Multisignature Scheme Based on Discrete Logarithms in the Plain Public Key Model

Zuhua Shao
Zhejiang University of Science and Technology, P. R. of China
E-mail: zhshao_98@yahoo.com

Keywords: discrete logarithm, random oracle model, group oriented cryptography, multisignature.

Received: November 18, 2008

In this paper, we propose a new multisignature scheme based on discrete logarithms. We show that this new scheme can resist existential forgeries against adaptive chosen-message attacks in the random oracle model. The main contribution is that our security model gets rid of the special security requirement on the generation of the signers' public keys. Adversaries are not required to reveal private keys corresponding to the public keys of its choice to the challenger in attack games. Thus the new multisignature scheme does not suffer from the problem identified by Micali et al., which is shared by many current multisignature schemes. Moreover, if the joint public key of a group of signers in this multisignature scheme is precomputed, the proposed multisignature scheme is optimal.

Povzetek: Opisana je shema podpisov za zaščito javnih ključev.

1 Introduction

Society oriented cryptography is a notion introduced by Desmedt [1]. A society oriented signature is essentially like a single signature except that is generated by plural individuals simultaneously.

A multisignature scheme is one kind of society oriented signature scheme, which allows multiple signers to sign the same message in a collaborative and simultaneous manner. A trivial solution is that every signer signs the message using a normal signature scheme respectively. Obviously, this simple solution will meet the security requirements for the multisignature scheme if the underlying signature scheme is secure. Its main drawback, however, is that both the data expansion and the computation costs for verification increase linearly with the number of signers in the group. Harn [2] submitted two additional properties that need to be achieved in the design of an optimal multisignature scheme:

1. The size of a multisignature should be identical to that of an individual signature.
2. The verification process of a multisignature should be almost identical to that of an individual signature.

Hence, in an optimal multisignature scheme, not only the size of signatures is independent of the number of signers participating in signing, but also the computation costs for verification.

Since the notion of multisignatures was introduced by Itakura and Nakamura [3], there have been many multisignature schemes proposed in literatures. However, most importantly, until the works of Ohta and Okamoto [4] and of Micali et al. [5], there were no formal security models for multisignatures. This lack of formalism has led not only to some confusion as to the precise security requirements for multisignatures, but also to some multisignature schemes having been subsequently broken [6, 7].

In group oriented cryptosystems, we must consider the possibility that an adversary could corrupt some fraction of participants, and thereby comes into possession of their private keys. We even allow the adversary to specify the public keys of the corrupted participants. In so-called rogue-key attacks, the adversary would register public keys created as a function of public keys of other honest participants. This kind of attacks could be extremely danger to break some multisignature schemes. The security notion of Ohta and Okamoto [4] is not strong enough to withstand such rogue-key attacks in the key generation.

Micali et al. [5] gave the first strong security notion for multisignature schemes in the plain public key model. They discussed a series of more sophisticated approaches based on zero-knowledge proofs, by which the private keys corresponding to the public keys can be extracted. Their scheme requires, as a pre-processing step, that the set of potential signers engage in an interactive key generation protocol to generate their key pairs. Besides expensive and resulting in complex public keys, this dedicated key generation enforces the set of potential signers to be static.

Boldyrva [8] proposed an efficient multisignature scheme based on the Gap-Diffie-Hellman group. Lu et al. [9] proposed the first multisignature scheme from pairings, provably secure without random oracles. Their security models allow an adversary to create arbitrary public keys for the corrupted signers possibly dependent on the public keys of the honest signers. But they require the adversary to prove the knowledge of the

corresponding secret keys (*KOSK*) during the public key registration. For simplicity, it has the adversary to hand over the secret keys of the corrupted signers in key generation algorithm. However, this *KOSK* assumption is not realized by existing public key infrastructure (PKI). Key registration protocols specified by the most widely used standards, PKCS#10 [10] - used by VeriSign and RFC 4210 [11, 12] do not specify that the Certification Authority (*CA*) should require proofs of knowledge, instead, specify that the *CA* should require proofs of possession (*POP*). That is, applicant is required to hand over the *CA* a signature, under the public key it is attempted to get certified, of some message that includes the public key and the identity of the applicant.

While such requirement might intuitively appear to stop adversaries from picking rogue keys, it does not suffice to realize the security models of [8, 9]. Ristenpart and Yilek [13] analyzed these schemes when key registration requires *POP*s. They showed that the standardized *POP* mechanism does not lead to secure multisignatures. Both schemes fall to rogue-key attacks despite the use of standardized *POP*. They presented a straightforward and natural fix for this problem: simple use separate hash function for *POP*s and multisignatures at the cost of upgrading existing PKI.

In 2006, Bellare and Neven [14] proposed a new multisignature scheme in the plain public key model, requiring nothing more than each signer has a (certified) public key in GF(p), which means neither *KOSK* or *POP* is required in key registration protocols. They provided a security proof in the random oracle model. However, their scheme is less efficient than the original Schnorr signature since the computation of verification increases linearly with the number of signers in the group.

In this paper, we also propose a new multisignature scheme based on Discrete Logarithms (DL) in the plain public key model. We show that this new scheme can resist existential forgeries against adaptive chosen-message attacks in the random oracle model. The main contribution is that our security model gets rid of the special security requirement on the generation of participants' public keys. Namely, like [14], our security model not only allows so-called rogue-key attacks in the key generation, but also gives the adversary complete freedom in specifying the public keys of the corrupted signers. The adversary is no longer enforced to prove either knowledge or possession of the private keys corresponding to the public keys of its choice. The second contribution is that our multisignature scheme can provide sequentially accountability, which means that not only individual signers can be identified from the multisignatures, but also the order of accountability. The main technique of this paper is the joint public key composed of the public keys of a group of users, which has been used in self-certified signatures [15, 16] and joint encryption scheme [17] to achieve provably secure.

Moreover, if the joint public key of a group of signers is precomputed, the proposed multisignature scheme is optimal, since the size of multisignatures and the verification costs are the same as those for the single-signer Schnorr signature scheme, regardless of the number of signers.

2 The new multisignature scheme

In this section, we first present a formal definition for the multisignature scheme, and then provide an implementation of Multisignature Scheme based on Discrete Logarithms (MSDL). Let $U = \{U_1, U_2, \ldots, U_n\}$ be a group of n signers.

2.1 Definition for multisignature scheme

Definition 1. A multisignature scheme is specified as four randomized algorithms: ParaGen, KeyGen, Sign and Verify:

ParaGen: takes a security parameter 1^k as input and returns a system parameter P, including some cryptographic hash functions.

KeyGen: takes the system parameter P as input, each signer U_i of the group U chooses its keypair (x_i, y_i) respectively.

Sign: takes as input P, the signers of any subset S of the group U (without loss of generality $S = \{U_1, U_2, \ldots, U_t\}$) cooperatively generate a multisignature σ for a message M by using their keypairs (x_i, y_i). The joint public key Y_S of the subset S is composed of the individual public keys $\{y_1, \ldots, y_t\}$.

Verify: takes as input P, M, the joint public key Y_S and a multisignature σ, it returns *invalid* or *valid*, with the property that if $(P) \leftarrow$ **ParaGen**(1^k), $(\{x_1, \ldots, x_t\}, Y_S) \leftarrow$ **KeyGen**(P) and $\sigma \leftarrow$ **Sign**$(P, M, \{x_1, \ldots, x_t\}, Y_S)$, then **Verify**$(P, M, \sigma, Y_S) = valid$.

2.2 An Implementation of Multisignature Scheme based on Discrete Logarithms (MSDL)

We use the Schnorr signature [18] as the underlying signature, which has been proved to be secure in the random oracle model [19].

ParaGen: A trusted party takes a security parameter 1^k as input and returns the system parameter P, which includes a subgroup $G_{g,p} = \{g^0, g^1, \ldots, g^{q-1}\}$ of a prime order q in the multiplicative group Z_p^*, where g is a generator with the prime order q, and two (ideal) hash functions H and F, where

$$H: G_{g,p} \times \ldots \times G_{g,p} \rightarrow Z_q^* \text{ and } F: \{0,1\}^* \times Z_p^* \times Z_q^* \rightarrow Z_q^*$$

KeyGen: takes the system parameter P as input, each signer U_i of a group U chooses its private key $x_i \in Z_q^*$ and computes its public key $y_i = g^{x_i}$ respectively.

Sign: takes as input P, the signers of a subset S of the group U (without loss of generality $S = \{U_1, U_2, \ldots, U_t\}$) cooperatively generate a multisignature σ for a message M by using their keypairs (x_i, y_i) as follows:

1. Each signer U_i of the subset S chooses a random number $k_i \in Z_q^*$, computes $r_i = g^{k_i}$ and broadcasts (y_i, r_i).
2. After receiving (y_j, r_j), $(j = 1, 2, \ldots, t)$, each signer

computes $R = r_1r_2\ldots r_t$, $h = H(y_1, y_2,\ldots, y_t)$ and $f = F(M, R, h)$.

3. The first signer U_1 computes $s_1 = k_1 - fx_1 \pmod q$ and sends it to the second signer U_2.
4. After receiving s_1 from the first signer U_1, the second signer U_2 first verifies $g^{s_1} y_1^{f} = r_1$ and then computes $s_2 = s_1 + k_2 - hfx_2 \pmod q$, and sends it to the third signer U_3.
5. After receiving s_{t-1} from the $(t-1)$th signer U_{t-1}, the last signer U_t first verifies $g^{s_{t-1}}(y_1 y_2^{h}\ldots y_{t-1}^{h^{t-2}})^{f} = r_1r_2\ldots r_{t-1}$, and then computes $s = s_{t-1} + k_t - h^{t-1}fx_t \pmod q$. The multisignature for the message M is $\sigma = (f, s)$.

Verify: The verifier first computes the joint public key of the subset $Y_S = y_1 y_2^{h}\ldots y_t^{h^{t-1}}$, where $h = H(y_1, y_2,\ldots, y_t)$, and then checks the verification equation of the multisignature $f = F(M, g^{s} Y_S^{f}, H(y_1, y_2,\ldots, y_t))$.

Completeness: Because $s_1 = k_1 - fx_1 \pmod q$ implies $g^{s_1} y_1^{f} = r_1$, $s_2 = s_1 + k_2 - hfx_2 \pmod q$ implies $g^{s_1}(y_1 y_2^{h})^{f} = r_1r_2$. By the same reason, $g^{s_{t-1}}(y_1 y_2^{h}\ldots y_{t-1}^{h^{t-2}})^{f} = r_1r_2\ldots r_{t-1}$ and $s = s_{t-1} + k_t - h^{t-1}fx_t \pmod q$ imply the verification equation. Hence, the signature $\sigma = (s, f)$ produced by the algorithm **Sign** is always *valid*.

Notice that our results can also be carried over to other groups, such as those built on elliptic curves.

Notice that the algorithm **Verify** requires that a verifier computes the joint public key $Y_S = y_1 y_2^{h}\ldots y_t^{h^{t-1}}$ from the individual public keys $\langle y_1, y_2,\ldots, y_t\rangle$ of a subset of signers. However, this time-consuming computation is independent of messages to be signed, and hence can be done once for all. Once the joint public key Y_S of a subset of signers is precomputed, the performance of the multisignature scheme is optimal.

3 Security model and security proof

In this section, we first define a new security model for multisignature schemes, which gives the adversary complete freedom in specifying the public keys of the corrupted signers without handing over the corresponding private keys. Then we provide the security proof in this strong security model.

3.1 Security model for multisignature schemes

Security model

Existential unforgeability against adaptive chosen message attacks (EUF-CMA) [20] is the well-accepted security model for signature schemes, where the adversary is allowed to ask the challenger to sign any message of its choice adaptively, i.e. he can adapt its queries according to previous answers. Finally, the adversary could not provide a new message-signature pair with a non-negligible advantage. Hence, it is natural to require that the multisignatures also satisfy this strong security notion.

Accordingly, existential unforgeability for group oriented setting means that the adversary attempts to generate a new multisignature without the knowledge of all private keys. We formalize this intuition as a chosen key model. In this model, the adversary is given a single public key, while the adversary is allowed to choose the key pairs of other signers of the group, and to ask the sign query for any multisignature under any joint public key. His goal is to generate a new multisignature under the joint public keys of the group composed of the given public key and the public keys of its choice.

We say that a multisignature scheme is existential unforgeable against adaptive chosen message attacks if no polynomial bounded adversary A has a non-negligible advantage against the challenger in the following game:

Setup: The challenger takes a security parameter 1^k as input and runs the randomized system parameter generation algorithm and the key generation algorithm to generate the system parameter P and a public key y. The challenger gives them to the adversary A.

Queries: Processing adaptively, the adversary A requests multisignatures queries (M_i, Y_i) under any joint public key Y_i on any message M_i of its choice, where the challenged public key y might be included in the joint public key Y_i.

Response: Finally, the adversary A outputs a new multisignature σ for a message M under a joint public key Y.

The adversary A wins the game if the output multisignature σ is nontrivial, i.e. it is not an answer of a sign query (M, Y) for the message M under a joint public key Y, and the joint public key Y is composed of the individual public keys $\{y_1, \ldots, y_{j-1}, y, y_{j+1}, \ldots, y_t\}$, where the individual public keys $\{y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_t\}$, $(j \in \{1, \ldots, t\})$, are chosen by the adversary A and y is the given public key.

The probability is over the random bits used by the challenger and the adversary.

Notice that our security model does not suffer from the same special limitation as the multisignature schemes proposed before. The adversary is given complete freedom in specifying the public keys but the given public key and is not enforced to disclose any knowledge of the corresponding private keys.

Notice that the Schnorr signature generation is not deterministic, there may be several signatures corresponding to a given message. Hence, our security model actually adopts the more liberal rule, which makes the adversary successful when it outputs a fresh signature of a given message different from previously obtained

signatures of the same message. Thus, our security model achieves non-malleability (NM) [21].

3.2 Security proof of the MSDL scheme

The security of the proposed MSDL scheme is based on the DL assumption.

Definition 2 (DL assumption)

A probabilistic algorithm A is said to (t, ε)-break DL in a group $G_{g,p}$, if on input $(g, p, q, y = g^x)$ and after running in time at most t, A solves the discrete logarithm problem $x = log_{g,p}y$ with probability at least ε, where the probability is over the uniform random choice of g from the group $G_{g,p}$, of x from Z_q^*, and the coin tosses of A. The (t, ε)-DL assumption on the group $G_{g,p}$ is that if no algorithm (t, ε)-breaks DL in $G_{g,p}$.

We have the following theorem about the security of the MSDL scheme.

Theorem. Let the hash functions H, F be random oracles. Then the Multisignature Signature scheme based on DL is existentially unforgeable against adaptive chosen message attacks (EUF-CMA) under the DL assumption.

Concretely, suppose that there is a EUF-CMA adversary A, which has an advantage ε^{MSDL} against the MSDL scheme of t signers and A runs in time at most t^{MSDL}. Suppose that A makes at most q_S sign queries, and at most q_H, q_F queries to the hash functions H, F, respectively. Then there is a DL algorithm B that has an advantage ε^{DL} in $G_{g,p}$ with running time t^{DL}, where:

$$\varepsilon^{MSDL} \le (4q_Fq_H)(\varepsilon^{DL})^{1/(3t+1)} + 1/q + q_S(q_F + q_S)/p \quad (1)$$

$$t^{MSDL} \ge t^{DL}/(2t) - 2q_SC_{exp}(G_{g,p}) \quad (2)$$

Here $C_{exp}(G_{g,p})$ denotes the computation cost of a long exponentiation in the group $G_{g,p}$.

Proof: We use the random oracle model to show that the proposed multisignature scheme is secure. Concretely, suppose that there is a EUF-CMA adversary A that has an advantage ε^{MSDL} against the MSDL scheme and A runs in time at most t^{MSDL}. Suppose that A makes at most q_H, q_F queries to the hash functions H and F respectively, and at most q_S queries to the sign oracle. Then there is a DL algorithm B that has an advantage ε^{DL} in $G_{g,p}$ with running time t^{DL}.

We show how to construct a DL algorithm B that uses A as a computer program to gain an advantage ε^{DL} for a DL problem with running time t^{DL}. The challenger takes a security parameter 1^k and runs the system parameter generation algorithm and the key generation algorithm to obtain the group $G_{p,g}$ and y. Its goal to output $x = \log_{,g,p}y \in Z_q^*$.

Algorithm B simulates the challenger and interacts with the adversary A in the following attack games:

Algorithm B gives the adversary A the resulting system parameter P, and y as the public key of an honest signer.

At any time, the adversary A can query hash oracles H or F. To response to these queries, B maintains two lists of tuples for the hash oracles H and F respectively. We refer to these lists as H-list and F-list. The contents of the two lists are "dynamic" during the attack games. Namely, when the games start, they are initially empty, but at the end of the games, they record all pairs of queries/answers.

Answering *H*-oracle Queries. When A queries the oracle H with some message $<y_1, y_2, \ldots, y_t>$, algorithm B responds as follows:

1. If the query $<y_1, y_2, \ldots, y_t>$ already appears on the H-list in some tuple $<<y_1, y_2, \ldots, y_t>, h>$, then algorithm B responds with $h = H(y_1, y_2, \ldots, y_t)$.
2. Otherwise, algorithm B picks a random h in Z_q^*, and responds with $h = H(y_1, y_2, \ldots, y_t)$ and adds the tuple $<< y_1, y_2, \ldots, y_t >, h>$ to the H-list.

Answering *F*-oracle Queries. When A queries the oracle F with some message $<M, R, h>$, algorithm B responds as follows:

1. If the query $<M, R, h>$ already appears on the F-list in some tuple $<<M, R, h>, f>$, then algorithm B responds with $f = F(M, r, h)$.
2. Otherwise, B checks if h is in the H-list, then generates a random $f \in Z_q^*$, responds with $f = F(M, R, h)$, and adds the tuple $<<M, R, h >, f>$ to the F-list.

Obviously, in two ways, h and f are uniform in Z_q^*, and they are independent of A's current view as required.

Answering sign queries. When the adversary A requests a signature for $<M, Y>$ under a joint public key Y, algorithm B responds to this query as follows:

1. B checks if Y is a valid joint public key: $Y = y_1y_2^{\,h}\ldots y_t^{\,h^{t-1}}$, where $h = H(y_1, y_2,\ldots, y_t)$.
2. Algorithm B chooses at random s, $f \in Z_q^*$, and computes $R = g^sY^f$.
3. If there exists a tuple $<<M, R, h>, f'>$ in the F-list with $f \ne f'$, B reports failure and terminates. (The probability of this unfortunate coincidence is at most $(q_F + q_S)/p$).
4. Otherwise, B responds with (s, f) to the adversary A, and adds the tuple $<<M, R, h>, f>$ to the F-list.

Obviously, the outputs of the simulated oracles are indistinguishable from those in the real attacks.

Finally, the adversary A returns a new valid message M and its multisignature (s, f) under the joint public key Y composed of public keys $\{y_1, \ldots, y_{j-1}, y, y_{j+1}, \ldots, y_t\}$, where y is the challenged public key and others are chosen by the adversary A such that

$$f = F(M, g^s(y_1\ldots y_{j-1}^{\,h^{j-2}}\, y^{h^{j-1}}\, y_{j+1}^{\,h^j}\ldots y_t^{\,h^{t-1}})^f, h),$$

where $h = H(y_1, y_2,\ldots, y_t)$

If the adversary A has not queried $F(M, R, h)$ or $H(y_1, y_2,\ldots, y_t)$, the probability

$\Pr[f = F(M, g^s(y_1...y_{j-1}^{h^{j-2}} y^{h^{j-1}} y_{j+1}^{h^j}...y_t^{h^{t-1}})^f, h)$, where $h = H(y_1, y_2,\ldots, y_t)] \le 1/q$,

since both the responses $F(M, R, H(y_1, y_2,\ldots, y_t))$ and $H(y_1, y_2,\ldots, y_t)$ are picked randomly.

Hence, with the probability $(1- 1/q)(\varepsilon^{\mathrm{MSDL}} - q_S(q_F + q_S)/p)$ the adversary A returns a new multisignature (s, f) such that

$$f = F(M, g^s(y_1...y_{j-1}^{h^{j-2}} y^{h^{j-1}} y_{j+1}^{h^j}...y_t^{h^{t-1}})^f, h),$$

where $h = H(y_1, y_2,\ldots, y_t)$

and the responses $F(M, R, H(y_1, y_2,\ldots, y_t))$ and $H(y_1, y_2,\ldots, y_t)$ are in the F-list and the H-list.

The verification equation is equivalent to the equation

$$g^s(y_1...y_{j-1}^{h^{j-2}} y^{h^{j-1}} y_{j+1}^{h^j}...y_t^{h^{t-1}})^{F(M,R,h)} = R,$$

where $h = H(y_1, y_2,\ldots, y_t)$,

where y is the challenged public key and other public keys $y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_t$ are chosen by the adversary A.

Since Pointcheval and Stern proposed the forking reduction proof [19], oracle replay techniques have been used to provide formal security proofs for ElGamal–like triplet signature schemes. In this proof, we are required to find $x = \log_{g,p} y$. It is a knowledge extraction problem. Hence, we try to use the oracle replay techniques to solve this DL problem.

We use $2t$ copies of the adversary A. In the attack games, the adversary A would ask H-query for $<y_1, \ldots, y_{j-1}, y, y_{j+1}, \ldots, y_t>$. We first guess a fixed index $1 \le u \le q_H$ and hope that $(y_1, \ldots, y_{j-1}, y, y_{j+1}, \ldots, y_t)$ happens to be uth H-query used in the forged multisignature. Then we guess a fixed index $1 \le v \le q_F$ and hope that $<M, R, h>$ happens to be vth F-query used in the forged multisignature. Note that A must ask for $H(y_1, \ldots, y_{j-1}, y, y_{j+1}, \ldots, y_t)$ before for $F(M, R, h)$.

Suppose that we make two good guesses by chance, denoted by the event GoodGuess. The probability of the event GoodGuess is

$$\Pr[\mathrm{GoodGuess}] = 1/(q_H q_F).$$

Hence, with the probability

$$\varepsilon = (1 - 1/q)(\varepsilon^{\mathrm{MSDL}} - q_S(q_F + q_S)/p)/(q_F q_H)$$
$$\ge (\varepsilon^{\mathrm{MSDL}} - 1/q - q_S(q_F + q_S)/p)/(q_F q_H)$$

the adversary A generates a new multisignature.

B gives the same system parameter, the same public key y and same sequence of random bits to the $2t$ copies of the adversary A, and responds with the same random answers to their queries for oracles until they at the same time ask the H-oracle query for $< y_1, \ldots, y_{j-1}, y, y_{j+1}, \ldots, y_t>$. This is the first forking point. At that point, B gives t independent random answers h_1, h_2 and h_t to the hash queries H in the $2t$ runs, the first two, gives h_1, the second two, gives h_2, and the last two, gives h_t.

Then B gives the first two copies of the adversary A same sequence of random bits, and the same random answers to their oracle queries until they both ask for $F(M_1, R_1, h_1)$. This is the second forking point. At that point, B gives two independent random answers f_{11} and f_{12} to the hash queries $F(M_1, R_1, h_1)$ in the first two runs. Similarly, B gives two independent random answers f_{21} and f_{22} to the hash queries $F(M_2, R_2, h_2)$ (the third forking point) in the second two runs, f_{t1} and f_{t2} to the hash queries $F(M_t, R_t, h_t)$ (the $(t + 1)$th forking point) in the last two runs. Thus, we would obtain $2t$ multisignatures, satisfying the following equations:

$$g^{s_{11}}(y_1 y_2^{h_1}...y_t^{h_1^{t-1}})^{f_{11}} = R_1 \quad (3)$$

$$g^{s_{12}}(y_1 y_2^{h_1}...y_t^{h_1^{t-1}})^{f_{12}} = R_1 \quad (4)$$

$$g^{s_{21}}(y_1 y_2^{h_2}...y_t^{h_2^{t-1}})^{f_{21}} = R_2$$

$$g^{s_{22}}(y_1 y_2^{h_2}...y_t^{h_2^{t-1}})^{f_{22}} = R_2$$

……

$$g^{s_{t1}}(y_1 y_2^{h_t}...y_t^{h_t^{t-1}})^{f_{t1}} = R_t$$

$$g^{s_{t2}}(y_1 y_2^{h_t}...y_t^{h_t^{t-1}})^{f_{t2}} = R_t$$

From these equations, we can derive $\log_{g,p} y$ as follows:

From eqn.(3) and eqn.(4), we can derive $a_1 = (s_{11} - s_{12})/(f_{12} - f_{11}) \pmod q$ such that

$$(y_1 y_2^{h_1}...y_t^{h_1^{t-1}}) = g^{a_1} \quad (t+1)$$

By the same way, we can derive $a_2, \ldots, a_t$, such that

$$(y_1 y_2^{h_2}...y_t^{h_2^{t-1}}) = g^{a_2} \quad (t+2)$$

……

$$(y_1 y_2^{h_t}...y_t^{h_t^{t-1}}) = g^{a_t} \quad (t+t)$$

Then from eqn.$(t+1)$– eqn.$(t+t)$, we have a system of equations

$$x_1 + h_1 x_2 + \ldots + h_1^{t-1} x_t = a_1 \pmod q$$

$$x_1 + h_2 x_2 + \ldots + h_2^{t-1} x_t = a_2 \pmod q$$

……

$$x_1 + h_t x_2 + \ldots + h_t^{t-1} x_t = a_t \pmod q$$

We can derive $x_j = \log_{g,p} y$ since h_1, h_2 and h_t are different from each other.

We use the "splitting lemma" [19] to compute the probability that A works as hoped. Let X be the set of possible sequences of random bits and random function values that take the adversary up to the first forking point where A asks for $H(y_1, \ldots, y_{j-1}, y, y_{j+1}, \ldots, y_t)$; let Y_1 be the set of possible sequences of random bits and random function values from the first forking point to the second forking point; let Z_1 be the set of possible sequences of random bits and random function values from the second forking point. By assumption, for any $x \in X$, $y \in Y_1$, $z \in Z_1$, the probability that A, supplied the sequences of random bits and random values $(x||y||z)$, generates a new multisignature is ε.

Suppose that the sequences of random bits and random function values supplied up to the first forking point in the simulations is a. By "splitting lemma", the probability that $\Pr\{a \in$ "good" subset $\Omega\} \ge \varepsilon/2$, and whenever $a \in \Omega$, $y \in Y_1$, $z \in Z_1$, the probability that A, supplied the sequences of random bits and random values $(a||y||z)$, produces a forgery is at least $\varepsilon/2$.

Suppose that the sequences of random bits and random function values supplied from the first forking point up to the second forking point in the simulations is b. Thus, the probability that $\Pr\{b \in$ "good" subset $\Omega'\} \geq \varepsilon/4$, and whenever $a \in \Omega$, $b \in \Omega'$, $z \in Z_1$, the probability that A, supplied the sequences of random bits and random values $(a||b||z)$, produces a forgery is at least $\varepsilon/4$.

By the same reason, we can compute the same probability for the other t - 1 cases.

Hence the probability that B solves the discrete logarithm in the $2t$ simulations is

$$\varepsilon^{\mathrm{DL}} \geq (\varepsilon)^{(3t+1)}/2^{(6t+1)} \geq ((\varepsilon^{\mathrm{MSDL}} - 1/q - q_S(q_F + q_S)/p)/(4q_Fq_H))^{(3t+1)}$$

$$\varepsilon^{\mathrm{MSDL}} \leq (4q_Fq_H)(\varepsilon^{\mathrm{DL}})^{1/(3t+1)} + 1/q + q_S(q_F + q_S)/p.$$

The time required to run one simulation is $t^{\mathrm{MSDL}} + 2q_SC_{exp}(G_{g,p})$.

The time required by the simulator B to solve the discrete logarithm $\log_{g,p}y$ is

$$t^{\mathrm{DL}} \leq 2t(t^{\mathrm{MSDL}} + 2q_SC_{exp}(G_{g,p})).$$

$$t^{\mathrm{MSDL}} \geq t^{\mathrm{DL}}/(2t) - 2q_SC_{exp}(G_{g,p}).$$

Q.E.D.

4 Conclusion

We have proposed a Multisignature Scheme based on Discrete Logarithms (MSDL). We show that this new scheme can resist existential forgeries against adaptive chosen-message attacks in the random oracle model. The main contribution is that our security model gets rid of the special security requirement on the generation of the participants' public keys. Thus the new multisignature scheme does not suffer from the problem identified by Micali et al., which is shared by many current multisignature schemes.

The second contribution is that our multisignature scheme can provide sequentially accountability, which means that not only individual signers can be identified from the multisignature, but also the order of accountability. That is, the first signer U_1 is responsible for the first partial multisignature, the second signer U_2 is responsible for the second partial multisignature, and the last signer U_t is responsible for the last multisignature. Thus, our scheme is robust. Notice that here sequentially accountability means that verifiers can demand that the signers are responsible for multisignatures according to the specified order $<U_1, U_2, \ldots, U_t>$ rather than that the signers could generate multisignatures only according to the specified order.

Furthermore, the proposed multisignature scheme is more efficient, since the size of multisignatures is the same as that of the underlying signatures, regardless of the number of participants. If the joint public key Y of a group of signers is precomputed, the computation cost for verification a multisignature is identical to those of an individual's signature. Thus the proposed multisignature scheme is optimal.

However, the forking reduction proof we use makes our proof inefficient. Strictly speaking, our proof is only loosely related to the DL problem according to Micali and Reyzin [22]. Therefore, our multisignature scheme is only applicable to the group of polynomial bounded signers.

Although the Schnorr scheme provably secure by oracle replay technique is only loosely related to DL problem, there has been not any efficient forgery attack without solving DL problem first. By similar reasons, our more loosely reduction would also provide users with somewhat security confidence that there is no efficient forgery algorithm without solving DL problem first.

We have proposed a Multisignature Scheme based on Discrete Logarithms (MSDL). We show that this new scheme can resist existential forgeries against adaptive chosen-message attacks in the random oracle model. The main contribution is that our security model gets rid of the special security requirement on the generation of the participants' public keys. Thus the new multisignature scheme does not suffer from the problem identified by Micali et al., which is shared by many current multisignature schemes.

The second contribution is that our multisignature scheme can provide sequentially accountability, which means that not only individual signers can be identified from the multisignature, but also the order of accountability. That is, the first signer U_1 is responsible for the first partial multisignature, the second signer U_2 is responsible for the second partial multisignature, and the last signer U_t is responsible for the last multisignature. Thus, our scheme is robust. Notice that here sequentially accountability means that verifiers can demand that the signers are responsible for multisignatures according to the specified order $<U_1, U_2, \ldots, U_t>$ rather than that the signers could generate multisignatures only according to the specified order.

Furthermore, the proposed multisignature scheme is more efficient, since the size of multisignatures is the same as that of the underlying signatures, regardless of the number of participants. If the joint public key Y of a group of signers is precomputed, the computation cost for verification a multisignature is identical to those of an individual's signature. Thus the proposed multisignature scheme is optimal.

However, the forking reduction proof we use makes our proof inefficient. Strictly speaking, our proof is only loosely related to the DL problem according to Micali and Reyzin [22]. Therefore, our multisignature scheme is only applicable to the group of polynomial bounded signers.

Although the Schnorr scheme provably secure by oracle replay technique is only loosely related to DL problem, there has been not any efficient forgery attack without solving DL problem first. By similar reasons, our more loosely reduction would also provide users with somewhat security confidence that there is no efficient forgery algorithm without solving DL problem first.

Acknowledgement

The author would like to thank the anonymous reviewers for their valuable comments and suggestions that improve the presentation of this paper significantly.

References

[1] Y. Desmelt (1988). Society and group oriented cryptography: A new concept, *Advances in Cryptology-Crypto'87*, LNCS 293, Springer, Berlin, pp. 120-127.

[2] L. Harn (1999). Digital multisignature scheme with distinguished signing authorities, *Electron. Lett* ., 35(4), pp.294-295.

[3] K. Itakura and K. Nakamura (1983). A public key cryptosystem suitable for digital multisignatures, *NEC Research & Development*, (71): pp. 1- 8.

[4] K. Ohta and T. Okamoto (1999(. Multi-signature schemes secure against active insider attacks, *IEICE Transaction on Fundamentals of Electronics communications and computer Science*, E82-A(1), pp.21-31.

[5] S. Micali, K. Ohta, and L. Reyzin (2001). Accountable-subgroup mulitisignatures, *In ACM Conference on Computer and communications Security*, 2001.

[6] L. Harn (1994). Group-oriented (*t, n*) threshold digital signature scheme and digital multisignature, *IEE Proc.-Comput. Digit. Tech*., 141(5), pp.307-313.

[7] C.-M. Li, T. Hwang, and N.-Y. Lee (1994). Threshold-multisignature schemes where suspected forgery implies traceability of adversarial shareholders, *Advances in Cryptology – Eurocrypt 94*, LNCS 950, Springer-Verlag, pp. 194-204.

[8] A. Boldyreva (2003). Threshold signature, multisignature and blind signature schemes based on the gap-Diffie-Hellman-group signature scheme, *Proceedings of PKC 2003*, LNCS 2567, Springer-Verlag, pp. 31-46.

[9] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters (2006). Sequential Aggregate Signature and Multisignature without Random Oracle, In *EUROCRYPTO'06*, LNCS 4004, Springer-Verlag, Berlin, pp. 465-485.

[10] PKCS#10: Certification request syntax standard, RSA Data Security, Inc., 2000.

[11] C. Adams, S. Farrell, T. Kause, T. Monen (2005). Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP), Internet, Engineering Task Force RFC 4210.

[12] J. Schaad (2005). Internet X.509 Public Key Infrastructure Certificate, Request Message Format, Internet Engineering Task Force RFC, 4211.

[13] T. Ristenpart and S. Yilek (2007). The Power of Proofs-of-Possession: Securing Multiparty Signatures against Rogue-Key Attacks. *in Advances in Cryptology – EUROCRYPT 2007*, LNCS 4515, Springer-Verlag, pp. 228–245.

[14] M. Bellare and G. Neven (2006). Multi-signatures in the plain public-key model and a generalized forking lemma, CCS 2006, ACM, pp.390-399.

[15] Zuhua Shao (2007). Self-certified Signatures Based on Discrete Logarithms, *in Proceedings of WAIFI* 2007, LNCS 4547, Springer-Verlag, pp.252-263.

[16] Zuhua Shao (2007). Self-certified signature scheme from pairings, *Journal of Systems and Software*, 80(3), pp. 388-395.

[17] Zuhua Shao (2009). Dynamic and efficient joint encryption scheme in the plain public key model, *Computers and Electrical Engineering*, 35(1), pp. 189-196.

[18] C. P. Schnorr (1991). Efficient signature generation by smart cards, *Journal of Cryptology*, 3(3), pp.161-174.

[19] D. Pointcheval and J. Stern (2000). Security arguments for digital signatures and blind signatures, *Journal of Cryptology,* 13(3), pp. 361-396.

[20] S. Goldwasser, S. Micali, and R. Rivest (1988). A digital signature scheme secure against adaptive chosen-message attacks, *SIAM Journal on Computing*, 17(2), pp.281-308.

[21] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway (1998). Relation among notions of security for public-key encryption schemes, In *Crypto'98*, LNCS 1462, Springer-Verlag, Berlin, pp. 26-45.

[22] S. Micali and L. Reyzin (2002). Improving the exacting security of digital signature schemes, *Journal of Cryptology*, 15(1), pp.1-18.