

Homomorphic Polynomial Commitments with Efficiently Verifiable Evaluation Proofs

Alan Szepieniec¹ and Benedikt Bünz²

¹ Nervos Foundation

alan@nervos.org

² Stanford University

1 Introduction

interactive proof systems for arbitrary computation

- zero-knowledge proofs - arithmetization - polynomial relations

circuit model: QAP and Groth16 NILP

- construction - used algebra - advantages and disadvantages

Turing model: STARK

- construction - used algebra - advantages and disadvantages

Class group

- groups of unknown order - vdfs / accumulators

Our contribution:

- polynomial commitments with groups of unknown order - efficiently verifiable evaluation proofs - interactive proofs for arbitrary computation

2 Preliminaries

Notation.

- Let $f(x) \in \mathbb{F}_p[x]$ be a polynomial of degree at most $N - 1$ where N is a power of two. The coefficients of $f(x)$ are denoted by f_i such that $f(x) \triangleq \sum_{i=0}^{N-1} f_i x^i$.
- We work in a group \mathbb{G} of unknown order (*e.g.* an ideal class group) with a designated base element $g \in \mathbb{G}$ with unknown order. (It might be tempting refer to this element as the *generator* but that terminology would imply that \mathbb{G} is cyclic, which is not necessarily true.) We use multiplicative notation.
- Let $q \in \mathbb{N}$ be an integer with $q \gg p$.

2.1 Generic Group Model

2.2 Hardness Assumptions

3 Polynomial Commitment and Evaluation Proof

3.1 Commitment Scheme.

To commit to a polynomial $f(x) \in \mathbb{F}_p[x]$, the committer raises the designated group element g to the the power $\sum_{i=0}^{N-1} f_i q^i$; to open the commitment he pro-

duces this integer. Formally, the commitment scheme is described by the following functionalities:

- **com** : $\mathbb{F}_p[x] \rightarrow \mathbb{G}$, $f(x) = \sum_{i=0}^{N-1} f_i x^i \mapsto g^{\sum_{i=0}^{N-1} f_i q^i}$
- **open** : $\mathbb{G} \times \mathbb{Z} \rightarrow \{\perp\} \cup \mathbb{F}_p[x]$, $(c, z) \mapsto \begin{cases} \perp & \text{if } g^z \neq c \\ f(x) = \sum_{i=0}^{N-1} f_i x^i & \text{else:} \end{cases}$,
 where $\forall i \in \{0, \dots, N-1\} : f_i \equiv z_i \bmod p$ and where $(z_i)_{i=0}^{N-1}$ is the base- q expansion of z , i.e., $z = \sum_{i=0}^{N-1} z_i q^i$ and $z_i \in \{0, \dots, q-1\}$.

This commitment scheme is not meant to be hiding. It does have to be binding. The following theorem establishes this fact.

Theorem 1 (binding). *The commitment scheme (com, open) is computationally binding under the adaptive root assumption.*

Proof. Let $z_0 \neq z_1 \in \mathbb{Z}$ be two different valid openings of the commitment c then $g^{z_0} = g^{z_1} = c$, then $z_0 - z_1$ is a multiple of $\text{ord}(g)$, which can be used in combination with the extended Euclidean algorithm to compute the d th root of g , for any d . This establishes that the ability to produce multiple openings of a single commitment to different polynomials breaks the adaptive root assumption. \square

This commitment scheme is sort of homomorphic. The group of integers $\mathbb{Z}, +$ acts naturally on the group of unknown order \mathbb{G} ; we embed the group of polynomials $\mathbb{F}_p[x], +$ into the group of integers by identifying $f(x) \in \mathbb{F}_p$ with $\sum_{i=0}^{N-1} f_i q^i \in \mathbb{Z}$. This embedding implies that the coefficients of the polynomial are not reduced modulo p ; nevertheless, from the point of view of the commitment scheme, their representatives modulo p are understood. If these coefficients grow through a sequence of homomorphic operations to be larger than q , then the homomorphism fails. Therefore, q has to be large enough such that there will never be any overflow. While the group \mathbb{G} is written multiplicatively, the “group” of commitments is written additively.

3.2 Proof of Correct Evaluation.

We now describe a protocol that allows the prover to prove that e is the evaluation of $f(x)$ in the given point z , where $f(x)$ was committed to in $c = \text{com}(f(x))$. In particular, the verifier knows $c = \text{com}(f(x))$, z , and e ; and the prover knows $f(x)$. We denote the protocol by $\text{PoCEv}[e, z, c, f(x)]$.

The high-level intuition behind the proof is to recursively reduce it to a proof of a simpler statement. In every iteration, the commitment c is identifiable with a polynomial $f(x)$ whose evaluation in z is $f(z) = e$. The prover splits this polynomial into $f_{(0)}(x)$ and $f_{(1)}(x)$ such that $f_{(0)}(x) + x^{N/2} f_{(1)}(x) = f(x)$ and provides the verifier with two matching commitments, c_0 and c_1 , as well as matching evaluations, $e_0 = f_{(0)}(z)$ and $e_1 = f_{(1)}(z)$. The verifier responds by providing a randomizer α . The protocol then recurses on the randomized linear combination of the two components, namely in order to establish that

$f'(x) = f_{(0)}(x) + \alpha f_{(1)}(x)$, which corresponds to $c' = c_0 + \alpha c_1$, evaluates to $e' = e_0 + \alpha e_1$ in z . In the last step, the polynomial consists of a single coefficient which is simply transmitted to the verifier.

Formally, PoCEv is defined recursively as follows:

- **If $N > 1$:**
 - Split $f(x)$ into two polynomials: let $f_{(0)}(x) \triangleq \sum_{i=0}^{N/2-1} f_i x^i$ and $f_{(1)}(x) \triangleq \sum_{i=0}^{N/2-1} f_{N/2+i} x^i$.
 - Prover sends to Verifier: $e_0 \triangleq f_{(0)}(z) \bmod p$, $e_1 \triangleq f_{(1)}(z) \bmod p$, $c_0 \triangleq \text{com}(f_{(0)}(x))$, $c_1 \triangleq \text{com}(f_{(1)}(x))$.
 - Verifier checks that $z^{N/2} e_1 + e_0 \stackrel{?}{=} e \bmod p$ and that $q^{N/2} c_1 + c_0 \stackrel{?}{=} c$.
 - Verifier produces a random scalar, $\alpha \xleftarrow{\$} \mathbb{F}_p$ and sends it to Prover.
 - Prover and Verifier proceed with the protocol PoCEv[$e', z, c', f'(x)$] where $e' \triangleq e_0 + \alpha e_1 \bmod p$, $c' \triangleq c_0 + \alpha c_1$, and $f'(x) \triangleq f_{(0)}(x) + \alpha f_{(1)}(x)$. (The prover does not reduce the coefficients of this polynomial modulo p .)
- **If $N = 1$:**
 - Prover sends the integer f_0 to Verifier. (Remember, at this point $f(x) = f_0 \cdot$)
 - Verifier checks that $f_0 \stackrel{?}{\in} \{0, \dots, q-1\}$ and $f_0 \stackrel{?}{=} e \bmod p$ and $c \stackrel{?}{=} \text{com}(f(x))$.

3.3 Optimizations

- *Use a proof of correct exponentiation.* Computing $q^{N/2} c_1$ naïvely is expensive. However, there is a workaround that avoids this expense, namely by having the prover compute this value and send it to the verifier. The two then engage in the proof of correct exponentiation PoE due to Wesolowski [2], or the batched variant thereof, PoHP [1].
- *Choose q to be a power of 2.* In these proofs of correct exponentiation, the prover must perform a tail division by a random large-enough prime. This tail division is slightly faster when q is a power of 2. Moreover, choosing a power of two makes the integer representation of a polynomial straightforward: a fixed number of bits is allocated for each coefficient.
- *Choose $\alpha \ll p$.* The randomizer α only needs to have entropy equal to the security parameter λ ; when $|p|$ is larger, it is allowable to pick α randomly from $\{0, \dots, 2^\lambda - 1\}$. However, when $|p|$ is smaller, the protocol cannot meet the target security level.
- *Open at $N = 2$ or even earlier.* Rather than open all the way at $N = 1$, it might make sense to terminate the recursion at a shallower depth. The increased number of coefficients to be transmitted is compensated for by the decreased number of group elements.
- *Precompute g, g^q, g^{q^2}, \dots .* Rather than raising the single group element to the huge power $\sum_{i=0}^{N-1} q^i f_i$, it is preferable to raise g^{q^i} to the power f_i and then to combine all the $g^{q^i f_i}$ together. Note that the values g^{q^i} are independent of the witness and even of the statement being proved.

- *Send only c_1 and e_1 .* The prover need not send both c_0 and c_1 . Just c_1 suffices as c_0 can be inferred from $c - q^{N/2}c_1$. A similar inference holds for e_0 .
- *Fiat-Shamir heuristic.* The protocol can be made non-interactive in the random oracle model by replacing the verifier's public coins by the output of a hash function evaluated in the concatenation of all messages up until that point.

3.4 Complexity Analysis.

The protocol recurses to a depth of $\log_2 N$ before it halts, at which point it triggers the case $N = 1$. At this point, the constant f_0 was determined from the coefficients of the original $f(x)$ and via $\log_2 N$ multiplications by random elements from \mathbb{F}_p but without reduction modulo p . So $|f_0| \leq \log_2 N \times (1 + |p|) \leq |q|$. Every iteration, the prover sends two group elements and one field element (with the optimizations) so the total communication cost is $\log_2 N \times (|p| + 2|\mathbb{G}|) + |q|$, ignoring the cost incurred by the verifier. This analysis establishes that the communication cost is logarithmic in N .

The verifier's computation involves logarithmically many (as a function of N) exponentiations in \mathbb{F}_p (by exponent $N/2$) and as many exponentiations in \mathbb{G} (by exponent $q^{N/2}$). The last step requires another group exponentiation (by f_0). The total complexity of the field exponentiations is $O((\log N)^2)$. The total complexity of the group exponentiations is $O(N \times |q|)$ naïvely; but $O(\lambda \times \log_2 N)$ with the first optimization, where λ is the security parameter.

4 Security Analysis

4.1 Soundness

We start with proving that the evaluation proof is sound under the assumption that $(\text{com}, \text{open})$ is a perfectly binding commitment scheme. This assumption is ridiculous in light of the fact that \mathbb{G} is a finite group while the domain of com is infinite. However, this intermediate step provides the intuition behind soundness. Later, we drop the perfect binding assumption in favor of working in the generic group model, and prove that soundness still holds in this model. Later still, in Section 4.2, we show the existence of an extractor capable of extracting the polynomial $f(x)$ under the fractional root assumption (*not* the generic group model).

Theorem 2 (soundness). *Suppose $(\text{com}, \text{open})$ is a perfectly binding commitment scheme. Then the proof system $\text{PoCEv}\{e, z, c \mid e = f(z) \wedge c = \text{com}(f(x))\}$ is statistically sound with respect to the language $\mathcal{L}_c = \{(e, z) \in \mathbb{F}_p \times \mathbb{F}_p \mid f(z) = e \wedge \text{com}(f(x)) = c\}$.*

Proof. By induction on N .

- **If $N = 1$:** The fact that $f_0 \in \{0, \dots, q-1\}$ establishes that \mathcal{L}_c corresponds to a constant polynomial $f(x) = f_0$. Since there is only one coefficient, the equation $e \equiv f_0 \bmod p$ follows from the binding property of the commitment scheme.
- **If $N > 1$:** Suppose $(e, z) \notin \mathcal{L}_c$, meaning that $e \neq f(z)$, where $f(x)$ is the polynomial that corresponds to \mathcal{L}_c .

The binding property guarantees that $f_{(0)}(x)$ and $f_{(1)}(x)$ are uniquely defined as soon as c_0 and c_1 are sent. So the languages \mathcal{L}_{c_0} and \mathcal{L}_{c_1} are also uniquely defined.

From $(e_0, z) \in \mathcal{L}_{c_0} \Leftrightarrow e_0 = f_{(0)}(z)$ and $(e_1, z) \in \mathcal{L}_{c_1} \Leftrightarrow e_1 = f_{(1)}(z)$, one derives $(e_0, z) \in \mathcal{L}_{c_0} \wedge (e_1, z) \in \mathcal{L}_{c_1} \Rightarrow e_0 + z^{N/2}e_1 = f_{(0)}(z) + z^{N/2}f_{(1)}(z)$ and this last implicant is equivalent to $(e, z) \in \mathcal{L}_c$ because $c = c_0 + q^{N/2}c_1$ and $e = e_0 + z^{N/2}e_1$. Negation therefore yields $(e, z) \notin \mathcal{L}_c \Rightarrow (e_0, z) \notin \mathcal{L}_{c_0} \vee (e_1, z) \notin \mathcal{L}_{c_1}$.

Due to the binding commitment scheme, $f'(x) = f_{(0)}(x) + \alpha f_{(1)}(x)$ is unique as the opening of $c' = c_0 + \alpha c_1$. So $\mathcal{L}_{c'}$ is well defined and $(e', z) \in \mathcal{L}_{c'} \Leftrightarrow e_0 + \alpha e_1 = f_0(z) + \alpha f_1(z)$. When $e_0 \neq f_0(z)$ or $e_1 \neq f_1(z)$ this equation holds with probability $1/p$ for uniformly random α . With the complement probability, α is sampled such that $(e', z) \notin \mathcal{L}_{c'}$. The inductive assumption states that the adversary must fail (except with negligible probability) to prove this membership. \square

The reason why the proof of Theorem 2 fails when dropping the perfect binding assumption, is because a given commitment $c \in \mathbb{G}$ no longer uniquely determines a polynomial $f(x)$. As a result, the language \mathcal{L}_c is ill-defined. Not only does the proof fail, the theorem statement becomes nonsensical.

It is possible to modify the definition by taking the union of all sets $\{(e, z) \in \mathbb{F}_p \times \mathbb{F}_p \mid f(z) = e\}$ for all $f(x)$ satisfying $\text{com}(f(x)) = c$. The resulting language $\mathcal{L}_c^* = \{(e, z) \mid \exists f(x) \in \mathbb{F}_p[x]. c = \text{com}(f(x)) \wedge f(z) = e\}$ is well defined, but hardly meaningful, because in general $\mathcal{L}_c^* = \mathbb{F}_p \times \mathbb{F}_p$. To see the membership of any pair $(e, z) \in \mathcal{L}_c^*$, just find a $f(x)$ such that both $f(z) = e$ and $\text{com}(f(x)) = c$ are satisfied — there are plenty of options for a generic group \mathbb{G} .

There is a definitional workaround that identifies the language \mathcal{L}_c with “the only polynomial $f(x)$ that the computationally bounded prover could possess such that $c = \text{com}(f(x))$ ” in a rigorous manner. This workaround involves defining the language in terms of the generic group model. To this end, the next definition relaxes the notion of binding slightly: there ought to exist an efficiently computable function that, when provided with the commitment and the list of queries made to the generic group oracle along with the corresponding answers, outputs the value that was committed to. In fact, the next definition even makes abstraction of the nature of the oracle.

Definition 1 (relative binding). *Let $(\text{com}, \text{open})$ be a commitment scheme defined relative to some oracle \mathcal{O} . Then $(\text{com}, \text{open})$ is relative binding if both conditions are satisfied:*

1. For all algorithms \mathbf{C} making at most a polynomial number (as a function of the security parameter) of queries to the oracle, the probability that $\mathbf{C}^{\mathcal{O}}$ outputs a triple of values (c, o_0, o_1) such that $\text{open}(c, o_0) \neq \perp$ and $\text{open}(c, o_1) \neq \perp$ and $o_1 \neq o_0$, is negligible (in the security parameter).
2. There is an efficiently computable function \mathbf{E} that satisfies the following. Let \mathbf{C} be any algorithm making at most a polynomial number (as a function of the security parameter) of queries to the oracle and terminates by outputting a valid commitment $c \leftarrow \mathbf{C}^{\mathcal{O}}()$. And let $Q = ((q_i, r_i))_i$ be the corresponding list of queries and responses. Then $\text{open}(c, \mathbf{E}(c, Q)) \neq \perp$.

Armed with this notion, it is possible to restate and fix Theorem 2 and its proof. Let \mathbf{C} be the portion of the prover-side algorithm that outputs the original commitment c but before engaging in the proof protocol; and let \mathbf{P} be the portion that plays the part of the prover in the protocol. The list Q consists of the queries and responses originating from \mathbf{C} , and is oblivious of the queries of \mathbf{P} (or of \mathbf{V} , for that matter). Then we consider the language $\mathcal{L}_{\mathbf{C}} \triangleq \{(e, z) \in \mathbb{F}_p \times \mathbb{F}_p \mid \text{open}(c, \mathbf{E}(c, Q)) = f(x) \wedge e = f(z)\}$, and state and prove soundness with respect to this language.

Theorem 3. *Let $(\text{com}, \text{open})$ be a relative binding commitment scheme. Let (\mathbf{C}, \mathbf{P}) be a pair of algorithms that share a state and such that $c \leftarrow \mathbf{C}$ and such that \mathbf{P} is a prover in the proof system $\text{PoCEv}\{e, z, c\}$. Then this proof system is statistically sound with respect to the language $\mathcal{L}_{\mathbf{C}} \triangleq \{(e, z) \in \mathbb{F}_p \times \mathbb{F}_p \mid \text{open}(c, \mathbf{E}(c, Q)) = f(x) \wedge e = f(z)\}$.*

Proof. The proof is identical to that of Theorem 2, except that the language \mathcal{L}_c is replaced by $\mathcal{L}_{\mathbf{C}} \triangleq \{(e, z) \in \mathbb{F}_p \times \mathbb{F}_p \mid \text{open}(c, \mathbf{E}(c, Q)) = f(x) \wedge e = f(z)\}$, and Q is updated at every recursion step to also contain the list of queries and responses made up until that point. \square

What remains to be proven then, is that the commitment scheme $(\text{com}, \text{open})$ is relative binding. We show that this holds in the generic group model.

Theorem 4. *Let $(\text{com}, \text{open})$ be the commitment scheme with*

$$\begin{aligned}
& - \text{com} : \mathbb{F}_p[x] \rightarrow \mathbb{G}, f(x) = \sum_{i=0}^{N-1} f_i x^i \mapsto g^{\sum_{i=0}^{N-1} f_i q^i} \\
& - \text{open} : \mathbb{G} \times \mathbb{N} \rightarrow \mathbb{F}_p \cup \{\perp\}, (c, n) \mapsto \begin{cases} f(x) = \sum_{i=0}^{N-1} f_i x^i & \text{if } g^n = c; \\ \text{where } f_i = n_i \bmod p & \\ \text{and } \sum_{i=0}^{N-1} n_i q^i = n & \\ \perp & \text{else,} \end{cases}
\end{aligned}$$

with the group operations being computed via the group oracle. Then $(\text{com}, \text{open})$ is relative binding in the generic group model.

Proof. Suppose the negation of the first item that defines relative binding, namely that there is an algorithm that makes a polynomial number (in the security parameter) of queries to the group oracle and outputs a triple (c, o_0, o_1) such that

$\text{open}(c, o_0) \neq \perp$ and $\text{open}(c, o_1) \neq \perp$ and $o_1 \neq o_0$. This implies that $g^{o_0} = g^{o_1}$, meaning that $o_0 - o_1$ is a multiple of the order of g . Consequently, \mathbf{C} , together with the extended Euclidean algorithm, can be made into an algorithm that solves the adaptive root problem. Since this problem is impossible (with overwhelming probability) from only a polynomial number of queries in the generic group model, this ability provides the necessary contradiction to make the contrapositive argument complete. So the first item that defines relative binding holds in the generic group model.

As for the second item, consider the table of discrete logarithms where every row and every column represents a group element, and the cell at (i, j) either holds an integer $n_{i,j}$ indicating that group element j raised to the power $n_{i,j}$ gives group element i ; or the cell is empty, indicating that no such discrete logarithm exists. While the list of queries Q does not fix the entire table, it does fix some elements (modulo the group order), namely at most $\#Q$ rows and at most $\#Q$ columns. Moreover, this partial filling out (with concrete representatives for the equivalence classes modulo the group order) can be computed by starting from a table with one element containing the value 1, and iteratively expanding and updating this table as queries from the list are processed.

This table contains a discrete logarithm base g for c , because otherwise c cannot be a valid commitment. The output of \mathbf{E} is the value of the cell whose indices match those of g and c in the list of group elements. Since $\mathbf{E}(c, Q)$ is a discrete logarithm of c base g , we have $\text{open}(c, \mathbf{E}(c, Q)) \neq \perp$. \square

4.2 Knowledge-Soundness

References

1. Boneh, D., Bünz, B., Fisch, B.: Batching techniques for accumulators with applications to iops and stateless blockchains. IACR Cryptology ePrint Archive 2018, 1188 (2018), <https://eprint.iacr.org/2018/1188>
2. Wesolowski, B.: Efficient verifiable delay functions. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part III. Lecture Notes in Computer Science, vol. 11478, pp. 379–407. Springer (2019)