Ring Group Signatures

Liqun Chen

Hewlett-Packard Laboratories,
Long Down Avenue,
Stoke Gifford,
Bristol, BS34 8QZ,
United Kingdom.
liqun.chen@hp.com

Abstract. In many applications of group signatures, not only a signer's identity but also which group the signer belongs to is sensitive information regarding signer privacy. In this paper, we study these applications and combine a group signature with a ring signature to create a ring group signature, which specifies a set of possible groups without revealing which member of which group produced the signature. The main contributions of this paper are a formal definition of a ring group signature scheme and its security model, a generic construction and a concrete example of such a scheme. Both the construction and concrete scheme are provably secure if the underlying group signature and ring signature schemes are secure.[1]

Keywords: ring signatures, group signatures, ring group signatures, group-ambiguity.

1 Introduction

Digital signatures have widely been used to provide services such as entity authentication, data origin authentication, non-repudiation, and data integrity. Anonymous digital signatures are a special type of digital signature. In an anonymous digital signature scheme, given a digital signature, an unauthorised entity, including the verifier, cannot discover the signer's identifier. However such a scheme still has the property that only a legitimate signer can generate a valid signature.

One of the major differences between a conventional digital signature and an anonymous digital signature is the nature of the public keys used to perform the signature verification. To verify a conventional digital signature, the verifier makes use of a single public verification key which is bound to the signer's identifier. To verify an anonymous digital signature, the verifier makes use of either a group public key or multiple public keys,

[1] This paper is an updated version of [19].

which are not bound to an individual signer. In the literature, an anonymous signature using a group public key is commonly known as a Group Signature (GS for short) [17], and an anonymous signature using multiple public keys is commonly known as a Ring Signature (RS for short) [31].

1.1 Motivation of this work

Given a GS, as proposed over 20 years ago by Chaum and van Heyst [17], its signer's identity is hidden in a group, of which the signer is a member, but the group's identity is revealed. It is not difficult to see that in many applications, not only the signer identity but also the group identity might contain some sensitive information which affects signer privacy. Here are a few examples of such applications:

Example 1. Affiliation membership authentication. In a social network, only legitimate members from those affiliations having a contract with the network service provider are allowed to access to the network. A user need to prove that he is a legitimate member of a set of legitimate affiliations but does not have to shown his identity from a specific affiliation.

Example 2. Vehicle communications. A modern vehicular *ad hoc* network (VANET), as discussed in [21, 30], allows legitimate vehicles to communicate with each other. To take part in the network, a driver is required proving that his vehicle is properly registered, but is not required to show with which registration authority.

Example 3. Computing platform attestation. By using the trusted computing technology, such as this in [12, 35], a Trusted Platform Module (TPM) can attest correctness of platform configurations. The owner of a TPM wants not only to hide the TPM identity but also not to disclose from which authority the TPM obtains an attestation credential.

Example 4. Fair exchanges between enterprises. When two companies, say A and B, work on a sensitive contract, neither A nor B wants the other company to be able to tell a third party that A or B has signed the contract before both the companies exchange their signatures to each other, and neither of them wants to reveal which individual employee signs the contract on behalf of the company.

Protection of group identities in these applications is thus a matter of importance. Our motivation of this work is to design a special type of GSs with the property of hiding group identities. We observe the nature of each the applications that the number of legitimate groups is reasonably small compared with the number of legitimate members of each group, and that the identities of these groups, associated with the group public keys, are always made publicly available to verifiers. This observation leads us to the basic idea that we can hide the group identity by following the elegant method of Ring Signatures (RSs) [31]. As a result, we propose to merge two notions of a GS and RS to a notion of a Ring Group Signature (RGS). From a RGS, signer privacy is not only dependent on the size of the signer's group but also associated with the size of the group set chosen by the signer. In the other words, a RGS holds both *signer-anonymity* and *group-ambiguity*. By the signer-anonymity property, we mean that a signer is hidden in a group, and by the group-ambiguity property, we mean that the group is hidden in a set of specified groups.

Although the idea of merging these two signatures is straightforward, combining them smoothly, efficiently, generically and securely is not trivial since both group and ring signatures are delicate and complex. Furthermore, we need figure out how a group with a true signer can trace a signature to the signer and other groups cannot, and how to avoid an unbound adversary to break group-ambiguity.

1.2 Related prior works

Many cryptographic primitives have been developed to address the problem of simultaneously achieving user authentication and user privacy with a variety of mechanisms and varying degrees of success. Associated with the digital signature technology, two of the most attractive and significant primitives are group signatures and ring signatures.

The notion of GS schemes was introduced by Chaum and van Heyst [17] in 1991. A typical GS scheme involves a group manager, a set of group members and a set of verifiers. The manager is in charge of verifying the legitimation of group members and issuing a membership certificate to each member. With the certificate, a member can create digital signatures on the behalf of the group. A verifier makes use of a group public key to verify the signature, and cannot identify the individual signer. The group manager is able to trace the signer's identity from the signature, and to revoke any member who is no longer legitimate. Since the pioneering work [17], the group signature primitive has become one of the favorite primitives of cryptographers. Researchers have proposed a large number of group signature schemes, e.g. [2, 8, 10, 14–16, 20, 25, 29], formal definitions and security models for the group signature schemes, e.g. [5, 6, 26], and a variety of group member join and revocation solutions, e.g. [3, 10, 13, 20, 26].

The notion of RS schemes was formalized by Rivest, Shamir and Tauman [31] in 2001, although the general concept itself (under different terminology) was first introduced by Cramer, Damgård and Schoenmakers [24] in 1994. In such a scheme, a signer chooses a set of independent possible signers including himself, and signs a message by using his secret key and the others' public keys without their approval or assistance. A verifier makes use of the whole public key set to verify the signature, and is convinced that the signature was signed by one of the signers without revealing which one is the true signer. Ring signatures are another fruitful research topic loved by cryptographers. A large number of RS schemes have been developed, e.g. [7, 9, 11, 22, 36].

Both GSs and RSs preserve signer privacy, but they are suitable for two different usage models. As mentioned in [31], GSs are useful when the members want to cooperate, while RSs are useful when the members do not want to cooperate. In a GS, the signer's identity is hidden in his group, and in a RS, the signer's identity is hidden in a set of possible signers. In these two signatures, signer privacy is dependent on either the size of the group or the size of the possible signer set.

In the literature, there are a lot of notions which provide either special types of group signatures or special types of ring signatures in order to achieve different levels of signer privacy. For instance, hierarchical group signatures [33] require group managers are organized in a tree with a root which enables to trace a hierarchical group signature to a right group. RGSs do not require different groups having any relationship to one another, and do not allow roots. Threshold ring signatures [11] allow an ad-hoc groups of k members from the whole n ring members to make a k-out-of-n ring signature. Linkable ring signatures [28] allow anyone to tell whether two signatures are generated by the same signer while still maintaining the anonymity of the real signer. Linkable threshold ring signatures [34] hold the properties of linkable ring signatures and threshold ring signatures. Escrowed linkable ring signatures [23] allow a trusted authority to link two ring signatures created by the same signer. Revocable ring signatures [27] allow a set of authorities to revoke the anonymity of the real signer, and revocable-iff-linked ring signatures [4] allow anyone to revoke the actual signer if the signer has signed a single event more than allowed times.

1.3 Our contributions

In this paper, we investigate a new variant of special group signatures and ring signatures and name it Ring Group Signatures (RGSs). We aim to meet the requirements in both of the GS model and RS model, i.e., the members inside of a group want to cooperate and outside of their group do not want to cooperate. The most properties in our variant can be found in various of the special types of GSs and/or of RSs, mentioned above. From this point of view, RGSs are not a completely new notion. However, to our best knowledge, this notion has not been formally explored from the angle of merging GSs and RSs. It is worth doing this, since we can benefit from these two well-studied primitives, and also since RGSs have many interesting applications.

RGS schemes keep the main features from group signature schemes as well as ring signature schemes. Like group signatures, each group has a manager and a set of members. Similar to ring signatures, neither setup procedures nor coordination among groups is required. It is possible for any member in any group to choose a set of groups including his own, and to sign a message by using his membership private key and other groups' public keys without their approval or assistance.

We introduce the concept of RGSs in three aspects: formal definitions, a generic construction and a concrete scheme. Following a brief overview of the GS and RS definitions (Section 2), we give a formal definition of syntax of RGS schemes, of a multi-party adversarial model, and of what it means for such schemes to be secure (Section 3). Security of them is defined via the notions of signer-anonymity, signer-traceability, signer-non-frameability and group-ambiguity. We notice that any suitable GS and RS schemes can be used to produce a RGS scheme. We choose to base our generic construction on the generic 3-move type group signature construction and a 3-move type ring signature scheme (Section 4). We choose to base our concrete RGS scheme on the GS scheme enable strong exculpability of [8] and the discrete-logarithm RS scheme of [1] (Section 5). We prove that both the generic construction and concrete scheme are unconditionally group-ambiguous and secure in the random oracle model. Finally, we conclude the paper in Section 6.

2 Preliminaries

Before formally defining a ring group signature (RGS) scheme in the next section, we first recap on some standard notation used throughout this paper, and then briefly review syntax and security notions of a group signature (GS) scheme and ring signature (RS) scheme; these two are building blocks of the RGS scheme.

2.1 Notation

Let $\mathbb{N}$ be the set of positive integers $\{1, 2, \ldots\}$. If s is a string, then ℓ_s denotes its length in the binary representation. The empty string is denoted by ε. If S is a set, then $|S|$ denotes its size, and $x \leftarrow S$ denotes the action of sampling an element from S uniformly at random and assigning the result to x. The empty set is denoted by $\emptyset$, the set of binary strings of length t is denoted by $\{0, 1\}^t$ and the set of binary strings of arbitrary length is denoted by $\{0, 1\}^*$. If n is a positive integer, i.e. $n \in \mathbb{N}$, then 1^n denotes the string of n ones, $[n]$ denotes the set of $\{0, ..., n-1\}$.

If A is an algorithm, then $z \leftarrow A(y_1, \ldots, y_n)$ denotes the action of running A on inputs $y_1 \ldots, y_n$ and letting z be the output; $[A(y_1, \ldots, y_n)]$ denotes the set of all points having positive probability of being output by A on inputs $y_1, \ldots, y_n$; $z \overset{\mathrm{R}}{\leftarrow} A(y_1, \ldots, y_n)$ denotes the action as $z \leftarrow A(y_1, \ldots, y_n)$, plus the extra property that A is a randomized algorithm. If $\boldsymbol{d}$ is a vector of strings, then $|\boldsymbol{d}|$ denotes its size and $\boldsymbol{d}$ is written as (d_0, d_1, ..., $d_{|\boldsymbol{d}|-1}$), and $\ell_{\boldsymbol{d}}$ denotes its length of the binary representation, i.e. $\ell_{\boldsymbol{d}} = \Sigma_{\ i\ \in\ [|\boldsymbol{d}|]} \ell_{d_i}$. When $\boldsymbol{d}$ is taken as input by a hash-function H, written as $H(..., \boldsymbol{d}, ...)$.

2.2 Definition of group signature schemes

There are several definitions of GS schemes, such as [5, 6, 26]. We follow the one supporting dynamic groups by Bellare et al. [6]. A GS scheme involves a group manager which consists of a trusted party for initial group key generation, an issuer and an opener, and a set of users, each with a unique identity $j \in \mathbb{N}$. A GS scheme consists of a tuple of algorithms and protocols written as $\mathcal{GS} = (\mathsf{G}_{\mathcal{GS}}, \mathsf{J}_{\mathcal{GS}}, \mathsf{S}_{\mathcal{GS}}, \mathsf{V}_{\mathcal{GS}}, \mathsf{O}_{\mathcal{GS}}, \mathsf{Ju}_{\mathcal{GS}})$, where

- $(gpk, ik, ok) \overset{\mathrm{R}}{\leftarrow} \mathsf{G}_{\mathcal{GS}}(1^\kappa)$ is a group key generation algorithm run by the trusted party, in which $\kappa \in \mathbb{N}$ is a security parameter, gpk is a group public key, ik is the issuer's private key and ok is the opener's private key.
- $(msk_j, reg_j) \overset{\mathrm{R}}{\leftarrow} \mathsf{J}_{\mathcal{GS}}(ik, j)$ is a joining protocol run between the issuer and a member j and creates a membership secret key msk_j and its registration reg_j (cryptographically bound with j). j obtains msk_j and the issuer records reg_j in his registration table reg.

- $\sigma_{\mathcal{GS}} \stackrel{\mathrm{R}}{\leftarrow} \mathsf{S}_{\mathcal{GS}}(gpk, msk_j, m)$ is a signing algorithm that creates a group signature $\sigma_{\mathcal{GS}}$ on a give message m.
- $1/0 \leftarrow \mathsf{V}_{\mathcal{GS}}(gpk, \sigma_{\mathcal{GS}}, m)$ is a verification algorithm that outputs 1 for accepting a given candidate signature $\sigma_{\mathcal{GS}}$ on a message m, and 0 for rejecting.
- $(j, \tau_j)/\perp \leftarrow \mathsf{O}_{\mathcal{GS}}(ok, reg, \sigma_{\mathcal{GS}}, m)$ is an opening algorithm that outputs the identity j and proof τ_j with the meaning that $\sigma_{\mathcal{GS}}$ was signed by j, or $\perp$ for $\mathsf{nonmatch}$.
- $1/0 \leftarrow \mathsf{Ju}_{\mathcal{GS}}(gpk, j, reg_j, \sigma_{\mathcal{GS}}, m, \tau_j)$ is a judge algorithm that outputs 1 for accepting the proof τ_j and 0 for rejecting.

As shown in [6], $\mathcal{GS}$ must satisfy four properties: *correctness* (honestly generated signatures verify and trace correctly), *anonymity* (a signature does not reveal its signer's identity), *traceability* (all signatures trace to identities of their real signers), and *non-frameability* (no members of the group and not even the group manager can produce signatures on behalf of other group members).

2.3 Definition of ring signature schemes

A number of formal definitions of RS schemes are available in the literature, e.g. [1, 7]. We follow the one by Abe et al. in [1]. A RS scheme involves an n-set ($n \in \mathbb{N}$) of possible signers, and consists of a triple of algorithms written as $\mathcal{RS} = (\mathsf{G}_{\mathcal{RS}}, \mathsf{S}_{\mathcal{RS}}, \mathsf{V}_{\mathcal{GS}})$, where

- $(pk, sk) \stackrel{\mathrm{R}}{\leftarrow} \mathsf{G}_{\mathcal{RS}}(1^\kappa)$ is a signer key generation algorithm, in which $\kappa \in \mathbb{N}$ is a security parameter, and $(pk,\ sk)$ is a public and private key pair.
- $\sigma_{\mathcal{RS}} \stackrel{\mathrm{R}}{\leftarrow} \mathsf{S}_{\mathcal{RS}}(sk_i, m, \boldsymbol{pk})$ is a signing algorithm that takes n public keys written as an n-vector $\boldsymbol{pk} = (pk_0, ..., pk_{n-1})$, a signer i's private key sk_i for $i \in [n]$ and a message m, and outputs a ring signature $\sigma_{\mathcal{RS}}$.
- $1/0 \leftarrow \mathsf{V}_{\mathcal{RS}}(\boldsymbol{pk}, \sigma_{\mathcal{RS}}, m)$ is a verification algorithm that takes as input a candidate signature $\sigma_{\mathcal{RS}}$ along with m and $\boldsymbol{pk}$ and outputs 1 for accepting and 0 for rejecting.

As shown in [1], $\mathcal{RS}$ must hold two security properties, *signer-ambiguity* (a signature is not bound to its signer) and *unforgeability* (a signature cannot be created without knowning a private key corresponding to a public key in the ring).

3 Definition of RGS schemes

3.1 Syntax of RGS schemes

This syntax is based on the definitions of GSs and RSs as shown in the previous section. Let $\mathcal{R}$ be a ring with n groups, and let $\{\boldsymbol{\kappa}, \boldsymbol{l}, \boldsymbol{gpk}, \boldsymbol{gsk}\}$ $= \{\kappa_i, l_i, gpk_i,\ gsk_i\}_{i\in[n]}$ refer to the list of security parameters, sizes, related public and secret keys for the groups in $\mathcal{R}$. Let $gsk_i = (ik_i, ok_i)$ contain two secret keys for the group issuer and opener respectively if these two authorities are separated; otherwise $gsk_i = ik_i = ok_i$. Let (i, j) denote the j-th member of the i-th group. A ring group signature (RGS) scheme consists of a tuple of protocols and algorithms written as $\mathcal{RGS} =$ $(\mathsf{G}_{\mathcal{RGS}}, \mathsf{J}_{\mathcal{RGS}}, \mathsf{S}_{\mathcal{RGS}}, \mathsf{V}_{\mathcal{RGS}}, \mathsf{O}_{\mathcal{RGS}}, \mathsf{Ju}_{\mathcal{RGS}})$, where:

- $(gpk_i, ik_i, ok_i) \xleftarrow{\text{R}} \mathsf{G}_{\mathcal{RGS}}(1^{\kappa_i})$ is a group key generation algorithm, in which each group i uses $\mathsf{G}_{\mathcal{GS}}(1^{\kappa_i})$ to generate their group key.
- $(msk_{(i,j)}, reg_{(i,j)}) \xleftarrow{\text{R}} \mathsf{J}_{\mathcal{RGS}}(ik_i, (i, j))$ is a joining protocol, in which each group i's issuer and the j-th member run the $\mathsf{J}_{\mathcal{GS}}(ik_i, j)$ protocol to create the $msk_{(i,j)}$ and $reg_{(i,j)}$ values, and the latter is recorded in the group registration table reg_i.
- $\sigma_{\mathcal{RGS}} \xleftarrow{\text{R}} \mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})$ run by (i, j) is a ring group signing algorithm that creates a ring group signature $\sigma_{\mathcal{RGS}}$ on a given message m.
- $0/1 \leftarrow \mathsf{V}_{\mathcal{RGS}}(m, \boldsymbol{gpk}, \sigma_{\mathcal{RGS}})$ is a ring group signature verification algorithm that outputs 1 for accepting the candidate signature $\sigma_{\mathcal{RGS}}$ on m and 0 for rejecting.
- $((i, j), \tau_{(i,j)}/\varepsilon)/\perp \leftarrow \mathsf{O}_{\mathcal{RGS}}(ok_i, reg_i, m, \boldsymbol{gpk}, \sigma_{\mathcal{RGS}})$ is a group member opening algorithm, in which group i's opener outputs an identity (i, j) and a proof $\tau_{(i,j)}$ saying that $\sigma_{\mathcal{RGS}}$ was signed by (i, j) or a letter ε if the proof is not required, or the symbol $\perp$ to indicate $\mathsf{nonmatch}$.
- $1/0 \leftarrow \mathsf{Ju}_{\mathcal{RGS}}(\boldsymbol{gpk}, (i, j), reg_{(i,j)}, \sigma_{\mathcal{RGS}}, m, \tau_{(i,j)})$ run by a trust judger is a judge algorithm that outputs 1 for accepting the proof $\tau_{(i,j)}$ and 0 for rejecting. This algorithm is optional.

Remark 1. An RGS scheme is built on combination of *underlying GS and RS schemes*. The groups in $\mathcal{R}$ may use different GS schemes from each other. The algorithms and protocols of the RGS scheme are according to the underlying GS schemes of all the groups involved. Security of the entirely RGS scheme is set to the smallest item of $\boldsymbol{\kappa}$.

Remark 2. It is assumed that except the public key gpk_i, no further information about the group i, such as the values $l_i, reg_i, (i, j), \tau_{(i,j)}$, is accessible by the other groups in $\mathcal{R}$. This assumption is reasonable for a RGS scheme, since it is designed for the applications where the groups in $\mathcal{R}$ may not trust each other.

Remark 3. In order to find out who the signer of a given signature is, the opener of each group with the interest of this result runs the $\mathsf{O}_{\mathcal{RGS}}$ algorithm under their own key. For a secure RGS scheme, only the opener from the signer's group will output the signer identity and the corresponding proof (if required), and the others will output $\perp$.

Remark 4. It is assumed that a judger is a trusted authority out side of $\mathcal{R}$, and the input to $\mathsf{Ju}_{\mathcal{RGS}}$ is kept at secret to other groups of $\mathcal{R}$. The $\mathsf{Ju}_{\mathcal{RGS}}$ algorithm is not mandatory (e.g., there may not exist such a judger in an application). If $\mathsf{Ju}_{\mathcal{RGS}}$ is omitted, the $\mathsf{O}_{\mathcal{RGS}}$ algorithm outputs ε instead of $\tau_{(i,j)}$.

An RGS scheme is required to satisfy five properties: *correctness*, *signer-anonymity*, *signer-traceability*, *signer-non-frameability*, and *group-ambiguity*, as defined as follows.

3.2 Correctness

This property ensures that honestly generated signatures verify and trace correctly. Specifically, $\mathcal{RGS}$ must satisfy the following correctness requirement: For all $m \in \{0,1\}^*$, all $n, l_i, \kappa_i \in \mathbb{N}$, all $(i,j) \in ([n],[l_i])$, all $\boldsymbol{gpk}$ containing gpk_i, all $(gpk_i, ik_i, ok_i) \in [\mathsf{G}_{\mathcal{RGS}}(1^{\kappa_i})]$, and all $(msk_{(i,j)}, reg_{(i,j)}) \in [\mathsf{J}_{\mathcal{RGS}}(ik_i,(i,j))]$,

$$\begin{aligned} \mathsf{V}_{\mathcal{RGS}}(m, \boldsymbol{gpk}, \mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})) = 1, \\ \mathsf{O}_{\mathcal{RGS}}(ok_i, m, \boldsymbol{gpk}, \mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})) = ((i,j), \tau_{(i,j)}), \\ \mathsf{Ju}_{\mathcal{RGS}}(\boldsymbol{gpk}, (i,j), reg_{(i,j)}, \sigma_{\mathcal{RGS}}, m, \tau_{(i,j)}) = 1. \end{aligned}$$

The first equation shows that a *true signature* must always be valid; the second and third equations show that a signature signed under a *true group membership secret* must be traced to its *true signer* by the signer's own group opener.

Remark 5. Intuitively, we interpret the differentiation between a *true* and a *valid* signature, between a *true* and a *traced* signer, and between a *true* and a *valid* membership secret:

- We say that $\sigma_{\mathcal{RGS}}$ is a *true* signature on m if $\sigma_{\mathcal{RGS}} \in [\mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})]$, $(msk_{(i,j)}, .) \in [\mathsf{J}_{\mathcal{RGS}}(ik_i,(i,j))]$, and $(gpk_i, ik_i, .) \in [\mathsf{G}_{\mathcal{RGS}}(1^{\kappa_i})]$ where $gpk_i \in \boldsymbol{gpk}$; we say that $\sigma_{\mathcal{RGS}}$ is a *valid* signature on m with respect to $\boldsymbol{gpk}$ if $\mathsf{V}_{\mathcal{RGS}}(m, \boldsymbol{gpk}, \sigma_{\mathcal{RGS}}) = 1$.

- We say a player with the identity (i, j) is a *true* signer of $\sigma_{\mathcal{RGS}}$ on m if $\sigma_{\mathcal{RGS}} \in [\mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})]$, $(msk_{(i,j)}, .) \in [\mathsf{J}_{\mathcal{RGS}}(ik_i, (i, j))]$, $(gpk_i, ik_i, .) \in [\mathsf{G}_{\mathcal{RGS}}(1^{\kappa_i})]$ and $gpk_i \in \boldsymbol{gpk}$; we say the player is a *traced* signer of $\sigma_{\mathcal{RGS}}$ on m with respect to $\boldsymbol{gpk}$ if $\mathsf{Ju}_{\mathcal{RGS}}(\boldsymbol{gpk}, (i, j), reg_{(i,j)}, \sigma_{\mathcal{RGS}}, m, \tau_{(i,j)}) = 1$ for any given values of $reg_{(i,j)}$ and $\tau_{(i,j)}$.
- We say that a value of $msk_{(i,j)}$ is a *true* group membership secret key with respect to $\boldsymbol{gpk}$ if $(msk_{(i,j)}, .) \in [\mathsf{J}_{\mathcal{RGS}}(ik_i, (i, j))]$, $(gpk_i, ik_i, .) \in [\mathsf{G}_{\mathcal{RGS}}(1^{\kappa_i})]$ and $gpk_i \in \boldsymbol{gpk}$. We say that the value $msk_{(i,j)}$ is a *valid* group membership secret key with respect to $\boldsymbol{gpk}$ if $\mathsf{V}_{\mathcal{RGS}}(\boldsymbol{gpk}, m, \mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})) = 1$ for any m.

3.3 Signer-anonymity

This property extends the GS *anonymity* property of [6] by involving multiple groups and allowing the adversary to adaptively choose groups. The property is defined with the following signer-anonymity experiment, run between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ and denoted by $\mathbf{Exp}^{\mathsf{anon-b}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$. In the experiment, $\mathcal{A}$'s goal is to determine which one out of the two members generated a signature.

In the experiment, $\mathcal{A}$ can make the following queries to $\mathcal{C}$.

- Group setup. $\mathcal{A}$ requests for creating a new group i in one of the two cases: (1) $\mathcal{C}$ runs $(gpk_i, ik_i, ok_i) \xleftarrow{\text{R}} \mathsf{G}_{\mathcal{RGS}}(1^{\kappa_i})$ and sends gpk_i and ik_i to $\mathcal{A}$; (2) $\mathcal{A}$ suggests the values of (gpk_i, ik_i, ok_i) to $\mathcal{C}$.
- Join. $\mathcal{A}$ requests for creating a new member (i, j) in either cases: (1) $\mathcal{C}$ runs the join protocol locally $(msk_{(i,j)}, reg_{(i,j)}) \leftarrow \mathsf{J}_{\mathcal{RGS}}(ik_i, (i, j))$; (2) $\mathcal{C}$ as the member runs the protocol with $\mathcal{A}$ as the issuer.
- Sign. $\mathcal{A}$ requests a signature on a message m for a member (i, j) created in a Join query and $\boldsymbol{gpk}$ set to the groups which were created in a set of Group setup queries. $\mathcal{C}$ computes $\sigma \leftarrow \mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})$ and returns σ to $\mathcal{A}$.
- Signer corrupt. $\mathcal{A}$ requests (i, j)'s membership secret key, and $\mathcal{C}$ responds with $msk_{(i,j)}$.
- Opener corrupt. $\mathcal{A}$ requests the opener secret key of an existing group i, and $\mathcal{C}$ responds with ok_i.
- Open. $\mathcal{A}$ requests for opening a signature σ. $\mathcal{C}$ runs the open algorithm by using all the ok_i values in $\mathcal{C}$'s possession. If one outputs $((i, j), \tau_{(i,j)}/\varepsilon)$ and the remaining outputs $\perp$, $\mathcal{C}$ returns $((i, j), \tau_{(i,j)}/\varepsilon)$ to $\mathcal{A}$; otherwise $\mathcal{C}$ returns $\perp$.

For challenging, $\mathcal{A}$ outputs a message m^* and two members $(i,j)_0$ and $(i,j)_1$ which were created from two Join queries. $\mathcal{C}$ chooses a random $b \leftarrow \{0,1\}$ (or takes the existing experiment parameter b if available), computes $\sigma \leftarrow \mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)_b}, m^*, \boldsymbol{gpk})$, and sends σ^* to $\mathcal{A}$.

After the challenge phase, $\mathcal{A}$ can ask additional queries to $\mathcal{C}$ as before the challenge phase.

Throughout the experiment, $\mathcal{A}$ has not made a signer corrupt query on either $(i,j)_0$ or $(i,j)_1$, an opener corrupt query or a group setup query with an opening key suggested by $\mathcal{A}$ on any group associated with $(i,j)_0$ or $(i,j)_1$, nor an open query on σ^*. Furthermore, $\mathcal{A}$ is not allowed to assign a group key already belonging to one group to another group.

Finally, $\mathcal{A}$ outputs a bit b'. $\mathcal{A}$ wins if $b' = b$.

We denote the advantage of adversary $\mathcal{A}$ in breaking the signer-anonymity of $\mathcal{RGS}$ by

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{anon}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa}) = \\ &\qquad \Pr[\mathbf{Exp}^{\mathsf{anon-1}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa}) = 1] - \Pr[\mathbf{Exp}^{\mathsf{anon-0}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa}) = 0],\end{aligned}$$

We say that $\mathcal{RGS}$ is signer-anonymous if for any polynomial-time adversary $\mathcal{A}$, the function $\mathbf{Adv}^{\mathsf{anon}}_{\mathcal{RGS},\mathcal{A}}(.)$ is negligible (as a function of the minimum security parameter $\kappa = min(\kappa_0, ..., \kappa_{n-1})$).

3.4 Signer-traceability

This property extends the GS *traceability* property of [6] by involving multiple groups and allowing the adversary to adaptively choose groups. The property is defined with the following traceability experiment, denoted by $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$ and run between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, where $\mathcal{A}$'s goal is to forge a valid signature under the conditions that either $\mathcal{A}$ does not have a key or the signature cannot be opened properly.

In the experiment, $\mathcal{A}$ can make the following queries to $\mathcal{C}$.

- Group setup. $\mathcal{A}$ requests creating a new group i. $\mathcal{C}$ runs $(gpk_i, ik_i, ok_i) \xleftarrow{\mathrm{R}} \mathsf{G}_{\mathcal{RGS}}(1^{\kappa_i})$ and returns gpk_i and ok_i.
- Join. $\mathcal{A}$ requests for creating a new member (i,j) in one of the following two cases: (1) $\mathcal{C}$ runs the join protocol as the issuer with $\mathcal{A}$ as the member; (2) $\mathcal{C}$ runs the join protocol locally.
- Sign. $\mathcal{A}$ requests a signature on a message m for (i,j) created in Case (2) of the join query, and $\boldsymbol{gpk}$ chosen from the existing groups. $\mathcal{C}$ computes $\sigma \leftarrow \mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})$ and returns σ to $\mathcal{A}$.

– Signer corrupt. $\mathcal{A}$ requests the membership secret key of a member (i,j), which was created in Case (2) of the join query. $\mathcal{C}$ responds with $msk_{(i,j)}$.

Finally, $\mathcal{A}$ outputs a message m^* and a signature σ^*. $\mathcal{A}$ wins if $\mathsf{V}_{\mathcal{RGS}}(m^*, \boldsymbol{gpk}, \sigma^*) = 1$, $\mathsf{O}_{\mathcal{RGS}}(ok_i, reg_i, m^*, \boldsymbol{gpk}, \sigma^*) = ((i,j), \tau_{(i,j)})$, and $\mathsf{Ju}_{\mathcal{RGS}}(\boldsymbol{gpk}, (i,j), reg_{(i,j)}, \sigma^*, m^*, \tau_{(i,j)}) = 1$, with the conditions that (i,j) was created in Case 2 of a Join query, $\mathcal{A}$ has not asked the signer corrupt query on (i,j), and $\mathcal{A}$ did not obtain σ^* by making a sign query on m^* for (i,j); or if $\mathsf{V}_{\mathcal{RGS}}(m^*, \boldsymbol{gpk}, \sigma^*) = 1$, but either $\mathsf{O}_{\mathcal{RGS}}$ returns $\perp$ or $\mathsf{Ju}_{\mathcal{RGS}}$ returns 0. We define the advantage of adversary $\mathcal{A}$ in defeating signer-traceability of $\mathcal{RGS}$ by

$$\mathbf{Adv}^{\mathsf{trace}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa}) = \Pr[\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa}) = 1].$$

We say that $\mathcal{RGS}$ is signer-traceable if for any polynomial-time adversary $\mathcal{A}$, the function $\mathbf{Adv}^{\mathsf{anon}}_{\mathcal{RGS},\mathcal{A}}(.)$ is negligible in the minimum security parameter $\kappa = min(\kappa_0, ..., \kappa_{n-1})$.

3.5 Signer-non-frameability

This property extends the GS *non-frameability* property of [6] by involving multiple groups and allowing the adversary to adaptively choose groups. The property is defined with the signer-non-frameability experiment, denoted by $\mathbf{Exp}^{\mathsf{nf}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$. It is run between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows.

In the experiment, $\mathcal{A}$ can make the following queries to $\mathcal{C}$.

– Group setup. $\mathcal{A}$ requests for creating a new group i and suggests the values of (gpk_i, ik_i, ok_i) to $\mathcal{C}$.
– Join. $\mathcal{A}$ requests for creating a new member (i,j). $\mathcal{C}$ runs the join protocol as the member with $\mathcal{A}$ as the issuer.
– Sign. $\mathcal{A}$ requests a signature on a message m for (i,j) created in a Join query and $\boldsymbol{gpk}$ both at $\mathcal{A}$'s choice. $\mathcal{C}$ computes $\sigma \leftarrow \mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})$ and returns σ.
– Signer corrupt. $\mathcal{A}$ requests the membership secret key of (i,j) created in a Join query. $\mathcal{C}$ responds with $msk_{(i,j)}$.

Finally, $\mathcal{A}$ outputs a message m^*, a signature σ^*, an identity $(i,j)^*$, a register $reg_{(i,j)}$ and a proof $\tau^*_{(i,j)}$. Assume that $(i,j)^*$ was created in a Join query, and $\mathcal{A}$ did neither corrupt the signer $(i,j)^*$ nor obtain σ^* from making a Sign query on m^* with $(i,j)^*$. $\mathcal{A}$ wins the experiment if $\mathsf{V}_{\mathcal{RGS}}(m^*, \boldsymbol{gpk}, \sigma^*) = 1$ and $\mathsf{Ju}_{\mathcal{RGS}}(\boldsymbol{gpk}, (i,j)^*, reg_{(i,j)*}, \sigma^*, m^*, \tau^*_{(i,j)}) = 1$. We define the advantage of adversary $\mathcal{A}$ in defeating signer-non-frameability of $\mathcal{RGS}$ by

$$\mathbf{Adv}^{\mathsf{nf}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa}) = \Pr[\mathbf{Exp}^{\mathsf{nf}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa}) = 1].$$

We say that $\mathcal{RGS}$ is signer-non-frameable if for any polynomial-time adversary $\mathcal{A}$, the function $\mathbf{Adv}^{\mathsf{nf}}_{\mathcal{RGS},\mathcal{A}}(.)$ is negligible in the minimum security parameter $\kappa = min(\kappa_0, ..., \kappa_{n-1})$.

3.6 Group-ambiguity

This property ensures that a signature is not bound to identity of the group which its true signer belongs to. More formally, we say $\mathcal{RGS}$ is perfectly group ambiguous if, given a triple $(\boldsymbol{gpk}, m, \sigma_{\mathcal{RGS}})$ for any $m \in \{0,1\}^*$, any $n, l_i, \kappa_i \in \mathbb{N}$, any $\boldsymbol{gpk}$ with $|\boldsymbol{gpk}| = n$, any $i \in [n]$, any $gpk_i \in \boldsymbol{gpk}$, any $(gpk_i, gsk_i) \in [\mathsf{G}_{\mathcal{RGS}}(1^{\kappa_i})]$ s.t. $gsk_i = (ik_i, ok_i)$, any $j \in [l_i]$, any $msk_{(i,j)} \in [\mathsf{J}_{\mathcal{RGS}}(ik_i, (i,j))]$, any $\sigma_{\mathcal{RGS}} \in [\mathsf{S}_{\mathcal{RGS}}(msk_{(i,j)}, m, \boldsymbol{gpk})]$, any unbound adversary $\mathcal{A}$ outputs k such that $gsk_k = gsk_i$ with probability exactly $1/n$.

Definition 1. *An RGS scheme following the syntax of RGS schemes is secure if it satisfies the five properties: correctness, signer-anonymity, signer-traceability, signer-non-frameability and group-ambiguity.*

4 A generic RGS construction

We now introduce a generic construction of a RGS scheme, which extends the 3-move type RS scheme of Abe et al. [1] by replacing the underlying Schnorr signature with a generic 3-move type GS scheme. We first briefly describe a generic construction of 3-move type GS schemes and an overview of the Abe et al RS scheme, and then demonstrate how to use these two building blocks to create the RGS construction.

4.1 A 3-move type GS scheme

The generic 3-move type GS construction is based on the concept of 3-move type signature schemes, which are from 3-move honest verifier zero-knowledge proofs, i.e. the standard Σ-protocol. Most of the well-known GS schemes belong to this type, such as [2, 8, 10, 14–16, 25, 29].

In the generic construction of a GS scheme, the algorithms and protocols of $\mathsf{G}_{\mathcal{GS}}$, $\mathsf{J}_{\mathcal{GS}}$, $\mathsf{O}_{\mathcal{GS}}$ $\mathsf{Ju}_{\mathcal{GS}}$ are mechanism specific and do not require special functionalities to form the generic construction. We will therefore omit the details of them in the following description, and focus on the signing algorithm $\mathsf{S}_{\mathcal{GS}}$ and the verification algorithm $\mathsf{V}_{\mathcal{GS}}$. In the next section, we will provide a concrete RGS scheme with fully details. For simplicity, we will also omit the subscript i from msk_i.

The signing algorithm $\mathsf{S}_{\mathcal{GS}}$ consists of four functions, say T, A, H and Z, to provide a proof of two level commitment. The verification algorithm $\mathsf{V}_{\mathcal{GS}}$ consists of two functions, say V and H, to verify these two level commitment. The abstract of these two algorithms is described in Figure 1.

Signing	Verification
$\mathsf{S}_{\mathcal{GS}}(gpk, msk, m) =$	$\mathsf{V}_{\mathcal{GS}}(m, gpk, \sigma) =$
$\boldsymbol{t} \leftarrow \{0,1\}^{\ell_t}$, Parse $(msk,\ \boldsymbol{t})$ as $\boldsymbol{x}$, $\boldsymbol{e} \leftarrow T(\boldsymbol{x})$	Parse $\sigma_{\mathcal{GS}}$ as $(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{s})$
$\boldsymbol{r} \leftarrow \{0,1\}^{\ell_r}, \boldsymbol{a} \leftarrow A(\boldsymbol{x}, \boldsymbol{r})$	$\boldsymbol{a}' \leftarrow V(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{s}, gpk)$
$\boldsymbol{c} \leftarrow H(m, \boldsymbol{e}, \boldsymbol{a})$	$\boldsymbol{c}' \leftarrow H(m, \boldsymbol{e}, \boldsymbol{a}')$
$\boldsymbol{s} \leftarrow Z(\boldsymbol{x}, \boldsymbol{r}, \boldsymbol{c})$	Return 1 if $\boldsymbol{c}' = \boldsymbol{c}$
Return $\sigma_{\mathcal{GS}} = (\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{s})$	Return 0 otherwise

Fig. 1. The construction of a 3-move type GS scheme

Given a message m, a group public key gpk which was generated in $\mathsf{G}_{\mathcal{GS}}$, and a group membership secret msk which was created in $\mathsf{J}_{\mathcal{GS}}$, $\mathsf{S}_{\mathcal{GS}}$ first generates a set of random strings, written as a vector $\boldsymbol{t}$, to be used to randomize msk and to encrypt msk under gpk, in order to guarantee that the value of msk is not revealed (anonymity) but the identity associated with msk can be extracted by running $\mathsf{O}_{\mathcal{GS}}$ (traceability), and then interprets $(msk, \boldsymbol{t})$ as a vector of secret values, say $\boldsymbol{x}$, in which to commit. The function T takes as input $\boldsymbol{x}$ and outputs the first level commitment, say $\boldsymbol{e}$ written as a vector of public strings. Subsequently $\boldsymbol{e}$ and $\boldsymbol{x}$ serve as an ephemeral public and private key pair for a Σ-protocol, which is frequently referred to as a signature-based knowledge proof denoted by $\mathcal{SPK}\{(\boldsymbol{x}) : \boldsymbol{e} = T(\boldsymbol{x})\}m$.

The functions A, H and Z are used in each stage of the Σ-protocol. A generates the second level commitment $\boldsymbol{a}$ based on the second randomness $\boldsymbol{r}$. H is a hash function $\{0,1\}^* \to \Delta$, which takes the message m and the first and second commitment $\boldsymbol{e}$ and $\boldsymbol{a}$ as input and outputs a challenge, for generality, which is also written as a vector of strings $\boldsymbol{c}$. Z generates a response $\boldsymbol{s}$ to the challenge. $\mathsf{S}_{\mathcal{GS}}$ returns the three vectors $(\boldsymbol{e},\ \boldsymbol{c},\ \boldsymbol{s})$ as the signature $\sigma_{\mathcal{GS}}$.

Given a candidate signature $\sigma_{\mathcal{GS}}$, $\mathsf{V}_{\mathcal{GS}}$ first interprets it as $(\boldsymbol{e},\ \boldsymbol{c},\ \boldsymbol{s})$. The function V, as a mirror of the function A, takes $\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{s}$ and gpk as input and outputs $\boldsymbol{a}'$. If the output of H, say $\boldsymbol{c}'$, is the same as $\boldsymbol{c}$, implying $\boldsymbol{a}'$ is equal to $\boldsymbol{a}$, then $\mathsf{V}_{\mathcal{GS}}$ outputs 1; otherwise outputs 0.

Remark that for the purpose of security proof, it is required that all secrets, including *gsk*, *msk*, and randomness $\boldsymbol{t}$ and $\boldsymbol{r}$, must distribute uniformly over certain spaces. It is also required that the functions T and Z must guarantee their outputs uniformly distributed if their inputs uniformly distribute; therefore, if $\boldsymbol{x}$ distributes uniformly over $\{0,1\}^{\ell_{\boldsymbol{x}}}$, then $\boldsymbol{e}$ created by T also distributes uniformly over $\{0,1\}^{\ell_{\boldsymbol{e}}}$; if $\boldsymbol{r}$ distributes uniformly over $\{0,1\}^{\ell_{\boldsymbol{r}}}$, then $\boldsymbol{s}$ from Z also distributes uniformly over $\{0,1\}^{\ell_{\boldsymbol{s}}}$. The above conditions are satisfied by many well-known group signature schemes, e.g. [2, 8, 10]. More formally speaking, except for the four general properties, thus correctness, anonymity, traceability and non-frameability, which by following the security notions of [6] any GS scheme must satisfy, we require that the 3-move type GS schemes must hold two extra properties, namely *collision* and *simulatability*, defined as follows.

Definition 2. *(Collision property) There exists a polynomial time algorithm that computes $\boldsymbol{x}$ from $\boldsymbol{e}$, $\boldsymbol{c}$, $\boldsymbol{s}$, $\boldsymbol{c}$', $\boldsymbol{s}$' and gpk, where $(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{s})$ and $(\boldsymbol{e}, \boldsymbol{c}', \boldsymbol{s}')$ are two unequal valid group signatures that correspond to the same $(m, \boldsymbol{e}, \boldsymbol{a})$ given to the hash function H.*

This property is from the collision property of 3-move signature schemes [1], and is also called the rewinding property in the literature, based on the Forking Lemma. This property is frequently used for proving the security of 3-move type signature schemes, such as the Schnorr signature scheme [32] and a number of its variants.

Definition 3. *(Simulatability in the random oracle model) A group signature scheme, $\mathcal{GS}$, is $(\tau, \epsilon, q_j, q_s,\ q_h)$-simulatable in the random oracle model if for any group key (gpk, ik, ok) $\in [\mathsf{G}_{\mathcal{GS}}(1^{\kappa})]$ and for any algorithm $\mathcal{A}$ that accesses to the random oracle H at most q_h times, the join protocol $\mathsf{J}_{\mathcal{GS}}(gsk)$ at most q_j times and the signing algorithm $\mathsf{S}_{\mathcal{GS}}(msk)$ at most q_s times, there exists a triple of interactive machines, $\mathcal{M}_{sim} = (\mathsf{J}_{sim}, \mathsf{S}_{sim}, \mathsf{H}_{sim})$, that interacts with $\mathcal{A}$ in such a way that the total running time is at most τ, and statistical distance of the probability distribution of $\mathsf{view}_{\mathcal{A}}(gpk,\ \mathsf{J}_{\mathcal{GS}}(ik, .),\ \mathsf{S}_{\mathcal{GS}}(., msk, .),\ H)$ and $\mathsf{view}_{\mathcal{A}}(gpk,\ \mathcal{M}_{sim}(gpk))$ is at most ϵ. Here, the probability is taken over all coin flips of $\mathsf{G}_{\mathcal{GS}}$, $\mathsf{J}_{\mathcal{GS}}$, $\mathsf{S}_{\mathcal{GS}}$, H, $\mathcal{M}_{sim}$, and $\mathcal{A}$.*

Observe that a simulator in control of the random oracle, $H : \{0,1\}^* \rightarrow \Delta$, can forge a group signature $\sigma_{\mathcal{GS}} = (\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{s})$ by choosing $\boldsymbol{c} \leftarrow \Delta$, $\boldsymbol{x} \leftarrow \{0,1\}^{\ell_{\boldsymbol{x}}}$ and $\boldsymbol{s} \leftarrow \{0,1\}^{\ell_{\boldsymbol{s}}}$, and computing $\boldsymbol{e} \leftarrow T(\boldsymbol{x})$. Obviously the vectors $\boldsymbol{e}$ and $\boldsymbol{s}$

created from this simulation distribute uniformly over $\{0,1\}^{\ell_e}$ and $\{0,1\}^{\ell_s}$ respectively. This forging operation will be used to create our generic RGS construction in the next subsection. This property is from the simulatability of an ordinary signature scheme in the random oracle model [1]. The definition of this property can be generalized to deal with multiple oracles for group signature schemes that involves multiple hash functions if necessary. Simulatability is featured in the unforgeability property against adaptive chosen massage attacks (EUF-CMA) signature schemes such as in the Schnorr signature scheme.

4.2 The AOS RS scheme

This scheme is called all discrete-log scheme in [1]. Let $\boldsymbol{pk}$ be an n-vector of public keys, written as $\boldsymbol{pk} = (pk_0, ..., pk_{n-1})$, where for each $i \in [n]$, $pk_i = (y_i, p_i, q_i, g_i)$ is a dis-log setting public key such that p_i and q_i are large primes, g_i is a generator of a prime subgroup of $\mathbb{Z}^*_{p_i}$ with order q_i, and $y_i = g^{x_i} \bmod p_i$ and let $x_i \in \mathbb{Z}^*_{q_i}$ be the corresponding private key sk_i. Let each user i have such a pair of private and public keys (sk_i, pk_i). Suppose the user k ($k \in [n]$) is the *true* signer, the abstract of the $\mathsf{S}_{\mathcal{RS}}$ and $\mathsf{V}_{\mathcal{RS}}$ algorithms of the AOS RS scheme is described in Figure 2.

Signing
$\mathsf{S}_{\mathcal{RS}}(sk_k, m, \boldsymbol{pk}) =$
$r_k \leftarrow \{0,1\}^{\ell_{q_k}}, a_k \leftarrow g_k^{r_k} \bmod p_k$
$c_{k+1} \leftarrow H_{k+1}(m, \boldsymbol{pk}, a_k)$
For $i = k+1, ..., n-1, 0, 1, ..., k-1$
$\{s_i \leftarrow \{0,1\}^{\ell_{q_i}}, a_i \leftarrow g_i^{s_i} y_i^{c_i} \bmod p_i$
$j = i+1 \bmod n, c_j \leftarrow H_j(m, \boldsymbol{pk}, a_i)\ \}$
$s_k \leftarrow r_k - x_k \cdot c_k \bmod q_k$
Return $\sigma_{\mathcal{RS}} = (c_0, s_0, ..., s_{n-1})$

Verification
$\mathsf{V}_{\mathcal{RS}}(m, \boldsymbol{pk}, \sigma_{\mathcal{RS}}) =$
Parse σ as $(c_0, s_0, ..., s_{n-1})$
For $i = 0, 1, ..., n-1$
$\{a'_i \leftarrow g_i^{s_i} y_i^{c'_i} \bmod p_i)$
$j = i+1 \bmod n$
$c'_j \leftarrow H_j(m, \boldsymbol{pk}, a'_i)\}$
Return 1 if $c'_0 = c_0$
Return 0 otherwise

Fig. 2. The AOS RS scheme

As proved in [1], the above RS scheme holds the properties of unconditional signer-ambiguity and existential unforgeability against adaptive chosen massage and chosen public-key attacks; see [1] for the details of these properties.

4.3 A 3-move type RGS scheme

We recall the special requirements of RGS schemes: except $\boldsymbol{gpk}$ any other information about any group, such as the values of l_i, reg_i, (i, j) and $\tau_{(i,j)}$, is not accessible to the other groups in $\mathcal{R}$, and the judger (if exists) is a trusted authority and out side of $\mathcal{R}$. If these requirements do not meet, an adversary may maliciously take a member from another group as a *valid* but not *true* signer.

As described in Section 3.1, an RGS scheme consists of ($\mathsf{G}_{\mathcal{GS}}$, $\mathsf{J}_{\mathcal{RGS}}$, $\mathsf{S}_{\mathcal{RGS}}$, $\mathsf{V}_{\mathcal{RGS}}$, $\mathsf{O}_{\mathcal{RGS}}$, $\mathsf{Ju}_{\mathcal{RGS}}$), in which each group uses the $\mathsf{G}_{\mathcal{GS}}$, $\mathsf{J}_{\mathcal{GS}}$, $\mathsf{O}_{\mathcal{GS}}$ and $\mathsf{Ju}_{\mathcal{GS}}$ protocols and algorithms from their underlying GS schemes to achieve $\mathsf{G}_{\mathcal{RGS}}$, $\mathsf{J}_{\mathcal{RGS}}$, $\mathsf{O}_{\mathcal{RGS}}$ and $\mathsf{Ju}_{\mathcal{RGS}}$ respectively. A membership secret key for the j-th member of the i-th group is denoted by $msk_{(i,j)}$. In this section, for simplicity, we omit the letter j and use the single subscript i, where it does not cause confusion. In the following RGS scheme specification, the true signer is a member of group k and has the membership secret key msk_k, and he signs a given message m with regard to a list of n group public keys, $\boldsymbol{gpk} = (gpk_0, gpk_1, ..., gpk_{n-1})$ where $k \in [n]$.

Suppose each group has a 3-move type GS scheme, as described before. For the i-th group, the signing algorithm $\mathsf{S}_{\mathcal{GS}}$ consists of four functions, written as T_i, A_i, H_i and Z_i, to achieve a proof of two-level commitment, written as $\boldsymbol{e}_i$ and $\boldsymbol{a}_i$; the verification algorithm $\mathsf{V}_{\mathcal{GS}}$ consists of two functions, written as V_i and H_i; and the group signature is written as $\sigma_{\mathcal{GS}_i} = (\boldsymbol{e}_i, \boldsymbol{c}_i, \boldsymbol{s}_i)$. By following the approach used in the AOS RS scheme as shown in Appendix 4.2, an RGS scheme arranges n group signatures, one from each group, to form a ring by shifting each challenge string $\boldsymbol{c}_i$ to the next position, i.e., $\boldsymbol{c}_i = \boldsymbol{c}_{i-1 \bmod n}$. With this manipulation, the signer can forge $n-1$ group signatures except the one of his own group. The abstract of the signing and verification algorithms in such an RGS scheme is shown in Figure 3.

SIGNING
$\mathsf{S}_{\mathcal{RGS}}(msk_k, m, \boldsymbol{gpk}) =$
$\boldsymbol{t}_k \leftarrow \{0,1\}^{\ell_{t_k}}$, Parse $(msk_k, \boldsymbol{t}_k)$ as $\boldsymbol{x}_k$, $\boldsymbol{e}_k \leftarrow T_k(\boldsymbol{x}_k)$
$\boldsymbol{r}_k \leftarrow \{0,1\}^{\ell_{r_k}}, \boldsymbol{a}_k \leftarrow A_k(\boldsymbol{x}_k, \boldsymbol{r}_k)$
$\boldsymbol{c}_{k+1} \leftarrow H_{k+1}(m, \boldsymbol{gpk}, \boldsymbol{e}_k, \boldsymbol{a}_k)$
For $i = k+1, ..., n-1, 0, 1, ..., k-1$
{ $\boldsymbol{x}_i \leftarrow \{0,1\}^{\ell_{x_i}}, \boldsymbol{e}_i \leftarrow T_i(\boldsymbol{x}_i)$
$\boldsymbol{s}_i \leftarrow \{0,1\}^{\ell_{s_i}}$
$\boldsymbol{a}_i \leftarrow V_i(\boldsymbol{e}_i, \boldsymbol{c}_i, \boldsymbol{s}_i, gpk_i)$
$j = i+1 \bmod n, \boldsymbol{c}_j \leftarrow H_j(m, \boldsymbol{gpk}, \boldsymbol{e}_i, \boldsymbol{a}_i)$ }
$\boldsymbol{s}_k \leftarrow Z_k(\boldsymbol{x}_k, \boldsymbol{r}_k, \boldsymbol{c}_k)$
Return $\sigma_{\mathcal{RGS}} = (\boldsymbol{e}_0, ..., \boldsymbol{e}_{n-1}, \boldsymbol{c}_0, \boldsymbol{s}_0, ..., \boldsymbol{s}_{n-1})$

VERIFICATION
$\mathsf{V}_{\mathcal{RGS}}(m, \boldsymbol{gpk}, \sigma) =$
Parse $\sigma_{\mathcal{RGS}}$ as
$(\boldsymbol{e}_0, ..., \boldsymbol{e}_{n-1}, \boldsymbol{c}_0, \boldsymbol{s}_0, ..., \boldsymbol{s}_{n-1})$
For $i = 0, 1, ..., n-1$
{
$\boldsymbol{a}'_i \leftarrow V_i(\boldsymbol{e}_i, \boldsymbol{c}_i, \boldsymbol{s}_i, gpk_i)$
$j = i+1 \bmod n$
$\boldsymbol{c}'_j \leftarrow H_j(m, \boldsymbol{gpk}, \boldsymbol{e}_i, \boldsymbol{a}'_i)$
}
Return 1 if $\boldsymbol{c}'_0 = \boldsymbol{c}_0$
Return 0 otherwise

Fig. 3. The generic construction of a RGS scheme

4.4 Security of the 3-move type RGS scheme

In this subsection, we discuss that any RGS scheme with the above 3-move construction satisfies correctness, signer-anonymity, signer-traceability, signer-non-frameability and group-ambiguity, as defined in Section 3 and the following theorems hold. For each theorem, we give a sketch of proof.

Theorem 1. *The above RGS scheme is correct, if all the underlying GS schemes are correct 3-move type GS schemes.*

Proof. The proof is straightforward. The correctness property means that honestly generated signatures verify and trace correctly. Regarding validation, $\sigma_{\mathcal{RGS}}$ contains n group signatures, each from one underlying GS scheme. The k-th signature is truly created so it is valid. The other $n-1$ signatures are simulated. Based on the simulatability of the 3-move type GS schemes in the random oracle model, these $n-1$ signatures are also valid. Obviously since all of the n signatures are valid, then the ring of the n signatures, $\sigma_{\mathcal{RGS}}$, must be valid as well. Regarding traceability, since the k-th signature is a true signature, the $\mathsf{O}_{\mathcal{GS}}$ algorithm of the k-th GS scheme will correctly recovers the identity of the signer. In the $\mathsf{O}_{\mathcal{RGS}}$ algorithm, each group opener follows the $\mathsf{O}_{\mathcal{GS}}$ algorithm of the underlying GS scheme, therefore the true signer can always be traced. The theorem follows. □

Theorem 2. *This RGS scheme is signer-anonymous if all the n underlying GS schemes are anonymous and all the functions H_i for $i \in [n]$ in the scheme are random oracles.*

Proof. (sketch) The theorem can be proved by the following reduction: if there is a polynomial adversary $\mathcal{A}$ who is able to break signer-anonymity of the RGS scheme, then $\mathcal{A}$ can be used by another polynomial algorithm $\mathcal{B}$ to break anonymity of one of the underlying GS schemes. Suppose $\mathcal{B}$ has the target to break the k-th underlying GS scheme for $k \in [n]$. To serve as a challenger $\mathcal{B}$ runs $\mathbf{Exp}^{\mathsf{anon-b}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$ with $\mathcal{A}$, and simultaneously, to serve as an adversary, $\mathcal{B}$ runs $\mathbf{Exp}^{\mathsf{anon-b}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$ with another challenger $\mathcal{C}$. By controlling the random oracles H_i, $\mathcal{B}$ can answer $\mathcal{A}$'s oracle queries properly.

$\mathcal{A}$ adoptively chooses two challenge identities $(i,j)_0$ and $(i,j)_1$ from the $M = \Sigma_{i\in[n]} l_i$ potential signers of the n groups; each signer has been assigned with $(i,j) \in ([n],[l_i])$ as identity. Suppose with some probability δ, $\mathcal{A}$ chooses both the two identities in the group k, such as $(i,j)_0 = (k, j_0)$ and $(i,j)_1 = (k, j_1)$ for $j_0, j_1 \in [l_k]$. $\mathcal{B}$ then takes these two signers on as his choice and sends j_0 and j_1 to $\mathcal{C}$ in the challenge phase of $\mathbf{Exp}^{\mathsf{anon-b}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$. At the end of $\mathbf{Exp}^{\mathsf{anon-b}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$, $\mathcal{B}$ receives $\mathcal{A}$'s response, say b', and $\mathcal{B}$ forwards it to $\mathcal{C}$ as his response to the challenge in $\mathbf{Exp}^{\mathsf{anon-b}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$. It is easy to see, if the advantage of $\mathcal{A}$ in breaking the signer-anonymity of the RGS scheme is ϵ, then the advantage of $\mathcal{B}$ in breaking the anonymity of the k-th underlying GS scheme is $\epsilon \cdot \delta$. δ is not negligible, as it depends on n and l_i. If ϵ is not negligible, this result is contradict to the assumption that all the underlying GS schemes are anonymous. The theorem follows. $\square$

Theorem 3. *The above RGS scheme is signer-traceable if all the n underlying GS schemes are traceable and all the functions H_i for $i \in [n]$ in the scheme are random oracles.*

Proof. (sketch) As similar to proving signer-anonymity, this theorem can be proved by the following reduction: if there is a polynomial adversary $\mathcal{A}$ who is able to break signer-traceability of the RGS scheme, then $\mathcal{A}$ can be used by another polynomial algorithm $\mathcal{B}$ to break traceability of one of the underlying GS schemes. Suppose $\mathcal{B}$ has the target to break the k-th underlying GS scheme for $k \in [n]$, which means $\mathcal{B}$ outputs a forged GS signature $\sigma_{\mathcal{GS}} = (\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{s})$, which will verify and trace as a *true* GS signature. To serve as a challenger $\mathcal{B}$ runs $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$ with $\mathcal{A}$, and simultaneously, to serve as an adversary, $\mathcal{B}$ runs $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$ with another challenger $\mathcal{C}$. Recall that $\mathcal{B}$ is allowed to control the random oracles H_i in $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$. When $\mathcal{B}$ receives the H_k query from $\mathcal{A}$ with the input $(m, \boldsymbol{gpk}, \boldsymbol{e}_{k-1}, \boldsymbol{a}_{k-1})$, $\mathcal{B}$ outputs $\boldsymbol{c}_k = H'_k(m, \boldsymbol{e}_k, \boldsymbol{a}_k)$, where H'_k is the k-th H function of the GS scheme, and the values of $(m, \boldsymbol{e}_k, \boldsymbol{a}_k)$ are from $\mathcal{A}$'s H_{k+1} query. The value of $\boldsymbol{c}_k$ is obtained in $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$ either from $\mathcal{C}$ or computed by $\mathcal{B}$ itself. $\mathcal{B}$ maintains the consistence of all the H_i outputs of the RGS scheme as usual. At the end of $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$, $\mathcal{B}$ receives $\sigma_{\mathcal{RGS}} = (\boldsymbol{e}_0, ..., \boldsymbol{e}_{n-1}, \boldsymbol{c}_0, \boldsymbol{s}_0, ..., \boldsymbol{s}_{n-1})$ from $\mathcal{A}$.

Suppose $\mathcal{A}$ chooses the forged *true* signer randomly from one of the n groups associated with $\boldsymbol{gpk}$. With the probability of $1/n$, $\sigma_{\mathcal{RGS}}$ traces to a member of the group k. In that case, $\mathcal{B}$ sends $(\boldsymbol{e}_k, \boldsymbol{c}_k, \boldsymbol{s}_k)$ to $\mathcal{C}$ as his input to $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$. In the case that $\mathcal{A}$ outputs a signature which is valid but which cannot be opened, $\mathcal{B}$ can still use $\mathcal{A}$'s result in his $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$ with $\mathcal{C}$ in the same way, since $\mathcal{A}$'s output includes at least one underlying group signature which is valid but cannot be opened. Since $\mathcal{A}$'s operation is a random selection, with the probability of $1/n$, this is from the k-th group.

As a result, if the advantage of $\mathcal{A}$ in breaking the signer-traceability of the RGS scheme is ϵ, then the advantage of $\mathcal{B}$ in breaking the traceability of the k-th underlying GS scheme is ϵ/n. If ϵ is not negligible, this result is contradict to the assumption that all the underlying GS schemes are traceable. The theorem follows. $\square$

Theorem 4. *This RGS scheme is signer-non-frameable, if all the n underlying GS schemes are non-frameable and all the functions H_i for $i \in [n]$ in the scheme are random oracles.*

Proof. (sketch) As similar to proving signer-anonymity, this theorem can be proved by the following reduction: if there is a polynomial adversary $\mathcal{A}$ who is able to break signer-non-frameability of the RGS scheme, then $\mathcal{A}$ can be used by another polynomial algorithm $\mathcal{B}$ to break non-frameability of one of the underlying GS schemes. Suppose $\mathcal{B}$ has the target to break the k-th underlying GS scheme for $k \in [n]$, which means $\mathcal{B}$ outputs a forged GS signature $\sigma_{\mathcal{GS}} = (\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{s})$, which will verify and trace as a *true* GS signature. To serve as a challenger $\mathcal{B}$ runs $\mathbf{Exp}^{\mathsf{nf}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$ with $\mathcal{A}$, and simultaneously, to serve as an adversary, $\mathcal{B}$ runs $\mathbf{Exp}^{\mathsf{nf}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$ with another challenger $\mathcal{C}$. Recall that $\mathcal{B}$ is allowed to control the random oracles H_i in $\mathbf{Exp}^{\mathsf{nf}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$. When $\mathcal{B}$ receives the H_k query from $\mathcal{A}$ with the input $(m, \boldsymbol{gpk}, \boldsymbol{e}_{k-1}, \boldsymbol{a}_{k-1})$, $\mathcal{B}$ outputs $\boldsymbol{c}_k = H'_k(m, \boldsymbol{e}_k, \boldsymbol{a}_k)$, where H'_k is the k-th H function of the GS scheme, and the values of $(m, \boldsymbol{e}_k, \boldsymbol{a}_k)$ are from $\mathcal{A}$'s H_{k+1} query. The value of $\boldsymbol{c}_k$ is obtained in $\mathbf{Exp}^{\mathsf{nf}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$ either from $\mathcal{C}$ or computed by $\mathcal{B}$ itself. $\mathcal{B}$ maintains the consistence of all the H_i outputs of the RGS scheme as usual. At the end of $\mathbf{Exp}^{\mathsf{nf}}_{\mathcal{RGS},\mathcal{A}}(\boldsymbol{\kappa})$, $\mathcal{B}$ receives $\sigma_{\mathcal{RGS}} = (\boldsymbol{e}_0, ..., \boldsymbol{e}_{n-1}, \boldsymbol{c}_0, \boldsymbol{s}_0, ..., \boldsymbol{s}_{n-1})$ from $\mathcal{A}$.

Suppose $\mathcal{A}$ chooses the forged *true* signer randomly from one of the n groups associated with $\boldsymbol{gpk}$. With the probability of $1/n$, $\sigma_{\mathcal{RGS}}$ traces to a member of the group k. In that case, $\mathcal{B}$ sends $(\boldsymbol{e}_k, \boldsymbol{c}_k, \boldsymbol{s}_k)$ to $\mathcal{C}$ as his input to $\mathbf{Exp}^{\mathsf{nf}}_{\mathcal{GS},\mathcal{A}}(\kappa_k)$. As a result, if the advantage of $\mathcal{A}$ in breaking the signer-non-frameability of the RGS scheme is ϵ, then the advantage of $\mathcal{B}$ in breaking the non-frameability of the k-th underlying GS scheme is ϵ/n. If ϵ is not negligible, this result is contradict to the assumption that all the underlying GS schemes are non-frameable. The theorem follows. □

Theorem 5. *The above RGS scheme is unconditionally group ambiguous, if all the n underlying GS schemes are secure 3-move type GS schemes.*

Proof. (sketch) The idea of proving this theorem is to argue that all the values in $\sigma_{\mathcal{RGS}}$ distribute uniformly with respect to $\boldsymbol{gpk}$ and regardless to the position of the group of the true signer, k. Observe that except for $\boldsymbol{s}_k$, all $\boldsymbol{s}_i$ are taken uniformly at random from $\{0,1\}^{\ell_{\boldsymbol{s}}}$; also except for $\boldsymbol{e}_k$, all $\boldsymbol{e}_i$ are computed by using T_i from $\boldsymbol{t}_i$ which are uniformly chosen from $\{0,1\}^{\ell_{\boldsymbol{t}_i}}$. As mentioned earlier, the function T_i keep the uniformly distribution property between their inputs and outputs, therefore, $\boldsymbol{e}_i$ must also distribute uniformly over $\{0,1\}^{\ell_{e_i}}$. The vectors $\boldsymbol{e}_k$ and $\boldsymbol{s}_k$ are computed by following the k-th underlying 3-move type GS scheme with a minor modification that takes a challenge string different from $\boldsymbol{c}_k$ in the original GS scheme. Since all $\boldsymbol{c}_i$ in the RGS scheme are outputs from the random oracle H_i, so this modification does not change the uniformly distribution property of $\boldsymbol{e}_k$ and $\boldsymbol{s}_k$. By following the specification of the above 3-move type GS scheme, $\boldsymbol{e}_k$ and $\boldsymbol{s}_k$ also distribute uniformly over $\{0,1\}^{\ell_e}$ and $\{0,1\}^{\ell_s}$ respectively. Therefore, for any fixed $(\boldsymbol{gpk}, m)$, $(\boldsymbol{e}_0, ..., \boldsymbol{e}_{n-1}, \boldsymbol{s}_0, ..., \boldsymbol{s}_{n-1})$ has $\prod_{i=0}^{n-1} 2^{\ell_{e_i}+\ell_{s_i}}$ variations that equally likely regardless of the position of k. Remaining $\boldsymbol{c}_0$ in a signature is determined uniquely from $(\boldsymbol{gpk}, m)$, $\boldsymbol{e}_i$'s and $\boldsymbol{s}_i$'s. As a result of this argument, even an unbound adversary cannot find the value k with a higher probability than randomly guessing. The theorem follows. □

Note that the unbound adversary can compute any group secret keys (ik_i, ok_i) from any group public keys gpk_i in $\boldsymbol{gpk}$, and also can retrieve any corresponding values of msk_i from a given ring group signature. It is important to see that although the adversary can compute all the *valid* membership secret keys of all possible group members, he is not able to tell which values are *true* group membership keys. On the other hand, in a real RGS scheme, for any valid result (i.e. the signer identity (i, j) and its proof $\tau_{(i,j)}$) from the group opening algorithm $\mathsf{O}_{\mathcal{RGS}}$, the issuer/opener of the i-th group cannot deny that the member (i, j) has actually joined the group, because every valid msk_i value belongs to a *true* group member.

5 A concrete RGS scheme

In this section, we introduce a concrete RGS scheme. For the underlying GS scheme, we make use of the GS scheme by Boneh et al. [8], which is a modification of their original short GS scheme by adding a joining protocol to allow each member to contribute to his own secret key. This modification is mentioned in Section 7 of [8]. Since their original GS scheme is called the SDH GS scheme, in this paper, we name this modification the SDH$^+$ GS scheme. Actually this is one of many possible choices, because any suitable 3-move type of GS schemes can be used to build our RGS scheme following the generic construction of $\mathcal{RGS}$ described in Section 4.3.

5.1 The SDH$^+$ GS scheme

Choose a bilinear group pair $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order p with a computable isomorphism ψ, where $g_1 = \psi(g_2)$ for g_1 and g_2 being the respective generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, and a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, such that the Strong Diffie-Hellman (SDH) assumption holds on $(\mathbb{G}_1, \mathbb{G}_2)$, and the Linear assumption holds on $\mathbb{G}_1$; we refer the reader to [8] for the definition of these two assumptions. Also choose a hash function $H : \{0,1\}^* \leftarrow \mathbb{Z}_p$, which is treated as a random oracle in the proof of security in [8].

– $\mathsf{G}_{\mathcal{GS}}(1^\kappa)$. Given a security parameter κ, it generates the group issuer's secret key $ik = \gamma$, the group opener's secret key $ok = (\xi, \zeta)$ and the group public key $gpk = (g_1, g_2, w,\ d, h, u, v)$, where $\gamma, \xi, \zeta \xleftarrow{\mathrm{R}} \mathbb{Z}_p^*$, $w = g_2^\gamma$, $d, h \xleftarrow{\mathrm{R}} \mathbb{G}_1 \backslash \{1_{\mathbb{G}_1}\}$, $u, v \in \mathbb{G}_1$ such that $u^\xi = v^\zeta = h$.

– $\mathsf{J}_{\mathcal{GS}}(\gamma, j)$. It registers each user $j \in [l]$ (the value l is the number of members in the group) and generates the user's membership secret key $msk_j = (f_j, \chi_j, A_j)$. During the joining process, the user j chooses $f_j \xleftarrow{\mathrm{R}} \mathbb{Z}_p^*$ and sends $F_j = d^{f_j} \in \mathbb{G}_1$ along with a proof of possession of f_j to the group issuer, who verifies the proof and then returns $\chi_j \xleftarrow{\mathrm{R}} \mathbb{Z}_p^*$ and $A_j \leftarrow (g_1 \cdot F_j)^{1/(\gamma+\chi_j)} \in \mathbb{G}_1$ back. The issuer records $reg_j = (F_j, \chi_j, A_j) \in reg$.

– $\mathsf{S}_{\mathcal{GS}}(gpk, msk_j, m)$. It creates the signature $\sigma = (\boldsymbol{e}, c, \boldsymbol{s})$ on $m \in \{0,1\}^*$ by performing the following steps:
  1. Choose a 2-vector of the first level random integers as $\boldsymbol{t} = (t_1, t_2) \xleftarrow{\mathrm{R}} \mathbb{Z}_p$.
  2. Set a 6-vector of ephemeral secret keys as $\boldsymbol{x} = (x_0, x_1, x_2, x_3, x_4, x_5)$ by $x_0 = f_j$, $x_1 = t_1$, $x_2 = t_2$, $x_3 = \chi_j$, $x_4 = \chi_j \cdot t_1$ and $x_5 = \chi_j \cdot t_2$.
  3. Compute a 3-vector of ephemeral public keys as $\boldsymbol{e} = (e_1, e_2, e_3)$ by $e_1 \leftarrow u^{x_1}$, $e_2 \leftarrow v^{x_2}$ and $e_3 \leftarrow A_j \cdot h^{x_1+x_2}$.
  4. Choose a 6-vector of the second level random integers as $\boldsymbol{r} = (r_0,\ r_1,\ r_2,\ r_3,\ r_4,\ r_5) \xleftarrow{\mathrm{R}} \mathbb{Z}_p$.
  5. Compute a 5-vector of commitment values as $\boldsymbol{a} = (a_1, a_2, a_3, a_4, a_5)$ by $a_1 \leftarrow u^{r_1}$, $a_2 \leftarrow v^{r_2}$, $a_3 \leftarrow \hat{e}(e_3,\ g_2)^{r_3} \cdot \hat{e}(d,\ g_2)^{-r_0} \cdot \hat{e}(h, w)^{-r_1-r_2} \cdot \hat{e}(h, g_2)^{-r_4-r_5}$, $a_4 \leftarrow e_1^{r_3} \cdot u^{-r_4}$ and $a_5 \leftarrow e_2^{r_3} \cdot v^{-r_5}$.
  6. Compute a challenge string as $c = H(m, \boldsymbol{e}, \boldsymbol{a})$.
  7. Compute a 6-vector of response values as $\boldsymbol{s} = (s_0, s_1, s_2, s_3, s_4, s_5)$ by $s_i = r_i + c \cdot x_i$ $(0 \le i \le 5)$.
  8. Output the signature $\sigma = (\boldsymbol{e}, c, \boldsymbol{s})$.

– $\mathsf{V}_{\mathcal{GS}}(gpk, \sigma, m)$. It verifies a candidate signature σ on a message m in the following operations:
  1. Parse σ as $(e_1, e_2, e_3, c, s_0, s_1, s_2, s_3, s_4, s_5)$.
  2. Compute a 5-vector of commitment values as $\boldsymbol{a}' = (a_1', a_2', a_3', a_4', a_5')$ by $a_1' \leftarrow u^{s_1} \cdot e_1^{-c}$, $a_2' \leftarrow v^{s_2} \cdot e_2^{-c}$, $a_3' \leftarrow \hat{e}(e_3, g_2)^{s_3} \cdot \hat{e}(d, g_2)^{-s_0} \cdot \hat{e}(h, w)^{-s_1-s_2} \cdot \hat{e}(h,\ g_2)^{-s_4-s_5} \cdot (\hat{e}(e_3, w)/\hat{e}(g_1, g_2))^c$, $a_4' \leftarrow e_1^{s_3} \cdot u^{-s_4}$ and $a_5' \leftarrow e_2^{s_3} \cdot v^{-s_5}$.
  3. Check whether $c = H(m, \boldsymbol{e}, \boldsymbol{a}')$ holds; if the check succeeds output 1 and otherwise 0.

– $\mathsf{O}_{\mathcal{GS}}(\xi, \zeta, reg, \sigma, m)$. It computes $A_j \leftarrow e_3/(e_1^\xi e_2^\zeta)$, retrieves $reg_j = (F_j, \chi_j, A_j)$ and outputs reg_j and the proof τ_j: $\mathcal{SPK}\{(\xi, \zeta) : e_3/A_j = e_1^\xi e_2^\zeta \wedge h = u^\xi = v^\zeta\}$.

– $\mathsf{Ju}_{\mathcal{GS}}(gpk, reg_j, \sigma, m, \tau_j)$. It verifies τ_j and $\hat{e}(A_j, w \cdot g_2^{\chi_j}) = \hat{e}(g_1 \cdot F_j, g_2)$; outputs 1 if the verification succeeds or 0 otherwise.

The reader is referred to [8] for security proof of the original SDH GS scheme, and [18] demonstrates that the modification in SDH$^+$ does not affect security of SDH.

5.2 The concrete RGS scheme

Suppose n groups are taken into a ring, and all of them support the SDH$^+$ GS scheme. Let $\{\boldsymbol{gpk}, \boldsymbol{l}\} = \{gpk_i, l_i\}_{\mathrm{i} \in [n]}$ be the lists of the n group public keys and sizes, where $gpk_i = (g_{1i}, g_{2i}, w_i, d_i, h_i, u_i, v_i)$ and its issuer secret key is γ_i and opener secret key is (ξ_i, ζ_i). The j-th member in the i-th group $((i, j) \in ([n], [l_i]))$ has a membership secret key $msk_{(i,j)} = (f_{(i,j)}, \chi_{(i,j)}, A_{(i,j)})$ and registration value $reg_{(i,j)} = (F_{(i,j)}, \chi_{(i,j)}, A_{(i,j)})$. These values are generated by using $\mathsf{G}_{\mathcal{GS}}$ and $\mathsf{J}_{\mathcal{GS}}$ as described in SDH$^+$ GS. More specifically, we have $f_{(i,j)}, \chi_{(i,j)} \in \mathbb{Z}_p^*$, $w_i = g_{2i}^{\gamma_i}, u_i^{\xi_i} = v_i^{\zeta_i} = h_i$, $F_{(i,j)} = d_i^{f_{(i,j)}}$, and $A_{(i,j)} = (g_{1i} \cdot F_{(i,j)})^{1/(\gamma_i + \chi_{(i,j)})}$.

To sign a message $m \in \{0,1\}^*$ with respect to $\boldsymbol{gpk}$, the j-th member of the k-th group (k, j) $(k \in [n])$ performs the following steps by running the algorithm $\mathsf{S}_{\mathcal{RGS}}((f_{(k,j)}, \chi_{(k,j)}, A_{(k,j)}), m, \boldsymbol{gpk})$:

1. Initiate the ring by following the steps 1 - 5 of $\mathsf{S}_{\mathcal{GS}}$ in SDH$^+$ GS to compute $\boldsymbol{t}_k$, $\boldsymbol{x}_k$, $\boldsymbol{e}_k$, $\boldsymbol{r}_k$ and $\boldsymbol{a}_k$.
2. Compute $c_{k+1} = H(m, \boldsymbol{gpk}, \boldsymbol{e}_k, \boldsymbol{a}_k)$.
3. For $i = k+1, ..., n-1, 0, 1, ..., k-1$, choose two random vectors $\boldsymbol{e}_i$ and $\boldsymbol{s}_i$ by $e_{zi} \xleftarrow{\mathrm{R}} \mathbb{Z}_{p_i}$ $(1 \le z \le 3)$ and $s_{zi} \xleftarrow{\mathrm{R}} \mathbb{Z}_{p_i}$ $(0 \le z \le 5)$, compute $\boldsymbol{a}_i = (a_{1i}, a_{2i}, a_{3i}, a_{4i}, a_{5i})$ by following the second step of $\mathsf{V}_{\mathcal{GS}}$ in SDH$^+$ GS, and compute $c_{i+1 \bmod n} = H_{i+1 \bmod n}(m, \boldsymbol{gpk}, \boldsymbol{e}_i, \boldsymbol{a}_i)$.
4. Close the ring by computing a 6-vector of response values as $\boldsymbol{s}_k = (s_{0k}, s_{1k}, s_{2k}, s_{3k}, s_{4k}, s_{5k})$, where $s_{zk} = r_{zk} + c_k \cdot x_{zk}$ $(0 \le z \le 5)$.
5. Output the signature $\sigma = (\boldsymbol{e}_0, ..., \boldsymbol{e}_{n-1}, c_0, \boldsymbol{s}_0, ..., \boldsymbol{s}_{n-1})$.

To verify a candidate ring group signature σ, a verifier run the $\mathsf{V}_{\mathcal{RGS}}(m, \boldsymbol{gpk}, \sigma)$ algorithm as follows:

1. Parse σ as $(\boldsymbol{e}_0, ..., \boldsymbol{e}_{n-1}, c_0, \boldsymbol{s}_0, ..., \boldsymbol{s}_{n-1})$.
2. For $i = 0$ to $n-1$, compute a 5-vector of commitment values $\boldsymbol{a}_i'$ by following the second step of $\mathsf{V}_{\mathcal{GS}}$ in SDH$^+$ GS, and then $c'_{i+1 \bmod n} = H_{i+1 \bmod n}(m, \boldsymbol{gpk}, \boldsymbol{e}_i, \boldsymbol{a}_i')$.
3. Check whether $c_0' = c_0$ holds; output 1 if the check succeeds and 0 otherwise.

Both the $\mathsf{O}_{\mathcal{RGS}}$ and $\mathsf{Ju}_{\mathcal{RGS}}$ algorithms are identical to the $\mathsf{O}_{\mathcal{GS}}$ and $\mathsf{Ju}_{\mathcal{GS}}$ algorithms respectively in SDH$^+$ GS. The ring group signature can be traced to the *true* signer only by the opener in the right group. This result is based on the assumption that except $\boldsymbol{gpk}$ all the information of any group is not accessible to the other groups and the judger is a trust authority out side of $\mathcal{R}$. Otherwise, obviously a malicious user can insert a set of valid A values in the relevant positions of $\mathcal{R}$. This would cause the openers of all the relevant groups to output plausible identities. As mentioned before, this is a reasonable assumption for our target applications.

Theorem 6. *The concrete RGS scheme is secure, i.e. it is correct, signer-anonymous, signer-traceable, signer-non-frameable and group-ambiguous, under the random oracle model and the assumption that the underlying group signature scheme is secure.*

Proof. The RGS scheme is a 3-move type RGS scheme and has a single underlying GS scheme, the SDH$^+$ GS scheme. This underlying GS scheme is a secure 3-move type GS scheme, based on the security analysis given in [8]. Therefore, the theorem simply follows Theorems 1, 2, 3, 4 and 5. □

6 Conclusions

We proposed the concept of ring group signatures (RGSs) including a formal definition of its syntax and security. RGSs support a set of attractive properties: correctness, signer-anonymity, signer-traceability, signer-non-frameability and group-ambiguity. We presented a generic construction of RGS schemes and proved that any RGS scheme following this construction holds the five properties as long as the underlying GS schemes are secure. Our security analysis is under the random oracle model. The ring group signature solution will benefit any applications based on group signatures, in which group identities are sensitive.

References

1. M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n signatures from a variety of keys. In ASIACRYPT '02, LNCS 2501, pp. 415–432, 2002.
2. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO '00*, LNCS 1880, pp. 255–270, 2000.
3. G. Ateniese, D. X. Song, and G. Tsudik. Quasi-efficient revocation in group signatures. In *6th Financial Cryptography*, LNCS 2357, pp. 183–197, 2002.

4. M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen. Constant-size id-based linkable and revocable-iff-linked ring signature. In *INDOCRYPT '06*, LNCS 4329, pp. 364–378, 2006.
5. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT '03*, LNCS 2656, pp. 614–629, 2003.
6. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA 2005*, LNCS 3376, pp. 136–153. 2005.
7. A. Bender, J. Katz, and R. Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *Journal of Cryptology*, 22(1):114–138, 2009.
8. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO '04*, LNCS 3152, pp. 41–55, 2004.
9. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregrate and verifiably encrypted signatures from bilinear maps. In *EUROCRYPT '03*, LNCS 2656, pp. 416–432. 2003.
10. D. Boneh and H. Shacham. Group signatures with verifier-local revocation. In *11th ACMCCS*, pp. 168–177, ACM Press, 2004.
11. E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In *CRYPTO '02*, LNCS 2442, pp. 465–480, 2002.
12. E. Brickell, J. Camenisch and L. Chen. Direct anonymous attestation. In *the 11th ACM Conference on Computer and Communications Security.* ACM Press, pp. 132–145, 2004.
13. J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO '02*, LNCS 2442, pp. 61–76, 2002.
14. J. Camenisch and M. Michels. A group signature scheme with improved efficiency. In *ASIACRYPT '98*, LNCS 1514, pp. 160–174, 1998.
15. J. Camenisch and M. Michels. Separability and efficiency for generic group signature schemes. In *CRYPTO '99*, LNCS 1666, pp. 413–430, 1999.
16. J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *CRYPTO '97*, LNCS 1296, pp. 410–424, 1997.
17. D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT '91*, LNCS 547, p. 257–265, 1991.
18. L. Chen. A DAA scheme requiring less TPM resources. In *Inscrypt 2009*, LNCS 6151, pp. 350-365, 2010. The full paper can be found at Cryptology ePrint Archive, Report 2010/008.
19. L. Chen. Ring group signatures. To appear in the Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom-2012).
20. L. Chen and J. Li. VLR group signatures with indisputable exculpability and efficient revocation. In *PASSAT'10*, pp. 727-734, IEEE Computer Society Press, 2010.
21. L. Chen, S.-L. Ng and G. Wang. Threshold anonymous announcement in VANETs. IEEE Journal on Selected Areas in Communications, 29(3):605-615, 2011.
22. S. Chow, V.-W. Wei, J. Liu, and T. Yuen. Ring signatures without random oracles. In *ASIACCS 2006*, pp. 297–302, ACM Press, 2006.
23. S. S. Chow, W. Susilo, and T. H. Yuen. Escrowed linkability of ring signatures and its applications. In *VIETCRYPT 2006*, LNCS 4341, pp.s 175–192, 2006.
24. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *EUROCRYPT '94*, LNCS 839, pp. 174–187, 1994.

25. J. Furukawa and H. Imai. An efficient group signature scheme from bilinear maps. *IEICE Transactions*, 89-A(5):1328–1338, 2006.
26. A. Kiayias and M. Yung. Group signatures with efficient concurrent join. In *EUROCRYPT '05*, LNCS 3494, pp. 198–214, 2005.
27. D. Y. W. Liu, J. K. Liu, Y. Mu, W. Susilo, and D. S. Wong. Revocable ring signature. *Journal of Computer Science and Technology*, 22(6):785–794, 2007.
28. J. K. Liu, V. K. Wei, and D. S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups (extended abstract). In *ACISP 2004*, LNCS 3108, pp. 325–335, 2004.
29. T. Nakanishi and N. Funabiki. Verifier-local revocation group signature schemes with backward unlinkability from bilinear maps. In *ASIACRYPT '05*, LNCS 3788, pp. 533–548, 2005.
30. P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung and J.-P. Hubaux. Secure vehicular communication systems: design and architecture. IEEE Commun. Mag., pp. 100109, November 2008.
31. R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT '01*, LNCS 2248, pp. 552–565, 2001.
32. C. P. Schnorr. Efficient identification and signatures for smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.
33. M. Trolin and D. Wikström. Hierarchical group signatures. In *ICALP 2005*, LNCS 3580, pp. 446–458, 2005.
34. P. P. Tsang, M. H. Au, J. K. Liu, W. Susilo, and D. S. Wong. A suite of non-pairing id-based threshold ring signature schemes with different levels of anonymity. To appear to Proceedings of ProvSec 2010.
35. Trusted Computing Group. TCG TPM specification 1.2. Available at `http://www.trustedcomputinggroup.org`, 2003.
36. F. Zhang and K. Kim. Id-based blind signature and ring signature from pairings. In *ASIACRYPT '02*, LNCS 2501, pp. 533–547, 2002.