AN ASYMMETRIC CRYPTOGRAPHIC KEY ASSIGNMENT SCHEME FOR ACCESS CONTROL IN TOTALLY-ORDERED HIERARCHIES *†

Min-Shiang Hwang

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

January 12, 2001

*This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no.: NSC88-2213-E-324-002.

†Responsible for correspondence: Prof. Min-Shiang Hwang

AN ASYMMETRIC CRYPTOGRAPHIC KEY ASSIGNMENT SCHEME FOR ACCESS CONTROL IN TOTALLY-ORDERED HIERARCHIES

Abstract

Many methods based on cryptography have been proposed to solve the problem of access control in hierarchic structures. However, these schemes are only used in symmetric (or one-key) cryptosystems. In this article, we propose a new multilevel access control scheme for a totally-ordered hierarchy that can be used in asymmetric (or two-key) cryptosystem. It is well known that there are two main advantages in asymmetric cryptography: key distribution and authentication. Our scheme is, to our best knowledge, the first to be used as an asymmetric cryptosystem.

Keywords: Authentication; Cryptography; Data security.

C.R. Categories: E.3

1 INTRODUCTION

A multilevel access control scheme for a computer or database system is one in which each subject (e.g., a user, program, processor, etc.) is given a distinct clearance and each object (e.g., a file, a message, data, etc.) is assigned a security level. Subjects and objects are classified into a number of distinct security classes $C_1, C_2, \cdots, C_r$. The security classes form a partially-ordered set (poset, for short) hierarchy. In such a hierarchy, an object with a particular security class can be accessed only by subjects in the same or a higher security class.

In a symmetric or one-key multilevel access control scheme, each security class C_i is assigned a distinct secret key d_i for encrypting and decrypting objects in that class. The secret key owned by security class C_i can be derived only by a subject in the same or a higher security class. Thus, subjects in a higher security class can decipher objects in a lower or equal security class.

In an asymmetric or two-key multilevel access control scheme, each security class C_i has a distinct enciphering key e_i and a distinct deciphering key d_i for encrypting and decrypting objects in that class. Deciphering key d_i can be derived only by subjects in a higher security class. Thus, any subject can encipher the objects in C_i with e_i but only subjects in the same or a higher security class can decipher the objects in C_i. It is well known that there are two main advantages in asymmetric cryptography (i.e., disadvantages in symmetric cryptography): key distribution and authentication [4, 5].

Many methods have been proposed in the literature for the access control problem in a hierarchy [2, 3, 7, 8, 9, 10, 11, 12, 13, 17], but none of these schemes are used in asymmetric cryptosystems. Therefore they do not have the advantageous properties of public key cryptosystems [6]. In this article, we present an asymmetric scheme for access control in a totally-ordered hierarchy. Although the totally-ordered hierarchy is a special case of the partially-ordered hierarchy, it has many applications in the real world. For example, we often classify documents into top-secret, secret, confidential, and unclassified security classes in the order *top-secret* > *secret* > *confidential* > *unclassified* and classify users into Administrators (A), Programmers (P), and Ordinary users (O) in the order $A > P > O$ in database systems.

2 OUR ASYMMETRIC CRYPTOGRAPHY-BASED MULTILEVEL SCHEME

In this section we present an asymmetric cryptography-based multilevel scheme for access control in a totally-ordered hierarchy. The security of our scheme is based on the difficulty of factoring the product of two large primes. We assume that there is a central authority (CA, for short) in the system. The CA's task is to generate and distribute keys. In the key generation procedure, the CA executes the following steps for security class C_i:

Step 1: Choose two large primes P and Q. And then compute $m = P \cdot Q$, where " $\cdot$ " denotes a multiplication. P and Q are kept secret, and m is a public parameter.

Step 2: Choose enciphering key (e_i, m) and secret key s_i satisfying RSA cryptosystem [16] for all security class C_i. In other words, e_i, s_i, and m satisfy the relations $e_i s_i \bmod \phi(m) = 1$, where $\phi(\cdot)$ is the Euler's totient function. Although e_i and s_i use the same common modular, it is nothing to revealing s_i by common modular attack [14] because of s_i is olny keep secret by CA.

Step 3: Choose secret parameter β, $2 \leq \beta \leq \phi(m) - 1$ such that β and $\phi(m)$ are relatively prime.

Step 4: Choose a random secret parameter t, $2 \leq t \leq \phi(m) - 1$.

Step 5: Compute a secret parameter α such that $(\alpha\beta \bmod \phi(\phi(m)) = 1)$.

Step 6: Compute a secret parameter p_1, deciphering key d_1, and public parameter w_1 of security class C_1, the top-secret level, as follows.

$$\begin{cases} p_1 = \alpha^t \bmod \phi(\phi(m)), \\ d_1 = \beta^t \bmod \phi(\phi(m)), \\ w_1 = s_1^{p_1} \bmod \phi(m). \end{cases} \tag{1}$$

The other secret parameters, deciphering keys and public parameters are calculated as follows. For i from 2 to r (we assume that there are r security classes in the hierarchy),

$$\begin{cases} p_i = p_{i-1}^2 \bmod \phi(\phi(m)) = \alpha^{2^{L_i-1}t} \bmod \phi(\phi(m)), \\ d_i = d_{i-1}^2 \bmod \phi(\phi(m)) = \beta^{2^{L_i-1}t} \bmod \phi(\phi(m)), \\ w_i = s_i^{p_i} \bmod \phi(m) = s_i^{\alpha^{2^{L_i-1}t} \bmod \phi(\phi(m))} \bmod \phi(m), \end{cases} \tag{2}$$

where L_i is the level of C_i in the hierarchy. Each security class holds enciphering key (e_i, m), deciphering key d_i, and a public parameter w_i. s_i and p_i are no longer needed. They should be discarded, but never revealed. Publish (e_i, m) and w_i, and keep d_i secret.

Each user can encrypt confidential message in a lower or equal security class under enciphering key. Each user thus can decrypt the ciphertext in a lower or equal security class under his deciphering key and public parameter.

An example of our scheme is shown in Figure 1. If user 3 (u_3) belonging

C_1: $(e_1, m), s_1$
$p_1 = \alpha^t \bmod \phi(\phi(m)), d_1 = \beta^t \bmod \phi(\phi(m))$
$w_1 = s_1^{p_1} \bmod \phi(m)$

C_2: $(e_2, m), s_2$
$p_2 = \alpha^{2t} \bmod \phi(\phi(m)), d_2 = \beta^{2t} \bmod \phi(\phi(m))$
$w_2 = s_2^{p_2} \bmod \phi(m)$

C_3: $(e_3, m), s_3$
$p_3 = \alpha^{4t} \bmod \phi(\phi(m)), d_3 = \beta^{4t} \bmod \phi(\phi(m))$
$w_3 = s_3^{p_3} \bmod \phi(m)$

C_4: $(e_4, m), s_4$
$p_4 = \alpha^{8t} \bmod \phi(\phi(m)), d_4 = \beta^{8t} \bmod \phi(\phi(m))$
$w_4 = s_4^{p_4} \bmod \phi(m)$

Figure 1: An example of our scheme.

to security class C_3 wishes to transfer a document to user 4 (u_4) in C_4, he can encipher the document M with enciphering key (e_4, m) as

$$T = M^{e_4} \bmod m,$$

where T is the ciphertext of M under enciphering key (e_4, m). Thus, users in security class C_i, $C_i \geq C_4$, can retrieve and decipher the ciphertext. For example, users in security class C_1 can decipher the document by computing as follows:

$$T^{w_4^{d_1^{2(L_4-L_1)}}} \bmod m, \tag{3}$$

where L_i is the level of C_i and L_j is the level of C_j.

Next, we prove that the plaintext M can be got by computing Equation (3) as follows:

$$\begin{aligned}
& T^{w_4^{d_1^{2(L_4-L_1)}}} \bmod m, \\
= \;& M^{e_4 w_4^{d_1^8}} \bmod m, \\
= \;& M^{e_4 s_4^{(\alpha\beta)^{8t} \bmod \phi(\phi(m))} \bmod \phi(m)} \bmod m, \\
= \;& M^{e_4 s_4 \bmod \phi(m)} \bmod m, \\
= \;& M.
\end{aligned}$$

Since by Step 5 of our scheme, $(\alpha\beta)^{8t} \bmod \phi(\phi(m)) = 1$, the above equation holds.

Notes that any user in security class C_2, C_3, or C_4 does not keep any information of $\phi(\phi(m))$ and $\phi(m)$ in our scheme. Each security class only holds enciphering key (e_i, m), deciphering key d_i, and a public parameter w_i.

3 SECURITY ANALYSIS

Since the deciphering key d_i of a security class in our scheme is equal to the square of its ancestor's deciphering key, a user in a lower level security class cannot disclose its ancestor's deciphering key unless he or she is able to compute the square root (mod m) of her deciphering key. However, it has been suggested that computing the bth roots of $K^b \bmod m$ for any integer $b > 1$ is as difficult as factoring m [16], and this has been proven in [15] for the case of $b = 2$. Therefore, to break the deciphering key of our scheme is as difficult as factoring the product of two large primes.

Although p_i and the deciphering d_i use the same common modular, it is nothing to revealing d_i by common modular attack [14] because of p_i is keep secret.

In addition, Akl and Taylor proved that the following property is required to ensure security against the cooperation of two or more users at a lower level in a hierarchy [1, 2]:

$$\gcd_{C_j \not\geq C_i} PI_j \nmid PI_i, \tag{4}$$

where PI_i and PI_j are the public information of C_i and C_j, respectively. In our scheme, PI_i and PI_j are equal to 2^{L_i} and 2^{L_j}, respectively. Since $\gcd_{C_j \not\geq C_i} PI_j$ is equal to 2^{L_i+1}, property (4) is satisfied.

4 CONCLUSIONS

We have proposed a simple scheme for solving the multilevel key generation problem. The major merit of our scheme is its simplicity in terms of both the underlying idea and the algorithm for assigning secret keys. Our scheme can handle the keys of an asymmetric cryptosystem in a hierarchy, but previous schemes handle only symmetric cryptosystems. It is well known that there are two main advantages in asymmetric cryptography: key distribution and authentication. Our scheme is, to our best knowledge, the first to be used as an asymmetric cryptosystem.

Acknowledgements

We would like to thank professors Wen-Guey Tzeng and Wei-Pang Yang for helpful discussions.

References

[1] Akl, S.G. and Taylor, P.D. (1982). Cryptographic solution to a multilevel security problem, in: *Proceedings of Crypto'82*, pp.237–249.

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

[3] Chang, C.C., Hwang, R.J., and Wu, T.C. (1992). Cryptographic key assignment scheme for access control in a hierarchy, *Information Systems*, **17**, 243–247.

[4] Denning, D.E.R. (1982). *Cryptography and data security*, Addison-Wesley, Massachusetts.

[5] Diffie, W. (1980). The first ten years of public-key cryptography, *Proceedings of the IEEE*, **76**, 560–577.

[6] Diffie, W. and Hellman, M.E. (1976). New directions in cryptography, *IEEE Transactions on Information Theory*, **22**, 644–654.

[7] Harn, L. and Lin, H.Y. (1990). A cryptographic key generation scheme for multilevel data security, *Computers & Security*, **9**, 539–546.

[8] Hwang, M.S., Chang, C.C., and Yang, W.P. (1993). Modified chang-hwang-wu access control scheme, *IEE Electronics Letters*, **29**, 2095–2096.

[9] Hwang, M.S. (1997). A cryptographic key assignment scheme in a hierarchy for access control, To appear in *Mathematical and Computer Modelling*, **26**, 27–31.

[10] Hwang, M.S. (1999). An improvement of a dynamic cryptographic key assignment scheme in a tree hierarchy, *Computers & Mathematics with Applications*, **37**, 19-22.

[11] Hwang, M.S. (1999). An improvement of novel cryptographic key assignment scheme for dynamic access control in a hierarchy, *IEICE Transactions on Fundamentals*, **E82-A**, 548–550.

[12] M.S. Hwang. Extension of CHW cryptographic key assignment scheme in a hierarchy. Accepted and to appear in *IEE Proceedings Computers and Digital Techniques.*

[13] Mackinnon, S.J., Taylor, P.D., Meijer, H., and Akl, S.G. (1985). An optimal algorithm for assigning cryptographic keys to control access in a hierarchy, *IEEE Transactions on Computers*, **34**, 797–802.

[14] Moore, J.H. (1988). Protocol failures in cryptosystems, *Proceedings of IEEE*, **76**, 594–602.

[15] Rabin, M.O. (1979). Digitalized signatures and public-key functions as intractable as factorization, Technical Report MIT/LCS/TR-212, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Mass.

[16] Rivest, R.L., Shamir, A., and Adleman, L. (1978). A method for obtaining digital signatures and public key cryptosystems, *Communications of the ACM*, **21**, 120–126.

[17] Sandhu, R.S. (1988). Cryptographic implementation of a tree hierarchy for access control, *Information Processing Letters*, **27**, 95–98.