Computers Math. Applic. Vol. 25, No. 6, pp. 109–114, 1993
Printed in Great Britain.

0097-4943/93 $6.00 + 0.00

A DYNAMIC CRYPTOGRAPHIC KEY ASSIGNMENT SCHEME IN A TREE STRUCTURE

H. T. LIAW, S. J. WANG AND C. L. LEI
Department of Electrical Engineering, National Taiwan University
Taipei, Taiwan 106, R.O.C.

(Received June 1992)

Abstract—In this paper, we propose a dynamic cryptographic key assignment scheme based on Newton's interpolation method and a predefined one way function. All users are classified into disjoint sets of security classes that employ the relation of partial ordering, that is, a security class at higher level can derive from his own cryptographic key the keys of the other security class below him. Compared with the existing assignment schemes, our scheme always produces economic cryptographic keys, which are smaller than the keys generated by the previous work in a tree structure. Furthermore, whenever a new security class is inserted into the user hierarchy system, the corresponding keys can be determined without changing any existing keys. Therefore, our scheme is suitable for practical implementation.

1. INTRODUCTION

Information security is a very important topic in the computer communication system. Up to the present, the popularity of computer networks and fast progress of computer technologies on a multi-user system make sharing of expensive resources a reality. However, sharing may cause some undesired phenomena such as unauthorized accesses and inconsistent status of shared resources. Therefore, an important issue in a multi-user computer environment is the question of how to control the access to computers. Besides, how to identify whether a user has enough privileges to retrieve the messages and/or change the access rights of the other users is another important issue. To date, many papers [1–14] have addressed this problem. In this paper, we study this problem on a computer communication system organized in a tree structure. In order to broadcast information within a tree structured system, we need to have a method to ensure the security of the system. A method based on cryptography was proposed in [1] for controlling access to information among a group of users in a hierarchy. This method employs the relation of partial ordering, that is, a user at higher level can derive from his own cryptographic key the keys of the other users below him in the hierarchy organization, and then retrieve messages destined to the users at lower levels. In addition, this method has another important property of being able to provide security against two or more users at lower level of the system collaborating to derive a higher leveled key to which they are not entitled.

When the number of users becomes larger, the key generation algorithm in [1] becomes inefficient and infeasible. Therefore, an improved method was proposed in [12] for reducing the key values by using a canonical assignment method. Recently, Harn *et al.* [8] presented another improved method. Instead of using the top-down key assignment approach as in [1,12], their scheme adopted a bottom-up approach.

Tree structure is a special case of the hierarchy structure, yet it is very popular in our daily life. For example, the personnel organization of a company or an army, the pedigree chart and the lineal chart [15] are typical tree structures. In this paper, we propose an efficient dynamic

This work was supported in part by the National Science Council of the Republic of China under the Grant NSC 81-0416-E-002-20.

cryptographic key assignment scheme in a tree structure for access control. The rest of this paper is organized as follows. In Section 2, we discuss the related research. Our efficient dynamic cryptographic key assignment scheme for assigning cryptographic keys in a tree structure is described in Section 3. The correctness of our security enforcing scheme is established in Section 4. Finally, concluding remarks are given in Section 5.

2. RELATED RESEARCH

A method for assigning cryptographic keys for access control in a hierarchy was first proposed in [1]. It assumes that all users in a computer communication system are distributed into m disjoint classes C_i, $i \in S$, where $S = \{1, 2, \ldots, m\}$, and these classes receive messages from an authority C_0. The set of classes is partially ordered by the partial ordering relation "$\leq$". If there exists a relation between C_i and C_j, i.e., $C_i \leq C_j$, it means that users in C_j can retrieve information destined for users in C_i, while the opposite is not allowed. Besides, every class C_i, $i \in S$, must satisfy the relation $C_i \leq C_0$, since no user in the system is allowed to retrieve messages of C_0.

First, the authority C_0 generates a set of keys K_i's, and secretly sends each K_i to the class C_i, $i = 1, 2, \ldots, m$. In [1], the keys K_i can be generated as follows. A public integer t_i is assigned to each class C_i with the condition

$$C_i \leq C_j \text{ if and only if } t_j | t_i.^* \tag{2.1}$$

In addition, the authority C_0 chooses a random secret key K_0 and a secret pair (p, q) of large prime numbers, and makes $M = p \bullet q$ public. Then $K_i = K_0^{t_i} \pmod{M}$ is sent to C_i. By condition (2.1), t_i/t_j is an integer if $C_j \geq C_i$. Thus, C_j can compute K_i by the following deduction

$$K_i = K_0^{t_i} = (K_0^{t_j})^{t_i/t_j} = K_j^{t_i/t_j} \pmod{M}.$$

However, if $C_j \not\geq C_i$, then t_i/t_j is not an integer. Thus, C_j does not have the ability to retrieve the information destined to users in C_i in case the information is encrypted by K_i. Unfortunately, there exists a very serious problem: two or more users of the system can collaborate to derive a key to which they are not entitled. In order to avoid this drawback, a new scheme was proposed in [1]. This new scheme demands that each t_i, $1 \leq i \leq m$, should be computed from $t_i = \prod_{C_j \not\leq C_i} P_j$, where P_j, $j \in S$, is a distinct prime assigned to C_j. These t_i's can ensure that condition (2.1) is not violated. Moreover, collaborative attacks are not possible.

Furthermore, a canonical assignment method [12] was proposed to reduce the value of t_i. In this algorithm, the hierarchy structure is first decomposed into a set of disjoint chains. Each chain is a totally ordered subset and is assigned a distinct prime. For node i, they define the rule $n_i = P^{e_i}$, where e_i denotes the depth of the node from the top of the chain whose associated prime is P. After all n_i's are assigned, t_i's are computed from the formula $t_i = \text{lcm}_{C_j \not\leq C_i} n_j$.

Recently, Harn *et al.* [8] presented another improved method. Instead of using the top-down key assignment approach as in [1,12], their scheme adopts a bottom-up approach. It first assigns each security class C_i a distinct prime P_i, and makes these primes public. Then it calculates the multiplication inverse d_i for each class C_i. It means that $d_i = P_i^{-1} \bmod \Phi(M)$, where $\Phi(M)$ denotes the Euler's number of M. This new scheme demands that each t_j, $1 \leq j \leq m$, should be computed from $t_j = \prod P_i$, for all i such that C_i is the successor of C_j. Then $K_j = K_0^{\prod d_i \bmod \Phi(M)} \pmod{M}$ is sent to C_j. Once the secret keys and the corresponding public keys are generated, C_j can compute K_i by the following deduction

$$K_i = K_j^{t_j/t_i} \bmod M.$$

The common drawback of the previous work is the size of storage for storing public keys grows dramatically. In the following section, we present a dynamic cryptographic key assignment scheme based on Newton's interpolation method and a predefined one way function. Comparing with the existing assignment schemes, our scheme always produces economic cryptographic keys, which are smaller than the keys generated by the previous work in a tree structure.

*$t_j | t_i$ means t_i is divisible by t_j.

3. A DYNAMIC KEY ASSIGNMENT SCHEME FOR ACCESS CONTROL IN A TREE STRUCTURE

Since our proposed scheme is based on a predefined one way function and Newton's interpolation polynomials (NIP for short) [16–18] in this section, we first introduce their ideas and methods. Intuitively, a one way function is a function that is easy to apply but hard to reverse. Formally, if a function $F : A \rightarrow B$ is a one way function, it is a one-to-one function and implies that

(1) for every x in A, $F(x)$ can be computed easily, and
(2) for every $y = F(x)$ in B, it is infeasible to compute x.

On the other hand, suppose there exist $n+1$ points denoted as (x_i, y_i), $0 \leq i \leq n$. Then the NIP with degree n passing through these $n+1$ points can be derived as

$$NP(x) = \sum_{j=0}^{n} \left(f[x_0, x_1, \ldots, x_j] \prod_{i=0}^{j-1} (x - x_i) \right). \tag{3.1}$$

Each coefficient $f[x_0, x_1, \ldots, x_j]$ in equation (3.1) can be computed by the divided differences [16,17]. Without loss of generality, let each coefficient be represented in some residue class of prime P and $P > y_i$, $0 \leq i \leq n$. Then a NIP can be expressed as equation (3.2),

$$NP(x) = (\alpha_n(x - x_{n-1})(x - x_{n-2}) \ldots (x - x_0) + \cdots + \alpha_2(x - x_1)(x - x_0) + \alpha_1(x - x_0) + \alpha_0) \bmod P. \tag{3.2}$$

In equation (3.2), let $NP(x_i) = y_i \bmod P$, $1 \leq i \leq n$, and all α_i's are constant coefficients.

In our scheme, let P be a large prime number and F be a predefined one way function. Besides, both P and F are public. Furthermore, let $C_{i_1}, C_{i_2}, \ldots, C_{i_j}$ be the immediate successors of C_i with the public key pairs $(t1_{i_1}, t2_{i_1}), (t1_{i_2}, t2_{i_2}), \ldots, (t1_{i_j}, t2_{i_j})$, respectively. In the following, an efficient algorithm for assigning the secret key and the corresponding public key pairs for each security class is presented. Our algorithm essentially consists of two phases.

(1) In the first phase, we assign public key pair $(t1_i, t2_i)$ for each security class C_i, starting from the root and going down to leaves.
(2) In the second phase, we generate secret key K_i for each security class C_i, starting from the root and going down to leaves.

The first phase of our algorithm assigns each security class C_i of the tree T a key pair $(t1_i, t2_i)$, which is equal to the pair (α_i, x_i) in NIP. After all $(t1_i, t2_i)$'s are assigned, all K_i's are computed in phase 2.

ALGORITHM ASSIGNMENT
INPUT: A tree T.
OUTPUT: A set of $(t1_i, t2_i)$'s and K_i's being assigned to the nodes of T, respectively.
Phase 1: Assign public key pair $(t1_i, t2_i)$ for each node C_i, starting from the root and going down to leaves.
STEP 1.1: Assign distinct key pairs $(t1_i, t2_i)$'s for all the non-root nodes C_i's, $1 \leq i \leq n$.
Phase 2: Generate a secret key K_i for each node C_i, starting from the root and going down to leaves.
STEP 2.1: For root node C_0, do
(2.1.1) randomly select an integer K_0 among 1 and $P-1$.
(2.1.2) let $C_1, C_2, \ldots, C_k$ be immediate successors of C_0, then generate a NIP of degree k over the Galois field $GF(P)$, denoted as $NP(x) = (\alpha_k(x - x_{k-1})(x - x_{k-2}) \ldots (x - x_0) + \cdots + \alpha_1(x - x_0) + \alpha_0) \bmod P$, where $(a_0, x_0) = (0, K_0)$, and $(\alpha_i, x_i) = (t1_i, t2_i)$, $1 \leq i \leq k$.
(2.1.3) compute the secret key K_i of C_i as $K_i = F(NP(t2_i)) \bmod P$, $1 \leq i \leq k$.

STEP 2.2: For each non-root and non-leaf node C_i, do

(2.2.1) let $Ci_1, Ci_2, \ldots, Ci_k$ be immediate successors of C_i, then generate a NIP of degree k over the Galois field $GF(P)$, denoted as $NP(x) = (\alpha_k(x - x_{k-1})(x - x_{k-2}) \ldots (x - x_0) + \cdots + \alpha_1(x - x_0) + \alpha_0) \bmod P$, where $(\alpha_0, x_0) = (0, K_i)$, and $(\alpha_j, x_j) = (t1_{i_j}, t2_{i_j})$, $1 \leq j \leq k$.

(2.2.2) compute the secret key K_{i_j} of C_{i_j} as $K_{i_j} = F(NP(t2_{i_j})) \bmod P$, $1 \leq j \leq k$.

The following example illustrates how K_i's are assigned by the algorithm proposed above.

EXAMPLE 1. Consider the tree structure as depicted in Figure 1. Without loss of generality, let the prime number $P = 13$ and the pseudo one way function $F(x) = x^2 + 1 \pmod P$. By employing the algorithm Assignment, C_0 can assign public key pairs $(t1_i, t2_i)$ and generate K_i for each security class, $0 \leq i \leq 7$, as shown in Table 1.

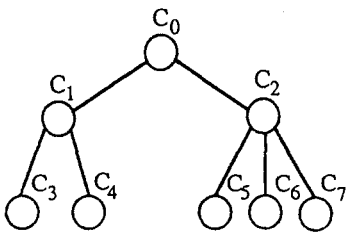

Figure 1. The tree structure.

Table 1. The public key pairs and secret keys in Figure 1.

	Public key pair $(t1_i, t2_i) = (\alpha_i, x_i)$	Secret key $K_i = F(NP(t2_i))$
C_0	—	1
C_1	(1,3)	5
C_2	(2,4)	4
C_3	(1,6)	2
C_4	(2,8)	5
C_5	(1,5)	2
C_6	(2,6)	11
C_7	(3,8)	10

Note that the security class C_i can compute the keys of the security classes below him, which is not an immediate successor, by repeatedly employing algorithm Derivation below.

ALGORITHM DERIVATION

INPUT: Secret key K_i.

OUTPUT: Secret key K_{i_j}.

STEP 1: Using Newton's interpolation method, reconstruct interpolating polynomial $NP(x) = (\alpha_k(x - x_{k-1})(x - x_{k-2}) \ldots (x - x_0) + \cdots + \alpha_1(x - x_0) + \alpha_0) \bmod P$, where $(\alpha_0, x_0) = (0, K_i)$, and $(\alpha_j, x_j) = (t1_{i_j}, t2_{i_j})$, $1 \leq j \leq k$.

STEP 2: Compute the secret key K_{i_j} as $K_{i_j} = F(NP(t2_{i_j})) \bmod P$, where $t2_{i_j}$ is the public key of C_{i_j}.

Now, a new node is inserted to the tree structure, say C_8. Figure 2 shows this condition.

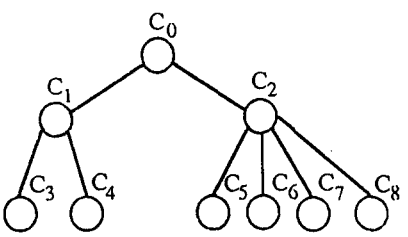

Figure 2. The new node C_8 is inserted into the system.

In our scheme, whenever a new node is inserted into the user hierarchy system, the corresponding secret key will be determined immediately without changing any previously defined secret keys and public key pairs. That is, by employing the algorithm Assignment, a new public key pair $(t1_8, t2_8) = (4, 7)$ is given and a secret key K_8 is computed as

$$NP(x) = 4(x-8)(x-6)(x-5)(x-4)+3(x-6)(x-5)(x-4)+2(x-5)(x-4)+1(x-4)+0) \bmod 13.$$

Thus, $NP(7) = (-24 + 18 + 12 + 3) \bmod 13 = 9$ and $K_6 = F(NP(7)) = (9^2 + 1) \bmod 13 = 4$.

4. THE SECURITY AND COMPUTATIONAL COMPLEXITY

In our scheme, each security class C_i can derive the secret keys of his immediate successors by using NIP and the predefined one way function F. However, it is impossible to reconstruct NIP by only knowing the public key pairs of C_i's immediate successors, because all the public key pairs of C_i's immediate successors can only construct an NIP', which has one less degree than that of the original NIP. Therefore, our scheme can provide security against two or more users at lower level of the system collaborating to derive a higher leveled key to which they are not entitled. In addition, each immediate successor C_{i_j} of C_i cannot reveal the root of NIP by using his secret key K_{i_j}, because the secret key is computed from the predefined one way function F. Hence, a user at lower level cannot derive from his own cryptographic key the keys of the other users above him in a tree structure. Hence, the correctness of our security enforcing scheme is established. Furthermore, it is easily seen that the computational complexity of our scheme heavily relies on the construction of NIP. However, Knuth [18] has proposed an efficient $0(k \log^2 k)$ time algorithm for NIP, where k is the number of immediate successors of the current security class. Therefore, our scheme only needs $O(n \log^2 n)$ time for practical implementation.

5. CONCLUDING REMARKS

In this paper, we have proposed a dynamic scheme to assign cryptographic keys in a tree structure for access control. It is ensured to prevent the cooperative attacks and effectively reduce the values of t_i's. Besides, no modification of the secret keys and public key pairs is needed when a new security class is inserted into the system or an old security class is deleted from the system, while almost all past access control schemes need. Hence, this scheme is feasible for assigning cryptographic keys in a tree structure.

REFERENCES

1. S.G. Akl and P.D. Taylor, Cryptographic solution to a problem of access control in a hierarchy, *ACM Transactions on Computer System* **1** (3), 239–247 (August 1983).
2. C.C. Chang, On the design of a key-lock-pair mechanism in information protection systems, *BIT* **26**, 410–417 (1986).
3. C.C. Chang, An information protection system scheme based upon number theory, *The Computer Journal* **30** (3), 249–253 (1987).
4. C.C. Chang, On the implementation of user hierarchy structure in information system, In *Proceedings of International Conference on Computer Software and Applications*, IEEE, Tokyo, Japan, 412–415, (October 1987).
5. R.W. Conway, W.L. Maxwell and H.L. Morgan, On the implementation of security measures in information systems, *Communication of ACM* **15** (4), 211–220 (1972).
6. G.S. Graham and P.L. Denning, Protection-principles and practices, In *Proc. Spring Jt. Computer Conference*, Vol. 40, AFIPS Press, Montvale, N.J., 417–429, (1972).
7. E. Gudes, The design of a cryptography-based secure file system, *IEEE Transactions on Software Engineering* (5), 411–419 (September 1980).
8. L. Harn and H.Y. Lin, A cryptographic key generation scheme for multilevel data security, *Computers & Security* **9**, 539–546 (1990).
9. J.K. Jan, A single-key access control scheme in information protection systems, *Information Sciences* **51**, 1–11 (1990).
10. J.K. Jan, C.C. Chang and S.J. Wang, A dynamic key-lock-pair access control scheme, *Computers & Security* **10** (2), 129–139 (1991).
11. C.H. Lin, R.C.T. Lee and C.C. Chang, A dynamic access control mechanism in information protection systems, *Journal of Information Science and Engineering* **6**, 25–35 (1990).

12. S.T. Mackinon, P.D. Taylor, H. Meijer and S.G. Akl, An optimal algorithm for assigning cryptographic keys to control access in a hierarchy, *IEEE Transactions on Computers* **C-34** (9), 797–802 (September 1985).
13. J.H. Saltzer and M.D. Schroeder, The protection of information in computer systems, In Working papers presented at the *Proc. IEEE*, 1278–1308, (September 1975).
14. M.L. Wu and T.Y. Hwang, Access control with single-key-lock, *IEEE Transactions On Software Engineering* **SE-10** (2), 185–191 (March 1984).
15. E. Horowitz and S. Sanni, *Fundamentals of Data Structures*, Computer Science Press, Rockvile, MD.
16. R.L. Burden, J.D. Faires and A.C. Reynolds, *Numerical Analysis*, 2nd ed., Prindle, Weber & Schmidt, Reading, MA, (1981).
17. M.K. Jain, S.R.K. Lyengar and R.K. Jain, *Numerical Methods for Scientific and Engineering Computation*, Wiley Eastern, New Delhi, (1985).
18. D.E. Knuth, *The Art of Computer Programming*, Vol. 2 (Seminumerical Algorithm), 2nd ed., Addison-Wesley, Reading, MA, (1980).