Batching non-membership proofs with bilinear accumulators

Abstract

In this short paper, we provide a protocol to batch multiple non-membership proofs into a single proof of constant size with bilinear accumulators via a succinct argument of knowledge for polynomial commitments. We use the same techniques to provide a constant-sized proof that a polynomial commitment as in [KZG10] is a commitment to a square-free polynomial. In the context of the bilinear accumulator, this can be used to prove non-repetition in a committed set.

We have designed all of the protocols so that the Verifier needs a constant amount of storage for verification despite the linear public parameter. We also provide ways to speed up the verification of membership and non-membership proofs and to shift most of the computational burden from the Verifier to the Prover. Since all the challenges are public coin, the protocols can be made non-interactive with a Fiat-Shamir heuristic.

1 Introduction

A commitment scheme is a fundamental cryptographic primitive which is the digital analog of a sealed envelop. *Committing* to a message `m` is akin to putting `m` in the envelop. *Opening* the commitment is like opening the envelop and revealing the content within. Commitments are endowed with two basic properties. The *hiding* property entails that a commitment reveals no information about the underlying message. The *binding* property ensures that one cannot alter the message without altering the commitment.

A cryptographic *accumulator* is a succinct binding commitment to a set or a multiset. A Prover with access to the set/multiset can prove membership or non-membership of an element with a proof verifiable against the succinct commitment held by a Verifier. Accumulators have been used for many applications including accountable certificate management [BLL00, NN98], timestamping [Bd94], group signatures and anonymous credentials [CL02], computations on authenticated data [ABC+12], anonymous e-cash [STS99b, MGGR13a], privacy-preserving data outsourcing [Sla12], updatable signatures [PS14, CJ10], and decentralized bulletin boards [FVY14, GGM14].

In this paper, we study a class of accumulators that is based on bilinear pairings of elliptic curves. First introduced by Nguyen in [Ngu05], these accumulators have the major advantage over the better known accumulator of a Merkle tree in that membership proofs are of constant size and multiple membership proofs can be batched or aggregated together into a single proof constant-sized proof. Furthermore, it was shown in [DT08] that they also allow for non-membership proofs for elements outside the committed set. In this paper, we provide a protocol to prove non-membership of an arbitrarily large set with a constant-sized proof.

A Vector Commitment (VC) is a closely related primitive [CF13]. It provides the same functionality as an accumulator, but for an ordered list of elements rather than a set. A VC is a position binding commitment and can be opened at any position to a unique value with a short proof (preferably independent of the length of the vector). Subvector commitments [LM18] are VCs where a subset of the vector positions can be opened in a single short proof.

Since bilinear accumulators require groups far smaller than RSA groups for the same level of security, we expect them to be substantially faster than RSA accumulators when it comes to accumulation, generation of membership proofs (witnesses) and verification. Furthermore, the hardness assumptions that underpin bilinear accumulators are the same as those in pairing-based Snarks and are *arguably* less brittle than the hardness assumptions for hidden order groups.

In recent years, cryptographic accumulators have seen a growing interest as a potential alternative to Merkle trees for blockchains. In particular, substantial progress was made in the paper [BBF19] where the authors showed how to provide constant sized non-membership proofs for arbitrarily large sets of data elements in accumulators based on groups of unknown order. In this paper, we show that this is also possible for bilinear accumulators and subsequently construct a bilinear Vector Commitment with constant-sized openings and a linear public parameter.

Furthermore, we adapt techniques from [BBF19] and [Wes18] in the bilinear accumulator setting to speed up verifications of membership/non-membership proofs and to shift most of the computational and storage burdens from the Verifier to the Prover. In particular, we provide a protocol to reduce the Verifier's task of verifying membership proofs to a constant run time independent of the number of data elements to be batched.

1.1 Structure of the paper

In section 1, we primarily provide some background and notations for bilinear accumulators and the KZG polynomial commitment scheme, including the hardness assumptions that underpin these schemes. In section 2, we describe the protocol `PoE` for verifiable computation and the succinct argument of knowledge `PoE` along with the security proofs.

In section 3, we use the protocols from section 2 to provide a constant-sized non-membership proof for an arbitrarily large set with respect to the accumulated digest. In section 4, we use the protocols from section 2 to construct a protocol that succinctly demonstrates that a KZG polynomial commitment is a commitment to a separable (square-free) polynomial. In the context of the bilinear accumulator, this can be used to prove non-repetition of elements inserted into the accumulator.

In section 5, we describe vector commitment with constant-sized openings that hinges on the universal accumulator with constant-sized membership and non-membership proofs. We discuss a protocol to demonstrate a polynomial relation between two discrete logarithms in the appendix.

1.2 Notations and terminology

As usual, $\mathbb{F}_q$ denotes the finite field with q elements for a prime power q and $\overline{\mathbb{F}}_q$ denotes its algebraic closure. $\mathbb{F}_q^*$ denotes the cyclic multiplicative group of the non-zero elements of $\mathbb{F}_q$. For polynomials $f(X)$, $g(X) \in \mathbb{F}_q[X]$, we denote by $\mathbf{gcd}(f(X), g(X))$ the unique *monic* polynomial that generates the (principal) ideal of $\mathbb{F}_q[X]$ generated by $f(X)$ and $g(X)$. A polynomial in $\mathbb{F}_q[X]$ is said to be *separable* or *square-free* if it is not divisible in $\mathbb{F}_q[X]$ by the square of any irreducible polynomial. Since a finite field is a perfect field, $f(X)$ being separable in $\mathbb{F}_q[X]$ is equivalent to $f(X)$ beign separable in the algebraic closure $\overline{\mathbb{F}}_q[X]$.

Batching and aggregation: Following the terminology of [BBF19], we use the term *batching* for the action of creating a single membership (or non-membership) witness for multiple data elements. *Aggregation* refers to the action of creating a single membership or non-membership proof for the data elements using individual witnesses that have already been created.

Neither of these mechanisms is afforded by Merkle trees, which is a primary reason for exploring other families of cryptographic accumulators and Vector Commitments. We now briefly introduce pairings.

Definition 1.1. For abelian groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, a *pairing*

$$\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$$

is a map equipped with the following properties.
1. Bilinearity: $\mathbf{e}(x_1 + x_2, y_1 + y_2) = \mathbf{e}(x_1, y_2) \cdot \mathbf{e}(x_1, y_2) \cdot \mathbf{e}(x_2, y_1) \cdot \mathbf{e}(x_2, y_2)$
$\forall\ x_1, x_2 \in \mathbb{G}_1,\ y_1, y_2 \in \mathbb{G}_2$.
2. Non-degeneracy: The image of $\mathbf{e}$ is non-trivial.
3. Efficient computability.

In pairing-based cryptography, we typically work in settings where the groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are cyclic of order p for some 256-bit prime p so as to have a 128-bit security level. Such pairings $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ are classified into three types:
- Type I: $\mathbb{G}_1 = \mathbb{G}_2$.
- Type II: $\mathbb{G}_1 \neq \mathbb{G}_2$ but there is an *efficiently computable* isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.
- Type III: There is no *efficiently computable* isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.

In practice, the groups $\mathbb{G}_1, \mathbb{G}_2$ are cyclic subgroups of the p-torsion subgroup of some pairing-friendly elliptic curve over a prime field $\mathbb{F}_\ell$ and $\mathbb{G}_T$ is the group of p-th roots of unity in the algebraic closure $\overline{\mathbb{F}}_\ell$. The most commonly used instantiation is the optimal ate pairing on the BLS12-381 curve. This curve is defined over a prime field $\mathbb{F}_\ell$ of bitsize 381 and the group of $\mathbb{F}_\ell$-points has a prime divisor p of bitsize 255. The curve has embedding degree 12 with respect to p. The pairing is efficiently computable via Miller's algorithm.

1.3 Cryptographic assumptions

We state the computationally infeasible problems that the security of our constructions hinge on.

Assumption 1.1. n-strong Diffie Hellman assumption: *Let $\mathbb{G}$ be a cyclic group of prime order p generated by an element g, and let $s \in \mathbb{F}_p^*$. Any probabilistic polynomial-time algorithm that is given the set $\{g^{s^i} : 1 \leq i \leq n\}$ can output a pair $(a, g^{1/(s+a)}) \in \mathbb{F}_p^* \times \mathbb{G}$ with at most negligible probability.*

Assumption 1.2. Knowledge of exponent assumption (KEA):. *Let $\mathbb{G}$ be a cyclic group of prime order p generated by an element g, and let $s \in \mathbb{F}_p^*$. Suppose there exists a PPT algorithm $\mathcal{A}_1$ that given pairs $(h_1, h_1^s), \cdots, (h_n, h_n^s)$ in $\mathbb{G}^2$, outputs a pair $(c_1, c_2) \in \mathbb{G}^2$ such that $c_2 = c_1^s$. Then there exists a PPT algorithm $\mathcal{A}_2$ that, with overwhelming probability, outputs a vector $(x_1, \cdots, x_n) \in \mathbb{F}_p^n$ such that*

$$c_1 = \prod_{i=1}^{n} h_i^{x_i} \ \ , \ \ c_2 = \prod_{i=1}^{n} (h_i^s)^{x_i}$$

A special case of the KEA assumption is that given the elements $\{g^{s^i} :\ 0 \leq i \leq n\}$, if a PPT algorithm $\mathcal{A}_1$ is able to output a triplet $(c_1, c_2, f(X)) \in \mathbb{G} \times \mathbb{G} \times \mathbb{F}_p[X]$ with $\deg(f(X) \geq 1$ such that $c_2 = c_1^{f(s)}$, then there is a PPT algorithm $\mathcal{A}_2$ that with overwhelming probability, outputs a polynomial $e(X)$ such that

$$c_1 = g_1^{e(s)} \ ,\ c_2 = g_1^{e(s) \cdot f(s)}.$$

The KEA assumption implies that breaking the stong Diffie-Hellman is equivalent to compjuting the trapdoor s. A PPT adversary $\mathcal{A}$ that can compute an element w such that $w^{s+\alpha} = g_1$ can also generate a polynomial $e(X)$ such that

$$w = g_1^{e(s)} \quad , \quad g_1 = g_1^{e(s)\cdot s}$$

and hence, s is a zero of the polynomial $e(X) \cdot X - 1$. So $\mathcal{A}$ can use a PPT algorithm such as [KS98] to factorize $e(X) \cdot X - 1$ and extract s in expected polynomial time.

Assumption 1.3. *Let $\mathbb{G}$ be a cyclic group of prime order p generated by an element g, and let $s \in \mathbb{F}_p^*$. Any probabilistic polynomial-time algorithm that is given the set $\{g^{s^i} : 1 \leq i \leq n\}$ can output a pair $(f(X), g^{1/(s+a)}) \in \mathbb{F}_p[X]_{\deg \geq 1} \times \mathbb{G}$ such that*

$$w^{f(s)} = g$$

with at most negligible probability.

This assumption is stronger than the n-strong Diffie Hellman assumption. However, for cryptosystems that use the KEA assumption, they are equivalent.

Lemma 1.1. *The n-strong Diffie Hellman and KEA assumptions imply Assumption 1.3.*

Proof. We show that a PPT adversary $\mathcal{A}$ that breaks Assumption 1.3 can break the n-strong Diffie Hellman with overwhelming probability. Suppose $\mathcal{A}$ outputs a non-constant polynomial $f(X)$ of degree $k \geq 1$ and an element w such that $w^{f(s)} = g$. Write $f(X) = \sum_{i=0}^{k} c_i X^i$. Then

$$(w^{\sum_{i=1}^{k} c_i s^{i-1}})^s = g \cdot w^{-c_0}$$

and the KEA assumption implies that with overwhelming probability, $\mathcal{A}$ can output a polynomial $e(X)$ such that

$$w^{\sum_{i=1}^{k} c_i s^{i-1}} = g^{e(s)} \quad , \quad g \cdot w^{-c_0} = g^{e(s)\cdot s}.$$

So

$$g = \left(g^{(1-e(s)\cdot s)\cdot c_0^{-1}}\right)^{f(s)}$$

and hence, $f(s)(e(s)\cdot s - 1) = c_0$. Now, $\mathcal{A}$ can use the [KS98] algorithm to factorize the polynomial $f(X)(e(X)X - 1) - c_0$ in expected polynomial runtime. Hence, $\mathcal{A}$ can derive the integer s - thus breaking the n-strong Diffie Hellman asusmption - with overwhelming probability. □

1.4 Argument Systems

An argument system for a relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{W}$ is a triple of randomized polynomial time algorithms $(\texttt{PGen}, \mathcal{P}, \mathcal{V})$, where $\texttt{PGen}$ takes an (implicit) security parameter λ and outputs a common reference string (CRS) $\texttt{pp}$. If the setup algorithm uses only public randomness we say that the setup is transparent and that the CRS is unstructured. The prover $\mathcal{P}$ takes as input a statement $x \in X$, a witness $w \in W$, and the CRS $\texttt{pp}$. The verifier $\mathcal{V}$ takes as input $\texttt{pp}$ and x and after interactions with $\mathcal{P}$ outputs 0 or 1. We denote the transcript between the prover and the verifier by $\langle \mathcal{V}(\texttt{pp}, x), \mathcal{P}(\texttt{pp}, x, w) \rangle$ and write $\mathcal{V}\langle(\texttt{pp}, x), \mathcal{P}(\texttt{pp}, x, w)\rangle = 1$ to indicate that the verifier accepted the transcript. If $\mathcal{V}$ uses only public randomness we say that the protocol is *public coin.*

We now define soundness and knowledge extraction for our protocols. The adversary is modeled as two algorithms $\mathcal{A}_0$ and $\mathcal{A}_1$, where $\mathcal{A}_0$ outputs the instance $x \in \mathcal{X}$ after $\texttt{PGen}$ is run, and $\mathcal{A}_1$ runs the interactive protocol with the verifier using a state output by $\mathcal{A}_0$. In a slight deviation from the soundness definition used in statistically sound proof systems, we do not universally quantify over the instance x (i.e. we do not require security to hold for all input instances x). This is due to the fact that in the computationally-sound setting the instance itself may encode a trapdoor of the common reference string, which can enable the adversary to fool a verifier. Requiring that an efficient adversary outputs the instance x prevents this. In our soundness definition the adversary $\mathcal{A}_1$ succeeds if he can make the verifier accept when no witness for x exists.

Definition 1.2. *We say an argument system* $(\mathtt{PGen}, \mathcal{P}, \mathcal{V})$ *for a relation* $\mathcal{R}$ *is* ***complete*** *if for all* $(x, w) \in \mathcal{R}$,

$$\mathtt{Pr}\big[\langle \mathcal{V}(\mathtt{pp}, x)\ ,\ \mathcal{P}(\mathtt{pp}, w)\rangle\big) = 1\ :\ \mathtt{pp} \xleftarrow{\$} \mathtt{PGen}(\lambda)\big]\ =\ 1.$$

Definition 1.3. *We say an argument system* $(\mathtt{PGen}, \mathcal{P}, \mathcal{V})$ *is* ***sound*** *if* $\mathcal{P}$ *cannot forge a fake proof except with negligible probability.*

Definition 1.4. *We say a sound argument system is an* ***argument of knowledge*** *if for any polynomial time adversary* $\mathcal{A}$*, there exists an extractor* $\mathcal{E}$ *with access to* $\mathcal{A}$*'s internal state that can, with overwhelming probability, extract a valid witness whenever* $\mathcal{A}$ *is convincing.*

Definition 1.5. *An argument system is* ***non-interactive*** *if it consists of a single round of interaction between* $\mathcal{P}$ *and* $\mathcal{V}$.

The Fiat-Shamir heuristic ([FS87]) can be used to transform interactive public coin argument systems into non-interactive systems. Instead of the Verifier generating the challenges, this function is performed by a public hashing algorithm agreed upon in advance.

2 Bilinear accumulators

We describe the setup in this section. Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be cyclic groups of order p for some prime p such that there exists a pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ which is *bilinear*, *non-degenerate* and *efficiently computable*. Fix generators g_1, g_2 of the cyclic groups $\mathbb{G}_1, \mathbb{G}_2$ respectively. Then $\mathbf{e}(g_1, g_2)$ is a generator of $\mathbb{G}_T$. Unlike in the case of accumulators based on groups of unknown order, the (common) order of the groups is public. Instead, the secret trapdoor is an integer s in the range $[1, p-1]$. Unfortunately, the generation of the trapdoor requires a trusted setup, which can be partially mitigated by using a secure multi-party computation ([BGM17]).
For a data set $\mathcal{D} = \{d_1, \cdots, d_n\}$, we define the accumulated digest

$$\mathrm{Acc}(g_1, \mathcal{D}) := g_1^{\prod\limits_{d \in \mathcal{D}} (d+s)} \in \mathbb{G}_1.$$

For a subset $\mathcal{D}_0 \subseteq \mathcal{D}$, the witness for $\mathcal{D}_0$ is defined by

$$\mathrm{wit}(\mathcal{D}_0) := g_1^{\prod\limits_{d \in \mathcal{D} \setminus \mathcal{D}_0} (d+s)}\ \in\ \mathbb{G}_1.$$

The Verifier then verifies the equation

$$\mathrm{wit}(\mathcal{D}_0)^{\prod\limits_{d_0 \in \mathcal{D}_0} (d_0+s)} = \mathrm{Acc}(g_1, \mathcal{D}).$$

via the pairing check

$$\mathbf{e}\Big(\mathrm{wit}(\mathcal{D}_0)\ ,\ g_2^{\prod\limits_{d_0 \in \mathcal{D}_0} (d_0+s)}\Big) = \mathbf{e}(\mathrm{Acc}(g_1, \mathcal{D}), g_2).$$

Since none of the parties are aware of the value of s, it is necessary to broadcast the sets $\{g_1, g_1^s, \cdots, g_1^{s^n}\}$ and $\{g_2, g_2^s, \cdots, g_2^{s^n}\}$ to the Prover.

The exponent $\prod\limits_{d \in \mathcal{D}} (d+s)$ can be interpreted as a degree n polynomial in the variable s. The coefficients of the polynomial are computed with a run time of $\mathbf{O}(n \log(n))$ using the Fast Fourier transform. Furthermore, the set $\mathbb{F}_p[X]$ of polynomials with $\mathbb{F}_p$-coefficients is a principal ideal domain whose maximal ideals are those generated by the irreducible polynomials. For a data set $\mathcal{D}$, the polynomial $f(X) = \prod\limits_{d \in \mathcal{D}} (X+d)$ is monic of degree $n = |\mathcal{D}|$. Let c_i denote the coefficient of X^i, i.e. $f(X) = \sum\limits_{i=0}^{n} c_i X^i$. The coefficients can be computed in run time $\mathbf{O}(n \log(n))$ using the Fast Fourier transform. The elements

$$g_1^{f(s)} = \prod_{i=0}^{n} (g_1^{s^i})^{c_i} \ \in \mathbb{G}_1 \ , \ \ g_2^{f(s)} = \prod_{i=0}^{n} (g_2^{s^i})^{c_i} \ \in \mathbb{G}_2$$

can then be computed by any party that possesses the elements $\{g_1^{s^i}, g_2^{s^i} \ : \ 0 \leq i \leq n\}$. The collision-resistance of the bilinear accumulator hinges on the n-Strong Diffie Hellman assumption.

The straightforward approach would be for the Verifier to compute the element

$$g_2^{\prod\limits_{d_0 \in \mathcal{D}_0} (d_0+s)} \ \in \mathbb{G}_2$$

in order to verify the equation

$$\mathbf{e}\big(\mathrm{wit}(\mathcal{D}_0), g_2^{\prod\limits_{d_0 \in \mathcal{D}_0} (d_0+s)}\big) = \mathbf{e}(\mathrm{Acc}(g_1, \mathcal{D}), g_2) \ \in \ \mathbb{G}_T.$$

However, this involves computing the coefficients c_i $(0 \leq i \leq |\mathcal{D}_0|)$ of the polynomial $\prod_{d_0 \in \mathcal{D}_0}(d_0 + s)$. The fastest known algorithm (the Fast Fourier transform) for this has run time complexity $\mathbf{O}(|\mathcal{D}_0| \log(|\mathcal{D}_0|))$ - followed by the exponentiations $(g_2^{s^i})^{c_i}$ that have run times $\mathbf{O}(\log(c_i))$. Furthermore, this straightforward verification mechanism makes it necessary for the Verifier to possess the entire set $\{g_1, g_1^s, \cdots, g_1^{s^n}\}$ which is not ideal. In the next subsection, we discuss protocols that allow us to sidestep this problem.

2.1 The protocols `PoE` and `PoKE`

In this subsection, we provide the protocols `PoE` and `PoKE` for bilinear accumulators which achieve three goals.

1. They speed up the verification process by replacing some exponentiation operations by polynomial division in $\mathbb{F}_p[X]$ which is substantially cheaper.

2. They shift most of the computational burden from the Verifier to the Prover. This is useful in settings where the Prover has more computational power at his disposal.

3. They reduce the Verifier's storage burden to the set $\{g_1, g_1^s, g_2, g_2^s\}$. This is potentially useful in settings where the Verifier has a low storage capacity.

Protocol 2.1. *Proof of exponentiation with base g_1 (`PoE`*):*

Parameters : A pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ of groups of prime order p; generators g_1, g_2 of $\mathbb{G}_1$, $\mathbb{G}_2$ respectively; a secret element $s \in \mathbb{F}_p^*$ such that the Prover possesses the elements $\{g_1^{s^i}, g_2^{s^i} \ : \ 0 \leq i \leq n\}$

Inputs: $a \in \mathbb{G}_1$; a polynomial $f(X) \in \mathbb{F}_p[X]$ of degree $\leq n$

Claim: $g_1^{f(s)} = a$

1. The Fiat-Shamir heuristic generates a challenge $\alpha \in \mathbb{F}_p^*$ (the challenge).

2. The Prover computes a polynomial $h(X) \in \mathbb{F}_p[X]$ and an element $\beta \in \mathbb{F}_p^*$ such that

$$f(X) = (X + \alpha)h(X) + \beta$$

and sends $Q := g_1^{h(s)}$ to the Verifier.

3. The Verifier computes $\beta := f(X) \pmod{(X+\alpha)}$ and accepts if and only if the equation

$$\mathbf{e}(Q, g_2^{s+\alpha}) \cdot \mathbf{e}(g_1^{\beta}, g_2) \overset{?}{=} \mathbf{e}(a, g_2)$$

holds. □

We refer to this as $\texttt{PoE}^*[g_1,\ f(X),\ a]$. The proof consists of a single element of $\mathbb{G}_1$ and in particular, is of constant size. Note that because of the bilinearity of the pairing, we have

$$Q^{s+\alpha} a^{\beta} = b \Longleftrightarrow \mathbf{e}(Q, g_2^{s+\alpha}) \cdot \mathbf{e}(a, g_2^{\beta}) = \mathbf{e}(b, g_2).$$

Thus, the pairing check is equivalent to verifying the equation $Q^{s+\alpha}a^{\beta} = b$.

Although the *asymptotic* complexity remains unchanged, this protocol swaps exponentiation operations in the group $\mathbb{G}_1$ with polynomial division operations in $\mathbb{F}_p[X]$ which are substantially cheaper. The most obvious application is that a Prover can use the protocol $\texttt{PoE}^*$ to convince a Verifier that an element $A \in \mathbb{G}_1$ is the accumulated digest $\mathrm{Acc}(g_1, \mathcal{D})$ of a data set $\mathcal{D}$. The Verifier just needs the four points $\{g_1, g_1^s, g_2, g_2^s\}$ to check the veracity of this claim.

Clearly, the protocol can be modified for the proof of an exponentiation $g_2^{f(s)} = b$ in the group $\mathbb{G}_2$. In this case, the proof would consist of the $\mathbb{G}_2$ element $g_2^{h(s)}$. We refer to this as $\texttt{PoE}^*[g_2,\ f(X),\ b]$.

Proposition 2.2. *The protocol* $\texttt{PoE}^*$ *is sound under the strong Diffie Hellman and KEA assumptions.*

Proof. We consider the case where the exponentiation is in $\mathbb{G}_1$ and with base g_1. The case where the exponentiation is in $\mathbb{G}_2$ and with base g_2 is virtually identical.

Suppose a PPT adversary $\mathcal{A}$ is able to output an accepting transcript $Q \in \mathbb{G}_1$ such that

$$\mathbf{e}(Q, g_2^{s+\alpha}) \cdot \mathbf{e}(g_1^{\beta}, g_2) = \mathbf{e}(a, g_2) \quad , \quad \beta := f(X) \pmod{(X+\alpha)}$$

in response to a challenge α. The pairing check implies that $Q^{s+\alpha} \cdot g_1^{\beta} = a$. The KEA assumption implies that with overwhelming probability, $\mathcal{A}$ can output a polynomial $h(X)$ such that

$$Q = g_1^{h(s)} \quad , \quad a = g_1^{h(s)(s+\alpha)+\beta}.$$

Setting $e(X) := h(X)(X+\alpha) + \beta$ yields $g_1^{e(s)} = a$. Now,

$$e(X) \equiv \beta \equiv f(X) \pmod{(X+\alpha)}$$

and since α was randomly and uniformly sampled from $\mathbb{F}_p$, it follows that with overwhelming probability, $e(X) = f(X)$. □

We now generalize this to bases $a \in \mathbb{G}_1$ other than g_1. We provide two versions. The second is more efficient (for the Prover) if there is the Prover knows a polynomial $e(X)$ of a small degree such that $g_1^{e(s)} = a$. The first is more efficient in all other cases.

Protocol 2.3. *Proof of exponent 1 for pairings* ($\texttt{PoE}-\texttt{1}$)*:*
Inputs: $a, b \in \mathbb{G}_1$; a polynomial $f(X) \in \mathbb{F}_p[X]$ of degree $\leq n$
Claim: $a^{f(s)} = b$

1. The Prover $\mathcal{P}$ sends the element $\widetilde{g}_2 := g_2^{f(s)} \in \mathbb{G}_2$ to the Verifier $\mathcal{V}$.
2. $\mathcal{P}$ sends a non-interactive proof for $\texttt{PoE}^*[g_2,\ f(X),\ \widetilde{g}_2]$ to $\mathcal{V}$.

3. $\mathcal{V}$ verifies the proof for $\texttt{PoE}^*[g_2,\ f(X),\ \widetilde{g}_2]$ and the pairing

$$\mathbf{e}(a, g_2) \overset{?}{=} \mathbf{e}(b, \widetilde{g}_2).$$

$\mathcal{V}$ accepts if and only if the pairing check holds and the $\texttt{PoE}^*$ is valid.

Proposition 2.4. *The protocol* $\texttt{PoE} - \texttt{1}$ *is secure under the strong Diffie Hellman and KEA assumptions.*

Proof. Suppose a PPT adversary $\mathcal{A}$ is able to output an element $\widetilde{g}_2 \in \mathbb{G}_2$ such that $\mathbf{e}(a, \widetilde{g}_2) = \mathbf{e}(b, g_2)$ along with a proof for $\texttt{PoE}^*[g_2,\ f(X),\ \widetilde{g}_2]$. The $\texttt{PoE}^*$ implies that with overwhelming probability, $g_2^{f(s)} = \widetilde{g}_2$. The pairing check then implies that the discrete logarithms between the pairs $(g_2, \widetilde{g}_2) \in \mathbb{G}_2$ and $(a, b) \in \mathbb{G}_1$ coincide and hence, $a^{f(s)} = b$. □

When the pairing is type III, the exponentiations in $\mathbb{G}_2$ are substantially more expensive than those in $\mathbb{G}_1$. Thus, in cases where the Prover possesses a polynomial $e(X)$ of a small degree such that $a = g_1^{e(s)}$, it can be cheaper to compute the element $a^{h(s)} = g_1^{e(s)\cdot h(s)}$ instead of $g_2^{h(s)}$.

Protocol 2.5. *Proof of exponent 2 for pairings* ($\texttt{PoE} - \texttt{2}$)*:*
Parameters : A pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ of groups of prime order p; generators g_1, g_2 of $\mathbb{G}_1$, $\mathbb{G}_2$ respectively; a secret element $s \in \mathbb{F}_p^*$ such that the Prover possesses the elements $\{g_1^{s^i}, g_2^{s^i} \ :\ 0 \leq i \leq n\}$
Inputs: $a, b \in \mathbb{G}_1$; a polynomial $f(X) \in \mathbb{F}_p[X]$ of degree $\leq n$
Claim: $a^{f(s)} = b$

1. The Fiat-Shamir heuristic generates a challenge $\alpha \in \mathbb{F}_p^*$ (the challenge).
2. The Prover computes a polynomial $h(X) \in \mathbb{F}_p[X]$ and an element $\beta \in \mathbb{F}_p^*$ such that

$$f(X) = (X + \alpha)h(X) + \beta$$

and sends $Q := a^{h(s)}$ to the Verifier.

3. The Verifier computes $\beta := f(X) \pmod{(X + \alpha)}$ and accepts if and only if the equation

$$\mathbf{e}(Q, g_2^{s+\alpha}) \cdot \mathbf{e}(a, g_2^{\beta}) \overset{?}{=} \mathbf{e}(b, g_2)$$

holds. □

We refer to these protocol as $\texttt{PoE} - \texttt{1}[a\ , f(X),\ b]$ and $\texttt{PoE} - \texttt{1}[a\ , f(X),\ b]$ respectively. We use the notation $\texttt{PoE}[a\ , f(X),\ b]$ to mean one of these two versions.

Proposition 2.6. *The protocol* $\texttt{PoE} - \texttt{2}$ *for bilinear accumulators is secure under the* n*-strong Diffie Hellman and KEA assumptions.*

Proof. Suppose, by way of contradiction, that a PPT adversary $\mathcal{A}$ produces fake witnesses Q_1, Q_2 in response to challenges α_1, α_2. Then Q_i satisfies the equation

$$Q_i^{s+\alpha} = b \cdot a^{-\beta_i} \quad , \quad \beta_i := f(X) \pmod{(X + \alpha_i)} \quad (i = 1, 2)$$

The KEA assumption implies $\mathcal{A}$ can output polynomials $e_1(X)$, $e_2(X)$ such that

$$Q_i = g_1^{e_i(s)} \quad , \quad b \cdot a^{-\beta_i} = g_1^{e_i(s)(s+\alpha_i)} \quad , \quad \beta_i \equiv f(X) \pmod{(X + \alpha_i)}.$$

For brevity, we write
$f_1(X) := (\beta_2 - \beta_1)^{-1} \cdot [e_1(X)(X+\alpha_1) - e_2(X)(X+\alpha_2)]$, $f_2(X) := e_1(X)(X+\alpha_1) + \beta_1 \cdot f_1(X)$.
So $a = g_1^{f_1(s)}$, $b = g_1^{f_2(s)}$. Furthermore, the equation

$$g_1^{f_2(s)} = b = Q_1^{s+\alpha_1} \cdot a^{\beta_1} = Q_1^{s+\alpha_1} \cdot g_1^{\beta_1 \cdot f_1(s)}$$

and the strong Diffie Hellman assumption imply that with overwhelming probability,

$$f_2(X) \equiv f_1(X) \cdot \beta_1 \equiv f_1(X) \cdot f(X) \pmod{(X+\alpha_1)}.$$

Since α_1 was randomly and uniformly sampled from $\mathbb{F}_p^*$, it follows that with overwhelming probability, $f_2(X) = f_1(X) \cdot f(X)$. □

We use the protocol PoE to modify the proof of membership for a data set. The goal is to reduce the storage and compuational burdens of the Verifier.

Protocol 2.7. *Protocol for set membership.*
Parameters : A pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ of groups of prime order p; generators g_1, g_2 of $\mathbb{G}_1$, $\mathbb{G}_2$ respectively; a trapdoor $s \in \mathbb{F}_p^*$ such that the Prover possesses the elements $\{g_1^{s^i}, g_2^{s^i} : 0 \leq i \leq n\}$ and the Verifier possesses the set $\{g_1, g_2, g_1^s, g_2^s\}$
Inputs: Data sets $\mathcal{D}$, $\mathcal{D}_0$; the accumulated digest $\mathrm{Acc}(D)$
Claim: $\mathcal{D}_0 \subseteq \mathcal{D}$.

1. The Prover computes the polynomial $f_0(X) := \prod_{d_0 \in \mathcal{D}_0} (X + d_0)$.
2. The Prover computes
$$\mathrm{wit}(\mathcal{D}_0) := g_1^{\prod_{d \in \mathcal{D} \setminus \mathcal{D}_0} (d+s)} \in \mathbb{G}_1$$
and sends it to the Verifier $\mathcal{V}$.
3. The Prover sends the Verifier a non-interactive proof of $\mathtt{PoE}[\mathrm{wit}(\mathcal{D}_0),\ f_0(X),\ \mathrm{Acc}(g_1, \mathcal{D})]$.
4. The Verifier computes $f_0(X)$ and accepts if and only if the PoE is valid. □

Thus, the proof of membership can be verified by a Verifier who possesses the set $\{g_1, g_1^s, g_2, g_2^s\}$. We next show how the last protocol can be adapted to provide an argument of knowledge of the logarithm. The goal is to construct a protocol with communication complexity much lower than simply sending the polynomial to the Verifier. This will be the key ingredient for batching non-memberships with a constant-sized proof.

Protocol 2.8. *Proof of knowledge of the exponent with base* g_1 (PoKE*):
Parameters: A pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$; generators g_1, g_2 for $\mathbb{G}_1, \mathbb{G}_2$ respectively.
Inputs: Element $a \in \mathbb{G}_1$
Claim: The Prover possesses a polynomial $f(X) \in \mathbb{F}_p[X]$ such that $g_1^{f(s)} = a$.

1. The Fiat-Shamir heuristic generates a challenge $\alpha \in \mathbb{F}_p^*$.
2. $\mathcal{P}$ computes the polynomial $h(X) \in \mathbb{F}_p[X]$ and the element $\beta \in \mathbb{F}_p$ such that
$$f(X) = (X + \alpha)h(X) + \beta.$$

$\mathcal{P}$ computes $Q := g_1^{h(s)}$ and sends $(Q, \beta) \in \mathbb{G}_1 \times \mathbb{F}_p$ to $\mathcal{V}$.

3. $\mathcal{V}$ then verifies the equations

$$\mathbf{e}(Q, g_2^{s+\alpha}) \cdot \mathbf{e}(g_1^{\beta}, g_2) \overset{?}{=} \mathbf{e}(a, g_2)$$

and accepts if and only if both equations hold. □

The proof consists of an element of $\mathbb{G}_1$ and an element of $\mathbb{F}_p$. We refer to this as $\texttt{PoKE}^*[g_1,\ a]$. We note that multiple $\texttt{PoKE}^*$s can be batched together. For elements $a_1, \cdots, a_k \in \mathbb{G}_1$, a Prover can demonstrate knowledge of polynomials $f_i(X)$ such that $g_1^{f_i(s)} = a_i$ by sending a proof for $\texttt{PoKE}^*[g_1,\ \prod_{i=1}^{k} a_i^{\gamma^i}]$ in response to a randomly generated challenge $\gamma \in \mathbb{F}_p$. This proof is constant-sized and independent of the number of polynomials or their degrees.

Clearly, the protocol can be modified for the proof of the knowledge of an exponent $g_2^{f(s)} = b$ in $\mathbb{G}_2$. In this case, the proof would consist of an element of $\mathbb{G}_2$ and an element of $\mathbb{F}_p$. We refer to this as $\texttt{PoKE}^*[g_2,\ b]$.

Proposition 2.9. *The protocol* $\texttt{PoKE}^*$ *is an argument of knowledge under the strong Diffie Hellman and KEA assumptions.*

Proof. We address the case where the exponentiation is in $\mathbb{G}_1$ and with base g_1. The case where the exponentiation is in $\mathbb{G}_2$ and with base g_2 is identical. We first show that the protocol is sound and then demonstrate witness extractability.

Suppose a PPT adversary $\mathcal{A}$ is able to output an accepting transcript $(Q, \beta) \in \mathbb{G}_1 \times \mathbb{F}_p$ such that $\mathbf{e}(Q, g_2^{s+\alpha}) \cdot \mathbf{e}(g_1^{\beta}, g_2) = \mathbf{e}(a, g_2)$ in response to a challenge α. The pairing check implies that $Q^{s+\alpha} \cdot g_1^{\beta} = a$. The KEA assumption implies that with overwhelming probability, $\mathcal{A}$ can output a polynomial $h(X)$ such that

$$Q = g_1^{h(s)} \quad , \quad a = g_1^{h(s)(s+\alpha)+\beta}.$$

Setting $f(X) := h(X)(X+\alpha) + \beta$ yields $g_1^{f(s)} = a$, which completes the proof.

We now demonstrate witness extractability to show that this is an argument of knowledge. An extractor $\mathcal{E}$ with access to the accepting transcripts and to the CRS proceeds as follows. Given accepting transcripts (Q_i, β_i) for challenges α_i $(i = 1, \cdots, N)$, $\mathcal{E}$ uses the Chinese remainder theorem to compute a polynomial $e_N(X)$ such that

$$e_N(X) \equiv \beta \pmod{(X + \alpha_i)} \quad , \quad i = 1, \cdots, N.$$

If $g_1^{e_N(s)} = a$, $\mathcal{E}$ halts. Otherwise, $\mathcal{E}$ samples the next accepting transcript (Q_{N+1}, β_{N+1}) and computes the polynomial $e_{N+1}(X)$ such that

$$e_{N+1}(X) \equiv e_N(X) \pmod{\prod_{i=1}^{N}(X + \alpha_i)} \quad , \quad e_{N+1}(X) \equiv \beta_{N+1} \pmod{(X + \alpha_{N+1})}.$$

via the Chinese remainder theorem. When the number of accepting transcripts sampled exceeds the degree of $f(X)$, the polynomial obtained by $\mathcal{E}$ is $f(X)$ with overwhelming probability. □

We now generalize this to bases $a \in \mathbb{G}_1$ other than g_1. We provide two versions. The second is more efficient (for the Prover) if the Prover knows a polynomial $e(X)$ of a small degree such that $g_1^{e(s)} = a$. The first is more efficient in all other cases. We will use $\texttt{PoKE}^*$ as a subprotocol for $\texttt{PoKE} - \texttt{1}$.

Protocol 2.10. *Proof of knowledge of the exponent for bilinear accumulators* ($\texttt{PoKE}-\texttt{1}$):
Parameters: A pairing $\mathbf{e}: \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$; generators g_1, g_2 for $\mathbb{G}_1, \mathbb{G}_2$ respectively.
Inputs: Elements $a, b \in \mathbb{G}_1$
Claim: The Prover possesses a polynomial $f(X) \in \mathbb{F}_p[X]$ such that $a^{f(s)} = b$.

1. The Prover $\mathcal{P}$ computes $\widetilde{g}_2 := g_2^{f(s)} \in \mathbb{G}_2$ and sends it to the Verifier $\mathcal{V}$.
2. $\mathcal{P}$ sends a non-interactive proof for $\texttt{PoKE}^*[g_2,\ \widetilde{g}_2]$.
3. $\mathcal{V}$ verifies the proof for $\texttt{PoKE}^*[g_2,\ \widetilde{g}_2]$ and the equation

$$\mathbf{e}(a, \widetilde{g}_2) \overset{?}{=} \mathbf{e}(b, g_2).$$

$\mathcal{V}$ accepts if and only if the $\texttt{PoKE}^*$ is valid and the pairing equation holds. □

Clearly, a virtually identical proof would work if (a, b) was a pair in $\mathbb{G}_2$ instead of $\mathbb{G}_1$. Henceforth, we refer to this succinct proof as $\texttt{PoKE}-\texttt{1}[a,\ b]$ for a pair (a, b) in $\mathbb{G}_1^2$ or $\mathbb{G}_2^2$.

Proposition 2.11. *The protocol* $\texttt{PoKE}-\texttt{1}$ *is an argument of knowledge under the strong Diffie Hellman and KEA assumptions.*

Proof. We consider the case where a, b are elements of $\mathbb{G}_1$. The case where they are elements of $\mathbb{G}_2$ is virtually identical.

Suppose a PPT adversary $\mathcal{A}$ is able to output an element $\widetilde{g}_2 \in \mathbb{G}_2$ such that $\mathbf{e}(b, g_2) = \mathbf{e}(a, \widetilde{g}_2)$ along with a proof for $\texttt{PoKE}^*[g_2,\ \widetilde{g}_2]$. The $\texttt{PoKE}^*$ implies that with overwhelming probability, $\mathcal{A}$ can output a polynomial $f(X)$ such that $g_2^{f(s)} = \widetilde{g}_2$. The pairing check implies that the discrete logarithms between $g_2, \widetilde{g}_2$ and a, b coincide and hence, $a^{f(s)} = b$.

An extractor $\mathcal{E}$ can simulate the extractor for $\texttt{PoKE}^*[g_2,\ \widetilde{g}_2]$ to extract the polynomial $f(X)$ in polynomial expected time. □

When the pairing is type III, the exponentiations in $\mathbb{G}_2$ are substantially more expensive than those in $\mathbb{G}_1$. Thus, in cases where the Prover possesses a polynomial $e(X)$ of a *small* degree such that $a = g_1^{e(s)}$, it can be cheaper to compute the element $a^{h(s)} = g_1^{e(s)\cdot h(s)} \in \mathbb{G}_1$ instead of $g_2^{h(s)} \in \mathbb{G}_2$.

Protocol 2.12. *Proof of knowledge of the exponent for bilinear accumulators* ($\texttt{PoKE}-\texttt{2}$):
Parameters: A pairing $\mathbf{e}: \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$; generators g_1, g_2 for $\mathbb{G}_1, \mathbb{G}_2$ respectively.
Inputs: Elements $a, b \in \mathbb{G}_1$
Claim: The Prover possesses a polynomial $f(X) \in \mathbb{F}_p[X]$ such that $a^{f(s)} = b$.

1. The Prover $\mathcal{P}$ computes $\widetilde{g} := g_1^{f(s)} \in \mathbb{G}_1$ and sends it to the Verifier $\mathcal{V}$.
2. The Fiat-Shamir heuristic generates a challenge $\alpha \in \mathbb{F}_p^*$.
3. $\mathcal{P}$ computes the polynomial $h(X) \in \mathbb{F}_p[X]$ and the element $\beta \in \mathbb{F}_p$ such that

$$f(X) = (X + \alpha)h(X) + \beta.$$

$\mathcal{P}$ computes

$$Q := a^{h(s)} \ \ , \ \ \widehat{g} := g_1^{h(s)}$$

and sends $(Q, \widehat{g}, \beta) \in \mathbb{G}_1^2 \times \mathbb{F}_p^*$ to $\mathcal{V}$.

4. $\mathcal{V}$ verifies the equations

$$\mathbf{e}(Q, g_2^{s+\alpha}) \cdot \mathbf{e}(a^{\beta}, g_2) \overset{?}{=} \mathbf{e}(b, g_2) \bigwedge \mathbf{e}(\widehat{g}, g_2^{s+\alpha}) \cdot \mathbf{e}(g_1^{\beta}, g_2) \overset{?}{=} \mathbf{e}(\widetilde{g}, g_2)$$

and accepts if and only if both equations hold. □

Clearly, a virtually identical proof would work if (a, b) was a pair in $\mathbb{G}_2$ instead of $\mathbb{G}_1$. Henceforth, we refer to this succinct proof as $\texttt{PoKE}-2[a,\ b]$ for a pair (a, b) in $\mathbb{G}_1^2$ or $\mathbb{G}_2^2$.

We now describe two attacks to show that the Protocol PoKE-2 needs both of the steps 1 and 3.

Attack 1: Suppose the pairing $\mathbf{e}$ is a type I pairing, i.e. $\mathbb{G}_1 = \mathbb{G}_2$. Suppose a malicious Prover $\mathcal{P}_{\text{mal}}$ possesses a polynomial $h(X) \in \mathbb{F}_p[X]$ such that $a^{h(s)} = g_2$. $\mathcal{P}_{\text{mal}}$ computes $Q' := b^{h(s)}$ and sends it to the Verifier. The Verifier then sees that $\mathbf{e}(a, Q') = \mathbf{e}(b, g_2)$ and is tricked into believing the veracity of the Prover's claim. Thus, this protocol is not secure for type I pairings. Clearly, a similar attack can be carried out for type II pairings as well.

Attack 2: Suppose a Prover $\mathcal{P}_{\text{mal}}$ possesses polynomials $h_1(X)$, $h_2(X)$ such that $g_1^{h_1(s)} = a$, $g_1^{h_2(s)} = b$ and $h_1(X)$ does not divide $h_2(X)$. With overwhelming probability, the challenge $\alpha \in \mathbb{F}_p^*$ is such that the polynomials $X + \alpha$ and $h_1(X)$ are relatively prime. On receiving the challenge α, $\mathcal{P}_{\text{mal}}$ could simply compute a polynomial $q(X) \in \mathbb{F}_p$ and an element $\beta \in \mathbb{F}_p$ such that

$$h_1(X)\beta + (X + \alpha)q(X) = h_2(X)$$

and send $Q := a^{q(s)}$, β to the Verifier. The Verifier then sees that $Q^{s+\alpha}a^{\beta} = b$ and is tricked into believing that the Prover possesses a polynomial $f(X)$ such that $a^{f(s)} = b$.

Note that when $h_1(X)$ divides $h_2(X)$, this does not constitute an attack since

$$a^{h_2(s)/h_1(s)} = b.$$

But in the case where $h_1(X)$ does not divide $h_2(X)$, this attack shows that it is not sufficient for the Prover to send the pair $(Q, \beta) \in \mathbb{G}_1 \times \mathbb{F}_p$ to the Verifier. To address this, we require the Prover to send the element $\widetilde{g} := g_1^{f(s)}$ *before* the challenge α is generated by the Fiat-Shamir heuristic.

In the case of protocol $\texttt{PoKE}-1$, the subprotocol $\texttt{PoKE}^*[g_2,\ \widetilde{g}_2]$ implies that the Prover possesses a polynomial $f(X)$ such that $g_2^{f(s)} = \widetilde{g}_2$ and the pairing check

$$\mathbf{e}(a, \widetilde{g}_2) = \mathbf{e}(b, g_2)$$

ensures that $a^{f(s)} = b$.

Proposition 2.13. *The protocol* $\texttt{PoKE}-2$ *is an argument of knowledge under the n-strong Diffie Hellman and KEA assumptions.*

Proof. Suppose a PPT adversary $\mathcal{A}$ is able to output accepting transcripts $(\widetilde{g}, Q_i, \widehat{g}_i, \beta_i)$ $(i = 1, 2)$ for challenges α_1, α_2 generated after $\widetilde{g}$ has been sent. Via the pairing checks, the Verifier verifies the equations

$$Q_i^{s+\alpha_i} = b \cdot a^{-\beta_i} \ \ , \ \ \widehat{g}_i^{s+\alpha_i} = \widetilde{g} \cdot g_1^{-\beta} \quad (i = 1, 2).$$

The KEA assumption implies that there is a PPT algorithm $\mathcal{A}$ that with overwhelming probability outputs polynomials $h_i(X)$ such that

$$g_1^{h_i(s)} = Q_i \ , \ g_1^{h_i(s)\cdot(s+\alpha)} = b \cdot a^{-\beta_i}$$

Furthermore,

$$a = g_1^{(\beta_1-\beta_2)^{-1}\cdot(h_1(s)-h_2(s))} \ , \ b = g_1^{\beta_1\cdot[(\beta_1-\beta_2)^{-1}(h_1(s)-h_2(s))]+h_1(s)}$$

For brevity, we write

$$f_1(X) := (\beta_1 - \beta_2)^{-1}[h_1(X) - h_2(X)] \ , \ f_2(X) := \beta_1 \cdot [(\beta_1 - \beta_2)^{-1} \cdot [h_1(X) - h_2(X))] + h_1(X).$$

So $a = g_1^{f_1(s)}$, $b = a^{f_2(s)}$ and a PPT adversary that can output $h_1(X)$, $h_2(X)$ can also efficiently output the polynomials $f_1(X)$, $f_2(X)$.

Since the equations $\widehat{g}_i^{s+\alpha_i} = \widetilde{g}_1 \cdot g_1^{-\beta}$ $(i = 1, 2)$ hold, the KEA assumption implies that with overwhelming probability, $\mathcal{A}$ can output polynomials $e_i(X)$ $(i = 1, 2)$ such that

$$g_1^{e_i(s)} = \widehat{g}_1 \ \ , \ \ g_1^{e_i(s)(s+\alpha_i)+\beta_i} = \widetilde{g}.$$

Set $f(X) := e_1(X)(X + \alpha_1) + \beta_1$. Then $\widetilde{g} = g_1^{f(s)}$. We argue that $f(X) \cdot f_1(X) = f_2(X)$ with overwhelming probability, which in turn will imply that $a^{f(s)} = b$ except with negligible probability.

Note that $f(X) \equiv \beta_1 \pmod{(X + \alpha_1)}$. In particular,

$$f(X) \cdot f_1(X) \equiv f_2(X) \pmod{(X + \alpha_1)}$$

and since α_1 is randomly and uniformly sampled from $\mathbb{F}_p^*$ after $\widetilde{g}$ has been sent, it follows that with overwhelming probability, $f(X) \cdot f_1(X) = f_2(X)$. Thus, $b = a^{f(s)}$

We now demonstrate witness extractability to show that this is an argument of knowledge. The extractor $\mathcal{E}$ with access to the accepting transcripts and to the CRS proceeds as follows. Given accepting transcripts (Q_i, β_i) for challenges α_i $(i = 1, \cdots, N)$, $\mathcal{E}$ uses the Chinese remainder theorem to compute the polynomial $e_N(X)$ such that

$$e_N(X) \equiv \beta \pmod{(X + \alpha_i)} \ \ , \ \ 1 = 1, \cdots, N.$$

If the equation

$$\mathbf{e}(a, g_2^{e_n(s)}) = \mathbf{e}(b, g_2)$$

holds, $\mathcal{E}$ halts. Otherwise, $\mathcal{E}$ samples the next accepting transcript (Q_{N+1}, β_{N+1}) and computes the polynomial $e_{N+1}(X)$ such that

$$e_{N+1}(X) \equiv e_N(X) \ (\mathrm{mod}\ \prod_{i=1}^{N}(X + \alpha_i)) \ \ , \ \ e_{N+1}(X) \equiv \beta_{N+1} \pmod{(X + \alpha_{N+1})}.$$

via the Chinese remainder theorem. When the number of accepting transcripts sampled exceeds the degree of $f(X)$, the polynomial obtained by $\mathcal{E}$ is $f(X)$ with overwhelming probability. □

We now discuss a zero-knowledge variant of the protocol PoKE for bilinear accumulators. This is a honest verifier zero-knowledge argument system. It just requires the Prover to add a blinding factor to his PoKE proof.

Protocol 2.14. *ZK Proof of knowledge of the exponent for bilinear accumulators* (ZKPoKE):
Parameters: A pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$; generators g_1, g_2 for $\mathbb{G}_1, \mathbb{G}_2$ respectively.
Inputs: Elements $a, b \in \mathbb{G}_1$
Claim: The Prover possesses a polynomial $f(X) \in \mathbb{F}_p[X]$ such that $a^{f(s)} = b$.

1. The Prover $\mathcal{P}$ chooses a random $k \in \mathbb{F}_p^*$ and sends $u := a^k \in \mathbb{G}_1$ to the Verifer $\mathcal{V}$.
2. The Fiat-Shamir heuristic generates a challenge $\alpha \in \mathbb{F}_p^*$.
3. $\mathcal{P}$ generates a non-interactive proof for the PoKE$[a,\ b^\alpha \cdot u]$ and sends it to $\mathcal{V}$.

4. $\mathcal{V}$ independently computes $b^\alpha \cdot u \in \mathbb{G}_1$ and accepts if and only if the proof for $\mathtt{PoKE}[a,\ b^\alpha \cdot u]$ is valid. □

As was the case with the protocol PoKE, the protocol ZKPoKE can be easily modified for the setting where the exponentiation is in the group $\mathbb{G}_2$ instead of the group $\mathbb{G}_1$.

3 Batching non-membership proofs

In this subsection, we show that we can have non-membership proofs of constant size with bilinear accumulators. As before, let $\mathcal{D}$ be the set of accumulated elements and let $\mathcal{D}_0$ be a set of elements disjoint from $\mathcal{D}$. For brevity, we write

$$f(X) := \prod_{d \in \mathcal{D}} (X+d) \ \ , \ \ f_0(X) := \prod_{d_0 \in \mathcal{D}_0} (X+d_0).$$

Since the polynomials $f(X)$, $f_0(X)$ are relatively prime, we may compute polynomials $h(X), h_1(X) \in \mathbb{F}_p[X]$ such that

$$f_0(X)h_0(X) - f(X)h(X) = 1 \in \mathbb{F}_p[X] \ \ , \ \ \deg h(X) < \deg f_0(X).$$

Set $w(\mathcal{D}_0) := g_1^{h_0(s)} \in \mathbb{G}_1$. Then

$$w(\mathcal{D}_0)^{f_0(s)} = \mathrm{Acc}(g_1, \mathcal{D})^{h(s)} g_1.$$

We use the pair $(w(\mathcal{D}_0), g_2^{h(s)}) \in \mathbb{G}_1 \times \mathbb{G}_2$ as the (constant-sized) non-membership witness for $\mathcal{D}_0$.

The problem that arises here is that a malicious Prover could provide a false witness since the Verifier does not know the polynomial $h(X)$. The most obvious solution to this would be to require the Prover to send $h(X)$ to the Verifier. However, that would require witnesses of size linear in the size of the set $\mathcal{D}_0$. Instead, following the idea presented in [BBF19] for the PoKE in accumulators based on groups of unknown order, we use the non-interactive PoKE for pairings to demonstrate that the element $g^{h(s)}$ was computed in an honest manner.

Protocol 3.1. *Protocol for non-membership proofs with the bilinear accumulator*
Parameters: A pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$;
Inputs: The accumulated digest $\mathrm{Acc}(g_1, \mathcal{D})$ for a data set $\mathcal{D}$; a data set $\mathcal{D}_0$
Claim: $\mathcal{D}_0 \cap \mathcal{D} = \emptyset$.

1. The Prover $\mathcal{P}$ computes the polynomials

$$f(X) := \prod_{d \in \mathcal{D}} (X+d) \ \ , \ \ f_0(X) := \prod_{d_0 \in \mathcal{D}_0} (X+d_0).$$

2. $\mathcal{P}$ computes polynomials $h_0(X),\ h(X) \in \mathbb{F}_p[X]$ such that

$$f_0(X)h_0(X) - f(X)h(X) = 1 \in \mathbb{F}_p[X] \ \ , \ \ \deg h(X) < \deg f_0(X).$$

3. $\mathcal{P}$ computes $w_1 := g_1^{h_0(s)}$, $w_2 := g_2^{h(s)}$ and sends the pair

$$(w_1, w_2) := (g_1^{h_0(s)}, g_2^{h(s)}) \in \mathbb{G}_1 \times \mathbb{G}_2$$

to the Verifier $\mathcal{V}$ along with non-interactive proofs for $\mathtt{PoKE}^*[g_2,\ w_2]$, $\mathtt{PoKE}^*[g_1,\ w_1]$.

4. $\mathcal{P}$ computes $\widetilde{g}_2 := g_2^{f_0(s)}$ and sends $\widetilde{g}_2$ to $\mathcal{V}$ along with a non-interactive PoE for the equation $\widetilde{g}_2 = g_2^{f_0(s)}$.

5. $\mathcal{V}$ verifies the equation

$$\mathbf{e}(w_1, \widetilde{g}_2) \overset{?}{=} \mathbf{e}(\mathrm{Acc}(g_1, \mathcal{D}), w_2) \cdot \mathbf{e}(g_1, g_2).$$

$\mathcal{V}$ then verifies the non-interactive proofs for $\mathtt{PoKE}^*[g_2,\ w_2]$, $\mathtt{PoKE}^*[g_1,\ w_1]$ and the non-interactive $\mathtt{PoE}^*[g_2,\ f_0(X),\ \widetilde{g}_2]$. $\mathcal{V}$ accepts if and only if all of these proofs are valid. □

Note that

$$w_1^{f_0(s)} = \mathrm{Acc}(g_1, \mathcal{D})^{h(s)} g_1 \iff \mathbf{e}(w_1, \widetilde{g}_2) = \mathbf{e}(\mathrm{Acc}(g_1, \mathcal{D}), g_2^{h(s)}) \cdot \mathbf{e}(g_1, g_2) \bigwedge \mathtt{PoE}^*[g_2,\ f_0(X),\ \widetilde{g}_2]$$

Thus, the pairing check in Step 5 would be equivalent to verifying the the equation

$$w_1^{f_0(s)} = \mathrm{Acc}(g_1, \mathcal{D})^{h(s)} g_1$$

if the Prover sent the polynomial $h(X)$ in the clear. However, the Verifier does not need the polynomial $h(X)$ in order to be convinced of the non-membership of $\mathcal{D}_0$ in $\mathcal{D}$. It suffices for the Verifier to know that $w_2 = g_2^{h(s)}$ for *some* polynomial $h(X)$ known to the Prover.

The Verifier needs just the four points $\{g_1, g_1^s, g_2, g_2^s\}$ to perform the verification. His computational burden is reduced to computing two pairings in addition to verifying a non-interactive PoE and a non-interactive PoKE. We now prove the security of this protocol under the n-strong Diffie Hellman and KEA assumptions.

Theorem 3.2. *The protocol for nonmembership proofs with the bilinear accumulator is secure under the n-strong Diffie Hellman and KEA assumptions.*

Proof. Set

$$f(X) = \prod_{d \in \mathcal{D}} (X + d) \ \ , \ \ f_0(X) = \prod_{d \in \mathcal{D}_0} (X + d_0).$$

As in the protocol, let g_1 be a randomly generated element of $\mathbb{G}_1$ and $\mathrm{Acc}(g_1, \mathcal{D})$ the accumulated digest

$$\mathrm{Acc}(g_1, \mathcal{D}) := g_1^{\prod\limits_{d \in \mathcal{D}} (s+d)}.$$

Suppose there exists a PPT adversary $\mathcal{A}$ that outputs a non-membership witness for a set $\mathcal{D}_0$ that is not disjoint from $\mathcal{D}$. Thus, $\mathcal{A}$ outputs elements $(w_1, w_2, \widetilde{g}_2) \in \mathbb{G}_1 \times \mathbb{G}_2^2$ such that

$$\mathbf{e}(w_1, \widetilde{g}_2) = \mathbf{e}(\mathrm{Acc}(g_1, \mathcal{D}), w_2) \cdot \mathbf{e}(g_1, g_2)$$

along with non-interactive proofs for $\mathtt{PoKE}^*[g_2,\ w_2]$, $\mathtt{PoKE}^*[g_1,\ w_1]$, $\mathtt{PoE}^*[g_2,\ f_0(X),\ \widetilde{g}_2]$.

The non-interactive $\mathtt{PoE}^*[g_2,\ f_0(X),\ \widetilde{g}_2]$ implies that $g_2^{f_0(s)} = \widetilde{g}_2$. The non-interactive $\mathtt{PoKE}^*[g_2,\ w_2]$ (in the group $\mathbb{G}_2$) implies that with overwhelming probability, the Prover possesses a polynomial $h(X) \in \mathbb{F}_p[X]$ such that $w_2 = g_2^{h(s)}$. Similarly, the non-interactive $\mathtt{PoKE}^*[g_1,\ w_1]$ (in the group $\mathbb{G}_1$) implies that with overwhelming probability, the Prover possesses a polynomial $h_0(X) \in \mathbb{F}_p[X]$ such that $w_1 = g_1^{h_0(s)}$. Finally, the pairing check implies that

$$w_1^{f_0(s)} = \mathrm{Acc}(g_1, \mathcal{D})^{h(s)} \cdot g_1.$$

Pick an element d_0 in $\mathcal{D}_0 \cap \mathcal{D}$. The PPT adversary $\mathcal{A}$ can compute the product

$$e(X) := \Big(\prod_{x \in (\mathcal{D} \cap \mathcal{D}_0) \setminus \{d_0\}} (X + d) \Big) \cdot h_0(X) \ \in \ \mathbb{F}_p[X]$$

and subsequently the element

$$w := w_1^{\prod_{x\in(\mathcal{D}\cap\mathcal{D}_0)\setminus\{d_0\}}(s+x)} = g_1^{e(s)} \;\in\; \mathbb{G}_1$$

using the CRS. Thus,

$$w^{s+d_0} = \mathrm{Acc}(g_1, \mathcal{D}) \cdot g_1 = g_1^{\prod_{d\in\mathcal{D}}(s+d)} \cdot g_1$$

and hence,

$$g_1 = \left(g_1^{\prod_{d\in\mathcal{D}\setminus\{d_0\}}(s+d)} \cdot w^{-1} \right)^{s+d_0} .$$

Thus, a PPT adversary who can create a fake proof of non-membership with non-negligible probability can also successfully break either the strong Diffie Hellman assumption or the KEA assmption with non-negligible probabilty. □

4 Non-repetition in committed sets

The non-membership proof in the preceding section boils down to succinctly proving that for elements $a_1,\ a_2 \in \mathbb{F}_p[X]$, the Prover possesses relatively prime polynomials $f_1(X),\ f_2(X)$ such that $a_1 = g_1^{f_1(s)}$, $a_1 = g_1^{f_2(s)}$. For the non-membership proof of $\mathcal{D}_0$ in $\mathcal{D}$, this is achieved by setting

$$a_1 := \mathrm{Acc}(g_1, \mathcal{D}) \ \ , \ \ a_2 := \mathrm{Acc}(g_1, \mathcal{D}_0)$$

The same technique can also be used to show that for an element $a \in \mathbb{G}_1$, the Prover possesses a square-free polynomial $f(X)$ such that $a = g_1^{f(s)}$. We note that $\mathbb{F}_p$ is a perfect field. Hence, a polynomial $f(X)$ being square-free in $\mathbb{F}_p[X]$ is equivalent to $f(X)$ being square-free in $\overline{\mathbb{F}}_p[X]$.

The protocol hinges on the simple fact that a polynomial $f(X) \in \mathbb{F}_p[X]$ is square-free if and only if it is relatively prime with its derivative $f'(X)$. Furthermore, a polynomial $e(X)$ is the derivative of $f(X)$ if and only if $f(X) = e(X)\cdot X + c_0$ for some constant $c_0 \in \mathbb{F}_p$. We now describe how a Prover can use these simple properties to succinctly demonstrate the square-free nature of a committed polynomial.

Given an element $a' \in \mathbb{F}_p[X]$, if the Prover can demonstrate that $a^{h(s)} \cdot a_1^{h_1(s)} = g_1$ for some polynomials $h(X),\ h_1(X)$ known to him, it implies that $a,\ a'$ are commitments (with base g_1) to relatively prime polynomials. If the Prover can also show that $(a')^s \cdot g_1^{c_0} = a$ for some constant $c_0 \in \mathbb{F}_p$, it then follows that the polynomial committed in a' is the derivative of the polynomial committed in a.

An obvious application is that the protocol can be used to demonstrate that a multiset commitment is actually a commitment to a *set* rather than to a multiset with some elements of multiplicity ≥ 2.

Protocol 4.1. *Protocol for separable polynomial commitment*
Parameters: A pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$;
Inputs: Element $a \in \mathbb{G}_1$
Claim: The Prover knows a *separable* polynomial $f(X) \in \mathbb{F}_p[X]$ such that $g_1^{f(s)} = a$
1. The Prover computes the derivative $f'(X)$ and sends $a' := g_1^{f'(s)}$ and the constant term $c_0 \in \mathbb{F}_p$ of $f(X)$ to the Verifier.
2. $\mathcal{P}$ computes polynomials $h_1(X),\ h_0(X)$ such that

$$f(X)h(X) + f'(X)h_1(X) = 1 \;\in\; \mathbb{F}_p[X] \ \ , \ \ \deg(h(X)) < \deg(f'(X)).$$

3. $\mathcal{P}$ computes $w := g_2^{h(s)}$, $w' := g_2^{h_1(s)}$ and sends them to $\mathcal{V}$ along with non-interactive proofs for $\texttt{PoKE}^*[g_2,\ w]$, $\texttt{PoKE}^*[g_2,\ w']$.

4. $\mathcal{V}$ verifies the two $\texttt{PoKE}^*$s and verifies the equations

$$\mathbf{e}(a, w) \cdot \mathbf{e}(a', w') \stackrel{?}{=} \mathbf{e}(g_1, g_2) \quad \bigwedge \quad \mathbf{e}(a', g_2^s) \cdot \mathbf{e}(g_1^{c_0}, g_2) \stackrel{?}{=} \mathbf{e}(a, g_2).$$

$\mathcal{V}$ accepts if and only if the two $\texttt{PoKE}^*$s are valid and both of the pairing equations hold. □

The Verifier just needs the four points $\{g_1, g_1^s, g_2, g_2^s\}$ to efficiently verify the proof. Clearly, the protocol is easy to modify if the element a lies in the group $\mathbb{G}_2$ instead of $\mathbb{G}_1$.

Theorem 4.2. *The preceding protocol is secure under the n-strong Diffie Hellman and KEA assumptions.*

Proof. It suffices to show that in case of an accepting transcript, a PPT adversary can - with overwhelming probability- output a polynomial $f(X) \in \mathbb{F}_p[X]$ such that $a = g_1^{f(s)}$ and $f(X)$ is relatively prime with its derivative $f'(X)$.

Suppose a PPT adversary $\mathcal{A}$ is able to output an accepting transcript. The element $c_0 \in \mathbb{F}_p$ and the element $a' \in \mathbb{F}_p$ sent by $\mathcal{A}$ pass the pairing check

$$\mathbf{e}(a', g_2^s) \cdot \mathbf{e}(g_1^{c_0}, g_2) \stackrel{?}{=} \mathbf{e}(a, g_2),$$

which implies that $(a')^s \cdot g_1^{c_0} = a$. The KEA assumption implies that with overwhelming probability, $\mathcal{A}$ can output a polynomial $e(X)$ such that

$$a' = g_1^{e(s)} \quad , \quad a = g_1^{e(s) \cdot s + c_0}.$$

Setting $f(X) := e(X) \cdot X + c_0$ yields

$$a' = g_1^{f'(s)} \quad , \quad a = g_1^{f(s)}.$$

Since the proofs for $\texttt{PoKE}^*[g_2,\ w]$, $\texttt{PoKE}^*[g_2,\ w']$ are valid, it follows that with overwhelming probability, the Prover possesses polynomials $h(X)$, $h_1(X)$ such that

$$w = g_2^{h(s)} \quad , \quad w' = g_2^{h_1(s)}.$$

The pairing check

$$\mathbf{e}(a, w) \cdot \mathbf{e}(a', w') \stackrel{?}{=} \mathbf{e}(g_1, g_2)$$

now implies that $f(X)h(X) + f'(X)h_1(X) = 1$. Thus, with overwhelming probability, the polynomials $f(X)$ and $f'(X)$ are relatively prime. □

5 Vector Commitments

The aim of this section is to construct a Vector Commitment with constant sized openings using the accumulator constructed in the preceding section. Informally, a Vector Commitment is a binding commitment to a vector in the same way that an accumulator is a binding commitment to a set.

The first Vector Commitment with public parameters as well as openings of constant size was constructed in [BBF19] using their universal accumulator based on groups of unknown order. Unfortunately, this does not seem feasible for a bilinear Vector Commitment since the bilinear accumulator has linear public parameters. But our construction does yield a bilinear VC with linear public parameters and openings of constant size which we expect to have a significant speed advantage over a group-based VC. Furthermore, rather than storing the entire public parameter, the Verifier only needs to store the set $\{g_1, g_1^s, g_2, g_2^s\}$ in addition to the membership proofs which are of constant size. Thus, his total amount of storage is of constant size.

Definition 5.1. A *Vector Commitment* (VC) is a tuple consisting of the following PPT algorithms:
1. $\texttt{VC.Setup}(\lambda, n, \mathcal{M})$: Given security parameter λ, length n of the vector and message space $\mathcal{M}$ of vector components, output public parameters $\texttt{pp}$ which are implicit inputs to all the following algorithms.
2. $\texttt{VC.Com}(\mathbf{m}) \longrightarrow \tau$: Given an input $\mathbf{m} = (m_1, \cdots, m_n)$ output a commitment *com*.
3. $\texttt{VC.Update}(com, m, i, \tau)$: Given an input message m and a position i, output a commitment *com* and advice τ.
4. $\texttt{VC.Open}(com, m, i, \tau)$: On input $m \in \mathcal{M}$ and $i \in [1, n]$, the commitment *com* and advice τ, output an opening π that proves m is the i-th committed element of *com*.
5. $\texttt{VC.Verify}(com, m, i, \tau) \longrightarrow 0/1$: On input commitment *com*, an index $i \in [1, n]$ and an opening proof π, output 1 (accept) or 0 (reject).

A vector commitment is said to be a *subvector commitment* (SVC) if given a vector $\mathbf{m}$ and a subvector $\mathbf{m}'$, the committer may open the commitments at all the positions of $\mathbf{m}'$ simultaneously. This notion was first introduced in [LM18]. It is necessary for each opening to be of size independent of the length of $\mathbf{m}'$, since otherwise it would be no more efficient than opening the positions separately. For instance, a Merkle tree is an example of a Vector Commitment that is not a subvector commitment since its position openings are not constant sized and the openings of several positions cannot be compressed into a single proof. In the rest of this section, we construct a SVC using the accumulator constructed in Section 2.

We start by constructing a bilinear accumulator as in the last section. The message space $\mathcal{M}$ is the set $\{0,1\}^*$. Our construction associates the element $i + p\mathbb{Z} \in \mathbb{F}_p^*$ for each index i of the vector. We now define a bit-vector $\mathbf{m} = (m_1, \cdots, m_{p-1})$ of length $p - 1$ as follows. For each index i, we set

$$m_j = \begin{cases} 1 & \text{if } j + p\mathbb{Z} \text{ was accumulated .} \\ 0 & \text{otherwise.} \end{cases}$$

The bit-vector $\mathbf{m}$ is *sparse*, i.e. most of its entries are 0. The opening of the i-th index is a membership proof of $i + p\mathbb{Z}$ if $m_i = 1$ and a non-membership proof if $m_i = 0$. With the accumulator we constructed in the last section, each opening is of constant size. Furthermore, the openings of multiple indices can be batched into a constant sized proof by aggregating all the membership witnesses for $\mathbb{F}_p^*$-elements on the indices opened to 1 and batching all the non-membership witnesses for elements at the indices opened to 0.

We use our accumulator to commit to the set of elements corresponding to indices such that $m_i = 1$. The opening of the i-th index to m_i is an inclusion proof for d_i and the opening to $m_i = 0$ is an exclusion proof for d_i. With our bilinear accumulator, the opening of each index is constant-size. Furthermore, the openings of multiple indices can be batched into a single constant sized proof using membership proofs for elements on the indices opened to elements of $\mathbb{F}_p^*$ and non-membership proofs for elements opened to 0.

5.1 A key-value map commitment

Following the ideas of [BBF19], we use our sparse VC to construct a key-value map commitment as follows. The key-space is represented by positions in the vector and the associated value is the data at the keys position. The vector length is exponential in the key length and most positions are zero. The complexity of the commitment is proportional to the number of bit indices that are set to 1 and hence, is independent of the length of the vector.

References

[ABC+12] Jae Hyun Ahn, Dan Boneh, Jan Camenisch, Susan Hohenberger, Abhi Shelat and Brent Waters. *Computing on authenticated data.* In Ronald Cramer, editor, TCC 2012, volume 7194 of LNCS, pages 1-20.Springer, Heidelberg, March 2012.

[BBF19] D. Boneh, B. Bunz, B. Fisch, *Batching Techniques for Accumulators with Applications to IOPs and Stateless Blockchains*, In Alexandra Boldyreva and Daniele Micciancio, editors, CRYPTO 2019, Part I, volume 11692 of LNCS, pages 561–586. Springer, Heidelberg, August 2019.

[BGG17] S. Bowe, A. Gabizon, M. Green, *A multi-party protocol for constructing the public parameters of the Pinocchio zk-SNARK*

[BGM17] S. Bowe, A. Gabizon, I. Myers, *Scalable Multi-party Computation for zk-SNARK Parameters in the Random Beacon Model* Cryptology ePrint Archive, Report 2017/1050, 2017. http://eprint.iacr.org/2017/1050.

[CF13] D. Catalano, D. Fiore, *Vector commitments and their applications*, In Kaoru Kurosawa and Goichiro Hanaoka, editors, PKC 2013, volume 7778 of LNCS, pages 55–72. Springer, Heidelberg, February / March 2013

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In Moti Yung, editor, CRYPTO 2002, volume 2442 of LNCS, pages 61-76. Springer, Heidelberg, August 2002.

[CPZ18] A. Chepurnoy, C. Papamanthou, Y. Zhang, *EDRAX : A Cryptocurrency with Stateless Transaction Validation*, Cryptology ePrint Archive, Report 2018/968, 2018. https://eprint.iacr.org/2018/968.

[DT08] I. Damgard, N. Triandopolous, *Supporting Non-membership Proofs with Bilinear-map Accumulators*, Cryptology ePrint Archive, Report 2008/538, 2008. http://eprint.iacr.org/2008/538.

[FVY14] Conner Fromknecht, Dragos Velicanu, and Sophia Yakoubov. *A decentralized public key infrastructure with identity retention.* Cryptology ePrint Archive, Report 2014/803, 2014. http://eprint.iacr.org/2014/803.

[FST06] D. Freeman, M. Scott, E. Teske, *A taxonomy of pairing-friendly elliptic curves*

[FS87] A. Fiat, A. Shamir, *How to prove yourself: Practical solutions to identification and signature problems.* In Andrew M. Odlyzko, editor, CRYPTO'86, volume 263 of LNCS, pages 186–194. Springer, Heidelberg, August 1987

[GGM14] Christina Garman, Matthew Green, and Ian Miers. Decentralized anonymous credentials. In NDSS 2014.

[KS98] E. Kaltofen and V. Shoup. Subquadratic-time factoring of polynomials over finite fields. Mathematics of computation, 67(223):1179–1197, 1998

[KZG10] A. Kate, G. Zaverucha, and I. Goldberg. Constant-size commitments to polynomials and their applications. In Masayuki Abe, editor, ASIACRYPT 2010, volume 6477 of LNCS, pages 177–194. Springer, Heidelberg, December 2010.

[LM18] R. Lai, G. Malavolta, *Optimal succinct arguments via hidden order groups*, Cryptology ePrint Archive, Report 2018/705, 2018. https://eprint.iacr.org/2018/705

[Mil86] V. Miller, *Short Programs for functions on Curves*

[MGGR13a] Ian Miers, Christina Garman, Matthew Green, and Aviel D. Rubin. *Zerocoin: Anonymous distributed E-cash from Bitcoin.* In 2013 IEEE Symposium on Security and Privacy, pages 397411. IEEE Computer Society Press, May 2013

[Ngu05] L. Nguyen, *Accumulators from bilinear pairings and applications*, CT-RSA, 3376:275–292, 2005

[Sla12] Daniel Slamanig. Dynamic accumulator based discretionary access control for outsourced storage with unlinkable access - (short paper). In Angelos D. Keromytis, editor, FC 2012, volume 7397 of LNCS, pages 215222. Springer, Heidelberg, February / March 2012.

[STS99b] Tomas Sander and Amnon Ta-Shma. Flow control: A new approach for anonymity control in electronic cash systems. In Matthew Franklin, editor, FC'99, volume 1648 of LNCS, pages 46-61. Springer, Heidelberg, February 1999

[Tre13] Edward Tremel, *Real world performance of cryptographic accumulators*

[Wes18] B. Wesolowski, *Efficient verifiable delay functions*, In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 379–407. Springer, 2019.

A A protocol for a polynomial relation between discrete logarithms

Protocol A.1. *Protocol for polynomial relation between logarithms*
Parameters: A pairing $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$;
Inputs: Elements $a, b \in \mathbb{G}_1$; a public polynomial $e(X)$
Claim: The Prover knows a polynomial $f(X) \in \mathbb{F}_p[X]$ such that:
- $g_1^{f(s)} = a$, $g_1^{e(f(s))} = b$
1. The Fiat-Shamir heuristic generates a challenge α.
2. The Prover computes polyonmials $h_1(X)$, $h_2(X)$ and $\mathbb{F}_p$-elements β_1, β_2 such that

$$f(X) = h_1(X)(X + \alpha) + \beta_1 \ \ , \ \ e(f(X)) = h_2(X)(X + \alpha) + \beta_2.$$

$\mathcal{P}$ computes $Q_1 := g_1^{h_1(s)}$, $Q_2 := g_1^{h_2(s)}$ and sends (Q_1, β_1, Q_2) to $\mathcal{V}$.
3. The Verifier computes $\beta_2 := e(\beta_1) \in \mathbb{F}_p$ and verifies the equations

$$\mathbf{e}(Q_1, g_2^{s+\alpha}) \cdot \mathbf{e}(g_1^{\beta_1}, g_2) \overset{?}{=} \mathbf{e}(a_1, g_2) \ \bigwedge \ \mathbf{e}(Q_2, g_2^{s+\alpha}) \cdot \mathbf{e}(g_1^{\beta_2}, g_2) \overset{?}{=} \mathbf{e}(b, g_2).$$

$\mathcal{V}$ accepts if and only if both equations hold. □

The two pairing equations in the last step can be batched so that the Verifier computes three pairings instead of six. The pairing checks and the strong Diffie-Hellman assumption imply that with overwhelming probability, the Prover knows polynomials $f(X)$, $f_1(X)$ such that $a = g_1^{f(s)}$, $b = g_1^{f_1(s)}$. Furthermore,

$$f_1(X) \equiv \beta_2 \equiv e(\beta_1) \equiv e(f(X)) \ (\mathrm{mod}\ (X + \alpha)).$$

Since α was randomly and uniformly sampled from $\mathbb{F}_p$ it follows that with overwhelming probability, $f_1(X) = e(f(X)$.

Example of an application: Let $\mathcal{D}$ be a data multiset and let $A := \mathrm{Acc}(g_1, \mathcal{D})$ be its accumulated digest which is known to a Verifier. The last protocol (setting $e(X) = X^n$) can be used to prove that an element $A_1 \in \mathbb{G}_1$ is the accumulated digest of the multiset $n \cdot \mathcal{D}$ (i.e. the multiset such that every $x \in \mathcal{D}$ has multiplicity $n \cdot \mathrm{mul}(x, \mathcal{D})$). The Verifier just needs access to the element $\mathrm{Acc}(g_1, \mathcal{D})$ and the elements $\{g_1, g_1^s, g_2, g_2^s\}$ to verify this proof in constant runtime.

B Aggregation of membership witnesses

Let d_1, d_2 be two accumulated data elements and w_1, w_2 their witnesses with respect to the accumulator $\mathrm{Acc}(g_1, \mathcal{D})$. Thus,

$$\mathrm{Acc}(g_1, \mathcal{D}) = w_1^{s+d_1} = w_2^{s+d_2}.$$

Set $c := (d_1 - d_2)^{-1} \in \mathbb{F}_p^*$ and $w_{1,2} := w_1^{-c} w_2^c$. Then $w_{1,2}^{(s+d_1)(s+d_2)} = \mathrm{Acc}(g_1, \mathcal{D})$. Thus, $w_{1,2}$ is a witness for $\{d_1, d_2\}$. This technique can be generalized to larger sets, but the runtime for doing so is linear in the size of the accumulated set. This makes it impractical for many use cases and is one of the major downsides to the bilinear accumulator.

Let $\mathcal{D}_1$, $\mathcal{D}_2$ be two disjoint accumulated data sets and w_1, w_2 their witnesses with respect to the accumulator $\mathrm{Acc}(g_1, \mathcal{D})$. For brevity, we write

$$f_1(X) := \prod_{d_1 \in \mathcal{D}_1} (X + d_1) \ \ , \ \ f_2(X) := \prod_{d_2 \in \mathcal{D}_2} (X + d_2).$$

Thus,

$$\mathrm{Acc}(g_1, \mathcal{D}) = w_1^{f_1(s)} = w_2^{f_2(s)}.$$

Since the polynomials $f_1(X)$, $f_2(X)$ are relatively prime and $\mathbb{F}_p[X]$ is a principal ideal domain, we may compute polynomials $h_1(X), h_2(X) \in \mathbb{F}_p[X]$ such that

$$h_1(X)f_1(X) + h_2(X)f_2(X) = 1 \in \mathbb{F}_p[X], \ \ \deg h_i(X) < \deg f_{3-i}(X).$$

Furthermore, the run time complexity of computing these polynomials is $\mathbf{O}(N \log^2(N) \log(\log(N))$ where $N = \max(\deg f_1(X), \deg f_2(X))$. Now, the element $w_{1,2} := w_1^{h_2(s)} w_2^{h_1(s)}$ is such that $w_{1,2}^{f_1(s)f_2(s)} = \mathrm{Acc}(g_1, \mathcal{D})$. Thus, $w_{1,2}$ is a witness for the union $\mathcal{D}_1 \cup \mathcal{D}_2$.

In order to compute the elements $w_{1-i}^{f_i(s)}$ (where $i = 0, 1$), the Prover needs to know (or compute from scratch) the polynomials $e_i(s)$ such that $w_{1-i} = g^{e_i(s)}$ and then compute

$$w_{1-i}^{f_i(s)} = g^{e_i(s) f_i(s)}.$$

The runtime complexity for this is linear in the number of elements in the accumulator. This is a major drawback when compared to accumulators based on hidden order groups.

B.1 Aggregation of non-membership witnesses

As was the case with membership proofs, the runtime complexity for aggregating non-membership proofs is linear in the number of elements in the accumulator. Consequently, it is impractical for many use cases.

As before, let $\mathcal{D}$ be the accumulated data set of size n. Let $\mathcal{D}_1$, $\mathcal{D}_2$ be disjoint data sets such that $|\mathcal{D}_1| + |\mathcal{D}_1| << n$. Set

$$f_1 := \prod_{d_1 \in \mathcal{D}_1} (X + d_1) \ \ , \ \ f_1 := \prod_{d_2 \in \mathcal{D}_2} (X + d_2).$$

Suppose we possess non-membership witnesses $(w_1, g_2^{e_1(s)})$, $(w_2, g_2^{e_2(s)})$ for $\mathcal{D}_1$, $\mathcal{D}_2$ respectively along with non-interactive proofs of knowledge for the polynomials $e_1(X), e_2(X)$. We compute polynomials $\widetilde{f_1}(X)$, $\widetilde{f_2}(X)$ such that

$$f_1(X)\widetilde{f_1}(X) + f_2(X)\widetilde{f_2}(X) = 1 \ \ , \ \ \deg \widetilde{f_2}(X) < \deg f_1(X)$$

and define

$$\widetilde{e}_{1,2}(X) := e_1(X) f_1(X) \widetilde{f_1}(X) + e_2(X) f_2(X) \widetilde{f_2}(X).$$

We compute polynomials $q(X), e_{1,2}(X) \in \mathbb{F}_p[X]$ such that

$$\widetilde{e}_{1,2}(X) = q(X) f_1(X) f_2(X) + e_{1,2}(X), \ \ \deg e_{1,2}(X) < \deg f_1(X) f_2(X).$$

Now set

$$w_{1,2} := \left(w_1^{\widetilde{f_2}(s)} \cdot w_2^{\widetilde{f_1}(s)} \right)^{q(s)}.$$

By construction,

$$w_{1,2}^{f_1(s)f_2(s)} = \mathrm{Acc}(g_1, \mathcal{D})^{e_{1,2}(s)} \cdot g_1.$$

A Prover who possesses proofs of knowledge of the polynomials $e_1(X), e_2(X)$, can efficiently generate a proof of the knowledge of $e_{1,2}(X)$. Thus, the pair $(w_{1,2}, g_2^{e_{1,2}(s)})$ along with a non-interactive proof of knowledge of $e_{1,2}(X)$ serves as a non-membership witness for the union $\mathcal{D}_1 \cup \mathcal{D}_2$.