Accumulators from Bilinear Pairings and Applications to ID-based Ring Signatures and Group Membership Revocation

Lan Nguyen

Centre for Information Security,
University of Wollongong, Wollongong 2522, Australia
{ldn01}@uow.edu.au

[1]

Abstract. We propose a dynamic accumulator scheme from bilinear pairings, whose security is based on the Strong Diffie-Hellman assumption. We show applications of this accumulator in constructing an identity-based (ID-based) ring signature scheme with constant-size signatures and its interactive counterpart, and providing membership revocation to group signature, traceable signature and identity escrow schemes and anonymous credential systems. The ID-based ring signature scheme and the group signature scheme have extremely short signature sizes. The size of our group signatures with membership revocation is only half the size of the well-known ACJT00 scheme, which does not provide membership revocation. The schemes do not require trapdoor, so system parameters can be shared by multiple groups belonging to different organizations. All schemes proposed are provably secure in formal models. We generalize the definition of accumulators to model a wider range of practical accumulators. We provide formal models for ID-based ad-hoc anonymous identification schemes and identity escrow schemes with membership revocation, based on existing ones.

Keywords: Dynamic accumulators, ID-based, ring signatures, ad-hoc anonymous identification, group signatures, identity escrow, membership revocation, privacy and anonymity.

1 Introduction

An *accumulator* scheme, introduced by Benaloh and de Mare [9] and further developed by Baric and Pfitzmann [4], allows aggregation of a large set of inputs into one constant-size value. For a given element, there is a *witness* that the element was included into the accumulated value whereas it is not possible to compute a witness for an element that is not accumulated. Camenisch and Lysyanskaya [20] extended the concept to *dynamic* accumulators, that means the costs of adding or deleting elements and updating individual witnesses do not

[1] This paper is the full version of “Accumulators from Bilinear Pairings and Applications” [32] presented at Cryptographers’ Track, RSA (CT-RSA) 2005.

depend on the number of elements aggregated. Accumulators have been found in a number of privacy-enhancing applications, including *ad-hoc anonymous identification*, *ring signatures* [24], *identity escrow* and *group signature* schemes with *membership revocation* [20].

Ring signature schemes, introduced by Rivest, Shamir and Tauman [34] and further studied in [16], allows a user to form an ad-hoc group without a central authority and sign messages on behalf of the group. A user might not even know that he has been included in a group and even a party with unlimited computing resources can not find out who the signer is. Zhang and Kim [42] extended the concept to *ID-based* ring signature schemes, where the group is formed by using members' identities rather than their public keys. ID-base cryptography was introduced by Shamir [37] to simplify key management in public key primitives. Since then, it has been well studied and developed in many cryptographic systems [12, 21, 14, 11, 7]. In any ID-based system, there is a central authority, called *Private Key Generator* (PKG), to extract private keys from identities. In ID-based ring signature schemes, to comply with the ad-hoc property, the involvement of a central authority is limited to only setting up initial public parameters and generating private keys from identities, and not for forming groups.

While having simple group formation set up as an advantage, the size of ring signatures linearly depends on the group size, as the verifier needs to know at least the group description. However, as pointed out in [24], in many scenarios, the group does not change for a long time or has a short description. So an appropriate measurement of ring signature sizes does not need to include the group description and it is a good direction to find constant-size ring signatures without the group description part. A ring signature scheme (DKNS04) with such a property has been proposed by Dodis, Kiayias, Nicolosi and Shoup [24]. They provide an ad-hoc anonymous identification scheme, where a user can form ad-hoc groups and anonymously prove membership in such groups, and use the Fiat-Shamir heuristics [25] to convert it into the ring signature scheme.

Although providing constant-size ring signatures, the DKNS04 scheme requires user public keys to be primes, that does not seem to allow an ID-based extension. So is there an ID-based ring signature scheme with constant-size signatures (without counting the list of identities to be included in the ring)? This paper provides the first affirmative answer to this question.

The notion of ring signatures is originated from the notion of group signatures, which was introduced by Chaum and Van Heyst [22]. A group signature scheme allows a group member to sign a message on behalf of the group without revealing his identity, and without allowing the message to be linkable to other signed messages that are verifiable under the same public key. The main difference with ring signature schemes lies in the role of a *group manager*. The role of the group manager is to register new users by issuing membership certificates that contains registration details, and in case of dispute revoke anonymity of a signed message by 'opening' the signature. In some schemes the functions of the group manager can be split between two managers: an *issuer* and an *opener*. This is a desirable property and allows a distribution of trust in the system.

Group signature schemes are closely related to a number of other cryptographic primitives. They are known to be the non-interactive counterpart of identity escrow systems [28]. In an identity escrow system a user can prove his membership of a group without revealing his identity and anonymity is revocable if a dispute occurs. An identity escrow system can be converted into a group signature scheme using the Fiat-Shamir heuristic [25]. Kiayias, Tsiounis and Yung [27] introduced the *traceable signature* primitive, which is basically the group signature system with added properties, that allow a variety of levels for protecting user privacy. Group signatures have also been used as building blocks for *anonymous credential* systems [3].

In early group signature schemes [18, 22, 23] the size of the public key and the signature grew with the size of the group and so the schemes were impractical for large groups. Schemes with fixed size group public key and signature length have been first proposed in [17] and later extended in [19, 1, 3]. In Crypto 2000, Ateniese et al. (ACJT00) [1] proposed an efficient group signature scheme with very short length and low computation cost. Ateniese and de Medeiros later proposed an efficient group signature scheme (AdM03) [3] that is 'without trapdoor' in the sense that none of the parties in the system, including the group manager, need to know the trapdoor. That is the system trapdoor is used during the initialisation and to generate system parameters. The advantage of this property is that the same trapdoor information can be used to initiate different groups. The importance and usefulness of this property in real-world applications, for example when used as a building block of anonymous credential systems when numerous organisations need to communicate and transfer information about users while protecting their privacy, have been outlined in [3].

Security of a group signature scheme has been traditionally proved by showing that it satisfies a list of requirements. However the list was informally defined and the relationship between various properties was unclear. Bellare et al. [6] gave a formal security model for group signature schemes in static group and reduced the number of requirements to three, correctness, full anonymity and full traceability, hence simplifying security goals and analysis. This model was later extended [8] to a model (BSZ04) for (partially) dynamic groups with four security requirements (correctness, anonymity, traceability and non-frameability).

Providing efficient fully dynamic group signature schemes, where users can be revoked from the group, has been a serious challenge. Early approaches [2, 15, 38] have costs linearly dependent either on the current group size or on the total number of deleted members. The most notable scheme with membership revocation (CL02) was proposed in [20], where these linear dependencies are removed; and Tsudik and Xu [41] later proposed another scheme (TX03), which requires less exponentiations in some operations. Both schemes use dynamic accumulator as the key for efficiency improvements and this method can provide membership revocation for other primitives, such as identity escrow and anonymous credential systems.

Both CL02 and TX03 schemes are based on the Strong RSA assumption, that requires the group manager to keep some trapdoor information. Using the approach in these schemes to extend AdM03 scheme to be a trapdoor-free group signature scheme with membership revocation does not seem to be easy, as the user certificates are not suitable to be accumulated by the dynamic accumulator used in CL02 and TX03 schemes. So is there a trapdoor-free group signature scheme with membership revocation as good as CL02 and TX03 schemes? This paper also provides the first affirmative answer to this question.

The security of our schemes are based on the q-Strong Diffie-Hellman (q-SDH) assumption. This assumption was strengthened by Boneh and Boyen [10] from a weaker assumption proposed in [31].

Our contribution

In this paper, we propose a new dynamic accumulator and its provably secure applications with a number of attractive properties. The applications are an ID-based ring signature scheme, a group signature scheme with membership revocation and their interactive counterparts, an ID-based ad-hoc anonymous identification scheme and an identity escrow scheme with membership revocation. The dynamic accumulator can also be used to provide membership revocation for traceable signature and anonymous credential systems. We also generalize the model of accumulators and provide formal models of ID-based ad-hoc anonymous identification schemes and identity escrow schemes with membership revocation, based on the models in [24, 8].

The schemes have a number of attractive properties. Both signature schemes provide the shortest signature sizes compared to corresponding schemes previously proposed. For example, at a comparable level of security when the CL02 and ACJT00 schemes use 1024 bit composite modulus and our group signature scheme with membership revocation uses elliptic curve groups of order 160 bit prime, the signature size in our scheme is just nearly one fourth and one half of the size of an CL02 signature and an ACJT00 signature, respectively. For higher security levels this ratio will be smaller, and ACJT00 scheme does not provide membership revocation. Like CL02 scheme, no procedure in our scheme linearly depends either on the current group size or the total number of revoked members. Our ID-based ring signature scheme is the first one providing signatures with fixed size. All previous normal ring signature schemes, except for the one in [24], have signature sizes linearly dependent on the group size. When using elliptic curve groups of order 160 bit prime, our ring signature size is only about 220 bytes.

Our schemes are completely trapdoor-free. Though also being trapdoor-free, the AdM03 scheme uses a trapdoor in the initialisation of the system and assumes that the initialising party "safely forgets" the trapdoor. Besides, the AdM03 scheme does not provide membership revocation.

Finally in our group signature scheme, the interactive protocol underlying the signature scheme achieves perfect zero-knowledge whereas in many previous schemes, including the ACJT00 and CL02 schemes, the corresponding protocols achieve statistical zero-knowledge. We note that all these zero-knowledge proofs including ours, is in honest verifier model.

The organisation of the paper is as follows. We recall some background knowledge in section 2 and present the models of dynamic accumulators, ID-based ad-hoc anonymous identification, ID-based ring signature and identity escrow with membership revocation schemes in section 3. Section 4 and 5 give descriptions of our dynamic accumulator, the ID-based ad-hoc anonymous identification and an ID-based ring signature schemes and their security proofs. Section 6 exemplifies the application of our dynamic accumulator to membership revocation by providing an identity escrow scheme with membership revocation. Section 7 provides efficiency comparison and section 8 concludes the paper.

2 Preliminaries

In this section, we briefly describe groups from bilinear pairing and their properties. Appendix A presents complexity assumptions, including Strong Diffie-Hellman, Decisional Bilinear Diffie-Hellman, Discrete Log and Decisional Diffie-Hellman assumptions, and a bilinear pairing version for El Gamal public key system (El Gamal^{BP}). It also provides a description of Digital Signature Primitives and its security requirement, Unforgeability against Chosen Message Attacks (UNF-CMA).

Notation. Let $\mathbb{N}$ be the set of positive integers. For a function $f : \mathbb{N} \to \mathbb{R}^+$, if for every positive number α, there exists a positive integer l_0 such that for every integer $l > l_0$, it holds that $f(l) < l^{-\alpha}$, then f is said to be *negligible*. Let PT denote polynomial-time, PPT denote probabilistic PT and DPT denote deterministic PT. An adversary is an interactive Turing machine. For a PT algorithm $\mathcal{A}(\cdot)$, "$x \leftarrow A(\cdot)$" denotes an output from the algorithm. For a set $\mathbf{X}$, "$x \leftarrow \mathbf{X}$" denotes an element uniformly chosen from $\mathbf{X}$. For interactive Turing machines $\mathcal{A}(\cdot)$ and $\mathcal{B}(\cdot)$, "$(a \leftarrow \mathcal{A}(\cdot) \leftrightarrow \mathcal{B}(\cdot) \to b)$" denotes that a and b are random variables corresponding to outputs of the joint computation between $\mathcal{A}(\cdot)$ and $\mathcal{B}(\cdot)$. Finally, "$\mathsf{Pr}[Procedures|Predicate]$" denotes the probability that *Predicate* is true after executing the *Procedures*.

2.1 Bilinear Pairings

Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic additive groups generated by P_1 and P_2, respectively, whose orders are a prime p, and $\mathbb{G}_M$ be a cyclic multiplicative group with the same order p. Suppose there is an isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ such that $\psi(P_2) = P_1$. Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_M$ be a bilinear pairing with the following properties:

1. **Bilinearity:** $e(aP, bQ) = e(P, Q)^{ab}$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}_2, a, b \in \mathbb{Z}_p$
2. **Non-degeneracy:** $e(P_1, P_2) \neq 1$
3. **Computability:** There is an efficient algorithm to compute $e(P, Q)$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}_2$

For simplicity, hereafter, we set $\mathbb{G}_1 = \mathbb{G}_2$ and $P_1 = P_2$. But our schemes can be easily modified for the general case when $\mathbb{G}_1 \neq \mathbb{G}_2$. For a group $\mathbb{G}$ of prime order, hereafter, we denote the set $\mathbb{G}^* = \mathbb{G}\backslash\{\mathcal{O}\}$ where $\mathcal{O}$ is the identity element of the group.

We define a Bilinear Pairing Instance Generator as a PPT algorithm $\mathcal{BPG}$ that takes as input a security parameter 1^l and returns a uniformly random tuple $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P)$ of bilinear pairing parameters, including a prime number p of size l, a cyclic additive group $\mathbb{G}_1$ of order p, a multiplicative group $\mathbb{G}_M$ of order p, a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_M$ and a generator P of $\mathbb{G}_1$.

3 Models

3.1 Accumulators

We generalize definitions of accumulators provided in [20, 24] as follows (in [20, 24], $\mathbf{U}_f = \mathbf{U}_g$ and the bijective function g is the identity function $g(x) = x$).

Definition 1. *An* accumulator *is a tuple (*$\{\mathbf{X}_l\}_{l\in\mathbb{N}}, \{\mathbf{F}_l\}_{l\in\mathbb{N}}$*), where* $\{\mathbf{X}_l\}_{l\in\mathbb{N}}$ *is called the* value domain *of the accumulator; and* $\{\mathbf{F}_l\}_{l\in\mathbb{N}}$ *is a sequence of families of pairs of functions such that each* $(f, g) \in \mathbf{F}_l$ *is defined as* $f : \mathbf{U}_f \times \mathbf{X}_f^{ext} \rightarrow \mathbf{U}_f$ *for some* $\mathbf{X}_f^{ext} \supseteq \mathbf{X}_l$*, and* $g : \mathbf{U}_f \rightarrow \mathbf{U}_g$ *is a bijective function. In addition, the following properties are satisfied:*

- (efficient generation) *There exists an efficient algorithm* $\mathcal{G}$ *that takes as input a security parameter* 1^l *and outputs a random element* $(f, g) \in_R \mathbf{F}_l$*, possibly together with some auxiliary information* a_f*.*
- (quasi commutativity) *For every* $l \in \mathbb{N}$*,* $(f, g) \in \mathbf{F}_l$*,* $u \in \mathbf{U}_f$*,* $x_1, x_2 \in \mathbf{X}_l$*:* $f(f(u, x_1), x_2) = f(f(u, x_2), x_1)$*. For any* $l \in \mathbb{N}$*,* $(f, g) \in \mathbf{F}_l$*, and* $\mathbf{X} = \{x_1, ..., x_q\} \subset \mathbf{X}_l$*, we call* $g(f(...f(u, x_1)..., x_q))$ *the* accumulated value *of the set* $\mathbf{X}$ *over* u*. Due to quasi commutativity, the value* $f(...f(u, x_1)..., x_q)$ *is independent of the order of the* x_i*'s and is denoted by* $f(u, \mathbf{X})$*.*
- (efficient evaluation) *For every* $(f, g) \in \mathbf{F}_l$*,* $u \in \mathbf{U}_f$ *and* $\mathbf{X} \subset \mathbf{X}_l$ *with size bound by a polynomial of* l*:* $g(f(u, \mathbf{X}))$ *is computable in time polynomial in* l*, even without the knowledge of* a_f*.*

Definition 2. ***(Collision Resistant Accumulator)*** *An accumulator is defined as* collision resistant *if for every PPT algorithm* $\mathcal{A}$*, the following function* $Adv_{\mathcal{A}}^{\mathsf{col.acc}}(l)$ *is negligible.*

$$\begin{aligned} Adv_{\mathcal{A}}^{\mathsf{col.acc}}(l) = \mathit{Pr}[&(f, g) \leftarrow \mathbf{F}_l; u \leftarrow \mathbf{U}_f; (x, w, \mathbf{X}) \leftarrow \mathcal{A}(f, g, u)| \\ &(\mathbf{X} \subseteq \mathbf{X}_l) \wedge (w \in \mathbf{U}_g) \wedge (x \in \mathbf{X}_f^{ext}\backslash\mathbf{X}) \\ &\wedge(f(g^{-1}(w), x) = f(u, \mathbf{X}))] \end{aligned}$$

We say that w *is a* witness *for the fact that* $x \in \mathbf{X}_l$ *has been accumulated in* $v \in \mathbf{U}_g$ *whenever* $g(f(g^{-1}(w), x)) = v$*. The notion of witness for a set of values* $\mathbf{X} \subseteq \mathbf{X}_l$ *can be defined similarly.*

Definition 3. ***(Dynamic Accumulator)*** *A* dynamic *accumulator is defined as a collision resistant accumulator with the following properties:*

- (efficient addition) *there exist PT algorithms* $\mathcal{D}_a, \mathcal{W}_a$ *such that, if* $v = g(f(u, \mathbf{X}))$, $x \in \mathbf{X}$, $x' \notin \mathbf{X}$ *and* $g(f(g^{-1}(w), x)) = v$, *then (i)* $\mathcal{D}_a(a_f, v, x') = v'$ *such that* $v' = g(f(u, \mathbf{X} \cup \{x'\}))$; *and (ii)* $\mathcal{W}_a(f, g, v, v', x, x', w) = w'$ *such that* $g(f(g^{-1}(w'), x)) = v'$.
- (efficient deletion) *there exist PT algorithms* $\mathcal{D}_d, \mathcal{W}_d$ *such that, if* $v = g(f(u, \mathbf{X}))$, $x, x' \in \mathbf{X}$, $x \neq x'$ *and* $g(f(g^{-1}(w), x)) = v$, *then (i)* $\mathcal{D}_d(a_f, v, x') = v'$ *such that* $v' = g(f(u, \mathbf{X} \backslash \{x'\}))$; *and (ii)* $\mathcal{W}_d(f, g, v, v', x, x', w) = w'$ *such that* $g(f(g^{-1}(w'), x)) = v'$.

Similar to Theorem 2 in [20], we can easily prove the following theorem about security of dynamic accumulators against adaptive attacks.

Theorem 1. *Suppose* $\mathcal{DA}$ *is a dynamic accumulator and* $\mathcal{O}$ *is an interactive Turing machine, which operates as an oracle as follows. It receives input* $(f$, g, a_f, $u)$, *where* $(f, g) \in \mathbf{F}_l$ *and* $u \in \mathbf{U}_f$. *It maintains a list of values* $\mathbf{X}$ *which is initially empty, and the current accumulated value,* v, *which is initially* $g(u)$. *It responds to two types of messages: when receiving the* (add, x) *message, it checks that* $x \in \mathbf{X}_l$, *and if so, adds* x *to the list* $\mathbf{X}$ *and updating the accumulated value (using efficient addition* $\mathcal{D}_a$*), it then sends back this updated value; similarly, when receiving the* (delete, x) *message, it checks that* $x \in \mathbf{X}$, *and if so, deletes it from the list and updates* v *(using efficient deletion* $\mathcal{D}_d$*) and sends back the updated value. In the end of the computation,* $\mathcal{O}$ *returns the current values for* $\mathbf{X}$ *and* v. *Let* $\mathbf{U}_f^{ext} \times \mathbf{X}_f^{ext}$ *denote the domains for which the computational procedure for function* f *is defined. For every PPT adversary* $\mathcal{A}$, *the following function* $Adv_{\mathcal{A}}^{\mathsf{adap.col}}(l)$ *is negligible.*

$$
\begin{aligned}
Adv_{\mathcal{A}}^{\mathsf{adap.col}}(l) = \mathsf{Pr}[&(f, g) \leftarrow \mathbf{F}_l; u \leftarrow \mathbf{U}_f; (x, w) \leftarrow \mathcal{A}(f, g, u) \leftrightarrow \\
&\mathcal{O}(f, g, a_f, u) \rightarrow (\mathbf{X}, v) \text{ — } (\mathbf{X} \subseteq \mathbf{X}_l) \wedge (w \in \mathbf{U}_g) \wedge \\
&(x \in \mathbf{X}_f^{ext} \backslash \mathbf{X}) \wedge (f(g^{-1}(w), x) = f(u, \mathbf{X}))]
\end{aligned}
$$

3.2 Identity-based Ad-hoc Anonymous Identification schemes

Syntax. The following definition is quite the same as the definition of an ad-hoc anonymous identification scheme in [24] except for some ID-based-related features: a KeyGen algorithm replaces the Register algorithm and the Setup does not maintain a database of users' public keys.

An identity-based ad-hoc anonymous identification scheme is defined as a tuple $\mathcal{IA}$ =(Setup, KeyGen, MakeGPK, MakeGSK, IAID_P , IAID_V) of PT algorithms, which are described as follows.

- Setup takes as input a security parameter 1^l and returns the public parameters *params* and a master key *mk*. The master key is only known to the Private Key Generator (PKG).
- KeyGen, run by the PKG, takes as input *params*, *mk* and an arbitrary identity of an user and outputs a private key for the user. The identity is used as the corresponding public key.

- $\mathsf{MakeGPK}$ takes as input *params* and a set of identities and deterministically outputs a single group public key which is used in the identification protocol IAID described below. Its cost linearly depends on the number of identities being aggregated. The algorithm is *order invariant* that means the order of aggregating the identities does not matter.
- $\mathsf{MakeGSK}$ takes as input *params*, a set of identities and a pair of an identity and the corresponding private key and deterministically outputs a single group secret key which is used in the identification protocol IAID described below. Its cost linearly depends on the number of identities being aggregated. It can be observed that a group secret key $gsk \leftarrow \mathsf{MakeGSK}(params, \mathbf{S}', (s_{id}, id))$ corresponds to a group public key $gpk \leftarrow \mathsf{MakeGPK}(params, \mathbf{S})$ if and only if $\mathbf{S} = \mathbf{S}' \cup \{id\}$. And more than one group secret key might corresponds to the same group public key.
- $\mathsf{IAID} = (\mathsf{IAID}_P, \mathsf{IAID}_V)$ is the two party identification protocol, which allows the prover (IAID_P) to anonymously show his membership in a group of identities he constructed by himself. Both of the prover and the verifier (IAID_V) takes as input *params* and a group public key; IAID_P is also given a corresponding group secret key; and IAID_V finally outputs 0 (reject) or 1 (accept). The cost of the protocol is independent from the number of identities that were aggregated in the group public key.

Security Requirements. The requirements are quite the same as thoses for ad-hoc anonymous identification schemes in [24], including Correctness, Soundness and Unconditional Anonymity, which are described in Appendix B.

3.3 ID-based Ring Signature schemes

Based on the model in [42], an ID-based ring signature scheme is as a tuple $\mathcal{IR}$ =(RSetup, RKeyGen, RSign, RVerify) of PT algorithms. RSetup and RKeyGen are defined the same as Setup and KeyGen in ID-based ad-hoc anonymous identification schemes. The PPT algorithm RSign takes as input the public parameter *params*, a user private key s_{id}, a set of identities, which includes the identity corresponding to s_{id}, and a message m; and outputs a signature for m. The DPT algorithm RVerify takes as input a set of identities, a message and a ring signature; and outputs either accept or reject.

There are three security requirements for ID-based ring signature schemes: Correctness, Unforgeability against Chosen Message, Group and Signer Attacks (UNF-CMGSA), and Unconditional Anonymity. Correctness intuitively requires that if RSign is given a valid private key correponding to an identity in the input set of identities, then its output signature is accepted by RVerify with overwhelming probability. UNF-CMGSA intuitively requires an adversary, who can adaptively play a chosen message-group-signer attack many times, can not forge a new ring signature with non-negligible probability. The chosen message-group-signer attack allows the adversary to adaptively choose a message, a group of identities, specify a signer in that group and query RSign for the corresponding signature. Unconditional Anonymity intuitively requires that given a ring signature, the adversary cannot tell the identity of the signer with a probability non-negligibly larger than a random guess, even assuming that the adversary has unlimited computing resources.

An ID-based ad-hoc anonymous identification scheme $\mathcal{IA}$ can be converted to an ID-based ring signature scheme $\mathcal{IR}$ by applying the Fiat-Shamir heuristics. Based on arguments similar to those in [24], we have the following lemma.

Lemma 1. *If $\mathcal{IA}$ provides Correctness, Soundness and Unconditional Anonymity, then the non-interactive dual $\mathcal{IR}$ provides Correctness, UNF-CMGSA (in the random oracle model), and Unconditional Anonymity.*

3.4 Identity Escrow schemes with Membership Revocation

Based on the BSZ04 formal model for group signature schemes, we propose a formal model for identity escrow schemes with membership revocation. The model can be used for many existent schemes, such as ones in [20, 41], where some public information needs to be updated after each addition or deletion of group members. The main extensions from the BSZ04 formal model are as follows.

- A public archive *arc* records history of the public information that needs to be updated. After each addition or deletion of group members, the issuer needs to add new information to *arc*.
- The issuer, with access to *arc* and *reg*, uses an algorithm Revoke to remove a specified member from the group by updating *arc*.
- Apart from the unchanged *membership secret key* (private signing key in the BSZ04 model), each group member also keeps a *membership witness*. Based on information in the public archive, each group member can run an algorithm Update to update the membership witness.
- There is an algorithm CheckArchive, that can be run by any party after each change in the public archive. This algorithm checks if the issuer updates the archive *arc* correctly. With such an algorithm, we can assume *arc* is always updated correctly.

An identity escrow scheme with membership revocation is a tuple $\mathcal{IE}$ =(GKg, UKg, Join, Iss, IEID_P, IEID_V, Open, Judge, Revoke, Update, CheckArchive) of PT algorithms, where GKg generates public parameters and secret keys, UKg generates personal public and private keys (different from membership secret keys) for users, the protocol (Join, Iss) allows a user to join the group and get a membership secret key and a membership witness, the protocol IEID=(IEID_P, IEID_V) allows a group member to anonymously prove his membership, Open revokes a IEID transcript to find the prover and Judge decides if the Open finds the right prover. The security requirements are Correctness, Anonymity, Traceability and Non-frameability. More details about the model are provided in Appendix B.

4 A Dynamic Accumulator from Bilinear Pairings

We propose a dynamic accumulator $\mathcal{DA}1 = (\{\mathbf{X}_l\}_{l \in \mathbb{N}}, \{\mathbf{F}_l\}_{l \in \mathbb{N}})$ from Bilinear Pairings as follows.

- Efficient Generation: To generate an instance of the accumulator from a security parameter l, use $\mathcal{BPG}$ to generate a tuple $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P)$ and $s \in_R \mathbb{Z}_p^*$. Compute a tuple $\mathbf{t}' = (P, sP, \dots, s^q P)$, where q is the upper bound on the number of elements to be accumulated by the accumulator. The corresponding functions (f, g) for $\mathbf{t}, \mathbf{t}'$ are defined as:
$f: \ \mathbb{Z}_p \times \mathbb{Z}_p \to \mathbb{Z}_p$ and $g: \mathbb{Z}_p \to \mathbb{G}_1$
$f: (u, x) \mapsto (x + s)u$ $\quad g: \ u \mapsto uP$
The corresponding domain for elements to be accumulated is $\mathbb{Z}_p \backslash \{-s\}$ and the auxiliary information is $a_f = s$. The tuple $\mathbf{t}' = (P^{(0)} = P, P^{(1)} = sP, \dots, P^{(q)} = s^q P)$ can be distributively constructed by many parties so that all of them need to cooperate to find s. Any party can verify validity of the tuple $\mathbf{t}'$ by checking if $e(P^{(0)}, P^{(q)}) = e(P^{(1)}, P^{(q-1)}) = e(P^{(2)}, P^{(q-2)}) = \dots$.
- Quasi Commutativity: It holds that: $f(f(u, x_1), x_2) = f(u, \{x_1, x_2\}) = (x_1 + s)(x_2 + s)u$.
- Efficient Evaluation: For $u \in \mathbb{Z}_p$ and a set $\mathbf{X} = \{x_1, ..., x_k\} \subset \mathbb{Z}_p \backslash \{-s\}$, where $k \leq q$, the value $g(f(u, \mathbf{X})) = \prod_{i=1}^{k} (x_i + s)uP$ is computable in time polynomial in l from the tuple $\mathbf{t}' = (P, sP, \dots, s^q P)$ and without the knowledge of the auxiliary information s.
- Efficient Addition: Suppose $V = g(f(u, \mathbf{X}))$, $x \in \mathbf{X}$, $x' \notin \mathbf{X}$ and $g(f(g^{-1}(W), x)) = V$, then $V' = g(f(u, \mathbf{X} \cup \{x'\}))$ can be computed as $V' = (x' + s)V$. And the value W' such that $g(f(g^{-1}(W'), x)) = V'$ can be computed as $W' = V + (x' - x)W$.
- Efficient Deletion: Suppose $V = g(f(u, \mathbf{X}))$, $x, x' \in \mathbf{X}$, $x \neq x'$ and $g(f(g^{-1}(W), x)) = V$, then $V' = g(f(u, \mathbf{X} \backslash \{x'\}))$ can be computed as $V' = 1/(x' + s)V$. And the value W' such that $g(f(g^{-1}(W'), x)) = V'$ can be computed as $W' = (1/(x' - x))(W - V')$.

Theorem 2 states the collision resistant property of $\mathcal{DA}1$ based on the Strong Diffie Hellman assumption as follows. Its proof is provided in Appendix C.

Theorem 2. *The accumulator $\mathcal{DA}1$ provides Collision Resistance if the q-SDH assumption holds, where q is the upper bound on the number of elements to be accumulated by the accumulator.*

5 An ID-based Ad-hoc Anonymous Identification scheme

This section presents an ID-based ad-hoc anonymous identification scheme that is based on the accumulator from bilinear pairings. We do not need the dynamic properties (efficient addition and deletion) of the accumulator for this construction.

5.1 Descriptions

As defined in the formal model, our scheme is a tuple $\mathcal{IA}1 =$(Setup, KeyGen, MakeGPK, MakeGSK, IAID_P , IAID_V) of PT algorithms, which are described as follows.

Setup, on a security parameter l, generates an instance of the accumulator above, including functions (f, g) and tuples $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P)$ and $\mathbf{t}' = (P, P_{pub} = sP, \ldots, s^q P)$, where $s \in_R \mathbb{Z}_p^*$ and q is the upper bound on the number of identities to be aggregated. The auxiliary information s can be safely deleted, as it will never be used later. It also generates $G_1, G_2, H, Q \in_R \mathbb{G}_1^*$, $u, s_m \in_R \mathbb{Z}_p^*$ and computes $Q_{pub} = s_m Q$. Let $\mathcal{H}$ be a collision-free hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_p$. Then, the public parameters are $params = (l, \mathbf{t}, \mathbf{t}', f, g, G_1, G_2, H, Q, Q_{pub}, u, \mathcal{H})$ and the master key is $mk = s_m$.

KeyGen extracts a private key $s_{id} = R_{id}$ for an identity id as $R_{id} = 1/(\mathcal{H}(id) + s_m)Q$. The user can verifies the private key by checking $e(\mathcal{H}(id)Q + Q_{pub}, R_{id}) \stackrel{?}{=} e(Q, Q)$.

MakeGPK, given a set of identities $\{id_i\}_{i=1}^k$, computes the set $\mathbf{X} = \{\mathcal{H}(id_i)\}_{i=1}^k$ and generates the group public key for the set $gpk = V = g(f(u, \mathbf{X}))$.

MakeGSK generates the group secret key gsk for a user id and a set of identities $\{id_i\}_{i=1}^k$ by computing the set $\mathbf{X}' = \{\mathcal{H}(id_i)\}_{i=1}^k$, $h_{id} = \mathcal{H}(id)$ and the witness $W = g(f(u, \mathbf{X}'))$. The group secret key is $gsk = (h_{id}, s_{id}, W)$.

(IAID$_P$, IAID$_V$) This protocol IAID has the common input $params$ and gpk and the prover (user id) also has gsk. It is a combination of the proof that an identity is accumulated and a proof of knowledge of the user private key correponding to that identity. The protocol proves the knowledge of (h_{id}, R_{id}, W) satisfying equations $e(h_{id}Q + Q_{pub}, R_{id}) = e(Q, Q)$ and $e(h_{id}P + P_{pub}, W) = e(P, V)$.

1. IAID_P generates $r_1, r_2, r_3, k_1, ..., k_7 \in_R \mathbb{Z}_p$ and computes
$$U_1 = R_{id} + r_1 H;\ U_2 = W + r_2 H;\ R = r_1 G_1 + r_2 G_2 + r_3 H;$$
$$T_1 = k_1 G_1 + k_2 G_2 + k_3 H;\ T_2 = k_4 G_1 + k_5 G_2 + k_6 H - k_7 R;$$
$$\mathit{\Pi}_1 = e(Q, U_1)^{-k_7} e(Q, H)^{k_4} e(Q_{pub}, H)^{k_1};$$
$$\mathit{\Pi}_2 = e(P, U_2)^{-k_7} e(P, H)^{k_5} e(P_{pub}, H)^{k_2}$$
2. $\mathsf{IAID}_P \longrightarrow \mathsf{IAID}_V$: $U_1, U_2, R, T_1, T_2, \mathit{\Pi}_1, \mathit{\Pi}_2$
3. $\mathsf{IAID}_P \longleftarrow \mathsf{IAID}_V$: $c \in_R \mathbb{Z}_p$
4. IAID_P computes $s_1 = k_1 + cr_1$; $s_2 = k_2 + cr_2$; $s_3 = k_3 + cr_3$; $s_4 = k_4 + cr_1 h_{id}$; $s_5 = k_5 + cr_2 h_{id}$; $s_6 = k_6 + cr_3 h_{id}$; $s_7 = k_7 + ch_{id}$
5. $\mathsf{IAID}_P \longrightarrow \mathsf{IAID}_V$: $s_1, ..., s_7$
6. IAID_V verifies that $T_1 \stackrel{?}{=} s_1 G_1 + s_2 G_2 + s_3 H - cR$; $T_2 \stackrel{?}{=} s_4 G_1 + s_5 G_2 + s_6 H - s_7 R$; $\mathit{\Pi}_1 \stackrel{?}{=} e(Q, U_1)^{-s_7} e(Q, H)^{s_4} e(Q_{pub}, H)^{s_1} e(Q, Q)^c e(Q_{pub}, U_1)^{-c}$; $\mathit{\Pi}_2 \stackrel{?}{=} e(P, U_2)^{-s_7} e(P, H)^{s_5} e(P_{pub}, H)^{s_2} e(P, V)^c e(P_{pub}, U_2)^{-c}$

5.2 Security

Security of the scheme $\mathcal{IA}1$ is stated in Theorem 3, whose proof is provided in Appendix C.

Theorem 3. *The ID-based ad-hoc anonymous identification scheme $\mathcal{IA}1$ provides Correctness and Unconditional Anonymity. The scheme $\mathcal{IA}1$ provides Soundness if the q-Strong Diffie-Hellman assumption holds, where q is the upper bound of the group size.*

5.3 Constant-size Identity-based Ring Signatures

Applying the FiatShamir heuristics to the ID-based adhoc anonymous identification scheme $\mathcal{IA}1$ results in an ID-based ring signature scheme $\mathcal{IR}1$ with constant-size signatures. More specifically, each signature contains (U_1, U_2, R, c, s_1, ... s_7), where c is computed from a hash function (a random oracle). Both the signer and the verifier only need to perform a computation proportional to the ring size once, and get some constantsize information (the group secret key and the group public key, respectively), on which they can produce/verify many subsequent signatures in constant time. The security of the scheme is stated in Theorem 4, which is based on results in Theorem 3 and Lemma 1.

Theorem 4. *The ID-based ring signature scheme $\mathcal{IR}1$ provides Correctness and Unconditional Anonymity. It also provides UNF-CMGSA in the random oracle model under the q-SDH assumption, where q is the upper bound of the group size.*

6 Application to Membership Revocation

We show how dynamic accumulators can be used to achieve membership revocation for group signature, traceable signature, identity escrow and anonymous credential systems. In particular, we provide membership revocation to an identity escrow scheme proposed in [33], and prove its security in the formal model above. The scheme can be easily converted to a group signature scheme (using Fiat-Shamir heuristics) or extended to a traceable signature scheme or an anonymous credential system; all of them provide membership revocation.

6.1 An Identity Escrow scheme with Membership Revocation

As defined in the formal model, our identity escrow scheme involves a trusted party for initial set-up, two group managers (the issuer and the opener), and users, each with a unique identity $i \in \mathbb{N}$, that may become group members. The scheme is a tuple $\mathcal{IE}1$ =(GKg, UKg, Join, Iss, IEID_P, IEID_V, Open, Judge, Revoke, Update, $\mathsf{CheckArchive}$) of PT algorithms which are defined as follows.

GKg: Suppose l is a security parameter and the generator $\mathcal{BPG}$ generates a tuple of bilinear pairing parameters $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P) \leftarrow \mathcal{G}(1^l)$, that is also the publicly shared parameters. Choose a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_p$, which is assumed to be a random oracle in the security proofs. Choose $P_0, G, G_1, G_2, H \in_R \mathbb{G}_1$, $x, x' \in_R \mathbb{Z}_p^*$ and compute $P_{pub} = xP$, $\Theta = e(G,G)^{x'}$.
An instance of the dynamic accumulator $\mathcal{DA}1$ is also generated by choosing $Q \in_R \mathbb{G}_1$, $s \in_R \mathbb{Z}_p^*$, computing $Q_{pub} = sQ$ and defining functions (f, g), correponding to the domain $\mathbb{Z}_p \backslash \{-s\}$ for elements to be accumulated and the auxiliary information $a_f = s$, as:
$f : \ \mathbb{Z}_p \times \mathbb{Z}_p \to \mathbb{Z}_p$ and $g : \mathbb{Z}_p \to \mathbb{G}_1$
$f : (u, a) \mapsto (a + s)u$ $g : u \mapsto uQ$
Note that unlike the definition of $\mathcal{DA}1$, the tuple $\mathbf{t}' = (Q, sQ, \ldots, s^q Q)$ is not needed to be generated here. The reason is that the evaluation of the accumulated value can be done by the issuer with the knowledge of the auxiliary information s; and the efficient addition and efficient deletion properties allow witnesses to be updated without the knowledge of the tuple $\mathbf{t}'$.
Besides tables *reg* and *upk*, there is also a *public archive*, as a table *arc*. Each entry j (row j^{th}) on the table will have three attributes, the first attribute contains a certificate part of an user, who was added to or deleted from the group. The second attribute is just one bit, to indicate whether the user was added (1) or deleted (0). The third attribute contains the *group accumulated value* V_j (more description of this value will be given) after adding or deleting that user. Initially, the public archive is empty, a $u \in_R \mathbb{Z}_p^*$ is generated and the group accumulated value is set to $V_0 = uQ$. The group public key is $gpk = (u, Q, Q_{pub}, P, P_0, P_{pub}, H, G, G_1, G_2, \Theta)$, the issuing key is $ik = (s, x)$, and the opening key is $ok = x'$.

UKg: This algorithm generates keys that provide authenticity for messages sent by the user in the (Join, Iss) protocol. This algorithm is the key generation algorithm K_S of any digital signature scheme (K_S, $Sign$, Ver) that is unforgeable against chosen message attacks (UNF-CMA). A user i runs the UKg algorithm that takes as input a security parameter 1^l and outputs a personal public and private signature key pair $(upk[i], usk[i])$. Public Key Infrastructure (PKI) can be used here. Although any UNF-CMA signature scheme can be used, but using schemes whose security is based on DBDH or SDH assumptions, will reduce the underlying assumptions of our group signature scheme.

(Join, Iss): In this protocol, an user i and the issuer first generate a value $x_i \in \mathbb{Z}_p^*$ so that its randomization is contributed by both parties, but its value is only known by the user. The issuer then generates (a_i, S_i) for the user so that $e(a_i P + P_{pub}, S_i) = e(P, x_i P + P_0)$. The user uses $usk[i]$ to sign his messages in the protocol. Suppose the current group accumulated value, which is publicly known, is V_j (there have been j entries on the table *arc*), the issuer computes a new group accumulated value $V_{j+1} = (a_i + s)V_j$ and appends an entry $(a_i, 1, V_{j+1})$ to the table. Note that the formal model assumes the communication to be private and authenticated. In case the user i was revoked and now rejoins the group again ($reg[i]$ has been filled), he and the issuer just need to perform the steps 8, 9, 10 of the protocol. The protocol is as follows.

1. $\mathsf{user}\ i \longrightarrow \mathsf{issuer}$: $I = yP + rH$, where $y, r \in_R \mathbb{Z}_p^*$.
2. $\mathsf{user}\ i \longleftarrow \mathsf{issuer}$: $u, v \in_R \mathbb{Z}_p^*$.
3. The user computes $x_i = uy + v$, $P_i = x_iP$.
4. $\mathsf{user}\ i \longrightarrow \mathsf{issuer}$: P_i and a proof of knowledge of (x_i, r') such that $P_i = x_iP$ and $vP + uI - P_i = r'H$ (see [19] for this proof).
5. The issuer verifies the proof, then chooses $a_i \in_R \mathbb{Z}_p^*$ different from all corresponding elements previously issued, and computes $S_i = \frac{1}{a_i+x}(P_i + P_0)$.
6. $\mathsf{user}\ i \longleftarrow \mathsf{issuer}$: a_i, S_i.
7. The user computes $\Delta_i = e(P, S_i)$, verifies if $e(a_iP + P_{pub}, S_i) = e(P, x_iP + P_0)$, and stores the *membership secret key* $gsk[i] = (x_i, a_i, S_i, \Delta_i)$. Note that only the user knows x_i. The issuer also computes Δ_i and makes an entry in the table *reg*: $reg[i] = (i, \Delta_i, \langle \mathsf{Join}, \mathsf{Iss}\rangle$ transcript so far).
8. Suppose the current group accumulated value is V_j, the issuer computes a new *group accumulated value* $V_{j+1} = (a_i + s)V_j$ and appends $(a_i, 1, V_{j+1})$ to the table *arc*.
9. $\mathsf{user}\ i \longleftarrow \mathsf{issuer}$: $j + 1, V_{j+1}$
10. The user verifies that $e(a_iQ + Q_{pub}, V_j) = e(Q, V_{j+1})$, then set his current *membership witness* to be $(j + 1, W_{i,j+1})$ where $W_{i,j+1} = V_j$.

($\mathbf{IEID}_P$, $\mathbf{IEID}_V$): This protocol IEID shows an user i's knowledge of (a_i, S_i) and a secret x_i such that: $e(a_iP + P_{pub}, S_i) = e(P, x_iP + P_0)$ and a_i has been accumulated in the current group accumulated value. The protocol does not reveal any information about his knowledge to anyone, except for the opener, who can only compute Δ_i by decrypting an encryption of that value. Before the protocol is started, user i checks the table *arc* to find the latest group accumulated value V_j and runs Update algorithm to compute his current membership witness $(j, W_{i,j})$ (or the issuer asks the users to run Update after changes in the table *arc*). The protocol is then run between user i (as IEID_P) and a verifier IEID_V as follows.

1. IEID_P computes $E = tG$, $\Lambda = \Delta_i\Theta^t$ (Δ_i is encrypted by El Gamal^{BP} public key (G, Θ)).
2. The following sub-protocol, which we call the Proving protocol, is performed.
   (a) Generate $r_1, r_2, r_3, k_0, ..., k_8 \leftarrow \mathbb{Z}_p$ and compute
   (b) IEID_P generates $r_1, r_2, r_3, k_0, ..., k_8 \in_R \mathbb{Z}_p$ and computes: $U_1 = S_i + r_1H$; $U_2 = W_{i,j} + r_2H$; $R = r_1G_1 + r_2G_2 + r_3H$; $T_1 = k_1G_1 + k_2G_2 + k_3H$; $T_2 = k_4G_1 + k_5G_2 + k_6H - k_7R$; $T_3 = k_8G$; $\Pi_1 = e(P, P)^{k_0}e(P, U_1)^{-k_7}e(P, H)^{k_4}$ $e(P_{pub}, H)^{k_1}$; $\Pi_2 = e(Q, U_2)^{-k_7}e(Q, H)^{k_5}e(Q_{pub}, H)^{k_2}$; $\Pi_3 = e(P, H)^{-k_1}$ Θ^{k_8}
   (c) $\mathsf{IEID}_P \longrightarrow \mathsf{IEID}_V$: $E, \Lambda, U_1, U_2, R, T_1, T_2, T_3, \Pi_1, \Pi_2, \Pi_3$.
   (d) $\mathsf{IEID}_P \longleftarrow \mathsf{IEID}_V$: $c \in_R \mathbb{Z}_p$.
   (e) IEID_P computes in $\mathbb{Z}_p$: $s_0 = k_0 + cx_i$; $s_1 = k_1 + cr_1$; $s_2 = k_2 + cr_2$; $s_3 = k_3 + cr_3$; $s_4 = k_4 + cr_1a_i$; $s_5 = k_5 + cr_2a_i$; $s_6 = k_6 + cr_3a_i$; $s_7 = k_7 + ca_i$; $s_8 = k_8 + ct$

(f) $\mathsf{IEID}_P \longrightarrow \mathsf{IEID}_V$: $s_0, ..., s_8$.

(g) IEID_V verifies that $T_1 \overset{?}{=} s_1G_1 + s_2G_2 + s_3H - cR$; $T_2 \overset{?}{=} s_4G_1 + s_5G_2 + s_6H - s_7R$; $T_3 \overset{?}{=} s_8G - cE$; $\Pi_1 \overset{?}{=} e(P,P)^{s_0} e(P,U_1)^{-s_7} e(P,H)^{s_4}$ $e(P_{pub},H)^{s_1} e(P,P_0)^{c} e(P_{pub},U_1)^{-c}$; $\Pi_2 \overset{?}{=} e(Q,U_2)^{-s_7} e(Q,H)^{s_5}$ $e(Q_{pub},H)^{s_2} e(Q,V_j)^{c} e(Q_{pub},U_2)^{-c}$; $\Pi_3 \overset{?}{=} e(P,H)^{-s_1} \Theta^{s_8} \Lambda^{-c} e(P,U_1)^{c}$.

Open: To open an IEID transcript $(E, \Lambda, ...)$ to find the prover, the opener computes $\Delta_i = \Lambda e(E,G)^{-x'}$ and a non-interactive zero-knowledge proof ϱ of knowledge of x so that $\Theta = e(G,G)^{x'}$ and $\Lambda/\Delta_i = e(E,G)^{x'}$ (see [19] for this proof); and finds the corresponding entry i in the table reg. If no entry is found, it returns $(0, \Delta_i, \varrho)$. Otherwise, it returns $(reg[i], \varrho)$.

Judge: Anyone can run the Judge algorithm as follows. On an output $(reg[i], \varrho)$ by the Open algorithm for an IEID transcript $(E, \Lambda, ...)$, it returns reject if verification of the proof ϱ rejects. Otherwise, it returns accept. On an output $(0, \Delta_i, \varrho)$ by Open, it returns reject if verification of the proof ϱ rejects; otherwise, it returns accept.

Revoke: To remove an user i from the group, the issuer retrieves the user's a_i from the table reg and the current group accumulated value V_j and computes a new group accumulated value $V_{j+1} = (1/(a_i + s))V_j$. The issuer appends a new entry $(a_i, 0, V_{j+1})$ on the table arc.

Update: Given access to the arc table, which currently has n rows, an user i with a membership witness $(j, W_{i,j})$ computes a new witness as follows. Its cost is about $n - j$ scalar multiplications.

```
for $(k = j+1; k++; k \leq n)$ do
    retrieve from row $k^{th}$ of $arc$ the entry $(a, b, V_k)$;
    if $b = 1$, then $W_{i,k} = V_{k-1} + (a - a_i)W_{i,k-1}$
    else $W_{i,k} = (1/(a - a_i))(W_{i,k-1} - V_k)$ end if;
end for;
return $(n, W_{i,n})$;
```

CheckArchive: Any party, after a change on the public archive, can run this algorithm as follows.

```
retrieve from the new row of $arc$ the entry $(a, b, V_k)$;
if $(b = 1)$ then return $(e(aQ + Q_{pub}, V_{k-1}) = e(Q, V_k))$
else return $(e(aQ + Q_{pub}, V_k) = e(Q, V_{k-1}))$;
```

6.2 Security

Security of the scheme $\mathcal{IE}1$ is stated in Theorems 5, 6, 7 and 8. Proofs of Theorems 6, 7 and 8 are provided in Appendix C. Theorem 5 can easily be proved by checking equations.

Theorem 5. *The identity escrow scheme with membership revocation $\mathcal{IE}1$ provides Correctness.*

Theorem 6. *The scheme $\mathcal{IE}1$ provides Anonymity if the Decisional Bilinear Diffie-Hellman assumption holds.*

Theorem 7. *The scheme $\mathcal{IE}1$ provides Traceability if the q-Strong Diffie-Hellman assumption holds, where q is the upper bound of the group size.*

Theorem 8. *The scheme $\mathcal{IE}1$ provides Non-frameability if the Discrete Logarithm assumption holds over the group $\mathbb{G}_1$ and the digital signature scheme (K_S, Sign, Ver) is UNF-CMA.*

7 Efficiency Comparison

We compare our ID-based ring signature scheme and group signature scheme with membership revocation with previous corresponding schemes at the same level of security. Comparisons for the interactive dual schemes (ID-based ad-hoc anonymous identification scheme and identity escrow scheme with membership revocation) can be similarly made. Our ID-based ring signature scheme is the first to provide constant-size signatures. Although the tuple $\mathbf{t}'$ is long, users just need to download it once, and they do not need to obtain the whole $\mathbf{t}'$. The signature size is also very much smaller than that of the current state-of-the-art normal ring signature scheme DKNS04 [24]. For elliptic curve group of 160-bit prime order, the signature size is only about 220 bytes. In the future, when higher levels of security are required, this difference even grows much larger. The same conclusion can be drawn for the size of our group signatures in comparison with those in the best schemes CL02 [20], TX03 [41], and even the ACJT00 scheme, which does not have membership revocation. Although *arc* is long (like CL02 and TX03 schemes), but the issuer can remove old entries after all users have updated. In some procedures, such as Make-GPK, it seems that for n users, the DKNS04 scheme requires only one modular exponentiation and our ID-based ring signature requires about n scalar multiplications. But the modular exponent is a product of n big numbers, so the cost of the modular exponentiation is only comparable to the total cost of our n scalar multiplications. Similar observations can be made for Make-GSK and Update. Note that most of pairing operations in IAID and IEID can be precomputed and published before the executions of the protocol. Besides, some recent papers [5, 35, 36] have shown rapid improvements in implementing the pairing operation.

We now make a specific comparison of sizes in our new group signature scheme with membership revocation with those in the ACJT00 and CL02 schemes. We assume that our scheme is implemented by an elliptic curve or hyperelliptic curve over a finite field. p is a 160-bit prime, $\mathbb{G}_1$ is a subgroup of an elliptic curve group or a Jacobian of a hyperelliptic curve over a finite field with order p and compression techniques are used. $\mathbb{G}_M$ is a subgroup of a finite field of size approximately 2^{1024}. A possible choice of these parameters can be from

Boneh *et al.*'s short signature scheme [13], where $\mathbb{G}_1$ is derived from the curve $E/GF(3^\iota)$ defined by $y^2 = x^3 - x + 1$. In addition, we assume that system parameters in the ACJT00 and CL02 schemes are $\epsilon = 1.1$, $l_p = 512$, $k = 160$, $\lambda_1 = 838$, $\lambda_2 = 600$, $\gamma_1 = 1102$ and $\gamma_2 = 840$ (as the CL02 scheme extends the ACJT00 scheme, these parameters are specified in [1]). We summarize the result in Table 1.

Table 1. Comparison of sizes (in Bytes)

	Signature	*gpk*	*gsk*	*ik*	*ok*	Membership Revocation
ACJT00	1087	768	370	128	128	No
CL02 scheme	1968	1280	370	256	128	Yes
Our scheme	410	329	188	40	20	Yes

8 Conclusions

We proposed a dynamic accumulator from bilinear pairings and showed its applications, including an ID-based ad-hoc anonymous identification scheme, an identity escrow scheme with membership revocation and their non-interactive counterparts, an ID-based ring signature scheme and a group signature scheme with membership revocation. Security proofs for these schemes were also provided. Our ID-based ring signature scheme is the first to provide signatures with constant-size (without counting the list of identities to be included in the ad-hoc group). Signature sizes in our schemes are also much smaller than those in the corresponding state-of-the-art schemes. Another advantage of our group signature scheme is perfect trapdoor-freeness, which allows sharing of public parameters among groups and organizations. The dynamic accumulator can also be used to provide membership revocation for other primitives, such as traceable signature schemes and anonymous credential systems. We also provided a generalized model of accumulators and formal models of ID-based ad-hoc anonymous identification schemes and identity escrow schemes with membership revocation.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. CRYPTO 2000, Springer-Verlag, LNCS 1880, pp. 255-270.
2. G. Ateniese and G. Tsudik. Quasi-efficient revocation of group signatures. http://eprint.iacr.org/2001/101, 2001.
3. G. Ateniese, and B. de Medeiros. Efficient Group Signatures without Trapdoors. ASIACRYPT 2003, Springer-Verlag, LNCS 2894, pp. 246-268.

4. N. Baric and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. EUROCRYPT 1997, Springer-Verlag, LNCS 1233, pp. 480-494.
5. P. Barreto, B. Lynn, and M. Scott. On the Selection of Pairing-Friendly Groups. SAC 2003.
6. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions. EUROCRYPT 2003, Springer-Verlag, LNCS 2656, pp. 614-629.
7. M. Bellare, C. Namprempre, and G. Neven. Security Proofs for Identity-Based Identification and Signature Schemes. EUROCRYPT 2004, Springer-Verlag, LNCS 3027, pp. 268-286.
8. M. Bellare, H. Shi, and C. Zhang. Foundations of Group Signatures: The Case of Dynamic Groups. Cryptology ePrint Archive: Report 2004/077.
9. J. Benaloh and M. de Mare. One-way accumulators: A decentralized alternative to digital signatures. EUROCRYPT 1993, Springer-Verlag, LNCS 765, pp. 274-285.
10. D. Boneh, and X. Boyen. Short Signatures Without Random Oracles. EUROCRYPT 2004, Springer-Verlag, LNCS 3027, pp. 56-73.
11. D. Boneh, and X. Boyen. Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. EUROCRYPT 2004, Springer-Verlag, LNCS 3027, pp. 223-238.
12. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. CRYPTO 2001, Springer-Verlag, LNCS 2139, pp. 213-229.
13. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. ASIACRYPT 2001, Springer-Verlag, LNCS 2248, pp.514-532.
14. X. Boyen. Multipurpose Identity-Based Signcryption (A Swiss Army Knife for Identity-Based Cryptography). CRYPTO 2003, Springer-Verlag, LNCS 2729, pp. 383-399.
15. E. Bresson and J. Stern. Group signatures with efficient revocation. PKC 2001, Springer-Verlag, LNCS 1992, pp. 190-206.
16. E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to adhoc groups. CRYPTO 2002, Springer-Verlag, LNCS 2442, pp. 465-480.
17. J. Camenisch, and M. Stadler. Efficient group signature schemes for large groups. CRYPTO 1997, Springer-Verlag, LNCS 1296.
18. J. Camenisch. Efficient and generalized group signatures. EUROCRYPT 1997, Springer-Verlag, LNCS 1233, pp. 465-479.
19. J. Camenisch, and M. Michels. A group signature scheme with improved efficiency. ASIACRYPT 1998, Springer-Verlag, LNCS 1514.
20. J. Camenisch, and A. Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. CRYPTO 2002, Springer-Verlag, LNCS 2442, pp. 61-76.
21. J. Cha, and J. Cheon. An Identity-Based Signature from Gap Diffie-Hellman Groups. PKC 2003, Springer-Verlag, LNCS 2567, pp. 18-30.
22. D. Chaum, and E. van Heyst. Group signatures. CRYPTO 1991, LNCS 547, Springer-Verlag.
23. L. Chen, and T. P. Pedersen. New group signature schemes. EUROCRYPT 1994, Springer-Verlag, LNCS 950, pp. 171-181.
24. Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous Identification in Ad Hoc Groups. EUROCRYPT 2004, Springer-Verlag, LNCS 3027, pp. 609-626.
25. A. Fiat, and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. CRYPTO 1986, Springer-Verlag, LNCS 263, pp. 186-194.

26. O. Goldreich. Foundations of Cryptography, Basic Applications. Cambridge University Press 2004.
27. A. Kiayias, Y. Tsiounis and M. Yung. Traceable Signatures. EUROCRYPT 2004, Springer-Verlag, LNCS 3027, pp. 571-589.
28. J. Killian, and E. Petrank. Identity escrow. CRYPTO 1998, Springer-Verlag, LNCS 1642, pp. 169-185.
29. S. Kim, S. Park, and D. Won. Convertible group signatures. ASIACRYPT 1996, Springer-Verlag, LNCS 1163, pp. 311-321.
30. M. Michels. Comments on some group signature schemes. TR-96-3-D, Department of Computer Science, University of Technology, Chemnitz-Zwickau, Nov. 1996.
31. S. Mitsunari, R. Sakai, and M. Kasahara. A new traitor tracing. IEICE Trans. Vol. E85-A, No.2, pp.481-484, 2002.
32. L. Nguyen. Accumulators from Bilinear Pairings and Applications. Cryptographers' Track, RSA (CT-RSA) 2005, Springer-Verlag, LNCS.
33. L. Nguyen, and R. Safavi-Naini. Efficient and Provably Secure Trapdoor-free Group Signature Schemes from Bilinear Pairings. ASIACRYPT 2004, Springer-Verlag, LNCS.
34. R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. ASIACRYPT 2001, Springer-Verlag, LNCS 2248, pp.552-565.
35. M. Scott, and P. Barreto. Compressed Pairings. Cryptology ePrint Archive, Report 2004/032.
36. M. Scott. Computing the Tate Pairing. Manuscript.
37. A. Shamir, Identity-based cryptosystems and signature schemes. CRYPTO 1984, LNCS 196, Springer-Verlag, pp. 47-53.
38. D. Song. Practical forward secure group signature schemes. ACM CCS 2001, ACM press, pp. 225-234.
39. V. To, R. Safavi-Naini, and F. Zhang. New traitor tracing schemes using bilinear map. DRM Workshop 2003.
40. Y. Tsiounis and M. Yung. On the security of El Gamal based encryption. First International Workshop on Practice and Theory in Public Key Cryptography - PKC '98, pages 117-134, LNCS 1431, 1998.
41. G. Tsudik, and S. Xu. Accumulating Composites and Improved Group Signing. ASIACRYPT 2003, Springer-Verlag, LNCS 2894, pp. 269-286.
42. F. Zhang, and K. Kim. ID-Based Blind Signature and Ring Signature from Pairings. ASIACRYPT 2002, Springer-Verlag, LNCS 2501, pp. 533-547.

A Preliminaries

A.1 Complexity Assumptions

The q-SDH assumption originates from a weaker assumption introduced by Mitsunari et al. [31] to construct traitor tracing schemes [39] before being well stated by Boneh and Boyen [10]. It intuitively means that there is no PPT algorithm that can compute a pair $(c, \frac{1}{s+c}P)$, where $c \in \mathbb{Z}_p$, from a tuple $(P, sP, \ldots, s^qP)$, where $s \in_R \mathbb{Z}_p^*$.

q-Strong Diffie-Hellman (q-SDH) Assumption. *For every PPT algorithm $\mathcal{A}$, the following function $Adv_{\mathcal{A}}^{\mathsf{q\text{-}SDH}}(l)$ is negligible.*

$$Adv_{\mathcal{A}}^{\mathsf{q\text{-}SDH}}(l) = \mathsf{Pr}[(\mathcal{A}(\mathbf{t}, P, sP, \ldots, s^qP) = (c, \frac{1}{s+c}P)) \wedge (c \in \mathbb{Z}_p)]$$

where $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P) \leftarrow \mathcal{G}(1^l)$ *and* $s \leftarrow \mathbb{Z}_p^*$.

Intuitively, the DBDH assumption [11] states that there is no PPT algorithm that can distinguish between a tuple $(aP, bP, cP, e(P,P)^{abc})$ and a tuple (aP, bP, cP, Γ), where $\Gamma \in_R \mathbb{G}_M^*$ (i.e., chosen uniformly random from $\mathbb{G}_M^*$) and $a, b, c \in_R \mathbb{Z}_p^*$. It is defined as follows.

Decisional Bilinear Diffie-Hellman (DBDH) Assumption. *For every PPT algorithm* $\mathcal{A}$*, the following function* $Adv_{\mathcal{A}}^{\mathsf{DBDH}}(l)$ *is negligible.*

$$Adv_{\mathcal{A}}^{\mathsf{DBDH}}(l) = |\mathsf{Pr}[\mathcal{A}(\mathbf{t}, aP, bP, cP, e(P,P)^{abc}) = 1] - \mathsf{Pr}[\mathcal{A}(\mathbf{t}, aP, bP, cP, \Gamma) = 1]|$$

where $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P) \leftarrow \mathcal{G}(1^l)$*,* $\Gamma \leftarrow \mathbb{G}_M^*$ *and* $a, b, c \leftarrow \mathbb{Z}_p^*$.

The Discrete Logarithm assumption in the group $\mathbb{G}_1$ is as follows.

Discrete Logarithm (DL) Assumption. *For every PPT algorithm* $\mathcal{A}$*, the following function* $Adv_{\mathcal{A}}^{\mathsf{DL}}(l)$ *is negligible.*

$$Adv_{\mathcal{A}}^{\mathsf{DL}}(l) = \mathsf{Pr}[\mathcal{A}(\mathbf{t}, Q, xQ) = x]$$

where $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P) \leftarrow \mathcal{G}(1^l)$*,* $Q \leftarrow \mathbb{G}_1^*$ *and* $x \leftarrow \mathbb{Z}_p^*$.

We now present the Decisional Diffie-Hellman assumption in the group $\mathbb{G}_M$. It can also be stated in many other cyclic groups of prime order, such as the subgroup of order p of group $\mathbb{Z}_{p'}$, where p, p' are large primes and $p \mid p' - 1$.

Decisional Diffie-Hellman (DDH) Assumption. *For every PPT algorithm* $\mathcal{A}$*, the following function* $Adv_{\mathcal{A}}^{\mathsf{DDH}}(l)$ *is negligible.*

$$Adv_{\mathcal{A}}^{\mathsf{DDH}}(l) = |\mathsf{Pr}[\mathcal{A}(\mathbf{t}, \Gamma, \Gamma^r, \Gamma^x, \Gamma^{xr}) = 1] - \mathsf{Pr}[\mathcal{A}(\mathbf{t}, \Gamma, \Gamma^r, \Gamma^x, \Gamma^s) = 1]|$$

where $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P) \leftarrow \mathcal{G}(1^l)$*,* $\Gamma \leftarrow \mathbb{G}_M^*$ *and* $x, r, s \leftarrow \mathbb{Z}_p^*$.

We also present a Decisional Diffie-Hellman Variant assumption and show that it is weaker than DBDH assumption in Theorem 9. This assumption is very similar to the DDH assumption, but it works over groups $\mathbb{G}_1$ and $\mathbb{G}_M$.

Decisional Diffie-Hellman Variant (DDHV) Assumption. *For every PPT algorithm* $\mathcal{A}$*, the following function* $Adv_{\mathcal{A}}^{\mathsf{DDHV}}(l)$ *is negligible.*

$$\begin{aligned} Adv_{\mathcal{A}}^{\mathsf{DDHV}}(l) = |&\mathsf{Pr}[\mathcal{A}(\mathbf{t}, P, rP, e(P,P)^x, e(P,P)^{xr}) = 1] - \\ &\mathsf{Pr}[\mathcal{A}(\mathbf{t}, P, rP, e(P,P)^x, e(P,P)^s) = 1]| \end{aligned}$$

where $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P) \leftarrow \mathcal{G}(1^l)$ *and* $x, r, s \leftarrow \mathbb{Z}_p^*$.

Theorem 9. *If the DBDH assumption holds then the DDHV assumption also holds.*

Proof. To prove the theorem, we show that if a PPT algorithm $\mathcal{A}$ has non-negligible $Adv_{\mathcal{A}}^{\mathsf{DDHV}}(l)$ (i.e., DDHV assumption does not hold), then we can build an algorithm $\mathcal{B}$ that has non-negligible $Adv_{\mathcal{B}}^{\mathsf{DBDH}}(l)$ (i.e., DBDH assumption does not hold). Suppose $a, b, c \in \mathbb{Z}_p^*$ and $\Gamma \in \mathbb{G}_M^*$, we observe that if a and b are uniformly distributed in $\mathbb{Z}_p^*$, then $x = ab$ is also uniformly distributed in $\mathbb{Z}_p^*$

and if $\varGamma$ is uniformly distributed in $\mathbb{G}_M^*$, then s is also uniformly distributed in $\mathbb{Z}_p^*$, where $\varGamma = e(P,P)^s$. So to distinguish between $(aP, bP, cP, e(P,P)^{abc})$ and $(aP, bP, cP, \varGamma)$, the algorithm $\mathcal{B}$ can simply return the outputs by $\mathcal{A}$ when it takes as input $(\mathbf{t}, P, cP, e(aP, bP), e(P,P)^{(ab)c})$ or $(\mathbf{t}, P, cP, e(aP, bP), \varGamma)$.

A.2 Bilinear Pairing versions for El Gamal public key system - El Gamal^{BP}

Key generation: Let $p, \mathbb{G}_1, \mathbb{G}_M, e$ be bilinear pairing parameters, as defined above, and G be a generator of $\mathbb{G}_1$. Suppose $x \in_R \mathbb{Z}_p^*$ and $\varTheta = e(G,G)^x$. The public key $pk = (G, \varTheta)$ and the secret key is $sk = x$.

Encryption: Plaintext $\varDelta \in \mathbb{G}_M$ can be encrypted by choosing an $t \in_R \mathbb{Z}_p^*$ and computing the ciphertext $(E, \varLambda) = (tG, \varDelta\varTheta^t)$.

Decryption: Ciphertext $(E, \varLambda)$ can be decrypted as $\varDelta = \varLambda / e(E,G)^x$.

Security: The security of El Gamal^{BP} system is stated in Theorem 10. The first statement can be proved exactly the same way as the proof for the El Gamal encryption scheme [40], except that it is based on DDHV assumption instead of DDH assumption. The second statement can be seen as a result of the first statement and Theorem 9.

Theorem 10. *El Gamal^{BP} encryption scheme is IND-CPA if and only if the DDHV assumption holds. El Gamal^{BP} encryption scheme is IND-CPA if the DBDH assumption holds.*

A.3 Digital Signatures

(Syntax) A digital signature scheme consists of three PT algorithms (K_S, $Sign$, Ver). The key generation algorithm K_S on input 1^l outputs (pk, sk) where pk is a public verification key, sk is the secret signing key and l is a security parameter. The signing algorithm $Sign$ takes as input the secret key sk and a message and outputs a signature for the message. The verification algorithm Ver takes as input the public key pk, a message and a signature and outputs accept if the signature is valid for the message, or reject otherwise.

(Security) Here we briefly recall definitions and notions of security for digital signature schemes. More details can be found in [26]. Digital signatures should be unforgeable against chosen message attacks (UNF-CMA). A *chosen message attack* means that the adversary may obtain signatures corresponding to messages the adversary adaptively chooses. Unforgeability intuitively means that the adversary can not output a valid signature for a message, for which it has not requested a signature during the attacks.

B Models

B.1 Security Requirements for ID-based Ad-hoc Anonymous Identitification schemes

Correctness. This property intuitively requires that in any execution of the IAID protocol, if IAID_P is given a group secret key corresponding to the common input group public key, then IAID_V outputs accept with overwhelming probability.

Soundness. This requirement is modelled by a game being played between an honest dealer and an adversary, and the adversary can send queries to the transcript oracle, who takes as input an identity of a user and a set of other identities and outputs a valid transcript of the IAID protocol's execution, where the user anonymously proves his membership of a group formed by the set of identities and himself.

The game is played as follows. The honest dealer first runs the Setup algorithm and sends the resulting public parameters to the adversary. Then, the adversary can adaptively send queries to the transcript oracle during the game, even during the execution of the IAID protocol later. At a point, the adversary returns a target group of identities and then executes the IAID protocol with the honest dealer, in the role of the prover and the verifier respectively, on common inputs of the public parameters and the group public key corresponding to the target group. The adversary wins the game if the honest dealer outputs accept, and the adversary does not have a private key corresponding to an identity in the target group. The ID-based ad hoc anonymous identification scheme provides Soundness if the probability that the adversary wins the game is negligible.

Anonymity. This requirement is modelled by a game being played between an honest dealer and an adversary, and the adversary can send only one query to a challenge oracle. This oracle takes as input two identity-privatekey pairs and a set of other identities and returns a transcript of the IAID protocol's execution, where the prover randomly uses one of the two private keys to prove membership of a group formed by the set of identities and two identities from the pairs.

The honest dealer first runs the Setup algorithm and sends the resulting public parameters to the adversary. Then, the adversary can find many pairs of identity and private key during the game, even after receiving the challenge transcript from the challenge oracle later. At a point, he queries the challenge oracle and gets a challenge transcript. The adversary then can do any experiments with the system before outputting an identity of the two pairs he queried the challenge oracle. The adversary wins the game if the identity he outputs corresponds to the private key the challenge oracle used to generate the challenge transcript. The ID-based ad hoc anonymous identification scheme provides Anonymity if the probability that the adversary wins the game is negligibly larger than a random guess. If the condition holds even assuming that the adversary has unlimited computing resources, then the scheme is said to provide *Unconditional Anonymity.*

Similar to the model in [24], we just define these security notions strong enough to capture security for ID-based ring signatures obtained by applying the Fiat-Shamir heuristics.

B.2 Identity Escrow schemes with Membership Revocation

We first describe participants and procedures in an identity escrow schemes with membership revocation (IEMR) and then model oracles accessible to the adversaries and finally define formal security requirements. This model is based on the BSZ04 model [8].

Participants and Procedures An IEMR scheme consists of a trusted party for initial set-up, two group managers (the issuer and the opener), and users, each with a unique identity $i \in \mathbb{N}$ (the set of positive integers). Each user can join the group and become a group member. There are two publicly readable tables *upk* and *arc* (public archive) and a table *reg* is readable by the opener and writable by the issuer. The scheme is specified as a tuple $\mathcal{IE}$ =(GKg, UKg, Join, Iss, IEID_P, IEID_V, Open, Judge, Revoke, Update, $\mathsf{CheckArchive}$) of PT algorithms, which are described as follows.

- GKg: In the setup phase, the trusted party runs the group-key generation algorithm GKg that takes as input a security parameter 1^l and outputs a triple of keys (gpk, ik, ok), where ik is given to the issuer, and ok is given to the opener. The group public key gpk for signature verification is published.
- UKg: A user i runs the user-key generation algorithm UKg that takes as input a security parameter 1^l and outputs a personal public and private key pair $(upk[i], usk[i])$, where $upk[i]$ is stored in the table upk.
- Join, Iss: These interactive algorithms are performed by a user, who has a personal public and private key pair, and the issuer as two sides of a group-joining protocol. Each takes as input an incoming message (unless the party is initiating the protocol) and a current state, and outputs an outgoing message, an updated state, and a decision which is one of accept, reject, cont. The communication is assumed to be secure (i.e., private and authenticated), and the user i is assumed to send the first message. If the issuer accepts, it makes an entry $reg[i]$ for i, in the registration table reg, and fills this entry with a new membership certificate, which is the final state output by Iss. It also update the arc table and sends i a membership witness. If i accepts, it stores the final state output by Join as its membership secret key $gsk[i]$ and membership witness $w[i]$.
- IEID_P, IEID_V: This IEID protocol takes as common input the group public key and the prover also has a membership secret key and his updated membership witness. The verifier output accept if and only if the prover is a current member of the group.
- Open: The opener, has read-access to the registration table reg, and can run the deterministic opening algorithm Open that takes as input the opening key ok, the registration table reg, and a valid transcript of the IEID protocol

under gpk and returns a pair (i, τ), where i is a non-negative integer and τ is a proof of this claim. If $i \geq 1$, the algorithm is claiming that the group member i took part in the protocol execution that produced the transcript, and if $i = 0$, it is claiming that no group member produced ω.

- Judge: Anyone can run the deterministic judge algorithm Judge that takes as input the group public key gpk, an integer $j \geq 1$, the public key $upk[j]$ of the user j (this is an empty string if this user has no public key), a valid transcript of the IEID protocol, and a proof-string τ. It aims to check that τ is a proof that user j took part in the protocol execution that produced the transcript. The judge will base its verification on the public key of j. As the IEID protocol should be simulatable, the transcript is assumed to be from a reliable source.
- Revoke: The issuer, with access to the table arc, can apply this algorithm to a user identity, that removes the user out of the group by updating the table arc.
- Update: This algorithm takes as input the group public key, an group member identity and the table arc and returns the updated membership witness for the member.
- CheckArchive: This algorithm can be run by any party after each change in the public archive. It returns accept if and only if the issuer updates the archive arc correctly.

The Oracles The security requirements are formulated via experiments in which an adversary capabilities are modelled by providing it access to certain oracles. It is assumed that each experiment has run GKg on input 1^l to obtain keys gpk, ik, ok that are used by the oracles, and all entries of the tables upk, reg and arc are assumed initially to be empty strings. It is also assumed that the experiment maintains the following sets which are initially empty and manipulated by the oracles: a set HU of honest users and a set CU of corrupted users. Different experiments will provide the adversary with different subsets of the following set of oracles. Compared to the BSZ04 model, there are two more oracles, RevokeU$(\cdot)$ and Witness$(\cdot)$, and there is no Open oracle. The reason for omitting Open is that in all of the recent efficient IEMR schemes, the IEID has the form (e, ρ_1, c, ρ_2), where e is an encryption of the user's identity-bound information and (ρ_1, c, ρ_2) is a $(commitment, challenge, response)$ tuple of a zero-knowledge protocol. So given a challenge transcript (e, ρ_1, c, ρ_2), the adversary could simulate another transcript $(e, \rho_1', c', \rho_2')$ and used Open oracle to find the prover of the transcript.

- AddU$(\cdot)$: This add user oracle with argument an identity $i \in \mathbb{N}$ allows the adversary to add i to the group as an honest user. The oracle adds i to the set HU of honest users, and picks a personal public and private key pair $(upk[i], usk[i])$ for i. It then executes the group-joining protocol by running Join (on behalf of i, initialized with $gpk, upk[i], usk[i]$) and Iss (on behalf of the issuer, initialized with $gpk, ik, i, upk[i]$). When Iss accepts, its final state is recorded as entry $reg[i]$ in the registration table and an update on the arc

table. When Join accepts, its final state is recorded as the membership secret key $gsk[i]$ and witness $w[i]$ of i. The calling adversary is returned $upk[i]$, but not the transcript of the interaction generated by the oracle.

- CrptU$(\cdot, \cdot)$: This corrupt user oracle with arguments an identity $i \in \mathbb{N}$ and a string upk allows the adversary to corrupt user i and set its personal public key $upk[i]$ to the value upk chosen by the adversary. The oracle initializes the issuer's state in anticipation of a group-joining protocol with i (so i is not yet in the group).
- SndToI$(\cdot, \cdot)$: Having corrupted user i, the adversary can use this send to issuer oracle to engage in a group-joining protocol with the honest issuer, itself playing the role of i and not necessarily executing the interactive algorithm Join prescribed for an honest user. The adversary provides the oracle with i and a message M_{in} to be sent to the issuer. The oracle, which maintains the issuer's state (the latter having been initialized by an earlier call to CrptU$(i, \cdot)$), computes a response as per Iss, returns the outgoing message to the adversary, sets entry $reg[i]$ of the registration table to Iss's final state and updates arc if the latter accepts.
- SndToU$(\cdot, \cdot)$: In some definitions we will want to consider an adversary that has corrupted the issuer. This send to user oracle can be used by such an adversary to engage in a group-joining protocol with an honest user, itself playing the role of the issuer and not necessarily executing the interactive algorithm Iss prescribed for the honest issuer. The adversary provides the oracle with i and a message M_in to be sent to i. The oracle maintains the state of user i, initializing this the first time it is called by choosing a personal public and private key pair for i, computes a response as per Join, returns the outgoing message to the adversary, and sets the membership secret key and membership witness of i to Join's final state if the latter accepts.
- USK$(\cdot)$: The adversary can call this user secret keys oracle with argument the identity $i \in \mathbb{N}$ of a user to expose both the membership secret key $gsk[i]$ and the personal private key $usk[i]$ of this user.
- RReg$(\cdot)$: The adversary can read the contents of entry i of the registration table reg by calling this read registration table oracle with argument $i \in \mathbb{N}$.
- WReg$(\cdot, \cdot)$: In some definitions we will allow the adversary to write/modify the contents of entry i of the registration table reg by calling this write registration table oracle with argument $i \in \mathbb{N}$.
- IEID$_P(\cdot)$: The adversary can use this oracle to make a user i (specified in the argument) perform the IEID protocol with a honest verifier, and get the transcript of the protocol execution.
- Ch$(b, \cdot, \cdot)$: A challenge oracle provides to an adversary attacking anonymity, and depending on a challenge bit b set by the overlying experiment. The adversary provides a pair i_0, i_1 of identities and obtains the transcript of an IEID protocol's execution under the membership secret key of i_b, as long as both i_0, i_1 are honest users with defined membership secret keys.
- RevokeU$(\cdot)$: This revoke user oracle makes the issuer run the Revoke algorithm to remove a user (specified in the argument) from the group. The user is also removed from HU $\cup$ SU.

- Witness$(\cdot)$: This oracle returns the membership witness of a user specified in the argument.

Security Requirements The security requirements are modelled by experiments, which are quite the same as experiments in [8]. An IEMR scheme must satisfy the following security requirements.

- Correctness: In this experiment the adversary is not computationally restricted and has access to AddU$(\cdot)$ and RReg$(\cdot)$ oracles. The adversary returns the identity of an honest group member and the group member performs the IEID protocol with an honest verifier. The correctness condition holds if the probability that one of the following steps fails is 0: IEID$_V$ accepts; Open algorithm returns the correct group member; and Judge algorithm accepts the proof returned by Open algorithm.
- Anonymity: The anonymity experiment involves a PT adversary, who knows the issuing key ik and has access to Ch$(b, \cdot, \cdot)$, SndToI$(\cdot, \cdot)$, SndToU$(\cdot, \cdot)$, WReg$(\cdot, \cdot)$, USK$(\cdot)$, CrptU$(\cdot, \cdot)$, RevokeU$(\cdot)$ and Witness$(\cdot)$ oracles. The adversary provides the Ch$(b, \cdot, \cdot)$ oracle identities of two honest members and is returned a transcript of the IEID protocol executed by one of the members (according to bit b). The anonymity condition holds if the probability that the adversary can correctly guess the bit b is negligible. Note that the adversary can not send the identity of challenge members to RevokeU$(\cdot, \cdot)$ oracle and the opener is uncorrupted.
- Traceability: The traceability experiment involves a PT adversary, who knows the opening key ok and has access to AddU$(\cdot)$, RReg$(\cdot)$, SndToI$(\cdot, \cdot)$, USK$(\cdot)$, CrptU$(\cdot, \cdot)$, RevokeU$(\cdot)$ and Witness$(\cdot)$ oracles. The adversary then performs the IEID protocol with a honest verifier. The traceability condition holds if the probability that all of the following steps succeed is negligible: IEID$_V$ accepts; Open algorithm can not return the identity of the prover or Open algorithm can return the identity of the prover but Judge algorithm rejects the proof returned by Open algorithm. Note that the issuer is uncorrupt and the opener is at wost partially corrupted, that means he performs correctly but his secret key is available to the adversary.
- Non-frameability: The non-frameability experiment involves a PT adversary, who knows the opening key ok and the issuing key ik, and has access to SndToU$(\cdot, \cdot)$, WReg$(\cdot, \cdot)$, GSig$(\cdot, \cdot)$, USK$(\cdot)$, CrptU$(\cdot, \cdot)$, RevokeU$(\cdot)$ and Witness$(\cdot)$ oracles. The adversary then performs the IEID protocol with a honest verifier and returns an identity of a honest user and a proof of a opening claim. The non-frameability condition holds if the probability that all of the following steps succeed is negligible: IEID$_V$ accepts; and Judge algorithm accepts the proof returned by the adversary, which claims that the honest user is the prover. Note that the adversary can not send the challenge user's identity to USK$(\cdot)$.

C Proofs

C.1 Proof of Theorem 2

Suppose there is a PPT adversary $\mathcal{A}$ that can break Collision-Resistance property of $\mathcal{DA}1$, we show a construction of a PPT adversary $\mathcal{B}$ that can break the q-SDH assumption. Suppose a tuple $challenge = (P, zP, \ldots, z^q P)$ is given, where $z \in_R \mathbb{Z}_p^*$, we show that $\mathcal{B}$ can compute $(c, 1/(z+c)P)$, where $c \in \mathbb{Z}_p$ with non-negligible probability. Let $u \in_R \mathbb{Z}_p^*$, as $\mathcal{A}$ breaks Collision-Resistance property of $\mathcal{DA}1$, he can output $\mathbf{X} = \{x_1, ..., x_k\} \subset \mathbb{Z}_p \backslash \{-z\}$, $x \in \mathbb{Z}_p \backslash (\{-z\} \cup \mathbf{X})$ and $W \in \mathbb{G}_1$ such that $k \leq q$ and $(x+z)W = \prod_{i=1}^{k}(x_i + z)uP$. From this equation and the tuple *challenge*, $(1/(x+z))P$ can be computed and hence the q-SDH assumption is broken.

C.2 Proofs of Theorem 3

We first prove the zero-knowledge property of IAID, as stated in Lemma 2.

Lemma 2. *Under the Discrete Log assumption on $\mathbb{G}_1$, the IAID protocol is a (honest-verifier) perfect zero-knowledge proof of knowledge (h_{id}, R_{id}, W) satisfying equations $e(h_{id}Q + Q_{pub}, R_{id}) = e(Q, Q)$ and $e(h_{id}P + P_{pub}, W) = e(P, V)$.*

Proof. As the proof for completeness is straightforward, we present the proofs for Soundness and Zero-knowledge property only, as follows.
Soundness: If the protocol accepts with non-negligible probability, we show that under the Discrete Log assumption on $\mathbb{G}_1$, a PPT prover must have the knowledge of (h_{id}, R_{id}, W) with the relations stated in the lemma. Suppose the protocol accepts for the same commitment (U_1, U_2, R, T_1, T_2, Π_1, Π_2) with two different pairs of challenges and responses $(c, s_1, ...s_7)$ and $(c', s'_1, ..., s'_7)$. Let $f_i = \frac{s_i - s'_i}{c - c'}$, $i = 1, ..., 7$, then $R = f_1G_1 + f_2G_2 + f_3H$; $f_7R = f_4G_1 + f_5G_2 + f_6H$; $e(Q_{pub}, U_1)e(Q, Q)^{-1} = e(Q, U_1)^{-f_7}e(Q, H)^{f_4}e(Q_{pub}, H)^{f_1}$; $e(P_{pub}, U_2)e(P, V)^{-1} = e(P, U_2)^{-f_7}e(P, H)^{f_5}e(P_{pub}, H)^{f_2}$.

From the first two equations, the prover has $\mathcal{O} = (f_4 - f_7f_1)G_1 + (f_5 - f_7f_2)G_2 + (f_6 - f_7f_3)H$ ($\mathcal{O}$ is the identity element of $\mathbb{G}_1$). Under the Discrete Log assumption on $\mathbb{G}_1$, it implies that $f_4 = f_7f_1$ and $f_5 = f_7f_2$.

Let $h_{id} = f_7$, $R_{id} = U_1 - f_1H$ and $W = U_2 - f_2H$, then $e(h_{id}Q + Q_{pub}, R_{id}) = e(Q, Q)$ and $e(h_{id}P + P_{pub}, W) = e(P, V)$. So the prover has the knowledge of (h_{id}, R_{id}, W) satisfying these relations.
Zero-knowledge: The simulator chooses $c, s_1, ...s_7 \in_R \mathbb{Z}_p$, $U_1, U_2, R \in_R \mathbb{G}_1$ and computes $T_1 = s_1G_1 + s_2G_2 + s_3H - cR$; $T_2 = s_4G_1 + s_5G_2 + s_6H - s_7R$; $\Pi_1 = e(Q, U_1)^{-s_7}e(Q, H)^{s_4}e(Q_{pub}, H)^{s_1}e(Q, Q)^c e(Q_{pub}, U_1)^{-c}$; $\Pi_2 = e(P, U_2)^{-s_7}$ $e(P, H)^{s_5}e(P_{pub}, H)^{s_2}e(P, V)^c e(P_{pub}, U_2)^{-c}$. We can see that the distribution of the simulation is the same as the distribution of the real transcript.

Theorems 3 can be easily concluded from the results of the above lemma. Correctness and Unconditional Anonymity is based on the completeness and perfect zero-knowledge properties of the IAID protocol, respectively. Soundness of $\mathcal{IA}1$ is based on the soundness property of the IAID protocol, the collision-resistance property of the accumulator $\mathcal{DA}1$ and the fact that: if a PPT adversary $\mathcal{A}$ can compute a new pair of hashed identity and private key $(h^*_{id}, R^*_{id} = 1/(h^*_{id} + s_m)Q)$ from a set of $\{(h^{(i)}_{id}, R^{(i)}_{id} = 1/(h^{(i)}_{id} + s_m)Q)\}^q_{i=1}$, then $\mathcal{A}$ can break the q-SDH assumption.

C.3 Security proofs for $\mathcal{IE}1$

Before proving security of $\mathcal{IE}1$, we prove the Zero-knowledge property of the Proving protocol in IEID protocol and the Coalition-Resistance of $\mathcal{IE}1$. In our definition, Coalition-Resistance intuitively means that a group of colluded members, with knowledge of the opening key and access to some oracles (as in the Traceability requirement), should not be able to generate a new valid pair of a membership secret key and a current membership witness. For an IEMR scheme $\mathcal{IE}$, a PPT adversary $\mathcal{A}$, a PPT predicate $\mathcal{U}$ that can determine the validity of a pair of a membership secret key and a current membership witness, and any security parameter $l \in \mathbb{N}$, the formula of the experiment for Coalition-Resistance is as follows.

Experiment $Exp^{\mathsf{coal.re}}_{\mathcal{IE},\mathcal{A},\mathcal{U}}(l)$

$(gpk, ik, ok) \leftarrow \mathsf{GKg}(1^l)$; $\mathsf{CU} \leftarrow \emptyset$; $\mathsf{HU} \leftarrow \emptyset$

$(gsk', w') \leftarrow \mathcal{A}(gpk, ok : \mathsf{CrptU}(\cdot,\cdot), \mathsf{SndToI}(\cdot,\cdot), \mathsf{AddU}(\cdot), \mathsf{RReg}(\cdot), \mathsf{USK}(\cdot), \mathsf{RevokeU}(\cdot), \mathsf{Witness}(\cdot))$

If $(gsk', w') \in \{(gsk[i], \mathsf{Update}(i))|\ i \in \mathsf{CU} \cup \mathsf{HU}\}$ then return 0 else return $\mathcal{U}(gpk, arc, gsk', w')$

The scheme $\mathcal{IE}$ provides Coalition-Resistance if the following function $Adv^{\mathsf{coal.re}}_{\mathcal{IE},\mathcal{A},\mathcal{U}}(l)$ is negligible.

$$Adv^{\mathsf{coal.re}}_{\mathcal{IE},\mathcal{A},\mathcal{U}}(l) = \mathsf{Pr}[Exp^{\mathsf{coal.re}}_{\mathcal{IE},\mathcal{A},\mathcal{U}}(l) = 1]$$

Lemma 3. *Under the Discrete Log assumption on $\mathbb{G}_1$, the Proving protocol in the IEID protocol is a (honest-verifier) perfect zero-knowledge proof of knowledge of $W_{i,j}$, (a_i, S_i), x_i and t such that $e(a_iQ + Q_{pub}, W_{i,j}) = e(Q, V_j)$, $e(a_iP + P_{pub}, S_i) = e(P, x_iP + P_0)$, $E = tG$ and $\Lambda = e(P, S_i)\Theta^t$.*

Proof. As the proof for completeness is straightforward, we present the proofs for Soundness and Zero-knowledge property only, as follows.

$\mathsf{Soundness}$: If the protocol accepts with non-negligible probability, we show that under the Discrete Log assumption on $\mathbb{G}_1$, a PPT prover must have the knowledge of $W_{i,j}$, (a_i, S_i), x_i and t with the relations stated in the lemma. Suppose the protocol accepts for the same commitment $(U_1, U_2, R, T_1, T_2, T_3, \Pi_1, \Pi_2, \Pi_3)$ with two different pairs of challenges and responses $(c, s_0, ...s_8)$ and $(c', s'_0, ..., s'_8)$. Let $f_i = \frac{s_i - s'_i}{c - c'}$, $i = 0, ..., 8$, then $R = f_1G_1 + f_2G_2 + f_3H$; $f_7R = f_4G_1 + f_5G_2 + f_6H$; $E = f_8G$; $e(P_{pub}, U_1)e(P, P_0)^{-1} = e(P, P)^{f_0}e(P, U_1)^{-f_7}e(P, H)^{f_4}$

$e(P_{pub}, H)^{f_1}$; $e(Q_{pub}, U_2)e(Q, V_j)^{-1} = e(Q, U_2)^{-f_7} e(Q, H)^{f_5} e(Q_{pub}, H)^{f_2}$; $\Lambda e(P, U_1)^{-1} = e(P, H)^{-f_1} \Theta^{f_8}$.

From the first two equations, the prover has $\mathcal{O} = (f_4 - f_7 f_1)G_1 + (f_5 - f_7 f_2)G_2 + (f_6 - f_7 f_3)H$ ($\mathcal{O}$ is the identity element of $\mathbb{G}_1$). Under the Discrete Log assumption on $\mathbb{G}_1$, it implies that $f_4 = f_7 f_1$ and $f_5 = f_7 f_2$.

Let $a_i = f_7$, $S_i = U_1 - f_1 H$, $x_i = f_0$, $t = f_8$ and $W_{i,j} = U_2 - f_2 H$, then $E = tG$, $\Lambda = e(P, S_i)\Theta^t$, $e(a_i P + P_{pub}, S_i) = e(P, x_i P + P_0)$ and $e(a_i Q + Q_{pub}, W_{i,j}) = e(Q, V_j)$. So the prover has the knowledge of $W_{i,j}$, (a_i, S_i), x_i and t satisfying these relations.

Zero-knowledge: The simulator chooses $c, s_0, ...s_8 \in_R \mathbb{Z}_p$, $U_1, U_2, R \in_R \mathbb{G}_1$ and computes $T_1 = s_1 G_1 + s_2 G_2 + s_3 H - cR$; $T_2 = s_4 G_1 + s_5 G_2 + s_6 H - s_7 R$; $T_3 = s_8 G - cE$; $\Pi_1 = e(P, P)^{s_0} e(P, U_1)^{-s_7} e(P, H)^{s_4} e(P_{pub}, H)^{s_1} e(P, P_0)^c e(P_{pub}, U_1)^{-c}$; $\Pi_2 = e(Q, U_2)^{-s_7} e(Q, H)^{s_5} e(Q_{pub}, H)^{s_2} e(Q, V_j)^c e(Q_{pub}, U_2)^{-c}$; $\Pi_3 = e(P, H)^{-s_1} \Theta^{s_8} \Lambda^{-c} e(P, U_1)^c$. We can see that the distribution of the simulation is the same as the distribution of the real transcript.

Lemma 4. *If the q-SDH assumption holds, then the scheme $\mathcal{IE}1$, whose group size is bounded by q, provide Coalition-Resistance, where the predicate $\mathcal{U}$ is defined as:*
$\mathcal{U}(\langle u, Q, Q_{pub}, P, P_0, P_{pub}, ...\rangle, V_j, \langle x_i, a_i, S_i, \Delta_i\rangle, W_{i,j}) = 1 \Leftrightarrow (e(a_i P + P_{pub}, S_i) = e(P, x_i P + P_0) \wedge e(a_i Q + Q_{pub}, W_{i,j}) = e(Q, V_j))$*, where V_j is the latest group accumulated value.*

Proof. We first prove that if there is a PPT adversary $\mathcal{A}$ with the capabililies as specified in the Coalition-Resistance definition, who can, with non-negligible probability, output a new membership secret key $(x^*, a^*, S^*, \Delta^*)$ (satisfying $e(a^* P + P_{pub}, S^*) = e(P, x^* P + P_0)$) not in the set of membership secret keys $\{(x_i, a_i, S_i, \Delta_i)\}_{i=1}^q$ generated during $\mathcal{A}$'s attack, then there is a PPT adversary $\mathcal{B}$ that can break the q-SDH assumption. (1)

Suppose a tuple *challenge* $= (R, zR, \ldots, z^q R)$ is given, where $z \in_R \mathbb{Z}_p^*$, we show that $\mathcal{B}$ can compute $(c, 1/(z+c)R)$, where $c \in \mathbb{Z}_p$ with non-negligible probability. We consider two cases.

Case 1: This is a trivial case, where $\mathcal{A}$ outputs $S^* \in \{S_1, ..., S_q\}$ with non-negligible probability. In this case, $\mathcal{B}$ chooses $s, u, x, x' \in_R \mathbb{Z}_p^*$ and $G, G_1, G_2, H, Q \in_R \mathbb{G}_1$, gives $\mathcal{A}$ the group public key $(u, Q, Q_{pub} = sQ, P = R, P_0 = zR, P_{pub} = xP, H, G, G_1, G_2, \Theta = e(G, G)^{x'})$ and the opening key x' and simulates a set of possible users. Then $\mathcal{B}$ can simulate all oracles $\mathcal{A}$ needs to access. Suppose a set of membership secret keys $\{(x_i, a_i, S_i, \Delta_i)\}_{i=1}^q$ is generated and $\mathcal{A}$ outputs a new $(x^*, a^*, S^*, \Delta^*)$ with non-negligible probability such that $S^* \in \{S_1, ..., S_q\}$. Suppose $S^* = S_j$, where $j \in \{1, ..., q\}$, then $\frac{1}{a^*+x}(x^* P + P_0) = \frac{1}{a_j+x}(x_j P + P_0)$, so $(a_j - a^*)P_0 = (a^* x_j - a_j x^* + x_j x - x^* x)P$. Therefore, z is computable by $\mathcal{B}$ from this, and so is $(c, 1/(z+c)R)$, for any $c \in \mathbb{Z}_p$.

Case 2: This is when the first case does not hold. That means $\mathcal{A}$ outputs $S^* \notin \{S_1, ..., S_q\}$ with non-negligible probability. Then $\mathcal{B}$ plays the following game:

1. Generate $\alpha, a_i, x_i \in_R \mathbb{Z}_p^*$, $i = 1, ..., q$, where a_is are different from one another, then choose $m \in_R \{1, ..., q\}$.

2. Let $x = z - a_m$, then the following P, P_{pub}, P_0 are computable by $\mathcal{B}$ from the tuple *challenge*.

$$P = \prod_{i=1, i \neq m}^{q} (z + a_i - a_m)R$$

$$P_{pub} = xP = (z - a_m) \prod_{i=1, i \neq m}^{q} (z + a_i - a_m)R$$

$$P_0 = \alpha \prod_{i=1}^{q} (z + a_i - a_m)R - x_m \prod_{i=1, i \neq m}^{q} (z + a_i - a_m)R$$

3. Generate $u, s, x' \in_R \mathbb{Z}_p^*$ and $G, G_1, G_2, H, Q \in_R \mathbb{G}_1$ and give $\mathcal{A}$ the group public key
$(u, Q, Q_{pub} = sQ, P, P_0, P_{pub}, H, G, G_1, G_2, \Theta = e(G, G)^{x'})$ and the opening key x' and simulates a set of possible users.
4. With the capabilities above, $\mathcal{B}$ can simulate oracles $\mathsf{CrptU}(\cdot, \cdot)$, $\mathsf{RReg}(\cdot)$, $\mathsf{USK}(\cdot)$, $\mathsf{RevokeU}(\cdot)$ and $\mathsf{Witness}(\cdot)$ $\mathcal{A}$ needs to access. For $\mathsf{AddU}(\cdot)$ or $\mathsf{SndToI}(\cdot, \cdot)$, $\mathcal{B}$ simulates the addition of a honest or corrupted user i as follows. As playing both sides of the Join, Iss protocol or being able to extract information from $\mathcal{A}$, $\mathcal{B}$ simulates the protocol as specified so that the prepared a_i, x_i above are computed in the protocol to be the corresponding parts of the user i's membership secret key. $\mathcal{B}$ can compute S_i as follows:
   - If $i = m$, then

$$S_m = \frac{1}{a_m + x}(x_m P + P_0) = \alpha \prod_{i=1, i \neq m}^{q} (z + a_i - a_m)R$$

   This is computable from the tuple *challenge*.
   - If $i \neq m$, then

$$\begin{aligned} S_i &= \frac{1}{a_i + x}(x_i P + P_0) \\ &= (x_i - x_m) \prod_{j=1, j \neq m, i}^{q} (z + a_j - a_m)R + \alpha \prod_{j=1, j \neq i}^{q} (z + a_j - a_m)R \end{aligned}$$

   This is computable from the tuple *challenge*.
5. Get the output $(x^*, a^*, S^*, \Delta^*)$ from $\mathcal{A}$, where

$$\begin{aligned} S^* &= \frac{1}{a^* + x}(x^* P + P_0) \\ &= \frac{1}{z + a^* - a_m}(\alpha z + x^* - x_m) \prod_{i=1, i \neq m}^{q} (z + a_i - a_m)R \end{aligned}$$

We can see that the case $\alpha z + x^* - x_m = \alpha(z + a^* - a_m)$ happens with negligible probability, as it results in $S^* = S_m$. So the case $\alpha z + x^* - x_m \neq \alpha(z+a^*-a_m)$ happens with non-negligible probability ϵ. Suppose the probability that $a^* \in \{a_1, ..., a_q\}$ is ε, then the probability that $a^* \notin \{a_1, ..., a_q\} \backslash \{a_m\}$ is $\epsilon - \frac{q-1}{q}\varepsilon$, which is also non-negligible if q is bound by a polynomial of l. And when $a^* \notin \{a_1, ..., a_q\} \backslash \{a_m\}$, $\frac{1}{z+a^*-a_m}R$ is computable from the tuple *challenge* and S^*. So $\mathcal{B}$ can compute $(c, \frac{1}{z+c}R)$, where $c = a^* - a_m$.

So, the statement (1) has been proved. Combined with the collision-resistant property of the dynamic accumulator $\mathcal{DA}1$ (Theorem 1), it results in the Coalition-Resistance of $\mathcal{IE}1$.

Proof of Theorem 6. Suppose there is a PPT adversary $\mathcal{A}$ that can break Anonymity property of $\mathcal{IE}1$, we show a construction of a PPT adversary $\mathcal{B}$ that can break IND-CPA property of El Gamal^{BP}. Suppose an El Gamal^{BP} public key (G, Θ) is given, $\mathcal{B}$ constructs an instance of $\mathcal{IE}1$ by generating the issuing key $ik = (s, x)$ and the group public key $gpk = (u, Q, Q_{pub}, P, P_0, P_{pub}, H, G, G_1, G_2, \Theta)$. The opening key *ok* is the private key of the El Gamal^{BP} public key, and unknown to $\mathcal{B}$. Let $\mathcal{B}$ play the role of the issuer, simulate the set of possible users and provides $\mathcal{A}$ with gpk, ik and access to the following simulated oracles:

- SndToI$(\cdot, \cdot)$, SndToU$(\cdot, \cdot)$, WReg$(\cdot, \cdot)$, USK$(\cdot)$ CrptU$(\cdot, \cdot)$, RevokeU$(\cdot)$ and Witness$(\cdot)$. With the above capabilities, $\mathcal{B}$ can easily simulate these oracles.
- Ch$(d, \cdot, \cdot)$. When receiving a query (i_0, i_1, m) from $\mathcal{A}$, $\mathcal{B}$ finds Δ_{i_d} and asks for an El Gamal^{BP} challenge encryption $cip = (E, \Lambda)$ of Δ_{i_d}. From that, $\mathcal{B}$ simulates $c, s_0, ...s_8$, U_1, U_2, R, T_1, T_2, T_3 and Π_1, Π_2, Π_3 as in the Zero-knowledge proof of Lemma 3. Then $\mathcal{B}$ returns to $\mathcal{A}$ the challenge transcript $(E, \Lambda, U_1, U_2, R, T_1, T_2, T_3, \Pi_1, \Pi_2, \Pi_3, c, s_0, ..., s_8)$.

At last, $\mathcal{B}$ outputs the bit returned by $\mathcal{A}$. As $\mathcal{A}$ can break Anonymity property, $\mathcal{B}$ outputs the correct bit with non-negligible probability.

Proof of Theorem 7. Suppose there is a PPT adversary $\mathcal{A}$ that can break Traceability property of $\mathcal{IE}1$, we show that there exists a PPT adversary $\mathcal{B}$ that can break Coalition-Resistance of $\mathcal{IE}1$. Suppose $\mathcal{A}$ can performs the IEID protocol with a honest verifier so that the opener can not trace the identity of the prover or the opener can find the identity but can not prove that to the Judge. As the IEID protocol has Soundness (Lemma 3), $\mathcal{B}$ can find $W_{i,j}$, a_i, S_i, x_i and t so that $E = tG$, $\Lambda = e(P, S_i)\Theta^t$, $e(a_iQ + Q_{pub}, W_{i,j}) = e(Q, V_j)$ and $e(a_iP + P_{pub}, S_i) = e(P, x_iP + P_0)$. So the opener, which is assumed to operate accurately, should find $\Delta_i = e(P, S_i)$ from the transcript. The issuer is assumed to be uncorrupted and no oracle accessible by the adversaries can write on *reg* table or overwrite $upk[j]$ of a group member j (CrptU does not apply to group members). So if Δ_i can not be found on *reg*, $\mathcal{B}$ has produced a new valid pair of membership secret key and witness.

Proof of Theorem 8. Suppose there is a PPT adversary $\mathcal{A}$ that can break Non-frameability property of $\mathcal{IE}1$ we show that there exists a PPT adversary $\mathcal{B}$ that can break Discrete Logarithm Assumption over $\mathbb{G}_1$. Suppose that $\mathcal{B}$ is given a challenge $(P, P^* = zP)$, where $P \leftarrow \mathbb{G}_1^*$ and $z \leftarrow \mathbb{Z}_p^*$, and $\mathcal{B}$ needs to compute z. $\mathcal{B}$ constructs an instance of $\mathcal{IE}1$ by generating $u, s, x, x', d \in_R \mathbb{Z}_p^*$ and $G, G_1, G_2, H, Q \in_R \mathbb{G}_1$ and give $\mathcal{A}$ the group signature public key $(u, Q, Q_{pub}, P, P_0 = dP, P_{pub} = xP, H, G, G_1, G_2, \Theta_a = e(G,G)^{x'})$, the issuing key $ik = (x, s)$ and the opening key x'. $\mathcal{B}$ simulates a set of possible users $\{1, ..., q\}$, where q is the upper bound of the group size, chooses $i_0 \in_R \{1, ..., q\}$ and provides $\mathcal{A}$ access to the following simulated oracles:

- $\mathsf{SndToU}(i, M_{in})$. If $i \neq i_0$, $\mathcal{B}$ just plays as a honest user i and executes Iss as specified in M_{in}. If $i = i_0$, $\mathcal{B}$ simulates the Join, Iss protocol so that $P_{i_0} = P^*$ (by controlling the random oracle, $\mathcal{B}$ can simulate the proof of knowledge in the protocol). Suppose the membership secret key obtained for i_0 is $(x_{i_0}, a_{i_0}, S_{i_0}, \Delta_{i_0})$, where $x_{i_0} = z$ is unknown to $\mathcal{B}$.
- $\mathsf{WReg}(\cdot,\cdot)$, $\mathsf{GSig}(\cdot,\cdot)$, $\mathsf{USK}(\cdot)$, $\mathsf{CrptU}(\cdot,\cdot)$, $\mathsf{RevokeU}(\cdot)$ and $\mathsf{Witness}(\cdot)$. With the capabilities above, $\mathcal{B}$ can simulate all these oracles, except the case when he gets a query $\mathsf{USK}(i_0)$. In this case, $\mathcal{B}$ fails.

If $\mathcal{A}$ succeeds with probability ϵ, then the probability that he can impersonate i_0 in IEID is at least ϵ/q, as $i_0 \in_R \{1, ..., q\}$. As the IEID protocol has $\mathsf{Soundness}$ (Lemma 3), $\mathcal{B}$ can find a_{i_1}, S_{i_1}, x_{i_1} and t so that $E_a = tG$, $\Lambda_a = e(P, S_{i_1})\Theta_a^t$ and $e(a_{i_1}P + P_{pub}, S_{i_1}) = e(P, x_{i_1}P + P_0)$. The digital signature scheme (K_S, $Sign$, Ver) is UNF-CMA, therefore $e(P, S_{i_0}) = e(P, S_{i_1})$ or $S_{i_0} = S_{i_1}$. So $\frac{1}{a_{i_0}+x}(x_{i_0}P + dP) = \frac{1}{a_{i_1}+x}(x_{i_1}P + dP)$, from that, $\mathcal{B}$ can compute $z = x_{i_0}$.