Structure-Preserving Compressing Primitives: Vector Commitments and Accumulators and Applications

Stephan Krenn[1], Omid Mir[1], and Daniel Slamanig[2]

[1] AIT Austrian Institute of Technology
{omid.mir, stephan.krenn}@ait.ac.at
[2] Research Institute CODE, Universität der Bundeswehr München, Germany
daniel.slamanig@unibw.de

Abstract. Compressing primitives such as accumulators and vector commitments, allow to represent large data sets with some compact, ideally constant-size value. Moreover, they support operations like proving membership or non-membership with minimal, ideally also constant-size, storage and communication overhead. In recent years, these primitives have found numerous practical applications, with many constructions based on various hardness assumptions. So far, however, it has been elusive to construct these primitives in a strictly structure-preserving setting, i.e., in a bilinear group in a way that messages, commitments and openings are all elements of the two source groups. Interestingly, backed by existing impossibility results, not even conventional commitments with such constraints are known in this setting. However, in many practical applications it would be convenient or even required to be structure-preserving, e.g., to commit or accumulate group elements.
In this paper we investigate whether strictly structure-preserving compressing primitives can be realized. We close this gap by presenting the first strictly structure-preserving commitment that is shrinking (and in particular constant-size). We circumvent existing impossibility results by employing a more structured message space, i.e., a variant of the Diffie-Hellman message space. Our main results are constructions of structure-preserving vector commitments as well as structure-preserving accumulators. We first discuss generic constructions and then present concrete constructions under the Diffie-Hellman Exponent assumption. To demonstrate the usefulness of our constructions, we discuss various applications. Most notable, we present the *first* entirely practical constant-size ring signature scheme in bilinear groups (i.e., the discrete logarithm setting). Concretely, using the popular BLS12-381 pairing-friendly curve, our ring signatures achieve a size of roughly 3 Kbits.

1 Introduction

NOTE: *As recently communicated to us by Gennaro Avitabile, Dario Fiore, and Gonzalo Martinez de Sola, the SPVC construction in Section 4 of an earlier version of this paper, published at PKC 2026 [64], is not position-binding. More precisely, the issue affects only the construction based on the AGM that does not include* VerifyMsg *in the verification procedure. They shared with us their full manuscript (which is accepted at Asiacrypt 2026 [14]), including both the attacks and their independently developed solutions addressing the identified issues. We likewise shared with them our manuscript and our independently developed fix, after which the two groups exchanged and compared their respective solutions.*

Our fix adds an additional degree-zero check on the message. This repairs the affected SPVC construction and all constructions built on top of it while introducing only minimal overhead. We also take this as a chance to update our ring signature scheme into a simpler and more efficient construction thereby also resolving an issue spotted by Avitabile et al.

Compressing primitives like accumulators, vector commitments or key-value commitments (key-value maps) are cryptographic techniques to efficiently represent and manage large datasets in a space-efficient form. They allow to represent large datasets with a compact, ideally constant-sized, value and support operations like proving membership or non-membership with minimal storage and communication overhead (ideally constant-size witnesses). These primitives have many interesting applications for blockchain privacy, e.g., Zcash[3], and scalability, e.g., use of Verkle trees[4],

[1] Corresponding author: omid.mir@ait.ac.at

[3] https://z.cash

[4] https://vitalik.eth.limo/general/2021/06/18/verkle.html

stateless clients for blockchains[5], data availability sampling [60] or batching [27]. Moreover, they are extensively used in privacy-preserving systems, such as anonymous authentication primitives, e.g., ring signatures [41] or revocation in anonymous credentials [8,11,37], or data outsourcing [20,29,87]. In this paper, we focus mainly on vector commitments [29] and accumulators [15,21].

Vector Commitments. We recall that a vector commitment (VC) scheme enables a user to commit to a vector $\mathbf{m}$, with the ability to later open the commitment at specific positions without being able to cheat (position binding). Moreover, they might also support to update values committed at certain position. Crucially, both the commitment size and the size of the opening must remain succinct, ideally of constant size. In recent years we have seen significant advancements and growing interest in the development and applications of vector commitments. Beginning with the foundational Merkle tree [76], which leverages collision-resistant hash functions, the field has expanded to include a diverse array of algebraic constructions. These include schemes based on pairing-based assumptions [29,55,62,65,70,71,89] as well as those relying on assumptions over groups of unknown order, such as RSA groups or class groups [26,29,65] and post-quantum constructions based on lattices [10,67,84,90]. For an extensive overview of schemes, see [82]. As we have already mentioned above, vector commitments have been widely utilized across various applications. Moreover, generalizations of vector commitments and in particular polynomial commitments [62] and functional commitments [70], have become foundational components in many recent constructions of succinct non-interactive arguments of knowledge (SNARKs).

Accumulators. We also recall that a (static) accumulator [15,21] is another type of compressing primitive, which allows to represent a set of elements $X = \{x_1, \ldots, x_q\}$ in the form of a succinct accumulator value acc_X. For each element $x_i \in X$, a witness wit_{x_i} can be efficiently computed to certify its membership in the set, while ensuring that it is computationally infeasible to forge witnesses for non-members $y \notin X$ (collision resistance). Universal accumulators in addition provide the functionality of non-membership witnesses for values $y \notin X$ and dynamic accumulators supporting efficient additions and deletions from the accumulated set. It is worth remarking that Catalano and Fiore have shown that any vector commitment can be used to construct universal dynamic accumulators [29]. Unfortunately, it is unclear whether this approach can be immediately applied to the SP setting or effectively hide the index, a property often required in privacy-preserving applications. Like vector commitments, accumulators have found wide-ranging applications and actually evolved into a foundational component in various advanced cryptographic constructs. They are integral to revocation in anonymous credentials [8, 11, 37], membership revocation for group signatures [25] and the construction of ring signatures [39,41,66].

Over time, cryptographic accumulators have evolved through various instantiations. Initially based on the RSA assumption [15], early schemes were later expanded upon [25,41]. Nguyen introduced pairing-based accumulators [80], sparking further advancements in this area [11,12,24,36,53]. More recent developments include lattice-based accumulators [61,67,83].

Structure-Preserving Compressing Primitives. While, as outlined above, numerous constructions exist for both vector commitments and accumulators, a significant challenge remains in developing such schemes that are structure-preserving (SP). We recall that a cryptographic scheme is called structure preserving [3] if all public inputs and outputs consist of elements of the source groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of a bilinear group $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P, \hat{P})$ and functional correctness can be verified only by testing group membership, computing group operations, and evaluating pairing product equations (PPEs) [1] of the form $\prod_i \prod_j e(A_i, \hat{B}_j)^{c_{i,j}} = 1$, where $A_i \in \mathbb{G}_1$ and $\hat{B}_j \in \mathbb{G}_2$ are group elements and $c_{i,j} \in \mathbb{Z}_p$ are constants. This domain is very rich in its constructions and allows for a modular design of (advanced) cryptographic primitives, e.g., structure-preserving signatures (SPS) [3] (cf. [52] for a recent overview on the large body of works), threshold SPS [34,79], blind signatures [3,48], group signatures [3,69], traceable signatures [2], homomorphic signatures [68] or delegatable anonymous credentials [35,47,78]. Such constructions are highly desirable because they maintain the algebraic structure of the committed values and proofs, ensuring compatibility with a broader range of cryptographic protocols and in particular the Groth-Sahai [59] and related proof systems. Moreover, especially for compressing primitives they are of concrete practical interest beyond modular protocol design, as we will discuss in this paper.

[5] https://ethresear.ch/t/the-stateless-client-concept/172

Unfortunately, there are known impossibility results of Abe et al. [6] establishing that strictly structure-preserving commitments to source-group elements cannot be smaller than the input message size and thus scale linearly with the number of group elements. In order to be compressing though, for conventional commitments, there are some approaches that relax the SP setting by allowing elements from the target group $\mathbb{G}_T$ in the commitment [5, 6]. Another approach by Abe et al. [7] is to construct SP commitments that have a relaxed notion of binding, where the message space and the verification space differ (e.g., being $\mathbb{Z}_p$ and $\mathbb{G}_1$ respectively). Nevertheless, these are not strictly structure-preserving commitments and in particular do not allow to commit to commitments, a feature that is interesting for practical applications. When it comes to accumulators, to the best of our knowledge, the approach that conceptually comes closest is that of determinantal accumulators [72]. While they are in spirit of SP primitives and their combination with Groth-Sahai proofs [59], the focus of determinantal primitives is on designing primitives that can be modularly combined with algebraic NIZKs in the vein of Couteau and Hartmann [32] and Couteau et al. [33]. Consequently, they are not structure-preserving. For vector commitments, there is an impossibility result for algebraic vector commitments in pairing-free groups [30], but this does not rule out the existence of structure-preserving ones. While [30] notes that the impossibility of strictly SP commitments [5, 6] rules out constructing succinct vector commitments in this structure-preserving setting, we are not aware of any work trying to bypass such an impossibility and thus this state of affairs is not satisfactory. This leads us to the following question:

Can we design (vector) commitments and accumulators that retain algebraic structure, i.e., being strictly structure-preserving, while being succinct?

Addressing this gap is crucial for advancing the field and our aim is to close it.

1.1 Our Results

Our contributions can be summarized as follows:

- We present the first construction of group-to-group commitments that are shrinking and strictly structure-preserving, i.e., messages, commitments and openings are all elements in the source groups $\mathbb{G}_1$ and $\mathbb{G}_2$. We obtain this by bypassing known impossibility results due to Abe et al. [5,6] as we require a more structured message space and in particular a Diffie-Hellman (DH) message space [3, 46]. Such a message space has recently shown to be relevant for various applications [34, 77].
- We present the first structure-preserving vector commitments (SPVCs). As a warm up, we show a (probably folklore) construction of weak-binding WSPVC from any EUF-CMA secure SPS scheme. However as our main result, we turn to SPVC's providing the conventional notion of position-binding and present a construction of SPVC under the Diffie-Hellman Exponent (q-DHE) assumption with a message space that is defined with respect to some global parameters (inspired by the recent work by Griffy et al. [56] and q-DHE VC schemes [55, 71]). Notably, our construction extends the applicability of vector commitments to settings where the prover cannot directly access Dlog representations, making it particularly suitable for applications such as ring signatures, where public keys are group elements and the signer can accumulate (or commit to) them without knowing their underlying exponents.
- We present the first structure-preserving accumulators (SPAs). We first discuss how a weak-binding vector commitment WSPVC naturally yields an accumulator. Second, and more importantly, we introduce a generic construction that allows any SPVC to be applied in a black-box manner to build an SPA in a different model with a new security property we call bounded soundness, i.e., for a q-bounded accumulator the adversary needs to output $q + 1$ valid value-witness pairs, even when the accumulator is adversarially generated. This security definition aligns with our black-box approach.
  Third, we extend SPA and construct perfect randomizable q-DHE accumulator starting for another and in particular DH-type message space. Here we aim to hide the index (or position), i.e., it should be possible to give out a witness and accumulated value such that they cannot be linked back to the accumulated value. This is accomplished through a randomization technique that ensures the randomized accumulated values do not reveal anything about the position of

the original accumulated value and DH-type messages. The security of the randomizable accumulator is based on a variant of collision resistance, which we formalize through a theorem and show to be sufficient for the security of ring signatures.

– Finally, we outline some applications that showcase the benefits of our schemes. Most notable, we present the *first* entirely practical constant-size ring signature scheme in bilinear groups (i.e., the discrete logarithm setting) and prove its security in the strongest model of Bender et al. [22]. It is inspired by the key-homomorphic signature based ring signature construction in [40] but uses an accumulator for the membership proof. We base our construction on our randomized SP accumulator and a variant of BLS signatures that we call accumulator-compatible BLS, which ensures that BLS public keys are compatible with the message space of our accumulator. Concretely, using the popular BLS12-381 pairing-friendly curve, ring signatures are of size roughly 3 Kbits. Moreover, we discuss the potential applications of our SPVC to succinct data availability sampling and algebraic Verkle trees. We believe that our schemes unlock a wide range of new applications.

1.2 Technical Overview

We provide here a brief technical overview of the techniques used before presenting the full constructions.

Strictly Structure-Preserving Commitments: Firstly, we present shrinking and strictly structure-preserving group-to-group commitments, thereby bypassing aforementioned impossibility results by using a structured message space. Consider therefore an asymmetric Type-3 pairing group: $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P, \hat{P}) \leftarrow \mathsf{BGSetup}(1^\lambda)$.

We start from the structure-preserving (bilinear pairing-based) Pedersen commitment scheme proposed by Abe et al. [4], where the commitment public key consists of q group elements $(X_1, \ldots, X_q)$ from $\mathbb{G}_1$. To commit to $(\hat{M}_1, \ldots, \hat{M}_q) \in \mathbb{G}_2$, a random element $\hat{R} \in \mathbb{G}_2$ is selected, and the commitment takes the form:

$$C = e(H, \hat{R}) \cdot \prod_{i=1}^{q} e(X_i, \hat{M}_i).$$

This commitment scheme can be shown to be binding under the Double Pairing Problem (DBP) assumption.

Now, to *shrink* commitments into a single element in the source group $\mathbb{G}_1$, we have overcome the impossibility results of Abe et al. [6]. Progress on this has been elusive so far. To overcome this open problem, we take inspiration from previous work on circumventing impossibility results and lower bounds in SPS [51, 52], achieved by switching from arbitrary group elements in the message space to a more structured message space. In particular we leverage a variant of the Diffie-Hellman (DH) message space, which has been used in various structure-preserving constructions in the past [3, 4, 46, 51, 52]. Specifically, we consider messages of the following form:

$$(\mathbf{M}, \hat{\mathbf{N}}) = (M_1, \ldots, M_q, \hat{N}_1, \ldots, \hat{N}_q),$$

such that there exist $m_i \in \mathbb{Z}_p$ for $1 \leq i \leq q$, satisfying:

$$M_i = X_i^{m_i}, \quad \hat{N}_i = \hat{P}^{m_i}, \quad \forall i \in [q].$$

In other words, the commitment public keys are embedded in $\mathbf{M}$, providing flexibility to generate commitments in $\mathbb{G}_1$ as:

$$\mathsf{com} := P^r \cdot \prod_{i=1}^{q} M_i \quad \text{and} \quad \mathsf{Open} := \hat{P}^r.$$

Meanwhile, $\hat{\mathbf{N}}$ is used for verification. This still allows us to prove security under the DBP assumption as the second group elements $\hat{\mathbf{N}}$ along with $\hat{P}^r$ can be used to break the assumption, similar to the security proof in [4]. Thus, relying on a DH message space circumvents certain constraints by enabling commitments in one group while verification occurs in another. At the same time, the random group elements or commitment keys remain/encode in the same group as the committed messages. Overall this gives a surprisingly simple construction of shrinking structure-preserving commitments and resolves a longstanding open problem.

Vector Commitments. Then we shift our attention to vector commitments. As a preliminary step, we present a weakly binding structure-preserving vector commitment (WSPVC), where commitments are generated honestly. We demonstrate that this can be realized through a generic construction based on any compact structure-preserving signature (SPS) scheme, effectively yielding a structure-preserving accumulator. We defer this construction in [63]. This construction directly yields a first SP accumulator, though only in a weak setting and in particular a setting where the accumulator has been computed honestly.

We then construct a SP vector commitment from the q-DHE assumption. Our construction is inspired by the q-DHE based mercurial commitment scheme from Libert and Yung [71], which however is not SP and just commits to scalars. To turn the underlying idea into a full-blown SPVC we need two essential ideas. We again rely on a type of DH message space, but this time the messages to be committed are generated with respect to the public parameters pp, which serve as the commitment keys, i.e., instead of using a random vector of group elements $\mathbf{X}$ we use q-DHE parameters to build messages. The latter idea is inspired by the recent construction of strongly private mercurial signatures by Griffy et al. [56].

More precisely, each message is represented as a vector over $\mathbb{G}_1$ and consists of two main components: i) the main message component $\mathbf{M}$ and ii) a tag vector $\mathbf{T}$. Latter is derived from the pp, and helps to encode the position of the message in the vector (i.e., the commitments and the witnesses). This structure is leveraged alongside mercurial commitments of Libert and Yung [55, 71] to construct our SPVC. A mercurial commitment takes the form: $C = P^{\sum \alpha^i \cdot m_i}$, where $(P^{\alpha^i})_{i\in[q]}$ is the pp (i.e., q-DHE elements). A key challenge in the structure-preserving setting is the lack of direct access to discrete logarithms. To address this, we introduce a new messages structure such that we encode messages implicitly in a mercurial vector commitment using $M_{i,0} = P^{m_i} \in \mathbb{G}_1$ and $M_{i,1} = M_{i,0}^{\alpha^i} = B_i^{m_i} \in \mathbb{G}_1$, where the α^i are now available in encoded messages and $B_i = P^{\alpha^i}$ are the pp elements. This allows us to build a VC over $\mathbf{M}_{i,1}$ for all $i \in [q]$.

To compute an opening for a committed message at some index i, mercurial commitments employ a shifting technique to create a proof for a message m_i as: $P^{\sum_{j\neq i} m_j \alpha^{q+1-i+j}}$. This transformation ensures that the message for which we want to generate a proof is always positioned at $q+1$. Since we do not have $P^{\alpha^{q+1}}$, it is naturally the missing element in the proof. However, during verification, it can still be checked using $g_t^{m_i}$ and this element can actually be explicitly included in the proof π – indeed, the security follows from the fact that the adversary does not know α^{q+1}, so it is not able to prove a message at this position – while other messages are shifted to positions within the range $[q+1, 2q]$. The commitments are also shifted accordingly in the same way, i.e., $e(C, \hat{P}^{\alpha^{q-i+1}})$. Clearly, when m_i is known and one has access to q-DHE parameters, this proof can be computed efficiently.

In our case, since m_i is unknown, it is pre-encoded with α using our message space and q-DHE elements. This ensures that when messages are shifted (as required for mercurial VC proofs), the necessary elements remain available for proof generation. We assign these elements as tag components and as part of our messages e.g., $\mathbf{T} = (P^{\alpha^j \cdot m_i})$ for $i \in [q]$ and $j \in [i+1, q+i] \setminus \{i\}$ (note that the details of how j is defined will be clarified and further explained in the construction section), playing a crucial role in encoding the missing terms during proof construction. We note that verifying messages of the form $M = P^{\alpha^i \cdot m_i}$ requires a slight modification of the verification equations. Instead of checking

$$e(C, \hat{P}^{\alpha^{q-i+1}}) = e(\pi, \hat{P}) \cdot g_t^{m_i} \text{ or } e(\pi, \hat{P}) \cdot e(P^{\alpha}, \hat{P}^{\alpha^q})^{m_i},$$

we set it to

$$e(C, \hat{P}^{\alpha^{q-i+1}}) = e(\pi, \hat{P}) \cdot e(P^{m_i}, \hat{P}^{\alpha^{q+1}}).$$

Additionally, we include a second verification equation

$$e(M_{i0}, \hat{B}_{2q}) = e(T_{i,2q}, \hat{P})$$

which ensures that the message is well-formed and has the required degree with respect to the public parameters. This modification enables verification while still allowing the binding proof to go through.

We can view this message structure as a generalized DH message space built on q-DHE parameters. Finally, we structure the messages based on the q-DHE assumption in such a way that verifying

the correctness of the message structure enables the extraction of discrete logarithms in the Generic Group Model (GGM) and ultimately reduces binding to the q-DHE assumption. The message verification has the form $e(M_{i1}, \hat{B}_{j-i}) = e(T_{ij}, \hat{P})$ and $e(M_{i1}, \hat{P}) = e(M_{i0}, \hat{B}_i)$, where one uses the public elements $\hat{B}_{j-i}$ from the pp. Alternatively, we also show the proof of binding in the AGM, which allows us to avoid explicit extraction. The use of the AGM leads to a slightly more efficient construction, since we no longer need to explicitly verify the message structure during verification. We present the AGM proof in the main body and the GGM proof in Appendix C.1.

We note that our construction does not require tags (even if they are linear) for verification; they are used only during proof generation and included as part of the auxiliary data. This is particularly appealing in the context of vector commitments and accumulator applications, as it enables efficient and modular verification.

SP Accumulators (SPA). We then turn to the construction of SP accumulators with stronger guarantees as the ones obtained from WSPVC. While there are results due to Catalano and Fiore [29] which show how to construct accumulators from vector commitments in a generic way, this idea does not immediately apply to the structure-preserving setting.

Thus, we present a *compiler (or generic construction) from vector commitments to accumulators*, demonstrating how any SPVC can be used to construct an SPA. Although our generic construction is applicable to all types of vector commitments and can be extended to accumulators, we focus here on the SP setting. An interesting observation when building accumulators from VCs is that in contrast to collision resistance, we allow the accumulator to be fully controlled by the adversary. We call this new notion bounded soundness, and it asks the adversary for a q-bounded accumulator to output an accumulator along with $q+1$ valid value and witness pairs. Therefore, we introduce our generic constructions, along with the new above mentioned security notion specifically tailored for black-box accumulator constructions derived from VCs.

q-DHE SPA with Perfect Randomizability. Since our generic construction utilizes SPVC, one inherently reveals the positions of elements within the set. This can potentially compromise sensitive information in privacy-preserving primitives. For example, in ring signatures where an accumulator could be used to accumulate public keys, one would expose the indices of the ring members and thus breaking anonymity. To enhance privacy in such applications, we aim for an approach where witnesses do not disclose any information about the elements. Thus, we introduce the notion of *the q-DHE accumulator with perfect randomizability* and modify SPA construction to satisfy this property.

In the q-DHE SPVC the relations that are required to verify the witness:

$$e\left(C, \hat{B}_{q+1-i}\right) = e\left(\pi_i, \hat{P}\right) \cdot e\left(M_{i0}, \hat{B}_{q+1}\right) \wedge e(M_{i0}, \hat{B}_{2q}) = e(T_{i,2q}, \hat{P}).$$

where $\hat{B}_{j-i}$ are public elements from the pp, one reveals the positions of elements in the vector. To address this, we use a randomization technique to break this connection. We randomize the witness using a random $\rho \in \mathbb{Z}_p^*$ and include $(\hat{B}_{q+1-i})^\rho$ as part of the witness. Assume that $W_1 = \pi_i^\rho$ and $\hat{W}_2 = (\hat{B}_{q+1-i})^\rho$, then the verification equation takes the form:

$$e\left(C, \hat{W}_2\right) = e\left(W_1, \hat{P}\right) \cdot e\left(M_{i0}, \hat{B}_{q+1}\right) \wedge e(M_{i0}, \hat{W}_4 = \hat{B}_{2q}^\rho) = e(W_3 = T_{i,2q}^\rho, \hat{P}).$$

Note that the verification prevents any leakage of the initial witnesses. We show that a variant of collision resistance (tailored to the construction) of this construction can be reduced to the position-binding property of SPVC.

Application to Ring Signatures: VCs and accumulators have many natural applications such as anonymous authentication, revocation in anonymous credentials, and data outsourcing. We explicitly provide a construction of a constant-size ring signature in the discrete logarithm setting—the first of its kind—by combining an accumulator-compatible variant of BLS with our q-DHE-based SPA and an efficient Schnorr-type zero-knowledge proof.

We recall that for BLS the signing key is $\mathsf{sk}_i := x_i \in \mathbb{Z}_p$, the public-key component is P^{x_i}, and a signature on a message m is $\delta := H(m)^{x_i}$, where H maps messages to $\mathbb{G}_2$. To make BLS public keys compatible with the message space of our accumulator, we extend them to $\mathsf{pk}_i = (M_{i0} = P^{x_i}, M_{i1} = B_i^{x_i})$ and associate a tag $\tau_i = \mathbf{T}_i$, yielding a tagged public key $t\mathsf{pk}_i = (\mathsf{pk}_i, \tau_i)$ that is a

valid element of the accumulator message space. A signer can therefore form a ring $\mathcal{R}$ of public keys, compute the corresponding accumulator $\mathsf{Acc}_{\mathcal{R}}$, and derive a membership witness for its own public key. In parallel, it computes a $\mathsf{BLS}_{\mathsf{Acc}}$ signature on $m\|\mathcal{R}$. Rather than revealing the signer-specific public key, signature, or membership witness, the signer proves in zero knowledge that it knows a secret key corresponding to a public key contained in the accumulated ring and that the same secret key authenticates $m\|\mathcal{R}$ under $\mathsf{BLS}_{\mathsf{Acc}}$. Thus, the proof links the BLS verification relation with the accumulator-opening relation without revealing which ring member is used. The proof can be instantiated using standard Schnorr-type techniques and made non-interactive via the Fiat–Shamir transform. This avoids SNARKs and other expensive machinery while preserving constant signature size. See Section 6 for details. More generally, the construction follows a generic paradigm of combining a signature scheme in the DL setting with an accumulator whose membership relation can be efficiently linked to the signing-key relation in zero knowledge.

We note that in scenarios where a party must prove membership of group elements without access to discrete logarithms (e.g., secret keys) directly, existing vector commitment or accumulators schemes are insufficient, as they typically require knowledge of the discrete logarithms of all public keys to compute a valid commitment. Our construction circumvents this limitation by design, enabling applications that are infeasible with prior approaches.

2 Preliminaries

Notations. The main security parameter will be denoted by λ. We use $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P, \hat{P}, g_t) \leftarrow \mathsf{BGSetup}(1^\lambda)$ to denote a bilinear group generator for asymmetric type-3 bilinear groups, where p is a prime of bit length λ. We use $[q]$ to denote the set $\{1, 2, \ldots, q\}$. When drawing multiple values from a set, we may omit notation for products of sets, e.g. $(x, y) \in \mathbb{Z}_p$ is the same as $(x, y) \in (\mathbb{Z}_p)^2$. For a map from the set Z to the set S, $m : Z \to S$, we will denote $m[i] \in S$ as the output of the map in S with input $i \in Z$. We use bold font to denote a vector (e.g. $\mathbf{V}$). For brevity, we will sometimes denote the elements in a vector as $\mathbf{V} = (V_i)_{i\in[q]} = (V_1, \ldots, V_q)$. Given a finite set S, we denote by $x \leftarrow S$ or $x \leftarrow\!\$\ S$ the sampling of an element uniformly at random from S. For an algorithm A, let $y \leftarrow A(x)$ be the process of running A on input x with access to uniformly random coins and assigning the result to y. With A^B we denote that A has oracle access to B. We assume all algorithms are polynomial-time (PPT) unless otherwise specified and public parameters are an implicit input to all algorithms in a scheme. We use $[a, b]$ to denote the range $\{a, a+1, \ldots, b\}$. Additionally, $|N|$ represents the length of N. By the symbol $\approx$, we denote indistinguishability between two distributions $D_1 \approx D_2$.

Diffie-Hellman Message Space. Over an asymmetric bilinear group, a pair $(M, \hat{N}) \in \mathbb{G}_1 \times \mathbb{G}_2$ is called a Diffie-Hellman (DH) message $\mathcal{M}_{\mathsf{DH}}$ [4, 46] if there exists $m \in \mathbb{Z}_p$ s.t. $M = P^m$ and $\hat{N} = \hat{P}^m$. One can efficiently verify whether $(M, \hat{N}) \in \mathcal{M}_{\mathsf{DH}}$ by checking $e(M, \hat{P}) = e(P, \hat{N})$. More formally, the message space are elements of the subgroup of $\mathbb{G}_1 \times \mathbb{G}_2$ defined as the image of the map $\psi' : \mathbb{Z}_p \to \mathbb{G}_1 \times \mathbb{G}_2 : x \mapsto (P^x, \hat{P}^x)$. One can easily extend the message space to a vector of Diffie-Hellman pairs $(\mathbf{M}, \hat{\mathbf{N}}) = (M_1, \ldots, M_q, \hat{N}_1, \ldots, \hat{N}_q)$ s.t. for all $i \in [q]$, $(M_i, \hat{N}_i) = (P^{m_i}, \hat{P}^{m_i}) \in \mathcal{M}_{\mathsf{DH}}$ for $m_i \in \mathbb{Z}_p$.

We note that related notations exist, such as equivalence classes [49]. Given the vector $\mathbf{M}$, the message $\mathbf{M}$ represents an equivalence class of all scaled messages $\mathbf{M}^\mu$. This allows for randomizing the message vector by switching between $\mathbf{M}$ and $\mathbf{M}^\mu$ for any non-zero μ. This concept extends to Diffie-Hellman message spaces, including Indexed DH [34] and tag-based DH message spaces [77], where randomized DH message vectors retain their validity with respect to their tags. Although latter variants are not directly related to our primitives, it provides useful context.

Definition 1 (Diffie-Hellman Exponent (q-DHE) [24]). *Let $\mathbb{G}$ be a finite cyclic group of order p, P be a generator of $\mathbb{G}$. The q-DHE problem is, given a tuple of elements $(P, B_1, \ldots, B_q, B_{q+2}, \ldots, B_{2q})$ such that $B_i = P^{\alpha^i}$ for $i = 1, \ldots, q, q+2, \ldots, 2q$ for $\alpha \leftarrow\!\$\ \mathbb{Z}_p^*$, to compute the missing group element $B_{q+1} = P^{\alpha^{q+1}}$ in the sequence. As shown in [55, 71], q-DHE can be modified so as to work in the asymmetric pairing setting, given $\{P^{\alpha^1}, \ldots, P^{\alpha^q}, P^{\alpha^{q+2}}, \ldots, P^{\alpha^{2q}}; \hat{P}^{\alpha^1}, \ldots, \hat{P}^{\alpha^q}; e(P, \hat{P})^{\alpha^{q+1}}\}$ it is hard to find $P^{\alpha^{q+1}}$.*

Generalized Version. *In our setting, we consider a further generalization in asymmetric bilinear groups, where the adversary is given: $\{P^{\alpha^1}, \ldots, P^{\alpha^q}, P^{\alpha^{q+2}}, \ldots, P^{\alpha^{2q}} \in \mathbb{G}_1;\ \hat{P}^{\alpha^1}, \ldots, \hat{P}^{\alpha^{2q}} \in \mathbb{G}_2\}$ and asked to compute $P^{\alpha^{q+1}} \in \mathbb{G}_1$.*

This assumption can be naturally framed as an instance of the Uber assumption framework for the AGM [18], where both the challenge and target exponents are univariate monomials in a formal variable X of degree at most $2q$. Specifically, the challenge set includes exponents of the form X^i for $i \in \{1, \ldots, q\} \cup \{q+2, \ldots, 2q\}$ for $\mathbb{G}_1$ elements, X^i for $i \in \{1, \ldots, 2q\}$ for $\mathbb{G}_2$ elements, and the target exponent is X^{q+1} (in $\mathbb{G}_1$), which is clearly linearly independent of the challenge polynomials. This satisfies the non-triviality condition of the Uber assumption. As shown in Corollary 3.6 (and related Lemmas 2.5 and 2.6) of [18], such instances in the AGM reduce to the $(2q-1, 2q)$-DLog problem.

Definition 2 ((m,n)-Discrete Logarithm [18, 50]). *Let $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ be an asymmetric bilinear group of prime order p. For integers m, n, the (m,n)-Discrete Logarithm ((m,n)-*dlog*) problem is, given*

$$(P, B_1 = P^{\alpha^1}, B_2 = P^{\alpha^2}, \ldots, B_m = P^{\alpha^m}, \hat{P}, \hat{B}_1 = \hat{P}^{\alpha^1}, \ldots, \hat{B}_n = \hat{P}^{\alpha^n}),$$

where $\alpha \leftarrow\!\$\, \mathbb{Z}_p^*$, $P \leftarrow\!\$\, \mathbb{G}_1$, $\hat{P} \leftarrow\!\$\, \mathbb{G}_2$, to compute $\alpha \in \mathbb{Z}_p^*$.

This assumption is similar to the n-discrete logarithm assumption considered in [50], except that powers α^i are given in the exponents in both groups $\mathbb{G}_1$ and $\mathbb{G}_2$, where this case is considered in Lemma 2.6 in [18].

Definition 3 ((Double Pairing Problem (DBP)) [4]). *We say the double pairing problem holds relative to* BG *if for any PPT algorithm $\mathcal{A}$*

$$\Pr\left[\begin{array}{c} \mathsf{BG} \leftarrow \mathsf{BGSetup}(1^\lambda); h_z \leftarrow\!\$\, \mathbb{G}_1 \setminus \{0\} \\ (\hat{Z}, \hat{R}) \leftarrow \mathcal{A}(\mathsf{BG}, h_z) : (\hat{Z}, \hat{R}) \in \mathbb{G}_2 \setminus \{0\} \text{ and } 1 = e(h_z, \hat{Z})e(P, \hat{R}) \end{array}\right] \leq \epsilon(\lambda)$$

This assumption follows from the decisional Diffie-Hellman (DDH) assumption in $\mathbb{G}_1$. By swapping $\mathbb{G}_1$ and $\mathbb{G}_2$ (assuming DDH in $\mathbb{G}_2$), we obtain a dual assumption. Hence, if DDH holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$, DBP holds in both groups.

Remark 1. Our CRS naturally corresponds to a generalized q-DHE challenge with the $(q+1)$-st power omitted. In the AGM, this can be viewed as an instance of the Uber assumption framework, where the challenge exponents are the monomials X^i for $i \in \{1, \ldots, q\} \cup \{q+2, \ldots, 2q\}$ in $\mathbb{G}_1$ and X^i for $i \in \{1, \ldots, 2q\}$ in $\mathbb{G}_2$, while the target exponent is the missing monomial X^{q+1}. This Uber assumption reduces in the AGM to the (m,n)-DLog assumption. Hence, our AGM proofs may equivalently be viewed as reductions to (m,n)-DLog.

2.1 Structure-Preserving Commitments

A structure-preserving commitment scheme was proposed by Abe et al. [4]. The public key consists of $q+1$ group elements $(H, U_1, \ldots, U_q)$ from $\mathbb{G}_1$. To commit to $(\hat{M}_1, \ldots, \hat{M}_q) \in \mathbb{G}_2$, a random element $\hat{R} \in \mathbb{G}_2$ is selected, and the commitment is computed as $C = e(H, \hat{R}) \prod_{i=1}^{q} e(U_i, \hat{M}_i)$. This commitment scheme is computationally binding under the DBP assumption. Moreover, it is both a length-reducing (constant-size) scheme and a homomorphic trapdoor commitment.

2.2 The Algebraic Group Model

The Algebraic Group Model (AGM) [50] is an idealized model, in which all algorithms inducing adversaries are meant to be algebraic. Algebraic algorithms generalize the notion of a generic algorithm [86] in the following sense, that whenever adversaries compute a group element, they do so by means of generic operations, by computing linear combinations of previously known group elements. So, whenever they output a group element $X \in \mathbb{G}$, they also output a representation $\{\alpha_i\}_{i=1}^{q}$ of $X = \prod_{i=1}^{q} g_i^{\alpha_i}$ as a function of previously observed group elements $(g_1, \ldots, g_q) \in \mathbb{G}^q$ in the same group. In contrast to generic algorithms, algebraic algorithms can exploit the group structure and learn more than in the generic group model. The AGM is a very powerful tool to establish the security of efficient protocols via reductions.

2.3 Zero-Knowledge Proofs of Knowledge

We define zero-knowledge proofs of knowledge (ZKPOK) and discuss non-interactive versions thereof (NIZK).

ZKPoK. Let $L_{\mathbb{R}} = \{x \mid \exists\, w : (x,w) \in \mathbb{R}\} \subseteq \{0,1\}^*$ be a formal language, where $\mathbb{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ is a binary, polynomial-time (witness) relation. For such a relation, the membership of $x \in L_{\mathbb{R}}$ can be decided in polynomial time (in $|x|$) when given a witness w of length polynomial in $|x|$ certifying $(x,w) \in \mathbb{R}$. We assume an interactive protocol $(\mathcal{P},\mathcal{V})$ between a prover $\mathcal{P}$ and a PPT verifier $\mathcal{V}$ and denote the outcome of the protocol as $(\cdot,b) \leftarrow \big(\mathcal{P}(\cdot,\cdot),\mathcal{V}(\cdot)\big)$ where $b = 0$ indicates that $\mathcal{V}$ rejects and $b = 1$ that it accepts the conversation with $\mathcal{P}$. We require the following properties completeness, zero knowledge (ZK) and soundness. For the formal definitions, see e.g. [54].

Non-Interactive Zero-Knowedge Proofs (of Knowledge). One can use the Fiat-Shamir heuristic [43] to transform any Sigma protocol into a non-interactive zero-knowledge proof of knowledge (NIZK). Whenever one requires multiple-extractions in a security proof, a standard measure is to opt for interactive ZKPOK. We however note that when willing to pay some extra costs, one could instead use straight-line extractable NIZK, e.g., obtained via Fischlin's transformation [44].

Definition 4 (Completeness). *We call an interactive protocol $(\mathcal{P},\mathcal{V})$ for a relation $\mathbb{R}$ complete if for all $x \in L_{\mathbb{R}}$ and w s.t $(x,w) \in R$ we have $(\cdot,1) \leftarrow \big(\mathcal{P}(x,w),\mathcal{V}(x)\big)$ with probability 1.*

Definition 5 (Zero knowledge (ZK)). *An $(\mathcal{P},\mathcal{V})$ for a language L is ZK if for any (malicious) verifier $\mathcal{V}^*$, there exists a PPT algorithm $\mathcal{S}$ (the simulator) such that:*

$$\{\mathcal{S}(x)\}_{x\in L} \approx \{< (\mathcal{P},\mathcal{V}^\star)(x) >\}_{x\in L},$$

where $(\mathcal{P},\mathcal{V}^*)(x)$ shows the transcript of communication between $\mathcal{P}$ and $\mathcal{V}^\star$ on the common input x.

Definition 6 (Knowledge soundness). *We say that $(\mathcal{P},\mathcal{V})$ is a proof of knowledge (PoK) relative to an NP relation $\mathbb{R}$ if for any malicious prover $\mathcal{P}^*$ such that $(\cdot,1) \leftarrow \big(\mathcal{P}^*(x),\mathcal{V}(x)\big)$ with probability greater than ϵ there exists a PPT knowledge extractor K (with rewinding black-box access to $\mathcal{P}^*$) such that $K^{\mathcal{P}^*}(x)$ returns a value w satisfying $(x,w) \in \mathbb{R}$ with probability polynomial in ϵ.*

2.4 Vector Commitments

Definition 7 (Vector Commitment [29]). *A vector commitment is a tuple of algorithms defined as follows:*

- $\mathsf{KeyGen}(1^\lambda, q)$: Given the security parameter λ and the size q of the committed vector, the key generation outputs some public parameters pp (which implicitly define the message space $\mathcal{M}$ and are input to all algorithms).
- $\mathsf{Commit}(m_1,\ldots,m_q)$: On input a vector of q messages $(m_1,\ldots,m_q) \in \mathcal{M}$ and the public parameters pp, the committing algorithm outputs a commitment com and auxiliary information aux.
- $\mathsf{Open}(m,i,\mathsf{aux})$: This algorithm is run by the committer to produce a proof π_i that m is the i-th committed message.
- $\mathsf{Verify}(\mathsf{com},m,i,\pi_i)$: The verification algorithm accepts (i.e., it outputs 1) if and only if π_i is a valid proof that com was created for a vector $(m_1,\ldots,m_q)$ such that $m = m_i$.

In addition, some applications require an update property (to update the commitment and the corresponding openings), which is defined using two additional (and optional) algorithms:

- $\mathsf{Update}(\mathsf{com},m,m',i)$: This algorithm is run by the committer who produced com and wants to update it by changing the i-th message to m'. It takes as input the old message m, the new message m', and the position i. The algorithm outputs a new commitment com' together with update information U. In particular, notice that in the case when some updates have occurred, the auxiliary information aux can include the update information produced by these updates.
- $\mathsf{ProofUpdate}(\mathsf{com},\pi_j,m',i,U)$: This algorithm is run by any user who holds a proof π_j for the message at position j with respect to com. It allows the user to compute an updated proof π'_j (and an updated commitment com'), such that π'_j will be valid with respect to com', where m' is the new message at position i. The value U contains the update information needed to compute these updated values.

VC should satisfy the property of position binding as follows:

Position Binding: This property ensures that a PPT adversary (with knowledge of pp) cannot produce two proofs for the same position in a fixed commitment com that open to different values. There are two flavors of this property:

- **Weak Position Binding:** Holds only for honestly generated commitments.
- **Position Binding:** Holds even for adversarially generated commitments.

Weak binding suffices for stateless validation (e.g., in Byzantine agreement on updates [82]) and can be easily achieved using accumulators. Position binding is essential in adversarial scenarios, like transparency logs, where commitments are generated by log servers.

Definition 8 (Position Binding [29]). *A* VC *satisfies position binding if,* $\forall i = 1, \ldots, q$*, and for every PPT adversary* $\mathcal{A}$*, the following probability (taken over all honestly generated* pp*) is at most negligible:*

$$\Pr\left[\begin{array}{c}\mathsf{Verify}(\mathsf{com}, m, i, \pi) = 1 \wedge \\ \mathsf{Verify}(\mathsf{com}, m', i, \pi') = 1 \wedge \\ m \neq m'\end{array}\middle|(\mathsf{com}, m, m', i, \pi, \pi') \leftarrow \mathcal{A}(\mathsf{pp})\right] \leq \epsilon(\lambda)$$

If we relax the above definition to hold only for honestly generated commitments com, we obtain the weak position binding notion.

Definition 9 (Weak Position Binding [28,55]). *A* VC *satisfies weak position binding if* $\forall i = 1, \ldots, q$ *and for every PPT adversary* $\mathcal{A}$*, the following probability (which is taken over all honestly generated parameters) is at most negligible:*

$$\Pr\left[\begin{array}{c}\mathsf{Verify}(\mathsf{com}, m, i, \pi) = 1 \wedge \mathsf{Verify}(\mathsf{com}, m', i, \pi') = 1 \wedge m \neq m' \; st: \\ (\mathsf{pp}) \leftarrow \mathsf{KeyGen}(1^\lambda), (\mathbf{m}, \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pp}), \\ (\mathsf{com}, \mathsf{aux}) \leftarrow \mathsf{Commit}(\mathbf{m}), (i, m, \pi, m', \pi') \leftarrow \mathcal{A}(\mathsf{com}, \mathsf{state})\end{array}\right] \leq \epsilon(\lambda).$$

Indeed, our constructions will satisfy a variant of this notion, where the adversary is also given aux in the second phase.

Conciseness: This property requires that both the commitment and the opening for a position i are of constant size, independent of the vectors length.

Hiding. VC can also be required to be hiding, meaning that one should not be able to distinguish whether a commitment was created to $(m_1, \ldots, m_q)$ or to $(m'_1, \ldots, m'_q)$ (both chosen by the adversary), even after seeing some proofs for positions where $m_i = m'_i$. However, hiding is not a crucial property in the realization of VC and typically not considered.

2.5 Accumulators

We provide a formal a definition of static accumulators based on the definition given by Derler et al. in [38].

Definition 10 (Static Accumulator). *A static accumulator is a tuple of algorithms defined as follows:*

$\mathsf{Setup}(1^\lambda, q)$: Take a security parameter λ and a parameter q. If $q \neq \infty$, then q is an upper bound on the number of elements to be accumulated. Return a key pair $(\mathsf{sk}_{acc}, \mathsf{pk}_{acc})$, where $\mathsf{sk}_{acc} = \emptyset$ if no trapdoor exists.

$\mathsf{Eval}((\mathsf{sk}_{acc}, \mathsf{pk}_{acc}), \mathcal{X})$: This (probabilistic) algorithm takes a key pair $(\mathsf{sk}_{acc}, \mathsf{pk}_{acc})$ and a set $\mathcal{X}$ to be accumulated and returns an accumulator $\mathsf{Acc}_{\mathcal{X}}$ together with some auxiliary information aux.

$\mathsf{WitCreate}((\mathsf{sk}_{acc}, \mathsf{pk}_{acc}), \mathsf{Acc}_{\mathcal{X}}, \mathsf{aux}, x_i)$: This algorithm takes a key pair $(\mathsf{sk}_{acc}, \mathsf{pk}_{acc})$, an accumulator $\mathsf{Acc}_{\mathcal{X}}$, auxiliary information aux, and a value x_i. It returns $\perp$, if $x_i \notin \mathcal{X}$, and a witness wit_{x_i} for x_i otherwise.

$\mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}_{\mathcal{X}}, wit_{x_i}, x_i)$: This algorithm takes a public key pk_{acc}, an accumulator $\mathsf{Acc}_{\mathcal{X}}$, a witness wit_{x_i}, and a value x_i. It returns *true* if wit_{x_i} is a witness for $x_i \in \mathcal{X}$ and *false* otherwise.

The above definition focuses on static accumulators. In contrast, dynamic accumulators enable the accumulated value and witnesses to be publicly updated whenever an element is added or removed from the set. Various properties, such as being trapdoorless, universal, or supporting subset queries, are associated with this primitive. For a comprehensive list of definitions, functionalities, and properties, we refer to [17,38]. Here, we only consider the basic properties.

Security. One requires collision resistance which states that it should be computationally infeasible to find a witness for any non-accumulated value $x \notin \mathcal{X}$.

Definition 11 (Collision resistance [17,38]). *An accumulator scheme is said to satisfy collision resistance if for all PPT adversaries $\mathcal{A}$, the following advantage is negligible:*

$$\Pr\left[\begin{array}{l}(\mathsf{sk}_{acc}, \mathsf{pk}_{acc}) \leftarrow \mathsf{Setup}(1^\lambda, q), (\mathsf{Acc}_{\mathcal{X}}, \mathsf{aux}) \leftarrow \mathsf{Eval}((\mathsf{sk}_{acc}, \mathsf{pk}_{acc}), \mathcal{X}), \\ (\mathcal{X}, wit_{x_i}, x_i) \leftarrow \mathcal{A}^{\mathcal{O}^W}(\mathsf{pk}_{acc}) : \mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}_{\mathcal{X}}, wit_{x_i}, x_i) = 1 \wedge x_i \notin \mathcal{X}\end{array}\right]$$

where $\mathcal{O}^{\mathrm{W}}$ represent an oracle for the algorithm WitCreate. An adversary is allowed to query them an arbitrary number of times. Note that if $\mathcal{O}^{\mathrm{W}}$ is queried for an element $x \notin \mathcal{X}$, the oracle outputs a reject symbol $\perp$.

2.6 Ring Signatures

Ring signature (RS) schemes [85] allow members of an ad-hoc group $\mathcal{R}$ (known as a ring) to anonymously sign messages on behalf of this group defined by their public keys. While it is possible to verify a ring signature against the public keys associated with $\mathcal{R}$, determining the actual signer remains infeasible, thus guaranteeing unconditional anonymity. This characteristic makes ring signatures particularly useful in various applications, especially in whistleblowing and ensuring transaction privacy in cryptocurrencies.

Definition 12 (Ring Signature Scheme). *A ring signature scheme* RS *is a tuple of the following algorithms:*

- $\mathsf{Setup}(1^\lambda, q)$: Takes as input a security parameter λ, an upper bound $q = \mathsf{poly}(\lambda)$ on the ring size and outputs public parameters pp.
- $\mathsf{KeyGen}(\mathsf{pp}, i)$: Takes as input the public parameter pp, an index $i \in [q]$ and outputs a key pair $(\mathsf{sk}_i, \mathsf{pk}_i)$.
- $\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}_i, m, \mathcal{R})$: Takes as input the public parameters pp, a secret key sk_i, a message $m \in \mathcal{M}$, and a ring $\mathcal{R} = (\mathsf{pk}_j)_{j\in[n]}$ of n public keys such that $\mathsf{pk}_i \in \mathcal{R}$. It outputs a signature σ.
- $\mathsf{Verify}(\mathsf{pp}, m, \sigma, \mathcal{R})$: Takes as input the public parameters pp, a message $m \in \mathcal{M}$, a signature σ, and a ring $\mathcal{R}$. It outputs a bit $b \in \{0,1\}$, If σ is valid with respect to the ring $\mathcal{R}$.

Remark 2. In contrast to standard ring signature schemes, key generation in our construction is index-aware. Each user generates a key pair with respect to a unique index $i \in [q]$, and at most one key pair may be generated per index. This reflects the structure of the underlying accumulator, where each public key occupies a fixed position. We assume that indices are globally coordinated. Moreover, in case of rings of size $\ell < q$, they are internally padded using the padding mechanism of the underlying accumulator.

We formally define ring signature schemes in alignment with [22,40].

Unforgeability. This property requires that without any secret key sk_i that corresponds to a public key $\mathsf{pk}_i \in \mathcal{R}$, it is infeasible to produce valid signatures with respect to arbitrary such rings $\mathcal{R}$.

Definition 13 (Unforgeability [22, 40]). *A* RS *scheme provides unforgeability if for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\varepsilon(\cdot)$ such that:*

$$\Pr\left[\begin{array}{c}\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q), \\ \{(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, i)\}_{i\in[q]}, \\ \mathcal{O} \leftarrow \{\mathsf{Sig}(\cdot,\cdot,\cdot), \mathsf{Key}(\cdot)\}, \\ (m^*, \sigma^*, \mathcal{R}^*) \leftarrow \mathcal{A}^{\mathcal{O}}(\{\mathsf{pk}_i\}_{i\in[q]})\end{array} : \begin{array}{c}\mathsf{Verify}(m^*, \sigma^*, \mathcal{R}^*) = 1 \wedge \\ (\cdot, m^*, \mathcal{R}^*) \notin Q^{\mathsf{Sig}} \wedge \\ \mathcal{R}^* \subseteq \{\mathsf{pk}_i\}_{i\in[q]\setminus Q_{\mathsf{Key}}}\end{array}\right] \leq \varepsilon(\lambda),$$

where $\mathsf{Sig}(i, m, \mathcal{R}) := \mathsf{Sign}(\mathsf{sk}_i, m, \mathcal{R})$, Sig *returns* $\perp$ *if* $\mathsf{pk}_i \notin \mathcal{R} \vee i \notin [q]$, *and* Q^{Sig} *records the queries to* Sig. *Furthermore,* $\mathsf{Key}(i)$ *returns* sk_i *and* Q_{Key} *records the queries to* Key.

Anonymity. This property ensures that it is computationally infeasible to determine which ring member generated a particular signature, provided there are at least two honest members in the ring. Our anonymity notion follows the one in [40] that is corresponds to the strongest definition in [22], known as anonymity against full key exposure.

Definition 14 (Anonymity [22, 40]). *A* RS *scheme provides anonymity if for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\epsilon(\cdot)$ such that:*

$$\Pr\left[\begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q), b \leftarrow \{0,1\}, \\ \{(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, i)\}_{i\in[q]}, \\ \mathcal{O} \leftarrow \{\mathsf{Sig}(\cdot,\cdot,\cdot)\}, \\ (m, (j_0, j_1) \in [q], \mathcal{R}, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}}(\{\mathsf{pk}_i\}_{i\in[q]}), \\ \sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_{j_b}, m, \mathcal{R}), \end{array} : \begin{array}{c} b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{st}, \sigma, \{\mathsf{sk}_i\}_{i\in[q]}) \\ b = b' \wedge \{\mathsf{pk}_{j_0}, \mathsf{pk}_{j_1}\} \subseteq \mathcal{R} \subseteq \{\mathsf{pk}_i\}_{i\in[q]} \end{array}\right] \leq \frac{1}{2} + \varepsilon(\lambda)$$

where $\mathsf{Sig}(i, m, \mathcal{R}) := \mathsf{Sign}(\mathsf{sk}_i, m, \mathcal{R})$ *if* $\mathsf{pk}_i \notin \mathcal{R} \vee i \notin [q]$ *return* $\perp$.

2.7 Structure-Preserving and Key-Homomorphic Signatures

We recall two types of signatures, which we will use throughout the rest of the paper. The first concept is key-homomorphic signatures [40], which, informally, allows for the randomization of keys and the adaptation of signatures to these randomized keys. Before providing the formal definitions, we first need to recall the definition of secret key to public key Homomorphism.

Definition 15 (Secret Key to Public Key Homomorphism [40]). *A signature scheme Σ provides a secret key to public key homomorphism if there exists an efficiently computable map $\mu : \mathbb{H} \to \mathbb{E}$ such that for all $\mathsf{sk}, \mathsf{sk}' \in \mathbb{H}$, it holds that: $\mu(\mathsf{sk} + \mathsf{sk}') = \mu(\mathsf{sk}) \cdot \mu(\mathsf{sk}')$, and for all $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}$, it holds that: $\mathsf{pk} = \mu(\mathsf{sk})$.*

Definition 16 (Key-Homomorphic Signatures [40]). *A signature scheme* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is called* key-homomorphic *if it provides a secret key to public key homomorphism (see Def. 15) and an additional PPT algorithm* Adapt*, defined as follows:*

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$*: Takes a public key* pk*, a message m, a signature σ, and a function Δ as input and outputs a public key* pk' *and a signature σ'.*

The following property must hold: For all $\Delta \in \mathbb{H}$, all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, all messages $m \in \mathcal{M}$, and all σ satisfying $\mathsf{Verify}(\mathsf{pk}, m, \sigma) = 1$, if we compute: $(\mathsf{pk}', \sigma') \leftarrow \mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$ then it holds with probability 1 that: $\Pr[\mathsf{Verify}(\mathsf{pk}', m, \sigma') = 1] = 1$ and $\mathsf{pk}' = \mu(\Delta) \cdot \mathsf{pk}$.

The key property of key-homomorphic signatures is whether adapted signatures are indistinguishable from freshly generated signatures, which is defined as follows:

Definition 17 (Adaptability of Signatures). *A key-homomorphic signature scheme provides* adaptability of signatures *if for every $\lambda \in \mathbb{N}$ and every message $m \in \mathcal{M}$, the following distributions are identical: $\big((\mathsf{sk}, \mathsf{pk}), \mathsf{Adapt}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m), \Delta)\big)$, where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $\Delta \leftarrow\!\!\$\ \mathbb{H}$, $\big((\mathsf{sk}, \mu(\mathsf{sk})), (\mu(\mathsf{sk}) \cdot \mu(\Delta), \mathsf{Sign}(\mathsf{sk} + \Delta, m))\big)$, where $\mathsf{sk} \leftarrow\!\!\$\ \mathbb{H}$ and $\Delta \leftarrow\!\!\$\ \mathbb{H}$.*

We recall BLS signatures as an instantiation of key-homomorphic signatures.

Definition 18 (Type-3 BLS Signatures).

$\mathsf{Setup}(1^\lambda)$*: Run* $\mathsf{BG} \leftarrow \mathsf{BGSetup}(1^\lambda, 3)$*, choose a hash function $H : \mathcal{M} \to \mathbb{G}_1$ uniformly at random from a hash function family $\{H_k\}_k$, and set* $\mathsf{pp} \leftarrow (\mathsf{BG}, H)$.

$\mathsf{KeyGen}(\mathsf{pp})$*: Parse* pp *as* (BG, H)*, choose $x \leftarrow\!\!\$\ \mathbb{Z}_p$, set* $\mathsf{pk} \leftarrow \hat{P}^x$, $\mathsf{sk} \leftarrow x$*, and return* $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, m)$*: Parse* sk *as $x \in \mathbb{Z}_p^*$ and return $\sigma \leftarrow H(m)^x$.*

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$*: Parse* pk *as $\hat{P}^x$, verify whether $e(H(m), \hat{P}^x) = e(\sigma, \hat{P})$, and return 1 if true, otherwise return 0.*

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$*: Let $\Delta \in \mathbb{Z}_p$ and $\mathsf{pk} = \hat{P}^x$. Return (pk', σ'), where $\mathsf{pk}' \leftarrow \hat{P}^x \cdot \hat{P}^\Delta$, $\sigma' \leftarrow \sigma \cdot H(m)^\Delta$. It is immediate that adapted signatures are identical to fresh signatures under $\mathsf{pk}' = \hat{P}^{x+\Delta}$.*

Structure-preserving signatures (SPSs) [4] are signatures where public keys and the signatures are source group elements of a bilinear group, and verification will be done only using group-membership tests and pairing-product equations. We recall the structure-preserving signature scheme FHS [49], which also allows for the efficient and joint randomization of messages and signatures in public, where the message space consists of group-element vectors. In this paper, message randomization is not required; thus, we remove the randomization algorithms and the equivalence class relation concept and to sign messages $\mathbf{M}$ of length $\ell - 1$ one signs message $(\mathbf{M}, P)$ and in verificaton checks whether the ℓ'th element of the message equals P. For further details, see [49].

$\mathsf{BG}_{\mathcal{R}}(1^{\lambda})$: This algorithm on input of a security parameter λ outputs a bilinear group BG.
$\mathsf{KeyGen}(\mathsf{BG}, \ell)$: This algorithm on input of a bilinear group BG and a vector length $\ell > 1$ outputs a key pair $(\mathsf{sk}, \mathsf{vk})$ as $\mathsf{sk} \leftarrow (x_i)_{i\in[\ell]}$, and the public key $\mathsf{vk} \leftarrow (\hat{X}_i)_{i\in[\ell]} = (\hat{P}^{x_i})_{i\in[\ell]}$.
$\mathsf{Sign}(\mathsf{sk}, \mathbf{M})$: This algorithm on input a $\mathbf{M} \in (\mathbb{G}_i^*)^{\ell-1}$and a secret key sk, sets $\mathbf{M}' = (\mathbf{M}, P)$ outputs a signature σ as $\sigma = (Z \leftarrow (\prod_{i\in[\ell]} M_i'^{x_i})^{y}, Y \leftarrow P^{\frac{1}{y}}, \hat{Y} \leftarrow \hat{P}^{\frac{1}{y}})$.
$\mathsf{Verify}(\mathbf{M}, \sigma, \mathsf{vk})$: This algorithm on input of a representative $\mathbf{M} \in (\mathbb{G}_i^*)^{\ell}$, a signature σ and a public key vk outputs a bit $b \in \{0, 1\}$ if $\mathbf{M}_\ell = P$ and $\prod_{i\in[\ell]} e(M_i, \hat{X}_i) = e(Z, \hat{Y}) \wedge e(Y, \hat{P}) = e(P, \hat{Y})$.

3 Shrinking SP Commitments for DH Messages

We begin by constructing *constant-size* and *strict* structure-preserving commitments for group elements using a variant of Diffie-Hellman (DH) messages. By *strict*, we refer to the definition provided in [6], which means that the messages, commitments, and openings are all confined to the source groups $\mathbb{G}_1$ and $\mathbb{G}_2$.

Our construction is notable for bypassing the impossibility result of Abe et al. [6], which establishes that strictly structure-preserving commitments to source-group elements cannot be smaller than the input message size and scale linearly with the number of group elements.

In contrast, in a relaxed structure-preserving setting, constant-size commitments to group elements can be achieved by allowing elements from the target group $\mathbb{G}_T$ in the commitment [5, 6]. While including $\mathbb{G}_T$ elements is acceptable when only witness indistinguishability is required for accompanying Groth-Sahai (GS) proofs, it becomes problematic when zero-knowledge is necessary (see [6]). Therefore, ensuring that group-to-group commitments remain entirely within the source groups is crucial for achieving zero-knowledge, and it also allow one to commit to other commitments, latter being very interesting for applications. Another way of bypassing this impossibility is done by Abe et al. in [7], who construct SP commitments that are shrinking in a relaxed binding setting, e.g., where the message space for committing is $\mathbb{Z}_p$ and the one for verification is $\mathbb{G}_1$.

Our approach is inspired by circumventing impossibility results and lower bounds in SPS [51,52], achieved by switching from arbitrary group elements in the message space to a more structured message space and in particular a Diffie-Hellman (DH) message space [3,46]. This has also recently been used to construct the first threshold SPS in [34]. In doing so we obtain shrinking strictly structure-preserving (group-to-group) commitments. The resulting SP commitment scheme is similar in nature to the γ-binding commitment scheme of [7], but it does not output a trapdoor and is strictly structure-preserving; that is, messages, commitments, and openings are all elements of the source groups $\mathbb{G}_1$ and $\mathbb{G}_2$.

We first define a new DH message space $\mathcal{M}_{\mathsf{DH}}$ and then present a construction for shrinking strictly structure-preserving (group-to-group) commitments based on this new DH message type.

New DH Message Space $\mathcal{M}_{\mathsf{DH}}$. We slightly adapt the DH message technique (cf. Sec. 2) to use a distinct base element in $\mathbb{G}_1$ for each index i:

Definition 19 (DH Message Space ($\mathcal{M}_{\mathsf{DH}}^{new}$)). *Let the public parameters be a vector $\mathbf{X}$ of random elements in $\mathbb{G}_1$ (e.g., $\mathbf{X} = (P^{x_i})_{i\in[q]}$). We then define $\mathcal{M}_{\mathsf{DH}}^{new}$ as a DH message space, if the following property holds: For the message vector $(\mathbf{M}, \hat{\mathbf{N}}) = (M_1, \ldots, M_q, \hat{N}_1, \ldots, \hat{N}_q)$ there exist $m_i \in \mathbb{Z}_p$ $(1 \leq i \leq q)$ s.t. $M_i = X_i^{m_i}$ and $\hat{N}_i = \hat{P}^{m_i}$ for all i.*

Note that membership in this message space can be efficiently checked by $e(M_i, \hat{P}) = e(X_i, \hat{N}_i)$. We can obtain the plain messages $\hat{P}^{m_i}$ in $\mathbb{G}_1$ as well by simply switching the vector $\mathbf{X}$ to reside in $\mathbb{G}_2$ and keeping the vector $\mathbf{M} = P^{m_i}$ in $\mathbb{G}_1$.

Construction. We introduce our group-to-group commitments, where the messages, commitments are all restricted to the source groups $\mathbb{G}_1$ and $\mathbb{G}_2$. Despite this, the construction remains concise and compact, achieving a compactness property.

Scheme 1 (Shrinking Strictly Structure-Preserving Commitment) *Our commitment scheme is composed of the following PPT algorithms:*

$\mathsf{KeyGen}(1^\lambda)$: Run $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P, \hat{P}) \leftarrow \mathsf{BGSetup}(1^\lambda)$. For $i \in [q]$, choose $X_i \leftarrow\!\!\$\ \mathbb{G}_1$. Note that $\mathbf{X}$ does not need to have structure, and therefore, we also can use a random oracle to generate these elements.
Output the commitment key $\mathsf{pk} := (\mathsf{BG}, X_1, \ldots, X_q)$.

$\mathsf{Commit}(\mathsf{pk}, msg)$: Parse msg as $((M_1, \hat{N}_1), \ldots, (M_q, \hat{N}_q)) \in \mathcal{M}_{\mathsf{DH}}^{new}$ (check if they are generated correctly by $e(M_i, \hat{P}) = e(X_i, \hat{N}_i)$ for $i \in [q]$). Choose $r \leftarrow\!\!\$\ \mathbb{Z}_p$, and compute

$$\mathsf{com} := P^r \cdot \prod_{i=1}^{q} M_i \ \wedge\ \mathsf{Open} := (\hat{R} = \hat{P}^r).$$

$\mathsf{Verify}(\mathsf{pk}, \mathsf{com}, msg, \mathsf{Open})$: Parse msg as $((M_1, \hat{N}_1), \ldots, (M_q, \hat{N}_q)) \in \mathcal{M}_{\mathsf{DH}}^{new}$, and Open as $\hat{R}$. Check for well-formedness of the messages by $e(M_i, \hat{P}) = e(X_i, \hat{N}_i)$ for $i \in [q]$, and abort otherwise. Output 1 if and only if:

$$e(\mathsf{com}, \hat{P}) = e(P, \hat{R}) \prod_{i=1}^{q} e(X_i, \hat{N}_i)\,.$$

Theorem 1. *Scheme 1 is perfectly hiding. Furthermore, it is binding under the DBP assumption.*

Proof. **Hiding.** This property is obvious as P^r is a uniformly random element.

Binding. For an adversary $\mathcal{A}$ breaking the binding property of the commitment scheme, consider the following adversary $\mathcal{B}$: On input a DBP challenge (BG, h_z), $\mathcal{B}$ sets $X_i := h_z^{x_i} \cdot P^{\xi_i}$ for random $(x_i, \xi_i) \leftarrow\!\!\$\ \mathbb{Z}_p$ for $i \in [q]$. $\mathcal{B}$ aborts if $X_i = 1$ for any i, but this happens only with negligible probability. $\mathcal{B}$ then runs $\mathcal{A}$ on $\mathsf{pk} = (\mathsf{BG}, X_1, \ldots, X_q)$. If $\mathcal{A}$ returns a commitment com together with two different valid messages and openings $(\hat{N}_1, \ldots, \hat{N}_q, \hat{R})$ and $(\hat{N}'_1, \ldots, \hat{N}'_q, \hat{R}')$, then $\mathcal{B}$ computes

$$\hat{Z}^* := \prod_{i=1}^{q} \left(\frac{\hat{N}_i}{\hat{N}'_i}\right)^{x_i}, \quad \hat{R}^* := \frac{\hat{R}}{\hat{R}'} \prod_{i=1}^{q} \left(\frac{\hat{N}_i}{\hat{N}'_i}\right)^{\xi_i}.$$

As it holds that $e(\mathsf{com}, \hat{P}) = e(P, \hat{R}) \prod_{i=1}^{q} e(X_i, \hat{N}_i) = e(P, \hat{R}') \prod_{i=1}^{q} e(X_i, \hat{N}'_i)$, and by the structure of X_i, it follows that

$$\begin{aligned}
1 &= e\left(P, \frac{\hat{R}}{\hat{R}'}\right) \cdot \prod_{i=1}^{q} e\left(h_z^{x_i} \cdot P^{\xi_i}, \frac{\hat{N}_i}{\hat{N}'_i}\right) \\
&= e\left(P, \frac{\hat{R}}{\hat{R}'}\right) \cdot e\left(P, \prod_{i=1}^{q} \left(\frac{\hat{N}_i}{\hat{N}'_i}\right)^{\xi_i}\right) \cdot e\left(h_z, \prod_{i=1}^{q} \left(\frac{\hat{N}_i}{\hat{N}'_i}\right)^{x_i}\right) \\
&= e\left(P, \frac{\hat{R}}{\hat{R}'} \prod_{i=1}^{q} \left(\frac{\hat{N}_i}{\hat{N}'_i}\right)^{\xi_i}\right) \cdot \left(h_z, \prod_{i=1}^{q} \left(\frac{\hat{N}_i}{\hat{N}'_i}\right)^{x_i}\right) = e\left(P, \hat{R}^*\right) \cdot e\left(h_z, \hat{Z}^*\right).
\end{aligned}$$

We first verify that the simulated inputs to $\mathcal{A}$ are correctly distributed. In KeyGen, each X_i distributes uniformly over $\mathbb{G}_1$. Thus, the simulated parameters are statistically close to the real ones.

Moreover, note that since a valid output from $\mathcal{A}$ satisfies $msg \neq msg'$, there exists an index $i^* \in [q]$ such that $\hat{N}_{i^*} \neq \hat{N}'_{i^*}$. The view of $\mathcal{A}$ is independent of x_{i^*} as x_{i^*} is information theoretically hidden in X_{i^*}. Thus $\hat{Z}^*$ follows the distribution of $\left(\frac{\hat{N}_{i^*}}{\hat{N}'_{i^*}}\right)^{x_i^*}$. Since $\frac{\hat{N}_{i^*}}{\hat{N}'_{i^*}} \neq 1$ and x_{i^*} is uniform over $\mathbb{Z}_p^*$, we conclude that $\hat{Z}^* \neq 1$ with overwhelming probability.

Thus $\hat{Z}^*, \hat{R}^* \in \mathbb{G}_2^*$ is a solution to the given DBP instance, and $\mathcal{B}$ succeeds with roughly the same probability as $\mathcal{A}$. □

4 Structure Preserving Vector Commitments

We next introduce our structure preserving vector commitment (SPVC) scheme over a message space $\mathbf{M} \in \mathbb{G}_1$ (or $\mathbb{G}_2$) within the standard position-binding model based on the q-DHE assumption, referred to as q-DHE SPVC. Before diving into the construction, we define a structured message space built on the q-DHE parameters, which can be viewed as a pp (and commitment keys). This design of a message structure on top of a pp is inspired by [56] (although their pp and the way it is used differ from ours). Indeed, this message space enables the construction of the SPVC based on the q-DHE VC schemes [55, 71] in the strong model, i.e., transforming these constructions for $m \in \mathbb{Z}_p$ into a structure-preserving commitment. Furthermore, it allows us to extract the discrete logarithm of the messages in the proof, reducing the construction to q-DHE.

Message Structure. Each message msg_i consists of $q+2$ elements in $\mathbb{G}_1$, partitioned into a main message component $\mathbf{M}$ and a tag vector $\mathbf{T}_i$. Specifically, $\mathbf{M}_i := M_{i0}$ contains the message element, and $\mathbf{T}_i := \left(M_{i1}, \{T_{ij}\}_{j\in([i+1,i+q]\setminus\{q+1\})\cup\{2q\}}\right)$ contains $q+1$ tag elements. The structure of these vectors, and hence the message space $\mathcal{M}^{\mathsf{pp},q}$, is determined by the q-DHE parameters. The message space $\mathcal{M}^{\mathsf{pp},q}$ is defined according to these parameters. We stress that only M_{i0} represents the underlying message to be committed and later verified. The element M_{i1} is deterministically derived from M_{i0} and is used solely to enforce the algebraic structure of the message space and to construct the commitment. Consequently, although we sometimes write $\mathbf{M}_i = (M_{i0}, M_{i1})$ for notational convenience, M_{i1} is conceptually regarded as part of the tag structure together with the elements T_{ij}.

$$\mathcal{M}^{\mathsf{pp},q} = \left\{ \begin{array}{l} ((\mathbf{M}_1, \mathbf{T}_1), \ldots, (\mathbf{M}_q, \mathbf{T}_q)) \mid \exists \mathbf{m} = (m_1, \ldots, m_q) \in \mathbb{Z}_p^q \text{ s.t. } \forall i \in [q], \\ msg_i = \left(M_{i0} = P^{m_i},\ M_{i1} = B_i^{m_i} = P^{\alpha^i m_i},\ T_{ij} = B_j^{m_i} = P^{\alpha^j m_i}\right) \\ \text{where } j \in \left([i+1, i+q] \setminus \{q+1\}\right) \cup \{2q\}. \end{array} \right\} \tag{1}$$

Also, pp does not include $B_{q+1} = P^{\alpha^{q+1}}$, meaning that no index $j = q+1$ exist. We use bases in $\mathbb{G}_2$ meaning $\hat{\mathbf{B}} = \{\hat{B}_1, \ldots, \hat{B}_{2q}\}$ to verify whether a vector is in the message space. Specifically, given a message msg, we can check whether the pairing satisfies the relationship with respect to i and j as defined in (1):

$$e(M_{i1}, \hat{B}_{j-i}) = e(T_{ij}, \hat{P}) \wedge e(M_{i1}, \hat{P}) = e(M_{i0}, \hat{B}_i) \tag{2}$$

Example: To clarify this process, let's consider an example with $q = 3$.

$$msg_1 = ((M_{10}, M_{11}), (T_{12}, T_{13})) = (\underbrace{(P^{m_1}, P^{\alpha^1 \cdot m_1})}_{\text{main part}}, \underbrace{(B_2^{m_1}, B_3^{m_1})}_{\text{tag}}),$$
$$msg_2 = ((M_{20}, M_{21}), (T_{23}, T_{25})) = ((P^{m_2}, P^{\alpha^2 \cdot m_2}), (B_3^{m_2}, B_5^{m_2})), \text{ and}$$
$$msg_3 = ((M_{30}, M_{31}), (T_{35}, T_{36})) = ((P^{m_3}, P^{\alpha^3 \cdot m_3}), (B_5^{m_3}, B_6^{m_3})).$$

Note that j cannot be $q+1$ (which is 4 in this example). Considering that $T_{ij} = B_j^{m_i} = P^{m_i \cdot \alpha^j}$, we get:

$$msg_1 = ((P^{m_1}, P^{\alpha^1 \cdot m_1}), (P^{\alpha^2 \cdot m_1}, P^{\alpha^3 \cdot m_1})),$$
$$msg_2 = ((P^{m_2}, P^{\alpha^2 \cdot m_2}), (P^{\alpha^3 \cdot m_2}, P^{\alpha^5 \cdot m_2})), \text{ and}$$
$$msg_3 = ((P^{m_3}, P^{\alpha^3 \cdot m_3}), (P^{\alpha^5 \cdot m_3}, P^{\alpha^6 \cdot m_3})).$$

One can verify each message as follows: For msg_1, (1) corresponds to the following three equations:

$$i = 1, j = 2 : e(M_{11}, \hat{B}_{2-1}) = e(T_{12}, \hat{P}) \Rightarrow e(P^{m_1 \cdot \alpha^1}, \hat{P}^{\alpha^1}) = e(P^{m_2 \cdot \alpha^2}, \hat{P})$$
$$i = 1, j = 3 : e(M_{11}, \hat{B}_{3-1}) = e(T_{13}, \hat{P}) \Rightarrow e(P^{m_1 \cdot \alpha^1}, \hat{P}^{\alpha^2}) = e(P^{m_1 \cdot \alpha^3}, \hat{P})$$
$$e(M_{11}, \hat{P}) = e(M_{10}, \hat{B}_1) \Rightarrow e(P^{m_1 \cdot \alpha^1}, \hat{P}) = e(P^{m_1}, \hat{P}^{\alpha^1})$$

Similarly, for msg_2 we obtain:

$$i = 2, j = 3 : e(M_{21}, \hat{B}_{3-2}) = e(T_{23}, \hat{P}) \Rightarrow e(P^{m_2 \cdot \alpha^2}, \hat{P}^{\alpha^1}) = e(P^{m_2 \cdot \alpha^3}, \hat{P})$$

$$i = 2, j = 5 : e(M_{21}, \hat{B}_{5-2}) = e(T_{25}, \hat{P}) \Rightarrow e(P^{m_2 \cdot \alpha^2}, \hat{P}^{\alpha^3}) = e(P^{m_2 \cdot \alpha^5}, \hat{P})$$
$$e(M_{21}, \hat{P}) = e(M_{20}, \hat{B}_2) \Rightarrow e(P^{m_2 \cdot \alpha^2}, \hat{P}) = e(P^{m_2}, \hat{P}^{\alpha^2})$$

This process holds also the same for msg_3.

Construction. Based on the q-DHE assumption, we now present a SPVC in the standard model which ensures that the commitment can be generated even in the presence of malicious behavior. In this model, tags are used to generate the proof/witness, while the main messages are used for verification and commitment.

In the definition, we introduce an auxiliary subroutine (VerifyMsg) to verify the correctness of messages with respect to the scheme parameters. This is only to ease presentation and avoid duplication in the presentation.

Scheme 2 (q-DHE SPVC) *Our q-DHE* SPVC *scheme is defined as follows:*

$\mathsf{KeyGen}(1^\lambda, q)$: Given the security parameter λ and the size q of the committed vector, the key generation picks $\alpha \leftarrow\$ \mathbb{Z}_p$ and outputs some public parameters which serves as commitment keys:

$$\mathsf{pp} = \begin{pmatrix} \mathsf{BG}, B_1 = P^{\alpha^1}, \ldots, B_q = P^{\alpha^q}, B_{q+2} = P^{\alpha^{q+2}}, \ldots, B_{2q} = P^{\alpha^{2q}}; \\ \hat{B}_1 = \hat{P}^{\alpha^1}, \ldots, \hat{B}_{2q} = \hat{P}^{\alpha^{2q}}; g_t^{\alpha^{q+1}} \end{pmatrix}$$

$\mathsf{Commit}(msg_1, \ldots, msg_q)$: On input a vector of q messages $(msg_1, \ldots, msg_q) \in \mathcal{M}^{\mathsf{pp},q}$, such that each $msg_i = (\mathbf{M}_i = (M_{i0}, M_{i1}), \mathbf{T}_i)$, check if messages are correctly generated via $1 \leftarrow \mathsf{VerifyMsg}(\mathsf{pp}, msg_i)$ for all $i \in [q]$. Pick $r \leftarrow\$ \mathbb{Z}$ and output a commitment com and auxiliary information aux:

$$\mathsf{com} = \left(C = P^r \cdot \prod_{i \in [q]} M_{i1} \right) \wedge \mathsf{aux} = (r, \mathbf{T})$$

$\mathsf{Open}(msg, i, \mathsf{aux})$: This algorithm is run by the committer to produce a proof π_i that msg is the i-th committed message:

$$\pi_i = \left(\pi_1 = B_{q+1-i}^r \cdot \prod_{j \in [q] \setminus \{i\}} T_{j,q+1-i+j}, \quad T_{i,2q} \right).$$

Where $T_{i,2q} = B_{2q}^{m_i}$ is an element of the tag.

$\mathsf{Verify}(\mathsf{com}, M_{0i}, i, \pi_i)$: The verification algorithm accepts (i.e., it outputs 1) if and only if π_i is a valid proof that com was created for M_{i0} :

$$e\left(C, \hat{B}_{q+1-i}\right) = e\left(\pi_1, \hat{P}\right) \cdot e\left(M_{i0}, \hat{B}_{q+1}\right) \wedge e(M_{i0}, \hat{B}_{2q}) = e(T_{i,2q}, \hat{P}).$$

$\mathsf{VerifyMsg}(\mathsf{pp}, msg)$: Takes a message $msg_i = (\mathbf{M} = (M_{i0}, M_{i1}), \mathbf{T} = (T_{ij}))$ and verifies whether it is well-formed with respect to the pp. It returns 1 if the following equation holds, and 0 otherwise.

$$e\left(M_{i1}, \hat{B}_{j-i}\right) = e\left(T_{ij}, \hat{P}\right) \wedge e\left(M_{i1}, \hat{P}\right) = e\left(M_{i0}, \hat{B}_i\right)$$

where $j \in [i+1, i+q] \setminus \{q+1\}$.

Our commitment scheme also allows for updates algorithm. Moreover, since all elements of the scheme are group elements, we can consistently randomize commitments and messages. To emphasize this randomization property, we define a new algorithm, Rand, specifically for this purpose.

$\mathsf{Update}(\mathsf{com}, msg, msg', i)$: This algorithm is run by the committer who produced com and wants to update it by changing the i-th message to $msg_i' = (\mathbf{M}', \mathbf{T}')$. Check if the new message is created correctly via VerifyMsg then outputs a new commitment com′ as: $C' = (C / M_{i1}) \cdot M_{i1}'$.

$\mathsf{Rand}(\mathsf{com}, \mathsf{aux}, \pi_i, msg_i) \to (\mathsf{com}', \mathsf{aux}', \pi_i', msg_i')$: Randomize the commitment and proof for a randomized message msg_i' as: Pick a random $\mu \leftarrow\!\!\$\, \mathbb{Z}_p$ and compute:

$$\mathsf{com}' = C' = C^{\mu} \wedge \pi_i' = \pi_i^{\mu},$$

which is valid for the randomized message $msg_i' = (\mathbf{M}^{\mu}, \mathbf{T}^{\mu})$. Update $\mathsf{aux} = (r, \mathbf{T})$ as $\mathsf{aux}' = (\mu \cdot r, \mathbf{T}^{\mu})$ and output $(\mathsf{com}', \mathsf{aux}', \pi_i', msg_i')$.

Correctness of Scheme 2 is straightforward, for completeness we provide it in Appendix C.

Theorem 2. *Scheme 2 is binding in the Algebraic Group Model (AGM) based on the q-DL assumption.*

Proof. To show binding, we proceed as follows. Let $\mathcal{A}$ be an algebraic adversary that breaks Scheme 2, meaning it produces a commitment C, an index $i \in [q]$, and two valid openings $(\pi_1, M_{i0}, T_{i,2q})$ and $(\pi_1', M_{i0}', T_{i,2q}')$ with $M_{i0} \neq M_{i0}'$, both satisfying

$$e\big(C, \hat{B}_{q+1-i}\big) = e\big(\pi_1, \hat{P}\big) \cdot e\big(M_{i0}, \hat{B}_{q+1}\big), \tag{3}$$

$$e\big(M_{i0}, \hat{B}_{2q}\big) = e\big(T_{i,2q}, \hat{P}\big). \tag{4}$$

Algebraic representations. Since $\mathcal{A}$ is algebraic, every group element it outputs comes with an explicit linear representation in terms of the public parameters. We therefore associate with the commitment C, the tag element $T_{i,2q}$, the openings π_1, π_1', and the message elements M_{i0}, M_{i0}' the formal polynomials $C(X)$, $T(X)$, $R(X)$, $R'(X)$, $M(X)$, and $M'(X)$, respectively. We write these as formal polynomials evaluated at α:

$$C(X) = \sum_{j \in [2q]\setminus\{q+1\}} a_j \cdot X^j + a_0, \quad T(X) = \sum_{j \in [2q]\setminus\{q+1\}} t_j \cdot X^j + t_0,$$

$$R(X) = \sum_{j \in [2q]\setminus\{q+1\}} r_j \cdot X^j + r_0, \quad R'(X) = \sum_{j \in [2q]\setminus\{q+1\}} r_j' \cdot X^j + r_0',$$

$$M(X) = \sum_{j \in [2q]\setminus\{q+1\}} y_j \cdot X^j + y_0, \quad M'(X) = \sum_{j \in [2q]\setminus\{q+1\}} y_j' \cdot X^j + y_0'.$$

Step 1: $M(X)$ and $M'(X)$ are constants (degree-0 polynomials). From the second verification equation (4), as a polynomial identity at $X = \alpha$, we have:

$$M(\alpha) \cdot \alpha^{2q} \;=\; T(\alpha).$$

Define the auxiliary polynomial

$$Q(X) \;=\; M(X) \cdot X^{2q} \;-\; T(X).$$

Then $Q(\alpha) = 0$. We consider two sub-cases.

Sub-case A: $Q(X) \neq 0$. Then α is a root of the known nonzero polynomial $Q(X)$ of degree at most $4q$. A reduction $\mathcal{B}$ factors $Q(X)$ over $\mathbb{Z}_p$, obtaining at most $4q$ candidate roots $\beta_1, \ldots, \beta_t$ (with $t \leq 4q$). For each candidate β_k, $\mathcal{B}$ checks whether $P^{\beta_k} \stackrel{?}{=} B_1 = P^{\alpha}$. so $\mathcal{B}$ outputs α, solving the q-DL instance except with negligible probability under.

Sub-case B: $Q(X) \equiv 0$. Then $M(X) \cdot X^{2q} = T(X)$ holds as a polynomial identity, and coefficient comparison is now valid. Expanding the left-hand side:

$$y_0 \cdot X^{2q} \;+\; \sum_{j \geq 1} y_j \cdot X^{j+2q} \;=\; T(X).$$

The term $y_0 \cdot X^{2q}$ has degree $2q$, which lies in the support of $T(X)$; it matches t_{2q}, giving the scalar relation $y_0 = t_{2q}$. Every term $y_j \cdot X^{j+2q}$ with $j \geq 1$ has degree $j + 2q > 2q$, exceeding the maximal degree of $T(X)$. No such term can be matched on the right-hand side, so $y_j = 0$ for all $j \geq 1$. Hence $M(X) = y_0$ is a constant polynomial.

Note carefully: the factor X^{2q} arises from *multiplying* $M(X)$ by X^{2q} in the identity; it is not a degree-$2q$ term inside $M(X)$ itself. The identical argument applied to $M'(X)$ gives $M'(X) = y'_0$ for some scalar $y'_0 \in \mathbb{Z}_p$. As if $Q(X) \neq 0$, the adversary already breaks q-DL; otherwise, except with negligible probability under q-DL, we assume $Q(X) \neq 0$ and conclude M and M' are constant, and we can proceed with the proof.

Step 2: Polynomial identity for the main verification equation. Rewrite Eq. (3) as a polynomial identity:

$$C(X) \cdot X^{q+1-i} \;=\; R(X) + y_0 \cdot X^{q+1}.$$

Since both openings verify at $X = \alpha$, define

$$P(X) = C(X) \cdot X^{q+1-i} - y_0 \cdot X^{q+1} - R(X),$$
$$P'(X) = C(X) \cdot X^{q+1-i} - y'_0 \cdot X^{q+1} - R'(X),$$

so $P(\alpha) = 0$ and $P'(\alpha) = 0$.

Step 3: Deriving a contradiction or breaking q-DL. Form the difference polynomial

$$D(X) \;=\; P(X) - P'(X) \;=\; (y'_0 - y_0) \cdot X^{q+1} - \sum_{j \in [2q] \setminus \{q+1\}} (r_j - r'_j) X^j - (r_0 - r'_0). \tag{5}$$

Since $P(\alpha) = P'(\alpha) = 0$, we have $D(\alpha) = 0$. We note that $D(X)$ follows directly from equating right-hand sides of both verification equations:

$$e\left(\pi / \pi',\ \hat{P}\right) = e\left(M'_{i0} / M_{i0},\ \hat{B}_{q+1}\right),$$

where $C(X) \cdot X^{q+1-i}$ cancels due to the shared commitment. We now consider the following cases.

Case 1: $D(X) = 0$**.** All coefficients of $D(X)$ vanish. In particular, the coefficient of X^{q+1} gives $y'_0 - y_0 = 0$, i.e. $y_0 = y'_0$. Hence $M_{i0} = P^{y_0} = P^{y'_0} = M'_{i0}$, contradicting $M_{i0} \neq M'_{i0}$.

Case 2: $D(X) \neq 0$**.** Since $D(\alpha) = 0$ and $D(X) \neq 0$, the α is a root of the known nonzero polynomial $D(X)$ of degree $\leq 2q$. $\mathcal{B}$ factors $D(X)$ over $\mathbb{Z}_p$, obtaining candidate roots $\beta_1, \ldots, \beta_t$. For each candidate β_j, $\mathcal{B}$ checks whether: $P^{\beta_j} = B_1$, where $B_1 = P^{\alpha}$ is the first public parameter. The unique β_j satisfying this is α. $\mathcal{B}$ outputs the scalar α, breaking the q-DL assumption. □

Remark 3. We also present an alternative strategy for proving binding in the GGM in [63]. This second approach proceeds via an explicit extraction argument, formalized as a Lemma, showing that any valid binding forgery reveals $P^{\alpha^{q+1}}$ and thus breaks the q-DHE assumption. The lemma itself is generic and modular—it isolates the key algebraic property of valid messages in our commitment scheme and can be reused in other constructions or proofs involving similarly structured commitments. We note that in this case, one needs to check the correctness of message structure VerifyMsg as part of the Verify proof.

5 Structure-Preserving Accumulator

In this section, we introduce the concept of a structure-preserving accumulator (SPA) and demonstrate how our vector commitment can be adapted into an accumulator, resulting in the first structure-preserving accumulator of this kind.

SPA **from (signature-based) Weakly Binding VC.** A weakly binding vector commitment can naturally be expressed as an accumulator, as both schemes are essentially equivalent when the accumulator and the vector commitment are honestly generated. Consequently, the witness becomes a signature for the element M_i, which is bound to specific public parameters

5.1 Blackbox Accumulators from SPVC

SPA **from q-DHE VC**: Catalano and Fiore [29] proposed a black-box construction of accumulators based on vector commitments. Their approach involves creating a succinct commitment C to a vector $\mathbf{X} = (x_1, \ldots, x_q)$ through a vector commitment. The commitment C ensures that it is computationally infeasible to open any position i to a value x_i' different from the original x_i. In their construction, the accumulation domain is represented by the set $D = \{1, \ldots, q\}$, and the accumulator is modeled as a commitment to a binary vector of length t. Each bit i indicates whether the element $i \in D$ is included in the accumulator. Membership or non-membership of an element is verified by revealing the corresponding position i of the commitment as either 1 or 0. However, it is not clear how to apply this result to the SP setting as our SPVC is not suitable for committing to messages that are mapped to integers i (i.e., message index).

Compiler From Vector Commitments to Accumulators. We rather show in the following how any SPVC can be used to construct a SPA. Actually, our generic construction achieves a somewhat different security notion that we call *bounded soundness*, where the accumulator is fully controlled by the adversary and the adversary is considered to win if it is able to generate an accumulator and openings to more (i.e., at least $q+1$) distinct values than the upper bound q of the accumulator allows for. We actually introduce a parametrized version, to also cover the accumulator introduced in Section 5.2 and comment on this parametrization below.

Bounded Soundness. We introduce the notion of bounded soundness, which naturally fits our setting and can be easily used when working with vector-commitment-based constructions. Intuitively, it is similar to collision resistance, but instead of preventing any two distinct openings to the same accumulator, it bounds the number of valid distinct openings that can exist. Specifically, the adversary should not be able to produce more than q distinct valid openings, where q is the upper bound defined by the accumulator parameter.

Definition 20 (Bounded soundness). *An accumulator scheme is said to satisfy parametrized bounded soundness if for all PPT adversaries $\mathcal{A}$, the following probability is negligible:*

$$\Pr\left[\begin{array}{l}(\mathsf{sk}_{acc}, \mathsf{pk}_{acc}) \leftarrow \mathsf{Setup}(1^\lambda, q), (\{(x_k, wit_{x_k})\}_{k=1}^{q+1}, \mathsf{Acc}) \leftarrow \mathcal{A}(\mathsf{pk}_{acc}, \mathsf{sk}_{acc}) : \\ \forall k \in [q+1] : \mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}, wit_{x_k}, x_k) = 1 \wedge \forall m \neq n \in [q+1] : x_m \neq x_n\end{array}\right]$$

Remark 4. We note that in the context of vector commitments (VC), the commitments do not need to be hiding, which makes the inclusion of randomness r seem unnecessary. Moreover, the randomness r is not required for the binding proof. Nevertheless, we retain it here as it might be useful for other applications in the future or for achieving properties like hiding, which are not directly required in our current setting.

Reducing Trust in pp. The pairing-based construction typically requires either public parameters generated in a trusted setup, which are linear in the number of elements added to the Acc, or a trusted party with a trapdoor to compute it. Trust in parameter generation can be reduced or removed using MPC protocols, such as [19]. Alternatively, Groth et al. [58] proposed updatable reference strings, which allow any party to update them securely, as demonstrated in Ethereum's 'powers of tau' ceremony [81].

Generic Construction: The construction is quite straightforward (see more details in [63]) as follow:

- $\mathsf{Setup}(1^\lambda, q)$ first samples pp as the public parameters of the SPVC. Furthermore, it samples pairwise distinct group elements $(U_1, \ldots, U_q)$.[6] The algorithm sets $\mathsf{pk}_{acc} \leftarrow (\mathsf{pp}, (U_1, \ldots, U_q))$, as well as sk_{acc} is set to $\perp$.
- To compute the accumulator, Eval first pads $\mathcal{X}$ with random elements in the case that $|\mathcal{X}| \leq q$. It then defines a mapping of the elements to the index set $\{1, \ldots, q\}$, which is stored as aux. The actual accumulator value Acc is then obtained by simply running the Commit algorithm.
- To create a witness for a value x, WitCreate first uses aux to re-identify the position i_x corresponding to x, and then runs the Open algorithm on the corresponding inputs. It outputs $wit_x = (U_{i_x}, \pi_x)$.
- The Verify now works canonically by outputting the result of the SPVC's Verify algorithm.

[6] E.g., one can set $U_i = P^i$, it is only used for binding messages to specific public elements.

The bounded soundness of this construction can now be seen by a simple observation. If the adversary can output $q+1$ values and valid witnesses (corresponding to messages and openings of a commitment) for a fixed accumulator value Acc (corresponding to a commitment value), at least two witnesses $wit_x, wit_{x'}$ for $x \neq x'$ must contain the same index $i_x = i_{x'}$. Therefore, the corresponding $\pi_x, \pi_{x'}$ are valid openings for the same position, immediately breaking the position binding property of the SPVC.

5.2 (Perfect Randomizable) Accumulator based on the q-DHE VC

The above construction reveals the position of an element in the underlying vector commitment, which unfortunately turns out to be insufficient for some privacy-enhancing primitives such as ring signatures. In this case, revealing the index of public key which was used for signing would immediately violate anonymity. Therefore, we are interested in an approach where the witness does not reveal sensitive information (beyond the message), not even the index, which however turns out to be non-trivial.

In the following, we now introduce such a (perfectly randomizable) q-DHE Structure-Preserving Accumulator (q-DHE SPA) in which the witness does not disclose any information beyond the message itself – not even its index. By making minor adjustments to the vector commitment scheme outlined in Scheme 2, we achieve that the message is no longer tied to any specific position or element in pp.

To achieve this, we randomize the elements that reveal the positions. Specifically, the verification equation $e(C, \hat{B}_{q+1-i}) = e(\pi_i, \hat{P}) \cdot e(M_{i0}, \hat{B}_{q+1})$ is initially designed to confirm the positions of the messages with respect to $\hat{B}_{q+1-i}$. We randomize this verification by picking a new random $\rho \in \mathbb{Z}_p$, modifying the equation to $e(C, \hat{W}_2) = e(\pi_i, \hat{P}) \cdot e(M_{i0}, \hat{B}_{q+1})$, where $\hat{W}_2 = (\hat{B}_{q+1-i})^\rho$ is part of the witness. We apply the same technique to the second verification equation involving $T_{i,2q}$ as $W_3 = T_{i,2q}^\rho$. By using the bilinear pairing property, we can appropriately randomize π_i with ρ, thereby ensuring that the verification remains valid as intended.

We now handle the randomization of witnesses (or proofs in SPVC) and commitments by defining an additional algorithm Rand, which generates tags for messages in the set and randomizes the witness/accumulators, respectively. We slightly adjust standard definitions to reflect these properties.

Definition 21 (Extended Static Accumulator). *A Extended static accumulator has the following additional algorithms in addition to Static Accumulator:*

- $\mathsf{VerifyMsg}(\mathsf{pp}, msg)$: Takes a message and tag $msg = (\mathbf{M}, \mathbf{T})$, verifies whether it is well-formed with respect to the pp.
- $\mathsf{Rand}(\mathsf{Acc}_\mathcal{X}, \mathsf{aux}, wit_{msg_i}, msg_i, (\mu, \gamma))$: On input an accumulator $\mathsf{Acc}_\mathcal{X}$, witness wit_{msg_i}, message msg_i and randomness $(\mu, \gamma) \in \mathbb{Z}_p$, compute a randomized accumulator, auxiliary information and witness $(\mathsf{Acc}'_\mathcal{X}, \mathsf{aux}', wit'_{msg_i},)$ for a randomized message msg'_i and output $(\mathsf{Acc}'_\mathcal{X}, \mathsf{aux}', wit'_{msg_i}, msg'_i)$.

We present the complete construction as follows:

Scheme 3 (q-DHE Accumulator) *Let* SPVC *be the vector commitment in Scheme 2, our q-DHE randomizable* SPA *is defined as follows:*

- $\mathsf{Setup}(1^\lambda, q)$: Given a security parameter λ and a parameter q, run $\mathsf{pp} \leftarrow \mathsf{SPVC.KeyGen}(1^\lambda, q)$ and set $\mathsf{pk}_{acc} = \mathsf{pp}$ and $\mathsf{sk}_{acc} = \perp$.
- $\mathsf{Eval}(\mathsf{pk}_{acc}, \mathcal{X})$: Given a public key pk_{acc} and set $\mathcal{X}$ it returns an accumulator $\mathsf{Acc}_\mathcal{X}$ together with auxiliary information aux as follows:
  - Check for all if $1 \leftarrow \mathsf{VerifyMsg}(\mathsf{pp}, msg_i)$ for all $msg_i = (\mathbf{M} = (M_{i0}, M_{i1}), \mathbf{T}) \in \mathcal{X}$.
  - For a random $r \in \mathbb{Z}_p$, compute:

$$\mathsf{Acc}_\mathcal{X} = C = \left(P^r \cdot \prod_{i \in [q]} M_{i1} \right) \ \wedge \ \mathsf{aux} = (r, \mathbf{T}).$$

Note that $\mathsf{Acc}_\mathcal{X}$ is similar to run $\mathsf{SPVC.Commit}(msg_1, \ldots, msg_q)$, where $msg_i = (\mathbf{M}, \mathbf{T})$.

$\mathsf{WitCreate}(\mathsf{pk}_{acc}, \mathsf{Acc}_{\mathcal{X}}, \mathsf{aux}, msg_i, \rho)$: This algorithm takes a key pair pk_{acc}, an accumulator $\mathsf{Acc}_{\mathcal{X}}$, auxiliary information aux, and a value msg_i. It returns $\perp$, if $msg_i \notin \mathcal{X}$ or $0 \leftarrow \mathsf{VerifyMsg}(\mathsf{pp}, msg_i)$, otherwise, compute a witness wit_{msg_i} for msg_i: For $\rho \in \mathbb{Z}_p^*$, and compute wit_{msg_i}

$$\Big(W_1 = \big(B_{q+1-i}^{r} \cdot \prod_{j \in [q] \setminus \{i\}} T_{j,q+1-i+j}\big)^{\rho},$$
$$\hat{W}_2 = \hat{B}_{q+1-i}^{\rho} = (\hat{P}^{\alpha^{q+1-i}})^{\rho}, \quad W_3 = T_{i,2q}^{\rho}, \quad M_{i0}^{\rho}\Big).$$

where the $M_{i0}^{\rho} = M'_{i0}$ is a new randomized message.

$\mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}_{\mathcal{X}}, wit_{msg_i}, M'_{i0})$: This algorithm takes a public key pk_{acc}, an accumulator $\mathsf{Acc}_{\mathcal{X}}$, a witness wit_{msg_i}, and a value M_{i0}. It returns *true* (i.e., it outputs 1) check if wit_{msg_i} is a valid witness and *false* (i.e., it outputs 0) otherwise as:

$$e(\mathsf{Acc}_{\mathcal{X}} = C, \hat{W}_2) = e(W_1, \hat{P}) \cdot e(M_{i0}^{\rho}, \hat{B}_{q+1}) \wedge e(M_{i0}^{\rho}, \hat{B}_{2q}) = e(W_3 = T_{i,2q}^{\rho}, \hat{P}).$$

$\mathsf{Rand}(\mathsf{Acc}_{\mathcal{X}}, \mathsf{aux}, wit_{msg_i}, M_{i0}, (\mu, \gamma)) \rightarrow (\mathsf{Acc}'_{\mathcal{X}}, \mathsf{aux}', wit'_{msg_i}, M''_{i0})$: On input an accumulator $\mathsf{Acc}_{\mathcal{X}}$, auxiliary information aux, witness wit_{msg_i}, message msg_i and randomness $(\mu, \gamma) \in \mathbb{Z}_p$, compute a randomized accumulator and witness for a randomized message msg'_i as:

$$\mathsf{Acc}'_{\mathcal{X}} = \mathsf{Acc}_{\mathcal{X}}^{\mu} \wedge wit'_{msg_i} = (W_1^{\mu\gamma}, \hat{W}_2^{\gamma}, W_3^{\mu\gamma}) \wedge M''_{i0} = M_{i0}^{\mu\gamma\rho}.$$

This is valid accumulator-witness pair for the randomized message. Finally, $\mathsf{aux} = (r, \mathbf{T})$ is updated.

The verification procedure does not require knowledge of the message position (index) or any public values that bind messages to specific positions. Moreover, the index cannot be derived from the CRS. Since only M_{i0} is revealed, and this value is re-randomized, consistency checks from other tags are not possible. Equivalently, only a single re-randomized element of the extended key is exposed, which prevents linking it to a specific position.

Bounded Domain and Padding. The accumulator is parameterized by a global bound $q = \mathsf{poly}(\lambda)$. All accumulated sets are interpreted as subsets of a universe indexed by $[q]$. For a set of size $\ell \leq q$, we represent it as a length-q vector, where positions corresponding to elements in the set are active and all remaining positions are filled with dummy values (or marked inactive). Equivalently, the accumulator commits to a vector of fixed length q with unused positions padded with a default element.

Security: The following presents our main theorem on the security of the accumulator, which we have tailored to the concrete construction after personal communication with the researchers involved in [88]. Intuitively, it can be seen as a variant of existing collision-resistance notions. While we allow the adversary to compute the accumulator, we additionally require it to reveal the randomness r used during the computation, this requirement lies between the standard and the strong forms of collision resistance. Moreover, we also require the adversary to output auxiliary information ρ, whose validity is verified and which pertains to the witnesses involved in the collision. This additional requirement slightly weakens the guarantees compared to standard definitions. Nevertheless, this modeling proves sufficiently strong for our practical applications, as demonstrated later in the ring signature section. In practice, such additional outputs can be enforced by higher-level protocols through zero-knowledge proofs allowing the reduction to extract them.

Theorem 3. *If the q-DHE Vector Commitment in Scheme 2 is position-binding, then the q-DHE Accumulator in Scheme 3 satisfies for all PPT adversaries $\mathcal{A}$ the following probability is negligible for every $\mathcal{X} = \{(\mathbf{M}_i, \mathbf{T}_i)\}_{i=1}^{q} \in \mathcal{M}^{\mathsf{pp},q}$:*

$$\Pr\left[\begin{array}{l}(\mathsf{sk}_{acc}, \mathsf{pk}_{acc}) \leftarrow \mathsf{Setup}(1^{\lambda}, q), ((r, \rho), (\mathbf{M}', wit'), \mathsf{Acc}) \leftarrow \mathcal{A}(\mathsf{pk}_{acc}, \mathsf{sk}_{acc}, \mathcal{X}) : \\ \mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}, wit', M'_{i0}) = 1 \wedge \forall i \in [q] : M'_0 \neq M_{i0} \wedge \\ \mathsf{Acc} = P^r \cdot \prod_{i \in [q]} M_{i1} \quad \textit{and} \quad \exists n' \in [q] : \hat{W}'_2 = \hat{B}^{\rho}_{n'} \wedge W'_3 = T^{\rho}_{i,2q}\end{array}\right],$$

where $wit' = (W'_1, \hat{W}'_2, W'_3)$.

Proof. Let $\mathcal{A}$ be an adversary succeeding with non-negligible probability in the experiment of the theorem. We construct an equally efficient adversary $\mathcal{B}$ against the position-binding property of the underlying vector commitment. The adversary $\mathcal{B}$ receives the public parameters pp of the vector commitment scheme and forwards them to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs a tuple $(\mathsf{aux} = (r, \rho), (M'_{0i}, wit\ acc)$ such that all verification checks in the theorem are satisfied. From the condition $\hat{W}'_2 = \hat{B}^{\rho}_{n'}$, and the knowledge of ρ, the reduction computes $(\hat{W}'_2)^{1/\rho} = \hat{B}_{n'}$. Since the CRS elements $\{\hat{B}_j\}_{j\in[q]}$ are public and indexed, $\mathcal{B}$ can efficiently recover the corresponding position n'. Furthermore, from the theorem assumptions, $W'_3 = T^{\rho}_{i,2q}$. Hence $\mathcal{B}$ can de-randomize the witness components and obtain

$$\pi'_{n'} = \Big((W'_1)^{1/\rho}, (\hat{W}'_2)^{1/\rho}, (W'_3)^{1/\rho}, \Big).$$

By construction,

$$(\hat{W}'_2)^{1/\rho} = \hat{B}_{n'}, \qquad (W'_3)^{1/\rho} = T_{i,2q}, \qquad (M')^{1/\rho} = M_{0i},$$

which is exactly the structure of a valid vector-commitment opening proof at position n' for the commitment $C = \mathsf{Acc}$. On the other hand, since the tag vector $\mathbf{T}_i$ is publicly available, $\mathcal{B}$ can compute the honest witness $\pi_{n'} = wit_{n'}$ for the message $\mathbf{M}_{n'}$ at the same position n'. By assumption, $M' \neq M_{n'}$ while both $\pi'_{n'}$ and $\pi_{n'}$ verify with respect to the same commitment $C = \mathsf{Acc}$ and the same position n'. Therefore $\mathcal{B}$ obtains two valid openings of the same commitment at the same position to two distinct messages, violating the position-binding property of the underlying vector commitment.

Hence, any adversary succeeding in the above experiment can be transformed into an adversary breaking position-binding. Since the vector commitment is position-binding, the success probability of $\mathcal{A}$ must be negligible. □

Perfect Randomization. We define a privacy property here inspired by signature adaptation in SPS-EQ signatures [49]. We want to guarantee that fresh and randomized accumulators, messages and witnesses are indistinguishable, which is important to guarantee the unlinkability of witness presentation. As fresh and randomized instances look identical and do not leak any information, the messages are not associated with particular values or public parameters, thus making it infeasible to identify their location in the set.

Remark 5. The auxiliary information aux is not required to be hidden for the perfect-randomization property. It is used only for witness generation and may therefore be treated as public auxiliary input. Accordingly, we include aux in the distinguisher's view.

Definition 22 (Perfect Randomization of Acc). *An accumulator scheme provides perfect randomization if for all λ, and for all $\mathsf{pk}_{acc} \leftarrow \mathsf{Setup}(1^{\lambda}, q)$, and for honestly generated $(\mathsf{pk}_{acc}, msg_i, \mathsf{Acc}_{\mathcal{X}}, \mathsf{aux}, wit_{msg_i})$, the following holds: If $\mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}_{\mathcal{X}}, wit_{msg_i}, M_{i0}) = 1$, then $(\mathsf{Acc}'_{\mathcal{X}}, wit'_{msg_i}, \mathsf{aux}', M'_{i0}) \leftarrow \mathsf{Rand}(\mathsf{Acc}, wit_{msg_i}, \mathsf{aux}, M_{i0})$ satisfies that $\mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}'_{\mathcal{X}}, wit'_{msg_i}, M'_{i0}) = 1$ and the distributions satisfy $(\mathsf{Acc}'_{\mathcal{X}}, wit'_{msg_i}, \mathsf{aux}, M'_{i0}) \approx (\mathsf{Acc}_{\mathcal{X}}, wit_{msg_i}, \mathsf{aux}', M'_{i0})$.*

Theorem 4. *The q-DHE Accumulator in Scheme 3 is perfectly randomizable.*

The proof is straightforward and provided in Appendix D.

6 Applications

Compressing primitives such as accumulators and vector commitments are highly versatile tools and can be used in many applications, as mentioned in the introduction. However, in this section, we describe how our SPVC and SPA primitives can lead to new and interesting applications, in addition to common ones. As the main application, we present the first constant-size ring signature in the dlog setting. The size of the signature consists of only a few group elements, and the computation involves only group exponentiation operations along with an efficient Schnorr NIZK for a simple AND statement. We also informally discuss some other potential applications for blockchains and demonstrate how these can result in significant efficiency improvements in blockchain applications.

6.1 Constant-Size Ring Signatures

We take inspiration from the framework in [40], which employs a key-homomorphic signature and a NIWI/NIZK for an OR relation (giving linear sized signatures). Moreover we consider the approach introduced in [41] that employs an accumulator for the membership proof in the ring. In particular, [41] rely on an RSA accumulator and so far no construction for ring signatures in the discrete logarithm setting following this approach, due to incompatibility of public key and accumulation domains, is known. Basically, the idea is to use an accumulator to compactly represent all public keys in the ring and to use a NIZK proof to demonstrate knowledge of a membership witness in this accumulator as well as the corresponding secret key, e.g., via an explicit signature.

We can achieve a constant-size ring signature in the discrete logarithm setting by integrating these two approaches i.e., using an accumulator with a key-homomorphic signature and leveraging very efficient (Schnorr style) NIZK proofs.

Construction Overview. We first define an accumulator-compatible variant of BLS, allowing tagged public keys to be accumulated using the q-DHE accumulator of Definition 3. To sign with respect to a ring $\mathcal{R}$, the signer computes $\mathsf{Acc}_{\mathcal{R}}$, a membership witness for its public key, and a $\mathsf{BLS}_{\mathsf{acc}}$ signature on $m\|\mathcal{R}$. A Schnorr-type proof then shows, without revealing the signer, that the same secret key underlies a valid BLS signature and a valid accumulator opening. Applying Fiat–Shamir yields a constant-size ring signature. Rings of size $\ell < q$ are internally padded according to the mechanism of Acc.

We begin with an accumulator-based BLS signature. We assume that each key has a specific position within the given ring.

Scheme 4 (Accumulator-Compatible BLS Scheme $\mathsf{BLS}_{\mathsf{acc}}$) $\mathsf{BLS}_{\mathsf{acc}}$ *with respect to q-DHE* SPA *accumulator* Acc *is defined by the following algorithms:*

$\mathsf{Setup}(1^\lambda, q)$: Run $\mathsf{pk}_{acc} \leftarrow \mathsf{Acc.Setup}(1^\lambda, q)$, choose a hash function $H : \mathcal{M} \to \mathrm{G}_2$, output $\mathsf{pp} := (H, \mathsf{pk}_{acc})$.

$\mathsf{KeyGen}(\mathsf{pp})$: Parse pp, user i chooses $x_i \leftarrow\!\!\$\ \mathbb{Z}_p$, set $(\mathbf{M} = (M_{i0} = P^{x_i}, M_{i1} = B_i^{x_i}), \mathbf{T} = (B_j^{x_i}))$, where $|j| = q-1 \wedge j \in [i+1, 2q] \wedge j \neq q+1$ output the secret key $\mathsf{sk}_i = (x_i)$ and the tagged public-key $t\mathsf{pk}_i = (\mathsf{pk}_i = \mathbf{M}_i, \tau = (\mathbf{T}_i))$ such that $1 \leftarrow \mathsf{VerifyMsg}(\mathsf{pp}, t\mathsf{pk})$. Here, we assume pk represents pure public keys, while τ denotes the associated tag. For simplicity, we treat both together as a tagged public key.

$\mathsf{Sign}(\mathsf{sk}, m)$: Compute and return signature as $\sigma \leftarrow H(m)^{\mathsf{sk}}$.

$\mathsf{Verify}(\mathsf{pk}_i, m, \sigma_i)$: Parse pk_i as $(M_{i0} = P^{x_i}, M_{i1} = B_i^{x_i})$. Return 1 if the following holds and 0 otherwise: $e(P^x, H(m)) = e(P, \sigma)$. We only need the first component of pk to verify the signatures.

Theorem 5 (EUF-CMA Security of $\mathsf{BLS}_{\mathsf{acc}}$). *If the BLS signature scheme is EUF-CMA secure, then the accumulator-based variant $\mathsf{BLS}_{\mathsf{acc}}$ is also EUF-CMA secure.*

Proof (Sketch). To prove the EUF-CMA security of $\mathsf{BLS}_{\mathsf{acc}}$, we obtain the public key pk from the BLS challenger. Using the trapdoor α that we can freely choose, we generate the corresponding tag and provide it to the adversary. For each of the adversary's queries, we forward them to the BLS challenger. Thus, any forgery in the $\mathsf{BLS}_{\mathsf{acc}}$ scheme corresponds directly to a forgery in the BLS scheme.

Now we are ready to present our ring signature:

Our Construction. In Figure 1, we present our constant-size ring signature construction based on the accumulator-compatible signature scheme $\mathsf{BLS}_{\mathsf{acc}}$, our q-DHE structure-preserving accumulator Acc, and a non-interactive zero-knowledge proof of knowledge. The latter is instantiated using standard Schnorr-type techniques and made non-interactive via the Fiat–Shamir transform [42,75]. The basic idea of the construction is as follows. Each user i generates an accumulator-compatible tagged public key $t\mathsf{pk}_i = (\mathsf{pk}_i, \tau_i)$ using $\mathsf{BLS}_{\mathsf{acc}}.\mathsf{KeyGen}$. The tag τ_i contains the auxiliary information required by the underlying q-DHE accumulator, while pk_i denotes the public-key component that is included in a ring. To sign a message m with respect to a set of tagged public keys $\mathcal{X}$, the signer first extracts the corresponding public keys and forms the ring $\mathcal{R} = \{\mathsf{pk}_j \mid (\mathsf{pk}_j, \tau_j) \in \mathcal{X}\}$. The signer then computes the accumulator $\mathsf{Acc}_{\mathcal{R}}$ of the ring and the auxiliary information required for witness generation. Using its tagged public key $t\mathsf{pk}_i$, it computes a membership witness wit_{pk_i} certifying

that its public key belongs to the accumulated ring. In parallel, the signer computes an accumulator-compatible BLS signature $\delta \leftarrow \mathsf{BLS}_{\mathsf{acc}}.\mathsf{Sign}(\mathsf{sk}_i, m\|\mathcal{R})$ on the message together with the ring. Rather than revealing the signer's public key, its accumulator witness, or the BLS signature directly, the signer proves knowledge of values $(\mathsf{sk}_i, W_1', \hat{W}_2', W_3', \delta)$ satisfying the relations of accumulator and signature verification. The first equation proves consistency between the hidden secret key and a valid $\mathsf{BLS}_{\mathsf{acc}}$ signature on $m\|\mathcal{R}$. The second equation proves a valid hidden opening of the accumulator for the signer's public key, while simultaneously linking that opening to the same secret-key exponent. The final equation provides the additional exponent-linking relation required by the structured q-DHE message space. Consequently, the proof establishes, without revealing which member is used, that the signer knows a secret key corresponding to a valid member of the accumulated ring and that this same secret key authenticates $m\|\mathcal{R}$. The proof π can be realized using standard Schnorr-type proof techniques [75]: the prover commits to the relevant hidden exponent representations, proves the required pairing relations and equality of the shared secret-key exponent across them, and applies the Fiat–Shamir transform to obtain a non-interactive proof.

$\mathsf{Setup}(1^\lambda, q)$: Run $\mathsf{pp}_{\mathsf{bls}} \leftarrow \mathsf{BLS}_{\mathsf{acc}}.\mathsf{Setup}(1^\lambda, q)$ and then output the public parameters $\mathsf{pp} = \mathsf{pp}_{\mathsf{bls}}$.
$\mathsf{KeyGen}(\mathsf{pp}, i)$: Run $(\mathsf{sk}_i, t\mathsf{pk}_i) \leftarrow \mathsf{BLS}_{\mathsf{acc}}.\mathsf{KeyGen}(\mathsf{pp})$ and output a keypair and its tag $(\mathsf{sk}_i, t\mathsf{pk}_i)$. For simplicity we assume that τ_i is included as part of the $t\mathsf{pk}_i$ so $t\mathsf{pk}_i = (\mathsf{pk}_i, \tau_i)$.
$\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}_i, m, \mathcal{R})$: Choose randomness $(\mu, \gamma) \in \mathbb{Z}_p$ and for a set of public keys $\mathcal{X} = \{t\mathsf{pk}_i\}$ do:

- Set $\mathcal{R} \leftarrow \{\mathsf{pk}_i \mid (\mathsf{pk}_i, \tau_i) \in \mathcal{X}\}$ // extract public keys from tagged keys in $\mathcal{X}$ to form ring $\mathcal{R}$
- $(\mathsf{Acc}_{\mathcal{R}}, \mathsf{aux}) \leftarrow \mathsf{Acc.Eval}(\mathsf{pp}, \mathcal{R})$ // q-DHE SPA accumulator
- $\delta \leftarrow \mathsf{BLS}_{\mathsf{acc}}.\mathsf{Sign}(\mathsf{sk}_i, m||\mathcal{R})$ // $\mathsf{BLS}_{\mathsf{acc}}$ signature on $m||\mathcal{R}$
- $wit_{\mathsf{pk}_i} \leftarrow \mathsf{Acc.WitCreate}(\mathsf{pp}, \mathsf{Acc}_{\mathcal{R}}, \mathsf{aux}, t\mathsf{pk}_i, 1)$ // witness for pk_i
- Compute a Schnorr-type proof

$$\pi \leftarrow \mathsf{Prove}\left\{ \begin{array}{l} (\mathsf{sk}_i, W_1', \hat{W}_2', W_3', \delta) : e(P^{\mathsf{sk}_i}, H(m\|\mathcal{R})) = e(P, \delta) \\ \wedge\ e(\mathsf{Acc}_{\mathcal{R}}, \hat{W}_2') = e(W_1', \hat{P})\, e(P^{\mathsf{sk}_i}, \hat{B}_{q+1}) \wedge\ e(P^{\mathsf{sk}_i}, \hat{B}_{2q}) = e(W_3', \hat{P}) \end{array} \right\}.$$

// Homomorphic Schnorr proof for the hidden SPVC opening, together with the exponent-linking relations.

Finally, output $\sigma = (r, \pi)$.
$\mathsf{Verify}(\mathsf{pp}, m, \sigma, \mathcal{R})$: Given pp, message $m \in \mathcal{M}$, signature $\sigma = (r\pi)$, and ring $\mathcal{R}$, parse $\mathsf{pk}' = M_{i0}'$ output 1 if the following holds and 0 otherwise:

$$P^r \cdot \prod_{M_{i1} \in \mathcal{R}} M_{i1} == \mathsf{Acc}_{\mathcal{R}} \wedge \pi \text{ is correct}$$

Note that a verifier might also recompute the accumulator locally with the knowledge of r and one can then omit to send it explicitly. The NIZK proof can be instantiated using standard Schnorr-type techniques: the prover first commits to the intermediate exponent values corresponding to the products and inverse terms, and then proves their consistency with the witnesses and the public group elements via the Fiat–Shamir transform.

Fig. 1. Constant size ring signature in the dlog setting (Σ is BLS and Acc is the accumulator in Sec. 5.2)

Theorem 6. *Let* $\mathsf{BLS}_{\mathsf{Acc}}$ *be EUF-CMA secure, let NIZK be zero-knowledge and simulation extractable, and let the q-DHE* SPA *satisfy Theorem 3. Then Scheme 1 is unforgeable and anonymous.*

We provide the full proof of this theorem in Appendix E. Intuitively, unforgeability follows from the simulation extractability of the proof π, the security of the q-DHE SPA, and the EUF-CMA security of $\mathsf{BLS}_{\mathsf{Acc}}$. Given a valid forgery $(m^*, \sigma^*, \mathcal{R}^*)$, simulation extractability allows us to extract

$$(\mathsf{sk}^*, W_1^*, \hat{W}_2^*, W_3^*, \delta^*)$$

satisfying

$$e(P^{\mathsf{sk}^*}, H(m^*\|\mathcal{R}^*)) = e(P, \delta^*),$$
$$e(\mathsf{Acc}_{\mathcal{R}^*}, \hat{W}_2^*) = e(W_1^*, \hat{P})\, e(P^{\mathsf{sk}^*}, \hat{B}_{q+1}),$$

and

$$e(P^{\mathsf{sk}^*}, \hat{B}_{2q}) = e(W_3^*, \hat{P}).$$

The latter two relations constitute a valid accumulator opening for the public key determined by sk^*. If this public key were not contained in $\mathcal{R}^*$, then together with the valid openings of the accumulated ring elements it would yield a violation of the security property of the q-DHE SPA. Hence the extracted public key must correspond to one of the members of $\mathcal{R}^*$.

Conditioned on this event, the first extracted relation shows that δ^* is a valid $\mathsf{BLS}_{\mathsf{Acc}}$ signature under that public key on $m^* \| \mathcal{R}^*$. Since the forgery is fresh and the ring contains no corrupted key, guessing the corresponding ring member allows us to reduce the forgery to the EUF-CMA security of $\mathsf{BLS}_{\mathsf{Acc}}$, incurring only a polynomial loss.

Anonymity follows from the zero-knowledge property of π. All signer-dependent values, namely the secret key, the $\mathsf{BLS}_{\mathsf{Acc}}$ signature, and the accumulator membership witness, occur only as witnesses to the proof. Replacing the real proof by a simulated proof therefore removes any dependence of the signature transcript on the identity of the signer. Consequently, signatures generated by any two eligible members of the same ring are indistinguishable.

Comparison with Related Work. Looking at the recent systematization of knowledge (SoK) by Chator et al. [31], we observe several constant-size ring signatures. The work in [23], which is based on composite order groups, remains inefficient at around 100 group elements and with computations being a factor of roughly 50 times slower than in prime order groups [45].[7] The approach by Malavolta and Schröder in [74] for building ring signatures from signatures with re-randomizable keys, when instantiated with their variant of Hofheinz and Kiltz signatures secure under the q-strong Diffie-Hellman assumption and the Groth16 zk-SNARK [57], yields constant size signatures of size $4\mathbb{G}_1 + 2\mathbb{G}_2 + \mathbb{Z}_p$. A popular pairing-friendly curve for the type-3 setting with 128 bit security is the BLS12-381 curve [16], where elements in $\mathbb{Z}_p$, $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ require 32, 48, 92, and 576 bytes, respectively. For the construction in [74] this would give around 3300 bits. However, while the size is very low, using a zk-SNARK and in particular constant-sized ones such as Groth16 (or the recent Polymath [73], which is even more compact) comes with drawbacks. First they are non-universal and thus require a trusted setup per circuit (here ring size, unless one fixes an upper bound) and the concrete proving times will be significant. The signer needs to prove the verification equation of the signature scheme among $|\mathcal{R}|$ keys, involving the evaluation of a programmable hash function. In the non discrete-logarithm setting, the RSA-accumulator based ring signatures by Dodis et al. in [41] gives the most efficient solution in the factorization setting and requires around $\approx 38,500$ bits - significantly larger than in the dlog setting.

The Concrete Size of Our RS**.** The efficiency of our scheme can be evaluated in terms of communication costs and the bit size of the involved elements. The signature consists of the proof π is a Schnorr proof for four scalar witnesses satisfying certain multiplicative relations. Using standard techniques, this can be realized with a size of: $3\mathbb{G}_1 + \mathbb{G}_2 + 3\mathbb{Z}_p$. Using BLS12–381, where compressed elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ require 48 and 96 bytes, respectively, and field elements in $\mathbb{Z}_p$ require 32 bytes, the total signature size is $3 \cdot 48 + 1 \cdot 96 + 3 \cdot 32 = 336$ bytes $= 2688$ bits. Hence, our ring signatures are constant-size and require roughly 3 Kbits, independently of the ring size. Thus, our construction provides the first entirely practical and concretely efficient constant-size ring signature in the dlog setting without using heavy machinery such as SNARKs to prove bilinear group arithmetic.

6.2 Succinct Data Availability Sampling

Data availability sampling addresses a major blockchain challenge—scalability [9,60]. This approach allows light clients to verify the availability and integrity of block data using multi-dimensional Reed-Solomon codes within an erasure coding strategy. Ethereum has shifted from fraud proofs, as highlighted in [9], to validity proofs based on polynomial commitments, leveraging their homomorphic properties. As a result, Ethereum aims to integrate this mechanism into its sharding protocol[8]. In this setting, each blockchain validator, similar to light clients, only needs to store the commitment instead of the full dataset and proof. However, within this multi-dimensional structure, clients with

[7] While there are approaches to convert such constructions to prime-order bilinear groups [45], this will not yield to anything near practical.

[8] https://notes.ethereum.org/@vbuterin/protodankshardingfaq

limited resources must store a tuple $\mathsf{com} = (C_1, \ldots, C_q)$, where each C_i is a KZG commitment [62] to a Reed-Solomon code (tensor code) [60]. We think this application can benefit from our SPVC schemes by reducing the commitment size (and, consequently, the communication size) from 256 elements per block to just one, allowing client storage to decrease from 256 KZG commitments per shard to a single commitment. This is achieved by treating the commitment com of a block of data as a single SPVC commitment.

In our approach, one can create a commitment to these KZG commitments for a more succinct representation. Using our weakly binding SPVC, this can be achieved without significant modifications to the KZG scheme or the SPVC itself. Also for q-DHE PSVC, we can easily extend the public parameters of KZG with those received from the q-DHE PSVC to compute KZG commitments in different bases. For example, assume we have $(B_1, B_2, \hat{B}_1, \hat{B}_2)$ alongside $(P^{x^i})_{i\in[q]}$, where x is the KZG trapdoor. We need to compute $(B_1^{x^i})_{i\in[q]}$ and $(B_2^{x^i})_{i\in[q]}$ to derive a KZG commitment with respect to our SPVC message space. We leave the full construction for future work.

6.3 Potential Application: Algebraic Verkle Trees

In a stateless blockchain client model, nodes do not store the entire state but rely on "witnesses" — compact proofs that verify the necessary state for transaction validation. Verkle Trees[9], which improve upon Merkle Trees by using VCs at the leaf nodes, enable smaller witness sizes and more efficient verification.

From a theoretical point of view, a key challenge has been the inability to commit to commitments within the same algebraic structure due to the lack of structure-preserving VCs. Algebraic Verkle Trees (AVTs) with the WSPVC resolve this issue, enabling seamless commitment to commitments without switching between cryptographic primitives like hash functions. As already mentioned, WSPVC is sufficient for stateless validation as commitments are always honestly produced through (Byzantine) agreement on a sequence of updates. This can potentially simplify verification and reduce witness sizes. AVTs also expand the Verkle Tree in both depth and width while maintaining constant-size proofs by incrementally committing to VC commitments. Moreover, we can efficiently prove knowledge of a message without needing to prove the pre-image of a hash in a SNARK. We believe that the use of SPVC in AVTs can pave the way for future research, particularly in optimizing scalability by integrating structure-preserving VCs into frameworks, which could lead to more efficient techniques in stateless blockchain. We leave this application as an interesting direction for future work.

7 Conclusion and Future Work

In this paper, we show that strictly structure-preserving compressing primitives can be realized. We present the first strictly structure-preserving commitment that is shrinking, and in particular, constant-size. By employing a more structured message space—specifically, a variant of the DH message space—we circumvent existing impossibility results. As our main contribution, we construct structure-preserving vector commitments (SPVC) and accumulators (SPA). We begin by discussing generic constructions and then provide concrete implementations under the Diffie-Hellman Exponent assumption. Our main application is the first entirely practical constant-size ring signature scheme in bilinear groups (i.e., the discrete logarithm setting).

We view this work as the initial step toward building structure-preserving compressing primitives, and the first stepping stone for further study in this direction. While our scheme gives good insight, building a fully-featured, unrestricted solution remains an open problem for future research. One interesting direction for future work is designing a vector commitment scheme that utilizes a more natural message space—such as messages as simple as P^m —instead of relying on a pp. Apart from that, it would be interesting to explore methods of compressing and shrinking every message msg, currently of size q, to constant or sublinear size as future work. Extending our accumulators to achieve standard soundness (i.e., collision resistance) is an interesting direction for future work. Moreover, further exploration of applications for our schemes appears to be a promising research direction.

[9] Vitalik Buterin, Guillaume Ballet, Dankrad Feist. Verkle tree EIP, 2021.

Acknowledgments. Omid Mir and Stephan Krenn were supported by the European Union's Horizon Europe projects SUNRISE and LICORICE (grant agreement no. 101073821 and 101168311), PREPARED, a project funded by the Austrian security research programme KIRAS of the Federal Ministry of Finance (BMF), and the CHIST-ERA project REMINDER through the Austrian Science Fund (FWF) project number I 6650-N. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the funding agencies. Neither European Union nor the granting authorities can be held responsible for them. We want to thank anonymous reviewers for their helpful comments. We are also grateful to Gonzalo Martínez de Sola, Gennaro Avitabile and Dario Fiore for helpful comments on our strong collision resistance notion of accumulators, which we now called bounded soundness to avoid confusion and we now cleanly separate the earlier parametrization of the assumption. Their comments helped us to significantly improve the presentation and to make the role of the previously used value *opt* more explicit and clear. Moreover, we thank them for sharing their writeup which made us aware of the fact that we have to extend the NIZK proof in our ring signature application to be able to extract all the values required. Finally, we want to thank them for pointing out an issue with position-binding in our previous construction.

References

1. Abe, M.: Structure-preserving cryptography. Invited Talk at Asiacrypt 2015, https://www.iacr.org/archive/asiacrypt2015/94520356/94520356.pdf
2. Abe, M., Chow, S.S.M., Haralambiev, K., Ohkubo, M.: Double-trapdoor anonymous tags for traceable signatures. In: Lopez, J., Tsudik, G. (eds.) ACNS 11. LNCS, vol. 6715, pp. 183–200 (Jun 2011). https://doi.org/10.1007/978-3-642-21554-4_11
3. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236 (Aug 2010). https://doi.org/10.1007/978-3-642-14623-7_12
4. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. Journal of Cryptology **29**(2), 363–421 (Apr 2016). https://doi.org/10.1007/s00145-014-9196-7
5. Abe, M., Groth, J., Kohlweiss, M., Ohkubo, M., Tibouchi, M.: Efficient fully structure-preserving signatures and shrinking commitments. Journal of Cryptology **32**(3), 973–1025 (Jul 2019). https://doi.org/10.1007/s00145-018-9300-5
6. Abe, M., Haralambiev, K., Ohkubo, M.: Group to group commitments do not shrink. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 301–317 (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_19
7. Abe, M., Kohlweiss, M., Ohkubo, M., Tibouchi, M.: Fully structure-preserving signatures and shrinking commitments. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 35–65 (Apr 2015). https://doi.org/10.1007/978-3-662-46803-6_2
8. Acar, T., Nguyen, L.: Revocation for delegatable anonymous credentials. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 423–440 (Mar 2011). https://doi.org/10.1007/978-3-642-19379-8_26
9. Al-Bassam, M., Sonnino, A., Buterin, V.: Fraud and data availability proofs: Maximising light client security and scaling blockchains with dishonest majorities. arXiv preprint arXiv:1809.09044 (2018)
10. Albrecht, M.R., Cini, V., Lai, R.W.F., Malavolta, G., Thyagarajan, S.A.K.: Lattice-based SNARKs: Publicly verifiable, preprocessing, and recursively composable - (extended abstract). In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 102–132 (Aug 2022). https://doi.org/10.1007/978-3-031-15979-4_4
11. Au, M.H., Tsang, P.P., Susilo, W., Mu, Y.: Dynamic universal accumulators for DDH groups and their application to attribute-based anonymous credential systems. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 295–308 (Apr 2009). https://doi.org/10.1007/978-3-642-00862-7_20
12. Au, M.H., Wu, Q., Susilo, W., Mu, Y.: Compact e-cash from bounded accumulator. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 178–195 (Feb 2007). https://doi.org/10.1007/11967668_12
13. Avitabile, G., Fiore, D., de Sola, G.M.: Structure-preserving compressing primitives, revisited. Cryptology ePrint Archive, Paper 2026/XXX (2023), https://eprint.iacr.org/2026/XXX, to appear at Asiacrypt 2026
14. Avitabile, G., Fiore, D., de Sola, G.M.: Structure-preserving compressing primitives, revisited. Cryptology ePrint Archive, Paper 2026/1952 (2026), https://eprint.iacr.org/2026/1952
15. Bari, N., Pfitzmann, B.: Collision-free accumulators and fail-stop signature schemes without trees. In: Fumy, W. (ed.) EUROCRYPT'97. LNCS, vol. 1233, pp. 480–494 (May 1997). https://doi.org/10.1007/3-540-69053-0_33

16. Barreto, P.S.L.M., Lynn, B., Scott, M.: Constructing elliptic curves with prescribed embedding degrees. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 02. LNCS, vol. 2576, pp. 257–267 (Sep 2003). https://doi.org/10.1007/3-540-36413-7_19
17. Barthoulot, A., Blazy, O., Canard, S.: Cryptographic accumulators: new definitions, enhanced security, and delegatable proofs. In: International Conference on Cryptology in Africa. pp. 94–119. Springer (2024)
18. Bauer, B., Fuchsbauer, G., Loss, J.: A classification of computational assumptions in the algebraic group model. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part II. LNCS, vol. 12171, pp. 121–151 (Aug 2020). https://doi.org/10.1007/978-3-030-56880-1_5
19. Ben-Sasson, E., Chiesa, A., Green, M., Tromer, E., Virza, M.: Secure sampling of public parameters for succinct zero knowledge proofs. In: 2015 IEEE Symposium on Security and Privacy. pp. 287–304. IEEE Computer Society Press (May 2015). https://doi.org/10.1109/SP.2015.25
20. Benabbas, S., Gennaro, R., Vahlis, Y.: Verifiable delegation of computation over large datasets. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 111–131 (Aug 2011). https://doi.org/10.1007/978-3-642-22792-9_7
21. Benaloh, J.C., de Mare, M.: One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In: Helleseth, T. (ed.) EUROCRYPT'93. LNCS, vol. 765, pp. 274–285 (May 1994). https://doi.org/10.1007/3-540-48285-7_24
22. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. Journal of Cryptology **22**(1), 114–138 (Jan 2009). https://doi.org/10.1007/s00145-007-9011-9
23. Bose, P., Das, D., Rangan, C.P.: Constant size ring signature without random oracle. In: Foo, E., Stebila, D. (eds.) ACISP 15. LNCS, vol. 9144, pp. 230–247 (Jun / Jul 2015). https://doi.org/10.1007/978-3-319-19962-7_14
24. Camenisch, J., Kohlweiss, M., Soriente, C.: An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 481–500 (Mar 2009). https://doi.org/10.1007/978-3-642-00468-1_27
25. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76 (Aug 2002). https://doi.org/10.1007/3-540-45708-9_5
26. Campanelli, M., Fiore, D., Greco, N., Kolonelos, D., Nizzardo, L.: Incrementally aggregatable vector commitments and applications to verifiable decentralized storage. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part II. LNCS, vol. 12492, pp. 3–35 (Dec 2020). https://doi.org/10.1007/978-3-030-64834-3_1
27. Campanelli, M., Fiore, D., Han, S., Kim, J., Kolonelos, D., Oh, H.: Succinct zero-knowledge batch proofs for set accumulators. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) ACM CCS 2022. pp. 455–469. ACM Press (Nov 2022). https://doi.org/10.1145/3548606.3560677
28. Campanelli, M., Nitulescu, A., Ràfols, C., Zacharakis, A., Zapico, A.: Linear-map vector commitments and their practical applications. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part IV. LNCS, vol. 13794, pp. 189–219 (Dec 2022). https://doi.org/10.1007/978-3-031-22972-5_7
29. Catalano, D., Fiore, D.: Vector commitments and their applications. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 55–72 (Feb / Mar 2013). https://doi.org/10.1007/978-3-642-36362-7_5
30. Catalano, D., Fiore, D., Gennaro, R., Giunta, E.: On the impossibility of algebraic vector commitments in pairing-free groups. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part II. LNCS, vol. 13748, pp. 274–299 (Nov 2022). https://doi.org/10.1007/978-3-031-22365-5_10
31. Chator, A., Green, M., Tiwari, P.R.: Sok: Privacy-preserving signatures. Cryptology ePrint Archive (2023)
32. Couteau, G., Hartmann, D.: Shorter non-interactive zero-knowledge arguments and ZAPs for algebraic languages. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 768–798 (Aug 2020). https://doi.org/10.1007/978-3-030-56877-1_27
33. Couteau, G., Lipmaa, H., Parisella, R., Ødegaard, A.T.: Efficient NIZKs for algebraic sets. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part III. LNCS, vol. 13092, pp. 128–158 (Dec 2021). https://doi.org/10.1007/978-3-030-92078-4_5
34. Crites, E.C., Kohlweiss, M., Preneel, B., Sedaghat, M., Slamanig, D.: Threshold structure-preserving signatures. In: ASIACRYPT 2023, Part II. pp. 348–382. LNCS (Dec 2023). https://doi.org/10.1007/978-981-99-8724-5_11
35. Crites, E.C., Lysyanskaya, A.: Delegatable anonymous credentials from mercurial signatures. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 535–555 (Mar 2019). https://doi.org/10.1007/978-3-030-12612-4_27
36. Damgård, I., Triandopoulos, N.: Supporting non-membership proofs with bilinear-map accumulators. Cryptology ePrint Archive, Report 2008/538 (2008), https://eprint.iacr.org/2008/538
37. Derler, D., Hanser, C., Slamanig, D.: A new approach to efficient revocable attribute-based anonymous credentials. In: Groth, J. (ed.) Cryptography and Coding - 15th IMA International Conference, IMACC 2015, Oxford, UK, December 15-17, 2015. Proceedings. Lecture Notes in Computer Science, vol. 9496, pp. 57–74. Springer (2015). https://doi.org/10.1007/978-3-319-27239-9_4, https://doi.org/10.1007/978-3-319-27239-9_4

38. Derler, D., Hanser, C., Slamanig, D.: Revisiting cryptographic accumulators, additional properties and relations to other primitives. In: Nyberg, K. (ed.) CT-RSA 2015. LNCS, vol. 9048, pp. 127–144 (Apr 2015). https://doi.org/10.1007/978-3-319-16715-2_7
39. Derler, D., Ramacher, S., Slamanig, D.: Post-quantum zero-knowledge proofs for accumulators with applications to ring signatures from symmetric-key primitives. In: Lange, T., Steinwandt, R. (eds.) Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10786, pp. 419–440. Springer (2018). https://doi.org/10.1007/978-3-319-79063-3_20, https://doi.org/10.1007/978-3-319-79063-3_20
40. Derler, D., Slamanig, D.: Key-homomorphic signatures: definitions and applications to multiparty signatures and non-interactive zero-knowledge. DCC **87**(6), 1373–1413 (2019). https://doi.org/10.1007/s10623-018-0535-9
41. Dodis, Y., Kiayias, A., Nicolosi, A., Shoup, V.: Anonymous identification in ad hoc groups. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 609–626 (May 2004). https://doi.org/10.1007/978-3-540-24676-3_36
42. Faust, S., Kohlweiss, M., Marson, G.A., Venturi, D.: On the non-malleability of the Fiat-Shamir transform. In: Galbraith, S.D., Nandi, M. (eds.) INDOCRYPT 2012. LNCS, vol. 7668, pp. 60–79 (Dec 2012). https://doi.org/10.1007/978-3-642-34931-7_5
43. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194 (Aug 1987). https://doi.org/10.1007/3-540-47721-7_12
44. Fischlin, M.: Communication-efficient non-interactive proofs of knowledge with online extractors. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 152–168 (Aug 2005). https://doi.org/10.1007/11535218_10
45. Freeman, D.M.: Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 44–61 (May / Jun 2010). https://doi.org/10.1007/978-3-642-13190-5_3
46. Fuchsbauer, G.: Automorphic signatures in bilinear groups and an application to round-optimal blind signatures. Cryptology ePrint Archive, Report 2009/320 (2009), https://eprint.iacr.org/2009/320
47. Fuchsbauer, G.: Commuting signatures and verifiable encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 224–245 (May 2011). https://doi.org/10.1007/978-3-642-20465-4_14
48. Fuchsbauer, G., Hanser, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 233–253 (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_12
49. Fuchsbauer, G., Hanser, C., Slamanig, D.: Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. Journal of Cryptology **32**(2), 498–546 (Apr 2019). https://doi.org/10.1007/s00145-018-9281-4
50. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 33–62 (Aug 2018). https://doi.org/10.1007/978-3-319-96881-0_2
51. Ghadafi, E.: More efficient structure-preserving signatures - or: Bypassing the type-iii lower bounds. In: Foley, S.N., Gollmann, D., Snekkenes, E. (eds.) Computer Security - ESORICS 2017 - 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10493, pp. 43–61. Springer (2017). https://doi.org/10.1007/978-3-319-66399-9_3, https://doi.org/10.1007/978-3-319-66399-9_3
52. Ghadafi, E.: Further lower bounds for structure-preserving signatures in asymmetric bilinear groups. In: Buchmann, J., Nitaj, A., eddine Rachidi, T. (eds.) AFRICACRYPT 19. LNCS, vol. 11627, pp. 409–428 (Jul 2019). https://doi.org/10.1007/978-3-030-23696-0_21
53. Ghosh, E., Ohrimenko, O., Papadopoulos, D., Tamassia, R., Triandopoulos, N.: Zero-knowledge accumulators and set algebra. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 67–100 (Dec 2016). https://doi.org/10.1007/978-3-662-53890-6_3
54. Goldreich, O.: Foundations of cryptography: Volume 1, basic tools, 2001. Google Scholar Google Scholar Digital Library Digital Library **5**
55. Gorbunov, S., Reyzin, L., Wee, H., Zhang, Z.: Pointproofs: Aggregating proofs for multiple vector commitments. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 2007–2023. ACM Press (Nov 2020). https://doi.org/10.1145/3372297.3417244
56. Griffy, S., Lysyanskaya, A., Mir, O., Kempner, O.P., Slamanig, D.: Delegatable anonymous credentials from mercurial signatures with stronger privacy. Cryptology ePrint Archive, Paper 2024/1216, to appear at ASIACRYPT'24 (2024), https://eprint.iacr.org/2024/1216
57. Groth, J.: On the size of pairing-based non-interactive arguments. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 305–326 (May 2016). https://doi.org/10.1007/978-3-662-49896-5_11

58. Groth, J., Kohlweiss, M., Maller, M., Meiklejohn, S., Miers, I.: Updatable and universal common reference strings with applications to zk-SNARKs. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part III. LNCS, vol. 10993, pp. 698–728 (Aug 2018). https://doi.org/10.1007/978-3-319-96878-0_24
59. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432 (Apr 2008). https://doi.org/10.1007/978-3-540-78967-3_24
60. Hall-Andersen, M., Simkin, M., Wagner, B.: Foundations of data availability sampling. Cryptology ePrint Archive (2023)
61. Jhanwar, M.P., Safavi-Naini, R.: Compact accumulator using lattices. Cryptology ePrint Archive, Report 2014/1015 (2014), https://eprint.iacr.org/2014/1015
62. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 177–194 (Dec 2010). https://doi.org/10.1007/978-3-642-17373-8_11
63. Krenn, S., Mir, O., Slamanig, D.: Structure-preserving compressing primitives: Vector commitments and accumulators and applications. Cryptology ePrint Archive, Paper 2024/1619 (2024), https://eprint.iacr.org/2024/1619
64. Krenn, S., Mir, O., Slamanig, D.: Structure-preserving compressing primitives: Vector commitments and accumulators and applications. In: Bai, S., Persichetti, E. (eds.) Public-Key Cryptography – PKC 2026. pp. 389–423. Springer Nature Switzerland, Cham (2026)
65. Lai, R.W.F., Malavolta, G.: Subvector commitments with application to succinct arguments. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 530–560 (Aug 2019). https://doi.org/10.1007/978-3-030-26948-7_19
66. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 1–31 (May 2016). https://doi.org/10.1007/978-3-662-49896-5_1
67. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. Journal of Cryptology **36**(3), 23 (Jul 2023). https://doi.org/10.1007/s00145-023-09470-6
68. Libert, B., Peters, T., Joye, M., Yung, M.: Linearly homomorphic structure-preserving signatures and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 289–307 (Aug 2013). https://doi.org/10.1007/978-3-642-40084-1_17
69. Libert, B., Peters, T., Yung, M.: Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 296–316 (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_15
70. Libert, B., Ramanna, S.C., Yung, M.: Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions. In: Chatzigiannakis, I., Mitzenmacher, M., Rabani, Y., Sangiorgi, D. (eds.) ICALP 2016. LIPIcs, vol. 55, pp. 30:1–30:14. Schloss Dagstuhl (Jul 2016). https://doi.org/10.4230/LIPIcs.ICALP.2016.30
71. Libert, B., Yung, M.: Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 499–517 (Feb 2010). https://doi.org/10.1007/978-3-642-11799-2_30
72. Lipmaa, H., Parisella, R.: Set (non-)membership NIZKs from determinantal accumulators. pp. 352–374. LNCS (2023). https://doi.org/10.1007/978-3-031-44469-2_18
73. Lu, D., Yu, A., Kate, A., Maji, H.: Polymath: Low-latency MPC via secure polynomial evaluations and its applications. Cryptology ePrint Archive, Report 2021/978 (2021), https://eprint.iacr.org/2021/978
74. Malavolta, G., Schröder, D.: Efficient ring signatures in the standard model. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part II. LNCS, vol. 10625, pp. 128–157 (Dec 2017). https://doi.org/10.1007/978-3-319-70697-9_5
75. Maurer, U.M.: Unifying zero-knowledge proofs of knowledge. In: Preneel, B. (ed.) AFRICACRYPT 09. LNCS, vol. 5580, pp. 272–286 (Jun 2009). https://doi.org/10.1007/978-3-642-02384-2_17
76. Merkle, R.C.: A digital signature based on a conventional encryption function. In: Pomerance, C. (ed.) CRYPTO'87. LNCS, vol. 293, pp. 369–378 (Aug 1988). https://doi.org/10.1007/3-540-48184-2_32
77. Mir, O., Bauer, B., Griffy, S., Lysyanskaya, A., Slamanig, D.: Aggregate signatures with versatile randomization and issuer-hiding multi-authority anonymous credentials. In: ACM CCS 2023. pp. 30–44. ACM Press (Nov 2023). https://doi.org/10.1145/3576915.3623203
78. Mir, O., Slamanig, D., Bauer, B., Mayrhofer, R.: Practical delegatable anonymous credentials from equivalence class signatures. In: The 23rd Privacy Enhancing Technologies Symposium. pp. 488–513. de Gruyter (2023)
79. Mitrokotsa, A., Mukherjee, S., Sedaghat, M., Slamanig, D., Tomy, J.: Threshold structure-preserving signatures: Strong and adaptive security under standard assumptions. In: PKC 2024, Part I. pp. 163–195. LNCS (May 2024). https://doi.org/10.1007/978-3-031-57718-5_6

80. Nguyen, L.: Accumulators from bilinear pairings and applications. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 275–292 (Feb 2005). https://doi.org/10.1007/978-3-540-30574-3_19
81. Nikolaenko, V., Ragsdale, S., Bonneau, J., Boneh, D.: Powers-of-tau to the people: Decentralizing setup ceremonies. pp. 105–134. LNCS (Jun 2024). https://doi.org/10.1007/978-3-031-54776-8_5
82. Nitulescu, A.: Sok: Vector commitments (2021), https://www.di.ens.fr/~nitulesc/files/vc-sok.pdf
83. Papamanthou, C., Shi, E., Tamassia, R., Yi, K.: Streaming authenticated data structures. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 353–370 (May 2013). https://doi.org/10.1007/978-3-642-38348-9_22
84. Peikert, C., Pepin, Z., Sharp, C.: Vector and functional commitments from lattices. In: Nissim, K., Waters, B. (eds.) TCC 2021, Part III. LNCS, vol. 13044, pp. 480–511 (Nov 2021). https://doi.org/10.1007/978-3-030-90456-2_16
85. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565 (Dec 2001). https://doi.org/10.1007/3-540-45682-1_32
86. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT'97. LNCS, vol. 1233, pp. 256–266 (May 1997). https://doi.org/10.1007/3-540-69053-0_18
87. Slamanig, D.: Dynamic accumulator based discretionary access control for outsourced storage with unlinkable access - (short paper). In: Keromytis, A.D. (ed.) FC 2012. LNCS, vol. 7397, pp. 215–222 (Feb / Mar 2012). https://doi.org/10.1007/978-3-642-32946-3_16
88. Martınez de Sola Monereo, G.: Structure preserving accumulators with applications to short threshold ring signatures (2025)
89. Tomescu, A., Abraham, I., Buterin, V., Drake, J., Feist, D., Khovratovich, D.: Aggregatable subvector commitments for stateless cryptocurrencies. In: Galdi, C., Kolesnikov, V. (eds.) SCN 20. LNCS, vol. 12238, pp. 45–64 (Sep 2020). https://doi.org/10.1007/978-3-030-57990-6_3
90. Wee, H., Wu, D.J.: Succinct vector, polynomial, and functional commitments from lattices. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part III. LNCS, vol. 14006, pp. 385–416 (Apr 2023). https://doi.org/10.1007/978-3-031-30620-4_13

A Weakly Binding Vector Commitments

We present a simple compiler in which commitments are derived from signature public keys, with auxiliary data including the corresponding secret keys and messages, while the opening/proof is simply a signature. To bind a position to a message, we introduce public information $(U_1, \ldots, U_q) \in \mathcal{M}$ into the public parameters, where random elements U_i are signed together with M_i as indices to the messages. This approach is compatible with any compact SPS. We propose an instantiation of WSPVC using the FHS signature [49], noting that message randomization is not required in this context. This message randomization can be prevented by fixing the first element of the message vector to a predetermined element U, which needs to be verified during the verification. We note that WSPVC only applies in applications where the committer (or the party generating accumulators) is honest, such as in algebraic verkle trees, where commitments are always honestly produced through (Byzantine) agreement on a sequence of updates (see Sec 6.3 for more details).

In Sec. 4, we enhance our security model to present a structure-preserving vector commitment with standard binding by incorporating message tags (identifiers). This ensures that messages will not verify unless they are computed using a compatible setup pp.

We present our weakly binding vector commitment WSPVC, which, as mentioned above, can be constructed from a structure-preserving (SP) signature, assuming the committer is honest and the commitments are generated correctly.

Scheme 5 (Weakly Binding VC) *A* WSPVC *is a tuple of the following algorithms:*

$\mathsf{KeyGen}(1^\lambda, q)$: Given the security parameter λ and the size q of the committed vector, the key generation run $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P, \hat{P}) \leftarrow \mathsf{BGSetup}(1^\lambda)$, for $i = 1, \ldots, q$, choose $U_i \leftarrow\!\!\$\ \mathbb{G}_1$ and outputs public parameters $\mathsf{pp} = (\mathsf{BG}, U_1, \ldots, U_q)$.

$\mathsf{Commit}(M_1, \ldots, M_q)$: On input a vector of q messages as $(M_1, \ldots, M_q) \in \mathcal{M} \in \mathbb{G}_1^q$ and the public parameters pp, output a commitment com and auxiliary information aux as follows: Run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{SPS.KeyGen}(1^\lambda, 2)$ and then set

$$\mathsf{com} = \mathsf{pk} \quad \wedge \quad \mathsf{aux} = (\sigma_1, \ldots, \sigma_q)$$

Where σ_i is a SPS signature on (M_i, U_i). For random $y \leftarrow\!\!\$\ \mathbb{Z}_p$, and using the FHS SPS as example:

$$\mathsf{pk} = (\hat{X}_1 = \hat{P}^{x_1}, \hat{X}_2 = \hat{P}^{x_2}) \wedge \sigma_i = \left(Z = (U_i^{x_1} \cdot M_i^{x_2})^{1/y}, Y = P^y, \hat{Y} = \hat{P}^y\right)$$

$\mathsf{Open}(M, i, \mathsf{aux})$: This algorithm is run by the committer to produce a proof π_i that M is the i-th committed message. Pick the related signature $\sigma_i \leftarrow \mathsf{aux}$ and output a proof $\pi_i = \sigma_i$ for M_i.

$\mathsf{Verify}(\mathsf{com}, M, i, \pi_i)$: The verification algorithm accepts (i.e., it outputs 1) only if π_i is a valid proof that com was created for M_i by verifying $\mathsf{SPS.Verify}(\mathsf{pk} = \mathsf{com}, M_i, \sigma_i = \pi_i) = 1$. For the FHS signature scheme, this is defined as follows, parse the proof $\pi_i = (Z, Y, \hat{Y})$, the commitment $\mathsf{com} = (X_1, X_2)$ and check:

$$\mathsf{SPS.Verify} : e(Z, \hat{Y}) = e(M_i, \hat{X}_2)e(U_i, \hat{X}_1) \wedge e(Y, \hat{P}) = e(P, \hat{Y})$$

$\mathsf{Rand}(\mathsf{com}, \pi_i)$: Randomize a proof for a message M_i as: Pick a random $\mu \leftarrow\!\!\$\ \mathbb{Z}_p$:

$$\pi_i' = (\sigma_i' = (Z^{1/\mu}.Y^{\mu}, \hat{Y}^{\mu}))$$

Theorem 7. *If the SPS is unforgeable, then the* WSPVC *in Scheme 5 satisfies weak binding.*

Proof (Sketch). The proof of binding is straightforward. If the signature is unforgeable, then the commitment is position binding; specifically, a new proof requires a new signature on a different message, which would violate the unforgeability of our signature scheme. □

Remark 6. Note that since the commitment serves as a public key and is independent of the message, updating the commitments for a new message can be achieved simply by signing the new message. Moreover, the message space for the commitment can be adapted by selecting a suitable SPS. Specifically, we can configure the SPVC to handle unilateral messages, where messages are drawn solely from either $\mathbb{G}_1$ or $\mathbb{G}_2$, or bilateral messages, which allow for a mix of elements from both $\mathbb{G}_1$ and $\mathbb{G}_2$.

B Accumulator Construction from SPVC

Let $\mathsf{SPVC} = (\mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ be a structure-preserving vector commitment scheme supporting vectors of length q. We construct a bounded accumulator $\mathsf{Acc} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{WitCreate}, \mathsf{Verify})$ as follows.

- $\mathsf{Setup}(1^\lambda, q)$: First sample pp as the public parameters of the SPVC scheme. Then sample pairwise distinct group elements $(U_1, \ldots, U_q)$.[10]
  Set the accumulator public key as
  $$\mathsf{pk}_{acc} \leftarrow (\mathsf{pp}, (U_1, \ldots, U_q)),$$
  and set $\mathsf{sk}_{acc} \leftarrow \perp$.
- $\mathsf{Eval}(\mathsf{pk}_{acc}, \mathcal{X})$: Let $\mathcal{X} = \{x_1, \ldots, x_\ell\}$ with $\ell \leq q$. If $\ell < q$, pad $\mathcal{X}$ with dummy elements so that the resulting multiset has size exactly q. (The dummy elements are chosen such that no valid witness can be generated for them.)
  Define an injective mapping
  $$\mathsf{aux} : \mathcal{X} \to [q]$$
  assigning each $x \in \mathcal{X}$ a unique index $i_x \in [q]$. Let $\mathsf{msg} = (M_1, \ldots, M_q)$ be the vector defined by
  $$M_{i_x} := x \quad \text{for all } x \in \mathcal{X},$$
  and M_j equal to the dummy value at all remaining positions.
  Compute
  $$(\mathsf{Acc}, \mathsf{aux}_{vc}) \leftarrow \mathsf{SPVC.Commit}(\mathsf{pp}, \mathsf{msg}).$$
  Output the accumulator value Acc and auxiliary information
  $$\mathsf{aux}_{acc} := (\mathsf{aux}_{vc}, \mathsf{aux}).$$
- $\mathsf{WitCreate}(\mathsf{pk}_{acc}, x, \mathsf{aux}_{acc})$: Parse $\mathsf{aux}_{acc} = (\mathsf{aux}_{vc}, \mathsf{aux})$. Retrieve the index $i_x := \mathsf{aux}(x)$. Compute
  $$\pi_x \leftarrow \mathsf{SPVC.Open}(\mathsf{pp}, \mathsf{msg}, i_x, \mathsf{aux}_{vc}).$$
  Output the witness
  $$wit_x := (U_{i_x}, \pi_x).$$
- $\mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}, x, wit_x)$: Parse $wit_x = (U_{i_x}, \pi_x)$. Check that U_{i_x} corresponds to the claimed index i_x. Then output the result of
  $$\mathsf{SPVC.Verify}(\mathsf{pp}, \mathsf{Acc}, x, i_x, \pi_x).$$

[10] For example, one may set $U_i = P^i$ for a generator P. These elements are only used to bind committed messages to fixed public positions.

C Correctness for SPVC

Correctness: If commitments are properly generated, then proofs will always satisfy the verification, which can be seen as follows:

$$\begin{aligned}
e(C, \hat{B}_{q+1-i}) &= e\left(P^r \cdot \prod_{j\in[q]} M_j, \hat{B}_{q+1-i}\right) \\
&= e\left(P^r \cdot P^{\sum_{j\in[q]} \alpha^j \cdot m_j}, \hat{P}^{\alpha^{q+1-i}}\right) \\
&= e\left(P^{r\cdot\alpha^{q+1-i}} \cdot (P^{\sum_{j\in[q]} \alpha^j \cdot m_j})^{\alpha^{q+1-i}}, \hat{P}\right) \\
&= e\left(B_{q+1-i}^r, \hat{P}\right) \cdot e\left((P^{\sum_{j\in[q]} \alpha^j \cdot m_j})^{\alpha^{q+1-i}}, \hat{P}\right) \\
&= e\left(B_{q+1-i}^r, \hat{P}\right) \cdot e\left(P^{\sum_{j\in[q]} m_j \cdot \alpha^{q+1-i+j}}, \hat{P}\right) \\
&= e\left(B_{q+1-i}^r, \hat{P}\right) \cdot e\left(P^{\sum_{j\in[q]\setminus\{i\}} m_j \cdot \alpha^{q+1-i+j}}, \hat{P}\right) \cdot e\left(P^{m_i\alpha^{q+1}}, \hat{P}\right) \\
&= e\left(B_{q+1-i}^r \cdot \prod_{j\in[q]\setminus\{i\}} P^{m_j\cdot\alpha^{q+1-i+j}}, \hat{P}\right) \cdot e\left(P^{\alpha^i\cdot m_i}, \hat{P}^{\alpha^{q+1-i}}\right) \\
&= e\left((B_{q+1-i}^r \cdot \prod_{j\in[q]\setminus\{i\}} T_{j,q+1-i+j}), \hat{P}\right) \cdot e\left(M_{i1}, \hat{B}_{q+1-i}\right) \\
&= e\left(\pi_i, P\right) \cdot e\left(M_{i0}, \hat{B}_{q+1}\right)
\end{aligned}$$

The second verification holds trivially $e(M_{i0}, \hat{B}_{2q}) = e(T_{i,2q}, \hat{P})$.

C.1 Alternative Proof for SPVC

In this strategy, our main technical result is to prove that our scheme satisfies binding in the generic group model (GGM) [86] for asymmetric (type-3) bilinear groups, for which there are no efficiently computable homomorphism between P and $\hat{P}$. In this model, the adversary is only given handles of group elements, which are uniform random strings. To perform group operations, it uses an oracle to which it can submit handles and receives back the handle of the sum, inversion, etc., of the group elements for which it submitted handles.

Before proving the binding property of our scheme, we further need the following auxiliary lemma:

Lemma 1 (Extraction of Discrete Logarithms from Valid Messages). *Let* pp *be a public parameter. If the messages satisfy the following conditions for all* $i \in [q]$*:*

$$e(M_{i1}, \hat{B}_{j-i}) = e(T_{ij}, \hat{P}) \quad \textit{and} \quad e(M_{i1}, \hat{P}) = e(M_{i0}, \hat{B}_i),$$

then the adversary is able to extract the discrete logarithms $\{m_i\}_{i\in[q]}$ *such that:*

$$m_i = \mathsf{dlog}_{B_i}(M_{i1}) = \mathsf{dlog}_{B_j}(T_{ij}) \quad \textit{for } 1 \le i \le q,$$

where $j \in [i+1, i+q] \setminus \{q+1\}$.

Proof. For a fixed i, consider the values $M_{i0}, M_{i1}, \{T_{ij}\}_{j=i+1, j\neq q+1}^{i+q}$ output by an adversary. With P and $\{B_u\}_{u=1,u\neq q+1}^{2q}$ being the values specified in pp, these values must now have the form:

$$M_{i0} = P^{b_0} \cdot \prod_{\substack{k=1\\k\neq q+1}}^{2q} B_k^{b_k}, \quad M_{i1} = P^{c_0} \cdot \prod_{\substack{k=1\\k\neq q+1}}^{2q} B_k^{c_k}, \text{ and } \quad T_{ij} = P^{a_{j0}} \cdot \prod_{\substack{k=1\\k\neq q+1}}^{2q} B_k^{a_{jk}},$$

where all b_u, c_u, a_{ju} are known to the adversary. For the remainder of this proof, all sums and products are over $k = 1, \ldots, 2q$ with $k \neq q+1$, which will be omitted for notational convenience.

Using the structure of the B_j as defined in in Eq. 1 and taking the discrete logarithm in P we obtain:

$$m_{i0} = b_0 + \sum b_k \alpha^k \tag{6}$$
$$m_{i1} = c_0 + \sum c_k \alpha^k \tag{7}$$
$$t_{ij} = a_{j0} + \sum a_{jk} \alpha^k \quad \forall j = i+1, \ldots, i+q,\ j \neq q+1\,. \tag{8}$$

Furthermore, by the verification equations $e(M_{i1}, B_{j-i}) = e(T_{ij}, P)$ and $e(M_{i1}, P) = e(M_{i0}, B_i)$ we obtain by a similar argument that:

$$m_{i1}\alpha^{j-i} = t_{ij} \qquad \forall j = i+1, \ldots, i+q,\ j \neq q+1 \tag{9}$$
$$m_{i1} = m_{i0}\alpha^i\,. \tag{10}$$

$\mathbf{c_0, \ldots, c_{i-1} = 0}$: By combining Equations (6), (7) and (10) we obtain:

$$c_0 + \sum c_k \alpha^k = b_0 \alpha^i + \sum b_k \alpha^{k+i}\,.$$

Given that the lowest degree on the right hand side is i, we directly obtain that $c_0 = \cdots = c_{i-1} = 0$.

$\mathbf{c_{i+1}, \ldots, c_q = 0}$: By combining Equations (7) to (9) we obtain for all j:

$$c_0 + \sum c_k \alpha^{k+j-i} = a_{j0} + \sum a_{jk} \alpha^k\,. \tag{11}$$

As α^{q+1} does not occur on the right hand side, the term $c_{q+1-j+i}\alpha^{q+1}$ on the left hand side must be 0 for all $j = i+1, \ldots, i+q$ satisfying $j \neq q+1$. Thus, in particular for $j = i+1, \ldots, q$, it follows that $c_{i+1} = \cdots = c_q = 0$.

$\mathbf{c_{q+2}, \ldots, c_{q+i-1}, c_{q+i+1}, \ldots, c_{2q} = 0}$: In (11), the highest degree on the right hand side equals $2q$. Thus, the coefficients of α^{2q+1} on the right hand side equals 0, i.e., $c_{2q+1-j+i} = 0$ for all $j = i+1, \ldots, i+q$ satisfying $j \neq q+1$. This immediately yields $c_{q+2} = \cdots = c_{2q} = 0$ except for c_{q+i} corresponding to $j = q+1$.

$\mathbf{c_{q+i} = 0}$: In the case that $i = 1$, there is nothing to prove as there is no c_{q+1} in (7). For $i > 1$, consider the term $c_{q+i}\alpha^{q+j}$ in (11) for $j = q+2$. As α^{2q+2} does not exist on the right hand side, it directly follows that $c_{q+i} = 0$.

Combining the above observations we obtain that $c_k = 0$ for all $k \neq i$. Rewriting (11) now yields for all j that:

$$c_i \alpha^j = a_{j0} + \sum a_{jk} \alpha^k\,.$$

Comparing coefficients gives us that $a_{jk} = 0$ for all $k \neq j$ and $a_{jj} = c_i$, such that $t_{ij} = c_i \alpha^j$.

Overall, this implies that $M_{i1} = P^{c_0} \cdot \prod B_k^{c_k} = B_i^{c_i}$ and $T_{ij} = P^{a_{j0}} \cdot \prod B_k^{a_{jk}} = B_j^{c_i}$, or equivalently $m_i := c_i = \mathsf{dlog}_{B_i} M_i = \mathsf{dlog}_{B_j} T_{ij}$ for all $j = i+1, \ldots, i+q$ with $j \neq q+1$. Thus, the adversary must be able to extract the discrete log of the message, and thus by induction, must always know the discrete logs of messages during the game. □

Theorem 8. *Scheme 2 is binding in the Generic Group Model (GGM) assuming the q-DHE assumption.*

Proof. Let $\mathcal{A}$ come up with a commitment C, an index $i \in \{1, \ldots, q\}$, an valid openings π_i and π_i' to distinct messages $(\mathbf{M}, \mathbf{T}), (\mathbf{M}', \mathbf{T}')$. By Lemma 1, we can now extract the corresponding $m_i \neq m_i'$.

From the verification equations, it follows that:

$$e(\pi_i, \hat{P}) \cdot e(P^{\alpha^i}, \hat{P}^{\alpha^{q+1-i}})^{m_i} = e(\pi_i', \hat{P}) \cdot e(P^{\alpha^i}, \hat{P}^{\alpha^{q+1-i}})^{m_i'}$$

Simply rewriting yields $e(\pi_i/\pi_i', \hat{P}) = e(P^{\alpha^i}, \hat{P}^{\alpha^{q+1-i}})^{m_i' - m_i}$, or equivalently $e((\pi_i/\pi_i')^{1/(m_i'-m_i)}, \hat{P}) = e(P^{\alpha^i}, \hat{P}^{\alpha^{q+1-i}})$.

Now, since $m_i \neq m_i'$, the latter relation implies that $P^{\alpha^{q+1}} = (\pi_i/\pi_i')^{1/(m_i'-m_i)}$ is revealed by the collision, which contradicts the q-DHE assumption. □

D Proof of Perfect Randomization

We show that our accumulator provides the perfect randomization property (Def. 22) as follows.

Proof. Let $(\mathbf{M}, \mathbf{T}) \in (\mathbb{G}_1^*)^2 \times (\mathbb{G}_1^*)^q \times (\mathbb{G}_2^*)^q$, and $\mathsf{pk}_{acc} \in (\mathbb{G}_1^*)^{2q-1} \times (\mathbb{G}_2^*)^q$. An accumulator, axuiliary information, witness and messages $(\mathsf{Acc}_{\mathcal{X}}, \mathsf{aux}, wit_{msg_i}, msg_i)$ satisfying $\mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}_{\mathcal{X}}, wit_{msg_i}, msg_i) = 1$ are of the form:

$$\mathsf{Acc}_{\mathcal{X}} = \left(P^r \cdot \prod_{i \in [q]} M_{i1} \right)$$

$$wit_{msg_i} = \left(W_1 = (B^r_{q+1-i} \cdot \prod_{j \in [q] \setminus \{i\}} T_{j,q+1-i+j})^\rho, \hat{W}_2 = (\hat{B}_{q+1-i})^\rho, \quad W_3 = T^\rho_{i,2q} \right)$$

$$\text{and and } msg_i = (M^\rho_{i0}).$$

On the other hand, for randomness $(\mu, \gamma) \in \mathbb{Z}_p^*$, the randomization algorithm $\mathsf{Rand}(\mathsf{Acc}, \mathsf{aux}, wit_{msg_i}, msg_i, (\mu, \gamma))$ outputs:

$$\mathsf{Acc}' = \left(P^{r\mu} \cdot \prod_{i \in [q]} M^\mu_{i1} \right),$$

$$wit' = \left(W'_1 = (B^{r\mu \cdot \gamma}_{q+1-i} \cdot \prod_{j \in [q] \setminus \{i\}} T^{\mu\gamma}_{j,q+1-i+j})^\rho, \hat{W}'_2 = (\hat{B}_{q+1-i})^{\rho \cdot \gamma} \quad W_3 = T^{\rho\mu\gamma}_{i,2q} \right)$$

$$\text{and } msg = (M^{\mu\gamma\rho}_{i0})$$

which are uniformly random elements conditioned on $\mathsf{Verify}(\mathsf{pk}_{acc}, \mathsf{Acc}^\mu_{\mathcal{X}}, wit'_{msg_i}, M^{\mu\gamma\rho}_{i0}) = 1$. Indeed, each element is perfectly randomized with fresh randomness, so it is clear that Rand and Eval are identically distributed for all $(\mathsf{Acc}', \mathsf{aux}' wit'_{msg_i}, msg'_i)$. □

E Unforgeability and Anonymity Proofs of Ring Signatures

We first prove unforgeability and then anonymity.

Unforgeability. We prove unforgeability by a sequence of games.

Proof. **Game 0 (Real game).** This is the standard unforgeability experiment for our ring-signature scheme.

Game 1 (Embedding the $\mathsf{BLS}_{\mathsf{Acc}}$ challenge). As in Game 0, except that the reduction $\mathcal{B}$ interacts with an EUF-CMA challenger $\mathcal{C}$ for $\mathsf{BLS}_{\mathsf{Acc}}$. The reduction guesses an index $j \in [q]$ and embeds the challenge tagged public key $t\mathsf{pk}_j$ at this position. All remaining key pairs $(\mathsf{sk}_i, t\mathsf{pk}_i)$, for $i \neq j$, are generated honestly. If $\mathcal{A}$ queries $\mathsf{Key}(j)$, the reduction aborts. Otherwise, key queries are answered honestly. Signature queries are handled as follows.

$\mathsf{Key}(i)$**:** Return sk_i and set $Q_{\mathsf{Key}} := Q_{\mathsf{Key}} \cup \{i\}$.

$\mathsf{Sig}(i, m, \mathcal{R})$**:** If $i \neq j$, compute the ring signature honestly. If $i = j$, query $m \| \mathcal{R}$ to the signing oracle of $\mathcal{C}$ and obtain $\delta \leftarrow \mathcal{C}.\mathsf{Sign}(m \| \mathcal{R})$. Compute

$$(\mathsf{Acc}_{\mathcal{R}}, \mathsf{aux}) \leftarrow \mathsf{Acc.Eval}(\mathsf{pp}, \mathcal{R})$$

and obtain the corresponding auxiliary value r such that

$$\mathsf{Acc}_{\mathcal{R}} = P^r \cdot \prod_{M_{k1} \in \mathcal{R}} M_{k1}.$$

Since the reduction does not know sk_j, it uses the simulator of the NIZK system to generate a simulated proof π for the corresponding public statement. Finally, set $Q^{\mathsf{Sig}} := Q^{\mathsf{Sig}} \cup \{(i, m, \mathcal{R})\}$ and return $\sigma = (r, \pi)$.

By the zero-knowledge property of the proof system, the simulated proofs are computationally indistinguishable from honestly generated proofs. Hence Games 0 and 1 are computationally indistinguishable. The guess of the challenge index j affects only the success probability of the reduction and introduces a factor $1/q$.

Game 2 (Extracting from the forgery). Suppose that $\mathcal{A}$ outputs a valid forgery

$$(m^*, \sigma^*, \mathcal{R}^*), \qquad \sigma^* = (r^*, \pi^*).$$

Since the forgery is fresh, π^* is a new accepting proof. By simulation extractability of the NIZK system, except with negligible probability, the reduction extracts

$$(\mathsf{sk}^*, W_1^*, \hat{W}_2^*, W_3^*, \delta^*)$$

such that

$$e(P^{\mathsf{sk}^*}, H(m^* \| \mathcal{R}^*)) = e(P, \delta^*),$$

$$e(\mathsf{Acc}_{\mathcal{R}^*}, \hat{W}_2^*) = e(W_1^*, \hat{P})\, e(P^{\mathsf{sk}^*}, \hat{B}_{q+1}),$$

and

$$e(P^{\mathsf{sk}^*}, \hat{B}_{2q}) = e(W_3^*, \hat{P}).$$

Moreover, validity of the ring signature implies

$$\mathsf{Acc}_{\mathcal{R}^*} = P^{r^*} \cdot \prod_{M_{k1} \in \mathcal{R}^*} M_{k1}.$$

Game 3 (Reduction to accumulator security). Define the extracted public-key component $M_0^* := P^{\mathsf{sk}^*}$. The last two extracted relations,

$$e(\mathsf{Acc}_{\mathcal{R}^*}, \hat{W}_2^*) = e(W_1^*, \hat{P})\, e(M_0^*, \hat{B}_{q+1}),$$

and

$$e(M_0^*, \hat{B}_{2q}) = e(W_3^*, \hat{P}),$$

are exactly the accumulator-witness relations of Theorem 3 for the special case $\rho = 1$. We define the event Bad-Acc to occur if the extracted value M_0^* does not correspond to any member of $\mathcal{R}^*$. In this case,

$$\big(M_0^*, (W_1^*, \hat{W}_2^*, W_3^*)\big)$$

constitutes an additional valid value–witness pair for the same accumulator $\mathsf{Acc}_{\mathcal{R}^*}$ with $\rho = 1$.

Together with the valid value–witness pairs corresponding to the accumulated ring elements, including the padding elements when $|\mathcal{R}^*| < q$, this yields an additional valid opening for a value outside the accumulated set and therefore violates Theorem 3. Hence

$$\Pr[\text{Bad-Acc}] \le \epsilon_{\mathsf{Acc}}(\lambda)$$

for negligible ϵ_{Acc}. Consequently, conditioned on $\neg$Bad-Acc, the extracted value $M_0^* = P^{\mathsf{sk}^*}$ corresponds to the public-key component of some member $\mathsf{pk}_k \in \mathcal{R}^*$.

Game 4 (Reduction to $\mathsf{BLS}_{\mathsf{Acc}}$). Conditioned on successful extraction and on $\neg$Bad-Acc, let $\mathsf{pk}_k \in \mathcal{R}^*$ be the public key corresponding to the extracted secret key sk^*. The first extracted relation gives

$$e(\mathsf{pk}_k, H(m^* \| \mathcal{R}^*)) = e(P, \delta^*),$$

and hence δ^* is a valid $\mathsf{BLS}_{\mathsf{Acc}}$ signature on $m^* \| \mathcal{R}^*$ under pk_k. If $k = j$, then by freshness of the ring-signature forgery we have

$$(j, m^*, \mathcal{R}^*) \notin Q^{\mathsf{Sig}}.$$

Hence $m^* \| \mathcal{R}^*$ was never submitted to the $\mathsf{BLS}_{\mathsf{Acc}}$ signing oracle for the challenge key $t\mathsf{pk}_j$. Therefore, the reduction outputs

$$(m^* \| \mathcal{R}^*, \delta^*)$$

as an EUF-CMA forgery against $\mathsf{BLS}_{\mathsf{Acc}}$. Since j is chosen uniformly from $[q]$, the event $k = j$ occurs with probability $1/q$. Thus, up to the negligible losses from zero knowledge, simulation extractability, and accumulator security, the reduction succeeds with a polynomial loss of at most a factor q. Therefore, the ring-signature scheme is unforgeable.

Anonymity. We prove anonymity by showing that signatures generated by any two eligible members of the same ring are indistinguishable. Let $i_0, i_1 \in [q]$ be the two challenge users, with $\{\mathsf{pk}_{i_0}, \mathsf{pk}_{i_1}\} \subseteq \mathcal{R}$.

Game $\mathbf{A}_0$. This is the real anonymity experiment in which the challenge signature is generated using sk_{i_0}.

Game $\mathbf{A}_1$. As in Game A_0, except that the proof π in the challenge signature is replaced by a simulated proof. By the zero-knowledge property of the NIZK system,

$$\mathsf{Game}\ \mathrm{A}_0 \approx \mathsf{Game}\ \mathrm{A}_1.$$

Observe that all signer-dependent values, namely

$$(\mathsf{sk}_i, W_1', \hat{W}_2', W_3', \delta),$$

occur only as witnesses to π. In contrast, the public statement, including $\mathcal{R}$, $\mathsf{Acc}_{\mathcal{R}}$, and r, depends only on the ring and is identical for both challenge users. Consequently, once π is simulated, the resulting challenge signature is independent of whether the witness corresponds to i_0 or i_1.

Game $\mathbf{A}_2$. We reinterpret the simulated challenge proof as corresponding to signer i_1 instead of signer i_0. This does not change the distribution: the simulator generates π solely from the public statement, which is identical for both signers, and does not use either witness. Hence, Games A_1 and A_2 are identically distributed.

Game $\mathbf{A}_3$. Finally, replace the simulated proof with an honestly generated proof using the witness corresponding to i_1. By zero knowledge, $\mathsf{Game}\ \mathrm{A}_2 \approx \mathsf{Game}\ \mathrm{A}_3$. Game A_3 is exactly the real anonymity experiment in which the challenge signature is generated using sk_{i_1}. Therefore,

$$\left| \Pr[b' = b] - \frac{1}{2} \right| \leq \epsilon(\lambda),$$

for negligible ϵ, and hence the scheme is anonymous.