New Key Management Systems for Multilevel Security

Hwankoo Kim[1], Bongjoo Park[1], JaeCheol Ha[2], Byoungcheon Lee[3], and DongGook Park[4]

[1] Information Security Major, Div. of Computer Science and Engineering, Hoseo University, Asan 336-795, Korea
{hkkim, bjpark}@office.hoseo.ac.kr
[2] Dept. of Information and Communication, Korea Nazarene University, Cheonan 330-718, Korea
jcha@kornu.ac.kr
[3] Dept. of Information Security, Joongbu University, Kumsan-Gun 312-702, Korea
sultan@joongbu.ac.kr
[4] School of Information Technology, SunChon University, SunChon 540-742, Korea
dgpark6@sunchon.ac.kr

Abstract. In this paper, we review briefly Akl and Taylor's cryptographic solution of multilevel security problem. We propose new key management systems for multilevel security using various one-way functions.

1 Introduction

Secret data should be managed for access to authorized people only. In order to do so, secret keys must be distributed solely to those with access to the pertaining information. However, this is not a simple problem to solve.

First, let us recall notation and terminology from [1]. Assume that the users of a computer system are divided into a number of disjoint sets $S = \{U_1, U_2, \ldots, U_n\}$. The term *security class* (or *class*, for short) will be used to designate each of the U_i. The meaning of $U_i \leq U_j$ in the partially ordered set $(S, \leq)$ is that users in U_i have a *security clearance* lower than or equal to those in U_j. Simply put, this means that users in U_j can have access to information held by (or destined to) users in U_i, while the opposite is prohibited.

Let x_m be a piece of information, or object, that a central authority (CA) desires to store (or broadcast over) the system. The meaning of the subscript m is that object x is accessible to users in class U_m. The partial order on S implies that x_m is also accessible to users in all classes U_i such that $U_m \leq U_i$. It is required to design a system which, in addition to satisfying the above conditions, ensures that access to the information is as decentralized as possible. This means that authorized users should be able to retrieve x_m independently as soon as it is stored or broadcast by CA.

This access control problem arises in organizations where a hierarchical structure exists. Government, the diplomatic corps, and the military are examples of

O. Gervasi et al. (Eds.): ICCSA 2005, LNCS 3481, pp. 245–253, 2005.

such hierarchies. Applications also exist in business and in other areas of the private sector, for example in the management of databases containing sensitive information, or in the protection of industrial secrets. Finally, the model is employed in the design of computer operating systems to control information flow from one program to another. General references for any undefined terminology and notion are [5, 6, 10, 13, 14].

In sections 2 and 3, we review briefly Akl and Taylor's cryptographic solution of multilevel security problem from [1]. We propose new key management systems for multilevel security using various one-way functions from section 4 to 7.

2 Cryptographic Solution

Let E (resp., D) be an encryption (resp., a decryption) algorithm of a cryptosystem, such as DES (Data Encryption Standard) or AES (Advanced Encryption Standard). Then the simplest cryptographic solution to access control problem may be obtained as follows ([1]). The CA generates n keys $\{K_i\}$ and distributes to U_i its own key K_i and all keys K_j belonging to U_j below U_i in the hierarchy. When an object x_m is to be stored (or broadcast), it is first encrypted with K_m to obtain $x' = E_{K_m}(x_m)$ and then stored (or broadcast) as the pair $[x', m]$. This guarantees that only users in possession of K_m will be able to retrieve x_m from $x_m = D_{K_m}(x')$.

As pointed out in [1], this solution has the advantage that only one copy of x_m is stored or broadcast and the operations of encryption and decryption are performed just once. Its disadvantage is the large number of keys that must be held by each user. To avoid this problem, Akl and Taylor proposed in [1] a new scheme to manage keys such a way that a system is used by which K_i can be feasibly computed from K_j if and only if $U_i \leq U_j$.

3 Overview of Known Schemes

In the case where the structure of classes is totally ordered, we can distribute multilevel security keys using a function $H(M) = M^2 \bmod pq$ as shown in Fig. 1. Instead of this function, we can also use $H(M) = M^3 \bmod pq$. In 1982, applying these functions, Akl and Taylor proposed a cryptographic solution of key management for multilevel security for any poset. The following is a brief review of their method.

For any given poset, we add a top class if there is no any. The CA assigns an integer t_i to each class U_i so that $U_i \leq U_j$ if and only if $t_j | t_i$. The CA chooses a random K, computes $K^{t_i} \bmod pq$, and then distributes it to each class U_i. If $U_i \leq U_j$, then $t_i = d \cdot t_j$ for some integer d. Thus a user in U_j can get the key of U_i by computing $(K^{t_j})^d = K^{t_j d} = K^{t_i} \bmod pq$. This key management system (KMS for short) is not secure. For example, as shown in Fig. 2, U_3 and U_1 can conspire together to find the (master) key K of the top class. For $K^9/(K^4)^2 = K$.

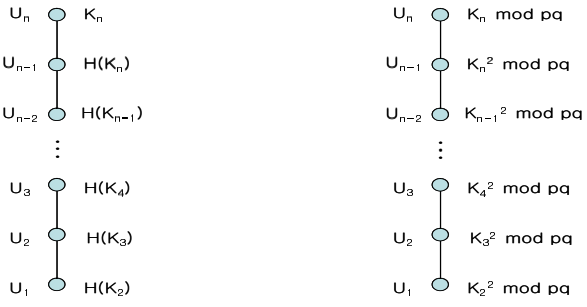

Fig. 1. Key management system for a totally ordered set

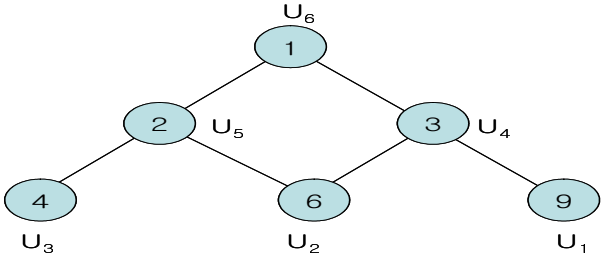

Fig. 2. Key management system using RSA - Conspired version

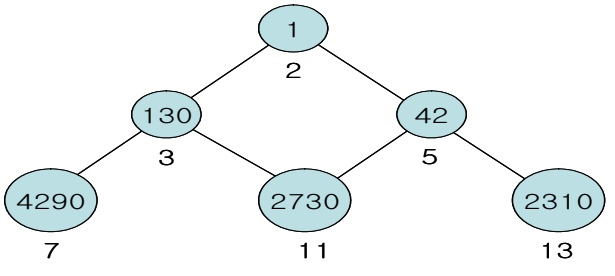

Fig. 3. Key management system using RSA - Improved version

Thus Akl and Taylor proposed a new method to avoid such a problem. They suggested a new algorithm for the assignment of t_i's. Each class U_i is assigned a distinct prime p_i and $t_i = \prod_{U_j \not\leq U_i} p_j$. (See Fig. 3.)

In [2], Akl and Taylor proposed a time-versus-storage trade-off for addressing their key management system. It was shown in [8, 9] that the key generation algorithm of [2] became inefficient when the number of users was large. As a result, an improved algorithm can be described and its optimality is shown.

4 KMS Using a One-Way (Cryptographic) Hash Function

Since Akl and Taylor's KMS for multilevel security uses exponentiation, the overload of computation of keys is high. However, we can compute keys faster than Akl and Taylor's method as shown below if we use a one-way hash function.

Once again, if there is no top class, we add a top class. Then the CA assigns a name to each class as in Fig. 4. Note that, in a lower tree, there is a unique class, denoted by $c(U_k)$, which covers a class U_k of a lower tree with top class.

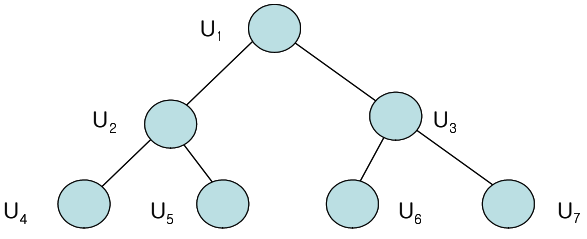

Fig. 4. A lower tree

Now the CA selects a key K belonging to the top class and a one-way hash function H. The CA computes $K_{U_k} = H(U_k, K_{c(U_k)})$ and distributes it together with H to each class U_k. Then the users in an upper class can compute all keys belonging to the classes lower than theirs using their keys, hash function H, and names of lower classes. (See Table 1.) Because of the one-wayness of the hash function, a user in U_k can not compute others' keys belonging to upper classes.

Table 1. Distribution of multilevel security keys for a lower tree using a hash function

Class	Keys
U_1	$K_{U_1} = K$
U_2	$K_{U_2} = H(U_2, K_{U_1})$
U_3	$K_{U_3} = H(U_3, K_{U_1})$
U_4	$K_{U_4} = H(U_4, K_{U_2})$
U_5	$K_{U_5} = H(U_5, K_{U_2})$
U_6	$K_{U_6} = H(U_6, K_{U_3})$
U_7	$K_{U_7} = H(U_7, K_{U_3})$

5 KMS Using RSA Algorithm

While we can manage multilevel security keys for a lower tree using one-way hash functions and names of classes, there is no known algorithm of multilevel security key management for an upper tree. Now we propose a multilevel security key management for an upper tree using the RSA algorithm [12].

Table 2. Distribution of multilevel security keys for an upper tree using the RSA

Class	Keys
U_1	$K_{U_1} = K$
U_2	$K_{U_2} = E(f_{U_2}(K_{U_1}))$
U_3	$K_{U_3} = E(f_{U_3}(K_{U_1}))$
U_4	$K_{U_4} = E(f_{U_4}(K_{U_2}))$
U_5	$K_{U_5} = E(f_{U_5}(K_{U_2}))$
U_6	$K_{U_6} = E(f_{U_6}(K_{U_3}))$
U_7	$K_{U_7} = E(f_{U_7}(K_{U_3}))$

If there is no bottom class, we add a bottom class. Now, we assign a name to each class. Since an upper tree is the dual poset of a lower tree, there is a unique class, denoted by $b(U_k)$, which is covered by a class U_k of an upper tree with bottom class.

Then, we select two large primes p and q and an encryption parameter e for the RSA algorithm and compute a decryption parameter d corresponding to e. Let $n = pq$ and we can define an encryption function E and a decryption function D as follows:

$$E(M) = M^e \bmod n \qquad D(C) = C^d \bmod n.$$

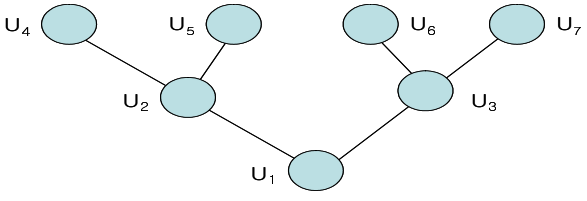

Fig. 5. An upper tree

Thereafter, the CA selects a key K belonging to the bottom class and a function f_{U_i}, the inverse of which is easy to compute. The CA distributes n, d and all f_{U_i}'s to each class and computes $K_{U_k} = E(f_{U_k}(K_{b(U_k)}))$ and distributes it to the class U_k. Here, the parameters p, q and e are secret to all users. Then the users in an upper class can compute all keys belonging to the classes lower than theirs using the decryption function D.

For example, assuming that the keys are distributed (See Table 2) for the poset in Fig. 5, a user belonging to U_7 can get $f_{U_7}(K_{U_3})$ from his/her key K_{U_7} using the RSA decryption function D. Furthermore, he/she can easily compute K_{U_3} by finding the inverse of f_{U_7}. Similarly he/she can get K_{U_1} from $K_{U_3} = E(f_{U_3}(K_{U_1}))$. To conclude, any user in class U_7 can compute the key K_{U_3} belonging to class U_3 and the key K_{U_1} belonging to class U_1.

6 KMS Using Poset Dimension

In this section, we propose a key management system using poset dimension. First we recall the notions of dimension and realizer in a poset from [15]. For any two posets (X, P) and (X, Q), Q is called an *extension* of P if $P \subseteq Q$. In particular, an extension Q of P is called a *linear extension* of P if Q is totally ordered. Let $\mathcal{E}(P)$ be the set of all linear extensions of P. Then it is easy to see that $P = \bigcap \mathcal{E}(P)$. Now, we define the *dimension* of any poset P as follows:

$$\dim(X, P) = \min\{|\Theta| : \Theta \text{ is a family of linear extension of } P, P = \bigcap \Theta\}.$$

A family Θ of linear extensions of P is called a *realizer* if $P = \bigcap \Theta$. It is easy to see that $\dim(X, P) = 1$ if and only if (X, P) is totally ordered.

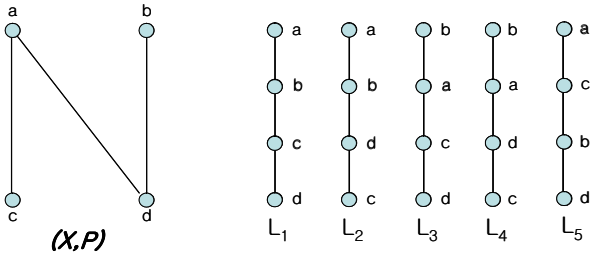

Fig. 6. Poset (X, P) and its linear extensions

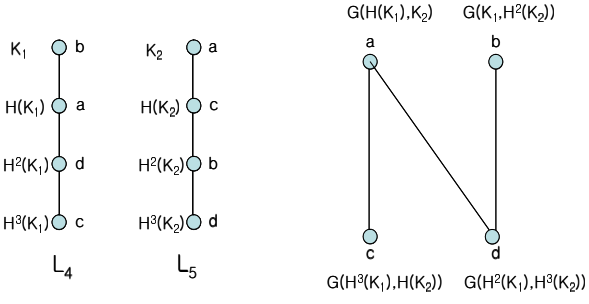

Fig. 7. Key management system using poset dimension

Example 1. Let $X = \{a, b, c, d\}$ be a set and let $P = \{(a, a), (b, b), (c, c), (d, d), (c, a), (d, a), (d, b)\}$ be a partial order on X. Then (X, P) is a poset and the number of extensions of P is 14. In particular, the number of linear extensions of P is 5, as indicated in Fig. 6. Since $L_4 \cap L_5 = P$, $\Theta = \{L_4, L_5\}$ is a realizer of (X, P), and $\dim(X, P) = 2$ since (X, P) is not totally ordered.

Let (X, P) be a poset such that $\dim(X, P) = n$. Then by the definition of dimension, there exist realizers $\{L_1, L_2, \ldots, L_n\}$. The CA selects two one-way

hash functions H, G and generates n keys $K_i (1 \leq i \leq n)$ to be assigned to the top class of each $L_i (1 \leq i \leq n)$, and then in each $L_i (1 \leq i \leq n)$ keys for multilevel security are distributed using the hash function H. Then n keys and two hash functions G, H are assigned to each class of the poset (X, P). The users in a class can get the key for multilevel security as the output of the hash function G.

Unfortunately, this key management system is vulnerable to cooperative attacks. However, if a user's key is stored in a smart card or a PCMCIA card, there will be no threat against conspiracy attacks, since it can not be read.

7 KMS Using a Clifford Semigroup

In this section, we propose a key management system using a Clifford semigroup. First we recall the notion of a commutative Clifford semigroup from [16]. A commutative semigroup S is said to be a *Clifford semigroup* if S is a semilattice of groups. For ease of understanding, this condition means: S is a disjoint union of groups $G_\lambda, \lambda \in \Lambda$, where Λ is a semilattice; if $\alpha, \beta \in \Lambda$ and $\beta \leq \alpha$, there exists a homomorphism $\phi_{\alpha,\beta} : G_\alpha \to G_\beta$; if $x, y \in S$, the multiplication $x.y$ in S is defined as follows: If $x \in G_\alpha, y \in G_\beta$ and $\alpha \wedge \beta = \inf\{\alpha, \beta\}$, then $x.y = \phi_{\alpha,\alpha\wedge\beta}(x)\phi_{\beta,\alpha\wedge\beta}(y)$, where the second member is a product in the group $G_{\alpha\wedge\beta}$; $\phi_{\alpha,\beta}$ is called a *bonding homomorphism.*

If S is any commutative semigroup and e is an idempotent of S, the subgroup G_e generated by e is $G_e = \{x \in S : xe = x$ and $\exists\ y \in S$ such that $xy = e\}$. Let us denote by E the set of idempotents of S; the union of disjoint groups $S_0 = \bigcup\{G_e : e \in E\}$ is the maximal Clifford semigroup contained in S. In fact, the set E is a semilattice with the partial order defined by: $e \leq f$ if and only if $ef = e$; therefore the family $\{G_e : e \in E\}$ inherits the semilattice structure of E; we write $G_e \preceq G_f$ if G_e is less than or equal to G_f in this partial order; in this case the bonding homomorphism $\phi_{f,e} : G_f \to G_e$ is given by $x \mapsto xe$.

To give a concrete example, we refer to [4, 7, 11] for some notions and terminologies of ideal theory in number fields.

Example 2. Let R be a non-maximal order of an imaginary quadratic number field K. We denote by D its integral closure and the index f of R in D as an abelian group, i.e., $f = |D/R|$. Assume that f is sufficiently large. Let $\mathrm{CL}(R)$ be the class semigroup of R. Then by [16–Theorem 11] $\mathrm{CL}(R)$ is a Clifford semigroup and by [16–Proposition 13] the idempotents of $\mathrm{CL}(R)$ are the equivalent classes of ideals of the form $E = (k, \eta)$, where $k \in \mathbb{Z}$ divides f. Note that any idempotent E which is not equivalent to R is not invertible. If E and F are idempotents where $E \leq F$, then the bonding homomorphism $\phi_{F,E} : G_F \to G_E$ is defined by $\phi_{F,E}(K_0) = EK_0$, where K_0 is the key ideal. A representation for an ideal is given in [7], while an efficient algorithm for multiplication of ideals is given in [3–pp. 113]. Note that the computation of K_0 from EK_0 seems to be difficult unless E is equivalent to R. Now, the CA assign an idempotent E_i to each class U_i. For example, we consider the diagram in Fig. 4. The CA selects a random key K_0 and computes E_2K_0, E_3K_0 and distributes each of them to each class U_2, U_3. The CA computes $E_2E_4K_0$ and distributes it to U_4. Similarly the CA computes

keys of all classes and distributes each of them respectively. Then the users in an upper class can compute all keys belonging to classes lower than itself.

8 Conclusion and Further Study

In this paper, we proposed new key management systems for multilevel security using various one-way functions and mathematical notions. In particular, we proposed a key management system for multilevel security for an upper tree structure using RSA. In section 7, we also proposed a KMS for multilevel security using poset dimension. However, this system can be vulnerable to cooperative attacks. Thus, it will take further work to solve this problem. In the final section, we proposed a KMS for multilevel security using a Clifford semigroup. Nevertheless, parameter sizes need to be considered for precise and efficient performance of our systems, which aim to provide practical security.

Acknowledgements

This research was supported by the MIC, Korea, under the ITRC support program supervised by the IITA.

References

1. Akl, Selim G., Taylor, Peter D.: Cryptographic Solution to a Multilevel Security Problem. CRYPTO 1982: 237-249.
2. Akl, Selim G., Taylor, Peter D.: Cryptographic Solution to a Problem of Access Control in a Hierarchy, ACM Trans. Comput. Syst. **1**(3): (1983) 239–248
3. Buchmann, J., Willams, H. C.: A key-exchange system based on imaginary quadratic fields. J. Cryptology **1** (1988) 107–118.
4. Cohen, H.: A course in Computational Algebraic Number Theory, Springer, Berlin, 1995.
5. Delfs, H., Knebel, H.: Introduction to Cryptography: Principles and Applications, Springer-Verlag, Berlin, 2002.
6. Denning, D. E.: Cryptography and Data Security, Addison-Wesley, Reading, Massachusetts, 1982.
7. Kim, H., Moon, S.: Public-key cryptosystems based on class semigroups of imaginary quadratic non-maximal orders, in *Information Security and Privacy – ACISP 2003*, LNCS 2727, Springer-Verlag, Berlin, 2003, pp. 488–497.
8. MacKinnon, Stephen J., Akl, Selim G.: New Key Generation Algorithms for Multilevel Security. IEEE Symposium on Security and Privacy (1983) 72–78.
9. MacKinnon, Stephen J., Taylor, Peter D., Meijer, Henk, Akl, Selim G.: An optimal algorithm for assigning cryptographic keys to control access in a hierarchy, IEEE Trans. computers **34** (1985) 797–802.
10. Menezes, A. J., Oorschot, P. C., Vanstone, S. A.: Handbook of Applied Cryptography, CRC Press, Boca Raton, 1997.
11. Mollin, R. A.: Quadratics, CRC Press, Boca Raton, 1996.

12. Rivest, R. L., Shamir, A., Adelman, L.: A method for obtaining digital signatures and public key cryptosystems. Communications of the ACM **21** (1978) 120–126.
13. Schneier, B.: Applied Cryptography, John Wiley & Sons, Inc., 1996.
14. Stinson, D. R.: Cryptography: Theory and Practice, CRC Press, Boca Raton, 2002.
15. Trotter, William T.: Combinatorics and Partially Ordered Sets: Dimension Theory, The Johns Hopkins Univ. Press, Baltimore and London, 1992.
16. Zanardo, P., Zannier, U.: The class semigroup of orders in number fields. Math. Proc. Camb. Phil. Soc. **115** (1994) 379–391.