2005

A Cryptographic Solution for General Access Control

Y. Kong
University of Wollongong, yk18@uow.edu.au

Jennifer Seberry
University of Wollongong, jennie@uow.edu.au

J. R. Getta
University of Wollongong, jrg@uow.edu.au

Ping Yu
University of Wollongong, ping@uow.edu.au

Publication Details

This article was originally published as: Kong, Y, Seberry, J, Getta, JR & Yu, P, A Cryptographic Solution for General Access Control, Proceedings of the 8th Information Security Conference (ICS 2005), Singapore, 20-23 September 2005, Lecture Notes in Computer Science 3650, Springer-Verlag, 2005, 461-473. The original publication is available " >here through Springerlink.

A Cryptographic Solution for General Access Control

Abstract
As one of the most popular information safeguarding mechanisms, access control is widely deployed in information systems. However, access control approach suffers from a tough problem, i.e. system administrators must be unconditionally trusted. Cryptographic substitutes have been developed to solve the above problem. In particular, hierarchical encryption, as an alternate solution of access control in a hierarchy, has been intensively studied. In this paper, we propose a cryptographic solution for general access control based on Chinese Remainder Theorem. Our solution has two categories: data based solution and key based solution. In contrast to the most recent hierarchical encryption system: Ray, Ray and Narasimhamurthi's system [1], our solution is more efficient, secure and flexible. Moreover, we introduce an efficient mechanism for authorization alterations. This paper ends with a set of experimental results that support our research.

Keywords
Chinese Remainder Theorem, Hierarchical Encryption

Disciplines
Physical Sciences and Mathematics

Publication Details
This article was originally published as: Kong, Y, Seberry, J, Getta, JR & Yu, P, A Cryptographic Solution for General Access Control, Proceedings of the 8th Information Security Conference (ICS 2005), Singapore, 20-23 September 2005, Lecture Notes in Computer Science 3650, Springer-Verlag, 2005, 461-473. The original publication is available " >here through Springerlink.

A Cryptographic Solution for General Access Control

Yibing Kong, Jennifer Seberry, Janusz R. Getta, and Ping Yu

School of Information Technology and Computer Science,
University of Wollongong,
Wollongong, NSW, Australia
{Yk18, Jennie, Jrg, Ping}@uow.edu.au

Abstract. As one of the most popular information safeguarding mechanisms, access control is widely deployed in information systems. However, access control approach suffers from a tough problem, i.e. system administrators must be unconditionally trusted. Cryptographic substitutes have been developed to solve the above problem. In particular, hierarchical encryption, as an alternate solution of access control in a hierarchy, has been intensively studied. In this paper, we propose a cryptographic solution for general access control based on Chinese Remainder Theorem. Our solution has two categories: data based solution and key based solution. In contrast to the most recent hierarchical encryption system: Ray, Ray and Narasimhamurthi's system [1], our solution is more efficient, secure and flexible. Moreover, we introduce an efficient mechanism for authorization alterations. This paper ends with a set of experimental results that support our research.

Keywords: Chinese Remainder Theorem, Hierarchical Encryption

1 Introduction

As one of the most popular information safeguarding mechanisms, access control is widely deployed in information systems. Great efforts have been made in this area over decades. Traditional access control has been replaced by more flexible and powerful systems, e.g. *Role-Based Access Control* (RBAC) [2] and *Flexible Authorization Framework* (FAF) [3]. However, in access control systems, unconditional trust in system administrators is always a potential threat to information security.

In order to overcome this threat, *hierarchical encryption* is developed as an alternate approach of access control. By using hierarchical encryption, all information in an information system is encrypted in a way such that data encrypted by a lower level security class can be decrypted by a higher level security class. The idea of hierarchical encryption is first proposed by Akl and Taylor [4, 5] in the early 1980s. Since then on more research work [1, 6–11] has been dedicated to this area. Ray, Ray and Narasimhamurthi's system [1] (*RRN* system) , to

our best knowledge, is the most recent development in this area. Compared to previous solutions, *RRN* system is a solution for general access control. That is, besides supporting access control policies following the hierarchical structure of an organization, *RRN* system also supports access control policies that do not follow the hierarchical structure. Furthermore, *RRN* is simple and can be easily incorporated in existing systems. However, *RRN* system has some disadvantages (e.g. lack of efficiency); this issue will be further discussed in section 3.

In this paper, we propose a cryptographic solution aiming at general access control, which performs much better than *RRN*. Our solution is based on Chinese Remainder Theorem (CRT) and has two categories: *data based solution* and *key based solution*. Assume a data item is to be shared with k sharers. In the data based solution, this data item is first encrypted by k sharers' public keys, respectively; then these k individual ciphertexts are combined by CRT. As a result, the final share ciphertext is k times bigger than the data item. In the key based solution, the data item is first encrypted by a symmetric key to produce a data ciphertext. Next, this symmetric key is encrypted by k sharers' public keys, respectively. Finally, these k individual ciphertexts are combined by CRT to produce a symmetric key share ciphertext. The data ciphertext and the symmetric key share ciphertext are concatenated and shared with those k sharers. The performance and security analysis shows that our solution is more efficient and secure than *RRN*. Moreover, in our solution, authorization alterations are efficiently supported. This paper ends with a set of experimental results that support our research.

The rest part of this paper is organized as follows. Section 2 introduces the fundamental knowledge of our solution. *RRN* system is briefly described in section 3. We propose a data based approach in section 4 and a key based approach in section 5. Section 6 depicts our experimental results. Section 7 concludes this paper.

2 Backgrounds

In this section, we will introduce the background knowledge on which our solution is based.

Theorem 1. *Chinese Remainder Theorem:*

If the integers $n_1, n_2, ..., n_k$ are pairwise relatively prime, then the system of simultaneous congruences

$$\begin{aligned} x &\equiv a_1 \ mod \ n_1 \ . \\ x &\equiv a_2 \ mod \ n_2 \ . \\ &... \\ x &\equiv a_k \ mod \ n_k \ . \end{aligned}$$

has a unique solution x, such that $0 \leq x < n = n_1 n_2 ... n_k$.

We call $n_1, n_2, ..., n_k$ the *CRT moduli* and x the *CRT solution*. The proof of CRT is available in most number theory books, e.g. [12]. *Garner's algorithm* is an efficient method for determining CRT solutions. This algorithm is listed as follows (For further details, please refer to Chapter 14.5 of [13]).

Algorithm: Garner's algorithm for CRT

`INPUT` : a positive integer $n = \prod_{i=1}^{k} n_i > 1$, with $\gcd(n_i, n_j) = 1$ for all $i \neq j$, and a modular representation $a(x) = (a_1, a_2, ..., a_k)$ of x for the n_i.

`OUTPUT` : the integer x in radix b representation.

1. For i from 2 to k do the following:
   1.1 $C_i \leftarrow 1$.
   1.2 For j from 1 to $(i-1)$ do the following:
   $u \leftarrow n_j^{-1} \bmod n_i$.
   $C_i \leftarrow u \cdot C_i \bmod n_i$.
2. $u \leftarrow a_1$, $x \leftarrow u$.
3. For i from 2 to k do the following:
   $u \leftarrow (a_i - x) \cdot C_i \bmod n_i$, $x \leftarrow x + u \cdot \prod_{j=1}^{i-1} n_j$.
4. Return(x).

The RSA algorithm [14] contains three parts: *key generation*, *encryption* and *decryption*. Key generation works as follows: find a *modulus* n (n is a product of two large primes) and choose a number e (e is a number less than n and relatively prime to $\phi(n)$, where $\phi(n)$ is the *Euler's totient function*). Find another number d such that $ed \equiv 1 \bmod \phi(n)$. The value e and d are called the *public* and *private exponents*, respectively. The public key K is the pair (e, n); the private key K^{-1} is the pair (d, n). The encryption of a message m with the public key $K = (e, n)$, denoted by $E_K(m)$, is defined as:

$$c = E_K(m) = m^e \bmod n \ .$$

where c is the ciphertext produced by the encryption algorithm E. The decryption of a ciphertext c with the private key $K^{-1} = (d, n)$, denoted by $D_{K^{-1}}(c)$, is defined as:

$$m = D_{K^{-1}}(c) = c^d \bmod n \ .$$

where m is the plaintext recovered by the decryption algorithm D.

3 *RRN* System

RRN system is a RSA based cryptosystem, which can be used not only for access control in a hierarchy but also for general cases. *RRN* system is based on the following principles [1].

Definition 1. Two RSA encryption keys $K_1 = (e_1, n_1)$ and $K_2 = (e_2, n_2)$ are said to be *compatible* if $e_1 = e_2$ and n_1 and n_2 are relatively prime.

Definition 2. For two compatible keys $K_1 = (e, n_1)$ and $K_2 = (e, n_2)$, their *product key*, $K_1 \times K_2$, is defined as $(e, n_1 n_2)$; K_1 and K_2 are called *factor keys* of the product key $K_1 \times K_2$.

Theorem 2. *For any two messages m and $\hat{m}$, such that $m, \hat{m} < n_1, n_2$,*

$$E_{K_1 \times K_2}(m) \equiv E_{K_1}(\hat{m}) \bmod n_1, \text{ if and only if } m = \hat{m} .$$
$$E_{K_1 \times K_2}(m) \equiv E_{K_2}(\hat{m}) \bmod n_2, \text{ if and only if } m = \hat{m} .$$

where $K_1 = (e, n_1)$, $K_2 = (e, n_2)$ and $K_1 \times K_2 = (e, n_1 n_2)$.

We call the ciphertext generated by a factor key (K_1 or K_2) *individual ciphertext* and the ciphertext generated by their product key ($K_1 \times K_2$) *share ciphertext*. Theorem 2 states that an individual ciphertext can be easily derived from its share ciphertext. Therefore, a message encrypted by a product key can be recovered by any of its factor keys' corresponding private keys. We will omit the proof of theorem 2. For details, please refer to Section 4 of Ray, Ray and Narasimhamurthi's paper [1].

In a *RRN* system, the personnel in an organization are organized in a hierarchical structure, which can be represented as a partially ordered set (poset), $(L, <)$. L is the set of *levels* of the organization and $<$ is the *dominance relation* between the levels. For each level $L_i \in L$, there is a pair of RSA keys assigned: $K_{L_i} = (e, n_{L_i})$, $K_{L_i}^{-1} = (d_{L_i}, n_{L_i})$ such that all RSA public keys in the system are compatible. Moreover, in order to enforce the access control in this hierarchy, a pair of *default keys* is used. The *default encryption key* for L_i is the product key of all its ancestors' public keys and its public key K_{L_i}; the *default decryption key* of L_i is its private key $K_{L_i}^{-1}$. In such a way, a message encrypted by L_i's default encryption key can be decrypted by L_i and its ancestors. *RRN* system also supports general cases of access control where *customized encryption keys* are used. Advantages of *RRN* system can be summarized as: supporting for general cases of access control, easily incorporated in existing systems, mutual access awareness and protecting for data consistency [1]. However, many problems remain unsolved.

- *RRN* system is strictly based on RSA cryptosystem, which restricts its application in a wide range of systems.
- *RRN* system is inefficient.
  - Generally, the modulus of a product key is a huge number (product of many moduli); it is time-consuming to perform RSA encryption on it.
  - For a message m, whenever its group of authorized users changes (e.g. a new user is granted to access m), *RRN* must re-encrypt m by using a newly generated encryption key.
- The sharing of the RSA public exponent e opens a potential security hole to attackers.
- Share ciphertext size increases proportionally as the number of sharers increases. Although this fact has been neglected in [1], it is of great importance if original data size is big or numerous sharers are involved.

4 A Data Based Solution

4.1 Overview

One popular way of enforcing access control is by means of *Access Control Lists* (ACLs). Each data is associated with an ACL, on which its authorized users/groups and corresponding access modes are listed. By looking at an ACL, it is easy to determine who is allowed to do what on the data associated with it. ACL covers the general cases of access control. For example, it supports hierarchical access control. If we generate ACLs according to the hierarchical structure of an organization, then hierarchical access control can be enforced. That is, a data owner and all his/her ancestors are listed on his/her data's ACLs.

From cryptographic perspective, to enforce general access control, each data must be encrypted such that only subjects on its ACL have ability to decrypt the data. One straightforward approach exists to solve this problem [1]. Assume each subject is assigned with a pair of keys: a public key and a private key. To share a message m with k subjects: $s_1, s_2, ..., s_k$, for each subject $s_i \in \{s_1, s_2, ..., s_k\}$, m is encrypted by s_i's public key. Together with a ciphertext for its owner, m is encrypted $k+1$ times. The system keeps these $k+1$ ciphertexts for sharing a single message m. One negative aspect of this approach has been identified, i.e. storing multiple copies of encrypted data (*individual ciphertexts*) can be a source of inconsistency [1]. In *RRN* system, to share the same data m, data owner calculates its *share ciphertext*. Instead of multiple individual ciphertexts, only one share ciphertext is kept. *RRN* system does not lead to inconsistencies but is more computation intensive.

Based on the above straightforward approach, if there exists an efficient method that converts multiple individual ciphertexts to one share ciphertext, then a new approach of enforcing general access control is established with the advantages of both efficiency and consistency. We have discovered such a method: Chinese Remainder Theorem (CRT). CRT provides a way of mapping a number $x \in \mathbb{Z}_n$ ($\mathbb{Z}_n$ is the set of nonnegative integers less than n) to a series of k numbers $a_i \in \mathbb{Z}_{n_i}$, where $1 \leq i \leq k$, $n = n_1 n_2 ... n_k$ and $n_1, n_2, ..., n_k$ are pairwise relatively prime. The mapping is a one-to-one correspondence (called a *bijection*) between $\mathbb{Z}_n$ and the Cartesian product $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times ... \times \mathbb{Z}_{n_k}$ [15]. This property of CRT enables it to construct a share ciphertext from a series of individual ciphertexts.

4.2 System Elements

Our data based solution consists of the following elements:

- A set of subjects $S = \{s_1, s_2, ..., s_\ell\}$, where a subject is either a user or a group.
- A *public key cryptosystem* that consists of three functions:
  (a) A *Key Generation* function KG: $\forall s_i \in S$, KG generates a pair of keys: a *public key* K_{s_i} and its corresponding *private key* $K_{s_i}^{-1}$.
  (b) An *Encryption* function E: $c = E_K(m)$, where c means *ciphertext*, m means *message* and K means *public key* (*encryption key*).

(c) A *Decryption* function D: $m = D_{K^{-1}}(c)$, where K^{-1} means *private key* (*decryption key*).

- A *Modulus Generator* MG: $\forall s_i \in S$, MG generates a modulus n_{s_i}, such that $n_{s_1}, n_{s_2}, ..., n_{s_\ell}$ are pairwise relatively prime. Please note, these moduli are publicly known and will be used as the CRT moduli.
- A *Shared DataBase* (or file system) SDB that stores shared data.

4.3 Cryptographic Access Control

Our data based solution is depicted by a scenario as follows. Assume that a subject s_i wants to share a message m with k subjects $s_{i_1}, s_{i_2}, ..., s_{i_k} \in S$, s_i performs the following operations (for simplicity, we assume that $m < n_{s_1}, n_{s_2}, ..., n_{s_\ell}$; for a longer message, encryption can be performed block by block):

A1. First, s_i computes k individual ciphertexts, i.e. $\forall s_j \in \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$, $E_{K_{s_j}}(m)$ is calculated;

A2. Second, s_i uses Garner's algorithm (see section 2) to calculate the *CRT solution* x, $0 \leq x < n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}$, such that x satisfies the following k simultaneous congruences:

(1). $x \equiv E_{K_{s_{i_1}}}(m) \bmod n_{s_{i_1}}$.
(2). $x \equiv E_{K_{s_{i_2}}}(m) \bmod n_{s_{i_2}}$.
...
(k). $x \equiv E_{K_{s_{i_k}}}(m) \bmod n_{s_{i_k}}$.

A3. Third, s_i stores x in SDB.

For a subject $s_j \in \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$, to access m, s_j needs to compute $E_{K_{s_j}}(m) = x \bmod n_{s_j}$. Then, s_j uses the private key $K_{s_j}^{-1}$ to recover m, i.e. $m = D_{K_{s_j}^{-1}}(E_{K_{s_j}}(m))$.

The method described above can be easily configured as an equivalent to RRN system. For instance, choose RSA as our public key cryptosystem. At the system initialization stage, assign each subject $s_i \in S$ a pair of RSA keys: a public key $K_{s_i} = (e, n_{s_i})$ and a private key $K_{s_i}^{-1} = (d_{s_i}, n_{s_i})$ such that all RSA moduli $n_{s_1}, n_{s_2}, ..., n_{s_\ell}$ are pairwise relatively prime. Note, that all subjects share a public exponent e. There is no need to use the modulus generator MG here, because we use the RSA moduli as the CRT moduli. To share a message m with k subjects $s_{i_1}, s_{i_2}, ..., s_{i_k} \in S$, our system and RRN system generate two share ciphertexts x and x', respectively. To verify the equivalence of the above customized system and RRN system, we need to prove that the share ciphertexts generated by the two systems are equal, i.e. $x = x'$.

Theorem 3. *In the two systems above, the share ciphertexts* $x = x'$.

Proof.

To prove $x = x'$, we first demonstrate that x and x' are both the CRT solutions of the same set of simultaneous congruences.

$$
\begin{aligned}
x' \bmod n_{s_{i_1}} &= (m^e \bmod n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}) \bmod n_{s_{i_1}} \\
&= (m^e - q n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}) \bmod n_{s_{i_1}} \\
&= m^e \bmod n_{s_{i_1}} \\
&= E_{K_{s_{i_1}}}(m) \ .
\end{aligned}
$$

where $m^e = q n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}} + r$ for some integers q and r ($r < n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}$).

Hence $x' \equiv E_{K_{s_{i_1}}}(m) \bmod n_{s_{i_1}}$. Similarly, the other $k-1$ congruences $x' \equiv E_{K_{s_{i_2}}}(m) \bmod n_{s_{i_2}}, ..., x' \equiv E_{K_{s_{i_k}}}(m) \bmod n_{s_{i_k}}$ can be proven.

Thus, $x' < n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}$ is a solution to the above k simultaneous congruences. We know that x is also a solution to these k simultaneous congruences. From the Chinese Remainder Theorem, we know that the solution for the k simultaneous congruences is unique in the range $[0, n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}})$. Therefore $x = x'$ holds.

□

The theorem above indicates that *RRN* system is covered as a special case by our data based solution.

4.4 Authorization Alterations

Alteration of a data item's authorizations, e.g. a subject is granted/revoked access to a data item, is a frequent event in information systems. The way *RRN* system dealing with authorization alterations is very inefficient because each time an authorization changes the affected data item must be re-encrypted with a new key.

Our data based solution handles authorization alterations according to the status of the affected data item. If the data item is *dynamic* (i.e. the data item changes at the time of authorization alteration), all operations from `A1` to `A3` (see section 4.3) are re-performed based on the new group of authorized subjects. If the data item is *static* (i.e. the data item remains the same at the time of authorization alteration), an efficient method is used to process authorization alterations.

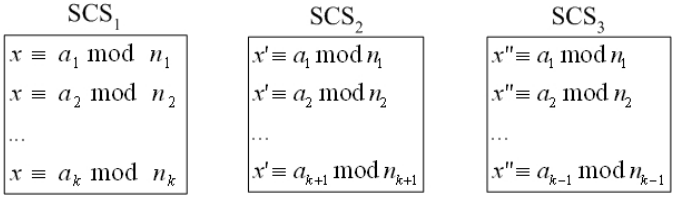

Fig. 1. Three Simultaneous Congruences Sets (SCSs)

The method is based on the following property of CRT. Consider the 3 Simultaneous Congruences Sets (SCSs) as shown in figure 1. SCS_1 contains k

simultaneous congruences, and its CRT solution is x; SCS_2 is created by adding one congruence to SCS_1, and its CRT solution is x'; SCS_3 is created by removing one congruence from SCS_1, and its CRT solution is x''. Assume, that the value of x has already been calculated. To get the value of x', we only need to find the CRT solution for the two congruences: $x' \equiv x \bmod n_1 n_2 ... n_k$ and $x' \equiv a_{k+1} \bmod n_{k+1}$; to get the value of x'', we only need one modular operation: $x'' = x \bmod n_1 n_2 ... n_{k-1}$. In a word, the values of x' and x'' can be easily derived from x.

In our data based solution, granting a subject access to a static data item is equivalent to the transformation from SCS_1 to SCS_2. The new share ciphertext x' can be derived from the old share ciphertext x efficiently. Revoking a subject from accessing a static data item is equivalent to the transformation from SCS_1 to SCS_3. The new share ciphertext x'' can be derived from the old share ciphertext x simply by a modular operation.

Let us analyze the security of the proposed method for static data item. First, let us consider a special situation: when $x < n_1 n_2 ... n_{k-1}$, $x'' = x \bmod n_1 n_2 ... n_{k-1} = x$. In this case, the above revocation method becomes useless because the revoked subject is still capable of decrypting x'' (which is equal to x). This problem is trivial because the probability of this situation is very low. In section 4.1, we have mentioned that CRT's mapping is a one-to-one correspondence between $\mathbb{Z}_n$ and the Cartesian product $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times ... \times \mathbb{Z}_{n_k}$ [15]. The data range $[0, n_1 n_2 ... n_{k-1})$ is only $\frac{1}{n_k}$ of $[0, n_1 n_2 ... n_k)$. If we choose 1024-bit numbers for CRT moduli, then the probability of $x < n_1 n_2 ... n_{k-1}$ is approximately 2^{-1024}. However, if $x < n_1 n_2 ... n_{k-1}$, we must re-perform all operations from `A1` to `A3` to revoke a subject. Finally, someone may argue that it is impossible to revoke a subject from accessing a static data item because the subject can simply store it before the revoking. Here, we assume some trusted workstations are used for subjects to access encrypted data items, on which saving a data item is disabled.

4.5 Performance and Security Analysis

This section compares the performance between *RRN* system and a *Data Based System* (*DBS*), which is configured as a *RRN* equivalent (see section 4.3). There are two algorithms used in these two systems: *fast modular exponentiation algorithm* and Garner's algorithm, whose complexity is detailed in [12, 13].

In both *RRN* and *DBS* systems, a message m is to be shared with k subjects, where m is of ℓ_m-bit in length, the RSA/CRT moduli of all subjects are of the same bit length: ℓ_n, the shared public exponent e is ℓ_e-bit and the private exponents are ℓ_d-bit (please note, ℓ_d is only an approximate value).

RRN system is purely based on RSA cryptosystem. Assume, that fast modular exponentiation algorithm is used. *RRN* encryption is calculated by $c = m^e \bmod n$, where n is the product of the k moduli and of $k\ell_n$-bit in length. Therefore the *RRN* encryption complexity is $O(\ell_e(k\ell_n)^2) = O(k^2 \ell_e \ell_n^2)$. The *RRN* decryption complexity is $O(\ell_d \ell_n^2)$. *DBS* system is based on RSA cryptosystem and CRT; fast modular exponentiation algorithm and Garner's algorithm are used. *DBS* encryption consists of k RSA encryptions and one CRT computation, its

Table 1. Performance Comparison between RRN and DBS

Systems	Encryption	Decryption	Granting access to a subject	Revoking access from a subject
RRN	$O(k^2\ell_e\ell_n^2)$	$O(\ell_d\ell_n^2)$	$O(k^2\ell_e\ell_n^2)$	$O(k^2\ell_e\ell_n^2)$
DBS	$O(k\ell_e\ell_n^2)$	$O(\ell_d\ell_n^2)$	$O(\ell_e\ell_n^2)$	$O(k\ell_n^2)$

complexity is $kO(\ell_e\ell_n^2)+O(k\ell_n^2) \approx O(k\ell_e\ell_n^2)$. The DBS decryption complexity is the same as that of RRN: $O(\ell_d\ell_n^2)$. We next analyze the complexity of authorization alterations. In RRN system, granting access to a subject (or revoking access from a subject) requires re-encrypting the affected data item. The complexity of this re-encryption is approximately $O(k^2\ell_e\ell_n^2)$. In our data based solution, granting a subject access to a static data item, we only need to generate a new individual ciphertext for the subject and then derive the new share ciphertext from the old one. The complexity of this process is: $O(\ell_e\ell_n^2)+2O(\ell_n^2) \approx O(\ell_e\ell_n^2)$. Revoking a subject from accessing a static data item only needs one modular operation. The complexity of this process is: $O(k\ell_n^2)$. Here we only illustrate authorization alterations for static data items; for dynamic data items, efficiency of authorization alterations is the same as that of encryption. The performance comparison between RRN and DBS is summarized in table 1, which shows that besides decryption, DBS system is more efficient than RRN system. Furthermore, our system has the flexibility of choosing an alternative public key cryptosystem which may results in more efficient system than DBS system.

As we know, RRN system requires the RSA public exponent e to be shared. This opens a potential security hole to attackers. The claim of [1] that "having multiple copies of the same data encrypted with different keys does not arise" is not true because with the knowledge of the RSA moduli and the sharers of a data item, an attacker can create those multiple copies by modular operations. In comparison with RRN system, if our data based solution uses the RSA cryptosystem, sharing the same RSA public exponent e is not required, i.e. different RSA public exponents can be used. Moreover, our data based solution has the flexibility of choosing an alternative public key cryptosystem which may results in more secure system.

5 A Key Based Solution

As discussed in section 1, our cryptographic solution of general access control has two categories: *data based solution* and *key based solution*. In data based solution, to share a message m with k sharers, the size of the share ciphertext is k times bigger than that of m. As a consequence, data based solution is not preferable if m or k is big. Moreover, data based solution is based on public key cryptosystem. This is because, to share a data item, its owner must know all sharers' encryption keys. In order to protect the confidentiality of decryption keys, we can only use a public key cryptosystem. Public key cryptosystems are

typically substantially slower than symmetric key cryptosystems [13]. Therefore, our data based solution is not so efficient, especially when m or k is big. In this section, we propose a key based solution, which solves the above problems. Our idea of key based solution is derived from our data based solution: instead of sharing a message, we share its encryption key. The technique used in our key based solution has been used by some secure broadcasting systems, e.g. [16, 17]. In contrast to those secure broadcasting systems, our key based solution applies to a different area: general access control.

In addition to the system elements listed for our data based solution (see section 4.2), key based solution requires a symmetric key cryptosystem. Here we denote its *encryption function* as SE and its *decryption function* as SD. This symmetric key cryptosystem is used to encrypt data items and the encryption keys are shared by a public key cryptosystem and CRT.

The key based solution is depicted by the following scenario. If a subject s_i wants to share a message m with k subjects $s_{i_1}, s_{i_2}, ..., s_{i_k} \in S$, s_i performs the following operations:

`B1`. randomly choose a symmetric key K_R;
`B2`. use K_R to encrypt m: $c = SE_{K_R}(m)$;
`B3`. $\forall s_j \in \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$, calculate $E_{K_{s_j}}(K_R)$;
`B4`. find the CRT solution x to the following k simultaneous congruences:
(1). $x \equiv E_{K_{s_{i_1}}}(K_R) \bmod n_{s_{i_1}}$.
(2). $x \equiv E_{K_{s_{i_2}}}(K_R) \bmod n_{s_{i_2}}$.
...
(k). $x \equiv E_{K_{s_{i_k}}}(K_R) \bmod n_{s_{i_k}}$.
`B5`. store $x||c$ in SDB, where the symbol $||$ means concatenation.

For a subject $s_j \in \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$, to access m, s_j needs to compute $E_{K_{s_j}}(K_R) = x \bmod n_{s_j}$; then uses private key $K_{s_j}^{-1}$ to retrieve the symmetric key K_R, i.e. $K_R = D_{K_{s_j}^{-1}}(E_{K_{s_j}}(K_R))$; finally uses K_R to recover m, i.e. $m = SD_{K_R}(c)$.

In our key based solution, authorization alterations are processed in the following way. For a dynamic data item, whenever its authorization changes, all operations from `B1` to `B5` are re-performed based on the new group of authorized subjects. For a static data item, if a subject is revoked from accessing the data item, to prevent the subject from using the old symmetric key to retrieve the data item, all operations from `B1` to `B5` are re-performed based on the new group of authorized subjects; if a subject is granted access to the data item, the re-encryption of data item is not needed because the old symmetric key can still be used. Thus the transformation from SCS_1 to SCS_2 (see section 4.4) can be used to generate a new share ciphertext for the old symmetric key such that the newly authorized subject can retrieve the old symmetric key to decrypt the data item.

In contrast to data based solution, multiple public key encryptions are performed on a symmetric key and one symmetric key encryption is performed on

a data item. Because the size of the symmetric key is usually much smaller than that of the data item, the public key encryptions are more efficient than those of the data based solution. Due to the same reason, the size of the share ciphertext is much smaller than that of the data based solution. In summary, key based solution is preferable when a data item or the number of sharers is big.

6 Experimental Results

As discussed in section 4.5 and 5, our solution is more efficient than *RRN* system. In this section, we list our experimental results as supporting evidence.

Table 2. Experimental Results

Systems	Encryption	Decryption	Granting access to a subject	Revoking access from a subject	Share ciphertext size
RRN	71,132 ms	11,707 ms	86,124 ms	57,683 ms	1,008,641 bytes
DBS	14,320 ms	11,476 ms	1,703 ms	401 ms	1,008,641 bytes
KBS	581 ms	491 ms	10 ms	561 ms	101,297 bytes

We have written Java programs to implement the following three systems: *RRN* system, *DBS* system and a *Key Based System* (*KBS*) using RSA and the *Advanced Encryption Standard* (AES). Our programs are running on *Java 2 Standard Edition* (J2SE) 1.4.2 and Windows XP; the test machine is a Pentium M 1.60GHz laptop with 512M memories. In our experiments, we share a 100,000-byte file with 10 sharers. The RSA public exponent is 16-bit; the RSA private exponents are approximately 1020-bit; the RSA/CRT moduli are 1024-bit and the AES keys are 128-bit. We have run four tests for each system: encryption, decryption, granting access to a subject and revoking access from a subject (Here we only measure authorization alterations for static data items; for dynamic data items, efficiency of authorization alterations is the same as that of encryption). The experimental results are shown in table 2, where times are measured in milliseconds (ms) and sizes are measured in bytes. The experimental results demonstrate the following facts, which conform to our earlier discussions.

- Our data based solution is more efficient than *RRN* system. The share ciphertext size grows proportionally as the number of sharers increases.
- Key based solution is more efficient than data based solution. And the share ciphertext size does not grow a lot when the number of sharers increases.
- Our authorization alteration mechanism is more efficient than that of *RRN* system.

7 Conclusion and Future Work

In this paper, we have proposed a cryptographic solution for general access control. Our solution is based on Chinese Remainder Theorem (CRT) and has two categories: data based solution and key based solution. *RRN* system is actually a special case of our data based solution. In contrast to *RRN*, our data/key based solution is more efficient and flexible. The technique used in our key based solution has been used by some secure broadcasting systems. However, our key based solution applies to a different area: general access control. We have proposed a mechanism for authorization alterations. This mechanism consists of very simple operations, which make it very efficient. Moreover, by using our solution, a system designer has the flexibility of choosing appropriate cryptosystems which may result in more efficient and secure system. Finally, we have utilized a set of experiments to verify our system; the experimental results provide evidence that supports our research.

In the future, our research will follow the following directions.

- Our solution can be applied to various systems where the need for access control arises. For example, multi-user file systems, database systems, message broadcasting systems and so on. In the future, we will develop one of such systems that is based on our solution.
- Explore methods other than Chinese Remainder Theorem that can be applied to cryptographic access control.

References

1. Ray, I., Ray, I., Narasimhamurthi, N.: A Cryptographic Solution to Implement Access Control in a Hierarchy and More. Proceedings of the Seventh ACM Symposium on Access Control Models and Technologies. ACM Press (2002) 65–73
2. Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R., Chandramouli, R.: Proposed NIST Standard for Role-based Access Control. ACM Transactions on Information and System Security, Vol. 4, No. 3. ACM Press (2001) 224–274
3. Jajodia, S., Samarati, P., Sapino, M. L., Subrahmanian, V. S.: Flexible Support for Multiple Access Control Policies. ACM Transactions on Database Systems, Vol. 26, No. 2. ACM Press (2001) 214–260
4. Akl, S. G., Taylor, P. D.: Cryptographic Solution to a Multilevel Security Problem. Advances in Cryptology: Proceedings of Crypto '82. Plenum Press (1982) 237–249
5. Akl, S. G., Taylor, P. D.: Cryptographic Solution to a Problem of Access Control in a Hierarchy. ACM Transactions on Computer Systems, Vol. 1, No. 3. ACM Press (1983) 239–248
6. MacKinnon, S. J., Taylor, P. D., Meijer, H., Akl, S. G.: An Optimal Algorithm for Assigning Cryptographic Keys to Access Control in a Hierarchy. IEEE Transactions on Computers, Vol. 34, No. 9 (1985) 797–802
7. Chick, G. C., Tavares, S. E.: Flexible Access Control with Master Keys. Advances in Cryptology: Proceedings of Crypto'89. Lecture Notes in Computer Science, Vol. 435. Springer-Verlag, Berlin Heidelberg New York (1990) 316–322
8. Harn, L., Lin, H. Y.: A Cryptographic Key Generation Scheme for Multi-level Data Security. Computer & Security, Vol. 9, No. 6 (1990) 539–546

9. Sandhu, R. S.: Cryptographic Implementation of a Tree Hierarchy for Access Control. Information Processing Letters, Vol. 27, No. 2 (1988) 95–98
10. Ohta, K., Okamoto, T., Koyama, K.: Membership Authentication for Hierarchical Multigroup using the Extended Fiat-Shamir Scheme. Advances in Cryptography: Proceedings of the EuroCrypt'90. Lecture Notes in Computer Science, Vol. 473. Springer-Verlag, Berlin Heidelberg New York (1991) 316–322
11. Zheng, Y., Hardjono, T., Seberry, J.: New Solutions to the Problem of Access Control in a Hierarchy. Technical Report Preprint 93-2, Department of Computer Science, University of Wollongong (1993)
12. Yan, S. Y.: Number Theory for Computing. Springer-Verlag, Berlin Heidelberg New York (2002)
13. Menezes, A. J., Oorschot, P. C. V., Vanstone, S. A.: Handbook of Applied Cryptography. CRC Press (1996)
14. Rivest, R. L., Shamir, A., Adleman, L.: A Method for Obtaining Digital Signatures and Public-key Cryptosystems. Communications of the ACM, Vol. 21, No. 2. ACM Press (1978) 120–126
15. Stallings, W.: Cryptography and Network Security: Principles and Practices. Prentice Hall (1999)
16. Chiou, G., Chen, W.: Secure Broadcasting Using the Secure Lock. IEEE Transactions on Software Engineering, Vol. 15, No. 8. (1989) 929–934
17. Zou, X., Ramamurthy, B., Magliveras, S.: Chinese Remainder Theorem Based Hierarchical Access Control for Secure Group Commmunication. Proceedings of the Third International Conference on Information and Communications Security. Lecture Notes in Computer Science, Vol. 2229. Springer-Verlag, Berlin Heidelberg New York (2001) 381–385