All-but-*k* Mercurial Commitments and their Applications[†]

Ryan Henry and Ian Goldberg

Cheriton School of Computer Science
University of Waterloo
Waterloo ON, Canada N2L 3G1

{rhenry, iang}@cs.uwaterloo.ca

***Abstract* — We introduce and formally define *all-but-k mercurial commitments*, a new kind cryptographic commitment that generalizes standard mercurial and non-mercurial (vector) commitments. We provide two concrete constructions for all-but-*k* mercurial commitments: the first is for committing to *unordered lists* (i.e., to *multisets*) and the second is for committing to *ordered lists* (i.e., to *vectors*). Both of our constructions build on Kate et al.'s polynomial commitments, leveraging the algebraic structure of polynomials to fine tune the ordinary binding property of mercurial commitments. To facilitate these constructions, we give novel zero-knowledge protocols for 1) proving knowledge of a point on a committed polynomial, 2) arguing knowledge of the committed polynomial itself, and 3) arguing that a committed polynomial has degree at most *k*.**

***Keywords* — Commitment schemes, mercurial commitments, vector commitments, zero-knowledge proofs, polynomial commitments, efficiency.**

I. Introduction

This paper introduces *all-but-k mercurial commitments*, a new type of cryptographic commitments with similarities to mercurial vector commitments [7]. Our new commitments are suitable for committing to *collections* of values, which may be unordered multisets or ordered vectors. In both instances, the all-but-*k* opening protocol explicitly reveals an upper bound on the "softness" of the opening, thus making all-but-*k* binding "tuneably" stronger than binding in an ordinary mercurial commitment to the same type of collection. One extreme in the binding spectrum occurs when the softness bound is $k \geq n$ for an n-element collection, in which case all-but-*k* binding is just regular mercurial binding; the other extreme occurs when the softness bound is $k = 0$, in which case all-but-*k* binding is regular *non*-mercurial binding. All-but-*k* binding for intermediate softness bounds $k \in [1, n-1]$ is intermediate between the above two extremes.

Contributions. The primary contribution in this paper is:

1. We introduce and formally define all-but-*k* mercurial commitments to multisets and vectors.

The secondary contributions are:

2. We describe three novel zero-knowledge protocols for proving statements about committed polynomials.
3. We construct all-but-*k* mercurial commitment schemes from polynomial commitments and our zero-knowledge proofs about committed polynomials. One of our new schemes is suitable for committing to multisets and the other for committing to vectors.

Outline. We begin with relevant background material in §II: first we cover our notation and cryptographic assumptions in §II-A, next we introduce polynomial commitments in §II-B and present our zero-knowledge proofs about committed polynomials in §II-C, and finally we provide background on mercurial commitments in §II-D. In §III we introduce all-but-*k* mercurial commitments, beginning with basic notation and terminology in §III-A and ending with formal definitions in §III-B. Our all-but-*k* constructions appear in §IV. The multiset construction is in §IV-A and the vector construction is in §IV-B. §V contains some brief concluding remarks.

II. Mathematical preliminaries

A. Notation and cryptographic assumptions

Throughout, $\mathbb{G}$ will denote a cyclic group of 2τ-bit prime order q with a fixed, publicly known generator g and an admissible bilinear pairing $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_{\mathrm{T}}$. (For ease of presentation, we use a symmetric pairing; however, generalizing our constructions to use asymmetric pairings is not difficult and may in fact improve efficiency. For an overview of the available choices of cryptographic pairings, we refer the reader to Galbraith et al. [10].) We use multiplicative notation for the groups $\mathbb{G}$ and $\mathbb{G}_{\mathrm{T}}$ and write $\alpha \in_{\mathrm{R}} \mathbb{G}$ to denote uniform random selection of an element α from $\mathbb{G}$. To prove the security of our constructions, we assume that $\mathbb{G}$ belongs to a polynomial-time generated sequence of groups for which certain computational hardness assumptions hold. A formal statement of each required assumption follows in Definitions 1 through 3; in these definitions — and throughout the paper — the term *negligible* describes a function $\varepsilon\colon \mathbb{N} \to \mathbb{R}$ with the property that, for every real number $c > 0$, there exists a positive integer τ_0 such that $\varepsilon(\tau) < 1/\tau^c$ for all $\tau > \tau_0$.

Our first assumption is the so-called discrete logarithm (DL) assumption, a venerable staple in the cryptographic literature.

[†] This is version 1.0 of CACR 2012-27 (2012-11-30); please consult the change log in Appendix A for differences between this version and prior versions.

Definition 1 (**Discrete Logarithm Assumption; [14, §3.6]**). *Let $\mathcal{G}$ be a probabilistic polynomial-time (PPT) algorithm that, on input $\tau \in \mathbb{N}$, outputs a representation of a group $\mathbb{G}$, its 2τ-bit order q, and a generator $g \in \mathbb{G}$. The **discrete logarithm assumption** holds in the sequence of groups output by $\mathcal{G}$ if there exists a negligible function $\varepsilon\colon \mathbb{N} \to \mathbb{R}$ such that, for every PPT adversary $\mathcal{A}$, we have $\Pr\left[\alpha \leftarrow \mathcal{A}(g, g^\alpha, q) \mid (\mathbb{G}, q, g) \leftarrow \mathcal{G}(\tau) \wedge \alpha \in_R \mathbb{Z}_q^*\right] \le \varepsilon(\tau)$.*

Our next two assumptions regard the hardness of relaxed Diffie-Hellman problems [14, §3.7]. First is the n-strong Diffie-Hellman assumption (n-SDH), which was introduced by Boneh and Boyen in 2004 [3, §2.3] and has since received considerable attention from the cryptographic community.

Definition 2 (***n*-Strong Diffie-Hellman Assumption; [4, §3]**). *Let $\mathcal{G}$ be a PPT algorithm that, on input $\tau \in \mathbb{N}$, outputs a representation of a group $\mathbb{G}$, its 2τ-bit order q, and a generator $g \in \mathbb{G}$. For a fixed positive integer $n \in \mathbb{N}$, the **n-strong Diffie-Hellman assumption** holds in the sequence of groups output by $\mathcal{G}$ if there exists a negligible function $\varepsilon\colon \mathbb{N} \to \mathbb{R}$ such that, for every PPT adversary $\mathcal{A}$, we have $\Pr\left[(c, g^{1/(\alpha+c)}) \leftarrow \mathcal{A}(g, g^\alpha, \ldots, g^{\alpha^n}, q) \mid (\mathbb{G}, q, g) \leftarrow \mathcal{G}(\tau) \wedge \alpha \in_R \mathbb{Z}_q^*\right] \le \varepsilon(\tau)$, where $\mathcal{A}$ is free to choose any $c \in \mathbb{Z}_q \setminus \{-\alpha\}$ during the experiment.*

Our last assumption concerns a problem similar to the n-Diffie-Hellman inversion problem [2, Appendix A] called the n-polynomial Diffie-Hellman (n-polyDH) problem. Au et al. [1, §5.4] implicitly used this latter assumption in their compact e-cash scheme to justify the claim that Nguyen's accumulator construction [15] is bounded. Kate et al. [11, Definition 2] identified and explicitly defined n-polyDH in a subsequent paper.

Definition 3 (***n*-Polynomial Diffie-Hellman Assumption; [11, §2]**). *Let $\mathcal{G}$ be a PPT algorithm that, on input $\tau \in \mathbb{N}$, outputs a representation of a group $\mathbb{G}$, its 2τ-bit order q, and a generator $g \in \mathbb{G}$. For a fixed positive integer $n \in \mathbb{N}$, the **n-polynomial Diffie-Hellman assumption** holds in the sequence of groups output by $\mathcal{G}$ if there exists a negligible function $\varepsilon\colon \mathbb{N} \to \mathbb{R}$ such that, for every PPT adversary $\mathcal{A}$, we have $\Pr\left[(f, g^{f(\alpha)}) \leftarrow \mathcal{A}(g, g^\alpha, \ldots, g^{\alpha^n}, q) \mid (\mathbb{G}, q, g) \leftarrow \mathcal{G}(\tau) \wedge \alpha \in_R \mathbb{Z}_q^*\right] \le \varepsilon(\tau)$, where $\mathcal{A}$ is free to choose any $f \in \mathbb{Z}_q[x]$ with $\sqrt{q} > \deg f > n$ during the experiment.*

Kate et al. remark [11, Footnote 4] that so bounding the degree of f in Definition 3 prevents $\mathcal{A}$ from computing n-polyDH solutions using standard number-theoretic identities (e.g., Fermat's little theorem). We conclude our discussion of computational assumptions by noting that the n-SDH and n-polyDH assumptions are both at least as strong as the DL assumption in that, if either the n-SDH or the n-polyDH assumption holds in the output of $\mathcal{G}$, then the DL assumption also holds in the output of $\mathcal{G}$.

B. Polynomial commitments

A *polynomial commitment scheme* is a cryptographic commitment scheme that lets a committer Alice commit to a polynomial $f \in \mathbb{Z}_q[x]$ and later open the commitment either *polynomial-wise* to f or *point-wise* to $\big(i, f(i)\big)$ for arbitrary $i \in \mathbb{Z}_q$. Kate et al.'s [11] $\textbf{PolyCommit}_{\text{DL}}$ scheme is a pairing-based construction for polynomial commitments in which the size of a commitment is constant with respect to the degree of the committed polynomial: using a common reference string of length $\Theta(n \lg q)$ bits, Alice can commit to any polynomial $f \in \mathbb{Z}_q[x]$ of degree at most $n < \sqrt{q}$ using only a single element from an order-q bilinear group $\mathbb{G}$. The degree-n $\textbf{PolyCommit}_{\text{DL}}$ reference string is an ordered list $\text{PK} = \langle (\mathbb{G}, q, g), g^{\alpha^j} \mid j \in [1, n] \rangle$, where $\langle g \rangle = \mathbb{G}$, $|g| = q$, and $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(\tau)$ for some PPT algorithm $\mathcal{G}$ whose output satisfies the above DL, n-SDH, and n-polyDH assumptions. In practice, we expect the maximum degree n in PK to be at most polylogarithmic in q (so that $n \ll \sqrt{q}$) or, equivalently, polynomial in τ; otherwise, the length of PK and the average-case cost of computing and opening a commitment are both superpolynomial in τ. A trusted initializer or a distributed protocol selects $\alpha \in_R \mathbb{Z}_q^*$ and securely discards it immediately after computing PK.[1]

To commit to a polynomial $f(x) = \sum_{j=0}^{k} a_j x^j \in \mathbb{Z}_q[x]$ of degree $k \le n$, Alice computes $C = \prod_{j=0}^{k} \big(g^{\alpha^j}\big)^{a_j} = g^{f(\alpha)}$ using the appropriate values from PK. Alice can of course open C to f by simply revealing f to Bob and having Bob redo the calculation. Alternatively, Alice can open C to $\big(i, f(i)\big)$ by appealing to the polynomial remainder theorem [17] as follows.

Protocol 1 (**Point-wise opening in $\textbf{PolyCommit}_{\text{DL}}$ [11]**).

A1: *Write $f(x) = Q(x)(x - i) + f(i)$, where $Q(x) = \sum_{j=0}^{k-1} b_j x^j$ is the polynomial quotient obtained by dividing $f(x) - f(i)$ by $(x - i)$. Compute the witness $\omega_i = \prod_{j=0}^{k-1} \big(g^{\alpha^j}\big)^{b_j} = g^{Q(\alpha)}$ and send $\big(i, f(i), \omega_i\big)$ to Bob.*

B2: *Return* "true" *if and only if $e(C/g^{f(i)}, g) \stackrel{?}{=} e(\omega_i, g^\alpha/g^i)$; otherwise, output* "false".

Note that the witness ω_i is itself a $\textbf{PolyCommit}_{\text{DL}}$ commitment whose length, like C, is independent of $\deg f$.

Both point-wise and polynomial-wise opening are complete by inspection. Hiding is unconditional when Bob knows at most $k - 1$ evaluations of a committed degree-k polynomial f, computational under the DL assumption when Bob knows exactly k evaluations of f and that $\deg f = k$, and trivially nonexistent when Bob knows $k + 1$ or more evaluations of f; point-wise evaluation binding is computational under the n-SDH and n-polyDH assumptions and polynomial-wise binding is computa-

[1] Alternatively, one can regard $\textbf{PolyCommit}_{\text{DL}}$ commitments as *trapdoor commitments* with α as the trapdoor: given α, it is possible to open any commitment C to any arbitrary point $(i, y_i) \in \big(\mathbb{Z}_q \setminus \{\alpha\}\big) \times \mathbb{Z}_q$. Note that knowledge of α affects binding but does *not* affect hiding.

tional under the DL assumption [11, Appendix C1]. A *trapdoor prover* can of course solve instances of the n-SDH and n-polyDH problems, which is why she can open arbitrary commitments to arbitrary evaluations. We refer the reader to Kate et al.'s paper [11] for further details on $\mathbf{PolyCommit}_{\text{DL}}$ commitments, including proofs of the above assertions.

C. Zero-knowledge proofs for polynomial commitments

Both of our all-but-k constructions in §IV extend Kate et al.'s $\mathbf{PolyCommit}_{\text{DL}}$ scheme. To facilitate these constructions, we introduce three new honest-verifier zero-knowledge protocols for assertions about committed polynomials; in particular, we give 1) an honest-verifier zero-knowledge proof of knowledge of a point on a committed polynomial, 2) an honest-verifier zero-knowledge argument of knowledge of the committed polynomial itself, and 3) an honest-verifier zero-knowledge argument that a (known) committed polynomial has degree at most k.

1) Proving knowledge of a point on a committed polynomial. In their technical report [12, Appendix E], Kate et al. suggested an honest-verifier zero-knowledge proof of knowledge of a point $\big(i, f(i)\big) \in \mathbb{Z}_q \times \mathbb{Z}_q^*$ on a polynomial f committed to by C, where $i \in \mathbb{Z}_q$ is public and $f(i) \neq 0$ is the prover's secret. Our own argument of knowledge of a committed polynomial, which we will present shortly, follows from the observation that if a non-trapdoor, PPT prover Alice can prove knowledge of such a point for an arbitrary *verifier-selected* challenge $i \in \mathbb{Z}_q$, then with all but negligible probability (in $\tau \approx (\lg q)/2$) she must indeed know a polynomial f such that $C = g^{f(\alpha)}$. Of course, Alice cannot simply let Bob challenge her to prove knowledge of arbitrary points on f using Kate et al.'s protocol, since doing so would leak information about the roots of f. (If honest Bob selects a challenge $i \in \mathbb{Z}_q$ uniformly at random, then with all but negligible probability we have that $f(i) \neq 0$ and so the protocol is indeed honest-verifier statistical zero-knowledge; however, if dishonest Bob conjectures that f has a root at some particular input i, he can use Alice as an oracle to prove or disprove his conjecture. One might object that such an attack is outside of the threat model since, absent this attack, our protocol is still only zero-knowledge with respect to the honest-verifier; nevertheless, we feel that the attack's simplicity and power makes Kate et al.'s protocol unpalatable for our particular application. Moreover, standard defenses like having Bob commit to his challenge input i ahead of time are clearly ineffectual against this particular attack.) We therefore propose the following alternative proof of knowledge, which is slightly more efficient than Kate et al.'s proof and, importantly, does not depend on $f(i)$ being nonzero. (However, the zero-knowledge property *does* require that $C/g^{f(i)} \neq 1$.)

Note that the subprotocol in Steps A1b through B4 of Protocol 2 is a standard Schnorr-like proof of knowledge of a value committed to by a Pedersen commitment [16] in $\mathbb{G}_T$.

Protocol 2 (Proof of knowledge of a point on a committed polynomial).

A1a: *Write* $f(x) = Q(x)(x - i) + f(i)$, *where* $Q(x) = \sum_{j=0}^{k-1} b_j x^j$ *is the polynomial quotient obtained by dividing* $f(x) - f(i)$ *by* $(x - i)$. *Compute the witness* $\omega_i = \prod_{j=0}^{k-1} \left(g^{\alpha^j}\right)^{b_j} = g^{Q(\alpha)}$ *and blind it as* $\tilde{\omega}_i = \omega_i^{1/r}$ *for some* $r \in_R \mathbb{Z}_q^*$.

A1b: *Choose* $s_0, s_1 \in_R \mathbb{Z}_q^*$ *and compute* $S = e(\tilde{\omega}_i, g^\alpha / g^i)^{s_0} \cdot e(g, g)^{s_1}$. *Send* $(\tilde{\omega}_i, S)$ *to Bob.*

B2: *Choose* $c \in_R \mathbb{Z}_q$ *and send it to Alice.*

A3: *Compute* $v_0 = s_0 - r\,c \bmod q$ *and* $v_1 = s_1 - y_i\,c \bmod q$, *where* $y_i = f(i)$. *Send* (v_0, v_1) *to Bob.*

B4: *Return* "true" *if and only if* $\left(e(\tilde{\omega}_i, g^\alpha / g^i)^{v_0} \cdot e(g, g)^{v_1}\right) \cdot e(C, g)^c \stackrel{?}{=} S$; *otherwise, return* "false".

Lemma 1. *Protocol 2 is a system for honest-verifier zero-knowledge proofs of knowledge of a witness-evaluation pair* $(\omega_i, y_i) \in \mathbb{G} \times \mathbb{Z}_q$ *such that* $e(C/g^{y_i}, g) = e(\omega_i, g^\alpha / g^i)$.

Proof. Completeness is clear from inspection of the protocol. The protocol is honest-verifier zero-knowledge because $C/g^{y_i} \neq 1$ implies that $\omega_i \neq 1$ and therefore that the blinded witness $\tilde{\omega}_i = \omega_i^{1/r}$ is statistically independent of ω_i. Since the subprotocol in Steps A1(b)–B4 is itself honest-verifier zero-knowledge [5, §2.4.3], a simulator for the honest verifier just chooses $\tilde{\omega}_i \in_R \mathbb{G}$ and then invokes the simulator for the honest verifier in the subprotocol. Soundness and extractability in Protocol 2 follow directly from soundness and extractability in the subprotocol, which is itself special sound. In particular, the subprotocol proves knowledge of (r, y_i) such that $e(C, g) = e(\tilde{\omega}_i, g^\alpha / g^i)^r \cdot e(g, g)^{y_i}$. This is equivalent to $e(C/g^{y_i}, g) = e(\tilde{\omega}_i^r, g^\alpha / g^i)$; thus, an extractor for the subprotocol can extract (r, y_i) and compute $\omega_i = \tilde{\omega}_i^r$ to get the desired tuple (ω_i, y_i). The knowledge error of Protocol 2 is clearly no worse than the knowledge error of the subprotocol. □

We denote Protocol 2 using a Camenisch-Stadler [6] inspired notation by $\mathrm{PK}\{(\omega_i, y_i) \mid C = g^{f(\alpha)} \wedge \omega_i = g^{Q(\alpha)} \wedge f(x) = Q(x)(x - i) + y_i\}$. As should be clear from this notation, Protocol 2 does *not* prove that Alice knows a polynomial f and output y_i such that $C = g^{f(\alpha)}$ and $f(i) = y_i$; all it proves is that she knows a witness-evaluation pair (ω_i, y_i) that, together with the public values (C, i), satisfies Bob's verification equation in the point-wise opening protocol. Alice may have learned (ω_i, y_i), for example, from some earlier point-wise opening of C by Carol, or perhaps she computed $\omega_i = \left(C/g^{y_i}\right)^{\frac{1}{\alpha - i}}$ using the trapdoor α or by solving an instance of the n-SDH problem. In each case, the proof is still valid: its soundness is not contingent on any computational assumptions.

2) Proving knowledge of a committed polynomial. Consider a variant of Protocol 2 that starts with an additional Step B0 in which Bob arbitrarily chooses the index $i \in_R \mathbb{Z}_q$ with respect to which Alice proves knowledge of (ω_i, y_i) in the remainder of the protocol. As we will show, this variant yields an honest-verifier zero-knowledge *argument* of knowledge of a polynomial f such that $C = g^{f(\alpha)}$. The arguments are computationally convincing under the n-SDH and n-polyDH assumptions. Completeness follows immediately from completeness in Protocol 2. A simulator for the honest verifier just chooses $i \in_R \mathbb{G}$ and invokes the simulator for Protocol 2. It follows that, if a non-trapdoor, PPT prover Alice is not privy to a (nonconstant) polynomial f such that $C = g^{f(\alpha)}$, then the probability that she knows an (ω_i, y_i) pair for Bob's challenge index $i \in_R \mathbb{Z}_q$ is at most about n/q. The restriction $n < \sqrt{q}$ implies that $n/q < 1/\sqrt{q}$ so that the protocol's knowledge error is negligible in $\tau \approx (\lg q)/2$. Thus, if a PPT prover Alice can convince honest Bob with a probability that is polynomial in τ, then a knowledge extractor for Protocol 2 can extract f from Alice in expected polynomial time by extracting $\deg f + 1$ distinct points on f and interpolating. The runtime of the extractor is linear in n, which may, in theory, be superpolynomial in τ; this is fine, however, because the extractor still runs in polynomial time *with respect to the same parameters as the PPT prover*). We thus get the following lemma.

Lemma 2. *The above variant of Protocol 2 is a system for honest-verifier zero-knowledge arguments of knowledge of a polynomial f such that $C = g^{f(\alpha)}$. It is computationally convincing under the n-SDH and the n-polyDH assumptions.*

We denote the above zero-knowledge argument of knowledge by $\mathrm{PK}\{f \mid C = g^{f(\alpha)} \wedge \deg f \le n\}$.

3) Proving that a committed polynomial has degree at most k. Suppose that the $\mathbf{PolyCommit}_{\mathrm{DL}}$ reference string PK bounds the degree of committed polynomials by $n < \sqrt{q}/2$ (which, in practice, is not a restriction since $\sqrt{q}/2$ is still superpolynomial in τ). Under the n-polyDH assumption, it should be infeasible for a non-trapdoor PPT prover Alice to output a commitment to any polynomial whose degree is greater than n but less than $\sqrt{q}$; in particular, if Alice knows a polynomial f such that $C = g^{f(\alpha)}$ and $k < \deg f \le n$, then she can exhibit a commitment C' to $f_k(x) = x^{n-k} f(x)$ with at most negligible probability (in $\tau \approx (\lg q)/2$). Note that the restriction $n < \sqrt{q}/2$ is necessary to ensure that $\deg f_k = \deg f + (n-k)$ will never exceed $\sqrt{q}$ (cf. Definition 3). The following is a noninteractive *trapdoor zero-knowledge* argument (i.e., an argument that is zero-knowledge with respect to all *trapdoor* verifiers) that exploits the above observation to prove that $f(x) = \sum_{j=0}^{k} a_j x^j$ has degree at most k.

Protocol 3 (Proof that a committed polynomial has degree at most k).

A1: *Compute* $C' = \prod_{j=0}^{k} \left(g^{\alpha^{n-k+j}}\right)^{a_j} = g^{f_k(\alpha)}$ *using the appropriate values from* PK. *Send* C' *to Bob.*

B2: *Output* "true" *if and only if* $e(C, g^{\alpha^{n-k}}) \stackrel{?}{=} e(C', g)$; *otherwise, output* "false".

For provable soundness under the n-polyDH assumption, Protocol 3 requires that the prover *knows* a polynomial f such that $C = g^{f(\alpha)}$. We denote Protocol 3 in combination with an argument of knowledge of f by $\mathrm{PK}\{f \mid C = g^{f(\alpha)} \wedge \deg f \le k\}$. It is straightforward to convert the protocol into a (non-trapdoor, interactive) honest-verifier zero-knowledge argument by blinding C' as $(C')^{1/r}$. Alice then uses a Schnorr proof in $\mathbb{G}_T$ to prove knowledge of r such that $e(C', g) = e(C, g^{\alpha^{n-k}})^r$. Fortunately, the simpler trapdoor zero-knowledge version we have presented here suffices for the security of our all-but-k constructions.

Lemma 3. *Protocol 3 is a system for noninteractive, trapdoor zero-knowledge arguments that, if Alice knows f such that $C = g^{f(\alpha)}$, then $\deg f \le k$. It is computationally convincing under the n-polyDH assumption.*

Proof Completeness for any $k \in [1, n]$ is clear from inspection of the protocol. The protocol is trapdoor zero-knowledge because a trapdoor simulator can trivially output $C' = C^{\alpha^{n-k}}$ given (C, n, k). If Alice knows f such that $C = g^{f(\alpha)}$ and $k < \deg f < n$, and C' such that $e(C, g^{\alpha^{n-k}}) = e(C', g)$, then she knows $f_k(x) = x^{n-k} f(x)$ such that $C' = g^{f_k(\alpha)}$ and $n < f_k < 2n$. Moreover, since $2n < \sqrt{q}$, we have that (f_k, C') is an n-polyDH tuple in $\mathbb{G}$ (cf. Definition 3). Under the n-polyDH assumption, Alice can exhibit such a C' with only negligible probability in τ; thus, the proof is computationally sound under the n-polyDH assumption when Alice knows f. □

D. Mercurial commitments

Chase et al. introduced *mercurial commitments* at Eurocrypt 2005 [8]. Mercurial commitments are a special type of commitment with a carefully relaxed binding property. A committer Alice can either *hard commit* to a value x or *soft commit* to no specific value. The two kinds of commitment look indistinguishable to a recipient Bob, but they have very different binding properties. A hard commitment is computationally binding in the traditional sense: if Alice hard commits to x, then she can later *hard open* or *soft open* the commitment to x and only to x. In contrast, a soft commitment is entirely nonbinding: if Alice soft commits, then she can later soft open (or *tease*) the commitment to an arbitrary value of her choosing (but she can never hard open it). Thus, when Alice soft opens a commitment to x, she effectively proves that "if this commitment can be hard opened at all. then it hard opens to x". A *mercurial vector commitment* scheme [7] is just a

mercurial commitment scheme that lets Alice hard commit to an arbitrary subset of components from some length-n vector in a single commitment. Alice can then soft open the commitment with respect to any position, and she can hard open it with respect to any position in which she initially hard committed. As in regular mercurial commitments, the binding guarantee is that a position-wise soft opening cannot contradict a position-wise hard opening at the same position.

III. All-but-k mercurial commitments

An *all-but-k mercurial commitment* scheme is similar to a mercurial vector commitment scheme. With all-but-k mercurial commitments, the committer Alice can commit to a collection of values $\mathcal{H}$ and later open that commitment to an arbitrary "supercollection" of $\mathcal{H}$ to which she has added up to k additional values. Exactly what it means to "add values" to $\mathcal{H}$ to form a "supercollection" of course depends on what sort of collection $\mathcal{H}$ is. We consider two basic collection types: ordered lists (i.e., *vectors*) and unordered lists (i.e., *multisets*).

A. Terminology and notation

We assume, without loss of generality, that all values to which Alice commits are from the interval $D = [0, d-1]$ for some fixed positive integer d. (To commit to elements from a finite commitment domain D' that is not of the above form, it suffices to set $d = |D'|$ and define an injective encoding function $\varphi\colon D' \to D$ that maps each element of D' to a representative element of $D = [0, d-1]$.)

Committing to multisets. If $\mathcal{H}$ and $\mathcal{S}$ are multisets over a common universe D, then $\mathcal{H} \uplus \mathcal{S}$ denotes the multiset sum of $\mathcal{H}$ and $\mathcal{S}$. We say that $\mathcal{H}$ is (D,n)-*committable* if it has universe D and cardinality at most n and that $(\mathcal{H},\mathcal{S})$ is (D,n,k)-*decommittable* if $\mathcal{S}$ has cardinality at most k and $\mathcal{H} \uplus \mathcal{S}$ has universe D and cardinality equal to n. We write $\mathcal{H} = \{(h_i, n_i) \mid i \in [1,m]\}$ to denote that $\mathcal{H}$ has m distinct elements $h_i \in D$ such that, for each $i \in [1,m]$, h_i occurs in $\mathcal{H}$ with multiplicity $n_i \in \mathbb{N}$. A polynomial $h(x) \in \mathbb{Z}_q[x]$ is a *polynomial representation* of $\mathcal{H} = \{(h_i, n_i) \mid i \in [1,m]\}$ if $h(x) = r \prod_{i=1}^{m} (x - h_i)^{n_i}$ for some $r \in \mathbb{Z}_q^*$. We denote the set of all polynomial representations of a set $\mathcal{H}$ by $\mathrm{PolyRep}(\mathcal{H})$. Note that $\deg h = |\mathcal{H}|$ if h is a polynomial representation of $\mathcal{H}$.

Committing to vectors. For committing to vectors $\vec{h}$ and $\vec{s}$, we introduce a *placeholder* element "$\star$" whose sum with any $a \in D$ (notably, with $a = 0$) we define to be a. We then define the sets $\mathcal{H} = \{i \mid h_i = \star\}$ and $\mathcal{S} = \{i \mid s_i = \star\}$, where h_i and s_i respectively denote the i^{th} components of $\vec{h}$ and $\vec{s}$. We say that $\vec{h}$ is (D,n)-*committable* if $\vec{h} \in (D \cup \{\star\})^n$ and that $(\vec{h},\vec{s})$ is (D,n,k)-*decommittable* if $\vec{h}$ and $\vec{s}$ are both (D,n)-committable, if $|\mathcal{H}| \le k$, and if $\mathcal{H} \cup \mathcal{S} = [1,n]$ and $\mathcal{H} \cap \mathcal{S} = \emptyset$. A pair of polynomials $(h(x), h'(x)) \in \mathbb{Z}_q[x] \times \mathbb{Z}_q[x]$ is a *polynomial representation* of $\vec{h}$ if $\deg h = n$, if $h(i) = h_i$ for every $i \in \mathcal{H}$, and if h' is a polynomial representation of $\mathcal{H}$. We denote the set of all polynomial representations of a (D,n)-committable vector $\vec{h}$ by $\mathrm{PolyRep}(\vec{h})$. Note that choosing $(h(x), h'(x)) \in_{\mathrm{R}} \mathrm{PolyRep}(\vec{h})$ is equivalent to choosing h such that $h(i) = h_i$ if $i \in [1,n] \setminus \mathcal{H}$ and $h(i) \in_{\mathrm{R}} \mathbb{Z}_q$ if $i \in \mathcal{H} \cup \{0\}$ and $h'(x) = r \prod_{i \in \mathcal{H}} (x - i)$ for $r \in_{\mathrm{R}} \mathbb{Z}_q^*$.

B. Formal definitions for all-but-k commitments

Definition 4 (All-but-k mercurial commitment scheme). *For a fixed collection type, let $\odot$ be the binary operator that maps two collections to a supercollection.*[2] *An* all-but-k mercurial commitment scheme *to such collections is a four-tuple of PPT algorithms:*

AllButK-Init(τ, d, n_0) takes as input a security parameter $\tau \in \mathbb{N}$, an upper bound $d \in \mathbb{N}$ for the commitment domain $D = [0, d-1]$, and a maximum cardinality $n_0 \in \mathbb{N}$ for a committable multiset. It outputs a common reference string PK suitable for committing to all-but-k elements of an n-element collection of values from D for any pair (n,k) of nonnegative integers with $k \le n$ and $n \le n_0$. Each of the other algorithms take PK as an implicit input.

AllButK-Commit$_{\mathrm{PK}}(n, H)$ takes as input a cardinality $n \le n_0$ and a (D,n)-committable collection H. It outputs an all-but-k commitment C and decommit information $\delta = (H, n, \ldots)$ for (C, H, n) suitable for opening C to $H \odot S$ for any collection S and $k \in \mathbb{N}$ such that (H,S) is (D,n,k)-decommittable.

AllButK-Open$_{\mathrm{PK}}(C, k, S, \delta)$ takes as input an all-but-k commitment C, an integer $k \in \mathbb{N}$, a collection S, and decommit information δ for (C,H,n). If (H,S) is (D,n,k)-decommittable, then it outputs a decommitment (C, O, k, ω), where $O = H \odot S$ and ω is a *witness* for (C, O, k); otherwise, it outputs $\perp$.

AllButK-Verify$_{\mathrm{PK}}(C, O, k, \omega)$ takes as input a decommitment (C, O, k, ω). It outputs "true" if ω is a witness for (C, O, k) and it outputs "false" otherwise.

Definition 5 (Position-wise opening). *An all-but-k mercurial commitment scheme* (AllButK-Init, AllButK-Commit, AllButK-Open, AllButK-Verify) *for vectors supports position-wise decommitment if it has the four additional PPT algorithms:*

AllButK-SoftOpen$_{\mathrm{PK}}(C, O, k, \omega, i)$ takes as input a decommitment (C, O, k, ω) and an index $i \in [1, |O|]$. It outputs a position-wise soft decommitment $(C, k, \omega, i, o_i, \pi)$, where $o_i \in D$ is the i^{th} component of O, and π is a noninteractive proof of knowledge of O such that ω is a witness for (C, O, k) and the i^{th} component of O is o_i.

[2] For vectors, we interpret $\odot$ as componentwise addition; for multisets we interpret it as multiset sum; with other collection types, the interpretation might differ.

$\textbf{AllButK-SoftVerify}_{\text{PK}}(\mathcal{C}, k, \omega, i, o_i, \pi)$ takes as input a position-wise soft decommitment $(\mathcal{C}, k, \omega, i, o_i, \pi)$. It outputs "true" if π proves that the committer knows O such that ω is a witness for $(\mathcal{C}, O, k)$ and the i^{th} component of O is o_i.

$\textbf{AllButK-HardOpen}_{\text{PK}}(\mathcal{C}, \delta, i)$ takes as input a commitment $\mathcal{C}$, decommit information $\delta = (H, n, \ldots)$, and an index $i \in [1, n]$. It outputs a position-wise hard decommitment $(\mathcal{C}, i, h_i, \omega_i)$, where $h_i \in D$ is the i^{th} components of H and ω_i is a witness for $(\mathcal{C}, i, h_i)$.

$\textbf{AllButK-HardVerify}_{\text{PK}}(\mathcal{C}, i, h_i, \omega_i)$ takes as input a position-wise hard decommitment $(\mathcal{C}, i, h_i, \omega_i)$. It outputs "true" if ω_i is a witness for $(\mathcal{C}, i, h_i)$ and it outputs "false" otherwise.

Definition 6 (**Secure all-but-k mercurial commitment scheme**). *An all-but-k commitment scheme is secure if it satisfies the following properties.*

1. **Correctness.** Let (H, S) be (D, n, k)-decommittable for $n \le n_0$, let $(\mathcal{C}, \delta) \leftarrow \textbf{AllButK-Commit}(H, |H \odot S|)$, and let $(\mathcal{C}, O, k, \omega) \leftarrow \textbf{AllButK-Open}(\mathcal{C}, k, S, \delta)$. Then $\textbf{AllButK-Verify}(\mathcal{C}, O, k, \omega) =$ "true".
2. **Hiding.** No adversary $\mathcal{A}$ can win the following indistinguishability game against the honest challenge C with a probability exceeding $(1/2) + \varepsilon(\tau)$:
   1. $\mathcal{A}$ chooses a positive integer $n \le n_0$, a nonnegative integer $k \le n$, and two (D, n, k)-decommittable pairs (H_0, S_0) and (H_1, S_1) such that $H_0 \odot S_0 = H_1 \odot S_1$, then sends $\left(PK, (H_0, S_0), (H_1, S_1), n, k\right)$ to C.
   2. C chooses $b \in_{\text{R}} \{0, 1\}$, computes $(\mathcal{C}, \delta) \leftarrow \textbf{AllButK-Commit}(H_b, n)$, and $(\mathcal{C}, O, k, \omega) \leftarrow \textbf{AllButK-Open}(\mathcal{C}, k, S_b, \delta)$, then sends ω to $\mathcal{A}$.
   3. $\mathcal{A}$ outputs b' and wins if and only if $b' = b$.

   If $\mathcal{A}$ can only run PPT algorithms, then hiding is *computational*; otherwise, it is *unconditional.*
3. **Binding.** For any adversary $\mathcal{A}$, if $\{\mathcal{C}, (O_i, k_i, \omega_i) \mid i \in [1, m]\} \leftarrow \mathcal{A}(\text{PK})$ and $\textbf{AllButK-Verify}(\mathcal{C}, O_i, k_i, \omega_i) =$ "true" for each $i \in [1, m]$, then with probability at least $\left(1 - \varepsilon(\tau)\right)^m$, there exist $H, S_1, \ldots, S_m$ such that, for all $i \in [1, m]$, the pair (H, S_i) is (D, n, k_i)-decommittable and $O_i = H \odot S_i$. If $\mathcal{A}$ can only run PPT algorithms, then binding is *computational*; otherwise, it is *unconditional.*

IV. Constructions

A. Multiset commitments from roots of polynomials

Our first construction implements all-but-k mercurial multiset commitments. It is provably secure under the DL, n-SDH, and n-polyDL assumptions in the random oracle model (a version with interactive opening/verification protocols would be secure in the standard model). The idea is quite simple. To hard commit to a multiset $\mathcal{H}$ we output a $\textbf{PolyCommit}_{\text{DL}}$ commitment $\mathcal{C}$ to a random polynomial representation $h(x) \in_{\text{R}} \text{PolyRep}(\mathcal{H})$. Suppose we want to open the commitment to a superset $\mathcal{H} \uplus \mathcal{S}$ of $\mathcal{H}$. If $h(x)$ has leading coefficient $r \in \mathbb{Z}_q^*$, we let $s(x) \in \text{PolyRep}(\mathcal{S})$ be the polynomial representation of $\mathcal{S}$ with leading coefficient $1/r$ and observe that $h(x)s(x) \in \text{PolyRep}(\mathcal{H} \uplus \mathcal{S})$ is the monic polynomial representation of $\mathcal{H} \uplus \mathcal{S}$. The prover reveals $\mathcal{D} = g^{s(\alpha)}$, $\mathcal{H} \uplus \mathcal{S}$, and $\Pi = \text{PK}\{s \mid \mathcal{D} = g^{s(\alpha)} \wedge \deg s \le k\}$. We use Fiat and Shamir's heuristic to make the above proof of knowledge non-interactive in the random oracle model [9].

Construction 1.

$\textbf{AllButK-Init}(\tau, d, n_0)$ *chooses* $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(\tau)$ *and outputs a* $\textbf{PolyCommit}_{DL}$ *common reference string* $\text{PK} = \langle (\mathbb{G}, q, g), g^{\alpha^i} \mid i \in [1, n_0] \rangle$ *if* $n_0 < \sqrt{q}/2$ *and* $q \ge d$*; otherwise, outputs* $\perp$.

$\textbf{AllButK-Commit}(n, \mathcal{H})$ *outputs* $\mathcal{C} = g^{h(\alpha)}$ *for a random polynomial representation* $h(x) \in_{\text{R}} \text{PolyRep}(\mathcal{H})$ *with leading coefficient* r *and decommit information* $\delta = (H, n, r)$ *if* $n \le n_0$ *and* $\mathcal{H}$ *is* (D, n)*-committable; otherwise, outputs* $\perp$.

$\textbf{AllButK-Open}(\mathcal{C}, k, \mathcal{S}, \delta)$ *outputs a witness* $\omega = (\mathcal{C}', \Pi, r \cdot r')$ *if* $n \le n_0$ *and* (H, S) *is* (D, n, k)*-decommittable, where* $\mathcal{C}' = g^{s(\alpha)}$ *for a random polynomial representation* $s(x)$ *of* $\mathcal{S}$ *with leading coefficient* $r' \in_{\text{R}} \mathbb{Z}_q^*$*, and* $\Pi = PK\{s \mid \mathcal{D} = g^{s(\alpha)} \wedge \deg s \le k\}$*; otherwise, it outputs* $\perp$.

$\textbf{AllButK-Verify}(\mathcal{C}, O, k, \omega)$ *computes* $g^{o(\alpha)}$ *for the polynomial representation* $o(x) \in \text{PolyRep}(O)$ *with leading coefficing* $r \cdot r'$*, then outputs "true" if* $e(g^{o(\alpha)}, g) \stackrel{?}{=} e(\mathcal{C}, \mathcal{C}')$ *and* Π *is correct.*

Theorem 1. *Construction 1 is a secure all-but-k mercurial multiset commitment scheme under the DL, n-SDH, and n-polyDH assumptions in the random oracle model.*

B. Vector commitments from evaluations of polynomials

Construction 2.

$\textbf{AllButK-Init}(\tau, d, n_0)$ *chooses* $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(\tau)$ *and outputs a* $\textbf{PolyCommit}_{DL}$ *common reference string* $\text{PK} = \langle (\mathbb{G}, q, g), g^{\alpha^i} \mid i \in [1, n_0] \rangle$ *if* $n_0 < \sqrt{q}$ *and* $q \ge d$*; otherwise, outputs* $\perp$.

$\textbf{AllButK-Commit}(n, \vec{h})$ *outputs a commitment* $\mathcal{C} = (g^{h(\alpha)}, g^{h'(\alpha)})$ *to a random polynomial representation* $\left(h(x), h'(x)\right) \in_{\text{R}} \text{PolyRep}(\vec{h})$ *and decommit information* $\delta = \left(\vec{h}, n, h(x), h'(x)\right)$.

$\textbf{AllButK-Open}(\mathcal{C}, k, \vec{s}, \delta)$ *outputs a witness* $\omega = (\mathcal{C}', \mathcal{D}, r, \pi)$ *if* $(\vec{h}, \vec{s})$ *is* (D, n, k)*-decommittable, where* $\mathcal{C}' = g^{s(\alpha)}$ *and* $s(i) = s_i - h(i)$ *for all* $i \in \mathcal{H}$*,* $\mathcal{D} = g^{t(\alpha)}$*,* $r = s(0) + h(0)$*, and* $t(x) = h'(x) \cdot s(x) / \prod_{i=1}^{n}(x - i)$*, and* $\pi = \text{PK}\{(s, t) : \mathcal{C} = g^{s(\alpha)} \wedge \mathcal{D} = g^{t(\alpha)} \wedge \deg s \le n_0 \wedge \deg t \le k\}$.

$\textbf{AllButK-Verify}(\mathcal{C}, \vec{o}, k, \omega)$ *computes* $g^{o(\alpha)}$ *for the polynomial representation* $o(x) \in \text{PolyRep}(\vec{o})$ *that has* $o(0) = r$*, and* $g^{z(\alpha)}$ *for* $z = \prod_{i=1}^{n}(x - i)$*, then outputs "true" if* $g^{o(\alpha)} \stackrel{?}{=} g^{h(\alpha)} g^{s(\alpha)}$*,* $e(g^{s(\alpha)}, g^{h'(\alpha)}) = e(g^{t(\alpha)}, g^{z(\alpha)})$*, and* π *is correct.*

Theorem 2. *Construction 2 is a secure all-but-k mercurial vector commitment scheme under the DL, n-SDH, and n-polyDH assumptions in the random oracle model.*

Optimization. Define $z_n(x) = \prod_{i=1}^{n}(x - i)$ and, for each $i \in [0,n]$, define the Lagrange coefficient $\lambda_i(x) = \prod_{j \neq i} \frac{x-j}{i-j}$ where the product is over $j \in [0,n] \setminus \{i\}$. Now consider the augmented common reference string $\mathrm{PK}' = \left(\mathbb{G}, q, g), g^{z(\alpha)}, g^{\lambda_i(\alpha)}, g^{\alpha^i} \mid i \in [1,n]\right\}$, which is of course computable from the standard reference string PK using $O(n^2\,\tau)$ multiplications in $\mathbb{G}$. We can write all commitment computations as a multiexponentiation in which all-but-k of the exponents is $\lg d$ bits long.

Theorem 3. *Using the above augmented reference string,* AllButK-Commit*,* AllButK-Open*, and* AllButK-Verify *in Construction 2 each have runtime in* $\Theta(n \lg d)$ *multiplications for any fixed k.*

V. Conclusion

We introduce and formally define *all-but-k mercurial commitments*, a new kind cryptographic commitment that generalizes standard mercurial and non-mercurial (vector) commitments. We provide two concrete constructions for all-but-k mercurial commitments: the first is for committing to *unordered lists* (i.e., to *multisets*) and the second is for committing to *ordered lists* (i.e., to *vectors*).

Please check back soon for a more detailed draft including proofs of correctness for the constructions and applications.

References

[1] Man Ho Au, Qianhong Wu, Willy Susilo, and Yi Mu. Compact e-cash from bounded accumulator. In *Proceedings of CT-RSA 2007*, volume 4377 of *LNCS*, pages 178–195, San Francisco, CA, USA, February 2007.

[2] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity-based encryption without random oracles. In *Proceedings of EUROCRYPT 2004*, pages 223–238, Interlaken, Switzerland, May 2004.

[3] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *Proceedings of EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73, Interlaken, Switzerland, May 2004.

[4] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology*, 21(2):149–177, April 2008.

[5] Stefan A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press, Cambridge, MA, USA, August 2000. First Edition.

[6] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups (extended abstract). In *Proceedings of CRYPTO 1997*, volume 1294 of *LNCS*, pages 410–424, Santa Barbara, CA, USA, August 1997.

[7] Dario Catalano, Dario Fiore, and Mariagrazia Messina. Zero-knowledge sets with short proofs. In *Proceedings of EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 433–450, Istanbul, Turkey, April 2008.

[8] Melissa Chase, Alexander Healy, Anna Lysyanskaya, Tal Malkin, and Leonid Reyzin. Mercurial commitments with applications to zero-knowledge sets. In *Proceedings of EUROCRYPT 2005*, volume 2494 of *LNCS*, pages 422–439, Aarhus, Denmark, May 2005.

[9] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Proceedings of CRYPTO 1986*, volume 263 of *LNCS*, pages 186–194, Santa Barbara, CA, USA, August 1986.

[10] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, September 2008.

[11] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *Proceedings of ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 177–194, Singapore, December 2010.

[12] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Polynomial commitments. Tech. Report CACR 2010-10, University of Waterloo, Waterloo, ON, Canada, December 2010.

[13] Benoît Libert and Moti Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In *Proceedings of TCC 2010*, volume 5978 of *LNCS*, pages 499–517, Zurich, Switzerland, February 2010.

[14] Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Boca Raton, FL, USA, October 1996. Fifth Printing, August 2001.

[15] Lan Nguyen. Accumulators from bilinear pairings and applications. In *Proceedings of CT-RSA 2005*, volume 3376 of *LNCS*, pages 275–292, San Francisco, CA, USA, February 2005.

[16] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Proceedings of CRYPTO 1991*, volume 576 of *LNCS*, pages 129–140, Santa Barbara, CA, USA, August 1991.

[17] Piotr Rudnicki. Little Bézout's theorem (factor theorem). *Formalized Mathematics*, 12(1):49–58, 2004.

Appendix A. Change log

No changes yet. Check back soon for a more complete draft.