Identity-Based Aggregate and Multi-Signature Schemes based on RSA

Ali Bagherzandi and Stanisław Jarecki

Department of Computer Science, University of California, Irvine
{zandi,stasio}@ics.uci.edu.

Abstract. We propose new identity-based multi-signature (IBMS) and aggregate signature (IBAS) schemes, secure under RSA assumption. Our schemes reduce round complexity of previous RSA-based IBMS scheme of Bellare and Neven [BN07] from three to two rounds. Surprisingly, this improvement comes at virtually no cost, as the computational efficiency and exact security of the new scheme are almost identical to those of [BN07]. The new scheme is enabled by a technical tool of independent interest, a class of zero-knowledge proofs of knowledge of preimages of one-way functions which is straight-line simulatable, enabling concurrency and good exact security, and *aggregatable*, enabling aggregation of parallel instances of such proofs into short multi/aggregate signatures.

1 Introduction

A multisignature protocol allows a group of players to sign the same message by generating a short string, called a multisignature, which can be verified against the set of the public keys of these players. Aggregate signature is a generalization of this notion to the case where each player signs a potentially different message. Such schemes reduce the bandwidth needed to transmit signatures, the space needed to store them, and the time needed to verify them, from linear in the number of the cosigners to a constant. Reducing bandwidth is especially important for low-energy devices, such as RFID chips and sensors, which communicate over energy-consuming wireless channels where data transmision consumes several orders of magnitude more energy than arithmetic operations (see *e.g.* [BA03]). Standard multi-/aggregate signatures reduce the space taken by n signatures from $O(n)$ to $O(1)$, but the verifiers still need the public keys of n signers. Therefore in applications where bandwidth is a bottleneck it can be useful to consider *identity-based* multi-/aggregate signatures where verifiers only need unique identifiers of signers, e.g. 32-bit IP addresses, instead of public keys.

Identity-Based (Multi-/Aggregate) Signatures: Identity-based cryptography [Sha84] simplifies public key management by replacing users' public keys with their identity *e.g.* their names, e-mails or IP addresses. In identity-based scheme a trusted party, a Private Key Generator (PKG), generates a private key corresponding to each user's identity, and messages signed using such keys can be then verified using the signer's identity and the PKG's master public key. In the case of identity-based multi-/aggregate signatures, if all signers have their private keys issued by the same PKG then the verifier needs only the PKG's master public key and the identities of all signers. Note that in many applications the identities of signers are often present in the protocol messages, *e.g.* the usernames or IP addresses in packet headers, in which case an identity-based multi-/aggregate signature adds only a constant bandwidth overhead over unauthenticated messages.

Current State of the Art: Standard signatures imply identity-based signatures following the "certification paradigm", *e.g.* [GHK06], *i.e.* by simply attaching signer's public key and certificate to each signature. However, it is not clear how to apply this idea to convert standard multi-/aggregate signatures, *e.g.* [BN06,BCJ08], into identity-based ones, because it is not clear how to aggregate n separate public keys and certificates, even if all certificates are signed by the same CA. (Standard aggregate signatures can be used to eliminate the overhead of CA's signatures on the certificates, but this would not eliminate the overhead due to the public keys.)

The first efficient IBAS/IBMS schemes designed from scratch are due to Gentry and Ramzan [GR06]. Their schemes employ a group with a bilinear map, their security relies, in the Random Oracle Model (ROM), on the hardness of GapDH problem, the schemes are non-interactive, and both the signing and verification times take $O(1)$ exponentiations and bilinear map operations. However, the IBAS scheme of [GR06] requires all cosigners to share a common token for every set of signatures they want to aggregate, and each cosigner must ensure that this token has not been previously used in signing a different message, hence in some applications this scheme will need an extra communication round for the participants to agree on a fresh common token. In subsequent work, Boldyreva *et. al* [BGOY10] (correcting a previous version of this paper) proposed an IBAS scheme which does not need these unique tokens but it requires sequential communication pattern, and it is based on a more complex bilinear map assumption. Note that while sequential communication is perfectly suited to some applications, *e.g.* secure route discovery [KT05], it introduces unnecessary overhead for players connected *e.g.* by a broadcast channel or a tree topology.

Without bilinear maps, Bellare and Neven [BN07] gave an IBMS scheme which relies on the RSA assumption in ROM. Their scheme also has fast multi-signature generation and verification, requiring $O(1)$ exponentiations, but it takes three rounds of interaction. Note that any 3-round IBMS implies a 4-round IBAS if all cosigners' messages are broadcast and the IBMS scheme is run on their concatenation. (Moreover, in the IBMS scheme of [BN07] this broadcast can be piggybacked on the first protocol round, giving a 3-round IBAS scheme.) However, such broadcast of all messages to all co-signers imposes bandwidth usage which might not be otherwise required, and so apart from this generic transformation it is interesting to consider IBAS schemes which do not require such broadcast. (As a side remark, we believe that the 3-round IBMS scheme of [BN07] can be modified to a 3-round IBAS scheme without such broadcast, *e.g.* using ideas similar to our IBAS scheme [BJ10].)

IBAS/IBMS Schemes	Underlying Problem$^{(1)}$	Restr-ictions$^{(2)}$	Number of Rounds	Verification Time$^{(3)}$	Signing Time$^{(3)}$	Signature Length$^{(4)}$
[GR06]-IBAS	GapDH	Stateful	1	3P+nM	5M	$2\|G_1\| + \kappa$
[BGOY10]	GapDH	Sequential	1	6P+nM	7M	$3\|G\|$
OUR IBAS	RSA	-	2	nE	2E	$\|\mathbb{Z}_n^*\| + 2\kappa + \log l$
[GR06]-IBMS	GapDH	-	1	3P	3M	$2\|G_1\|$
[BN07]	RSA	-	3	1E	2E	$\|\mathbb{Z}_n^*\| + \kappa$
OUR IBMS	RSA	-	2	1E	2E	$\|\mathbb{Z}_n^*\| + 2\kappa + \log l$

Fig. 1. (1) All schemes have been given security proofs only in the *ROM* model; (2) The IBAS scheme of [GR06] assumes that the players share a unique and common token for every instance of the IBAS scheme. This requirement can be avoided at the cost of an additional round of interaction, while the scheme of [BGOY10] requires sequential aggregation; (3) Signing time is measured per player. In both signing and verification costs, P is the cost of one pairing operation, M is the cost of scalar multiplication on an elliptic curve, and E is the cost of (multi-)exponentiation in $\mathbb{Z}_n^*$ (with about 80-bit exponents); (4) Signature length is measured in bits where κ is the security parameter, n is an RSA modulus, l is an upper bound on the number of players, G_1 and G_2 are two groups of elliptic curve points with an asymmetric bilinear map, G is a group of elliptic curve points with a symmetric bilinear map, and $|A|$ stands for the bitsize of representation of elements in group A. Typical values for these parameters are $\kappa = 160$, $|G_1| = 160$, $|G| = 512$, $\log l = 20$, and $|\mathbb{Z}_n^*| = 1024$ or 2048.

Our Contributions: We propose IBMS and IBAS schemes secure under RSA assumption in ROM which require only two rounds of communication. This provides alternatives to IBMS/IBAS schemes based on bilinear maps especially in applications which intrinsically take two communication rounds, such as authenticated route discovery or aggregation of broadcast acknowledgements. Since bilinear map operations are still more expensive than RSA exponentiation, our computational costs are slightly lower in signing and significantly lower in verification, compared to *e.g.* [GR06], although our signatures are longer. A summary of these comparisons is in Figure 1.

Further Related Work: Gregory Neven introduced two primitives, *sequential aggregate signed data* and *multi-signed data*, corresponding to aggregate signatures and multisignatures respectively, whose goal is to minimize the total bandwidth consumed by signatures *and messages* incurred in transmission of authenticated data originated by multiple sources [Nev08]. His constructions use message recovery techniques to squeeze message bits into a (multi/aggregate) signature. Comparing his work to ours, we note that (1) his schemes support only sequential aggregation when signing different messages; (2) bandwidth savings depend on message sizes (for small messages the bandwidth can be worse than with standard signatures); (3) these schemes do not address the overhead due to public keys, which raises an interesting question whether total bandwidth due to signatures and messages can be further reduced, perhaps using message-recovery techniques, with *identity-based* multi/aggregate signatures. In other related work Herranz and Galindo *et. al* [Her06,GHK06] show identity-based signatures which can be aggregated if they originate from *the same signer.*

Organization/Roadmap: Section 2 contains a technical overview of our constructions. In Section 3 we define IBMS schemes. (We relegate a formal description of IBAS schemes to [BJ10].) In Section 4 we develop our tools, namely we introduce structured-instance zero-knowledge (ZK) proofs and Σ-equivocable commitments and we show that Σ-equivocable commitments suffice to compile a class of Σ-protocols which includes an RSA-based identification protocol, a proof of knowledge of e-th root, to straight-line simulatable structured-instance ZK. In Section 5 we show *homomorphic* Σ-equivocable commitments secure under RSA. By the results from Section 4 this implies an *aggragatable* structured-instance ZK proof of knowledge of e-th root, which leads to an IBMS scheme construction, described in Section 6, and an IBAS scheme sketched in Section 7.

2 Technical Overview

Our IBAS/IBMS scheme is a multi-prover version of Guillou-Quisquater signature [GQ88]. The ID-based version of GQ signature is a non-interactive zero-knowledge (NIZK) proof of knowledge (PK) of e-th root modulo n (in ROM). Let $y = H(ID)$ be an element in $\mathbb{Z}_n^*$ and let x be the e-th root of y, a private key corresponding to identity ID. (Such private key can be computed the PKG who knows the factorization of n.) To sign message m, the signer with identity ID follows the ROM-based NIZK PK of e-th root of y: It computes the first proof message $a = k^e$ for random k in $\mathbb{Z}_n^*$, gets challenge c by querying (m, a) to a hash function (modeled as random oracle), and computes response z to this challenge as $z = kx^c$. The signature is (a, z) verified by checking if $z^e = ay^c$ for $c = H(m, a)$. Due to homomorphic property of exponentiation one might hope to obtain an IBAS/IBMS scheme by aggregating such ROM-based NIZK PK's of e-th root made by several cosigners. For instance, consider the two-round protocol built along the lines of the DL-based multisignature scheme of [MOR01]: In the first round each player broadcasts its first message a_i. All players obtain a common challenge c by querying the hash function on input including $a = \prod a_i$ and the message being signed. Finally each player broadcasts its response z_i to this challenge. The multi-signature is (a, z) where $z = \prod z_i$. Note that if $z_i^e = a_i y_i^c$ for each i then (a, z) satisfies the verification equation $z^e = a(\prod y_i)^c$ where $y_i = H(ID_i)$. We believe that an adaptation of the security proof of [MOR01] would show security of this scheme, but the resulting security argument would have several limitations: (1) The reduction would be only from expected-time hardness of RSA problem; (2) It would encounter substantial security degradation due to extensive use of rewindings; (3) It would therefore not extend to concurrent executions of multiple instances of this scheme.

To explain how we overcome these limitations we need to first explain why they appear in the above draft scheme. The simulator for the NIZK PK of e-th root picks a random challenge c and a random z in $\mathbb{Z}_n^*$, computes prover's first message as $a = z^e y^{-c}$ and defines the hash of (a, m) as c because it controls the hash function H. Note that since the adversary has no information about a, there is only a negligible chance that it queries H on the same (a, m) before the simulator attempts to define its value as c, and hence the simulator passes with overwhelming probability. The fundamental difference between this simulation and the simulation for aggregated proof in the draft scheme above is that in the aggregated proof corrupt cosigners can choose their contributions a_i on the basis of a_i's broadcasted by the honest cosigners. Consequently, the simulator can only guess the resulting a value with probability $1/q_h$ where q_h is the number of hash queries the adversary makes. This gives rise to a simulation procedure which rewinds the adversary expected q_h times in each signature instance, which causes all the limitations listed above: reduction to expected-time hardness, loose security reduction, and no argument for security of concurrent protocol instances.

Bellare and Neven [BN07] showed how to overcome all these issues in the ROM model by adding an extra communication round in which each player first commits to its a_i contribution by broadcasting a hash $H(a_i)$. By controlling the hash function H the simulator can learn the a_i's committed by the adversary and then decide on the a_i's published on behalf of the honest players. This way the simulator passes without rewinding with overwhelming probability, similarly to the NIZK simulation sketched above. The main technical challenge we handle in this work is how to achieve such straight-line simulation without introducing such extra communication round, i.e. with only two rounds of interaction. Our technique is a variant of Damgard's HVZK-to-ZK compilation [Dam00] which constructs a straight-line simulatable zero-knowledge proof from any Σ-protocol using an equivocable commitment scheme, but we introduce an interesting twist: In Damgard's scheme a signer commits to its a_i value using an equivocable commitment scheme, and the simulator, on any challenge c can open this commitments to the value $a_i = z_i^e y_i^{-c}$ needed for the proof to verify (where response z_i is chosen at random, to match the response distribution in the real proof). However, to create an IBMS/IBAS scheme by aggregating such proofs we need this commitment scheme to be multiplicatively homomorphic, and to the best of our knowledge no efficient commitment scheme is both equivocable and multiplicatively homomorphic. Instead, we show a commitment scheme which is multiplicatively homomorphic over $\mathbb{Z}_n^*$ and satisfies a restricted form of equivocability which we call *Σ-equivocability*, and which suffices for straight-line simulation of Σ-protocol compiled as above. For example, Σ-equivocable commitment for relation $R = \{(x, y) \,|\, y = x^e\}$ allows for equivocation of commitments to messages of the form $z^e y^{-c}$ for any c and z, and this is exactly the form of message a which the simulator needs in the above proof.

The idea to use commitments with similarly restricted equivocability appeared before in [BCJ08], where it was used to construct a straight-line simulatable and aggregatable proof of DL knowledge, and a DL-based multi-signature scheme. However, the equivocability notion (and the construction) of [BCJ08] gives rise to only single-instance zero-knowledge proofs. Intuitively, this suffices for security of multi-signatures (as opposed to identity-based multi-signatures) because in multi-signatures the adversary w.l.o.g. corrupts all players except of one, so the simulator needs to embed its challenge problem in just one public key, and needs to simulate multi-signature protocol on behalf of only that one player. Using this form of equivocation in security argument for identity-based schemes would introduce security degradation by factor of q_H, the number of hash function queries, because the simulator would have to guess the single identity into which to embed its challenge. Here we define a more general notion of Σ-equivocability which allows for straight-line simulatable "structured instance" zero-knowledge proofs in the CRS model: In structured-instance zero-knowledge proofs, formalized in this paper, the simulator can simulate on any statement in a class of *related* instances, in contrast to a single statement in single-instance ZK and any instance in (standard) multi-instance ZK. The class of instances which is particularly useful in showing a security reduction for an IBMS/IBAS scheme based on Σ-protocol for proving knowledge of preimage of function $f(x) = x^e$ are instances of the form $y = \mathring{y}f(\delta)$ where $\mathring{y}$ is the simulator's challenge. In this way the simulator can embed its challenge into any number of identities, picking random δ for each identity, and yet straight-line simulate the proofs performed on behalf of all these entities in parallel. Thus our main technical contribution is two-fold: First, we formalize the notion of Σ-equivocability and apply it to a compilation from Σ-protocols to straight-line simulatable structured-instance ZKPK (Section 4). Secondly, we construct a multiplicatively homomorphic and Σ-equivocable commitment scheme based on the RSA problem (Section 5). Together, these two parts immediately imply the IBMS and IBAS schemes we present in this paper (Section 6 and Section 7).

3 Identity-Based Multi-/Aggregate Signature Schemes

We define the notion of identity-based multisignature scheme (IBMS) building on the definitions given by [MOR01,BN06,GR06,BN07]. (Due to lack of space, we relegate the extension of our definitions to IBAS schemes to the full version of the paper [BJ10]). Our notion is more flexible than that of [BN07,MOR01,BN06] because we do not require the set of participants' identities as input to the multi-/aggregate signature protocol. The participating players must be aware of each other in the protocol execution, but this is needed only to ensure proper communication, and the participant identities are not required as inputs to the cryptographic protocol. The schemes secure in this setting provide flexibility to applications of multi-/aggregate signatures because sometimes signers might care only about the message they are signing and not about the identities of the cosigners. Otherwise the list of cosigners can always be attached to the message being signed.

Syntax of an IBMS Scheme: We define an identity-based multisignature as $\mathsf{IBMS} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{MSign}, \mathsf{Vrfy})$ where Setup, KeyDer and Vrfy are probabilistic poly-time algorithms, and MSign is a distributed protocol executed by a set of parties *s.t.*

– $(mpk, msk) \leftarrow \mathsf{Setup}(1^\kappa)$, run by a trusted party, on input the security parameter κ, generates master public key mpk and corresponding master secret key msk.

- $sk_{Id} \leftarrow \mathsf{KeyDer}(msk, Id)$, run by a trusted party, on input master secret key msk and an identity $Id \in \{0,1\}^*$ provides a secret key sk_{Id} to the user with identity Id.
- MSign is a multisignature protocol run by a group of players who intend to sign the same message m. Player with identity Id executes this protocol on public inputs mpk and message m and private input sk_{Id} which is his own secret key. The local output of the protocol for every participant is a multisignature denoted σ.
- $\{0,1\} \leftarrow \mathsf{Vrfy}(mpk, m, IdSet, \sigma)$ verifies whether σ is a valid multisignature on message m on behalf of the set of the identities $IdSet$.

In the random oracle model (ROM), KeyDer, MSign and Vrfy procedures additionally have access to a random oracle $\mathsf{H}(\cdot) : \{0,1\} \rightarrow D$, where D depends on the scheme. This set of procedures must satisfy the following *completeness* properties: For any integer n, any message m, and any (mpk, msk) output by $\mathsf{Setup}(1^\kappa)$, if for $i = 1..n$, one obtains $sk_{Id_i} \leftarrow \mathsf{KeyDer}(msk, Id_i)$ and correctly follows MSign on input m using secret keys sk_{Id_i}, then assuming all messages are delivered between players, each player outputs the same string σ which satisfies $\mathsf{Vrfy}(mpk, m, \{Id_1, ..., Id_n\}, \sigma) = 1$.

Security Notion of an IBMS Scheme: We model the security as existential unforgeability under an adaptive chosen message *and* adaptive chosen identity attack: The adversary participates in a game in which it issues a number of key derivation and signature queries. In a key derivation query, the adversary corrupts a player by submitting its identity Id to the key derivation oracle and receiving its secret key sk_{Id}. In a signature query the adversary specifies the message m and the identity Id that it wants to interact with; and the signing oracle performs MSign protocol on message m on behalf of Id. The adversary wins the game if it eventually outputs a message m, a multisignature σ and a set of identities $IdSet$ *s.t.* $\mathsf{Vrfy}(mpk, m, IdSet, \sigma) = 1$ and there exists an identity Id *s.t.*, the adversary never queried the key derivation oracle on Id and never queried the signing oracle on (m, Id). More formally we define the *adversarial advantage* of $\mathcal{A}$ against $\mathsf{IBMS} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{MSign}, \mathsf{Vrfy})$ as a probability that experiment $\mathbf{Exp}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A})$ described in Figure 2 outputs 1 *i.e.*

$$\mathbf{Adv}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A}) = Pr[\mathbf{Exp}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A}) = 1]$$

where the probability goes over the random coins of the adversary and all the randomness used in the experiment. We call an IBMS scheme $(t, \epsilon, n, q_K, q_S)$-secure if $\mathbf{Adv}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A}) \leq \epsilon$ for every adversary $\mathcal{A}$ that runs in time at most t, makes at most q_K key derivation queries and at most q_S signature queries, and produces a forgery on behalf of at most n parties. In the random oracle model we extend this notion to $(t, \epsilon, n, q_K, q_S, q_H)$-security, where $\mathcal{A}$ is additionally restricted to at most q_H hash queries and the probability in the experiment $\mathbf{Exp}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A})$ goes also over random choice of a hash function.

Experiment $\mathbf{Exp}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A})$

- $(mpk, msk) \leftarrow \mathsf{Setup}(1^\kappa)$; $\mathsf{MIdLst} \leftarrow \emptyset$; $\mathsf{CIdLst} \leftarrow \emptyset$;
- Run $\mathcal{A}(mpk)$, and handle $\mathcal{A}$'s key derivation and signature queries as follows:
- On a key derivation query on identity Id, add Id to CIdLst, run KeyDer on input (msk, Id) and return sk_{Id} to $\mathcal{A}$.
- On a signing query on pair (m, Id), add (m, Id) to MIdLst, run MSign protocol on behalf of identity Id on message m forwarding messages to and from $\mathcal{A}$.
- When $\mathcal{A}$ halts, parse its output as $(m, IdSet, \sigma)$.
- If $(\mathsf{Vrfy}(mpk, m, IdSet, \sigma){=}1) \wedge (\exists\, Id{\in}IdSet\ s.t.\ (Id{\notin}\mathsf{CIdLst}) \wedge ((m, Id){\notin}\mathsf{MIdLst}))$ then return 1, otherwise return 0.

Fig. 2. Chosen Message Attack against an Identity-Based Multisignature Scheme

4 Σ-Equivocable Commitments and Structured-Instance Zero-Knowledge

Homomorphic Σ-Protocols: Σ-protocol, a notion introduced by Cramer, Damgard and Schoenmakers [CDS94], is a three-move proof system with *special* honest-verifier zero-knowledge (HVZK) and strong soundness properties. Let $R = \{(x, y)\}$ be a relation whose membership can be verified in polynomial time. We consider a special case where X and Y are algebraic groups (for notational simplicity we use multiplicative notation for both), and $R = \{(x, f(x)) \mid x \in X\}$ where $f : X \to Y$ is a homomorphic one-way function. We consider a proof of knowledge system for relation R which we call *homomorphic Σ-protocol* (for R): The prover, on input $x \in X$, sends $a = f(k)$ where $k \overset{r}{\leftarrow} X$. The verifier, on input $y \in Y$, creates a challenge c as a random κ-bit string, and the prover responds with $z = kx^c$. The verifier accepts iff $f(z) = ay^c$. This is a form of several Σ-protocols for known homomorphic one-way functions, *e.g.* Guillou-Quisquater identification scheme [GQ88] for a power function $f_{e,n}(x) = x^e \bmod n$ and Schnorr's scheme [Sch89] for exponentiation $f_{g,p}(x) = g^x \bmod p$. The special HVZK property of a Σ-protocol says that there exists an efficient simulator which on input y computes pair (a, z) for any c with the distribution matching that of the prover. The special strong soundness says that there exists an efficient extractor which computes witness x *s.t.* $(x, y) \in R$ for any y from any pair of accepting conversations (a, c, z) and (a, c', z') *s.t.* $c \neq c'$.

Structured-Instance Zero-Knowledge: Multi-instance zero-knowledge (ZK) (*a.k.a.* multi-theorem ZK) in common reference string (CRS) model requires a two-phase probabilistic poly-time simulator *s.t.* (1) in the first phase, given public parameters, the simulator outputs the CRS string together with some trapdoor information; (2) In the second phase, given a statement and the trapdoor, simulator outputs the simulated proof for that statement. In the single-instance ZK, the simulator knows the statement beforehand and can set the CRS string as a function of this particular statement. Structured instance zero-knowledge proof for relation R introduced above is an intermediary notion: The simulator is given a "core statement" $\mathring{y} \in Y$ before it sets the CRS string, and then it can simulate the proof for statement $y = \mathring{y} \cdot f(\delta)$ for any $\delta \in X$. Here is the formal definition:

Definition 1. *Let X and Y be algebraic groups and $f : X \to Y$ be a surjective homomorphic one-way function, all indexed by a public parameter* par. *Let $\Pi = (\mathcal{G}, \mathcal{P}, \mathcal{V})$ be a proof system in CRS model for relation $R = \{(x, y) \in X \times Y | \ y = f(x)\}$ where $\mathcal{G}$ is an algorithm that outputs the common reference string. We say that Π is* straight-line ϵ-structured-instance zero-knowledge *if there exist efficient algorithms $\mathcal{S}_1, \mathcal{S}_2$* s.t. *$\mathcal{S}_1$ on input* par *and a core instance $\mathring{y} \in Y$, outputs the CRS string σ and trapdoor td, while $\mathcal{S}_2$ on input td and a "witness-shift" $\delta \in X$ outputs a simulated proof $\tilde{\pi}$ for instance $y = \mathring{y} f(\delta)$, and for all $(\mathring{x}, \mathring{y}) \in X \times Y$* s.t. *$f(\mathring{x}) = \mathring{y}$ the following two properties hold:*

1. *Statistical difference between the following two distributions is at most ϵ:*

$$\{\sigma \mid (\sigma, td) \leftarrow \mathcal{S}_1(\mathsf{par}, \mathring{y})\}$$
$$\{\sigma \mid \sigma \leftarrow \mathcal{G}(\mathsf{par})\}$$

2. *$\forall$ verifier $\mathcal{V}^*$ and $\forall \delta \in X$, the following two distributions are identical:*

$$\{\tilde{\pi} \mid \tilde{\pi} \leftarrow \mathcal{V}^*(y, \sigma)^{\mathcal{S}_2(td, \delta, \sigma)}; (td, \sigma) \leftarrow \mathcal{S}_1(\mathsf{par}, \mathring{y}); y \leftarrow \mathring{y} f(\delta)\}$$
$$\{\pi \mid \pi \leftarrow \mathcal{V}^*(y, \sigma)^{\mathcal{P}(x, y, \sigma)}; \sigma \leftarrow \mathcal{G}(\mathsf{par}); y \leftarrow \mathring{y} f(\delta); x \leftarrow \mathring{x}\delta\}$$

Commitment Schemes: A commitment scheme $\mathcal{C}$ in the CRS model consists of probabilistic poly-time algorithms CSetup, CKG, Com and Open. CSetup on input the security parameter κ, generates public parameters cpar, which also determine the commitment message space $\mathcal{M}$. $\mathsf{CKG}(\mathsf{cpar})$ generates the commitment key K, $\mathsf{Com}_K(m)$ generates the commitment $\mathcal{C}$ and the decommitment D on message $m \in \mathcal{M}$, and finally $\mathsf{Open}_K(\mathcal{C}, D, m)$ determines if D is a valid decommitment of commitment $\mathcal{C}$ to message m. A commitment scheme must satisfy that if $\mathsf{cpar} \leftarrow \mathsf{CSetup}(1^\kappa)$, $K \leftarrow \mathsf{CKG}(\mathsf{cpar})$, and $(\mathcal{C}, D) \leftarrow \mathsf{Com}_K(m)$, then $\mathsf{Open}_K(\mathcal{C}, D, m) = 1$. Below we define statistical hiding and computational binding properties of commitments because these will be variants of these notions which our scheme satisfies.

ϵ-Hiding: For all $\mathsf{cpar} \leftarrow \mathsf{CSetup}(1^\kappa)$, $m_0, m_1 \in \mathcal{M}$, and $K \leftarrow \mathsf{CKG}(\mathsf{cpar})$, there is less than ϵ statistical difference between the distribution of $\mathcal{C}$'s output by $\mathsf{Com}_K(m_0)$ and the distribution of $\mathcal{C}$'s output by $\mathsf{Com}_K(m_1)$. A commitment scheme is *perfectly* hiding if $\epsilon = 0$.

(t, ϵ)-Binding: For any algorithm $\mathcal{A}$ running in time t and any cpar output by $\mathsf{CSetup}(1^\kappa)$, the probability of $\mathsf{Open}_K(\mathcal{C}, D_0, m_0) = \mathsf{Open}_K(\mathcal{C}, D_1, m_1) = 1$ and $m_0 \neq m_1$ is less than ϵ where $(\mathcal{C}, D_0, D_1, m_0, m_1)$ is outputted by $\mathcal{A}$ on input K and $K \leftarrow \mathsf{CKG}(\mathsf{cpar})$ and probability is over the coins of CKG and $\mathcal{A}$.

Notation: In this paper we only deal with the commitment schemes in which the commitment is a deterministic function of the message and the decommitment. Therefore we assume there exist a decommitment space denoted as $\mathcal{R}$ and the Com procedure picks decommitment $D \overset{r}{\leftarrow} \mathcal{R}$ and computes the commitment $\mathcal{C}$ as the deterministic function of m and D.

Σ-Equivocable Commitments: A commitment scheme is *equivocable* if there exists an efficient simulator that generates commitment key K, indistinguishable from real key, together with a *trapdoor td*. The trapdoor allows simulator to create fake commitments indistinguishable from real ones, and later decommit them to *any* message. Using equivocable commitments, one can compile a Σ-protocol to a multi-instance ZK proof system with straight-line simulation [Dam00]. Here we define a rather restrictive form of equivocability called Σ-equivocability and we show that it is sufficient for compiling Σ-protocols into structured-instance ZK proofs with straight-line simulation. It turns out that structured-instance ZK is sufficient for our application of ZK proofs to multi-/aggregate signatures and multi-instance ZK is not required. Moreover the straight-line simulatability of this system allows us to have multi-/aggregate schemes with concurrency, better exact security and with improved round complexity.

Definition 2. *Let X and Y be algebraic groups and let $f : X \to Y$ be a homomorphic one-way function, all indexed by a commitment parameter* cpar*. We call a commitment scheme* ϵ-Σ-equivocable *for f if there exist probabilistic polytime algorithms* tdCKG*,* tdCom*, and* $\mathsf{RstEquiv}$*, where* $(K, td) \leftarrow \mathsf{tdCKG}(\mathsf{cpar}, \mathring{y})$*,* $(\tilde{\mathcal{C}}, st) \leftarrow \mathsf{tdCom}_K(td)$*, and* $(\tilde{D}, z) \leftarrow \mathsf{RstEquiv}_K(td, st, c, \delta)$*,* s.t. *for any* cpar *output by* CSetup *and any $\mathring{y} \in Y$ the following properties hold:*

1. *There is at most ϵ statistical difference between the distribution of K's output by* $\mathsf{CKG}(\mathsf{cpar})$ *and K's output by* $\mathsf{tdCKG}(\mathsf{cpar}, \mathring{y})$*.*
2. *For all* $(K, td) \leftarrow \mathsf{tdCKG}(\mathsf{cpar}, \mathring{y})$*, $\delta \in X$, and $c \in \{0,1\}^\kappa$, if $(\tilde{\mathcal{C}}, st)$ is output by* $\mathsf{tdCom}_K(td)$ *and $(\tilde{D}, z)$ is output by* $\mathsf{RstEquiv}_K(td, st, c, \delta)$ *then $\tilde{D}$ is distributed as random decommitment in $\mathcal{R}$ and* $\mathsf{Open}_K(\tilde{\mathcal{C}}, \tilde{D}, f(z)(\mathring{y} f(\delta))^{-c}) = 1$*.*

Intuitively definition 2 says that the equivocation procedure, given $(\mathring{y}, c, \delta)$, can open a fake commitment to a message of the form $a = f(z)(\mathring{y} f(\delta))^{-c}$ for some z. This is useful in straight-line simulation of a proof of knowledge for relation $R = \{(x, y) \in X \times Y \mid y = f(x)\}$. For example, let $f : \mathrm{QR}_n \to \mathrm{QR}_n$ where $f(z) = z^e \pmod{n}$. Consider the HVZK simulator of the Σ-protocol for proving knowledge of e-th root: This simulator picks random c and z and computes prover's first message $a = z^e y^{-c}$. Below we show that Damgard's compilation [Dam00] (see Figure 3 below) transforms such Σ-protocol to structured-instance zero-knowledge using only such Σ-equivocable commitments, because the simulator can output a fake commitment and then open it to what the Σ-protocol simulator would output as the prover's first message *i.e.* $a = z^e y^{-c}$. Definition 2 implies that a fake commitment can be opened to $a = z^e(\mathring{y}\delta^e)^{-c}$ for any δ and c. Hence the structured-instance zero-knowledge simulator can use this property to simulate a proof for any instance $y = \mathring{y}\delta^e$ where $\mathring{y}$ is set before the simulator creates the CRS string (see theorem 1).

Homomorphic Commitments: We call a commitment scheme multiplicatively homomorphic if there are efficiently computable operations $\otimes$ and $\oplus$ *s.t.* if $\mathsf{Open}_K(\mathcal{C}_1, D_1, m_1) = 1$ and $\mathsf{Open}_K(\mathcal{C}_2, D_2, m_2) = 1$, then $\mathsf{Open}_K(\mathcal{C}, D, m) = 1$

for $\mathcal{C} = \mathcal{C}_1 \otimes \mathcal{C}_2$, $D = D_1 \oplus D_2$, and $m = m_1 m_2$. Accordingly, a commitment scheme is l-restricted multiplicatively homomorphic if the homomorphic operation can be applied on only l commitment-decommitment pairs generated by Com procedure. Our construction is l-restricted multiplicatively homomorphic.

Common Reference String:
Commitment Key K of Σ-Equivocable Commitment Scheme

$\underline{\text{Prover } P(x)}$ $s.t.\ x \in X,\ f(x) = y$		$\underline{\text{Verifier } V(y)}$ $s.t.\ y \in Y$
$k \overset{r}{\leftarrow} X,\ a \leftarrow f(k)$		
$(\mathcal{C}, D) \leftarrow \mathsf{Com}_K(a)$	$\xrightarrow{\mathcal{C}}$	
	$\xleftarrow{c}$	$c \overset{r}{\leftarrow} \{0,1\}^{\kappa}$
$z \leftarrow kx^c$	$\xrightarrow{z\ ,\ D}$	acc iff $\mathsf{Open}_K(\mathcal{C}, D, f(z)y^{-c}) = 1$

Fig. 3. Straight-line simulatable structured-instance ZKPK of pre-image of f

Structured-Instance Zero-Knowledge from Homomorphic Σ-Protocol: Figure 3 shows a construction of a straight-line simulatable structured-instance zero-knowledge proof of knowledge system, in the CRS model, from homomorphic Σ-protocol and Σ-equivocable commitment. This is an identical construction to Damgard's compiler from Σ-protocol to ZKPK proof [Dam00]. Below we show that using only Σ-equivocable commitments the same compilation produces *structured-instance* zero-knowledge proof given *homomorphic* Σ-protocol. As in [Dam00] the resulting protocol is an argument of knowledge, subject to the binding property of the commitment scheme.

Theorem 1. *Let X and Y be algebraic groups, $f : X \leftarrow Y$ a homomorphic one-way function, $\mathcal{C}$ a Σ-equivocable commitment over message space $\mathcal{M} \subseteq Y$. Then the protocol in figure 3 is a straight-line simulatable structured-instance zero-knowledge proof of knowledge of pre-image of f in the CRS model.*

Proof. The straight-line simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, for structured-instance zero-knowledge proof acts as follows: In the first phase, given cpar and $\mathring{y} \in Y$, $\mathcal{S}_1$ runs $\mathsf{tdCKG}(\mathsf{cpar}, \mathring{y})$ to obtain (td, K) and sets the common reference string σ as K. In the second phase, given td and witness shift $\delta \in X$, $\mathcal{S}_2$ runs $\mathsf{tdCom}_K(td)$ to obtain the fake commitment $\tilde{\mathcal{C}}$ and state st and sends $\tilde{\mathcal{C}}$ to the verifier. Upon receiving the challenge c from the verifier, $\mathcal{S}_2$ runs $\mathsf{RstEquiv}_K(td, st, \mathring{y}, \delta)$ to get the response z and fake commitment $\tilde{D}$. According to Σ-equivocability property (definition 2) it immediately follows that $\mathcal{S}$ satisfies conditions in definition 4.

5 Aggregatable Zero-Knowledge Proof of Knowledge of e-th Root

Safe RSA Assumption: Since our construction relies on two *related* instances of RSA cryptosystems which share same RSA modulus n but use two different public exponents e and e', it is convenient for us to use the following notation for RSA instance generation: We call an algorithm $\mathsf{KG_{sRSA}}$ a safe RSA generator if on input security parameter κ and a prime e *s.t.* $2^\kappa \leq e \leq 2^{2\kappa}$, $\mathsf{KG_{sRSA}}$ generates a pair (n, d) where (1) $n = pq$ *s.t.* $p = 2p' + 1$, $q = 2q' + 1$ and p, q, p' and q' are all prime numbers *s.t.* $|p'| = |q'|$ and $p', q' > 2^{2\kappa}$ and (2) $d = e^{-1} \bmod \phi(n)$. For later use we define $n' = p'q'$. The advantage of an algorithm $\mathcal{A}$ in breaking the RSA(e) problem is defined as

$$\mathbf{Adv}^{\mathsf{ow\_RSA}}_{\mathsf{KG_{sRSA}},\mathcal{A},\mathsf{e}}(\kappa) = \Pr[x^e \overset{n}{\equiv} y \mid (n, d) \overset{r}{\leftarrow} \mathsf{KG_{sRSA}}(\kappa, e); y \overset{r}{\leftarrow} \mathbb{Z}_n^*; x \overset{r}{\leftarrow} \mathcal{A}(n, e, y)] \quad (1)$$

We say algorithm $\mathcal{A}$, (t, ϵ)-breaks the RSA(e) problem on security parameter κ if $\mathcal{A}$ runs in time at most t and $\mathbf{Adv}^{\mathsf{ow\_RSA}}_{\mathsf{KG_{sRSA}},\mathcal{A},\mathsf{e}}(\kappa) \geq \epsilon$. We say that the RSA($e$) problem is (t, ϵ)-hard (for security parameter κ) if no algorithm $\mathcal{A}$, (t, ϵ)-breaks it. We note that the requirement that $p', q' > 2^{2\kappa}$ is just a lower-bound we introduce to enable any party to choose "secondary" public exponent e' *s.t.* $\gcd(e', \phi(n)) = 1$ and $e' > le$ where l is a maximum number of participants in any single instance of the multi-signature scheme.

5.1 RSA-based Multiplicatively Homomorphic Σ-Equivocable Commitment

Let e and e' be two prime numbers *s.t.* $2^\kappa \leq e, e' \leq 2^{2\kappa}$ and $e \leq e'/l$ for some integer l and let (n, d) be output by $\mathsf{KG_{sRSA}}(\kappa, e)$. This assures that both (n, e) and (n, e') are safe RSA instances. We describe an efficient commitment scheme, which is computationally binding under the RSA(e') assumption, has l-restricted multiplicatively homomorphic property on message space $\mathcal{M} = \mathrm{QR}_n$, and is Σ-equivocable for $f(x) = x^e \pmod n$. Curiously, this commitment is statistically hiding only for the messages picked from a specific subset of the message space, but in our application of this commitment scheme to straight line simulatable ZKPK of e-th root, standard hiding property is not necessary, and Σ-equivocability property for the above function is sufficient.

- $\mathsf{CSetup}(\kappa)$: Pick prime numbers e and e' *s.t.* $2^\kappa \leq e, e' \leq 2^{2\kappa}$ and $e \leq e'/l$. Run $\mathsf{KG_{sRSA}}$ on input (κ, e) to obtain (n, d). Set $\mathsf{cpar} \leftarrow (n, e, e')$.
- $\mathsf{CKG}(n, e, e')$: Pick $h \overset{r}{\leftarrow} \mathrm{QR}_n$ and set $K \leftarrow (n, e, e', h)$. Note that it is easy to sample random elements in QR_n by squaring a random element in $\mathbb{Z}_n^*$.
- $\mathsf{Com}_K(m)$: Pick $r \overset{r}{\leftarrow} \mathbb{Z}_e$ and set $\mathcal{C} \leftarrow h^r m^{e'}$ and $D \leftarrow r$. (Hence the decommitment space is $\mathbb{Z}_e$.)
- $\mathsf{Open}_K(\mathcal{C}, r, m)$: Accept iff $\mathcal{C} = h^r m^{e'}$ and $0 \leq r < e'$.
- $\mathsf{tdCKG}((n, e, e'), \mathring{y})$: Pick $\gamma \overset{r}{\leftarrow} [n]$, and set $h \leftarrow (\mathring{y})^{\gamma e'}$, $K \leftarrow (n, e, e', h)$, and $td \leftarrow (\gamma, \mathring{y})$.
- $\mathsf{tdCom}_K(td)$: Pick $s \overset{r}{\leftarrow} \mathbb{Z}_e$ and return $(\tilde{\mathcal{C}}, st)$ where $\tilde{\mathcal{C}} = (\mathring{y})^{e's}$ and $st = s$.

– $\mathsf{RstEquiv}_K(td, st, c, \delta)$: Compute $r = (s + c)\gamma^{-1} \pmod e$ and $i = (s + c - \gamma r)/e$ (over integers) and return (r, z) where $z = (\mathring{y})^i(\delta)^c$.

Statistical Hiding: This commitment scheme is ϵ-hiding for the messages picked from $\tilde{\mathcal{M}} \subset \mathrm{QR}_n$ where $\tilde{\mathcal{M}} = \{h^{i(e')^{-1}} | i \in [\epsilon e/2]\}$ and h is determined by the commitment key. To argue this note that the maximum statistical difference between the distributions of the commitments to $m_0, m_1 \in \tilde{\mathcal{M}}$ happens when they correspond to $i = 0$ and $i = \epsilon e/2$ respectively. This way the distributions of the commitments would be $\{h^r\}_{r \xleftarrow{r} [e]}$ and $\{h^{r+\epsilon e/2}\}_{r \xleftarrow{r} [e]}$ respectively which has a statistical difference equal to ϵ.

Computational Binding: This commitment scheme is (t, ϵ)-binding if $\mathrm{RSA}(e')$ problem is (t, ϵ)-hard. Indeed given the challenge (n, e', h), one can use the attacker on binding to find the e'-th root of h. The reduction runs the binding attacker to obtain $(\mathcal{C}, r, m, r', m')$ *s.t.* $\mathsf{Open}_K(\mathcal{C}, r, m) = \mathsf{Open}_K(\mathcal{C}, r', m') = 1$ and $m \neq m'$. Since $\mathcal{C} = h^r m^{e'} = h^{r'} m'^{e'}$ it follows that $h^{r-r'} = (m'/m)^{e'}$. Now since $r, r' < e'$, then $\gcd(e', r-r') = 1$ and using extended Euclidian algorithm one can compute α, β *s.t.* $\alpha(r - r') + \beta e' = 1$. Thus $h = h^{\alpha(r-r')+\beta e'} = ((m'/m)^\alpha h^\beta)^{e'}$ and e'-th root of h can be computed as $(m'/m)^\alpha h^\beta$.

l-Restricted Multiplicative Homomorphism: This commitment scheme is multiplicatively homomorphic on QR_n in the sense that up to $l \leq \lfloor e'/e \rfloor$ messages can be combined: If $\{(\mathcal{C}_i, r_i)\}_{i=1..l}$ are commitment-decommitment pairs for messages $m_1, ..., m_l \in \mathrm{QR}_n$ each computed by the commitment procedure, then $r = \sum_{i=1}^{l} r_i$(over integers) is a valid decommitment for commitment $\mathcal{C} = \prod_{i=1}^{l} \mathcal{C}_i$ for message $m = \prod_{i=1}^{l} m_i$. Note that by setting $e' \geq e2^\kappa$, homomorphism can be used on any feasible set of messages.

Σ-Equivocability: This commitment scheme is $2^{-2\kappa}$-Σ-equivocable for function (family) $f_{(n,e)}(x) = x^e (\mathrm{mod}\ n)$. First note that for every (n, e, e') output by CSetup and every $\mathring{y} \in \mathrm{QR}_n$ *s.t.* $\mathring{y}$ is a generator of QR_n, the distributions of keys generated by $\mathsf{CKG}(n, e, e')$ and $\mathsf{tdCKG}((n, e, e'), \mathring{y})$ are at most $2^{-2\kappa}$ apart, because CKG chooses the key h as a random element in QR_n while tdCKG picks $h = (\mathring{y})^{e'\gamma}$ for e' *s.t.* $\gcd(e', \phi(n)) = 1$ and γ chosen at random in $[n]$. Moreover the statistical difference between $[n]$ and $[4n']$ is equal to $1 - 4n'/n < 2^{2\kappa}$. Secondly, if $\mathring{y}$ is a generator of QR_n then for every $\gamma \in [n]$, every $\delta \in \mathrm{QR}_n$ and every $c \in \{0, 1\}^\kappa$, according to the code of tdCom and $\mathsf{RstEquiv}$, r, z satisfy $s+c = \gamma r+ie$ and $z = (\mathring{y})^i(\delta)^c$, therefore for $m = z^e(\mathring{y}(\delta)^e)^{-c}$ we have $\tilde{\mathcal{C}} = h^r m^{e'}$, and hence $\mathsf{Open}(\tilde{\mathcal{C}}, r, m) = 1$. Moreover the distribution of the decommitments in the equivocation process *i.e.* $\{\tilde{r} | s \xleftarrow{r} \mathbb{Z}_e; \tilde{r} \leftarrow (s + c)\gamma^{-1} (\mathrm{mod}\ e)\}$ is identical to uniform distribution over $\mathbb{Z}_e$.

Corollary 1. *Consider prime number* $2^\kappa \leq e \leq 2^{2\kappa}$ *and let* n *be a safe* RSA *modulus output by* $\mathsf{KG_{sRSA}}$ *on input* e *and security parameter* κ*. Consider compilation shown in figure 3 and let the function (family)* f *be* $f_{(n,e)} : \mathrm{QR}_n \to \mathrm{QR}_n$ s.t. $f_{(n,e)}(x) = x^e \pmod n$ *and let the compilation be instantiated with the commitment scheme described in this section. Then from theorem 1, it immediately follows that the resulting scheme is a straight-line structured-instance zero-knowledge proof of knowledge of* e*-th root.*

6 Identity-Based Multisignature Scheme Based on RSA

We describe our IBMS scheme based on the RSA assumption. The scheme takes two communication rounds, requires two double-exponentiations per party for signing and one triple-exponentiation for verification. The scheme is based on the GQ ID-based identification protocol [GQ88], which is the Σ-protocol for proving knowledge of e-th root. Each party simply executes the aggregatable zero-knowledge proof of e-th root of its (hashed) identity string, using the straight-line simulatable aggregatable ZKPK of e-th root described in Section 5. Figure 4 contains the Setup, KeyDer, MSign and Vrfy algorithms for this IBMS scheme.

Note on multi-signature length: In Figure 4 the final multi-signature is a tuple $(z, \mathcal{C}, D)$ where $z \in \mathbb{Z}_n^*$ and $(\mathcal{C}, D) \in \mathbb{Z}_n^* \times \mathbb{Z}_e$ is a commitment-decommitment pair on message $a = z^e(\mathring{y})^{-c}$. However this commitment can be computed as a deterministic function of the committed message a and the decommitment D (see Section 4). Therefore $\mathcal{C}$ can be computed given (z, c, D), and hence one can use (z, c, D) as the final multi-signature, which reduces the multi-signature size to $|\mathbb{Z}_n^*| + |\mathbb{Z}_e| + \kappa < |n| + 2\kappa + \log l$.

Theorem 2. *If RSA(e) and RSA(e') problems are (t', ϵ')-hard, and the IBMS scheme in figure 4 is instantiated with commitment scheme in section 5, which is (t_B, ϵ_B)-binding and ϵ_E-Σ-equivocable for function $f_{(n,e)}(x) = x^e (\mathrm{mod}\ n)$, then the resulting IBMS scheme is $(t, \epsilon, n, q_k, q_s, q_h)$-secure in random oracle model where*

$$t \geq \frac{1}{2}\min(t', t_B) - (3q_s + q_h)t_{exp}$$

$$\epsilon \leq 4q_k\sqrt{(\epsilon' + \epsilon_B + \epsilon_E)q_h + \left(\frac{q_h}{2^{\kappa+1}}\right)^2} + \frac{q_k q_h}{2^{\kappa-1}} + \epsilon_E$$

and t_{exp} is the time of one exponentiation in $\mathbb{Z}_n^*$.

Proof. Let $\mathcal{C} = (\mathsf{CKG}, \mathsf{Com}, \mathsf{Open}, \mathsf{tdCKG}, \mathsf{tdCom}, \mathsf{RstEquiv})$ be a commitment scheme for public parameters $\mathsf{cpar} = (n, e, e')$ and the message space $\mathcal{M}$ equal to QR_n. Assume $\mathcal{C}$ is l-restricted multiplicatively homomorphic, (t_B, ϵ_B)-binding and ϵ_E-Σ-equivocable for $f_{(e,n)}(x) = x^e \pmod n$. Given a $(t, \epsilon, n, q_k, q_s, q_h)$-forger $\mathcal{F}$, consider two simulators $\mathcal{B}_0$ and $\mathcal{B}_1$ that simulate the role of the honest player as in the experiment $\mathbf{Exp}_{\mathsf{IBMS}}^{\mathsf{uu-cma}}$ interacting with the forger $\mathcal{F}$. $\mathcal{B}_0$ takes as an input a set $\{c_1, c_2, ..., c_{q_h}\}$ where c_i's are in $\{0,1\}^\kappa$ and runs Setup procedure to obtain (mpk, msk) and follows the real protocol *i.e.* answers to forger's key derivation queries and signing queries using procedures KeyDer and MSign respectively. Additionally, $\mathcal{B}_0$ answers the forger's hash queries and performs an extra finalization process by following the procedures $\mathsf{SimHash}$ and $\mathsf{Finalize}$ in Figure 5. The simulator $\mathcal{B}_1$, on the other hand, takes as an input an RSA challenge $(n, e, \mathring{y})$ and a set $\{c_1, c_2, ..., c_{q_h}\}$ where c_i's are in $\{0,1\}^\kappa$ and follows the Init, $\mathsf{SimKeyDer}$, $\mathsf{SimMSign}$, $\mathsf{SimHash}$ and $\mathsf{Finalize}$ procedures detailed in Figure 5 to perform the initialization, answering to key derivation, signing and hash

1. **Setup**(1^κ):

   Let l be the maximum number of players in the IBMS scheme. Pick prime numbers e and e' *s.t.* $2^\kappa \le e, e' \le 2^{2\kappa}$ and $2^{\kappa+1}l < e < e'/l$. Run $\mathsf{KG_{sRSA}}$ on input (κ, e) to obtain (n, d). Note that $\gcd(e', \phi(n)) = 1$ because $\phi(n) = 4p'q'$ where $p', q' > 2^{2\kappa}$. Run $\mathsf{CKG}(n, e, e')$ to obtain the commitment key K. Set $mpk = (n, e, e', K)$ and $msk = d$. Assume $\mathcal{H}_1 : \{0,1\}^* \to \mathrm{QR}_n$ and $\mathcal{H}_2 : \mathrm{QR}_n \times \{0,1\}^* \times \mathrm{QR}_n \to \{0,1\}^\kappa$ are random oracles that every other algorithm in the protocol has access to them.

2. **KeyDer**(msk, Id):

   The PKG computes $x_{Id} \leftarrow (\mathcal{H}_1(Id))^{2d} (\mathrm{mod}\ n)$, sets the private key of the user with identity Id as $sk_{Id} \leftarrow x_{Id}$ and sends it back to him via a secure and authenticated channel.

3. **MSign**: Let $\mathcal{P}$ be the set of players participating in the protocol. Each player determines $\mathcal{P}$ after the first step of MSign. Player with identity Id_i on input (mpk, m, sk_{Id_i}), performs the following steps:

   3.1 Pick $k_i \overset{r}{\leftarrow} \mathrm{QR}_n$, $a_i \leftarrow k_i^e$; Set $(\mathcal{C}_i, D_i) \overset{r}{\leftarrow} \mathsf{Com}_K(a_i)$ and broadcast $(Id_i, \mathcal{C}_i)$;
   3.2 Upon receiving $(Id_j, \mathcal{C}_j)\ \forall P_j \in \mathcal{P}$, Set $IdSet \leftarrow \{Id_j\}_{P_j \in \mathcal{P}}$ and $\mathcal{C} \leftarrow \bigotimes_{P_j \in \mathcal{P}} \mathcal{C}_j$;
   Set $c \leftarrow \mathcal{H}_2(\mathcal{C}, IdSet, m)$; Compute $z_i \leftarrow k_i (x_{Id_i})^c$ and broadcast (z_i, D_i);
   3.3 Output multisignature $\sigma = (z, \mathcal{C}, D)$, where $z = \prod_{P_j \in \mathcal{P}} z_j$ and $D = \bigoplus_{P_j \in \mathcal{P}} D_j$.

4. **Vrfy**$(mpk, m, IdSet, \sigma)$:

   Parse σ as $(z, \mathcal{C}, D)$ and mpk as (n, e, e', K); Set $c \leftarrow \mathcal{H}_2(\mathcal{C}, IdSet, m)$;
   $y \leftarrow \prod_{Id_i \in IdSet} \mathcal{H}_1(Id_i)^2$; If $\mathsf{Open}_K(\mathcal{C}, D, z^e y^{-c}) = 1$ then accept otherwise reject.

Fig. 4. Identity-based multisignature scheme based on RSA

queries and finalization processes, respectively. Intuitively, the simulator $\mathcal{B}_1$ uses Coron's technique [Cor00] to embed the RSA challenge in the hashes of the ID's of the players with some biased probability $1 - \rho$ hoping that the forgery be based upon the ID of the player for which the RSA challenge is indeed embedded. This way $\mathcal{B}_1$ passes the signing queries on behalf of identity Id just like real protocol using the procedure MSign if the RSA challenge is not embedded in the hash of Id and otherwise $\mathcal{B}_1$ uses the straight-line structured-instance zero-knowledge simulator for proof of knowledge of e-th root (see corollary 1) to simulate the signature protocol on behalf of the identity Id. Both $\mathcal{B}_0$ and $\mathcal{B}_1$, after receiving a valid forgery from $\mathcal{F}$, perform a finalization phase in which the forged multisignature is returned together with the index of the hash responses upon which they are based. Namely both $\mathcal{B}_0$ and $\mathcal{B}_1$ return $(j, (m, IdSet, \sigma))$ *s.t.* $\mathsf{Vrfy}(mpk, m, IdSet, \sigma) = 1$ and there exists at least one uncorrupted Id *s.t.* (m, Id) is never queried for signing. The simulators $\mathcal{B}_0$ and $\mathcal{B}_1$ set up empty tables H_1 and H_2 to simulate the hash functions $\mathcal{H}_1$ and $\mathcal{H}_2$ respectively and use the set $\{c_1, c_2, ..., c_{q_h}\}$ to answer to the hash queries to $\mathcal{H}_2$ which enables the utilization of forking lemma (as formulated *e.g.* in [BN06,BCJ08]).

Now for $I \in \{0,1\}$ let's lower-bound $acc_{\mathcal{B}_I}$ the probability that $\mathcal{B}_I$ generates a "useful" output *i.e.* an output other than $(0, \lambda)$. This happens when $\mathcal{B}_I$ does not abort in any of the key derivation queries or finalization procedure. Therefore $acc_{\mathcal{B}_I} \geq \rho^{q_K}(1-\rho)$. This function reaches its maximum when $\rho = q_K/(q_K+1)$. Substituting this value of ρ yields:

$$acc_{\mathcal{B}_I} \geq \left(\frac{q_K}{q_K+1}\right)^{q_K}\left(1-\frac{q_K}{q_K+1}\right) \geq \frac{1}{q_K}\left(\frac{q_K}{q_K+1}\right)^{q_K+1} \geq \frac{1}{4q_K}$$

For $I \in \{0,1\}$, consider $\mathcal{F}_{\mathcal{B}_I}$-the forking algorithm associated with $\mathcal{B}_I$. The success event of $\mathcal{F}_{\mathcal{B}_I}$ denoted by $E^{\mathcal{B}_I}$ is that the algorithm $\mathcal{B}_I$ outputs two tuples $(c_j,(x,n_1,m,IdSet,\sigma))$ and $(\tilde{c_j},(\tilde{x},\tilde{n_1},\tilde{m},Id\tilde{S}et,\tilde{\sigma}))$ *s.t.* $c_j \neq \tilde{c}_j$ where j is the index of the hash responses upon which the forged multisignature is based. Since the random coins of the algorithm $\mathcal{B}_I$ and the hash responses of the algorithm $\mathcal{B}_I$ previous to j^{th} query are the same in the first and second executions, all the computations and communications and in particular the queries submitted to the hash function $\mathcal{H}_2$ before j^{th} query must be the same, too. Thus the occurrence of $E^{\mathcal{B}_I}$ implies $IdSet = Id\tilde{S}et$, $\mathcal{C} = \tilde{\mathcal{C}}$ and $m = \tilde{m}$. Note that $IdSet = Id\tilde{S}et$ also implies $y = \tilde{y}$. This is because $y = \prod_{Id_i \in IdSet}(\mathcal{H}_1(Id_i))^2$, $\tilde{y} = \prod_{Id_i \in Id\tilde{S}et}(\mathcal{H}_1(Id_i))^2$ and the values for $\mathcal{H}_1(Id_i)$ for all $Id_i \in IdSet$ is fixed before the fork. The success event $E^{\mathcal{B}_I}$ can be partitioned into two cases (1) event $E_1^{\mathcal{B}_I}$ in which $E^{\mathcal{B}_I}$ happens and $z^e y^{-c} = \tilde{z}^e\tilde{y}^{-\tilde{c}}$ (2) event $E_2^{\mathcal{B}_I}$ in which $E^{\mathcal{B}_I}$ happens and $z^e y^{-c} \neq \tilde{z}^e\tilde{y}^{-\tilde{c}}$. Obviously $E^{\mathcal{B}_I} = E_1^{\mathcal{B}_I} \cup E_2^{\mathcal{B}_I}$ and hence $\Pr[E^{\mathcal{B}_I}] \leq \Pr[E_1^{\mathcal{B}_I}] + \Pr[E_2^{\mathcal{B}_I}]$. On the other hand, according to the forking lemma, $E^{\mathcal{B}_I}$ can be lower bounded by $\epsilon_{\mathcal{B}_I}$, the success probability of the simulator $\mathcal{B}_I$:

$$acc_{\mathcal{B}_I}\cdot\left(\frac{acc_{\mathcal{B}_I}}{q_h} - \frac{1}{2^\kappa}\right) \leq \Pr[E^{\mathcal{B}_I}] \leq \Pr[E_1^{\mathcal{B}_I}] + \Pr[E_2^{\mathcal{B}_I}] \tag{2}$$

If c_i's are uniformly distributed in $\{0,1\}^\kappa$ then $\mathcal{F}$'s view in interaction with $\mathcal{B}_0$ is identical to the real execution of the protocol. As for $\mathcal{B}_1$, since $\mathcal{C}$ is ϵ_E-Σ-equivocable, by straight-line structured-instance simulatability of ZKPK of e-th root, firstly the distributions of the commitment keys in the simulation and in the real protocol are at most ϵ_E apart and secondly the distribution of the tuples $(\mathcal{C}_1, D_1, z_1)$ generated in each signature instance in the interaction between $\mathcal{F}$ and $\mathcal{B}_1$ is identical the distributions of the same variables in the real execution. Thus, since our simulation is straight line, total distance between $\mathcal{F}$'s view in interaction with $\mathcal{B}_1$ and in real execution is at most ϵ_E. This implies in particular that $\epsilon_{\mathcal{B}_0} = \epsilon$, $|\epsilon_{\mathcal{B}_1} - \epsilon| \leq \epsilon_E$ and $|\Pr[E_2^{\mathcal{B}_0}] - \Pr[E_2^{\mathcal{B}_1}]| \leq \epsilon_E$. So $\epsilon/4q_k \leq acc_{\mathcal{B}_0}$ and $(\epsilon - \epsilon_E)/4q_k \leq acc_{\mathcal{B}_0}$. Thus equation (2) becomes:

$$\frac{\epsilon-\epsilon_E}{4q_k}\left(\frac{\epsilon-\epsilon_E}{4q_k q_h} - \frac{1}{2^\kappa}\right) \leq \Pr[E_1^{\mathcal{B}_1}] + \Pr[E_2^{\mathcal{B}_0}] + \epsilon_E \tag{3}$$

The actual reduction algorithm $\mathcal{R}$, runs both $\mathcal{F}_{\mathcal{B}_0}$ and $\mathcal{F}_{\mathcal{B}_1}$. If ${E_1}^{\mathcal{B}_1}$ happens, then $z^e y^{-c_j} = \tilde{z}^e\tilde{y}^{-\tilde{c}_j}$. Substituting $y = \tilde{y} = (\mathring{y})^{2n_1}x^{2e}$ where n_1 is the number of players for whom the reduction has embedded the challenge (see figure 5) yields

$$\left((z/\tilde{z})x^{2(c_j-\tilde{c}_j)}\right)^e = (\mathring{y})^{2n_1(c_j-\tilde{c}_j)} \tag{4}$$

```
**Init**$(n, e, \mathring{y})$:
  Pick prime $e'$ where $el \leq e' \leq 2^{2\kappa}$ and run $\mathsf{tdCKG}((n, e, e'), \mathring{y})$ to get $(td, K)$,
  set $mpk$ as $(n, e, e', K)$ and run $\mathcal{F}$ on input $mpk$;

**SimKeyDer**$(Id)$:
  Query $\mathcal{H}_1$ on $Id$ and look up $\mathsf{H}_1[Id]$ to get $(b, \delta, y)$. If $b = 0$, return $\delta$
  otherwise abort the simulation with failure outputting $(0, \lambda)$.

**SimMSign**$(m, Id)$:
  Query $\mathcal{H}_1$ on $Id$ and look up $\mathsf{H}_1[Id]$ to get $(b, \delta, y)$. If $(b = 0)$ then run
  $\mathsf{MSign}(m, Id)$; otherwise:
  $-(\tilde{\mathcal{C}}, st) \leftarrow \mathsf{tdCom}_K(td)$;
    Send $(Id, \tilde{\mathcal{C}})$ to $\mathcal{F}$;
  $-$Upon receiving $(Id_j, \mathcal{C}_j)$ for $P_j \in \mathcal{P}$,
    $IdSet \leftarrow \{Id_j\}_{P_j \in \mathcal{P}}$; $\mathcal{C} \leftarrow \bigotimes_{P_j \in \mathcal{P}} \mathcal{C}_j$;
    $c \leftarrow \mathcal{H}_2(\mathcal{C}, IdSet, m)$;
    $(\tilde{D}, \tilde{z}) \leftarrow \mathsf{RstEquiv}_K(td, st, c, \delta)$;
    Send $(\tilde{z}, \tilde{D})$ to $\mathcal{F}$;
  $-z \leftarrow \prod_{P_j \in \mathcal{P}} z_j$; $D \leftarrow \bigoplus_{P_j \in \mathcal{P}} D_j$;
    Output $\sigma = (z, \mathcal{C}, D)$;

**SimHash**:
  $\mathcal{H}_1(Id)$: If $Id$ is not previously queried, pick $\delta$ uniformly at random from $\mathrm{QR}_n$,
  toss a biased coin $b$ so that $b = 0$ with probability $\rho$ and $b = 1$ with probability $1 - \rho$.
  If $b = 0$, set $y \leftarrow \delta^e$ otherwise set $y \leftarrow \mathring{y}\delta^e$. Store $(b, \delta, y)$ to $\mathsf{H}_1[Id]$.
  Return $\mathsf{H}_1[Id]$.

  $\mathcal{H}_2(\mathcal{C}, IdSet, m)$: If $(\mathcal{C}, IdSet, m)$ is an $i^{\text{th}}$ distinct query of $\mathcal{F}$ to $\mathcal{H}_2$,
  then query $\mathcal{H}_1(Id_i)$ for every $Id_i \in IdSet$ and set $\mathsf{H}_2[(\mathcal{C}, IdSet, m)] \leftarrow c_i$;
  Return $\mathsf{H}_2[(\mathcal{C}, IdSet, m)]$;

**Finalize**:
  Upon receiving a valid forgery $(m, IdSet, \sigma)$ from $\mathcal{F}$, parse $\sigma$ as $(z, \mathcal{C}, D)$ and query
  $\mathcal{H}_2$ on $(\mathcal{C}, IdSet, m)$. Let $IdSet_0 = \{Id_i | b_i = 0\}$ and $IdSet_1 = \{Id_i | b_i = 1\}$.
  If $IdSet_1 = \emptyset$ then abort the simulation with failure outputting $(0, \lambda)$. Otherwise set
  $x \leftarrow \prod_{Id_i \in IdSet}(x_i)$, $n_1 = |IdSet_1|$ and return $(j, (x, n_1, m, IdSet, \sigma))$ where $j$
  is the index of $c$ in the hash table $\mathsf{H}_2$.
```

Fig. 5. The procedures SimHash and Finalize that $\mathcal{B}_0$ and $\mathcal{B}_1$ use and the procedures Init, SimKeyDer, and SimMSign that $\mathcal{B}_1$ uses.

Now since $l2^{\kappa+1} < e$, therefore $\gcd(e, n_1(c_j - \tilde{c_j})) = 1$ and one can easily compute the e-th root of $\mathring{y}$ using the extended Euclidean algorithm.

If ${E_2}^{\mathcal{B}_0}$ happens, then $\mathcal{R}$ immediately translates it into an attack against binding property of commitment scheme $\mathcal{C}$ by returning $(c, D, \tilde{D}, z^e y^{-c_j}, \tilde{z}^e \tilde{y}^{-\tilde{c}_j})$. To see this note that as argued before, $y = \tilde{y}$, $\mathcal{C} = \tilde{\mathcal{C}}$ and since ${E_2}^{\mathcal{B}_0}$ is occurred, thus $z^e y^{-c_j} \neq \tilde{z}^e \tilde{y}^{-\tilde{c}_j}$ and due to validity of the forgeries we have $\mathsf{Open}_K(\mathcal{C}, D, z^e y^{-c_j}) = \mathsf{Open}_K(\mathcal{C}, \tilde{D}, \tilde{z}^e \tilde{y}^{-\tilde{c}_j}) = 1$. Moreover the commitment key K is outputted by CKG in the execution of $\mathcal{B}_0$. Thus $\Pr[E_1^{\mathcal{B}_1}] \leq \epsilon'$ and $\Pr[E_2^{\mathcal{B}_0}] \leq \epsilon_B$ and hence equation (3) becomes

$$\frac{\epsilon - \epsilon_E}{4q_k}\left(\frac{\epsilon - \epsilon_E}{4q_k q_h} - \frac{1}{2^\kappa}\right) \leq \epsilon' + \epsilon_B + \epsilon_E \tag{5}$$

The running time t_R of the reduction algorithm $\mathcal{R}$ is twice the maximum of running time of the algorithms $\mathcal{B}_0$ and $\mathcal{B}_1$. But the running time of $\mathcal{B}_0$ and $\mathcal{B}_1$ is dominated by the running time of the forger $\mathcal{F}$ plus the time spent by the simulators to answer the hash, signing and key derivation queries. Thus $t_R \leq 2(t + (3q_s + q_h)t_{exp})$ where t_{exp} is the time required for exponentiation in

$\mathbb{Z}_n^*$. On the other hand since $\mathcal{R}$ either answers the RSA challenge or returns an attack against the binding property of the commitment $\mathcal{C}$, it must be true that $\min(t', t_B) \le t_R$. Thus:

$$t \ge \frac{1}{2}\min(t', t_B) - (3q_s + q_h)t_{exp}$$

7 Identity-Based Aggregate Signature Scheme

The construction in the previous section can be easily modified to obtain a 2-round identity based aggregate signature (IBAS) scheme provably secure under RSA assumption. For this purpose, one needs to modify the verification algorithm to support the case where different challenges are acquired in step 4.2 of the protocol due to querying $\mathcal{H}_2$ on different messages. More precisely, the resulting IBAS scheme is exactly the same as the scheme described in figure 4 except that its verification algorithm would be as follows: Parse σ as $(z, \mathcal{C}, D)$ and mpk as (n, e, e', K); Compute $R \leftarrow z^e \prod_{Id_i \in IdSet} (\mathcal{H}_1(Id_i))^{2c_i}$ where c_i is output of $\mathcal{H}_2$ on input $(\mathcal{C}, IdSet, m_i)$ and check whether $\mathsf{Open}_K(\mathcal{C}, D, R) = 1$.

The security proof for this IBAS scheme is similar to the proof given in the previous section. Namely the reduction runs two simulators; in one simulator the challenge is embedded in the commitment key and in the other it is embedded in hashes of IDs. Therefore with high probability, if the forgery happens the reduction translates it either to either attack the binding property of the commitment scheme (event E_2 in the previous proof) or to find e-th root of the challenge (event E_1 in the previous proof). The security proof of the IBAS scheme is similar to the security proof of IBMS scheme described in the previous section. The most important difference is that in order to find the e-th root of the challenge we have the following equation instead of equation 4 in the previous proof:

$$(\mathring{y})^{2\sum_{Id_i \in IdSet_1}(\tilde{c_i} - c_i)} = \left((z/\tilde{z}) \prod_{Id_i \in IdSet} x_i^{2(\tilde{c_i} - c_i)} \right)^e$$

Therefore to be able to compute e-th root of $\mathring{y}$, we need $\gcd(e, 2\sum_{Id_i \in IdSet_1}(\tilde{c_i} - c_i)) = 1$. In particular, the reduction succeeds as long as $\sum_{Id_i \in IdSet_1}(\tilde{c_i} - c_i) \ne 0 \bmod e$, *i.e.* unless the challenges in the two branches of the forking algorithm sum up to the same value mod e, which happens with only negligible probability.

References

[BA03] Kenneth Barr and Krste Asanovic. Energy aware lossless data compression. In *MobiSys*, 2003.

[BCJ08] Ali Bagherzandi, Jung Hee Cheon, and Stanislaw Jarecki. Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In *ACM Conference on Computer and Communications Security*, pages 449–458, 2008.

[BGOY10] Alexandra Boldyreva, Craig Gentry, Adam O'Neill, and Dae Hyun Yum. Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In *Cryptology ePrint Archive, Report 2007/438, Revised 21/02/2010*, 2010.

[BJ10] Ali Bagherzandi and Stanislaw Jarecki. Identity-based aggregate and multi-signature schemes based on RSA. (full version), 2010.

[BN06] Mihir Bellare and Gregory Neven. Mult-signatures in the plain public-key model and a general forking lemma. In *Conference on Computer and Communications Security, CCS'06*, pages 390– 399, 2006.

[BN07] Mihir Bellare and Gregory Neven. Identity-based multi-signatures from rsa. In *CT-RSA*, pages 145–162, 2007.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, pages 174–187, 1994.

[Cor00] Jean-Sébastien Coron. On the exact security of full domain hash. In *CRYPTO*, pages 229–235, 2000.

[Dam00] Ivan Damgård. Efficient concurrent zero-knowledge in the auxiliary string model. In *EUROCRYPT*, pages 418–430, 2000.

[GHK06] David Galindo, Javier Herranz, and Eike Kiltz. On the generic construction of identity-based signatures with additional properties. In *ASIACRYPT*, pages 178–193, 2006.

[GQ88] Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In *CRYPTO*, pages 216–231, 1988.

[GR06] Craig Gentry and Zulfikar Ramzan. Identity-based aggregate signatures. In *Public Key Cryptography*, pages 257–273, 2006.

[Her06] Javier Herranz. Deterministic identity-based signatures for partial aggregation. *Comput. J.*, 49(3):322–330, 2006.

[KT05] Jihye Kim and Gene Tsudik. Srdp: Securing route discovery in dsr. In *MobiQuitous*, pages 247–260, 2005.

[MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-subgroup multisignatures. In *ACM Conference on Computer and Communications Security, CCS'01*, October 2001.

[Nev08] Gregory Neven. Efficient sequential aggregate signed data. In *EUROCRYPT*, pages 52–69, 2008.

[Sch89] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO*, pages 239–252, 1989.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.