Hierarchical Group Signatures

Mårten Trolin and Douglas Wikström

Royal Institute of Technology (KTH),
Nada, SE-100 44 Stockholm, Sweden
{marten, dog}@nada.kth.se

Abstract. We introduce the notion of *hierarchical group signatures*. This is a proper generalization of group signatures, which allows multiple group managers organized in a tree with the signers as leaves. When opening a signature a group manager only learns to which of its subtrees, if any, the signer belongs.

We provide definitions for the new notion and construct a scheme that is provably secure given the existence of a family of trapdoor permutations. We also present a construction which is relatively practical, and prove its security in the random oracle model under the strong RSA assumption and the DDH assumption.

1 Introduction

Consider the notion of group signatures introduced by Chaum and van Heyst [13]. A group member can compute a signature that reveals nothing about the signer's identity except that he is a member of the group. On the other hand the group manager can always reveal the identity of the signer.

An application for group signatures is anonymous credit cards. The cardholder wishes to preserve his privacy when he pays a merchant for goods, i.e., he is interested in unlinkability of payments. The bank must obviously be able to extract the identity of a cardholder from a payment or at least an identifier for an account, to be able to debit the account. To avoid fraud, the bank, the merchant, and the cardholder all require that a cardholder cannot pay for goods without holding a valid card. To solve the problem using group signatures we let the bank be the group manager and the cardholders be signers. A cardholder signs a transaction and hands it to the merchant. The merchant then hands the signed transaction to the bank, which debits the cardholder and credits the merchant. Since signatures are unlinkable, the merchant learns nothing about the cardholder's identity. The bank on the other hand can always extract the cardholder's identity from a valid signature and debit the correct account.

The above scenario is somewhat simplified since normally there are many banks that issue cards of the same brand which are processed through the same payment network. The payment network normally works as an administrator and routes transactions to several independent banks. Thus, the merchant hands a payment to the payment network which hands the payment to the issuing bank. We could apply group signatures here as well by making the payment network act as the group manager. The network would then send the extracted identity to the issuing bank. Another option is to set up several independent group signatures schemes, one for each issuer. In the first approach, the payment network learns the identity of the customer, and in the second approach the merchant learns which bank issued the customer's card. A better solution would reveal nothing except what is absolutely necessary to each party. The merchant needs to be convinced that the credit card is valid, the payment network must be able to route the payment to the correct card issuer, and the issuer must be able to determine the identity of the cardholder.

In this extended abstract we introduce and investigate the notion of *hierarchical group signatures.* These can be employed to solve the above problem. When using a hierarchical group signature scheme there is not one single group manager. Instead there are several group managers organized in a tree, i.e., each group manager either manages a group of signers or a group of group managers. In the original notion the group manager can always identify the signer of a message, but nobody else can distinguish between signatures by different signers. The corresponding property for hierarchical group signatures is more complicated. When opening a signature from a signer in its subtree, a group manager learns to which of the subtrees directly below it the signer belongs. Signatures from other signers are indistinguishable. Hence a group manager on the level directly above the signers can identify its signers, whereas group managers higher in the hierarchy only learns to which subtree the signer belongs.

When we use hierarchical group signatures to construct anonymous credit cards for the more realistic setting we let the payment network be the root manager that manages a set of group managers, i.e., the issuing banks, and we let the cardholders be signers. The credit card application also demonstrates what kind of responsibility model is likely to be used with a hierarchical group signature scheme. With a valid signature on a transaction, the merchant has a valid demand on the payment network. If the payment network has a signature that can be shown to belong to a certain bank, the network has a valid demand on that bank. Thus, it is in the network's interest to open the signatures it receives from merchants, and it is in the issuing banks' interest to open the signatures they receive from the network.

1.1 Previous Work

The concept of group signatures was first introduced by Chaum and van Heyst [13] in 1991. This and the group signature schemes that followed [14, 7] all have the property that the complexity of the scheme grows with the number of participants. In [11] Camenisch and Stadler presented a system where the key does not grow with the number of participants. This system, however, relies on a non-standard number-theoretic assumption. The assumption was actually found to be incorrect and modified in [2]. An efficient system whose security rests on the strong RSA assumption and the Diffie-Hellman decision assumption was presented by Camenisch and Michels in 1998 [10]. This system was improved in [1]. The currently most efficient scheme that is secure under standard assumptions is [8]. More efficient schemes do exist [6, 9], but they are based on bilinear maps and thus relies on less well-studied assumptions for security.

A related notion is *traceable signatures* introduced by Kiayias et al. [19], where signatures belonging to a member can be opened, or traced, in a distributed way without revealing the group secret.

Bellare et al. [4] give a definitional framework for group signatures for static groups, i.e., when the set of members cannot be changed after the initial setup. The paper also contains a scheme based on general methods in this setting. Kiayias and Yung [20] define security for dynamic groups and prove that a modification of [1] is secure under these definitions. Independently, Bellare et al. [5] extend the definitions of [4] in a similar way to handle dynamic groups, and present a scheme that is secure under general assumptions.

In [2] the concepts of *multi-group signatures* and *subgroup signatures* are described, and in [21] a system for hierarchical multi-groups is given. It may be worthwhile to consider the differences between these concepts and hierarchical signatures introduced here. Subgroup signatures make it possible for an arbitrary number i of signers to produce a joint signature which can be verified to stem from i distinct group members. Multi-group signature schemes allow a signer who is a member of two groups to produce a signature that shows membership of either both groups or just one of them. In hierarchical multi-groups a signer who is a member of a supergroup with subgroups can produce a signature that reveals membership either of the supergroup or of a subgroup of his choice. However, the opening procedure is not hierarchical, i.e., there are no group managers for the subgroups.

1.2 Notation

Throughout the text, κ denotes a security parameter. A function $f : \mathbb{N} \rightarrow [0, 1]$ is said to be negligible if for each $c > 0$ there exists a $\kappa_0 \in \mathbb{N}$ such that $f(\kappa) < \kappa^{-c}$ for $\kappa_0 < \kappa \in \mathbb{N}$. We write $\emptyset$ to denote both the empty set and the empty string. If T is a tree we denote by $\mathcal{L}(T)$ its set of leaves and by $\mathcal{V}(T)$ the set of all vertices. We write G_q for the unique subgroup of order q of $\mathbb{Z}_p^*$ for a prime $p = 2q + 1$. In the ElGamal cryptosystem a secret key is a randomly generated $x \in \mathbb{Z}_q$ and the public key is $y = g^x$. To encrypt message $m \in G_q$, $r \in \mathbb{Z}_q$ is chosen randomly and the cryptotext is given by $(u, v) = E_y(m, r) = (g^r, y^r m)$. To decrypt a cryptotext $D_x(u, v) = u^{-x} v = m$ is computed. We denote by $N = PQ$ an RSA module for two strong primes P and Q, and let QR_N be the subgroup of squares in $\mathbb{Z}_N^*$ with generators $\mathbf{g}$ and $\mathbf{h}$. The adversaries in this paper are modeled as polynomial time Turing machines with *non-uniform* auxiliary advice string. We denote the set of such adversaries by PPT^*.

2 Hierarchical Group Signatures

In this section we discuss the notion of hierarchical group signatures. We begin by describing the parties of a hierarchical group signature system. Then we proceed by giving formal definitions.

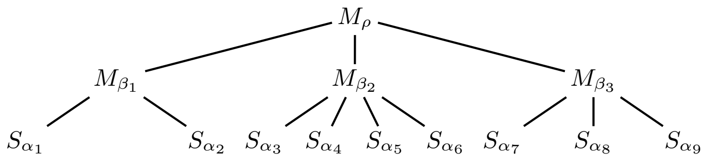

Fig. 1. A tree of group managers and signers, where $\rho = \{\beta_1, \beta_2, \beta_3\}$, $\beta_1 = \{\alpha_1, \alpha_2\}$, $\beta_2 = \{\alpha_3, \alpha_4, \alpha_5, \alpha_6\}$, and $\beta_3 = \{\alpha_7, \alpha_8, \alpha_9\}$

There are two types of parties: signers denoted S_α for α in some index set $\mathcal{I}$, and group managers denoted M_α for indices α described below. The parties form a tree T, where the signers are leaves and the group managers are inner nodes. The indices of the group managers are formed as follows. If a group manager manages a set of signers $\beta \subset \mathcal{I}$ we denote it by M_β. This corresponds to M_β having S_α for $\alpha \in \beta$ as children. If a group manager M_γ manages a set of group managers $\{M_{\beta_1}, \ldots, M_{\beta_l}\}$ we denote it by M_γ where $\gamma = \{\beta_1, \ldots, \beta_l\}$. This corresponds to M_γ having M_{β_i} for $i = 1, \ldots, l$ as children. Let M_ρ denote the root group manager. We define the root group manager to be at depth 0 and assume that all leaves in the tree are at the same depth δ. Figure 1 illustrates a tree of parties.

Note that standard group signatures correspond to having a single group manager $M_{\{1,\ldots,l\}}$ that manages all signers $S_1, \ldots, S_l$.

2.1 Definition of Security

Bellare et al. [4] give a definition of a group signature scheme, but more importantly they argue that two properties of group signatures, full anonymity and full traceability, imply any reasonable security requirements one can expect from a group signature scheme. We follow their definitional approach closely.

Definition 1 (Hierarchical Group Signature). *A* hierarchical group signature scheme $\mathcal{HGS} = (\mathsf{HKg}, \mathsf{HSig}, \mathsf{HVf}, \mathsf{HOpen})$ *consists of four polynomial-time algorithms*

1. *The randomized* key generation algorithm HKg *takes as input* $(1^\kappa, T)$*, where* T *is a tree of size polynomially bounded in the security parameter* κ *with all leaves at the same depth, and outputs a pair of maps* $\mathrm{hpk}, \mathrm{hsk} : \mathcal{V}(T) \rightarrow \{0,1\}^*$*. For each node (or leaf)* α*,* $\mathrm{hpk}(\alpha)$ *is the public key and* $\mathrm{hsk}(\alpha)$ *is the secret key.*
2. *The randomized* signature algorithm HSig *takes as input a message* m*, a tree* T*, a public key* hpk*, and a secret signing key* $\mathrm{hsk}(\alpha)$*, and returns a signature of* m*.*
3. *The deterministic* signature verification algorithm HVf *takes as input a tree* T*, a public key* hpk*, a message* m *and a candidate signature* σ *of* m *and returns either* 1 *or* 0*.*

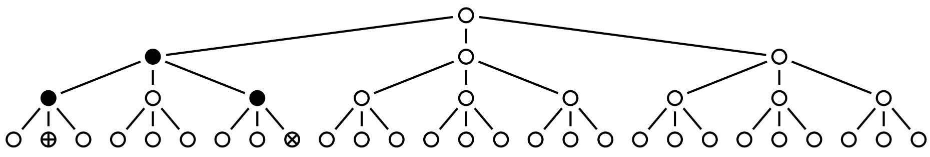

Fig. 2. Nodes in black represent group managers able to distinguish between signatures by $S_{\alpha^{(0)}}$ and $S_{\alpha^{(1)}}$, the two marked leaves

4. *The deterministic* opening algorithm HOpen *takes as input a tree* T*, a public key* hpk*, a secret opening key* $\mathrm{hsk}(\beta)$*, a message* m*, and a candidate signature* σ*. It outputs an index* $\alpha \in \beta$ *or* $\perp$*.*

We need to define what we mean by security for a hierarchical group signature scheme. We begin with anonymity. Consider Figure 2, where two signers $S_{\alpha^{(0)}}$ and $S_{\alpha^{(1)}}$ are marked. Assume that it is known that a message has been signed by one of them. Then any group manager on the path leading from $S_{\alpha^{(0)}}$ or $S_{\alpha^{(1)}}$ to their first common ancestor can determine which of them signed the message. In the figure those group managers are marked with black. In the definition of anonymity we capture the property that unless the adversary corrupts one of these group managers, it cannot determine whether $S_{\alpha^{(0)}}$ or $S_{\alpha^{(1)}}$ signed the message, even if the adversary is given the private keys of all signers and is allowed to select $\alpha^{(0)}$, $\alpha^{(1)}$ and the message itself.

We define Experiment 1 to formalize these ideas. Throughout the experiment the adversary has access to an $\mathsf{HOpen}(T, \mathrm{hpk}, \mathrm{hsk}(\cdot), \cdot, \cdot)$ oracle. At the start of the experiment the adversary is given the public keys of all parties and the private keys of all signers. Then it can adaptively ask for the private keys of the group managers. At some point it outputs the indices $\alpha^{(0)}$ and $\alpha^{(1)}$ of two leaves and a message m. The $\mathsf{HSig}(\cdot, T, \mathrm{hpk}, \mathrm{hsk}(\cdot))$ oracle computes the signature of m using the private key $\mathrm{hsk}(\alpha^{(b)})$ and hands it to the adversary. The adversary finally outputs a guess d of the value of b. If the scheme is anonymous the probability that $b = d$ should be negligibly close to $1/2$ when b is a randomly chosen bit. The labels $\mathsf{corrupt}$, choose and guess below distinguish between the phases of the experiment.

Experiment 1 (Hierarchical Anonymity, $\mathbf{Exp}^{\mathsf{anon}-b}_{\mathcal{HGS},A}(\kappa, T)$).

```
$(\mathrm{hpk}, \mathrm{hsk}) \leftarrow \mathsf{HKg}(1^\kappa, T)$;  $s_{\mathrm{state}} \leftarrow (\mathrm{hpk}, \mathrm{hsk}(\mathcal{L}(T)))$;  $\mathcal{C} \leftarrow \emptyset$;  $\alpha \leftarrow \emptyset$;
Do
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\alpha\}$
    $(s_{\mathrm{state}}, \alpha) \leftarrow A^{\mathsf{HOpen}(T,\mathrm{hpk},\mathrm{hsk}(\cdot),\cdot,\cdot)}(\mathsf{corrupt}, s_{\mathrm{state}}, \mathrm{hsk}(\alpha))$
While  $(\alpha \in \mathcal{V}(T) \setminus \mathcal{C})$
$(s_{\mathrm{state}}, \alpha^{(0)}, \alpha^{(1)}, m) \leftarrow A^{\mathsf{HOpen}(T,\mathrm{hpk},\mathrm{hsk}(\cdot),\cdot,\cdot)}(\mathsf{choose}, s_{\mathrm{state}})$
$\sigma \leftarrow \mathsf{HSig}(m, T, \mathrm{hpk}, \mathrm{hsk}(\alpha^{(b)}))$
$d \leftarrow A^{\mathsf{HOpen}(T,\mathrm{hpk},\mathrm{hsk}(\cdot),\cdot,\cdot)}(\mathsf{guess}, s_{\mathrm{state}}, \sigma)$
```

Let B be the set of nodes on the paths from $\alpha^{(0)}$ and $\alpha^{(1)}$ up to their first common ancestor α_t excluding $\alpha^{(0)}$ and $\alpha^{(1)}$ but including α_t, i.e., the set of nodes $\alpha_l^{(0)}$, $\alpha_l^{(1)}$, $l = t, \ldots, \delta - 1$, such that

$$\alpha^{(0)} \in \alpha^{(0)}_{\delta-1} \in \alpha^{(0)}_{\delta-2} \in \ldots \in \alpha^{(0)}_{t+1} \in \alpha_t \ni \alpha^{(1)}_{t+1} \ni \ldots \ni \alpha^{(1)}_{\delta-2} \ni \alpha^{(1)}_{\delta-1} \ni \alpha^{(1)} .$$

If $B \cap \mathcal{C} \neq \emptyset$ or if A asked its $\mathsf{HOpen}(T, \mathrm{hpk}, \mathrm{hsk}(\cdot), \cdot, \cdot)$ oracle a question $(\alpha_l^{(0)}$, m, $\sigma)$ or $(\alpha_l^{(1)}, m, \sigma)$ return 0. Otherwise return d.

Consider the above experiment with a depth one tree T and root ρ. In that case we may assume that $\mathrm{hsk}(\rho)$ is never handed to the adversary, since the adversary fails in that case anyway. Similarly the $\mathsf{HOpen}(T, \mathrm{hpk}, \mathrm{hsk}(\cdot), \cdot, \cdot)$ oracle reduces to the Open oracle in [4]. Thus, our experiment reduces to the experiment for full anonymity given in [4] where the adversary gets the secret keys of all signers, but only the public key of the group manager.

Next we consider how the notion of full traceability can be defined in our setting. Full traceability as defined in [4] is similar to security against chosen message attacks (CMA-security) as defined by Goldwasser, Micali and Rivest [18] for signatures. The only essential difference is that the group manager must always be able to open a signature and identify the signer. In our setting this amounts to the following. Given a signature deemed valid by the HVf algorithm, the root should always be able to identify the child directly below it of which the signer is a descendent. The child should have the same ability for the subtree of which it is a root and so on until the child itself is a signer.

Again we define an experiment consisting of two phases. The adversary is given the secret keys of all group managers. Then the adversary adaptively chooses a set of signers to corrupt. Then in a second phase the adversary outputs a message and a signature. If the output amounts to a signature deemed valid by HVf and the signer cannot be traced, or if the signature is traced to a non-corrupted signer, the adversary has succeeded and the experiment outputs 1. Otherwise it outputs 0. Thus, the distribution of the experiment should be negligibly close to 0 for all adversaries if the scheme is secure.

Experiment 2 (Hierarchical Traceability, $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{HGS},A}(\kappa, T)$).

```
$(\mathrm{hpk}, \mathrm{hsk}) \leftarrow \mathsf{HKg}(1^\kappa, T)$;  $s_{\mathrm{state}} \leftarrow (\mathrm{hpk}, \mathrm{hsk}(\mathcal{V}(T) \backslash \mathcal{L}(T))$;  $\mathcal{C} \leftarrow \emptyset$;  $\alpha \leftarrow \emptyset$;
Do
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\alpha\}$
    $(s_{\mathrm{state}}, \alpha) \leftarrow A^{\mathsf{HSig}(\cdot, T, \mathrm{hpk}, \mathrm{hsk}(\cdot))}(\mathsf{corrupt}, s_{\mathrm{state}}, \mathrm{hsk}(\alpha))$
While   $(\alpha \in \mathcal{V}(T) \setminus \mathcal{C})$
$(m, \sigma) \leftarrow A^{\mathsf{HSig}(\cdot, T, \mathrm{hpk}, \mathrm{hsk}(\cdot))}(\mathsf{guess}, s_{\mathrm{state}})$
```

If $\mathsf{HVf}(T, \mathrm{hpk}, m, \sigma) = 0$ return 0. Define $\alpha_0 = \rho$ and $\alpha_l = \mathsf{HOpen}(T$, hpk, $\mathrm{hsk}(\alpha_{l-1})$, m, $\sigma)$ for $l = 1, \ldots, \delta$. If $\alpha_l = \perp$ for some $0 < l \leq \delta$ return 1. If $\alpha_\delta \notin \mathcal{C}$ and the $\mathsf{HSig}(\cdot, T, \mathrm{hpk}, \mathrm{hsk}(\cdot))$ oracle did not get a question (m, α_δ) return 1. Otherwise return 0.

Consider the experiment above with a depth one tree. This corresponds to giving the adversary the secret key of the group manager, and letting it adaptively choose additional signing keys. This is precisely the setting of [4].

The advantage of the adversary is defined in the natural way by setting $\mathbf{Adv}^{\mathsf{anon}}_{\mathcal{HGS},A}(\kappa,T) = |\Pr[\mathbf{Exp}^{\mathsf{anon}-0}_{\mathcal{HGS},A}(\kappa,T)=1] - \Pr[\mathbf{Exp}^{\mathsf{anon}-1}_{\mathcal{HGS},A}(\kappa,T)=1]|$ and $\mathbf{Adv}^{\mathsf{trace}}_{\mathcal{HGS},A}(\kappa,T) = \mathbf{Exp}^{\mathsf{trace}}_{\mathcal{HGS},A}(\kappa,T)$.

Definition 2 (Security of Hierarchical Group Signatures). *A hierarchical group signature scheme* $\mathcal{HGS} = (\mathsf{HKg},\mathsf{HSig},\mathsf{HVf},\mathsf{HOpen})$ *is* secure *if for all trees* T *of polynomial size in* κ *with all leaves at the same depth, and all* $A \in \mathrm{PPT}^*$, $\mathbf{Adv}^{\mathit{trace}}_{\mathcal{HGS},A}(\kappa,T) + \mathbf{Adv}^{\mathit{anon}}_{\mathcal{HGS},A}(\kappa,T)$ *is negligible.*

3 Our Constructions

We construct two hierarchical group signature schemes, one under general assumptions, and one under standard assumptions in the random oracle model. Both require a trusted key generator at the start of the protocol. The two constructions are different, but based on similar ideas. In this extended abstract we only give the main ideas behind our constructions. Detailed descriptions and proofs of our claims are given in the full paper [26].

3.1 Our Approach

All known group signatures are based on the idea that the signer encrypts a secret of some sort using the group manager's public key, and then proves that the resulting cryptotext is on this special form. The security of the cryptosystem used implies anonymity, since no adversary can distinguish cryptotexts of two distinct messages if they are encrypted using the *same* public key. We generalize this approach.

First we consider the problem of forwarding partial information on the identity of the signer to group managers without leaking information. Each group manager M_β is given a secret key sk_β and a public key pk_β of a cryptosystem. We also give each signer S_α a public key pk_α that is used to identify the signer. Each signer is associated in the natural way with the path $\alpha_0, \alpha_1, \ldots, \alpha_\delta$ from the root $\rho = \alpha_0$ to the leaf $\alpha = \alpha_\delta$ in the tree T of group managers and signers. To compute a signature, the signer computes as part of the signature a chain

$$(C_0, C_1, \ldots, C_{\delta-1}) = \left(E_{\mathrm{pk}_{\alpha_0}}(\mathrm{pk}_{\alpha_1}), E_{\mathrm{pk}_{\alpha_1}}(\mathrm{pk}_{\alpha_2}), \ldots, E_{\mathrm{pk}_{\alpha_{\delta-1}}}(\mathrm{pk}_{\alpha_\delta})\right) .$$

Note that each cryptotext C_l in the list encrypts the public key $\mathrm{pk}_{\alpha_{l+1}}$ used to form the following cryptotext. The particular structure of the chain and the fact that all leaves are on the same depth in the tree ensures that a group manager M_β on depth l can try to open a signature by decrypting C_l, i.e., it computes $\mathrm{pk} = D_{\mathrm{sk}_\beta}(C_l)$. If $\alpha_l = \beta$, then $\mathrm{pk} = \mathrm{pk}_{\alpha_{l+1}}$. Thus, if M_β manages signers, it learns the identity of the signer S_α, and if it manages other group managers it learns the identity of the group manager below it in the tree which (perhaps indirectly) manages the signer S_α.

Now suppose that $\alpha_l \neq \beta$, so $\mathrm{pk} \neq \mathrm{pk}_{\alpha_{l+1}}$. What does M_β, or indeed any outsider, learn about the identity of the signer S_α? It clearly does not learn anything from a cryptotext C_l about the encrypted cleartext, as long as the cryptosystem is semantically secure. However, if the cryptotext C_{l+1} somehow indicates which public key was used to form it, M_β, or any outsider, can simply look at C_{l+1} and recover the cleartext of C_l. This means that it can look at the chain of cryptotexts and extract information on the identity of the signer. We conclude that using the approach above, we need a cryptosystem which not only hides the cleartext, but also hides the public key used to form the cryptotext. Such a cryptosystem is called an *anonymous* cryptosystem [3].

Next we consider the problem of ensuring hierarchical traceability. This problem consists of two parts. We must ensure chosen message security to avoid that an illegitimate signer is able compute a valid signature at all. However, the more difficult problem is to ensure that the signer S_α not only formed $(C_0, \ldots, C_{\delta-1})$ as described above for some public keys $\mathrm{pk}_{\alpha_0}, \ldots, \mathrm{pk}_{\alpha_\delta}$, but also that the public keys used correspond to the unique path $\alpha_0, \alpha_1, \ldots, \alpha_\delta$ from the root $\rho = \alpha_0$ to the leaf $\alpha = \alpha_\delta$ corresponding to the signer S_α. This is the main obstacle to construct an efficient hierarchical group signature scheme.

3.2 A Construction Under General Assumptions

We sketch the construction under general assumptions. To achieve hierarchical anonymity we employ the cryptosystem of Goldwasser and Micali [17] and prove that this cryptosystem is anonymous. To achieve traceability we use the group signature scheme of Bellare et al. [4] and a non-interactive adaptive zero-knowledge unbounded simulation sound proof (NIZK) as defined and constructed for any language in NP in Feige, Lapidot and Shamir [16], Sahai [23], and De Santis [24]. Both constructions are provably secure under the existence of a trapdoor permutation family.

The signer proves using the NIZK that the chain of cryptotexts is formed correctly, and the group signature scheme ensures that only legitimate signers can form a signature, without losing anonymity. The group signature also allows us to use a semantically secure cryptosystem for the chain (cf. [4, 8]), since any query to the HOpen oracle obviously can be answered correctly by the simulator if we know the full identity of the signer, i.e., we use a variant of the double-cryptotext trick of Naor and Yung [22]. The following theorem is proved in the full version [26].

Theorem 1. *If there exists a family of trapdoor permutations, then there exists a secure hierarchical group signature scheme.*

3.3 A Construction Under the DDH Assumption and the Strong RSA Assumption

To achieve hierarchical anonymity in the practical construction we employ the ElGamal cryptosystem, which is semantically secure under the DDH assumption. It is easy to see that ElGamal is also anonymous, as long as a fixed group is used for each security parameter. Thus, each group manager M_β holds a secret key x_β and a public key $y_\beta = g^{x_\beta}$, and the chain of cryptotexts is on the form

$$((u_0, v_0), \ldots, (u_{\delta-1}, v_{\delta-1})) = (E_{y_{\alpha_0}}(y_{\alpha_1}), \ldots, E_{y_{\alpha_{\delta-1}}}(y_{\alpha_\delta})) \ .$$

To achieve chosen message security we employ the Fiat-Shamir heuristic to turn an identification scheme into a signature scheme. The secret key of a signer S_α is a Cramer-Shoup [15] signature $\sigma_\alpha = \mathsf{Sig}^{\mathsf{cs}}(y_{\alpha_1}, \ldots, y_{\alpha_{\delta-1}})$ of the public keys corresponding to the path $\alpha_0, \alpha_1, \ldots, \alpha_\delta$ from the root $\rho = \alpha_0$ to the leaf $\alpha = \alpha_\delta$. The Cramer-Shoup scheme is provably secure under the strong RSA assumption.

To form a signature of a message m the signer first computes a commitment $C(\sigma_\alpha)$ of the signature σ_α. Then it computes an honest verifier zero-knowledge public coin proof $\pi(m)$ that the cryptotexts $((u_0, v_0), \ldots, (u_{\delta-1}, v_{\delta-1}))$ form a chain and that $C(\sigma_\alpha)$ hides a signature of the list $(y_{\alpha_1}, \ldots, y_{\alpha_{\delta-1}})$ of public keys used to form the chain of cryptotexts. The proof is given in the random oracle model and the message m to be signed is given as a prefix to every query to the random oracle. Thus, the complete signature is given by

$$(E_{y_{\alpha_0}}(y_{\alpha_1}), \ldots, E_{y_{\alpha_{\delta-1}}}(y_{\alpha_\delta}), C(\sigma_\alpha), \pi(m)) \ .$$

Intuitively, this means that if a signer S_α can produce a valid signature, we can by rewinding extract a signature of the list of public keys corresponding to the path from the root to the signer. Thus, a signature can only be formed if the signer is legitimate and if it has formed the chain correctly. Denote the hierarchical group signature scheme sketched above by $\mathcal{HGS}$. We prove the following theorem in the full version [26].

Theorem 2. *The hierarchical signature scheme $\mathcal{HGS}$ is secure under the DDH assumption and the strong RSA assumption in the random oracle model.*

Efficiency Analysis. The complexity of the protocol is largely determined by the proof sketched in the next subsection. This protocol has soundness $1 - O(\delta 2^{-\kappa'})$, where κ' is a secondary security parameter. Using standard computational tricks we estimate the complexity of the protocol to correspond to roughly $\kappa'(\delta + 3)$ general exponentiations modulo a κ-bit integer. If we set $\delta = 3$ and $\kappa' = 160$ this corresponds to less than 1000 general exponentiations. The size of a signature is about 1 Mb. The full version [26] contains a more detailed analysis.

Construction of the Proof of Knowledge. The main obstacle to find an efficient hierarchical group signature scheme following our approach is how to prove efficiently that $C(\sigma_\alpha)$ is a commitment of a signature σ_α of the list of public keys $(y_{\alpha_1},\ldots,y_{\alpha_{\delta-1}})$ used to form the chain $((u_0,v_0),\ldots,(u_{\delta-1},v_{\delta-1}))$. We construct a reasonably practical honest verifier zero-knowledge public coin proof for this relation by carefully selecting and combining a variety of cryptographic primitives and techniques. Due to the complexity of this protocol we can only sketch the main ideas of this protocol in this extended abstract. Details are given in the full version [26].

Let $q_0,\ldots,q_3$ be primes such that $q_i = 2q_{i+1}+1$ for $i = 0,1,2$. A list of such primes is called a Cunningham chain and exists under mild assumptions on the distribution of primes. There is a subgroup $G_{q_{i+1}} \subset \mathbb{Z}_{q_i}^*$ of order q_{i+1} for $i = 0,1,2$. Denote by g_i and y_i fixed and independently chosen generators of G_{q_i} for $i = 1,2,3$, i.e., $\log_{g_i} y_i$ is not known to any party in the protocol. Thus, we can form a commitment of a value $y_\alpha \in G_{q_3}$ in three ways, as

$$(y_3^{t'''} g_3^{s'''}, y_3^{s'''} y_\alpha) \ , \quad (y_2^{t''} g_2^{s''}, y_2^{s''} g_2^{y_\alpha}) \ , \ \text{ and } \quad (y_1^{t'} g_1^{s'}, y_1^{s'} g_1^{g_2^{y_\alpha}}) \ ,$$

where $t''', s''' \in \mathbb{Z}_{q_3}$, $t'', s'' \in \mathbb{Z}_{q_2}$, and $t', s' \in \mathbb{Z}_{q_1}$ are randomly chosen. By extending the ideas of Stadler [25] we can give a reasonably practical cut-and-choose proof that the elements hidden in two such commitments are identical.

Recall that the collision-free Chaum-Heijst-Pfitzmann [12] hash function is defined by $H^{\mathsf{CHP}} : \mathbb{Z}_{q_2}^\delta \to G_{q_2}$, $H^{\mathsf{CHP}} : (z_1,\ldots,z_\delta) \mapsto \prod_{l=1}^{\delta} h_l^{z_l}$, where the bases $h_1,\ldots,h_\delta \in G_{q_2}$ are randomly chosen, i.e., no party knows a non-trivial representation of $1 \in G_{q_2}$ in these elements.

We employ ElGamal over G_{q_3}. This means that the public keys $y_{\alpha_1},\ldots,y_{\alpha_\delta}$ belong to G_{q_3}. Although it is not trivial, the reader should not find it too hard to imagine that Stadler-techniques can be used to prove that the public keys used for encryption are identical to values hidden in a list of commitments formed as

$$((\mu_0,\nu_0),\ldots,(\mu_{\delta-1},\nu_{\delta-1})) = ((y_2^{t_0''} g_2^{s_0''}, y_2^{s_0''} h_1^{y_{\alpha_1}}),\ldots,(y_2^{t_{\delta-1}''} g_2^{s_{\delta-1}''}, y_2^{s_{\delta-1}''} h_\delta^{y_{\alpha_\delta}})) \ .$$

The importance of this is that if we take the product of the commitments we get a commitment of $H^{\mathsf{CHP}}(y_{\alpha_1},\ldots,y_{\alpha_\delta})$, i.e.,

$$\left(\prod_{i=0}^{\delta-1}\mu_i, \prod_{i=0}^{\delta-1}\nu_i\right) = \left(y_2^{t''} g_2^{s''}, y_2^{s''} \prod_{i=1}^{\delta} h_i^{y_{\alpha_i}}\right) \ , \tag{1}$$

for some $t'', s'' \in \mathbb{Z}_{q_2}$. Thus, at this point we have devised a way for the signer to verifiably commit to the hash value of the keys it used to form the chain of cryptotexts. This is a key step in the construction.

Recall that the signer commits to a Cramer-Shoup signature σ_α of the list of public keys it uses to form the chain of cryptotexts. This signature scheme uses an RSA-modulus N and elements from the subgroup QR_N of squares in $\mathbb{Z}_N^*$, and it is parameterized by two collision-free hash functions. The first hash function is used to compute a message digest of the message to be signed, i.e., the list $(y_{\alpha_1}, \ldots, y_{\alpha_\delta})$ of public keys. Above we have sketched how the signer can verifiably form a commitment of the H^{CHP} hash value of this message, so it is only natural that we let this be the first of the two hash functions in the signature scheme. However, in the signature scheme the message digest lives in the exponent of an element in QR_N in the signature scheme. To move the hash value up in the exponent and to change group from G_{q_1} to QR_N, the signer forms two commitments

$$\left(y_1^{t'} g_1^{s'}, y_1^{s'} g_1^{H^{\mathsf{CHP}}(y_{\alpha_1},\ldots,y_{\alpha_\delta})}\right) \text{ and } \mathbf{h}^t \mathbf{g}^{H^{\mathsf{CHP}}(y_{\alpha_1},\ldots,y_{\alpha_\delta})} .$$

Then it gives a cut-and-choose proof that the exponent in the left commitment equals the value committed to in the product (1). It also proves that the exponents in the two commitments are equal. Thus, at this point the signer has proved that it holds a commitment over QR_N of the hash value of the public keys it used to form the chain of cryptotexts.

The second hash function used in the Cramer-Shoup signature scheme is applied to a single element in QR_N. Since H^{CHP} is not collision-free on such inputs, we use the Shamir hash function $H^{\mathsf{Sh}}_{(\mathbf{g},N)} : \mathbb{Z} \to \mathrm{QR}_N$, $x \mapsto \mathbf{g}^x \bmod N$ instead. Using similar techniques as explained above the signer evaluates the hash function and moves the result into the exponent, by two Stadler-like cut-and-choose proofs.

Given the two hash values in the exponents of two commitments, standard techniques can be used to prove that the commitment $C(\sigma_\alpha)$ is a commitment of the Cramer-Shoup signature σ_α of the list of public keys used to form the chain of cryptotexts.

4 Conclusion

We have introduced and formalized the notion of hierarchical group signatures and given two constructions. The first is provably secure under general assumptions, whereas the second is provably secure under the DDH assumption and the strong RSA assumption in the random oracle model. The latter is practical, i.e., it can be implemented and run on modern workstations, bit it is still slow. Both require a trusted key generator. Thus, an interesting open problem is to eliminate these deficiencies.

Acknowledgments

We thank Johan Håstad for excellent advice. We also thank Torsten Ekedahl for explaining the heuristic view on the existence of Cunningham chains. We thank Ronald Cramer for providing us with references. We had valuable discussions on anonymity of cryptosystems with Gustav Hast.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO 2000*, volume 1880 of *LNCS*, 2000.
2. G. Ateniese and G. Tsudik. Some open issues and directions in group signatures. In *Financial Cryptography '99*, volume 1648 of *LNCS*, 1999.
3. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *ASIACRYPT 2001*, volume 2248 of *LNCS*, 2001.
4. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT 2003*, volume 2656 of *LNCS*, 2003.
5. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *RSA-CT 2005*, volume 3376 of *LNCS*, 2005.
6. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO 2004*, volume 3152 of *LNCS*, 2004.
7. J. Camenisch. Efficient and generalized group signatures. In *EUROCRYPT'97*, volume 1233 of *LNCS*, 1997.
8. J. Camenisch and J. Groth. Group signatures: Better efficiency and new theoretical aspects. In *SCN 2004*, volume 3352 of *LNCS*, 2005.
9. J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *CRYPTO 2004*, volume 3152 of *LNCS*, 2004.
10. J. Camenisch and M. Michels. A group signature scheme with improved effiency. In *ASIACRYPT'98*, volume 1514 of *LNCS*, 1999.
11. J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *CRYPTO'97*, volume 1294 of *LNCS*, 1997.
12. D. Chaum, E. van Heijst, and B. Pfitzmann. Cryptographically strong undeniable signatures, unconditionally secure for the signer. In *CRYPTO'91*, volume 576 of *LNCS*, 1991.
13. D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT'91*, volume 547 of *LNCS*, 1991.
14. L. Chen and T.P. Pedersen. New group signature schemes. In *EUROCRYPT'94*, volume 950 of *LNCS*, 1994.
15. R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. In *6th CCS*, 1999.
16. U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero-knowledge proofs under general assumptions. *SIAM Journal of Computing*, 29(1), 1999.
17. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2), 1984.
18. S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing*, 17(2), 1988.
19. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable signatures. In *EUROCRYPT 2004*, volume 3027 of *LNCS*, 2004.
20. A. Kiayias and M. Yung. Group signatures: Provable security, efficient constructions and anonymity from trapdoor-holders. Cryptology ePrint Archive, Report 2004/076, 2004. `http://eprint.iacr.org/2004/076`.
21. S. Kim, S. Park, and D. Won. Group signatures for hierarchical multigroups. In *ISW'97*, volume 1396 of *LNCS*, 1998.
22. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd STOC*, 1990.

23. A. Sahai. Non-malleable non-interactive zero-knowledge and adaptive chosen-ciphertext security. In *40th FOCS*, 1999.
24. A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. Robust non-interactive zero knowledge. In *CRYPTO 2001*, volume 2139 of *LNCS*, 2001.
25. M. Stadler. Publicly verifiable secret sharing. In *EUROCRYPT'96*, volume 1070 of *LNCS*, 1996.
26. M. Trolin and D. Wikström. Hierarchical group signatures. Cryptology ePrint Archive, Report 2004/076, 2004. `http://eprint.iacr.org/2004/311`.