A Structured ElGamal-Type Multisignature Scheme

Mike Burmester[1], Yvo Desmedt[2], Hiroshi Doi[3], Masahiro Mambo[4], Eiji Okamoto[5], Mitsuru Tada[6], and Yuko Yoshifuji[6,*]

[1] Information Security Group, Royal Holloway, University of London
m.burmester@rhbnc.ac.uk
[2] Department of Computer Science, Florida State University
desmedt@cs.fsu.edu
[3] Department of Mathematics, Faculty of Science, Okayama University
hdoi@math.okayama-u.ac.jp
[4] Education Center for Information Processing & Graduate School of Information Sciences, Tohoku University, mambo@ecip.tohoku.ac.jp
[5] Center for Cryptography, Computer and Network Security University of Wisconsin, Milwaukee, okamoto@cs.uwm.edu
[6] School of Information Science, Japan Advanced Institute of Science and Technology, {mt,yosifuji}@jaist.ac.jp

Abstract. We propose a structured multisignature scheme which is based on a modified ElGamal signature scheme and analyze its security. The structure takes into account the order of the signers. With serial structures, different signing orders produce different multisignatures. In contrast, with parallel structures the multisignatures are independent of the signing order. Our structured multisignatures can deal with structures which are composed of serial and parallel signing orders. We give reductions for the security of the proposed scheme, and for the specified order of the signers in the serial and mixed cases.

Keywords: Multisignature, Structured multisignature, Group structure, Series-parallel graph, ElGamal signature.

1 Introduction

When multiple entities sign a document, the signing order often reflects the role of each signer and signatures with different signing orders are regarded as multisignatures with different meanings. A typical example is a multisignature in a company. Usually, the head of a section should sign a document after the other members of the section have signed it. Other examples involve banks and command structures.

Of course the signing order is of little relevance to authentication. However there are other aspects one should take into account, such as the liability of the

*Present affiliation of the last author: NTT Service Integration Laboratories

H. Imai, Y. Zheng (Eds.): PKC 2000, LNCS 1751, pp. 466–483, 2000.

signers. This could be related to their ranking and determined by the signing order. Depending on the application, a different signing order may be required. Moreover, each signer may only wish to sign after the previous signers have done so, and the verifier may require that the correct order has been adhered to.

We can consider two cases for the signing order: 1) *serial signing*, in which the signing order can be detected by a verifier from a signature, 2) *parallel signing*, in which the signing order cannot be detected by a verifier from a signature. A multisignature scheme is said to be *structured* if the group of signers is structured. The structure takes into account the signing order of the entities of the signing groups, if this is serial. In this case different signing orders yield different multisignatures. In contrast, the multisignature for parallel signing orders is group independent. Our structured multisignature scheme can deal with group structures which are composed of serial and parallel signing orders.

In this paper we propose a structured multisignature scheme based on a modified ElGamal signature scheme and give reductions for its security, including the security for the specified order of the signers. This paper is organized as follows. After this introduction we discuss related work in Section 2. In Section 3 we represent signer groups by series-parallel graphs and in Section 4 we give definitions and specify our setting. In Section 5 we define our model, and the reductions and functions which we shall use for the security proof. After describing the basic modified ElGamal signature scheme in Section 6, we present a serial multisignature scheme in Section 7, a parallel multisignature scheme in Section 8, and a mixed multisignature scheme in Section 9. In Section 10 we modify our scheme to get a more efficient scheme by exploiting the decomposition tree of series-parallel graphs. Finally, conclusions are given in Section 11.

2 Related Work

So far there are several proposals for multisignature schemes, as for example in [Oka88,OO91,HZ92,Shim94,DOMU94,HMP95,DOM98,OO99]. Multisignature schemes have advantages over multiple iterations of a single signature scheme either in the length of the signature or in the computational cost of generating and verifying the signature. Another property one should pay attention to is the signing order. Some multisignature schemes are independent of the order of the signers [OO91,HZ92,Shim94,HMP95], while other schemes are order specified [Oka88,DOMU94,DOM98,OO99].

Among the schemes which are sensitive to the signing order, the scheme in [Oka88] is constructed using bijective functions such as the RSA signing function [RSA78]. The scheme in [DOMU94] is also based on the RSA signature scheme, but this scheme has been shown to be insecure. A modified version of this scheme is proposed in [DOM98], however its security analysis is not complete. Another scheme [DOM98] uses an ElGamal-type signature [ElG85,HMP94], but as with the RSA based scheme, its security analysis is not complete.

The Ohta-Okamoto schemes in [OO91] are converted from corresponding identification schemes such as the Schnorr scheme [Schn91] and the Fiat-Shamir

scheme [FS86]. For single signatures, the security proof for the Schnorr signature scheme and a modified ElGamal signature scheme are given in [PS96] using the Random Oracle model [BR93] (however there are some problems with this model [CGH98]). Ohta-Okamoto prove the security of their schemes using an ID reduction to an identification scheme [OO91]. For single signatures their proof gives a tighter reduction than [PS96]. However the Ohta-Okamoto schemes do not consider mixed structures composed of serial and parallel signing orders. In particular, the security of multisignature schemes for such a composite group structure has not been proven.

A structured multisignature scheme based on an ElGamal-type signature is also proposed in [DOM98] but its security for the signing order is not complete.

3 Series-Parallel Graph

We represent the group of signers by a graph. We consider directed graphs which satisfy the following conditions:

(G-1) Each signer corresponding to an edge appears only once in the graph.
(G-2) The graphs are restricted to series-parallel type graphs.

A *series-parallel graph* is a graph which is generated by series and parallel compositions of series-parallel graphs. The simplest series-parallel graph is a base graph of two vertices and an edge. We refer the reader to [Len90] for more details. The set of all graphs satisfying the above conditions is denoted by SPG.

In Figure 1 we illustrate a typical series-parallel graph. A series-parallel graph can be decomposed into a unique tree called the decomposition tree of the graph. In Figure 2 we show the decomposition tree of the graph in Figure 1. The labels of the vertices show the type of composition of the subordinate edges. Vertices labeled SC show series compositions whereas vertices labeled PC show parallel compositions. Series-parallel graphs can be recognized in linear time and the decomposition tree of a series-parallel graph can be found in linear time [Len90].

4 Notation and Definition

We will use a discrete logarithm setting. Let p and q be appropriate primes such that $q|(p-1)$, and let $g \in \mathrm{Z}_p^*$ be a q-th root of 1 modulo p. P_i is a signer with secret key $a_i \in \mathrm{Z}_q^*$, and k_i is a random number in Z_q^* selected by P_i. $M \in \{0,1\}^*$ is the message to be signed. $a \in_R A$ indicates that the element a is selected from the set A uniformly at random.

Constructible groups. Let $\mathcal{G} \subset SPG$ be a collection of signer groups, $G_k \in \mathcal{G}$ a signer group represented by a series-parallel graph, P_i a signer in G_k, $a_i \in \mathrm{Z}_q^*$ the secret key of P_i, and $y_i^{(k)} \in \mathrm{Z}_p^* \backslash \{1\}$ the partial public key of P_i corresponding to a_i. We say that $\mathcal{G}$ is an *available group* of a signature

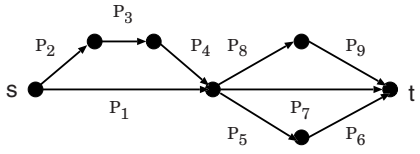

Fig. 1. A series-parallel graph G

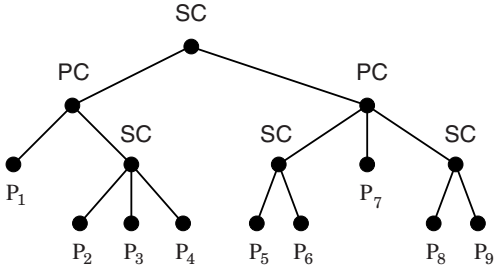

Fig. 2. The decomposition tree of G

scheme if there is a set of secret keys $\{a_i | P_i \in G_k, G_k \in \mathcal{G}\}$ of this scheme such that,

$$y_i^{(k)} \neq y_j^{(l)} \quad \text{for } \forall G_k, G_l \in \mathcal{G},\ \forall (P_i, P_j) \in (G_k, G_l),\ i \neq j. \tag{1}$$

We allow $k = l$ in (1). The set of keys $\{a_i | P_i \in G_k, G_k \in \mathcal{G}\}$ of an available group is called an *appropriate secret-key set* for $\mathcal{G}$ or simply, *appropriate for* $\mathcal{G}$. If it is feasible to generate an appropriate secret-key set from an available group $\mathcal{G}$, then $\mathcal{G}$ is called a *constructible group*.

Signature structure. When signers sign a document sequentially, the signing order has a special meaning. We call such a signature structure, *serial*. On the other hand, when all or a part of the signers create a partial signature for a document in arbitrary order and a complete signature is created from these partial signatures, the signing order of the signers has no meaning. We call such a signature structure, *parallel*.

The signer group of a serial structure is denoted by $(P_1, P_2, \ldots, P_n)$. This means that P_1 signs first, P_2 second and so on. After all $P_1, P_2, \ldots, P_{n-1}$ have signed, P_n signs. For simplicity, we write

$$G_{i,\ldots,j} = (\ldots, P_{i-1}, P_i, \ldots, P_{j-1}, P_j, \ldots),$$
$$G_{j,\ldots,i} = (\ldots, P_{i-1}, P_j, \ldots, P_{j-1}, P_i, \ldots),$$

where $1 \leq i < j \leq n$.

The signer group of a parallel structure is denoted by $\wedge(P_1, P_2, \ldots, P_n)$.

In the scheme we propose the partial signature created by signer P_i is a triple $(s_i, r_i, r) \in \mathrm{Z}_q \times \mathrm{Z}_p^* \times \mathrm{Z}_p^*$. The partial signature created by the last signer P_n is the multisignature of the entire group. For (s_n, r_n, r) we have $r_n = r$, so that the final form of the multisignature is (s_n, r_n).

Let $i, j \in \{1, 2, \ldots, n\}$, $i < j$, and $K \subset \{1, 2, \ldots, n\}$. For a variable v, let $v_{[i,j,K]}$ denote the sequence $v_i, v_{i+1}, \ldots, v_j$ in which all v_l, $l \in K$, are excluded. $v_{[1,n,\phi]}$ is simply denoted by $v_{[i,j]}$.

For simplicity, $(\mathrm{mod} p)$ is sometimes omitted in this paper.

Hash function. In this paper we shall use a *target-collision intractable* hash function, hash, which takes values of arbitrary length as input and outputs a value in $\mathbf{Z}_q^*$. (A hash function is target-collision intractable if for a given x it is infeasible to compute a y such that $\mathrm{hash}(x) = \mathrm{hash}(y)$.)

5 Attacks and Reductions

Attack model. For our security analysis, we consider a model in which all insiders (signers) except an honest signer may collude in the attack. The attackers give a partial signature of a message M to the honest signer P_i and obtain a valid partial signature by P_i of M. With this information, the attackers try to obtain forged signatures or to derive P_i's secret key.

Reductions. We use the following reductions in our security proof (*cf.* [Woll87]):

- $f \leq_m^p g$ denotes that f reduces to g with respect to the polynomial-time many-one ($\leq_m^p$-) reducibility. (That is, there exists a polynomial-time computable function h such that $f(x) = g(h(x))$.)
- $f \leq_{k-tt}^p g$ denotes that f reduces to g with respect to the polynomial-time truth-table ($\leq_{k-tt}^p$-) reducibility. In a $\leq_{k-tt}^p$-reducibility only k non-adaptive queries to an oracle are allowed. If we do not wish to stress the number of queries, we simply write $f \leq_{tt}^p g$.
- $f \leq_T^p g$ denotes that f reduces to g with respect to the polynomial-time Turing ($\leq_T^p$-) reducibility. In a $\leq_T^p$-reducibility, a polynomial-time bounded oracle Turing machine can access an oracle adaptively.

For $type \in \{m, k-tt, T\}$, if $f \leq_{type}^p g$ and $g \leq_{type}^p f$, then $f \equiv_{type}^p g$.

We use the following functions for the proof of security.

- $\mathbf{DLP}_q(X, g, p, q)$ is a function that on input two primes p, q with $q|(p-1)$, $X \in \mathrm{Z}_p^*$, $g \in \mathrm{Z}_p^*$ of order q, outputs $x \in \mathrm{Z}_q$ such that $X \equiv g^x (\mathrm{mod} p)$, if such an x exists. This function solves the discrete logarithm problem in a subgroup of prime order q.
- $\mathbf{Forge}(y, r, M, g, p, q)$ is a function that on input two primes p, q with $q|(p-1)$, $y \in \mathrm{Z}_p^*$, $r \in \mathrm{Z}_p^*$ satisfying $gcd(r, q) = 1$, $M \in \{0,1\}^*$, $g \in \mathrm{Z}_p^*$ of order q, outputs $s' \in \mathrm{Z}_q$ with $g^{s'} \equiv y r^{r \cdot \mathrm{hash}(r,M)} (\mathrm{mod} p)$ for a given hash function hash(), if such an s' exists.
  This function forges a valid signature s' using the pair (M, r) of a message M and a number r and available public information, but does not use the signer's secret key.
- $\mathbf{Secret1}(y, r, s, M, g, p, q)$ is a function that on input two primes p, q with $q|(p-1)$, $y \in \mathrm{Z}_p^*$, $r \in \mathrm{Z}_p^*$ with $gcd(r, q) = 1$, $s \in \mathrm{Z}_q$, $M \in \{0,1\}^*$, $g \in \mathrm{Z}_p^*$ of order q, outputs $(a, k) \in (\mathrm{Z}_q^*, \mathrm{Z}_q^*)$ such that

$$y \equiv g^a,\ r \equiv g^k,\ s \equiv a + kr \cdot \mathrm{hash}(r, M) \,(\mathrm{mod} q)$$

for a given hash function hash(), if such a pair of (a, k) exists.

This function computes the secret key a of P and the random number k using a valid signature (r, s) by P and available public information.

- **Secret2** $[(P_1, P_2, \ldots, P_n)]\,(y_i, r_i, \tilde{r}, s_i, \tilde{k}, a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]}, M, g, p, q)$ is a function that on input two primes p, q with $q|(p-1)$, $y_i, r_i, \in \mathrm{Z}_p^*$, $\tilde{r} \in \mathrm{Z}_p^*$ satisfying $gcd(\tilde{r}, q) = 1$, $s_i,\ \tilde{k} \in \mathrm{Z}_q$, $a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]} \in \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$, $M \in \{0,1\}^*$, $g \in \mathrm{Z}_p^*$ of order q, outputs $(a_i, k_i) \in (\mathrm{Z}_q^*, \mathrm{Z}_q^*)$ such that $a_{[1,n]}$ is appropriate for $(P_1, P_2, \ldots, P_n)$, and

$$\alpha = (\cdots((a_1+1)a_2+1)\cdots)a_{i-1} \,(\mathrm{mod} q),$$
$$y_i \equiv g^{(\alpha+1)a_i},\ r_i \equiv (g^{\tilde{k}})^{a_i} \cdot g^{k_i},$$
$$s_i \equiv ((\alpha + \tilde{k}\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M)) + 1)a_i + k_i\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M) \,(\mathrm{mod} q),$$

for a given hash function hash(), if such a pair of (a_i, k_i) exists, and otherwise outputs $\perp$.

This function computes the pair (a_i, k_i) consisting of the secret key a_i and the random number k_i of an honest signer P_i in the following situation. All signers except P_i collude to compute (a_i, k_i), using any available public information and a valid partial signature (r_i, s_i, r) created by P_i in a serial structure for a message M. The colluding entities do not need to follow the signature creation procedure for computing these values. A partial signature $(\tilde{r}_{i-1}, \tilde{s}_{i-1}, \tilde{r})$ given to P_i satisfies $y_{i-1}\tilde{r}_{i-1}^{\tilde{r}\mathrm{hash}(\tilde{r},M)} \equiv g^{\alpha} \cdot g^{\tilde{k}\tilde{r}\cdot\mathrm{hash}(\tilde{r},M)} \equiv g^{\tilde{s}_{i-1}}$, where $\tilde{r}_{i-1} \equiv g^{\tilde{k}}, \tilde{s}_{i-1} \equiv \alpha + \tilde{k}\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M) \,\mathrm{mod} q$.

- **Flip**$[G_{i,\ldots,j}, G_{j,\ldots,i}](y_{i-1}, y_i, y', \tilde{r}_{i-1}, r_i, \tilde{r}, \tilde{s}_{i-1}, s_i, \tilde{k}, a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]}, M, g, p, q)$ is a function that on input two primes p, q with $q|(p-1)$, y_{i-1}, y_i, y', $\tilde{r}_{i-1}, r_i \in \mathrm{Z}_p^*$, $\tilde{r} \in \mathrm{Z}_p^*$ satisfying $gcd(\tilde{r}, q) = 1$, $\tilde{s}_{i-1}, s_i, \tilde{k} \in \mathrm{Z}_q$, $a_{[1,n,\{i\}]}$, $k_{[1,n,\{i\}]} \in \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$, $M \in \{0,1\}^*$, $g \in \mathrm{Z}_p^*$ of order q, outputs $(s', r') \in (\mathrm{Z}_q, \mathrm{Z}_p^*)$ such that $s' \in \mathrm{Z}_q$, $r' \in \mathrm{Z}_p^*$ with $gcd(r', q) = 1$, $\alpha' \in \mathrm{Z}_q^*$, $a_i, k_i \in \mathrm{Z}_q^*$, $a_{[1,n,\{i\}]}$ is appropriate for $[G_{i,\ldots,j}, G_{j,\ldots,i}]$, and

$$\alpha = (\cdots((a_1+1)a_2+1)\cdots)a_{i-1} \,(\mathrm{mod} q),$$
$$\beta = \alpha a_j a_{i+1} \cdots a_{j-1}a_{j+1} \cdots a_n + a_j a_{i+1} \cdots a_{j-1}a_{j+1} \cdots a_n$$
$$+ a_{i+1} \cdots a_{j-1}a_{j+1} \cdots a_n + \cdots$$
$$+ a_{j-1}a_{j+1} \cdots a_n + a_{j+1} \cdots a_n \,(\mathrm{mod} q),$$
$$\gamma = a_{j+1} \cdots a_n + \cdots + a_n \,(\mathrm{mod} q),$$
$$y_{i-1} \equiv g^{\alpha},\ y_i \equiv (y_{i-1} \cdot g)^{a_i},\ y' \equiv g^{\beta a_i + \gamma},$$
$$\tilde{r}_{i-1} \equiv g^{\tilde{k}},\ r_i \equiv \tilde{r}_{i-1}^{a_i} \cdot g^{k_i},\ r' \equiv g^{\alpha'},$$
$$\tilde{s}_{i-1} \equiv \alpha + \tilde{k}\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M) \,(\mathrm{mod} q),$$
$$s_i \equiv (\tilde{s}_{i-1} + 1)a_i + k_i\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M) \,(\mathrm{mod} q),$$
$$s' \equiv (\beta a_i + \gamma) + \alpha' r' \cdot \mathrm{hash}(r', M) \,(\mathrm{mod} q)$$

for a given hash function hash(), if such a pair of (s', r') exists.

This function forges a signature of a signer structure $G_{j,\ldots,i}$ from the signature of a signer structure $G_{i,\ldots,j}$ in the following situation. All signers except the honest signer P_i collude in the forgery, using public information and a valid partial signature (r_i, s_i, r) created by P_i for the message M. The colluding entities do not need to follow the signature creation procedure for

computing these values. A partial signature $(\tilde{r}_{i-1}, \tilde{s}_{i-1}, \tilde{r})$ given to P_i satisfies $y_{i-1}\tilde{r}_{i-1}^{\tilde{r}\text{hash}(\tilde{r},M)} \equiv g^{\alpha} \cdot g^{\tilde{k}\tilde{r}\cdot\text{hash}(\tilde{r},M)} \equiv g^{\tilde{s}_{i-1}}$, where $\tilde{r}_{i-1} \equiv g^{\tilde{k}}, \tilde{s}_{i-1} \equiv \alpha + \tilde{k}\tilde{r} \cdot \text{hash}(\tilde{r}, M) \pmod{q}$.

- **Forge**$[G_o, G_f](y_{i-1}, y_i, y', \tilde{r}_{i-1}, r_i, \tilde{r}, \tilde{s}_{i-1}, s_i, \tilde{\beta}, a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]}, M, g, p, q)$ is a function that on input two primes p, q with $q|(p-1)$, $y_{i-1}, y_i, y' \in \mathrm{Z}_p^*$, $\tilde{r}_{i-1} \in \mathrm{Z}_p^*$ satisfying $gcd(\tilde{r}_{i-1}, q) = 1$, $r_i \in \mathrm{Z}_p^*$ satisfying $gcd(r_i, q) = 1$, $\tilde{s}_{i-1}, s_i \in \mathrm{Z}_q$, $\tilde{\beta} \in \mathrm{Z}_q^*$, $a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]} \in \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$, $M \in \{0,1\}^*$, $g \in \mathrm{Z}_p^*$ of order q, outputs $(s', r') \in (\mathrm{Z}_q, \mathrm{Z}_p^*)$ such that $s' \in \mathrm{Z}_q$, $r' \in \mathrm{Z}_p^*$ satisfying $gcd(r', q) = 1$, $\beta' \in \mathrm{Z}_q$, $a_i, k_i \in \mathrm{Z}_q^*$, $a_{[1,n]}$ is appropriate for $\mathcal{G} \subset SPG$, function f_α is determined from $G_o \in \mathcal{G}$, $f_{\alpha_1'}$ and $f_{\alpha_2'}$ are determined from $G_f \in \mathcal{G}$, $P_i \in G_o, P_i \in G_f$, and
  $y_{i-1} \equiv g^{\alpha}$, $y_i \equiv (y_{i-1} \cdot g)^{a_i}$, $y' \equiv g^{\alpha_1' a_i + \alpha_2'}$,
  $\alpha = f_\alpha(a_{[1,n,\{i\}]})$, $\alpha_1' = f_{\alpha_1'}(a_{[1,n,\{i\}]})$, $\alpha_2' = f_{\alpha_2'}(a_{[1,n,\{i\}]})$,
  $\tilde{r}_{i-1} \equiv g^{\tilde{\beta}}$, $r_i \equiv \tilde{r}_{i-1}^{a_i} \cdot g^{k_i}$, $r' \equiv g^{\beta'}$,
  $\tilde{s}_{i-1} \equiv \alpha + \tilde{\beta}\tilde{r}\text{hash}(\tilde{r}, M) \pmod{q}$,
  $s_i \equiv (\tilde{s}_{i-1} + 1)a_i + k_i\tilde{r}\text{hash}(\tilde{r}, M) \pmod{q}$,
  $s' \equiv \alpha_1' a_i + \alpha_2' + \beta' r' \cdot \text{hash}(r', M) \pmod{q}$,
  for a given hash function hash(), if such a pair of (s', r') exists.
  This function forges the signature of a signer structure G_f from the signature of a signer structure G_o in the following situation. All signers except the honest signer P_i collude in the forgery using public information and a valid partial signature (r_i, s_i, r) created by P_i for the message M. The colluding entities do not need to follow the signature creation procedure for computing these values. A partial signature $(\tilde{r}_{i-1}, \tilde{s}_{i-1}, \tilde{r})$ given to P_i satisfies $y_{i-1}\tilde{r}_{i-1}^{\tilde{r}\text{hash}(\tilde{r},M)} \equiv g^{\alpha} \cdot g^{\tilde{\beta}\tilde{r}\cdot\text{hash}(\tilde{r},M)} \equiv g^{\tilde{s}_{i-1}}$, where $\tilde{r}_{i-1} \equiv g^{\tilde{\beta}}, \tilde{s}_{i-1} \equiv \alpha + \tilde{\beta}\tilde{r} \cdot \text{hash}(\tilde{r}, M) \pmod{q}$.
- $FC(v, w)$ is a function which on input v, w, outputs the first component v.

6 Modified ElGamal Signature

We use a modified version [YL93] of the ElGamal signature scheme [ElG85]. To explain this scheme we consider a single entity signer group consisting of only P. The message is M. Let $a \in \mathrm{Z}_q^*$ be the secret key and $y = g^a \pmod{p}$ the public key of P.

Basic signature protocol

1. P selects $k \in_R \mathrm{Z}_q^*$ and computes $r = g^k \bmod p$. P repeats this process until r satisfies $gcd(r, q) = 1$.
2. P computes $s = a + kr \cdot \text{hash}(r, M) \pmod{q}$. The signature of M is (s, r).
3. The signature (s, r) is verified by checking that $g^s \stackrel{?}{\equiv} yr^{r\cdot\text{hash}(r,M)} \pmod{p}$.

A valid signature passes the verification check because

$$g^s \equiv g^a \cdot g^{kr\cdot\text{hash}(r,M)} \equiv yr^{r\cdot\text{hash}(r,M)} \pmod{p}.$$

6.1 Security Considerations

Generation of a forged signature (r, s): To show the difficulty of forging a signature we first consider the case when r is given.

- **Forge** $\leq_m^p$ **DLP**$_q$: **Forge**$(y, r, M, g, p, q) =$ **DLP**$_q(yr^{r\cdot \mathrm{hash}(r,M)}, g, p, q)$
- **DLP**$_q \leq_m^p$ **Forge** : **DLP**$_q(g^s, g, p, q) =$ **Forge**(y, r, M, g, p, q),
  where $r \in_R \mathrm{Z}_p^*$, $gcd(r, q) = 1$, $M \in_R \{0,1\}^*, y = g^s \cdot r^{-r\cdot \mathrm{hash}(r,M)}$.
  Observe that

$$y \cdot r^{r\cdot \mathrm{hash}(r,M)} \equiv (g^s \cdot r^{-r\cdot \mathrm{hash}(r,M)}) \cdot r^{r\cdot \mathrm{hash}(r,M)} \equiv g^s.$$

Thus **DLP**$_q \equiv_m^p$ **Forge**.
This proof shows only one aspect of the difficulty of signature forgery.

Security of secret values

- **Secret1** $\leq_{2-tt}^p$ **DLP**$_q$:
  Secret1$(y, r, s, M, g, p, q) =$ (**DLP**$_q(y, g, p, q)$, **DLP**$_q(r, g, p, q)$)
- **DLP**$_q \leq_{1-tt}^p$ **Secret1** : **DLP**$_q(g^a, g, p, q) = (r \cdot \mathrm{hash}(r, M))^{-1} \cdot$
  $\cdot(-s(1 - r \cdot \mathrm{hash}(r, M)) + FC($**Secret1**$(y, r, s, M, g, p, q))) \pmod q$,
  where $s \in_R \mathrm{Z}_q^*, M \in_R \{0,1\}^*$, $r = (g^a)^{-1} \cdot g^s$, $gcd(r \cdot \mathrm{hash}(r, M), q) = 1$, and
  $y = g^s \cdot r^{-r\cdot \mathrm{hash}(r,M)}$.
  Observe that

$$y \cdot r^{r\cdot \mathrm{hash}(r,M)} = g^s,$$

  and that

$$y \equiv g^s \cdot r^{-r\cdot \mathrm{hash}(r,M)} \equiv g^s \cdot g^{(-a+s)(-r\mathrm{hash}(r,M))}$$
$$\equiv g^{s(1-r\mathrm{hash}(r,M))+ar\cdot \mathrm{hash}(r,M)}.$$

Thus **DLP**$_q \equiv_{tt}^p$ **Secret1**.

7 Serial Multisignature Scheme

A partial public key y_i of P_i of a multisignature scheme must belong to $\mathrm{Z}_p^* \backslash \{1\}$ and must also satisfy the condition (1). This is achieved by selecting appropriate secret-key groups.

For example, an appropriate secret-key group $(a_1, \ldots, a_n)$ for $(P_1, \ldots, P_n)$ is obtained by choosing $(\alpha_{i-1} + 1)a_i \neq a_j$ for $i = 2, \ldots, n$, $j = 1, \ldots, i-1$, where $\alpha_{i-1} = (\ldots((a_1 + 1)a_2 + 1)\ldots)a_{i-1} \pmod q$ (See Section 7.2).

Suppose n signers P_i sign a message M sequentially. Let $a_i \in \mathrm{Z}_q^*$ be the secret key of P_i. Then the partial public key y_i of P_i in $(P_1, \ldots, P_n)$, $i = 1, 2, \ldots, n$, is computed as follows:

$$y_1 = g^{a_1}, \; y_i = (y_{i-1} \cdot g)^{a_i}.$$

The public key of the group $(P_1, \ldots, P_n)$ is $y = y_n (\bmod p)$. The secret and public keys are generated either by a trusted center or by each signer using distributed

protocols. In the latter case, each signer repeatedly selects his secret key until the keys form an appropriate set. Furthermore, each signer needs to prove that they know the secret key which corresponds to the partial public key computed. For this purpose they may use the protocol in [Schn91], which does not reveal any secret information.

Serial multisignature protocol

1. **Generation of** r**.** $P_1, \ldots, P_n$ generate r together as follows.
   (a) P_1 selects $k_1 \in_R \mathrm{Z}_q^*$ and computes $r_1 = g^{k_1} \bmod p$. P_1 repeats this process until r_1 satisfies $gcd(r_1, q) = 1$.
   (b) For $i \in \{2, \ldots, n\}$:
   P_{i-1} gives r_{i-1} to P_i.
   P_i selects $k_i \in_R \mathrm{Z}_q^*$ and computes $r_i = {r_{i-1}}^{a_i} g^{k_i} \bmod p$; P_i repeats this process until r_i satisfies $gcd(r_i, q) = 1$.
   (c) $r = r_n$.
2. **Generation of** s**.** $P_1, \ldots, P_n$ generate s together as follows.
   (a) P_1 computes $s_1 = a_1 + k_1 r \cdot \mathrm{hash}(r, M) \ (\mathrm{mod} q)$.
   (b) For $i \in \{2, \ldots, n\}$:
   P_{i-1} gives s_{i-1} to P_i.
   P_i verifies that $g^{s_{i-1}} \equiv y_{i-1} r_{i-1}^{r \cdot \mathrm{hash}(r,M)} \ (\mathrm{mod} p)$ and if so computes

$$s_i = (s_{i-1} + 1)a_i + k_i r \cdot \mathrm{hash}(r, M) \ \ (\mathrm{mod} q).$$

   (c) $s = s_n$.

   Multisignature. (r, s) is the multisignature on M by $(P_1, \ldots, P_n)$.
   Verification. A multisignature (r, s) is verified by checking the congruence

$$g^s \stackrel{?}{\equiv} y r^{r \cdot \mathrm{hash}(r,M)} \ \ (\mathrm{mod} p).$$

7.1 Security Considerations

Security of a_i **and** k_i**.** We consider the case when all the signers P_j except an honest signer P_i, $i > 1$, collude to derive a_i and k_i from any information they can obtain.

Let α be the exponent of y_{i-1} to the base g modulo p, *i.e.* $y_{i-1} = g^{\alpha}$, $\alpha = (\ldots((a_1 + 1)a_2 + 1)\ldots)a_{i-1} \ (\mathrm{mod} q)$. Likewise, let $\tilde{k}$ be the exponent of $\tilde{r}_{i-1}$ to the base g modulo p, *i.e.* $\tilde{r}_{i-1} = g^{\tilde{k}}$.

– $\mathbf{Secret2}[(P_1, \ldots, P_i, \ldots, P_n)] \leq_T^p \mathbf{DLP}_q$:
$\mathbf{Secret2}[(P_1, \ldots, P_i, \ldots, P_n)](y_i, r_i, \tilde{r}, s_i, \tilde{k}, a_{[1,n,\{i\}]} k_{[1,n,\{i\}]}, M, g, p, q) = (A, \mathbf{DLP}_q(r_i \cdot (g^{-A\tilde{k}}), g, p, q))$, where $A = \mathbf{DLP}_q(y_i, g^{\alpha+1}, p, q)$.

– $\mathbf{DLP}_q \leq_m^p \mathbf{Secret2}[(P_1, \ldots, P_i, \ldots, P_n)]$:
At first $a_{[1,i-1]} \in \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$ is selected so that it is appropriate for $(P_1, \ldots, P_i, \ldots, P_{i-1})$. Then $a_{[i+1,n]} \in_R \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$ is randomly selected. This procedure is repeated until the set of keys $a_{[1,n,\{i\}]}$ and a_i becomes appropriate for $(P_1, \ldots, P_i, \ldots, P_n)$. We can recognize that $a_{[1,n]}$ is not appropriate from the output of $\mathbf{Secret2}[(P_1, \ldots, P_i, \ldots, P_n)]$. In Section 7.2 we shall see that this procedure is feasible (with probability almost one) if n is bounded by a polynomial in $|p|$. When $a_{[1,n]}$ becomes appropriate, $\mathbf{DLP}_q(g^{a_i}, g, p, q)$ is computed in the following way:
$\mathbf{DLP}_q(g^{a_i}, g, p, q) = FC(\mathbf{Secret2}[(P_1, \ldots, P_i, \ldots, P_n)]((g^{a_i})^{\alpha+1}, (g^{a_i})^{\tilde{k}} \cdot g^{k_i}, \tilde{r},\ k_i\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M) \,(\mathrm{mod}q),\ \tilde{k}, a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]}, M, g, p, q))$,
where $\tilde{r} \in_R \mathrm{Z}_p^*$, $gcd(\tilde{r}, q) = 1$, $k_i \in_R \mathrm{Z}_q^*$, $M \in_R \{0,1\}^*$, $a_{[1,n]}$ is appropriate for $(P_1, \ldots, P_i, \ldots, P_n)$, $a_{[1,n,\{i\}]} \in \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$, $k_{[1,n,\{i\}]} \in_R \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$, $\alpha \in \mathrm{Z}_q^* \backslash \{q-1\}$, $\tilde{k} \in \mathrm{Z}_q^* \backslash \{1\}$,
$\alpha = (\ldots((a_1 + 1)a_2 + 1)\ldots)a_{i-1} \,(\mathrm{mod}q)$,
$\tilde{k} = -(\alpha + 1)(\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M))^{-1} \,(\mathrm{mod}q)$.

Therefore $\mathbf{DLP}_q \equiv_T^p \mathbf{Secret2}[(P_1, \ldots, P_i, \ldots, P_n)]$.

Security of Signing Order. We study the order of signature creation. Namely we assess the difficulty of forgery when all attackers $P_k \in G_{i,\ldots,j}$ excluding P_i try to collude to change the signing order from $G_{i,\ldots,j}$ to $G_{j,\ldots,i}$, where $1 \leq i < j \leq n$, after P_i, who is given a possibly forged partial signature from the previous signer, has signed M. Attackers can use any information except a_i and k_i.

Let y' be a public key of signature structure $(\ldots, P_{i-1}, P_j, \ldots, P_i, \ldots)$ and (r', s') be a forged signature. $\tilde{k}$ and $\tilde{r}$ are freely selected by attackers.

– $\mathbf{Flip}[G_{i,\ldots,j}, G_{j,\ldots,i}] \leq_{1-tt}^p \mathbf{DLP}_q$:
$\mathbf{Flip}[G_{i,\ldots,j}, G_{j,\ldots,i}](y_{i-1}, y_i, y', \tilde{r}_{i-1}, r_i, \tilde{r}, \tilde{s}_{i-1}, s_i, \tilde{k}, a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]}, M, g, p, q) = (\mathbf{DLP}_q(y', g, p, q) + \alpha' r' \cdot \mathrm{hash}(r', M) \ (\mathrm{mod}q),\ r')$,
where $\alpha' \in_R \mathrm{Z}_q^*, r' = g^{\alpha'}, gcd(r', q) = 1$.
– $\mathbf{DLP}_q \leq_{1-tt}^p \mathbf{Flip}[G_{i,\ldots,j}, G_{j,\ldots,i}]$:
$\mathbf{DLP}(g^{a_i}, g, p, q) = \beta^{-1} \cdot \{-\gamma - \alpha' r' \cdot \mathrm{hash}(r', M) + FC(\mathbf{Flip}[G_{i,\ldots,j}, G_{j,\ldots,i}]$
$(g^{\alpha}, (g^{a_i})^{\alpha+1}, (g^{a_i})^{\beta} \cdot g^{\gamma}, g^{\tilde{k}}, (g^{s_i} \cdot (g^{a_i})^{-(\alpha+1)})^{(\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M))^{-1}}, \tilde{r}, \alpha + \tilde{k}\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M),$
$s_i, \tilde{k}, a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]}, M, g, p, q))\} \ (\mathrm{mod}q)$,
where $\tilde{k} \in_R \mathrm{Z}_q^*, s_i \in_R \mathrm{Z}_q, \tilde{r} \in_R \mathrm{Z}_q^*$ with $gcd(\tilde{r}, q) = 1, a_{[1,n,\{i\}]} \in \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$ is appropriate for $\{G_{i,\ldots,j}, G_{j,\ldots,i}\}$, $k_{[1,n,\{i\}]} \in_R \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*, M \in_R \{0,1\}^*$, $\alpha' \in_R \mathrm{Z}_q^*, r' = g^{\alpha'}, gcd(r'q) = 1$,

$$\begin{aligned}
&\alpha = (\cdots((a_1 + 1)a_2 + 1)\cdots)a_{i-1} \ (\mathrm{mod}q),\\
&\beta = \alpha a_{i+1} \cdots a_n + a_{i+1} \cdots a_n + a_{i+1} \cdots a_{j-1}a_{j+1} \cdots a_n + \cdots +\\
&\qquad + a_{j-1}a_{j+1} \cdots a_n + a_{j+1} \cdots a_n \ (\mathrm{mod}q),\ \ gcd(\beta, q) = 1,\\
&\gamma = a_{j+1} \cdots a_n + \cdots + a_n \,(\mathrm{mod}q).
\end{aligned}$$

Therefore $\mathbf{DLP}_q \equiv_{1-tt}^p \mathbf{Flip}[G_{i,\ldots,j}, G_{j,\ldots,i}]$.

7.2 Appropriate Secret Keys

Given $a_i \in Z_q^*$ select $a_{[i+1,n]} \in_R Z_q^* \times \cdots \times Z_q^*$ after selecting $a_{[1,i-1]}$ appropriately for $(P_1, \ldots, P_i)$. The probability that $a_{[1,n]}$ is appropriate for $(P_1, \ldots, P_i, \ldots, P_n)$ can be estimated as follows.

Let $\alpha_{i-1} = (\cdots((a_1+1)a_2+1)\cdots)a_{i-1} \,(\mathrm{mod} q)$. Then a_i should satisfy

$$a_i \neq \alpha_j(\alpha_{i-1}+1)^{-1} \,(\mathrm{mod} q) \quad \text{for } j = 1, \ldots, i-1$$

and

$$a_i \neq (a_u - \sum_{l=i+1}^{m} \prod_{t=l} a_t)(\alpha_{i-1}+1)^{-1}(\prod_{k=i+1}^{m} a_k)^{-1} \,(\mathrm{mod} q)$$

for $u = 1, \ldots, m-1$ and $m = i+1, \ldots, n$. Thus the probability is

$$1 - (i - 1 + \sum_{v=i}^{n-1} v)/2^{|a_i|} = 1 - (n(n-1) - i^2 + 3i - 2)/2^{|a_i|-1}$$

for $i = 2, \ldots, n$. If $n = O(poly(|a_i|))$ this probability can be estimated to be almost 1.

8 Parallel Multisignature Scheme

The public key of a parallel multisignature scheme is $y = \prod_{i=1}^{n} y_i \,(\mathrm{mod} p)$, where $y_i = g^{a_i}$ are the partial public keys. We must have $y_i \in Z_p^* \backslash \{1\}$ and the product of any partial keys should not be 1.

Parallel multisignature protocol

1. P_i selects $k_i \in_R Z_q^*$ and computes $r_i = g^{k_i} (\mathrm{mod} p)$. P_i repeats this process until $gcd(r_i, q) = 1$
2. Each P_i broadcasts r_i to all the other signers. Then each P_i checks if there is a combination of r_i whose product is equal to 1 modulo p. If there is such a combination, step 1 is repeated.
3. $r = \prod_{i=1}^{n} r_i \,(\mathrm{mod} p)$.
4. P_i computes $s_i = a_i + k_i r \cdot \mathrm{hash}(r, M) \,(\mathrm{mod} q)$, and broadcasts s_i to all the other signers.
5. $s = \sum_{i=1}^{n} s_i \,(\mathrm{mod} q)$.

The multisignature of M by $\wedge(P_1, \ldots, P_n)$ is (s, r). It is verified by checking that

$$g^s \equiv g^{\sum_{i=1}^{n} s_i} \equiv \prod_{i=1}^{n} y_i \,(\prod_{i=1}^{n} r_i)^{r \cdot \mathrm{hash}(r,M)} \equiv y r^{r \cdot \mathrm{hash}(r,M)} \,(\mathrm{mod} p).$$

Observe that,

$$g^s \equiv g^{\sum_{i=1}^{n} s_i} \equiv \prod_{i=1}^{n} y_i \, (\prod_{i=1}^{n} r_i)^{r \cdot \mathrm{hash}(r,M)}$$
$$\equiv y r^{r \cdot \mathrm{hash}(r,M)} \pmod{p}.$$

As noted in [DOMU94,DOM98] parallel multisignatures schemes are essentially threshold schemes [DF91].

9 Multisignature for a Mixed Structure

The parallel and serial signature structures can be combined in an arbitrary order. The public keys and signatures are generated and verified by using the methods of the serial and parallel cases in a straightforward way.

The partial public keys of the signers in a mixed structure are computed as follows. Let G_{init} be the group of signers with no incoming edges in the graph representation G. The partial public key of signer P_i in G_{init} is simply $y_i = g^{a_i}$. This is given to every signer in $G_{next,init}$, the group of signers whose edges are connected to the edges of G_{init}. Then for all i such that P_i does not belong to G_{init}, P_i's partial public key is $y_i = (g \prod_{j:\, P_j \in G_{prev,i}} y_j)^{a_i}$ and is given to every signer in $G_{next,i}$, where $G_{prev,i}$ and $G_{next,i}$ denote a group of signers whose edges are connected to and from P_i in the graph representation, respectively. The public key of the group is $y = \prod_{j\,:P_j \in G_{last}} y_j$, where G_{last} denotes a group of signers whose edges are combined into the output in the graph representation.

General multisignature protocol

1. **Computing r:** For all P_i which belong to G_{init}, $r_i = g^{k_i}$, with $k_i \in_R \mathrm{Z}_q^*$. For all P_i that do not belong to G_{init}, $r_i = (\prod_{j:\, P_j \in G_{prev,i}} r_j)^{a_i} g^{k_i}$ where $k_i \in_R \mathrm{Z}_q^*$. The created r_i is given to every signer in $G_{next,i}$. Then $r = \prod_{j:\, P_j \in G_{last}} r_j$.
2. **Computing s:** For all P_i which belong to G_{init}, a partial signature (s_i, r_i, r) of P_i is computed by $s_i = a_i + k_i r \cdot \mathrm{hash}(r, M) \pmod{q}$. Then for all P_i that do not belong to G_{init}, the partial signature (s_i, r_i, r) of P_i is computed by $s_i = ((\sum_{j:\, P_j \in G_{prev,i}} s_j) + 1) a_i + k_i r \cdot \mathrm{hash}(r, M) \pmod{q}$, and given to every signer in $G_{next,i}$. Finally $s = \sum_{j:\, P_j \in G_{last}} s_j$.

   The multisignature of M by the group G is (s, r). It is verified by checking that $g^s \stackrel{?}{\equiv} y r^{r \cdot \mathrm{hash}(r,M)} \pmod{p}$.

To show how this is done more explicitly we compute the multisignature of the group $G_{fig1} = [\wedge(P_1, (P_2, P_3, P_4)), \wedge((P_5, P_6), P_7, (P_8, P_9))]$ in Figure 1. The partial public key of P_i in G_{fig1} is computed from the secret key $a_i \in \mathrm{Z}_q^*$ and other partial keys y_j $(j \neq i)$ as follows: $y_1 = g^{a_1} \pmod{p}$ for the first

lower branch, $y_2 = g^{a_2} (\mathrm{mod} p)$, $y_3 = (y_2 g)^{a_3} (\mathrm{mod} p)$, $y_4 = (y_3 g)^{a_4} (\mathrm{mod} p)$ for the first upper branch, $y_5 = ((y_1 y_4) g)^{a_5} (\mathrm{mod} p)$, $y_6 = (y_5 g)^{a_6} (\mathrm{mod} p)$ for the second lower branch, $y_7 = ((y_1 y_4) g)^{a_7} (\mathrm{mod} p)$ for the second middle branch, $y_8 = ((y_1 y_4) g)^{a_8} (\mathrm{mod} p)$, $y_9 = (y_8 g)^{a_9} (\mathrm{mod} p)$ for the second upper branch. Finally the public key y of G_{fig1} is computed by $y = y_6 y_7 y_9 (\mathrm{mod} p)$.

- **Computing r:** We have $r_4 = r_3^{a_4} g^{k_4} (\mathrm{mod}\ p)$, $r_5 = (r_1 r_4)^{a_5} g^{k_5} (\mathrm{mod} p)$, $r_6 = (r_5)^{a_6} g^{k_6} (\mathrm{mod} p)$, so $r = r_6 r_7 r_9 \, (\mathrm{mod} p)$.
- **Computing s:** We have $s_4 = (s_3 + 1) a_4 + k_4 r \cdot \mathrm{hash}(r, M) \, (\mathrm{mod} q)$, $s_5 = ((s_1 + s_4) + 1) a_5 + k_5 r \cdot \mathrm{hash}(r, M) \, (\mathrm{mod} q)$ and $s_6 = (s_5 + 1) a_6 + k_6 r \cdot \mathrm{hash}(r, M) \, (\mathrm{mod} q)$, so $s = s_6 + s_7 + s_9 \, (\mathrm{mod} q)$.

The multisignature of M by G_{fig1} is (s, r). It is verified by checking that

$$g^s \stackrel{?}{\equiv} y r^{r \cdot \mathrm{hash}(r, M)} \, (\mathrm{mod} p).$$

9.1 Security Considerations

The partial public key of a signer P_i in a directed series-parallel graph can be expressed as $y = g^{A a_i + B}$, for some $A, B \in Z_q$. We study the difficulty of signature forgery for a structure G_f, given a partial signature (r_i, s_i) of P_i for the original structure G_o.

- $\mathbf{Forge}[G_o, G_f] \leq^p_{1-tt} \mathbf{DLP}_q$:
  $\mathbf{Forge}[G_o, G_f](y_{i-1}, y_i, y', \tilde{r}_{i-1}, r_i, \tilde{r}, \tilde{s}_{i-1}, s_i, \tilde{\beta}, a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]}, M, g, p, q)$
  $= (\mathbf{DLP}(y', g, p, q) + \beta' r' \mathrm{hash}(r', M) \, (\mathrm{mod} q), \, r')$, where $\beta' \in_R \mathrm{Z}_q^*$, $r' = g^{\beta'}$, $gcd(g^{\beta'}, q) = 1$.
- $\mathbf{DLP}_q \leq^p_{1-tt} \mathbf{Forge}[G_o, G_f]$:
  $\mathbf{DLP}_q(g^{a_i}, g, p, q) = \alpha_1'^{-1} \{ -\beta' r' \mathrm{hash}(r', M) - \alpha_2' + FC(\mathbf{Forge}[G_o, G_f](g^{\alpha}, (g^{a_i})^{\alpha+1}, (g^{a_i})^{\alpha_1'} \cdot g^{\alpha_2'}, g^{\tilde{\beta}}, (g^{s_i} \cdot (g^{a_i})^{-\alpha-1})^{(\tilde{r} \cdot \mathrm{hash}(\tilde{r}, M))^{-1}}, \tilde{r}, \alpha + \tilde{\beta} \tilde{r} \cdot \mathrm{hash}(\tilde{r}, M), s_i, \tilde{\beta}, a_{[1,n,\{i\}]}, k_{[1,n,\{i\}]}, M, g, p, q)) \} \, (\mathrm{mod} q)$,
  where $a_{[1,n]} \in \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$ is appropriate for $\{G_o, G_f\}$, $G_o, G_f \in \mathcal{G}$, $\tilde{\beta} \in_R \mathrm{Z}_q^*$ with $gcd(g^{\tilde{\beta}}, q) = 1$, $s_i \in_R \mathrm{Z}_q$ with $gcd((g^{s_i} \cdot (g^{a_i})^{-\alpha-1})^{(\tilde{r} \mathrm{hash}(\tilde{r}, M))^{-1}}, q) = 1$, $\tilde{r} \in_R \mathrm{Z}_p^*$ with $gcd(\tilde{r}, q) = 1$, $M \in_R \{0,1\}^*$, $k_{[1,n,\{i\}]} \in_R \mathrm{Z}_q^* \times \cdots \times \mathrm{Z}_q^*$, $\alpha = f_\alpha(a_{[1,n,\{i\}]})$, $\alpha_1' = f_{\alpha_1'}(a_{[1,n,\{i\}]})$, $\alpha_2' = f_{\alpha_2'}(a_{[1,n,\{i\}]})$, with f_α determined by G_o, $f_{\alpha_1'}$, $f_{\alpha_2'}$ determined by G_f, $P_i \in G_o, G_f$,

Therefore $\mathbf{DLP}_q \equiv^p_{1-tt} \mathbf{Forge}[G_o, G_f]$.

This proof does not hold when: (i) $\alpha_1' = 0$, (ii) β' satisfies $gcd(g^{\beta'}, q) = 1$. In both cases we cannot show $\mathbf{DLP}_q \leq^p_{1-tt} \mathbf{Forge}[G_o, G_f]$.

A typical example of the latter case is when the signer structure from the first signer to P_i in G_o is preserved in G_f. For example, it is easy to forge a signature for $G_f = (\cdots, P_i, P_{i+2}, P_{i+1}, \cdots))$ from an original graph $G_o = (\cdots, P_i, P_{i+1}, P_{i+2}, \cdots)$.

9.2 Constructible and Available Groups

We give a very rough estimate of the number of signers for which the complete set of signer groups becomes constructible.

Let T_n be the number of all signer structures for n signers whose directed graph G satisfies conditions (G-1) and (G-2). Then,

$$T_n < 2^n(2^n-1)(2^n-2)\cdots(2^n-(n-1)) < 2^{n^2}.$$

A bound for the number K_n of all partial public keys for G is given by,

$$K_n \leq 2(2^n-1)+1 = 2^{n+1}-1.$$

Therefore, the number of all partial public keys for all directed graphs with n signers is bounded by,

$$T_n \cdot K_n < 2^{n^2}(2^{n+1}-1) = 2^{n^2+n+1} - 2^{n^2}.$$

Using the birthday paradox we can now get an upper bound on the number of signers for which we have constructible groups for any set. We have

$$T_n \cdot K_n < 2^{n^2+n+1} - 2^{n^2} << \sqrt{q}.$$

Thus when $n < \sqrt{\log \sqrt{q}}$ any signer group should have a different public key.

Figure 3 and Figure 4 show directed non-series-parallel graphs whose signer groups are not available. The partial public keys of these structures are the same for any choice of signers' secret keys.

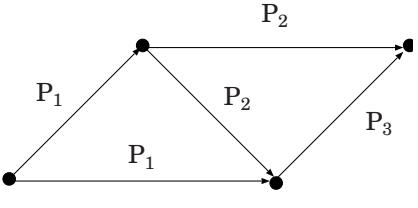

Fig. 3. A non series-parallel graph

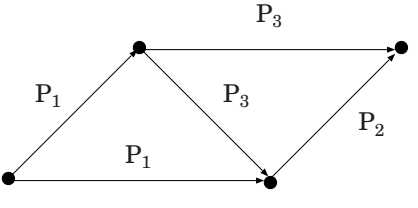

Fig. 4. A non series-parallel graph

Indeed, let $y_{\text{fig.3}}$ and $y_{\text{fig.4}}$ be public keys of these graphs respectively. We can easily check that

$$\begin{aligned} y_{\text{fig.3}} &\equiv g^{((a_1+(a_1+1)a_2)+1)a_3+(a_1+1)a_2} \\ &= g^{((a_1+(a_1+1)a_3)+1)a_2+(a_1+1)a_3} \\ &\equiv y_{\text{fig.4}}\,. \end{aligned}$$

10 An Efficient Structured Multisignature Scheme

We have not fully exploited the structure of the decomposition tree in our earlier approach. We shall now show how this can be done by considering the group structure $G_{fig1} = [\wedge(P_1,(P_2,P_3,P_4)),\wedge((P_5,P_6),P_7,(P_8,P_9))]$ in Figure 1. From the decomposition tree in Figure 2, using our earlier approach for serial/parallel executions, we see that may take as group secret key, the key

$$a = [\,[\,a_1 + ((a_2+1)a_3+1)a_4\,]+1\,]\cdot[\,(a_5+1)a_6 + a_7 + (a_8+1)a_9\,]$$

with corresponding group public key $y = g^a$. Here a secret key is computed for every sub-tree in Figure 2 precisely once: $[a_1 + ((a_2+1)a_3+1)a_4]$ is the secret key for the left sub-tree $[\wedge(P_1,(P_2,P_3,P_4)]$. This makes use of $((a_2+1)a_3+1)a_4$ which is the secret key for (P_2,P_3,P_4). Similarly $[(a_5+1)a_6+a_7+(a_8+1)a_9]$ is the secret key for the right sub-tree $\wedge((P_5,P_6),P_7,(P_8,P_9))$ which uses $(a_5+1)a_6$ as secret key for (P_5,P_6) and $(a_8+1)a_9$ as secret key for (P_8,P_9).

The resulting scheme is far much more efficient than the earlier scheme for which the group key was,

$$\begin{aligned} a' = ((\,[\,a_1 + ((a_2+1)a_3+1)a_4\,]+1)a_5+1)a_6 \;+\\ +\;(\,[\,a_1+((a_2+1)a_3+1)a_4\,]+1)a_7 \;+\\ +\;((\,[\,a_1+((a_2+1)a_3+1)a_4]\;+1)a_8+1)a_9\,. \end{aligned}$$

Since the security for the general case can be reduced to that of the serial and parallel structures, we only need to prove that these are secure. This follows from our earlier reductions.

Let us describe the multisignature protocol for the example of the group structure G_{fig1} with the decomposition tree shown in Figure 2. The partial public key of P_i in G_{fig1} is computed from the secret key $a_i \in Z_q^*$ and other partial keys y_j $(j \neq i)$ as follows: $y_1 = g^{a_1} (\mathrm{mod} p)$ for the first left branch, $y_2 = g^{a_2} (\mathrm{mod}\ p)$, $y_3 = (y_2 g)^{a_3} (\mathrm{mod} p)$, $y_4 = (y_3 g)^{a_4} (\mathrm{mod} p)$, for the second left branch, $y_5 = ((y_1 y_4) g)^{a_5} (\mathrm{mod} p)$, $y_6 = ((y_1 y_4) y_5 g)^{a_6} (\mathrm{mod} p)$ for the first right branch, $y_7 = ((y_1 y_4) g)^{a_7} (\mathrm{mod} p)$, for the second right branch, $y_8 = ((y_1 y_4) g)^{a_8} (\mathrm{mod} p)$, $y_9 = ((y_1 y_4) y_8 g)^{a_9} (\mathrm{mod}\ p)$ for the third right branch. Finally the public key y of G_{fig1} for the new protocol is computed by $y = y_6 y_7 y_9 (\mathrm{mod} p)$.

– **Computing r:** We have

$$\begin{aligned} r_4 &= r_3^{a_4} g^{k_4} \pmod p,\\ r_5 &= (r_1 r_4)^{a_5} g^{k_5} \pmod p,\\ r_6 &= ((r_1 r_4) r_5)^{a_6} g^{k_6} \pmod p. \end{aligned}$$

Similarly,

$$\begin{aligned} r_7 &= (r_1 r_4)^{a_7} g^{k_7} \pmod p,\\ r_8 &= (r_1 r_4)^{a_8} g^{k_8} \pmod p,\\ r_9 &= ((r_1 r_4) r_8)^{a_9} g^{k_9} \pmod p. \end{aligned}$$

Then $r = r_6 r_7 r_9 \,(\mathrm{mod} p)$.

– **Computing s:** We have

$$\begin{aligned} s_4 &= (s_3 + 1)a_4 + k_4 r \cdot \mathrm{hash}(r, M) \pmod{q}, \\ s_5 &= ((s_1 + s_4) + 1)a_5 + k_5 r \cdot \mathrm{hash}(r, M) \pmod{q}, \\ s_6 &= ((s_1 + s_4) + s_5 + 1)a_6 + k_6 r \cdot \mathrm{hash}(r, M) \pmod{q}. \end{aligned}$$

Similarly,

$$\begin{aligned} s_7 &= ((s_1 + s_4) + 1)a_7 + k_7 r \cdot \mathrm{hash}(r, M) \pmod{q}, \\ s_8 &= ((s_1 + s_4) + 1)a_8 + k_8 r \cdot \mathrm{hash}(r, M) \pmod{q}, \\ s_9 &= ((s_1 + s_4) + s_8 + 1)a_9 + k_9 r \cdot \mathrm{hash}(r, M) \pmod{q}. \end{aligned}$$

So $s = s_6 + s_7 + s_9 \,(\mathrm{mod} p)$.

The new multisignature of M by G_{fig1} is (s, r). It is verified by checking that

$$g^s \stackrel{?}{\equiv} y r^{r \cdot \mathrm{hash}(r, M)} \,(\mathrm{mod} p).$$

11 Conclusions

A structured multisignature scheme is a scheme for which the group of signers is structured. We have studied such schemes and proposed structured multisignature schemes based on a modified ElGamal signature scheme. For the security proof we have considered attacks for which all the signers except one honest signer P_i collude. The attackers give a partial signature for a message M to P_i and obtain a valid partial signature for M by P_i. Under this attack model, the security of the proposed scheme is proven by showing reductions of the discrete logarithm problem to the problems of extracting a secret key of a target signer, changing the signing order in a serial multisignature and changing the signing structure in a mixed structure.

Acknowledgements

The authors are grateful to Chandana Gamage and Yuliang Zheng for their kind assistance during the printing trouble of this paper.

References

[BR93] M. Bellare, and P. Rogaway, Random Oracles are Practical: a paradigm for designing efficient protocols, *Proc. of 1st ACM Conference on Computer and Communication Security*, pp. 62–73, 1993. 468

[CGH98] R. Canetti, O. Goldreich, and S. Halevi, The Random Oracle Methodology, Revisited, *Proc. of the 30th Annual ACM Symposium on Theory of Computing, STOC*, pp. 209–218, 1998. 468

[DF91] Y. Desmedt, and Y. Frankel, Shared generation of authenticators and signatures, *Lecture Notes in Computer Science 576, Advances in Cryptology -Crypto '91*, pp. 457-469, 1991. 477

[DOMU94] H. Doi, E. Okamoto, M. Mambo, and T. Uyematsu, Multisignature Scheme with Specified Order, *Proc. of the 1994 Symposium on Cryptography and Information Security*, SCIS94-2A, January 27-29, 1994. 467, 477

[DOM98] H. Doi, E. Okamoto, and M. Mambo,Multisignature Schemes for Various Group Structures, *The 36-th Annual Allerton Conference on Communication, Control, and Computing,* pp. 713-722,1999. 467, 468, 477

[ElG85] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Trans. on Inform. Theory*, Vol. IT-31, No. 4, pp. 469-472, 1985. 467, 472

[FS86] A. Fiat, and A. Shamir, How to Prove Yourself: Practical Solution to Identification and Signature Problems, *Lecture Notes in Computer Science 263, Advances in Cryptology -Eurocrypt '86*, Spring-Verlag, pp. 186-194 1987. 468

[HMP94] P. Horster, M. Michels, and H. Petersen, Meta-ElGamal Signature Schemes, *Proc. of the 2nd ACM Conference on Computer and Communications Security,* pp. 96-107, November 1994. 467

[HMP95] P. Horster, M. Michels, and H. Petersen Meta-multisignature schemes based on the discrete logarithm problem, *Information Security -the Next Decade, Proc. of IFIP/Sec'95,* Chapman & Hall pp. 128-142 1995. 467

[HZ92] T. Hardjono, and Y. Zheng A practical digital multisignature scheme based on discrete logarithms, *Lecture Notes in Computer Science 718, Proc. of Auscrypt'92*, Springer-Verlag, pp. 122-132, 1993. 467

[Len90] T. Lengauer, Combinatorial Algorithms for Integrated Circuit Layout, B. G. Teubner Stuttgart, John Wiley & Sons, 1990. 468

[OO91] K. Ohta, and T. Okamoto, A digital multisignature scheme based on the Fiat-Shamir scheme, *Lecture Notes in Computer Science 739, Advances in Cryptology -Asiacrypt'91*, Springer-Verlag, pp. 139-148, 1993. 467, 468

[OO99] K. Ohta, and T. Okamoto, Multisignature schemes secure against active insider attacks, *IEICE Trans. Fundamentals* Vol. E82-A, No. 1, pp. 21-31, 1999. 467

[Oka88] T. Okamoto, A Digital Multisignature Scheme Using Bijective Public-Key Cryptosystems, *ACM Trans. on Computer Systems*, Vol. 6, No. 8, pp. 432-441, November 1988. 467

[PS96] D. Pointcheval, and J. Stern, Security Proofs for Signature Schemes, *Lecture Notes in Computer Science 1070, Advances in Cryptology -Eurocrypt '96,* Springer-Verlag, pp. 387-398, 1996. 468

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Communcation of the ACM*, Vol. 21, No. 2, pp. 120-126, 1978. 467

[Schn91] C. P. Schnorr, Efficient Signature Generation by Smart Cards, *Journal of Cryptology*, Vol. 4, No. 3, pp. 161-174 1991. 467, 474

[Shim94] A. Shimbo, Multisignature Schemes Based on the ElGamal Scheme, *Proc. of The 1994 Symposium on Cryptography and Information Security*, SCIS94-2C, January 27-29, 1994. 467

[Woll87] H. Woll, Reductions among number theoretic problems, *Information and Computation*, Vol. 72, pp. 167-179, 1987. 470

[YL93] S. Yen, and C. Laih, New Digital Signature Scheme Based on Discrete Logarithm, *Electronics Letters*, Vol. 29, No. 12, pp. 1120-1121, 1993. 472