Practical Forward Secure Group Signature Schemes

Dawn Xiaodong Song*
University of California, Berkeley
dawnsong@cs.berkeley.edu

ABSTRACT

A group signature scheme allows a group member to sign messages anonymously on behalf of the group, while in case of a dispute, a designated entity can reveal the identity of a signature's originator. Group signature schemes can be used as a basic building block for many security applications such as electronic banking systems and electronic voting. Two important issues – forward security and efficient revocation – have not been addressed by prior schemes. We construct the first *forward-secure* group signature schemes. While satisfying all the security properties proposed in previous group signature schemes, our schemes provide a new desired security property, *forward-security*: while the group public key stays fixed, a group signing key of a group member evolves over time such that compromise of a group signing key of the current time period does not enable an attacker to forge group signatures pertaining to the past time periods. Such forward-security is important to mitigate the damage caused by key exposure and particularly desirable for group signature schemes because the risk of signing key exposure escalates as the size of the group increases. Our schemes are provably secure in the random oracle model and under the strong RSA and decisional Diffie Hellman assumptions.

Furthermore, we extend our forward-secure group signature scheme to provide a solution for the problem of group member exclusion without the need to re-key all other group members. When a group member is excluded, he should not be able to generate valid signatures any more and yet his previous signatures remain anonymous. We provide the first solutions which support both *retroactive public revocation* and *backward unlinkability* and the signature size is independent of the number of revoked members.

*We gratefully acknowledge funding support for this research. This research was sponsored in part by the United States Defense Advanced Research Projects Agency (contract N66001-99-2-8913), and by the United States National Science Foundation (grant FD99-79852). DARPA Contract N66001-99-2-8913 is under the supervision of the Space and Naval Warfare Systems Center, San Diego. This paper represents the opinions of the authors and do not necessarily represent the opinions or policies, either expressed or implied, of the United States government, of DARPA, NSF, or any of its agencies.

CCS'01, November 5-8, 2001, Philadelphia, Pennsylvania, USA.

1. INTRODUCTION

Group signature schemes are an important building block for many security applications. In contrast to ordinary signature schemes where there is only one signer, group signature schemes allow any member of a group of signers to sign documents on behalf of the group. In general, a *group manager* controls the group membership and issues *group signing keys* to group members. The group signing keys allow a group member to sign documents on behalf of the group. In particular, a group signature scheme provides anonymity and unlinkability to the signer, i.e. everybody can verify that the signature is valid on behalf of a group, but nobody except for the group manager can identify the signing member. Furthermore it is computationally hard for anybody but the group manager to decide whether two different valid signatures were generated by the same group member. These attractive security properties make group signature schemes appealing to applications such as electronic voting, electronic auctions and many applications where it is desirable to hide organizational structure. Group signature schemes are also used in electronic cash systems to conceal the cash-issuing banks' identities [27] and identity escrow systems [25].

The concept of group signatures was first proposed by Chaum and van Heyst [16]. Several different group signature schemes have been proposed [16, 18, 13, 11, 27, 12, 3]. The most recent scheme by Ateniese et al. is particularly efficient and provably secure [3]. Unfortunately several limitations still render all previous solutions unsatisfactory in practice. One important problem is how to deal with exposure of group signing keys. Another important problem is how to allow efficient exclusion of group members. We discuss these two problems in more detail in the rest of this section and introduce our approaches to address these problems.

1.1 Exposure of Group Signing Keys and Forward Secure Group Signatures

Exposure of secret keys for "non-cryptographic" reasons, such as a compromise of the underlying storage system or human errors, is one of the greatest threats to many cryptographic protocols in practice. In group signature schemes, if a group member's group signing key is exposed to an at-

tacker, the attacker can then sign any documents on behalf of the group. And the danger of the exposure of signing keys escalates as the group size increases. In prior group signature schemes, the exposure of one group member's group signing key not only requires changing the group public key and signing keys, but also renders all previously obtained group signatures invalid, because one cannot distinguish whether a signature is generated by an attacker after it obtained the group signing key or by the legitimate group member before the attacker obtained the group signing key.

We propose to use the concept of *forward security* to reduce the damage of exposure of a group signing key, i.e. even when a group signing key is exposed, previously generated group signatures remain valid and do not need to be re-signed. In a *forward secure group signature* scheme, the group signing keys evolve over time. We divide the time that a group public key is desired to be valid into T periods. Assume the group public key is GPK, and a group member A obtained an initial group signing key SK_0 when he initially joins the group. While the group public key stays fixed, A's group signing key evolves over time: in time period i, A's group signing key evolves from SK_{i-1} to SK_i using a public one-way function. After time period i, SK_i is erased from the system. And a signature on a message m is a pair $\langle i, s\rangle$ where i represents the time period indicated in the signature. The verification procedure ensures that a signature $\langle i, s\rangle$ is valid for message m only if s was produced using the group signing key for time period i.

Here is an example to illustrate how such a forward-secure group signature scheme can reduce the damage of the exposure of a group signing key. Assume an attacker breaks into a group member's system in time period j and obtains the member's group signing key SK_j. Because of the one-way property, the attacker cannot compute the group signing keys corresponding to previous time periods. Assume a user B obtained a group signature on a document m prior to time period j. B expects to be able to use the signed document m for a long time (long after time period j). When it is discovered that SK_j has been exposed, the group public key is revoked. If the group signature scheme were without the forward-secure property, obviously the group signature B obtained on document m would be rendered invalid and B would need to obtain a new signature on document m. But when the group signature scheme is constructed as a forward-secure scheme, the attacker cannot compute the group signing keys corresponding to previous time periods, and hence the group only revokes the group public key for any period following the time period j. Thus any valid signatures with corresponding time period before j is still accepted. Because in the signature B obtained $\langle t, s\rangle$, $t < j$. Hence the signature $\langle t, s\rangle$ is still a valid signature on m and B would not need to obtain a new signature on m.

The concept of forward secure signatures was first proposed by Ross Anderson [2] for traditional (non-group) signatures. The challenge is to design an efficient scheme. In particular the size of the secret key, public key and signatures should not be dependent on the number of time periods during the lifetime of the public key. Several schemes have recently been proposed for traditional (non-group) signatures and threshold signatures that satisfy this efficiency property [6, 1, 26, 28, 24].

Previous group signature schemes do not provide forward security. We propose the first forward secure group signature schemes. In particular we extend the group signature scheme proposed by Ateniese et al. [3] in two different ways to construct two different forward secure group signature schemes. The first scheme leverages techniques used in [6] and the second scheme employs a new technique to achieve forward security which is also proposed in [24]. In addition, we show that forward secure group signature schemes enables other desirable security properties at little extra cost. For example, when a new group member C joins, with forward secure group signature schemes, we can restrict the new member to generate signatures that are only valid for future time periods and not before his joining period with no extra overhead. More generally, we can support at little extra cost the property of *time-limited group membership*, i.e. a group member is only allowed to sign on behalf of the group during a limited time.

1.2 Retroactively Publicly Revokable Group Membership with Backward Unlinkability

Any practical group signature scheme must support dynamic group membership. In practice, group members may join, leave, or be excluded from the group during at any time. Previous group signature schemes can support group member joins efficiently, but not group member exclusion events. Imagine the scenario where an attacker stole a group member A's group signing key during time period i. The leakage of A's group signing key is only discovered later in time period j. The group manager revokes A's group signing key in time period j. Therefore nobody should be able to generate group signatures valid for time periods after j using A's group signing key. We call this property *public revokability*. Furthermore, because the attacker could have signed documents on behalf of the group any time after time period i using A's group signing key, signatures generated using A's group signing key after time period i should become invalid. At the same time, signatures generated using other members' group signing keys should still remain valid and anonymous and unlinkable. We call this property *retroactive public revokability*. Note that retroactive public revokability implies public revokability, not vice versa. Moreover, signatures generated using A's group signing key before time period i should remain valid, anonymous and unlinkable because these signatures are generated by A not the attacker. We call this property *backward unlinkability*. Ideally a group signature scheme should support both retroactive public revokability and backward unlinkability.

A naive approach to achieve both public revokability and backward unlinkability is to use a forward secure group signature scheme and re-issue a new group public key and new group signing keys to legitimate group members whenever a new group member is expelled. Obviously this approach does not support retroactive public revokability and is impractical when the group is large or highly dynamic. In another approach the group manager issues a Certification Revocation List (CRL) when a group member is expelled. When a user obtains a group signature on a document for time period i, he checks against the CRL to see whether the signer has been expelled for time period i. The challenge for this approach is to design such a group signature scheme that enables CRL and still provides anonymity, unlinkability and backward unlinkability. Ateniese and Tsudik mentioned similar problems as open issues for group signa-

ture schemes [4]. Bresson and Stern [10] proposed a first solution providing public revokability for the group signature scheme in [13]. Unfortunately their approach does not provide retroactive public revokability and the signature size is linear to the number of revoked members. We extend our forward secure group signature schemes to provide the first solutions that support both retroactive public revokability and backward unlinkability and the signature size in our schemes is independent of the number of revoked members.

1.3 Our Contribution and Outline

We propose forward-secure group signature schemes. In particular we extend the group signature scheme proposed by Ateniese et al. [3] in two different ways to construct two different forward secure group signature schemes. The first scheme leverages techniques used in [6] and the second scheme employs a new technique to achieve forward security and is also proposed in [24]. Our schemes satisfy forward security as well as all the traditional security properties shared with previous group signature schemes. Our schemes are efficient in the sense that they are independent of the number of group members, and the size of signatures and group keys are independent of the number of time periods during the lifetime of the group public key. As a side benefit, our schemes can also be extended to support flexible time-limited group membership at little extra overhead.

Furthermore, we extend our forward-secure group signature schemes to support revocation. Our approaches are the first ones that support retroactive public revokable group membership with backward unlinkability and the signature size in our approach is independent of the number of revoked members.

The rest of the paper is organized as follows. We first introduce the model and formal security requirements in section 2, and the preliminaries and our notations in section 3. We then describe our forward secure group signature Scheme I and extensions in section 4 and the Scheme II in section 5. We conclude in section 7 and give security proofs the lemmas in the appendix.

2. THE MODEL AND SECURITY REQUIREMENTS

In a group signature scheme, a principal who can sign documents on behalf of the group is called a *group member*. A *group manager (GM)* controls the group membership and assigns group signing keys to group members which allow group members to sign documents on behalf of the group. A traditional group signature scheme consists of five procedures: SETUP, JOIN, SIGN, VERIFY, and OPEN. We add another procedure, EVOLVE, for the forward-secure group signature scheme. Below is a brief description of the six procedures in a forward-secure group signature scheme.

- SETUP: On input a security parameter ℓ, this probabilistic procedure outputs the system parameters, the group public key and the secret key for the group manager.
- JOIN: For a user to join the group, the group manager and the user execute this protocol interactively. The user receives a group signing key and becomes a new group member.
- EVOLVE: Given input of a group signing key for time period i, this procedure outputs the corresponding group signing key for time period $i+1$.
- SIGN: Given input of a group public key, a member's group signing key, a message m and a time period i, this probabilistic procedure outputs a signature $\langle i, s\rangle$ on message m.
- VERIFY: Given input of a group public key, a group signature $\langle i, s\rangle$ and a message m, this procedure verifies whether s is a valid group signature on m signed with a group signing key of time period i. If s is a valid group signature signed with a group signing key of time period i, we say $\langle i, s\rangle$ is a signature *valid for time period* i.
- OPEN: Given input of a message, a valid group signature on the message, a group public key and the group manager's secret key, this procedure determines the identity of the signer.

A traditional group signature scheme should satisfy the following properties:

- *Correctness*: Signatures produced by a group member using SIGN must be accepted by VERIFY.
- *Unforgeability*: Only group members are able to sign messages on behalf of the group.
- *Anonymity*: Given a valid signature of a message, it is computationally hard for everybody but the group manager to identify the actual signer.
- *Unlinkability*: It is computationally hard for everybody but the group manager to decide whether two different valid signatures were computed by the same group member.
- *Exculpability*: Neither a coalition of group members nor the group manager can generate signatures that will be opened by the OPEN procedure as generated from another group member. This means a group member cannot be blamed to have generated a signature that he actually did not generate.
- *Traceability*: A trusted entity can always open a valid signature using the OPEN procedure and identify the actual signer. This trusted entity can either be the group manager or some other entity, usually called the *revocation manager*. For simplicity we assume this trusted entity is the group manager in this case. If a separate entity is desired, the scheme can be easily adapted to support a separate revocation manager.

We define two degrees of forward security:

- *Weak Forward security*: Assume a set of group signing keys $\Phi = \{k_{i,t_i}\}_{1\le i\le L}$ where k_{i,t_i} represents the group signing key of member i for time period t_i, and $t = \min(t_1, \ldots, t_L)$. We call $\Omega(\Phi)$ is the *weak-span* of Φ where $\Omega(\Phi)$ represents the set of group signing keys $\{k_{i,w_i}\}_{1\le i\le L, t\le w_i\le T}$. We say the group signature scheme satisfies the weak forward security if an attacker given a set of group signing keys Φ cannot generate a valid group signing key not in $\Omega(\Phi)$.

- *Strong Forward security*: Given a set of group signing keys $\Phi = \{k_{i,t_i}\}_{1 \le i \le L}$ where k_{i,t_i} represents the group signing key of member i for time period t_i, we call $\Psi(\Phi)$ is the *span* of Φ where $\Psi(\Phi)$ represents the set of group signing keys $\{k_{i,w_i}\}_{1 \le i \le L, t_i \le w_i \le T}$. Then the group signature scheme satisfies the strong forward security if an attacker given a set of group signing keys Φ cannot generate a valid group signing key not in $\Psi(\Phi)$.

In this paper, we also discuss the following desired security properties:

- *Time-limited membership*: The group manager can limit a member's group membership by issuing him group signing keys which can only generate group signatures valid for some periods of time.

- *Retroactive public revokability and backward unlinkability*: At a time period i, the group manager can exclude a group signing key starting from time period j such that any signatures generated using this group signing key after time period j become invalid to any verifier. Moreover, all signatures generated using this group signing key before time period j should still remain anonymous and unlinkable to everybody but the group manager.

Finally the scheme should be efficient. In particular, the signature size and key lengths should be independent of the number of group members and the number of time periods during the lifetime of the public key. Furthermore, in the scheme to support public revokability, the signature size should be independent of the number of revoked members.

3. PRELIMINARIES

Our schemes rely on the strong RSA assumption [5, 22] and the decisional Diffie-Hellman assumption [20, 7]. Let $n = pq$ be an RSA-like modulus and let G be a cyclic subgroup of $\mathcal{Z}_n^*$. The strong RSA assumption is that given n and $z \in G$, it is computationally hard to find $v \in G$ and $e \in \mathcal{Z}_{>1}$ such that $z \equiv v^e (\mod n)$. Let $G = \langle g \rangle$ be a cyclic group generated by g. The decisional Diffie-Hellman assumption is that given g, g^x, g^y, and g^z, it is computationally hard to decide whether g^{xy} and g^z are equal.

We use several existing zero-knowledge proof protocols as building blocks in our scheme. These zero-knowledge protocols can be performed non-interactively using an ideal hash function (a.k.a. the Fiat-Shamir heuristics [21]) and we refer to the resulting constructs as *signatures of knowledge* [13]. Due to space limitation, we do not review the details of these protocols here. We give the references to these protocols and introduce the notations we use in the paper here. To simplify the representation, we use *PK* to represent that the protocol is a signature of knowledge protocol, Greek letters to denote the secret knowledge that is being proved, and all other parameters are known to the prover and the verifier.

- Signature of knowledge of the discrete logarithm:
  Let $G = \langle g \rangle$ denote a group of prime order q and $y \in G$. We use $\mathrm{PK}\{(\alpha) : y = g^\alpha\}(m)$ to denote the signature of knowledge of $\log_g y$ in group G. This protocol was designed by [29, 15] and shown to be zero-knowledge in the auxiliary string model [19].

- Signature of knowledge of the discrete logarithm in QR_n:
  Let $n = pq$, where $p = 2p' + 1, q = 2q' + 1$, and p, q, p' and q' are all primes. Let g be the generator of QR_n, and $y \in \mathrm{QR}_n$. We use $\mathrm{PK}\{(\alpha) : y = g^\alpha\}(m)$ to denote the signature of knowledge of $\log_g y$ in group QR_n [22]. In general, it is not easy to for the prover to prove that y is a quadratic residue. So we use the protocol $\mathrm{PK}\{(\alpha) : y^2 = (g^2)^\alpha\}$ instead since $\log_{g^2} y^2 = \log_g y$ in case y is a quadratic residue.

- Signature of knowledge of a representation:
  Let $\mathrm{PK}\{(\alpha_1, \ldots, \alpha_v) : y = g_1^{\alpha_1} \cdots g_v^{\alpha_v}\}(m)$ denote a signature of knowledge of a representation of an element $y \in G$ with respect to bases $g_1, \ldots, g_v \in G$ [15].

- Signature of knowledge of equality of discrete logarithm:
  Let $\mathrm{PK}\{(\alpha) : y_1 = g^\alpha \wedge y_2 = h^\alpha\}(m)$ denote a signature of knowledge of equality of discrete logarithms of two group elements $y_1, y_2 \in G$ to the bases $g \in G$ and $h \in G$ [14, 17].

- Signature of knowledge of ranges:
  Let $\mathrm{PK}\{(\alpha) : y = g^\alpha \wedge \alpha \in [a, b]\}(m)$ denote a signature of knowledge of a discrete logarithm of $y \in G$ with respect to $g \in G$ such that $\log_g y$ lies in the integer interval $[a, b]$. This protocol can be efficiently done under the strong RSA assumption and if the prover is not provided the factorization of the modulus [8].

4. FORWARD SECURE GROUP SIGNATURE I

4.1 The Scheme

SETUP Procedure. The group manager (GM) chooses two $(\ell_n/2)$-bit primes $p = 2p'+1$ and $q = 2q'+1$ where p' and q' are also primes. Set $n := pq$. *GM* also randomly chooses elements $a, d, g, g_1 \in_R \mathrm{QR}_n$, a secret element $x \in_R Z^*_{p'q'}$, and set $y := g^x \mod n$. It stores (p, q, x) as its secret key and publishes (n, a, d, g, g_1, y) as the group public key. It also divides the time during which the group public key is valid into T time period and makes the time intervals public.

In the rest of the paper, we also use the following notation. Let Γ and Λ denote integer intervals: $\Gamma = (-2^{\ell_\Gamma}, 2^{\ell_\Gamma}), \Lambda = (2^{\ell_\Lambda}, 2^{\ell_\Lambda + 1})$, where $\ell_\Lambda > T + \ell_\Gamma + 2$.

Intuition. The intuition of the scheme is as the following. When a user U joins the group, he and *GM* randomly select $x_u \in_R \Gamma$ together such that U knows x_u and *GM* only knows $y_u := a^{x_u}$. The group manager then randomly selects a prime $e_u \in \Lambda$, computes $c_{u,0} := (y_u d)^{1/(e_u 2^T)} \mod n$, and sends U $(c_{u,0}, e_u)$. Thus $(x_u, c_{u,0}, e_u)$ is U's group signing key. Under the strong RSA assumption, nobody can generate valid group signing keys except the group manager. The group members then evolve group signing keys using squaring as a public one-way function. In particular, U's group signing key for time period i is $(x_u, c_{u,i}, e_u)$ where $c_{u,i} = c_{u,i-1}^2 \mod n$. To sign a message m for time period i, U produces non-interactive proofs of knowledge that he knows $(x_u, c_{u,i}, e_u)$ such that $(c_{u_i}^{2^{T-i}})^{e_u} = da^{x_u}$, $x_u \in \Gamma$, and $e_u \in \Lambda$, where the challenge in the non-interactive proof

is dependent on m as in the standard Fiat-Shamir heuristics [21]. To enable the OPEN procedure, the signer also encrypts $c_{u,i}$ with the public key of the group manager and produces the non-interactive proof that the encryption is of the correct form. In case of a dispute, the group manager can simply decrypt the value of $c_{u,i}$ to identify the actual signer.

JOIN Procedure

1. When a user U joins the group, he first generates a secret $r_u \in_R \Gamma$, $r_1 \in_R \{0,1\}^{2\ell_n}$, and sends GM $s_1 := g^{r_u} g_1^{r_1}$. U proves to GM that s_1 is formed correctly: $\mathrm{PK}\{(\alpha,\beta) : s_1 = g^{\alpha} g_1^{\beta}\}$.

2. GM then randomly select $r_m \in_R \Gamma$ and send U r_m. U then computes $x_u = (r_u + r_m \mod (2^{\ell_\Gamma+1} - 1)) - 2^{\ell_\Gamma} + 1, y_u = a^{x_u}$. U then sends GM y_u.
   U computes $s_2 = \lfloor \frac{r_u + r_m}{2^{\ell_\Gamma+1}-1} \rfloor$ and selects $r_2 \in_R \{0,1\}^{2\ell_n}$, sets $s_3 := g^{s_2} g_1^{r_2}$, and sends s_3 to GM. Then U proves to GM that y_u is formed correctly:
   $\mathrm{PK}\{(\alpha,\beta,\gamma,\delta,\epsilon,\theta) : s_1 = g^{\alpha} g_1^{\beta} \wedge s_3 = g^{\gamma} g_1^{\delta} \wedge$
   $s_1 g^{r_m - 2^{\ell_\Gamma}+1} s_3^{-(2^{\ell_\Gamma+1}-1)} = g^{\epsilon} g_1 \theta \wedge y_u = a^{\epsilon}\}$.

3. GM then randomly selects a prime $e_u \in \Lambda$, computes $c_{u,0} := (y_u d)^{1/(e_u 2^T)} \mod n$, and sends U $(c_{u,0}, e_u)$. U verifies that $a^{x_u} d \equiv c_{u,0}^{e_u 2^T} \mod n$. $(c_{u,0}, e_u, x_u)$ is U's group signing key.

EVOLVE Procedure. Assume U has group signing key $(c_{u,j}, e_u, x_u)$ at time period j. Then at time period $j+1$, his group signing key becomes $(c_{u,j+1}, e_u, x_u)$, where $c_{u,j+1} := c_{u,j}^2 \mod n$.

SIGN and VERIFY Procedures. Assume U has group signing key $(c_{u,j}, e_u, x_u)$ at time period j. To sign a message m in time period j, he first chooses $r_1 \in_R \{0,1\}^{2\ell_n}$, computes $A = c_{u,j} y^{r_1}, B = g^{r_1}$, and generates
$\mathrm{PK}\{(\alpha,\beta,\delta,\epsilon) : d^2 = (A^{2^{T-j+1}})^{\alpha} (1/(a^2))^{\beta} (1/(y^2))^{\delta} \wedge$
$B^2 = (g^2)^{\epsilon} \wedge 1 = (B^{2^{T-j+1}})^{\alpha} (1/(g^2))^{\delta} \wedge \beta \in \Gamma \wedge \alpha \in \Lambda\}(m)$.
A verifier simply checks the validity of the above signature of knowledge.

OPEN Procedure. In the event that the actual signer must be subsequently identified (e.g., in case of a dispute), GM first checks the validity of the signature via the VERIFY procedure, and then recover $c_{u,0}$ (and thus the identity of U) as $c_{u,0} = (A/B^x)^{1/2^j} \mod n$. GM also proves that $\log_g y = \log_B (A/(c_{u,0}^{2^j}) \mod n)$.

4.2 Security Analysis

In this subsection we show that Scheme I is a secure group signature scheme and satisfies weak forward security. We state our theorems here and would like to refer the reader to the appendix A for detailed proofs.

LEMMA 1. *$PK\{(\alpha,\beta,\delta,\epsilon) : B^2 = (g^2)^{\epsilon} \wedge$ $d^2 = (A^{2^{T-j+1}})^{\alpha} (1/(a^2))^{\beta} (1/(y^2))^{\delta} \wedge 1 = (B^{2^{T-j+1}})^{\alpha} (1/(g^2))^{\delta}$ $\wedge \beta \in \Gamma \wedge \alpha \in \Lambda\}(m)$ is a statistical zero-knowledge proof of knowledge of the group membership key.*

LEMMA 2. *In Scheme I, under the strong RSA assumption, a group signing key for some time period t, (x_u, e_u, c_u) where $x_u \in \Gamma$, $e_u \in \Lambda$, and $c_u^{2^{T-t} e_u} = d a_{x_u} \mod n$, can only be generated by the group manager given that the number of group signing keys the group manager issues is polynomially bounded.*

LEMMA 3. *Scheme I satisfies weak forward security under the strong RSA assumption and given that the number of group signing keys the group manager issues is polynomially bounded.*

COROLLARY 1. *In the random oracle model, Scheme I is secure and satisfies the weak forward security under the strong RSA assumption and the decisional Diffie-Hellman assumption.*

4.3 Time-Limited Group Membership

When a group member U joins at time period i, if he only obtains the group signing key for that time period $(c_{u,i}, e_u, x_u)$, then he can only generate group signatures valid for time periods following time period i and not for the time periods prior to time period i. Thus his group membership is only valid for the time after he joined.

We can easily extend Scheme I to support a more general form of *time-limited group membership*, i.e. a group member is only allowed to sign on behalf of the group during a limited time. The basic idea is to have two chains of the group signing key. One chain evolves forward over time (as in Scheme I) and the other chain backwards. Note that the two chains share the same x_u. A signer needs to know both the signing key in the forward chain and the backward chain for time period i to generate a signature valid for time period i. Knowing the signing key for time period i in the forward chain enables the signer to compute all the keys in the forward chain after time period i, and knowing the signing key for time period i in the backward chain enables the signer to compute all the keys in the backward chain prior to time period i. Therefore knowing the signing key for time period i in the forward chain and the signing key for time period j in the backward chain allow the member to sign documents between time period i and j, given $i < j$. And the expence of two chains is not much, i.e. the signature size is less than double of the basic one-chain signature size.

4.4 Revocation

We can extend Scheme I to support retroactive revocation with backward unlinkability. Let h be a generator in a cyclic group of order n where the decisional Diffie Hellman problem is hard [9, 7]. When a group member U signs a message in time period j with his group signing key $(c_{u,j}, e_u, x_u)$, besides the basic SIGN procedure, U also does the following. He randomly selects $r \in \mathcal{Z}_n$ and compute $g_2 = h^r$, $C = g_2^{c_{u,j}}$, and reveals (g_2, C) which we call a *revocation token*. U proves through signature of knowledge that g_2 and C are of the right form. When a user V is expelled from the group starting from time period i, $c_{v,i}$ and i will be published in the CRL. Assume a verifier has a signature for time period j where $j \geq i$ and the revocation token in the signature is (g_2', C'). To see whether the signing key has been expelled, the verifier simply compute $c_{v,j}$ and check whether $C' = (g_2')^{c_{v,j}}$. If they equal, it means that the signature is revoked. In more detail, for U to sign a message m in time period j, the protocol is as the following:

U chooses $r_1 \in_R \{0,1\}^{2\ell_n}, r_2 \in_R \mathcal{Z}_n$, computes $A = c_{u,j}y^{r_1}, B = g^{r_1}, g_2 = h^{r_2}, C = g_2^{c_{u,j}}$, and generates
$\mathrm{PK}\{(\alpha,\beta,\delta,\epsilon,\gamma) : g_2 = h^\gamma \wedge B^2 = (g^2)^\epsilon \wedge$
$d^2 = (A^{2^{T-j+1}})^\alpha (1/(a^2))^\beta (1/(y^2))^\delta \wedge 1 = (B^{2^{T-j+1}})^\alpha (1/(g^2))^\delta$
$\wedge \beta \in \Gamma \wedge \alpha \in \Lambda\}(m)$,
and $\mathrm{PK2}\{(\eta,\theta) : A = \eta y^\theta \wedge B = g^\theta \wedge C = g_2^\eta\}(m)$.

The first protocol PK is the same as in our original forward-secure group signature scheme. The second protocol PK2 is a new zero-knowledge proof protocol that we construct and we explain the details in the appendix B.

It is clear that this scheme supports publicly-revokable group membership. However, the backward unlinkability property relies on a new cryptographic assumption which we call *log-square assumption*: Let n be a product of two safe primes as in Scheme I. Let G be a cyclic group of order n where the decisional Diffie Hellman problem is hard, and let g_2 be a generator of G. Given a random $v \in_R \mathrm{QR}_n$, and $w = g_2^u \in_R G$, it is computationally hard for an attacker to decide whether $v = u^2 \mod n$ without knowing the factor of n. Under this assumption, it is easy to see that our scheme supports backward unlinkability. The cryptographic assumption reduces to either factoring or discrete logarithm problem. We conjecture this problem is hard although more research still needs to be done to study this problem.

5. FORWARD SECURE GROUP SIGNATURE II

5.1 The Scheme

SETUP Procedure. The SETUP procedure is almost the same as in Scheme I. In addition to the SETUP procedure in Scheme I, let Λ_i denote integer interval $(2^{\ell_\Lambda}(1+i2^{\ell_\Lambda}/(T+1)), 2^{\ell_\Lambda}(1+(i+1)2^{\ell_\Lambda}/(T+1)))$, for $0 \le i \le T$. The group manager also specifies a deterministic one-way method such that given a randomly chosen prime $e_{u,i} \in \Lambda_i$, one can generate a sequence of prime numbers $e_{u,i}, \ldots, e_{u,T}$ where $e_{u,j} \in \Lambda_j$, for $i \le j \le T$. Given only $e_{u,j}$ for some $i \le j \le T$, it is hard to compute backwards, i.e. $e_{u,k}$ for $k < j$.

Intuition. This forward secure group signature scheme is similar to Scheme I in section 4 except that it uses a different one-way function to evolve the group signing keys. When a user U joins the group, similarly to Scheme I, he and GM randomly select $x_u \in_R \Gamma$ together such that U knows x_u and GM only knows $y_u := a^{x_u}$. The group manager then randomly selects a prime $e_{u,0} \in \Lambda_0$ which then generates a sequence of prime numbers $e_{u,i} \in \Lambda_i, 0 \le i \le T$. Let $b_u = \prod_{0 \le i \le T} e_{u,i}$. The group manager then computes $f_u := (y_u d)^{1/(b_u)} \mod n$, and sends U $(f_u, e_{u,0})$. Thus from $(x_u, f_u, e_{u,0})$, U can derive his group signing key for each time period. In particular, let $v_0 = f_u, v_{i+1} = v_i^{e_{u,i}}, w_0 = \prod_{1 \le i \le T} e_{u,i}, w_{i+1} = w_i / e_{u,i+1}$. U's group signing key for time period i is $(x_u, c_{u,i}, e_{u,i})$ where $c_{u,i} = v_i^{w_i} \mod n$. To sign a message m for time period i, U produces non-interactive proofs of knowledge that he knows $(x_u, c_{u,i}, e_{u,i})$ such that $c_{u,i}^{e_{u,i}} = da^{x_u}$, $x_u \in \Gamma$, and $e_{u,i} \in \Lambda_i$. To enable the OPEN procedure, the signer also encrypts $c_{u,i}$ with the public key of the group manager and produces the non-interactive proof that the encryption is of the correct form. In case of a dispute, the group manager can simply decrypt the value of $c_{u,i}$ to identify the actual signer.

JOIN Procedure. The JOIN procedure is the same as in Scheme I except for the third step. In the third step, the group manager randomly selects a prime $e_{u,0} \in \Lambda_0$ which then generates a sequence of prime numbers $e_{u,i} \in \Lambda_i, 0 \le i \le T$. Let $b_u = \prod_{0 \le i \le T} e_{u,i}$. The group manager then computes $f_u := (y_u d)^{1/(b_u)} \mod n$, and sends U $(f_u, e_{u,0})$. U computes b_u from the given $e_{u,0}$ and verifies that $a^{x_u} d = f_u^{b_u} \mod n$.

EVOLVE Procedure. Let $v_0 = f_u, w_0 = \prod_{1 \le i \le T} e_{u,i}$. In each time period i, U stores $(v_i, e_{u,i})$. To evolve in time period $i+1$, U computes $e_{u,i+1}, \ldots, e_{u,T}$ from $e_{u,i}$ and computes $v_{i+1} = v_i^{e_{u,i}}, w_{i+1} = e_{u,i+2} \cdots e_{u,T}$. Thus U's group signing key for time period $i+1$ is $(x_u, c_{u,i+1}, e_{u,i+1})$ where $c_{u,i+1} = v_{i+1}^{w_{i+1}} \mod n$. And U update the storage $(v_i, e_{u,i})$ to $(v_{i+1}, e_{u,i+1})$.

SIGN and VERIFY Procedures. Assume U has group signing key $(c_{u,j}, e_{u,j}, x_u)$ at time period j. To sign a message m in time period j, he first chooses $r_1 \in_R \{0,1\}^{2\ell_n}$, computes $A = c_{u,j}y^{r_1}, B = g^{r_1}$, and generates
$\mathrm{PK}\{(\alpha,\beta,\delta,\epsilon) : d^2 = (A^2)^\alpha (1/(a^2))^\beta (1/(y^2))^\delta \wedge$
$B^2 = (g^2)^\epsilon \wedge 1 = (B^2)^\alpha (1/(g^2))^\delta \wedge \beta \in \Gamma \wedge \alpha \in \Lambda_j\}(m)$.

A verifier simply checks the validity of the above signature of knowledge.

OPEN Procedure. In the event that the actual signer must be subsequently identified (e.g., in case of a dispute), GM first checks the validity of the signature via the VERIFY procedure, and then recover $c_{u,j}$ (and thus the identity of U) as $c_{u,j} = (A/B^x) \mod n$. GM also proves that $\log_g y = \log_B (A/(c_{u,j}) \mod n)$.

5.2 Security Analysis

We show that Scheme II is a secure group signature scheme and satisfies strong forward security. We state our theorems here and would like to refer the reader to the appendix A for detailed proofs.

LEMMA 4. *$PK\{(\alpha,\beta,\delta,\epsilon) : d^2 = (A^2)^\alpha (1/(a^2))^\beta (1/(y^2))^\delta \wedge B^2 = (g^2)^\epsilon \wedge 1 = (B^2)^\alpha (1/(g^2))^\delta \wedge \beta \in \Gamma \wedge \alpha \in \Lambda_i\}(m)$ is a statistical zero-knowledge proof of knowledge of the group membership key.*

LEMMA 5. *In Scheme II, under the strong RSA assumption, a group signing key for some time period t, $(x_u, e_{u,t}, c_{u,t})$, where $c_{u,t}^{e_{u,t}} = da_{x_u} \mod n$, $x_u \in \Gamma$, and $e_u \in \Lambda_t$, can only be generated by the group manager given that the number of group signing keys the group manager issues is polynomially bounded.*

LEMMA 6. *Under the strong RSA assumption, Scheme II satisfies strong forward security given that the number of group signing keys the group manager issues is polynomially bounded.*

COROLLARY 2. *In the random oracle model, Scheme II is secure and satisfies the strong forward security under the strong RSA assumption and the decisional Diffie-Hellman assumption.*

5.3 Time-limited Group Membership

Using Scheme II, we can in fact support group membership valid for any subset of the time periods with almost

no overhead. For example, when user U joins the group, if GM only wants to issue U group membership valid for time periods $t_1, \ldots, t_m$, then in step 3 instead of computing b_u as the product of the whole sequence $e_{u,0}, \ldots, e_{u,T}$, GM computes $b_u = \prod_{1 \le i \le m} e_{u,t_i}$ and then compute $f_u = (y_u d)^{1/b_u} \bmod n$. Thus U can only sign on behalf the group in time periods $t_1, \ldots, t_m$ with no extra overhead in secret storage and computation time comparing to the basic Scheme II.

5.4 Revocation

Revocation in Scheme II is easy. When U signs with his group membership $(x_u, e_{u,i}, c_{u,i})$ in time period i, besides the basic SIGN procedure in Scheme II, he also does the following: he randomly generates an element $g_3 \in QR_n$ and reveals g_3 and $D = g_3^{e_{u,i}}$ and proves in zero-knowledge that D is formed correctly. We call (g_3, D) the *revocation token* of the signature. More precisely, the new SIGN procedure is as the following: U chooses $r_1 \in_R \{0,1\}^{2\ell_n}, g_3 \in_R QR_n$, computes $A = c_{u,i} y^{r_1}, B = g^{r_1}, D = g_3^{e_{u,i}}$, and generates
$\mathrm{PK}\{(\alpha, \beta, \delta, \epsilon) : B^2 = (g^2)^{\epsilon} \wedge 1 = (B^2)^{\alpha}(1/(g^2))^{\delta} \wedge$
$D^2 = (g_3^2)^{\alpha} \wedge d^2 = (A^2)^{\alpha}(1/(a^2))^{\beta}(1/(y^2))^{\delta} \wedge \beta \in \Gamma$
$\wedge \alpha \in \Lambda_i\}(m)$.

A verifier simply checks the validity of the above signature of knowledge.

Now if the group manager wants to revoke U's signature starting from time period i, he simply reveals $e_{u,i}$ and i. If a user V obtained a signature for time period $j \ge i$ and the revocation token of the signature is (g', D'), to check whether the signature is revoked, (i.e. it was signed by U), V computes $e_{u,j}$ from $e_{u,i}$ and simply checks whether $D' = (g')^{e_{u,j}}$. If they equal, it means the signature is revoked because it is signed by U after time period i.

This revocation scheme supports retroactive public revocation. It is easy to see that all signatures signed by other members stay anonymous and unlinkable. This scheme also supports backward unlinkability for the revoked member with the assumption that DDH is hard, because without knowing the exponents, given two revocation token in two different signatures (g', D') and (g'', D''), if one can tell whether $\log_{g'} D' = \log_{g''} D''$, then he can solve the DDH problem.

Moreover, with $O(logT)$ storage overhead, we can extend Scheme II to allow a *time-limited* revocation, i.e. to revoke a group member's signatures only during some periods of time. The basic idea is that instead of generating the sequence $e_{u,0}, \ldots, e_{u,T}$ using a one-way chain, we can generate the sequence using a top-down one-way tree. The squence $e_{u,0}, \ldots, e_{u,T}$ forms the leaf nodes in the top-down one-way tree. Knowing the root, one can generate the entire tree; and knowing an internal node, one can generate the entire subtree but not any other nodes in the tree. Thus, when the group manager only wants to revoke a group member U's signature from time period t_1 to time period t_2, he reveals the internal nodes that serve as the roots of the subtrees that cover the time periods from t_1 to t_2. Thus any user can compute $e_{u,i}$ for $t_1 \le i \le t_2$ and therefore can check whether a signature is revoked.

6. DISCUSSION

Our two forward secure group signature schemes are both based on the group signature scheme proposed in [3] and use different one-way function for evolving the group signing keys to achieve forward security. The two schemes have different performance and security tradeoffs. Scheme I has efficient EVOLVE procedure (only requiring a squaring), although the signing and verification are less efficient (requiring $O(T)$ squaring). Scheme II has efficient signing and verification procedure and has no extra overhead comparing to the non-forward-secure group signature scheme in [3]. But Scheme II has less efficient EVOLVE procedure (requiring $O(T)$ exponentiations), although similar techniques as in [24] can be used to reduce the overhead with a storage space tradeoff. Scheme II can also achieve a more flexible time-limited group membership than Scheme I. Scheme II supports strong forward security and Scheme I can only be proven satisfying weak forward security. Scheme II has a much more efficient revocation mechanism than Scheme I and requires no new cryptographic assumptions for security while Scheme I requires a new assumption for revocation with backward unlinkability. Note that both two schemes also satisfy an interesting property that even if an attacker learns a group member U's group signing key for time period i, he still cannot identify which signatures generated before time period i were generated by U, i.e. all U's signatures before time period i still remain anonymous and unlinkable.

7. CONCLUSION

In this paper, we present our forward-secure group signature schemes. Our schemes satisfy forward security as well as all the traditional security properties shared with previous group signature schemes. Our schemes are efficient in the sense that they are independent of the number of group members and the size of signatures and group keys are independent on the number of time periods during the lifetime of the group public key. In addition, we extend our schemes to provide the first solutions to enable retroactive-publicly-revokable group membership with backward unlinkability and the signature size is independent of the number of revoked members.

Our approach illustrates that forward security is a particularly important property for group signature schemes, not only because the danger of the exposure of signing keys escalates as the group size increases, but also because forward security help enable other desired security properties at little extra cost, such as time-limited group membership and retroactive revocation. Our forward secure group signature schemes also have a side benefit that even if an attacker learns a group member U's group signing key for time period i, he still cannot identify which signatures generated before time period i were generated by U, i.e. all U's signatures before time period i still remain anonymous and unlinkable.

Acknowledgements

We would like to thank Giuseppe Ateniese and Gene Tsudik for their many helpful discussions and contribution to section 5 (and particularly section 5.4). We would also like to thank anonymous reviewers for their helpful feedback, Doug Tygar and Adrian Perrig for their encouragement and help on the paper.

8. REFERENCES

[1] Michel Abdalla and Leonid Reyzin. A new forward-secure digital signature scheme. In *ASIACRYPT*, pages 116–129, 2000.

[2] Ross Anderson. Invited Lecture, 4th ACM Computer and Communications Security, 1997.

[3] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In M. Bellare, editor, *Advances in Cryptology - CRYPTO 2000*, pages 255–270. Springer-Verlag, 2000. Lecture Notes in Computer Science Volume 1880.

[4] Giuseppe Ateniese and Gene Tsudik. Some open issues and new directions in group signatures. In *Financial Crypto 1999*. Springer-Verlag, 1999.

[5] N. Baric and B. Pfitzman. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology - EUROCRYPT 1997*, pages 480–494. Springer-Verlag, 1997. Lecture Notes in Computer Science Volume 1233.

[6] Mihir Bellare and Sara Miner. A forward-secure digital signature scheme. In *Advances in Cryptology - CRYPTO'99*, 1999.

[7] D. Boneh. The decision diffie-hellman problem. In *Proceedings of the Third Algorithmic Number Theory Symposium*, pages 48–63. Springer-Verlag, 1998. Lecture Notes in Computer Science Volume 1423.

[8] F. Boudot. Efficient proofs that a committed number lies in an interval. In B. Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000*, pages 431–444, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science Volume 1807.

[9] Stefan Brands. An efficient off-line electronic cash system based on the representation problem. Technical Report CS-R9323, CWI, 1993.

[10] Emmanuel Bresson and Jacques Stern. Efficient revocation in group signatures. In *Proceeding of Public Key Cryptography (PKC 2001)*, 2001.

[11] J. Camenisch and M. Michels. A group signature with improved efficiency. In *Advances in Cryptology - ASIACRYPT '98*, pages 160–174, Berlin, 1998. Springer-Verlag. Lecture Notes in Computer Science Volume 1514.

[12] J. Camenisch and M. Michels. Separability and efficiency for generic group signature schemes. In M. Wiener, editor, *Advances in Cryptology - Crypto '99*, pages 413–430, Berlin, 1999. Springer-Verlag. Lecture Notes in Computer Science Volume 1666.

[13] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In B. Kaliski, editor, *Advances in Cryptology - CRYPTO'97*, pages 410–424. Springer-Verlag, 1997. Lecture Notes in Computer Science Volume 1296.

[14] D. Chaum. Zero-knowledge undeniable signatures (extended abstract). In Ivan B. Damgård, editor, *Advances in Cryptology - EuroCrypt '90*, pages 458–464, Berlin, 1990. Springer-Verlag. Lecture Notes in Computer Science Volume 473.

[15] D. Chaum, J. H. Evertse, and J. van de Graaf. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In David Chaum and Wyn L. Price, editors, *Advances in Cryptology - EuroCrypt '87*, pages 127–142, Berlin, 1987. Springer-Verlag. Lecture Notes in Computer Science Volume 304.

[16] D. Chaum and E. van Heyst. Group signatures. In Donald W. Davies, editor, *Advances in Cryptology - EuroCrypt '91*, pages 257–265, Berlin, 1991. Springer-Verlag. Lecture Notes in Computer Science Volume 547.

[17] D. Chaum and T. P. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *Advances in Cryptology - Crypto '92*, pages 89–105, Berlin, 1992. Springer-Verlag. Lecture Notes in Computer Science Volume 740.

[18] L. Chen and T. P. Pedersen. New group signature schemes. In Alfredo De Santis, editor, *Advances in Cryptology - EuroCrypt '94*, pages 171–181, Berlin, 1995. Springer-Verlag. Lecture Notes in Computer Science Volume 950.

[19] I. Damgard. Efficient concurrent zero-knowledge in the auxiliary string model. In B. Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000*, pages 431–444, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science Volume 1807.

[20] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 6(IT-22):644–654, 1976.

[21] A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. In A. M. Odlyzko, editor, *Advances in Cryptology - Crypto '86*, pages 186–194, Berlin, 1986. Springer-Verlag. Lecture Notes in Computer Science Volume 263.

[22] E. Fujisaki and T. Okamoto. Statistical zero-knowledge protocols to prove modular polynomial relations. In B. Kaliski, editor, *Advances in Cryptology - Crypto '97*, pages 16–30, Berlin, 1997. Springer-Verlag. Lecture Notes in Computer Science Volume 1294.

[23] A. Herzberg, M. Jarecki, H. Krawczyk, and M. Yung. Proactive secret sharing or: How to cope with perpetual leakage. In *Advances in Cryptology - CRYPTO'95*. Springer-Verlag, 1995. Lecture Notes in Computer Science Volume 1807.

[24] Gene Itkis and Leonid Reyzin. Forward-secure signatures with optimal signing and verifying. In *To Appear in CRYPTO 2001*, 2001.

[25] J. Kilian and E. Petrank. Identity escrow. In *Advances in Cryptology - CRYPTO'98*, pages 169–185. Springer-Verlag, 1998. Lecture Notes in Computer Science Volume 1642.

[26] Hugo Krawczyk. Simple forward-secure signatures from any signature scheme. In *7th ACM Conference on Computer and Communication Security*, 2000.

[27] A. Lysyanskaya and Z. Ramzan. Group blind digital signatures: A scalable solution to electronic cash. In *Financial Cryptography (FC'98)*, pages 184–197. Springer-Verlag, 1998. Lecture Notes in Computer Science Volume 1465.

[28] Chanathip Namprempre Michel Abdalla, Sara Miner. Forward security in threshold signature schemes. In *RSA 2001*, 2001.

[29] C. P. Schnorr. Efficient identification and signatures for smart cards. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology - EuroCrypt '89*, pages 688–689, Berlin, 1989. Springer-Verlag. Lecture Notes in Computer Science Volume 434.

APPENDIX

A. SECURITY ANALYSIS

We provide here the proofs of the lemmas stated in the paper. The proofs are related to proofs in [3]. The proof for Lemma 3 is similar to the proof in Lemma 2. The proofs for Lemma 4 and 5 and are similar to the proofs in [3]. Corollary 1 and 2 can be easily proven from the lemmas. Therefore we omit the proofs for Lemma 2, 4 and 5 and Corollary 1 and 2 here.

Lemma 1. $PK\{(\alpha,\beta,\delta,\epsilon) : d^2 = (A^{2^{T-j+1}})^{\alpha}(1/(a^2))^{\beta}(1/(y^2))^{\delta} \wedge B^2 = (g^2)^{\epsilon} \wedge 1 = (B^{2^{T-j+1}})^{\alpha}(1/(g^2))^{\delta} \wedge \beta \in \Gamma \wedge \alpha \in \Lambda\}(m)$ *is a statistical zero-knowledge proof of knowledge of a group signing key valid for time period* j.

PROOF SKETCH. It is easy to see that the protocol is statistical zero-knowledge. Now we show how to build a knowledge extractor for the group signing key. Under the properties of the PK protocols and under the strong RSA assumption, the knowledge extractor can produce values $\alpha, \beta, \delta, \epsilon$ such that the statement after the colon holds. In particular

from $B^2 = (g^2)^\epsilon$ and $1 = (B^{2^{T-j+1}})^\alpha (1/(g^2))^\delta$, we get that $\delta \equiv 2^{T-j}\alpha\epsilon \mod \mathrm{ord}(g)$. Furthermore, we have $d^2(a^2)^\beta = (A^{2^{T-j+1}})^\alpha (1/(y^2)^\delta = (A^2/(y^2)^\epsilon)^{2^{T-j}\alpha}$. As $\beta \in \Gamma, \alpha \in \Lambda$, we see that $(A^2/(y^2)^\epsilon, \alpha, \beta, \gamma)$ is a valid group signing key for time period j. Hence the signer must know a valid group signing key for time period j. □

Lemma 2. In Scheme I, under the strong RSA assumption, a group signing key for some time period t, (x_u, e_u, c_u) where $c_u^{2^{T-t}e_u} = da^{x_u} \mod n$ and $x_u \in \Gamma$, and $e_u \in \Lambda$, can only be generated by the group manager given that the number of group signing keys the group manager issues is polynomially bounded.

PROOF SKETCH. Let $\mathcal{M}$ be an attacker that is allowed to adaptively run the JOIN procedure and thereby obtain group signing keys $\{(x_i, e_i, c_i)\}_{1\le i\le K}$ where $d = a^{x_i}c_i^{e_i} \mod n$. We show in the following that if $\mathcal{M}$ outputs a tuple $(\hat{x}, \hat{e}, \hat{c}, t)$ with $\hat{x} \in \Gamma, \hat{e} \in \Lambda, da^{\hat{x}} = \hat{c}^{2^{T-t}\hat{e}} \mod n$, and $(\hat{x}, \hat{e}) \neq (x_i, e_i)$ for all $1 \le i \le K$, i.e. a new group signing key valid for time period t, with non-negligible probability, then we can use $\mathcal{M}$ to break the strong RSA assumption.

Given a pair (n, z), where n is the product of two safe primes and $z \in \mathrm{QR}_n$ and we would like to solve the strong RSA problem, meaning we would like to find a pair $(v, e) \in \mathcal{Z}_n \times \mathcal{Z}_{>1}$ such that $v^e \equiv z \mod n$. We look for such a pair (v, e) by repeatedly playing a random one of the following two games with $\mathcal{M}$.

Game 1.

1. Select $v_1, \ldots, v_K \in_R \Gamma$ and primes $e_1, \ldots, e_K \in_R \Lambda$.
2. Let $\alpha = \prod_{1\le i\le K} e_i$, and $a = z^{2^T\alpha} \mod n$.
3. Choose $r_2 \in_R \Gamma$, let $d = a^{r_2} \mod n$.
4. For $1 \le i \le K$, let $\alpha_i = \prod_{1\le l\le K; l\neq i} e_l \mod n$, and $c_i = z^{(v_i+r_2)\alpha_i} \mod n$.
5. Select $g \in_R \mathrm{QR}_n$ and publish (n, a, d, g, h) as the group public key.
6. When $\mathcal{M}$ sends $s_i = g^{\tilde{x}_i}h^{t_1}$ and asks for the corresponding group signing key, we first extract $\tilde{x}_i$ from the zero-knowledge proof. We then compute w_i such that $v_i = (w_i + \tilde{x}_i \mod (2^{\ell_\Gamma+1} - 1)) - 2^{\ell_\Gamma} + 1$, and send (c_i, e_i, w_i) to $\mathcal{M}$.
7. After K requests, $\mathcal{M}$ outputs a new group signing key $(\hat{x}, \hat{e}, \hat{c})$ valid for time interval t, i.e. $da^{\hat{x}} \equiv \hat{c}^{2^{T-t}\hat{e}} \mod n$. In particular, $\hat{c}^{2^{T-t}\hat{e}} \equiv z^{2^T\alpha(\hat{x}+r_2)} \mod n$. Because squaring is a permutation in QR_n, we have $(\hat{c}^2)^{\hat{e}} \equiv z^{2^{t+1}\alpha(\hat{x}+r_2)} \mod n$.
8. If $\gcd(\hat{e}, e_j) \neq 1$ for some $1 \le j \le K$, then output $\perp$ and quit. Otherwise, Let $\tilde{e} := 2^{t+1}(\hat{x} + r_2)$. Because $\gcd(\hat{e}, \alpha) = 1$, we have $\gcd(\hat{e}, \alpha\tilde{e}) = \gcd(\hat{e}, \tilde{e})$. By the extended Euclidean algorithm, there exists $\sigma, \tau \in \mathcal{Z}$ s.t. $\sigma\hat{e} + \tau(\tilde{e}\alpha) = \gcd(\hat{e}, \tilde{e})$. Let $v := z^\sigma(\hat{c}^2)^\tau \mod n$, and $e := \hat{e}/\gcd(\hat{e}, \tilde{e})$. We have $v^e \equiv z \mod n$. Because $\hat{e} \in \Lambda$, and $\tilde{e} < 2^{T+1}(2^{\ell_\Gamma} + 2^{\ell_\Gamma}) < 2^{\ell_\Lambda}$, we have $e > 1$. Therefore, (v, e) is a pair that solves the strong RSA problem, i.e. $v^e \equiv z \mod n$ and $e > 1$.

Game 2.

1. Select $v_1, \ldots, v_K \in_R \Gamma$ and primes $e_1, \ldots, e_K \in_R \Lambda$.
2. Choose $k \in_R \{1, \ldots, K\}$. Let $\alpha = \prod_{1\le i\le K} e_i, \beta_i = \prod_{1\le l\le K; l\neq i,k} e_l \mod n$, for all $1 \le i \le K$. In particular $\beta_k = \prod_{1\le l\le K; l\neq k} e_l \mod n$. Set $a = z^{2^T\beta_k} \mod n$.
3. Choose $r_1 \in_R \Gamma$, let $c_k = z^{\beta_k r_1} \mod n, d = (c_k^{e_k}/z^{\beta_k v_k})^{2^T} \mod n$.
4. For all $1 \le j \le K, j \neq k$, compute $c_j = z^{(v_j + e_k r_1 - v_k)\beta_j} \mod n$.
5. Select $g \in_R \mathrm{QR}_n$ and publish (n, a, d, g) as the public key.
6. When $\mathcal{M}$ sends $s_i = g^{\tilde{x}_i}h^{t_1}$ and asks for the corresponding group signing key, we first extract $\tilde{x}_i$ from the zero-knowledge proof. We then compute w_i such that $v_i = (w_i + \tilde{x}_i \mod (2^{\ell_\Gamma+1} - 1)) - 2^{\ell_\Gamma} + 1$, and send (c_i, e_i, w_i) to $\mathcal{M}$.
7. After K requests, $\mathcal{M}$ outputs a new group signing key $(\hat{x}, \hat{e}, \hat{c})$ valid for time period t, i.e. $da^{\hat{x}} \equiv \hat{c}^{2^{T-t}\hat{e}} \mod n$. In particular, $\hat{c}^{2^{T-t}\hat{e}} \equiv z^{2^T\beta_k(r_1e_k - v_k + \hat{x})} \mod n$.
8. If $\gcd(\hat{e}, e_k) \neq e_k$, then output $\perp$ and quit. Otherwise we have $\hat{e} = e_k$ because $\hat{e} \in \Lambda$. Therefore $(\hat{c}/c_k^{2^t})^{2^{T-t}e_k} = z^{2^T\beta_k(\hat{x}-v_k)} \mod n$, and hence $(\hat{c}^2/c_k^{2^{t+1}})^{e_k} = z^{2^{t+1}\beta_k(\hat{x}-v_k)} \mod n$. Let $\tilde{e} := 2^{t+1}(\hat{x} - v_k)$. Because $\gcd(e_k, \beta_k) = 1$, we have $\gcd(e_k, \beta_k\tilde{e}) = \gcd(e_k, \tilde{e})$. By the extended Euclidean algorithm, there exists $\sigma, \tau \in \mathcal{Z}$ s.t. $\sigma e_k + \tau(\tilde{e}\beta_k) = \gcd(e_k, \tilde{e})$. Let $v := z^\sigma(\hat{c}^2/(c_k^{2^{t+1}}))^\tau \mod n$, and $e := e_k/\gcd(e_k, \tilde{e})$. We have $v^e \equiv z \mod n$. Because $e_k \in \Lambda$, and $\tilde{e} < 2^{T+1}(2^{\ell_\Gamma} + 2^{\ell_\Gamma}) < 2^{\ell_\Lambda}$, we have $e > 1$. Therefore, (v, e) is a pair that solves the strong RSA problem, i.e. $v^e \equiv z \mod n$ and $e > 1$.

Consequently, we can use $\mathcal{M}$ to solve the strong RSA problem in expected running-time polynomial in K by playing randomly Game 1 or Game 2 until the result is not $\perp$. Therefore no one but the group manager can generate group signing keys in our scheme under the strong RSA assumption.

□

Lemma 6. Under the strong RSA assumption, Scheme II satisfies strong forward security given that the number of group signing keys the group manager issues is polynomially bounded.

PROOF SKETCH. Let $\mathcal{M}$ be an attacker who can break the strong forward security of Scheme II with non-negligible probability. We show that we can use $\mathcal{M}$ to break the strong RSA assumption.

Given a pair (n, z), where n is the product of two safe primes and $z \in \mathrm{QR}_n$, we would like to solve the strong RSA problem, meaning we would like to find a pair $(v, e) \in \mathcal{Z}_n \times \mathcal{Z}_{>1}$ such that $v^e \equiv z \mod n$. We look for such a pair (v, e) by repeatedly playing a random one of the following two games with $\mathcal{M}$.

Game 1.

We first randomly select primes $\Phi = \{e_{1,t_1}, \ldots, e_{K,t_K}\}$ where $e_{i,t_i} \in \Lambda_{t_i}$.
Let $\alpha = \prod_{1\le i\le K, t_i\le w_i\le T} e_{i,w_i}$ where e_{i,w_i} represents the

element for time period w_i in the one-way chain generated from e_{i,t_i}. Let $a = z^\alpha \mod n$. Choose $r \in_R \Gamma$, and let $d = a^r$. For the i-th request, $\mathcal{M}$ asks for a group signing key for some time period s_i at his will. We answer the request by randomly selecting a $e_{a_i,t_{a_i}} \in \Phi$ such that $t_{a_i} = s_i$ and $e_{a_i,t_{a_i}}$ has not been used to answer $\mathcal{M}$'s previous requests. If no such $e_{a_i,t_{a_i}}$ exists, then abort. Otherwise, we randomly select $x_i \in \Gamma$, and computes $c_i = z^{(x_i+r)\alpha/(e_{a_i,t_{a_i}})} \mod n$ and reply $\mathcal{M}$ with $(x_i, c_i, e_{a_i,t_{a_i}})$.

Assume after L requests, $\mathcal{M}$ outputs a valid group signing key $(\tilde{x}, \tilde{c}, \tilde{e})$ not in the span of the group signing keys from his requests. If $\gcd(\tilde{e}, e_{a_i,w_{a_i}}) \neq 1$ for some $1 \leq i \leq L, t_{a_i} \leq w_{a_i} \leq T$ then abort. Otherwise, similar to the argument shown in the previous proof, using the extended Euclidean algorithm, one can break the strong RSA assumption, i.e. find a pair (v, e) such that $v^e \equiv z \mod n$ and $e > 1$.

Game 2.. We first randomly select primes $\Phi = \{e_{1,t_1}, \ldots, e_{K,t_K}\}$ where $e_{i,t_i} \in \Lambda_{t_i}$. Let $\alpha = \prod_{1 \leq i \leq K, t_i \leq w_i \leq T} e_{i,w_i}$ where e_{i,w_i} represents the element for time period w_i in the one-way chain generated from e_{i,t_i}. We randomly select $k \in_R \{1, \ldots, K\}$. Let $\beta_i = \alpha/e_{i,t_i}$. Let $a = z^{\beta_k} \mod n$. Choose $r, x_k \in_R \Gamma$, and let $c_k = a^r \mod n, d = a^{re_{k,t_k} - x_k}$. For the i-th request, $\mathcal{M}$ asks for a group signing key for time period s_i. We answer the request by randomly selecting a $e_{a_i,t_{a_i}} \in \Phi$ such that $t_{a_i} = s_i$ and $e_{a_i,t_{a_i}}$ has not been used to answer $\mathcal{M}$'s previous requests. If no such $e_{a_i,t_{a_i}}$ exists, then abort. Otherwise, we randomly select $x_i \in \Gamma$, and computes $c_i = z^{(x_i + re_{k,t_k} - x_k)\beta_k / e_{a_i,t_{a_i}}} \mod n$ and reply $\mathcal{M}$ with $(x_i, c_i, e_{a_i,t_{a_i}})$.

Assume $\mathcal{M}$ outputs a valid group signing key $(\tilde{x}, \tilde{c}, \tilde{e})$ not in the span of the group signing keys from his requests. If $\gcd(\tilde{e}, e_{k,t_k}) \neq e_{k,t_k}$ then abort. Otherwise, similar to the argument shown in previous proof, using the extended Euclidean algorithm, one can break the strong RSA assumption, i.e. find a pair (v, e) such that $v^e \equiv z \mod n$ and $e > 1$.

Note that when the number of group signing keys issued by the group manager is polynomially bounded, Game 1 or Game 2 will succeed with non-negligible probability given $\mathcal{M}$. □

B. OUR ZERO-KNOWLEDGE PROOF PROTOCOL

We now explain the details of the zero-knowledge proof protocol PK2$\{(\alpha, \beta) : A = \alpha y^\beta \wedge B = g^\beta \wedge C = g_2^\alpha\}$ we used in section 4.4. This protocol is to show that given values $A, B \in \mathrm{QR}_n$ and $C \in G$, where g_2 is a generator of G which is a group of order n and in which the DDH problem is hard, the prover knows (c_u, r_1, r_2) such that $A = c_u y^{r_1}, B = g^{r_1}, C = g_2^{c_u}$. The protocol repeats the following protocol for sufficient times:

1. P select $t_1, t_2, t_3 \in_R \mathcal{Z}_n, t_4 \in_R \{0,1\}^{\epsilon(\ell_n+k)+1}$, and compute $E = A^{t_2}, a_1 = g_2^{t_1}, a_3 = t_3^{t_2}, a_4 = g^{t_4}, a_5 = y^{t_4}$ and send V (E, a_1, a_3, a_4, a_5)
2. V selects $c_2 \in \{0,1\}^k$.
3. P computes $b_1 = t_1 + c_2 c_u t_3 \mod n, b_2 = t_4 + c_2 r_1$.
4. V sends P $c_1 \in \{0,1\}$.
5. If $c_1 = 0$, P sends V $f_0 = c_u t_3 \mod n$; If $c_1 = 1$, P sends V $f_1 = t_3 \mod n$.
6. - If $c_1 = 1$, then V verifies ${g_2}^{b_1} = a_1 C^{c_2 f_1}, {g}^{b_2} = a_4 B^{c_2}$. And P and V engage in a zero knowledge proof, PK$\{(\alpha) : a_3 = f_1^\alpha \wedge E = A^\alpha\}$.
   - If $c_1 = 0$, then V verifies ${g_2}^{b_1} = a_1 {g_2}^{c_2 f_0}, {g}^{b_2} = a_4 B^{c_2}$. And P and V engage in a zero knowledge proof, PK$\{(\alpha) : (Ea_3)^{c_2} = (f_0^{c_2}(y)^{b_2}(a_5)^{-1})^\alpha \wedge E = A^\alpha\}$.

LEMMA 7. *PK2$\{(\alpha, \beta) : A = \alpha y^\beta \wedge B = g^\beta \wedge C = g_2^\alpha\}$ is a zero-knowledge proof protocol.*

PROOF SKETCH. We first show the knowledge extractor. Assume we can rewind the prover so that we give it two different values $c_1 = 0$ and $c_1 = 1$ after the third step. Thus $g_2^{b_1} = a_1 C^{c_2 f_1} = a_1 g_2^{c_2 f_0}$. Assume $C = g_2^{x_1}$, then $f_0 = f_1 x_1 \mod n$. From rewinding the second step, we get (x_2) such that $B = g^{x_2}$. From rewinding the proof of knowledge protocol in the verification step, we can get x_4 such that $a_3 = f_1^{x_4}, E = A^{x_4}$, and $(Af_1)^{c_2 x_4} = (f_1 x_1 y^{x_2})^{c_2 x_4}$. So $A = x_1 y^{x_2}$.

We now show the simulation argument. The following algorithm constitutes a simulator S for the output of any verifier V.

1. S chooses $c_1 \in_R \{0,1\}$.
2. If $c_1 = 1$, S randomly chooses b_1, f_1, t_2, b_2, a_5 in their appropriate ranges. S then also randomly chooses $c_2 \in_R \{0,1\}^k$, and computes $a_1 = g_2^{b_1}(C^{c_2 f_1})^{-1}, a_3 = f_1^{t_2}, E = A^{t_2}, a_4 = g^{b_2} B^{-c_2}$.

   If $c_1 = 0$, S randomly chooses b_1, b_2, t_2, t_3, f_0 from their appropriate ranges. S then also randomly chooses $c_2 \in_R \{0,1\}^k$, and computes $a_1 = g_2^{b_1}({g_2}^{c_2 f_0})^{-1}, E = A^{t_2}$, $a_3 = t_3^{t_2}, a_4 = f_0^{c_2} h^{b_2} A^{-c_2} t_3^{-c_2}, a_4 = g^{b_2} B^{-c_2}$, $a_5 = y^{b_2} f_0^{c_2} (At_3)^{-c_2}$.
3. S runs V, sends it the generated values (E, a_1, a_3, a_4, a_5) and receives a c_2'.
4. If $c_2' = c_2$, then S sends the generated values (b_1, b_2) to V and receives a c_1'. Otherwise S continues with Step 1.
5. If $c_1' = c_1$, then S sends f_0 if $c_1' = 0$ or f_1 if $c_1' = 1$, and completes the verification step with V. Otherwise S continues with Step 1.

By construction, the output of the simulator is statistically distributed to the output of the verifier. Hence if we choose $k = \Theta(\mathrm{poly}(\ell))$ then the protocol is zero-knowledge. □