys SK_2 and SK_4 following the equations $SK_2 = DK_1^{e_4} \bmod n$ and $SK_4 = DK_1^{e_2} \bmod n$, but they cannot obtain the secret key of C_3. Similarly, the users in C_2 can also use their own derivation key to derive the secret keys SK_3 and SK_4 following the equations $SK_3 = DK_2^{e_4} \bmod n$ and $SK_4 = DK_2^{e_3} \bmod n$. The uses in C_3 cannot derive any secret key. The secret key of C_3 can also be derived by the users in C_4 with the equation $SK_2 = DK_4 \bmod n$. With the secret key, users can decipher the plaintext and access the information they want. The scheme is simple, and the access policies are more flexible than traditional hierarchy structures.

5 Discussions

In this section, we shall examine the security of our proposed key assignment scheme. In addition, we shall also discuss the required storage and computational complexity in our proposed scheme.

5.1 Security Analysis

The security features of our proposed scheme are described as follows.

1. **Difficulty for factoring the modular n.** From the public parameter e_i and the modular n, no one can derive the multiplicative inverse d_i. The security is similar to that of the RSA cryptosystem [13]; it is based on the difficulty of factoring the modular n. Any adversary who wants to derive the multiplicative inverse d_i from the public parameter e_i and the modulus n has to factor n into its two prime factors. Currently, there are many factoring algorithms, but they are all time-consuming. Furthermore, if an adversary tries all possible corresponding multiplicative inverses, d_i, until he/she finds the correct one, it is in fact not more efficient than trying to factor n [13].

2. **Preventing the unauthorized users to access.** Without the authority to access the information in C_j, the users in C_i cannot derive the secret key of the users in C_j. If the users in C_i has the authority to access the information in C_j, the derivation key DK_i have a hidden multiplicative inverse d_j. Therefore, the secret key SK_j will be able to be derived by Equation (3). However, if users are not be authorized, the derivation key DK_i will reveal no information about C_i, and there will be no way for the users to derive the secret key from the derivation key or the public parameters.

3. **Resisting common modulus attack.** If everyone gives the same modulus n, but different values for the exponents d_i and e_i, the RSA cryptosystem is not secure. The problem occurs when the same message is encrypted by two different exponents (both having the same modulus), and the two exponents are relatively prime, then the message can be recovered without using the private key d_i [13]. The attack is called

common modulus attack. For example, a message m is encrypted by using the keys e_1 and e_2. The two ciphertexts are

$$\begin{aligned} c_1 &= m^{e_1} \mod n \ \ and \\ c_2 &= m^{e_2} \mod n. \end{aligned}$$

Since the e_1 and e_2 are relatively prime, we can derive r and s using Euclidean algorithm, such that

$$re_1 + se_2 = 1.$$

Thus, the plaintext m can be recovered by using the following equation without the private key d.

$$(c_1)^r \times (c_2)^s = m \mod n.$$

In our scheme, even though all user classes use the same modulus and different values for the exponents e and d, our system will not reveal any information under the common modulus attack. Since $SK_i = g^{d_i}\ mod\ n$ must be kept secure by the users in class C_i, no public parameters are calculated by power of e_i modular n. Even through several users reveal their secret keys SK_i in different user classes C_i, only the parameter g can be derived by using the common modulus attack. However, revealing the parameter g does not harm the security of our scheme, because the multiplicative inverses d_i are unknown.

4. **Resisting collaborate attack.** In collaborate attack, several user classes may reveal their derivation and secure keys to try to derive the derivation and secure keys of the unauthorized classes. Using the common modulus attack with the revealed secure keys, only the parameter g can be derived. Furthermore, the $DK_i = g^{\prod_{C_j<C_i}(d_j)}\ mod\ n$, the exponential part only contains the multiplicative inverses d_j of the authorized classes. Therefore, the collaboration is not helpful to derive the derivation

and secure keys of the unauthorized classes. The key pointer is that the multiplicative inverses d_i are unknown, they are kept secure by CA.

5.2 Required Storage and Computational Complexity

Assume that there be m user classes in the hierarchy. From the algorithm of our proposed scheme, the public parameters $\{e_1, e_2, \cdots, e_m\}$ and n are $m+1$ integers, where the binary value of $\{e_1, e_2, \cdots, e_m\}$ are between 1 and $\phi(n)$. That is, the size of the public parameters must be less than or equal to $\log_2(n)$. Let the size of each public parameter is k bits, where $2^{k-1} < n \leq 2^k$. Generally, the length of k in the range of 512 bits to 1024 bits is secure. Therefore, the amount of the required storage for storing the public parameters is $(m+1)\lceil \log_2(n) \rceil$ bits, i.e. $(m+1)k$ bits. Recall the Akl-Taylor scheme, the public parameter of the user class C_i is a product of the primes dedicated to the user classes which are not the descendants of C_i. Obviously, the amount of the required storage in our scheme is much less than the Akl-Taylor scheme, especially when the number of user classes in the hierarchy is large.

The computational complexity of our scheme is also simple. When a user in class C_i and the relationship $C_j < C_i$ holds, the user can derive the secret key of class C_j with the derivation key DK_i from Equation 3. It requires r multiplications and 1 modular exponential computations, where r is the number of immediate successors of the processed user class. Therefore, our scheme is efficient to implement.

6 Conclusions

We have proposed a secure key assignment scheme for solving the multilevel access control problem in complicated access control policies. The main contribution of our scheme is that it can be used in more flexible applications than that of the scheme proposed previously. In this scheme, the access control policy cannot only be enforced in a hierarchy but also employed for more complicated policies with anti-symmetrical arrangements and transitive exceptions. Furthermore, our scheme does not require large amount of storage for storing public parameters.

References

[1] S. G. Akl and P. D. Taylor, "Cryptographic solution to a problem of access control in a hierarchy," *ACM Transactions on Computer Systems*, vol. 1, pp. 239–248, Jul 1983.

[2] C. C. Chang, R. J. Hwang, and T. C. Wu, "Cryptographic key assignment scheme for access control in a hierarchy," *Information Systems*, vol. 17, no. 3, pp. 243–247, 1992.

[3] Lein Harn and Hung Yu Lin, "A cryptographic key generation scheme for multilevel data security," *Computers & Security*, vol. 9, pp. 539–546, Oct. 1990.

[4] Min-Shiang Hwang, "A cryptographic key assignment scheme in a hierarchy for access control," *Mathematical and Computer Modelling*, vol. 26, no. 2, pp. 27–31, 1997.

[5] Min-Shiang Hwang, "Extension of CHW cryptographic key assignment scheme in a hierarchy," *IEE Proceedings Computers and Digital Techniques*, vol. 146, no. 4, p. 219, 1999.

[6] Min-Shiang Hwang, "An improvement of a dynamic cryptographic key assignment schemes in a tree hierarchy," *Computers & Mathematics with Applications*, vol. 37, no. 3, pp. 19–22, 1999.

[7] Min-Shiang Hwang, "An improvement of novel cryptographic key assignment scheme for dynamic access control in a hierarchy," *IEICE Trans-*

actions on Fundamentals Of Electronics, Communications and Computer Sciences, vol. 82-A, no. 3, pp. 548–550, 1999.

[8] Min-Shiang Hwang, "A new dynamic cryptographic key generation scheme in a hierarchy," *Nordic Journal of Computing*, vol. 6, no. 4, pp. 363–371, 1999.

[9] Min-Shiang Hwang, "Cryptanalysis of YCN key assignment scheme in a hierarchy," *Information Processing Letters*, vol. 73, no. 3, pp. 97–101, 2000.

[10] Min-Shiang Hwang, "An asymmetric cryptographic scheme for a totally-ordered hierarchy," *International Journal of Computer Mathematics*, vol. 73, pp. 463–468, 20009.

[11] Stephen J. Mackinnon, Peter D. Taylor, Henk Meijer, and Selim G. Akl, "An optimal algorithm for assigning cryptographic keys to control access in a hierarchy," *IEEE Transactions on Computers*, vol. 34, pp. 797–802, Sep. 1985.

[12] R. S. Sandhu, "Cryptographic implementation of a tree hierarchy for access control," *Information Processing Letters*, vol. 27, pp. 95–98, 1988.

[13] Bruce Schneier, *Applied Cryptography, 2nd Edition.* New York: John Wiley & Sons, 1996.

[14] J. H. Yeh, R. Chow, and R. Newman, "A key assignment for enforcing access control policy exceptions," in *Proceedings on International Symposium on Internet Technology*, pp. 54–59, Taipei, 1998.