Front. Comput. Sci., 2016, 10(4): 741–754
DOI 10.1007/s11704-015-5138-2

RESEARCH ARTICLE

Identity-based aggregate signcryption in the standard model from multilinear maps

Hao WANG[1,2], Zhen LIU[3], Zhe LIU (✉)[4], Duncan S. WONG[3]

1 School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China
2 Shandong Provincial Key Laboratory for Novel Distributed Computer Software Technology, Jinan 250014, China
3 Security and Data Sciences, Hong Kong Applied Science and Technology Research Institute (ASTRI), Hong Kong, China
4 Laboratory of Algorithmics, Cryptology and Security (LACS), University of Luxembourg, Luxembourg L-1359, Luxembourg

Abstract Signcryption is a public key cryptographic method that achieves unforgeability and confidentiality simultaneously with significantly smaller overhead than that required by "digital signature followed by public key encryption". It does this by signing and encrypting a message in a single step. An aggregate signcryption scheme allows individual signcryption ciphertexts intended for the same recipient to be aggregated into a single (shorter) combined ciphertext without losing any of the security guarantees. We present an aggregate signcryption scheme in the identity-based setting using multilinear maps, and provide a proof of security in the standard model. To the best of our knowledge, our new scheme is the first aggregate signcryption scheme that is secure in the standard model.

Keywords identity-based aggregate signcryption, multilinear maps, standard model, GGH framework

1 Introduction

Signcryption, first proposed by Zheng [1], is a cryptographic primitive that performs signature and encryption simultaneously in one logical steps at lower computational costs and communication overheads than those required by the traditional sign-then-encrypt approach. Due to its advantages, there have been many signcryption schemes proposed after Zheng's publication. Zheng's original schemes were only proven secure by Baek et al. [2] who described a formal security model in a multiuser setting. Many public key signcryption schemes have been proposed after Ref. [1], such as Refs. [3–5].

Identity-based cryptography (IBC) was introduced by Shamir in 1984 [6]. The unique property of identity-based cryptography is that a user's public key can be any binary string, such as an email address, IP address, that can identify the user. This removes the need for senders to look up the recipient's public key before sending out an encrypted message. Identity-based cryptography is supposed to provide a more convenient alternative to conventional public key infrastructure (PKI). Identity based signcryption (IBSC) with formal security proof was introduced by Malone-Lee [7]. But Malone-Lee's scheme was not secure and its weakness was pointed out by Libert and Quisquater in Ref. [8]. Libert and Quisquater also proposed three different types of IBSC schemes which satisfy either public verifiability or forward security. Then, Chow et al. [9] designed an IBSC scheme that provides both public verifiability and forward security. Boyen [10] presented an IBSC scheme that provides not only public verifiability and forward security but also ciphertext unlinkability and anonymity. Chen and Malone-Lee [11] improved Boyen's scheme in efficiency and Barreto et al. [12] constructed the most efficient IBSC scheme to date.

Received April 8, 2015; accepted September 5, 2015

E-mail: zhe.liu@uni.lu

In many application scenarios, one may have to process many signcryption quickly and simultaneously. In order to reduce the compute cost and overhead of transmission, Selvi et al. [13] introduced the concept of aggregate signcryption in the identity-based setting. An aggregate signcryption scheme allows individual signcryption ciphertexts intended for the same recipient to be aggregated into a single (shorter) combined ciphertext without losing any of the security guarantees that would be present if the original signcryption ciphertexts were transmitted individually.

In other words, the aggregate ciphertext still provides a mechanism for message transmission with data confidentiality, data integrity and origin authentication, but with significantly reduced overhead. We stress that the aggregation algorithm is entirely public — it can be performed by any entity given a number of signcryption ciphertexts and the corresponding public keys. This allows for signcryption ciphertexts intended for a single receiver to be combined within the network. If a network node receives two signcryption ciphertext which need to be routed to the same receiver, then the node can run the aggregation algorithm and forward the aggregate signcryption ciphertext at a reduced bandwidth cost.

After Selvi et al. [13] proposed the first identity-based aggregate signcryption scheme (IBASC) along with a formal security model and a formal security proof, there are many researches focusing on this topic. Ren et al. [14], Qi et al. [15], and Kar [16] proposed aggregate signcryption schemes in the identity-based setting. Dent [17] proposed aggregate signcryption scheme in public-key setting, and Eslami et al. [18]. Lu et al. [19] proposed aggregate signcryption schemes in the certificateless setting. However, none of these aggregate signcryption schemes was proven secure without random oracle.

Many researchers have expressed doubts about the wisdom of relying on the random-oracle model (ROM). In particular, Canetti, Goldreich and Halevi [20] showed that there are signature and encryption schemes which are secure in the ROM, but insecure for any instantiation of the random oracle. Furthermore, RSA-FDH (Full Domain Hash) and many other schemes provably secure in the ROM require a cryptographic hash function whose output size does not match any of the standard hash functions. Therefore, constructing aggregate signcryption schemes that are secure in the standard model (without random oracle) is a meaningful work.

In this paper, we construct an identity-based aggregate signcryption scheme that is secure in the standard model. To realize this, we use the new method for implementing the Full Domain Hash proposed by Hohenberger et al. [21]. We present our results in a generic multilinear map [22] setting and then show how they can be translated to the GGH [23] graded algebras analogue of multilinear maps. To the best of our knowledge, this is the first aggregate signcryption scheme that is secure in the standard model.

We introduce the leveled multilinear maps, the GGH graded encoding, and the complexity assumption in Section 2, and review the definitions and security model for IBASC in Section 3 and 4. Then, we present our IBASC in generic multilinear map setting in Section 5, and make it translated to the GGH framework in Section 6. In Section 7, we give the conclusion and open problems.

2 Leveled multilinear maps and the GGH graded encoding

2.1 Generic leveled multilinear maps

We give a description of generic leveled multilinear maps. More details of the GGH graded algebras analogue of mulitlinear maps are included in Appendix, and for further details, please refer to Ref. [23].

For generic leveled multilinear maps, we assume the existence of a group generator $\mathcal{G}$, which takes as input a security parameter 1^λ and a positive integer k to indicate the number of allowed pairing operations. $\mathcal{G}(1^\lambda, k)$ outputs a sequence of groups $\vec{G} = (\mathbb{G}_1, \ldots, \mathbb{G}_k)$ each of large prime order $p > 2^\lambda$. In addition, we let g_i be a canonical generator of $\mathbb{G}_i$ (and it is known from the group's description). We let $g = g_1$.

We assume the existence of a set of bilinear maps $\{e_{i,j} : \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_{i+j} | i, j \geqslant 1; i + j \leqslant k\}$. The map $e_{i,j}$ satisfies the following relation:

$$e_{i,j}(g_i^a, g_j^b) = g_{i+j}^{ab} : \forall a, b \in \mathbb{Z}_p.$$

We observe that one consequence of this is that $e_{i,j}(g_i, g_j) = g_{i+j}$ for each valid i, j.

When the context is obvious, we will sometimes abuse notation and drop the subscripts i, j. For example, we may simply write $e(g_i^a, g_j^b) = g_{i+j}^{ab}$.

2.2 Algorithmic components of GGH encodings

Garg, Gentry and Halevi (GGH) [23] defined an "approximate" version of a multilinear group family, which they call a *graded encoding system*. As a starting point, they view g_i^α in a multilinear group family as simply an encoding of α at "level-i". This encoding permits basic functionalities, such as equality testing (it is easy to check that two level-i encodings encode the same exponent), additive homomorphism

(via the group operation in $\mathbb{G}_i$), and bounded multiplicative homomorphism (via the multilinear map e). They retain the notion of a somewhat homomorphic encoding with equality testing, but they use probabilistic encodings, and replace the multilinear group family with "less structured" sets of encodings related to lattices.

Abstractly, their k-graded encoding system for a ring R includes a system of sets $\mathcal{S} = \{S_i^{(\alpha)} \subset \{0,1\}^* : i \in [0,k], \alpha \in R\}$ such that, for every fixed $i \in [0,k]$, the sets $\{S_i^{(\alpha)} : \alpha \in R\}$ are disjoint (and thus form a partition of $S_i = \bigcup_\alpha S_i^{(\alpha)}$). The set $S_i^{(\alpha)}$ consists of the "level-i encodings of α". Moreover, the system comes equipped with efficient procedures, as follows:

- **Instance generation** The randomized $\mathsf{InstGen}(1^\lambda, 1^k)$ takes as input the security parameter λ and integer k. The procedure outputs $(\mathsf{params}, \mathbf{p}_{zt})$, where params is a description of a k-graded encoding system as above, and $\mathbf{p}_{zt}$ is a level-k "zero-test parameter".
- **Ring sampler** The randomized $\mathsf{samp}(\mathsf{params})$ outputs a "level-zero encoding" $a \in S_0$, such that the induced distribution on α such that $a \in S_0^{(\alpha)}$ is statistically uniform.
- **Encoding** The (possibly randomized) $\mathsf{enc}(\mathsf{params}, i, a)$ takes $i \in [k]$ and a level-zero encoding $a \in S_0^{(\alpha)}$ for some $\alpha \in R$, and outputs a level-i encoding $u \in S_i^{(\alpha)}$ for the same α.
- **Re-randomization** The randomized $\mathsf{reRand}(\mathsf{params}, i, u)$ re-randomizes encodings to the same level, as long as the initial encoding is under a given noise bound. Specifically, for a level $i \in [k]$ and encoding $u \in S_i^{(\alpha)}$, it outputs another encoding $u' \in S_i^{(\alpha)}$. Moreover for any two encodings $u_1, u_2 \in S_i^{(\alpha)}$ whose noise bound is at most some b, the output distributions of $\mathsf{reRand}(\mathsf{params}, i, u_1)$ and $\mathsf{reRand}(\mathsf{params}, i, u_2)$ are statistically the same.
- **Addition and negation** Given params and two encodings at the same level, $u_1 \in S_i^{(\alpha_1)}$ and $u_2 \in S_i^{(\alpha_2)}$, we have $\mathsf{add}(\mathsf{params}, u_1, u_2) \in S_i^{(\alpha_1+\alpha_2)}$, and $\mathsf{neg}(\mathsf{params}, u_1) \in S_i^{(-\alpha_1)}$, subject to bounds on the noise.
- **Multiplication** For $u_1 \in S_{i_1}^{(\alpha_1)}$, $u_2 \in S_{i_2}^{(\alpha_2)}$, we have $\mathsf{mult}(\mathsf{params}, u_1, u_2) \in S_{i_1+i_2}^{(\alpha_1 \cdot \alpha_2)}$.
- **Zero-test** The procedure $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u)$ outputs 1 if $u \in S_k^{(0)}$ and 0 otherwise. Note that in conjunction with the procedure for subtracting encodings, this gives us an equality test.
- **Extraction** This procedure extracts a "canonical" and "random" representation of ring elements from their level-k encoding. Namely $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u)$ outputs (say) $K \in \{0,1\}^\lambda$, such that:
  1) With overwhelming probability over the choice of $\alpha \in R$, for any two $u_1, u_2 \in S_k^{(\alpha)}$, $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_1) = \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_2)$,
  2) The distribution $\{\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u): \alpha \in R, u \in S_k^{(\alpha)}\}$ is statistically uniform over $\{0,1\}^\lambda$.

The realization method of GGH's graded encoding system is included in Appendix.

2.3 Complexity assumption

Assumption 1 (multilinear computational Diffie-Hellman assumption) [21] The k-multilinear computational Diffie-Hellman (k-MCDH) assumption states the following: A challenger runs $\mathcal{G}(1^\lambda, k)$ to generate groups and generators of order p. Then it picks random $c_1, \ldots, c_k \in \mathbb{Z}_p$. The assumption then states that given $g = g_1, g^{c_1}, \ldots, g^{c_k}$ it is hard for any poly-time algorithm to compute $g_{k-1}^{\prod_{j\in[1,k]} c_j}$ with better than negligible advantage (in security parameter λ).

Assumption 2 (GGH analogue of k-MCDH assumption) [21] The GGH k-multilinear computational Diffie-Hellman (GGH k-MCDH) assumption states the following: A challenger runs $\mathsf{InstGen}(1^\lambda, 1^k)$ to obtain $(\mathsf{params}, \boldsymbol{p}_{zt})$. Note that params includes a level 1 encoding of 1, which we denote as g. Then it picks random $c_1, \ldots, c_k$ each equal to the result of a fresh call to $\mathsf{samp}()$.

The assumption then states that given params, $\boldsymbol{p}_{zt}$, $\mathsf{enc}(1, c_1)$, …, $\mathsf{enc}(1, c_k)$, it is hard for any poly-time algorithm to compute an integer $t \in [1, 2^\lambda]$ and an encoding z such that

$$\mathsf{isZero}(\boldsymbol{p}_{zt}, \mathsf{mult}(g, z) - \mathsf{enc}(k, t \cdot \prod_{j\in[1,k]} c_j))$$

outputs true.

Assumption 3 (multilinear decisional Diffie-Hellman assumption) [24] The k-multilinear decisional Diffie-Hellman (k-MDDH) assumption states the following: A challenger runs $\mathcal{G}(1^\lambda, k)$ to generate groups and generators of order p. Then it picks random $c_1, \ldots, c_{k+1} \in \mathbb{Z}_p$. The assumption then states that given $g = g_1, g^{c_1}, \ldots, g^{c_{k+1}}$ it is hard for any poly-time algorithm to distinguish $g_k^{\prod_{j\in[1,k+1]} c_j}$ from a uniform $\mathbb{G}_k$-element with better than negligible advantage (in security parameter λ).

Assumption 4 (GGH analogue of k-MDDH assumption) [24] The GGH k-multilinear decisional Diffie-Hellman (GGH k-MDDH) assumption states the following: A challenger runs $\mathsf{InstGen}(1^\lambda, 1^k)$ to obtain $(\mathsf{params}, \boldsymbol{p}_{zt})$. Note that params includes a level 1 encoding of 1, which we denote as g. Then it picks random $c_1, \ldots, c_{k+1}$ and each equals to the result of a fresh call to $\mathsf{samp}()$.

The assumption then states that given params, $\boldsymbol{p}_{zt}, \mathsf{enc}(1, c_1), \ldots, \mathsf{enc}(1, c_{k+1})$ and a level-k encoding T, it is hard for any poly-time algorithm to decide the output of $\mathsf{isZero}(\boldsymbol{p}_{zt}, \mathsf{reRand}(T) - \mathsf{reRand}(\mathsf{enc}(\mathsf{params}, \mathrm{k}, \prod_{j\in[1,k+1]} c_j)))$ is 1 or 0 with better than negligible advantage (in security parameter λ).

3 Definitions for identity-based aggregate signcryption schemes

We now give our definitions for identity-based aggregate signcryption schemes. In our setting, each signcryption is associated with a *multiset* $\mathcal{S}$ over identities. A set $\mathcal{S}$ is of the form $\{ID_1, \ldots, ID_{|\mathcal{S}|}\}$. Since $\mathcal{S}$ is a multiset it is possible to have $ID_i = ID_j$ for $i \neq j$. All signcryptions, including those that come out of the signcrypt algorithm, are considered to be aggregate signcryptions. The aggregation algorithm is general in that it can take any two aggregate signcryptions and combine them into a new aggregate signcryption.

Our definition allows for an initial trusted setup that will generate a set of common public parameters PP. This will define message space and identity space. The authority also produces a master secret key used later to run the key generation algorithm.

We emphasize a few features of our setting. First, aggregation is very general in that it allows for the combination of any two aggregate signcryptions into a single one. The aggregation operation does not require any secret keys. The multiset structure allows one to combine two aggregate signcryptions which both are signcrypted from the same sender.

An identity-based aggregate signcryption scheme (IBASC) consists of the following five probabilistic polynomial time (PPT) algorithms:

- **Setup**(1^λ) The trusted setup algorithm takes the security parameter λ as input, and outputs a common set of public parameters PP and master secret key MSK. The description of message space $\mathcal{M}$ and identity space $\mathcal{ID}$ are included in PP, where $\mathcal{ID} = \mathcal{ID}_S \cup \mathcal{ID}_R$, $\mathcal{ID}_S$ is the sender identity space, $\mathcal{ID}_R$ is the receiver identity space, and $\mathcal{ID}_S \cap \mathcal{ID}_R = \emptyset$.
- **KeyGen**$(MSK, ID \in \mathcal{ID})$ The key generation algorithm is run by the authority. It takes the master secret key MSK and the identity information $ID \in \mathcal{ID}$ of user $\mathcal{U}_{ID}$ as input and outputs a private key SK_{ID}. The authority sends SK_{ID} to user $\mathcal{U}_{ID}$ through a secure channel.
- **Signcrypt**$(PP, M_i \in \mathcal{M}, ID_R \in \mathcal{ID}_R, ID_{S_i} \in \mathcal{ID}_S, SK_{ID_{S_i}})$ For generating the signcryption of a message M_i from user $\mathcal{U}_{ID_{S_i}}$ to user $\mathcal{U}_{ID_R}$, the sender $\mathcal{U}_{ID_{S_i}}$ provides the system parameters PP, the message M_i, the identity information $ID_R \in \mathcal{ID}_R$ of receiver $\mathcal{U}_{ID_R}$, the identity information $ID_{S_i} \in \mathcal{ID}_S$ of its own, and the private key $SK_{ID_{S_i}}$ for ID_{S_i} as input to this algorithm. The signcryption algorithm outputs the valid signcryption σ_i for the message M_i from user $\mathcal{U}_{ID_{S_i}}$ to user $\mathcal{U}_{ID_R}$ and an identities set of senders $\mathcal{S} = \{ID_{S_i}\}$. We emphasize that a single signcryption that is output by this algorithm is considered to also be an aggregate signcryption.
- **Aggregate**$(PP, \mathcal{S}_x, \mathcal{S}_y, \sigma_x, \sigma_y)$ The aggregation algorithm takes as input two multisets $\mathcal{S}_x$ and $\mathcal{S}_y$ and purported signcryption σ_x and σ_y. The elements of $\mathcal{S}_x = \{ID_{S_{x,1}}, \ldots, ID_{S_{x,|S_x|}}\}$ consist of the identities corresponding to σ_x and the elements of $\mathcal{S}_y = \{ID_{S_{y,1}}, \ldots, ID_{S_{y,|S_y|}}\}$ consist of the identities corresponding to σ_y. The process produces a signcryption σ_{Agg} on the multiset $\mathcal{S}_{Agg} = \mathcal{S}_x \bigcup \mathcal{S}_y$, where $\bigcup$ is a multiset union.
- **Unsigncrypt**$(PP, SK_{ID_R}, \mathcal{S}, \sigma)$ The unsigncryption algorithm takes as input the public parameters, the private key SK_{ID_R} for the receiver with identity ID_R, a multiset $\mathcal{S} = \{ID_{S_1}, \ldots, ID_{S_{|\mathcal{S}|}}\}$ and an aggregate signcryption σ corresponding to $\mathcal{S}$. It outputs a set of messages $\{M_1, \ldots, M_{|\mathcal{S}|}\}$ or a false symbol $\perp$ to indicate whether verification succeeded, where M_i is the message sent from user $\mathcal{U}_{ID_{S_i}}$ with identity $ID_{S_i} \in \mathcal{S}$ to user $\mathcal{U}_{ID_R}$ with identity ID_R, for $i \in [1, |\mathcal{S}|]$.

Correctness The correctness property states that all valid aggregate signcryption will pass the unsigncryption algorithm and output corresponding messages, where a valid aggregate is defined recursively as an aggregate signcryption derived by an application of the aggregation algorithm on two valid inputs or the signcryption algorithm. More formally, for all PP, $MSK \in$ **Setup**(1^λ), all $ID_{S_1}, \ldots, ID_{S_k} \in \mathcal{ID}_S$ $(k \geqslant 1)$, all $SK_{ID_{S_i}} \in$ **KeyGen**(MSK, ID_{S_i}), $\{M_1, \ldots, M_{|\mathcal{S}|}\} \leftarrow$ **Unsigncrypt**$(PP, SK_{ID_R}, \mathcal{S}, \sigma)$, if σ is a valid aggregate signcryption for multiset $\mathcal{S}$ under PP. We say that an aggregate signcryp-

tion σ is valid for multiset $\mathcal{S}$ if: (1) $\mathcal{S} = \{ID_{S_i}\}$ for some $i \in [1,k]$, and $\sigma \in$ **Signcrypt**$(PP, M_i \in \mathcal{M}, ID_R \in \mathcal{ID}_R, ID_{S_i} \in \mathcal{ID}_S, SK_{ID_{S_i}})$; or (2) there exist multisets $\widetilde{\mathcal{S}}, \widehat{\mathcal{S}}$ where $\mathcal{S} = \widetilde{\mathcal{S}} \cup \widehat{\mathcal{S}}$ and valid aggregate signcryption $\widetilde{\sigma}, \widehat{\sigma}$ on them respectively such that $\sigma \in$ **Aggregate**$(PP, \widetilde{\mathcal{S}}, \widehat{\mathcal{S}}, \widetilde{\sigma}, \widehat{\sigma})$.

4 Security model for identity based aggregate signcryption schemes

Security of signcryption consists of two different mechanism: one ensuring authenticity, and the other privacy.

4.1 Unforgeability

Informally, in the unforgeability game, it should be computationally infeasible for any adversary to produce a forgery implicating an honest identity, even when the adversary can control all other identities involved in the aggregate and can mount a chosen message attack on the honest identity. This is defined using a game between a challenger and an adversary $\mathcal{A}$ with respect to scheme Π = (**Setup**, **KeyGen**, **Signcrypt**, **Aggregate**, **Unsigncrypt**).

IBASC-Unforg $(\Pi, \mathcal{A}, \lambda)$:

- **Setup** The challenger runs **Setup**(1^λ) to obtain PP. It sends PP to $\mathcal{A}$.
- **Queries** Proceeding adaptively, $\mathcal{A}$ can make three types of requests:
  1) Private key query On input an identity $ID \in \mathcal{ID}$, the challenger runs **KeyGen**(MSK, ID) to obtain SK_{ID}. Then, it sends SK_{ID} to the adversary $\mathcal{A}$.
  2) Signcrypt query On input an identity $ID_S \in \mathcal{ID}_S$, an identity $ID_R \in \mathcal{ID}_R$, and a message $M \in \mathcal{M}$, the challenger runs **KeyGen**(MSK, ID_S) to obtain SK_{ID_S}, and returns the signcryption resulting from **Signcrypt**$(PP, M, ID_R, ID_S, SK_{ID_S})$ to $\mathcal{A}$.
  3) Unsigncrypt query On input a multiset $\mathcal{S} = \{ID_{S_1}, \ldots, ID_{S_{|\mathcal{S}|}}\} \in \mathcal{ID}_S^{|\mathcal{S}|}$, an identity $ID_R \in \mathcal{ID}_R$, and an aggregate signcryption σ (from $\mathcal{S}$ to ID_R), the challenger obtains the private key SK_{ID_R} for ID_R from **KeyGen**(MSK, ID_R) and returns corresponding messages $\{M_1, \ldots, M_{|\mathcal{S}|}\}$ or a false symbol $\perp$ to indicate whether verification succeeded, where M_i is the message sent from user $\mathcal{U}_{ID_{S_i}}$ with identity $ID_{S_i} \in \mathcal{S}$ to user $\mathcal{U}_{ID_R}$ with identity ID_R, for $i \in [1, |\mathcal{S}|]$.
- **Response** Finally, $\mathcal{A}$ outputs a multiset $\mathcal{S}^* \in \mathcal{ID}_S^{|\mathcal{S}^*|}$ of identities, a receiver identity $ID_R^* \in \mathcal{ID}_R$ and a purported aggregate signcryption σ^*.

We say the adversary "wins" or that the output of this experiment is 1 if: (1) $\{M_1^*, \ldots, M_{|\mathcal{S}^*|}^*\} \leftarrow$ **Unsigncrypt**$(PP, SK_{ID_R^*}, \mathcal{S}^*, \sigma^*)$, (2) there exists an identity $ID_{S_i}^* \in \mathcal{S}^*$ such that $ID_{S_i}^*$ was not queried for a private key query by the adversary, and (3) there exists a message M_i^* corresponding to $ID_{S_i}^*$ such that $(M_i^*, ID_{S_i}^*)$ was not queried for a signcrypt query by the adversary. Otherwise, the output is 0. Define **IBASC-Forg**$_{\mathcal{A}}$ as the probability that **IBASC-Unforg**$(\Pi, \mathcal{A}, \lambda) = 1$, where the probability is over the coin tosses of the **Setup**, **KeyGen**, and **Signcrypt** algorithms and of $\mathcal{A}$.

Definition 1 (adaptive unforgeability) An ID-based aggregate signcryption scheme Π is existentially unforgeable with respect to adaptive chosen-message attacks if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the function IBASC-Forg$_{\mathcal{A}}$ is negligible in λ.

Selective security: We consider a selective variant to **IBASC-Unforg** (selective in both the identity and the message) where there is an **Init** phase before the **Setup** phase, wherein $\mathcal{A}$ gives to the challenger a forgery identity/ message pair $(ID_S^* \in \mathcal{ID}_S, M^* \in \mathcal{M})$. Note that M^* is the message sent form ID_S^*. The adversary only "wins" causing the experiment output to be 1 if: (1) $\{M_1^*, \ldots, M_{|\mathcal{S}^*|}^*\} \leftarrow$ **Unsigncrypt**$(PP, SK_{ID_R^*}, \mathcal{S}^*, \sigma^*)$, (2) ID_S^* was not queried for a private key query by the adversary, and (3) $M^* \in \{M_1^*, \ldots, M_{|\mathcal{S}^*|}^*\}$ is the message corresponding to $ID_S^* \in \mathcal{S}^*$, and (M^*, ID_S^*) was not queried for a signcrypt query by the adversary.

4.2 Confidentiality

Similar to unforgeability game, it should be infeasible for any adversary to distinguish the challenge signcryption ciphertexts, even when the adversary can mount any adaptive chosen identity and chosen ciphertext attacks. This is defined using a game between a challenger and an adversary $\mathcal{A}$ with respect to scheme Π = (**Setup**, **KeyGen**, **Signcrypt**, **Aggregate**, **Unsigncrypt**).

IBASC-IND-ID-CCA$(\Pi, \mathcal{A}, \lambda)$:

- **Setup** The challenger runs **Setup**(1^λ) to obtain PP. It sends PP to $\mathcal{A}$.
- **Phase 1** Proceeding adaptively, $\mathcal{A}$ can make three types of requests:
  1) Private key query: On input an identity $ID \in \mathcal{ID}$,

the challenger runs **KeyGen**(MSK, ID) to obtain SK_{ID}. Then, it sends SK_{ID} to the adversary $\mathcal{A}$.

2) Signcrypt query: On input an identity $ID_S \in \mathcal{ID}_S$, an identity $ID_R \in \mathcal{ID}_R$, and a message $M \in \mathcal{M}$, the challenger runs **KeyGen**(MSK, ID_S) to obtain SK_{ID_S}, and returns the signcryption resulting from **Signcrypt**(PP, M, ID_R, ID_S, SK_{ID_S}) to $\mathcal{A}$.

3) Unsigncrypt query: On input a multiset $\mathcal{S} = \{ID_{S_1}, \ldots, ID_{S_{|\mathcal{S}|}}\} \in \mathcal{ID}_S^{|\mathcal{S}|}$, an identity $ID_R \in \mathcal{ID}_R$, and an aggregate signcryption σ (from $\mathcal{S}$ to ID_R), the challenger obtains the private key SK_{ID_R} for ID_R from **KeyGen**(MSK, ID_R) and returns corresponding messages $\{M_1, \ldots, M_{|\mathcal{S}|}\}$ or a false symbol $\perp$ to indicate whether verification succeeds, where M_i is the message sent from user $\mathcal{U}_{ID_{S_i}}$ with identity $ID_{S_i} \in \mathcal{S}$ to user $\mathcal{U}_{ID_R}$ with identity ID_R, for $i \in [1, |\mathcal{S}|]$.

- **Challenge** $\mathcal{A}$ chooses a multiset of identities $\mathcal{S}^* = \{ID^*_{S_1}, \ldots, ID^*_{S_{|\mathcal{S}^*|}}\} \in \mathcal{ID}_S^{|\mathcal{S}^*|}$, a receiver identity $ID^*_R \in \mathcal{ID}_R$, and two multisets of messages $M^*_1 = \{M^*_{1,1}, \ldots, M^*_{1,|\mathcal{S}^*|}\}$ and $M^*_2 = \{M^*_{2,1}, \ldots, M^*_{2,|\mathcal{S}^*|}\}$, with the restriction that $|M^*_{1,i}| = |M^*_{2,i}|$, for $\forall i \in [1, |\mathcal{S}^*|]$. The Challenger generates the private keys for all identities in $\mathcal{S}^*$. Then, it chooses a random bit $\beta \in \{0, 1\}$, and computes the signcryptions from **Signcrypt**(PP, $M^*_{\beta,i}$, ID^*_R, $ID^*_{S_i}$, $SK_{ID^*_{S_i}}$) for $\forall i \in [1, |\mathcal{S}^*|]$. Finally, the challenger aggregates these signcryptions using the aggregation algorithm, and returns the result $\{\sigma^*, \mathcal{S}^*\}$ to $\mathcal{A}$. The adversary's choice of ID^*_R is restricted to the identities those are not queried for any private key queries in Phase 1.
- **Phase 2** Phase 1 is repeated with the restriction that the adversary cannot make a private key query on the identity ID^*_R, and cannot unsigncrypt $\{\sigma^*, S^*\}$.
- **Guess** $\mathcal{A}$ outputs a bit b'.

$\mathcal{A}$ wins the game if $b' = b$. The advantage of $\mathcal{A}$ is given by $\mathbf{ADV}_{\mathcal{A}} = |Pr[b' = b] - \frac{1}{2}|$.

Definition 2 (IND-ID-CCA) An ID-based aggregate signcryption scheme Π is indistinguishable with respect to adaptive chosen-identity and chosen-ciphertext attacks if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage $\mathbf{ADV}_{\mathcal{A}}$ is negligible in λ.

4.2.1 Chosen-plaintext attack

We consider a restricted variant to **IBASC-IND-ID-CCA** where the adversary $\mathcal{A}$ is not allowed to make **Unsigncrypt query** in Phase 1 and Phase 2. We call this **IBASC-IND-ID-CPA** game.

Definition 3 (IND-ID-CPA) An ID-based aggregate signcryption scheme Π is indistinguishable with respect to adaptive chosen-identity and chosen-plaintext attacks if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage $\mathbf{ADV}_{\mathcal{A}}$ is negligible in λ in **IBASC-IND-ID-CPA** game.

4.2.2 Selective security

We consider a selective variant to **IBASC-IND-ID-CPA** (selective in receiver identity) where there is an **Init** phase before the **Setup** phase, wherein $\mathcal{A}$ gives to challenger the receiver identity $ID^*_R \in \mathcal{ID}_R$. We call this **IBASC-IND-sID-CPA** game.

Definition 4 (IND-sID-CPA) An ID-based aggregate signcryption scheme Π is indistinguishable with respect to selective chosen-identity and chosen-plaintext attacks if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage $\mathbf{ADV}_{\mathcal{A}}$ is negligible in λ in **IBASC-IND-sID-CPA** game.

5 Identity-based aggregate signcryption construction

5.1 Generic multlinear construction

The idea of this construction is inspired by the related work [21] and [25], the former proposed an aggregate signature and the latter proposed a key-encapsulation mechanism. Both of them are constructed in the identity-based setting and using multilinear maps.

• **Setup**(1^λ, l, n) The trusted setup algorithm is run by PKG, the master authority of the ID-based system. It takes as input the security parameter as well the bit-length l of messages and bit-length n of identities. The message space is defined as $\{0, 1\}^l$, and the identity space is defined as $\{0, 1\}^n$, where the sender identity space is defined as $\{0\} \times \{0, 1\}^{n-1}$, and the receiver identity space is defined as $\{1\} \times \{0, 1\}^{n-1}$. It first runs $\mathcal{G}(1^\lambda, k = l + n)$ and outputs a sequence of groups $\vec{\mathbb{G}} = (\mathbb{G}_1, \ldots, \mathbb{G}_k)$ of prime order p, with canonical generators $g_1, \ldots, g_k$, where we let $g = g_1$.

Next, it chooses random exponents $(a_{1,0}, a_{1,1}), \ldots, (a_{l,0}, a_{l,1}) \in \mathbb{Z}_p^2$ and sets $A_{i,\alpha} = g^{a_{i,\alpha}}$ for $i \in [1, l]$ and $\alpha \in \{0, 1\}$. It also chooses random exponents $(b_{1,0}, b_{1,1}), \ldots, (b_{n,0}, b_{n,1}) \in \mathbb{Z}_p^2$ and sets $B_{j,\beta} = g^{b_{j,\beta}}$ for $j \in [1, n]$ and $\beta \in \{0, 1\}$.

These will be used to define two functions $\bar{H}(ID, M)$:

$\{0,1\}^n \times \{0,1\}^l \to \mathbb{G}_k$ and $\widetilde{H}(ID) : \{0,1\}^n \to \mathbb{G}_n$. Let $m_1, \ldots, m_l$ be the bits of message M and $id_1, \ldots, id_n$ as the bits of ID. It is computed iteratively as

$$\bar{H}_1(ID, M) = \widetilde{H}_1(ID) = B_{1,id_1}.$$

For $i \in [2, n]$,

$$\bar{H}_i(ID, M) = e(\bar{H}_{i-1}(ID, M), B_{i,id_i}),$$

$$\widetilde{H}_i(ID) = e(\widetilde{H}_{i-1}(ID), B_{i,id_i}).$$

For $i \in [n+1, n+l=k]$,

$$\bar{H}_i(ID, M) = e(\bar{H}_{i-1}(ID, M), A_{i-n,m_{i-n}}).$$

It defines $\bar{H}(ID, M) = \bar{H}_{k=n+l}(ID, M)$, $\widetilde{H}(ID) = \widetilde{H}_n(ID)$.

Then, it sets a randomness extractor, which can extract a uniform random l bits string form $\mathbb{G}_k$, i.e., $\mathsf{ext} : \mathbb{G}_k \to \{0,1\}^l$.

The public parameters, PP, consist of the group sequence description plus: $(A_{1,0}, A_{1,1}), \ldots, (A_{l,0}, A_{l,1}), (B_{1,0}, B_{1,1}), \ldots, (B_{n,0}, B_{n,1})$, ext.

The master secret key MSK includes PP together with the values $(b_{1,0}, b_{1,1}), \ldots, (b_{n,0}, b_{n,1})$.

• **KeyGen**(MSK, $ID \in \{0,1\}^n$) The private key for identity $ID = id_1 \cdots id_n$ is $SK_{ID} = g_{n-1}^{\prod_{j\in[1,n]} b_{j,id_j}} \in \mathbb{G}_{n-1}$.

• **Signcrypt**($M_i \in \{0,1\}^l$, $ID_R \in \{1\} \times \{0,1\}^{n-1}$, $ID_{S_i} \in \{0\}\times\{0,1\}^{n-1}$, $SK_{ID_{S_i}}$, PP) The Signcryption algorithm takes $M_i = m_{i,1} \cdots m_{i,l}$, $ID_R = id_{R,1} \cdots id_{R,n}$, $ID_{S_i} = id_{S_i,1} \cdots id_{S_i,n}$, $SK_{ID_{S_i}} = g_{n-1}^{\prod_{j\in[1,n]} b_{j,id_{S_i,j}}}$ and PP as input, lets temporary variable $D_{i,0} = SK_{ID_{S_i}}$, and chooses $t_i \in \mathbb{Z}_p$ randomly. Then for $j = 1$ to l it computes $D_{i,j} = e(D_{i,j-1}, A_{j,m_{i,j}}) \in \mathbb{G}_{n-1+i}$. The output signcryption is $\sigma_{S_i} = (\sigma_{S_i,0}, \sigma_{S_i,1}, \sigma_{S_i,2})$, where

$$\sigma_{S_i,0} = \mathsf{ext}(e(\widetilde{H}(ID_R), g_l^{t_i})) \oplus M_i, \quad \sigma_{S_i,1} = g_{l+1}^{t_i},$$

$$\sigma_{S_i,2} = D_{i,l} = (g_{k-1})^{\prod_{j\in[1,n]} b_{j,id_{S_i,j}} \cdot \prod_{j\in[1,l]} a_{i,m_{i,j}}}.$$

This serves as an ID-based aggregate signcryption for the (single element) multiset $\mathcal{S} = \{ID_{S_i}\}$.

• **Aggregate**(PP, $\mathcal{S}_x$, $\mathcal{S}_y$, σ_x, σ_y) The aggregation algorithm simply computes the output signcryption σ_z as $\sigma_z = (\sigma_{x,0}\|\sigma_{y,0}, \sigma_{x,1}\|\sigma_{y,1}, \sigma_{x,2} \cdot \sigma_{y,2})$, where $\|$ is a concatenation symbol. This serves as a signcryption on the multiset $\mathcal{S}_z = \mathcal{S}_x \bigcup \mathcal{S}_y$, where $\bigcup$ is a multiset union.

• **Unsigncrypt**(PP, SK_{ID_R}, $\mathcal{S}$, σ) For $\mathcal{S} = \{ID_{S_1}, \ldots, ID_{S_{|\mathcal{S}|}}\}$, $\sigma = \{\ \sigma_{ID_{S_1},0} \| \cdots \| \sigma_{ID_{S_{|\mathcal{S}|}},0}, \sigma_{ID_{S_1},1} \| \cdots \| \sigma_{ID_{S_{|\mathcal{S}|}},1}, \sigma_2\ \}$, the unsigncryption algorithm computes that

$$M_i = \sigma_{ID_{S_i},0} \oplus \mathsf{ext}(e(SK_{ID_R}, \sigma_{ID_{S_i},1})),$$

for $i \in \{1, \ldots, |\mathcal{S}|\}$.

It then accepts if and only if

$$e(\sigma_2, g) \stackrel{?}{=} \prod_{i=1,\ldots,|\mathcal{S}|} \bar{H}(ID_{S_i}, M_i).$$

5.2 Correctness

For correct "decryption", we have

$$\begin{aligned}
\widetilde{H}(ID_R) &= \widetilde{H}_n(ID_R) \\
&= e(\widetilde{H}_{n-1}(ID_R), B_{n,id_{R,n}}) \\
&= e(e(\widetilde{H}_{n-2}(ID_R), B_{n-1,id_{R,n-1}}), B_{n,id_{R,n}}) \\
&\cdots \\
&= e(B_{1,id_{R,1}}, \ldots, B_{n,id_{R,n}}) \\
&= e(g_1^{b_{1,id_{R,1}}}, \ldots, g_1^{b_{n,id_{R,n}}}) \\
&= g_n^{\prod_{j\in[1,n]} b_{j,id_{R,j}}},
\end{aligned}$$

$$\begin{aligned}
e(SK_{ID_R}, \sigma_{ID_{S_i},1}) &= e(g_{n-1}^{\prod_{j\in[1,n]} b_{j,id_{R,j}}}, g_{l+1}^{t_i}) \\
&= g_{n+l}^{\prod_{j\in[1,n]} b_{j,id_{R,j}} \cdot t_i}.
\end{aligned}$$

Therefore, for $i \in [1, |\mathcal{S}|]$:

$$\begin{aligned}
e(\widetilde{H}(ID_R), g_l^{t_i}) &= g_{n+l}^{\prod_{j\in[1,n]} b_{j,id_{R,j}} \cdot t_i} \\
&= e(SK_{ID_R}, \sigma_{ID_{S_i},1}),
\end{aligned}$$

then,

$$\sigma_{ID_{S_i},0} \oplus \mathsf{ext}(e(SK_{ID_R}, \sigma_{ID_{S_i},1})) = M_i.$$

For "authentication", we have

$$\begin{aligned}
\sigma_2 &= \prod_{i\in[1,|\mathcal{S}|]} \sigma_{S_i,2} \\
&= \prod_{i\in[1,|\mathcal{S}|]} (g_{k-1})^{\prod_{j\in[1,n]} b_{j,id_{S_i,j}} \cdot \prod_{j\in[1,l]} a_{j,m_{i,j}}} \\
&= (g_{k-1})^{\sum_{i\in[1,|\mathcal{S}|]} (\prod_{j\in[1,n]} b_{j,id_{S_i,j}} \cdot \prod_{j\in[1,l]} a_{j,m_{i,j}})},
\end{aligned}$$

$$\begin{aligned}
&\prod_{i=1,\ldots,|\mathcal{S}|} \bar{H}(ID_{S_i}, M_i) \\
&= \prod_{i=1,\ldots,|\mathcal{S}|} (g_k)^{\prod_{j\in[1,n]} b_{j,id_{S_i,j}} \cdot \prod_{j\in[1,l]} a_{j,m_{i,j}}} \\
&= (g_k)^{\sum_{i\in[1,|\mathcal{S}|]} (\prod_{j\in[1,n]} b_{j,id_{S_i,j}} \cdot \prod_{j\in[1,l]} a_{j,m_{i,j}})}.
\end{aligned}$$

Therefore, if $(\mathcal{S}, \sigma)$ is a "correct" aggregate signcryption ciphertext, then

$$\begin{aligned}
e(\sigma_2, g) &= \prod_{i=1,\ldots,|\mathcal{S}|} \bar{H}(ID_{S_i}, M_i) \\
&= (g_k)^{\sum_{i\in[1,|\mathcal{S}|]} (\prod_{j\in[1,n]} b_{j,id_{S_i,j}} \cdot \prod_{j\in[1,l]} a_{j,m_{i,j}})}.
\end{aligned}$$

5.3 Security

5.3.1 Unforgeability

Theorem 1 The ID-based aggregate signcryption scheme for message length l and identity length n in Section 5.1 is selectively secure in the unforgeability game under the $(l+n)$-MCDH assumption.

Proof We show that if there exists a PPT adversary $\mathcal{A}$ that can break the selective security of the ID-based aggregate signcryption scheme in the unforgeability game with a non-negligible advantage for message length l, identity length n and security parameter λ, then there exists a PPT simulator that can break the $(l+n)$-MCDH assumption. The simulator takes as input an MCDH instance $(g, g^{c_1}, \ldots, g^{c_k})$ together with the group descriptions where $k = l+n$. Let m_j denote the j-th bit of M and id_j denote the j-th bit of ID. The simulator plays the role of the challenger in the game as follows.

- **Init** Let $ID_S^* \in \{0\} \times \{0,1\}^{n-1}$ and $M^* \in \{0,1\}^l$ be the forgery identity/message pair output by $\mathcal{A}$.
- **Setup** The simulator chooses random $x_1, \ldots, x_l$, $y_1, \ldots, y_n \in \mathbb{Z}_p$. For $i = 1$ to l, let $A_{i,m_i^*} = g^{c_{i+n}}$ and $A_{i,1-m_i^*} = g^{x_i}$. For $i = 1$ to n, let $B_{i,id_{S,i}^*} = g^{c_i}$ and $B_{i,1-id_{S,i}^*} = g^{y_i}$. We remark that the parameters are distributed independently and uniformly at random as in the real scheme.
- **Queries** Conceptually, the simulator will be able to generate keys or signcrypt or unsigncrypt for the adversary, because its requests will differ from the challenge identity or message in at least one bit. More specifically,
  1) Private key query:
     (a) On input an identity $ID_S \in \{0\} \times \{0,1\}^{n-1}$, if $ID_S = ID_S^*$, the simulator will abort. Otherwise, the simulator computes the private key as follows. Let β be the first index such that $id_{S,\beta} \neq id_{S,\beta}^*$. Use $n-2$ pairings on the $B_{\gamma,id_{S,\gamma}}$ values to compute $s = (g_{n-1})^{\prod_{\gamma=1,\ldots,n\wedge\gamma\neq\beta} b_{\gamma,id_{S,\gamma}}}$. Then compute $SK_{ID_S} = s^{y_\beta} = (g_{n-1})^{\prod_{\gamma=1,\ldots,n} b_{\gamma,id_{S,\gamma}}}$. The simulator sends SK_{ID_S} to the adversary $\mathcal{A}$.
     (b) On input an identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$, the simulator uses $n-2$ pairings on the $B_{\gamma,id_{R,\gamma}}$ values to compute $s = (g_{n-1})^{\prod_{\gamma=2,\ldots,n} b_{\gamma,id_{R,\gamma}}}$. Then compute $SK_{ID_R} = s^{y_1} = (g_{n-1})^{\prod_{\gamma=1,\ldots,n} b_{\gamma,id_{R,\gamma}}}$. The simulator sends SK_{ID_R} to the adversary $\mathcal{A}$.

     Private keys are unique and perfectly distributed as in the real game.
  2) Signcrypt query: On input an identity $ID_{S_i} \in \{0\} \times \{0,1\}^{n-1}$, an identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$, and a message $M_i \in \{0,1\}^l$, if $ID_{S_i} \neq ID_S^*$, the simulator creates signcryption ciphertext in the usual way, because it can generate the private keys of any identities $ID_{S_i} \neq ID_S^*$ as in the private key query phase. If $ID_{S_i} = ID_S^*$, then we know $M_i \neq M^*$. The simulator creates $\sigma_{S_i,0}$ and $\sigma_{S_i,1}$ in the usual way. Let β be the first index such that $m_{i,\beta} \neq m_\beta^*$. Use $l-2$ pairings on the $A_{j,m_{i,j}}$ values to compute $\sigma'_{i,2} = (g_{l-1})^{\prod_{j=1,\ldots,l\wedge j\neq\beta} a_{j,m_{i,j}}}$. Next, compute $\sigma''_{i,2} = \sigma'_{i,2}{}^{x_\beta} = (g_{l-1})^{\prod_{j=1,\ldots,l} a_{j,m_{i,j}}}$. Use $n-1$ pairings on the $B_{j,id_{i,j}}$ values to compute $t = (g_n)^{\prod_{j=1,\ldots,n} b_{j,id_{i,j}}}$. Finally, compute $\sigma_{i,2} = e(t, \sigma''_{i,2}) = (g_{k-1})^{\prod_{j\in[1,n]} b_{j,id_{i,j}} \cdot \prod_{j\in[1,l]} a_{i,m_{i,j}}}$. Return $\sigma_{S_i} = (\sigma_{S_i,0}, \sigma_{S_i,1}, \sigma_{S_i,2})$ and $\mathcal{S} = \{ID_{S_i}\}$ to $\mathcal{A}$. Signcryption ciphertexts are unique and perfectly distributed as in the real game.
  3) Unsigncrypt query: The simulator can run unsigncryption in the usual way, because it can generate the private keys of any identities $ID_R \in \{1\} \times \{0,1\}^{n-1}$ freely.
- **Response** $\mathcal{A}$ outputs an aggregate signcryption ciphertext σ^* on multiset $\mathcal{S}^*$ where $ID_S^* \in \mathcal{S}^*$ and M^* is the corresponding message sent from ID_S^*. The simulator will extract from this a solution to the MCDH problem. This works by iteratively computing all the other signcryption in $\mathcal{S}^*$ and then dividing them out of the aggregate until only one or more signcryption ciphertexts on (ID_S^*, M^*) remain. That is, the simulator takes an aggregate for $\mathcal{S}^*$ and computes an aggregate signcryption ciphertext for $\mathcal{S}'$ where $\mathcal{S}'$ has one less identity than $\mathcal{S}^*$ at each step. These signcryption will be computed as in the query phase. Eventually, we have an aggregate instances σ' on $w \geqslant 1$ of (ID_S^*, M^*). We have that $e(\sigma'_2, g) = \bar{H}(ID_S^*, M^*)^w = (g_k)^{w\cdot(\prod_{j\in[1,n]} b_{j,id_{S,j}^*} \cdot \prod_{j\in[1,l]} a_{j,m_j^*})}$ and thus $\sigma'_2 = (g_{k-1})^{w\prod_{j\in[1,k]} c_j}$. The simulator computes $\sigma'^{1/w}_2$ (recall that w is not 0 mod p) which gives $(g_{k-1})^{\prod_{j\in[1,k]} c_j}$ and this is given as the solution to the MCDH problem.

As remarked in the **Setup** and **Query** phase, the responses of the challenger are distributed identically to the real unforgeability game. The simulator succeeds whenever $\mathcal{A}$ does.

5.3.2 Confidentiality

Theorem 2 The ID-based aggregate signcryption scheme for message length l and identity length n in Section 5.1 is IND-sID-CPA under the $(l+n)$-MDDH assumption.

Proof We show that if there exists a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ that can break the IND-sID-CPA security of the ID-based aggregate signcryption scheme in **IBASC-IND-sID-CPA** game with a non-negligible advantage for message length l, identity length n and security parameter λ, then there exists a PPT simulator $\mathcal{B}$ that can break the $(l+n)$-MDDH assumption. The simulator takes as input an MDDH instance $(g, g^{c_1}, \ldots, g^{c_{k+1}}, T)$ together with the group descriptions, where $k = l + n$, and T is identical to $g_k^{\prod_{j\in[1,k+1]} c_j}$ or uniform and independent in $\mathbb{G}_k$. $\mathcal{B}$'s goal is to output 1 if $T = g_k^{\prod_{j\in[1,k+1]} c_j}$ and 0 otherwise. Let m_j denote the jth bit of M and id_j denote the jth bit of ID. The simulator plays the role of the challenger in the game as follows.

- **Init** $\mathcal{A}$ outputs an identity $ID_R^* = id_{R,1}^* \ldots id_{R,n}^*$, where it wishes to be challenged, the $id_{R,j}^*$ is the j-th bit of ID_R^*, and $id_{R,1}^* = 1$.
- **Setup** $\mathcal{B}$ chooses random exponents $(a_{1,0}, a_{1,1}), \ldots, (a_{l,0}, a_{l,1}) \in \mathbb{Z}_p^2$ and sets $A_{i,\alpha} = g^{a_{i,\alpha}}$ for $i \in [1,l]$ and $\alpha \in \{0,1\}$. It also chooses random exponents $b_1, \ldots, b_n \in \mathbb{Z}_p$ and sets $(B_{j,id_j^*} = g^{c_j}, B_{j,(1-id_j^*)} = g^{b_j})$ for $j \in [1,n]$. Then, it sets a randomness extractor $\mathsf{ext} : \mathbb{G}_k \to \{0,1\}^l$, and sends $(A_{1,0}, A_{1,1}), \ldots, (A_{l,0}, A_{l,1}), (B_{1,0}, B_{1,1}), \ldots, (B_{n,0}, B_{n,1})$, ext as well as the group sequence description to $\mathcal{A}$. We remark that the parameters are distributed independently and uniformly at random as in the real scheme.
- **Phases 1 and 2** Conceptually, the simulator will be able to generate keys or signcrypt for the adversary, because its requests will differ from the challenge identity in at least one bit.
  1) Private key query:
     (a) On input an identity $ID_S \in \{0\} \times \{0,1\}^{n-1}$, the simulator uses $n-2$ pairings on the $B_{\gamma,id_{S,\gamma}}$ values to compute $s = (g_{n-1})^{\prod_{\gamma=2,\ldots,n} b_{\gamma,id_{S,\gamma}}}$. Then compute $SK_{ID_S} = s^{y_1} = (g_{n-1})^{\prod_{\gamma=1,\ldots,n} b_{\gamma,id_{S,\gamma}}}$. The simulator sends SK_{ID_S} to the adversary $\mathcal{A}$.
     (b) On input an identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$, if $ID_R = ID_R^*$, the simulator will abort. Otherwise, the simulator computes the private key as follows. Let β be the first index such that $id_{R,\beta} \neq id_{R,\beta}^*$. Use $n-2$ pairings on the $B_{\gamma,id_{R,\gamma}}$ values to compute $s = (g_{n-1})^{\prod_{\gamma=1,\ldots,n\wedge j\neq\beta} b_{\gamma,id_{R,\gamma}}}$. Then compute $SK_{ID_R} = s^{b_\beta} = (g_{n-1})^{\prod_{\gamma=1,\ldots,n} b_{\gamma,id_{R,\gamma}}}$. The simulator sends SK_{ID_R} to the adversary $\mathcal{A}$.

     Private keys are unique and perfectly distributed as in the real game.
  2) Signcrypt: On input an identity $ID_{S_i} \in \{0\} \times \{0,1\}^{n-1}$, an identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$, and a message $M_i \in \{0,1\}^l$, the simulator creates signcryption ciphertext in the usual way, because it can generate the private keys of any identities $ID_{S_i} \in \{0\} \times \{0,1\}^{n-1}$.
- **Challenge** $\mathcal{A}$ chooses a multiset of identities $\mathcal{S}^* = \{ID_{S_1}^*, \ldots, ID_{S_{|\mathcal{S}^*|}}^*\}$, and two multisets of messages $M_1^* = \{M_{1,1}^*, \ldots, M_{1,|\mathcal{S}^*|}^*\}$ and $M_2^* = \{M_{2,1}^*, \ldots, M_{2,|\mathcal{S}^*|}^*\}$. $\mathcal{B}$ generates $SK_{ID_{S_i}^*}$ for $\forall\ ID_{S_i}^* \in \mathcal{S}^*$. Then, it chooses a random bit $\beta \in \{0,1\}$, chooses a random integer $t_i' \in \mathbb{Z}_p$, and calculates the signcryption ciphertexts $\sigma_{S_i}^* = (\sigma_{S_i,0}^*, \sigma_{S_i,1}^*, \sigma_{S_i,2}^*)$ for $ID_{S_i}^* \in \mathcal{S}^*$, where $\sigma_{S_i,2}^*$ is calculated in the usual way, and

$$\sigma_{S_i,0}^* = \mathsf{ext}(T^{t_i'}) \oplus M_{\beta,i}^*,\quad \sigma_{S_i,1}^* = g_{l+1}^{t_i' \prod_{j\in[1,l+1]} c_{n+j}}.$$

  Finally, the challenger aggregates these signcryptions using the aggregation algorithm, and returns the result $\{\sigma^*, \mathcal{S}^*\}$ to $\mathcal{A}$.
- **Guess** $\mathcal{A}$ outputs his guess $b' \in \{0,1\}$ for b.

If $b = 1$ then $\mathcal{A}$ played the proper security game. On the other hand, if $b = 0$, all information about the message M_b^* is lost. Therefore the advantage of $\mathcal{A}$ is exactly 0. As a result if $\mathcal{A}$ breaks the proper security game with a non-negligible advantage, then $\mathcal{B}$ has a non-negligible advantage in breaking the $(l+n)$-MDDH assumption.

6 IB-ASC in the GGH framework

In this section, we show how to modify our ID-based construction to use the GGH [23] graded algebras analogue of multilinear maps. For ease of notation on the reader, we suppress repeated params arguments that are provided to every algorithm. Thus, for instance, we will write $\alpha \leftarrow \mathsf{samp}()$ instead of $\alpha \leftarrow \mathsf{samp}(\mathsf{params})$. Note that in our scheme, there will only ever be a single uniquely chosen value for params throughout the scheme, so there is no cause for confusion. For further details on the GGH framework, please refer to Ref. [23]. The realization method of GGH's graded encoding

system is included in Appendix.

6.1 Construction in the GGH framework

• **Setup**$(1^\lambda, l, n)$ The trusted setup algorithm is run by PKG, the master authority of the ID-based system. It takes as input the security parameter as well the bit-length l of messages and bit-length n of identities. The message space is defined as $\{0,1\}^l$, and the identity space is defined as $\{0,1\}^n$, where the sender identity space is defined as $\{0\}\times\{0,1\}^{n-1}$, and the receiver identity space is defined as $\{1\}\times\{0,1\}^{n-1}$. It then runs $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^k)$, where $k = l+n$. Recall that params will be implicitly given as input to all GGH-related algorithms below.

Next, it chooses random encodings $a_{i,\alpha} = \mathsf{samp}()$ for $i \in [1,l]$, $\alpha \in \{0,1\}$, and $b_{i,\beta} = \mathsf{samp}()$ for $i \in [1,n]$, $\beta \in \{0,1\}$. Then it assigns $A_{i,\alpha} = \mathsf{enc}(1, a_{i,\alpha})$ for $i \in [1,l]$, $\alpha \in \{0,1\}$, and $B_{j,\beta} = \mathsf{enc}(1, b_{j,\beta})$ for $j \in [1,n], \beta \in \{0,1\}$.

These will be used to define two functions $\bar{H}(ID, M) : \{0,1\}^n \times \{0,1\}^l \to \mathbb{G}_k$ and $\widetilde{H}(ID) : \{0,1\}^n \to \mathbb{G}_n$. Let $m_1, \ldots, m_l$ be the bits of message M and $id_1, \ldots, id_n$ as the bits of ID. It is computed iteratively as

$$\bar{H}_1(ID, M) = \widetilde{H}_1(ID) = B_{1,id_1}.$$

For $i \in [2,n]$,

$$\bar{H}_i(ID, M) = \mathsf{mult}(\bar{H}_{i-1}(ID, M), B_{i,id_i}),$$

$$\widetilde{H}_i(ID) = \mathsf{mult}(\widetilde{H}_{i-1}(ID), B_{i,id_i}).$$

For $i \in [n+1, n+l=k]$,

$$\bar{H}_i(ID, M) = \mathsf{mult}(\bar{H}_{i-1}(ID, M), A_{i-n,m_{i-n}}).$$

It defines $\bar{H}(ID, M) = \mathsf{reRand}(k, \bar{H}_k(ID, M))$, $\widetilde{H}(ID) = \mathsf{reRand}(n, \widetilde{H}_n(ID))$.

Then, it sets a pseudorandom number generator $\mathcal{F}(s) \to s'$, where $s \in \{0,1\}^\lambda$ is the seed, and $s' \in \{0,1\}^l$ is a pseudorandom number.

The public parameters, PP, consist of the group sequence description plus: $(A_{1,0}, A_{1,1}), \ldots, (A_{l,0}, A_{l,1}), (B_{1,0}, B_{1,1}), \ldots, (B_{n,0}, B_{n,1}), \mathcal{F}$.

The master secret key MSK includes PP together with the values $(b_{1,0}, b_{1,1}), \ldots, (b_{n,0}, b_{n,1})$.

• **KeyGen**$(MSK, ID \in \{0,1\}^n)$ The private key for identity $ID = id_1 \cdots id_n$ is $SK_{ID} = \mathsf{reRand}(n-1, \mathsf{enc}(n-1, \prod_{i\in[1,n]} b_{i,id_i}))$.

• **Signcrypt**$(M_i \in \{0,1\}^l, ID_R \in \{1\}\times\{0,1\}^{n-1}, ID_{S_i} \in \{0\}\times\{0,1\}^{n-1}, SK_{ID_{S_i}}, PP)$ The Signcryption algorithm takes $M_i = m_{i,1} \cdots m_{i,l}$, $ID_R = id_{R,1} \cdots id_{R,n}$, $ID_{S_i} = id_{S_i,1} \cdots id_{S_i,n}$, $SK_{ID_{S_i}} = \mathsf{reRand}(n-1, \mathsf{enc}(n-1, \prod_{j\in[1,n]} b_{j,id_{S_i,j}}))$ and PP as input, lets temporary variable $D_{i,0} = SK_{ID_{S_i}}$, and chooses encodings $t_i = \mathsf{samp}()$ randomly. Then for $j = 1$ to l it computes $D_{i,j} = \mathsf{mult}(D_{j-1}, A_{j,m_{i,j}}) \in \mathbb{G}_{n-1+i}$. The output signcryption is $\sigma_{S_i} = (\sigma_{S_i,0}, \sigma_{S_i,1}, \sigma_{S_i,2})$, where

$$\sigma_{S_i,0} = \mathcal{F}(\mathsf{ext}(\mathsf{mult}(\widetilde{H}(ID_R), \mathsf{enc}(l, t_i)))) \oplus M_i,$$

$$\sigma_{S_i,1} = \mathsf{enc}(l+1, t_i),$$

$$\sigma_{S_i,2} = \mathsf{reRand}(k-1, D_l).$$

This serves as an ID-based aggregate signcryption for the (single element) multiset $\mathcal{S} = \{ID_{S_i}\}$.

• **Aggregate**$(PP, \mathcal{S}_x, \mathcal{S}_y, \sigma_x, \sigma_y)$ The aggregation algorithm simply computes the output signcryption σ_z as $\sigma_z = (\sigma_{x,0}\|\sigma_{y,0}, \sigma_{x,1}\|\sigma_{y,1}, \sigma_{x,2} + \sigma_{y,2})$, where $\|$ is a concatenation symbol. The serves as a signcryption on the multiset $\mathcal{S}_z = \mathcal{S}_x \cup \mathcal{S}_y$, where $\cup$ is a multiset union.

• **Unsigncrypt**$(PP, SK_{ID_R}, \mathcal{S}, \sigma)$ For $\mathcal{S} = \{ID_{S_1}, \ldots, ID_{S_{|\mathcal{S}|}}\}$, $\sigma = \{\ \sigma_{ID_{S_1},0} \|\ldots\| \sigma_{ID_{S_{|\mathcal{S}|}},0}, \sigma_{ID_{S_1},1} \|\ldots\| \sigma_{ID_{S_{|\mathcal{S}|}},1}, \sigma_2\ \}$, the unsigncryption algorithm computes that

$$M_i = \sigma_{ID_{S_i},0} \oplus \mathcal{F}(\mathsf{ext}(\mathsf{mult}(SK_{ID_R}, \sigma_{ID_{S_i},1}))),$$

for $i \in \{1, \ldots, |\mathcal{S}|\}$.

It then accepts if and only if the under zero testing procedure outputs true.

$$\mathsf{isZero}(\mathbf{p}_{zt}, \mathsf{mult}(\sigma_2, g) - \sum_{i=1,\ldots,|\mathcal{S}|} \bar{H}(ID_{S_i}, M_i)).$$

Correctness Correctness follows from the same argument as for the IBASC scheme in the generic multilinear setting.

6.2 Security

6.2.1 Unforgeability

Theorem 3 The ID-based aggregate signcryption scheme for message length l and identity length n in Section 6.1 is selectively secure in the unforgeability game under the GGH $(l+n)$-MCDH assumption.

Proof We show that if there exists a PPT adversary $\mathcal{A}$ that can break the selective security of the ID-based aggregate signcryption scheme in the unforgeability game with a non-negligible advantage for message length l, identity length n and security parameter λ, then there exists a PPT simulator that can break the GGH $(l+n)$-MCDH assumption. The simulator takes as input a GGH MCDH instance, params, $\mathbf{p}_{zt}$, $C_1 = \mathsf{enc}(1, c_1), \ldots, C_k = \mathsf{enc}(1, c_k)$ where $k = l+n$. Let m_j denote the j-th bit of M and id_j denote the j-th bit of ID. The simulator plays the role of the challenger in the game as follows.

- **Init** Let $ID_S^* \in \{0\} \times \{0,1\}^{n-1}$ and $M^* \in \{0,1\}^l$ be the forgery identity/message pair output by $\mathcal{A}$.
- **Setup** The simulator chooses random $x_1, \ldots, x_l$, $y_1, \ldots, y_n$ with fresh calls to $\mathsf{samp}()$. For $i = 1$ to l, let $A_{i,m_i^*} = C_{i+n}$ and $A_{i,1-m_i^*} = \mathsf{enc}(1, x_i)$. For $i = 1$ to n, let $B_{i,id_{S_i}^*} = C_i$ and $B_{i,1-id_{S_i}^*} = \mathsf{enc}(1, y_i)$. We remark that the parameters are distributed independently and uniformly at random as in the real scheme.
- **Queries** Conceptually, the simulator will be able to generate keys or signcrypt or unsigncrypt for the adversary, because its requests will differ from the challenge identity or message in at least one bit. More specifically,
  1) Private key query:
     (a) On input an identity $ID_S \in \{0\} \times \{0,1\}^{n-1}$, if $ID_S = ID_S^*$, the simulator will abort. Otherwise, the simulator computes the private key as follows. Let β be the first index such that $id_{S,\beta} \neq id_{S,\beta}^*$. Use $n-2$ multiplications on the $B_{\gamma,id_{S,\gamma}}$ values to compute $s = \prod_{\gamma=1,\ldots,n \wedge j \neq \beta} B_{\gamma,id_{S,\gamma}}$. Then compute $SK_{ID_S} = \mathsf{reRand}(n-1, s \cdot y_\beta)$. The simulator sends SK_{ID_S} to the adversary $\mathcal{A}$.
     (b) On input an identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$, the simulator uses $n-2$ multiplications on the $B_{\gamma,id_{R,\gamma}}$ values to compute $s = \prod_{\gamma=2,\ldots,n} B_{\gamma,id_{R,\gamma}}$. Then compute $SK_{ID_R} = \mathsf{reRand}(n-1, s \cdot y_1)$. The simulator sends SK_{ID_R} to the adversary $\mathcal{A}$.

     Private keys are unique and perfectly distributed as in the real game.
  2) Signcrypt: On input an identity $ID_{S_i} \in \{0\} \times \{0,1\}^{n-1}$, an identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$, and a message $M_i \in \{0,1\}^l$, if $ID_{S_i} \neq ID_S^*$, the simulator creates signcryption ciphertext in the usual way. If $ID_{S_i} = ID_S^*$, then we know $M_i \neq M^*$. The simulator creates $\sigma_{S_i,0}$ and $\sigma_{S_i,1}$ in the usual way. Let β be the first index such that $m_{i,\beta} \neq m_\beta^*$. Use $l-2$ multiplications on the $A_{j,m_{i,j}}$ values to compute $\sigma'_{S_i,2} = \prod_{j=1,\ldots,l \wedge j \neq \beta} A_{j,m_{i,j}}$. Next, compute $\sigma''_{S_i,2} = \sigma'_{S_i,2} \cdot x_i$. Use $n-1$ multiplications on the $B_{j,id_{S_i,j}}$ values to compute $\gamma = \prod_{j=1,\ldots,n} B_{j,id_{S_i,j}}$. Finally, compute $\sigma_{S_i,2} = \mathsf{reRand}(k-1, \gamma \cdot \sigma''_{S_i,2})$. Return $\sigma_{S_i} = (\sigma_{S_i,0}, \sigma_{S_i,1}, \sigma_{S_i,2})$ and $\mathcal{S} = \{ID_{S_i}\}$ to $\mathcal{A}$. Signcryption ciphertexts are unique and perfectly distributed as in the real game.
  3) Unsigncrypt: The simulator can run unsigncryption in the usual way, because it can generate the private keys of any identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$ freely.
- **Response** $\mathcal{A}$ outputs an aggregate signcryption ciphertext σ^* on multiset $\mathcal{S}^*$ where $ID_S^* \in \mathcal{S}$ and M^* is the corresponding message sent from ID_S^*. The simulator will extract from this a solution to the GGH MCDH problem. This works by iteratively computing all the other signatures in $\mathcal{S}^*$ and then dividing them out of the aggregate until only one or more signcryption ciphertexts on (ID_S^*, M^*) remain. That is, the simulator takes an aggregate for $\mathcal{S}^*$ and computes an aggregate signcryption ciphertext for $\mathcal{S}'$ where $\mathcal{S}'$ has one less identity than $\mathcal{S}^*$ at each step. These signcryption will be computed as in the query phase. Eventually, we have an aggregate instances σ' on $w \geqslant 1$ of (ID_S^*, M^*). However recall that $\bar{H}(ID_S^*, M^*)$ is a level k encoding of $(\prod_{j\in[1,n]} b_{j,id_{S,j}^*}) \cdot (\prod_{j\in[1,l]} a_{j,m_j^*}) = \prod_{j\in[1,k]} c_j$. Thus verification of the signcryption ciphertexts implies that (w, σ'_2) is a solution to the GGH k-MCDH problem, and so the simulator returns (w, σ'_2) to break the GGH k-MCDH assumption.

As remarked in the **Setup** and **Query** phase, the responses of the challenger are distributed identically to the real unforgeability game. The simulator succeeds whenever $\mathcal{A}$ does.

6.2.2 Confidentiality

Theorem 4 The ID-based aggregate signcryption scheme for message length l and identity length n in Section 6.1 is IND-sID-CPA under the GGH $(l+n)$-MDDH assumption.

Proof We show that if there exists a PPT adversary $\mathcal{A}$ that can break the IND-sID-CPA security of the ID-based aggregate signcryption scheme in **IBASC-IND-sID-CPA** game with a non-negligible advantage for message length l, identity length n and security parameter λ, then there exists a PPT simulator $\mathcal{B}$ that can break the GGH $(l+n)$-MDDH assumption. The simulator takes as input an GGH MDDH instance, params, $\mathbf{p}_{zt}$, $C_1 = \mathsf{enc}(1, c_1), \ldots, C_{k+1} = \mathsf{enc}(1, c_{k+1})$ and a level-k encoding T, where $k = l+n$. Algorithm $\mathcal{B}$'s goal is to output 1 if $\mathsf{isZero}(\mathbf{p}_{zt}, \mathsf{reRand}(T)\text{-}\mathsf{reRand}(\mathsf{enc}(\mathsf{params}, k, \prod_{j\in[1,k+1]} c_j)))$=1 and 0 otherwise. Let m_j denote the j-th bit of M and id_j denote the j-th bit of ID. The simulator plays the role of the challenger in the game as follows.

- **Init** $\mathcal{A}$ outputs an identity $ID_R^* = id_{R,1}^* \cdots id_{R,n}^*$, where

it wishes to be challenged, the $id^*_{R,j}$ is the jth bit of ID^*_R.

- **Setup** $\mathcal{B}$ chooses random $a_{1,0}, a_{1,1} \ldots, a_{l,0}, a_{l,1}, b_1, \ldots, b_n$ with fresh calls to $\mathsf{samp}()$, and sets $(A_{i,0} = \mathsf{enc}(1, a_{i,0}), A_{i,1} = \mathsf{enc}(1, a_{i,1}))$ for $i \in [1, l]$, sets $(B_{i,id^*_{R,i}} = C_i,\ B_{i,(1-id^*_{R,i})} = \mathsf{enc}(1, b_i))$ for $i \in [1, n]$. We remark that the parameters are distributed independently and uniformly at random as in the real scheme.
- **Phases 1 and 2** Conceptually, the simulator will be able to create keys or signcrypt for the adversary, because his requests will differ from the challenge identity in at least one bit.
  1) Private key query:
     (a) On input an identity $ID_S \in \{0\} \times \{0,1\}^{n-1}$, the simulator uses $n-2$ multiplications on the $B_{\gamma,id_{S,\gamma}}$ values to compute $s = \prod_{\gamma=2,\ldots,n} B_{\gamma,id_{R,\gamma}}$. Then compute $SK_{ID_S} = \mathsf{reRand}(n-1, s \cdot b_1)$. The simulator sends SK_{ID_S} to the adversary $\mathcal{A}$.
     (b) On input an identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$, if $ID_R = ID^*_R$, the simulator will abort. Otherwise, the simulator computes the secret key as follows. Let β be the first index such that $id_{R,\beta} \neq id^*_{R,\beta}$. Use $n-2$ multiplications on the $B_{\gamma,id_{R,\gamma}}$ values to compute $s = \prod_{\gamma=1,\ldots,n \wedge \gamma \neq \beta} B_{\gamma,id_{R,\gamma}}$. Then compute $SK_{ID_R} = \mathsf{reRand}(n-1, s \cdot b_\beta)$. The simulator sends SK_{ID_R} to the adversary $\mathcal{A}$.

     Private keys are unique and perfectly distributed as in the real game.
  2) Signcrypt: On input an identity $ID_{S_i} \in \{0\} \times \{0,1\}^{n-1}$, an identity $ID_R \in \{1\} \times \{0,1\}^{n-1}$, and a message $M_i \in \{0,1\}^l$, the simulator creates signcryption ciphertext in the usual way.
- **Challenge** $\mathcal{A}$ chooses a multiset of identities $\mathcal{S}^* = \{ID^*_{S_1}, \ldots, ID^*_{S_{|S^*|}}\}$, and two multisets of messages $M^*_1 = \{M^*_{1,1}, \ldots, M^*_{1,|S^*|}\}$ and $M^*_2 = \{M^*_{2,1}, \ldots, M^*_{2,|S^*|}\}$. $\mathcal{B}$ generates $SK_{ID^*_{S_i}}$ for $\forall\ ID^*_{S_i} \in \mathcal{S}^*$. Then, it chooses a random bit $\beta \in \{0,1\}$, chooses random t'_i with fresh calls to $\mathsf{samp}()$, and calculates the the signcryption ciphertexts $\sigma^*_{S_i} = (\sigma^*_{S_i,0}, \sigma^*_{S_i,1}, \sigma^*_{S_i,2})$ for $ID^*_{S_i} \in \mathcal{S}^*$, where $\sigma_{S_i,2}$ is calculated in the usual way, and

$$\sigma_{S_i,0} = \mathcal{F}(\mathsf{ext}(T \cdot t'_i)) \oplus M^*_{\beta,i},$$

$$\sigma_{S_i,1} = t'_i \cdot \prod_{j \in [1,l+1]} C_{n+j}.$$

Finally, the challenger aggregates these signcryptions using the aggregation algorithm, and returns the result $\{\sigma^*, \mathcal{S}^*\}$ to $\mathcal{A}$.

- **Guess** $\mathcal{A}$ outputs his guess $b' \in \{0,1\}$ for b.

If $b = 1$ then $\mathcal{A}$ played the proper security game. On the other hand, if $b = 0$, all information about the message M^*_b is lost. Therefore the advantage of $\mathcal{A}$ is exactly 0. As a result if $\mathcal{A}$ breaks the proper security game with a non-negligible advantage, then $\mathcal{B}$ has a non-negligible advantage in breaking the GGH $(l+n)$-MDDH assumption.

7 Conclusions

We construct a new identity-based aggregate signcryption scheme in multilinear map setting, and prove its security in the standard model. To the best of my knowledge, this is the first identity-based aggregate signcryption scheme that is secure in the standard model.

This work motivates two open problems. The first is to find an efficient identity-based aggregate signcryption scheme (without random oracles). The second is to find an identity-based aggregate signcryption scheme with full security.

Acknowledgements This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 61272434, 61502218, and 61572294), the Natural Science Foundation of Shandong Province (ZR2013FQ021), and the Outstanding Young Scientists Foundation Grant of Shandong Province (BS2014DX016).

Appendix

Realization of graded encoding system

GGH's n-graded encoding system works as follows. (This is a whirlwind overview, see [23] for details.) The system uses three rings. First, it uses the ring of integers O of the mth cyclotomic field. This ring is typically represented as the ring of polynomials $O = \mathbb{Z}[x]/(\Phi_m(x))$, where $\Phi_m(x)$ is m-th cyclotomic polynomial, which has degree $N = \phi(m)$. Second, for some suitable integer modulus q, it uses the quotient ring $O/(q) = \mathbb{Z}_q[x]/(\Phi_m(x))$, similar to the NTRU encryption scheme [26]. The encodings live in $O/(q)$. Finally, it uses the quotient ring $R = O/\mathcal{I}$, where $\mathcal{I} = \langle g \rangle$ is a principal ideal of O that is generated by g and where $|O/\mathcal{I}|$ is a large prime. This is the ring "R" referred to above; elements of R are what is encoded.

What does a GGH encoding look like? For a fixed random $z \in O/(q)$, an element of $S_i^{(\alpha)}$ — that is, a level-i encoding of $\alpha \in R$ — has the form $e/z^i \in O/(q)$, where $e \in O$

is a "small" representative of the coset $\alpha + \mathcal{I}$ (it has coefficients that are very small compared to q). To add encodings $e_1/z^i \in S_i^{(\alpha_1)}$ and $e_2/z^i \in S_i^{(\alpha_2)}$, just add them in $\mathcal{O}/(q)$ to obtain $(e_1 + e_2)/z^i$, which is in $S_i^{(\alpha_1+\alpha_2)}$ if $e_1 + e_2$ is "small". To mult encodings $e_1/z^{i_1} \in S_{i_1}^{(\alpha_1)}$ and $e_2/z^{i_2} \in S_{i_2}^{(\alpha_2)}$, just multiply them in $\mathcal{O}/(q)$ to obtain $(e_1 \cdot e_2)/z^{i_1+i_2}$, which is in $S_{i_1+i_2}^{(\alpha_1\cdot\alpha_2)}$ if $e_1 \cdot e_2$ is "small". This smallness condition limits the GGH encoding system to degree polynomial in the security parameter. Intuitively, dividing encodings does not "work", since the resulting denominator has a nontrivial term that is not z.

The GGH params allow everyone to generate encodings of random (known) values. The params include a level-1 encoding of 1 (from which one can generate encodings of 1 at other levels), and (for each $i \in [n]$) a sufficient number of level-i encodings of 0 to enable re-randomization. To encode (say at level-1), run samp(params) to sample a small element a from $\mathcal{O}$, e.g., according to a discrete Gaussian distribution. For a Gaussian with appropriate deviation, this will induce a statistically uniform distribution over the cosets of $\mathcal{I}$. Then, multiply a with the level-1 encoding of 1 to get a level-1 encoding u of $a \in R$. Finally, run reRand(params, 1, u), which involves adding a random Gaussian linear combination of the level-1 encodings of 0, whose noisiness (i.e., numerator size) "drowns out" the initial encoding. The parameters for the GGH scheme can be instantiated such that the re-randomization procedure can be used for any pre-specified polynomial number of times.

To permit testing of whether a level-n encoding $u = e/z^n \in S_n$ encodes 0, GGH publishes a level-n zero-test parameter $\mathbf{p}_{zt} = hz^n/g$, where h is "somewhat small" and g is the generator of $\mathcal{I}$. The procedure isZero(params, $\mathbf{p}_{zt}$, u) simply computes $\mathbf{p}_{zt}\cdot u$ and tests whether its coefficients are small modulo q. If u encodes 0, then $e \in I$ and equals $g \cdot c$ for some (small) c, and thus $\mathbf{p}_{zt}\cdot u = h\cdot c$ has no denominator and is small modulo q. If u encodes something nonzero, $\mathbf{p}_{zt}\cdot u$ has g in the denominator and is not small modulo q. The ext(params, $\mathbf{p}_{zt}$, u) procedure works by applying a strong extractor to the most significant bits of $\mathbf{p}_{zt}\cdot u$. For any two $u_1, u_2 \in S_n^{(\alpha)}$, we have (subject to noise issues) $u_1 - u_2 \in S_n^{(0)}$, which implies $\mathbf{p}_{zt}(u_1 - u_2)$ is small, and hence $\mathbf{p}_{zt}\cdot u_1$ and $\mathbf{p}_{zt}\cdot u_2$ have the same most significant bits (for an overwhelming fraction of α's).

Garg et al. provide an extensive cryptanalysis of the encoding system, which we will not review here. We remark that the underlying assumptions are stronger, but related to, the hardness assumption underlying the NTRU encryption scheme: that it is hard to distinguish a uniformly random element from $\mathcal{O}/(q)$ from a ratio of "small" elements i.e., an element $u/v \in \mathcal{O}/(q)$ where $u, v \in \mathcal{O}/(q)$ both have coefficients that are on the order of (say) q^ϵ for small constant ϵ.

References

1. Zheng Y L. Digital signcryption or how to achieve cost (signature & encryption) << cost(signature) + cost(encryption). In: Proceedings of the 17th Annual International Cryptology Conference. 1997, 165–179
2. Baek J, Steinfeld R, Zheng Y L. Formal proofs for the security of signcryption. In: Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptosystems. 2002, 80–98
3. Zheng Y L, Imai H. How to construct efficient signcryption schemes on elliptic curves. Information Processing Letters, 1998, 68(5): 227–233
4. Bao F, Deng R H. A signcryption scheme with signature directly verifiable by public key. In: Proceedings of the 1st International Workshop on Practice and Theory in Public Key Cryptography. 1998, 55–59
5. Hwang R S, Lai C H, Su F F. An efficient signcryption scheme with forward secrecy based on elliptic curve. Applied Mathematics and Computation, 2005, 167(2): 870–881
6. Shamir A. Identity-based cryptosystems and signature schemes. In: Proceedings of CRYPTO. 1984, 47–53
7. Malone-Lee J. Identity-based signcryption. IACR Cryptology ePrint Archive, 2002, 98
8. Libert B, Quisquater J J. New identity based signcryption schemes from pairings. IACR Cryptology ePrint Archive, 2003, 23
9. Chow S S M, Yiu S M, Hui L C K, Chow K P. Efficient forward and provably secure id-based signcryption scheme with public verifiability and public ciphertext authenticity. In: Proceedings of the 6th International Conference on Information Security and Cryptology (ICISC 2003). 2003, 352–369
10. Boyen X. Multipurpose identity-based signcryption. In: Proceedings of the 23rd Annual International Cryptology Conference. 2003, 383–399
11. Chen L Q, Malone-Lee J. Improved identity-based signcryption. In: Proceedings of the 8th International Workshop on Theory and Practice in Public Key Cryptography. 2005, 362–379
12. Barreto P S L M, Libert B, McCullagh N, Quisquater J J. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In: Proceedings of the 11th International Conference on the Theory and Application of Cryptology and Information Security. 2005, 515–532
13. Selvi S S D, Vivek S S, Shriram J, Kalaivani S, Rangan C P. Identity based aggregate signcryption schemes. In: Proceedings of the 10th International Conference on Cryptology in India. 2009, 378–397
14. Ren X Y, Qi Z H, Geng Y. Provably secure aggregate signcryption scheme. ETRI Journal, 2012, 34(3): 421–428
15. Qi Z H, Ren X Y, Geng Y. Provably secure general aggregate signcryption scheme in the random oracle modele. China Communications, 2012, 9(11): 107–116
16. Kar J. Provably secure identity-based aggregate signcryption scheme in random oracles. IACR Cryptology ePrint Archive, 2013, 37
17. Dent A W. Aggregate signcryption. IACR Cryptology ePrint Archive, 2012, 200
18. Eslami Z, Pakniat N. Certificateless aggregate signcryption schemes. IACR Cryptology ePrint Archive, 2011, 360
19. Lu H J, Xie Q. An efficient certificateless aggregate signcryption scheme from pairings. In: Proceedings of 2011 International Conference on Electronics, Communications and Control (ICECC-2011).

2011, 132–135

20. Canetti R, Goldreich O, Halevi S. The random oracle methodology, revisited. Journal of the ACM, 2004, 51(4): 557–594
21. Hohenberger S, Sahai A, Waters B. Full domain hash from (leveled) multilinear maps and identity-based aggregate signatures. In: Proceedings of the 33rd Annual Cryptology Conference, Part I. 2013, 494–512
22. Boneh D, Silverberg A. Applications of multilinear forms to cryptography. Contemporary Mathematics, 2003, 324(1): 71–90
23. Garg S, Gentry C, Halevi S. Candidate multilinear maps from ideal lattices. In: Proceedings of the 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques. 2013, 1–17
24. Freire E S V, Hofheinz D, Paterson K G, Striecks C. Programmable hash functions in the multilinear setting. In: Proceedings of the 33rd Annual Cryptology Conference, Part I. 2013, 513–530
25. Wang H, Zheng Z H, Yang B. New identity-based key-encapsulation mechanism and its applications in cloud computing. International Journal of High Performance Computing and Networking, 2015, 8(2): 124–134
26. Hoffstein J, Pipher J, Silverman J H. NTRU: a ring-based public key cryptosystem. In: Proceedings of the 3rd Intemational Symposium on Algorithmic Number Theory (ANTS-III). 1998, 267–288

Hao Wang received his BS degree in information and computing science from Qufu Normal University, China in 2007, and his PhD degree in computer science from Shandong University, China in 2012. He is currently a lecturer in Shandong Normal University. His primary interest is public-key cryptography, in particular, designing cryptographic primitives and provable security. Currently, he is focusing on attribute-based cryptography and security in cloud computing.

Zhen Liu received the BS degree in applied mathematics and MS degree in mathematics from Shanghai Jiao Tong University, China, and received the PhD degrees in computer science from City University of Hong Kong, China and Shanghai Jiao Tong University, China. His primary interest is applied cryptography, in particular, encryption and signature schemes. With a mathematical background, he is interested in studying provable security and designing cryptographic primitives, for the research problems motivated by practical applications. Currently, he is focusing on lattice-based cryptography (mainly on lattice-based functional encryption) and Fully Homomorphic Encryption (FHE).

Zhe Liu received his BE degree in software engineering from Shandong University, China with first class honors, MS degrees in computer science from University of Luxembourg, Luxembourg and Shandong University, China in 2010 and 2011, respectively. Since 2011, he has been a PhD student in the Laboratory of Algorithmics, Cryptology and Security (LACS), University of Luxembourg. His research interests include computer arithmetics, with special focus on efficient implementation of Public key Cryptography (PKC) on Wireless Sensor Networks (WSNs).

Duncan S. Wong received the BE degree from the University of Hong Kong, China in 1994, the MPhil degree from the Chinese University of Hong Kong, China in 1998, and the PhD degree from Northeastern University, USA in 2002. He has been with the Chinese University of Hong Kong and then the City University of Hong Kong. He is currently a director in Security and Data Sciences at Hong Kong Applied Science and Technology Research Institute (ASTRI), China. His primary research interest is cryptography, in particular, cryptographic protocols, encryption and signature schemes, and anonymous systems. He is also interested in other topics in information security, such as network security, wireless security, database security, and security in cloud computing.