Traffic Filtering and Routing in Partially-Hidden Networks

Deholo Nali, Ali Miri, Carlisle Adams
(Corresponding author: Deholo Nali)

School of Information Technology and Engineering, University of Ottawa
800 King Edward Avenue, P.O. Box 450, Station A, Ottawa, Ontario, Canada K1N 6N5
(Email: {deholo, samiri, cadams}@site.uottawa.ca)

(Received June 30, 2005; revised and accepted July 31, 2005)

Abstract

Suppose that an organization $\mathcal{O}$ wants to do the following: to reveal only part of its network map; to filter unwanted traffic passing through its network; and to route, to the right destination, legitimate traffic intended for hidden parts of its network. This would enable $\mathcal{O}$ to perform traffic filtering and routing with improved resistance against network mapping attacks. This paper proposes a provably-secure hierarchical identity-based group signature scheme which meets the above requirements, and is scalable to large, distributed, and hierarchically-structured networks.

Keywords: Group signature, hierarchical networks, identity-based cryptography, sender anonymity, traffic filtering

1 Introduction

Let $\mathcal{G}$ be a group of users composed of hierarchically-structured and non-overlapping subgroups. Hierarchical anonymous authentication (HAA) refers to the process whereby: (1) any third party can determine which subgroup of $\mathcal{G}$ a given user is part of; and (2) no one, except a designated member[1] of a user's subgroup, is able to determine the identity of this user. One application of HAA is *traffic filtering and routing in partially-hidden networks.*

Suppose, for instance, that a company $\mathcal{O}$ wants to achieve the following goals: (G1) $\mathcal{O}$ wants to reveal only part of its network map to third parties; (G2) $\mathcal{O}$ wishes to stop unwanted traffic passing through its network; and, (G3) when legitimate network traffic is intended for hidden parts of its network, $\mathcal{O}$ wants to route this traffic to the right destination nodes.

G1 may be motivated by the desire to achieve both increased privacy and improved resistance against network mapping attacks. (These attacks precede most harmful network-based attacks.) G2 is useful to combat denial of service attacks, and this goal could be achieved by stopping all network messages passing through $\mathcal{O}$'s network, except those which come from $\mathcal{O}$'s subnets and those which respond to recent messages coming from $\mathcal{O}$'s subnets. (Remark that, if $\mathcal{O}$'s network is virtual, then network messages coming from $\mathcal{O}$'s subnets may either enter or exit $\mathcal{O}$'s physical network. In particular, network messages coming from a mobile device belonging to $\mathcal{O}$ can send messages which virtually come from $\mathcal{O}$'s network, but physically come from another network.) Finally, G3 is essential to the use of networks. In particular, $\mathcal{O}$ could use a designated node of each one of its subnets in order to route messages intended for specific nodes of this subnet. Consequently, $\mathcal{O}$ could organize its network nodes into hierarchically-structured and non-overlapping subnets, and use HAA in order to achieve G1, G2 and G3. HAA is therefore useful for traffic filtering and routing in partially-hidden and hierarchically-structured networks.

1.1 A First Attempt to Support HAA

One attempt to support HAA is to use group signatures (GS)[2] [1, 14]. GS schemes are cryptographic primitives enabling each member of a given group to sign on behalf of the group, in such a way that only a designated group leader is able to identify the issuer of any given valid group signature. Thus, one can associate a group signature scheme with each subgroup of a group, and use these schemes to support HAA, when the subgroups are hierarchically structured. This methodology, however, has the following two limitations. First, a central trusted party must generate a distinct set of public parameters for each

[1] The designated member of this subgroup will henceforth be called the *subgroup leader*.

[2] Note that schemes which do not provide source authenticity (e.g. through digital signatures) are not suitable for HAA, since network messages can be spoofed (i.e. misattributed to legitimate sources).

subgroup. Ideally, there would be one small set of public parameters which could be used to verify signatures issued on behalf of all subgroups. Second, one central trusted party (associated with the large group) must both generate each one of the subgroup parameters, and secretly send secret parameters to each subgroup leader. When the hierarchy is large (or becomes large as subgroups are added), it is more efficient to have a scheme whereby a central entity dynamically delegates to other entities (e.g. subgroup leaders) the task of generating and distributing secret subgroup parameters. Hence, this attempt to support HAA is neither space efficient nor computationally efficient in the case of large hierarchies (e.g. hierarchies composed of 10,000 nodes and 500 subgroups). Consequently, a novel cryptographic primitive needs to be designed in order to support HAA in large networks.

1.2 Related Work

Scalable Cryptography. This paper is concerned with mechanisms which are scalable to large, distributed, and hierarchically-structured networks. While certificate-based public-key cryptography requires the distribution and management of public-key certificates, identity-based (ID-based) cryptography removes this requirement by allowing public keys to be derived from entity identifiers (e.g. email addresses, telephone numbers, or IP addresses). In large and distributed networks, ID-based cryptography is therefore more convenient than certificate-based cryptography. Moreover, a class of schemes called *hierarchical ID-based* (HIB) cryptosystems are suitable for hierarchically-structured networks. HIB cryptosystems enable a root private key generator (PKG) to delegate the generation and secure distribution of ID-based private keys to lower PKGs, which recursively do the same, in such a way that end-nodes are authenticated and securely obtain their private keys. ID-based cryptography was introduced by Shamir [34], and Horwitz and Lynn [22] described the first hierarchical ID-based encryption (HIBE) scheme. Since then, various HIBE and hierarchical ID-based signature (HIBS) schemes have been proposed [5, 6, 17, 20, 25, 29].

Hierarchical Group Signatures and Related Primitives. Trolin and Wickstrōm [35] recently presented the first formal treatment of hierarchical group signature (HGS). Their work focuses on groups whose structure (i.e. the number and identity of subgroups) is hidden to third parties. (This class of HGS scheme is henceforth called *HGS-Class-1* schemes, while HGS schemes supporting HAA are called *HGS-Class-2* schemes.) Trolin and Wickstrōm describe a provably secure *HGS-Class-1* scheme, but the scheme is suboptimal (for reasonable security parameters, producing group signature requires a few hundred exponentiations). Group signatures for hierarchical multigroups had already been considered by Kim et al. [26], but the scalability of Kim et al.'s scheme is hampered by the fact that both group leaders and signers must register in a central authoritative party before signers are able to issue group signatures. Moreover, Kim et al.'s scheme does not feature a hierarchical identification procedure for group leaders, which makes the scheme non adequate for *HGS-Class-1* applications. Anonymous authentication schemes were presented by Brickell et al. [10], featuring identity escrow (i.e. a designated party is able, under special circumstances, to identity the issuer of anonymous authenticating tokens). However, these schemes do not efficiently handle hierarchically structured groups (as HGS schemes would). Anonymous authentication schemes with subset queries were also proposed (cf. [18, 19, 33]). These schemes enable an authenticating party to determine whether the issuer of a given authenticating token belongs to a subset of the system users. Such a subset may be the subset of another one – thereby yielding a hierarchical setting. However, [18, 19, 33] do not provide identity escrow. Boneh and Franklin [7] presented the first anonymous authentication scheme with subset queries, identity escrow, and signer revocation capability. However, the work required by provers (e.g. signers) and verifiers, in this scheme, is linear with respect to the number of subsets. This is suboptimal. For instance, when subsets are structured as a rooted tree, the work required by provers may be expected to be constant, and the work required by verifiers to be linear with respect to the depth of a specific subset (i.e. logarithmic in the total number of subsets). This is what our proposed scheme achieves. (Table 1 presents an overview of the aforementioned schemes' limitations.)

1.3 Contributions

The aim of this paper is twofold. First, we seek to formalize the security of *HGS-Class-2* schemes. (Note that the formal security of *HGS-Class-1* schemes has already been investigated by Trolin and Wickstrōm [35], in the framework of certificate-based cryptography.) Second, we seek to devise a provably secure *HGS-Class-2* scheme.

We propose a hierarchical ID-based group signature (HIBGS) scheme, with the following group signature properties. First, the scheme allows each signature verifier to use a locally kept list of revoked-user tokens, in order to determine the revocation status of signers, at signature verification time. (Group signatures having this feature are said have *Verfier-Local Revocation* (VLR) capability [9].) Second, the scheme enables subgroup leaders to co-generate their subgroup members' keys, but prevents subgroup leaders from being able to sign on behalf of their subgroup members. This stands in contrast with Boneh and Shacham's recent group signature scheme [9]. Third, the proposed scheme provides a mechanism for signers to be able to reuse the same private key to generate multiple group signatures (unlike Chen et al.'s ID-based GS scheme [16].) Fourth, the scheme describes a signature verification procedure whose computational cost is only logarithmic with respect to the number $|SGL|$ of subgroups in a given hierarchy. This improves Boneh and Franklin's anonymous authentication scheme with sub-

Table 1: Limitations of previous work related to HIBGS

Category	**Limitations**
Group Signature [1, 2, 3, 9, 11, 12, 13, 14, 15, 16, 23, 24, 28, 30, 31, 32, 36]	No hierarchical key generation
(*HGS-Class-1*) Hierarchical Group Signature [35]	Very expensive signing and verifying Does not support HAA
Hierarchical ID-based Signature [5, 6, 17, 20, 25, 29]	No signer anonymity
Group Signature for Hierarchical Subgroups [26]	Mandatory Central Registration for group leaders and signers Not suitable for *HGS-Class-1* applications
Anonymous Authentication with ID Escrow [10]	Not efficient for hierarchically structured groups
Anonymous Authentication with Subset Queries [18, 33, 19]	No identity escrow
Anonymous Authentication with Subset Queries and ID Escrow [7]	Not efficient for hierarchically structured groups

set queries [7], whose signing and signature verification costs grow linearly with respect to $|SGL|$. Fifth, the suggested scheme is provably secure in the the random oracle model [4], assuming the intractability of the bilinear Diffie-Hellman problem (see Section 2.) Sixth, the proposed scheme enables signature verifiers to assess the validity of group signatures, without the need to obtain certified copies of group public keys. This stands in contrast to all certificate-based group signature schemes, including [7, 9].

Additionally, our proposed scheme has the following features, as a *hierarchical* GS scheme. First, it prevents the ancestors of any subgroup leader gl from impersonating gl in her interactions with existing members of gl's subgroup. This is achieved by requiring each subgroup leader to run confidential and authenticated communications with her subgroup members, based on a secret which is unique to the pair (subgroup member, subgroup leader). Second, it is a *HGS-Class-2* scheme, and thereby supports hierarchical anonymous authentication. Third, it is scalable, in the sense that it enables subgroup leaders to generate (on their own) their descendants' private keys. This stands in contrast with Kim et al.'s multi-group scheme [26], which requires both subgroup leaders and signers to register with a central trusted party. Fourth, it is efficient, featuring constant-length keys and a constant-time signing procedure.

Noteworthy is the fact that our HIBGS scheme features constant-size signatures (even in the hierarchical setting). This property was obtained from the key generation algorithm of a related hierarchical ID-based encryption scheme [29], whose description is beyond the scope of this paper. However, let us underscore two steps and two conditions required to obtain our proposed HIBGS scheme, from this HIBE scheme.

◇ Leader (i.e. subgroup leader) keys had to be modified in such a way that they are different from subgroup member's signing keys. This is due to the fact that signing keys should be co-generated with subgroup members (so that leaders are not able to sign on behalf of subgroup members), while subgroup leaders should be able to generate the full key of each one of their children (i.e. sub-leaders). This difference between sub-leader keys and signer keys also implies that a new algorithm had to be designed for the generation of signing keys. Avoiding member impersonation by key-issuing leaders is not a trivial task: one must describe a mechanism which guarantees that any given signature was issued with a subgroup member's secret, without revealing this secret, yet showing that the secret is bound to the key given by the subgroup leader to the signer.

◇ The user key generation procedure had to be redesigned in such a way that one signing key enables its intended user to issue multiple group signatures. Note that this feature cannot be achieved using Chen et al's [16] approach, since that approach requires each user to obtain a signing key for each issued group signature. Our approach is different, as shown in §3.

◇ The new key generation algorithm needed to be modified in such a way that it remains provably secure, in the random oracle model.

◇ Keys of the resulting scheme also had to remain of constant length, and signatures had to remain of constant size.

For our proposed scheme, it was also necessary to devise a threat model for *HGS-Class-2* schemes, and to prove the security of our proposed scheme with respect to this model. To devise this model, we extended Boneh and Schacham's security model for group signature schemes with VLR capability in such a way that attempts of leaders to sign on behalf of their subgroup members are taken into consideration.

The sequel is organized as follows. Section 2 presents the number theoretic assumptions on which the security of our proposed HIBGS scheme is based. Section 3 describes our proposed HIBGS scheme, and Section 4 discusses its computational and space requirements in comparison with related schemes. Section 5 summarizes the security guarantees of our scheme, and Section 6 concludes the paper.

2 Number Theoretic Assumptions

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two Abelian groups of prime order q, where $\mathbb{G}_1$ is additive and $\mathbb{G}_2$ is multiplicative. Let $P_0^{(1)} \in \mathbb{G}_1^*$ be a generator of $\mathbb{G}_1$. A *Bilinear pairing* $\hat{e}$ is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ such that $\hat{e}(aP_0^{(1)}, bP_0^{(1)}) = \hat{e}(P_0^{(1)}, P_0^{(1)})^{ab}$ for all $a, b \in \mathbb{Z}_q^*$. The map $\hat{e}$ is said to be an *admissible pairing* if it is a *non-degenerate, computable Bilinear pairing*. Let $\mathcal{A}$ be an attacker modelled as a probabilistic Turing machine. The *computational Diffie-Hellman* (*CDH*) problem [8] is that in which $\mathcal{A}$ is to compute $abP_0^{(1)}$, given $(\mathbb{G}_1, q, P_0^{(1)}, aP_0^{(1)}, bP_0^{(1)})$ and a security parameter k, where $a, b \in \mathbb{Z}_q^*$ are unknown. The *decisional Diffie-Hellman* (*DDH*) problem [8] is that in which $\mathcal{A}$ is to guess whether $cP_0^{(1)} = abP_0^{(1)}$, given $(\mathbb{G}_1, q, P_0^{(1)}, aP_0^{(1)}, bP_0^{(1)}, cP_0^{(1)})$ and a security parameter k, where $a, b, c \in \mathbb{Z}_q^*$ are unknown. $\mathbb{G}_1$ is called a *Gap-Diffie-Hellman group* if the CDH is intractable in $\mathbb{G}_1$, but the the DDH can be solved in polynomial time in $\mathbb{G}_1$. The *Bilinear Diffie-Hellman* (*BDH*) problem [8] is that in which $\mathcal{A}$ is to compute $\hat{e}(P_0^{(1)}, P_0^{(1)})^{abc}$ given a security parameter k, the tuple $(\mathbb{G}_1, q, P_0^{(1)}, aP_0^{(1)}, bP_0^{(1)}, cP_0^{(1)})$ where $a, b, c \in \mathbb{Z}_q^*$ are unknown, and given the fact that the *CDH* problem cannot be solved in polynomial time with non-negligible advantage in both $\mathbb{G}_1$ and $\mathbb{G}_2$. The *Double Discrete Logarithm* (*DDL*) problem [12] is that in which $\mathcal{A}$ is to compute an unknown $c \in \mathbb{Z}_q^*$ given both $(a^c P_0^{(1)}, a, P_0^{(1)})$. Konama et al. [27] proved that the DDL problem is as at least as hard as the discrete logarithm (DL) problem modulo a prime.

3 Proposed HIBGS Scheme

Since the design of a HIBGS scheme is complex, Section 3.1 presents an overview of our HIBGS scheme, while the actual scheme is described in Section 3.2. (Important symbols used in the proposed scheme are presented in Table 2.)

3.1 Overview

Principals. The following parties are involved in our scheme: one root leader (*rLead*), one set of $|SGL|$ subgroup leaders, $|SGL|$ sets of subgroup members, one set of signature verifiers (i.e. third parties who can verify HGS signatures), and one arbitrator (i.e. a trusted third party). The subgroups are organized in a tree-shaped hierarchy, and each subgroup is identified by the ID-tuple of its subgroup leader. For instance, any second-level subgroup is identified by an ID-tuple of the form $\overline{ID}_2 = (ID_1, ID_2)$, where $\overline{ID}_1 = (ID_1)$ refers to the ID-tuple of a first-level subgroup leader. Moreover, each member of a subgroup is identified by an indexed ID-tuple whose prefix is the ID-tuple of the subgroup. For example, the i^{th} member of the $\overline{ID}_2$ is identified by $\overline{ID}_2[i]$.

Key Generation. The proposed HIBGS scheme has three key generation algorithms. These are the *Root Setup* algorithm, the *Leader-Key Generation* algorithm, the *User-Key Generation* algorithm. $\diamond$ The *Root Setup* algorithm defines various secret and public parameters which are used by others methods of the scheme. $\diamond$The *Leader-Key Generation* algorithm is used by each subgroup leader to generate the private key of her children subgroups (i.e. the subgroups placed directly below the leader's subgroup). The private key of each t^{th}-level subgroup leader is a tuple $(S_{\overline{ID}_t}, E_{\overline{ID}_t}, \widetilde{S}_{\overline{ID}_t}, R_{\overline{ID}_{t+1}}, \widetilde{R}_{\overline{ID}_{t+1}}, s_t)$, where the first five entries are used to issue signing keys and the last entry is unique to the t^{th} hierarchical level. Moreover, the first five entries of a subgroup leader's private key are generated with random elements $\alpha_{\overline{ID}_t}$ and $\widetilde{\alpha}_{\overline{ID}_t}$ which make the private key unique to the branch and level of the corresponding subgroup. Furthermore, the first five entries of the a subgroup leader's private key are generated in such a way that: none can be derived from the others; none can be derived from the private key of a subgroup leader of the same level; and none can be derived from the private key of a subgroup leader or a subgroup member located below the current subgroup leader. These conditions imply that subgroup leaders located on different branches of a subgroup hierarchy cannot derive one another's private keys. $\diamond$ The third key generation algorithm of the proposed HIBGS scheme is the *User-Key Generation* algorithm. This procedure is used by each subgroup member (say $\overline{ID}_t[i]$) and her subgroup leader to co-generate $\overline{ID}_t[i]$'s private key. This process is initiated by $\overline{ID}_t[i]$ who generates a key $mgt_{\overline{ID}_t[i]}$ and hides it from $\overline{ID}_t$ by raising a known basis to the exponent $mgt_{\overline{ID}_t[i]}$. This hiding process yields a membership token $mt_{\overline{ID}_t[i]}$ which $\overline{ID}_t[i]$ gives to $\overline{ID}_t$. $\overline{ID}_t$ then verifies (in zero knowledge) that $\overline{ID}_t[i]$ knows $mgt_{\overline{ID}_t[i]}$, and computes both a revocation token and $\overline{ID}_t[i]$'s private key. The revocation token $rt_{\overline{ID}_t[i]}$ is given the value of $mt_{\overline{ID}_t[i]}$, but $rt_{\overline{ID}_t[i]}$ is not sufficient to identify $\overline{ID}_t[i]$. $\overline{ID}_t[i]$'s private key has the form $(S_{\overline{ID}_t[i]}, E_{\overline{ID}_t[i]}, \widetilde{S}_{\overline{ID}_t[i]}, \widetilde{E}_{\overline{ID}_t[i]}, s_{t+1}, \gamma_{\overline{ID}_t[i]})$, where the first four entries have the same properties as $\overline{ID}_t$'s private key entries, and where the last entry must be used by $\overline{ID}_t[i]$ when she issues group signatures.

Signing. The signing algorithm is composed of three steps. The first step consists of using the signer's hidden key $mgt_{\overline{ID}_t[i]}$ in order to obtain a new version of $S_{\overline{ID}_t[i]}$, $E_{\overline{ID}_t[i]}$, and $\widetilde{S}_{\overline{ID}_t[i]}$ (namely $\underline{S}_{\overline{ID}_t[i]}$, $\underline{E}_{\overline{ID}_t[i]}$, and $\underline{\widetilde{S}}_{\overline{ID}_t[i]}$). These newly computed values are bound to $mgt_{\overline{ID}_t[i]}$. The second step of the signature algorithm consists of using $\gamma_{\overline{ID}_t}$ to set up a zero knowledge proof that $\overline{ID}_t[i]$ knows $\gamma_{\overline{ID}_t[i]}$. This proof is randomized (using the variable r) in such a way that third parties cannot determine whether two group signatures are issued by the same user. The third step of the signing algorithm mimics the *Leader-Key Generation* procedure, but uses a one-way function of the signed message in order to generate values which

Table 2: Important symbols used in the HIBGS scheme

$\mathcal{M}$	Message Space	$mgt_{\overline{ID}_t[i]}$	$ID_t[i]$'s membership generating token
$\mathcal{S}$	Signature Space	$mt_{\overline{ID}_t}$	$\overline{ID}_t[i]$'s membership token
$\mathbb{G}_i$	Scheme's i^{th} Abelian group	$rt_{\overline{ID}_t[i]}$	$\overline{ID}_t[i]$'s revocation token
$\mathcal{H}_i$	Scheme's i^{th} hash function	$d_{\overline{ID}_t}$	$\overline{ID}_t$'s secret key
$P_0^{(i)}$	Scheme's i^{th} generator of $\mathbb{G}_1$	$d_{\overline{ID}_t[i]}$	$\overline{ID}_t[i]$'s secret key
s_0	Root Leader's secret key	$\gamma_{\overline{ID}_t[i]}$	$\overline{ID}_t[i]$'s membership checking key
$L_t^{(i)}$	Scheme's i^{th} t^{th}-level Public parameter	SGL	Set of all subgroups forming a large hierarchy
$\overline{ID}_t$	Subgroup leader's identifier	$RL_{\overline{ID}_t}\|$	Set of revocation tokens associated with $\overline{ID}_t$'s subgroup
$\overline{ID}_t[i]$	Identifier of $\overline{ID}_t$'s i^{th} subgroup member	I_m	$\mathcal{H}_4(m)$, where m is a message
J_0	Tuple of the form (I_0, I_0, I_0)	J_i	Tuple of the form $(I_i, p_{(i,1)}, p_{(i,3)})$ if $i \geq 1$

demonstrate that the signer has the required private key.

Signature Verification. The signature verification algorithm proceeds in two steps. In the first step, the verifiers computes the public key of the claimed signer (i.e. $A_{t+1}P_0^{(1)} + B_{t+1}P_0^{(2)}$ and $C_{t+1}P_0^{(1)} + D_{t+1}P_0^{(2)}$), and binds this public key with a one-way function of the signed message (thereby obtaining $\rho_t^{(1)}$ and $\rho_t^{(2)}$). Then, in the second step, the verifier determines whether the signer used coherent values (by checking whether $\hat{e}(U_1 + U_2, P_0^{(5)} + P_0^{(6)}) = \hat{e}(U_5 + U_6, P_0^{(1)} + P_0^{(2)})$ and $\hat{e}(U_1' + U_2', P_0^{(5)} + P_0^{(6)}) = \hat{e}(U_5' + U_6', P_0^{(1)} + P_0^{(2)})$), had the right signing key (by checking whether $\hat{e}(L_{t+2}^{(4)}, \rho_t^{(1)})\hat{e}(P_0^{(3)}, \widetilde{V}) = \hat{e}(P_0^{(4)}, V)\hat{e}(L_{t+2}^{(3)}, \rho_t^{(2)})$), had a hidden key that matches with his private key (by checking whether $\mathcal{H}_2(\varpi||(W_2^{\mathcal{H}_3(W_1)})^{\theta}U_1||\chi^{\theta}U_2') = \kappa$), and is not revoked (by checking whether $U_1' \neq rt_{\overline{ID}_t[j]}U_1$ and $(\beta - \gamma_{\overline{ID}_t[j]})U_1' = rt_{\overline{ID}_t[j]}P_0^{(1)}$, where $rt_{\overline{ID}_t[j]}$ is any revocation token given to the verifier by $\overline{ID}_t$).

Opening. The opening algorithm essentially consists of the very last part of the verification algorithm. Since the opener is the subgroup leader of a given signer, this leader is able to link a given revocation token with the identity (index) of each member of her subgroup.

Arbitrating. The arbitrating procedure is executed by a trusted third party (TTP). While the TTP does not have to hold any secret piece of information, it is trusted to follow to the following line of reasoning: since the private key used to issue a group signature is bound with the membership token of a signer, and since the revocation token of a user is equal to the membership token of this user, then the possession of a valid group signature which matches with a given revocation token uniquely identifies the corresponding user. Thus, the author of group signature can be identified.

3.2 Scheme

- `Instance Generator` (k). This procedure, denoted by IG, is a randomized algorithm which takes a security parameter $k > 0$, runs in $O(k)$ steps, and outputs $(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, SG)$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are two Abelian Gap-Diffie-Hellman groups of prime order $q \geq 2^k$, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is an admissible pairing with respect to which the BDH problem is intractable, and SG is a prime order cyclic subgroup of $\mathbb{Z}_q^*$ in which the DL problem is intractable.

- `Root Setup` (k). Given a security parameter $k > 0$, the root PKG:

  1) runs IG with input k and obtains ($\mathbb{G}_1$, $\mathbb{G}_2$, $\hat{e}$, SG).
  2) picks, randomly and uniformly[3], $P_0^{(1)}$, $P_0^{(2)}$, $P_0^{(3)}$, $P_0^{(4)}$, $P_0^{(5)}$, $P_0^{(6)} \in \mathbb{G}_1$;
  3) picks $g \in_R SG$, and $s_0, I_0 \in_R \mathbb{Z}_q^*$, and sets $d_{\overline{ID}_0} = (s_0)$;
  4) computes $n = poly_1(k)$, $\ell = \log(k)$, $n_\tau = poly_2(k)$, and $n_{gs} = poly_3(k)$, where $poly_i$ is a polynomial over the positive integers, for $i = 1, 2, 3$;
  5) chooses cryptographic hash functions: $\mathcal{H}_1 : \{0,1\}^* \to (\mathbb{Z}_q^*)^{3\ell}$, $\mathcal{H}_2 : \{0,1\}^* \to \mathbb{Z}_q^*$, $\mathcal{H}_3 : \{0,1\}^{n_\tau} \to \mathbb{Z}_q^*$, $\mathcal{H}_4 : \{0,1\}^n \to \mathbb{Z}_q^*$, $\mathcal{H}_5 : \mathbb{Z}_q^* \to \mathbb{Z}_q^*$, where the image through $\mathcal{H}_1$ of a (t-long) ID-tuple $\overline{ID}_t = (ID_1, \cdots, ID_t)$ is $\mathcal{H}_1(\overline{ID}_t) = (J_1, \cdots, J_t, J_0, \cdots, J_0) \in (\mathbb{Z}_q^*)^{3\ell}$, where: $J_0 = (I_0, I_0, I_0)$; $J_i = (I_i, p_{(i,1)}, p_{(i,3)})$ for $1 \leq i \leq t$; $I_i = \mathcal{H}_6(ID_i), p_{(i,1)} = \mathcal{H}_7(I_i), p_{(i,3)} = \mathcal{H}_7(p_{(i,1)})$ for $1 \leq i \leq t$; $\mathcal{H}_6$ is any cryptographic hash function from $\{0,1\}^*$ to $\mathbb{Z}_q^*$, and $\mathcal{H}_7$ is any cryptographic hash function from $\mathbb{Z}_q^*$ to $\mathbb{Z}_q^*$.
  6) computes, $(s_i = \mathcal{H}_5(s_{i-1}))_{i=1}^{\ell-1}$, $L_1^{(j)} = s_0 P_0{}^{(j)}$ for $j = 3, 4$, and $(L_i^{(j)} = s_{i-1}L_{i-1}^{(j)})_{i=2}^{\ell}$ for $j = 3, 4$.

  The message space is $\mathcal{M} = \{0,1\}^n$ and the signature space is $\mathcal{S} = \mathbb{G}_1^6 \times \{0,1\}^{n_\tau} \times \mathbb{Z}_q^{*5}$. The system's public parameters (which must be certified) are $pubParams = (g, q, n, \hat{e}, I_0, P_0^{(1)}, P_0^{(2)}, P_0^{(3)}, P_0^{(4)}, P_0^{(5)}, P_0^{(6)}, \mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \mathcal{H}_4, \mathcal{H}_5, ((L_i^{(j)})_{i=1}^{\ell})_{j=3}^{4})$. The *root PKG* keeps s_0 secret, so that $params = (pubParams, s_0)$.

- `Sub-Leader-Key Generation` $(ID_{t+1}, d_{\overline{ID}_t})$: For each child-leader $\overline{ID}_{t+1} = (ID_1, \cdots, ID_{t+1})$ of a leader $\overline{ID}_t$, the following takes place:

[3] In the sequel, we shall use the notation $x \in_R X$ to indicate that the element x is chosen uniformly at random from the set X.

1) $\overline{ID}_t$ picks $\alpha_{\overline{ID}_{t+1}}, \widetilde{\alpha}_{\overline{ID}_{t+1}} \in_R \mathbb{Z}_q^*$.

2) – If $t = 0$, then $rLead$ computes $J_1 = (I_1, p_{(1,1)}, p_{(1,3)})$ (with ID_1 and $\mathcal{H}_1$), $p_{(1,2)} = \mathcal{H}_7(p_{(1,3)})$, $p_{(1,4)} = \frac{p_{(1,2)}p_{(1,3)}}{p_{(1,1)}}$, $I'_1 = -p_{(1,2)}$, $I''_1 = -\frac{p_{(1,1)}}{p_{(1,2)}}I'_1, \widetilde{I}_1 = I_1 + p_{(1,1)}, \widetilde{I}'_1 = I'_1 + p_{(1,2)}, \widetilde{I}''_1 = I''_1 + p_{(1,4)}$, $R_{\overline{ID}_1} = p_{(1,1)}\alpha_{\overline{ID}_1} + p_{(1,2)}\widetilde{\alpha}_{\overline{ID}_1}$, $\widetilde{R}_{\overline{ID}_1} = p_{(1,3)}\alpha_{\overline{ID}_1} + p_{(1,4)}\widetilde{\alpha}_{\overline{ID}_1}$, $s_1 = \mathcal{H}_5(s_0)$, and the following:

$$\begin{aligned}
S_{\overline{ID}_1} &= s_0(I_1P_0^{(1)} + I'_1P_0^{(2)} + \alpha_{\overline{ID}_1}P_0^{(3)}),\\
E_{\overline{ID}_1} &= s_0(\frac{I_1I''_1}{I'_1}P_0^{(1)} + I''_1P_0^{(2)} + \widetilde{\alpha}_{\overline{ID}_1}P_0^{(3)}),\\
\widetilde{S}_{\overline{ID}_1} &= s_0(\alpha_{\overline{ID}_1}P_0^{(4)} + \widetilde{I}_1P_0^{(5)} + \widetilde{I}'_1P_0^{(6)}),\\
\widetilde{E}_{\overline{ID}_1} &= s_0(\widetilde{\alpha}_{\overline{ID}_1}P_0^{(4)} + \frac{\widetilde{I}_1\widetilde{I}''_1}{\widetilde{I}'_1}P_0^{(5)} + \widetilde{I}''_1P_0^{(6)});
\end{aligned}$$

– Otherwise (i.e. if $t \geq 1$), $\overline{ID}_t$ computes: $J_t = (I_t, p_{(t,1)}, p_{(t,3)})$ and $J_{t+1} = (I_{t+1}, p_{(t+1,1)}, p_{(t+1,3)})$ (with ID_t, ID_{t+1} and $\mathcal{H}_1$); and

$$\begin{aligned}
I'_{t+1} &= -p_{(t,2)},\\
I''_{t+1} &= -\frac{p_{(t,2)}p_{(t,3)}}{p_{(t,1)}},\\
\widetilde{I}_{t+1} &= I_{t+1} + p_{(t,1)},\\
\widetilde{I}'_{t+1} &= I'_{t+1} + p_{(t,2)},\\
\widetilde{I}''_{t+1} &= I''_{t+1} + p_{(t,4)};\\
p_{(t+1,2)} &= -\frac{p_{(t,1)}}{p_{(t,3)}}p_{(t+1,1)},\\
p_{(t+1,4)} &= \frac{p_{(t+1,2)}p_{(t+1,3)}}{p_{(t+1,1)}},
\end{aligned}$$

$\alpha_{\overline{ID}_{t+1}}$, $\widetilde{\alpha}_{\overline{ID}_{t+1}} \in_R \mathbb{Z}_q^*$, $R_{\overline{ID}_{t+1}} = p_{(t+1,1)}\alpha_{\overline{ID}_{t+1}} + p_{(t+1,2)}\widetilde{\alpha}_{\overline{ID}_{t+1}}$, $\widetilde{R}_{\overline{ID}_{t+1}} = p_{(t+1,3)}\alpha_{\overline{ID}_{t+1}} + p_{(t+1,4)}\widetilde{\alpha}_{\overline{ID}_{t+1}}$; $s_{t+1} = \mathcal{H}_5(s_t)$, and the following:

$$\begin{aligned}
S_{\overline{ID}_{t+1}} &= s_t(I_{t+1}S_{\overline{ID}_t} + I'_{t+1}E_{\overline{ID}_t}) + \alpha_{\overline{ID}_{t+1}}L_{t+1}^{(3)},\\
E_{\overline{ID}_{t+1}} &= s_t(\frac{I_{t+1}I''_{t+1}}{I'_{t+1}}S_{\overline{ID}_t} + I''_{t+1}E_{\overline{ID}_t} + \widetilde{\alpha}_{t+1}L_{t+1}^{(3)},\\
\widetilde{S}_{\overline{ID}_{t+1}} &= s_t(\widetilde{I}_{t+1}\widetilde{S}_{\overline{ID}_t} + \widetilde{I}'_{t+1}\widetilde{E}_{\overline{ID}_t}) + (\alpha_{\overline{ID}_{t+1}} - R_{\overline{ID}_t})L_{t+1}^{(4)}, \text{ and}\\
\widetilde{E}_{\overline{ID}_{t+1}} &= s_t(\frac{\widetilde{I}_{t+1}\widetilde{I}''_{t+1}}{\widetilde{I}'_{t+1}}\widetilde{S}_{\overline{ID}_t} + \widetilde{I}''_{t+1}\widetilde{E}_{\overline{ID}_t} + (\widetilde{\alpha}_{t+1} - \widetilde{R}_{\overline{ID}_t})L_{t+1}^{(4)};
\end{aligned}$$

3) Then, $\overline{ID}_t$ secretly gives $d_{\overline{ID}_{t+1}} = (S_{\overline{ID}_{t+1}}, E_{\overline{ID}_{t+1}}, \widetilde{S}_{\overline{ID}_{t+1}}, R_{\overline{ID}_{t+1}}, \widetilde{R}_{\overline{ID}_{t+1}}, s_{t+1})$ to $\overline{ID}_{t+1}$.

- `User-Key Generation` ($mt_{\overline{ID}_t}$): For each member $\overline{ID}_t[i]$ of the group leader identified by $\overline{ID}_t$ (where $1 \leq i \leq n_{gs}$), the following takes place:

1) $\overline{ID}_t[i]$ picks $mgt_{\overline{ID}_t[i]}, \lambda \in_R \mathbb{Z}_q^*$, computes $mt_{\overline{ID}_t[i]} = (g^{\mathcal{H}_3(T)})^{mgt_{\overline{ID}_t[i]}} \in SG$, $\kappa = \mathcal{H}_2(||(g^{\mathcal{H}_3(T)})^{\lambda}||)$, and $\theta = \lambda - \kappa \cdot mgt_{\overline{ID}_t[i]}$, and secretly sends $(mt_{\overline{ID}_t[i]}, T, \kappa, \theta)$ to $\overline{ID}_t$, where $T \in \{0,1\}^{\tau}$ is the validity period of $mt_{\overline{ID}_t[i]}$, and the symbol $||$ denotes the string "$||$".

2) Upon reception of $(mt_{\overline{ID}_t[i]}, T, \kappa, \theta)$, $\overline{ID}_t$ checks whether $\mathcal{H}_2(||(g^{\mathcal{H}_3(T)})^{\theta}mt_{\overline{ID}_t[i]}^{\kappa}||) = \kappa$. If this condition is not satisfied, $\overline{ID}_t$ sends "Invalid", and interrupts the key generation process. Otherwise:

3) $\overline{ID}_t$ picks $\alpha_{\overline{ID}_t[i]}, \widetilde{\alpha}_{\overline{ID}_t[i]} \in_R \mathbb{Z}_q^*$, sets $rt_{\overline{ID}_t[i]} = mt_{\overline{ID}_t[i]}$, and stores $(\overline{ID}_t[i], rt_{\overline{ID}_t[i]})$.

4) – If $t = 0$, the root leader computes $I'_0 = \mathcal{H}_7(I_0)$, $I''_0 = \mathcal{H}_7(I'_0)$, $\widetilde{I}_0 = \mathcal{H}_7(I''_0)$, $\widetilde{I}'_0 = \mathcal{H}_7(\widetilde{I}_0)$, $\widetilde{I}''_0 = \mathcal{H}_7(\widetilde{I}'_0)$, $s_1 = \mathcal{H}_5(s_0)$, and the following:

$$\begin{aligned}
S_{\overline{ID}_0[i]} &= s_0(I_0P_0^{(1)} + I'_0mt_{\overline{ID}_0[i]}P_0^{(2)} + \alpha_{\overline{ID}_0[i]}P_0^{(3)}),\\
E_{\overline{ID}_1} &= s_0(\frac{I_0I''_0}{I'_0}P_0^{(1)} + I''_0mt_{\overline{ID}_0[i]}P_0^{(2)} + \widetilde{\alpha}_{\overline{ID}_0[i]}P_0^{(3)}),\\
\widetilde{S}_{\overline{ID}_0[i]} &= s_0(\alpha_{\overline{ID}_0[i]}P_0^{(4)} + \widetilde{I}_0P_0^{(5)} + \widetilde{I}'_0mt_{\overline{ID}_0[i]}P_0^{(6)}),\\
\widetilde{E}_{\overline{ID}_0[i]} &= s_0(\widetilde{\alpha}_{\overline{ID}_0[i]}P_0^{(4)} + \frac{\widetilde{I}_0\widetilde{I}''_0}{\widetilde{I}'_0}P_0^{(5)} + \widetilde{I}''_0mt_{\overline{ID}_0[i]}P_0^{(6)});
\end{aligned}$$

– Otherwise (i.e. if $t \geq 1$):

a. $\overline{ID}_t$ sends $mt_{\overline{ID}_t[i]}$ to its parent $\overline{ID}_{t-1}$;

b. $\overline{ID}_{t-1}$ computes $I'_0 = \mathcal{H}_7(I_0)$, $I''_0 = \mathcal{H}_7(I'_0)$, $\widetilde{I}_0 = \mathcal{H}_7(I''_0)$, $\widetilde{I}'_0 = \mathcal{H}_7(\widetilde{I}_0)$, $\widetilde{I}''_0 = \frac{I''_0\widetilde{I}'_0}{I'_0}$, $R'_{\overline{ID}_t} = (\widetilde{I}_0 - I_0)\alpha_{\overline{ID}_t} + mt_{\overline{ID}_t[i]}(\widetilde{I}'_0 - I'_0)\widetilde{\alpha}_{\overline{ID}_t}$, $\widetilde{R}'_{\overline{ID}_t} = (\frac{\widetilde{I}_0\widetilde{I}''_0I'_0 - I_0I''_0I'_0}{I'_0I'_0})\alpha_{\overline{ID}_t} + mt_{\overline{ID}_t[i]}(\widetilde{I}''_0 - I''_0)\widetilde{\alpha}_{\overline{ID}_t}$ (where $\alpha_{\overline{ID}_t}$ and $\widetilde{\alpha}_{\overline{ID}_t}$ were used to compute $\overline{ID}_t$ private key components), and returns $(R'_{\overline{ID}_t}, \widetilde{R}'_{\overline{ID}_t})$ to $\overline{ID}_t$;

c. $\overline{ID}_t$ picks computes $I'_0 = \mathcal{H}_7(I_0)$, $I''_0 = \mathcal{H}_7(K'_0)$, $\widetilde{I}_0 = \mathcal{H}_7(I''_0)$, $\widetilde{I}'_0 = \mathcal{H}_7(\widetilde{I}_0)$, $\widetilde{I}''_0 = \frac{I''_0\widetilde{I}'_0}{I'_0}$, $s_{t+1} = \mathcal{H}_5(s_t)$, and the following:

$$\begin{aligned}
S_{\overline{ID}_{t+1}} &= s_t(I_0S_{\overline{ID}_t} + I'_0mt_{\overline{ID}_t[i]}E_{\overline{ID}_t}) + \alpha_{\overline{ID}_t[i]}L_{t+1}^{(3)},\\
E_{\overline{ID}_{t+1}} &= s_t(\frac{I_0I''_0}{I'_0}S_{\overline{ID}_t} + I''_0mt_{\overline{ID}_t[i]}
\end{aligned}$$

$$E_{\overline{ID}_t}) + \widetilde{\alpha}_{\overline{ID}_t[i]} L^{(3)}_{t+1},$$

$$\widetilde{S}_{\overline{ID}_{t+1}} = s_t(\widetilde{I}_0 \widetilde{S}_{\overline{ID}_t} + \widetilde{I}'_0 mt_{\overline{ID}_t[i]} \widetilde{E}_{\overline{ID}_t}) + (\alpha_{\overline{ID}_t[i]} - R'_{\overline{ID}_t}) L^{(4)}_{t+1}, \text{ and}$$

$$\widetilde{E}_{\overline{ID}_{t+1}} = s_t(\frac{\widetilde{I}_0 \widetilde{I}''_0}{\widetilde{I}'_0} \widetilde{S}_{\overline{ID}_t} + \widetilde{I}''_0 mt_{\overline{ID}_t[i]} \widetilde{E}_{\overline{ID}_t}) + (\alpha_{\overline{ID}_t[i]} - \widetilde{R}'_{\overline{ID}_t}) L^{(4)}_{t+1};$$

5) $\overline{ID}_t$ picks $\gamma_{\overline{ID}_t[i]} \in_R \mathbb{Z}^*_q$. (This value is used by $\overline{ID}_t$ to verify $\overline{ID}_t[i]$'s signatures.)

6) $\overline{ID}_t$ secretly gives $(S_{\overline{ID}_t[i]}, E_{\overline{ID}_t[i]}, \widetilde{S}_{\overline{ID}_t[i]}, \widetilde{E}_{\overline{ID}_t[i]}, s_{t+1}, \gamma_{\overline{ID}_t[i]})$ to $\overline{ID}_t[i]$.

7) Then, $\overline{ID}_t[i]$ sets her private key $d_{\overline{ID}_t[i]} = (mgt_{\overline{ID}_t[i]}, S_{\overline{ID}_t[i]}, E_{\overline{ID}_t[i]}, \widetilde{S}_{\overline{ID}_t[i]}, \widetilde{E}_{\overline{ID}_t[i]}, s_{t+1}, \gamma_{\overline{ID}_t[i]})$. Note that, for robustness purposes, $\overline{ID}_t[i]$ can, via the *Signing* and *Verifying* algorithms, make sure that $(S_{\overline{ID}_t[i]}, E_{\overline{ID}_t[i]}, \widetilde{S}_{\overline{ID}_t[i]}, \widetilde{E}_{\overline{ID}_t[i]}, s_{t+1})$ matches $mt_{\overline{ID}_t[i]}$.

8) Finally, $\overline{ID}_t[i]$ and $\overline{ID}_t$ co-generate a random *authentication key* to be used (e.g. via a message authentication code) for the authentication of $\overline{ID}_t$ in subsequent communications.

- `Signing` $(m, d_{\overline{ID}_t}[i])$: Given a message $m \in \mathcal{M}$, the membership generating token $mgt_{\overline{ID}_t}[i]$ of a signer $\overline{ID}_t[i]$, and the system's public parameters, this algorithm:

  1) picks $r, \lambda \in_R \mathbb{Z}^*_q$, computes $\beta_r = (r^{\mathcal{H}_3(T)})^{mgt_{\overline{ID}_t[i]}}$, and derives the following:

  $\underline{S}_{\overline{ID}_t[i]} = \beta_r S_{\overline{ID}_t[i]}$, $\underline{\widetilde{S}}_{\overline{ID}_t[i]} = \beta_r \widetilde{S}_{\overline{ID}_t[i]}$,
  $U_1 = \beta_r P^{(1)}_0$, $\underline{E}_{\overline{ID}_t[i]} = \beta_r E_{\overline{ID}_t[i]}$,
  $\underline{\widetilde{E}}_{\overline{ID}_t[i]} = \beta_r \widetilde{E}_{\overline{ID}_t[i]}$, $U'_1 = \beta_r mt_{\overline{ID}_t[i]} P^{(1)}_0$,
  $U_2 = \beta_r P^{(2)}_0$, $U'_2 = \beta_r mt_{\overline{ID}_t[i]} P^{(2)}_0$,
  $U_5 = \beta_r P^{(5)}_0$, $U'_5 = \beta_r mt_{\overline{ID}_t[i]} P^{(5)}_0$,
  $U_6 = \beta_r P^{(6)}_0$, $U'_6 = \beta_r mt_{\overline{ID}_t[i]} P^{(6)}_0$,
  $\theta = \lambda - mgt_{\overline{ID}_t[i]}$, $\beta = \beta_r^{-1} + \gamma_{\overline{ID}_t[i]}$,
  $\chi = (r \cdot g)^{\mathcal{H}_3(T)}$, $W_1 = T$,
  $W_2 = r$, $\varpi = \mathcal{H}_2(mgt_{\overline{ID}_t[i]})$,
  $\kappa = \mathcal{H}_2(\varpi || (r^{\mathcal{H}_3(T)})^{\lambda} P^{(1)}_0 || \chi^{\lambda} P^{(2)}_0)$;

  2) picks $\alpha_m \in_R \mathbb{Z}^*_q$, and computes:
  $I_m = \mathcal{H}_4(m)$, $I'_m = \mathcal{H}_7(I_m)$,
  $V = s_{t+1}(I_m \underline{S}_{\overline{ID}_t[i]} + I'_m \underline{E}_{\overline{ID}_t[i]}) + \alpha_m L^{(3)}_{t+2}$,
  $\widetilde{V} = s_{t+1}(I_m \underline{\widetilde{S}}_{\overline{ID}[i]} + I'_m \underline{\widetilde{E}}_{\overline{ID}_t[i]}) + \alpha_m L^{(4)}_{t+2}$;

  3) outputs $\sigma = (U_1, U_1, U_2, U'_2, U_5, U'_5, U_6, U'_6, V, \widetilde{V}, W_1, W_2, \beta, \kappa, \theta, \varpi)$.

- `Verifying` $(RL_{\overline{ID}_t}, m, \sigma)$: Given a message m, a signature $\sigma = (U_1, U_1, U_2, U'_2, U_5, U'_5, U_6, U'_6, V, \widetilde{V}, W_1, W_2, \beta, \kappa, \theta, \varpi)$, and the system's public parameters, this algorithm:

  1) computes $(J_1, \cdots, J_t, J_0, \cdots, J_0) = \mathcal{H}_1(\overline{ID}_t) \in (\mathbb{Z}^*_q)^{3\ell}$, $I_m = \mathcal{H}_4(m)$, $I'_m = \mathcal{H}_7(I_m)$, $\rho^{(1)}_t = A_{t+2} P^{(1)}_0 + B_{t+2} P^{(2)}_0$, and $\rho^{(2)}_t = C_{t+2} P^{(5)}_0 + D_{t+2} P^{(6)}_0$ where:
     – If $t = 0$, then
     $I'_0 = \mathcal{H}_7(I_0)$, $I''_0 = \mathcal{H}_7(I'_0)$,
     $\widetilde{I}_0 = \mathcal{H}_7(I''_0)$, $\widetilde{I}'_0 = \mathcal{H}_7(\widetilde{I}_0)$,
     $\widetilde{I}''_0 = \mathcal{H}_7(\widetilde{I}'_0)$,
     $A_2 P^{(1)}_0 = (I_m I_0 + I'_m \frac{I_0 I''_0}{I'_0}) U_1$,
     $B_2 P^{(2)}_0 = (I_m I'_0 + I'_m I''_0) U'_2$,
     $C_2 P^{(5)}_0 = (I_m \widetilde{I}_0 + I'_m \frac{\widetilde{I}_0 \widetilde{I}''_0}{\widetilde{I}'_0}) U_5$,
     $D_2 P^{(6)}_0 = (I_m \widetilde{I}'_0 + I'_m \widetilde{I}''_0) U'_6$,
     – Otherwise:
     $p_{(1,2)} = \mathcal{H}_7(p_{(1,3)})$,
     $p_{(1,4)} = \frac{p_{(1,2)} p_{(1,3)}}{p_{(1,1)}}$,
     $I'_1 = -p_{(1,2)}$, $I''_1 = -\frac{p_{(1,1)}}{p_{(1,2)}} I'_1$,
     $\widetilde{I}_1 = I_1 + p_{(1,1)}$, $\widetilde{I}'_1 = I'_1 + p_{(1,2)}$,
     $\widetilde{I}''_1 = I''_1 + p_{(1,4)}$.

     $A_1 = I_1$, $B_1 = I'_1$, $C_1 = \widetilde{I}_1$,
     $D_1 = \widetilde{I}'_1$, $\widetilde{A}_1 = \frac{I_1 I''_1}{I'_1}$, $\widetilde{B}_1 = I''_1$,
     $\widetilde{C}_1 = \frac{\widetilde{I}_1 \widetilde{I}''_1}{\widetilde{I}'_1}$, $\widetilde{D}_1 = \widetilde{I}''_1$,
     and, for $1 \le i < t$,
     $I'_{i+1} = -p_{(i,2)}$, $I''_{i+1} = -\frac{p_{(i,2)} p_{(i,3)}}{p_{(i,1)}}$,
     $\widetilde{I}_{i+1} = I_{i+1} + p_{(i,1)}$,
     $\widetilde{I}'_{i+1} = I'_{i+1} + p_{(i,2)}$,
     $\widetilde{I}''_{i+1} = I''_{i+1} + p_{(i,4)}$;
     $p_{(i+1,2)} = -\frac{p_{(i,1)}}{p_{(i,3)}} p_{(i+1,1)}$,
     $p_{(i+1,4)} = \frac{p_{(i+1,2)} p_{(i+1,3)}}{p_{(i+1,1)}}$,
     $A_{i+1} = I_{i+1} A_i + I'_{i+1} \widetilde{A}_i$, and
     $\widetilde{A}_{i+1} = \frac{I_{i+1} I''_{i+1}}{I'_{i+1}} A_i + I''_{i+1} \widetilde{A}_i$,
     $B_{i+1} = I_{i+1} B_i + I'_{i+1} \widetilde{B}_i$, and
     $\widetilde{B}_{i+1} = \frac{I_{i+1} I''_{i+1}}{I'_{i+1}} B_i + I''_{i+1} \widetilde{B}_i$,
     $C_{i+1} = \widetilde{I}_{i+1} C_i + \widetilde{I}'_{i+1} \widetilde{C}_i$, and
     $\widetilde{C}_{i+1} = \frac{\widetilde{I}_{i+1} \widetilde{I}''_{i+1}}{\widetilde{I}'_{i+1}} C_i + \widetilde{I}''_{i+1} \widetilde{C}_i$,
     $D_{i+1} = \widetilde{I}_{i+1} D_i + \widetilde{I}'_{i+1} \widetilde{D}_i$, and
     $\widetilde{D}_{i+1} = \frac{\widetilde{I}_{i+1} \widetilde{I}''_{i+1}}{\widetilde{I}'_{i+1}} D_i + \widetilde{I}''_{i+1} \widetilde{D}_i$.
     Then,
     $I'_0 = \mathcal{H}_7(I_0)$, $I''_0 = \mathcal{H}_7(I'_0)$,
     $\widetilde{I}_0 = \mathcal{H}_7(I''_0)$, $\widetilde{I}'_0 = \mathcal{H}_7(\widetilde{I}_0)$,
     $\widetilde{I}''_0 = \frac{I''_0 \widetilde{I}'_0}{I'_0}$,
     $A_{t+1} P^{(1)}_0 = I_0 A_t U_1 + I'_0 \widetilde{A}_t U'_1$,
     $\widetilde{A}_{t+1} P^{(1)}_0 = \frac{I_0 I''_0}{I'_0} A_t U_1 + I''_0 \widetilde{A}_t U'_1$,
     $B_{t+1} P^{(2)}_0 = I_0 B_t U_2 + I'_0 \widetilde{B}_t U'_2$,
     $\widetilde{B}_{t+1} P^{(2)}_0 = \frac{I_0 I''_0}{I'_0} B_t U_2 + I''_0 \widetilde{B}_t U'_2$,
     $C_{t+1} P^{(5)}_0 = \widetilde{I}_0 C_t U_5 + \widetilde{I}'_0 \widetilde{C}_t U'_5$,
     $\widetilde{C}_{t+1} P^{(5)}_0 = \frac{\widetilde{I}_0 \widetilde{I}''_0}{\widetilde{I}'_0} C_t U_5 + \widetilde{I}''_0 \widetilde{C}_t U'_5$,
     $D_{t+1} P^{(6)}_0 = \widetilde{I}_0 D_t U_6 + \widetilde{I}'_0 \widetilde{D}_t U'_6$,
     $\widetilde{D}_{t+1} P^{(6)}_0 = \frac{\widetilde{I}_0 \widetilde{I}''_0}{\widetilde{I}'_0} D_t U_6 + \widetilde{I}''_0 \widetilde{D}_t U'_6$,

$$A_{t+2}P_0^{(1)} = I_m A_{t+1}P_0^{(1)} + I'_m \widetilde{A}_{t+1}P_0^{(1)},$$
$$B_{t+2}P_0^{(2)} = I_m B_{t+1}P_0^{(2)} + I'_m \widetilde{B}_{t+1}P_0^{(2)},$$
$$C_{t+2}P_0^{(5)} = I_m C_{t+1}P_0^{(5)} + I'_m \widetilde{C}_{t+1}P_0^{(5)},$$
$$D_{t+2}P_0^{(6)} = I_m D_{t+1}P_0^{(6)} + I'_m \widetilde{D}_{t+1}P_0^{(6)}.$$

2) – outputs "valid" if the following conditions are satisfied:
   * $\hat{e}(U_1 + U_2, P_0^{(5)} + P_0^{(6)}) = \hat{e}(P_0^{(1)} + P_0^{(2)}, U_5 + U_6)$ and $\hat{e}(U'_1 + U'_2, P_0^{(5)} + P_0^{(6)}) = \hat{e}(P_0^{(1)} + P_0^{(2)}, U'_5 + U'_6)$;
   * $\hat{e}(L_{t+2}^{(4)}, \rho_t^{(1)})\ \hat{e}(P_0^{(3)}, \widetilde{V}) = \hat{e}(P_0^{(4)}, V)\ \hat{e}(L_{t+2}^{(3)}, \rho_t^{(2)})$;
   * $\mathcal{H}_2(\varpi || (W_2^{\mathcal{H}_3(W_1)})^\theta U_1 || \chi^\theta U'_2) = \kappa$, where $\chi = (W_2 \cdot g)^{\mathcal{H}_3(W_1)}$;
   * $U'_1 \neq rt_{\overline{ID}_t[i]} U_1$, for each revocation token $rt_{\overline{ID}_t[i]}$ included in $RL_{\overline{ID}_t}$.
   * $(\beta - \gamma_{\overline{ID}_t[i]})U'_1 = rt_{\overline{ID}_t[i]} P_0^{(1)}$ for some unrevoked member $\overline{ID}_t[i]$ of a $\overline{ID}_t$'s group, whenever the *Verifying* algorithm is run by $\overline{ID}_t$.

   – outputs "invalid" otherwise.

Note that the identifier $\overline{ID}_t[i]$ may be kept secret from the verifier, by including, in $RL_{\overline{ID}_t}$, only the revocation tokens of $\overline{ID}_t$'s revoked group members. Note also that the fourth condition of acceptance is a zero-knowledge proof that the signer used (in the computation of β_r) the same discrete logarithm that was used to generate the key sent by $\overline{ID}_t$ to $\overline{ID}_t[i]$, where i is unknown to the verifier.

- `Opening` $(RL_{\overline{ID}_t}, m, \sigma)$: Given a signature $\sigma = (U_1, U_1, U_2, U'_2, U_5, U'_5, U_6, U'_6, V, \widetilde{V}, W_1, W_2, \beta, \kappa, \theta, \varpi)$ of m which is valid with respect to both $\overline{ID}_t$ and $RL_{\overline{ID}_t}$, this algorithm:

  1) looks for the user identifier $\overline{ID}_t[i]$ $(1 \leq i \leq n_{gs})$ such that $U'_1 = rt_{\overline{ID}_t[i]} U_1$.

  2) If no identifier $\overline{ID}_t[i]$ can be found such that the equality holds, the algorithm returns "Fail". Otherwise, $\overline{ID}_t$ outputs $\overline{ID}_t[i]$ as the claimed issuer of σ.

- `Arbitrating` $(RL_{\overline{ID}_t}, m, \sigma)$: Given a signature $\sigma = (U_1, U_2, U_3, U_4, V, \widetilde{V}, W_1, W_2, \beta, \kappa, \theta, \varpi)$ of m such that σ is valid with respect to both $\overline{ID}_t$ and $RL_{\overline{ID}_t}$, and such that a valid group member $\overline{ID}_t[i]$ refutes $\overline{ID}_t$'s claim that $\overline{ID}_t[i]$ issued σ, the algorithm proceeds as follows:

  1) A trusted third party (TTP) requests $rt_{\overline{ID}_t[i]}$ from $\overline{ID}_t$.

  2) – If $U'_1 = rt_{\overline{ID}_t[j]} U_1$, then the TTP outputs "$\overline{ID}_t[i]$ issued σ".

     – Otherwise, TTP outputs "$\overline{ID}_t[i]$ did not issue σ".

  Note that $\overline{ID}_t[i]$ needs not reveal $mgt_{\overline{ID}_t[i]}$ to the TTP.

4 Efficiency

Table 3 compares the computational[4] requirements of our HIBGS scheme with Chen et al.'s ID-based group signature scheme (CZK-IDGS [16]), and Boneh and Shacham's group signature scheme with Verifier-Local Revocation (BS-GS with VLR [9].) t denotes both the hierarchical level of a signer and the hierarchical level of a subgroup leader or subgroup member whose key is to be generated. It is assumed that all hash functions of the compared schemes have the same computational cost, denoted by H. M_X and A_X respectively denote computational costs of scalar multiplication and addition in the Abelian group X. R_X denotes the computational cost of uniformly selecting a random element in the set X. The computational cost of exponentiation in the group X is denoted by Ex_X, $\mathbf{P}$ denotes the computational cost of a bilinear pairing operation, Inv_X denotes the computational cost of inversion in X, and ψ denotes the computational cost of an efficiently computable function (cf. [9]). In order to provide a realistic computational cost average for the *Verifying* algorithm, it is assumed in Table 3 that the verifier is any party, except the signature issuer's subgroup leader. When the verifier is the issuer's subgroup leader, the computational cost of verification is at most $(2M_{\mathbb{G}_1} + A_{\mathbb{Z}_q^*})n_{gs}$ higher, and the information learned by the verifier (i.e. the fact that $\gamma_{\overline{ID}_t[i]}$ was used to generate the signature) is only used to formally prove the traceability of signatures by the appropriate subgroup leaders (cf. the proof of Theorem 2). Since all verifiers except the appropriate subgroup leaders should not be able to identify the issuers of subgroup signatures, the assumption of Table 3 seems reasonable to us. We note that all algorithms have constant storage requirements, apart from the following two exceptions: (1) the *Verifying* procedures of both our HIBGS scheme and Boneh and Shacham's scheme have storage requirements which grow linearly with respect to the size of the revocation list; (2) the *Root Setup* method of our HIBGS scheme has space requirements which grow linearly with the depth of the underlying hierarchy; (3) the storage requirements of Chen et al.'s *User-Key Generation* algorithm grow linearly with the number of issued signatures.

By the *(UNABLE)* entries of its second row, Table 3 notes that, among the three analyzed schemes, ours is the only one that (efficiently[5]) handles hierarchical set-

[4] The storage requirements of the compared schemes linearly depend on the computational requirements of these schemes. Moreover, a space unit of 256 kb can be assumed [21].

[5] Note that the two other schemes could be used in hierarchal settings by having a central entity manage the key generation and secure key distribution/update of each subgroup leader. However,

Table 3: Efficiency comparison of our HIBGS, with related schemes

	Our HIBGS	**IDGS** [16]	**GS with VLR** [9]
Root Setup	$(\ell-1)H + 2\ell M_{\mathbb{G}_1} + 6R_{\mathbb{G}_1} + 3R_{\mathbb{Z}_q^*}$	$M_{\mathbb{G}_1} + R_{\mathbb{Z}_q^*}$	$\psi + M_{\mathbb{G}_1} + R_{\mathbb{G}_1} + R_{\mathbb{Z}_q^*}$
Sub-Leader Key Generation	$A_{\mathbb{G}_1} + 7A_{\mathbb{Z}_q^*} + 6H + 3Inv_{\mathbb{Z}_q^*} + 12M_{\mathbb{G}_1} + 22M_{\mathbb{Z}_q^*} + 2R_{\mathbb{Z}_q^*}$	(UNABLE)	(UNABLE)
User Key Generation	$8A_{\mathbb{G}_1} + 5A_{\mathbb{Z}_q^*} + 4Ex_{\mathbb{Z}_q} + 11H + 4Inv_{\mathbb{Z}_q^*} + 12M_{\mathbb{G}_1} + 35M_{\mathbb{Z}_q^*} + 5R_{\mathbb{Z}_q^*}$	$\lvert SIG\rvert(2H + 3M_{\mathbb{G}_1} + 2\mathbf{P} + R_{\mathbb{Z}_q^*}) + H + M_{\mathbb{G}_1} + R_{\mathbb{Z}_q^*}$	$Inv_{\mathbb{Z}_q^*} + M_{\mathbb{G}_1} + R_{\mathbb{Z}_q^*}$
Signing	$4A_{\mathbb{G}_1} + A_{\mathbb{Z}_q^*} + 2Ex_{\mathbb{Z}_q^*} + 5H + Inv_{\mathbb{Z}_q^*} + 19M_{\mathbb{G}_1} + 8M_{\mathbb{Z}_q^*} + 3R_{\mathbb{Z}_q^*}$	$A_{\mathbb{G}_1} + A_{\mathbb{Z}_q^*} + 3H + 3M_{\mathbb{G}_1} + R_{\mathbb{Z}_q^*}$	$2A_{\mathbb{G}_1} + 3A_{\mathbb{Z}_q^*} + 2H + 3Ex_{\mathbb{G}_2} + Inv_{\mathbb{G}_1} + Inv_{\mathbb{G}_2} + 5M_{\mathbb{G}_1} + 2M_{\mathbb{G}_2} + 4M_{\mathbb{Z}_q^*} + 3\mathbf{P} + 5R_{\mathbb{Z}_q^*} + 2\psi$
Verifying	$16A_{\mathbb{G}_1} + (11t-7)A_{\mathbb{Z}_q^*} + (2t+11)H + (5t-1)Inv_{\mathbb{Z}_q^*} + 2Ex_{\mathbb{Z}_q^*} + (28+\lvert RL\rvert)M_{\mathbb{G}_1} + 2M_{\mathbb{G}_2} + (26t+4)M_{\mathbb{Z}_q^*} + 8\mathbf{P}$	$2A_{\mathbb{G}_1} + 3H + M_{\mathbb{G}_1} + 4\mathbf{P}$	$(2+\lvert RL\rvert)A_{\mathbb{G}_1} + 4Ex_{\mathbb{G}_2} + 2H + Inv_{\mathbb{G}_1} + Inv_{\mathbb{G}_2} + 4M_{\mathbb{G}_1} + 4M_{\mathbb{G}_2} + 2\lvert RL\rvert\mathbf{P} + 2\psi$
Opening	$\lvert RL\rvert M_{\mathbb{G}_1}$	$H + 4\mathbf{P}$	$\lvert RL\rvert A_{\mathbb{G}_1} + 2\lvert RL\rvert\mathbf{P}$
Arbitrating	$M_{\mathbb{G}_1}$	$H + 4\mathbf{P}$	(UNABLE)

tings. Moreover, the last row's *(UNABLE)* entry show that, unlike our scheme, Boneh and Shacham's scheme allows key generating entities to impersonate subgroup members, which removes the possibility for trusted third parties to deny any subgroup member's claim that she has been the victim of an impersonation.

Table 3 also reveals that, while being executable in constant time, the *User-Key Generation* algorithm of our proposed scheme is more computationally demanding than Boneh and Shacham's, yet significantly more efficient than Chen et al.'s (which requires one signature certificate to be issued for each signature). The *Signing* procedure of Chen et al.'s scheme is the most efficient among the three compared ones. However, due to the very high storage and computational requirements of Chen et al.'s *User-Key Generation* procedure, our *Signing* procedure improves on Boneh and Shacham's due to the fact that it requires no pairing computation. The greatest advantage of our scheme concerns the verification of signatures. Our scheme requires only 8 pairings, compared to $2|RL|$ in the case of Boneh and Schacham's scheme. Hence, our scheme is advantageous when $|RL| > 8$ (using the computational costs presented [21]). Moreover, this improvement is significant when revocation lists are either large or growing. The *Opening* procedure of our scheme is similarly more efficient than Boneh et Shacham's. Note that Chen et al.'s scheme did not present a mechanism to check the revocation status of signers at signature verification time. Such a mechanism could however be added, and would require $|RL|$ additional pairings. For Chen et al.'s *Opening* algorithm, a revocation bit could be associated with each signature certificate entry in the leader's member list. This would allow *Opening* procedures to be executed in constant time, but would not remove the $O(|SIG|)$ storage requirement, where $|SIG|$ is the number of signatures to be issued by all users of the system. Another interesting comparison can be made between our HIBGS scheme and Boneh and Franklin's anonymous authentication scheme with subset queries [7]. Compared to this scheme, our scheme has constant computational requirements for *Signing* (compared to $O(|GM|)$, where $|GM|$ is the number of subgroup leaders in the whole hierarchy.) Furthermore, the computational cost of our scheme's *Verifying* procedure grows only logarithmically with respect to the number of subgroup leaders in the underlying hierarchy. This stands in contrast with Boneh and Franklin's analogous algorithm, which grows linearly with respect to $|GM|$.

such an approach is not computationally scalable to large hierarchies. Moreover, such an approach would allow attackers to easily target the key distribution entity, and launch efficient *denial of service attacks*.

5 Security

This section presents the security guarantees of our proposed HIBGS scheme. We first present the threat model, and then state three security theorems. Due to space limitations, proofs of these theorems are only presented in the extended version of this paper.

5.1 Threat Model

We propose a threat model for *HGS* schemes with PSI, using Bellare et al.'s framework [3]. The model is presented in terms of three fundamental properties which this class of HGS schemes may be expected to have.

The first property (*correctness*) ensures that the signing and verifying procedures of the HGS scheme are complementary (i.e. that each group signature is valid if

and only if it was properly issued). The second property (*full traceability*) ensures that no coalition of subgroup members and subgroup leaders can generate a group signature which cannot be opened or traced back to some member of the coalition. The third property (*full selfless anonymity*) avoids the possibility that valid group signatures leak information about their signers. Following the approach of Boneh and Shacham [9], we consider full *selfless* anonymity (as opposed to *full anonymity*) in order to underline the fact that signers are assumed to be able to recognize (open) their own signatures. In other words, signatures are not assumed to be anonymous to their own issuers. The second and the third properties are described via two games in which an attacker attempts to break the desired property.

Note that, unlike Boneh and Shacham's threat model [9], our model assumes that key issuers can be compromised without subgroup members running the risk of being impersonated by key issuers. We believe that this subtle difference is significant, because, in practice, subgroup leaders (by whom key issuers are typically instantiated) cannot be assumed never to attempt to maliciously impersonate their subgroup members. Each of the three security requirements is formally presented below.

Correctness:

A HGS scheme is said to satisfy the *correctness* property if:

$$\forall\, m \in \mathcal{M}, \forall\, i \in \{1, 2, \cdots, n_{\overline{ID}_t}\}$$

Verifying(RL, m, Signing(m, $d_{\overline{ID}_t[i]}$) = valid $\Longleftrightarrow$ $rt_{\overline{ID}_t[i]} \notin RL$

Full Traceability:

Let Ψ be a *HIBGS* scheme. The following game, between a challenger $\mathcal{C}h$ and an attacker $\mathcal{A}$, is considered:

Setup: Given a security parameter k, $\mathcal{C}h$ uses Ψ's *Root Setup* algorithm to generate the cryptosystem's public and private parameters – keeping the private parameters secret while giving the public ones to $\mathcal{A}$. For each subgroup leader $\overline{ID}_t$, $\mathcal{C}h$ also provides $\mathcal{A}$ with $\overline{ID}_t$'s revocation token vector $RL_{(i,\overline{ID}_t)}$ and sets $U_{\overline{ID}_t}$ to be the empty set, where $U_{\overline{ID}_t}$ is the set of $\overline{ID}_t$'s subgroup members who collude against $\mathcal{C}h$.

Queries: $\mathcal{A}$ makes a polynomially bounded number of queries of the following type:

- **Public Key**: Given a subgroup leader's ID-tuple $\overline{ID}_{(i,t_i)} = (ID_{(i,1)}, \cdots, ID_{(i,t_i)})$, $\mathcal{C}h$ must return the public key associated with $\overline{ID}_{(i,t_i)}$.
- **Signing**: Given a signer's identifier $\overline{ID}_{(i,t_i)}[j]$ (where $1 \leq j \leq n_{\overline{ID}_{(i,t_i)}}$) and an arbitrary message m, $\mathcal{C}h$ computes and returns to $\mathcal{A}$ the signature σ = Signing(m, $d_{\overline{ID}_t[j]}$) of m.
- **Compromise**:
  - ∗ *Leader Compromise*: Given a subgroup leader's ID-tuple $\overline{ID}_{(i,t_i)}$, $\mathcal{C}h$ responds with $(d_{\overline{ID}_{(i,t_i)}}, sk_{\overline{ID}_{(i,t_i)}}, RL_{\overline{ID}_{(i,t_i)}})$, where $d_{\overline{ID}_{(i,t_i)}}$ is $\overline{ID}_{(i,t_i)}$'s private key, $sk_{\overline{ID}_{(i,t_i)}}$ is $\overline{ID}_{(i,t_i)}$'s user secret key vector, and $RL_{\overline{ID}_{(i,t_i)}}$ is $\overline{ID}_{(i,t_i)}$'s user revocation token vector.
  - ∗ *User Compromise*: Given a signer's identifier $\overline{ID}_{(i,t_i)}[j]$ (where $1 \leq j \leq n_{\overline{ID}_{(i,t_i)}}$), $\mathcal{C}h$ appends j to $U_{\overline{ID}_{(i,t_i)}}$ (the known coalition, from $\overline{ID}_{(i,t_i)}$'s subgroup, that is attempting to forge a signature) and responds with $d_{\overline{ID}_{(i,t_i)}[j]} = (mgt_{\overline{ID}_{(i,t_i)}[j]}, sk_{\overline{ID}_{(i,t_i)}[j]})$, where $mgt_{\overline{ID}_{(i,t_i)}[j]}$ is $\overline{ID}_{(i,t_i)}[j]$'s membership generating token and $sk_{\overline{ID}_{(i,t_i)}[j]}$ is $\overline{ID}_{(i,t_i)}[j]$'s secret key.

Forgery: $\mathcal{A}$ chooses a subgroup leader's ID-tuple $\overline{ID}_t^*$, and sends to $\mathcal{C}h$ the following: a message m^*, and a signature σ^*.

The attacker $\mathcal{A}$ wins the *Traceability* game if either one of the following conditions is satisfied: (A) $\mathcal{A}$ proves that $\mathcal{C}h$ issued incoherent answers during the game; or (B) the following are true: (B.1) σ^* is accepted by the verification algorithm as a valid signature on m^*, with respect to $\overline{ID}_t^*$; (B.2) either σ^* traces to some user outside of the coalition $U_{\overline{ID}_t^*} \backslash RL^*_{\overline{ID}_t^*}$[6] (using the *Opening* procedure), or the *Opening* algorithm fails; and (B.3) σ^* is nontrivial, i.e., $\mathcal{A}$ did not obtain σ^* by making a signing query for m^* with any of $\overline{ID}_t^*$'s signers.

The following definition is predicated on the assumption that the security of HGS schemes is to be proved in the random oracle model.

Definition: A forger $\mathcal{A}$ $(\tau, n_{comp}, n_{sig}, n_{pk}, \epsilon)$-breaks traceability in a HGS scheme if: (1) $\mathcal{A}$ runs in time at most τ; (2) $\mathcal{A}$ makes at most n_{comp} *Compromise* queries, at most n_{sig} *Signing* queries, and at most n_{pk} *Public Key* queries; and (3) the probability that $\mathcal{A}$ wins the game is at least ϵ, where the probability is taken over the coin tosses of $\mathcal{A}$ and the randomized key generation and signing algorithms.

Selfless Full Anonymity:

For *selfless full anonymity*, the following game, between a challenger $\mathcal{C}h$ and an attacker $\mathcal{A}$, is considered:

Setup: Given a security parameter k, $\mathcal{C}h$ uses Ψ's *Root Setup* algorithm to generate the cryptosystem's public and private parameters - keeping the private parameters secret while giving the public ones to $\mathcal{A}$.

Queries: $\mathcal{A}$ makes a polynomially bounded number of queries of the following type:

[6] Note that, without loss of generality, the revocation list $RL_{\overline{ID}_t^*}$ of $\overline{ID}_t^*$'s subgroup could be set to the empty set.

- **Public Key**: Given a subgroup leader's ID-tuple $\overline{ID}_{(i,t_i)} = (ID_{(i,1)}, \cdots, ID_{(i,t_i)})$, $\mathcal{C}h$ must return the public key associated with $\overline{ID}_{(i,t_i)}$.
- **Signing**: Given a signer's identifier $\overline{ID}_{(i,t_i)}[j]$ (where $1 \leq j \leq n_{\overline{ID}_{(i,t_i)}}$) and an arbitrary message m, $\mathcal{C}h$ computes and returns to $\mathcal{A}$ the signature $\sigma = \text{Signing}(m, d_{\overline{ID}_t[j]})$ of m.
- **Compromise**:
  - *Leader Compromise*: Given a subgroup leader's ID-tuple $\overline{ID}_{(i,t_i)}$, $\mathcal{C}h$ responds with $(d_{\overline{ID}_{(i,t_i)}}, sk_{\overline{ID}_{(i,t_i)}}, rt_{\overline{ID}_{(i,t_i)}})$, where $d_{\overline{ID}_{(i,t_i)}}$ is $\overline{ID}_{(i,t_i)}$'s private key, $sk_{\overline{ID}_{(i,t_i)}}$ is $\overline{ID}_{(i,t_i)}$'s user secret key vector, and $rt_{\overline{ID}_{(i,t_i)}}$ is $\overline{ID}_{(i,t_i)}$'s user revocation token vector.
  - *User Compromise*: Given a signer's identifier $\overline{ID}_{(i,t_i)}[j]$ (where $1 \leq j \leq n_{\overline{ID}_{(i,t_i)}}$), $\mathcal{C}h$ appends j to $U_{\overline{ID}_{(i,t_i)}}$ (the known coalition from $\overline{ID}_{(i,t_i)}$'s subgroup) and responds with $d_{\overline{ID}_{(i,t_i)}[j]} = (mgt_{\overline{ID}_{(i,t_i)}[j]}, sk_{\overline{ID}_{(i,t_i)}[j]})$, where $mgt_{\overline{ID}_{(i,t_i)}[j]}$ is $\overline{ID}_{(i,t_i)}[j]$'s membership generating token, and $sk_{\overline{ID}_{(i,t_i)}[j]}$ is $\overline{ID}_{(i,t_i)}[j]$'s secret key.
- **Revocation**: Given a signer's identifier $\overline{ID}_{(i,t_i)}[j]$ (where $1 \leq j \leq n_{\overline{ID}_{(i,t_i)}}$), $\mathcal{C}h$ responds with $\overline{ID}_{(i,t_i)}[j]$'s revocation token $rt_{\overline{ID}_{(i,t_i)}[j]}$.
- **Opening**: Given the ID-tuple $\overline{ID}_t$ of a claimed subgroup leader, and a signature $\sigma \in \mathcal{S}$ which is valid with respect to $\overline{ID}_t$, $\mathcal{C}h$ returns either "Fail" or the identifier $\overline{ID}_t[i]$ of σ's signer.

Challenge: $\mathcal{A}$ outputs a target subgroup leader's ID-tuple $\overline{ID}_t^*$, a message m^*, and two indices i_0 and i_1 (where $1 \leq i_0, i_1 \leq n_{\overline{ID}_t^*}$). With respect to $\overline{ID}_t^*$, $\mathcal{A}$ must have made no *Leader Compromise*, *User Compromise*, or *User Revocation* queries for either index. Moreover, no *Leader Compromise* query should have been issued on $\overline{ID}_t^*$'s parent. The challenger $\mathcal{C}h$ chooses a bit $b \in \{0, 1\}$ uniformly at random, computes a signature $\sigma^* = \text{Signing}(m^*, d_{\overline{ID}_t^*[i_b]})$ on m^* by user $\overline{ID}_t^*[i_b]$, and provides $\mathcal{A}$ with σ^*.

Restricted Queries: After obtaining the challenge, algorithm $\mathcal{A}$ can make additional queries of the challenger, restricted as follows:

- **Public Key**: $\mathcal{A}$ issues *Public Key* queries as before.
- **Signing**: $\mathcal{A}$ issues *Signing* queries as before.
- **Compromise**: As before, but $\mathcal{A}$ cannot make *Leader Compromise* queries with $\overline{ID}_t^*$, or *User Compromise* queries for i_0 or i_1, with $\overline{ID}_t^*$.
- **Revocation**: As before, but $\mathcal{A}$ cannot make *Revocation* queries for i_0 and i_1, with $\overline{ID}_t^*$.
- **Opening**: As before, but $\mathcal{A}$ cannot make *Opening* queries for i_0 and i_1, with $\overline{ID}_t^*$.

Output: $\mathcal{A}$ outputs a bit b', its guess of b.

$\mathcal{A}$ wins the *Anonymity* game if either one of the following conditions is satisfied: (A) $\mathcal{A}$ proves that $\mathcal{C}h$ issued incoherent answers during the game; or (B) $b' = b$.

Definition: An exposer $\mathcal{A}$ $(\tau, n_{comp}, n_{sig}, n_{pk}, \epsilon)$-breaks selfless full anonymity if: (1) $\mathcal{A}$ runs in time at most τ; (2) $\mathcal{A}$ makes at most n_{comp} *Compromise* queries, at most n_{sig} *Signing* queries, and at most n_{pk} *Public Key* queries; and (3) the advantage $|Pr[b = b'] - \frac{1}{2}|$ is at least ϵ, where the probabilities are taken over the coin tosses of $\mathcal{A}$, of the randomized key generation and signing algorithms, and the choice of b.

Definition: A hierarchical group signature scheme is $(\tau, n_{comp}, n_{sig}, n_{pk}, \epsilon)$-secure if it is correct, if no polynomially-bounded adversary $(\tau, n_{comp}, n_{sig}, n_{pk}, \epsilon)$-breaks full traceability, and if no polynomially-bounded adversary $(\tau, n_{comp}, n_{sig}, n_{pk}, \epsilon)$-breaks selfless anonymity.

5.2 Impersonation Threats

Since there are many different classes of impersonation attacks, we specify in this section the assumptions made in our threat model regarding impersonation.

Consider the following question: what happens if a malicious party creates a new subgroup ? This could be done either to maliciously study the behavior of joining subgroup members, or to control the security privileges of joining members. Our threat model does not address such a possibility, and our proposed scheme is therefore not intended to be protect against attacks of this nature. A related concern is the following: what happens if a malicious party impersonates a valid subgroup leader, in such a way that a legitimate user joins the impersonated leader's subgroup ? Once again, our threat model does not take into consideration such a possibility, and our HIBGS is therefore not designed to protect against this class of impersonation attacks.

The strength of our scheme, however, is to provide a mechanism which detects the impersonation of subgroup leaders in their relationship with existing subgroup members. Indeed, we require each joining subgroup member to co-generate, with her subgroup leader, an authentication key which does not solely depend on the leader's HIBGS-based private key. This authentication key should be unique to the pair formed by the joining subgroup member and the subgroup leader. The authentication key is then used, in subsequent communications, to authenticate the subgroup leader. This avoids the risk of impersonation of subgroup leaders which have not been compromised.

5.3 Security Theorems

Theorem 1. *Our proposed HIBGS scheme satisfies the correctness property.*

Theorem 2. *Let* T *be a tree-shaped hierarchy in which any proportion of ID-tuples $(ID_1, \cdots, ID_t)$ having a common* root branch *ID_1 is a non-negligible quantity with respect to a security parameter k. Assume that, in our proposed HIBGS scheme, all hash functions are random oracles. Suppose also that there exists an attacker $\mathcal{A}$, which $(\tau, n_{comp}, n_{sig}, n_{pk}, \varepsilon(k))$-breaks traceability in our scheme. Then, there exists an algorithm $\mathcal{B}$ which solves the BDH problem, in time $O(\tau)$, with non-negligible advantage at least*

$$\varepsilon(k)\frac{n_{rb}-1}{n_{rb}}(1-\delta)^{n_{comp}+n_{sig}}\frac{1}{pol(k)},$$

where n_{rb} is the number of root branches in T*, δ is the largest proportion of ID-tuples having a common root branch, and pol is a polynomial such that $\frac{1}{p(k)}$ is a lower bound of all proportions of ID-tuples having a common root branch.*

Theorem 3. *Let k be a security parameter. Assume that the hash functions of our proposed HIBGS scheme are random oracles. Suppose also that there exists an attacker $\mathcal{A}$, which $(\tau, n_{comp}, n_{sig}, n_{pk}, \varepsilon(k))$-breaks selfless full anonymity in our scheme. Then, there exists an algorithm $\mathcal{B}$ which solves the DDL problem, in time $O(\tau)$, with non-negligible advantage at least $\varepsilon(k)$.*

6 Conclusion

The aim of this paper was to propose a cryptographic scheme suitable for hierarchical anonymous authentication (HAA). HAA enables routing and filtering, in large and hierarchically-structured partially-hidden networks. Moreover, HAA supports privacy-enhanced access control in settings in which the hierarchical subgroup structure of a large group is public, and one seeks to determine whether a user belongs to a given subgroup without compromising the anonymity of this user.

We devised a threat model for a class of hierarchical group signature schemes that are suitable for HAA, and we described a hierarchical ID-based group signature (HIBGS) scheme which is provably secure, in the random oracle model, under the proposed threat model. Important features of the proposed scheme include its efficiency and its security. In terms of computational efficiency: the scheme exempts signature verifiers from the need to obtain certified copies of subgroup public keys; moreover, the proposed scheme requires the least number of pairing computations, for signing and verifying procedures (compared with other pairing-based group signature schemes); furthermore, the scheme's signing and verifying costs grow only logarithmically with respect to the number of subgroups in the hierarchy. In terms of security: the scheme provides a mechanism for signature verifiers to locally assess the revocation status of signers (and thereby the validity of group signatures); moreover, the scheme offers the guarantee that non-compromised subgroup members cannot be impersonated by their subgroup leaders.

One limitation of the proposed scheme is the fact that the computational cost of signature verification grows linearly with respect to the size of revocation lists. This is therefore an area of future research. Extensions of our scheme include the design of threshold and forward-secure ID-based group signature/signcryption schemes.

References

[1] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," in *Proceedings of CRYPTO'00 on Advances in Cryptology*, LNCS 1880, pp. 255–270, Springer-Verlag, 2000.

[2] G. Ateniese and G. Tsudik, "Some open issues and new directions in group signatures," in *Financial Cryptography 1999*, LNCS 1648, pp. 196–211, Springer-Verlag, 1999.

[3] M. Bellare, D. Micciancio, and B. Warinschi, "Foundations of group signatures: formal definitions, simplified requirements, and a construction based on general assumptions," in *Proceedings of EUROCRYPT'03 on Advances in Cryptology*, LNCS 2656, pp. 614–629, Springer Verlag, 2003.

[4] M. Bellare and P. Rogaway, "Random oracles are practical: a paradigm for deisigning efficient protocols," in *Proceedings of the 1^{st} ACM Conference on Computer and Communications Security*, pp. 62–73, ACM, 1993.

[5] D. Boneh and X. Boyen, "Efficient selective-ID secure identity-based encryption without random oracles," in *Proceedings of EUROCRYPT'04 on Advances in Cryptology*, LNCS 3027, pp. 223–238, Springer-Verlag, 2004.

[6] D. Boneh, X. Boyen, and E.-J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Proceedings of EUROCRYPT'05 on Advances in Cryptology*, LNCS 3494, pp. 440–456, Springer-Verlag, 2005.

[7] D. Boneh and M. K. Franklin, "Anonymous authentication with subset queries," in *Proceedings of the 6^{th} ACM Conference on Computer and Communications Security*, pp. 113–119, ACM, 1999.

[8] D. Boneh and M. K. Franklin, "Identity-based encryption from the weil pairing," in *Proceedings of CRYPTO'01 on Advances in Cryptology*, LNCS 2139, pp. 213–229, Springer-Verlag, 2001.

[9] D. Boneh and H. Shacham, "Group signatures with verifier-Local revocation," in *Proceedings of the 11^{th} ACM Conference on Computer and Communications Security*, pp. 168–177, ACM, 2004.

[10] E. Brickell, P. Gemmell, and D. Kravitz, "Trustee-based tracing extensions to anonymous cash and

the making of anonymous change," in *Proceedings of ACM-SIAM Symposium on Discrete Algorithm - SODA'95*, pp. 457–466, ACM, 1995.

[11] J. Camenish, "Efficient and generalized group signatures," in *Proceedings of EUROCRYPT'97 on Advances in Cryptology*, LNCS 1233, pp. 465–479, Springer-Verlag, 1997.

[12] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups," in *Proceedings of CRYPTO'97 on Advances in Cryptology*, LNCS 1070, pp. 410–424, Springer-Verlag, 1997.

[13] C. Castellucia, "How to convert any ID-based signature scheme into a group signature scheme," in *Cryptology ePrint Archive, Report 2002/116*, 2002.

[14] D. Chaum and E. van Heijst, "Group signature," in *Proceedings of EUROCRYPT'91 on Advances in Cryptology*, LNCS 547, pp. 257–265, Springer-Verlag, 1991.

[15] L. Chen and T. P. Pedersen, "New group signature schemes," in *Proceedings of EUROCRYPTO'94 on Advances in Cryptology*, LNCS 950, pp. 171–181, Springer-Verlag, 1994.

[16] X. Chen, F. Zhang, and K. Kim, "A new ID-based group signature scheme from bilinear pairings," `http://eprint.iacr.org/`, 2003.

[17] S. S. M. Chow, L. C. K. Hui, S. M. Yiu, and K. P. Chow, "Secure hierarchical identity-based signature and its applications," in *Proceedings of the Sixth Conference on Information Security and Cryptology (ICICS'04)*, LNCS 3269, pp. 480–494, Springer-Verlag, 2004.

[18] R. Cramer, I. Damgārd, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Proceedings of CRYPTO'94 on Advances in Cryptology*, LNCS 839, pp. 174–187, Springer-Verlag, 1994.

[19] U. Feige, A. Fiat, and A. Sharair, "Zero-knowledge proofs of identity," *Journal of Cryptology*, vol. 1, pp. 77–94, 1988.

[20] C. Gentry and A. Silverberg, "Hierarchical ID-based cryptography," in *Proceedings of ASIACRYPT'02 on Advances in Cryptology*, LNCS 2501, pp. 548–566, Springer-Verlag, 2002.

[21] K. Harrison, D. Page, and N. P. Smart, "Software implementation of finite fields of characteristic three, for use in pairing based cryptosystems," *London Mathematical Society Journal of Computing Mathemetics*, vol. 5, pp. 181–193, 2002.

[22] J. Horwitz and B. Lynn, "Towards hierarchical identity-based encryption," in *Proceedings of EUROCRYPT'02 on Advances in Cryptology*, LNCS 2332, pp. 466–481, Springer-Verlag, 2002.

[23] M. Joye, "On the difficulty of coalition-resistance in group signature schemes (II)," in *Technical Report*, LCIS-99-6B, 1999.

[24] M. Joye, S. Kim, and N. Lee, "Cryptanalysis of two group signature schemes," in *Information Security 1999*, LNCS 1729, pp. 271–275, Springer-Verlag, 1999.

[25] E. Kiltz, A. Mityagin, S. Panjwani, and B. Raghavan, "Append-only signatures," in *To Appear in Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, LNCS 3580, pp. 434–445, Springer-Verlag, 2005.

[26] S. Kim, S. Park, and D. Won, "Group signatures for hierarchical multigroups," in *Information Security Workshop - ISW'97*, LNCS 1396, pp. 273–281, Springer-Verlag, 1997.

[27] C. Konoma, M. Mambo, and H. Shizuya, "The computational difficulty of solving cryptographic primitive problems related to the discrete logarithm problem," *IEICE Transactions on Fundamentals*, vol. E88-A, no. 1, pp. 81–88, 2005.

[28] W. Mao and C. H. Lim, "Cryptanalysis in prime order subgroup of $\mathbb{Z}_n$," in *Proceedings of ASIACRYPT'98 on Advances in Cryptology*, LNCS 1514, pp. 214–226, Springer-Verlag, 1998.

[29] D. Nali, C. Adams, and A. Miri, "Hierarchical identity-based encryption with constant ciphertext and key length," *Manuscript*, 2005.

[30] S. Park, S. Kim, and D. Won, "ID-based group signature," *Electronics Letters*, vol. 33, pp. 1616–1617, 1997.

[31] H. Petersen, "How to convert any digital signature scheme into a group signature scheme," in *Proceedings of the Security Protocols Workshop 1997*, LNCS 1361, pp. 177–190, Springer-Verlag, 1997.

[32] C. Popescu, "Group signature schemes based on the difficulty of computation of approximate e-th roots," in *Proceedings of Protocols for Multimedia Systems (PROMS 2000)*, pp. 325–331, Poland, 2000.

[33] A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung, "On monotone formula closure of SZK," in *IEEE Foundations of Computer Science*, pp. 454–465, IEEE Press, 1994.

[34] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proceedings of CRYPTO'84 on Advances in Cryptology*, pp. 47–53. Springer-Verlag New York, Inc., 1984.

[35] M. Trolin and D. Wickstrōm, *Hierarchical Group Signature*, `http://eprint.iacr.org/2004/311.pdf`, 2004.

[36] Y. M. Tseng and J. K. Jan, "A novel ID-based group signature," in *Workshop on Cryptology and Information Security*, pp. 159–164, Tainan, 1998.

Deholo Nali is a Ph.D. student in computer science, at the University of Ottawa. He holds a M.Sc. in mathematics. His research interests include the design and analysis of identity-based cryptographic protocols.

Ali Miri received his BSc and MSc in Mathematics from the University of Toronto in 1991 and 1993 respectively, and his PhD in Electrical and Computer Engineering from the University of Waterloo in 1997. Having worked as an NSERC Postdoctoral Fellow at the University of Waterloo and the University of Toronto, he joined the School of Information Technology and Engineering (SITE) at the University of Ottawa in July of 2001, where he is currently working as an associate professor. His research interests include security and privacy technologies and their applications in e-business and e-commerce, such as network security and the role of Public Key Cryptography.

Carlisle Adams is an Associate Professor in the School of Information Technology and Engineering (SITE) at the University of Ottawa. Prior to his academic appointment in 2003, he worked for 13 years in industry in the design and standardization of a variety of cryptographic and security technologies for the Internet. His research and technical contributions include the CAST family of symmetric ciphers, protocols for authentication and management in PKI environments, and an architecture and policy language for access control in electronic networks. Dr. Adams is co-author of Understanding PKI: Concepts, Standards, and Deployment Considerations, Second Edition (Addison-Wesley, 2003).