A Fully Distributed Proactively Secure Threshold-Multisignature Scheme

Johann van der Merwe, Dawoud S. Dawoud, *Member*, *IEEE*, and Stephen McDonald, *Member*, *IEEE*

Abstract—Threshold-multisignature schemes combine the properties of threshold *group-oriented* signature schemes and *multisignature* schemes to yield a signature scheme that allows a threshold (t) or more group members to collaboratively sign an arbitrary message. In contrast to threshold *group* signatures, the individual signers do not remain anonymous, but are *publicly* identifiable from the information contained in the valid threshold-multisignature. The main objective of this paper is to propose such a secure and efficient threshold-multisignature scheme. The paper uniquely defines the fundamental properties of threshold-multisignature schemes and shows that the proposed scheme satisfies these properties and eliminates the latest attacks to which other similar schemes are subject. The efficiency of the proposed scheme is analyzed and shown to be superior to its counterparts. The paper also proposes a discrete logarithm based *distributed-key management infrastructure* (DKMI), which consists of a round optimal, publicly verifiable, distributed-key generation (DKG) protocol and a one round, publicly verifiable, distributed-key redistribution/updating (DKRU) protocol. The round optimal DKRU protocol solves a major problem with existing secret redistribution/updating schemes by giving group members a mechanism to identify malicious or faulty share holders in the first round, thus avoiding multiple protocol executions.

Index Terms—Security and protection, distributed systems, group-oriented cryptography, threshold-multisignature, secret sharing, distributed-key management infrastructure, publicly verifiable distributed-key generation, publicly verifiable distributed-key update, publicly verifiable distributed-key redistribution.

1 Introduction

In distributed systems it is sometimes necessary for users to share the power to use a cryptosystem [1], [2]. The system secret is divided up into shares and securely stored by the entities forming the distributed cryptosystem. The main advantage of a distributed cryptosystem is that the secret is never computed, reconstructed, or stored in a single location, making the secret more difficult to compromise [3].

In many applications, a threshold (t) or more shareholders are required to cooperatively generate a digital signature, in contrast to the conventional single signer. This may also be seen as a distribution of trust since the shareholders must collaborate and contribute equally to produce a valid *multiparty* signature.

Threshold-multisignature schemes [4] combine the properties of threshold *group-oriented* signature schemes [2] and *multisignature* schemes [5]. In the literature, threshold-multisignature schemes are also referred to as threshold signature schemes with *traceability* [6], [7], [8]. The combined properties guarantee the signature verifier that at least t members participated in the generation of the group-oriented signature and that the identities of the signers can be easily established. The majority of the existing threshold-multisignature schemes belong to variants of the single signatory, *generalized ElGamal signatures* [9], [10], extended to a group/multiparty setting.

In [11], Wang defines the properties of threshold *group* signature schemes [12] in order to guarantee the security of the system. To the best of the authors' knowledge, a complete set of definitions does not exist for *threshold-multisignature* schemes. The authors' studies have shown that secure threshold-multisignature schemes must satisfy the following five main properties:

Correctness: All threshold-multisignatures on an arbitrary message m, generated by an honest authorized subset β of group members, forming subgroup P_β, can be verified by any outsider V (with respect to the group). This implies that the group-oriented signature is publicly verifiable.

Threshold property: Only a threshold of t or more *authorized* group members are able to collaboratively generate a valid threshold-multisignature. This property thus incorporates *unforgeability*.

Traceability: Any outsider V can learn the identities of the individual signers belonging to P_β from the threshold-multisignature on m without interaction with any of the group members and/or a group manager. This implies that the signers are publicly traceable with public information. Traceability implies accountability; the individual signers participating in the threshold-multisignature scheme can be held accountable for their contribution to the group-oriented signature.

Coalition-resistance: No colluding subset of group members can generate a valid threshold-multisignature not satisfying the traceability property. Coalition-resistance subsumes *framing-resistance*, i.e., no subset of group members can sign on behalf of any other subset of group members.

• *The authors are with the School of Electrical, Electronic, and Computer Engineering, University of KwaZulu-Natal, King George V Avenue, Room 4-05, Glenwood, Durban, South Africa, 4041.*
E-mail: {vdmerwe, dawoudd, mcdonalds}@ukzn.ac.za.

Manuscript received 16 Dec. 2004; revised 5 Apr. 2006; accepted 18 Apr. 2006; published online 9 Jan. 2007.
Recommended for acceptance by M. Singhal.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-0307-1204.
Digital Object Identifier no. 10.1109/TPDS.2007.1005.

Break-resistance: An adversary in possession or control of the group secret key and/or the individual secret shares of any number of group members cannot generate a valid threshold-multisignature and/or partial/individual signatures; thus, although the underlying *threshold cryptosystem* has been broken, the threshold-multisignature signature scheme should not be breakable.

Threshold-multisignature schemes can be differentiated from threshold *group* signatures [12] by the fact that by definition, in the latter, the individual signers remain anonymous since it is computationally hard to derive the identities from the group signature, with the exception of the group managers. In contrast, by the above-defined *traceability property* of threshold-multisignature schemes, the individual signers are publicly traceable and do not enjoy anonymity. Consequently, the traceability property of threshold-multisignature schemes allows the individual signers to be held accountable in the public domain and renders the *unlinkability* property of threshold *group* signature schemes, as defined in [11], inapplicable.

An important component of any threshold digital signature scheme is the sharing of the group key [2]. If the group key sharing does not involve a trusted third party (TTP), the key sharing is performed by a distributed collaborative protocol. A threshold-multisignature scheme that is suitable for distributed systems (for example, ad hoc networks [13]) may require a *distributed-key generation* (DKG) protocol that effectively eliminates the TTP. The first DKG protocol, for discrete logarithm-based threshold cryptosystems, was introduced in [14] and a number of promising proposals for DKG have been published since [15], [3], [16], [17]. In these schemes, the distributed-key is generated in a distributed fashion as a function of the random input of all protocol participants while preserving the symmetric relationship between them.

In threshold cryptosystems, it is impractical to assume that an adversary cannot compromise more than t shareholders during the entire lifetime of the distributed secret [18]. The secret shares therefore have to be periodically updated using a *distributed-key updating* (DKU) protocol. The DKU scheme allows for only an impractical period T in which an active/mobile adversary has time to compromise a sufficient percentage of the group secret shares [18].

The access structure $\Gamma_P^{(n,t)}$ of the initial share distribution will not necessarily remain constant [19]. Assuming the same shareholders to be present at all times is unrealistic and the availability/security trade-off (set by the total number of group members (n) and threshold (t)) may need to be changed as a function of system vulnerability, networking environment, and current functionality of the cryptosystem. The shares must then be redistributed to a new access structure $\Gamma_{P'}^{(n',t')}$ using a *distributed-key redistribution* (DKR) protocol [19], [20].

The main objective of this paper is to propose a new *threshold-multisignature* scheme without a trusted third party (TTP), based on a round optimal, publicly verifiable DKG protocol. The proposed scheme can be easily adapted to incorporate a TTP; a version of the proposed scheme with the assistance of a TTP will therefore not be presented. The proposed discrete logarithm-based threshold-multisignature scheme is also *proactively* secure, allowing for DKR to a new access structure $\Gamma_{P'}^{(n',t')}$ and periodic DKU to mitigate attacks from an active/mobile adversary. The proposed discrete logarithm-based threshold-multisignature scheme is made proactively secure by periodically updating secret shares and facilitating changes in group membership by allowing an authorized subset β of existing group members to redistribute secret shares to a new access structure $\Gamma_{P'}^{(n',t')}$.

The paper is organized as follows: In Section 2, the new threshold-multisignature scheme is proposed. Section 3 introduces a new distributed-key redistribution/updating (DKRU) scheme. The security and features of the proposed threshold-multisignature scheme are discussed in Section 4. Some conclusions are provided in Section 5.

2 Proposed Threshold-Multisignature Scheme

In this section, a novel threshold-multisignature scheme, without assistance from a trusted key distribution center (KDC), is proposed. The scheme consists of the following six parts (Section 2.1 through Section 2.6):

2.1 System Parameter Setup

The group members P_i, for $i = 1 : n$, agree on and publish the following system parameters:

- p, q two large primes such that $q \mid (p-1)$,
- g generator of the cyclic subgroup of order q in $(Z)_p^*$,
- $H(\cdot)$ collision-free one-way hash function,
- (n, t) threshold parameter t and total number of group members n, and
- T threshold cryptosystem secret update period.

Protocol participants P_i, for $i = 1 : n$, are assumed to have a long-term public/private key pair PK_i/SK_i and an authentic certificate, verifiable with the public key of a common trusted third party. The certificate of P_i binds the public key $PK_i = g^{SK_i}$ to user P_i's identity ID_i. The certificates are distributed to (or can be traced by) all communication entities P_j, for $j = 1 : n$, $j \neq i$.

2.2 Initial Publicly Verifiable Distributed-Key Generation

In this section, a publicly verifiable *distributed-key generation* (DKG) protocol is presented. The round optimal DKG protocol is developed from the scheme of Zhang and Imai [17]. The purpose of the protocol is to realize secure, *initial* secret sharing to an access structure $\Gamma_P^{n,t}$, without a trusted dealer or KDC.

The protocol executes in four steps:

1. For $1 \leq i \leq n$, P_i chooses a random number $d_i \in [1, q-1]$. P_i then shares the random value d_i with all of the other protocol participants, using Shamir's secret sharing scheme [21] as follows:
   a. Sets $a_{i,0} = s_{i,0} = d_i$ and chooses at random $a_{i,k} \in [1, q-1]$, for $1 \leq k \leq t-1$, such that the numbers $a_{i,0}, \ldots, a_{i,t-1}$ define a polynomial $f_i(X) = \sum_{k=0}^{t-1} a_{i,k} X^k$ over $\mathbb{Z}_q$.
   b. For $j = 0, \ldots, t-1$, P_i computes, $A_{i,j} = g^{a_{i,j}} \bmod p$. For $j = 1, \ldots, n$, P_i computes $s_{i,j} = f_i(ID_j) \bmod q$, $y_{i,j} = g^{s_{i,j}} \bmod p$ and an encryption

$E_{PK_j}(s_{i,j})$ of each secret shadow. P_i then binds itself to the public values by generating the digital signature $(r_{P_i} s_{P_i})$ on message $([y_{i,1}, E_{PK_1}(s_{i,1})] \parallel \cdots \parallel [y_{i,n}, E_{PK_n}(s_{i,n})] \parallel A_{i,0} \parallel \cdots \parallel A_{i,(t-1)})$ using the ElGamal type signature variant: $GES = (M.EG\text{II}.3.\sigma(1), r, s, h(m,r), 1)$, developed from the generalized ElGamal scheme presented in [9]. The signature is generated with P_i's long-term private key SK_i. P_i broadcasts the message and the signature to all protocol participants. The encryption $E_{PK_j}(s_{i,j})$ of the secret shadow for protocol participant P_j is performed using an appropriate *publicly verifiable encryption scheme* [15], [22]. P_i keeps the share when $j = i$.

2. Protocol participants $P_1, \ldots, P_n$ each verify the following:

   a. (r_{P_i}, s_{P_i}) is a valid signature on the public values

   $$([y_{i,1}, E_{PK_1}(s_{i,1})] \parallel \cdots \parallel [y_{i,n}, E_{PK_n}(s_{i,n})] \parallel A_{i,0} \parallel \cdots \parallel A_{i,(t-1)})$$

   received from P_i. This serves as a proof of integrity and binds participant P_i to these public values.

   b. $E_{PK_j}(s_{i,j})$ is the correct encryption of $s_{i,j}$ under the public key PK_j. This implies that anybody can verify that $y_{i,j}$ and $s_{i,j}$ satisfy the following relationship: $y_{i,j} = g^{s_{i,j}} \bmod p$ and that protocol participant P_j can correctly retrieve the subshare $s_{i,j}$ (note that only P_j can decrypt $E_{PK_j}(s_{i,j})$ with its corresponding private key SK_j to recover $s_{i,j}$).

   c. P_j knows that the *subshare* distribution from member P_i is correct if (1) holds.

$$\prod_{k=0}^{t-1} A_{i,k}^{(ID_j)^k} = \prod_{k=0}^{t-1} (g^{a_{i,k}})^{(ID_j)^k} = g^{\sum_{k=0}^{t-1}(a_{i,k})(ID_j)^k} = g^{f_i(ID_j)} (mod\ p) = y_{i,j}. \quad (1)$$

3. Participants are disqualified if they do not follow the protocol correctly. The remaining set of participants form the set Q. The protocol aborts if Q contains less than t members. P_j uses the correct shares received from $i \in Q$ to compute the following:

$$x_j = \sum_{i \in Q} (s_{i,j} L_i) \bmod q, \quad (2)$$

where the Lagrangian coefficient is given as:

$$L_i = \prod_{k \in Q, k \neq i} \frac{ID_k}{ID_k - ID_i} \bmod q. \quad (3)$$

The secret key x_j is P_j's *new* share in the distributed group secret key x_Q. The group public key can be computed (verified) by any group member or outsider as:

$$y_Q = \prod_{i \in Q} A_{i,0}^{L_i} = g^{\sum_{i \in Q} f_i(0) L_i} = g^{x_Q} \bmod p. \quad (4)$$

Each P_j, $j \in Q$ calculates its public key as $y_j = g^{x_j} \bmod p$. Using the ElGamal type signature variant GES, P_j binds itself to the public key y_j by generating a signature (r_{P_j}, s_{P_j}) on y_j using its long-term private key SK_j. P_j broadcasts (y_j, r_{P_j}, s_{P_j}) to all protocol participants. P_j also verifies the authenticity of the received public keys after calculating the public key y_i of all P_i, $i \in Q$ as:

$$y_i = g^{x_i} = g^{\sum_{j \in Q} (s_{j,i} L_j)} = \prod_{j \in Q} y_{j,i}^{L_j} \bmod p. \quad (5)$$

4. Each P_j, $j \in Q$ completely erases *all* traces of its initial randomly chosen value d_j and all generated subshares $s_{j,i} = f_j(ID_i) \bmod q$ $(i = 1, \ldots, n)$ and received subshares $s_{i,j}$ $(i \in Q)$.

The secret can be constructed if t or more honest players, from the set Q, collect:

$$x_Q = \sum_{j \in Q} \left(x_j \prod_{k \in Q, k \neq j} \frac{ID_k}{ID_k - ID_j} \right) \bmod q. \quad (6)$$

Taking the Lagrange polynomial of the original random values d_j, for $j \in Q$, also yields the group secret x_Q, which highlights the importance of erasing these values securely.

In Section 3, a publicly verifiable *distributed-key redistribution/update* (DKRU) protocol is proposed that is compatible with the initial DKG protocol given above. The differences and the significance of the similarity are discussed in Section 4.6 and Section 5.

2.3 Individual Signature Generation

Any subset β of t or more members can represent the group and sign an arbitrary message m. Each member P_i, $i \in \beta$, selects a random integer $k_i \in [1, q-1]$ and computes $r_i = g^{k_i} \bmod p$. Each member *verifiably encrypts* k_i with its *own* long-term public key PK_i using a *verifiable encryption scheme* [15], [22] to generate $E_{PK_i}(k_i)$. Using the ElGamal type signature variant GES, P_i generates a signature (r'_{P_i}, s'_{P_i}) on $[r_i, E_{PK_i}(k_i)]$ using its long-term private key SK_i. P_i broadcasts $(r_i, E_{PK_i}(k_i), r'_{P_i}, s'_{P_i})$ to all protocol participants. This implies that each member commits to its public value r_i and provides a proof of knowledge of its corresponding discrete logarithm, k_i.

After all committed r_is are available and members with invalid r_is are excluded from the set β, the value R is calculated as follows:

$$R = \prod_{i \in \beta} r_i \bmod p. \quad (7)$$

Each P_i uses public values ID_j authentically bound to PK_j, for $j \in \beta$, to form the set B.

P_i uses its (secret share)/(private key) set (x_i, SK_i) and random number k_i to compute its individual signature s_i on message m as follows:

$$s_i = [H(m, R, B)][(x_i) \cdot L_{\beta i} + SK_i] + k_i \bmod q, \quad (8)$$

where the Lagrange interpolating coefficient $L_{\beta i}$ is given by:

$$L_{\beta i} = \prod_{k \in \beta, k \neq i} \frac{ID_k}{ID_k - ID_i} \bmod q.$$

The set $(s_i, r_i, L_{\beta i}, B)$ is the individual signature of P_i on message m, which is broadcast to all other group members.

2.4 Individual Signature Verification

On receiving all of the signatures $(s_i, r_i, L_{\beta i}, B)$, P_j, $j \in \beta$, performs the functionality of a clerk and uses the public key set (y_i, PK_i) to authenticate the individual signature of P_i by verifying if $L_{\beta i}$ is correct and whether the following equation holds for $i \in \beta$, $i \neq j$:

$$g^{s_i} = \left(y_i^{L_{\beta i}} PK_i\right)^{[H(m,R,B)]} r_i \bmod p. \tag{9}$$

If (9) fails to hold, the individual signature of P_i on message m is invalid. Participants are disqualified if their individual signatures are found to be invalid. The remaining honest participants form the set α and repeat the *individual signature generation* part from (7), with $\beta = \alpha$. The protocol aborts if α contains less than t members.

2.5 Threshold-Multisignature Generation

After P_j, $j \in \alpha$, has received and verified t or more individual signatures, the second signature parameter S of the threshold-multisignature (R, S) on message m can be computed as:

$$S = \sum_{i \in \alpha} s_i \bmod q. \tag{10}$$

The set of identities B is appended to (R, S) and provides an explicit link between the threshold signature and the subset of individual signers who collaborated to generate the threshold signature (R, S, B) on message m.

2.6 Threshold-Multisignature Verification and Individual Signer Identification

Any outsider can use the group public key y_Q and public keys PK_i bounded to the identities ID_i, for $i \in \alpha$, to verify the validity of the threshold-multisignature (R, S, B) on an arbitrary message m. The signature verifier calculates the subgroup public key PK_α as follows:

$$PK_\alpha = g^{\sum_{i \in \alpha} SK_i} = \prod_{i \in \alpha} PK_i \bmod p. \tag{11}$$

The signature verifier now has all the information to check if the following congruency holds:

$$g^S \equiv (y_Q PK_\alpha)^{[H(m,R,B)]} R \bmod p. \tag{12}$$

If (12) holds, the threshold-multisignature (R, S, B) on message m is valid and the subgroup B of individual signers is positively identified. (See Section 4.1 for *proof of correctness*.)

3 Proposed Publicly Verifiable Distributed-Key Redistribution/Update Protocol

The proposed publicly verifiable *distributed-key redistribution/update* (DKRU) protocol is as follows:

A distributed group secret x_Q is redistributed from $\Gamma_P^{n,t}$ to a *new* access structure $\Gamma_{P'}^{n',t'}$, using the shares of the authorized subset $\beta \in \Gamma_P^{n,t}$. The system parameter setup given in Section 2.1 is applicable.

The initial secret distribution to the access structure $\Gamma_P^{n,t}$ is performed using the publicly verifiable *distributed-key generation* (DKG) scheme given in Section 2.2. Note that the proposed DKRU protocol is compatible with the initial DKG protocol. The minor differences and the significance of the similarity are discussed in Section 4.6 and Section 5. The proposed DKRU protocol can be used as a publicly verifiable *distributed-key updating* (DKU) protocol by simply keeping the access structure constant, $\Gamma_{P'}^{n',t'} = \Gamma_P^{n,t}$. For the sake of proactive security, the threshold cryptosystem shares should be periodically updated within a limited time period T.

The *publicly verifiable property* by definition means that any outsider can obtain all of the required information from the broadcast channel to validate the operation of both the *initial* DKG and DKRU protocols.

Assume that a subset α of new members join or existing members leave the threshold cryptosystem.

The protocol executes in four steps:

1. P_i, $i \in \beta$ share their secret share x_i of group secret key x_Q with $P_j \in \Gamma_{P'}^{n',t'}$, using Shamir's secret sharing scheme [21]:
   For $i \in \beta$:
   a. P_i sets $a'_{i,0} = s'_{i,0} = x_i$ and chooses at random $a'_{i,k} \in [1, q-1]$, for $1 \leq k \leq t'-1$, such that the numbers $a'_{i,0}, \ldots, a'_{i,t'-1}$ define a polynomial $f'_i(X) = \sum_{k=0}^{t'-1} a'_{i,k} X^k$ over $\mathbb{Z}_Q$.
   b. For $j = 0, \ldots, t'-1$, P_i computes $A'_{i,j} = g^{a'_{i,j}} \bmod p$. For $j = 1, \ldots, n'$, P_i computes $s'_{i,j} = f'_i(ID_j) \bmod q$, $y'_{i,j} = g^{s'_{i,j}} \bmod p$ and an encryption $E_{PK_j}(s'_{i,j})$ of each secret shadow. P_i then binds itself to the public values by generating a digital signature (r'_{P_i}, s'_{P_i}) on message $([y'_{i,1}, E_{PK_1}(s'_{i,1})] \| \cdots \| [y'_{i,n'}, E_{PK_{n'}}(s'_{i,n'})] \| A'_{i,0} \| \cdots \| A'_{i,(t'-1)} \| y_\beta)$ using the ElGamal type signature variant: $GES = (M.EG\text{II}.3.\sigma(1), r, s, h(m, r), 1)$, developed from the generalized ElGamal scheme presented in [9]. The signature is generated with P_i's long-term private key SK_i. P_i broadcasts the message and signature to all protocol participants. Note that P_j, $j \in \beta$, knows the authentic public values y_i of $P_i \in \Gamma_P^{n,t}$, from a previous DKG or DKRU operation (see (5) and (16)). The set y_β is formed by y_i, $i \in \beta$, and only included in the broadcast message if *new* members join the group. The encryption $E_{PK_j}(s'_{i,j})$ of the secret shadow for protocol participant P_j is performed using an appropriate *publicly verifiable encryption scheme* [15], [22]. P_i keeps the share when $j = i$.
2. Protocol participants $P_1, \ldots, P_{n'}$ each verify the following:
   a. (r'_{P_i}, s'_{P_i}) is a valid signature on the public values $([y'_{i,1}, E_{PK_1}(s'_{i,1}] \| \cdots \| [y'_{i,n'}, E_{PK_{n'}}(s'_{i,n'})] \| A'_{i,0} \| \cdots \| A'_{i,(t'-1)} \| y_\beta)$.

b.

$$A'_{i,0} = y_i \ \forall \ i \in \beta. \tag{13}$$

If P_j, $j \in \alpha$ are *new* members joining the group, then they must only use y_is found to be consistant in the broadcast messages received from t or more members of subset β. If the conditions do not hold for more than t members from the authorized subset β, abort the protocol; otherwise, evaluate the following equation:

$$\prod_{k=0}^{t'-1} A_{i,k}^{\prime (ID_j)^k} = \prod_{k=0}^{t'-1} (g^{a'_{i,k}})^{(ID_j)^k} = g^{\sum_{k=0}^{t'-1} (a'_{i,k})(ID_j)^k} = g^{f'_i(ID_j)} (mod\ p) = y'_{i,j}. \tag{14}$$

P_j knows that the *subshare* distribution received from P_i is correct if (14) holds.

3. Participants are disqualified if they do not follow the protocol correctly. The remaining set of participants forms the set Q'. The protocol aborts if Q' contains less than t members of the authorized subset β. P_j uses the correct shares received from $i \in Q'$ to compute the following: $x'_j = \sum_{i \in Q'} s'_{i,j} L'_i$, where the Lagrangian coefficient is given as:

$$L'_i = \prod_{k \in Q', k \neq i} \frac{ID_k}{ID_k - ID_i} \ mod\ q. \tag{15}$$

The secret key x'_j is P_j's *new* share in the distributed group secret key $x_{Q'}$.[1] If the conditions presented in Step-2 hold, then P'_j knows that its *new* secret key x'_j is a valid share of x'_Q. Each P_j, $j \in Q'$ calculates its public key as $y'_j = g^{x'_j} \ mod\ p$. Using the ElGamal type signature variant GES and its long-term private key SK_j, P_j generates a signature (r'_{P_j}, s'_{P_j}) on y'_j, binding itself to the public value. P_j broadcasts $(y'_j, r'_{P_j}, s'_{P_j})$ to all network participants. P_j also calculates the public key of all P_i, $i \in Q'$, as:

$$y'_i = g^{x'_i} = g^{\sum_{j \in Q'} (s'_{j,i} L_j)} = \prod_{j \in Q'} y'^{L_j}_{j,i} \ mod\ p. \tag{16}$$

4. Each P_j, $j \in Q'$ completely erases *all* traces of its old share x_j and all generated subshares $s'_{j,i} = f'_j(ID_i) \ mod\ q$ $(i = 1, \ldots, n')$ and received subshares $s'_{i,j}$ $(i \in Q')$.

The secret can be constructed if t' or more honest players, from the set Q', collect:

$$x_{Q'} = \sum_{j \in Q'} \left(x'_j \prod_{k \in Q' k \neq j} \frac{ID_k}{ID_k - ID_j} \right) mod\ q. \tag{17}$$

1. The group secret remains constant, irrespective of distributed-key redistribution or distributed-key updates, $\therefore y_{Q'} = g^{x_{Q'}} = y_Q = g^{x_Q}$. Group members P_j, for $j \in Q'$ belonging to the new access structure $\Gamma^{n',t'}_{P'}$ thus share the same group secret and public key as the old access structure $\Gamma^{n,t}_{P}$.

Any network participant can calculate (verify) the group public key when needed, using the available public information, as follows:

$$y_{Q'} = \prod_{i \in Q'} A'^{L'_i}_{i,0} = g^{\sum_{i \in Q'} f'_i(0) L'_i} = g^{x_{Q'}} \ mod\ p. \tag{18}$$

4 Discussion on the Security and Features of the Proposed Threshold-Multisignature Scheme

The proposed threshold-multisignature scheme is based on a multiparty extension of the ElGamal type signature variant: $GES = (M.EG\text{II}.3.\sigma(1), r, s, h(m, r), 1)$ [9]. The proposed threshold-multisignature scheme can equally use any other secure and efficient signature variant of the ElGamal type signature scheme. References [9], [10] help in selecting such a variant. The main reason for using the defined GES is to minimize the computational cost of generating and verifying the individual signatures and group-oriented signature in a multiparty setting without compromising security.

The security analysis considers a straightforward general *adversary model*. An adversary is a malicious party that uses every means available to break the proposed threshold-multisignature scheme. Any *active* adversary can eavesdrop on all of the communication between group members, modify the content of messages, and inject them back into the channel. When an honest party is compromised, all of its public and private information is exposed to the adversary.

To argue the security and validity of the proposed threshold-multisignature scheme, it is enough to show that the scheme fulfills all of the fundamental properties of generic threshold-multisignature schemes given in Section 1 and resists attacks to which other similar schemes are subject.

4.1 Correctness and Threshold Property

Any t or more honest group members of the authorized subset α can collaborate and use the threshold-multisignature scheme to produce a valid threshold-multisignature (R, S, B) on an arbitrary message m. In the context of the statement, *honest* implies that the group members follow the protocol as specified in Section 2.

Proof.

$$\begin{aligned}
&s_i = [H(m, R, B)][(x_i) \cdot L_{\alpha i} + SK_i] + k_i \ mod\ q \ (i \in \alpha),\\
&S = \sum_{i \in \alpha} (s_i) = [H(m, R, B)] \sum_{i \in \alpha} [(x_i) \cdot L_{\alpha i}] + \\
&\quad [H(m, R, B)] \sum_{i \in \alpha} (SK_i) + \sum_{i \in \alpha} (k_i) \ mod\ q,\\
&g^S = g^{[H(m,R,B)][\sum_{i \in \alpha} [(x_i) \cdot L_{\alpha i}]]} g^{[H(m,R,B)] \sum_{i \in \alpha} (SK_i)} g^{\sum_{i \in \alpha} (k_i)} \ mod\ p\\
&\quad = g^{[H(m,R,B)] x_Q} \left[\prod_{i \in \alpha} (PK_i) \right]^{[H(m,R,B)]} \prod_{i \in \alpha} (r_i) \ mod\ p\\
&\quad = (y_Q PK_\alpha)^{[H(m,R,B)]} R \ mod\ p.
\end{aligned}$$

□

4.2 Traceability of Signers

The reason for the intractability of [6] and [8], as presented by [23] and [24], [25], respectively, is due to the non-existence of a relationship between the threshold signature (R, S) and the Lagrange polynomial $h(y)$ used by the signature verifier to identify the individual signers. The polynomial $h(y)$ is constructed by the designated combiner with t pairs of public values belonging to members of the subgroup β that submitted valid partial signatures. The Lagrange interpolation polynomial, $h(y)$ is given as [6]:

$$h(y) = \sum_{I=1}^{t} x_i \prod_{j=1, j\neq i}^{t} \frac{y - y_j}{y_i - y_j} = b_{t-1}y^{t-1} + \ldots + b_1 y + b_0. \quad (19)$$

It is noted here that it is possible to create such a relationship between $h(y)$ and (R, S) and make the schemes proposed in [6], [8] traceable by including $h(y)$ within the *individual signatures*. In the view of the authors, the schemes presented in [6], [8], with a strong binding between $h(y)$ and (R, S), still fail to provide an optimum solution to ensure traceability. Using the polynomial function $h(y)$ to capture the signers' identities results in additional computational overhead for the threshold signature verifiers ($n(t-1)$ multiplications and $n(t-2)$ exponentiations) and fails to reduce the threshold group-oriented signature size. The polynomial $h(y)$ of degree $t-1$, constructed from t data points, requires at least t coefficients $(b_0, b_1, \ldots, b_{t-1})$ to be uniquely defined. Rather than including the coefficients in the individual signatures and appending the coefficients to the threshold signature, the proposed threshold-multisignature scheme uses the subset of identities B instead. This saves the signature verifier the computational overhead of evaluating $h(y)$ to learn the identities of the individual signers. The computational cost of generating $h(y)$ is considered in Section 4.7.4.

The traceability property of the proposed threshold-multisignature scheme is verified in Section 4.3 by showing that the signature verifier can only validate the threshold signature if an authentic subset B of individual signers collaborated to generate the signature. Since the subset B of identities is used to validate the threshold signature (12) the individual signers are publicly traceable and explicitly bound to the threshold signature.

If a verifier were to know and use the certified public keys of the individual signers, why not simply use a list of individual (RSA) signatures for the group signature?

The threshold property must be guaranteed by cryptographic means during the generation of the threshold group-oriented signature to effectively share the power of the cryptosystem between the group members [1], [2]. The actual value of the threshold t may be (dynamically) determined by an authorized subset of group members [19] and thus forms part of the group policy. Expecting that all (future) threshold group-oriented signature verifiers will consistently enforce a (dynamic) group policy makes the system vulnerable to attack.

4.3 Coalition-Resistance and Break-Resistance

The coalition and break-resistance properties can be verified by showing that the proposed threshold-multisignature scheme is break-resistant:

Assume a malicious subgroup β of t or more members conspires to obtain the group secret x_Q and wants to frame valid subgroup α for a signature on an arbitrary message m. This implies β wants to generate a valid untraceable group-oriented signature. Any malicious party P_j chooses a random number $k \in [1, q-1]$ and generates $R = g^k$. P_j uses the publicly known values (ID_i), for $i \in \alpha'$, to form the subgroup A. (The public value PK_α can easily be calculated using (11).) The malicious member can attempt to forge the threshold-multisignature by computing $S = [H(m, R, A)](x_Q + SK_\alpha) + k$, which is impractical since P_j cannot compute SK_α from PK_α based on the intractability of the discrete logarithm problem. An alternative method is to solve for S to satisfy the verification equation $g^S = (y_Q PK_\alpha)^{[H(m,R,A)]} R \bmod p$, which is again impractical based on the intractability of the discrete logarithm problem. An attacker's last resort is to randomly select a value F and signature parameter S and then attempt to determine a value R' that satisfies $g^S = (y_Q PK_\alpha)^F R \bmod p$ and $F = [H(m, R', A)]$ simultaneously. This is known to be impractical based on the properties of a secure collision-free one-way hash function $H(\cdot)$.

This shows that using the proposed threshold-multisignature scheme to produce a valid untraceable signature or signing on behalf of any other subset of group members is not possible, even if the threshold cryptosystem is broken, i.e., the group secret x_Q is known or controlled by an adversary. It also confirms that the individual signers that collaborate to generate a threshold-multisignature are always traceable. With little modification to the above argument, it can be shown that forging *individual signatures* is also impractical, even if a coalition of t or more group members conspire to obtain an individual's partial share x_i of the group key x_Q.[2] It can thus be concluded that the proposed threshold-multisignature is *break-resistant*.

4.4 Attacks on Threshold-Multisignature Schemes

4.4.1 Collusion Attack

A *collusion attack* enables dishonest group members (and/or a designated clerk or combiner node: see Section 4.5) that work together to control the group key.

Wang et al. [25] report a collusion attack on the threshold-multisignature scheme (without a trusted third party) presented by Li et al. in [8]. Wang et al. [25] propose some countermeasures against the attack. The first solution requires each member to publish its public value y_i simultaneously to mitigate the *rushing attack* on the group secret. This solution is impractical since there is no method in existing networking protocols of accommodating n simultaneous broadcasts nor can devices receive n simultaneous messages without advanced hardware. Any network participant will therefore always be able to make a legitimate excuse for a late arriving public value. The second proposed solution requires members to commit to their public values and then open their commitments to reveal the public values. This solution, however, makes the protocol interactive and thus vulnerable to an *adaptive* adversary [16], [17]. The third

2. A formal security proof for the defined ElGamal signature variant GES in the Random Oracle and Generic Model (ROM + GM) can be found in [26]. The security proof of the underlying individual signature scheme supports the break-resistance property.

solution requires members to provide a noninteractive proof of knowledge of the discrete logarithm for the public value y_i to the base of the primitive element g. This solution is well known and has been used by both Fouque and Stern [16] and Zhang and Imai [17] to eliminate the need for a *complaint phase*. As pointed out by Zhang and Imai, this approach was first introduced in [15] by Stadler to realize a *publicly verifiable* secret sharing scheme. The *complaint phase* also makes distributed-key generation (DKG) schemes such as [14], [3], [27] impractical considering the complexity of current multiparty computations [16], [17].

The proposed threshold multisignature scheme prevents adversaries from controlling the group key by using the publicly verifiable, one round DKG protocol, given in Section 2.2, and, thus, inherently eliminates attacks from an adaptive adversary during the construction of the threshold cryptosystem. The proposed scheme uses *publicly verifiable encryption* [15], [22] that allows protocol participants to provide a zero knowledge proof of their public values' discrete logarithm. Publicly verifiable encryption, however, does not offer a mechanism to bind participants to their public values. Committing participants to their public values and ensuring the public values' integrity are both essential if members are to be disqualified if their public values fail to satisfy the verification equations (for example, (1) or (9)). The existing round optimal DKG protocols [16], [17] and threshold-multisignature schemes [4], [6], [7], [8] fail to guarantee a strong binding between participants and their public values. The weak binding allows an adversary to manipulate public values that will result in the disqualification of honest participants. The proposed threshold-multisignature scheme and DKG protocol use the ElGamal type signature variant GES to generate secure and efficient digital signatures, which explicitly binds participants to their public values and simultaneously provides a mechanism to ensure the values' integrity.

4.4.2 Universal Forgery Attack

In [23], Tseng and Jan present a universal forgery attack on the threshold signature schemes with traceability by Wang et al. [6]. The attack allows any group member or outsider to generate a valid forged threshold signature (R', S') on an arbitrary message m' from an existing valid threshold signature (R, S) on message m. Tseng and Jan also show in [23] that Wang et al.'s schemes [6] are completely insecure since the trapdoor information $\alpha^{f(0)}$, which is, in fact, the group secret $(y = g^{\alpha^{f(0)}})$, can be calculated as $\alpha^{f(0)} = SR^{-1}H(m)^{-t}$ from any valid threshold signature (R, S) in message m.

The scheme by Wang et al. [6] is an example of an insecure ElGamal type signature variant extended to a multiparty setting. The universal forgery attack in [6] is a direct consequence of the insecurity of the *modified* ElGamal signature scheme on which the group signature is based. Caution should be taken in the modification of the ElGamal signature or when choosing an appropriate variant. The modification of the basic ElGamal signature scheme for the sake of efficiency can easily introduce an insecurity. The threshold-multisignature scheme proposed in Section 2 takes this fact into consideration. It is developed from a secure and optimally efficient variant that was selected using the guidelines given in [9], [10].

4.4.3 Rushing Attack

In the context of group-oriented signature schemes, a rushing attack is defined as a manipulation of the signature parameters or keying material by an adversary based on the public values received from all the other participants. The adversary thus waits until all other participants have played before defining its own value [16]. It is noted that the collusion attack by Wang et al. [25] and universal forgery attack by Wu and Hsu [24], both rely directly on a rushing attack. Assuming the threshold signature scheme is based on a secure and efficient variant [9], [10], it should be clear that the majority of attacks on group-oriented signature schemes can thus only come from a vulnerability introduced by the extension of the ElGamal type signature variant to accommodate multiple signers. Since group members must all make a valid contribution to the threshold signature parameters and group secret, it is evident that manipulation of these values can occur if an adversary can delay playing its own contribution. The adversary can collect the contributed values of all other participants and then compute its own contribution to force the signature parameters or group secret to a known value when calculated as a function of all contributed values. Fouque and Stern [16] and Zhang and Imai [17] construct their distributed-key generation schemes on a *synchronous* network to prevent the adversary from delaying its response, thereby eliminating rushing attacks. Fouque and Stern propose a solution to mitigate rushing attacks without using a *synchronous* network. Fouque and Stern define an Incorruptible Third Party (ITP) to always play at the end, therefore eliminating the requirement for synchrony and preventing a malicious player from manipulating the signature parameters. The ITP unfortunately becomes a single point of vulnerability.

It is noted here that synchrony is not always required in a discrete logarithm-based threshold group-oriented signature scheme based on a variant of the generalized ElGamal type signature. Forcing players to provide a zero knowledge proof of the discrete logarithm of all public values is sufficient to completely eliminate the rushing attack in an *asynchronous* network. Take, for example, the signature parameter R, which is generally calculated as $R = \prod_{i\in\beta} r_i \bmod p$, where $r_i = g^{k_i}$ is the public value of each player P_i with corresponding randomly chosen secret k_i. The malicious player P_j waits for each P_i to play its r_i. P_j chooses a random value k and sets $R = g^k$. After receiving all r_is for $i \in \beta, i \neq j$, P_j factors out its public value as $r_j = R\prod_{i\in\beta, i\neq j} r_i^{-1} \bmod p$, which will force the other players to compute R at a later stage (of which the discrete logarithm is known by P_j). Note that P_j does not know the discrete logarithm of its public value r_j nor can it be calculated based on the intractability of discrete logarithms in finite fields. This will, in particular, be true if the modular arithmetic is done in an appropriate multiplicative subgroup $((Z)_p^*)$ or a cyclic subgroup of order q.

The attack presented by Michels and Horster [28] on the threshold-multisignature proposed by Li et al. [4] is a good example of the rushing attack principle. As in the above example, the malicious player P_j in this attack does not know the discrete logarithm of its own share r_j; thus, if the threshold group-oriented signature protocol requires each

player to provide a zero knowledge proof of all public values' discrete logarithms, the benefit the adversary gains from a rushing attack in an *asynchronous* network is nullified. It will also prevent a malicious participant from disrupting the protocol since the participant can simply be disqualified if it does not play a fair game.

The proposed threshold multisignature scheme in this paper requires players to provide a zero knowledge proof of the discrete logarithm of their public values by generating a *publicly verifiable encryption* [15], [22] on the discrete logarithm of the public values. With this mechanism, the proposed scheme eliminates rushing attacks without any requirement for synchrony.

4.4.4 Conspiracy Attack

It is noted here that the threshold signature scheme proposed by Li et al. [8] is also vulnerable to a conspiracy attack by t or more malicious members. Any t or more members forming the subgroup β' can use their subshares $f(ID_i)_j$, $(i, j \in \beta')$ to construct the group secret key $f(0)$ as $f(0) = \sum_{j=1}^{t}(f(ID_i)_j \prod_{k=1,k\neq j}^{t} \frac{ID_j}{ID_j - ID_i})$. Any malicious member generates $R = g^{k_i}$ and solves the following congruence for integer S: $SR \equiv (R + h(m))k_i + f(0) \ mod \ q$. The set (R, S) will be a universally forged signature on the arbitrary message m, verifiable with the group-oriented signature verification equation, $g^{SR} = R^{(R+h(m))}y \ mod \ p$.

As shown in Section 4.3, conspiracy attacks in the proposed threshold-multisignature schemes are avoided by each member contributing an individual secret (long-term private key SK_i) to the group-oriented signature known to be *explicitly* associated with the member. A mechanism is provided for verifying the members' secret contributions from the threshold-multisignature without revealing any additional information of the secrets, except the information that is publicly know to be associated with the secrets, i.e., the members' public keys PK_i, for $i \in \alpha$. Li et al. [4] defend against conspiracy attacks in a similar approach by adding a random number to the secret key, which effectively conceals the member's secret share of the group key. Note that this is, however, only accurate for Li et al.'s scheme *with* a trusted authority. Since the random numbers are not *explicitly* linked to the member's identity, the scheme presented in [4] does not have a strong *traceability property* [6].

Conspiracy attacks in threshold-multisignature schemes can also be avoided if the secret shares of members are generated in such a way that construction of the threshold cryptosystem's secret polynomial or derivation of the group secret from the members' secret shares is computationally infeasible. Wang et al. [6] propose such a scheme which prevents conspiracy attacks without attaching a random secret to secret shares. It is, however, noted that, since a member's public value in [6] is not explicitly linked to a secret known to be associated with a group member, the scheme by Wang et al. also lacks a strong traceability property. The same comment can be made on the scheme proposed by Lee and Chang in [7].

4.5 Symmetric Relationships—Eliminating the Combiner

A centralized combiner node is a specialized server which can be seen as a single point of vulnerability and should be replicated in distributed networks (for example, ad hoc networks) to ensure a correct combination [13]. The scheme proposed in this paper eliminates the need for a designated combiner/clerk as the construction of the threshold-multisignature is done by the shareholders themselves. This eliminates attacks relying on a corrupt clerk and mitigates the problem of ensuring the availability of a combiner node in distributed networks [13]. The scheme also preserves the symmetric relationship between the shareholders by placing on each node the same computational, memory, and communication overhead.

4.6 Proactive Security and Secret Redistribution

The proposed threshold-multisignature scheme defends against mobile/active adversaries by proactively updating the group key shares every period T. To allow for *dynamic* group membership and modification of the availability/security trade-off, the shares can be redistributed to a new access structure $\Gamma_{P'}^{n',t'}$. The proposed publicly verifiable *distributed-key redistribution/update* (DGRU) protocol proposed in Section 3 was designed to support dynamic group membership and allow for share updates by merely keeping the access structure constant, i.e., $\Gamma_{P'}^{n',t'} = \Gamma_{P}^{n,t}$. Comparing the DGRU protocol to the *initial* DKG protocol presented in Section 2.2, the reader will note that these protocols are almost exactly equivalent. By deliberately keeping the phrasing and notation as far as possible the same, the following minor differences are easily identifiable:

- For the *initial* DKG, each user must choose a random number d_i, while the DKRU protocol uses the user's share x_i of the group secret x_Q.
- Members of the authorized subset β append the public values y_i of $P_i \in \beta$ to $([y'_{i,1}, E_{PK_1}(s'_{i,1})] \parallel \cdots \parallel [y'_{i,n'}, E_{PK_{n'}}(s'_{i,n'})] \parallel A'_{i,0} \parallel \cdots \parallel A'_{i,(t'-1)})$ if *new* members join the group.
- In the DKRU protocol, the participants must also check that the condition presented in (13) holds before verifying whether the *subshare* distribution is correct by using (14). New members should only use public values that have been found to be consistent in t or more broadcast messages. If *both* (13) and (14) are satisfied for t or more members of β, the users are guaranteed that their *new* share in the *original* group key x_Q is valid.

The DKG protocol due to Zhang and Imai [17] was modified to complement the proposed DKRU protocol given in Section 3 and to allow for practical key redistribution/updating. Besides the differences given above, the *initial* DKG protocol, DKU protocol, and DGR protocol are conveniently performed by a single, publicly verifiable, round optimal protocol. A protocol suite capable of servicing *all* aspects of secret sharing is defined here as a ***distributed-key*** *management infrastructure* (DKMI).

It is noted that the proposed DKMI is independent of the proposed *threshold-multisignature* scheme and can be used in other applications to construct and maintain a threshold cryptosystem.

The proposed DKMI presented in Section 2.2 and Section 3 solves a major problem with existing secret *redistribution/update* protocols. This problem is identified by Wong et al. [20]: By definition, a threshold cryptosystem allows t or more members holding a secret share to reconstruct the group secret. In a proactively secure threshold cryptosystem, not more than $t-1$ members can be compromised or become faulty within an update time period T; otherwise, the system is broken. Referring to the proposed DKRU protocol presented in Section 3 and the secret redistribution scheme by Wong et al. [20], the authorized subset β may thus contain, in the worst-case scenario, $t-1$ malicious members out of the total n group members. The $t-1$ malicious members in β can thus distribute subshares $\overline{s_{i,j}}$ of an incorrect share $\overline{x_i}$ to new members joining the system. Although new members can detect the discrepancy in the broadcast values, they cannot identify which members are malicious. It is noted here that this is not only applicable to the new members in the scheme proposed by Wong et al., but also *existing* members cannot identify the malicious members in β and, therefore, malicious members cannot be disqualified. Consequently, the redistribution protocol proposed by Wong et al. [20] must be repeated each time with a different authorized subset β', with $|\beta'| = t$, until all values are consistent and all the verification conditions hold. Wong et al. give a worst-case repetition of the protocol bounded by $\binom{n}{t} - \binom{n-t+1}{t} = \sum_{i=1}^{t-1} \binom{t-1}{i}\binom{n-t+1}{t-i}$, which makes the scheme impractical.

This paper presents a solution to the above problem with existing secret *redistribution/update* protocols as follows:

In the proposed DKMI, the protocol participant P_j in the *initial* DKG protocol uses (5) to calculate the authentic public values y_i corresponding to the secret share x_i of each P_i, for $i \in Q$. In the DKRU protocol, members of the authorized subset β construct a polynomial with their secret share set to $a'_{i,0} = s'_{i,0} = x_i$. In order to ensure that (14) holds and to avoid identification, the malicious members are forced to compute and broadcast $A'_{i,0} = g^{a'_{i,0}}$ with their computed subshare witnesses $y_{i,j}$. Assume that a subset α of new members are joining the threshold cryptosystem. P_i, $i \in \beta$, broadcast to all members a message that includes their calculated public keys y_i, for all other P_i, $i \in \beta$. All existing members forming part of the new access structure $\Gamma_{P'}^{(n',t')}$ can detect and positively identify a malicious member that distributes subshares $\overline{s_{i,j}}$ of an incorrect share $\overline{x_i}$, by checking if (13) holds. The only feasible option is to require the authorized subset β to include all the existing members that will form part of the new access structure $\Gamma_{P'}^{n',t'}$. By the fundamental definition of a threshold cryptosystem, this will always ensure at least t members of β with consistent public values. This requirement places no limitation on the practicality of the proposed distributed-key management infrastructure (DKMI)[3] and keeps the scheme round optimal.

3. It is obvious that existing members that do not participate in the distributed-key redistribution *or* distributed-key update procedure will be excluded from the threshold cryptosystem, therefore, old members wanting to form part of P' are always available to help with distributing subshares to new members.

The new members will detect the discrepancies in the public values broadcast by the members in β. Since the new members will receive consistent public values from at least t participants, the participants with inconsistent values can therefore be considered malicious or faulty and are therefore positively identified.[4] Before joining members have obtained a *valid* share of the group secret, they do not have any authority and, therefore, can only *implicitly* aid in the disqualification of malicious members by only using the subshares of the t or more members with consistent public values to compute their own *new* share in the group key.

4.7 Efficiency Analysis

The efficiency of threshold-multisignatures may be based on the following five criteria (Section 4.7.1 through Section 4.7.5):

4.7.1 Group Public Key Length

The public key length in similar schemes will be briefly considered. In order to make a feasible comparison, only schemes that attempt to eliminate conspiracy attacks are evaluated:

Li et al. [4]: The group public key of [4] not only consists of y, but also implicitly includes the following public values $\{x_i, z_{ij}\}$. In the Li et al. scheme, protocol participants attach a secret random number to their group secret shares in an attempt to avoid a *conspiracy attack*. Consequently, the public key includes the public values $\{x_i, z_{ij}\}$ since these values are required by the signature verifier to validate a threshold signature. It can thus be concluded that the public key is dependent on the number of group members n.

Wang et al. [6]: The scheme proposed in [6] is an improvement on the scheme presented in [4] since conspiracy attacks are prevented without attaching a random number to secret shares. The group public key is thus independent of the group size n. The protocol, however, requires the participants to order themselves into a ring topology, which can be limiting in some network scenarios.

Lee and Chang [7]: The threshold-multisignature scheme in [7] also avoids conspiracy attacks without attaching a random secret to shares. The group public key is dependent on the number of group members n as the signature verifier needs the individual public values y_i of all group members and two additional parameters c_Q and c_w to compute the subgroup public key, Y, that is required to verifying the threshold signature. Difficulty will be experienced with this scheme when trying to eliminate the need for a trusted authority to distribute the initial group key shares.

A robust authentication mechanism is essential for securing a distributed system against *active* adversaries and central to ensuring the traceability of individual signers. As shown in Section 4.3, the proposed threshold-multisignature scheme uses the long-term private keys of members SK_i, provided by a public key infrastructure, to

4. The proposed scheme does have another mechanism for *all* network participants to identify malicious members. Since both the *initial* DKG protocol (Section 2.2) and DKRU protocol (Section 3) are both *publicly verifiable*, all network participants have exactly the same information as existing members of the threshold cryptosystem, except for a valid share in the group secret. A joining member who has observed a periodic DKRU procedure can calculate the authentic public values of each group member using (16) and, therefore, identify malicious members during the next DKRU procedure.

avoid conspiracy attacks even if colluding members derive or control the group secret x_Q. As a result of members including their private keys in their *individual signatures*, the public key of the scheme consists of y_Q and the public key PK_α of the subgroup α that collaborated to generate the threshold signature. The public key PK_α of the subgroup α is a function of the long-term public keys PK_i of the group members P_i, for $i \in \alpha$. Although the group public key may be perceived to be dependent on the group size n, the scheme does not introduce any additional storage requirements since the public keys PK_i, for $i \in \alpha$ used to calculate PK_α is publicly known (traceable) and primarily required for authentication purposes.

4.7.2 Group-Oriented Signature Size

The main contribution to the communication overhead, *post* signature generation, is made by the size of the threshold group signature. The threshold signature size of *threshold-multisignature* schemes is bound to be dependent on the threshold parameter or, more precisely, $t + c$, where $0 \leq c \leq (n - t)$. This conclusion is naturally drawn from the *traceability property* of threshold-multisignature schemes, as given in Section 1, which specifies that any outsider must be able to retrieve the identities of the individual signers from the threshold signature. The threshold signature must thus be bound to information explicitly linked to each of the signers that collaborated to generate the threshold signature. In the case of the proposed scheme, the information is the identities of the individual signers contained in B. The individual identities of the group members can be carefully chosen to significantly reduce the size of the threshold-multisignature.

4.7.3 Communication Cost of Signature Generation and Verification

In terms of communication cost, the individual and threshold signature generation mechanisms of all the existing threshold-multisignature schemes and the proposed scheme are almost equivalent. Multiparty signature schemes constructed from ElGamal type (discrete logarithm-based) signature variants are bound to be interactive. In round one, each participant generates a commitment r_i and, in the second round, generates an individual signature (s_i, r_i) on an arbitrary message m. In the third round, participants send their contribution to a combiner or designated clerk which constructs the threshold signature. Assume the authorized subset β of group members collaborate to sign m. This yields a three round protocol for existing schemes, which requires $(2|\beta|)$ broadcast messages and $(|\beta|)$ unicast messages. The proposed threshold-multisignature scheme, is to the best of the authors' knowledge, the first threshold *group-oriented* signature scheme with *traceability* that allow malicious members to be positively identified and disqualified from the group. The proposed scheme also eliminates the need for a combiner. Assuming β contains at least one malicious or faulty participant, the proposed protocol will still require three rounds and only two rounds if all individual signatures satisfy (9). Therefore, in the three round scenario, the proposed protocol requires $(3|\beta|)$ broadcast messages and, in the latter two round scenario only, $(2|\beta|)$ broadcast messages.

4.7.4 Computational Cost of Signature Generation and Verification

To make a feasible comparison between the computational cost of the proposed threshold-multisignature scheme and similar schemes [4], [6], [7], it is assumed that the system parameters are chosen to yield the same time complexity for exponentiations, multiplications, and summations. Although summations and, in some cases, multiplications contribute to an insignificant fraction of the overall time complexity, these operations are still included for the sake of completeness. Values that remain constant between different signature generations (for example, the *inner parts* of the Lagrange coefficients) can be precomputed and are therefore not included in the analysis. The computational cost of the schemes will be given in terms of the minimum threshold (t) members (out of the total (n) group members) required to collaboratively sign an arbitrary message m.

The threshold-multisignature schemes presented in [4], [6], [7] were chosen for analysis since they all make an attempt to eliminate a conspiracy attack by t or more colluding members.

In Table 1 and Table 2, the schemes, with the assistance of a trusted key distribution center (KDC), are compared. Note that, although this paper does not present a threshold-multisignature scheme with a KDC, it can be trivially extended to incorporate a KDC *without* making any changes to the individual/threshold signature generation and verification procedures. Table 3 and Table 4 compare the proposed threshold-multisignature scheme, without a KDC, with the existing schemes.

In the schemes proposed by Wang et al. [6] and Li et al. [8], the public values of the individual signers are captured within the Lagrange interpolation polynomial $h(y)$ as given in (19). The coefficients $(b_0, b_1, \ldots, b_{t-1})$ of $h(y)$ are appended to the threshold signature by the combiner/clerk, which enables the threshold signature verifier to evaluate $h(y)$ in order to identify the individual signers. In Section 4.2, it was shown that appending the set of identities B to the threshold signature is a better solution, saving the threshold signature verifier the computational effort of evaluating $h(y)$. Another factor to consider is the computational cost of computing the coefficients $(b_0, b_1, \ldots, b_{t-1})$. It can be shown that the total computational cost of computing $(b_0, b_1, \ldots, b_{t-1})$ from t data pairs is given as:

$$\text{Multiplications} : t\left(\sum_{i=1}^{t-2}\left[\binom{t-1}{i+1} i\right] + (t-2) + t + 2\right),$$

$$\text{Summations} : t\left(\sum_{i=1}^{t-2}\left[\binom{t-1}{i+1} - 1\right] + (t-1) + t\right),$$

$$\text{Inversions} : t.$$

It should be clear that using an interpolation polynomial $h(y)$ to identify the individual signers is impractical for large values of threshold t.

The computational overhead that causes the most concern is the number of exponentiations in the individual signature verification equation ((9)) and threshold signature

TABLE 1
Computational Cost Comparison of Individual Signature Generation and Verification (with the Assistance of a Trusted Share Distribution Center)

Protocols	Signature generation			
	Summations	Multiplications	Exponentiations	Inversions
Li *et al.* [4]	1	$(t-1)+(t-2)+2$	1	0
Wang *et al.* [6]	1	$(t-2)+4$	3	1
Lee *et al.* [7]	2	$(t-1)+(t-2)+4$	1	0
PTMS*(Section-II-C)	2	$(t-1)+(t-2)+2$	1	0
Protocols	**Signature verification**			
	Summations	Multiplications	Exponentiations	Inversions
Li *et al.* [4]	0	$t((t-2)+1)$	$3t$	0
Wang *et al.* [6]	0	t	$3t$	0
Lee *et al.* [7]	0	$t[(t-2)+3]$	$5t^{\dagger}$	0
PTMS*(Section-II-D)	0	$2(t-1)$	$3(t-1)$	0

* *Proposed threshold-multisignature scheme (PTMS), see Section 2.*
† *Note that computations of the individual signatures are performed modulo* n.

TABLE 2
Computational Cost Comparison of Threshold Signature Generation and Verification (with the Assistance of a Trusted Share Distribution Center)

Protocols	Signature generation			
	Summations	Multiplications	Exponentiations	Inversions
Li *et al.* [4]	$t-1$	$t-1$	0	0
Wang *et al.* [6]	$s^{\ddagger}$	$2(t-1)+m^{\dagger}$	0	t
Lee *et al.* [7]	$t-1$	$t-1$	0	0
PTMS*(Section-II-E)	$t-1$	0	0	0
Protocols	**Signature verification**			
	Summations	Multiplications	Exponentiations	Inversions
Li *et al.* [4]	0	$t(t-2)+(t-1)+2$	$t+2$	0
Wang *et al.* [6]	0	$n(t-1)+1$	$n(t-2)+3$	0
Lee *et al.* [7]	0	$(t-1)+2$	5	0
PTMS*(Section-II-F)	0	$(t-1)+2$	2	0

* *Proposed threshold-multisignature scheme (PTMS), see Section 2.*
† $m = t(\sum_{i=1}^{t-2}[\binom{t-1}{i+1}i] + (t-2)+t+2)$.
‡ $s = t(\sum_{i=1}^{t-2}[\binom{t-1}{i+1}-1] + (t-1)+t)$.

verification equation ((12)), which are anticipated to contribute the bulk of the verification time complexity. The justification for looking critically at the verification processes is substantiated by the notion that a signature is normally generated only once, but verified many times. The optimum number of exponentiations for an ElGamal type signature variant is 2 [9]. It can thus be concluded that the proposed threshold-multisignature scheme is superior to existing schemes since it requires only two exponentiations for threshold signature verification, while guaranteeing *break-resistance*. For individual signature verification, three exponentiations are required, one more than the optimal two exponentiations. The additional exponentiation is as a consequence of satisfying the stronger break-resistance property. The proposed scheme also provides improved efficiency for all other operations, with or without the assistance of a trusted authority, which includes, individual signature generation (Table 1 and Table 3) and threshold signature generation (Table 2 and Table 4).

4.7.5 Efficiency of Initial Key Distribution and Key Redistribution/Update Protocols

The proposed threshold cryptosystem is constructed with the round optimal DKG protocol presented in Section 2.2, which

TABLE 3
Computational Cost Comparison of Individual Signature Generation and Verification (without the Assistance of a Trusted Share Distribution Center)

Protocols	Signature generation			
	Summations	Multiplications	Exponentiations	Inversions
Li *et al.* [4]	0	$(t-1)+(n-t)[(t-2)+1]+1$	1	0
Wang *et al.* [6]	1	$(n-t)(t-2)+(n-t-1)+5$	$(n-t)+2$	1
PTMS*(Section-II-C)	2	$(t-1)+(t-2)+2$	1	0
Protocols	**Signature verification**			
	Summations	Multiplications	Exponentiations	Inversions
Li *et al.* [4]	0	$t[(t-2)+(n-t-1)+2]$	$3t$	0
Wang *et al.* [6]	0	t	$3t$	0
PTMS*(Section-II-D)	0	$2(t-1)$	$3(t-1)$	0

* *Proposed threshold-multisignature scheme (PTMS), see Section 2.*

TABLE 4
Computational Cost Comparison of Threshold Signature Generation and Verification (without the Assistance of a Trusted Share Distribution Center)

Protocols	Signature generation			
	Summations	Multiplications	Exponentiations	Inversions
Li *et al.* [4]	$t-1$	$t-1$	0	0
Wang *et al.* [6]	$s^{\ddagger}$	$2(t-1)+m^{\dagger}$	0	t
PTMS*(Section-II-E)	$t-1$	0	0	0
Protocols	**Signature verification**			
	Summations	Multiplications	Exponentiations	Inversions
Li *et al.* [4]	0	$t[(t-2)+(n-t-1)]+(t-1)+2$	$t+2$	0
Wang *et al.* [6]	0	$n(t-1)+1$	$n(t-2)+3$	0
PTMS*(Section-II-F)	0	$(t-1)+2$	2	0

* *Proposed threshold-multisignature scheme (PTMS), see Section 2.*

$^{\dagger} m = t(\sum_{i=1}^{t-2}[\binom{t-1}{i+1} i] + (t-2) + t + 2)$.

$^{\ddagger} s = t(\sum_{i=1}^{t-2}[\binom{t-1}{i+1} - 1] + (t-1) + t)$.

effectively eliminates the need for a mutually trusted key distribution center (KDC). Existing threshold-multisignature schemes that do not need the assistance of a trusted authority [4], [6], [8], do not use secure and efficient one round DKG protocols to initialize the threshold cryptosystem. These schemes also do not consider *proactive* security because they do not allow for the periodic updating of secret shares nor do they permit *dynamic* group membership since they cannot redistribute the secret shares to a new access structure. Based on the above, the existing systems are rigid concerning the threshold security parameters (n, t), which sets the security/availability trade-off. Accordingly, such systems cannot respond to changes in the networking environment. For these reasons, the efficiency of the proposed threshold-multisignature scheme's system setup and maintenance procedures cannot be compared to the existing schemes.

It is the view of the authors that the security of the underlying DKMI influences the security of the signature scheme to such an extent that the threshold group-oriented signature scheme cannot be considered separately from the DKMI.

The *worst-case* computational cost of the *initial* DKG protocol and DKRU protocol are briefly considered.

Efficiency analysis on the initial DKG protocol (Section 2.2) and DKRU protocol (Section 3) shows that these protocols require $O(nt)$ and $O(n't')$ exponentiations, respectively. Multiplications yield a similar growth rate to the exponentiations. Each protocol participant is required to generate random numbers $O(t)$ for initial DKG and $O(t')$ for DKRU.

The threshold cryptosystem accommodates a total of n^2 shares. Each participant broadcasts $O(t) + O(n)$ messages for initial DKG and $O(t') + O(n')$ for DKRU. The network as a whole has a communication cost of $O(nt) + O(n^2)$ and $O(n't') + O(n'^2)$ for initial DKG and DKRU, respectively.

5 Conclusion

Investigations within the fields of threshold *group-oriented* signature schemes, threshold *group* signature schemes, *multisignature* schemes, and *threshold-multisignature* schemes resulted in explicitly defining the properties of threshold-multisignature schemes. The main aim of this paper is to introduce such a secure threshold-multisignature scheme. To reach this objective, the secure and optimally efficient ElGamal type signature variant, $GES = (M.EG\text{II}.3.\sigma(1), r, s, h(m, r), 1)$, was extended to a multiparty setting to yield a threshold-multisignature scheme, which from the authors' point of view is the first to provide a guaranteed *traceability property*. The proposed threshold-multisignature scheme was shown to satisfy all of the specified security requirements and to fulfill the stronger *break-resistant property*. The threshold-multisignature signature scheme thus remains secure, even if the threshold cryptosystem has been broken, i.e., the group secret or individual secret shares are known or controlled by an adversary. It was shown that the proposed threshold-multisignature scheme eliminates the latest attacks on similar threshold signature schemes with traceability. The efficiency analysis showed that the proposed threshold-multisignature scheme outperforms other existing schemes and is optimal in terms of exponentiations with respect to threshold signature verification and near optimal for individual signature verification, while providing break resistance.

The paper also addressed the issue of *initial* distributed-key generation (DKG) to construct a threshold cryptosystem without the assistance of a trusted authority. To realize initial DKG, improvements were made to the round optimal DKG scheme of Zhang and Imai. The paper proposes a novel publicly verifiable, round optimal (one round) distributed-key redistribution/updating (DKRU) protocol that eliminates a major problem with existing schemes; malicious or faulty protocol participants from the original access structure are positively identifiable by the existing honest members as well as the new members joining the threshold cryptosystem, which avoids repeating the secret redistribution protocol until all the verification conditions holds.

The paper proposes the first integrated (single protocol) solution for DKG, distributed-key updating (DKU), and distributed-key redistribution (DKR). Although the initial DKG protocol and DKRU protocol are presented separately for clarity, it was shown that the two protocols are essentially equivalent. The same *initial* DKG protocol can thus be used to realize DKR and DKU by merely keeping the access structure constant. The paper defines the notion of a *distributed-key management infrastructure* (DKMI) as a protocol suite that considers all aspects of secret distribution (DKG), secret updating (DKU), and secret redistribution (DKR). Considering the minor differences between the *initial* DKG protocol and DKRU protocol, the proposed DKMI requires considerably less implementation effort than implementing three unrelated protocols to realize distributed-key management.

Use of the DKRU mechanism makes the proposed *fully* distributed threshold-multisignature scheme proactively secure, allows for *dynamic* group membership, and gives the group members the capability of adjusting the availability/security trade-off by redistributing the existing access structure $\Gamma_P^{(n,t)}$ to a new access structure $\Gamma_{P'}^{(n',t')}$.

Acknowledgments

This work was supported by ARMSCOR, the Armaments Corporation of South Africa. The authors thank the reviewers and editors for their time, effort, and positive input.

References

[1] Y. Desmedt, "Society and Group Oriented Cryptography: A New Concept," *Proc. Advances in Cryptology—CRYPTO '87,* 1987.
[2] Y. Desmedt, "Threshold Cryptography," *European Trans. Telecomm.,* vol. 5, no. 4, pp. 449-457, 1994.
[3] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems," *Proc. Advances in Cryptology—EUROCRYPT '99,* May 1999.
[4] C.-M. Li, T. Hwang, and N.-Y. Lee, "Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders," *Proc. Advances in Cryptology—EUROCRYPT '94,* May 1994.
[5] A. Boldyreva, "Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme," *Proc. Public Key Cryptography—PKC '03,* 2003.
[6] C.-T. Wang, C.-H. Lin, and C.-C. Chang, "Threshold Signature Schemes with Traceable Signers in Group Communications," *Computer Comm.,* vol. 21, no. 8, pp. 771-776, 1998.
[7] W.-B. Lee and C.-C. Chang, "(t, n) Threshold Digital Signature with Traceability Property," *J. Information Science and Eng.,* vol. 15, no. 5, pp. 669-678, 1999.
[8] Z.-C. Li, J.-M. Zhang, J. Luo, W. Song, and Y.-Q. Dai, "Group-Oriented (t, n) Threshold Digital Signature Schemes with Traceable Signers," *Proc. Second Int'l Symp. Topics in Electronic Commerce (ISEC '01),* Apr. 2001.
[9] P. Horster, M. Michels, and H. Petersen, "Generalized ElGamal Signatures for One Message Block," *Proc. Second Int'l Workshop IT-Security,* Sept. 1994.
[10] L. Harn and Y. Xu, "Design of Generalised ElGamal Type Digital Signature Schemes Based on Discrete Logarithms," *Electronics Letters,* vol. 30, no. 24, pp. 2025-2026, 1994.
[11] G. Wang, "On the Security of the Li-Hwang-Lee-Tsai Threshold Group Signature Scheme," *Proc. Fifth Int'l Conf. Information Security and Cryptography (ICISC '02),* Nov. 2002.
[12] H. Pedersen, "How to Convert Any Digital Signature Scheme into a Group Signature Scheme," *Proc. Fifth Int'l Workshop Security Protocols,* Apr. 1997.
[13] L. Zhou and Z.J. Haas, "Securing Ad Hoc Networks," *IEEE Network,* special issue on network security, vol. 13, no. 6, pp. 24-30, 1999.
[14] T.P. Pedersen, "A Threshold Cryptosystem without a Trusted Party," *Proc. Advances in Cryptology—EUROCRYPT '91,* 1991.
[15] M. Stadler, "Publicly Verifiable Secret Sharing," *Proc. Advances in Cryptology—EUROCRYPT '96,* 1996.
[16] P.-A. Fouque and J. Stern, "One Round Threshold Discrete-Log Key Generation without Private Channels," *Proc. Public Key Cryptography—PKC '01,* Feb. 2001.
[17] R. Zhang and H. Imai, "Round Optimal Distributed Key Generation of Threshold Cryptosystem Based on Discrete Logarithm Problem," *Proc. Applied Cryptography and Network Security (ACNS '03),* Oct. 2003.
[18] A. Herzberg, S. Jaracki, H. Krawczyk, and M. Yung, "Proactive Secret Sharing or: How to Cope with Perpetual Leakage," *Proc. Advances in Cryptology—CRYPTO '95,* 1995.
[19] Y. Desmedt and S. Jajodia, "Redistributing Secret Shares to New Access Structures and Its Applications," Technical Report ISSE-TR-97-01, Dept. of Information and Software Eng., School of Information Technology and Eng., George Mason Univ., July 1997.
[20] T.M. Wong, C. Wang, and J.M. Wing, "Verifiable Secret Redistribution for Archive System," *Proc. First Int'l IEEE Security in Storage Workshop,* Dec. 2002.

[21] A. Shamir, "How to Share a Secret," *Comm. ACM,* vol. 22, no. 11, pp. 612-613, 1979.
[22] J. Camenisch and I. Damgård, "Verifiable Encryption, Group Encryption, and Their Applications to Separable Group Signatures and Signature Sharing Schemes," *Proc. Advances in Cryptology—ASIACRYPT '00,* 2000.
[23] Y.-M. Tseng and J.-K. Jan, "Attacks on Threshold Signature Schemes with Traceable Signers," *Information Processing Letters,* vol. 71, no. 1, pp. 1-4, 1999.
[24] T.-S. Wu and C.-L. Hsu, "Cryptanalysis of Group-Oriented (t, n) Threshold Digital Signature Schemes with Traceable Signers," *Computer Standards and Interfaces,* vol. 26, no. 5, pp. 477-481, 2004.
[25] G. Wang, X. Han, and B. Zhu, "On the Security of Two Threshold Signature Schemes with Traceable Signers," *Proc. First Int'l Conf. Applied Cryptography and Network Security (ACNS '03),* Oct. 2003.
[26] C.P. Schnorr and M. Jakobsson, "Security of Discrete Log Cryptosystems in the Random Oracle and Generic Model," *Proc. Math. of Public-Key Cryptography,* June 1999.
[27] R. Canetti, R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Adaptive Security for Threshold Cryptosystems," *Proc. Advances in Cryptology—CRYPTO '99,* Aug. 1999.
[28] M. Michels and P. Horster, "On the Risk of Disruption in Several Multiparty Signature Schemes," *Proc. Advances in Cryptology—ASIACRYPT '96,* Nov. 1996.

Johann van der Merwe received the BSc degree in electronic engineering from the University of Natal in 2003 and the MSc degree from the University of KwaZulu-Natal in 2005. He is currently pursuing the PhD degree in distributed communication systems. His research interests include security issues in ad hoc networks, with particular focus on key management, threshold group-oriented cryptography, and digital signature schemes.

Dawoud S. Dawoud received the BSc (1965) and MSc (1969) degrees in electronic engineering from Cairo University and the PhD (1973) degree in computer engineering from Russia. He has been a full professor since 1984. He has taught courses in computer engineering, digital signal processing, and digital telecommunications in numerous universities, including Cairo University, University of Lagos, and University of Botswana. He has supervised many PhD and MSc theses and published more than 90 papers and books. He holds two patents in computer architecture. He is currently a professor of computer engineering and signal processing at the University of KwaZulu-Natal, South Africa. He is a member of the IEEE.

Stephen McDonald received the BSc (1987) and MSc (1989) degrees in electronic engineering from the University of Natal, South Africa. After working in the communications industry for five years, he rejoined the University as a senior lecturer in 1997. His current research interests include ad hoc networks and various aspects of image processing. He is a member of the IEEE.