Asian Journal of Informaion Technology 5 (1) : 75-78, 2006
© Medwell Online, 2006

An Incremental Encryption Scheme Based on the Parallelizable and Integrity-assured Block Encryption Mode

Chuan-Chi Wang
Department of Computer Science and Information Engineering,
Ching-Yun University, Jung-Li, Taiwan ROC

Abstract: The aim of this study an incremental encryption scheme that can reduce the necessary recomputation of blocks insertion and deletion on an encrypted data. Both of privacy and integrity of the encrypted data are guaranteed. The scheme has very good property of incrementality in the case that the underlying block cipher is far complicated than the simple mathematic operations such as Addition/Subtraction. It bases on the block encryption mode that is parallelizable and integrity assured.

Key words: Incremental cryptography, block cipher, hash function, encryption mode

INTRODUCTION

The notion of incrementality of cryptographic functions was proposed by Bellare, Goldreich and Goldwasser[1]. The concept is that for a document M and the corresponding cryptographic value C, having once applied a transformation to M, the time to recompute C should be proportional to the amount of modification done to M. The modification includes the operations of data replacement, insertion and deletion. Thereby one obtains much faster cryptographic primitives for environments where closely related documents are undergoing the same cryptographic transformations. They also proposed an incremental collision-free hashing function based on discrete exponentiation in a group of prime order.

It is very expensive to use modular exponentiation operations for data hashing. Thus, Bellare and Micciancio then introduced a new paradigm for incremental data hashing by the concept of randomize-then-combine[2]. They suggested using standard hash functions to randomize each data blocks and then combining together with a group operation such as addition or multiplication.

Both of the schemes mentioned above support the properties of incrementality and integrity. However, in some case, the property of privacy is also concerned. In the case of concerning privacy and incrementality, the ECB mode encryption[3] obviously satisfy the requirement. Because ECB mode operates each blocks independently and parallelizable. However, in the case that all of the three properties (privacy, integrity and incrementality) are concerned, we need more complicated cryptographic schemes. In fact, the encryption mode that is parallelizable and integrity assured[4,5] can solve the problem partially. The methods can solve the problem of replacement, but not insertion and deletion. The reason is that they use a series of values, named as Whitening-Random-Sequence, which have fixed order to combine with the data blocks in the plaintext to protect data integrity avoiding the possible substitution attacks. However, for a document M and the corresponding cryptographic value C, the insertion and deletion operations will change the order of the data blocks. The blocks which order is changed have to be recomputed. Thus, not only the inserted block has to be encrypted, but also the blocks behind it have to be recomputed. The deletion operation has the same situation.

In this study , we proposed a method that can reduce the necessary recomputation of C when M is modified by block insertion or deletion. The idea is to update Whitening-Random-Sequence to fit the new sequence of M. We give a full detail of the idea by a model presented in section 2. Applying the proposed model, we design a cryptographic scheme and present it in section 3. We then discuss the cost for each operation in section 4, and the security properties in section 5. Finally, a brief conclusion is described in section 6.

THE ICSID MODEL

Encryption modes those are parallelizable and integrity assured[4,5] can be described generally as the following mathematic expressions. For a sequence of data blocks $M = <M_1, M_2, \ldots, M_n>$, a nonce r is generated and applied to derive a sequence of random numbers $R = <r_1, r_2, \ldots, r_p>$ by a method G() where p should not be less then n.

$$r \xrightarrow{G()} < r_1, r_2, \ldots, r_p >$$

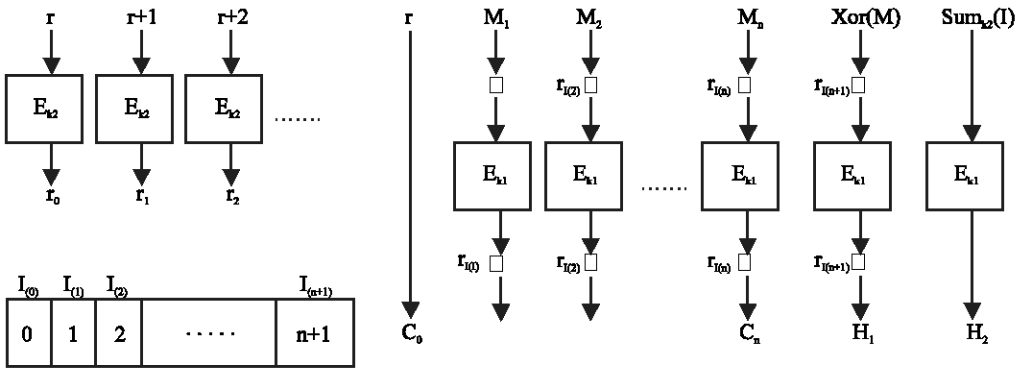

Fig. 1: An application of ICSID model

Each data block M_i goes to be combined with the corresponding random value r_i by some simple operations such as bit-wise exclusive-or, and then be encrypted to derive the corresponding ciphertext C_i by an encryption algorithm E().

$$\{r_i, M_i\} \xrightarrow{E()} C_i \ , \ C = \{C_1, C_2, ..., C_n\}$$

To guarantee integrity, parts of the value of $<r_1,r_2,...,r_p>$, $<M_1, M_2,..., M_n>$ and $<C_1,C_2,...,C_n>$are combined to be the integrity check value H by a combination method denoted as Com().

$$\{<r_1,r_2,...,r_p>,<M_1,M_2,...,M_n>,<C_1,C_2,...,C_n>\} \xrightarrow{Com()} H$$

The output set is {r, C, H}.

In above model, Mi has to be encrypted with ri. Thus, in the case that a data block is removed or a new data block is inserted, the sequence index of the blocks behind the modified one will be changed therefore having to be recomputed with the new corresponding random value.

For improving incrementality, we change this model by adding an additional component that is a set of index denoted as I. Specifically, the set I is an ordered set that is a permutation of {1,2,…,p} and I(i) is a function return the ith item in I. We then change the Step 2 of the above model to encrypted Mi with $r_{I(i)}$.

$$\{r_{I(i)}, M_i\} \xrightarrow{E()} C_i$$

This means that we can change the relation between M and R by updating the index set I. Therefore, in the case of block insertion and deletion, we are not necessary to recompute the blocks behind the modified one, but only need to change the index set appropriately. The detail operations are described in the next section illustrated by an application of the model.

In the new model, the index set I should be included in the combination of H for checking the possible illicit modification.

$$\{I,<r_1,r_2,...,r_p>,<M_1,M_2,...,M_n>,<C_1,C_2,...,C_n>\} \xrightarrow{Com()} H$$

The output set of the new model is {r,I,C,H}. We call the new model as ICSID standing for Incremental Cryptographic Scheme with efficient data Insertion and Deletion.

The incremental encryption scheme: Applying the ICSID model, we present an application that is based on the encryption mode scheme proposed in[4]. It is depicted in (Fig 1).

We use E_k to denote encryption function under key k and D_k for decryption. The data M which is to be encrypted is divided into blocks in length of the block size of the underlying block cipher. Let these blocks be $M_1,M_2, \ldots, M_n$ when M has n blocks. For simplicity, we assume that all operations, including replacement, insertion and deletion, work on individual data blocks. The problem of variable length data modification is not discussed here.

Initially, a random value r is chosen. It is used to generate random values $S = <S_0,S_1, S_2, \ldots>$, by applying the underlying block cipher and a key different with the one used for data block encryption. The function is as follows:

$$S_i = E_{k2}(i + r)$$

Obviously, this is a count mode encryption[6]. The output has good random property[4]. Furthermore, the random values do not need to be generated in sequence. We can directly get any single random value by giving the input. Therefore, in the application, the items in the index set are extended to be a series of unequal integer.

The ciphertext $C = <C_0, C_1, \ldots, C_n>$ and the integrity check value $H = <H_1, H_2>$ are computed as follows:

$C_0 = r$,

$C_i = E_{k1}(M_i \oplus r_{I(i)}) \oplus r_{I(i)}$ for i = 1 to n,

$H_1 = E_{k1}(Xor(M) \oplus r_{I(n+1)}) \oplus r_{I(0)}$,

$H_2 = E_{k1}(Sum_{k2}(I))$.

The notation □ means XOR. I is the index set which records the map between M and R, that is, if I(i) = j then Mi is encrypted with S_j. Xor(M) means to use the operation of XOR to compress the plaintext into a block. $Sum_{k2}(I)$ partitions the index set into a sequence of q-bits blocks, denoted as $v_0, v_1, \ldots, v_t$ and then sum them together. The expression is as follows:

$$Sum_{k2}(I) = \sum_{i=0}^{t} v_i \mod 2^b$$

where I is partitioned into $v_0, v_1, \ldots, v_t$, and b is the block length of the underlying cipher.

The length of v_i, =q, is derived from the key k2 by the following expression:

$$q = B + (k2 \bmod A),$$

where A=b/2 and B=b/2. Let the length (=q) be in the range from b/2 to b.

In this application, the process for data block modifications, including replacement, insertion and deletion, are described as follows:

The process of replacement: For the output set {r,I,C,H}, when a data block M_i is replaced by another block denoted as P, the output set should be recomputed as follows:

- The random number r is not changed.
- The index set I is not changed.
- The new ciphertext C' is generated from C by changing C_i to $E_{k1}(P \square r_{I(i)}) \square r_{I(i)}$.
- The new integrity check value H' = $<H_1', H_2'>$ is derived by

$H_1' = E_{k1}(N \oplus r_{I(n+1)}) \oplus r_{I(0)}$ where

$N = D_{k1}(H_1 \oplus r_{I(0)}) \oplus r_{I(n+1)} \oplus M_i \oplus P$

$H_2' = H_2$

The process of insertion: For the output set {r,I,C,H}, when a data block P is inserted into the sequence of data at the position behind block M_{i-1}, the output set should be recomputed as follows:

- The random number r is not changed.
- The index set I is changed to be I' as follows:
  I'(j) = I(j) for j = 0 to i-1.
  I'(j+1)=I(j) for j = i to n+1.
  I'(i) = z, where z is an arbitrary integer which is not in the original index set I.
- The new ciphertext C' is generated by expressions as follows:
  $C'_j = C_j$ for j = 0 to i-1.
  $C'_{j+1} = C_j$ for j = i to n.
  $C'_i = E_{k1}(P \square r_{I'(i)}) \square r_{I'(i)}$, where $r_{I'(i)} = E_{k2}(r + I'(i)) = E_{k2}(r+z)$.
- The new integrity check value H' = $<H_1', H_2'>$ is derived by

$H_1' = E_{k1}(N \oplus r_{I(n+2)}) \oplus r_{I(0)}$, where

$N = D_{k1}(H_1 \oplus r_{I(0)}) \oplus r_{I'(n+2)} \oplus M_i \oplus P$

$H_2' = E_{k1}(Sum_{k2}(I')) = E_{k1}(\sum_{i=0}^{t'} v'_t \bmod 2^b)$ where I' is partitioned into $v'_0, v'_1, \ldots, v'_{t'}$.

The process of deletion: For the output set {r,I,C,H}, when a data block M_i is removed from the sequence of data M, the output set should be recomputed as follows:

- The random number r is not changed.
- The index set I is changed to be I' as follows:
  I'(j) = I(j) for j = 0 to i-1.
  I'(j)=I(j+1) for j = i to n.
- The new ciphertext C' is generated by removing C_i from C.
- The new integrity check value H' = $<H_1', H_2'>$ is derived by

$H_1' = E_{k1}(N \oplus r_{I(n)}) \oplus r_{I(0)}$, where $N = D_{k1}(H_1 \oplus r_{I(0)}) \oplus r_{I(n)} \oplus M_i$.

$H_2' = E_{k1}(Sum_{k2}(I')) = E_{k1}(\sum_{i=0}^{t'} v'_t \bmod 2^b)$ where I' is partitioned into $v'_0, v'_1, \ldots, v'_{t'}$.

Cost for replacement, insertion and deletion: According to the expressions in the previous section, we can figure out the number of operations that is needed for a single data block update, including replacement, insertion and deletion, on data stream in length of n (blocks). The result is shown in Table 1. We can see that all the necessary operations are constant except the operations of Addition/Subtraction that is caused by updating the integrity check value of the new index set. In this part, the cost of the update is related to the length of the whole data. Therefore, the application in Section 3 is not a perfect incremental algorithm because that by a perfect incremental algorithm, the time to update the result upon modification of a sequence of data blocks M should be proportional" to the amount of modification" done to M.[1] However, the part of operations is very simple mathematic Addition/Subtraction. The cost is usually far less than block cipher Encryption/Decryption. Therefore,

Table 1: The number of operations needed for update of a data block

	Replacement	Insertion	Deletion
Encryption/Decryption	4	9	5
XOR	6	7	5
Addition/Subtraction	0	O(n)	O(n)
Multiplication/Modulo	0	1	1

the application still has good property of incrementality, especially in the case that the underlying cipher is far complicated than the simple mathematic Addition/Subtraction.

Security Properties: The security of the ICSID model obviously is based on the security of the underlying encryption scheme. Besides, integrity of the index set should be guaranteed. Otherwise, malicious people may make a forgery data by applying the substitution attack.

For the application we presented in section 3, the security of the underlying encryption scheme has been proved in[4]. Although we change the method of random numbers generation to be a count mode encryption, this will not violate the condition mentioned in[4]. The result is still a sequence of pair-wise differentially-uniform vectors.

For the integrity of the index set in the application, an attacker is difficult to modify the index set without being found because a new index value will fail the decryption if the corresponding ciphertexts do not be changed consistently. Only the one who knows the secret keys can do it. However, a case of substitution attack may destroy the security. That is, an attacker can rearrange the items of the index set and the corresponding ciphertexts. The integrity will be verified for the rearranged data if we do not add any other authentic mechanism into the system. Thus, we use an additional key-dependent integrity check value, H_2 in the application, to ensure that the order of items in the index set does not be changed surreptitiously. The method is as described in Section 3. The whole index set is partitioned into blocks in length that is dependent on the secret key and then is combined together with the mathematic sum. In the application, the possible length is from 64 to 127. Without known the secret keys, the attacker has to guess the length before modifying the index set. Moreover, the integrity check value is encrypted by another key value in the system, thus the attacker is difficult to verify that his guess is right or not from the encrypted integrity check value.

CONCLUSIONS

The ICSID model can be used with any block encryption scheme that is parallelizable and integrity assured[4,5]. The most important issue is to ensure the integrity of the applied index set efficiently. A loose condition is required: The order of the items in the index set can not be changed without being found. We can combine the index set into the integrity check code of the underlying encryption scheme to gain integrity. Otherwise, in the case that the underlying scheme is not applicable to protect the index set, such as the case proposed in Section 3, we can use other specific function to do it. For example, the method of key dependent partition-sum that we propose in Section 3.

The application in Section 3 has no perfect incrementality because the cost of the update is related to the length of the whole data. This is caused by the integrity check function of the index set. The situation may not happen if the underlying encryption scheme is changed, although we still not find one. However, the part of operations is very simple mathematic Addition/Subtraction. Therefore, the application in Section 3 still has good property of incrementality, especially, in the case that the underlying cipher is far complicated than the simple mathematic Addition/Subtraction.

REFERENCES

1. Bellare, M., O. Goldreich and S. Goldwasser, 1994. Incremental cryptography: The Case Of Hashing And Signing, Advances in Cryptology - Crypto 94 Proceedings, Lecture Notes in Computer Science 839, Y. Desmedt Ed., Springer-Verlag.
2. Bellare, M. and D. Micciancio, 1997. A New Paradigm for Collision-free Hashing: Incrementality at Reduced Cost. Advances in Cryptology - Eurocrypt 97 Proceedings, Lecture Notes in Computer Science 1233, W. Fumy (Ed.), Springer-Verlag.
3. DES Modes of Operation, 1980. Federal Information Processing Standard Publication.
4. Charanjit, S.J., 2001. Encryption Modes with Almost Free Message Integrity. Advances in Cryptology –EUROCRYPT Proceedings, Lecture Notes in Computer Science 2045, Y. B. P.tzmann, Ed., Springer-Verlag.
5. Black, J. and P. Rogaway, 2002. A Block-Cipher Mode of Operation for Parallelizable Message Authentication. Advances in Cryptology EUROCRYPT Proceedings, Lecture Notes in Computer Science 2332 , L. Knudsen, (Ed.), Springer-Verlag.
6. Diffie, W., and M. Hellman, 1979. Privacy and Authentication: An Introduction to Cryptography. Proceedings of the IEEE, 67: 397-427.