On the Impossibility of Algebraic Vector Commitments in Pairing-Free Groups

Dario Catalano[1 [0000−0001−9677−944X]], Dario Fiore[2 [0000−0001−7274−6600]], Rosario Gennaro[3 [0000−0002−3297−3750]], and Emanuele Giunta[2,4 [0000−0001−5294−6648]]

[1] University of Catania, Italy.
catalano@dmi.unict.it
[2] IMDEA Software Institute, Madrid, Spain.
{dario.fiore, emanuele.giunta}@imdea.org
[3] Protocol Labs.
rosario.gennaro@protocol.ai
[4] Universidad Politecnica de Madrid, Spain.

Abstract. Vector Commitments allow one to (concisely) commit to a vector of messages so that one can later (concisely) open the commitment at selected locations. In the state of the art of vector commitments, *algebraic* constructions have emerged as a particularly useful class, as they enable advanced properties, such as stateless updates, subvector openings and aggregation, that are for example unknown in Merkle-tree-based schemes. In spite of their popularity, algebraic vector commitments remain poorly understood objects. In particular, no construction in standard prime order groups (without pairing) is known.

In this paper, we shed light on this state of affairs by showing that a large class of concise algebraic vector commitments in pairing-free, prime order groups are impossible to realize.

Our results also preclude any cryptographic primitive that implies the algebraic vector commitments we rule out, as special cases. This means that we also show the impossibility, for instance, of succinct polynomial commitments and functional commitments (for all classes of functions including linear forms) in pairing-free groups of prime order.

1 Introduction

Vector commitments [27, 9] (VC) are a class of commitment schemes that allow a sender to commit to a vector $\mathbf{v}$ of n messages, in such a way that she can later open the commitment at selected positions. Namely, the sender can convince anyone that the i-th message in the committed vector is v_i. A secure scheme shall satisfy *position binding*, i.e. generating valid openings to different values $v_i \neq v_i'$ for the same position i is computationally infeasible.

The distinguishing feature of vector commitments is that commitments and openings must be *succinct*. In the original notion of [27, 9], this means that their size is independent of n, the length of the vector, but a relaxed notion allowing a logarithmic dependence in n may be considered, as in the case of the celebrated Merkle tree construction [29].

Mainly thanks to their succinctness property, vector commitments have been shown to be a useful building block in several applications, such as zero-knowledge sets [31, 27, 9], verifiable databases [3, 9], succinct arguments [22, 30, 4, 25], proofs of retrievability [20, 12], and stateless blockchains [10, 4].

Analyzing the state of the art of VC schemes, we see that VC constructions are based on two main approaches.

On one side, we have tree-based VCs, notably Merkle trees [29] and their generalizations [24]. These constructions have the advantage of being realizable from collision resistant hash functions, and thus can be based on the hardness of virtually any cryptographic problem including factoring, discrete logarithm, SIS and many more. In fact, we notice that VCs with logarithmic-size openings are *equivalent* to collision-resistant hash functions. The main drawback of tree-based schemes is that their openings are of size $O(\log n)$. Additionally, the tree-based approach seems to inherently impede the realization of properties such as subvector openings [4, 25] and aggregation [8], that turn useful in both theoretical and practical applications of VCs.

On the other side, we have *algebraic* vector commitments, notably based on bilinear pairings [27, 21, 9], groups of unknown order [9], and lattices [34, 35]. Roughly speaking, an algebraic VC is one in which the commitment and verification algorithm only use algebraic operations over the group that underlies the construction (this rules out hashing group elements for example). The main advantage of these constructions is that they admit openings of constant size,[5] that are virtually optimal – a single group element in most constructions. Moreover, algebraic schemes naturally achieve useful properties such as (additive) homomorphism, stateless updatability [9], subvector openings [4, 25] and aggregation [8]. Yet, the powerful versatility of existing VCs with constant-size openings contrasts with the limited theoretical understanding of their foundations.

We see two main open questions related to algebraic VCs. The first one concerns the minimal general assumption that implies them. While tree-based schemes with logarithmic openings are well understood, being de facto equivalent to collision-resistant hash functions[6], we have no generic recipe to build algebraic VCs with constant-size openings.[7] The second question is whether algebraic VCs can be built from "standard" prime-order groups without pairings. In this setting, known constructions rely either on the tree-based approach (e.g., building a Merkle tree on top of Pedersen hash function), or on inner-product arguments in the random oracle model [6, 7]. Both these approaches entail logarithmic-size openings and a non-algebraic verification.

[5] We include lattice-based schemes in the 'algebraic' category although they do not perfectly fit our notion of using a group in a black box way; also, existing schemes still need (poly) logarithmic-size openings.

[6] A Merkle tree is a VC with logarithm openings that can be realized from any CRHF. Conversely, in any non trivial VC the commitment procedure has to be shrinking and collision resistant, from which CRHF can be built.

[7] The only generic construction with constant size opening is the folklore one that combines a hash function and a constant-size SNARK; yet this is non-algebraic due to the need of encoding the hash computation in the SNARK's constraint system.

We believe that settling these two questions would improve our understanding of vector commitments. In this work, we focus on the second question for two important reasons: (i) on the theoretical front, studying algebraic VCs in this minimal setting helps us understand conceptually what are the "ingredients" needed to build them; (ii) on the practical side, pairing-free groups of known order are the simplest and most efficient cryptographic setting, and yet we know of no construction of algebraic VCs there.

Our results are negative: we show that a broad class of VC schemes in this setting cannot both be succinct and satisfy position binding.

1.1 Our Results

We informally call a vector commitment built on top of a group $\mathbb{G}$ of prime order q "algebraic" if all its procedures use $\mathbb{G}$ in a black box way, i.e. without relying on the representation of group elements. We show the following two main results.

Impossibility of algebraic VCs with linear verification. We start by looking at the class of algebraic VC schemes in which the verification algorithm is a set of linear equations over $\mathbb{G}$. Specifically, for a message m and position i the verification consists of checking that

$$A(\mathbf{z}, m, i) \cdot \mathbf{X} \stackrel{?}{=} B(\mathbf{z}, m, i) \cdot \mathbf{Y} \tag{1}$$

where $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2)$ are the group elements appearing respectively in the public parameters and the commitment, openings are of the form $(\mathbf{Y}, \mathbf{z})$ with $\mathbf{Y}$ being a vector of group elements and $\mathbf{z}$ of field elements, and A, B are functions defining matrices with coefficients in $\mathbb{F}_q$.

We believe this to be the simplest and most natural form of verification using only group operations. However we show that whenever A depends affinely on $\mathbf{z}, m$ and B is independent from them (we say such a scheme has *strictly* linear verification), then it is impossible to achieve both position binding and succinctness. More specifically we prove that if a scheme has position binding, commitments of bit-length ℓ_c and opening proofs of bit-length ℓ_π, then asymptotically their product is lower bounded by the length of the vector we are committing to, i.e. $\ell_c \cdot \ell_\pi = \Omega(n)$. Thus either $\ell_c = \Omega(\sqrt{n})$ or $\ell_\pi = \Omega(\sqrt{n})$. Interestingly, this family of schemes captures generalizations of Pedersen commitments [2] which, as we show in the full version, achieve this lower bound.

Next, we investigate how crucial are our requirements on the dependence of $A(\cdot)$ and $B(\cdot)$ on $\mathbf{z}, m$. We show they are necessary. Indeed, if we allow either A to depend quadratically, or B affinely, on $\mathbf{z}, m$ then there exist succinct VC constructions whose verification can be written in the above form over a group $\mathbb{G}$. We provide examples in the full version. The schemes we find however rely on arithmetization techniques to encode arbitrary circuits as constraint systems of degree 2 over a finite field [13]. This for instance means that, for proper choice of A and B, it is possible to express, using an algebraic verification equation as

(1), computations like the validity tests for a Merkle tree path, or any arbitrary VC verification algorithm.

Despite being secure and succinct, VC schemes built this way do not satisfactorily answer our question in a positive way, as they appear to bypass the underlying group as their source of hardness. Indeed, either their security comes from problems unrelated to $\mathbb{G}$, or if they depend on $\mathbb{G}$, they must do it in a non-black-box way[8].

Impossibility of algebraic VCs with generic group verification. Motivated by these findings, we investigate whether VCs can be built given *only* black-box access to a cryptographic group. To study this case, we just assume the VC (which we call *algebraic with generic verification*) to use the underlying group generically, without any further constraint on its verification procedure.

Eventually we provide a black-box separation in Maurer's Generic Group Model [28]. This informally implies that any VC using $\mathbb{G}$ generically and whose position binding reduces to a hard problem in $\mathbb{G}$ (such as DLP or CDH) cannot be succinct, as it must hold $\ell_c \cdot \ell_\pi = \Omega(n)$.

1.2 Our Techniques

Our strategy to prove our impossibility results on algebraic vector commitments consists of two main steps. (A) We show that from a VC it is possible to construct a class of signature schemes. In particular, if the VC is algebraic with linear (resp. generic) verification, the resulting signature scheme's verification has analogous algebraic properties. (B) We prove the insecurity of this class of signature schemes in pairing-free groups of known order. To achieve the latter result we build on, and extend, the recent techniques of [11], that provide negative results for a somewhat smaller family of algebraic signatures.

In what follows we give an overview on each step.

From VCs to Signatures. Given a VC scheme for vectors of length n our transformation produces a signature scheme with polynomially bounded message space $\{1, \ldots, n\}$. In a nutshell, the public key is a commitment c to a vector of n random values $(s_1, \ldots, s_n)$. The signature on the message $i \in [n]$ is the pair (s_i, π_i) where π_i is the VC opening proof that c opens to s_i at position i. Verification simply runs the VC verification algorithm to check that the opening is valid.

Conveniently, this transformation maps algebraic VCs with linear/generic verification to signature schemes with the analogous property, which we then call algebraic signatures with linear/generic verification. This happens since the verification algorithm is essentially the same in both primitives.

The resulting signature however may not be proved existentially unforgeable if it comes from a VC satisfying only position binding. Indeed the latter property does not imply that every opening proof is hard to compute. However, assuming

[8] For example, one may consider a Merkle-tree of Pedersen commitments which must use the group representation to go from one level to another.

that the scheme is also succinct, an adversary who produces 'many' correct openings should have to correctly guess the value of several messages s_i used to generate the commitment. This can be shown to be information-theoretically hard if the commitment and opening proofs provided have significantly smaller bit-length than the min-entropy of those messages.

For this reason we introduce a relaxed security notion, called ϑ-unforgeability, where an adversary must provide not only one but at least more than ϑ-many[9] forgeries for non-queried messages. Setting ϑ as a proper function of the number of queries made by the adversary, we prove that signatures from VCs are ϑ-unforgeable.

Impossibility of Algebraic Signatures, revisited. To conclude our impossibility result for VCs, we finally provide an impossibility result and a black-box separation for algebraic signatures with strictly linear and generic verification respectively. In particular, we show in both cases that the message space in a ϑ-unforgeable construction is upper-bounded by $n + \vartheta$ with n being the number of group elements in the verification key. We also show this to be tight by providing a construction that achieves this bound in the full version.

Notice that similar results were already proved in [11]. In their work signatures are assumed to be of the form $(\mathbf{Y}, t)$ with $\mathbf{Y}$ a vector of group elements and $t \in \{0,1\}^\kappa$. Moreover the verification procedure is assumed to consist of a linear check as in Equation 1. For this class of signatures, which can be shown equivalent to our notion of *algebraic with linear verification*, they provide an attack running in time $O(2^\kappa \cdot \mathsf{poly}(\lambda))$.

Thus their adversary is efficient only when $t = O(\log \lambda)$, whereas our impossibility result applies to schemes with strictly linear verification, where signatures may contain several field elements. Likewise, their black-box separation only captures schemes with *linear verification*, while we extend it to signatures where all procedures are simply required to be generic. To show that this class of schemes is indeed more general we provide examples in the full version.

We finally stress that, as in [11], our results hold in Maurer's Generic Group Model [28]. For a comparison with other models of generic computation, such as Shoup's Generic Group Model [40], we refer to the discussion in [42].

1.3 Interpretation of our impossibility and further implications

As mentioned earlier, both our impossibility results specify precise bounds and conditions under which VCs cannot be built generically in pairing-free groups. The bottom line is that, whenever a position-binding VC scheme uses the group in a black box way (and relies on it for security), then it cannot be succinct, which we recall is the distinguishing feature of this primitive.

Another interesting aspect of our impossibility results is that they imply analogous impossibilities for any primitive that allows one to construct algebraic VCs (with either strictly linear or generic-group verification) in pairing-free groups. Notably, our impossibility applies to polynomial commitments [21],

[9] Where ϑ may depend on the public parameters as well as the number of queries.

and functional commitments [26] supporting any class of functions that includes projections, i.e., $C_i(\mathbf{v}) = v_i$ (already captured by linear forms). Indeed, each of these primitives allows one to build a VC with exactly the same succinctness and type of verification.[10] Therefore we obtain that any secure functional commitment or polynomial commitment using a pairing-free group in a black-box way cannot be succinct (or, more precisely, they must satisfy $\ell_c \cdot \ell_\pi = \Omega(n)$).

Our impossibility for algebraic signatures instead can be shown to imply analogous results for verifiable random functions [32] and identity-based encryption [39, 5], the latter through the Naor-trick reduction, as observed in [11]. In this way our black-box separation for signatures yield a simpler argument for the tight result in [38].

An interesting question left open by our work is understanding if our results can imply the impossibility of further cryptographic primitives via a connection to the classes of algebraic signatures and vector commitments that we rule out. Another open question concerns the minimal assumptions required to describe a VC with constant-size commitment and openings. We notice that our impossibility for VCs with generic verification holds in Maurer's generic group model [28]. When using Shoup's GGM [40], our results may not hold as one could use the group oracle as a random oracle [43], e.g., to build a Merkle tree of Pedersen hashes (see a similar discussion for signatures in [11]). However, to the best of our knowledge all these techniques would in the best case lead to schemes with logarithmic-size openings.

1.4 Related Work

The study of impossibility results about the construction of cryptographic primitives in restricted models is an important area of research that provides insights on the foundations of a cryptographic problem. Starting with the seminal paper of Impagliazzo and Rudich [19], a line of works study the (in)feasibility of constructing cryptographic primitives in a black-box way from general assumptions, such as one-way functions or trapdoor permutations (e.g. [41, 23, 15, 16, 17, 14]).

Another line of works (more closely related to ours), initiated by Papakonstantinou, Rackoff and Vahlis [33], considers the problem of proving impossibility of cryptographic primitives that make black-box use of a cryptographic group without pairings. Specifically, [33] prove that identity-based encryption (IBE) algorithms built in this model of computation cannot be secure. Following [33], more recent works study the impossibility, in generic group models for pairing-free groups of known order, of other cryptographic primitives, such as verifiable delay functions [36], identity-based encryption (with a result tighter than [33]) [38] and signature schemes [11]. In addition to proving impossibility for algebraic signatures with generic-group algorithms, [11] also prove the generic impossibility of a class of algebraic signatures whose verification is a system of linear equations over a group.

[10] These constructions are trivial/folklore and we do not elaborate further on them.

In [37], Schul-Ganz and Segev prove a lower bound on the number of group operations needed to verify batch membership proofs in accumulators that make black-box use of a cryptographic group. Their lower bound applies analogously to the verification of subvector openings in vector commitments. Despite the result and the techniques of [37] differ from ours, both [37] and our work show certain limitations of constructing VCs in prime order groups.

Finally, we mention the work of Abe, Haralambiev and Ohkubo [1] that also considers a question related to constructing vector commitments. Following a research line on structure-preserving cryptography, Abe et al. [1] investigate if it is possible to construct commitment schemes in bilinear groups in which messages, keys, commitments, and decommitments are elements of bilinear groups, and whose openings are verified by pairing product equations. For this class of schemes, they prove that the commitment cannot be shrinking. Implicitly this result also implies the impossibility of constructing succinct vector commitments in this structure-preserving setting in bilinear groups.

1.5 Organization of the paper

In Section 3 we define algebraic VCs and show our transformation to ϑ-unforgeable signatures. Section 4 presents the definition of algebraic signatures and our impossibility results for strictly linear verification and generic group verification. Finally, in Section 5 we illustrate how to relate the parameters of our VC-to-signatures transformation with those needed by the impossibility of algebraic signatures.

2 Preliminaries

Notation. We denote the security parameter by λ and negligible functions with $\mathsf{negl}(\lambda)$. We say that an algorithm is PPT if it runs in probabilistic polynomial time. For a positive integer n, $[n]$ denotes the set $\{1, \ldots, n\}$. We use $(\mathbb{G}, +)$ to denote a group of known prime order q with canonical generator G, and $\mathbb{F}_q$ for the field of order q. The identity (or zero) element is denoted as $0 \in \mathbb{G}$. Given a vector $\mathbf{x} \in \mathbb{F}_q^n$, we denote $\mathbf{x} \cdot G = (x_1 G, \ldots, x_n G)$.

$\mathbb{F}_q^{n,m}$ is the space of matrices A with m columns and n rows and entries in $\mathbb{F}_q$. $\operatorname{rk} A$ denotes the rank of A, i.e. the maximum number of linearly independent rows. $A^\top$ is the transposed of A. All $\mathbf{x} \in \mathbb{F}_q^n$ are assumed to be column vectors, whereas row vectors are denoted as $\mathbf{x}^\top$.

In what follows 'GGM' stands for Maurer's Generic Group Model [28] for a group of known prime order q. This model can be defined through two stateful oracles $\mathcal{O}_{\mathsf{add}}$ and $\mathcal{O}^0_{\mathsf{eq}}$ such that: group element are labeled with progressively increasing indices, the first being associated to the canonical generator G, $\mathcal{O}_{\mathsf{add}}(X, Y)$ associate the next index to the element $X + Y$ and $\mathcal{O}^0_{\mathsf{eq}}(X)$ returns 1 if X equals the identity element, 0 otherwise.

2.1 Vector Commitments

We recall the definition of vector commitments from [9].

Definition 1 (VC). *A Vector Commitment scheme is a tuple of algorithms* $(\mathsf{VC.Setup}, \mathsf{VC.Com}, \mathsf{VC.Open}, \mathsf{VC.Vfy})$ *and a message space* $\mathsf{VC.M}$ *such that*

- $\mathsf{VC.Setup}(1^\lambda) \xrightarrow{\$} \mathsf{pp}$ *generates the public parameters.*
- $\mathsf{VC.Com}(\mathsf{pp}, m_1, \ldots, m_n) \xrightarrow{\$} c, \mathsf{aux}$ *produce a commitment to* $m_1, \ldots, m_n \in \mathsf{VC.M}$ *together with some auxiliary information.*
- $\mathsf{VC.Open}(\mathsf{pp}, m, i, \mathsf{aux}) \xrightarrow{\$} \pi$ *return an opening proof that the* i*-th entry of a given commitment is* m_i*.*
- $\mathsf{VC.Vfy}(\mathsf{pp}, c, m, i, \pi) \to 0/1$ *verifies the opening proof's correctness.*

We require a vector commitment scheme to satisfy *perfect correctness*, that is, given public parameters $\mathsf{pp} \leftarrow^{\$} \mathsf{VC.Setup}(1^\lambda)$, commitment $c, \mathsf{aux} \leftarrow^{\$} \mathsf{VC.Com}(\mathsf{pp}, m_1, \ldots, m_n)$ for any $m_i \in \mathsf{VC.M}$, and opening $\pi \leftarrow^{\$} \mathsf{VC.Open}(\mathsf{pp}, m_i, i, \mathsf{aux})$, it holds

$$\Pr\left[\mathsf{VC.Vfy}(\mathsf{pp}, c, m, i, \pi) \to 1\right] = 1$$

Moreover, to avoid trivial cases, in this paper we assume $|\mathsf{VC.M}| \geq 2$.

The main security property for a vector commitments is the so called *position binding*, which informally states that no adversary can open the same position of a given commitment to two different values. Formally

Definition 2 (Position binding). *A vector commitment scheme satisfies position binding if for any* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* $\varepsilon(\lambda)$ *such that*

$$\Pr\left[\begin{array}{l}\mathsf{VC.Vfy}(\mathsf{pp}, c, m, i, \pi) \to 1 \\ \mathsf{VC.Vfy}(\mathsf{pp}, c, m', i, \pi') \to 1 \\ m \neq m'\end{array} \middle| \begin{array}{l}\mathsf{pp} \leftarrow^{\$} \mathsf{VC.Setup}(1^\lambda) \\ \mathcal{A}(\mathsf{pp}) \to (c, m, m', i, \pi, \pi')\end{array}\right] \leq \varepsilon(\lambda).$$

The property that distinguishes VCs from classical binding commitments is *succinctness* Following [27, 9], a VC scheme is said succinct if there is a fixed $p(\lambda) = \mathsf{poly}(\lambda)$ such that for any n the size of honestly generated commitments and openings is bounded by $p(\lambda)$. One may also consider weaker notions where the size may be bounded by $p(\lambda) \log n$ or $p(\lambda, \log n)$.

Since in our work we are interested in understanding the feasibility of VCs based on their level of succinctness, we consider a parametric notion. We say that a VC has succinctness (ℓ_c, ℓ_π) if for any $m_1, \ldots, m_n \in \mathsf{VC.M}$, commitment $c, \mathsf{aux} \leftarrow^{\$} \mathsf{VC.Com}(\mathsf{pp}, m_1, \ldots, m_n)$ and opening $\pi \leftarrow^{\$} \mathsf{VC.Open}(\mathsf{pp}, m_i, i, \mathsf{aux})$ for any $i \in [n]$, we have that c (resp. π) has bit-length $\ell_c(\lambda, n)$ (resp. $\ell_\pi(\lambda, n)$).

2.2 Digital Signatures

Definition 3. *A signature scheme is a tuple of* PPT *algorithms* $(\mathsf{S.Setup}, \mathsf{S.Sign}, \mathsf{S.Vfy})$ *and a message space set* $\mathsf{S.M}$ *such that*

- $\mathsf{S.Setup}(1^\lambda) \xrightarrow{\$} (\mathsf{sk}, \mathsf{vk})$ *generates the secret and verification keys*
- $\mathsf{S.Sign}(\mathsf{sk}, m) \xrightarrow{\$} \sigma$ *returns the signature of a message* $m \in \mathsf{S.M}$
- $\mathsf{S.Vfy}(\mathsf{vk}, m, \sigma) \to 0/1$ *verifies the signature* σ *for a message* $m \in \mathsf{S.M}$

We further require a signature scheme to satisfy *perfect correctness*, meaning that if $(\mathsf{sk}, \mathsf{vk}) \xleftarrow{\$} \mathsf{S.Setup}(1^\lambda)$ and $\sigma \xleftarrow{\$} \mathsf{S.Sign}(\mathsf{sk}, m)$ for any $m \in \mathsf{S.M}$ then the verification algorithm accepts always, i.e.

$$\Pr\left[\mathsf{S.Vfy}(\mathsf{vk}, m, \sigma) \to 1\right] = 1.$$

3 Algebraic Vector Commitments

In this paper we focus on vector commitments built on a pairing-free group of known order, using it in a black box way. We start by introducing a notion of algebraic vector commitments where the verification algorithm only consists of a system of linear equations.

Definition 4 (Algebraic VCs with linear verification). *A vector commitment scheme is said to be* algebraic with linear verification *if the message space is* $\mathsf{VC.M} = \mathbb{F}_q$ *and*

- $\mathsf{VC.Setup}(1^\lambda) \xrightarrow{\$} \mathsf{pp}$ *such that* $\mathsf{pp} = (\mathbf{X}_1, s_1) \in \mathbb{G}^{\nu_1} \times \{0,1\}^*$.
- $\mathsf{VC.Com}(\mathsf{pp}, m_1, \ldots, m_n) \xrightarrow{\$} c, \mathsf{aux}$ *such that* $c = (\mathbf{X}_2, s_2) \in \mathbb{G}^{\nu_2} \times \{0,1\}^*$.
- $\mathsf{VC.Open}(\mathsf{pp}, m, i, \mathsf{aux}) \to \pi$ *such that* $\pi = (\mathbf{Y}, \mathbf{z})$ *with* $\mathbf{Y} \in \mathbb{G}^k$ *and* $\mathbf{z} \in \mathbb{F}_q^h$.
- *There exist* $A : \mathbb{F}_q^{h+1} \times [n] \times \{0,1\}^* \to \mathbb{F}_q^{\ell,n}$ *and* $B : \mathbb{F}_q^{h+1} \times [n] \times \{0,1\}^* \to \mathbb{F}_q^{\ell,k}$ *matrices such that* $\mathsf{VC.Vfy}(\mathsf{pp}, c, m, i, \pi) \to 1$ *if and only if, calling* $\mathbf{X} = \mathbf{X}_1 || \mathbf{X}_2$ *and* $s = s_1 || s_2$

$$A(\mathbf{z}, m, i, s) \cdot \mathbf{X} = B(\mathbf{z}, m, i, s) \cdot \mathbf{Y}.$$

For the ease of presentation we will omit s in A and B when clear from the context. Notice that the definition imposes linearity only with respect to group elements while it allows procedures A, B to depend non-linearly on the field vector element $\mathbf{z}$.

As we shall see, our first impossibility result states that whenever A is an affine function of $\mathbf{z}, m$ and B does not depends on $\mathbf{z}, m$, then the resulting scheme cannot be both "succinct" and position binding. We call these schemes *strictly* linear since their verification equations depend linearly both in $\mathbf{z}$ and $\mathbf{Y}$.

Definition 5 (Algebraic VCs with strictly linear verification). *A vector commitment is said to be algebraic with strictly linear verification if it satisfies Definition 4,* $A(\mathbf{z}, m, i)$ *is an affine function*[11] *of* $\mathbf{z}, m$ *and* $B(i)$ *does not depends on* $\mathbf{z}, m$.

[11] i.e. $A(\mathbf{z}, m, i) = A_0(i) + z_1 A_1(i) + \ldots + z_h A_h(i) + m A_{h+1}(i)$

However, if we allow A to depend quadratically, or B linearly, on $\mathbf{z}, m$ then we could use arithmetization techniques, such as R1CS, to encode a circuit representing for example a Merkle tree verification into the verification equation of Definition 4. This means that we can construct algebraic VC schemes with linear verification that are succinct and position binding. Explicit examples of such schemes are provided in the full version.

This technique however either bypasses the underlying group and may reduce security to external problems, or rely on non-black-box usage of the group. An example of the latter comes by encoding a Merkle tree built using an hash function whose collision resistance is based on discrete logarithm over the same group $\mathbb{G}$, such as Pedersen hash. Note that this construction would not retain algebraic properties from the underlying group. For this reason, following an approach similar to [33, 11], we study whether in the Generic Group Model (GGM) the security of a VC can be reduced to hard problems on the underlying group. To this aim we provide the following more general definition.

Definition 6 (Algebraic VCs with generic verification). *A vector commitment scheme is said to be* algebraic with generic verification *if, in the GGM, the algorithms* $\mathsf{VC.Setup}, \mathsf{VC.Com}, \mathsf{VC.Open}, \mathsf{VC.Vfy}$ *are oracle machines with access to* $\mathcal{O}_{\mathsf{add}}$ *and* $\mathcal{O}^0_{\mathsf{eq}}$.

3.1 Generic Transformation from VCs to Signatures

The strategy we adopt to show our impossibility results is to establish a connection between vector commitments and signatures, providing a way to construct the latter from the former generically. This way we will be able to bridge extensions of the impossibility results in [11] for algebraic signatures to algebraic vector commitments.

More specifically, for a given VC (not necessarily algebraic) our transformation produces a signature scheme with polynomially bounded message space $\{1, \ldots, n\}$. The high-level idea is to compute a commitment c to random messages $m_1, \ldots, m_n$, and use (pp, c) as the verification key and the auxiliary information aux as the secret key. In order to sign a message $i \in \{1, \ldots, n\}$, the signer returns m_i and π, the message and opening proof for the i-th position, while verification is performed by checking the correctness of π. A formal description of the transformation is presented in Fig. 1.

3.2 ϑ-Unforgeability

In terms of security the transformation in Fig. 1 fails in general to realize a UF-CMA-secure signature scheme. Informally, the problem is that position binding and succinctness do not imply, per se, that every opening proof is hard to compute, after having seen other openings. Indeed the latter property could be easily violated, for example, by a VC where $\mathsf{VC.Open}$ attaches to every opening the proof (m_1, π_1) for position 1. Notice that one could modify any VC to do so without violating succinctness nor position binding. Yet starting from such a

$\mathsf{S}_{\mathsf{VC}}.\mathsf{Setup}(1^\lambda)$:

1 : $\mathsf{VC.Setup}(1^\lambda) \to \mathsf{pp}$
2 : $m_1, \ldots, m_n \leftarrow^{\$} \mathsf{VC.M}$
3 : $c, \mathsf{aux} \leftarrow^{\$} \mathsf{VC.Com}(\mathsf{pp}, m_1, \ldots, m_n)$
4 : $\mathsf{vk} \leftarrow (\mathsf{pp}, c)$ $\mathsf{sk} \leftarrow (\mathsf{aux}, \{m_i\}_{i=1}^n)$
5 : Return vk, sk

$\mathsf{S}_{\mathsf{VC}}.\mathsf{Sign}(\mathsf{sk}, i)$:

1 : Parse $\mathsf{sk} = (\mathsf{aux}, \{m_i\}_{i=1}^n)$
2 : $\pi \leftarrow \mathsf{VC.Open}(\mathsf{pp}, m_i, i, \mathsf{aux})$
3 : $\sigma \leftarrow (m_i, \pi)$
4 : Return σ

$\mathsf{S}_{\mathsf{VC}}.\mathsf{Vfy}(\mathsf{vk}, i, \sigma)$:

1 : Parse $\mathsf{vk} = (\mathsf{pp}, c)$ and $\sigma = (m_i, \pi)$. Return $\mathsf{VC.Vfy}(\mathsf{pp}, c, m_i, i, \pi)$

Fig. 1. Generic transformation from VCs to signature schemes

VC would allow an adversary to easily forge a signature for message 1 in the scheme in Fig. 1.

Observe that, informally, if the VC scheme were *hiding*, meaning that no information about messages in unopened positions is leaked, and $|\mathsf{VC.M}|$ is large enough, then the associated signature would be secure, since an adversary would have to guess the right message in the i-th position. This intuition can be extended to general VC assuming that the scheme is *succinct*. Indeed, even though the commitment c or its openings π may leak information about unopened messages among $m_1, \ldots, m_n$, if their bit length is significantly smaller than n, no adversary can produce "too many" forgeries given only a few openings, as correctly guessing these message would be information-theoretically hard.

For this reason we introduce a relaxed notion of unforgeability for signatures, called ϑ-unforgeability, which is enough for our purposes. In a nutshell, it requires a winning adversary to produce at least ϑ forgeries on *distinct* messages, with ϑ being a function of the queries performed and the public parameters. Next, using the intuition above, we prove that signature schemes obtained through the transformation in Fig. 1 satisfy this weaker notion.

Definition 7 (ϑ-UF). *Given a function $\vartheta : \{0,1\}^* \to \mathbb{N}$ and a signature scheme we define the ϑ-Unforgeability Experiment as in Fig. 2. The advantage of an adversary $\mathcal{A}$ is defined as*

$$\mathsf{Adv}^{\vartheta\text{-}\mathsf{UF}}(\mathcal{A}) \;=\; \Pr\left[\mathsf{Exp}^{\vartheta\text{-}\mathsf{UF}}_{\mathcal{A}} = 1\right].$$

A scheme is ϑ-Unforgeable if any PPT *adversary has negligible advantage.*

To provide more intuition about this notion we observe that setting $\vartheta = 0$ yields the classic unforgeability under chosen message attacks (UF-CMA) [18] security definition. For higher values of ϑ we obtain progressively weaker definitions until $\vartheta(\mathsf{vk}, Q) = |\mathsf{S.M}|$, which is trivially true for any scheme. The notion

$\mathsf{Exp}^{\vartheta\text{-}\mathsf{UF}}_{\mathcal{A}}$ **with adversary** $\mathcal{A}$:

1 : Initialize $Q \leftarrow \varnothing$, generate $\mathsf{sk}, \mathsf{vk} \leftarrow^{\$} \mathsf{S.Setup}(1^\lambda)$ and send $\mathcal{A} \leftarrow \mathsf{vk}$
2 : **When** $\mathcal{A} \rightarrow m \in \mathsf{S.M}$:
3 : Sign $\sigma \leftarrow^{\$} \mathsf{S.Sign}(\mathsf{sk}, m)$, store $Q \leftarrow Q \cup (m, \sigma)$ and send $\mathcal{A} \leftarrow \sigma$
4 : **When** $\mathcal{A} \rightarrow F$:
5 : Return 1 if the following conditions are satisfied:
6 : For all $(m, \sigma) \in F$, the signature is correct, i.e. $\mathsf{S.Vfy}(\mathsf{vk}, m, \sigma) \rightarrow 1$
7 : Messages in F were not queried, i.e. $(m, \sigma) \in F \;\Rightarrow\; (m, \cdot) \notin Q$
8 : $|\{m \,:\, (m, \cdot) \in F\}| > \vartheta(\mathsf{vk}, Q)$
9 : Else return 0

Fig. 2. ϑ-Unforgeability Experiment for a given signature scheme

of t-time security is also captured by our definition setting

$$\vartheta(\mathsf{vk}, Q) \;=\; \begin{cases} 0 & \text{If } |Q| \leq t \\ |\mathsf{S.M}| & \text{If } |Q| > t \end{cases}$$

Finally we can show that a signature scheme obtained from a "succinct" VC satisfy this notion. A proof appears in the full version.

Theorem 1. *Given a Vector Commitment with commitments of bit-length $\ell_c = \ell_c(n, \lambda)$ and opening proofs of bit-length $\ell_\pi = \ell_\pi(n, \lambda)$, then there exists a* PPT *black box reduction $\mathcal{R}$ of ϑ-UF for the derived signature scheme described in Fig. 1 to the position binding property, where*

$$\vartheta(\mathsf{vk}, Q) \;=\; \frac{\lambda + \ell_c + |Q| \cdot (\ell_\pi + \log |\mathsf{VC.M}|)}{\log |\mathsf{VC.M}|}.$$

In particular for any position binding VC, the resulting signature is ϑ-UF with ϑ as specified above.

4 Algebraic Signatures

Having established a connection between VC and signatures we now provide the analogous of algebraic VC with (strictly) linear/generic verification in the signature setting. The first one is equivalent to the notion of *algebraic signature* in [11] and simply constrain the verification procedure to test a system of linear equations, albeit with a minor addition: as these signatures may come in our case from a VC, we split $\mathsf{S.Setup}$ in a CRS-generator $\mathsf{S.SetupCRS}$ which returns the public parameters (a list of group elements $\mathbf{X}_1$) and the actual key generation algorithm $\mathsf{S.SetupKey}(\mathbf{X}_1)$ which produces vk and sk. Note there is no loss of generality assuming this structure as $\mathsf{S.SetupCRS}$ may return an empty vector which could then be ignored by $\mathsf{S.SetupKey}$.

Definition 8. *A signature scheme* $(\mathsf{S.Setup}, \mathsf{S.Sign}, \mathsf{S.Vfy})$ *is said to be algebraic with linear verification if*

- $\mathsf{S.Setup}$ *is divided into two algorithms* $\mathsf{S.SetupCRS}$ *and* $\mathsf{S.SetupKey}$ *such that* $\mathsf{S.SetupCRS}(1^\lambda) \xrightarrow{\$} (\mathbf{X}_1, s_1) \in \mathbb{G}^{n_1}$ *and* $\mathsf{S.SetupKey}(1^\lambda, \mathbf{X}_1, s_1) \xrightarrow{\$} \mathsf{sk}, \mathsf{vk}$ *with*

$$\mathsf{vk} = (\mathbf{X}, s) \in \mathbb{G}^n \times \{0,1\}^* \quad : \quad \mathbf{X} = \mathbf{X}_1 || \mathbf{X}_2, \qquad \mathbf{X}_2 \in \mathbb{G}^{n_2}, \qquad s = s_1 || s_2.$$

- $\mathsf{S.Sign}(\mathsf{sk}, m) \xrightarrow{\$} \sigma$ *where* $\sigma = (\mathbf{Y}, \mathbf{z})$ *with* $\mathbf{Y} \in \mathbb{G}^k$ *and* $\mathbf{z} \in \mathbb{F}_q^h$.
- *There exist* $A : \mathbb{F}_q^h \times \mathsf{S.M} \times \{0,1\}^* \to \mathbb{F}_q^{\ell,n}$ *and* $B : \mathbb{F}_q^h \times \mathsf{S.M} \times \{0,1\}^* \to \mathbb{F}_q^{\ell,k}$ *matrices such that* $\mathsf{S.Vfy}(\mathsf{vk}, m, \sigma) \to 1$ *if and only if* $\sigma = (\mathbf{z}, \mathbf{Y})$ *and*

$$A(\mathbf{z}, m, s)\mathbf{X} \;=\; B(\mathbf{z}, m, s)\mathbf{Y}.$$

Furthermore the scheme is said to have strictly linear verification if $A(\mathbf{z}, m, s)$ *is an affine function of* $\mathbf{z}$ *and* $B(m, s)$ *does not depend on* $\mathbf{z}$.

When clear from the context we will omit for clarity the argument s in the matrices A, B above. Next we provide an analogous for algebraic vector commitments with generic verification. As in the previous definition we split the setup algorithm into a procedure that prepares the CRS and another one that uses the CRS, oblivious to any trapdoor information about it, to compute the secret and verification keys.

Definition 9. *A signature scheme* $(\mathsf{S.Setup}, \mathsf{S.Sign}, \mathsf{S.Vfy})$ *is said to be algebraic with generic verification if, in the GGM, all algorithms have access to* $\mathcal{O}_{\mathsf{add}}$ *and* $\mathcal{O}^0_{\mathsf{eq}}$. *Furthermore we require* $\mathsf{S.Setup}$ *to be divided into two algorithms* $\mathsf{S.SetupCRS}$ *and* $\mathsf{S.SetupKey}$ *such that* $\mathsf{S.SetupCRS}(1^\lambda) \xrightarrow{\$} (\mathbf{X}_1, s_1) \in \mathbb{G}^{n_1} \times \{0,1\}^*$ *and* $\mathsf{S.SetupKey}(1^\lambda, \mathbf{X}_1, s_1) \xrightarrow{\$} \mathsf{sk}, \mathsf{vk}$ *with*

$$\mathsf{vk} = (\mathbf{X}, s) \in \mathbb{G}^n \times \{0,1\}^* \quad : \quad \mathbf{X} = \mathbf{X}_1 || \mathbf{X}_2, \qquad \mathbf{X}_2 \in \mathbb{G}^{n_2}, \qquad s = s_1 || s_2.$$

4.1 Attack to Schemes with Strictly Linear Verification

We now provide an attack for algebraic signatures with strictly linear verification. The same notation of Definition 8 will be used below without further reference.

Theorem 2. *Given a signature scheme with strictly linear verification, for any* ϑ *polynomially bounded such that* $n_2 + \vartheta \leq |\mathsf{S.M}|$ *there exists a* PPT *algorithm* $\mathcal{A}$ *that in the unforgeability experiment in Fig. 2 performs at most* n_2 *queries and produces* ϑ *distinct forgeries with significant probability.*

Proof. For the sake of presentation we build $\mathcal{A}$ describing first a subroutine $\mathcal{B}$ which could break security by doing potentially more signing queries that n_2. Next, we show how $\mathcal{A}$ can use $\mathcal{B}$ in a black-box way to realize the full attack with n_2 queries.

Similarly to the attack described in [11], upon receiving the verification key $(\mathbf{X}, s)$, the subroutine $\mathcal{B}$ (described formally in Figure 3) keeps track of all possible exponents of $\mathbf{X}$ in an affine space $L \subseteq \mathbb{F}_q^n$. Then for each message m_i either a forgery can be produced or a new condition on $\mathbf{X}$ is found, thus decreasing $\dim L$, at the cost of a signature query. This is done by checking if the system $A(\mathbf{z}, m)\mathbf{x} = B(m)\mathbf{y}$ can be solved for a given $\mathbf{z} \in \mathbb{F}_q^h$ and all $\mathbf{x} \in L$. More specifically we define $S(L, m)$, the *solutions* set, as the collection of all those $\mathbf{z}$ for which any $\mathbf{x} \in L$ makes the systems solvable, formally

$$S(L, m) \;=\; \{\mathbf{z} \in \mathbb{F}_q^h \;:\; A(\mathbf{z}, m) \cdot L \subseteq \operatorname{Im} B(m)\}.$$

If $S(L, m)$ is easy to compute, a strategy for $\mathcal{B}$ is to check whether $S(L, m) \neq \varnothing$ and in this case to get any $\mathbf{z} \in S(L, m)$ and find, using pseudo-inverses or Gaussian elimination, a vector $\mathbf{Y} \in \mathbb{G}^k$ such that $A(\mathbf{z}, m)\mathbf{X} = B(m)\mathbf{Y}$. Conversely, if $S(L, m) = \varnothing$, $\mathcal{B}$ may request a signature $(\mathbf{Y}, \mathbf{z})$, which implies that the exponent $\mathbf{x}$ of $\mathbf{X}$ satisfies the condition $A(\mathbf{z}, m)\mathbf{x} \in \operatorname{Im} B(m)$. Notice that, unlike the attack presented in [11], $\mathcal{B}$ is required to be PPT and thus computing $S(L, m)$ efficiently is essential in our argument. This will follow as we assumed the verification to be strictly linear, implying that $S(L, m)$ is an affine space.

Although $\mathcal{B}$ effectively breaks security, we can only upper bound the number of signatures queried by $n_1 + n_2$, i.e. one for each group element in the CRS $\mathbf{X}_1$ and verification key $\mathbf{X}_2$, since initially $L = \mathbb{F}_q^{n_1+n_2}$ with dimension $n_1 + n_2$. In order to reduce the requested signatures to be at most n_2 we introduce a preprocessing phase to find as many linear relations among group elements of the CRS as possibile and then run $\mathcal{B}$ providing as input a refined space L. Informally, if $\mathcal{B}$ is unable to find new relations among the elements of $\mathbf{X}_1$, then $\dim L$ can at most decrease by n_2, yielding the desired upper bound.

To conclude we then need to describe how the preprocessing is carried out: The core idea is to initialize the set of possible exponents $V = \mathbb{F}_q^{n_1}$ and execute several times $\mathcal{B}(\mathsf{vk}^*, V)$ replying to signing queries with $\mathsf{S.Sign}(\mathsf{sk}^*, \cdot)$ where $\mathsf{vk}^*, \mathsf{sk}^* \leftarrow^{\$} \mathsf{S.SetupKey}(1^\lambda, \mathbf{X}_1, s_1)$ is freshly sampled each time. If in some of those executions $\mathcal{B}$ is able to find a new relation among the group elements, then V is updated accordingly (lowering its dimension by at least 1), and a new round of simulations is run. Conversely if $\mathcal{B}(\mathsf{vk}^*, V)$ fails to find new relations several times in this simulated environment, then it is executed one last time with the real verification key vk and signing oracle. If no new relation is found in this last execution, $\mathcal{A}$ concludes by returning the forgeries found by $\mathcal{B}$. Otherwise $\mathcal{A}$ aborts.

Informally $\mathcal{A}$ aborts with low probability since the simulated and real executions are identically distributed from $\mathcal{B}$ perspective and in particular since no relation is found among the many simulated executions, it is unlikely this will happen in the real one. Finally we remark that simulating the signature challenger in this preprocessing phase is crucial. In this way the only signature queries performed by $\mathcal{A}$ are those requested by the last execution of $\mathcal{B}$.

Having provided the intuition behind the attacker $\mathcal{A}$ built on top of $\mathcal{B}$, we now proceed to prove the theorem through a sequence of claims. We begin by

Adversary $\mathcal{B}(\mathsf{vk}, V)$:

1 : Set $L \leftarrow V \times \mathbb{F}_q^{n_2} \subseteq \mathbb{F}_q^{n_1+n_2}$
2 : Initialize the set of forgeries $F \leftarrow \varnothing$ and call $\theta \leftarrow n_2 + \vartheta$
3 : Sample $m_1, \ldots, m_\theta \xleftarrow{\$} \mathsf{S.M}$ distinct messages
4 : **For** $i \in \{1, \ldots, \theta\}$:
5 : **If** $S(L, m_i) \neq \varnothing$:
6 : Get a vector $\mathbf{z} \in S(L, m_i)$
7 : Find a solution $\mathbf{Y} \in \mathbb{G}^k$ such that $A(\mathbf{z}, m_i)\mathbf{X} = B(m_i)\mathbf{Y}$
8 : Set $\sigma \leftarrow (\mathbf{Y}, \mathbf{z})$ and store $F \leftarrow F \cup \{(m_i, \sigma)\}$
9 : **Else**:
10 : Query m_i to the challenger and get $\sigma = (\mathbf{Y}, \mathbf{z})$
11 : Update $L \leftarrow L \cap \{\mathbf{x} \in \mathbb{F}_q^n \ : \ A(\mathbf{z}, m_i)\mathbf{x} \in \operatorname{Im} B(m_i)\}$
12 : Return F, L

Fig. 3. $\mathcal{B}$ breaking ϑ-UF of an algebraic signature with strictly linear verification.

stating the following properties about $\mathcal{B}(\mathsf{vk}, V)$ where we denote $\mathsf{vk} = (\mathbf{X}, s)$ with $\mathbf{X} = \mathbf{X}_1 || \mathbf{X}_2$, $\mathbf{x}_1$ the discrete logarithm of $\mathbf{X}_1$ and $\mathbf{x}$ the discrete logarithm of $\mathbf{X}$. Finally we denote $\pi : \mathbb{F}_q^{n_1} \times \mathbb{F}_q^{n_2} \to \mathbb{F}_q^{n_1}$ the projection on first component, i.e. $\pi(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1$.

Claim 1 *If L is an affine space, $S(L, m)$ is an affine space. Moreover an affine base for $S(L, m)$ can be computed in polynomial time.*

Claim 2 *If $\mathbf{x}_1 \in V$ then at any step of $\mathcal{B}(\mathsf{vk}, V)$, $\mathbf{x} \in L$.*

Claim 3 *If $\mathbf{x}_1 \in V$, $\mathcal{B}$ is* PPT *and upon returning (F, L), F is a set of valid forgeries.*

Claim 4 *For a given m_i, if the condition at step 5 is not satisfied, i.e. $S(L, m_i) = \varnothing$, then after step 11 the dimension of L decreases strictly.*

Claim 5 *After the execution of line 1, Fig 3, $\dim L = n_2 + \dim V$ and if $\mathcal{B}(\mathsf{vk}, V)$ returns (F, L) with $\pi(L) = V$ then $\dim L \geq \dim V$.*

Next we state the following properties about $\mathcal{A}$

Claim 6 *$\mathcal{A}$ is* PPT.

Claim 7 *At any step of $\mathcal{A}$ execution, $\mathbf{x}_1 \in V$.*

Claim 8 *$\mathcal{A}$ fails with probability $\Pr[\mathcal{A}(\mathsf{vk}) \to \mathsf{fail}] \leq 1/2$.*

Adversary $\mathcal{A}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})$:

1 : Parse $\mathsf{vk} = (\mathbf{X}, s)$ with $\mathbf{X} = \mathbf{X}_1||\mathbf{X}_2$ and $s = s_1||s_2$
2 : Initialize $V \leftarrow \mathbb{F}_q^{n_1}$ the space of potential exponents of $\mathbf{X}_1$
3 : **Do**:
4 : **For** $2n_1 + 1$ times:
5 : $\mathsf{vk}^*, \mathsf{sk}^* \leftarrow^{\$} \mathsf{S.SetupKey}(1^\lambda, \mathbf{X}_1, s_1)$
6 : Execute $F^*, L^* \leftarrow^{\$} \mathcal{B}^{\mathsf{S.Sign}(\mathsf{sk}^*,\cdot)}(\mathsf{vk}^*, V)$
7 : Set $V^* \leftarrow \{\mathbf{x}_1 \,:\, \exists \mathbf{x}_2 \,:\, \mathbf{x}_1||\mathbf{x}_2 \in L^*\}$ the projection of L^* on $\mathbb{F}_q^{n_1}$
8 : **If** $V^* \neq V$:
9 : Update $V \leftarrow V^*$, **break**
10 : **Until** the for-cycle ends without interruptions
11 : Execute $F, L \leftarrow^{\$} \mathcal{B}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{vk}, V)$
12 : Compute V^* as the projection of L on $\mathbb{F}_q^{n_1}$
13 : **If** $V^* \neq V$: Return fail
14 : **Else**: Return F

Fig. 4. $\mathcal{A}$ breaking the ϑ-UF of an algebraic signature using as subroutine an algorithm $\mathcal{B}$, which is that of Fig. 3 in the case of schemes with strictly linear verification, or that of Fig. 5 in the case of schemes with generic verification.

First we observe these claims imply the thesis. Indeed by Claim 8, with probability greater than $1/2$, $\mathcal{A}$ does not return fail. By construction, this implies that in the last execution $\mathcal{B}(\mathsf{vk}, V)$ returns (F, L) with $\pi(L) = V$. Thus by Claim 5 $n_2 + \dim V \geq L \geq \dim V$ at any step of $\mathcal{B}$ during its last execution. As a consequence $\dim L$ can decrease at most n_2 times. Applying Claim 4 we conclude that $S(L, m_i) = \varnothing$ can happen at most n_2 times because each time this occurs, $\dim L$ decreases. It follows then that for at least $\theta - n_2 = \vartheta$ messages, the condition $S(L, m_i) \neq \varnothing$ is satisfied, meaning that $\mathcal{B}$ adds a new signature to the set F, which in the end will have cardinality $|F| \geq \vartheta$. Finally, since $\mathbf{x} \in V$ by Claim 7, we can apply Claim 3 to conclude that F is a valid set of forgeries, implying that $\mathcal{A}$ breaks ϑ-UF.

Next, we provide a proof for each of these claims:

Proof of Claim 1. We start observing that if L is any set and $\mathbf{x}_1, \ldots, \mathbf{x}_d \in L$ is a base for the linear span of L then $S(L, m) = \bigcap_{i=1}^d S(\mathbf{x}_i, m)$. By construction, $\mathbf{x}_i \in L$ implies $S(L, m) \subseteq S(\mathbf{x}_i, m)$, and in particular $S(L, m) \subseteq \cap_{i=1}^d S(\mathbf{x}_i, m)$. Conversely let $\mathbf{z}$ be a vector in the intersection of all $S(\mathbf{x}_i, m)$. We can find vectors $\mathbf{u}_i \in \mathbb{F}_q^k$ such that $A(\mathbf{z}, m)\mathbf{x}_i = B(m)\mathbf{u}_i$. Since $\mathbf{x}_1, \ldots, \mathbf{x}_d$ is a base for the linear span of L, for any $\mathbf{x} \in L$ we can express it as a linear combination $\alpha_1\mathbf{x}_1 + \ldots + \alpha_d\mathbf{x}_d$. In conclusion

$$A(\mathbf{z}, m)\mathbf{x} \;=\; \sum_{i=1}^{d} \alpha_i A(\mathbf{z}, m)\mathbf{x}_i \;=\; \sum_{i=1}^{d} \alpha_i B(m)\mathbf{u}_i \;=\; B(m) \sum_{i=1}^{d} \alpha_i \mathbf{u}_i.$$

Thus $A(\mathbf{z}, m)\mathbf{x} \in \operatorname{Im} B(m)$ and in particular $\mathbf{z} \in S(L, m)$.

In order to show that $S(L, m)$ is efficiently computable it suffices to show that $S(\mathbf{x}, m)$ can be computed in polynomial time for any point $\mathbf{x}$. To this aim let $f_{\mathbf{x}} : \mathbb{F}_q^h \to \mathbb{F}_q^\ell$ be such that $f(\mathbf{z}) = A(\mathbf{z}, m)\mathbf{x}$. Since the scheme has strictly linear verification (Definition 8) $A(\,\cdot\,, m)$ is an affine map and so is f. Furthermore by construction $S(\mathbf{x}, m) = f_{\mathbf{x}}^{-1}(\operatorname{Im} B(m))$ since $\mathbf{z} \in S(\mathbf{x}, m)$ if and only if $A(\mathbf{z}, m)\mathbf{x} \in \operatorname{Im} B(m)$. This concludes the argument as the preimage through an affine map of a linear space is an affine space which can be computed in polynomial time.

Proof of Claim 2. If $\mathbf{x}_1 \in V$ then $\mathbf{x} = \mathbf{x}_1 || \mathbf{x}_2 \in V \times \mathbb{F}_q^{n_2}$ which by construction implies that, when L is initialized, $\mathbf{x} \in L$. Next assume by induction $\mathbf{x} \in L$ in all previous steps. The only instruction in $\mathcal{B}$ that may modify L is in step 11 and when this is executed, since $\sigma = (\mathbf{Y}, \mathbf{z})$ is a valid signature by perfect correctness, we have

$$A(\mathbf{z}, m_i)\mathbf{X} = B(m_i)\mathbf{Y} \quad \Rightarrow \quad A(\mathbf{z}, m_i)\mathbf{x} \in \operatorname{Im} B(m_i).$$

Proof of Claim 3. To prove that $\mathcal{B}$ is a PPT algorithm, observe that the for-loop is executed $\theta = n_2 + \vartheta$, that is polynomially bounded, times. Inside the loop, checking $S(L, m_i) \neq \varnothing$ and possibly computing a $\mathbf{z} \in S(L, m_i)$ can be done efficiently from Claim 1 by computing a base for it. Next, calling $\mathbf{x}$ the discrete logarithm of $\mathbf{X}$, we have that $A(\mathbf{z}, m_i)\mathbf{x} \in \operatorname{Im} B(m_i)$ because

$$\mathbf{z} \in S(L, m_i) \quad \Rightarrow \quad A(\mathbf{z}, m_i) \cdot L \subseteq \operatorname{Im} B(m_i) \quad \Rightarrow \quad A(\mathbf{z}, m_i)\mathbf{x} \in \operatorname{Im} B(m_i)$$

where the last implication follows as $\mathbf{x} \in L$ by Claim 2 and the assumption $\mathbf{x}_1 \in V$. Thus, calling H a weak-inverse[12] of $B(m_i)$, which can be computed efficiently, the vector $\mathbf{Y}$ can be set as $H \cdot A(\mathbf{z}, m_i)\mathbf{X}$. Indeed, as $A(\mathbf{z}, m_i)\mathbf{X} \in \operatorname{Im} B(m_i)$ there exists a vector $\mathbf{Z} \in \mathbb{G}^k$ such that $A(\mathbf{z}, m_i)\mathbf{X} = B(m_i)\mathbf{Z}$ and in particular

$$B(m_i)\mathbf{Y} = B(m_i)HA(\mathbf{z}, m_i)\mathbf{X} = B(m_i)HB(m_i)\mathbf{Z} = B(m_i)\mathbf{Z} = A(\mathbf{z}, m_i)\mathbf{X}.$$

Finally, given the bases of two affine spaces, a base of their intersection can be computed efficiently. This conclude the proof that $\mathcal{B}$ is PPT.

For the second part, by construction each entry in F is of the form $(m_i, \mathbf{Y}, \mathbf{z})$ such that

$$A(\mathbf{z}, m_i)\mathbf{X} = B(m_i)\mathbf{Y}.$$

Therefore, by our definition of signatures with linear verification scheme, the verifier accepts $(m_i, \mathbf{Y}, \mathbf{z})$. The claim is thus proven.

Proof of Claim 4. Since the condition at step 5 is not satisfied, $S(L, m_i) = \varnothing$ and in particular $\mathbf{z} \notin S(L, m_i)$ implying that $A(\mathbf{z}, m_i)\mathbf{x} \notin \operatorname{Im} B(m_i)$ for some $\mathbf{x} \in L$. Therefore L is not contained in the space of all $\mathbf{x}$ such that $A(\mathbf{z}, m_i)\mathbf{x} \in \operatorname{Im} B(m_i)$ and in particular its dimension decreases after the execution of step 11

[12] H is the weak-inverse of A if $A \cdot H \cdot A = A$

Proof of Claim 5. The first part follow as L is initially $V \times \mathbb{F}_q^{n_2}$ of dimension $\dim V + n_2$. The second part follows by linear algebra since $\dim L \geq \dim \pi(L) = \dim V$.

Proof of Claim 6. Since $\mathsf{S.SetupKey}$, $\mathsf{S.Sign}$ and $\mathcal{B}$ are PPT algorithm, by Claim 3 in the last case, each step in the loop can be computed efficiently. In particular, as $2n_1+1$ is polynomially bounded, each for-loop in $\mathcal{A}$ can be performed efficiently.

Next we show that the procedure inside the Do-Until loop is repeated at most $n_1 + 1$ times. The key observation is that during the execution of $\mathcal{B}$, the space L forms a monotone decreasing sequence, implying that when $\mathcal{B}(\mathsf{vk}^*, V) \to (F^*, L^*)$ then $L^* \subseteq V \times \mathbb{F}_q^{n_2}$. In particular this implies that $\pi(L^*) \subseteq \pi(V \times \mathbb{F}_q^{n_2}) = V$. Thus if at any point the for-loop is halted, $\pi(L^*) = V^* \neq V$ implies $V^* \subseteq V$. Hence the dimension of V strictly decreases, and since initially $\dim(V) = n_1$, the foor-loop can be halted at most n_1 times.

Finally, using again that $\mathcal{B}$ is an efficient algorithm, computing F, L can be done in polynomial time. It follows that $\mathcal{A}$ is PPT.

Proof of Claim 7. We proceed by induction. Initially $V = \mathbb{F}_q^{n_1}$ implies $\mathbf{x}_1 \in V$. Next we observe that the value of V is only changed if, within the for-loop, $V^* \neq V$ (see step 8, Fig. 4). Assume by induction that before this step is executed $\mathbf{x}_1 \in V$. Then, when this happens, $\mathcal{B}(\mathsf{vk}^*, V) \to (F^*, L^*)$ had been executed with $\mathbf{x}_1 \in V$. By Claim 2 this implies that $\mathbf{x} \in L^*$ and in particular $\mathbf{x}_1 = \pi(\mathbf{x}) \in \pi(L^*) = V^*$. Thus when $\mathcal{A}$ sets $V \leftarrow V^*$, $\mathbf{x}_1 \in V$.

Proof of Claim 8. Define the following events:

- $\mathcal{E}_{i,j}$ = "During the i-th iteration of the Do-Until loop, and the j-th iteration of the for loop, $\mathcal{B}^{\mathsf{S.Sign}(\mathsf{sk}^*, \cdot)}(\mathsf{vk}^*, V)$ returns (F^*, L^*) such that $\pi(L^*) = V$".
- $\mathcal{E}_{\mathsf{last}}$ = "$\mathcal{B}^{\mathsf{S.Sign}(\mathsf{sk}, \cdot)}(\mathsf{vk}, V)$ returns F, L with $\pi(L) = V$".

Furthermore let $I \sim \{1, \ldots, n_1 + 1\}$ be the random variable such that $\mathcal{A}$ terminates the Do-Until loop after the I-th execution. Then we observe that, conditioned on $\mathbf{X}_1, s_1$ and the V at iteration i, the event $\mathcal{E}_{i,j}$ depends only on the random coins used for $\mathcal{B}$, $\mathsf{S.SetupKey}$ and $\mathsf{S.Sign}$ which are chosen independently at each execution of $\mathcal{B}$. In particular, for a fixed i, the events $\{\mathcal{E}_{i,j}\}_j$ are independent and, since for $\mathcal{E}_{i,j}, \mathcal{E}_{i,k}$ with $j \neq k$ the procedure $\mathcal{B}$ is invoked with the same input

$$\Pr[\mathcal{E}_{i,j}] = \Pr[\mathcal{E}_{i,k}].$$

We may therefore define $p_i = \Pr[\mathcal{E}_{i,1}]$ as the success probability of each execution of $\mathcal{B}$ during the i-th loop. Similarly, if $I = i$, the vector space V given in input to $\mathcal{B}$ is by construction equal to the one used during the i-th execution of the Do-Until loop. In particular

$$p_i = \Pr[\mathcal{E}_{\mathsf{last}} | I = i].$$

To conclude we show that

$$\begin{aligned}
\Pr[\mathcal{A} \to \mathsf{fail}] &= \Pr[\neg\mathcal{E}_{\mathsf{last}}] = \sum_{i=1}^{n_1+1} \Pr[\neg\mathcal{E}_{\mathsf{last}} | I = i] \cdot \Pr[I = i] \\
&\leq \sum_{i=1}^{n_1+1} \Pr[\neg\mathcal{E}_{\mathsf{last}} | I = i] \cdot \Pr[\mathcal{E}_{i,1} \wedge \ldots \wedge \mathcal{E}_{i,2n_1+1}] \\
&= \sum_{i=1}^{n_1+1} \Pr[\neg\mathcal{E}_{\mathsf{last}} | I = i] \cdot \prod_{j=1}^{2n_1+1} \Pr[\mathcal{E}_{i,j}] \\
&= \sum_{i=1}^{n_1+1} (1 - p_i) \cdot p_i^{2n_1+1} \\
&\leq \sum_{i=1}^{n_1+1} \frac{1}{2n_1+2} = \frac{n_1+1}{2n_1+2} = \frac{1}{2}.
\end{aligned}$$

where the first inequality comes from the fact that $I = i$ implies $\mathcal{E}_{i,j}$ for all $j \in \{1, \ldots, 2n_1 + 1\}$, while the second inequality comes from the fact that the function $f_t(x) = (1-x)x^t$ is upper bounded by $1/(t+1)$ when $x \in [0, 1]$. Indeed $f_t(0) = f_t(1) = 0$ and its derivative vanishes only at $t/(t+1)$, which has to be the maximum point, implying that

$$(1 - x) \cdot x^t \leq \left(1 - \frac{t}{t+1}\right) \cdot \left(\frac{t}{t+1}\right)^t \leq \frac{1}{t+1}.$$

4.2 Attack to Schemes with Generic Verification

Theorem 3. *Given an algebraic signature scheme with generic verification, for any ϑ such that $n_2 + \vartheta \leq |\mathsf{S.M}|$ there exists an adversary $\mathcal{A}$ that in the unforgeability experiment in Fig. 2 performs at most n_2 signature queries and produces ϑ distinct forgeries.*

Moreover, calling κ an upper bound on the signature bit-length, and χ an upper bound on the number of queries $\mathsf{S.Vfy}$ performs to $\mathcal{O}^0_{\mathsf{eq}}$, then $\mathcal{A}$ runs in time $O(\vartheta \cdot 2^\kappa \cdot 2^\chi \cdot \mathsf{poly}(\lambda))$ and performs $O(\vartheta \cdot \mathsf{poly}(\lambda))$ queries to $\mathcal{O}_{\mathsf{add}}$ and $\mathcal{O}^0_{\mathsf{eq}}$.

Proof. As done in Theorem 4 we begin by providing an attack $\mathcal{B}$ which breaks the scheme but performs potentially $n_1 + n_2$ signature queries.

At a high level $\mathcal{B}$, given the verification key $\mathsf{vk} = (\mathbf{X}, s)$, will keep track of all possible exponents of $\mathbf{X}$ in a set L and for each message m either the dimension of L decreases by one or $\mathcal{B}$ finds a forgery. Assume without loss of generality that signatures are of the form $(\mathbf{Y}', t')$ with $\mathbf{Y}' \in \mathbb{G}^k$ and $t' \in \{0, 1\}^\kappa$.

For any m, our adversary attempts to produce a forgery as follows: For all possible $t \in \{0, 1\}^\kappa$, it executes the verification algorithm by simulating a generic group $\widetilde{\mathbb{G}}$ with oracles $\widetilde{\mathcal{O}}_{\mathsf{add}}$ and $\widetilde{\mathcal{O}}^0_{\mathsf{eq}}$. More specifically, since $\mathsf{S.Vfy}$ requires as input the verification key $(\mathbf{X}, s)$, the message m and the signatures $(\mathbf{Y}, t)$, $\mathcal{B}$

reproduces all the group elements involved by assigning dummy indexes for $\widetilde{\mathbf{X}}$, $\widetilde{\mathbf{Y}}$ and runs $\mathsf{S.Vfy}((\widetilde{\mathbf{X}}, s), m, (\widetilde{\mathbf{Y}}, t))$. During the execution, each query to $\widetilde{\mathcal{O}}_{\mathsf{add}}$ is emulated by simply returning new incremental indexes, while to emulate $\widetilde{\mathcal{O}}^0_{\mathsf{eq}}$, χ bits $\beta_1, \ldots, \beta_\chi$ are chosen at the beginning of the execution so that the answer to the i-th query will be β_i. Note that each element T_i the verifier queries to $\widetilde{\mathcal{O}}^0_{\mathsf{eq}}$ has to be a linear combination of the initial group elements he received, i.e. $T_i = \mathbf{a}_i^\top \widetilde{\mathbf{X}} - \mathbf{b}_i^\top \widetilde{\mathbf{Y}} - c_i \cdot \widetilde{G}$ obtained though $\widetilde{\mathcal{O}}_{\mathsf{add}}$, and $\mathcal{B}$ can extract these coefficients.

Repeating the execution of $\mathsf{S.Vfy}$ for different values of $\beta_1, \ldots, \beta_\chi$ implicitly defines a tree of height χ in which paths are determined by the replies $\mathcal{B}$ gave at the i-th query to $\widetilde{\mathcal{O}}^0_{\mathsf{eq}}$. If at some point a path $\beta_1, \ldots, \beta_\chi$ that makes the verifier accept is found, $\mathcal{B}$ can try to find a vector $\mathbf{Y}$ in the real GGM, such that the i-query $\mathsf{S.Vfy}$ would do to $\mathcal{O}^0_{\mathsf{eq}}$ will be answered with β_i. If such a $\mathbf{Y}$ is found, then $(\mathbf{Y}, t)$ will be a valid forgery for m.

Recalling that the i-th query has the form $T_i = \mathbf{a}_i \widetilde{\mathbf{X}} - \mathbf{b}_i \widetilde{\mathbf{Y}} - c_i \cdot G$, then $\mathcal{B}$ needs to find a vector $\mathbf{Y}$ such that for all $i \in \{1, \ldots, \chi\}$

$$\mathbf{a}_i^\top \mathbf{X} = \mathbf{b}_i^\top \mathbf{Y} + c_i \cdot G \;\text{ when }\; \beta_i = 1, \qquad \mathbf{a}_i^\top \mathbf{X} \neq \mathbf{b}_i^\top \mathbf{Y} + c_i \cdot G \;\text{ when }\; \beta_i = 0$$

Regarding the equations on the left side, they can be packed up into a system $A\mathbf{X} = B\mathbf{Y} + \mathbf{c} \cdot G$. Through pseudo-inverses or Gaussian elimination is easy to check if solutions exists for all $\mathbf{x} \in L$ (as in the proof of Theorem 2). If this is not the case $\mathcal{B}$ simply discards this path and continues its brute-force search. However, even if the previous condition is satisfied, for some of the points $\mathbf{x}$ in L it may be the case that any vector $\mathbf{y}$ satisfying $A\mathbf{x} = B\mathbf{y} + \mathbf{c}$ fails to satisfy some of the inequalities above $\mathbf{a}_i^\top \mathbf{x} \neq \mathbf{b}_i^\top \mathbf{y} + c_i$, implying that no solution $\mathbf{Y} \in \mathbb{G}^k$ can be found if $\mathbf{x}$ is the discrete logarithm of $\mathbf{X}$. We call these points $\mathbf{x} \in L$ *faulty* and, more specifically, the set of faulty points is defined as

$$\mathcal{F}^{A,B,\mathbf{c}}_{\mathbf{a},\mathbf{b},c} \;=\; \{\mathbf{x} : A\mathbf{x} \in \operatorname{Im} B + \mathbf{c}, \quad \forall \mathbf{y} \in \mathbb{F}_q^m \quad A\mathbf{x} = B\mathbf{y} + \mathbf{c} \;\Rightarrow\; \mathbf{a}^\top \mathbf{x} = \mathbf{b}^\top \mathbf{y} + c\}.$$

Three possible cases may occur now:

- If all points in L are faulty with respect to some inequality constraint, then $\mathcal{B}$ gives up on the path as the solution $\mathbf{Y}$ does not exist.
- If not all points are faulty $\mathcal{B}$ attempts to solve the system, which requires expensive queries to $\mathcal{O}_{\mathsf{add}}, \mathcal{O}^0_{\mathsf{eq}}$: if a solution $\mathbf{Y}$ satisfying all constraints is found, this is a valid forgery.
- If not all points are faulty, but no solution can be found, it means that $\mathbf{x}$, the discrete log of $\mathbf{X}$, has to be a faulty point. This information reduces the dimension of L as not all points in L are faulty.

Finally, if no solution can be found for any $t \in \{0,1\}^\kappa$ and path $\beta_1, \ldots, \beta_\chi$, $\mathcal{B}$ queries a signature for m and uses this information to reduce the dimension of L. As for the proof of Theorem 2, $\mathcal{B}$ might overall query $n_1 + n_2$ signatures (as opposed to the desired n_1) since initially it has no information on the exponents

of $\mathbf{X}$, i.e. $\dim L = n_1 + n_2$, and each signature query may reveal only one new linear combination among these group elements. To address this issue we use the same strategy presented in Theorem 2, that is, we use $\mathcal{B}$ in a black-box way inside the algorithm $\mathcal{A}$, formally described in Fig. 4. The main idea is again that $\mathcal{A}$ initially extracts linear combinations among CRS elements that could be found by $\mathcal{B}$, and finally executes $\mathcal{B}$ providing the retrieved information as input. In this way $\mathcal{B}$ will, with significant probability, only find relations among elements of $\mathbf{X}_2$, thus requesting at most n_2 signatures.

A detailed description of $\mathcal{A}$ appears in Fig. 5, while a more detailed proof of the Theorem appears in the full version.

5 Conclusions

5.1 Impossibility of Algebraic Vector Commitments

Using both the negative results provided in the previous sections for algebraic signatures and Theorem 1 connecting the efficiency of a VC to the security of the associated signature scheme, we obtain two lower bounds for algebraic vector commitments

Theorem 4. *Given a position binding algebraic VC with strictly linear verification, let $\ell_c = \ell_c(n)$ and $\ell_\pi = \ell_\pi(n)$ be respectively the commitment and opening bit length to commit to a vector of n entries. Then*

$$\nu_2 + \frac{\lambda + \ell_c + \nu_2 \cdot (\ell_\pi + \log |\mathsf{VC.M}|)}{\log |\mathsf{VC.M}|} \;\geq\; n.$$

Proof. Assume there exists an algebraic VC with strictly linear verification contradicting the above inequality and satisfying position binding. Then by Theorem 1 the signature scheme obtained through the transformation in Fig. 1 would satisfy ϑ-UF with

$$\vartheta(\mathsf{vk}, Q) \;=\; \frac{\lambda + \ell_c + |Q| \cdot (\ell_\pi + \log |\mathsf{VC.M}|)}{\log |\mathsf{VC.M}|}$$

and its message space would have size $|\mathsf{S_{VC}.M}| = n$. Since vk contains ν_2 group elements excluding those that belong to the CRS, i.e. the public parameters of the original Vector Commitment, the attacker $\mathcal{A}$ from Theorem 2 can produce at least $n - \nu_2$ forgeries performing at most ν_2 queries. Called Q the set of queries performed by $\mathcal{A}$ we would have that

$$\vartheta(\mathsf{vk}, Q) \;\leq\; \frac{\lambda + \ell_c + \nu_2 \cdot (\ell_\pi + \log |\mathsf{VC.M}|)}{\log |\mathsf{VC.M}|} \;<\; n - \nu_2$$

where we use the fact that $|Q| \leq \nu_2$ in the first inequality. This is then a contradiction since $\mathcal{A}$ would breaks the ϑ-UF of the derived signature, implying that the given vector commitment was not binding.

Adversary $\mathcal{B}(\mathsf{vk}, V)$:

1 : Initialize $F \leftarrow \varnothing$ the set of forgeries
2 : Call $L = V \times \mathbb{F}_q^{n_2}$ the set of possible exponents of $\mathbf{X}$
3 : Call $\theta = n + \vartheta$ and sample $m_1, \ldots, m_\theta \leftarrow^{\$} \mathsf{S.M}$ distinct messages
4 : **For** $m \in \{m_1, \ldots, m_\theta\}$:
5 : **For** $t \in \{0,1\}^\kappa$ and $(\beta_1, \ldots, \beta_\chi) \in \{0,1\}^\chi$:
6 : Simulate a Generic Group $\widetilde{\mathbb{G}}$ with generator $\widetilde{G}$ and oracles $\widetilde{\mathcal{O}}_{\mathsf{add}}$ and $\widetilde{\mathcal{O}}^0_{\mathsf{eq}}$
7 : Assign indices for two vectors $\widetilde{\mathbf{X}} \in \widetilde{\mathbb{G}}^n$ and $\widetilde{\mathbf{Y}} \in \widetilde{\mathbb{G}}^k$
8 : Run $\mathsf{S.Vfy}((\widetilde{\mathbf{X}}, s), m, (\widetilde{\mathbf{Y}}, t))$ using $\widetilde{\mathbb{G}}$
9 : **When** $\mathsf{S.Vfy}$ queries $\widetilde{\mathcal{O}}_{\mathsf{add}}(T, S)$:
10 : Store a way to express $T + S$ as a linear combination of $\widetilde{\mathbf{X}}$, $\widetilde{\mathbf{Y}}$ and $\widetilde{G}$
11 : Return to $\mathsf{S.Vfy}$ a label for $T + S$
12 : **When** $\mathsf{S.Vfy}$ queries $\widetilde{\mathcal{O}}^0_{\mathsf{eq}}(T_i)$ the i-th time:
13 : Store $\mathbf{a}_i \in \mathbb{F}_q^n$, $\mathbf{b}_i \in \mathbb{F}_q^k$ and $c_i \in \mathbb{F}_q$ such that $T_i = \mathbf{a}_i^\top \widetilde{\mathbf{X}} - \mathbf{b}_i^\top \widetilde{\mathbf{Y}} - c_i \cdot \widetilde{G}$
14 : Return β_i to $\mathsf{S.Vfy}$
15 : **When** $\mathsf{S.Vfy}$ halts and returns $b \in \{0,1\}$:
16 : Let $A = (\mathbf{a}_i \ : \ \beta_i = 1)$, $B = (\mathbf{b}_i \ : \ \beta_i = 1)$ and $\mathbf{c} = (c_i \ : \ \beta_i = 1)$
17 : **If** $b = 0$:
18 : **Continue** cycle in line 5
19 : **Elif** $A \cdot L \nsubseteq \operatorname{Im} B + \mathbf{c}$:
20 : **Continue** cycle in line 5
21 : **Elif** $\exists i \ : \ \beta_i = 0$ and $L \subseteq \mathcal{F}^{A,B,\mathbf{c}}_{\mathbf{a}_i,\mathbf{b}_i,c_i}$:
22 : **Continue** cycle in line 5
23 : **Elif** $\exists i \ : \ \beta_i = 0$ and $\mathbf{X} \in \mathcal{F}^{A,B,\mathbf{c}}_{\mathbf{a}_i,\mathbf{b}_i,c_i} \cdot G$:
24 : Update $L \leftarrow L \cap \mathcal{F}^{A,B,\mathbf{c}}_{\mathbf{a}_i,\mathbf{b}_i,c_i}$
25 : **Break** cycle in line 5
26 : **Else**:
27 : Find $\mathbf{Y} \in \mathbb{G}^k$ s.t. $A\mathbf{X} = B\mathbf{Y} + \mathbf{c}G$ and $\mathbf{a}_i^\top \mathbf{X} \neq \mathbf{b}_i^\top \mathbf{Y} + c_i G$ for $\beta_i = 0$
28 : Store $\sigma \leftarrow (\mathbf{Y}, t)$ and $F \leftarrow F \cup \{(m, \sigma)\}$
29 : **Break** cycle in line 5
30 : **If** the cycle ended without interruptions:
31 : Query a signature for m and wait for $(\mathbf{Y}, t)$
32 : Reconstruct $A, B, \mathbf{c}$ as in step 16 using $(\mathbf{X}, s, m, \mathbf{Y}, t)$ and the group $\mathbb{G}$
33 : Update $L \leftarrow L \cap \{\mathbf{x} \in \mathbb{F}_q^n \ : \ A\mathbf{x} \in \operatorname{Im} B + \mathbf{c}\}$
34 : Return F, L

Fig. 5. $\mathcal{B}$ breaking security of an algebraic signature scheme with generic verification.

Theorem 5. *Given an algebraic VC with generic verification that is position binding against unbounded adversaries performing polynomially bounded queries to the GGM oracles $\mathcal{O}_{\mathsf{add}}$, $\mathcal{O}^0_{\mathsf{eq}}$, using the same notation of Theorem 4, then*

$$\nu_2 + \frac{\lambda + \ell_c + \nu_2 \cdot (\ell_\pi + \log |\mathsf{VC.M}|)}{\log |\mathsf{VC.M}|} \geq n.$$

Proof. Assuming again by contradiction that the above inequality is not satisfied, Theorem 1 implies that the associated signature scheme is ϑ-UF against any unbounded adversary $\mathcal{C}$ making at most polynomially many signature and group operations queries, or otherwise $\mathcal{R}^{\mathcal{C}}$ would break position binding with significant advantage. Notice that since $\mathcal{R}$ is PPT, $\mathcal{R}^{\mathcal{C}}$ still performs polynomially many generic group operations. As in the proof of Theorem 4 then, our initial assumption implies $\vartheta \leq n - \nu_2$. Since the adversary $\mathcal{A}$ of Theorem 3 returns $n - \nu_2$ signatures performing at most ν_2 queries, this contradicts the ϑ-UF of the associated signature against this adversary.

Corollary 1. *Given an algebraic vector commitment with strictly linear verification, then $\ell_c \cdot \ell_\pi = \Omega(n)$. Analogously, given an algebraic vector commitment with generic verification position binding against unbounded adversary performing at most polynomially many queries to the GGM oracles, $\ell_c \cdot \ell_\pi = \Omega(n)$.*

Note that this lower bound implies in both cases that either $\ell_c = \Omega(\sqrt{n})$ or $\ell_\pi = \Omega(\sqrt{n})$.

5.2 Impossibility of Algebraic Signatures

As a by-product of our study on VC we also obtain the following two impossibility results for algebraic signatures which extend the one presented in [11] to a broader family of schemes.

Theorem 6. *For any UF-CMA algebraic signature scheme with strictly linear verification, $n_1 \geq |\mathsf{S.M}|$.*

Theorem 7. *For any algebraic signature scheme with generic verification UF-CMA secure against any unbounded adversary performing at most polynomially many queries to the GGM oracles, $n_1 \geq |\mathsf{S.M}|$.*

Acknowledgements

This work has received funding in part from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under project PICOCRYPT (grant agreement No. 101001283), by the Spanish Government under projects SCUM (ref. RTI2018-102043-B-I00), RED2018-102321-T, and SECURING (ref. PID2019-110873RJ-I00), by the Madrid Regional Government under project BLOQUES (ref. S2018/TCS-4339), by a research grant from Nomadic Labs and the Tezos Foundation, by the Programma ricerca di ateneo UNICT 35 2020-22 linea 2 and by research gifts from Protocol Labs.

References

1. 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. Springer, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_19
2. Bayer, S., Groth, J.: Efficient zero-knowledge argument for correctness of a shuffle. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 263–280. Springer, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_17
3. 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. Springer, Heidelberg (Aug 2011). https://doi.org/10.1007/978-3-642-22792-9_7
4. Boneh, D., Bünz, B., Fisch, B.: Batching techniques for accumulators with applications to IOPs and stateless blockchains. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 561–586. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26948-7_20
5. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_13
6. Bootle, J., Cerulli, A., Chaidos, P., Groth, J., Petit, C.: Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 327–357. Springer, Heidelberg (May 2016). https://doi.org/10.1007/978-3-662-49896-5_12
7. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy. pp. 315–334. IEEE Computer Society Press (May 2018). https://doi.org/10.1109/SP.2018.00020
8. 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. Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64834-3_1
9. Catalano, D., Fiore, D.: Vector commitments and their applications. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 55–72. Springer, Heidelberg (Feb / Mar 2013). https://doi.org/10.1007/978-3-642-36362-7_5
10. Chepurnoy, A., Papamanthou, C., Zhang, Y.: Edrax: A cryptocurrency with stateless transaction validation. Cryptology ePrint Archive, Report 2018/968 (2018), https://eprint.iacr.org/2018/968
11. Döttling, N., Hartmann, D., Hofheinz, D., Kiltz, E., Schäge, S., Ursu, B.: On the impossibility of purely algebraic signatures. In: Theory of Cryptography Conference. pp. 317–349. Springer (2021)
12. Fisch, B.: PoReps: Proofs of space on useful data. Cryptology ePrint Archive, Report 2018/678 (2018), https://eprint.iacr.org/2018/678
13. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct NIZKs without PCPs. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 626–645. Springer, Heidelberg (May 2013). https://doi.org/10.1007/978-3-642-38348-9_37

14. Gennaro, R., Gertner, Y., Katz, J.: Lower bounds on the efficiency of encryption and digital signature schemes. In: 35th ACM STOC. pp. 417–425. ACM Press (Jun 2003). https://doi.org/10.1145/780542.780604
15. Gennaro, R., Trevisan, L.: Lower bounds on the efficiency of generic cryptographic constructions. In: 41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA. pp. 305–313. IEEE Computer Society (2000)
16. Gertner, Y., Kannan, S., Malkin, T., Reingold, O., Viswanathan, M.: The relationship between public key encryption and oblivious transfer. In: 41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA. pp. 325–335. IEEE Computer Society (2000)
17. Gertner, Y., Malkin, T., Reingold, O.: On the impossibility of basing trapdoor functions on trapdoor predicates. In: 42nd FOCS. pp. 126–135. IEEE Computer Society Press (Oct 2001). https://doi.org/10.1109/SFCS.2001.959887
18. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing **17**(2), 281–308 (Apr 1988)
19. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: 21st ACM STOC. pp. 44–61. ACM Press (May 1989). https://doi.org/10.1145/73007.73012
20. Juels, A., Kaliski Jr., B.S.: Pors: proofs of retrievability for large files. In: Ning, P., De Capitani di Vimercati, S., Syverson, P.F. (eds.) ACM CCS 2007. pp. 584–597. ACM Press (Oct 2007). https://doi.org/10.1145/1315245.1315317
21. 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. Springer, Heidelberg (Dec 2010). https://doi.org/10.1007/978-3-642-17373-8_11
22. Kilian, J.: On the complexity of bounded-interaction and noninteractive zero-knowledge proofs. In: 35th FOCS. pp. 466–477. IEEE Computer Society Press (Nov 1994). https://doi.org/10.1109/SFCS.1994.365744
23. Kim, J.H., Simon, D.R., Tetali, P.: Limits on the efficiency of one-way permutation-based hash functions. In: 40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA. pp. 535–542. IEEE Computer Society (1999)
24. Kuszmaul, J.: Verkle trees (2018), https://math.mit.edu/research/highschool/primes/materials/2018/Kuszmaul.pdf
25. 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. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26948-7_19
26. 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
27. 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. Springer, Heidelberg (Feb 2010). https://doi.org/10.1007/978-3-642-11799-2_30

28. Maurer, U.M.: Abstract models of computation in cryptography (invited paper). In: Smart, N.P. (ed.) 10th IMA International Conference on Cryptography and Coding. LNCS, vol. 3796, pp. 1–12. Springer, Heidelberg (Dec 2005)
29. Merkle, R.C.: A digital signature based on a conventional encryption function. In: Pomerance, C. (ed.) CRYPTO'87. LNCS, vol. 293, pp. 369–378. Springer, Heidelberg (Aug 1988). https://doi.org/10.1007/3-540-48184-2_32
30. Micali, S.: CS proofs (extended abstracts). In: 35th FOCS. pp. 436–453. IEEE Computer Society Press (Nov 1994). https://doi.org/10.1109/SFCS.1994.365746
31. Micali, S., Rabin, M.O., Kilian, J.: Zero-knowledge sets. In: 44th FOCS. pp. 80–91. IEEE Computer Society Press (Oct 2003). https://doi.org/10.1109/SFCS.2003.1238183
32. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable random functions. In: 40th FOCS. pp. 120–130. IEEE Computer Society Press (Oct 1999). https://doi.org/10.1109/SFFCS.1999.814584
33. Papakonstantinou, P.A., Rackoff, C., Vahlis, Y.: How powerful are the DDH hard groups? Electron. Colloquium Comput. Complex. p. 167 (2012), https://eccc.weizmann.ac.il/report/2012/167
34. 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. Springer, Heidelberg (May 2013). https://doi.org/10.1007/978-3-642-38348-9_22
35. Peikert, C., Pepin, Z., Sharp, C.: Vector and functional commitments from lattices. In: Theory of Cryptography Conference. pp. 480–511. Springer (2021)
36. Rotem, L., Segev, G., Shahaf, I.: Generic-group delay functions require hidden-order groups. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part III. LNCS, vol. 12107, pp. 155–180. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45727-3_6
37. Schul-Ganz, G., Segev, G.: Accumulators in (and beyond) generic groups: Non-trivial batch verification requires interaction. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part II. LNCS, vol. 12551, pp. 77–107. Springer, Heidelberg (Nov 2020). https://doi.org/10.1007/978-3-030-64378-2_4
38. Schul-Ganz, G., Segev, G.: Generic-Group Identity-Based Encryption: A Tight Impossibility Result. In: Tessaro, S. (ed.) 2nd Conference on Information-Theoretic Cryptography (ITC 2021). Leibniz International Proceedings in Informatics (LIPIcs), vol. 199, pp. 26:1–26:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2021). https://doi.org/10.4230/LIPIcs.ITC.2021.26, https://drops.dagstuhl.de/opus/volltexte/2021/14345
39. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO'84. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (Aug 1984)
40. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT'97. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (May 1997). https://doi.org/10.1007/3-540-69053-0_18
41. Simon, D.R.: Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In: Advances in Cryptology - EUROCRYPT '98. Lecture Notes in Computer Science, vol. 1403, pp. 334–345. Springer (1998)
42. Zhandry, M.: To label, or not to label (in generic groups). Cryptology ePrint Archive, Report 2022/226 (2022), https://eprint.iacr.org/2022/226
43. Zhandry, M., Zhang, C.: The relationship between idealized models under computationally bounded adversaries. Cryptology ePrint Archive, Report 2021/240 (2021), https://eprint.iacr.org/2021/240