An Asymmetric Cryptographic Key Assignment Scheme for Access Control in Tree Structural Hierarchies

Debasis Giri and Parmeshwary Dayal Srivastava
(Corresponding author: Debasis Giri)

Department of Mathematics, Indian Institute of Technology, Kharagpur 721 302, India
(E-mail:{dgiri, pds}@maths.iitkgp.ernet.in)

(Received Oct. 08, 2005; revised and accepted Nov. 5, 2005 & Mar. 12, 2006)

Abstract

In a hierarchical structure, a user in a security class has access to information items of another class if and only if the former class is a predecessor of latter. Based upon cryptographic techniques, several schemes have been proposed for solving the problem of access control in hierarchical structures. In this paper, we propose a new scheme for an access control in tree structural hierarchies based on asymmetric cryptographic key assignment scheme. Further, our encryption and decryption procedures are based on asymmetric cryptographic technique. We show that proposed scheme requires less amount of storage space to store public parameters and also retains the same security level compared to the previous published schemes. Furthermore, our scheme achieves better generality compared to the Hwang's scheme.

Keywords: Access control, authentication, cryptography, data security

1 Introduction

In real life, hierarchical structures are used in many applications organizations like the military, government organizations, school systems, college systems, private corporations, computer network systems [16, 17, 19, 20], operating systems [10] and database management systems [5, 7, 8, 9], etc.

We consider an organizational structure in which the users and their own information items (e.g., a message, data, etc.) are divided into a number of disjoint set of security classes, say $C_1, C_2, \cdots, C_n$. We can define a binary relation $\leq$, which partially orders the set $C = \{C_1, C_2, \cdots, C_n\}$. In the partially ordered, $(C, \leq)$, $C_i \leq C_j$ means that C_i has security clearance lower than or equal to C_j. In other words, the users in C_j can access the encrypted information items held by the users in C_i. However, the converse is not permitted. Figure 1 shows an example of four-level hierarchial structure. Top level class possesses the greatest authority, and authority decreases with the increase in level. Thus, users in bottom level classes have the least authority. For the partially ordered set structure, $C_i \leq C_j$, C_i is called a successor of C_j, where as C_j is called a predecessor of C_i. If there does not exist C_k such that $C_i \leq C_k \leq C_j$, C_i is called an immediate successor of C_j, and C_j is called an immediate predecessor of C_i. If there does not exist C_i such that $C_i \leq C_j$, C_j is called a leaf security class. Without loss of generality, we identify the classes in a hierarchical system as follows. Let G be a set consists of integers $1, 2, \cdots, g$, i.e., $G = \{1, 2, \cdots, g\}$, where g is the degree of a tree structure. Let C_i be a class. Then, the immediate successors of C_i are represented by C_{i_j}, where $i_j = i \cdot g + j - 2$ for some $j \in G$ and i is the identity of the class C_i. Let us consider an example as follows. Assume that the user in the security class C_{15} in Figure 1 encrypts a message (information items) with her own encryption key e_{15}. Because of access control in a hierarchical structure, only the users in the security class C_{15} and her predecessors classes (i.e., C_5, C_2, C_1) can decrypt this encrypted message, whereas nobody else can decrypt this encrypted message.

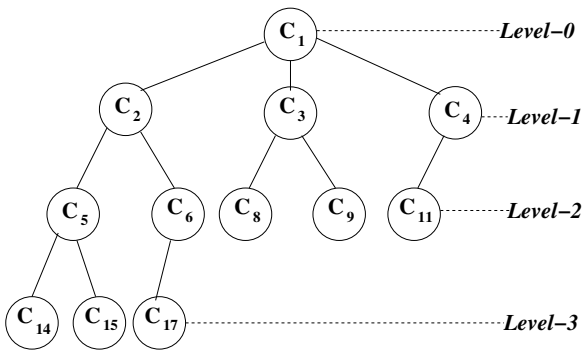

Figure 1: An example of a tree hierarchical structure

A straightforward access control scheme for poset hierarchy is to assign each security class with a key, and each class has the keys of all its successors. The information items belonging to a class is encrypted with the key assigned to that class. As a result, if a class encrypts the information items, its predecessors can only decrypt the encrypted information items. The drawback of such scheme is to store the keys in higher hierarchical classes. Several methods have been proposed in order to solve such type of problems based on the concept of the master key [4]. In 1983, Akl and Taylor [2] proposed a scheme based on symmetric key cryptosystem. Each security class C_i is assigned with a public parameter, PB_i and a secret key $K_i = K_0^{PB_i} \bmod N$, where N is widely separated secret pair of primes and K_0 is kept secret by the central authority (CA, for short). If $C_i \leq C_j$, PB_i/PB_j is an integer, C_j can derive the secret key, K_i as $K_i = K_0^{PB_i} = K_0^{PB_j \cdot (PB_i/PB_j)} = K_j^{(PB_i/PB_j)} \bmod N$ of the class C_i.

In 1985, Mackinnon et al. [18] proposed an improved algorithm for the Akl-Tayllor scheme based on top-down approach of poset hierarchy for reducing the value of public parameters. In 1988, Sandhu [24] introduced a cryptographic implementation of a tree hierarchy for access control based on one-way function. In 1990, Harn and Lin [11] proposed a scheme which is similar to the scheme of Akl-Taylor, but, it is based on bottom-up approach for key generation. In 1992 and 1993, both Chang et al. [3] and Liaw et al. [13, 14] proposed the scheme based on Newton's interpolation method and one-way function. The key generation procedures of the above schemes in such a way that higher level security class can derive the secret key of lower level security class using her secret key and the public parameters. In 2000, Hwang [12] proposed an access control scheme for a totally ordered hierarchy based on asymmetric cryptosystem. Recently, many related schemes have been proposed [15, 26]. In 2003, Lin-Hwang-Chang [15] proposed a scheme for access control, where each security class contains a secret key SK_i and derivation key DK_i which are kept secret by the class C_i. If $C_i \leq C_j$, the class C_j can derive the secret key of the class C_i using the derivation key DK_j and public parameters. In this scheme requires only small amount of storage space to store public parameters compared to the Akl-Taylor's [2]. In this paper, we propose a new scheme for access control in tree structural hierarchy based on asymmetric cryptosystem which is the generalization of Hwang's [12] proposed scheme. Besides, our scheme requires less amount of storage space to store public parameters. Moreover, our encryption and decryption techniques are based on asymmetric cryptographic technique.

In a multilevel access control scheme based on asymmetric cryptosystem, each security class C_i has a distinct encryption key e_i and a distinct decryption key d_i for encryption and decryption respectively. A user can encrypt the information items (message) in C_i with e_i. The only user in the same security class C_i can decrypt using the decryption key d_i and public parameters, whereas the users in the higher security classes can decrypt that encrypted information items using their own decryption key and public parameters. But, no one else can decrypt that encrypted information items.

The remainder of this paper is organized as follows. Section 2 gives a brief review of the Hwang's scheme. In Section 3, we describe our proposed scheme for access control in tree structural hierarchies. Section 4 shows the space and time complexity of our scheme. In Section 5, we discuss the security analysis. Section 6 shows the advantages of our scheme. In section 7, our scheme is compared with previous published schemes. Finally, Section 8 concludes the paper.

2 Review of the Hwang's Scheme

In this section, we now review briefly the Hwang's scheme [12].

In the key generation phase, for n security classes $C_1, C_2, \cdots, C_n$ in totally-ordered hierarchy, CA performs the following techniques to generate and distribute keys. At first CA chooses a large number N so that N is product of two large primes. Then CA chooses e_i so that e_i and $\phi(N)$ are relatively prime and computes $s_i = e_i^{-1} \bmod \phi(N)$, where $\phi(\cdot)$ represents the usual Euler's totient function. After that CA selects parameters β, $2 \leq \beta \leq \phi(\phi(N)) - 1$ and t, $2 \leq t \leq \phi(N) - 1$ such that $\gcd(\beta, \phi(\phi(N))) = 1$. CA computes $\alpha = \beta^{-1} \bmod \phi(\phi(N))$. Then, CA also computes p_i, d_i, w_i, where

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

and

$$\begin{cases} p_i = p_{i-1}^2 \bmod \phi(\phi(N)), \\ d_i = d_{i-1}^2 \bmod \phi(\phi(N)), \\ w_i = s_i^{p_i} \bmod \phi(N), \end{cases}$$

for $i = 2, 3, \cdots, n$. After that CA sends securely encryption key (e_i, N), decryption key d_i to the security class C_i, where (e_i, N) is public and d_i is kept secret by C_i. CA keeps w_i as public.

2.1 Encryption Technique

The encryption technique of this scheme is as follows. Let M be the message to be encrypted. Encrypted message T of $M(< N)$ for a user in a security class C_i is defined as

$$T = M^{e_i} \bmod N.$$

2.2 Decryption Technique

Let us assume that $C_i \leq C_j$. If a user in security class C_j wants to decrypt the message which is encrypted by a

user in security class C_i, the following is the technique of this scheme:

$$M = T^{w_i^{d_j^{2^{(L_i - L_j)}}}} \bmod N,$$

where L_i and L_j are the level of security classes C_i and C_j respectively, and M is the decrypted message.

3 Our Scheme

In this section, we present a new key assignment scheme for access control in a tree structural hierarchy based on asymmetric cryptosystem. We assume that there is a trusted CA in the system. The main purpose of CA is to generate keys and distribute them securely to the classes. We use the following notations for describing key generation procedure.

- C_i: A class with identity i.
- C_{i_j}: A successor class of a class C_i, where $i_j = i \cdot g + j - 2, j \in \{1, 2, \cdots, g\}$ and g is the degree of a tree. i_j is the identity of the class C_{i_j}.
- E_i, D_i, p_i, w_i, s_i: These parameters are assigned for the class C_i, which are kept secret by CA.
- e_i, d_i: Encryption key and decryption key for the class C_i respectively.
- $E_{i_j}, D_{i_j}, p_{i_j}, w_{i_j}, s_{i_j}$: These parameters are assigned for the class C_{i_j}, which is a successor class of C_i. These parameters are kept secret by CA.
- e_{i_j}, d_{i_j}: Encryption key and decryption key of the class C_{i_j}.
- SI_{ij}: SI_{ij} (secret information) is computed and kept secret by CA in between the classes C_i and C_j with $C_i \leq C_j$.
- SRI_{ij}: SRI_{ij} (secret relational information) is computed by CA in between the classes C_i and C_j with $C_i \leq C_j$ and is kept secret by C_j.

3.1 Key Generation Procedure

In this subsection, we discuss the procedure to generate keys for all classes.

Step 1: CA chooses a large number N, so that N is a product of two large primes which are widely separated and $\phi(\phi(N))$ has at least two large prime factors. $\phi(\cdot)$ is Euler's totient function. CA keeps N as public parameter.

Step 2: CA chooses a prime h ($h > 2$) and another number h_i distinct from h so that $2 \leq h \cdot h_i \leq \phi(N) - 1$ and $\gcd(hh_i, \phi(N)) = 1$. Then CA computes the distinct encryption key $e_i = hh_i$ and secret key s_i such that $e_i s_i = 1 \bmod \phi(N)$ for the security class C_i. Although e_i and s_i use the same common modular, it is not possible to derive s_i by common modular attack [21] because of the fact that s_i are kept secret only by CA.

Step 3: CA chooses a prime E_i so that $2 \leq E_i \leq \phi(\phi(N)) - 1$ and $\gcd(E_i, \phi(\phi(N))) = 1$. Then CA calculates the multiplicative inverse, D_i, for each E_i, where $D_i E_i = 1 \bmod \phi(\phi(N))$. E_i and D_i are kept secret by CA.

Step 4: CA chooses a secret parameter t, $2 \leq t \leq \phi(\phi(\phi(N))) - 1$.

Step 5: CA computes a secret parameters p_1, w_1, and decryption key d_1 of the security class C_1 as follows:

$$\begin{cases} p_1 = D_1^t \bmod \phi(\phi(N)), \\ d_1 = E_1^t \bmod \phi(\phi(N)), \\ w_1 = s_1^{p_1} \bmod \phi(N). \end{cases}$$

Let us consider CA has computed the secret parameters p_i, w_i and decryption key d_i for the class C_i. Let C_{i_j} be an immediate successor of the class C_i. The secret parameters p_{i_j}, w_{i_j} and decryption key d_{i_j} of the class C_{i_j} as follows:

$$\begin{aligned} p_{i_j} &= (D_{i_j} p_{\lceil \frac{i_j - 1}{g} \rceil})^2 \bmod \phi(\phi(N)) \\ &= (D_{i_j} p_i)^2 \bmod \phi(\phi(N)), \\ d_{i_j} &= (E_{i_j} d_{\lceil \frac{i_j - 1}{g} \rceil})^2 \bmod \phi(\phi(N)) \\ &= (E_{i_j} d_i)^2 \bmod \phi(\phi(N)), \\ w_{i_j} &= s_{i_j}^{p_{i_j}} \bmod \phi(N), \end{aligned}$$

where g is the degree of the tree structure.

Step 6: CA computes secret information SI_{ij} in between two same or different classes C_i and C_j with $C_i \leq C_j$ using the following algorithm:

SI_Function(i, j)

Step 6.1: Set $k := 1, SI := 1$;

Step 6.2: If $(i == j)$ then
Goto **Step 6.5**;

Step 6.3: If $(i > j)$
$SI := SI \cdot E_i^{2^k} \bmod \phi(\phi(N))$;
$i = \lceil \frac{i-1}{g} \rceil$;
$k := k + 1$;

Step 6.4: Go to **Step 6.2**;

Step 6.5: Return SI;

All SI_{ij} are kept secret by CA.

Step 7: Then CA computes secret relation information SRI_{ij} in between two same or different classes C_i and C_j with $C_i \leq C_j$ as follows:

$$SRI_{ij} = w_i^{SI_{ij}} \bmod \phi(N).$$

Then, CA transmits securely the decryption key d_i and encryption key (e_i, N), and secret relational information SRI_{ki} for all k with $C_k \leq C_i$ to the class C_i in the system. The class C_i keeps secret d_i and SRI_{ki} for all k with $C_k \leq C_i$. The encryption key (e_i, N) of each security classes C_i are published by CA. E_i, D_i, s_i, p_i, w_i and SI_{ki} for all k with $C_k \leq C_i$ are kept secret by CA.

3.2 Encryption Technique

We define our encryption technique as follows. Let M be the information items or message to be encrypted. Encrypted message T of M for a user in a security class C_i is defined as

$$T = M^{e_i} \bmod N.$$

3.3 Decryption Technique

Let us assume that $C_i \leq C_j$. If a user in security class C_j wants to decrypt this encrypted message which is encrypted by a user in security class C_i, we use the following technique.

$$M = T^{SRI_{ij}^{d_j^{2(L_i - L_j)}}} \bmod N,$$

where $L_i = \lceil \log_g((g-1)\cdot(\text{identity of the class } C_i + 1)) - 1 \rceil$ and L_i and L_j are the level of security classes C_i and C_j respectively, M is the decrypted message, and SRI_{ij} are secret relational information of the class C_j.

3.4 Correctness

In this subsection, we prove that plaintext M can be derived using our decryption technique. We have already shown that encrypted message by the security class C_{15} is decrypted by the security class C_2 because of the fact that the security class C_2 is in the higher level security class than C_{15}. The proof as follows:
Let $T = M^{e_{15}} \bmod N$, where T is the ciphertext encrypted by a user in the security class C_{15}. Then

$$M = T^{SRI_{ij}^{d_j^{2(L_i - L_j)}}} \bmod N,$$

where

$$\begin{aligned} L_{15} &= \lceil \log_3(2 \cdot 15 + 1) - 1 \rceil \\ &= 3, \\ L_2 &= \lceil \log_3(2 \cdot 2 + 1) - 1 \rceil \\ &= 1, \\ d_2 &= E_2^2 \cdot E_1^{2t} \bmod \phi(\phi(N)), \\ SI_{15,2} &= E_{15}^2 \cdot E_5^4 \bmod \phi(\phi(N)). \end{aligned}$$

Now,

$$\begin{aligned} & SRI_{15,2}^{d_2^{2(L_{15}-L_2)}} \\ = \; & w_{15}^{(d_2^{2(L_{15}-L_2)})SI_{15,2}} \\ = \; & s_{15}^{p_{15}\cdot(d_2^{2(L_{15}-L_2)})SI_{15,2}} \\ = \; & s_{15}^{(D_{15}^2\cdot D_5^4\cdot D_2^8\cdot D_1^{8t})\cdot(E_2^2\cdot E_1^{2t})^4\cdot(E_{15}^2\cdot E_5^4)} \\ = \; & s_{15} \bmod \phi(N), \end{aligned}$$

where $p_{15} = D_{15}^2 \cdot D_5^4 \cdot D_2^8 \cdot D_1^{8t} \bmod \phi(\phi(N))$.

Therefore,

$$\begin{aligned} & T^{SRI_{15,2}^{d_2^{2(L_{15}-L_2)}}} \bmod N \\ = \; & M^{e_{15}\cdot s_{15}} \bmod N \\ = \; & M. \end{aligned}$$

4 Storage Requirement and Computational Complexity

Storage Requirement:
Let us consider k be the number of successors of the class C_i. Then from the key generation procedure, the class C_i has to store $k+1$ secret relational information, where each secret relational information lies between 1 and $\phi(N)$ $(< N)$ and the decryption key d_i lies between 1 and $\phi(\phi(N))$ $(< N)$. Therefore, the storage requirement for storing the secret information (parameters) is the sum of storing $k+1$ secret relational information and one decryption key. Thus, the required storage for storing the secret information is $(k+2)\lceil \log_2 N \rceil$ bits for the class C_i. Let us assume that there are u classes in the hierarchical systems. So, the total number of public parameters, $\{e_i | i \in \{\text{identity of the classes in the hierarchy}\}\}$ and N is $u+1$. Also, each e_i lies between 1 and $\phi(N)$ $(< N)$. Therefore, the total amount of space required for storing the public parameters is $(u+1)\lceil \log_2 N \rceil$ bits.

Time Complexity:
The time requirement for encryption and decryption techniques in our scheme using the repeated square and multiply algorithm are described as follows.

Time Requirement for Encryption:
Since the encryption key lies between 1 and $\phi(N)$ ($\leq N$), the time requirement to encrypt a message is $O(\log_2^3(N))$ in terms of bit operations.

Time Requirement for Decryption:
Suppose C_i encrypts a message M, where encrypted message is T and C_j plans to decrypt this encrypted message. The total time required for decryption can be attributed to three basic stages.

1) Computation of $d_j^{2^{(L_i-L_j)}}$: The number of bit operations required is $O(2^{L_i-L_j}\log_2^2(N))$.

2) Computation of $SRI_{ij}^{d_j^{2^{(L_i-L_j)}}}$: The number of bit operations required is $O(N^{2^{L_i-L_j}}\log_2^2(N))$.

3) Computation of $T^{SRI_{ij}^{d_j^{2^{(L_i-L_j)}}}} \bmod N$: The number of bit operations required is $O(N^{2^{L_i-L_j}}\log_2^3(N))$.

Thus, in our scheme, computational time is $O(N^{2^L}\log_2^3(N))$ in terms of bit operations, where $L = L_i - L_j$. So, computational time required for encryption and decryption of our scheme are same as the Hwang's scheme.

5 Security Analysis

In our proposed scheme, the decryption key d_i of a security class C_i is equal to the square of multiplication of the parameter E_i, which is kept secret by CA and its immediate predecessor's decryption key $d_{\lceil\frac{i-1}{g}\rceil}$. So, a user in a lower level security class C_i can derive its predecessor's decryption key unless that lower level class is able to compute the square root $\bmod \phi(\phi(N))$ of her decryption key as well as to compute the D_i which is the inverse of E_i. Since N is product of two large primes. So, it is difficult to compute $\phi(N)$ from N. Hence, it is also difficult to compute $\phi(\phi(N))$ from N. Also, it is known that the problem to compute n-th root of $x^n \bmod m$ for any integer $n \geq 2$ is as difficult as factoring m [23], where m is product of two large primes and this has been proven in [22] for the case of $n = 2$. Again, $\phi(\phi(N)$ has at least two large prime factors. As a result, it is hard to compute square root $\bmod \phi(\phi(N))$. Further, E_i and D_i are kept by CA. So, it is hard to compute D_i or D_i^2 from secret relational information of C_i. Therefore, in our scheme, it is difficult to compute the decryption key of upper level class by a class of lower level class is as difficult as factoring the product of two large primes.

Collaboration Attacks:
Collaboration attack is the case when two or more security classes at the lower level in the hierarchy wish to derive the decryption key of their predecessor class. Let C_i and C_j be the immediate successors of the class C_k. The decryption keys of C_i, C_j, and C_k are $d_i(= (E_i d_k)^2 \bmod \phi(\phi(N)))$, $d_j(= (E_j d_k)^2 \bmod \phi(\phi(N)))$ and d_k respectively. Let us assume that C_i and C_j compromise their d_i, d_j and their secret relational information SRI. Because of the factorization problem, it is hard to compute $\phi(\phi(N))$ from N. Again, it is difficult to compute D_i^2 (inverse of E_i^2) or D_j^2 (inverse of E_j^2) from SRI as well as it is also difficult to compute the square root of $d_k^2 \bmod \phi(\phi(N))$. Thus, it is hard to compute d_k from d_i, d_j and SRI. Hence our scheme is secure against such type of attacks.

Common Subordinate Attacks:
This is the case when the subordinate class C_k is accessible by two or more predecessor classes C_i and C_j. Let us consider $C_k \leq C_i \leq C_j$, where C_k and C_j are the immediate successor and predecessor of the class C_i respectively. Let us assume that C_i and C_k compromise their decryption keys and their secret relational information SRI. $d_i\ (= (E_i d_j)^2 \bmod \phi(\phi(N)))$ and $d_k\ (= (E_k d_j)^2 = E_k^2 E_i^4 d_j^4 \bmod \phi(\phi(N)))$ are the decryption keys of the classes C_i and C_k respectively. But, in our proposed scheme, it is difficult to compute d_j using d_i, d_k and secret relational information SRI of the classes C_i and C_k because it is difficult to compute n-th root $(n = 4)$ of $d_j^n \bmod \phi(\phi(N))$ for any integer $n \geq 2$. Further, it is difficult to compute D_i^2 as well as D_k^2 by a user in C_i is same as in collaboration attacks. Therefore, in our scheme, it is difficult to compute d_j using d_i, d_k and SRI by a user in C_i. As a result, proposed scheme is secure against such type of attacks.

Common Modulus Attacks:
There are two types of common modulus attacks.

1) The first type of common modulus attacks uses the same message M and same modulus N for two different encryption keys (public keys) e_1 and e_2. Then, $T_1 = M^{e_1} \bmod N$ and $T_2 = M^{e_2} \bmod N$. If e_1 and e_2 are relatively prime, there exist integers x and y such that $xe_1 + ye_2 = 1$. In this case, message M can be retrieved by the following technique [21, 25]:

$$\begin{aligned} T_1^x(T_2^y) \bmod N &= (M^{e_1})^x (M^{e_2})^y \\ &= M^{xe_1+ye_2} \\ &= M. \end{aligned}$$

But, in our proposed scheme, adversary can calculate $\gcd(e_i, e_j) = h$, where $h > 2$. As a result, it is difficult to compute h-th root of $M^h \bmod N$ without factoring to N. So, our scheme is secure against this type of attacks.

2) The second type of common modulus attacks [6, 21] is that a user in a class can use her own encryption key (public key) e_2 and decryption key d_2 together to retrieve the decryption key d_1 of another user of a class C_1 using encryption key e_1. The user first finds the gcd of e_1 and $e_2 d_2 - 1$ by the Euclidean algorithm. Let $u = \gcd(e_1, e_2 d_2 - 1)$. Then, the user finds v such that $v = (e_2 d_2 - 1)/u$. Since u divides e_1 and $\gcd(e_1, \phi(m)) = 1$, u must be relatively prime to $\phi(m)$. As $e_2 d_2 - 1 = 0 \bmod \phi(m)$ and $uv = e_2 d_2 - 1$, uv is a multiple of $\phi(m)$. As a result, v must be a multiple of $\phi(m)$. Since v is relatively prime to e_1, there exist integers x and y such that the relation $xv + ye_1 = 1$ holds. Therefore, $ye_1 = 1 \bmod \phi(m)$. Since v is multiple of $\phi(m)$. Thus, $y = d_1$. Hence,

the user in a security class can derive the decryption key d_1 of another user.

Let us consider our scheme. Assume that a user in a security class C_i wants to retrieve decryption key d_j of another user in a class C_j. A user in a class knows encryption key e_i, decryption key d_i of itself and encryption key e_j of C_j. But, encryption and decryption keys are on different modulus. So, C_i have to compute p_i from d_i, where p_i is the inverse of d_i. As it is hard to compute $\phi(\phi(N))$ from N, it is difficult to compute p_i from d_i by the class C_i. Hence, a user in C_i cannot derive d_j from e_i, d_i and e_j. As a result, our scheme is secure against this type common modulus attacks.

6 Advantages

In this section, we discuss the various kind of advantages achieved from our proposed scheme. The following advantages are as follows:

- Encryption and decryption techniques are based on asymmetric cryptosystem.
- The security is equivalent as RSA cryptosystem.
- Key generation procedure is based on asymmetric cryptosystem which is done by CA. So, it supports authentication.
- This scheme is secure against all possible attacks.
- No class can derive the decryption key of other class. An encrypted message of a class can be decrypted by the class itself and its predecessor classes. But, the reverse is not true.

7 Comparison

In this section, we compare our method with previous published schemes.

1) Our proposed scheme provides a hierarchical access control based on asymmetric key assignment in tree (non linear) hierarchical structure, whereas the M.S. Hwang's scheme provides totally order (linear order) hierarchical structure for access control although both schemes provide encryption and decryption techniques based on asymmetric cryptosystem.

2) In our scheme, the size of public parameters only depends on the magnitude of N and does not depend on the number of security classes in the hierarchical system. As a result, our scheme may be applicable even if the number of security classes is more. On the other hand, if the number of security classes is large, the Akl and Taylor [2], Mackinnon et al. [18] and Harn and Lin [11] scheme cannot be applicable.

8 Conclusion

In this paper, we have proposed a new scheme for solving the multilevel key generation technique. Our scheme is based on asymmetric cryptosystem for access control of information items in an organization. In fact, this scheme does not require large amount of storage space to store public parameters. Our scheme also provides encryption and decryption techniques using asymmetric cryptosystem. Furthermore, our proposed scheme retains the same security level compared to the schemes previously published.

Hence, we conclude that our scheme is a novel scheme to be used as asymmetric cryptosystem in tree structured access control hierarchies for key generation as well as for encryption and decryption.

References

[1] S. G. Akl and P. D. Taylor, "Cryptographic solution to a multilevel security problem", in *Proceding of Crypto'82*, pp. 237-249, 1982.

[2] 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, no. 2, pp 239-248, 1983.

[3] 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.

[4] G. C. Chick and S. E. Tavares, "Flexible access control with master keys", in *Advances in Cryptology (CRYPTO'89)*, pp. 316-322, 1990.

[5] G. I. Davida, D. L. Wells, and J. B. Kam, "A database encryption system with subkeys", *ACM Transactions on Database Systems*, vol. 6, no. 2, pp 312-328, 1981.

[6] J. M. DeLaurentis, "A forther weakness in the common modulus protocol for RSA cryptosystem", *Cryptologia*, vol. 8, no. 3, pp. 253-259, 1984.

[7] D. E. Denning, S. G. Akl, M. Morgenstern, P. G. Neumann, R. R. Schell, and M. Heckman, "Views for multilevel database security", in *Proceeding of the IEEE Symposium on Security and Privacy, Oakland*, pp. 156-172, 1986.

[8] D. E. Denning, *Cryptography and Data Security*. Massachusetts: Addison-Wesley, 1983.

[9] D. E. Denning, "Cryptographic checksums for multilevel database security", in *Proceeding of the IEEE Symposium on Security and Privacy, Oakland*, pp. 52-61, 1984.

[10] L. J. Fraim, "SCOMP: a solution to the multilevel security problem", *IEEE Computer*, vol. 16, no.7, pp. 26-34, 1983.

[11] L. Harn and H. Y. Lin, "A cryptographic key generation scheme for multilevel data security", *Computers and Security*, vol. 9, no. 6, pp. 539-546, 1990.

[12] M. S. Hwang, "An asymmetric cryptographic key assignment scheme for access control in totally-ordered hierarchies", *International Journal Computer Mathematics*, vol. 73, pp. 463-468, 2000.
[13] H. T. Liaw, and C. L. Lei, "An optimal algorithm to assign cryptographic keys in a tree structure for access control", BIT 33, pp. 46-56, 1993.
[14] H. T. Liaw, S. J. Wang, and C. L. Lei, "An dynamic cryptographic key assignment scheme in a tree structure", *Computers and Mathematics with Applications*, vol. 25, no. 6, pp. 109-114, 1993.
[15] I. C. Lin, M. S. Hwang, and C. C. Chang, "A new key assignment scheme for enforcing complicated access control policies in hierarchy", *Future Generation Computer Systems*, vol. 19, no. 4, pp. 457-462, 2003.
[16] W. P. Lu and M. K. Sundareshan, "A model for multilavel security in computer networks", in *Proceedings of the INFOCOM 1988*, pp. 1095-1104, New Orleans, LA, 1988.
[17] W. P. Lu and M. K. Sundareshan, "Enhanced protocols for hierarchical encryption key management for secure communication in internet environments", *IEEE transactions on communications*,vol. 40, no. 4, pp. 658-660, 1992.
[18] S. J. Mackinnon, 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*, vol. 34, no. 9, pp. 797-802, 1985.
[19] D. McCullough, "Specifications for multi-level security and a hook-up property", in *Proceeding of the IEEE Symposium on Security and Privacy*, pp. 161-166, 1987.
[20] J. McHugh and A. P. Moore, "A security policy and formal top level specification for a multi-level secure local area network", in *Proceeding of the IEEE Symposium on Security and Privacy*, pp. 34-39, 1986.
[21] J. H. Moore, "Protocol failures in cryposystems", *Proceedings of IEEE*, vol. 76, pp. 594-602, 1988.
[22] M. O. Rabin, "Digitalized signatures and public-key functions as intractable as factorization", *Technical Report MIT/LCS/TR-212, Laboratory for Computer Science, Massaachusetts Institute of Technology, Cambridge, Mass*, 1979.
[23] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaning digital signatures and public key cryptosystems", *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.
[24] R. S. Sandhu, "Cryptographic implimentation of a tree hierarchy for access control", *Information Processing Letters*, vol. 27, pp. 95-98, 1988.
[25] B. Schneier, *Applied Cryptography*, 2nd ed., J. Wiley and Sons, New York, 1996.
[26] T. C. Wu and C. C. Chang, "Cryptographic key assignment scheme for hierarchical access control", *International journal of computer systems science and engineering*, Vol. 16, no. 1, pp. 25-28, 2001.

Debasis Giri received his M.Sc. degree in Mathematics from the Indian Institute of Technology, Kharagpur 721 302, India in 1998, and his M.Tech. degree in Computer Science and Data Processing from the same institute in 2001. He is now working toward the Ph.D. degree from the Indian Institute of Technology, Kharagpur 721 302, India. Before joining the Ph.D. program, he worked as a lecturer in the department of Computer Science and Engineering of Haldia Institute of Technology, West Bengal, India from March, 2001 to January, 2004. His current research interests include cryptography, network security and information security.

Parmeshwary Dayal Srivastava received his M.Sc. degree in Mathematics from Kanpur University, Kanpur (U.P.), India in 1975 and Ph.D. in Mathematics from Indian institute of Technology, Kanpur (U.P.), India in 1980. Dr. Srivastava joined as Faculty in the department of Mathematics, I.I.T. Kharagpur (India) in May, 1980. During his 26 years of teaching, he taught various courses of pure & Applied Mathematics such as Real Analysis, Complex Analysis, Algebra, Measure theory, Numerical Analysis etc. to UG & PG students at IIT, Kharagpur. He has published more than 35 papers in a journal of International repute. He is referee of Indian Journal of Pure & Appl. Maths. (India); Demonstratio Mathematica (Warsa, Poland); Soochow J. Mathematics (China); Tamkang J. Mathematics (China); Bull. National Metallurgical Lab. (CSIR) Jamshedpur (India); ISTAM, IIT Kharagpur (India); J. Natural Sciences & Mathematics (Pakistan); Journal of Orissa Mathematical Society (India) and reviewer for Mathematical Review. Professor Srivastava is the life member of Indian Mathematical Society, Allahabad (India) & Indian Academy of Social Science, Allahabad (India). Presently, Dr. Srivastava is Professor of Mathematics at I.I.T. Kharagpur (India). His current research interests are Functional Analysis and Cryptography & Network Security.