Compressed Σ-Protocols for Bilinear Group Arithmetic Circuits and Applications

Thomas Attema[1,2,4,*], Ronald Cramer[1,2,†], and Matthieu Rambaud[3,‡]

[1] CWI, Cryptology Group, Amsterdam, The Netherlands
[2] Leiden University, Mathematical Institute, Leiden, The Netherlands
[3] Telecom Paris, Institut Polytechnique de Paris, France
[4] TNO, Cyber Security and Robustness, The Hague, The Netherlands

Version 3 - March 10, 2021[5,6]

Abstract. Recent developments in zero-knowledge have yielded various communication-efficient protocols for proving correctness of statements captured by an arithmetic circuit. Since any relation can be translated into an arithmetic circuit relation, these primitives are extremely powerful and widely applied. However, this translation often comes at the cost of losing *conceptual simplicity* and *modularity* in cryptographic protocol design.

For this reason, Lai et al. (CCS 2019), show how Bulletproof's circuit zero-knowledge protocol (Bootle et al., EUROCRYPT 2016 and Bünz et al., S&P 2018) can be generalized to work for *bilinear group arithmetic circuits* directly, without requiring these circuits to be translated into arithmetic circuits. For many natural relations their approach is actually more efficient than the indirect circuit ZK approach.

We take a different approach by generalizing Compressed Σ-Protocol Theory (CRYPTO 2020). Besides its conceptual simplicity our approach also has practical advantages; we reduce the communication costs by roughly a factor 3.

As a first application, we construct the first k-out-of-n threshold signature scheme (TSS) with both *transparent setup* and threshold signatures of size logarithmic in n. Each individual signature is of a so-called BLS type, the threshold signature hides the identities of the k signers and the threshold k can be dynamically chosen at aggregation time. Prior TSSs either have signature size linear in k or require a trusted setup.

As a second application, we give an explicit and direct construction for a proof that many signed messages satisfy a public predicate. The applications of this generic functionality are numerous. For instance, it implies a compiler from crash-fault tolerant distributed protocols into maliciously secure ones. Recently, it was shown that this in turn allows the so-far quadratic costs of certain consensus mechanisms to be reduced down to quasi linear in the number of players. Moreover, our construction finds applications in digital transaction and anonymous credential systems. The main benefit of this direct construction is that it avoids the large arithmetic circuits that are typically encountered when relying on the black-box application of circuit ZK systems, thereby achieving concretely efficient protocols.

Keywords: Zero-Knowledge, Bilinear Groups, Pairings, Compressed Σ-Protocol Theory, Threshold Signature Schemes, Consensus.

1 Introduction

Bulletproofs [BCC+16, BBB+18] introduced an ingenious technique to compress the communication complexity of discrete logarithm (DL) based circuit zero-knowledge (ZK) protocols from linear to logarithmic.

[*] thomas.attema@tno.nl

[†] cramer@cwi.nl, cramer@math.leidenuniv.nl

[‡] rambaud@enst.fr

[5] **Change log** w.r.t. Version 1 - November 17, 2020: (a) minor editorial changes throughout, and (b) included a second application of our techniques.

[6] **Change log** w.r.t. Version 2 - March 10, 2021: corrected a typo in the abstract.

Their approach was presented as a drop-in replacement for the well-established Σ-protocol theory and it results in efficient zero-knowledge protocols for relations captured by a circuit defined over $\mathbb{Z}_q \cong \mathbb{Z}/(q\mathbb{Z})$. In [AC20], Bulletproofs and Σ-protocol theory were reconciled by repurposing an appropriate adaptation of Bulletproofs as a black-box compression mechanism for basic Σ-protocols. They first show how to handle linear arithmetic relations by deploying a basic Σ-protocol. Second, they show how an adaptation of Bulletproofs allows the communication complexity of the basic Σ-protocol to be compressed from linear to logarithmic. Hence, the resulting *compressed Σ-protocol* allows a prover to prove *linear* statements with a communication complexity that is *logarithmic* in the size of the witness. Finally, to handle arbitrary non-linear relations, arithmetic secret sharing based techniques [CDP12] are deployed to *linearize* these non-linearities. Cryptographic protocol design can now follow well-established approaches from Σ-protocol theory, but with the additional black-box compression mechanism to reduce the communication complexity down to logarithmic.

These, and other, recent advances in communication-efficient circuit ZK lead to an obvious, but *indirect*, approach for efficient protocols for arbitrary relations:

1. Construct an arithmetic circuit capturing the relation.
2. Apply an efficient circuit ZK protocol to this arithmetic circuit.

However, for some relations, the associated arithmetic circuits can be large and complex. Thereby losing the conceptual simplicity and possibly even the concrete efficiency over a more *direct* approach. The work of [ACF20], for instance, describes a number of efficiency advantages of their direct approach for proving knowledge of k discrete logarithms out of n public group elements.

Moreover, Lai et al. [LMR19] construct a zero-knowledge proof system for directly handling relations captured by *bilinear group arithmetic circuits*. A bilinear group is a tuple $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$, where $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map, also called a pairing, and $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are groups (group operations are written additively) of prime order q generated by G, H and $e(G, H)$, respectively. A bilinear group arithmetic circuit, or a bilinear circuit, is a circuit in which each wire takes values in $W \in \{\mathbb{Z}_q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$ and the gates all have fan-in 2 and unbounded fan-out. Gates are either group operations, $\mathbb{Z}_q$-scalar multiplications or bilinear pairings. For more details see Section 6.3. Bilinear circuits directly capture relations encountered in, e.g., identity based encryption [SW05] and structure preserving signatures [AFG$^+$10]. We note that, for a highly optimized group of order $q \approx 2^{256}$, multiplying a single group element with a $\mathbb{Z}_q$-scalar requires an arithmetic circuit with approximately 800 multiplication gates [HBHW20], instead of a single gate in the bilinear circuit model. Hence, besides conceptual simplicity there can be significant efficiency advantages of the *direct* approach over the *indirect* approach that uses generic solutions for arithmetic circuit ZK.

In this work, we focus on two other applications of ZK protocols for relations defined over bilinear circuits: threshold signature schemes (TSSs) [DF89] and proving knowledge of a set of signed messages satisfying a public predicate.

A k-out-of-n TSS is a standard signature scheme, allowing each of the n players to individually sign arbitrary messages m, enriched with a public k-aggregation algorithm. The k-aggregation algorithm takes as input k signatures, issued by *any* k distinct players, on the same message m and outputs a *threshold signature* σ. A naive TSS is obtained by exhibiting the k individual signatures directly. However, this approach results in threshold signatures with size linear in the threshold k. The main goal for TSSs is to have *succinct* threshold signatures, i.e., with size sub-linear in k. The first succinct construction [Sho00] immediately found an application in reducing the communication complexity of consensus protocols [CKS05]. This application was revived recently [LM18, YMR$^+$19, ADD$^+$19, AMS19]. The impact of succinctness is significant since, in consensus applications, the threshold k is of the same order of magnitude as n (typically $k = n/2$ or $k = 2n/3$). Although desirable in some applications, it is not required that a threshold signature *hides* the k-subset of signers.

A TSS can be considered a special case of the following stronger functionality: proving knowledge of a set of signed messages satisfying a public predicate. In a TSS the predicate simply states that at least k-out-of-n signed messages are equal to a public message m. This stronger functionality has numerous applications. First, it allows for a compiler of crash-fault tolerant distributed protocols into maliciously secure ones, i.e., protocols that are secure against maliciously corrupted players. Lamport [Lam11] describes a similar compiler. A key feature of his compiler, called "Byzantizing an Algorithm", is that a player appends its outgoing message m with a cryptographic proof that shows that the output m is computed honestly. Lamport's proof proceeds by naively appending m with the messages $m_{i_1}, \dots, m_{i_k}$ and their signatures $\sigma_{i_1}, \dots, \sigma_{i_k}$. Removing these linear communication costs would have a direct impact on the complexity of algorithms. This is evidenced by the recent and unpublished work of [Ram20], where it is shown how to reduce the so-far quadratic communication costs of certain desirable security properties of consensus mechanism down to quasi-linear. Their approach is to replace the naive proofs of Lamport's compiler by communication-efficient transparent proofs-of-knowledge, thereby reducing the size of the proofs from $O(n)$ down to $O(\log(n))$. Other applications are proving the correctness of large blocks of transactions [AGM18, BBHR19] and anonymous credential systems [CGM16].

1.1 Contributions

In this work, we present a novel ZK protocol for relations captured by bilinear circuits. We show that there is a generalization of the approach of [AC20] for arithmetic circuit relations to bilinear circuit relations. The main ingredient required for this generalization is a homomorphic commitment scheme that allows a prover to commit to vectors $\mathbf{x} \in \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2} \times \mathbb{G}_T^{n_T}$ [AFG$^+$10, LMR19]. Generalizing [AC20], our approach is to first *compress* a basic Σ-protocol for proving linear statements about committed vectors $\mathbf{x}$, and second to show how to handle arbitrary bilinear circuit relations by *linearizing* non-linearities. This leads to a conceptually simple and modular construction of ZK protocols for bilinear circuit relations. We actually show that our generalization works for any circuit model in which all gates have fan-in 2 and are either linear of bilinear.

The communication complexity of our approach is derived from the properties of the commitment scheme. The size of a commitment to a vector $\mathbf{x} \in \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2} \times \mathbb{G}_T^{n_T}$ is constant in the dimensions n_0, n_1 and n_2, but it is linear in the dimension n_T. For this reason the communication complexity of our approach is logarithmic in n_0, n_1 and n_2, but linear in n_T. We consider a strictly stronger application scenario than the prior work of [LMR19], i.e., their results are restricted to a more limited class of circuits,[7] *and* we improve upon their communication costs by roughly a factor 3. More precisely, we reduce the constant in the logarithmic component of the communication costs from 16 down to 6, and the constant in the linear component from 3 down to 1. See Section 6.5 for a detailed comparison.

Another application of the commitment scheme of [AFG$^+$10, LMR19] is that it allows a prover to commit to Pedersen commitments in a pairing-based platform. This layered approach, of committing to commitments, was already suggested in [AFG$^+$10] and it allows a prover to commit to n^2 $\mathbb{Z}_q$-coefficients using only $2n+1$ public group elements, instead of the n^2+1 public group elements required when using Pedersen commitments directly. Replacing the Pedersen commitment scheme, in circuit ZK protocols derived from Bulletproofs [BCC$^+$16, BCC$^+$16] or Compressed Σ-Protocol Theory [AC20], by this layered commitment scheme immediately gives a square root reduction in the size of the set of public parameters while leaving the logarithmic communication costs exactly the same.

An additional advantage of our approach is that we can handle linear relations directly. By contrast, Lai et al. [LMR19] generalize the Bulletproof approach [BCC$^+$16, BBB$^+$18] where the pivotal protocol handles a specific non-linear inner-product relation. Applying this approach to a linear relation requires a cumbersome approach of capturing this linear relation by a set of non-linear inner-product constraints, leading to unnecessarily complicated protocols.

As an application of our compressed Σ-protocol for proving linear relations, we construct a transparent k-out-of-n threshold signature scheme (TSS) with threshold signatures that are $O(\kappa \log(n))$ bits. Recall that a TSS enables any set of at least k players, in a group of n, to issue a "threshold" signature on a message m, but no subset of less than k players is able to issue one. A TSS is called *transparent* if it does not require a trusted setup phase, i.e., all public parameters are random coins. Given recent advances in efficient circuit zero-knowledge, an obvious solution is to construct a threshold signature as a proof of knowledge attesting the knowledge of k-out-of-n signatures. With the appropriate ZK protocol this would immediately

[7] This is perhaps not immediate from the paper [LMR19], but it has been confirmed by the authors.

result in a transparent TSS with sublinear size threshold signatures. However, we have not encountered this obvious approach in literature. Perhaps because this approach would require an inefficient reduction from the corresponding threshold signature relation to a relation defined over an arithmetic circuit.

For this reason, we follow a more *direct* approach avoiding this inefficient reduction. Namely, we append the BLS signature scheme [BLS01] with a k-aggregation algorithm. The BLS signature scheme is defined over a bilinear group. In particular, the BLS verification algorithms naturally fit with our compressed Σ-protocols for relations defined over bilinear groups. To derive the required threshold functionality, we use an appropriated adaptation of k-out-of-n proofs of partial knowledge from a recent unpublished work [ACF20].

The compressed Σ-protocols are interactive and can be made non-interactive by the Fiat-Shamir transform [FS86]. The non-interactive proofs contain precisely the messages sent from the prover to the verifier. Hence, the logarithmic proof size is inherited by the logarithmic communication complexity of the compressed Σ-protocol. More precisely, a k-out-of-n threshold signature contains $4\lceil\log_2(n)\rceil + 3$ elements of $\mathbb{G}_T$, 1 element of $\mathbb{G}_1$ and 1 element of $\mathbb{Z}_q$.

The k-aggregation algorithm can be evaluated by any party with input at least k valid signatures from distinct signers. Besides the signatures, the k-aggregation algorithm only takes public input values. Moreover, the threshold k can be chosen at aggregation time independent of the set-up phase. By contrast, Shoup's construction [Sho00] requires a different trusted setup phase for every threshold k. Since the compressed Σ-protocol is zero-knowledge, an additional property of our TSS is that a threshold signature hides the k-subset of signers $\mathcal{S}$. Our TSS does not require a trusted setup and is therefore transparent. More precisely, the players can generate their own public-private key-pairs and the Σ-protocol only requires an unstructured public random string defined by the public parameters of the commitment scheme.

Finally, we construct a protocol for a generalized functionality: proving that a set of signed messages satisfies a public predicate. So far all known protocols with logarithmic communication rely on the black-box application of communication-efficient proofs-of-knowledge for arithmetic circuit relations. This typically requires hash-functions, commitment-functions and/or signature verification algorithms to be expressed in terms of an arithmetic circuit. In turn, leading to large circuits influencing the concrete communication-efficiency. We give an explicit and direct construction, with logarithmic complexity, avoiding black-box reductions to circuit ZK techniques.

As in our TSS, we use a signature scheme for which the verification equations are defined over a bilinear group. However, since its verification equation involves a hash-function, the BLS signature does not suffice. Note that this was not an issue for our TSS, because there the input to the hash function is a public message m. Here, the input messages $m_1, \dots, m_n$ can be distinct and are not public. Therefore, we use instead a structure-preserving signature (SPS) scheme that avoids hash-functions. In particular, we use the SPS scheme of [AGHO11]. The second challenge that we encounter is that the verification algorithm of this SPS scheme is non-linear. More precisely, the verification of n signatures is captured by a verification circuit with n bilinear pairing gates. Directly applying our linearization strategy results in a communication complexity linear in the number of bilinear pairing gates, i.e., this approach does not suffice. Fortunately, our framework can be naturally composed with existing inner product arguments, such as [BCC$^+$16, BBB$^+$18, LMR19, BMMV19]. Namely, we demonstrate in Section 7 how our protocol can transform these n non-linearities into a single inner-product to which existing techniques apply. This emphasizes the versatility and plug-and-play nature (modularity) of compressed Σ-protocol theory. Non-standard and more complicated functionalities can be implemented directly, by composing various building-blocks in a conceptually simple manner.

1.2 Related Work

Zero-Knowledge Proof Systems. Groth and Sahai [GS08] were the first to consider zero-knowledge proof systems for relations defined over bilinear groups *directly*. In contrast to more standard indirect approaches, their work avoids inefficient reductions to arithmetic circuit relations. Bilinear groups have found applications in many areas of cryptography. For instance, in digital signatures, identity based encryption and efficient zero-knowledge proof systems. For this reason many relevant relations are naturally defined over bilinear groups. The goal is not only to achieve efficiency, but also modularity in the design of cryptographic protocols.

A drawback of the Groth-Sahai proof system is that its proof sizes are linear in the size of the statements. By contrast, Bulletproofs [BCC$^+$16, BBB$^+$18] are practically efficient DL-based proof systems for arithmetic circuit relations with logarithmic proof sizes. Their main building block is an efficient protocol for proving a specific non-linear inner-product relation. Arbitrary relations captured by an arithmetic circuit are reduced to a set of inner-product constraints. Lai et al. [LMR19] adapted the techniques from Bulletproofs to the bilinear circuit model achieving a communication-efficient ZKP system for relations defined over bilinear circuits. More precisely, the communication complexity is logarithmic in the number of $\mathbb{Z}_q$, $\mathbb{G}_1$ and $\mathbb{G}_2$ inputs, but linear in the number of $\mathbb{G}_T$ inputs. They first reduce the bilinear circuit relation to a set of inner-products constraints, and subsequently describe protocols for proving various inner-product relations. The work of [BMMV19] improves the efficiency for a specific subset of bilinear inner-product relations. Hence, although these approaches avoid reductions to arithmetic circuits, they do rely on the reduction to a set of inner-product constraints.

In [AC20], an alternative approach for arithmetic circuit relations is described. Their pivotal protocol is a basic Σ-protocol for proving linear relations. They show how to compress the communication complexity down to logarithmic and how to handle non-linearities in arbitrary arithmetic circuit relations. This approach is compatible with standard Σ-protocol theory and avoids the need for reinventing cryptographic protocol design around non-linear inner-product relations. Here, we generalize compressed Σ-protocols to the bilinear circuit model.

Threshold Signature Schemes. Shoup's TSS [Sho00] already achieves threshold signatures of constant size. However, his approach, and all other approaches with threshold signature sizes sub-linear in k and n are not transparent [GJKR96, GJKR03, Bol03, LJY16, HAP18, KG20, KSM20, GG20]. These works require either an explicit trusted dealer, or they have implemented this trusted dealer by an MPC (or other interactive) protocol that is evaluated before messages are signed. At first glance it might seem that [GG20] also achieves a transparent setup. However, in their protocol the k signing players first have to run an interactive protocol before they can generate threshold signatures. This interactive protocol has to be evaluated before players can produce their inputs to the aggregation algorithm, therefore we consider this as a trusted setup.

The standard approach, introduced by Desmedt and Frankel [DF89], works by secret sharing the private key amongst the n players. This requires the private key to be generated by either a trusted dealer or an MPC protocol, i.e., this approach has a trusted set-up and is not transparent. Moreover, in contrast to our scheme, the threshold k should be fixed during the setup phase.

By contrast, all known *transparent* TSSs have size at least linear in the threshold k. Besides the naive implementation of simply outputting k valid signatures, there is also the following approach used by the decentralized transaction system Libra [Lib19] and by [NRS$^+$20]. Every player generates its own public-private key-pair. A threshold signature is computed as the sum of k individual BLS signatures, and it can be verified by running the BLS verification algorithm using the sum of the public keys of the k signers. Hence, the threshold signature should contain a list of the k signers, i.e., it is of size $O(n)$ or $O(k\log(n))$ depending on the exact encoding of this list. Moreover, these threshold signatures clearly do not hide the k-subset of signers. By contrast, Haque et al. [HKSS20] construct a transparent TSS that does hide the k-subset of signers. However, while individual signature sizes are logarithmic in n, the threshold signature are linear in the threshold k.

Finally, a recent unpublished work [BCG20] presents a different variant of a TSS, which they call *succinctly reconstructed distributed signatures* (SRDS). Their SRDS is most similar to the obvious approach of reducing the problem to an arithmetic circuit relation. It indeed applies a general (unspecified) SNARK in a black-box manner to achieve $O(\mathsf{poly}\log)$-size signatures. However, their SRDS can only tolerate up to $n/3$ corruptions.

1.3 Organization of the Paper

The remainder of the paper is organized as follows. In Section 2, we recall basic notation and definitions regarding bilinear groups and zero-knowledge proof systems. In Section 3, we define a number of commitment schemes generalizing Pedersen vector commitments. In Section 4, we describe a compressed Σ-protocol for proving linear relations about committed vectors. The compressed Σ-protocol has a logarithmic communication complexity. In Section 5, as an application of our compressed Σ-protocol, we describe a novel threshold signature scheme. In Section 6, we describe our linearization strategy that allows handling non-linear relations. Finally, in Section 7 we describe our protocol for proving knowledge of a set of signed messages satisfies a public predicate.

2 Preliminaries

2.1 Bilinear Groups

We consider the ring $\mathbb{Z}_q \cong \mathbb{Z}/(q\mathbb{Z})$ for a prime q. Moreover, we let $\mathbb{G}_1, \ldots, \mathbb{G}_k$ and $\mathbb{G}_T$ be groups of prime order q supporting discrete-log (DL) based cryptography, hence $\log(q) = O(\kappa)$ for security parameter κ. Some properties of commitment schemes used in this work rely on the stronger *Decisional Diffie-Hellman* (DDH) assumption. Therefore, we assume the DDH assumption to hold in all groups.

We write the group operations additively. Clearly, all groups $\mathbb{G}_i$ are $\mathbb{Z}_q$-modules and, for all $a \in \mathbb{Z}_q$ and $g \in \mathbb{G}_i$, the product $ag \in \mathbb{G}_i$ is well-defined. We write vectors in boldface and inner-products are defined naturally, i.e., for all $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_q^n$ and $\mathbf{g} = (g_1, \ldots, g_n) \in \mathbb{G}_i^n$ we define $\langle \mathbf{a}, \mathbf{g} \rangle := \sum_{i=1}^n a_i g_i$.

Let $G \in \mathbb{G}_1$ and $H \in \mathbb{G}_2$ be generators and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a non-trivial bilinear mapping (i.e., a pairing), i.e., $e(G, H)$ generates $\mathbb{G}_T$. Then, a tuple $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$ defines a *bilinear group*. For vectors $\mathbf{G} \in \mathbb{G}_1^n$ and $\mathbf{H} \in \mathbb{G}_2^n$ the following inner-product is defined $e(\mathbf{G}, \mathbf{H}) := \sum_{i=1}^n e(G_i, H_i)$.

We say that the *Symmetrical External Diffie-Hellman* (SXDH) holds in a bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$, if the DDH assumption holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$ [BGdMM05]. By the above assumption that the DDH assumption holds in all $\mathbb{G}_i$, it follows that the SXDH assumption holds for all bilinear groups that are considered in this work. The SXDH assumption implies that there is no efficiently computable isomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$, or from $\mathbb{G}_2$ to $\mathbb{G}_1$ [ACHdM05], i.e., we only consider bilinear groups of Type III [GPS08].

2.2 Proofs of Knowledge

We recall some standard notions regarding Proofs of Knowledge (PoKs) following the notation and definitions of [AC20, ACF20]. A relation R is a set of statement-witness pairs $(x; w)$. A μ-move protocol Π for relation R is an interactive protocol with μ communication rounds between a prover $\mathcal{P}$ and verifier $\mathcal{V}$. It allows $\mathcal{P}$ to convince $\mathcal{V}$ that it knows a witness w for statement x, i.e., $(x; w) \in R$. Protocol Π is also called an interactive proof for relation R. The statement x is *public input* for both $\mathcal{P}$ and $\mathcal{V}$ and the witness w is *private input* only for $\mathcal{P}$. In our protocol descriptions this is written as $\textsc{Input}(x; w)$, i.e., the public and private input are separated by a semicolon. As the output of the protocol $\mathcal{V}$ either accepts or rejects $\mathcal{P}$'s claim. The messages sent between $\mathcal{P}$ and $\mathcal{V}$ in one protocol execution are also referred to as a *conversation* or *transcript*. If $\mathcal{V}$ accepts the associated transcript, it is called accepting.

An interactive proof is said to be *public coin*, if all message from $\mathcal{V}$ are chosen uniformly at random and independent from prior messages. Interactive protocols that are public-coin can be made *non-interactive* by the Fiat-Shamir transformation [FS86], as proven in [BR93], without increasing the communication costs from $\mathcal{P}$ to $\mathcal{V}$. All interactive proofs in this work are public-coin.

Let us now describe some desirable (security) properties.

Definition 1 (Completeness). *An interactive proof Π is called perfectly* complete*, if on any input $(x; w) \in R$, the verifier $\mathcal{V}$ always accepts.*

Definition 2 (Knowledge Soundness). *An interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ is said to be knowledge sound with knowledge error $\kappa : \mathbb{N} \to [0, 1)$, if there exists a polynomial $q : \mathbb{N} \to \mathbb{N}$ and an algorithm χ (extractor) with the following properties. For each (potentially dishonest) PPT prover $\mathcal{P}^\star$, for each $x \in \{0, 1\}^\star$, whenever $(\mathcal{P}^*, \mathcal{V})(x)$ outputs accept with probability $\epsilon(x) \geq \kappa(|x|)$, the extractor χ, given input x and rewindable oracle access to the $\mathcal{P}^*$, runs in expected polynomial time and successfully outputs a witness w for statement x with probability at least $(\epsilon(x) - \kappa(|x|))/q(|x|)$.*

Definition 3 (Proof/Argument of Knowledge). *An interactive proof that is both complete and knowledge sound is said to be a Proof or Knowledge (PoK). PoKs for which knowledge soundness only holds under computational assumptions are also referred to as Arguments of Knowledge.*

Witness extended emulation [Lin03] gives an alternative notion for knowledge soundness, sufficient for most practical scenarios, and it is known to be implied by knowledge soundness [Lin03]. For details we refer to [Lin03, HKR19, AC20].

We now recall a generalization of the *special soundness* property. Special soundness is in general easier to handle than knowledge soundness. We first introduce the notion of a tree of accepting transcripts.

Definition 4 (Tree of Accepting Transcripts). *Let Π be a $(2\mu+1)$-move protocol. A $(k_1, k_2, \ldots, k_\mu)$-tree of accepting transcripts for protocol Π is a set of $\prod_{i=1}^{\mu} k_i$ accepting transcripts that are arranged in the following tree structure. The nodes in this tree correspond to the prover's messages and the edges correspond to the verifier's challenges. Every node at depth i has precisely k_i children corresponding to k_i pairwise distinct challenges. Every transcript corresponds to exactly one path from the root node to a leaf node.*

Definition 5 (Special Soundness). *A $(2\mu+1)$-move protocol is said to be $(k_1, k_2, \ldots, k_\mu)$-special sound, if there exists an efficient algorithm that on input a $(k_1, k_2, \ldots, k_\mu)$-tree of accepting transcripts for statement x, outputs a witness w for x.*

An interactive proof that is $(k_1, k_2, \ldots, k_\mu)$-special sound is known to have witness extended emulation [BCC+16, AC20].[8] Therefore, protocols that are complete and special sound are also referred to as proofs of knowledge (PoKs).

In some protocols there are rounds in which $\mathcal{V}$ sends multiple challenges per round, i.e., μ challenges are sent in less than $2\mu+1$ rounds. For these protocols we also consider the $(k_1, \ldots, k_\mu)$-special soundness property. However, in this case a tree of accepting transcripts contains nodes that do not correspond to a message sent from $\mathcal{P}$ to $\mathcal{V}$.

Definition 6 (Honest Verifier Zero-Knowledge (HVZK)). *An interactive proof Π is said to be honest verifier zero-knowledge (HVZK), if there exists a PPT simulator that, on input a statement x that admits a witness w, outputs an accepting transcript, such that simulated transcripts follow exactly the same distribution as transcripts between an honest prover and an honest verifier. If the simulator proceeds by first sampling the random challenges, the protocol is said to be special honest verifier zero-knowledge (SHVZK).*

Finally, we recall that two protocols, Π_a for relation R_a and Π_b for relation R_b, are said to be *composable*, if the final message of protocol Π_a contains a witness for relation R_b [AC20]. In this case, the composition $\Pi_b \diamond \Pi_a$ runs Protocol Π_a but replaces the witness for relation R_b in its final message by an appropriate instantiation of Protocol Π_b. If protocol Π_a is $(k_1, \ldots, k_{\mu_1})$-special sound and protocol Π_b is $(k'_1, \ldots, k'_{\mu_2})$-special sound, then the composition $\Pi_b \diamond \Pi_a$ is easily seen to be $(k_1, \ldots, k_{\mu_1}, k'_1, \ldots, k'_{\mu_2})$-special sound.

3 Commitment Schemes

The techniques in this paper work for any *homomorphic* commitment scheme of the following form:

$$\textsc{Com} : \mathbb{G}_S \times \mathbb{Z}_q^r \to \mathbb{G}_C, \quad (\mathbf{x}, \gamma) \mapsto \textsc{Com}(\mathbf{x}, \gamma), \tag{1}$$

where $\mathbb{G}_S := \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \cdots \times \mathbb{G}_k^{n_k}$ for groups $\mathbb{G}_1, \ldots, \mathbb{G}_k$ of prime order q, and $\gamma \in \mathbb{Z}_q^r$ is the commitment randomness, typically $r = 1$ or $r = 2$, i.e., to commit to a vector $\mathbf{x} \in \mathbb{G}_S$, a prover samples $\gamma \in \mathbb{Z}_q^r$ uniformly at random and outputs the commitment $\textsc{Com}(\mathbf{x}, \gamma) \in \mathbb{G}_C$. We assume that this commitment scheme is hiding and binding, possibly under computational hardness assumptions, and that it is homomorphic, i.e., $\textsc{Com}(\mathbf{x}_1, \gamma_1) + \textsc{Com}(\mathbf{x}_2, \gamma_2) = \textsc{Com}(\mathbf{x}_1 + \mathbf{x}_2, \gamma_1 + \gamma_2)$ for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{G}_S$ and $\gamma_1, \gamma_2 \in \mathbb{Z}_q$.

[8] In a recent unpublished work [ACK21], it is shown that $(k_1, k_2, \ldots, k_\mu)$-special soundness actually implies the stronger notion of knowledge soundness.

In this section, we describe a number of instantiations of this abstract commitment scheme. Subsequently, in the next sections, we show that the compressed Σ-protocol theory of [AC20] immediately generalizes from proving statements about Pedersen commitments to vectors $\mathbf{x} \in \mathbb{Z}_q^n$ to proving statements about commitments of vectors $\mathbf{x} \in \mathbb{G}_S$. For the techniques of [AC20] to work, it is only required that the commitment function is a *homomorphism*. The compression techniques are applicable if the commitments are *compact*, i.e., the size of the commitments is independent of the dimensions n_i of the variables that are to be compressed.

The first and best-known instantiation of this abstract commitment scheme is the Pedersen vector commitment scheme [Ped91], where $n_1 = \cdots = n_k = 0$. This commitment scheme is perfectly hiding and computationally binding under the discrete logarithm assumption. Applying this work to the Pedersen commitment scheme simply results in the compressed Σ-protocols of [AC20]. Recall that group operations are written additively.

Definition 7 (Pedersen Vector Commitment [Ped91]). *Let $\mathbb{G}$ be an Abelian group of prime order q. Pedersen vector commitments to vectors $\mathbf{x} \in \mathbb{Z}_q^n$ are defined by the following setup and commitment phase:*

- *Setup:* $\mathbf{g} = (g_1, \dots, g_n) \leftarrow_R \mathbb{G}^n$, $h \leftarrow_R \mathbb{G}$.
- *Commit:* $\textsc{Com}_1 : \mathbb{Z}_q^n \times \mathbb{Z}_q \to \mathbb{G}, \quad (\mathbf{x}, \gamma) \mapsto h\gamma + \langle \mathbf{g}, \mathbf{x} \rangle$.

Abe et al. [AFG+10] constructed a similar commitment scheme that works with bilinear groups $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$ and allows a prover to commit to vectors of group elements $\mathbf{x} \in \mathbb{G}_1^n$. A straightforward generalization shows that this approach allows a prover to commit to vectors $\mathbf{x} \in \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1}$ [LMR19]. The commitment scheme is perfectly hiding and computationally binding under the DDH assumption in $\mathbb{G}_1$. Analogously, this construction results in a commitment scheme for vectors $\mathbf{x} \in \mathbb{Z}_q^{n_0} \times \mathbb{G}_2^{n_2}$.

Definition 8 (Commitment to $(\mathbb{Z}_q, \mathbb{G}_1)$-vectors [AFG+10, LMR19]). *Let $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$ be a bilinear group and let $n_0, n_1 \geq 0$. The following setup and commitment phase define a commitment scheme for vectors in $\mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1}$:*

- *Setup:* $\mathbf{g} = (g_1, \dots, g_{n_0}) \leftarrow_R \mathbb{G}_T^{n_0}$, $h \leftarrow_R \mathbb{G}_T$, $\mathbf{H} = (H_1, \dots, H_{n_1}) \leftarrow_R \mathbb{G}_2^{n_1}$.
- *Commit:* $\textsc{Com}_1 : \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{Z}_q \to \mathbb{G}_T, \ (\mathbf{x}, \mathbf{y}, \gamma) \mapsto h\gamma + \langle \mathbf{g}, \mathbf{x} \rangle + e(\mathbf{y}, \mathbf{H})$.

Remark 1. As an application of the commitment scheme of Definition 8, Abe et al. [AFG+10] mentioned commitments to Pedersen vector commitments. A commitment to n n-dimensional Pedersen vector commitments is namely a commitment to an n^2-dimensional $\mathbb{Z}_q$-vector. This two-tiered commitment scheme only requires $2n + 1$ public group elements. By contrast, Pedersen's commitment scheme requires $n^2 + 1$ public group elements to commit to an n^2-dimensional $\mathbb{Z}_q$-vector. Replacing the Pedersen vector commitment scheme in [BCC+16, BBB+18, AC20] by this two-tiered commitment scheme results in arithmetic circuit ZK protocols with exactly the same communication complexity, but with a square root improvement in the size of the public parameters.

In addition, Lai et al. [LMR19] show how this approach can be extended to construct a commitment scheme for vectors with coefficients in $\mathbb{Z}_q, \mathbb{G}_1$ and $\mathbb{G}_2$. In contrast to the previous commitments, a commitment to a vector $\mathbf{x} \in \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2}$ consists of two target group elements. Informally, the reason is that, with high probability, $(S, -R) \in \mathbb{G}_1 \times \mathbb{G}_2$ is a non-trivial solution for the equation $e(x, R) + e(S, y) = 1$, where $(S, R) \in \mathbb{G}_1 \times \mathbb{G}_2$ is sampled uniformly at random. Such a solution would break the binding property of the naive generalization in which commitments consist of only one target group element. However, with high probability, there does not exist a solution $(x, y) \in \mathbb{G}_1 \times \mathbb{G}_2$ to the system of equations $e(x, R_1) + e(S_1, y) = 1$ and $e(x, R_2) + e(S_2, y) = 1$, where $(S_1, R_1), (S_2, R_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ are sampled uniformly at random. For this reason, the commitments consist of two target group elements and breaking their binding property can be reduced to solving a similar system of equations. The resulting commitment scheme is described in Definition 9. It is computationally hiding under the DDH assumption in $\mathbb{G}_T$, and it is computationally binding under the SXDH assumption [LMR19]. The scheme can be made perfectly hiding by introducing an additional randomizer $\gamma_2 \in \mathbb{Z}_q$.

Definition 9 (Commitment to $(\mathbb{Z}_q, \mathbb{G}_1, \mathbb{G}_2)$-vectors [LMR19]). *Let $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$ be a bilinear group and let $n_0, n_1, n_2 \geq 0$. The following setup and commitment phase define a commitment scheme for vectors in $\mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2}$:*

- *Setup:* $\mathbf{g} \leftarrow_R \mathbb{G}_T^{2\times n_0}$, $h \leftarrow_R \mathbb{G}_T^2$, $\mathbf{H} \leftarrow_R \mathbb{G}_2^{2\times n_1}$,$\mathbf{G} \leftarrow_R \mathbb{G}_1^{2\times n_2}$.
- *Commit:* $\text{Com}_1 : \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2} \times \mathbb{Z}_q \to \mathbb{G}_T^2, \quad (\mathbf{x}, \mathbf{y}, \mathbf{z}, \gamma) \mapsto h\gamma + \langle \mathbf{g}, \mathbf{x}\rangle + e(\mathbf{y}, \mathbf{H}) + e(\mathbf{G}, \mathbf{z})$, *where*

$$h\gamma + \langle \mathbf{g}, \mathbf{x}\rangle + e(\mathbf{y}, \mathbf{H}) + e(\mathbf{G}, \mathbf{z}) := \begin{pmatrix} h_1\gamma + \langle \mathbf{g}_1, \mathbf{x}\rangle + e(\mathbf{y}, \mathbf{H}_1) + e(\mathbf{G}_1, \mathbf{z}) \\ h_2\gamma + \langle \mathbf{g}_2, \mathbf{x}\rangle + e(\mathbf{y}, \mathbf{H}_2) + e(\mathbf{G}_2, \mathbf{z}) \end{pmatrix}. \tag{2}$$

The aforementioned commitment schemes do not allow a prover to commit to elements of the target group $\mathbb{G}_T$ of the bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. For this reason, we introduce the homomorphic commitment scheme of Definition 10. This scheme is based on the El Gamal encryption scheme [Gam84]. The commitment scheme is unconditionally binding and hiding under the DDH assumption in $\mathbb{G}_T$.

Definition 10 (Commitment to $(\mathbb{G}_T)$-vectors [Gam84, LMR19]). *Let $\mathbb{G}_T$ be an Abelian group of prime order q. The following setup and commitment phase define a commitment scheme for vectors in $\mathbb{G}_T^{n_T}$:*

- *Setup:* $\mathbf{g} \leftarrow_R \mathbb{G}_T^{n_T}$, $h \leftarrow_R \mathbb{G}_T$.
- *Commit:* $\text{Com}_2 : \mathbb{G}_T^{n_T} \times \mathbb{Z}_q \to \mathbb{G}_T^{n_T+1}, \quad (\mathbf{x}, \gamma) \mapsto \begin{pmatrix} h\gamma \\ \mathbf{g}\gamma + \mathbf{x} \end{pmatrix}.$

Note that, in contrast to the schemes of Definitions 7, 8 and 9, this commitment scheme is not compact, i.e, a commitment to a vector $\mathbf{x} \in \mathbb{G}_T^{n_T}$ contains $n_T + 1$ group elements. For this reason, the compression techniques applicable to compact commitments are of no benefit for commitments to $\mathbb{G}_T$-vectors, and we will treat commitments to target group elements separately.

Altogether, for a bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$, we obtain the following commitment scheme:

$$\text{Com} : \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2} \times \mathbb{G}_T^{n_T} \times \mathbb{Z}_q^2 \to \mathbb{G}_T^{n_T+2}, \quad (\mathbf{x}, \mathbf{y}, \gamma_1, \gamma_2) \mapsto \begin{pmatrix} \text{Com}_1(\mathbf{x}; \gamma_1) \\ \text{Com}_2(\mathbf{y}; \gamma_2) \end{pmatrix}, \tag{3}$$

where $\mathbf{x} \in \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2}$, $\mathbf{y} \in \mathbb{G}_T^{n_T}$, Com_1 is the commitments scheme from Definition 9, and Com_2 is the commitment scheme from Definition 10.

4 Compressed Σ-Protocol for Opening Homomorphisms

In this section, we describe a Compressed Σ-Protocol for opening homomorphisms on committed vectors with coefficients in $\mathbb{Z}_q, \mathbb{G}_1, \dots, \mathbb{G}_k$. More precisely, we construct a protocol for proving that a secret committed vector $\mathbf{x} \in \mathbb{G}_S = \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \cdots \times \mathbb{G}_k^{n_k}$ satisfies $f(\mathbf{x}) = y$ for a public homomorphism f and a public value y. From now on we assume to have access to a homomorphic commitment scheme

$$\text{Com} : \mathbb{G}_S \times \mathbb{Z}_q^r \to \mathbb{G}_C.$$

Our Compressed Σ-Protocol is a generalization of the approach of [AC20] for opening linear forms on committed $\mathbb{Z}_q$-vectors.

4.1 Basic Σ-Protocol

We describe a basic Σ-protocol for opening homomorphisms on committed vectors $\mathbf{x} \in \mathbb{G}_S$, i.e., a Σ-protocol for proving that the secret vector $\mathbf{x}$ satisfies that $f(\mathbf{x}) = y$ for a public homomorphism $f : \mathbb{G}_S \to \mathbb{H}$ and a public element $y \in \mathbb{H} := \mathbb{Z}_q \times \mathbb{G}_1 \times \cdots \times \mathbb{G}_k \times \mathbb{G}_h$, where $\mathbb{G}_h$ is an arbitrary group. More precisely, we describe a basic Σ-protocol for the following relation,

$$R = \big\{ \, (P, f, y; \mathbf{x}, \gamma) : P = \textsc{Com}\,(\mathbf{x}, \gamma)\,, f(\mathbf{x}) = y \big\}. \tag{4}$$

where $(P, f, y) \in \mathbb{G}_C \times \mathrm{Hom}(\mathbb{G}_S, \mathbb{H}) \times \mathbb{H}$ and $(\mathbf{x}, \gamma) \in \mathbb{G}_S \times \mathbb{Z}_q^r$.

Protocol 0, denoted by Π_0, describes a basic Σ-protocol for relation R and its main properties are summarized in Theorem 1. The Σ-protocol is a generalization of the Σ-protocol for opening linear forms of [AC20], and the Σ-protocol for opening homomorphisms of [ACF20], where the committed vectors have coefficients only in $\mathbb{Z}_q$.

Protocol 0 Σ-protocol Π_0 for relation R
Σ-protocol for opening a homomorphism.

$$\textsc{Input}(P, f, y; \mathbf{x}, \gamma)$$

$$\begin{array}{c} P = \textsc{Com}(\mathbf{x}, \gamma) \in \mathbb{G}_C \\ y = f(\mathbf{x}) \in \mathbb{H} \end{array}$$

$$\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier} \\
\mathbf{r} \leftarrow_R \mathbb{G}_S, \rho \leftarrow_R \mathbb{Z}_q, & & \\
A = \textsc{Com}(\mathbf{r}, \rho), t = f(\mathbf{r}) & \xrightarrow{A, t} & \\
 & & c \leftarrow_R \mathbb{Z}_q \\
 & \xleftarrow{c} & \\
\mathbf{z} = c\mathbf{x} + \mathbf{r} & & \\
\phi = c\gamma + \rho & & \\
 & \xrightarrow{\phi, \mathbf{z}} & \textsc{Com}(\mathbf{z}, \phi) \overset{?}{=} A + cP \\
 & & f(\mathbf{z}) \overset{?}{=} cy + t
\end{array}$$

Theorem 1 (Homomorphism Evaluation). *Π_0 is a Σ-protocol for relation R. It is perfectly complete, special honest-verifier zero-knowledge and unconditionally special sound. Moreover, the communication costs are:*

- *$\mathcal{P} \rightarrow \mathcal{V}$: 1 element of $\mathbb{G}_C$, 1 of $\mathbb{H}$, 1 of $\mathbb{G}_S$ and r elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \rightarrow \mathcal{P}$: 1 element of $\mathbb{Z}_q$.*

4.2 Reduction

The factors in the codomain $\mathbb{H}$ of the homomorphism f that are in $\{\mathbb{Z}_q, \mathbb{G}_1, \ldots, \mathbb{G}_{k-1}\}$ can be "incorporated into the commitment". The goal is not to hide the evaluation $y = f(\mathbf{x})$, in fact y is still public, but to reduce the overall communication complexity that is achieved after compression. Ultimately, this step will reduce a relevant constant in the communication complexity of our compressed Σ-protocol by a factor 1/2. This technique was first deployed in [BBB$^+$18] to improve the communication complexity of the protocols for certain inner-product relations from [BCC$^+$16]. Here, this technique is generalized to our setting.

We assume the homomorphic commitment scheme to be of the following form $\textsc{Com} = (\textsc{Com}_1, \textsc{Com}_2)$, with $\textsc{Com}_1$ compact and where

$$\begin{array}{l} \textsc{Com}_1 : \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \cdots \times \mathbb{G}_{k-1}^{n_{k-1}} \times \mathbb{Z}_q^{r_1} \rightarrow \mathbb{G}_{C_1}, \\ \textsc{Com}_2 : \mathbb{G}_k^{n_k} \times \mathbb{Z}_q^{r_2} \rightarrow \mathbb{G}_{C_2}, \\ \textsc{Com} \ \ : \mathbb{G}_S \times \mathbb{Z}_q^r \rightarrow \mathbb{G}_C := \mathbb{G}_{C_1} \times \mathbb{G}_{C_2}, \end{array} \tag{5}$$

for some r_1 and r_2 with $r = r_1 + r_2$. Hence, the size of the codomain $\mathbb{G}_{C_1}$ is independent from the input dimensions $n_0, n_1, \dots, n_{k-1}$, while the size of the codomain $\mathbb{G}_{C_2}$ does depend on the input dimension n_k. Recall that the commitment scheme of eq. (3) is of this form.

To describe the reduction, we write $f = (f_1, f_2)$, where $f_1 : \mathbb{G}_S \to \mathbb{Z}_q \times \mathbb{G}_1 \times \cdots \times \mathbb{G}_{k-1}$ and $f_2 : \mathbb{G}_S \to \mathbb{G}_k \times \mathbb{G}_h$, i.e., $\textsc{Com}$ is compact on the codomain of f_1. We extend the domain of our commitment scheme and write $\textsc{Com} : \mathbb{G}_S \times (\mathbb{Z}_q \times \mathbb{G}_1 \dots \mathbb{G}_{k-1}) \times \mathbb{Z}_q^r \to \mathbb{G}_C$ for the scheme that allows a prover to commit to vectors $(\mathbf{x}, \mathbf{y}) \in \mathbb{G}_S \times \mathbb{Z}_q \times \mathbb{G}_1 \times \cdots \times \mathbb{G}_{k-1}$. We assume that $\textsc{Com}_{old}(\mathbf{x}, \gamma) = \textsc{Com}_{new}(\mathbf{x}, 0, \gamma)$ for all $\mathbf{x} \in \mathbb{G}_S$ and $\gamma \in \mathbb{Z}_q^r$, justifying the fact that we use the same notation for both commitment schemes. For the commitment schemes of Section 3 this extension only requires additional public parameters to be sampled. Using this notation, Protocol 1, denoted by Π_1, gives another protocol for relation R. It is perfectly complete and special sound under the assumption that the commitment scheme $\textsc{Com}$ is binding. However, this protocol is not special honest verifier zero-knowledge (SHVZK); the witness $(\mathbf{x}, \gamma)$ is sent in the clear. The properties of this protocol are summarized in Lemma 1. Note that the trivial proof of knowledge, in which the prover just sends its witness, has better properties than protocol Π_1. Hence, protocol Π_1 is only useful when it is combined with the compression mechanism Π_2 of the next section.

Protocol 1 Argument of Knowledge Π_1 for R
Reduction from relation R to relation R_1.

$$\textsc{Input}(P, f, y; \mathbf{x}, \gamma)$$

$$P = \textsc{Com}(\mathbf{x}, 0, \gamma) \in \mathbb{G}_C$$
$$(y_1, y_2) = y = f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}))$$

Prover		Verifier
	$\xleftarrow{c}$	$c \leftarrow_R \mathbb{Z}_q$
	$\xrightarrow{\mathbf{x},\gamma}$	
		$\textsc{Com}(\mathbf{x}, cf_1(\mathbf{x}), \gamma) \overset{?}{=}$ $P + \textsc{Com}(0, cy_1, 0)$ $f_2(\mathbf{x}) \overset{?}{=} y_2$

Lemma 1. *Π_1 is a 2-move protocol for relation R. It is perfectly complete and computationally special sound, under the assumption that the commitment scheme is binding. Moreover, the communication costs are:*

- *$\mathcal{P} \to \mathcal{V}$: 1 element of $\mathbb{G}_S$ and r elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \to \mathcal{P}$: 1 element of $\mathbb{Z}_q$.*

Proof. **Completeness** follows directly.

Special soundness: We show that there exists an efficient algorithm χ that, on input two accepting transcripts, either extracts a witness for R_1, or finds two different openings to the same commitment, and thereby breaks the binding property of the commitment scheme.

So let $(c, \mathbf{x}, \gamma)$ and $(c', \mathbf{x}', \gamma')$ be two accepting transcripts with $c \neq c'$, then by subtracting the two verification equations and since $\textsc{Com}(\cdot)$ is a homomorphism,

$$\textsc{Com}\left(\mathbf{x} - \mathbf{x}', cf_1(\mathbf{x}) - c'f_1(\mathbf{x}'), \gamma - \gamma'\right) = \textsc{Com}\left(0, (c - c')y_1, 0\right).$$

Hence, either we have extracted two different openings to the same commitment, or $\mathbf{x} = \mathbf{x}'$, $cf_1(\mathbf{x}) - c'f_1(\mathbf{x}') = (c - c')y_1$ and $\gamma = \gamma'$. In the latter case, it follows that $f(\mathbf{x}) = f(\mathbf{x}') = y$. Moreover, from this it follows that

$$\textsc{Com}\left(\mathbf{x}, cf_1(\mathbf{x}), \gamma\right) = P + \textsc{Com}\left(0, cy_1, 0\right),$$

which implies that $\textsc{Com}\left(\mathbf{x}, 0, \gamma\right) = P$. Hence, $(\mathbf{x}, \gamma)$ is a witness for relation R, which completes the proof. □

We observe that the final message of Protocol $\mathit{\Pi}_1$ is a witness for relation

$$\begin{aligned} R_1 = \big\{ \big(Q \in \mathbb{G}_C, g = (g_1, g_2) \in \mathrm{Hom}(\mathbb{G}_S, \mathbb{H}), y_2 \in \mathbb{G}_k \times \mathbb{G}_h; \mathbf{x} \in \mathbb{G}_S, \gamma \in \mathbb{Z}_q^r \big) : \\ Q = \mathrm{C{\scriptstyle OM}}\left(\mathbf{x}, g_1(\mathbf{x}), \gamma\right), g_2(\mathbf{x}) = y_2 \big\}, \end{aligned} \tag{6}$$

where $Q = P + \mathrm{C{\scriptstyle OM}}(0, cy_1, 0)$ and $(g_1, g_2) = (cf_1, f_2)$ for a random challenge c. In other words, Protocol $\mathit{\Pi}_1$ has reduced relation R to relation R_1. The benefit of this reduction is that the number of public elements in R_1 is smaller, i.e., a statement of relation R_1 does not contain $y_1 \in \mathbb{Z}_q \times \mathbb{G}_1 \times \cdots \times \mathbb{G}_{k-1}$.

Note that the $\mathbb{G}_k$ factors of f can also be incorporated into the commitment. However, this will not result in a reduction of the communication complexity, because the commitment scheme $\mathrm{C{\scriptstyle OM}}$ in not compressing in its $\mathbb{G}_k$ component. For this reason we make the distinction between the $(\mathbb{Z}_1, \mathbb{G}_1, \dots, \mathbb{G}_{k-1})$-part and the $(\mathbb{G}_k, \mathbb{G}_h)$-part of the homomorphism $f = (f_1, f_2)$. In the next section, we show how to compress a protocol for relation R_1. Alternatively, we can compress a protocol for relation R directly, but this will yield larger communication costs.

4.3 Compression Mechanism

In this section, we describe a compression mechanism for relation R_1, i.e., a protocol for relation R_1 where the communication costs are smaller than simply sending the witness. Subsequently, we will show how this compression mechanism can reduce the communication complexity of the basic $\mathit{\Sigma}$-protocol $\mathit{\Pi}_0$.

We introduce the following notation. First note that, by reordering coefficients, a witness $(\mathbf{x}, \gamma) \in \mathbb{G}_S \times \mathbb{Z}_q^r$ for relation R_1 can be written as $(\mathbf{z}, \mathbf{x}_k, \gamma_2) = ((\mathbf{x}_0, \gamma_1), \mathbf{x}_1, \dots, \mathbf{x}_k, \gamma_2) \in \mathbb{Z}_q^{n_0+r_1} \times \mathbb{G}_1^{n_1} \times \cdots \times \mathbb{G}_k^{n_k} \times \mathbb{Z}_q^{r_2}$, where the commitment randomness $\gamma_1 \in \mathbb{Z}_q^{r_1}$ for the commitment scheme $\mathrm{C{\scriptstyle OM}}_1$ is combined with the secret $\mathbb{Z}_q$-coefficients of $\mathbf{x} \in \mathbb{G}_S$. In this notation $\mathrm{C{\scriptstyle OM}}_1(\mathbf{z})$ is a commitment to the vector $(\mathbf{x}_0, \dots, \mathbf{x}_{k-1})$ and the randomness γ_1 is no longer explicit. The reason for this change of notation is that the compression mechanism does not have to be zero-knowledge. For this reason the hiding property and the associated randomness γ_1 are irrelevant in this section.

For such a vector $\mathbf{z}$, we define the left and right halves $\mathbf{z}_L, \mathbf{z}_R \in \mathbb{Z}_q^{(n_0+r_1)/2} \times \mathbb{G}_1^{n_1/2} \times \cdots \times \mathbb{G}_{k-1}^{n_{k-1}/2}$, such that $\mathbf{z} = (\mathbf{z}_L, \mathbf{z}_R)$ up to reordering of the coefficients. We assume that $n_0 + r_1, n_1, \dots, n_{k-1}$ are all even; if not, the vector $\mathbf{z}$ can be appended with the appropriate number zeros. We extend the domain of the homomorphisms $g = (g_1, g_2)$ to vectors $\mathbf{z}$ of this form by simply ignoring the randomness γ_1, i.e., $g(\mathbf{z}, \mathbf{x}_k) = g(\mathbf{x}, \gamma_1) := g(\mathbf{x})$ for all $(\mathbf{z}, \mathbf{x}_k)$ and for all $g \in \mathrm{Hom}(\mathbb{G}_S, \mathbb{H})$. As before, we will extend the domain of the commitment scheme with the codomain of g_1, i.e., we define commitments of the form $\mathrm{C{\scriptstyle OM}}_1(\mathbf{z}, g_1(\mathbf{z}, \mathbf{x}_k)) := \mathrm{C{\scriptstyle OM}}_1(\mathbf{x}_0, \dots, \mathbf{x}_k, g_1(\mathbf{x}), \gamma_1)$ and $\mathrm{C{\scriptstyle OM}}\left(\mathbf{z}, \mathbf{x}_k, g_1(\mathbf{z}, \mathbf{x}_k), \gamma_2\right) := \mathrm{C{\scriptstyle OM}}\left(\mathbf{x}, g_1(\mathbf{x}), \gamma\right)$.

Moreover, for any $\mathbf{z}' \in \mathbb{Z}_q^{(n_0+r_1)/2} \times \mathbb{G}_1^{n_1/2} \times \cdots \times \mathbb{G}_{k-1}^{n_{k-1}/2}$, we define $(0, \mathbf{z}') := \big((0, \mathbf{z}_1'), (0, \mathbf{z}_2'), \dots, (0, \mathbf{z}_{k-1}')\big) \in \mathbb{Z}_q^{n_0+r_1} \times \mathbb{G}_1^{n_1} \times \cdots \times \mathbb{G}_{k-1}^{n_{k-1}}$. The vector $(\mathbf{z}', 0)$ is defined analogously. We use sub-brackets, e.g., $((\mathbf{z}_L, \mathbf{z}_R), \mathbf{x}_k, \gamma_2)$, to emphasize that a sub-vector $(\mathbf{z}_L, \mathbf{z}_R)$ takes values in $\mathbb{Z}_q^{(n_0+r_1)} \times \mathbb{G}_1^{n_1} \times \cdots \times \mathbb{G}_{k-1}^{n_{k-1}}$.

The compression mechanism is a generalization of the compression mechanism of [AC20]. It is described in Protocol 2 and its main properties are summarized in Theorem 2. Recall that we consider commitment schemes of the following form $\mathrm{C{\scriptstyle OM}} = (\mathrm{C{\scriptstyle OM}}_1, \mathrm{C{\scriptstyle OM}}_2)$, where

$$\begin{aligned} &\mathrm{C{\scriptstyle OM}}_1 : \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \cdots \times \mathbb{G}_{k-1}^{n_{k-1}} \times \mathbb{Z}_q^{r_1} \to \mathbb{G}_{C_1}, \\ &\mathrm{C{\scriptstyle OM}}_2 : \mathbb{G}_k^{n_k} \times \mathbb{Z}_q^{r_2} \to \mathbb{G}_{C_2}, \end{aligned} \tag{7}$$

and $\mathrm{C{\scriptstyle OM}}_1$ is compact. Note that we only apply the compression on the part of the commitment scheme that is compact, i.e., not on the $\mathbb{G}_k$ part $\mathrm{C{\scriptstyle OM}}_2$.

Theorem 2 (Compression Mechanism). *Let $n_i \in \mathbb{Z}_{>0}$ be even for all $0 \leq i \leq k$. Then $\mathit{\Pi}_2$ is a 3-move protocol for relation R_1. It is perfectly complete and unconditionally 3-special sound. Moreover, the communication costs are:*

Protocol 2 Compression Mechanism $\mathit{\Pi}_2$ for relation R_1.

$\textsc{Input}\,(Q, g, y_2; \mathbf{z}, \mathbf{x}_k, \gamma_2)$

$$\mathbf{z} = (\mathbf{x}_0, \gamma_1, \mathbf{x}_1, \ldots, \mathbf{x}_{k-1})$$
$$g = (g_1, g_2) \in \mathrm{Hom}(\mathbb{G}_S, \mathbb{H})$$
$$Q = (Q_1, Q_2) \in \mathbb{G}_{C_1} \times \mathbb{G}_{C_2}$$
$$Q_1 = \textsc{Com}_1(\mathbf{z}, g_1(\mathbf{x}))$$
$$Q_2 = \textsc{Com}_2(\mathbf{x}_k, \gamma_2)$$
$$y_2 = g_2(\mathbf{x}) \in \mathbb{G}_k \times \mathbb{G}_h$$

Prover		Verifier
$(a_1, a_2) = g\left((0, \mathbf{z}_L), 0\right)$ $A = \textsc{Com}_1\left((0, \mathbf{z}_L), a_1\right)$ $(b_1, b_2) = g\left((\mathbf{z}_R, 0), 0\right)$ $B = \textsc{Com}_1\left((\mathbf{z}_R, 0), b_1\right)$	$\xrightarrow{A, B, a_2, b_2}$	
		$c \leftarrow_R \mathbb{Z}_q$
	$\xleftarrow{c}$	
$\mathbf{z}' = \mathbf{z}_L + c\mathbf{z}_R$		
	$\xrightarrow{\mathbf{z}', \mathbf{x}_k, \gamma_2}$	$(d_1, d_2) = g\left((c\mathbf{z}', \mathbf{z}'), c\mathbf{x}_k\right)$ $\textsc{Com}_1\left((c\mathbf{z}', \mathbf{z}'), d_1\right) \overset{?}{=} A + cQ_1 + c^2 B$ $\textsc{Com}_2\left(\mathbf{x}_k, \gamma_2\right) \overset{?}{=} Q_2$ $d_2 \overset{?}{=} a_2 + cy_2 + c^2 b_2$

- $\mathcal{P} \to \mathcal{V}$: 2 *elements of* $\mathbb{G}_{C_1}$, 2 *elements of* $\mathbb{G}_h$, $n_i/2$ *elements of* $\mathbb{G}_i$ *for all* $1 \le i \le k-1$, $n_k + 2$ *elements of* $\mathbb{G}_k$ *and* $n_0/2 + r_2$ *elements of* $\mathbb{Z}_q$.
- $\mathcal{V} \to \mathcal{P}$: 1 *element of* $\mathbb{Z}_q$.

The proof of Theorem 2 is almost identical to the proofs of [AC20, Theorem 2] and [ACF20, Theorem 2].

Proof. **Completeness** follows directly.

Special Soundness: We show that the protocol is 3-special sound, i.e., there exists an efficient algorithm that, on input three accepting transcripts, computes a witness for relation R_1.

Let $(A, B, a_2, b_2; c_1; \mathbf{z}_1, \mathbf{x}_{k,1}, \gamma_{2,1})$, $(A, B, a_2, b_2; c_2; \mathbf{z}_2, \mathbf{x}_{k,2}, \gamma_{2,2})$ and $(A, B, a_2, b_2; c_3; \mathbf{z}_3, \mathbf{x}_{k,3}, \gamma_{2,3})$ be three accepting transcripts for distinct challenges $c_1, c_2, c_3 \in \mathbb{Z}_q$ and with common first message (A, B, a_2, b_2). Let $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{Z}_q$ be such that

$$\begin{pmatrix} 1 & 1 & 1 \\ c_1 & c_2 & c_3 \\ c_1^2 & c_2^2 & c_3^2 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$

Note that, since the challenges are distinct, this Vandermonde matrix is invertible and a solution to this equation exists. We define $\bar{\mathbf{z}} = \sum_{i=1}^{3} \alpha_i (c_i \mathbf{z}_i, \mathbf{z}_i)$, $\bar{\mathbf{x}}_k = \sum_{i=1}^{3} \alpha_i c_i \mathbf{x}_{k,i}$ and $\bar{\gamma}_2 = \sum_{i=1}^{3} \alpha_i c_i \gamma_{2,i}$. Since $\textsc{Com}_1$ and $\textsc{Com}_2$ are homomorphisms, it is straightforward to see that $\textsc{Com}_1(\bar{\mathbf{z}}, g_1(\bar{\mathbf{z}}, \bar{\mathbf{x}}_k)) = Q_1$, $\textsc{Com}_2(\bar{\mathbf{x}}_k, \bar{\gamma}_2) = Q_2$ and $g_2(\bar{\mathbf{z}}, \bar{\mathbf{x}}_k) = y_2$. Hence, $(\bar{\mathbf{z}}, \bar{\mathbf{x}}_k, \bar{\gamma}_2)$ is a witness for relation R_1, which completes the proof.

□

4.4 Composition of the Protocols

We observe that the final message $(\mathbf{z}', \mathbf{x}_k, \gamma_2)$ of the compression mechanism is a witness for exactly the same relation R_1, but now with public statement (Q', g', y_2') where $Q' := (A + cQ_1 + c^2 B, Q_2)$, $g'(\mathbf{z}', \mathbf{x}_k) := g\left((c'\mathbf{z}', \mathbf{z}'), c\mathbf{x}_k\right)$ and $y_2' = a_2 + cy_2 + c^2 b_2$. In other words, the dimensions of the $\mathbb{Z}_q, \mathbb{G}_1, \ldots, \mathbb{G}_{k-1}$

components of the witness halved. Hence, the compression mechanism is *composable* with another appropriate instantiation of the *same* compression mechanism. This composition can be applied to further reduce the dimensions of the witness and thereby the communication complexity. Recall that, in this composition the final message $(\mathbf{z}', \mathbf{x}_k, \gamma_2)$ in protocol Π_2 is replaced by another instantiation of Π_2. Altogether we see that we can compose the following compressed Σ-protocol:

$$\Pi_c = \underbrace{\Pi_2 \diamond \cdots \diamond \Pi_2}_{\mu \text{ times}} \diamond \Pi_1 \diamond \Pi_0, \tag{8}$$

where $\mu = \lceil \log_2 (\max_{1 \le i \le k-1}(n_i)) \rceil$. The properties of composition Π_c are summarized in Theorem 3. We say Π_c is a Compressed Σ-Protocol for relation R.

Theorem 3 (Compressed Σ-Protocol for Opening Homomorphisms). *Π_c is a $(2\mu + 3)$-move protocol for relation R, where $\mu = \lceil \log_2 (\max(n_0 + r_1, n_1, \ldots, n_{k-1})) \rceil$. It is perfectly complete, special honest-verifier zero-knowledge and computationally $(2, 2, 3, \ldots, 3)$-special sound, under the assumption that the commitment scheme is binding. Moreover, the communication costs are:*

- *$\mathcal{P} \to \mathcal{V}$: $2\mu + 1$ elements of $\mathbb{G}_{C_1}$, 1 of $\mathbb{G}_{C_2}$, $2\mu + 1$ of $\mathbb{G}_H$, 2 of $\mathbb{G}_i$ for all $1 \le i \le k-1$, $n_k + 2\mu + 1$ of $\mathbb{G}_k$, and $r_2 + 1$ of $\mathbb{Z}_q$.*
- *$\mathcal{V} \to \mathcal{P}$: $\mu + 2$ elements of $\mathbb{Z}_q$.*

4.5 Amortization

Standard amortization techniques apply to the basic Σ-protocol Π_0 for relation R, and thereby also to compressed Σ-protocol Π_c. These amortization techniques allow a prover to open *many* homomorphisms on *one* commitment, or *one* homomorphism on *many* commitments, without increasing the communication costs from the prover to the verifier. For details we refer to [AC20, §5.1].

These amortization techniques allow us to restrict ourselves to homomorphisms with the codomain $\mathbb{Z}_q \times \mathbb{G}_1 \times \cdots \times \mathbb{G}_k \times \mathbb{G}_h$. Namely opening a homomorphism $h : \mathbb{G}_S \to \mathbb{Z}_q^{s_0} \times \mathbb{G}_1^{s_1} \times \cdots \times \mathbb{G}_k^{s_k} \times \mathbb{G}_h$ is equivalent to opening $\max(s_i)$ homomorphisms with codomain $\mathbb{Z}_q \times \mathbb{G}_1 \times \cdots \times \mathbb{G}_k \times \mathbb{G}_h$.

5 Threshold Signature Schemes

In this section, we describe a threshold signature scheme (TSS), as an application of the compressed Σ-protocol Π_c for proving linear statements on committed vectors $\mathbf{x}$. Informally a k-out-of-n threshold signature can only be computed given k valid signatures issued by a k-subset of n players. We first describe the formal definition of a TSS. Subsequently, we give our construction based on the compressed Σ-protocol Π_c.

5.1 Definition and Security Model

In this work, we deviate from standard TSS definitions by aiming for a strictly stronger functionality. In standard TSS definitions [Sho00, Bol03], a trusted dealer generates a single public key and n private keys that are distributed amongst the n players. The private keys allow individual players to generate *partial* signatures on messages m. There is a public algorithm to aggregate k partial signatures into a threshold signature. The threshold signature can be verified with the public key generated by the trusted dealer.

By contrast, we define a TSS as an *extension* of a digital signature scheme. Our fundamental strengthening of the definitions of [Sho00, Bol03] and related works, is that the public and private keys are generated by the players locally. Public keys are published on a *bulletin board* and thereby publicly tied to the player's identities. This setup is thus *transparent* (called "bulletin board" in [BCG20] and formalized as $\mathcal{F}_{CA}$ in the UC framework [Can04]). The players can individually sign messages by using their private keys. The aggregation algorithm now takes as input k signatures, instead of partial signatures, to generate a threshold signature.

For simplicity we assume the threshold k to be fixed. We will explain later why our construction (trivially) satisfies some stronger properties.

Let us first give a definition for the basic building block of our TSS.

Definition 11 (Digital Signature). *A digital signature scheme consists of three algorithms:*

- KEYGEN *is a randomized key generation algorithm that outputs a public-private key-pair* $(\mathsf{pk}, \mathsf{sk})$*.*
- SIGN *is a (possibly randomized) signing algorithm. On input a message* $m \in \{0,1\}^*$ *and a secret key* sk*, it outputs a signature* $\sigma = \textsc{sign}(\mathsf{sk}, m)$*.*
- VERIFY *is a deterministic verification algorithm. On input a public key* pk*, a message* m *and a signature* σ*, it outputs either* accept *or* reject*.*

A signature scheme is *correct* if $\textsc{verify}\,(\mathsf{pk}, m, \textsc{sign}(\mathsf{sk}, m)) = \mathsf{accept}$ for all key-pairs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \textsc{keygen}$ and messages $m \in \{0,1\}^*$. If $\textsc{verify}(\mathsf{pk}, m, \sigma) = \mathsf{accept}$ we say that σ is a *valid* signature on message m. Moreover, an adversary that does not know the secret key sk should not be able to forge a valid signature. This security property is formally captured in the widely accepted definition *Existential Unforgeability under Chosen-Message Attacks* (EUF-CMA) [Bol03]. We assume digital signature schemes to be correct and EUF-CMA by definition.

Definition 12 (Threshold Signature). *A k-out-of-n threshold signature scheme (TSS) is a digital signature scheme* (KEYGEN, SIGN, VERIFY) *appended with two algorithms:*

- k-AGGREGATE *is a (possibly randomized) aggregation algorithm. On input n public keys* $(\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$*, k signatures* $(\sigma_i)_{i \in \mathcal{S}}$ *for a k-subset* $\mathcal{S} \in \{1, \ldots, n\}$ *and a message* $m \in \{0,1\}^*$*, it outputs a threshold signature* Σ*.*
- k-VERIFY *is a deterministic verification algorithm. On input n public keys* $(\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$*, a message m and a threshold signature* Σ*, it outputs either* accept *or* reject*.*

Let $\mathcal{S} \subset \{1, \ldots, n\}$ be some k-subset of indices and let $(\sigma)_{i \in \mathcal{S}}$ be signatures, such that $\textsc{verify}(\mathsf{pk}_i, m, \sigma_i) = \mathsf{accept}$, for all $i \in \mathcal{S}$, and for some message $m \in \{0,1\}^*$. Then a TSS is *correct*, if for all $(\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$, m, $\mathcal{S}$ and $(\sigma)_{i \in \mathcal{S}}$,

$$k\text{-}\textsc{verify}\Big((\mathsf{pk}_1, \ldots, \mathsf{pk}_n), m, k\text{-}\textsc{aggregate}\big(m, (\sigma_i)_{i \in \mathcal{S}}\big)\Big) = \mathsf{accept}.$$

If $k\text{-}\textsc{verify}\Big((\mathsf{pk}_1, \ldots, \mathsf{pk}_n), m, \Sigma\Big) = \mathsf{accept}$ we say that Σ is a valid threshold signature. Moreover, an adversary with at most $k-1$ valid signatures on a message m should not be able to construct a valid threshold signature. This *unforgeability* property can be formalized by the following security game. Consider an adversary that is allowed to choose a subset of $k-1$ indices $\mathcal{I} \subset \{1, \ldots, n\}$ and impose the values of the keys pk_i in this subset. Assume that all remaining keys pk_i were generated honestly from KEYGEN and therefore correspond to secret keys sk_i. The adversary is allowed to query polynomially many signatures $\sigma_i' = \textsc{sign}(sk_i, m')$ for arbitrary messages m'. The TSS is said to be *unforgeable*, if the adversary is incapable of producing a valid k-out-of-n threshold signature on some message m that has not been queried. We assume threshold signatures schemes to be robust and unforgeable by definition.

5.2 Our Threshold Signature Scheme

We follow a non-standard, but conceptually simple, approach for constructing a threshold signature scheme. The starting point of our TSS is a digital signature scheme (KEYGEN, SIGN, VERIFY) and the k-aggregation algorithm k-AGGREGATE simply produces a proof of knowledge of k valid signatures on a message m, i.e., a PoK for the following relation:

$$\begin{aligned} R_T = \big\{\, &(\mathsf{pk}_1, \ldots, \mathsf{pk}_n, m; \mathcal{S}, (\sigma_i)_{i \in \mathcal{S}}) : \\ &|\mathcal{S}| = k,\ \textsc{verify}(\mathsf{pk}_i, m, \sigma_i) = \mathsf{accept}\ \forall i \in \mathcal{S} \big\}. \end{aligned} \tag{9}$$

The obvious approach is to capture this relation by an arithmetic circuit, i.e., reduce it to a number of constraints defined over $\mathbb{Z}_q$, and apply a communication-efficient proof of knowledge for arithmetic circuit relations in a black-box manner. Although, for the appropriate choice of proof system, this approach would immediately result in a transparent TSS with sub-linear size threshold signatures, we have not encountered it in literature. The closest resemblance can be found in a recent unpublished work [BCG20] that considers a different TSS scenario.

A significant drawback of this *indirect* approach is that it relies on an inefficient reduction to arithmetic circuit relations. For this reason, we follow a *direct* approach avoiding these inefficient reductions.

We instantiate our TSS with the BLS signature scheme [BLS01] defined over a bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$. Let us now briefly recall the BLS signature scheme, instantiated in our n-player setting. All players i, $1 \leq i \leq n$, generate their own private key $u_i \in \mathbb{Z}_q$, and publish the associated public key $P_i = u_i H \in \mathbb{G}_2$. To sign a message $m \in \{0,1\}^*$, player i computes signature $\sigma_i = u_i \mathcal{H}(m) \in \mathbb{G}_1$. The public verification algorithm accepts a signature σ_i if

$$e(\sigma_i, H) = e(\mathcal{H}(m), P_i). \tag{10}$$

By the bilinearity of e, all honestly generated signatures are accepted. The unforgeability follows from the co-CDH assumption [BLS01]. Note that in [BLS01], where $\mathbb{G}_1 = \mathbb{G}_2$, this collapses to the “gap-group” assumption.

We will be using the commitment scheme from Definition 8:

$$\textsc{Com}_1 : \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{Z}_q \to \mathbb{G}_T,\ (\mathbf{x}_{\mathbb{Z}_q}, \mathbf{x}_{G_1}, \gamma) \mapsto h\gamma + \langle \mathbf{g}, \mathbf{x}_{\mathbb{Z}_q} \rangle + e(\mathbf{x}_{G_1}, \mathbf{H}).$$

This commitment scheme requires the slightly stronger DDH assumption in $\mathbb{G}_1$ to hold.

Instantiating relation R_T with the BLS signature scheme therefore results in the following relation,

$$R_{TSS} = \{(P_1, \dots, P_n, m; \mathcal{S}, (\sigma_i)_{i \in \mathcal{S}}) : |\mathcal{S}| = k,\ e(\sigma_i, H) = e(\mathcal{H}(m), P_i)\ \forall i \in \mathcal{S}\}.$$

The k-AGGREGATE algorithm simply computes a proof of knowledge for relation R_{TSS}. The main challenge is that the prover only knows k-out-of-n signatures. To handle this problem the k-out-of-n case is reduced to the n-out-of-n as follows. The k signatures are appended with $n-k$ signatures $\sigma_i = 0$ and a binary vector that allows the prover to eliminate the $n-k$ new and invalid signatures. The left hand side of the verification remains the same, while the right hand side is multiplied by corresponding coefficient of the binary vector. This approach results in a TSS with the desired properties. However, it requires the prover to prove a number of *non-linear* statements, i.e., that the committed binary vector is binary and contains at most $n-k$ zeros. Although this can be done efficiently, e.g., with the range proofs of [AC20], a recent result on *k-out-of-n proofs of partial knowledge* [ACF20] gives an even more efficient solution, that completely avoids non-linearities.

The proof of partial knowledge technique allows us to reduce relation R_{TSS} to a linear relation defined over the bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$. Let $p(X) = 1 + \sum_{j=1}^{n-k} a_j X^j \in \mathbb{Z}_q[X]$ be the unique polynomial of degree at most $n-k$ with $p(i) = 0$ for all $i \in \{1, \dots, n\} \backslash \mathcal{S}$. Note that this polynomial defines an $(n-k, n)$ secret sharing of 1, with shares $s_i = 0$ for all $i \notin \mathcal{S}$. The k-aggregator defines $\widetilde{\sigma}_i = p(i)\sigma_i$, where $\widetilde{\sigma}_i$ is understood to be equal to 0 for $i \notin \mathcal{S}$, i.e., the secret sharing defined by $p(X)$ *eliminates* the signatures $(\sigma_i)_{i \notin \mathcal{S}}$ that the k-aggregator does not know. Subsequently, the k-aggregator commits to

$$\mathbf{x} = (a_1, \dots, a_{n-k}, \widetilde{\sigma}_1, \dots, \widetilde{\sigma}_n) \in \mathbb{Z}_q^{n-k} \times \mathbb{G}_1^n.$$

Now note that the committed vector $\mathbf{x}$ satisfies $f_i(\mathbf{x}) = f_i(a_1, \dots, a_{n-k}, \widetilde{\sigma}_1, \dots \widetilde{\sigma}_n) = e(\mathcal{H}(m), P_i)$ for all $1 \leq i \leq n$, where

$$f_i : \mathbb{Z}_q^{n-k} \times \mathbb{G}_1^n \to \mathbb{G}_T, \quad \mathbf{x} \to e(\widetilde{\sigma}_i, H) - \sum_{j=1}^{n-k} a_j i^j e(\mathcal{H}(m), P_i). \tag{11}$$

Hence, by proving that the committed vector satisfies these relations, it follows that the k-aggregator knows a non-zero polynomial $p(X)$ of degree at most $n-k$ and group elements $\widetilde{\sigma}_1, \dots \widetilde{\sigma}_n \in \mathbb{G}_1$ such that $e(\widetilde{\sigma}_i, H) = p(i)e(\mathcal{H}(m), P_i)$ for all $1 \leq i \leq n$. Therefore, the k-aggregator must know valid signatures for all indices i with $p(i) \neq 0$, and since $p(X)$ is non-zero and of degree at most $n-k$ at least, k of its evaluations are non-zero. Because the mappings f_i are homomorphisms, the required proof of knowledge follows from an appropriate instantiation of compressed Σ-protocol Π_c. We apply the amortization techniques of Section 4.5 to prove all n relations of eq. (11) for essentially the price of one. Moreover, we apply the Fiat-Shamir transform to make protocol Π_c non-interactive. Altogether the threshold signature contains a commitment $P \in \mathbb{G}_T$ to the vector $\mathbf{x}$ together with a non-interactive proof of knowledge π of an opening of P that satisfies the aforementioned linear constraints. The k-AGGREGATE algorithm is summarized in Algorithm 3. The associated k-verification algorithm k-VERIFY simply runs the verifier of Π_c. Robustness of the resulting threshold signature follows immediately from the completeness of Π_c, and unforgeability follows from the soundness of Π_c. The properties of the TSS are summarized in Theorem 4. Note that our TSS has some additional properties not required by the definition of Section 5.1. For instance, since the proof of knowledge Π_c is special honest-verifier zero-knowledge, our threshold signatures hide the k-subset $\mathcal{S}$ of signers.

Algorithm 3 k-Aggregation Algorithm k-AGGREGATE

```
PUBLIC INPUT :     Public Keys $P_1, \dots, P_n \in \mathbb{G}_2$
                   Message $m \in \{0,1\}^*$
PRIVATE INPUT :    $k -$ Subset $\mathcal{S} \subset \{1, \dots, n\}$
                   Signatures $(\sigma_i)_{i \in \mathcal{S}} \in \mathbb{G}_1^k$

OUTPUT :           Threshold Signature $\Sigma = (\pi, P) \in \mathbb{Z}_q \times \mathbb{G}_1 \times \mathbb{G}_T^{4\lceil \log_2(n) \rceil + 3} \cup \{\perp\}$

1. If $\exists i \in \mathcal{S}$ such that $e(\sigma_i, H) \neq e(\mathcal{H}(m), P_i)$ output $\perp$ and abort.
2. Compute the unique polynomial $p(X) = 1 + \sum_{i=1}^{n-k} a_j X^j \in \mathbb{Z}_q[X]$ of degree at most $n-k$ such that $p(i) = 0$ for
   all $i \in \{1, \dots, n\} \backslash \mathcal{S}$.
3. Compute $\widetilde{\sigma}_i := p(i)\sigma_i$ for all $i \in \mathcal{S}$ and set $\widetilde{\sigma}_i = 0$ for all $i \notin \mathcal{S}$.
4. Let $\mathbf{x} = (a_1, \dots, a_{n-k}, \widetilde{\sigma}_1, \dots, \widetilde{\sigma}_n) \in \mathbb{Z}_q^{n-k} \times \mathbb{G}_1^n$ and compute commitment $P = \textsc{Com}_1(\mathbf{x}, \gamma) \in \mathbb{G}_T$ for $\gamma \in \mathbb{Z}_q$
   sampled uniformly at random.
5. Run the non-interactive variant of $\Pi_c$ to produce a proof $\pi$ attesting that the committed vector $\mathbf{x}$ satisfies
   $f_i(\mathbf{x}) = f_i(a_1, \dots, a_{n-k}, \widetilde{\sigma}_1, \dots \widetilde{\sigma}_n) = e(\mathcal{H}(m), P_i)$ for all $1 \leq i \leq n$, where $f_i$ are homomorphisms defined
   in Equation (11).
6. Output commitment $P$ and the non-interactive proof $\pi \in \mathbb{Z}_q \times \mathbb{G}_1 \times \mathbb{G}_T^{4\lceil \log_2(n) \rceil + 2}$.
```

Theorem 4 (Threshold Signature Scheme). *The k-out-of-n threshold signature scheme defined by the BLS signatures scheme [BLS01] appended with the algorithms (k-AGGREGATE, k-VERIFY) is robust and unforgeable. Moreover:*

- *A threshold signature contains exactly $4\lceil \log_2(n) \rceil + 3$ elements of $\mathbb{G}_T$, 1 element of $\mathbb{G}_1$ and 1 element of $\mathbb{Z}_q$.*
- *A threshold signature is zero-knowledge on the identities of the k signers.*
- *The threshold k can be chosen at aggregation time.*
- *It resists against an adaptive adversary which can replace the public keys of corrupted players.*

Proof. **Robustness** immediately follows from the completeness of Π_c.

Unforgeability. The proof is similar to the proof of [ACF20, Theorem 6]. From special soundness of Π_c (Theorem 3), it follows that there exists an efficient extractor χ that outputs a vector $\mathbf{x}' = (\mathbf{a}', S_1, \dots, S_n) \in \mathbb{Z}_q^{n-k} \times \mathbb{G}_1^n$ such that $f_i(\mathbf{x}) = e(\mathcal{H}(m), P_i)$ for all $1 \leq i \leq n$, where f_i are as in Equation (11). Let us denote

$p'(X) = 1 + \sum_{i=1}^{n-k} a'_j X^k \in \mathbb{Z}_q[X]$, then $\mathcal{S}' = \{i : p'(i) \neq 0\}$ has cardinality at least k. Moreover, it is easily seen that $p'(i)^{-1} S_i$ is a valid BLS signature on message m associated to public key P_i. Hence, an adversary capable of forging a threshold signature is also capable of computing k distinct valid signatures on m. Since the adversary is capable of corrupting at most $k-1$ players, this contradicts the unforgeability of the BLS signature scheme.

The remaining properties are trivially verified.

□

6 Generalized Circuit Zero-Knowledge Protocols

The Compressed Σ-Protocol of Section 4 allows a prover to open homomorphisms on committed vectors $\mathbf{x} \in \mathbb{G}_S$, i.e., it allows a prover to prove linear statements. In this section, we show how to handle non-linearities. The approach is again a generalization of that of [AC20], where it was shown how to linearize non-linearities in arithmetic circuit relations.

In this generalization we consider circuits $C : \mathbb{G}_S \to \mathbb{Z}_q^{s_0} \times \mathbb{G}_1^{s_1} \times \cdots \times \mathbb{G}_k^{s_k}$ and we aim to find a HVZK PoK for proving knowledge of a witness $\mathbf{x}$ such that $C(\mathbf{x}) = 0$, i.e., for the following relation:

$$R_{cs} = \{(C; \mathbf{x}) : C(\mathbf{x}) = 0\}. \tag{12}$$

Each wire of C corresponds to a variable that takes values in a group $W \in \{\mathbb{Z}_q, \mathbb{G}_1, \ldots, \mathbb{G}_k\}$. We assume all gates to have fan-in two and unbounded fan-out. The gates are either addition gates that add two elements of the same group, or bilinear gates mapping two group elements $a, b \in \mathbb{Z}_q \cup \mathbb{G}_1 \cup \cdots \cup \mathbb{G}_k$, not necessarily of the same group, to another group element $c \in \mathbb{Z}_q \cup \mathbb{G}_1 \cup \cdots \cup \mathbb{G}_k$. Note that these circuits are indeed generalizations of arithmetic circuits, where wires take values in $\mathbb{Z}_q$, and gates are either addition or multiplication gates.

Bilinear gates taking one constant and one variable input value are linear mappings. Hence, circuits C containing no bilinear gates with two variable inputs are handled directly by the techniques from Section 4. In this case, $C(\mathbf{x}) = f(\mathbf{x}) + a$ for a homomorphism f and a fixed constant a. A protocol for relation R_{cs} now goes as follows:

1. The prover commits to $\mathbf{x} \in \mathbb{G}_S$.
2. The prover and the verifier run Π_c to open the homomorphism f, i.e., the prover reveals a value y and proves that $f(\mathbf{x}) = y$.
3. The verifier checks that $y + a = 0$.

When C contains bilinear gates, we cannot express the circuit in the aforementioned manner. To handle these non-linearities, the prover appends the secret vector $\mathbf{x}$ with a vector aux containing auxiliary information, i.e., in the first step of the protocol the prover commits to the appended vector $(\mathbf{x}, \mathsf{aux})$. The approach is a generalization of the *secret sharing based* techniques from [AC20], in which non-linearities are linearized.

Let us define $\mathbf{c}$ to be the vector of wire values associated to the output wires of all the bilinear gates in $C(\mathbf{x})$. Note that $\mathbf{c}$ depends on the secret vector $\mathbf{x} \in \mathbb{G}_S$. Then, there exists a homomorphism f and a constant a, independent from $\mathbf{x}$, such that $C(\mathbf{x}) = f(\mathbf{x}, \mathbf{c}) + a$. A naive generalization of the above approach to arbitrary circuits is now obtained by taking $\mathsf{aux} = \mathbf{c}$. However, this approach does not guarantee that the committed vector $(\mathbf{x}, \mathbf{c})$ is of the appropriate form, i.e., that $\mathbf{c}$ indeed corresponds to the outputs of the bilinear gates.

To prove that the committed vector $(\mathbf{x}, \mathbf{c})$ is of the appropriate form we encode the inputs and outputs of the bilinear gates in polynomials $f \in A[X]$ where $A \in \{\mathbb{Z}_q, \mathbb{G}_1, \ldots, \mathbb{G}_k\}$. We first describe some properties of these polynomials.

6.1 Polynomials over Groups of Prime Order

The $\mathbb{Z}_q$-module structure of the groups $\mathbb{G}_i$ naturally extends to their polynomial rings, i.e., $\mathbb{G}_i[X]$ is a $\mathbb{Z}_q[X]$-module for all i, and the product $h(X)$ of two polynomials $f(X) = \sum_{i=0}^{n} a_i X^i \in \mathbb{Z}_q[X]$ and $g(X) =$

$\sum_{i=0}^{m} g_i X^i \in \mathbb{G}_i[X]$ is defined as follows

$$h(X) = f(X)g(X) := \sum_{i=0}^{n}\sum_{j=0}^{m}(a_i g_j)X^{i+j} \in \mathbb{G}_i[X].$$

Since $\mathbb{G}_i$ is a $\mathbb{Z}_q$-module, a polynomial $f = \sum_{i=0}^{n} a_i X^i \in \mathbb{G}_i[X]$ defines

$$f : \mathbb{Z}_q \to \mathbb{G}_i, \quad \rho \to f(\rho) = \sum_{i=0}^{n} a_i \rho^i,$$

called the "evaluation" mapping. Whereas every $\rho \in \mathbb{Z}_q$ defines a mapping:

$$F_\rho : \mathbb{G}_i[X] \to \mathbb{G}_i, \quad f = \sum_{i=0}^{n} a_i X^i \to f(\rho) = \sum_{i=0}^{n} a_i \rho^i,$$

called the "evaluation at ρ" mapping, which is linear.

A bilinear gate $\mathrm{Gate} : \mathrm{L} \times \mathrm{R} \to \mathrm{U}$ can be extended to act on polynomials in the following manner:

$$\mathrm{Gate} : \mathrm{L}[X] \times \mathrm{R}[X] \to \mathrm{U}[X], \ \left(\sum_{i=0}^{n} a_i X^i, \sum_{j=0}^{m} b_j X^j\right) \mapsto \sum_{i=0}^{n}\sum_{j=0}^{m} \mathrm{Gate}(a_i, b_j)X^{i+j}. \tag{13}$$

By the bilinearity of Gate it follows that this mapping commutes with polynomial evaluation, i.e., for all $\rho \in \mathbb{Z}_q$ it holds that $\mathrm{Gate}(f(\rho), g(\rho)) = \mathrm{Gate}(f, g)(\rho)$.

The following lemma shows that a non-zero polynomial f has at most $\deg(f)$ zeros. From this it follows that, for a fixed non-zero polynomial f and a random challenge c, the probability that $f(c) = 0$ is at most $\deg(f)/q$.

Lemma 2. *Let $f(X) \in A[X]$ be non-zero, for some $A \in \{\mathbb{Z}_q, \mathbb{G}_1, \ldots, \mathbb{G}_k\}$. Then $f(X)$ has at most $\deg(f)$ zeros.*

Proof. Recall that A has prime order q and let g be a generator of A. Then it is easily seen that $f(X) = f'(X)g$ for some polynomial $f'(X) \in \mathbb{Z}_q[X]$ with $\deg(f) = \deg(f')$. Moreover, since g is a generator of A, it holds that $f(a) = 0$ if and only if $f'(a) = 0$. The lemma now follows from the fact that a non-zero polynomial f' defined over a field has at most $\deg(f')$ zeros. □

The following lemma describes an approach for testing whether three polynomials $f(X)$, $g(X)$ and $h(X)$ satisfy a bilinear relation defined by Gate. More precisely, when the bilinear relation holds in a random evaluation point $c \in \mathbb{Z}_q$ then, with high probability, it holds for the polynomials $f(X)$, $g(X)$ and $h(X)$.

Lemma 3. *Let $f(X) \in \mathrm{L}[X]$, $g(X) \in \mathrm{R}[X]$ and $h(X) \in \mathrm{U}[X]$ with $\deg(f), \deg(g) \leq n$ and $\deg(h) \leq 2n$. Then, for $d \in \mathcal{C} \subset \mathbb{Z}_q$ sampled uniformly at random, it holds that*

$$\Pr\left(\mathrm{Gate}\left(f(d), g(d)\right) = h(d) \middle| \,\mathrm{Gate}\left(f(X), g(X)\right) \neq h(X)\right) \leq \frac{2n}{|\mathcal{C}|}.$$

Proof. The polynomial $h(X) - \mathrm{Gate}\left(f(X), g(X)\right) \in \mathrm{U}[X]$ has degree at most $2n$. The lemma now follows from Lemma 2. □

6.2 Linearization of Bilinear Gates

We are now ready to describe the linearization approach. To this end, let us assume that there exist ℓ different bilinear types of gates $\mathrm{Gate}_i : \mathrm{L}_i \times \mathrm{R}_i \to \mathrm{U}_i$, where $1 \leq i \leq \ell$. Moreover, for all i, we let m_i be the number of gates of type i in circuit C and, for a circuit evaluation $C(\mathbf{x})$, we let $\mathbf{a}_i \in \mathrm{L}_i^{m_i}$ and $\mathbf{b}_i \in \mathrm{R}_i^{m_i}$ be the vectors of left and right input values of these gates. Similarly, we let $\mathbf{c}_i \in \mathrm{U}_i^{m_i}$ be the vector of output values for the gates of type i.

The protocol now goes as follows. First, for each i, the prover samples two polynomials $f_i(X) \in \mathrm{L}_i[X]_{\leq m_i}$ and $g_i(X) \in \mathrm{R}_i[X]_{\leq m_i}$ of degree at most m_i uniformly at random under the condition that $f_i(j) = a_{i,j}$ and $g_i(j) = b_{i,j}$ for all $1 \leq j \leq m_j$. Note that these polynomials define *packed Shamir secret sharings* [Sha79] with (m_i+1)-reconstruction and 1-privacy of the vectors $\mathbf{a}_i$ and $\mathbf{b}_i$, i.e., the vectors $\mathbf{a}_i$ and $\mathbf{b}_i$ can be reconstructed from any m_i+1 evaluations of $f_i(X)$ and $g_i(X)$ and any single evaluation outside $\{1, \dots, m_i\}$ is independent from the vectors $\mathbf{a}_i$ and $\mathbf{b}_i$.

Second, the prover computes the polynomial $h_i(X) = \mathrm{Gate}_i\left(f_i(X), g_i(X)\right)$. By the *strong-multiplicativity* of Shamir's secret sharing scheme, it holds that $h_i(X) \in \mathrm{U}_i[X]$ defines a packed secret sharing of the vector $\mathbf{c}_i \in \mathrm{U}_i^{m_i}$ with $2m_i + 1$ reconstruction. More precisely, $h_i(X)$ is of degree at most $2m_i$ and $h_i(j) = c_{i,j}$ for all $1 \leq j \leq m_i$. Subsequently, the prover sends a commitment to the following secret vector to the verifier:

$$\mathbf{y} = \big(\mathbf{x}, f_1(0), g_1(0), h_1(0), \dots, h_1(2m_1), \dots, f_\ell(0), g_\ell(0), h_\ell(0), h_\ell(1), \dots, h_\ell(2m_\ell)\big).$$

The vector $\mathbf{y} = (\mathbf{x}, \mathsf{aux})$ contains the vector $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_\ell)$ of the output values of all bilinear gates as a sub-vector. Hence, all wires values can be expressed as the evaluation of some public homomorphism in $\mathbf{y}$ plus a public constant value. Moreover, the vector $\mathbf{y}$ contains $m_i + 1$ evaluations of $f_i(X)$ and $g_i(X)$ and $2m_i + 1$ evaluations of $h_i(X)$ for all i, i.e., it uniquely defines polynomials $f_i(X)$ and $g_i(X)$ of degree at most m_i and $h_i(X)$ of degree at most $2m_i + 1$. By the linearity of Lagrange interpolation it follows that, in addition to the wire values, all evaluations of the polynomials $f_i(X)$, $g_i(X)$ and $h_i(X)$ can be expressed as some homomorphism evaluated in $\mathbf{y}$ plus a constant value. These properties allow the prover to convince the verifier that the vector $\mathbf{y}$ is of the appropriate form by proving that certain linear constraints hold.

In the next step of the protocol, the verifier samples a random challenge $d \in \mathbb{Z}_q \setminus \{1, \dots, \max(m_i)\}$ uniformly at random and asks the prover to run protocol Π_c to open $C(\mathbf{x})$, $f_i(d)$, $g_i(d)$ and $h_i(d)$ for all $1 \leq i \leq \ell$. Note that all these values correspond to homomorphisms evaluated in the committed vector $\mathbf{y} = (\mathbf{x}, \mathsf{aux})$. To further reduce the communication costs, the amortization techniques mentioned in 4.5 are applied. Finally, the verifier verifies that $C(\mathbf{x}) = 0$ and that $\mathrm{Gate}\left(f_i(d), g_i(d)\right) = h_i(d)$ for all i. By Lemma 3 this final verification implies that $\mathrm{Gate}\left(f_i(X), g_i(X)\right) = h_i(X)$, and therefore that $\mathrm{Gate}\left(a_{i,j}, b_{i,j}\right) = c_{i,j}$ for all j, with probability at least $1 - 2m_i/(q - m_i)$. If m_i is polynomial and q is exponential in the security parameter, this probability is overwhelming. The protocol is SHVZK because the polynomials $f_i(X)$, $g_i(X)$ and $h_i(X)$ define secret sharings with 1-privacy, and because protocol Π_c is SHVZK. For a more detailed discussion we refer to [AC20] in which this approach is restricted to arithmetic circuits.

The resulting protocol, denoted by Π_{cs}, is described in Protocol 4. The protocol is perfectly complete, special honest-verifier zero-knowledge and computationally $(k_1, \dots, k_\mu)$-special sound under the assumption that the commitment scheme is binding. The precise properties of the protocol, such as the values of $k_1, \dots, k_\mu$ and the exact communication costs, depend on the commitment scheme and on the bilinear gates that are considered. For this reason, we will only specify these precise properties for the concrete example of bilinear group arithmetic circuits in Section 6.3.

6.3 Bilinear Group Arithmetic Circuits

In this section, we consider the set of bilinear circuits C defined over a bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$, i.e., circuits of the following form:

$$C : \mathbb{Z}_q^{n_0} \times \mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2} \times \mathbb{G}_T^{n_T} \to \mathbb{Z}_q^{s_0} \times \mathbb{G}_1^{s_1} \times \mathbb{G}_2^{s_2} \times \mathbb{G}_T^{s_T}.$$

Protocol 4 Circuit Satisfiability Argument $\mathit{\Pi}_{cs}$ for Relation R_{cs}
The polynomials f_i and g_i are sampled uniformly at random such that their evaluations in $1, \dots, m_i \in \mathbb{Z}_q$ coincide with the left and, respectively, right inputs of the m_i type i gates of C evaluated in $\mathbf{x}$.

$$\textsc{Input}(C; \mathbf{x})$$

$$C : \mathbb{G}_S \to \mathbb{Z}_q^{s_0} \times \mathbb{G}_1^{s_1} \times \cdots \times \mathbb{G}_k^{s_k}$$
$$C(\mathbf{x}) = 0$$

Prover | Verifier

$$\gamma \leftarrow_R \mathbb{Z}_q^r$$
$$f_i \leftarrow_R \mathrm{L}_i[X]_{\le m_i}$$
$$g_i \leftarrow_R \mathrm{R}_i[X]_{\le m_i}$$
$$h_i(X) := \mathrm{Gate}_i(f_i(X), g_i(X))$$
$$\mathbf{y} = (\mathbf{x}, f_1(0), g_1(0), h_1(0),$$
$$h_1(1), \dots, h_1(2m_1)$$
$$\dots$$
$$f_\ell(0), g_\ell(0), h_\ell(0),$$
$$h_\ell(1), \dots, h_\ell(2m_\ell))$$

$$\xrightarrow{\textsc{Com}(\mathbf{y}, \gamma)}$$

$$d \leftarrow_R \mathbb{Z}_q \setminus \{1, \dots, \max(m_i)\}$$

$$\alpha_i = f_i(d) \qquad \xleftarrow{d}$$
$$\beta_i = g_i(d)$$
$$\gamma_i = h_i(d) \qquad \xrightarrow{\alpha_1, \beta_1, \gamma_1, \dots, \alpha_\ell, \beta_\ell, \gamma_\ell} \qquad \gamma_i \overset{?}{=} \mathrm{Gate}_i(\alpha_i, \beta_i), \ \forall\ 1 \le i \le \ell$$

Run $\mathit{\Pi}_c$ to prove that $\mathbf{y} = (\mathbf{x}, \mathsf{aux})$ satisfies
$C(\mathbf{x}) = 0, f_i(d) = \alpha_i, g_i(d) = \beta_i$
$h_i(d) = \gamma_i, \ \forall\ 1 \le i \le \ell$

These circuits are also called *bilinear group arithmetic circuits* [LMR19] and they are composed of addition gates and the following 5 types of bilinear gates:

$$\begin{aligned}
\mathrm{Gate}_0 &: \mathbb{Z}_q \ \times \mathbb{Z}_q \to \mathbb{Z}_q, && (a, b) \to ab, \\
\mathrm{Gate}_1 &: \mathbb{G}_1 \ \times \mathbb{Z}_q \to \mathbb{G}_1, && (g, a) \to ga, \\
\mathrm{Gate}_2 &: \mathbb{G}_2 \ \times \mathbb{Z}_q \to \mathbb{G}_2, && (h, a) \to ha, \\
\mathrm{Gate}_3 &: \mathbb{G}_T \ \times \mathbb{Z}_q \to \mathbb{G}_T, && (k, a) \to ka, \\
\mathrm{Gate}_4 &: \mathbb{G}_1 \ \times \mathbb{G}_2 \to \mathbb{G}_T, && (g, h) \to e(g, h).
\end{aligned} \tag{14}$$

Instantiating protocol $\mathit{\Pi}_{cs}$ for bilinear circuits and with the commitment scheme of Equation (3) results in a protocol $\mathit{\Pi}_{bi}$ for relation $R_{bi} = \{(C; \mathbf{x}) : C(\mathbf{x}) = 0, C \text{ is a bilinear circuit}\}$. This relation considers bilinear circuits C for which we let m_i denote the number of gates of type i for $0 \le i \le 4$. The variables m_i only count the bilinear gates that take two variable input values, the ones taking one constant input are linear and therefore handled directly by protocol $\mathit{\Pi}_c$. In the first step of this protocol instantiation, the prover commits to a vector

$$\mathbf{y} = (\mathbf{x}, \mathsf{aux}) \in \mathbb{Z}_q^{n_0+2m_0+6} \times \mathbb{G}_1^{n_1+2m_1+3} \times \mathbb{G}_1^{n_2+2m_2+3} \times \mathbb{G}_T^{n_T+2m_3+2m_4+3}.$$

For ease of notation we define the following parameters:

$$\begin{aligned}
m &:= \max(m_i), \quad s := \max(s_0 + 6, s_1 + 3, s_2 + 3, s_T + 3), \\
N &:= \max\left(n_0 + 2m_0 + 7, n_1 + 2m_1 + 3, n_2 + 2m_2 + 3\right), \\
N_T &:= n_T + 2m_3 + 2m_4 + 3.
\end{aligned}$$

Note that we make a distinction between the $(\mathbb{Z}_q, \mathbb{G}_1, \mathbb{G}_2)$-part, for which the commitment scheme is compact, and the $\mathbb{G}_T$-part of the vector $\mathbf{y}$. The properties of Protocol Π_{bi} are now summarized in the following theorem.

Theorem 5 (Circuit Zero-Knowledge Protocol for Bilinear Circuits). *Π_{bi} is a $(2\mu+7)$-move protocol for circuit relation R_{bi}, where $\mu = \lceil \log_2(N) \rceil$. It is perfectly complete, special honest-verifier zero-knowledge, under the DDH assumption in $\mathbb{G}_T$, and computationally $(2m+1, s, 2, 2, k_1, \dots, k_\mu)$-special sound, under the SXDH assumption, where $k_i = 3$ for all $1 \leq i \leq \mu$. Moreover, the communication costs are:*

- *$\mathcal{P} \rightarrow \mathcal{V}$: $6\lceil \log_2(N) \rceil + 3N_T + 9$ elements of $\mathbb{G}_T$, 5 elements of $\mathbb{G}_1$, 5 elements of $\mathbb{G}_2$ and 9 elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \rightarrow \mathcal{P}$: $\lceil \log_2(N) \rceil + 4$ elements of $\mathbb{Z}_q$.*

6.4 Improved Communication Efficiency for El Gamal Based Commitments

The basic Σ-protocol Π_0 of Section 4.1, for opening homomorphisms $f : \mathbb{G}_S \rightarrow \mathbb{H}$, follows the generic design for q-one-way group homomorphisms[9] [Cra96, CD98]. Similarly, the compression mechanism is generally applicable to a wide-class of relations captured by (structured) q-one-way group homomorphisms.[10] However, for some instantiations of the commitment scheme $\textsc{Com}$ this generic approach is sub-optimal as it leads to unnecessarily high communication costs. This is the case for the El Gamal based commitment scheme of Definition 10,

$$\textsc{Com}_2 : \mathbb{G}_T^{n_T} \times \mathbb{Z}_q \rightarrow \mathbb{G}_T^{n_T+1}, \quad (\mathbf{x}, \gamma) \mapsto \begin{pmatrix} h\gamma \\ \mathbf{g}\gamma + \mathbf{x} \end{pmatrix},$$

and for the commitment scheme $\textsc{Com} = (\textsc{Com}_1, \textsc{Com}_2)$ used by protocol Π_{bi} of Theorem 5. Here, we describe a more efficient approach tailored to the commitment scheme $\textsc{Com}_2$ and explain how the reduced communication costs translate to a reduction of the communication costs of protocol Π_{bi} for bilinear circuit relations.

The main observation is that to open a $\textsc{Com}_2$-commitment $P = (P_1, P_2) \in \mathbb{G}_T \times \mathbb{G}_T^{n_T}$, a prover merely has to reveal a $\gamma \in \mathbb{Z}_q$ such that $h\gamma = P_1$. The committed vector $\mathbf{x} \in \mathbb{G}_T^{n_T}$ can be computed from the commitment P and the (partial) opening γ, i.e., $\mathbf{x} = P_2 - \mathbf{g}\gamma$. Hence, proving knowledge of a commitment opening is equivalent to proving knowledge of a discrete logarithm (in base h). The natural Σ-protocol for the latter problem is much more efficient than the one for the former problem. More precisely, its communication costs are independent of the dimension n_T of committed vectors. A straightforward extension of this protocol allows a prover to prove that the committed vector satisfies a linear relation captured by a homomorphism $f : \mathbb{G}_T^{n_T} \rightarrow \mathbb{H}$.

The resulting protocol, denoted by Π_{EG}, is a protocol for the following relation:

$$R_{EG} = \left\{ \left(P \in \mathbb{G}_T^{n_T+1}, f \in \mathrm{Hom}(\mathbb{G}_T^{n_T}, \mathbb{H}), y \in \mathbb{H}; \mathbf{x} \in \mathbb{G}_T^{n_T}, \gamma \in \mathbb{Z}_q\right) : P = \textsc{Com}_2(\mathbf{x}, \gamma), f(\mathbf{x}) = y \right\}. \quad (15)$$

It is described in Protocol 5 and its properties are summarized in Theorem 6.

Theorem 6 (Σ-Protocol for El Gamal Based Commitments). *Π_{EG} is a Σ-protocol for relation R_{EG}. It is perfectly complete, special honest-verifier zero-knowledge and unconditionally special sound. Moreover, the communication costs are:*

- *$\mathcal{P} \rightarrow \mathcal{V}$: 1 element of $\mathbb{G}_T$, 1 element of $\mathbb{H}$, 1 element of $\mathbb{Z}_q$.*
- *$\mathcal{V} \rightarrow \mathcal{P}$: 1 element of $\mathbb{Z}_q$.*

Proof. **Completeness** follows directly.

Special Honest-Verifier Zero-Knowledge (SHVZK): Upon receiving a random challenge $c \in \mathbb{Z}_q$ a simulator proceeds as follows. The simulator samples $\phi \in \mathbb{Z}_q$ uniformly at random and computes $A = h\phi - cP_1$

[9] Here, applied to the homomorphism $\mathbb{G}_S \times \mathbb{Z}_q^r \rightarrow \mathbb{G}_C \times \mathbb{H}$, $(\mathbf{x}, \gamma) \mapsto (\textsc{Com}(x, \gamma), f(\mathbf{x}))$

[10] See [ACF20] for a general view on the compression mechanism.

and $t = f(cP_2 - \mathbf{g}\phi) - cy$. It is easily seen that the transcript (A, t, c, ϕ) is accepting and that simulated transcripts follow exactly the same distribution as transcripts between an honest prover and an honest verifier.

Special Soundness: We show that there exists an efficient algorithm, that on input two accepting transcripts, computes a witness for relation R_{EG}. Let (A, t, c, ϕ) and (A, t, c', ϕ') be accepting transcripts, for challenges $c \neq c'$ and with common first message (A, t). We define $\bar{\phi} = (\phi - \phi')/(c - c') \in \mathbb{Z}_q$ and $\bar{\mathbf{z}} = P_2 - \mathbf{g}\bar{\phi} \in \mathbb{G}_T^{n_T}$. Then it is easily verified that $\textsc{Com}_2(\bar{z}, \bar{\phi}) = P$ and that $f(\bar{\mathbf{z}}) = y$. Hence, $(\bar{\mathbf{z}}, \bar{\phi})$ is a witness for relation R_{EG}, which completes the proof. □

Protocol 5 *Σ*-protocol $\mathit{\Pi}_{EG}$ for relation R_{EG}
Σ-protocol for opening a homomorphism on a committed $\mathbb{G}_T$ vector.

$\textsc{Input}(P, f, y; \mathbf{x}, \gamma)$

$$P = (P_1, P_2) = \textsc{Com}(\mathbf{x}, \gamma) \in \mathbb{G}_T \times \mathbb{G}_T^{n_T}$$
$$y = f(\mathbf{x}) \in \mathbb{H}$$

$$\begin{array}{ccc} \text{Prover} & & \text{Verifier} \\ \rho \leftarrow_R \mathbb{Z}_q & & \\ A = h\rho & & \\ t = f(-\mathbf{g}\rho) & \xrightarrow{A,t} & \\ & & c \leftarrow_R \mathbb{Z}_q \\ & \xleftarrow{c} & \\ \phi = c\gamma + \rho & & \\ & \xrightarrow{\phi} & \mathbf{z} = cP_2 - \mathbf{g}\phi \\ & & h\phi \overset{?}{=} A + cP_1 \\ & & f(\mathbf{z}) \overset{?}{=} cy + t \end{array}$$

Theorem 6 shows that the communication costs of $\mathit{\Pi}_{EG}$ are indeed independent of n_T. By contrast, following the general design for q-one-way homomorphism would result in communication cost, from prover to verifier, of $2n_T+1$ G_T-elements, 1 $\mathbb{H}$-element and 1 $\mathbb{Z}_q$-element. This improvement can directly be inherited by the *compressed Σ*-protocols that use commitment scheme $\textsc{Com}_2$. For instance the communication costs of protocol $\mathit{\Pi}_{bi}$ can be reduced by $2n_T$ elements by incorporating this improved *Σ*-protocol. We denote the resulting protocol by $\mathit{\Pi}'_{bi}$ and summarize its properties in Theorem 7.

Theorem 7 (Improved ZK Protocol for Bilinear Circuits). *$\mathit{\Pi}'_{bi}$ is a $(2\mu+7)$-move protocol for circuit relation R_{bi}, where $\mu = \lceil \log_2(N) \rceil$. It is perfectly complete, special honest-verifier zero-knowledge, under the DDH assumption in $\mathbb{G}_T$, and computationally $(2m+1, s, 2, 2, k_1, \ldots, k_\mu)$ − special sound, under the SXDH assumption, where $k_i = 3$ for all $1 \leq i \leq \mu$. Moreover, the communication costs are:*

- $\mathcal{P} \rightarrow \mathcal{V}$: $6\lceil \log_2(N) \rceil + N_T + 9$ *elements of* $\mathbb{G}_T$, 5 *of* $\mathbb{G}_1$, 5 *of* $\mathbb{G}_2$ *and* 9 *of* $\mathbb{Z}_q$.
- $\mathcal{V} \rightarrow \mathcal{P}$: $\lceil \log_2(N) \rceil + 4$ *elements of* $\mathbb{Z}_q$.

6.5 Comparison of the Communication Costs

In this section, we compare the communication costs of our protocol $\mathit{\Pi}_{bi}$ to the bilinear circuit ZK protocol of [LMR19]. We note that, a rigorous comparison is difficult, for the following two reasons. First, we consider arbitrary bilinear circuits, whereas they assume certain structural properties, and therefore their result does not apply to the general bilinear circuit model, but only to a more limited class of circuits.Second, we consider a strictly stronger scenario in which the prover proves that the *committed* input values satisfy some bilinear relation, instead of merely proving knowledge of a satisfying input vector without being committed to this input vector. This difference explains why their communications costs are independent of the input dimensions n_0, n_1 and n_2.

Despite these two aspects, showing that we consider a stronger application scenario, it is interesting to note that our communication costs are smaller in certain parameter regimes. From Theorem 7 it follows that our Protocol Π_{bi} requires the prover to send a total of

$$6 \lceil \log_2 (N) \rceil + N_T + 28$$

elements (group and field elements) to the verifier, i.e., the communication costs associated to the $(\mathbb{Z}_q, \mathbb{G}_1, \mathbb{G}_2)$-part are logarithmic and the communications costs associated to the $\mathbb{G}_T$-part are linear. By contrast, the protocol of [LMR19] results in a total communication costs of

$$16 \log_2 (\ell_{mix}) + 3n_T + 71$$

elements, where $\ell_{mix} = 2m'_0 + m'_1 + m'_2 + n_T m'_3 + m'_4$. Here, the variable m'_i counts all gates of type i, including the ones taking a constant input value, i.e., $m'_i \geq m_i$. Hence, we have reduced the constant of the logarithmic part from 16 down to 6, and the constant of the linear part from 3 down to 1. However, when comparing the linear parts of the communication complexity, we note that there exist bilinear circuits for which $3n_T < N_T = n_T + 2m_3 + 2m_4 + 3$, e.g., circuits with $n_T = 0$ and $m_4 > 0$. Therefore, depending on the bilinear circuit our *linear* communication costs can be larger. This can partially be explained by the fact that Lai et al. [LMR19] make structural assumptions on the bilinear circuit. For instance, they assume that only input and output wires can take values in $\mathbb{G}_T$, whereas our protocol works for arbitrary bilinear circuits.

Nevertheless, as opposed to general bilinear circuits, there are specific quadratic inner-product relations for which the approach of Lai et al. [LMR19] can result in communication costs lower than those obtained by applying our generic approach. These relations exploit the fact that their approach reduces bilinear circuit relations to sets of inner-product constraints. These techniques are further improved in Bünz et al. [BMMV19], who merely focus on communication-efficient protocols for quadratic inner-product relations. By contrast, for the example of threshold signature schemes, which only rely on linear circuits, application of the latter approach would result in unnecessary overhead as compared to our compressed Σ-protocol approach. In the next section, we show that are techniques are easily combined with these inner-product arguments leveraging the benefits of both approaches.

7 Proving Knowledge of Signed Messages Satisfying a Public Predicate

In this section, we consider a functionality more general than that of threshold signature schemes. Namely, proving knowledge of a set of signed messages $m_1, \dots, m_n$ satisfying a public predicate $F(m_1, \dots, m_n) = 0$. Moreover, we allow the prover to choose $n - k$ input messages m_i freely, i.e., the prover is only required to prove knowledge of k-out-of-n valid signatures. This is precisely the building block used in [Ram20] for communication-efficient consensus.

More precisely, we aim to construct a proof-of-knowledge for relation

$$\begin{aligned} R_{\mathsf{pred}} = \big\{ \, &(\mathsf{pk}_1, \dots, \mathsf{pk}_n, \mathsf{tag}; m_1, \dots, m_n; \mathcal{S}, (\sigma_i)_{i \in \mathcal{S}}) : \\ &F(m_1, \dots, m_n) = 0, \ |\mathcal{S}| = k, \\ &\textsc{verify}(\mathsf{pk}_i, (m_i, \mathsf{tag}), \sigma_i) = \mathsf{accept} \ \forall i \in \mathcal{S} \big\}. \end{aligned} \tag{16}$$

The public statement the public-keys of the n players and a tag. The tag is a unique identifier that, for example, ensures that a dishonest player can not reuse signatures received in previous rounds of the protocol. Signatures σ_i on the pair (m_i, σ_i) can be verified by running the algorithm Verify.

The main contribution of [Ram20] stems from their idea of applying a communication-efficient PoK for relation R_{pred}; zero-knowledge is not required in their application. Black-box application of such a PoK suffices, but typically requires relation R_{pred} to be captured by an arithmetic circuit. As stated before, this generic black-box approach results in large arithmetic circuits with sub-optimal concrete efficiency. Using our techniques, we avoid this cumbersome transformation by *directly* construction a PoK for relation R_{pred}. However, the required functionality does not follow straightforwardly and some adaptations are required. We demonstrate the *plug-and-play* nature of compressed Σ-protocol theory by composing the different elements of this theory with a novel building block to construct the required protocol.

7.1 Communication-Efficient Protocol

Let us now describe our approach. By assuming that the input messages m_i are elements of $\mathbb{Z}_q$ and that F can be described by an arithmetic circuit, proving that the constraint $F(m_1, \dots, m_n) = 0$ is satisfied follows directly from the techniques from Section 6. Hence, for simplicity, we consider the relation

$$\begin{aligned} R'_{\mathsf{pred}} = \big\{ &(\mathsf{pk}_1, \dots, \mathsf{pk}_n, \mathsf{tag}; m_1, \dots, m_n; \mathcal{S}, (\sigma_i)_{i \in \mathcal{S}}) : \\ &|\mathcal{S}| = k,\ \textsc{verify}(\mathsf{pk}_i, (m_i, \mathsf{tag}), \sigma_i) = \mathsf{accept}\ \forall i \in \mathcal{S} \big\}. \end{aligned} \tag{17}$$

This relation is very similar to the threshold signature relation R_T of Equation 9. However, there is one important difference that requires us to make certain protocol adaptations. Namely, in the TSS relation all players sign the *same* public message. Here, the players sign secret and possibly different messages $m_1, \dots, m_n$.

The BLS verification algorithm involves the evaluation of a hash function $H(m_i)$. Because the messages in relation R'_{pred} are secret, a direct application of our circuit ZK approach to BLS signatures requires the cumbersome and inefficient approach of expressing the hash fucntion as an arithmetic circuit. Recall that, the protocol is not required to be zero-knowledge. However revealing the messages m_i requires in a linear communication complexity. For this reason, we consider the structure preserving signature (SPS) scheme of [AGHO11], that avoids the use of hash functions. For a complete description of the signature scheme we refer to [AGHO11]. Here, we are mainly interested in the verification algorithm.

Structure Preserving Signature Scheme. The SPS scheme woks over a bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$ and it allows a prover to sign a pair of group elements $(M, N) \in \mathbb{G}_1 \times \mathbb{G}_2$. The verification key is of the form $(U, V, W, Z) \in \mathbb{G}_1 \times \mathbb{G}_2^3$ and a signature is of the form $(R, S, T) \in \mathbb{G}_1^2 \times \mathbb{G}_2$. A signature (R, S, T) on a message (M, N) is accepted by a verifier if the following two equations hold

$$\begin{aligned} &(1) \quad e(R, V) + e(S, H) + e(M, W) = e(G, Z), \\ &(2) \quad e(R, T) + e(U, N) = e(G, H). \end{aligned} \tag{18}$$

Recall that group operations in $\mathbb{G}_T$ are written additively. Note that this signature scheme allows a prover to sign pairs of group elements, while what we actually need is a scheme for signing pairs of field elements $(m_i, \mathsf{tag}) \in \mathbb{Z}_q^2$. However, to sign a pair $(m, \mathsf{tag}) \in \mathbb{Z}_q^2$ the prover simply uses the SPS scheme to create a signature on $(G^m, H^{\mathsf{tag}}) \in \mathbb{G}_1 \times \mathbb{G}_2$. Because, group exponentiation is a linear operation this additional step is easily dealt with in our proofs of knowledge. For simplicity, we therefore assume that the input messages and tag of relation R'_{pred} are group elements, i.e., $(m, \mathsf{tag}) \in \mathbb{G}_1 \times \mathbb{G}_2$.

Hence, the SPS scheme avoids the hash function and the verification only consists of a small number of bilinear-group operations. This means that, using this signature scheme, both relations R_{pred} and R'_{pred} can be captured by small bilinear circuits. Our proofs for bilinear circuit relations, combined with the proofs of partial knowledge [ACF20], directly result in the required protocols.

Inner-Product Argument of Knowledge. However, this approach still results in communication complexity that is linear in the number of players. The second verification equation of the SPS scheme is not linear in the input, i.e., it contains a pairing gate $e(R, T)$ taking two variable input values. Combined with the fact that our bilinear circuit protocols have a complexity that is linear in the number of pairing gates with two variable input values, a linear communication complexity follows. To avoid this linear complexity, we make an adaptation to our generic approach. Instead of using the linearization strategy of Section 6, we deploy a protocol that proves a quadratic inner-product relation directly, i.e., without linearizing the non-linearities first. Combined with our techniques this results in PoK for relation R'_{pred} with a logarithmic communication complexity.

In Appendix A, we present a natural abstraction of an inner-product compression mechanism. Various instantiations of this protocol can be found in literature [BCC+16, BBB+18, LMR19, BMMV19]. The protocol is defined for arbitrary group homomorphisms Ψ. In our instantiation $\Psi : \mathbb{G}_1^n \times \mathbb{G}_2^n \to \mathbb{G}_T^2$ will simply be the commitment function of Definition 9 (omitting the commitment randomness). The compression mechanism Π_Ψ, is a proof-of-knowledge for the following inner-product relation,

$$R_\Psi = \{(P, u; x, y) : \Psi(x, y) = P,\ \langle x, y\rangle = u\}.$$

This protocol is knowledge sound under the assumption that a prover can not find a non-trivial pre-image in $\Psi^{-1}(0)$, i.e., the commitment scheme must be binding. When this protocol is applied recursively (as before), the communication complexity is roughly equal to $6\log_2(n)$. This can be reduced to roughly $4\log_2(n)$ by applying the techniques from Section 4.2.

Composing the Building Blocks. We are now ready to describe the composition of the different building blocks resulting in a protocol Π'_{pred} for relation R'_{pred}. Similar to our TSS approach, we use the proof-of-partial knowledge techniques [ACF20] to reduces the k-out-of-n case to the n-out-of-n. To this end, the prover uses an appropriate polynomial $p(X)$ to modify the signatures $\sigma_i = (R_i, S_i, T_i)$, messages m_i and verification equations such that n-out-of-n, instead of k-out-of-n will be satisfied. What remains is for the prover to prove knowledge of a polynomial $p(X) = 1 + \sum_{i=1}^{n-k} a_i X^i$ and modified messages $\widetilde{m}_i = p(i)m_i$ and signatures $(\widetilde{R}_i, \widetilde{S}_i, T_i) = (p(i)R_i, p(i)S_i, T_i)$ such that

$$\begin{aligned} &(1) \quad e(\widetilde{R}_i, V_i) + e(\widetilde{S}_i, H) + e(\widetilde{m}_i, W_i) = p(i)e(G, Z_i), \\ &(2) \quad e(\widetilde{R}_i, T_i) = p(i)e(G, H) - p(i)e(U_i, \mathsf{tag}). \end{aligned} \tag{19}$$

To this end the prover commits to a vector $\mathbf{y}$ containing the coefficients of $p(X)$ and the modified messages and signatures in a single compact commitment P. Using compressed Σ-protocol Π_c the linear constraint (1) can easily be proven to be satisfied for all i. To prove that the second non-linear constraint is satisfied for all i, we transform these n constraints into a single inner-product constraint and apply the inner-product protocol Π_Ψ described in Appendix A. To this end the verifier sends a challenge $\rho \in \mathbb{Z}_q$ sampled uniformly at random to the prover. The prover computes a compact (non-hiding) commitment $Q = \textsc{Com}((\mathbf{x}', \mathbf{y}'), 0)$ to the vector $(\mathbf{x}', \mathbf{y}') = (\widetilde{R}_1, \dots, \widetilde{R}_n, T_1, \rho T_2, \dots, \rho^{n-1}T_n)$ and sends Q to the verifier, together with the value

$$u := \sum_{i=1}^{n} \rho^{i-1} \left(p(i)e(G, H) - p(i)e(U_i, \mathsf{tag})\right) \in \mathbb{G}_T.$$

Note that Q is the output of a public homomorphism evaluated in $\mathbf{y}$. Hence, the prover can prove that it has been computed honestly by running compressed Σ-protocol Π_c on public input P. Similarly, the prover shows that u has been computed honestly. As final step of the protocol, the prover uses the inner-product protocol Π_Ψ on public input Q to prove that the committed vector $(\mathbf{x}', \mathbf{y}')$ satisfies the inner product relation

$$\sum_{i=1}^{n} e(x'_i, y'_i) = u. \tag{20}$$

From this it follows, with probability at least $1 - (n-1)/q$, that non-linear equation (2) is satisfied for all $\widetilde{R}_i$ and T_i committed to in commitment Q. Here, the probability is taken over the randomness of the challenge ρ.

Protocol Π'_{pred} for relation R'_{pred} is formally described in Algorithm 6. Its communication complexity is easily seen to be logarithmic, because all its components have logarithmic communication. The concrete efficiency of $\mathit{\Pi}'_{\mathsf{pred}}$ can be further improved by amortizing certain steps of the protocol. To improve the presentation we have omitted these concrete efficiency improvements.

As mentioned above, the protocol is easily adaptable to messages and tags that are field elements instead of group elements. From there it is straightforward to prove additional constraints (captured by an arithmetic circuit) on the input messages $m_1, \dots, m_n$. In particular, a protocol $\mathit{\Pi}_{\mathsf{pred}}$ for relation R_{pred}. The communication complexity of $\mathit{\Pi}_{\mathsf{pred}}$ is logarithmic in n and the size of the circuit capturing the predicate $F(m_1, \dots, m_n) = 0$.

Algorithm 6 Interactive Protocol $\mathit{\Pi}'_{\mathsf{pred}}$ for relation R'_{pred}
Proving knowledge of k-out-of-n signed messages.

Public Input : Public Keys $\mathsf{pk}_i = (U_i, V_i, W_i, Z_i) \in \mathbb{G}_1 \times \mathbb{G}_2^3$ for $1 \leq i \leq n$,
Tag $\mathsf{tag} \in \mathbb{G}_2$.
Private Input : Messages $m_i \in \mathbb{G}_1$ for $1 \leq i \leq n$,
Signatures $\sigma_i = (R_i, S_i, T_i) \in \mathbb{G}_1^2 \times \mathbb{G}_2$ for $1 \leq i \leq n$,
$k-$ Subset $\mathcal{S} \subset \{1, \dots, n\}$ such that
$\mathsf{Verify}(\mathsf{pk}_i, (m_i, \mathsf{tag}), \sigma_i) = \mathsf{accept}\ \forall i \in \mathcal{S}$.
Output : Proof π

1. Prover computes the unique polynomial $p(X) = 1 + \sum_{i=1}^{n-k} a_j X^j \in \mathbb{Z}_q[X]$ of degree at most $n-k$ such that $p(i) = 0$ for all $i \in \{1, \dots, n\} \backslash \mathcal{S}$.
2. Prover computes $\widetilde{m}_i = p(i) m_i$ and $(\widetilde{R}_i, \widetilde{S}_i, T_i) = (p(i)R_i, p(i)S_i, T_i)$ for all $1 \leq i \leq n$.
3. Prover sends a commitment P to the vector
$$\mathbf{y} = \left(a_1, \dots, a_{n-k}, \widetilde{m}_1, \dots, \widetilde{m}_n, \widetilde{R}_1, \widetilde{S}_1, T_1, \dots, \widetilde{R}_n, \widetilde{S}_n, T_n\right) \in \mathbb{Z}_q^{n-k} \times \mathbb{G}_1^{3n} \times \mathbb{G}_2^n.$$
4. Verifier sends a challenge $\rho \leftarrow_R \mathbb{Z}_q$ sampled uniformly at random.
5. Prover computes $u = \sum_{i=1}^n \rho^{i-1} \left(p(i)e(G, H) - p(i)e(U_i, \mathsf{tag})\right) \in \mathbb{G}_T$ and sends it together with a commitment $Q = \textsc{Com}_1((\mathbf{x}', \mathbf{y}'), 0)$ to the vector
$$(\mathbf{x}', \mathbf{y}') = \left(\widetilde{R}_1, \dots, \widetilde{R}_n, T_1, \rho T_2, \dots, \rho^{n-1} T_n\right) \in \mathbb{G}_1^n \times \mathbb{G}_2^n.$$
6. Prover and verifier run compressed $\mathit{\Sigma}$-protocol $\mathit{\Pi}_c$ on public input P to show that (1) of Equation 19 is satisfied, and to prove that u and Q were computed honestly.
7. Prover and verifier run the recursive composition of inner-product protocol $\mathit{\Pi}_{\mathit{\Psi}}$ to prove that the committed vector $(\mathbf{x}', \mathbf{y}')$ satisfies the inner-product relation of Equation 20.

8 Acknowledgements

We are grateful for the constructive comments and encouragement of Hieu Phan. We also thank Thijs Veugen for the numerous editorial comments. We thank Russell Lai for answering a number of questions regarding the prior work [LMR19] and improving our understanding of their techniques. Thomas Attema has been supported by the Vraaggestuurd Programma Veilige Maatschappij, supervised by the Innovation Team of the Dutch Ministry of Justice and Security, and the Vraaggestuurd Programma Cyber Security, part of the Dutch Top Sector High Tech Systems and Materials programme. Ronald Cramer has been supported by ERC ADG project No 74079 (ALGSTRONGCRYPTO) and by the NWO Gravitation Programme (QSC).

References

AC20. Thomas Attema and Ronald Cramer. Compressed sigma-protocol theory and practical application to plug & play secure algorithmics. In *CRYPTO (3)*, volume 12172 of *Lecture Notes in Computer Science*, pages 513–543. Springer, 2020.

ACF20. Thomas Attema, Ronald Cramer, and Serge Fehr. Compressing proofs of k-out-of-n-partial knowledge. *IACR Cryptol. ePrint Arch.*, 2020:753, 2020.

ACHdM05. Giuseppe Ateniese, Jan Camenisch, Susan Hohenberger, and Breno de Medeiros. Practical group signatures without random oracles. *IACR Cryptol. ePrint Arch.*, 2005:385, 2005.

ACK21. Thomas Attema, Ronald Cramer, and Lisa Kohl. A compressed Σ-protocol theory for lattices. *IACR Cryptol. ePrint Arch.*, 2021:307, 2021.

ADD$^+$19. Ittai Abraham, Srinivas Devadas, Danny Dolev, Kartik Nayak, and Ling Ren. Synchronous byzantine agreement with expected $O(1)$ rounds, expected $O(n^2)$ communication, and optimal resilience. In *Financial Cryptography*, volume 11598 of *Lecture Notes in Computer Science*, pages 320–334. Springer, 2019.

AFG$^+$10. Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 209–236. Springer, 2010.

AGHO11. Masayuki Abe, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Optimal structure-preserving signatures in asymmetric bilinear groups. In *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 649–666. Springer, 2011.

AGM18. Shashank Agrawal, Chaya Ganesh, and Payman Mohassel. Non-interactive zero-knowledge proofs for composite statements. In *CRYPTO (3)*, volume 10993 of *Lecture Notes in Computer Science*, pages 643–673. Springer, 2018.

AMS19. Ittai Abraham, Dahlia Malkhi, and Alexander Spiegelman. Asymptotically optimal validated asynchronous byzantine agreement. In *PODC*, pages 337–346. ACM, 2019.

BBB$^+$18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society, 2018.

BBHR19. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable zero knowledge with no trusted setup. In *CRYPTO (3)*, volume 11694 of *Lecture Notes in Computer Science*, pages 701–732. Springer, 2019.

BCC$^+$16. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *EUROCRYPT (2)*, volume 9666 of *Lecture Notes in Computer Science*, pages 327–357. Springer, 2016.

BCG20. Elette Boyle, Ran Cohen, and Aarushi Goel. Breaking the $O(\sqrt{n})$-bits barrier: Balanced byzantine agreement with polylog bits per-party. *IACR Cryptol. ePrint Arch.*, 2020:130, 2020.

BGdMM05. Lucas Ballard, Matthew Green, Breno de Medeiros, and Fabian Monrose. Correlation-resistant storage via keyword-searchable encryption. *IACR Cryptol. ePrint Arch.*, 2005:417, 2005.

BLS01. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, 2001.

BMMV19. Benedikt Bünz, Mary Maller, Pratyush Mishra, and Noah Vesely. Proofs for inner pairing products and applications. *IACR Cryptol. ePrint Arch.*, 2019:1177, 2019.

Bol03. Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In *Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2003.

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73. ACM, 1993.

Can04. Ran Canetti. Universally composable signature, certification, and authentication. In *CSFW*, page 219. IEEE Computer Society, 2004.

CD98. Ronald Cramer and Ivan Damgård. Zero-knowledge proofs for finite field arithmetic; or: Can zero-knowledge be for free? In *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 424–441. Springer, 1998.

CDP12. Ronald Cramer, Ivan Damgård, and Valerio Pastro. On the amortized complexity of zero knowledge protocols for multiplicative relations. In *ICITS*, volume 7412 of *Lecture Notes in Computer Science*, pages 62–79. Springer, 2012.

CGM16. Melissa Chase, Chaya Ganesh, and Payman Mohassel. Efficient zero-knowledge proof of algebraic and non-algebraic statements with applications to privacy preserving credentials. In *CRYPTO (3)*, volume 9816 of *Lecture Notes in Computer Science*, pages 499–530. Springer, 2016.

CKS05. Christian Cachin, Klaus Kursawe, and Victor Shoup. Random oracles in constantinople: Practical asynchronous byzantine agreement using cryptography. *J. Cryptol.*, 18(3):219–246, 2005.

Cra96. Ronald Cramer. *Modular Design of Secure yet Practical Cryptographic Protocols*. PhD thesis, CWI and University of Amsterdam, 1996.

DF89. Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 307–315. Springer, 1989.

FS86. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

Gam84. Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer, 1984.

GG20. Rosario Gennaro and Steven Goldfeder. One round threshold ECDSA with identifiable abort. *IACR Cryptol. ePrint Arch.*, 2020:540, 2020.

GJKR96. Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust threshold DSS signatures. In *EUROCRYPT*, volume 1070 of *Lecture Notes in Computer Science*, pages 354–371. Springer, 1996.

GJKR03. Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Secure applications of pedersen's distributed key generation protocol. In *CT-RSA*, volume 2612 of *Lecture Notes in Computer Science*, pages 373–390. Springer, 2003.

GPS08. Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discret. Appl. Math.*, 156(16):3113–3121, 2008.

GS08. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432. Springer, 2008.

HAP18. Yotam Harchol, Ittai Abraham, and Benny Pinkas. Distributed SSH key management with proactive RSA threshold signatures. In *ACNS*, volume 10892 of *Lecture Notes in Computer Science*, pages 22–43. Springer, 2018.

HBHW20. Daira Hopwood, Sean Bowe, Taylor Hornby, and Nathan Wilcox. *Zcash Protocol Specication - Version 2020.1.7*, 2020.

HKR19. Max Hoffmann, Michael Klooß, and Andy Rupp. Efficient zero-knowledge arguments in the discrete log setting, revisited. In *ACM Conference on Computer and Communications Security*, pages 2093–2110. ACM, 2019.

HKSS20. Abida Haque, Stephan Krenn, Daniel Slamanig, and Christoph Striecks. Logarithmic-size (linkable) threshold ring signatures in the plain model. *IACR Cryptol. ePrint Arch.*, 2020:683, 2020.

KG20. Chelsea Komlo and Ian Goldberg. FROST: flexible round-optimized schnorr threshold signatures. *IACR Cryptol. ePrint Arch.*, 2020:852, 2020.

KSM20. Eleftherios Kokoris-Kogias, Alexander Spiegelman, and Dahlia Malkhi. Asynchronous distributed key generation for computationally-secure randomness, consensus, and threshold signatures. In *ACM Conference on Computer and Communications Security*. ACM, 2020.

Lam11. Leslie Lamport. Byzantizing paxos by refinement. In *DISC*, volume 6950 of *Lecture Notes in Computer Science*, pages 211–224. Springer, 2011.

Lib19. Libra Team. *State Machine Replication in the LibraBlockchain*, 2019. Version 2019-10-24.

Lin03. Yehuda Lindell. Parallel coin-tossing and constant-round secure two-party computation. *J. Cryptology*, 16(3):143–184, 2003.

LJY16. Benoît Libert, Marc Joye, and Moti Yung. Born and raised distributively: Fully distributed non-interactive adaptively-secure threshold signatures with short shares. *Theor. Comput. Sci.*, 645:1–24, 2016.

LM18. Julian Loss and Tal Moran. Combining asynchronous and synchronous byzantine agreement: The best of both worlds. *IACR Cryptol. ePrint Arch.*, 2018:235, 2018.

LMR19. Russell W. F. Lai, Giulio Malavolta, and Viktoria Ronge. Succinct arguments for bilinear group arithmetic: Practical structure-preserving cryptography. In *ACM Conference on Computer and Communications Security*, pages 2057–2074. ACM, 2019.

NRS$^+$20. Kartik Nayak, Ling Ren, Elaine Shi, Nitin H. Vaidya, and Zhuolun Xiang. Improved extension protocols for byzantine broadcast and agreement. In *DISC*, volume 179 of *LIPIcs*, pages 28:1–28:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Ped91. Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

Ram20. Matthieu Rambaud. Malicious security comes for free in consensus with leaders. *IACR Cryptol. ePrint Arch.*, 2020:1480, 2020.

Sha79. Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

Sho00. Victor Shoup. Practical threshold signatures. In *EUROCRYPT*, volume 1807 of *Lecture Notes in Computer Science*, pages 207–220. Springer, 2000.

SW05. Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005.

YMR$^+$19. Maofan Yin, Dahlia Malkhi, Michael K. Reiter, Guy Golan-Gueta, and Ittai Abraham. Hotstuff: BFT consensus with linearity and responsiveness. In *PODC*, pages 347–356. ACM, 2019.

Appendix

A Compression Mechanism for Inner Product Relation

This section describes an abstract compression mechanism for proving that the pre-image of a one-way homomorphism satisfies an inner-product relation. When composing this compression mechanism recursively the communication complexity drops from linear down to logarithmic. This protocol is not zero-knowledge, but can easily be made zero-knowledge by composing it with the appropriate SHVZK Σ-protocol. The protocol, denoted by Π_Ψ, is described in Protocol 7 and its properties are summarized in Theorem 8.

Theorem 8 (Abstract Inner Product Argument). *Let $\mathbb{H} = \mathbb{H}' \oplus \mathbb{H}'$ for some group $\mathbb{H}'$ with prime exponent*[11] *q and let $\Psi : \mathbb{H} \oplus \mathbb{H} \to \mathbb{G}$ be a group homomorphism. Then Π_Ψ is a 3-move protocol for relation*

$$R_\Psi = \{(P, u; x, y) : \Psi(x, y) = P,\ \langle x, y\rangle = u\}.$$

It is perfectly complete and computationally 3-special sound, under the assumption that it is computationally hard to compute a non-trivial pre-image of 0 for Ψ. Moreover, the communication costs are:

- *$\mathcal{P} \to \mathcal{V}$: 2 elements of $\mathbb{G}$ and 2 element of $\mathbb{H}'$ and 2 elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \to \mathcal{P}$: 1 element of $\mathbb{Z}_q$.*

Proof. **Completeness** follows directly.

Special Soundness: We show that the protocol is *computationally* 3-special sound. Let $(A, B, a, b, c_1, z_{1,1}, z_{2,1})$, $(A, B, a, b, c_2, z_{1,2}, z_{2,2})$ and $(A, B, a, b, c_3, z_{1,3}, z_{2,3})$ be three accepting transcripts for distinct challenges $c_1, c_2, c_3 \in \mathbb{Z}_q$.

Let $V^{-1} \in \mathbb{Z}_q$ be the inverse of a Vandermonde matrix defined by c_1, c_2, c_3, i.e.,

$$V^{-1} \begin{pmatrix} 1 & c_1 & c_1^2 \\ 1 & c_2 & c_2^2 \\ 1 & c_3 & c_3^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Since the challenges are distinct, this Vandermonde matrix is invertible and V^{-1} indeed exists. We define

$$\begin{pmatrix} \bar{\mathbf{a}} \\ \bar{\mathbf{z}} \\ \bar{\mathbf{b}} \end{pmatrix} = \begin{pmatrix} \bar{a}_1 & \bar{a}_2 & \bar{a}_3 & \bar{a}_4 \\ \bar{x}_1 & \bar{x}_2 & \bar{y}_1 & \bar{y}_2 \\ \bar{b}_1 & \bar{b}_2 & \bar{b}_3 & \bar{b}_4 \end{pmatrix} = V^{-1} \begin{pmatrix} c_1 z_{1,1} & z_{1,1} & z_{2,1} & c_1 z_{2,1} \\ c_2 z_{1,2} & z_{1,2} & z_{2,2} & c_1 z_{2,2} \\ c_3 z_{1,3} & z_{1,3} & z_{2,3} & c_1 z_{2,3} \end{pmatrix}.$$

Then, by the first verification equation of Π_Ψ, it follows that $\Psi(\bar{\mathbf{a}}) = A$, $\Psi(\bar{\mathbf{z}}) = P$ and $\Psi(\bar{\mathbf{b}}) = B$.

Now note that, by these same verification equation, it either follows that, for all $1 \le i \le 3$,

$$\begin{aligned} z_{1,i} &= \bar{a}_2 + c_i \bar{x}_2 + c_i^2 \bar{b}_2, \\ z_{2,i} &= \bar{a}_3 + c_i \bar{y}_1 + c_i^2 \bar{b}_3, \\ c_i z_{1,i} &= \bar{a}_1 + c_i \bar{x}_1 + c_i^2 \bar{b}_1, \\ c_i z_{2,i} &= \bar{a}_4 + c_i \bar{y}_2 + c_i^2 \bar{b}_4, \end{aligned}$$

or we can deduce a pre-image of 0. Namely, suppose for example that the first equation does not hold for $i = 1$, then $\gamma := z_{1,1} - \bar{a}_2 - c_1 \bar{x}_2 - c_1^2 \bar{b}_2 \neq 0$ and $\Psi(\gamma) = 0$.

[11] Recall that the exponent of group is the smallest e such that $g^e = 1$ for all group elements g.

So let us assume that these equalities hold. Because they hold for three pairwise distinct challenges, it follows that

$$\begin{aligned} &\bar{a}_1 = \bar{a}_4 = \bar{b}_2 = \bar{b}_3 = 0,\\ &\bar{a}_2 = \bar{x}_1, \quad \bar{a}_3 = \bar{y}_2,\\ &\bar{b}_1 = \bar{x}_2, \quad \bar{b}_4 = \bar{y}_1. \end{aligned}$$

Hence, by the second verification equation, it holds that, for all $1 \leq i \leq 3$,

$$\begin{aligned} a + c_i u + c_i^2 b &= \langle z_{1,i}, z_{2,i} \rangle,\\ &= \langle \bar{x}_1 + c_i \bar{x}_2, \bar{y}_2 + c_i \bar{y}_1 \rangle,\\ &= \langle \bar{x}_1, \bar{y}_2 \rangle + c_i \langle (\bar{x}_1, \bar{x}_2), (\bar{y}_1, \bar{y}_2) \rangle + c_i^2 \langle \bar{x}_2, \bar{y}_1 \rangle. \end{aligned}$$

Since, this equality holds for three pairwise distinct challenges it follows that $\langle (\bar{x}_1, \bar{x}_2), (\bar{y}_1, \bar{y}_2) \rangle = u$. Hence $\bar{z} = (\bar{x}_1, \bar{x}_2, \bar{y}_1, \bar{y}_2)$ is a witness for relation R_Ψ, which completes the proof. □

Protocol 7 Generic Compression Mechanism Π_Ψ for inner-product relation R_Ψ.

$\textsc{Input}(P, u; x = (x_L, x_R), y = (y_L, y_R))$

$P = \Psi(x, y) \in \mathbb{G}$
$\langle x, y \rangle = u \in \mathbb{Z}_q$

Prover		Verifier
$A = \Psi(0, x_L, y_R, 0), a = \langle x_L, y_R \rangle$ $B = \Psi(x_R, 0, 0, y_L), b = \langle x_R, y_L \rangle$		
	$\xrightarrow{A,B,a,b}$	
		$c \leftarrow_R \mathbb{Z}_q$
	$\xleftarrow{c}$	
$z_1 = x_L + c x_R$ $z_2 = c y_L + y_R$		
	$\xrightarrow{z_1, z_2}$	$\Psi(c z_1, z_1, z_2, c z_2) \overset{?}{=} A P^c B^{c^2}$ $\langle z_1, z_2 \rangle \overset{?}{=} a + cu + c^2 b$