Malaysian Journal of Mathematical Sciences

Journal homepage: https://einspem.upm.edu.my/journal

Group Identity-Based Identification: Definitions, Construction and Implementation

Vangujar, A. K. *[1], **Ng, T. S.**[2], **Chia, J.**[1], **Chin, J. J.**[1,3], and **Yip, S. C.**[1]

[1]*Faculty of Engineering, Multimedia University, Malaysia*
[2]*School of Electrical & Electronic Engineering, Yonsei University, Seoul*
[3]*Information Security Lab, MIMOS Berhad, Malaysia*

E-mail: apurva710@gmail.com
**Corresponding author*

Received: 29 October 2020
Accepted: 5 July 2021

Abstract

As an extension of identification schemes in multiparty setting, we propose the first definitions and construction for a Group Identity-Based Identification (Group-IBI) scheme. The Group-IBI involves a group manager ($\mathcal{GM}$) that is in charge of a specific group, which in turn manages several group members. The $\mathcal{GM}$'s role is not only to control the registration and revocation of the members, but also to perform an identification protocol with a verifier as a whole entity, i.e., a group. The Group-IBI scheme that we proposed is potentially suitable for numerous real-world online applications such as e-shopping, e-banking, and e-voting where consensus of all members of a group is required to be derived before proceeding with authentication. In this paper, we propose the first definitions and security models for Group-IBI. We also show the first provable-secure construction that is pairing free by using the Schnorr identity-based identification (IBI) and Schnorr signature.

Keywords: multiparty schemes; identity-based identification scheme; provable security.

1 Introduction

The authors in [11] introduced the group signature scheme (GSS), an extended version of a generic signature scheme. The entities involved in the GSS include a group member, several group members, and an adversary group member. In the GSS, the group members are only able to mark messages anonymously on behalf of the group. Since the signatures are verified by a group public key instead of an individual public key, the identity of the signer is not disclosed, thus maintaining the anonymity of the signer.

In a practical setting, GSS can be used by employees to sign documents as a representative, or an employee of the company. However, since it is not secure to store the messages, the verifier has to be aware that the group public key is authentically from the group itself. A few years later, [22] combined the GSS with blind signatures for the application of electronic cash. In the proposed scheme, the setting of multiple banks is possible. To maintain several layers of anonymity, the identity of the spender is anonymous to the bank while the spender's bank is not visible to the vendor as well.

In some works related to the GSS such as the one by [12] introduced GSS which all permits each group member to sign messages on behalf of the group whereas [10] proposed GSS whose public key and signatures have length independent of the number of group members. Therefore, it can be used for the large groups. Same author [8] constructed novel GSS which is more efficient and based on the RSA assumption. A year later [9] provided GSS based on discrete logrithmic assumption making it more secure , and [19] presented strong security model for GSS. However, some GSS are said to be restricted and inefficient. As an example, one major challenge faced by the GSS is to be able to maintain large numbers of group signatures. Another related work would be the one by [2], where theoretical foundations for the group signature primitive were provided. In addition to that, Bellare et al. also constructed the GSS based on general assumptions.

Though there has not been much progress for some period of time, GSS schemes began to resurface in 2010, such as the works done by [15]. In particular, Gordon et al.'s work is a lattice version of the GSS using the learning with errors (LWE) assumption under the random oracle model. Then, [21] proposed another lattice version of the GSS, where the revocation is done locally during the verification. Considering that all the users in the GSS are associated to a group, the security of the users are affected once the group's security is compromised. Hence, we propose the Group-IBI scheme as a solution, to protect the security of the users in a group even if the group is compromised.

1.1 Motivations and Contributions

Our contribution in this work is three-fold. The first is to formalize and define the security notions and definitions for Group-IBI. The second is we show a concrete construction using Schnorr IBI [26] and signatures [25]. Last but not least, we show efficient instantiations of the scheme using the Libsodium library on Curve25519. We describe our motivations for Group-IBI as follows.

The security of most company systems is dependent on a specialized network administration that is specific to the company itself. While it is important to have an internal management to track any ongoing transactions, it is also a key property to be able to represent the company as a whole for any external transactions to prove that the transaction is authentically done by the company instead of an impostor. The proposed Group-IBI is able to handle this use-case, which is one of its

significant properties.

With the consideration that a group has to be handled internally within by the $\mathcal{GM}$ and the group members, a trusted authority ($\mathcal{TA}$) is utilized to generate the group key pairs for many groups, where the group keys will be distributed to each $\mathcal{GM}$. However, each user key pair is generated by the $\mathcal{GM}$ himself without the interference of the $\mathcal{TA}$ or any other parties to ensure that the members management stays within the group itself. The $\mathcal{TA}$ stores all the key pairs which is generated by the $\mathcal{GM}$. It is significant to have the $\mathcal{TA}$ to maintain the whole group security.

For example, consider an e-voting system in a parliament consisting of a few parties and a session chair, the Group-IBI aims to tackle this problem by capturing the respective party in a group. Each party consisting of party members is handled by a $\mathcal{GM}$ that is in charge of registering and revoking the members of a specific party. The $\mathcal{GM}$ is also in charge of obtaining the consents from all the party members before performing a transaction. Particularly, the $\mathcal{GM}$ performs a group identification to represent the party with a verifier: the session chair.

Given that the signature proposed by [25] is one of the de-facto digital signature (DS) schemes to date, the Group-IBI scheme is initialized using variants of the Schnorr IBI and Schnorr DS after a general form of the Group-IBI is proposed. [26] and [18]'s works are used for the Schnorr IBI and Schnorr DS respectively, where a security proof is provided after the application. In the security proof for the Group-IBI, we show that the security of the Group-IBI is tied to the scheme that is initialized with, in this case the Schnorr variants.

The paper is organized as follows. Section 2 begins with some preliminaries including assumptions and security models. In Section 3, we propose the Group-IBI scheme in a general form. The security proof and model for the Group-IBI scheme is then presented in Section 4. The application of the Group-IBI scheme using Schnorr variants is elaborated in Section 5. Finally we conclude with some discussions and future work in Section 7.

2 Definitions

In this section, we present the definition of Identity-Based Identification (IBI) and the Digital Signature (DS) schemes, where the Group-IBI is viewed as a combination of both schemes. In consideration that the Group-IBI involves a multiparty setting, we will only present the security model after we have defined the Group-IBI scheme in Section 4.

2.1 Identity-Based Identification

The identification scheme was initially proposed by [14]. The authors in [6] pioneered the identity-based encryption scheme that led to the flourishing of identity-based cryptography. In later years, [3] constructed a more secure IBI scheme and constructed based on the zero-knowledge proof that results in higher efficiency.

In recent years, some advances were made in the field of IBI with work such as [1] who proposed concrete IBI scheme secure against concurrent attack, [27] constructed Hierarchical Identity-based Identification scheme has no pairing, and [13]'s scheme based on Diffie-Hellman assumption which gives tighter security proof.

The definition of an Identity-Based Identification (IBI) from [20] is presented as follows:

Definition 2.1. *An identity-based identification scheme $IBI = (\mathcal{S}, \mathcal{E}, (\mathcal{P}, \mathcal{V}))$ consists of three polynomial-time algorithms: Setup, Extract, and Identification. The algorithms are described as follows:*

i) ***Setup.*** *$(\mathcal{S})$: Given the security parameter 1^k as an input, a pair of master public and secret keys (m_{pk}, m_{sk}) is generated. m_{pk} is known to the public but m_{sk} is only made known to the public key generator (PKG), or also known as the $\mathcal{TA}$.*

ii) ***Extract.*** *$(\mathcal{E})$: $\mathcal{TA}$ takes in a public identity ID and (m_{pk}, m_{sk}) as inputs to generate a corresponding user secret key, u_{sk}.*

iii) ***Identification Protocol.*** *$(\mathcal{P}, \mathcal{V})$: Using (m_{pk}, u_{sk}, ID) and (m_{pk}, ID) as inputs for $\mathcal{P}$ and $\mathcal{V}$ respectively, both parties will interact in a protocol as follows.*

- *(a)* ***commit*** *(CMT): $\mathcal{P}$ sends CMT to $\mathcal{V}$.*
- *(b)* ***challenge*** *(CHA): $\mathcal{V}$ responds $\mathcal{P}$ with a challenge CHA.*
- *(c)* ***response*** *(RSP): $\mathcal{P}$ returns a response RSP to $\mathcal{V}$.*

At the end of the protocol, $\mathcal{V}$ decides to accept or reject $\mathcal{P}$'s RSP with a Boolean decision (I/O). A legitimate $\mathcal{P}$ should always be accepted.

2.2 Digital Signatures

The definition of a Digital Signature (DS) by [20] is presented.

Definition 2.2. *A digital signature scheme $DS = (\mathcal{KG}, \mathcal{SN}, \mathcal{VR})$ consists of three polynomial-time algorithms: Key Generation, Signing, and Verification. The algorithms are described as follows:*

i) ***Key Generation.*** *$(\mathcal{KG})$: A pair of public and secret keys are generated based on the security parameter input 1^k. The public key pk can be aired on an open channel, while the secret key sk is kept secret by the user.*

ii) ***Signing.*** *$(\mathcal{SN})$: The user uses the secret key sk to sign on a message m to generate a signature, which is denoted as σ.*

iii) ***Verification.*** *$(\mathcal{VR})$: The verifier takes the public key pk and σ as the input to ensure that the signature is genuinely signed by the user. If the signature is authentic, the algorithm returns "I", and "O" otherwise.*

2.3 Mathematical Assumptions

DDH assumption is defined from [5].

Definition 2.3. *Decisional Diffie-Hellman Assumption (DDH). A Challenger $\mathcal{C}$ is said to (t, ε)-solve the DDH assumption if $\mathcal{C}$ runs in time at most t and furthermore:*

$$|\Pr[a, b, c \leftarrow \mathbb{Z}_q : \mathcal{C}(g, g^a, g^b, g^c) = 1] - \Pr[a, b \leftarrow \mathbb{Z}_q : \mathcal{C}(g, g^a, g^b, g^{ab}) = 1]| \geq \varepsilon$$

We say that the DDH assumption is (t, ε)-hard if no algorithm (t, ε)-solves the DDH assumption.

3 Group Identity-Based Identification

In this section, we present the Group Identity-Based Identification (Group-IBI) scheme in a general form alongside the security model for our scheme. Our proposed scheme may be viewed as a general framework for Group-IBI, where we will show an example instantiation in Section 5. The scheme is defined as transactions between $\mathcal{TA}$, $\mathcal{GM}$, and several group members.

i) **Setup** $(\mathcal{S})$**.** $\mathcal{TA}$ generates master key pairs (m_{pk}, m_{sk}).

ii) **Extract** $(\mathcal{E})$**.** Phase 1 is run by the $\mathcal{TA}$, whereas Phases 2 and 3 are run by the $\mathcal{GM}$.

Phase 1. Using (m_{pk}, m_{sk}), $\mathcal{TA}$ generates group key pairs (g_{pk}, g_{sk}) and passes them to $\mathcal{GM}$.

Phase 2. For each member of a group $(G_1, G_2, ..., G_n)$, a member is required to have an ID. To register as a member of the group, a group member sends ID to $\mathcal{GM}$. Using (g_{pk}, g_{sk}), $\mathcal{GM}$ generates user keys for member of the ID, (u_{pk}, u_{sk}).

Phase 3. Then, $\mathcal{GM}$ sends the keys (u_{pk}, u_{sk}) to the group member. At the same time, $\mathcal{GM}$ stores (ID, u_{pk}) that is associated with the group member.

iii) **Identification** $(\mathcal{P}, \mathcal{V})$**.**

Phase 1. Suppose a group member wants to perform verification protocol as a group (i.e. G_1 wants to verify as a group). Using u_{sk}, G_1 generates a signature σ_1. G_1 notifies $\mathcal{GM}$ for a request to verify, and sends (u_{pk}, σ_1, ID) to $\mathcal{GM}$.

Phase 2. $\mathcal{GM}$ then checks if the signature σ_1 is valid and verifies if the associated ID is within the list of members. If G_1 is a valid member in the list, $\mathcal{GM}$ then issues a notice to all other members in the group to generate their signatures and attach their u_{pk} as well. As $\mathcal{GM}$ receives the values for each members, $\mathcal{GM}$ also checks if the provided values are valid.

Phase 3. Once all the values are obtained and are valid [1], $\mathcal{GM}$ then performs verification with a verification party $\mathcal{V}$, by attaching a signature generated from g_{sk}, σ_g as a representation of the group verification. $\mathcal{GM}$ then carries out the protocols CMT, CHA, and RSP with $\mathcal{V}$.
Note: To revoke the membership of a group member, the $\mathcal{GM}$ only has to remove the associated ID and u_{pk} from the members list. Therefore, whenever a verification is done by a non-member, $\mathcal{GM}$ verifies if the produced signature is part of the member list and does not take into account if the signature is produced by a non-member for group verification.

4 Security Model and Security Proof

In this section, we cover two areas, namely the security model of the Group-IBI, and also the security proofs of impersonating a $\mathcal{GM}$ and a group member. We first define the security model of the Group-IBI, with reference to each of the security proofs described in Section 4.4.

[1] This means that consent from all members in the group are required to be able to perform a group verification.

4.1 Malicious Third Party

A malicious third party only has the capability to eavesdrop on messages and may try to impersonate other parties using the information obtained from eavesdropping to perform malicious activities on the group.

4.1.1 Impersonating $\mathcal{GM}$

Definition 4.1. *A malicious third party may try to impersonate a $\mathcal{GM}$ to perform a group verification. However, it is not possible if he does not have (g_{pk}, g_{sk}) which is tied to the (u_{pk}, u_{sk}) from the said group. Besides that, he does not have access to the members list within the group.*

4.1.2 Impersonating a Group Member

Definition 4.2. *A malicious third party may try to impersonate a group member to be part of the group (i.e., to be able to participate in activities as a legitimate part of the group). However, since a group member is required to register via the $\mathcal{GM}$, the malicious third party who tries to impersonate the group member may fail the member list checking if he is not able to produce σ that is generated from u_{sk}. Once he tries to register as a part of the group member via the $\mathcal{GM}$, he would fall into the case of a malicious group member.*

4.2 Malicious Group Member

Definition 4.3. *A malicious group member may try to replicate the role of the $\mathcal{GM}$ by generating his own (g_{pk}, g_{sk}) to target certain members within the group and trick them into giving him their consent to be able to perform group verification.*

4.2.1 Impersonating $\mathcal{GM}$

Definition 4.4. *A malicious group member impersonates the $\mathcal{GM}$ to perform group verification by himself without needing the consent of any group members or even the $\mathcal{GM}$. The security proof will be described in Section 4.4.2 to avoid impersonation of the$\mathcal{GM}$.*

4.2.2 Impersonating Another Group Member

Definition 4.5. *The malicious group member may try to replicate another group member using their (ID, u_{pk}). The security proof for a malicious group member impersonating another group member is described in Section 4.4.1.*

4.3 Malicious $\mathcal{GM}$

The $\mathcal{GM}$'s task is to generate user keys (u_{pk}, u_{sk}) for group members using the group keys (g_{pk}, g_{sk}), while keeping track of the group members in a list. Besides that, the $\mathcal{GM}$ is also able to perform verification as a group representative. Considering the authority and role that a $\mathcal{GM}$ has over his own group, a malicious $\mathcal{GM}$'s goal is to be able to impersonate another group's $\mathcal{GM}$. With that in mind, the security proof follows the one described in Section 4.4.2.

4.4 Security Proof

4.4.1 Security Against Impersonation as Another Group Member

We define the security proof against impersonation as another group member, where a simulation game between a Challenger $\mathcal{C}$ and an Impersonator $\mathcal{I}$ is constructed. The goals of $\mathcal{C}$ and $\mathcal{I}$ are defined to solve the hard problem of the scheme and to impersonate as a member of the group, respectively.

Theorem 4.1. *The Group-IBI scheme above is* $(t_{\text{Group-IBI}}, q_e, \varepsilon_{\text{Group-IBI}})$*-secure against impersonation in the random oracle model if the hard problem of the signature holds, such that:*

$$\varepsilon_{\text{Group-IBI}} \approx \varepsilon_{\text{Sign}}$$
$$t_{\text{Group-IBI}} \approx \mathcal{O}(t_{\text{Sign}})$$

Proof. In this game, we construct a Challenger $\mathcal{C}$ making use of an Impersonator $\mathcal{I}$.

Phase 1

Setup. $\mathcal{C}$ obtains master public key, m_{pk}.

Extract Query. For an extract query of ID queried by $\mathcal{I}$, $\mathcal{C}$ computes and sends $(u_{pk_{\mathcal{I}}}, u_{sk_{\mathcal{I}}})$ to $G_{\mathcal{I}}$.

Identification Query. For an identification query on ID queried by $\mathcal{I}$, $\mathcal{C}$ checks if ID has been queried an extract query before. If so, $\mathcal{C}$ uses the existing $(u_{pk_{\mathcal{I}}}, u_{sk_{\mathcal{I}}})$ to return a valid transcript/conversation for $\mathcal{I}$; else, $\mathcal{C}$ runs extract query algorithm to generate $(u_{pk_{\mathcal{I}}}, u_{sk_{\mathcal{I}}})$ and returns a valid transcript/conversation to $\mathcal{I}$. The transcript may be a well-formed conversation created by $\mathcal{C}$ alone if it is a passive attack, or it may be a full conversation with $\mathcal{I}$ as a prover while $\mathcal{I}$ acts as a cheating verifier.

Phase 2

$\mathcal{I}$ pretends to be a valid group member using ID^*, where ID^* was queried during the extract query. $\mathcal{I}$ generates a signature $\sigma_{\mathcal{I}}$ and then sends the signature to $\mathcal{C}$. After $\mathcal{C}$ obtains the signature, $\mathcal{C}$ checks the validity of the signature [2]. If the signature produced by $\mathcal{I}$ is not valid, $\mathcal{C}$ aborts and it fails in the security game. Else, $\mathcal{C}$ can use the forgery $\sigma_{\mathcal{I}}$ to solve the hard problem used in the scheme and wins in the security game.

[2]It is noted that $\mathcal{I}$ has to produce the ID of a valid member in the group, else ID^* will fail when $\mathcal{C}$ does cross-checking on the validity of ID^* as a group member.

We now analyze the probability of aborts during the whole simulation process.

$$\begin{aligned}\Pr[\mathcal{C} \text{ wins}] &= \Pr[\mathcal{C} \text{ accepts } \sigma_{\mathcal{I}}] - \Pr[\mathcal{C} \text{ no abort}] \\ &\approx \varepsilon_{\text{Sign}} - 0 \\ &\approx \varepsilon_{\text{Sign}}\end{aligned}$$

It is noted that during the query phase, the probability of aborts occurring is highly dependent on the signature scheme used. However, the abort that may occur during the query phase due to a hash collision is negligible. Therefore, it can be conjectured that if $\mathcal{I}$ is able to come up with a valid $\sigma_{\mathcal{I}}$, $\mathcal{I}$ has broken the signature scheme used in the Group-IBI.

It is noted that $\mathcal{O}(t_{\text{Sign}})$ is the time needed to query to the oracle. □

4.4.2 Security Against Impersonation as Group Manager

We define the security proof against impersonation as a group manager, where a simulation game between a Challenger $\mathcal{C}$ and an Impersonator $\mathcal{I}$ is constructed. The goals of $\mathcal{C}$ and $\mathcal{I}$ are defined to solve the hard problem of the scheme and to impersonate as a group manager, respectively.

Theorem 4.2. *The Group-IBI scheme above is* $(t_{\text{Group-IBI}}, q_e, \varepsilon_{\text{Group-IBI}})$*-secure against impersonation in the random oracle model if the hard problem of the IBI scheme used holds, such that:*

$$\begin{aligned}\varepsilon_{\text{Group-IBI}} &\approx \varepsilon_{\text{IBI}} \\ t_{\text{Group-IBI}} &\approx \mathcal{O}(t_{\text{IBI}})\end{aligned}$$

Proof. In this game, we construct a Challenger $\mathcal{C}$ making use of an Impersonator $\mathcal{I}$.

Phase 1

Setup. Similar to the proof described in Section 4.4.1.

Extract Query. Similar to the proof described in Section 4.4.1.

Identification Query. For an identification on a query ID by $\mathcal{I}$, $\mathcal{C}$ checks if ID is queried before during the extract query. If ID was queried before to $\mathcal{C}$, $\mathcal{C}$ uses the existing $(u_{pk_{\mathcal{I}}}, u_{sk_{\mathcal{I}}})$ to return as a valid transcript/conversation; else $\mathcal{C}$ runs extract query and then plays the role of a prover and performs the identification protocol with $\mathcal{I}$.

Phase 2

$\mathcal{I}$ pretends to be a valid identity ID^*. With the transcript/conversation produced by $\mathcal{I}$, $\mathcal{C}$ wins if the transcript/conversation is valid; else $\mathcal{C}$ aborts and loses the security game. *Note:* It is noted that the security proof for impersonation as $\mathcal{GM}$ is exactly the security proof for a generic IBI scheme. Therefore, we link the security proof for impersonation as $\mathcal{GM}$ to the IBI scheme that is used, which gives us Theorem 4.2, as required. □

5 Application of Group-IBI

5.1 Group-IBI with Schnorr IBI and Schnorr DS

We initialize the Group-IBI using the Schnorr IBI and Schnorr DS. The Group-IBI involves the $\mathcal{TA}$ and the $\mathcal{GM}$ to perform a collective verification as a group, whereas the Schnorr signature involves the transaction between the $\mathcal{GM}$ and the group members for the members to prove their identity as a valid group member. We refer to [26]'s tight Schnorr IBI variant and [18]'s tight Schnorr signature variant to instantiate the Group-IBI, with consideration that both schemes use the same hard problems and have the same key generation algorithms.

i) **Setup** $(\mathcal{S})$**.** On a security level 1^k, $\mathcal{TA}$ generates two large primes p and q, such that $q|(p-1)$. $\mathcal{TA}$ also generates $x \xleftarrow{R} \mathbb{Z}_q$ to compute $y_1 = g^{-x}$ and $y_2 = h^{-x}$ where $g, h \xleftarrow{R} \mathbb{G}$. Compute a hash function $H : \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \to \mathbb{Z}_q$. The master public key (m_{pk}) is $(p, q, g, h, y_1, y_2, H)$ while the master secret key (m_{sk}) is x.

ii) **Extract** $(\mathcal{E})$**.**

Phase 1. Given a group ID g_{ID}, selects a random integer $t \xleftarrow{R} \mathbb{Z}_q$. Then, $\mathcal{TA}$ computes $A = g^t, B = h^t$, and $s = t + x\alpha$ where $\alpha = H(g_{ID}, A, B, y_1, y_2)$. $\mathcal{TA}$ passes the group public keys $g_{pk} = (g_{ID}, g, h, y_1, y_2)$ and group secret keys $g_{sk} = (\alpha, s)$ to $\mathcal{GM}$.

Phase 2. Consider a group member G_1 wants to register as a group member, he sends ID_1 to $\mathcal{GM}$. $\mathcal{GM}$ generates a random integer $a_1 \xleftarrow{R} \mathbb{Z}_q$ and then computes $y_{1,1} = g^{a_1}$ and $y_{2,1} = h^{a_1}$. In addition, $\mathcal{GM}$ also computes a hash function $H : \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \to \mathbb{Z}_q$.

Phase 3. $\mathcal{GM}$ passes the (u_{pk_1}, u_{sk_1}) to G_1 as $((H, g, h, y_{1,1}, y_{2,1}), a_1)$. At the same time, $\mathcal{GM}$ stores $u_{pk_1} = (g, h, y_{1,1}, y_{2,1})$ that is associated with the ID of group member G_1, ID_1 in a list of members.

iii) **Identification** $(\mathcal{P}, \mathcal{V})$**.**

Phase 1. Suppose a group member wants to perform verification protocol as a group (i.e. G_1 wants to verify as a group). G_1 generates a random salt $r_1 \xleftarrow{R} \mathbb{Z}_q$ and computes $A_1 = g^{r_1}, B_1 = h^{r_1}, c_1 = H(PK, A_1, B_1, m_1), s_1 = a_1 c_1 + r_1 \pmod q$. G_1 then sends the generated signature $\sigma_1 = (c_1, s_1)$ alongside the message m_1 [3] to the $\mathcal{GM}$.

Phase 2. To check the validity of the signature σ_1, $\mathcal{GM}$ retrieves u_{pk_1} from the members list and computes $A_1' = g^{s_1} y_{1,1}$ and $B_1' = h^{s_1} y_{2,1}$. If the value of $H(PK, A_1', B_1', m_1) = c_1$, the signature is authentic and is a valid signature from the group member, since he is able to retrieve u_{pk_1} from the members list. $\mathcal{GM}$ then issues a notice to all other members in the group to generate their signatures with the transaction details (format of the message, such as $m_n = (tr_1 || ID_n)$). As $\mathcal{GM}$ receives the values for each members, $\mathcal{GM}$ also checks if the

[3] G_1 may enclose the request for transaction (i.e. tr_1) alongside his ID ID_1 for verification as the message $m_1 = (tr_1 || ID_1)$ G_1 may also enclose ID_1 separate from the message to notify $\mathcal{GM}$ that he is the member that is requesting for a group verification transaction.

provided signatures are valid. Once all the values are obtained and are valid, $\mathcal{GM}$ is then able to perform verification with a verification party as a group representative.

Phase 3. $\mathcal{GM}$ performs the transaction with a verifier $\mathcal{VR}$. The verification protocol are carried out as follows.

(a) CMT : $\mathcal{GM}$ computes $A = g^s {y_1}^{\alpha}$ and $B = h^s {y_2}^{\alpha}$. Then, $\mathcal{GM}$ generates a random salt $r \overset{R}{\leftarrow} \mathbb{Z}_q$, computes $X = g^r$ and then sends (A, B, X) to $\mathcal{VR}$.

(b) CHA : $\mathcal{VR}$ then generates a challenge $c \overset{R}{\leftarrow} \mathbb{Z}_q$ and sends c to $\mathcal{GM}$.

(c) RSP : $\mathcal{GM}$ computes the response value $y = r + cs \pmod q$ and then sends the value of y to $\mathcal{VR}$.

$\mathcal{VR}$ accepts the value of if and only if the value of $g^y = X \cdot (A/{y_1}^{\alpha})^c$, where $\alpha = H(g_{ID}, A, B, y_1, y_2)$.

5.2 Security Proof

5.2.1 Security Against Impersonation as Another Group Member

In this section, we present a full security proof for the Schnorr instantiated Group-IBI.

Theorem 5.1. *The Schnorr Group-IBI scheme above is* $(t_{\text{Group-IBI}}, q_e, \varepsilon_{\text{Group-IBI}})$*-secure against impersonation in the random oracle model if the Decisional Diffie-Hellman* (DDH) *problem holds, such that:*

$$\varepsilon_{\text{Group-IBI}} \geqslant \varepsilon_{\text{DDH}} + 2(q_e + 1)q^{-1}$$
$$t_{\text{Group-IBI}} \geqslant t_{\text{DDH}} + 2.4(q_e + 1)t_{exp}$$

where q_e *is the total extract queries that are queried by an impersonator* $\mathcal{I}$ *and assuming a two-exponent multi-exponentiation takes time* $1.2t_{exp}$.

Proof. In this game, we construct Challenger $\mathcal{C}$ making use of an Impersonator $\mathcal{I}$ in the Schnorr Group-IBI environment.

Phase 1

Setup. $\mathcal{C}$ sets master public key, m_{pk} as $(p, q, g, h, y_1, y_2, H)$.

Extract Query. For an extract query of ID queried by $\mathcal{I}$, $\mathcal{C}$ generates a random integer $a_{\mathcal{I}} \overset{R}{\leftarrow} \mathbb{Z}_q$ and then computes $y_{1,\mathcal{I}} = g^{a_{\mathcal{I}}}$ and $y_{2,\mathcal{I}} = h^{a_{\mathcal{I}}}$. $\mathcal{C}$ then sends $(u_{pk_{\mathcal{I}}}, u_{sk\mathcal{I}})$ to $\mathcal{I}$.

Identification Query. For an identification query on ID queried by $\mathcal{I}$, $\mathcal{C}$ checks if ID has been queried an extract query before. If so, $\mathcal{C}$ uses the existing $(u_{pk_{\mathcal{I}}}, u_{sk\mathcal{I}})$ to return a valid transcript/conversation for $\mathcal{I}$; else, $\mathcal{C}$ runs extract query algorithm to generate (u_{pk_I}, u_{sk_I}) and returns produces a valid transcript/conversation with $\mathcal{I}$.

Hash Query. In response to query $H(PK, A_1', B_1', m_1)$, $\mathcal{C}$ checks if the hash value for m_1 is predetermined. If so, $\mathcal{C}$ returns the predetermined value to $\mathcal{I}$; else $\mathcal{C}$ generates a random value that is chosen uniformly at $\mathbb{Z}_q$.

Sign Query. In response to a signature query by $\mathcal{I}$, $\mathcal{C}$ generates random values $c'', s'' \overset{R}{\leftarrow} \mathbb{Z}_q$ and computes $A_1'' = g^{s''} {y_1}^{c''}, B_1'' = h^{s''} {y_2}^{c''}$. $\mathcal{C}$ sets $H(PK, A_1', B_1', m_1) = c''$ and outputs the signature as $\sigma'' = (c'', s'')$.

Phase 2

$\mathcal{I}$ pretends to be a valid group member using ID^*, where ID^* was queried during the extract query. After $\mathcal{I}$ generates a signature $\sigma_{\mathcal{I}} = (c_{\mathcal{I}}, s_{\mathcal{I}})$ and then sends $\sigma_{\mathcal{I}}$ to $\mathcal{C}$. As $\mathcal{C}$ obtains the signature, $\mathcal{C}$ does checking on the validity of the signature. If the signature produced by $\mathcal{I}$ is not valid, $\mathcal{C}$ aborts and it fails in the security game. Else, $\mathcal{C}$ can determine whether the given tuple is a valid DH tuple, and wins in the security game with probability as follows:

$$\begin{aligned}\Pr[\mathcal{C} \text{ wins}] &= \Pr[\text{Signature } \sigma_{\mathcal{I}} \text{ is valid}] - \Pr[\mathcal{C} \text{ aborts if it is a DH tuple}] \\ &\quad - \Pr[\mathcal{C} \text{ not aborts if it is a random tuple}] \\ &\leqslant \varepsilon_{\text{Group-IBI}} - \Pr[\mathcal{C} \text{ aborts if it is a DH tuple}] \\ &\quad - \Pr[\mathcal{C} \text{ not aborts if it is a random tuple}]\end{aligned}$$

We examine the probability that $\mathcal{C}$ aborts if the given tuple is a DH tuple. If the tuple is a valid DH tuple, $\mathcal{C}$ is able to simulate the game perfectly from the setup to sign query in Phase 1 with a negligible probability of q^{-1} on the collision of the hash oracle when answering $\mathcal{I}$'s extract queries for $(q_e + 1)$ times.

Whereas in Phase 2, If the tuple is a random tuple, $\mathcal{C}$ does not abort with a probability of q^{-1} when responding to $\mathcal{I}$'s queries for q_e times. Therefore, by combining the probabilities, we get the result as follows:

$$\begin{aligned}\Pr[\mathcal{C} \text{ wins}] &\leqslant \varepsilon_{\text{Group-IBI}} - (q_e + 1)q^{-1} - (q_e + 1)q^{-1} \\ \varepsilon_{DDH} &\leqslant \varepsilon_{\text{Group-IBI}} - 2(q_e + 1)q^{-1}\end{aligned}$$

Based on the probability calculation, we are able to obtain Theorem 5.1, as required.

The time $t_{\text{Group-IBI}}$ is therefore the equivalent of running the DDH challenger simulator $\mathcal{C}$ with the addition of $q_e + 1$ total extract queries during both phases, multiplied by the two components of $(u_{pk_{\mathcal{I}}}, u_{sk_{\mathcal{I}}})$ in the extract query that require exponentiation at time $1.2t_{exp}$, thereby giving us $t_{\text{Group-IBI}} \geqslant t_{\text{DDH}} + 2.4(q_e + 1)t_{exp}$. □

5.2.2 Security Against Impersonation as $\mathcal{GM}$

Based on the proof defined in Section 4.4.2, the security proof for impersonation as a $\mathcal{GM}$ for the Group-IBI instantiated with Tan et al.'s Schnorr IBI and Katz-Wang's Schnorr DS follows vis-à-vis Tan et al.'s security proof under active and concurrent attacks. We omit this section due to lack of space, but reserve its presentation in the full version of the paper.

6 Implementation

We present an experimental implementation of our group IBI scheme in C/C++ by using libsodium[4]. The scheme was built on an abstraction layer of Curve25519 that is Ristretto255. Curve25519 is an elliptic curve designed and implemented by [4] that features high speed key exchanges and short signatures.

While Curve25519 is often used to perform the elliptic curve Diffie-Hellman key exchange (ECDH), we employ it for group identity-based identification. Curve25519 has an order that is a prime q multiplied with a cofactor $h = 8$. As a result, the order hq cannot be used for direct construction as a prime order is required. Ristretto is designed by [17]. It is an abstraction layer which builds on Decaf point compression by [16] that enabled building of prime order groups from curves with cofactor 8.

Ristretto is supported by libsodium. Six algorithms have been implemented which are Setup, Extract Phase 1-2 and Identification Phases 1-3. Extract Phase 3 is not implemented because it only involves key distribution and storage, which can vary depending on the use cases of the scheme. An optimization was made using pre-computation: g_{sk} contains 2 extra group elements A, B which were stored after Extract Phase 1. This reduces 2 exponentiation in $\mathbb{G}$ during Identification Phase 3. Storage requirements is NOT increased because y_1 and y_2 no longer need to be stored on g_{pk} as the values A, B no longer need to be computed.

The tests were run on two different machines. The specifications are given in Table 1. In total, we run the 6 algorithms and emulated an identification attempt from a number of group members. The number of group members were increased to observe the effects of the algorithms and record the CPU run-times as well as memory usage. Each experiment was conducted 100 times with a total of 25 experiments with linearly increasing group members.

Table 1: Machines Used for Implementation Test Runs

Machine	A	B
CPU	Intel(R) Core(TM) i7-8750H CPU 2.20GHz	RaspberryPi 3B+ ARMv7 Cortex A-53 600MHz
OS	64-bit Linux OS	arm71 Linux OS
#Cores	6	4
#Threads	12	4
Memory	23 GB	932 MB

6.1 CPU Run-time

The total run-times for each experiment were then averaged over the runs and were plotted with respect to the number of group members. The results are splitted into 4 figures. Figures 1, 2, 3, 4 show the CPU run-times. The algorithms Extract Phase 2 and Identification Phase 2 have linearly increasing run-times as the number of group members increases, while the algorithms Setup, Extract Phase 1 and Identification Phase 1, 3 has constant run-times. Note that the run-time on Identification Phase 2 can be drastically improved in a real world scenario where the group mem-

[4]libsodium is a fork of the networking cryptography library (nacl) which was created by Daniel J. Bernstein

bers perform signature generation concurrently as compared to our experiment where signature generation is performed on the same machine where the group master instance is running.

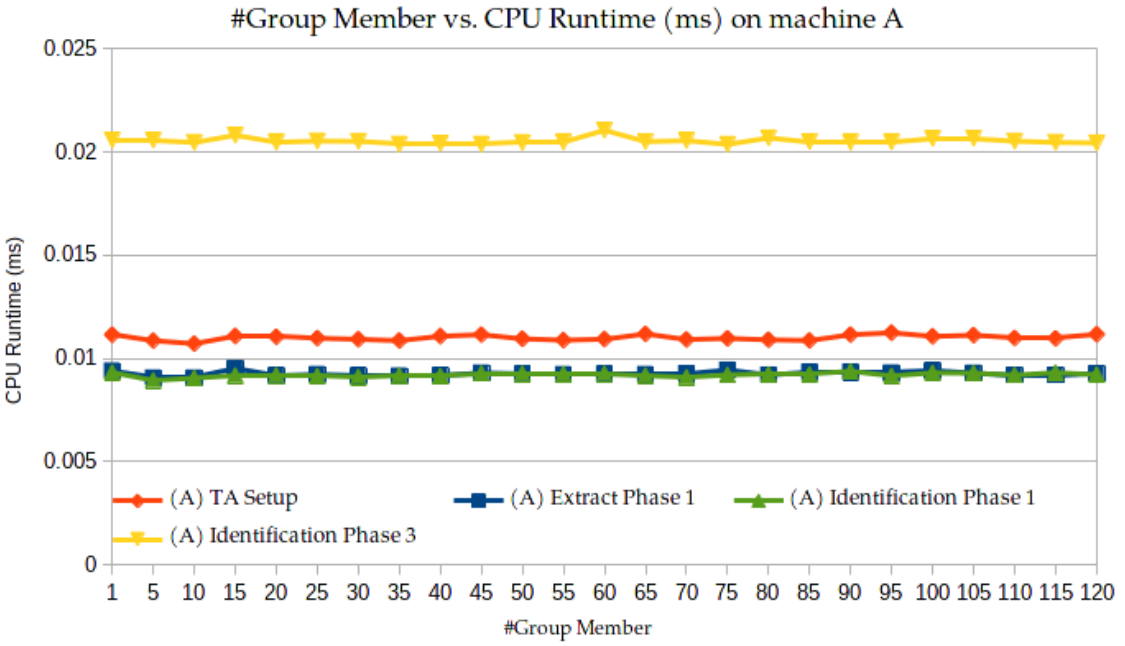

Figure 1: Algorithms with Constant Run-Times on Machine A

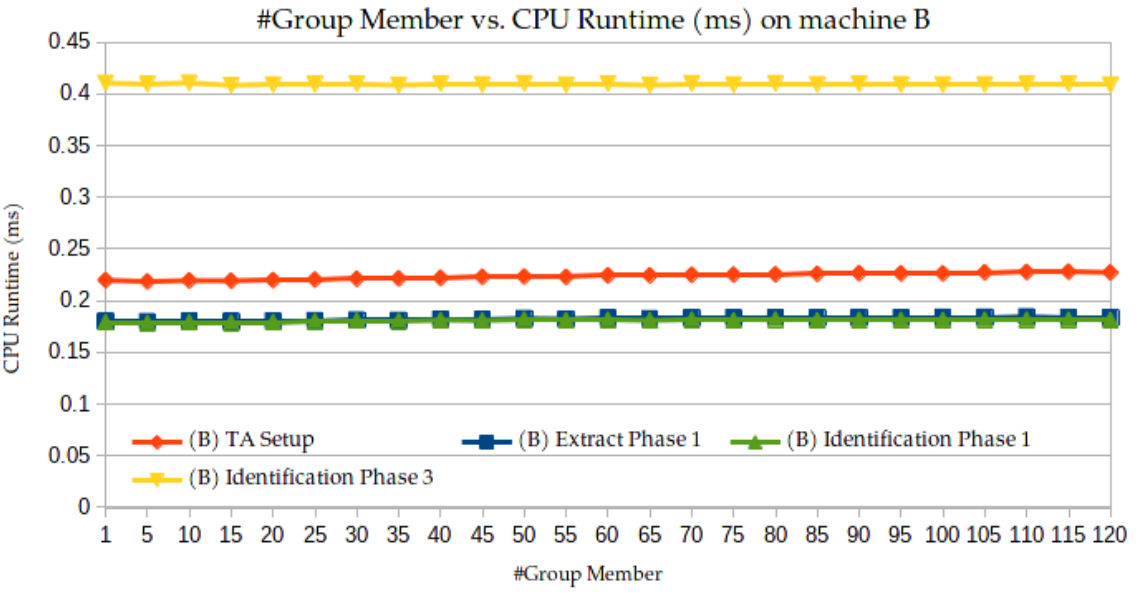

Figure 2: Algorithms with Constant Run-Times on Machine B

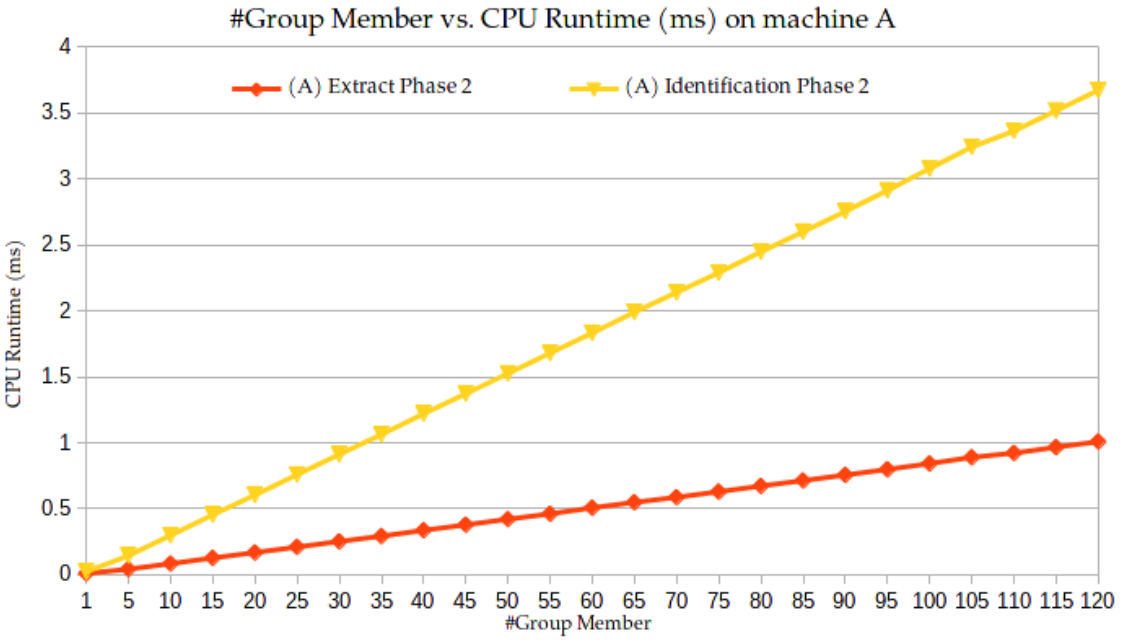

Figure 3: Algorithms with Linear Run-Times on Machine A

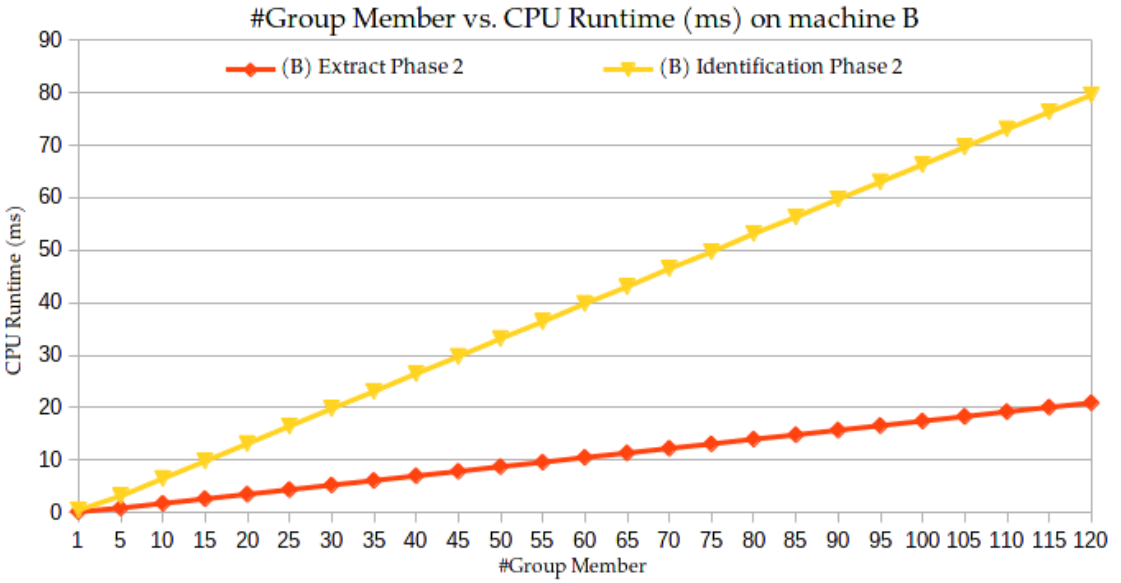

Figure 4: Algorithms with Linear Run-Times on Machine B

6.2 Memory Use

The memory use of the algorithms were also recorded using **valgrind**[5]. We record the heap usage output of valgrind for each experiment and plot a graph showing the number of group mem-

[5]valgrind is made by [23] and is used to check for heap usage and memory leaks

bers vs. heap memory usage in Figure 5. As expected, heap size increases linearly with number of group member due to Identification phase 2. We note that the total heap usage on any machine running Identification phase 2 in real world use-cases is much lower than our experiments as the group member instances will be running on different machines as compared to running on the same machine where the group master instance is running.

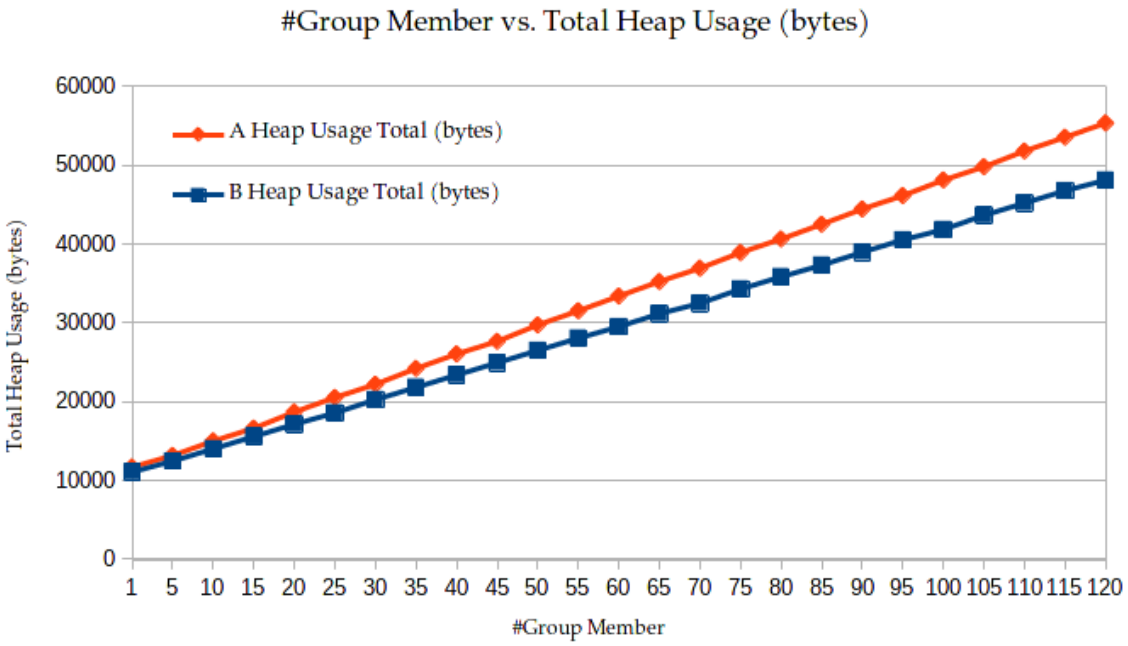

Figure 5: Heap Usages on Machine A and B

7 Conclusion

In this paper, we have presented the Group-IBI scheme, where the scheme is viewed as a combination of IBI and DS. The proposed Group-IBI enables easy members' registration and revocation within a group with the role of a group manager. Members' interactions are also not affected by the group's external interactions. Instead, the group manager is responsible for performing a verification with a verifier as a whole entity, thus providing a proof of consensus. To our knowledge, there are no existing consensus protocols yet that provide authentication using zero-knowledge proofs of knowledge.

In recent work, [24] showed that by using different hard problems such as the Decisional Square Diffie-Hellman (D-Square-DH) problem, the number of public keys used in the Schnorr DS can be reduced by one. Thus, by reducing the length of the overall public keys used in the Schnorr DS. It is stated that the number of public keys can be reduced using the same method in [26] 's work. Therefore, the number of public keys used in the Group-IBI can be reduced by applying a different hard problem.

We are unable to show any efficiency analysis comparisons due to this work being the first of its kind. Future work would be to construct a pairing-based instantiation using short signatures of the Boneh-Lynn-Shacham signature with tight security [7] combined with the recent tight-IBI

based on Kurosawa-Heng's original IBI by [13].

Acknowledgement The authors would like to thank the Multimedia University for the financial support through the Graduate Research Assistant scheme for the first author. The authors would also like to thank the Ministry of Education of Malaysia in providing financial support for this work through the Fundamental Research Grant Scheme (FRGS/1/2019/ICT04/MMU/02/5). The 3rd author would like to thank the Information Security Lab at MIMOS Berhad which hosted his visit during the curation of this journal. **Conflicts of Interest** The authors declare no conflict of interest.

References

[1] P. Barapatre & C. P. Rangan (2013). Identity-based identification schemes from ID-KEMs. In *International Conference on Security, Privacy, and Applied Cryptography Engineering*, pp. 111–129. Springer, Berlin, Heidelberg.

[2] M. Bellare, D. Micciancio & B. Warinschi (2003). Foundations of group signatures: Formal definition, simplified requirements and a construction based on trapdoor permutations. In *Advances in cryptology - EUROCRYPT 2003*, pp. 614–629. Springer, Berlin, Heidelberg.

[3] M. Bellare, C. Namprempre & G. Neven (2009). Security proofs for identity-based identification and signature schemes. *Journal of Cryptology*, *22*(1), 1–61.

[4] D. J. Bernstein (2006). Curve25519: New Diffie-Hellman speed records. In *Public Key Cryptography - PKC 2006*, pp. 207–228. Springer, Berlin, Heidelberg.

[5] D. Boneh (1998). The decision Diffie-Hellman problem. In *International Algorithmic Number Theory Symposium*, pp. 48–63. Springer, Berlin, Heidelberg.

[6] D. Boneh & M. Franklin (2003). Identity-based encryption from the Weil pairing. *SIAM Journal on Computing*, *32*(3), 586–615.

[7] D. Boneh, B. Lynn & H. Shacham (2004). Short signatures from the Weil pairing. *Journal of Cryptology*, *17*(4), 297–319.

[8] J. Camenisch & M. Michels (1998). A group signature scheme with improved efficiency. In *Advances in Cryptology- ASIACRYPT '98*, pp. 160–174. Springer, Berlin, Heidelberg.

[9] J. Camenisch & M. Michels (1999). Separability and efficiency for generic group signature schemes. In *Advances in Cryptology - CRYPTO '99*, pp. 413–430. Springer, Berlin, Heidelberg.

[10] J. Camenisch & M. Stadler (1997). Efficient group signature schemes for large groups. In *Advances in Cryptology - CRYPTO '97*, pp. 410–424. Springer, Berlin, Heidelberg.

[11] D. Chaum & T. P. Pedersen (1992). Transferred cash grows in size. In *Advances in Cryptology - EUROCRYPT'92*, pp. 390–407. Springer, Berlin, Heidelberg.

[12] L. Chen & T. P. Pedersen (1994). New group signature schemes. In *Advances in Cryptology - EUROCRYPT'94*, pp. 171–181. Springer, Berlin, Heidelberg.

[13] J. Chia & J. Chin (2020). An identity based-identification scheme with tight security against active and concurrent adversaries. *IEEE Access*, *8*, 61711–61725.

[14] A. Fiat & A. Shamir (1986). How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology-CRYPTO'86*, pp. 186–194. Springer, Berlin, Heidelberg.

[15] S. D. Gordon, J. Katz & V. Vaikuntanathan (2010). A group signature scheme from lattice assumptions. In *Advances in Cryptology - ASIACRYPT2010*, pp. 395–412. Springer, Berlin, Heidelberg.

[16] M. Hamburg (2015). *Decaf: Eliminating cofactors through point compression*. Cryptology ePrint Archive, Report 2015/673. https://eprint.iacr.org/2015/673.

[17] M. Hamburg, H. D. Valence, I. Lovecruft & T. Arcieri (2018). *The Ristretto Group*. https://ristretto.group/why_ristretto.html.

[18] J. Katz & N. Wang (2003). Efficiency improvements for signature schemes with tight security reductions. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, pp. 155–164. ACM, New York, NY.

[19] A. Kiayias & M. Yung (2005). Group signatures with efficient concurrent join. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 198–214. Springer, Berlin, Heidelberg.

[20] K. Kurosawa & S.-H. Heng (2004). From digital signature to ID-based identification/signature. In *Public Key Cryptography- PKC 2004*, pp. 248–261. Springer, Berlin, Heidelberg.

[21] A. Langlois, S. Ling, K. Nguyen & H. Wang (2014). Lattice-based group signature scheme with verifier-local revocation. In *Public Key Cryptography - PKC2014*, pp. 345–361. IACR, Buenos Aires.

[22] A. Lysyanskaya & Z. Ramzan (1998). Group blind digital signatures: A scalable solution to electronic cash. In *Financial Cryptography*, pp. 184–197. Springer, Berlin, Heidelberg.

[23] N. Nethercote & J. Seward (2007). How to shadow every byte of memory used by a program. In *Proceedings of the 3rd International Conference on Virtual Execution Environments*, pp. 65–74. ACM, New York, NY.

[24] T. S. Ng, S. Y. Tan & J. J. Chin (2017). A variant of Schnorr signature scheme with tight security reduction. In *Proceedings of the 8th International Conference on ICT Convergence*, pp. 411–415. IEEE, New York, NY.

[25] C.-P. Schnorr (1989). Efficient identification and signatures for smart cards. In *Advances in Cryptology- CRYPTO'89*, pp. 239–252. Springer, New York, NY.

[26] S.-Y. Tan, S.-H. Heng, R. C.-W. Phan & B.-M. Goi (2011). A variant of Schnorr identity-based identification scheme with tight reduction. In *Future Generation Information Technology*, pp. 361–370. Springer, Berlin, Heidelberg.

[27] A. Vangujar, J. Chin, S. Tan & T. Ng (2019). A hierarchical identity-based identification scheme without pairing. *Malaysian Journal of Mathematical Sciences*, *13*(*S*), 93–109.