electronics

MDPI

Article

An Approach to the Construction of a Recursive Argument of Polynomial Evaluation in the Discrete Log Setting

Sungwook Kim

Department of Information Security, College of Interdisciplinary Studies for Emerging Industries, Seoul Women's University, Seoul 01797, Korea; kim.sungwook@swu.ac.kr; Tel.: +82-2-970-5767

Abstract: Succinct Non-interactive Arguments of Knowledge (SNARKs) are receiving a lot of attention as a core privacy-enhancing technology for blockchain applications. Polynomial commitment schemes are important building blocks for the construction of SNARKs. Polynomial commitment schemes enable the prover to commit to a secret polynomial of the prover and convince the verifier that the evaluation of the committed polynomial is correct at a public point later. Bünz et al. recently presented a novel polynomial commitment scheme with no trusted setup in Eurocrypt'20. To provide a transparent setup, their scheme is built over an ideal class group of imaginary quadratic fields (or briefly, class group). However, cryptographic assumptions on a class group are relatively new and have, thus far, not been well-analyzed. In this paper, we study an approach to transpose Bünz et al.'s techniques in the discrete log setting because the discrete log setting brings a significant improvement in efficiency and security compared to class groups. We show that the transposition to the discrete log setting can be obtained by employing a proof system for the equality of discrete logarithms over multiple bases. Theoretical analysis shows that the transposition preserves security requirements for a polynomial commitment scheme.

Keywords: blockchain privacy; zero-knowledge proof; proof of knowledge; polynomial commitment; recursive argument; discrete log

Citation: Kim, S. An Approach to the Construction of a Recursive Argument of Polynomial Evaluation in the Discrete Log Setting. *Electronics* **2022**, *11*, 131. https://doi.org/10.3390/electronics11010131

Academic Editor: Martin Reisslein

Received: 19 November 2021
Accepted: 29 December 2021
Published: 1 January 2022

1. Introduction

Zero-Knowledge Succinct Non-interactive Arguments of Knowledge (zk-SNARKs) are non-interactive proof systems between the prover and the verifier. They provide a way for the prover to convince the verifier that the statement claimed by the prover is true without disclosing any other information except the validity of the statement while maintaining a short proof size and an efficient verification by the verifier. Since their adoption to cryptocurrency systems, such as Zcash [1] and Ethereum [2], zk-SNARKs are regarded as an essential technique for solving data privacy issues in blockchain-based applications. There have been numerous SNARK proposals in the literature. Some constructions present very efficient proof systems with the help of a trusted setup [3–5]. Because the transparent property is desirable for applications, such as cryptocurrency, recent constructions [6–8] have focused on a proof system with a transparent setting, i.e., they have no trusted setup.

The construction of SNARKs with no trusted setup heavily relies on a transparent and efficient polynomial commitment scheme. At a high level, transparent zk-SNARKs can be constructed using the framework from polynomial interactive oracle proofs (IOP) [3,6] as follows: (1) The prover expresses the required computation for proving a statement as a set of low-degree polynomials over a finite field $\mathbb{F}$, which is a representation of its witness. (2) The prover sends commitments to low degree polynomials to the verifier, and the verifier then checks the proof by querying evaluations of polynomials for points chosen uniformly at random from $\mathbb{F}$, where we crucially require a polynomial commitment scheme. (3) Finally, one can obtain the non-interactive version of the previous proof systems by applying the Fiat–Shamir heuristic [9].

In this paper, we focus on polynomial commitment schemes. Let $f(X)$ be the prover's secret polynomial over a field $\mathbb{F}$ with the degree at most d, i.e., $\deg(f) \leq d$. In polynomial commitment schemes, the prover sends the commitment to f to the verifier. Later, upon input of a public point $(x, y) \in \mathbb{F} \times \mathbb{F}$, the prover convinces the verifier that the committed polynomial f holds $y = f(X)$ with a proof. We call a polynomial commitment scheme transparent if it requires no trusted setup to generate public parameters for the scheme. Since the first construction was developed by Kate et al. [10], a variety of polynomial commitment schemes have been proposed in the literature.

For polynomial commitment schemes, the main factors of efficiency consist of the computation complexities of the prover (prover complexity) and verifier (verifier complexity), and the communication complexity between them. Usually, constructions with a trusted setup provide higher efficiency than those with a transparent setting. Recently, Bünz et al. [6] proposed an efficient polynomial commitment scheme with a transparent setting. Asymptotically, it achieves a logarithmic verifier complexity and proof size for evaluation (communication complexity). In brief, it improves efficiency by applying an evaluation protocol in a recursive manner. It reduces the degree of a polynomial f by half at each iteration; hence, $\log \deg(f)$ iterations overall. Transparency in the scheme relies on the use of a group of unknown order whose concrete candidate is an ideal class group of imaginary quadratic fields.

The security of a group of unknown order stems from the infeasibility of computing the order of the group. Previous cryptographic constructions over a class group considered concrete group parameters, such as a 1665-bit negative fundamental discriminant for 128-bit security [11,12], which was used in Bünz et al.'s scheme [6]. However, recent works report that the above parameters for class groups provide less security than expected. Notably, Dobson and Galbraith estimate that class groups with a 1665-bit discriminant only offer 55-bit security [13]. They, therefore, claim that orders of a random class group should be at least 2^{3328} for a 128-bit security level. Those parameters correspond to approximately a 6656-bit discriminant. This leads to a decreased efficiency for the cryptographic primitives based on class groups.

In this paper, we put forward a study to overcome the efficiency degradation of Bünz et al.'s construction caused by the use of class groups. To do this, we focus on transposing their techniques in the discrete log setting, preserving a no-trust setup. This approach brings about two advantages. First, the (elliptic curve) discrete log problem is one of the standard cryptographic assumptions as opposed to the order assumption of class groups. To date, its security has been well-understood. Second, the group operation in the discrete log setting (e.g., elliptic curve groups) is much more efficient than that in the class groups, which significantly reduces the actual computation cost for both the prover and the verifier. In addition, a group element in the discrete log setting is shorter than that in class groups. This advantage cuts the cost of bandwidth spent by the prover and the verifier when applying the evaluation protocol of a polynomial commitment scheme.

Our approach is built on an information-theoretic abstraction given in [6] to construct a polynomial commitment scheme. The abstraction requires two properties, a linear homomorphism and a monomial homomorphism, which the underlying commitment scheme should provide. These two properties enable the verifier to apply the computations among polynomials over their committed forms, such as a linear combination (a linear homomorphism) and a degree-shift operation (a monomial homomorphism) of polynomials. The two properties are necessary for an evaluation protocol using a recursive call, which is critical in achieving a logarithmic verifier and communication complexities. To realize these properties in a discrete log setting, we utilize a polynomial encoding method devised by Bootle et al. [14]. This method uses a variant of the Pedersen commitment scheme [15], which naturally provides a linear homomorphism. Unfortunately, however, the Pedersen commitment scheme is not a monomial homomorphism, which is easily obtained in a class group-based scheme [6]. Thus, we focus our attention on the study of a discrete

log-based proof system to prove that a monomial homomorphism is verifiably computed in the discrete log setting. The contribution of this work is as follows.

- We clarify a proof system that proves the correct computation of a monomorphism in the discrete log setting. Specifically, we show it suffices to have a proof system to check the equality of a discrete logarithm over multiple bases, say PoE_{mDL}. Given two subsets $\{g_1, \cdots, g_d\}$ and $\{h_1, \cdots, h_d\}$ of a group $\mathbb{G}$, PoE_{mDL} allows the prover to convince the verifier that $\prod_{i=1}^{d} g_i^{a_i}$ and $\prod_{i=1}^{d} h_i^{b_i}$ have equal exponents, i.e., $a_i = b_i$ for $i = 1, \ldots, d$, without disclosing raw exponents. A number of studies on PoE_{mDL} have been carried out independently of the construction of polynomial commitment schemes. This work bridges two rather independent proof systems and provides a blueprint to combine these proof systems for the construction of an efficient, transparent polynomial commitment scheme in the discrete log setting.
- We propose a recursive argument to show the correct polynomial evaluation by employing PoE_{mDL}. Our approach is to transpose a recursive argument from a class group in [6] to that from the discrete log setting. We present a security analysis to demonstrate the completeness and soundness of the proposed protocol. In addition, We present a zero-knowledge version of the obtained polynomial commitment scheme. A zero-knowledge version ensures that no information of the prover's secret polynomial $f(X)$ is leaked while the prover convinces the verifier that $y = f(x)$ holds for a point (x, y).

The remainder of this paper is organized as follows. In Section 2, we review related works. In Section 3, we provide the background on the hardness assumption and building blocks for polynomial commitment schemes. In Section 4, we present our approach to transpose Bünz et al.'s techniques in the discrete log setting and investigate a sub-routine protocol as a sufficient condition for our approach. In Section 5, we discuss the performance and security of our approach. In Section 6, we extend the polynomial commitment scheme in the previous section to the version with a zero-knowledge evaluation protocol. Finally, we provide some concluding remarks in Section 7.

2. Related Work

A lot of recent research on polynomial commitment schemes have been carried out in the context of Succinct Non-interactive ARguments of Knowledge (SNARKs). In particular, a polynomial commitment scheme provides a key tool to generate a zk-SNARK from a polynomial interactive oracle proof (IOP) [3,6].

Kate et al. first constructed efficient and succinct polynomial commitment schemes for univariate polynomials [10]. The construction is based on bilinear pairings over elliptic curves and requires a trusted setup. Its extension to multivariate polynomials has been proposed by Papamanthou et al. [16] and Zhang et al. [17]. Zhang et al. [18] also presented the zero-knowledge version of their work [17]. These schemes all use bilinear pairings and require a trusted setup.

Associated with transparent SNARKs, polynomial commitment schemes with a transparent setting have received significant attention and, along with the previously mentioned constructions, many schemes can be found in the literature. Bootle et al. [14] constructed a transparent polynomial commitment scheme in the discrete log setting. They represent a polynomial of degree d as a matrix with $\sqrt{d}$ rows and columns and then write a polynomial evaluation as matrix multiplications. This leads to a $O(\sqrt{d})$ commitment size, verifier complexity, and communication complexity. Wahby et al. presented a transparent polynomial commitment scheme [7] for multilinear polynomials under the discrete log assumption. The scheme is built on the ideas of a matrix commitment of Bootle et al. [14] and the inner-product argument of Bünz et al. [19]. For a polynomial of degree d, the $O(\sqrt{d})$ commitment size, verifier complexity, and communication complexity are required. Ben-Sasson et al. [20] introduced the Fast Reed Solomon IOP of Proximity (FRI), which implicitly yields a transparent polynomial commitment scheme. Kattis et al. [8] and Zhang et al. [21] independently presented a method for obtaining polynomial commitment schemes from FRI.

Their construction has $O(\lambda)$ size commitments for the security parameter λ and $O(\log^2 d)$ communication complexity and supports quantum resistance. In addition, Lee [22] proposed a multivariate polynomial commitment scheme with a transparent setting using pairing-based commitments. The scheme builds on inner product arguments given in Bootle et al. [14] and Bünz et al. [6]. Recently, Boneh et al. [23] studied additive polynomial commitment schemes, where commitments form an additive group [6,10,14,19,22]. They showed that the additive property yields a batch evaluation of polynomial commitments, which can be used for the efficient construction of SNARKs.

Groups of unknown order provide a mathematical structure for interesting cryptographic applications, such as delay functions [24], accumulators [25], and polynomial commitment schemes [6]. Most cryptographic applications consider two candidate groups of unknown order, i.e., RSA groups [26] and ideal class groups of imaginary quadratic fields [27]. RSA groups assume a trusted setup in generating the RSA modulus and hence do not meet our current interest. By contrast, class groups do not require a trusted setup and thus have been used in recent constructions with a transparent setting [6,24,25]. Dobson and Galbraith [13] analyzed the security of the candidate parameters for class groups proposed in [11,12]. They argued that the parameters in [11,12] do not meet the desired security level and present much larger parameters, which lead to an extremely large size-up for commitments in previous constructions. In this line of research, Belabas et al. [28] recently reported that the order assumption in class groups of imaginary quadratic fields does not hold in certain special classes of prime numbers. Some studies have explored alternative source groups of unknown order with a transparent setting. As an example, Dobson and Galbraith [13] suggested the Jacobian of hyperelliptic curves of genus 3, whereas Lee [29] pointed out that the order of the Jacobian of a hyperelliptic curve can be efficiently computed.

3. Preliminaries

Throughout the paper, λ denotes the security parameter written in unary. The function $\mathsf{negl} : \mathbb{N} \to [0,1]$ denotes a negligible function, i.e., $\mathsf{negl}(\lambda) = 1 - \lambda^{\omega(1)}$. For a set S, we use $e \xleftarrow{\$} S$ to denote that an element e is sampled uniformly at random from S. For a probabilistic algorithm A, we write $y \leftarrow A(x)$ to denote that y is returned as the result of A on input x together with a randomness r picked internally.

3.1. The Discrete Logarithm Assumptions

Let Ggen be an algorithm that takes on input λ and returns a λ-bit prime number p, cyclic group $\mathbb{G}$ of order p, and a generator g of $\mathbb{G}$.

Definition 1 (Discrete Logarithm Assumption)**.** *The discrete logarithm assumption holds relative to* Ggen *if for any polynomial-time adversary* $\mathcal{A}$*,*

$$\Pr\left[g^a = h \ : \begin{array}{c} (\mathbb{G}, p, g) \leftarrow \mathsf{Ggen}(1^\lambda), \\ h \xleftarrow{\$} \mathbb{G}, \\ a \leftarrow \mathcal{A}(\mathbb{G}, p, g, h) \end{array} \right] \leq \mathsf{negl}(\lambda).$$

Definition 2 (Discrete Logarithm Relation Assumption)**.** *The discrete logarithm relation assumption holds relative to* Ggen *if for any polynomial-time adversary* $\mathcal{A}$*,*

$$\Pr\left[\exists a_i \neq 0 \wedge \prod_{i=0}^{n} g_i^{a_i} = 1 \ : \begin{array}{c} (\mathbb{G}, p, g_0) \leftarrow \mathsf{Ggen}(1^\lambda), \\ g_1, \ldots, g_n \xleftarrow{\$} \mathbb{G}, \\ a_0, \ldots, a_n \leftarrow \mathcal{A}(\mathbb{G}, p, g_0, \ldots g_n) \end{array} \right] \leq \mathsf{negl}(\lambda).$$

In the above definition, $\prod_{i=0}^{n} g_i^{a_i} = 1$ for some $a_i \neq 0$ is called a non-trivial discrete logarithm relation. It is well-known that the discrete logarithm relation assumption is equivalent to the discrete logarithm assumption [14].

3.2. Zero-Knowledge Arguments of Knowledge

Let $\mathcal{R} \subset \mathcal{S} \times \mathcal{W}$ be a polynomial-time-decidable binary relation. $s \in \mathcal{S}$ and $w \in \mathcal{W}$ are called a statement and a witness, respectively. We define $L_{\mathcal{R}}$ as the set $\{s \in \{0,1\}^* : \exists w \in \{0,1\}^* \text{ such that } (s,w) \in \mathcal{R}\}$, which is called the language of $\mathcal{R}$. We consider an argument system for a relation $\mathcal{R}$ consisting of three probabilistic polynomial-time algorithms $(\mathsf{Pgen}, \mathcal{P}, \mathcal{V})$. A non-interactive algorithm Pgen takes the security parameter λ as an input and returns a common reference string (crs) pp. $\mathcal{P}$ and $\mathcal{V}$ are called a prover and a verifier, respectively, and both are interactive algorithms. In addition, $\mathcal{P}$ takes as input a triple of pp, a statement $s \in \mathcal{S}$, and a witness $w \in \mathcal{W}$. Moreover, $\mathcal{V}$ takes as input a pair of pp and a statement $s \in \mathcal{S}$ and outputs 0 or 1. We denote the transcript produced by $\mathcal{P}$ and $\mathcal{V}$ for an interaction by $\mathsf{tr} \leftarrow \langle \mathcal{P}(\mathsf{pp}, s, w), \mathcal{V}(\mathsf{pp}, s) \rangle$ and write $\langle \mathcal{P}(\mathsf{pp}, s, w), \mathcal{V}(\mathsf{pp}, s) \rangle = b$, where $b = 1$ if $\mathcal{V}$ accepts and $b = 0$ if $\mathcal{V}$ rejects.

Definition 3 (Argument of Knowledge)**.** *We call the triple* $(\mathsf{Pgen}, \mathcal{P}, \mathcal{V})$ *an argument of knowledge for relation* $\mathcal{R}$ *if it has completeness and witness-extended emulation, as defined below.*

Definition 4 (Perfect Completeness)**.** $(\mathsf{Pgen}, \mathcal{P}, \mathcal{V})$ *has perfect completeness if for all non-uniform polynomial-time adversaries* $\mathcal{A}$*,*

$$\Pr\left[(s,w) \notin \mathcal{R} \vee \langle \mathcal{P}(\mathsf{pp}, s, w), \mathcal{V}(\mathsf{pp}, s) \rangle = 1 \ : \ \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Pgen}(1^\lambda), \\ (s,w) \leftarrow \mathcal{A}(\mathsf{pp}) \end{array}\right] = 1.$$

Definition 5 (Witness-Extended Emulation [30,31])**.** $(\mathsf{Pgen}, \mathcal{P}, \mathcal{V})$ *has witness-extended emulation if for every deterministic polynomial-time prover* $\mathcal{P}^*$ *there exists an expected polynomial-time emulator* $\mathcal{E}$ *such that for all non-uniform polynomial-time adversaries* $\mathcal{A}$*, the difference between the following two probabilities is less than or equal to* $\mathsf{negl}(\lambda)$*:*

$$\Pr\left[\mathcal{A}(\mathsf{tr}) = 1 \ : \ \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Pgen}(1^\lambda), \\ (s, \mathsf{st}) \in \mathcal{A}(\mathsf{pp}), \\ \mathsf{tr} \leftarrow \langle \mathcal{P}^*(\mathsf{pp}, s, \mathsf{st}), \mathcal{V}(\mathsf{pp}, s) \rangle \end{array}\right] \text{ and}$$

$$\Pr\left[\mathcal{A}(\mathsf{tr}) = 1 \ \wedge \ \text{if } \mathsf{tr} \text{ is accepting, then } (s,w) \in \mathcal{R} \ : \ \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Pgen}(1^\lambda), \\ (s, \mathsf{st}) \in \mathcal{A}(\mathsf{pp}), \\ \mathsf{tr} \leftarrow \langle \mathcal{P}^*(\mathsf{pp}, s, \mathsf{st}), \mathcal{V}(\mathsf{pp}, s) \rangle \end{array}\right],$$

where the oracle called by $\mathcal{E}^{\langle \mathcal{P}^*(\mathsf{pp}, s, \mathsf{st}), \mathcal{V}(\mathsf{pp}, s) \rangle}$ *permits rewinding to any round and running again on fresh verifier randomness, and* st *is the initial state of* $\mathcal{P}^*$*.*

Definition 6 (Public Coin)**.** *An argument system* $(\mathsf{Pgen}, \mathcal{P}, \mathcal{V})$ *is called public coin if the verifier chooses its messages uniformly at random, and independently of the messages sent by the prover, i.e., the challenges correspond to the verifier's randomness.*

We recall special honest verifier zero-knowledge, which states that the view of the verifier can be simulated if the verifier follows the protocol honestly and if challenges made by the verifier are known in advance.

Definition 7 (Perfect SHVZK)**.** *A public coin argument system* $(\mathsf{Pgen}, \mathcal{P}, \mathcal{V})$ *is called a perfect special honest verifier zero-knowledge (SHVZK) argument for relation* $\mathcal{R}$ *if there exists a probabilistic polynomial-time simulator* Sim *such that for all interactive non-uniform polynomial-time adversaries* $\mathcal{A}$*,*

$$\Pr\left[\mathcal{A}(tr) = 1 \ \wedge \ (s,w) \in \mathcal{R} \ : \ \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Pgen}(1^\lambda), \\ (s,w,\rho) \leftarrow \mathcal{A}(\mathsf{pp}), \\ \mathsf{tr} \leftarrow \langle \mathcal{P}(\mathsf{pp}, s, w), \mathcal{V}(\mathsf{pp}, s) \rangle \end{array}\right]$$

$$= \Pr\left[\mathcal{A}(tr) = 1 \ \wedge \ (s,w) \in \mathcal{R} \ : \ \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Pgen}(1^\lambda), \\ (s,w,\rho) \leftarrow \mathcal{A}(\mathsf{pp}), \\ \mathsf{tr} \leftarrow \mathsf{Sim}(s,\rho) \end{array}\right],$$

where ρ is the public coin randomness used by the verifier.

The general forking lemma [6,14] is useful for proving that an argument system has witness-extended emulation. Consider a public coin interactive argument system with r rounds. We view $\prod_{i=1}^{r} n_i$ distinct accepting transcripts as having a tree format with depth r and $\prod_{i=1}^{r} n_i$ leaves, which we call an $(n_1, \ldots, n_r)$-tree. For $1 \leq i \leq r$, let c_i be the i-th round challenge chosen among exactly $n_i \geq 1$ values. The root node is labeled with a statement s and has exactly n_1 children labeled with a distinct value for c_1, where each edge from the root to a child is labeled with a message from the prover to the verifier on c_1. Similarly, each node in depth $1 \leq i \leq r-1$ is labeled with a distinct value for c_i and has n_{i+1} children labeled with a distinct value for c_{i+1}, where each edge from c_i to c_{i+1} is labeled with a message from the prover to the verifier on c_i. Note that each path from the root to a leaf then corresponds to an accepting transcript.

Lemma 1 (General Forking Lemma [6,14])**.** *Let $(\mathsf{Pgen}, \mathcal{P}, \mathcal{V})$ be a public coin argument system for relation $\mathcal{R}$ with r rounds. Let χ be a witness extraction algorithm that succeeds with overwhelming probability in extracting a witness from an $(n_1, \ldots, n_r)$-tree of accepting transcripts in probabilistic polynomial time. If $\prod_{i=1}^{r} n_i$ is bounded above by a polynomial in the security parameter λ, $(\mathsf{Pgen}, \mathcal{P}, \mathcal{V})$ has witness-extended emulation.*

3.3. Commitment Schemes

We review the definitions and security properties regarding the polynomial commitment schemes. In the following, we use a tuple $(a_0, \ldots, a_n; b_0, \ldots, b_{n'})$ for arguments or a returned tuple of the prover $\mathcal{P}$ and the verifier $\mathcal{V}$. In a tuple, $(a_0, \ldots, a_n)$ before the semicolon and $(b_0, \ldots, b_{n'})$ after it denotes public variables known to both $\mathcal{P}$ and $\mathcal{V}$, and secret variables known to only $\mathcal{P}$, respectively.

Definition 8 (Commitment Scheme)**.** *A commitment scheme is a triple* $(\mathsf{Cgen}, \mathsf{Commit}, \mathsf{Open})$ *of probabilistic polynomial-time algorithms defined as follows:*

- $\mathsf{pp} \leftarrow \mathsf{Cgen}(1^\lambda)$ *takes the security parameter λ on input, and outputs the public parameter* pp*, which specifies a message space, a randomness space, and a commitment space;*
- $(c; r) \leftarrow \mathsf{Commit}(\mathsf{pp}; m, r)$ *takes a secret message m and an optional random r chosen uniformly at random on input and returns a commitment c and (optionally) a secret opening hint r;*
- $b \in \{0,1\} \leftarrow \mathsf{Open}(\mathsf{pp}, c, m, r)$ *verifies the commitment c to the message m provided with the opening hint r. It outputs $b = 1$ if the commitment is valid and $b = 0$ otherwise.*

A commitment scheme is binding if for all non-uniform polynomial-time adversaries $\mathcal{A}$,

$$\Pr\left[b_0 = b_1 \neq 0 \ \wedge \ m_0 \neq m_1 \ : \ \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Cgen}(1^\lambda), \\ (c, m_0, m_1, r_0, r_1, \rho) \leftarrow \mathcal{A}(\mathsf{pp}), \\ b_0 \leftarrow \mathsf{Open}(\mathsf{pp}, c, x_0, r_0), \\ b_1 \leftarrow \mathsf{Open}(\mathsf{pp}, c, x_1, r_1) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

A commitment scheme is hiding if for all non-uniform polynomial-time adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$,

$$\left|\frac{1}{2} - \Pr\left[b' = b \ : \ \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Cgen}(1^\lambda), \\ (m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}_0(\mathsf{pp}), \\ b \xleftarrow{\$} \{0,1\}, \\ (c; r) \leftarrow \mathsf{Commit}(\mathsf{pp}; m_b, r), \\ b' \leftarrow \mathcal{A}_1(\mathsf{st}, c) \end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

In a polynomial commitment scheme, $\mathcal{V}$ additionally checks whether the evaluation at any point is correct with respect to the committed polynomial $f(X)$ given by $\mathcal{P}$. The below definition of polynomial commitment schemes is given by Bünz et al. [6], which extends that of Kate et al. [10].

Definition 9 (Polynomial Commitment Scheme [6,10])**.** *Let* $(\mathsf{Cgen}, \mathsf{Commit}, \mathsf{Open})$ *be a commitment scheme for a message space* $R[X]$ *over a ring* R*. A polynomial commitment scheme additionally consists of a protocol* Eval *as follows:*

- $b \in \{0,1\} \leftarrow \mathsf{Eval}(\mathsf{pp}, c, x, y, d; f(X), r)$ *is an interactive public coin protocol between* $\mathcal{P}$ *and* $\mathcal{V}$*. Both* $\mathcal{P}$ *and* $\mathcal{V}$ *have as input a commitment* c*, points* $x, y \in R$*, and a degree* d*. In addition,* $\mathcal{P}$ *knows the opening of* c *to a secret polynomial* $f(X) \in R[X]$ *with* $\deg(f(X)) \leq d$ *and a secret opening hint* r*.* $\mathcal{P}$ *convinces* $\mathcal{V}$ *that* $f(x) = y$ *by applying the protocol.*

3.4. Privacy-Preserving Blockchain with SNARKs

Recently, SNARKs have been receiving a lot of attention from the blockchain industry as a solution for balancing privacy and publicly-verifiable integrity. For instance, Zcash employs SNARKs to provide Bitcoin with user anonymity and privacy of transaction data with anonymous coins [1]. SNARKs are also used to verify Ethereum smart contracts over private input [2].

Figure 1 presents a high-level architecture of privacy-preserving blockchains with SNARKs. A typical way that SNARKs are used in blockchains is as follows. The real data is stored in off-chain storage. The data posted to the on-chain blockchain (blockchain ledger) consist of the commitment to the transaction and its proof that the target transaction is valid. Cryptographic commitment schemes ensure that it is very difficult to obtain the original input value from the committed value, and the proof generated using SNARKs can be verifiable by any node in the blockchain network. Therefore, the privacy problem is solved because the data is hidden in the public on-chain blockchain. In addition, since zero-knowledge techniques provide fast verification, they are being used in various ways to improve the performance and minimize the size of the blockchain. It is worth noting that a polynomial commitment scheme is a key building block to compile a polynomial IOP system, which is a formal representation of a proving statement, into a SNARK [3,6].

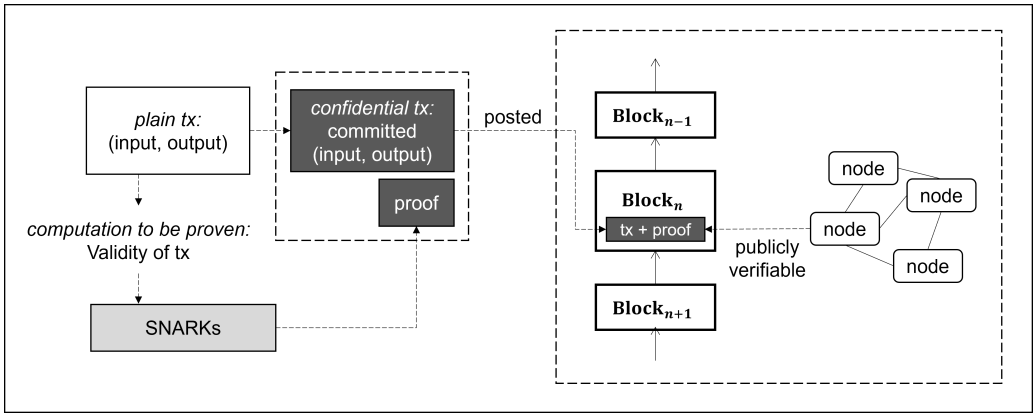

Figure 1. Overview of blockchain with SNARKs.

4. Our Approach

In this section, we present our approach to transpose Bünz et al.'s techniques in the discrete log setting. We investigate and identify a sufficient condition for a transposition. Specifically, we show how to employ a proof system for the equality of discrete logarithms. For the rest of this paper, we encode a polynomial $f(X) = \sum_{i=0}^{d-1} f_i X^i$ over a field $\mathbb{F}$ into a vector $\boldsymbol{f} = (f_0, \ldots, f_{d-1}) \in \mathbb{F}^d$. For a group $\mathbb{G}$ let $\boldsymbol{g}$ be a vector $(g_0, \ldots, g_{\ell-1}) \in \mathbb{G}^\ell$ for some positive integer ℓ. For $d \leq \ell$ we denote the multi-exponentiation $\prod_{i=0}^{d-1} g_i^{f_i}$ by $\boldsymbol{g}^{\boldsymbol{f}}$. When it is clear from the context, we write the commitment to a polynomial $f(X)$ as $\mathsf{Commit}(f)$

instead of $\mathsf{Commit}(\mathsf{pp}; f(X), r)$ for the sake of convenience. We also take a finite field $\mathbb{F}$ as $\mathbb{Z}_p$ for a prime p. Table 1 presents frequently-used notations in the paper.

Table 1. Notations.

Notation	Definition
$\mathcal{P}, \mathcal{V}$	the prover and verifier of a proof system
$\mathbb{F}$	finite field (usually $\mathbb{Z}_p$ with a prime p)
$\mathbb{G}$	a group of a prime order p
$\lvert\mathbb{G}\rvert$	the size of an element of $\mathbb{G}$
$[\mathbb{G}]$	the computation cost for group operation in $\mathbb{G}$
$f(X)$, $\deg(f(X))$	a polynomial $f(X) \in \mathbb{F}[X]$ and its degree, respectively
$f_L(X)$, $f_R(X)$	the left and right half parts of a polynomial $f(X)$, respectively
$\boldsymbol{f}$	the vector representation $(f_0, \ldots, f_{d-1})$ of a polynomial $f(X) = \sum_{i=0}^{d-1} f_i X^i \in \mathbb{F}[X]$
$\boldsymbol{g}$	$(g_0, \ldots, g_\ell) \in \mathbb{G}^\ell$ for some positive integer ℓ
$\boldsymbol{g}^{\boldsymbol{f}}$	$\prod_{i=0}^{d-1} g_i^{f_i}$ for $\boldsymbol{g} = (g_0, \ldots, g_\ell) \in \mathbb{G}^\ell$ and $\boldsymbol{f} = (f_0, \ldots, f_{d-1})$ where $d \leq \ell$
$\mathsf{Commit}(f)$	the commitment to a polynomial $f(X) \in \mathbb{F}[X]$
$\mathsf{Commit}_{\mathsf{H}}(f)$	the hiding commitment to a polynomial $f(X) \in \mathbb{F}[X]$
PoE_{mDL}	a proof system for equality of discrete logarithms over multiple bases
$\lvert PoE_{mDL}\rvert$	the communication complexity of PoE_{mDL}
$[PoE_{mDL}]_{\mathcal{P}}$, $[PoE_{mDL}]_{\mathcal{V}}$	the computation costs of $\mathcal{P}$ and $\mathcal{V}$ for PoE_{mDL}, respectively

4.1. Bünz et al.'s Abstraction

Bünz et al. [6] presented a polynomial commitment scheme for their construction of SNARKs. The proposed scheme operates in a recursive way by reducing the degree of polynomial f by half during each iteration, and hence, there are $\log \deg(f)$ iterations overall. More precisely, given an odd degree d polynomial $f(X) = \sum_{i=0}^{d} f_i X^i$, the prover splits it into two polynomials

$$f_L(X) = \sum_{i=0}^{\frac{d+1}{2}-1} f_i X^i \text{ and } f_R(X) = \sum_{i=0}^{\frac{d+1}{2}-1} f_{\frac{d+1}{2}+i} X^i,$$

which both polynomials have a degree of roughly $d/2$ satisfying

$$f(X) = f_L(X) + X^{d/2} f_R(X). \tag{1}$$

The prover then sends the verifier the commitments $\mathsf{Commit}(f_L)$ and $\mathsf{Commit}(f_R)$ to $f_L(X)$ and $f_R(X)$, respectively. At the end of each iteration, the prover takes the next input polynomial as

$$f(X) \leftarrow f_L(X) + \alpha f_R(X) \tag{2}$$

for a random α received from the verifier.

Because the verifier needs to check if $f(x) = y$ from committed polynomials, the verifier should be able to homomorphically compute a committed form of the current $f(X)$ and the next $f(X)$ from $\mathsf{Commit}(f_L)$ and $\mathsf{Commit}(f_R)$ to see whether (1) holds. The verifier also needs to compute a commitment to a polynomial in (2) for the next iteration from $\mathsf{Commit}(f_L)$ and $\mathsf{Commit}(f_R)$.

To support the computation of the committed form, Bünz et al. [6] define the following two abstract properties.

- linear homomorphism
  $\mathsf{Commit}(f)^a \cdot \mathsf{Commit}(g)^b = \mathsf{Commit}(a \cdot f + b \cdot g)$ for polynomials f and g, and scalars a and b (a linear homomorphism);

- monomial homomorphism
  $\mathsf{Commit}(f)^{X^d} = \mathsf{Commit}(X^d \cdot f)$ for some bounded positive integer d.

Figure 2 illustrates Bünz et al.'s abstraction for a polynomial commitment scheme with a recursive argument from the above two properties.

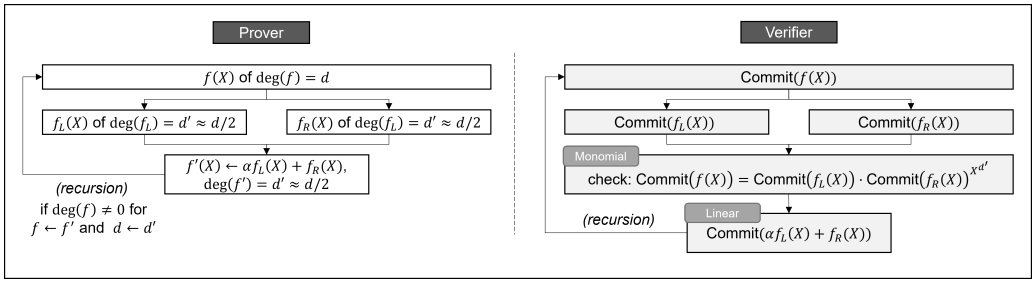

Figure 2. Bünz et al.'s approach for the recursive argument. The prover recursively reduces the degree of f at each iteration over a plain f. The verifier recursively reduces the degree of f at each iteration over a committed form of f using monomial and linear homomorphic properties.

4.2. Base Commitment Scheme to Polynomial

We construct a polynomial commitment scheme based on a generalization of the Pedersen commitment scheme [32]. In the generalized Pedersen commitment, pp consists of a group $\mathbb{G}$ of a prime order p, and group elements $g, g_0, ..., g_{n-1} \xleftarrow{\$} \mathbb{G}$. For a secret message vector $\boldsymbol{m} = (m_0, \ldots, m_{n-1}) \in \mathbb{Z}_p^n$ the prover chooses $r \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\left(g^r \prod_{i=0}^{n-1} g_i^{m_i}; r \right) \leftarrow \mathsf{Commit}(\mathsf{pp}; \boldsymbol{m}, r).$$

It is well-known that the generalized Pedersen commitment is perfectly hiding and computationally binding under the discrete logarithm relation assumption [32]. It is also important to note that the generalized Pedersen commitment scheme is homomorphic, i.e.,

$$\mathsf{Commit}(\mathsf{pp}; \boldsymbol{m}, r) \cdot \mathsf{Commit}(\mathsf{pp}; \boldsymbol{m'}, r') = \mathsf{Commit}(\mathsf{pp}; \boldsymbol{m} + \boldsymbol{m'}, r + r').$$

We consider a commitment scheme, which does not use the randomness in the generalized Pedersen commitment scheme, as follows:

- $\mathsf{Cgen}(1^\lambda)$: On input of the security parameter λ, it first samples $\mathbb{G} \leftarrow \mathsf{Ggen}(1^\lambda)$ of a prime order p of length λ. It then chooses $g_0, \ldots, g_d \xleftarrow{\$} \mathbb{G}$ and returns $\mathsf{pp} = (\mathbb{G}, p, \boldsymbol{g})$ where $\boldsymbol{g} = (g_0, \ldots, g_d)$.
- $\mathsf{Commit}(\mathsf{pp}; f(X))$: For a secret polynomial $f(X) = \sum_{i=0}^{d} f_i X^i \in \mathbb{Z}_p[X]$ with $\deg(f(X)) \leq d$, it computes and outputs $c \leftarrow \boldsymbol{g}^{\boldsymbol{f}} = \prod_{i=0}^{d} g_i^{f_i}$.
- $\mathsf{Open}(\mathsf{pp}, c, f(X))$: On input c and $f(X)$, the verifier computes $c' \leftarrow \prod_{i=0}^{d} g_i^{f_i}$ and checks if $c = c'$ in $\mathbb{G}$.

Because the generalized Pedersen commitment scheme is computationally binding under the discrete logarithm relation assumption, so is the commitment scheme above.

4.3. Evaluation Protocol

As presented in Section 4.1, Bünz et al.'s approach considers two properties (a linear homomorphism and a monomial homomorphism) for the underlying commitment schemes, which is crucial for the verifier to compute (1) from the commitments to the polynomials f_L and f_R on the right-hand side. However, our base commitment scheme does not provide monomial homomorphism, while it immediately holds the linear homomorphic property. A monomial homomorphic commitment scheme is not known thus far in the discrete log setting with no trusted setup. This is because a monomial homomorphic property may

require some special structures for base elements in the group, which is impossible to generate without a trusted setup.

Thus, our approach focuses on providing a way to check the integrity of f_L and f_R, i.e., $f = f_L + X^{d/2} \cdot f_R$ using the linear homomorphic property only. The idea behind our approach is presented in Figure 3. To avoid monomial homomorphism, our approach simply lets the prover send one additional commitment to $X^{d/2} \cdot f_R$ besides two commitments to f_L and f_R so that $f = f_L + X^{d/2} \cdot f_R$ can be verified using the linear homomorphic property.

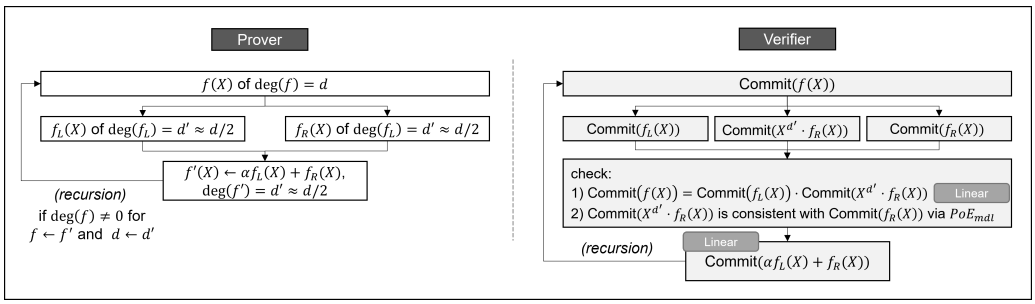

Figure 3. Our approach for recursive argument. Aside from two commitments, $\mathsf{Commit}(f_L)$ and $\mathsf{Commit}(f_R)$, the verifier additionally receives $\mathsf{Commit}(X^{d'} \cdot f_R)$ to confirm they properly come from the input polynomial f using linear homomorphic property only.

We present the evaluation protocol Eval in Algorithm 1, which is a transposition of Bünz et al.'s construction [6] in the discrete log setting. In Line 13 of the Eval protocol, the prover sends one additional commitment $c_{RR} \leftarrow \mathsf{Commit}(X^{d/2} \cdot f_R)$. The verifier is then able to compute $\mathsf{Commit}(f) = \mathsf{Commit}(f_L) \cdot \mathsf{Commit}(X^{d/2} \cdot f_R)$ using a linear homomorphic property of the underlying commitment scheme (Line 13). However, because the polynomial $f_R(X)$ is committed to c_R and c_{RR} independently, it is required for the prover to prove that c_R and c_{RR} are generated from the same polynomial $f_R(X)$. More precisely, given public parameter $\mathsf{pp}(= \{g_0, \ldots, g_d\})$ and two target instances $c_R \left(= \prod_{i=0}^{d'} g_i^{f_{d'+1+i}}\right)$ and $c_{RR} \left(= \prod_{i=0}^{d'} g_{d'+1+i}^{f_{d'+1+i}}\right)$, the prover needs to convince the verifier that c_R and c_{RR} have the same exponents. Thus, we require a proof for equality of discrete logarithms, which is invoked as a sub-protocol PoE_{mDL} in the Eval protocol (Line 12). PoE_{mDL} takes pp, c_R, c_{RR}, and $\deg(f_R(X)) = d'$ on input and returns 1 if c_R and c_{RR} have the same exponents and 0 otherwise. If the returned value is 0, $\mathcal{V}$ aborts the Eval protocol because $\mathcal{P}$ is a cheating prover. We remark that several cryptographic protocols on PoE_{mDL} have been proposed [33,34]. The works for PoE_{mDL} have been developed independently from the construction of polynomial commitment schemes.

Algorithm 1 $\mathsf{Eval}(\mathsf{pp}, c \in \mathbb{G}, x \in \mathbb{Z}_p, y \in \mathbb{Z}_p, d \in \mathbb{N}; f(X) \in \mathbb{Z}_p[X])$

Common input: public parameter pp, commitment $c = \prod_{i=0}^{d} g_i^{f_i}$ to $f(X)$, point $(x, y = f(x))$, degree bound d

Prover's witness: secret polynomial $f(X) = \sum_{i=0}^{d} f_i X^i$

1: **if** $d = 0$ **then**
2: $\mathcal{P}$ sends $f(X) = f_0 \in \mathbb{Z}_p$ to $\mathcal{V}$ // $f(X)$ is a constant f_0
3: $\mathcal{V}$ checks that $f_0 = y$ in $\mathbb{Z}_p$
4: $\mathcal{V}$ checks that $g_0^{f_0} = c$ in $\mathbb{G}$
5: $\mathcal{V}$ returns 1 if all checks pass, 0 otherwise
6: **else**
7: $\mathcal{P}$ and $\mathcal{V}$ compute $d' \leftarrow \lfloor \frac{d}{2} \rfloor$
8: $\mathcal{P}$ sets $f_L(X) \leftarrow \sum_{i=0}^{d'} f_i X^i$ and $f_R(X) \leftarrow \sum_{i=0}^{d'} f_{d'+1+i} X^i$ // If d is even, then $f_{2d'+1} = 0$
9: $\mathcal{P}$ computes $c_L \leftarrow \prod_{i=0}^{d'} g_i^{f_i}$, $c_R \leftarrow \prod_{i=0}^{d'} g_i^{f_{d'+1+i}}$, and $c_{RR} \leftarrow \prod_{i=0}^{d'} g_{d'+1+i}^{f_{d'+1+i}}$ in $\mathbb{G}$
10: $\mathcal{P}$ computes $y_L \leftarrow f_L(x)$ and $y_R \leftarrow f_R(x)$ in $\mathbb{Z}_p$
11: $\mathcal{P}$ sends $\{c_L, c_R, c_{RR}, y_L, y_R\}$ to $\mathcal{V}$
12: $\mathcal{P}$ and $\mathcal{V}$ run $PoE_{mDL}(pp, c_R, c_{RR}, d')$ // Checking that $mDL(c_R) = mDL(c_{RR})$
13: $\mathcal{V}$ checks that $c = c_L \cdot c_{RR}$ in $\mathbb{G}$ and returns 0 if the equation does not hold
14: $\mathcal{V}$ checks that $y = y_L + y_R \cdot x^{d'+1}$ in $\mathbb{Z}_p$ and returns 0 if the equation does not hold
15: $\mathcal{V}$ chooses $\alpha \xleftarrow{\$} \mathbb{Z}_p$ and sends it to $\mathcal{P}$
16: $\mathcal{P}$ and $\mathcal{V}$ compute $c' \leftarrow (c_L)^{\alpha} \cdot c_R$ in $\mathbb{G}$ and $y' \leftarrow \alpha \cdot y_L + y_R$ in $\mathbb{Z}_p$
17: $\mathcal{P}$ computes $f'(X) \leftarrow \alpha \cdot f_L(X) + f_R(X)$ in $\mathbb{Z}_p[X]$ // $\deg(f'(X)) = d'$
18: $\mathcal{P}$ and $\mathcal{V}$ run $\mathsf{Eval}(\mathsf{pp}, c', x, y', d'; f'(X))$

5. Discussion: Performance & Security Analysis

Let $\Pi = (\mathsf{Cgen}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Eval})$ be the polynomial commitment scheme described in Section 4. In this section, we analyze the performance and security of the Eval protocol.

5.1. Performance

We analyze the efficiency of our approach in comparison with the recently proposed schemes with a transparent setting found in the literature. For a concrete performance analysis, we borrow the examples of groups and parameters at the 128-bit security level given by Lee [22], which is presented in Table 2. In Table 2, $\mathbb{G}$ denotes a cyclic group of known order, which is implemented by curve25519-dalek [35]. An imaginary class group $\mathbb{G}_U$ [36] is taken as an example group of unknown order. The discriminant of $\mathbb{G}_U$ is fixed as $\Delta = -(2^{6656} - 26,745)$, which is estimated to offer the 128-bit security level [13]. This is implemented by ANTIC [37]. For a pairing-based construction, $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ denote the two source groups and the target group of a pairing P. The groups of pairing are implemented by RELIC [38] over the curve BLS12-381 [39].

We analyze the efficiency of our approach. Let $|\mathbb{G}|$ and $|PoE_{mDL}|$ be the size of an element in G and the communication complexity of PoE_{mDL}, respectively. Let $[\mathbb{G}]$, $[PoE_{mDL}]_{\mathcal{P}}$, and $[PoE_{mDL}]_{\mathcal{V}}$ be the computation cost in $\mathbb{G}$, the prover's computation cost for PoE_{mDL}, and the verifier's computation cost for PoE_{mDL}, respectively. Below we focus on the dominated terms for each complexity comprising the transmission and operations of the group elements, i.e., we neglect operations over a field $\mathbb{Z}_p$. The Eval protocol makes recursive calls roughly $\log d$ times. The messages between the prover and the verifier consist of $\log d$ times of three elements in $\mathbb{G}$, and $|PoE_{mDL}|$, respectively. Thus, the communication complexity is equal to $3 \log d \cdot |\mathbb{G}| \cdot |PoE_{mDL}|$. The prover applies $\log d$ times of three multi-exponentiations [40] of roughly $d/2$ size and one operation over $\mathbb{G}$, and PoE_{mDL} on the prover's side. This leads to $O(d) \cdot [G] \cdot [PoE_{mDL}]_{\mathcal{P}}$ computation complexity for the

prover. The verifier applies $\log d$ times of one exponentiation and two operations over $\mathbb{G}$, and PoE_{mDL} on the verifier's side, which leads to $O(\log d) \cdot [\mathbb{G}] \cdot [PoE_{mDL}]_{\mathcal{V}}$ computation complexity overall. The size of public parameter pp is $d \cdot |\mathbb{G}| \cdot |PoE_{mDL}|$.

We now provide a comparison for polynomial commitment schemes [6,22] with a transparent setting, which achieves a logarithmic verifier complexity in Table 3. The table focuses on the dominated terms for each complexity comprising the transmission and operations of the group elements. As mentioned above, We apply multi-exponentiation techniques [40] to both our approach and Bünz et al.'s construction to reduce the prover complexity by a factor of $\log d$. In the case of Bünz et al.'s construction, it is possible to reduce the size of public parameter pp to a single element of $\mathbb{G}_U$ when multi-exponentiation techniques are not applied. Table 3 summarizes the efficiency analysis on communication and computation complexities, and the size of the public parameter for recent polynomial commitment schemes with transparent setting and our approach.

Table 3 shows that the efficiency of our approach depends on that of PoE_{mDL}. If PoE_{mDL} has a constant communication/computation complexity, we observe that each complexity is almost the same across the schemes, and the efficiency of a scheme depends highly on the underlying group. The benchmark result on base groups in Table 2 shows that the sizes of the element in $\mathbb{G}_U$ and $\mathbb{G}_T$ are approximately 25× and 6× larger than that of $\mathbb{G}$, respectively. For the operation time, $\mathbb{G}_U$ and $\mathbb{G}_T$ are approximately 844× and 18× slower than that of $\mathbb{G}$, respectively.

Table 2. Efficiency comparison between groups. The time is measured for operations to multiply a random point (element) of a group by a 256-bit scalar.

Group, Operation		Size (bytes)	Time (μs)
Elliptic curve (EC) group	$\mathbb{G}$	32	45
Class group	$\mathbb{G}_U$	832	38,000
EC group with pairing	$\mathbb{G}_1$	48	220
	$\mathbb{G}_2$	96	490
	$\mathbb{G}_T$	192	820
EC pairing operation	P	-	1600

Table 3. Comparison between polynomial commitment schemes with a transparent setting. $|\cdot|$ and $[\cdot]$ denote the size of an element and the computation cost of a group operation in the corresponding group, respectively. We express communication complexity in the number of group elements and computation complexity in a number of group operations.

Scheme	Bünz et al. [6]	Lee [22]	Our Approach
Communication	$3 \cdot \|\mathbb{G}_U\| \cdot \log d$	$6 \cdot \|\mathbb{G}_T\| \cdot \log d$	$3 \cdot \|\mathbb{G}\| \cdot \|PoE_{mDL}\| \cdot \log d$
Prover Computation	$[\mathbb{G}_U] \cdot O(d)$	$[\mathbb{G}_1] \cdot O(d)$	$[\mathbb{G}] \cdot [PoE_{mDL}]_{\mathcal{P}} \cdot O(d)$
Verifier Computation	$[\mathbb{G}_U] \cdot O(\log d)$	$[\mathbb{G}_T] \cdot O(\log d)$	$[\mathbb{G}] \cdot [PoE_{mDL}]_{\mathcal{V}} \cdot O(\log d)$
Size of pp	$\|\mathbb{G}_U\| \cdot d$	$(\|\mathbb{G}_1\| + \|\mathbb{G}_2\|) \cdot d$	$\|\mathbb{G}\| \cdot \|PoE_{mDL}\| \cdot d$

The above discussion shows that our scheme based on an elliptic curve group in the discrete log setting is very promising in the case that we have an efficient PoE_{mDL}. Unfortunately, currently known PoE_{mDL} protocols have $O(d)$ communication complexity for the number of bases, i.e., degree of a polynomial in our setting, which is not desirable for our purpose of logarithmic complexity. However, we emphasize that it is meaningful to observe that two independent cryptographic primitives are closely connected and suggest a stepping stone for the construction of an efficient, transparent polynomial commitment scheme with a recursive argument in the discrete log setting.

5.2. Security

We analyze perfect completeness (Definition 4) and witness-extended emulation (Definition 5) of the proposed polynomial commitment scheme.

Theorem 1. *The* Eval *protocol of the polynomial commitment scheme* Π *has perfect completeness.*

Proof of Theorem 1. First, we show that the case of $d = 0$ satisfies the perfect completeness. When $d = 0$, the valid input consists of the constant polynomial $f(X) = f_0$, $c = g_0^{f_0} \leftarrow \mathsf{Commit}(\mathsf{pp}; f(X))$, and $y = f_0$. Thus, the verification equations checked by $\mathcal{V}$ immediately hold.

Next, we consider the case of $d > 0$. For the polynomial $f(X) = \sum_{i=0}^{d} f_i X^i \in \mathbb{Z}_p[X]$, let $t_{in} \leftarrow (c, x, y, d; f(X))$ and $t_{out} \leftarrow (c', x, y', d'; f'(X))$ be the input and output tuples for every recursive step in the Eval protocol. For the perfect completeness, it suffices to show that t_{out} satisfies the relations $c' = \boldsymbol{g}^{\boldsymbol{f}'}$, $y' = f'(x)$, and $\deg(f'(X)) \leq d'$ when the relations $c = \boldsymbol{g}^{\boldsymbol{f}}$, $y = f(x)$, and $\deg(f(X)) \leq d$ hold for t_{in}.

When $d+1$ is odd, we can see that $(c', y', f'(X))$ from t_{out} is exactly equal to $(c, y, f(X))$ from t_{in} and $\deg(f'(X)) = \deg(f(X)) \leq d \leq d' = d+1$. Thus, the relation holds for t_{out}. When $d+1$ is even, we have $f_L(X)$ and $f_R(X)$, such that $f(X) = f_L(X) + X^{\frac{d+1}{2}} f_R(X)$ and $f'(X) = \alpha \cdot f_L(X) + f_R(X)$. Thus, we can see that the following equations hold:

$$
\begin{aligned}
c' &= (c_L)^{\alpha} \cdot c_R = \boldsymbol{g}^{\alpha \boldsymbol{f_L}} \cdot \boldsymbol{g}^{\boldsymbol{f_R}} = \boldsymbol{g}^{\alpha \boldsymbol{f_L} + \boldsymbol{f_R}} = \boldsymbol{g}^{\boldsymbol{f}'}, \\
y' &= \alpha \cdot y_L + y_R = \alpha \cdot f_L(x) + f_R(x) = f'(x), \\
c &= \boldsymbol{g}^{\boldsymbol{f}} = \prod_{i=0}^{d} g_i^{f_i} = \prod_{i=0}^{\frac{d+1}{2}-1} g_i^{f_i} \cdot \prod_{i=\frac{d+1}{2}}^{d} g_i^{f_i} = \prod_{i=0}^{\frac{d+1}{2}-1} g_i^{f_i} \cdot \prod_{i=0}^{\frac{d+1}{2}-1} g_{\frac{d+1}{2}+i}^{f_{\frac{d+1}{2}+i}} = c_L \cdot c_{RR}, \\
y &= y_L + y_R \cdot x^{d'} = f_L(x) + x^{\frac{d+1}{2}} f_R(x) = f(x).
\end{aligned}
$$

Finally, we have,

$$\deg f'(X) \leq \max\{\deg f_L(X), \deg f_R(X)\} \leq d' = \frac{d+1}{2} - 1.$$

This completes the proof of the perfect completeness. □

We now prove that the Eval protocol is sound, i.e., it has witness-extended emulation. In brief, we need to show that we can extract a witness polynomial $f(X)$ from a tree of accepting transcripts, where the number of transcripts is bounded in a polynomial in λ. This can be done by extracting an intermediate secret polynomial at each iteration of *Eval*, i.e., from Level $i+1$ to Level i in the tree. In Lemma 2, we first show that given two accepting transcripts, we can extract an intermediate witness polynomial at each iteration of the Eval protocol. We then prove the witness-extended emulation for the whole Eval protocol by using the lemma from leaf nodes to the root node sequentially in Theorem 2.

Lemma 2. *Let* $\mathsf{pp} = (\mathbb{G}, p, g_0, \ldots, g_d)$ *be the public parameter generated by* Ggen. *Suppose we have two accepting transcripts* $(x, c_L, c_R, c_{RR}, y_L, y_R, \alpha, f(X), y)$ *and* $(x, c_L, c_R, c_{RR}, y_L, y_R, \alpha', f'(X), y')$ *for two distinct numbers* $\alpha, \alpha' \in \mathbb{Z}_p$, *such that* $\boldsymbol{g}^{\boldsymbol{f}} = c_L^{\alpha} c_R$ *and* $\boldsymbol{g}^{\boldsymbol{f}'} = c_L^{\alpha'} c_R$. *Furthermore, suppose* $f(X)$ *and* $f'(X)$ *are polynomials in* $\mathbb{Z}_p[X]$ *with a degree of at most* d *and* $y = f(x)$, $y' = f'(x) \in \mathbb{Z}_p$. *Then on the input of the above transcripts, there exists a probabilistic polynomial-time algorithm* $\mathcal{E}$ *that extracts either* $f_L(X), f_R(X) \in \mathbb{Z}_p[X]$ *with a degree of at most* d *such that* $y_L = f_L(x) \in \mathbb{Z}_p$, $y_R = f_R(x) \in \mathbb{Z}_p$, *or a breach of the binding property of the Pedersen commitment scheme relative to* Ggen.

Proof of Lemma 2. Because the two transcripts are valid, it holds that

$$g^f = (c_L)^\alpha \cdot c_R,\ g^{f'} = (c_L)^{\alpha'} \cdot c_R \in \mathbb{G}.$$

We then have

$$g^{f-f'} = (c_L)^{\alpha-\alpha'} = g^{(\alpha-\alpha')f_L},$$
$$g^{\alpha f'-\alpha' f} = (c_R)^{\alpha-\alpha'} = g^{(\alpha-\alpha')f_R}.$$

Thus, $\mathcal{E}$ is able to compute

$$f_L(X) \leftarrow \frac{f(X)-f'(X)}{\alpha-\alpha'},\ f_R(X) \leftarrow \frac{\alpha f'(X)-\alpha' f(X)}{\alpha-\alpha'}$$

from the binding property of the Pedersen commitment scheme. In addition, because it holds that

$$f(x) = y = \alpha \cdot y_L + y_R \text{ and } y' = f'(x) = \alpha' \cdot y_L + y_R,$$

we let $y_L \leftarrow \frac{y-y'}{\alpha-\alpha'}$ and $y_R \leftarrow \frac{\alpha y'-\alpha' y}{\alpha-\alpha'}$. Then, y_L and y_R are identical to the evaluations of the above $f_L(X)$ and $f_R(X)$ at $X = x$, respectively. □

Theorem 2. *The* Eval *protocol has witness-extended emulation for a relation*

$$\mathcal{R}_{\mathsf{Eval}} = \{((c,x,y,d), f(X)) : \deg(f(X)) \le d \ \wedge f(x) = y \ \wedge \mathsf{Open}(\mathsf{pp}, c, f(X)) = 1\}$$

if the discrete logarithm relation assumption holds for Ggen.

Proof of Theorem 2. For witness-extended emulation, we call the general forking lemma (Lemma 1). Thus, we need to construct an expected polynomial-time extractor $\mathcal{E}$ that extracts a witness from a tree whose number of leaves is bounded above by a polynomial in λ.

For the statement $(c,x,y,d) \in L_{\mathcal{R}_{\mathsf{Eval}}}$, we consider the following tree of the accepting transcripts. The root node is labeled with the first input statement (c,x,y,d) to Eval. Including the root node, let N be a node labeled with the statement (c,x,y,d). We denote the corresponding witness polynomial to (c,x,y,d) by $f^{(d)}(X) \in \mathbb{Z}_p[X]$. N has two child nodes as follows. By rewinding the oracle $\langle \mathcal{P}^*, \mathcal{V} \rangle$ two times with two different challenges α_1 and α_2 on the same input statement (c,x,y,d), each child node for the given challenge is labeled with the update statement (c',x,y',d'). Finally, nodes with $d = 0$ are leaf nodes of the tree. Because the number of levels with a branching factor of 2 is bounded by $\lceil \log(d+1) \rceil$, there are at most $2^{1+\lceil \log_2(d+1) \rceil} \le 4(d+1)$ transcripts in total, which is a polynomial in λ.

We now prove that there exists an extractor $\mathcal{E}$ that extracts a witness $f(X)$ from the above tree, which we construct in a recursive way. That is, we construct an extractor $\mathcal{E}^{(d)}$ to extract $f^{(d)}(X)$ for a statement (c,x,y,d) at each node starting from the leaf nodes in the tree. We note that $\mathcal{E}^{(d)}$ for the degree bound d in the root node is a desired extractor $\mathcal{E}$.

We fist consider $\mathcal{E}^{(0)}$ to extract a witness from the leaves of the tree, i.e., $d = 0$. In this case, $\mathcal{E}^{(0)}$ directly obtains a witness $f(X) = f_0 \in \mathbb{Z}_p$ from the transcript given by the prover, such that $f_0 = y$ and $c = g_0^{f_0}$. We now move to the case of $d > 0$. From the construction of the tree, the node has two child nodes, where each node is labeled with the update statement $\left(c', x, y', d' = \lfloor \frac{d}{2} \rfloor\right)$ on the same input statement (c,x,y,d) with two distinct challenges α_1 and α_2. We assume that we have the extractor $\mathcal{E}^{(d')}$ that returns the valid witness $f^{(d')}$ for each child node. We then construct the extractor $\mathcal{E}^{(d)}$. $\mathcal{E}^{(d)}$ extracts $f_L^{(d)}(X)$ and $f_R^{(d)}(X)$ with a degree of at most d' by applying Lemma 2. It then returns

$$f^{(d)}(X) = f_L^{(d)}(X) + f_R^{(d)}(X) X^{d'+1} \in \mathbb{Z}_p[X],$$

whose degree is bounded by $2d' = 2\lfloor \frac{d}{2} \rfloor \leq d$. Because the tree consists of the accepting transcripts, we have $c = g^{f^{(d)}}$ and $f^{(d)}(x) = f_L^{(d)}(x) + f_R^{(d)}(x)x^{d'+1} = y$. Then, by the general forking lemma, we conclude that Π has witness extended emulation. □

6. Extension to Zero-Knowledge Polynomial Evaluation

In this section, we extend the polynomial commitment scheme from Section 4 to a zero-knowledge version. The zero-knowledge protocol enables the prover to convince the verifier that the prover has a polynomial $f(X)$ with $\deg(f) \leq d$ such that $f(x) = y$ for a public point (x, y) but does not leak any other information about f that is formally defined in the notion of perfect SHVZK (Definition 7). For this, we require a hiding commitment scheme to polynomials, such as the generalization of the Pedersen commitment scheme, which uses randomness when generating a commitment [32]. Below, we give a formal description of the generalization of the Pedersen commitment scheme $(\mathsf{Cgen}_\mathsf{H}, \mathsf{Commit}_\mathsf{H}, \mathsf{Open}_\mathsf{H})$ over the polynomials in $\mathbb{Z}_p[X]$.

- $\mathsf{Cgen}_\mathsf{H}(1^\lambda)$: On input of the security parameter λ, it first samples $\mathbb{G} \leftarrow \mathsf{Ggen}(1^\lambda)$ of a prime order p of length λ. It then chooses $g, g_0, \ldots, g_d \xleftarrow{\$} \mathbb{G}$ and returns $\mathsf{pp}_\mathsf{H} = (\mathbb{G}, p, g, g_0, \ldots, g_d) \leftarrow \mathsf{Cgen}(1^\lambda)$.
- $\mathsf{Commit}_\mathsf{H}(\mathsf{pp}_\mathsf{H}; f(X), d, r)$: For a secret polynomial $f(X) = \sum_{i=0}^{d} f_i X^i \in \mathbb{Z}_p[X]$ it selects $r \xleftarrow{\$} \mathbb{Z}_p$ and outputs $c \leftarrow g^r \prod_{i=0}^{d} g_i^{f_i}$ with secret opening information $f(X), d, r$.
- $\mathsf{Open}_\mathsf{H}(\mathsf{pp}_\mathsf{H}, c, f(X), r)$: On input c and $f(X)$, a verifier $\mathcal{V}$ computes $c' \leftarrow g^r \prod_{i=0}^{d} g_i^{f_i}$ and checks if $c = c'$ in $\mathbb{G}$.

We present our zero-knowledge evaluation protocol EvalZK in Algorithm 2. The EvalZK protocol is also obtained by transposing the corresponding zero-knowledge evaluation protocol given by Bünz et al. under the discrete log setting [6]. The basic idea is to mask the prover's secret polynomial with a random polynomial using the blinding technique introduced in [14,19,41] and then run the Eval protocol on it.

Algorithm 2 $\mathsf{EvalZK}(\mathsf{pp}_\mathsf{H}, c_f \in \mathbb{G}, x \in \mathbb{Z}_p, y_f \in \mathbb{Z}_p, d \in \mathbb{N}; f(X) \in \mathbb{Z}_p[X], r_f \in \mathbb{Z}_p)$

Common input: public parameter pp_H, commitment $c_f = g^{r_f} \prod_{i=0}^{d} g_i^{f_i}$ to $f(X)$, point $(x, y_f = f(x))$, degree bound d

Prover's witness: secret polynomial $f(X) = \sum_{i=0}^{d} f_i X^i$, opening hint r_f to c_f

1: $\mathcal{P}$ samples a random polynomial $h(X) = \sum_{i=0}^{d} h_i X^i \xleftarrow{\$} \mathbb{Z}_p[X]$ of degree d
2: $\mathcal{P}$ computes $c_h \leftarrow g^{r_h} \prod_{i=0}^{d} g_i^{h_i}$ in $\mathbb{G}$ for $r_h \xleftarrow{\$} \mathbb{Z}_p$ and $y_h = h(x)$ in $\mathbb{Z}_p$ // $c_h \leftarrow \mathsf{Commit}_\mathsf{H}(\mathsf{pp}_\mathsf{H}; h(X), r_h)$
3: $\mathcal{P}$ sends $\{c_h, y_h\}$ to $\mathcal{V}$
4: $\mathcal{V}$ samples $\alpha_f \xleftarrow{\$} \mathbb{Z}_p$ and sends it to $\mathcal{P}$
5: $\mathcal{P}$ computes $\tilde{f}(X) \leftarrow h(X) + \alpha_f \cdot f(X)$ in $\mathbb{Z}_p[X]$ and $r_{\tilde{f}} \leftarrow r_h + \alpha_f \cdot r_f$ in $\mathbb{Z}_p$
6: $\mathcal{P}$ sends $r_{\tilde{f}}$ to $\mathcal{V}$
7: $\mathcal{P}$ and $\mathcal{V}$ compute $c \leftarrow c_h \cdot c_f^{\alpha_f} \cdot g^{-r_{\tilde{f}}}$ in $\mathbb{G}$ and $y \leftarrow y_h + \alpha_f \cdot y_f$ in $\mathbb{Z}_p$ // $c = \prod_{i=0}^{d} g_i^{\tilde{f}_i} \leftarrow \mathsf{Commit}(\mathsf{pp}; \tilde{f}(X))$ and $y = \tilde{f}(x)$
8: $\mathcal{P}$ and $\mathcal{V}$ run $\mathsf{Eval}(\mathsf{pp}, c, x, y, d; \tilde{f}(X))$

The EvalZK protocol receives a hiding commitment to the prover's secret polynomial $f(x)$ on input, i.e., $c_f \leftarrow \mathsf{Commit}_\mathsf{H}(\mathsf{pp}_\mathsf{H}; f(X), d, r_f)$, which is perfectly indistinguishable to a random element in $\mathbb{G}$. To hand it over to the Eval protocol, it is necessary to remove the randomization part g^{r_f} from $c_f = g^{r_f} \prod_{i=0}^{d} g_i^{f_i}$, which is equal to $\mathsf{Commit}(\mathsf{pp}, f(X))$. However, because this reveals information on $f(x)$, the protocol lets the prover and the verifier collaboratively blind $f(x)$ by $\tilde{f}(X) = h(X) + \alpha f(X)$ (Line 5). Here, $h(X) \in \mathbb{Z}_p[X]$ is a random polynomial selected by the prover (Line 1) and $\alpha \in \mathbb{Z}_p$ is a random number selected by the verifier (Line 4). Consequently, both the prover and the verifier succeed

in generating a non-hiding commitment to $\tilde{f}(X)$ under Π and the point $(x, y = \tilde{f}(x))$, and then start the Eval protocol (Lines 7–8).

Theorem 3. *The* EvalZK *protocol has perfect completeness, witness-extended emulation, and perfect SHVZK for a relation*

$$\mathcal{R}_{\mathsf{EvalZK}} = \left\{ ((c, x, y, d), (f(X), r)) : \begin{array}{l} \deg(f(X)) \leq d \ \wedge f(x) = y \\ \wedge \ \mathsf{Open}_{\mathsf{H}}(\mathsf{pp}, c, f(X), r) = 1 \end{array} \right\}$$

if the discrete logarithm relation assumption holds for Ggen.

Proof of Theorem 3. (perfect completeness) We show that Π_{H} has perfect completeness. Because the Eval protocol has perfect completeness (Theorem 1), it suffices to show that c and y are a valid input to Eval. That is, c is the correct commitment to $\tilde{f}(X) = h(X) + \alpha f(X)$ under Π and y is the evaluation of $\tilde{f}(X)$ at $X = x$ in $\mathbb{Z}_p$. Given $f(X) = \sum_{i=0}^{d} f_i X^i$ of a degree of at most d and $h(X) = \sum_{i=0}^{d} h_i X^i$ of degree d, we have

$$\begin{aligned} c &= c_h \cdot c_f^{\alpha_f} \cdot g^{-r_{\tilde{f}}} \\ &= \left(g^{r_h} \prod_{i=0}^{d} g_i^{h_i} \right) \cdot \left(g^{\alpha_f r_f} \prod_{i=0}^{d} g_i^{\alpha_f f_i} \right) \cdot g^{-r_{\tilde{f}}} \\ &= g^{r_h + \alpha_f r_f} \cdot g^{-r_{\tilde{f}}} \cdot \prod_{i=0}^{d} g_i^{h_i + \alpha_f f_i} \\ &= \prod_{i=0}^{d} g_i^{h_i + \alpha_f f_i} = \prod_{i=0}^{d} g_i^{\tilde{f}_i} = \mathsf{Commit}(\mathsf{pp}, \tilde{f}(X))) \text{ and} \\ y &= y_h + \alpha_f y_f = y(x) + \alpha_f f(x) = \tilde{f}(x) \mod p. \end{aligned}$$

(witness-extended emulation) We show that Π_{H} has witness-extended emulation. From Theorem 2, we have an expected polynomial-time extractor $\mathcal{E}$ that extracts $\tilde{f}(X)$ for the Eval protocol. Using $\mathcal{E}$, we construct an extractor $\mathcal{E}_{\mathsf{H}}$ to extract a witness $f(X)$ from EvalZK. The extractor $\mathcal{E}_{\mathsf{H}}$ runs the prover to obtain $\{c_h, y_h\}$. At this point, $\mathcal{E}_{\mathsf{H}}$ then rewinds the oracle $\langle \mathcal{P}^*, \mathcal{V} \rangle$ twice with distinct challenges α_f and α'_f and obtains the corresponding commitments (c, y) and (c', y') to the witnesses $\tilde{f}(X)$ and $\tilde{f}'(X)$, respectively. Then, $\mathcal{E}_{\mathsf{H}}$ runs $\mathcal{E}$ on inputs $(\mathsf{pp}, c, x, y, d)$ and $(\mathsf{pp}, c', x, y', d)$ and receives the corresponding witnesses $\tilde{f}(X)$ and $\tilde{f}'(X)$, respectively. Finally, $\mathcal{E}_{\mathsf{H}}$ is able to extract the witness $f(X)$ from $\tilde{f}(X)$ and $\tilde{f}'(X)$, similarly to Lemma 2. This completes the proof of the witness-extended emulation. (perfect SHVZK) We construct the simulator Sim. Given only the public input, the simulator Sim outputs a simulated transcript that is identical to the valid transcript produced by the prover and the verifier in the real interaction. The simulator Sim first samples a random polynomial $\tilde{f}(X)$ of degree d and $r_{\tilde{f}} \xleftarrow{\$} \mathbb{Z}_p$. In addition, Sim samples a random challenge $\alpha_f \xleftarrow{\$} \mathbb{G}$ and computes $c_h = c \cdot c_f^{-\alpha_f} \cdot g^{r_{\tilde{f}}}$ and $y_h = y - \alpha_f \cdot y_f$. The simulator Sim then simply applies the Eval protocol honestly using $\tilde{f}(X)$ as the witness. Because in a real execution, the values α_f and $r_{\tilde{f}}$ are distributed uniformly at random over $\mathbb{Z}_p$, the simulated α_f and $r_{\tilde{f}}$ are identically distributed to real values. In addition, the real c_f and $\tilde{f}(X))$ are distributed uniformly at random over $\mathbb{G}$ and $\mathbb{Z}_p[X]$ of degree d, respectively, and the same distributions hold for the simulated c_f and $\tilde{f}(X)$, respectively. The simulated c_h is also distributed uniformly at random over $\mathbb{G}$, and thus the real c_h is, because of the perfect hiding property of the underlying commitment scheme. Clearly, the simulated $(c_h, y_h, \alpha_f, r_{\tilde{f}})$ holds the relations

$$c = c_h \cdot c_f^{\alpha_f} \cdot g^{-r_{\tilde{f}}} \in \mathbb{G}, \ y = y_h + \alpha_f \cdot y_f \in \mathbb{Z}_p.$$

Finally, the Eval protocol does not leak more than $\tilde{f}(X)$ itself, which contains no information about $f(X)$. Therefore, the views of the simulated and real transcripts are identically distributed. This completes the proof of the perfect SHVZK. □

7. Conclusions

In this paper, we presented how to transpose a recursive argument of polynomial evaluation over a class group proposed by Bünz et al. to the discrete log setting as a way to improve the efficiency. The transposition follows from their information-theoretic abstraction. We found that the challenge for a transposition is to provide a monomial homomorphism for an underlying commitment scheme. We observed that when we use a polynomial encoding method that presents coefficients of the polynomial to the power of random group elements, an essential sufficient condition is a proof system for the equality of discrete logarithms (PoE_{mDL}) over multiple bases. We believe that our approach suggests a stepping stone for the construction of an efficient, transparent polynomial commitment scheme with a recursive argument in the discrete log setting. Currently, the efficiency of known proof systems for PoE_{mDL} is not sufficient to hava logarithmic communication and verifier complexities. Therefore, in future work, we will continue to research how to improve the efficiency of PoE_{mDL}, which leads to high-efficiency gains for the proposed construction in the discrete log setting.

Funding: This research received no external funding.

Acknowledgments: This work was supported by a research grant from Seoul Women's University (2020-0454 and 2021-0093).

Conflicts of Interest: The author declares no conflict of interest.

References

1. Ben-Sasson, E.; Chiesa, A.; Garman, C.; Green, M.; Miers, I.; Tromer, E.; Virza, M. Zerocash: Decentralized Anonymous Payments from Bitcoin. In Proceedings of the 2014 IEEE Symposium on Security and Privacy, San Jose, CA, USA, 18–21 May 2014; pp. 459–474.
2. Eberhardt, J.; Tai, S. ZoKrates–Scalable Privacy-Preserving Off-Chain Computations. In Proceedings of the 2018 IEEE International Conference on Blockchain, Halifax, NS, Canada, 30 July–3 August 2018; pp. 1084–1091.
3. Chiesa, A.; Hu, Y.; Maller, M.; Mishra, P.; Vesely, N.; Ward, N.P. Marlin: Preprocessing zkSNARKs with Universal and Updatable SRS. In *Advances in Cryptology—EUROCRYPT 2020*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 738–768.
4. Maller, M.; Bowe, S.; Kohlweiss, M.; Meiklejohn, S. Sonic: Zero-Knowledge SNARKs from Linear-Size Universal and Updatable Structured Reference Strings. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS 2019), London, UK, 11–15 November 2019; pp. 2111–2128.
5. Gabizon, A.; Williamson, Z.J.; Ciobotaru, O. PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge. *IACR Cryptol. ePrint Arch.* **2019**, 953.
6. Bünz, B.; Fisch, B.; Szepieniec, A. Transparent SNARKs from DARK Compilers. In *Advances in Cryptology—EUROCRYPT 2020*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 677–706.
7. Wahby, R.S.; Tzialla, I.; Shelat, A.; Thaler, J.; Walfish, M. Doubly-Efficient zkSNARKs Without Trusted Setup. In Proceedings of the 2018 IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 20–24 May 2018; pp. 926–943.
8. Kattis, A.; Panarin, K.; Vlasov, A. RedShift: Transparent SNARKs from List Polynomial Commitment IOPs. *IACR Cryptol. ePrint Arch.* **2019**, 1400.
9. Fiat, A.; Shamir, A. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Advances in Cryptology—CRYPTO 1986*; Springer: Berlin/Heidelberg, Germany, 1986; pp. 186–194.
10. Kate, A.; Zaverucha, G.M.; Goldberg, I. Constant-Size Commitments to Polynomials and Their Applications. In *Advances in Cryptology—ASIACRYPT 2010*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 177–194.
11. Hamdy, S.; Möller, B. Security of Cryptosystems Based on Class Groups of Imaginary Quadratic Orders. In *Advances in Cryptology—ASIACRYPT 2000*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 234–247.
12. Buchmann, J.; Hamdy, S. A survey on IQ cryptography. In *Public Key Cryptography and Computational Number Theory*; De Gruyter: Berlin, Germany; New York, NY, USA, 2011; pp. 1–16.
13. Dobson, S.; Galbraith, S.D. Trustless Groups of Unknown Order with Hyperelliptic Curves. *IACR Cryptol. ePrint Arch.* **2020**, *2020*, 196.
14. Bootle, J.; Cerulli, A.; Chaidos, P.; Groth, J.; Petit, C. Efficient Zero-Knowledge Arguments for Arithmetic Circuits in the Discrete Log Setting. In *Advances in Cryptology—EUROCRYPT 2016*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 327–357.

15. Pedersen, T.P. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *Advances in Cryptology—CRYPTO 1991*; Springer: Berlin/Heidelberg, Germany, 1991; pp. 129–140.
16. Papamanthou, C.; Shi, E.; Tamassia, R. Signatures of Correct Computation. In *Theory of Cryptography, Proceedings of the 10th Theory of Cryptography Conference (TCC), Tokyo, Japan, 3–6 March 2013*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 222–242.
17. Zhang, Y.; Genkin, D.; Katz, J.; Papadopoulos, D.; Papamanthou, C. vSQL: Verifying Arbitrary SQL Queries over Dynamic Outsourced Databases. In Proceedings of the 2017 IEEE Symposium on Security and Privacy, San Jose, CA, USA, 22–24 May 2017; pp. 863–880.
18. Zhang, Y.; Genkin, D.; Katz, J.; Papadopoulos, D.; Papamanthou, C. A Zero-Knowledge Version of vSQL. *IACR Cryptol. ePrint Arch.* **2017**, 1146.
19. Bünz, B.; Bootle, J.; Boneh, D.; Poelstra, A.; Wuille, P.; Maxwell, G. Bulletproofs: Short Proofs for Confidential Transactions and More. In Proceedings of the 2018 IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 21–23 May 2018; pp. 315–334.
20. Ben-Sasson, E.; Goldberg, L.; Kopparty, S.; Saraf, S. DEEP-FRI: Sampling Outside the Box Improves Soundness. In Proceedings of the 11th Innovations in Theoretical Computer Science Conference (ITCS), Washington, SA, USA, 12–14 January 2020; pp. 5:1–5:32.
21. Zhang, J.; Xie, T.; Zhang, Y.; Song, D. Transparent Polynomial Delegation and Its Applications to Zero Knowledge Proof. In Proceedings of the 2020 IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 18–20 May 2020; pp. 859–876.
22. Lee, J. Dory: Efficient, Transparent arguments for Generalised Inner Products and Polynomial Commitments. *IACR Cryptol. ePrint Arch.* **2020**, 1274.
23. Boneh, D.; Drake, J.; Fisch, B.; Gabizon, A. Halo Infinite: Recursive zk-SNARKs from any Additive Polynomial Commitment Scheme. *IACR Cryptol. ePrint Arch.* **2020**, 1536.
24. Wesolowski, B. Efficient Verifiable Delay Functions. In *Advances in Cryptology—EUROCRYPT 2019*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 379–407.
25. Boneh, D.; Bünz, B.; Fisch, B. Batching Techniques for Accumulators with Applications to IOPs and Stateless Blockchains. *Advances in Cryptology—CRYPTO 2019*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 561–586.
26. Rivest, R.L.; Shamir, A.; Wagner, D.A. *Time-Lock Puzzles and Timed-Release Crypto*; Technical Report; Massachusetts Institute of Technology: Cambridge, MA, USA, 1996.
27. Buchmann, J.; Williams, H.C. A Key-Exchange System Based on Imaginary Quadratic Fields. *J. Cryptol.* **1988**, *1*, 107–118. [CrossRef]
28. Belabas, K.; Kleinjung, T.; Sanso, A.; Wesolowski, B. A note on the low order assumption in class group of an imaginary quadratic number fields. *IACR Cryptol. ePrint Arch.* **2020**, 1310.
29. Lee, J. The security of Groups of Unknown Order based on Jacobians of Hyperelliptic Curves. *IACR Cryptol. ePrint Arch.* **2020**, 289.
30. Lindell, Y. Parallel Coin-Tossing and Constant-Round Secure Two-Party Computation. *J. Cryptol.* **2003**, *16*, 143–184. [CrossRef]
31. Groth, J.; Ishai, Y. Sub-linear Zero-Knowledge Argument for Correctness of a Shuffle. In *Advances in Cryptology—EUROCRYPT 2008*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 379–396.
32. Groth, J. Linear Algebra with Sub-linear Zero-Knowledge Arguments. In *Advances in Cryptology—CRYPTO 2009*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 192–208.
33. Chaum, D.; Evertse, J.; van de Graaf, J. An Improved Protocol for Demonstrating Possession of Discrete Logarithms and Some Generalizations. In *Advances in Cryptology—EUROCRYPT 1987*; Springer: Berlin/Heidelberg, Germany, 1987; pp. 127–141.
34. Chaum, D.; Pedersen, T.P. Transferred Cash Grows in Size. In *Advances in Cryptology—EUROCRYPT 1992*; Springer: Berlin/Heidelberg, Germany, 1992; pp. 390–407.
35. Lovecruft, I.A.; de Valence, H. curve25519-dalek: A pure-rust implementation of group operations on ristretto and curve25519. Available online: https://github.com/zkcrypto/curve25519-dalek-ng (accessed on 28 December 2021).
36. Jacobson, M.J., Jr.; van der Poorten, A.J. Computational Aspects of NUCOMP. In Proceedings of the 5th International Symposium Algorithmic Number Theory, Sydney, Australia, 7–12 July 2002; pp. 120–133.
37. Hart, W. ANTIC—Algebraic Number Theory in C. Available online: https://github.com/wbhart/antic (accessed on 28 December 2021).
38. Aranha, D.F.; Gouvêa, C.P.L.; Markmann, T.; Wahby, R.S.; Liao, K. RELIC Is an Efficient LIbrary for Cryptography. Available online: https://github.com/relic-toolkit/relic (accessed on 28 December 2021).
39. Bowe, S. BLS12-381: New zk-SNARK Elliptic Curve Construction. Available online: https://electriccoin.co/blog/new-snark-curve/ (accessed on 23 December 2021).
40. Pippenger, N. On the Evaluation of Powers and Monomials. *SIAM J. Comput.* **1980**, *9*, 230–250. [CrossRef]
41. Chiesa, A.; Forbes, M.A.; Spooner, N. A Zero Knowledge Sumcheck and its Applications. *arXiv* **2017**, arXiv:1704.02086.