兩種簡單的門檻式解密法
Two simple threshold decryption schemes

洪國寶
Gwoboa Horng

國立中興大學 資訊科學研究所
Institute of Computer Science, National Chung Hsing University

摘要

門檻式解密法,可讓一加密訊息在僅當參與解密人數超過某一門檻時,方能解出原文.該法具有可提供更具彈性之機密共享等的應用.本文提出兩種植基於機密共享的門檻式解密法:一無須會議金匙;另一則可偵測欺騙者,而正確的解出原文.

關鍵詞: 門檻式解密法, 機密共享

Abstract

A threshold decryption scheme allows a ciphertext sent to a group of n participants to be decrypted only when the number of participants from the group is larger than or equal to a predetermined threshold value t. Potential applications of such a scheme include document distribution and information sharing among a group of users. In this paper, we propose two simple threshold decryption schemes based on existent cryptosystems. The first scheme requires no session key and is based on public key cryptosystems. The second scheme can detect cheaters and is based on symmetric cryptosystems.

Keywords: Threshold decryption schemes, Secret sharing.

1 Introduction

In insecure environments, cryptosystems are often used to achieve secrecy. In such systems, a user A can send a secret message M to another user B by encrypting M with the shared secret key between A and B (symmetric cryptosystems) or the public key of B (public key cryptosystems).

Sometimes, a user may wish to send a ciphertext to a group of n participants such that only when the number of participants from the group is larger than or equal to a predetermined threshold value t can the ciphertext be decrypted. This concept is called threshold decryption, a simplified concept of group-oriented cryptography proposed by Desmedt in 1987 [5]. Potential applications include document distribution and information sharing among a group of users.

Many such schemes have been proposed [5, 6, 9, 12]. Most of them require setting up phase. In this paper, we try to design schemes based on existent cryptosystems. The first scheme requires no session key and is based on public key cryptosystems. The second scheme can detect cheaters and is based on symmetric cryptosystems.

The rest of this paper is organized as follows: Section 2 contains some related preliminaries including the notion of one-way functions, Diffie-Hellman key distribution scheme, Shamir's threshold scheme, and Rabin's information dispersal scheme. Section 3 introduces a threshold decryption scheme without session keys. Section 4 proposes a threshold decryption scheme against cheaters. Section 5 contains the conclusions.

2 Preliminaries

2.1 One-way function and key distribution scheme

The notion of one-way functions is the core of most cryptographic applications. A function $f : X \rightarrow Y$ is one-way if it is easy to compute $f(x)$ for every $x \in X$, yet it is hard for most $y \in Y$ to figure out an $x \in X$ such that $f(x) = y$. No one-way function has been found yet [17]. However, it is widely believed that one-way functions exist [8]. A candidate one-way function is *modular exponentiation*: Let g and n be integers and $Z_n = \{0, 1, \ldots, n-1\}$. The modular exponentiation with respect to basis g and modulus n is the function $f_{g,n} : Z_n \rightarrow Z_n$ defined by $f_{g,n}(x) = g^x \bmod n$. By using the well-known square-and-multiply algorithm [17], it is easy to compute $f_{g,n}(a)$ for any $a \in Z_n$. However, no efficient algorithms are known to solve the corresponding reverse problem, the discrete logarithm problem [13]. Usually, n is chosen to be a large prime p such that $p-1$ has at least one large prime factor, and g is a primitive element mod$p$.

Based on the hardness of the discrete logarithm problems, Diffie and Hellman proposed a key distribution scheme [7]. Suppose that A and B want to share a secret K_{AB}, where A has a secret x_A and B has a secret x_B. Let p be a large prime and g be a primitive element in $GF(p)$, both known. A computes $y_A = g^{x_A} \bmod p$, and sends y_A to B. Similarly B computes $y_B = g^{x_B} \bmod p$, and sends it to A. Then the secret is computed as $K_{AB} = g^{x_A x_B} \bmod p$.

2.2 Secret sharing and information dispersal algorithms

The concept of secret sharing was proposed independently by Blakely [1] and Shamir [15]. A secret sharing scheme allows a secret, K, to be shared among a set of n participants, $P = \{P_1, P_2, \ldots P_n\}$, such that only authorized subsets of P can recover K, but any unauthorized subset has no knowledge for the secret at all. Such a scheme is useful for protecting an important secret data, such as a cryptographic key, from being lost or destroyed without accidental or malicious exposure. It can also find application in a variety of settings, for examples, distributed decision making, hierarchical access, and efficient dispersal of information.

The secret sharing scheme proposed by Shamir [15] is called a threshold scheme. More precisely, a t-out-of-n threshold scheme, where $t \leq n$, goes as follows: The administrator, a person who generates and distributes the shares, randomly chooses a polynomial, called the secret polynomial, $f(x)$ of degree $t-1$ such that $f(0) = K$. All arithmetic is done in $GF(p)$, where p is a large prime number. The value $f(i)$, for $i = 1$ to n, are the shares to be distributed to the participants P_i respectively.

Suppose the participants $P_{i_1}, P_{i_2}, \ldots, P_{i_t}$ want to determine the secret K. Then each P_{i_j} submits his share. They will have t shares $y_{i_j} = f(i_j)$, $1 \leq j \leq t$. The secret polynomial can be reconstructed by either solving a system of linear equations or applying Lagrange interpolation formula.

However, if someone submits a false share, that is, a point not passed by the polynomial $h(x)$, then they will recover a wrong value. The participants submitting false shares are called cheaters. There are many papers, for examples [2, 3, 4, 11, 16, 18, 19], address the issue of cheater detection and identification.

Secret sharing schemes have been extensively investigated since their invention in 1979. A detailed bibliography can be found in Stinson's homepage [10].

One important application of secret sharing is for information dispersal. An information dispersal algorithm [14] is a method that breaks a file F of length L into n pieces S_i, $1 \leq i \leq n$, each of length L/t, so that every t pieces suffice for reconstructing F. One implementation goes as follows. Let $F = F_1F_2 \cdots F_t$ such that $|F_i| = L/t$, $1 \leq i \leq t$. First, we construct polynomial $F(x) = F_1x^{t-1} + F_2x^{t-2} + \cdots + F_{t-1}x + F_t$. Then we evaluate $F(x)$ at $x = 1, 2, \ldots, n$. Finally, the n pieces S_i are $(i, F(i))$, $1 \leq i \leq n$.

It is easy to see that any t or more pieces can be used to reconstruct the polynomial $F(x)$ and hence the file. However, unlike Shamir's threshold scheme, in the above scheme any $t-1$ pieces $S_{i_1}, S_{i_2}, \ldots, S_{i_{t-1}}$ may provide some information about F.

3 A threshold decryption scheme without session keys

In this section, we propose a threshold decryption scheme which is based on threshold scheme. However, our threshold scheme is used to encrypt the message not the session key. Therefore, our construction is more close to the information dispersal scheme. In our scheme, we assume that Diffie-Hellman key distribution scheme is incorporated. That is, each potential participant with identity u_i selects a secret key $x_i \in Z_p$ and publishes the corresponding public key $y_i \equiv g^{x_i} \bmod p$ where p is a large prime such that $|p| = 513$ and g is a primitive element of $GF(p)$.

To send a secret message M to a group G of n users $u_1, u_2, \ldots, u_n$, such that the ciphertext can be decrypted only if t of more users from G get together, that is, the threshold value is t, the sender performs the following steps:

- Step 1. Divide the message M into blocks of 512-bit each. Let $M = M_lM_{l-1} \cdots M_2M_1$, where $|M_i| = 512$.

- Step 2. Generate a random integer r from $[1, p-1]$ and compute $z_i \equiv y_i^r \bmod p$ for $i = 1, 2, \ldots, n$.

- Step 3. Construct a $(l+n)$-out-of-$(l+2n-t)$ threshold scheme by computing a degree $l+n-1$ polynomial $f(x) = M_lx^{l+n-1} + M_{l-1}x^{l+n-2} + \cdots + M_1x^n + A_{n-1}x^{n-1} + A_{n-2}x^{n-2} + \cdots + A_2x^2 + A_1x + A_0 \bmod p$ such that $f(u_i) = z_i$ for $i = 1, 2, \ldots, n$. The coefficients $A_{n-1}, \ldots, A_0$ can be easily be found by solving n simultaneous linear equations $f(u_i) = z_i$ for $i = 1, 2, \ldots, n$. The pairs (u_i, z_i), $i = 1, 2, \ldots, n$, are called secret shares.

- Step 4. Compute $R \equiv g^r \bmod p$, and $l+n-t$ shares $f(i)$ for $i = 1, 2, \ldots, l+n-t$ where none of them is a secret share. That is, $u_i > l+n-t$ for $i = 1, 2, \ldots, n$.

- Step 5. Send $\{t, R, (f(1), f(2), \ldots, f(l+n-t))\}$ to the group.

When t participants, say $u_{i_1}, u_{i_2}, \ldots, u_{i_t}$, get together and decide to recover the message. Then they can recover the message M by performing the following:

- Step 1. Compute individual secret share. For $j = 1, \ldots, t$, u_{i_j} computes his secret share $(u_{i_j}, R^{x_{i_j}} \bmod p)$ from R and his secret key x_{i_j}.

- Step 2. Apply Lagrange interploation formula to find a polynomial $f(x)$ together such that it passes througt the following points: $(u_{i_j}, R^{x_{i_j}} \bmod p)$ where $j = 1, \ldots, t$ and, $(k, f(k))$ for $k = 1, 2, \ldots, l + n - t$.

- Step 3. Recover the message by concatenating the first $m - n$ coefficients of $f(x)$ where m is the degree of $f(x)$.

The security of our scheme is very high. Without knowing a secret share, an intruder can only guess a share and compute a polynomial by Lagrange interpolation formula.

The secret keys are also protected. If a user, with identity u_i, is a recipient, then his secret share, $(u_i, R^{x_i} \bmod p)$ will be known to all other recipients. However, the secret key x_i is protected since to uncover x_i from $R^{x_i} \bmod p$ and R is equivalent to break the Diffie-Hellman key distribution scheme [7] which is a very hard problem [13].

The random integer r should be different every time the sender sends a message. Therefore, the secret shares are different. This can prevent the message recovered by an illegal recipient even though he is a group member in previous message.

The most time-consuming step in our scheme is to compute the secret shares since the computation of $R^{k_i} \bmod p$ may take a while. Furthermore, finding the coefficients $A_{n-1}, \ldots, A_0$ takes more time than encrypting M with a session key. Therefore, this scheme is most suitable for sending short messages.

4 A threshold decryption scheme against cheaters

In this section, we propose a threshold encryption scheme based on symmetric cryptography. We assume that there is a trusted center (TC) such that it shares a different secret key with everyone on the network. Knowledge of the secret key is used as proof of identity. Let k_i be the secret key shared between user u_i and TC. Let h be a one-way function known to everyone. The notation $\{m\}k_i$ is used to denote that a message m is encrypted with secret key k_i.

Suppose a user, say u_0, wishes to encrypt message M to a group G of n users $u_1, u_2, \ldots, u_n$, such that the ciphertext can be decrypted only if t or more users from G get together. Then the following steps are performed:

- Step 1. User u_0 requests for threshold decryption service by sending $C_1 = \{G, t\}k_0$ to TC.

- Step 2. TC decrypts C_1 and selects a session key k and a random integer R.

  Then TC computes $c_i = \{R\}k_i$ and $d_i = h(c_i)$ for $i = i, \ldots, n$.

  Next, TC constructs polynomial $f(x)$ passing through the following $n + 1$ points: $(0, k)$ and (u_i, c_i) for $i = i, \ldots, n$.

  Finally, TC computes $e_i = f(i)$ for $i = i, \ldots, n - t + 1$ and sends $C_2 = \{k, R, d_1, d_2, \ldots, d_n, e_1, \ldots, e_{n-t+1}\}k_0$ to u_0.

- Step 3. User u_0 decrypts C_2 to recover $k, R, d_1, d_2, \ldots, d_n, e_1, \ldots, e_{n-t+1}$.

  Then u_0 computes $C = \{M\}k$ and sends $C_3 = \{C, R, d_1, d_2, \ldots, d_n, e_1, \ldots, e_{n-t+1}\}$ to G.

When $l \geq t$ participants from group G, say $u_{i_1}, u_{i_2}, \ldots, u_{i_l}$, get together and decide to recover the message. Then they can recover the message M by performing the following step:

- Step 1. Compute secret shares.
  For $j = 1, \ldots, l$, u_{i_j} computes $c'_{i_j} = \{R\}k_{i_j}$.

- Step 2. Identify cheaters.
  For $j = 1, \ldots, l$, if $h(c'_{i_j}) \neq d_{i_j}$ then u_{i_j} is a cheater.
  Let $H = \{u_{a_1}, \ldots, u_{a_m}\}$ be the set of honest users. If $m < t$ then they cannot recover the message. Otherwise, they perform the following steps.

- Step 3. Recover the session key.
  Using Lagrange interpolating formula, the honest users compute a polynomial $f(x)$ passing through the points (i, e_i) for $i = 1, \ldots, n - t + 1$ and (u_{a_i}, c_{a_i}) for $i = 1, \ldots, l$. The session key $k = f(0)$.

- Step 4. Decrypt the ciphertext.
  The plaintext message $M = \{C\}k$.

It is easy to see that only the users specified in G will be able to recover the session key k. Since $h(c_i) = d_i$, every member in G can verify the correctness of a submitted share. Therefore, the session key and hence the message can be correctly recovered as long as the number of honest participants is equal to or greater than t.

This scheme can be adopted to enhance a key authentication server, such as Kerberos, to support threshold decryption service.

5 Conclusions

In this paper, two simple threshold decryption schemes based on existent cryptosystem are proposed. The first scheme requires no session key and is based on public key cryptosystems. It is most suitable for sending short messages. The second scheme can detect cheaters and is based on symmetric cryptosystems. It can be adopted to enhance a key authentication server, such as Kerberos, to support threshold decryption service.

Acknowledgment. This work is supported by the National Science Council Taiwan, Republic of China, under contract NSC86-2213-E-005-005.

References

[1] G. Blakley, Safeguarding cryptographic keys, *Proc. AFIPS 1979 Natl. Conf., New york*, **48**, 313-317, 1979.

[2] E.F. Brickell and D.R. Stinson, The detection of cheaters in threshold schemes, *SIAM J. Disc. Math.*, 4, 502-510, 1991.

[3] M. Carpenteri, A perfect threshold secret sharing scheme to identify cheaters, *Designs, codes, and cryptography*, **5**, 183-187, 1995.

[4] C C. Chang and R.J. Hwang, Efficient cheater identification method for threshold schemes, *IEE Proc.-Comput. Digit. Tech*, **144**, 23-27, 1997.

[5] Y. Desmedt, Society and group-oriented cryptography: a new concept, *Proc. Crypto '87*, pp. 120-127, 1987.

[6] Y. Desmedt and Y. Frankel, Threshold cryptosystems, *Proc. Crypto '89*, pp. 307-315, 1989.

[7] W. Diffie and M. Hellman, New directions in cryptography, *IEEE Transactions on Information Theory*, **IT-22**, pp. 644-654, 1976.

[8] S. Goldwasser, The search for provably secure cryptosystems, *Proceedings of Symposia in Applied Mathematics*, **42**, pp. 89-113, 1990.

[9] L. Harn, H. Lin, and S. Yang, Threshold cryptosystem with multiple secret sharing policies, *IEE Proc.-Comput. Digit. Tech.*, **141**, pp. 142-144, 1994.

[10] http://bibd.unl.edu/ stinson.

[11] E.D. Karnin, J.W. Greene and M.E. Hellman, On secret sharing systems, *IEEE Trans. Inform. Theory*, **IT-29**, 35-41, 1983.

[12] C. Lin and C. Chang, Method for constructing a group-oriented cipher system, *J. Computer Communications* **17**, pp. 805-808, 1994.

[13] S. Pohlig and M. Hellman, An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance, *IEEE Trans. Inform. Theory*, **IT-24**, pp. 106-110, 1978.

[14] M. Rabin, Efficient dispersal of information for security, load balancing, and fault tolerance, *Journal of ACM*, **36**, pp. 335-348, 1989.

[15] A. Shamir, How to share a secret, *Communications of ACM*, **22**, pp. 612-613, 1979.

[16] G. Simmons, Robust shared secret schemes or "how to be sure you have the right answer even though you don't know the question," *Congr. Numer.*, **68**, pp. 215-248, 1989.

[17] D.R. Stinson, *Cryptography: Theory and Practice*, CRC Press, Inc. 1995.

[18] M. Tompa and H. Woll, How to share a secret with cheaters, *J. Cryptology*, **1**, 133-138, 1988.

[19] T.C. Wu and T.S. Wu, Cheating detection and cheater identification in secret sharing schemes, *IEE Proc.-Comput. Digit. Tech.*, **142**, 367-369, 1995.