Weaknesses in Some Threshold Cryptosystems

Susan K. Langford

Atalla Corporation, 2304 Zanker Road, San Jose, CA 95131
E-mail: langford_susan@tandem.com

Abstract. Threshold cryptosystems allow n members of a group to share a private key such that any k of them can use the key without revealing its value. These systems can be divided into two categories, systems which use a trusted center to generate the shares and systems which create the shares in a distributed manner. This paper describes a number of security weaknesses which arise in systems which do not use a trusted center. We show that the n-out-of-n threshold undeniable signature scheme [8] has an actual security of only 2-out-of-n. The discrete log based threshold signature schemes [7, 11, 12] have a weakness in the key generation protocol. Finally, the generalized threshold cryptosystem [9] is not secure for some access structures.
Keywords: Threshold Cryptosystems, threshold signatures, secret sharing, cryptanalysis, digital signatures, undeniable signatures.

1 Introduction

The concepts of group oriented cryptography and threshold cryptosystems were developed by Frankel and Desmedt [2, 4, 6]. In a threshold cryptosystem, the private key is not held by a individual. Instead, the key is shared among a group such that a certain minimum number of them can work together to use the key without compromising its value. Any subset of the group with fewer than the threshold number of members will have no information about the key. This distribution of the key provides protection against dishonest group members and accidental disclosure of the key.

Threshold cryptosystems can be divided into two categories – systems which use a trusted center to generate the shares and systems which create the shares in a distributed manner. Systems which create shares in a distributed manner avoid the single point of vulnerability at key generation. However, the distributed generation is more complicated and can introduce weaknesses which do not occur in simpler protocols that assume a trusted center.

This paper describes a number of weaknesses in proposed schemes which do not use trusted key generation. Section 2 describes the threshold generation of undeniable signatures [8] and shows how two signers can sign an arbitrary message. Section 3 discusses the share generation phase of threshold digital signatures, showing why a commitment phase is necessary for security. Section 4 examines the generalized threshold cryptosystem of [9]. This system is not secure for certain access structures.

All of the systems described in this paper are based on the discrete logarithm problem. Each system has a large prime modulus p and an element, α, which is primitive mod p. These two parameters are public.

2 Undeniable Signatures

The concept of undeniable signatures was first proposed by D. Chaum and H. van Antwerpen [1] in 1989. An undeniable signature, like a digital signature, is a number computed from a message and a secret key known only to the signer. Unlike a digital signature, an undeniable signature can only be verified with the cooperation of the signer.

Harn and Yang [8] showed how to generate undeniable signatures using a threshold of 1-out-of-n or n-out-of-n shareholders. This section shows how two signers can forge a signature in the n-out-of-n scheme.

2.1 Chaum's Undeniable Signature Scheme

Harn and Yang's scheme is based on Chaum and van Antwerpen's first undeniable signature scheme based on discrete logarithms. User A chooses a private key x_A such that the $\gcd(x_A, p-1) = 1$ and calculates a public key $y_A = \alpha^{x_A} \bmod p$. To sign a message m, A computes $z = m^{x_A} \bmod p$.

A verifier and the signer can use a challenge-response protocol to validate a signature. Chaum also provides a disavowal protocol that allows the signer to prove that a forged commitment is not valid. The details of these protocols are beyond the scope of this paper.

2.2 Group Public Key Generation Phase

The first phase of Harn and Yang's protocol is the generation of the group public key. Shareholder i generates a private key x_i such that $x_i^{-1} \bmod p-1$ exists. The group public key y is calculated by all members as

$$y = \alpha^{x_1 x_2 \cdots x_n} \bmod p.$$

In order to calculate the group public key, the members must be connected in a ring. Member i calculates α^{x_i} and sends the result to member $i+1$. At the same time, member i receives $\alpha^{x_{i-1}}$ from member $i-1$. Member i raises this value to the x_i, calculating $\alpha^{x_{i-1}x_i}$ and again sends the result to member $i+1$. The process continues, calculating $\alpha^{x_{i-2}x_{i-1}x_i}$ in the third step, and so on. After n steps, every member should have a copy of the public key. Each member should verify that his copy of the public key matches that released by all other members.

2.3 Signature Creation

Calculating a signature for message m requires each member of the group to sign the message sequentially. The first signer signs the message with his private key x_1, calculating

$$z_1 = m^{x_1} \bmod p.$$

The first signer then sends z_1 to his successor, who calculates

$$z_2 = z_1^{x_2} = m^{x_1 x_2} \bmod p.$$

The process continues until all members have signed. This creates the undeniable signature

$$z = z_n = m^{x_1 x_2 \ldots x_n} \bmod p.$$

The disadvantage of the above procedure is that only the first signer knows the message. The other members are creating blind signatures; they have no way of verifying what is being signed. As Harn and Yang observe, in a threshold scheme the assumption is that the members do not trust each other, limiting the possible applications of this method.

Harn and Yang therefore recommended that a signature be generated using the same method used to generate the public key. The members are connected in a ring and the message m is used instead of the public element α. The members are required to release the signatures simultaneously. The group can then verify that all of the signatures match before releasing the signature.

2.4 Forging Signatures

The first problem with Harn and Yang's protocol is that it does not prevent a group member from getting an unauthorized message signed. Suppose a signer starts the protocol with an alternate message μ. The signature of μ is created and released at the same time as the signature of the valid message. Althought the invalid signature isn't officially released by the group, the cheating signer could secretly copy and use it. The other shareholders will know that the signatures did not all match, but they will not know if this was the result of cheating or an error. Group members can check which signer originated each version of the message, but a cheating signer can randomize other intermediate results, creating many different outputs.

This attack can be avoided by having the shareholders release a one-way function of the signatures to compare, and only releasing the actual signature if all values are equal. However, this procedure only increases the overall security slightly, because the scheme is still vulnerable to colluding signers.

The major weakness of the threshold undeniable signature scheme is that any two adjacent signers can undetectably sign an arbitrary message whenever the signing protocol is executed. Suppose signers i and $i+1$ are colluding and all other signers are acting honestly. In the first step of the signing protocol, signer $i+1$ calculates $\mu^{x_{i+1}}$, where μ is the alternate message the two want signed. Signer i calculates m^{x_i} like any legitimate signer.

After n steps, signer i will receive the signature started by signer $i+1$. This result is a valid signature of the message μ. Signer $i+1$ receives the signature started by signer $i+2$. This value is the signature of m. Signer $i+1$ sends a copy of this signature secretly to signer i. The two cheaters can now each release valid signatures of m. The other signers have no way of knowing that another signature has been generated.

3 Threshold Signatures

A threshold signature scheme allows k-out-of-n members of a group to create a valid signature. A number of these schemes have been proposed based on discrete log (modified ElGamal) signatures [7, 11, 12]. One of the benefits of these schemes is that they can be implemented without a single trusted center to generate the shares. These three papers all use similar protocols to accomplish this share generation.

This section explains the share generation protocol and describes how a shareholder can manipulate this protocol to control the group's public key. The basic attack is illustrated on an n-out-of-n scheme in section 3.1. Section 3.2 generalizes the attack to k-out-of-n schemes, and section 3.3 extends the attack to the scheme of Li, Hwang, and Lee [11]. Section 3.4 then explains how these attacks can be prevented. Since the attacks are only concerned with the key generation phase, the details of the signature generation for each of these schemes will not be described.

3.1 n-out-of-n Share Generation

To generate shares of the private key, each member of the group generates a value x_i and computes a corresponding public key,

$$y_i = \alpha^{x_i} \bmod p.$$

The group public key is then calculated by all members

$$y = \prod_{i=1}^{n} y_i \bmod p.$$

In this protocol, a weakness arises if the public keys are not all revealed simultaneously. Suppose that member n is the last member of the group to reveal her public key. Instead of broadcasting

$$y_n = \alpha^{x_n} \bmod p,$$

she reveals the quantity

$$\hat{y}_n = y_n \prod_{i=1}^{n-1} y_i^{-1}.$$

The computed value of the public key will then be

$$y = y_n.$$

Member n now knows the group public key, but does not know the value of her own share. Whenever the group signs a message, she must calculate her partial signature by calculating the actual signature for the group and then subtracting all of the other partial signatures.

3.2 k-out-of-n Share Generation

To generate shares for a k-out-of-n scheme, each member generates x_i and calculates $y_i = \beta^{x_i} \bmod p$, where q is a prime dividing p and β is an element of order q. The group public key is determined by the product of the y_i as before.

Each member i will then create a k-out-of-n secret sharing scheme for the value x_i by generating a kth order polynomial $f_i(z)$ with $f_i(0) = x_i \bmod q$. The share for member j is $f_i(z_j)$. Member i distributes the shares and makes public the value $y_{i,j} = \beta^{f_i(z_j)}$ for each of the members. Each member can check that the shares received from other members are valid by checking that $y_{i,j}$ does correspond to the share $f_i(z_j)$ and that for any group of k values for j, the public values $y_{i,j}$ can be combined to calculate y_i.

This system can be attacked if $n - k + 1$ of the members of the group collude. Note that this attack is interesting only if $n - k + 1 < k$; otherwise, the colluding members could recover the private key directly. [1] In the first stage of the attack, member n manipulates her public key as described in the previous section. She now knows the group's private key.

In order to prevent the honest group members from realizing that cheating is occurring, member n must distribute shares of the private key corresponding to $\hat{y}_n$. She generates $k - 1$ random quantities from $GF(q)$. These are the shares for the honest members. She distributes them and computes the corresponding $y_{i,j}$. She now needs to compute the remaining $y_{i,j}$. She can calculate these from her public key. (She cannot know the value of the remaining $f_n(j)$ without solving the discrete logarithm problem. However, she does not need to know these values because they are the shares for her conspirators.)

Assume that members one to $k - 1$ were honest and were given random $f_n(j)$. Member n knows that

$$\hat{y}_n = \prod_{j=1}^{k} y_{n,j}^{c_j} \bmod p,$$

where the c_j are interpolation constants. She can solve this equation for $y_{n,k}$. She then uses the same technique to find the remaining public keys.

[1] The scheme in [10] is technically not vulnerable to this attack because k must be larger that $n - k + 1$. However, since security through inefficiency is not a good practice, the commitment phase of section 3.4 should be used with the scheme in [10].

3.3 Non-anonymous Members

Li, Hwang, and Lee [11] proposed a scheme in which they claim that even if more than k members collude, they cannot recover the private key. Their scheme ties the signature to the identity of the signers. If k member collude, they can sign documents, but they cannot impersonate another set of members. Li, Hwang, and Lee propose two methods, one using a trusted center and one not requiring a trusted center. The scheme without a trusted center uses a key generation method similar to that described above, but slightly more complex. It is still vulnerable to the same basic attack.

In this scheme, the group generates a public key y as described in the previous sections. Each member of the group then generates shares for the other members of the group. Member i generates a value u_{ij} for member j such that u_{ij} is a share of x_i plus a random integer g_{ij}. Member i also calculates the public values $y_{ij} = \alpha^{u_{ij}} \bmod p$ and $z_{ij} = \alpha^{g_{ij}} \bmod p$.

As in the previous section, a cheating group member can manipulate y so that she knows the group private key. She can then generate $k-1$ shares for honest members by picking the u_{ij} and g_{ij} randomly. However, unlike the previous section, the cheating member can create more than $k-1$ honest shares. She simply generates a random u_{ij} and solves for the value of z_{ij}. Note that she does not know the value of g_{ij} for these additional shares. For dishonest members, she can pick a random g_{ij} and solve for the value of y_{ij}.

Once the shares are generated, any group of k honest members can create a signature. The cheating members can create a signature which looks as if it was signed by the $k-1$ honest members which received the first set of random shares. Since a signature must have k signers to appear valid, the dishonest group must include one of its members in the signature as well. Note that if more that $k-1$ honest shares are created, the dishonest signers cannot sign a message that appears to come from any of these extra shares.

3.4 Preventing The Attack

The attacks in this section are all easily avoided by having the group members generate a commitment to their public key before any of the values are revealed. This approach is the same as that used by Pedersen [13] to create a threshold cryptosystem without a trusted party. An alternate method of preventing the attack would require every shareholder to sign their own name using their private key and make the signature public along with the public key value.

4 Generalized Threshold Cryptosystem

Laih and Harn [9] proposed methods of creating a generalized threshold cryptosystem which could implement any secret sharing policy. This section describes their second scheme, which is based on the ElGamal encryption scheme. This method does not require the assistance of a trusted party to create the shares. This section shows that the scheme is not secure for some access structures.

4.1 Generalized Threshold Scheme

This scheme assumes that the access structure can be represented as

$$F = f_1 + f_2 + \cdots + f_t,$$

where each f_i is an access instance. The group public key is generated in the same manner as the previous section. Each member generates an x_i and broadcasts $y_i = \alpha^{x_i} \bmod p$. The group's key is the product of the public keys. Note that this may be vulnerable to the attack of the previous section.

Any user i who does not belong to the access instances f_j needs to publish a public key $T_{i,j}$ that satisfies

$$x_i = \sum_{h \in f_j} K_{i,h} + T_{i,j} \bmod p - 1, \tag{1}$$

where $K_{i,h} = y_h^{x_i} = \alpha^{x_i x_h} \bmod p$.

Encryption: Any outside sender uses ElGamal encryption with the group public key y to encrypt message m. The ciphertext is computed as

$$C_1 = \alpha^r \bmod p,$$

$$C_2 = my^r \bmod p,$$

where r is a random number between 1 and $p-1$. The value (C_1, C_2) is sent to all members of the group.

Decryption: Assume that the members of the access instance f_j will be decrypting (C_1, C_2). Each member i in f_j needs to calculate the partial result

$$C_{1,i} = C_1^{x_i + \sum_{h \notin f_j} K_{i,h}} \bmod p, \tag{2}$$

where $K_{i,h}$ is defined as above.

The partial results are then sent to one designated individual. The ciphertext is then decrypted by calculating

$$m = C_2((\prod_{i \in f_j} C_{1,i})(\prod_{h \notin f_j} C_1^{T_{h,j}}))^{-1} \bmod p \tag{3}$$

$$= C_2((\prod_{i \in f_j} C_1^{x_i + \sum_{h \notin f_j} K_{i,h}})(\prod_{h \notin f_j} C_1^{T_{h,j}}))^{-1} \bmod p \tag{4}$$

$$= C_2((\prod_{i \in f_j} C_1^{x_i})(\prod_{i \in f_j} C_1^{\sum_{h \notin f_j} K_{i,h}})(\prod_{h \notin f_j} C_1^{T_{h,j}}))^{-1} \bmod p \tag{5}$$

$$= C_2((\prod_{i \in f_j} C_1^{x_i})(\prod_{i \in f_j} \prod_{h \notin f_j} C_1^{K_{i,h}})(\prod_{h \notin f_j} C_1^{T_{h,j}}))^{-1} \bmod p \tag{6}$$

$$= C_2((\prod_{i \in f_j} C_1^{x_i})(\prod_{h \notin f_j} (\prod_{i \in f_j} C_1^{K_{i,h}})C_1^{T_{h,j}}))^{-1} \bmod p \tag{7}$$

$$= C_2((\prod_{i \in f_j} C_1^{x_i})(\prod_{h \notin f_j} C_1^{\sum_{i \in f_j} K_{i,h}+T_{h,j}}))^{-1} \bmod p \tag{8}$$
$$= C_2((\prod_{i \in f_j} C_1^{x_i})(\prod_{h \notin f_j} C_1^{x_h}))^{-1} \bmod p \tag{9}$$
$$= C_2(\prod_{i=1}^{n} C_1^{x_i})^{-1} \bmod p \tag{10}$$
$$= C_2(C_1^{x})^{-1} \bmod p \tag{11}$$
$$= my^r(\alpha^{xr})^{-1} \bmod p. \tag{12}$$

4.2 Access Structures

The weakness of this scheme arises from the linear equations (1). The total number of these equations will be determined by the access structure. However, the total number of variables, x_i and $K_{i,h}$, is determined by the total number of people in the group, independent of the access structure. Therefore, some access structures will allow unauthorized subsets of members to recover the private key.

Example: Suppose a group consisting of four members A, B, C, D has the access structure $F = AB + AC + AD + BC$. Member A knows the value of x_A, K_{A_B}, K_{A_C}, and K_{A_D}. The seven public $T_{i,j}$ values which were generated by group members B, C, and D give A enough information to find the private key.

5 Conclusion

This paper describes weaknesses in a number of threshold cryptosystems. These weaknesses are the result of trying to construct these systems without a trusted key generation center. This result does not imply that it is impossible to create secure schemes without a trusted party generating the shares. However, such schemes are more complicated than those that do allow a single trusted center and are therefore more vulnerable to manipulation.

The seriousness of these weaknesses vary. The signature schemes [7, 11, 12] can quite easily be made resistant to the attack described here by adding a commitment phase to the share generation protocol. The undeniable signature scheme [8] only has a security level of 2-out-of-n and should be used accordingly. The generalized threshold cryptosystem [9] has potential weaknesses both in its share generation and in the possibility that the public key may allow the calculation of the private key for certain access structures.

References

1. D. Chaum and H. van Antwerpen, "Undeniable Signatures," *Advances in Cryptology – Eurocrypt '89 proceedings*, Springer-Verlag, 1989, pp. 212-216.
2. Y. Desmedt, "Society and group oriented cryptography," *Advances in Cryptology–Crypto '87 proceedings*, Springer-Verlag, 1988, pp. 120-127.

3. Y. Desmedt, "Threshold Cryptography," *European Transactions on Telecommunications and Related Technologies*, Vol. 5, No. 4, July-August 1994, pp. 35-43.
4. Y. Desmedt and Y. Frankel, "Shared generation of authenticators and signatures," *Advances in Cryptology–Crypto '91 proceedings*, Springer-Verlag, 1992, pp. 457-269.
5. T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inform. Theory*, Vol. 31, 1985, pp. 521-539.
6. Y. Frankel and Y. Desmedt, "Parallel reliable threshold multisignature," *Tech. Report TR-92-04-02*, Dep. of EE & CS, Univ. of Wisconsin-Milwaukee, April 1992.
7. L. Harn, "Group-oriented (t, n) threshold digital signature scheme and digital multisignature," *IEE Proc.-Comput. Digit. Tech.*, Vol. 141, No. 5, September 1994, pp. 307-313.
8. L. Harn and S. Yang, "Group-Oriented Undeniable Signature Schemes without the Assistance of a Mutually Trusted Party," *Advances in Cryptology – Auscrypt '92 proceedings*, Springer-Verlag, 1992, pp. 133-142.
9. C. Laih and L. Harn, "Generalized threshold cryptosystems," *Advances in Cryptology – Asiacrypt '91 proceedings*, Springer-Verlag, 1993, pp. 159-166.
10. S. Langford, "Threshold DSS Signatures without a Trusted Party," *Advances in Cryptology – Crypto '95 proceedings*, Springer-Verlag, 1995, pp. 397-409.
11. C. Li, T. Hwang, N. Lee, "(t, n)-threshold signature scheme based on discrete logarithm," *Advances in Cryptology – Eurocrypt '94 proceedings*, Springer-Verlag, 1995, pp. 191-200.
12. C. Park and K. Kurosawa, "New ElGamal Type Threshold Digital Signature Scheme," pre-print.
13. T. Pedersen, "A Threshold Cryptosystem without a Trusted Party," *Advances in Cryptology – Eurocrypt '91 proceedings*, Springer-Verlag, 1992, pp. 522-526.
14. R. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Communications of the ACM*, Vol. 21, April 1978, pp. 294-299.
15. A. Shamir, "How to share a secret," *Commun. ACM*, 22:612-613, November 1979.