How (Not) to Simulate PLONK

Marek Sefranek[0009−0008−8987−9555]

TU Wien, Vienna, Austria
marek.sefranek@tuwien.ac.at

Abstract. PLONK is a zk-SNARK system by Gabizon, Williamson, and Ciobotaru with proofs of constant size (0.5 KB) and sublinear verification time. Its setup is circuit-independent supporting proofs of arbitrary statements up to a certain size bound.
Although deployed in several real-world applications, PLONK's zero-knowledge property had only been argued informally. Consequently, we were able to find and fix a vulnerability in its original specification, leading to an update of PLONK in eprint version 20220629:105924.
In this work, we construct a simulator for the patched version of PLONK and prove that it achieves statistical zero knowledge. Furthermore, we give an attack on the previous version of PLONK showing that it does not even satisfy the weaker notion of (statistical) witness indistinguishability.

Keywords: Zero knowledge · zk-SNARKs · Simulator · Vulnerability

1 Introduction

First introduced by Goldwasser, Micali, and Rackoff [GMR85] as a theoretical concept in the 1980s, *zero-knowledge proofs* have become efficient enough to be used in practice over the subsequent decades. Informally speaking, a zero-knowledge proof allows a party (the "prover") to convince another party (the "verifier") that a statement is true, without revealing any information beyond the validity of the statement. Moreover, such proofs need to be complete, i.e., the verifier accepts an honestly generated proof on a true statement, and sound, i.e., a malicious prover cannot convince the verifier of a false statement.

The most widely used type of zero-knowledge proof systems are *zk-SNARKs* (***s****uccinct* ***n****on-interactive* ***ar****guments of* ***k****nowledge*) [BCCT12]. An *argument* is a proof system with computational soundness, i.e., both prover and verifier are probabilistic polynomial-time algorithms. Furthermore, the prover must know a witness whenever producing a valid proof (hence argument *of knowledge*). *Non-interactive* means that only a single message, "the proof", is sent during the protocol. The term *succinct* refers to the proof size, in particular, zk-SNARKs have proofs that are much shorter than the proved statement, in some cases even of constant size.

Many of the currently most efficient zk-SNARKs are constructed using an elliptic-curve group over a prime field $\mathbb{F}_p$, i.e., their proofs are composed of

elliptic-curve group elements and field elements. To prove arbitrary statements in NP, they express them in the NP-complete language of *arithmetic circuits*.

Currently, the zk-SNARK with the shortest proof size is the construction by Groth [Gro16]. Its proofs are independent of the size of the associated arithmetic circuit and consist of just 3 group elements ($\approx$ 128 bytes). Nevertheless, it comes with the major drawback of requiring a circuit-specific common reference string, limiting its usability in practice. To solve this issue, a recent development is to construct zk-SNARKs with a *universal structured reference string* (SRS), i.e., once generated, it can be used to prove statements about any arithmetic circuit up to a certain size bound.

The first construction with a universal SRS that scales linearly in size is *Sonic* by Maller et al. [MBKM19], featuring a proof size of 20 group elements and 16 field elements ($\approx$ 1152 bytes). Improving upon the efficiency of Sonic, *PLONK* by Gabizon et al. [GWC19a] offers even shorter proofs consisting of only 9 group elements and 6 field elements ($\approx$ 480 bytes). Another universal zk-SNARK is *Marlin* by Chiesa et al. [CHM+20] with a proof size of 13 group elements and 8 field elements ($\approx$ 672 bytes). However, compared to PLONK, it has not found such a wide adoption in practice.

Beyond that, other directions in constructing succinct non-interactive zero-knowledge proofs include *Bulletproofs* [BBB+18] and *zk-STARKs* [BSBHR18], short for ***z**ero-**k**nowledge **s**calable **t**ransparent **ar**guments of **k**nowledge*. At the cost of having longer proofs—logarithmic (Bulletproofs) and polylogarithmic (zk-STARKs) in the size of the proved statement—these constructions do not require any trusted setup, i.e., they do not use a common reference string.[1] On top of that, zk-STARKs are currently considered post-quantum secure as opposed to proof systems which rely on the hardness of the discrete-logarithm problem in elliptic-curve groups.

1.1 PLONK

This work focuses on the PLONK zk-SNARK by Gabizon et al. [GWC19a]. It has constant-size proofs, sublinear proof verification time, and a circuit-independent setup which produces a universal and updatable structured reference string. Updatability means that the SRS can be rerandomized such that the honesty of only one party from all updaters up to that point is required for soundness.

The original PLONK construction employs the *KZG polynomial commitment scheme* by Kate, Zaverucha, and Goldberg [KZG10] due to its constant-size commitments and opening proofs that only consist of single group elements. This comes at the cost of a trusted setup which produces a structured reference string of linear size $O(d)$ group elements, where d is the maximum supported degree of the committed polynomials. In general, PLONK can be instantiated with any (extractable) polynomial commitment scheme.

[1] More precisely, both rely on a collision-resistant hash function, with Bulletproofs modeling it as a random oracle [BR93] and zk-STARKs using it directly.

PLONK is deployed in various real-world projects, a prominent example of which is the anonymous cryptocurrency Zcash [HBHW22]. Currently, Zcash uses *Halo 2* as its zk-SNARK [Ele22], which is built on top of the arithmetization introduced by PLONK [GWC19a, Sec. 6].

There are several other extensions and variants of PLONK. This includes support for custom gates [GW22], which can perform more complex operations than just field addition/multiplication, such as elliptic-curve group addition. Furthermore, using the *plookup* protocol of [GW20], *PlonKup* by Pearson et al. [PFM+22] extends PLONK to incorporate lookup gates, which are used to enforce that a value is contained in a predefined table.

The work by Chen et al. [CBBZ23] introduced *HyperPlonk*, which uses multilinear instead of univariate polynomial commitments. At the cost of longer proofs, this avoids the need for an FFT during proof generation, achieving a linear prover runtime as opposed to the quasilinear time required by PLONK.

1.2 Our contributions

Due to the lack of a formal security proof for PLONK's zero-knowledge property, we were able to identify a vulnerability in its original specification [GWC19b]. Consequently, we proposed a fix, and the PLONK protocol has been patched accordingly in eprint version 20220629:105924.[2] In this work, we give a formal proof establishing that the resulting version of PLONK achieves statistical zero knowledge. Towards this goal, we construct a simulator and argue why its outputs are distributed statistically close to real PLONK proofs.

Furthermore, we show that the previous specification of PLONK without our fix does not satisfy statistical zero knowledge by devising an attack that breaks its (statistical) witness indistinguishability. In this notion, which is weaker than zero knowledge, the adversary defines a statement with two different witnesses and then, given a proof computed under either of them, has to distinguish which of the two witnesses was used.

2 Preliminaries

We use $\lambda \in \mathbb{N}$ to denote the security parameter, $[n]$ for the set $\{1, \ldots, n\} \subset \mathbb{N}$, and $\mathsf{negl}(\cdot)$ for any negligible function $\mathsf{negl}(\lambda) = \lambda^{-\omega(1)}$. Let $\mathbb{F}_p[X]$ be the set of univariate polynomials over $\mathbb{F}_p$, and $\mathbb{F}_p^{(\leq d)}[X]$ its restriction to polynomials of degree $\leq d$.

A statement of the form $y \coloneqq \mathcal{A}(x)$ denotes a deterministic assignment, while $y \leftarrow \mathcal{A}(x)$ denotes a probabilistic assignment, i.e., running the algorithm $\mathcal{A}$ with uniform randomness on input x. We also use $y \leftarrow S$ for sampling y uniformly at random from the set S. We write $(y; z) \leftarrow (\mathcal{A} \parallel \mathcal{E})(x)$ when $\mathcal{A}$ on input x outputs y, and $\mathcal{E}$ on the same input (including $\mathcal{A}$'s randomness) outputs z.

An *indexed relation* $\mathcal{R}$ is a set of index-statement-witness triples $(\mathtt{i}, \mathtt{x}, \mathtt{w})$ with the corresponding *indexed language* $\mathcal{L}_{\mathcal{R}} \coloneqq \{(\mathtt{i}, \mathtt{x}) \mid \exists \mathtt{w} \colon (\mathtt{i}, \mathtt{x}, \mathtt{w}) \in \mathcal{R}\}$. For

[2] https://twitter.com/rel_zeta_tech/status/1542474186664210432

a size bound $n \in \mathbb{N}$, we denote by $\mathcal{R}_n$ the restriction of $\mathcal{R}$ to triples $(\mathtt{i}, \mathtt{x}, \mathtt{w})$ with $|\mathtt{i}| \leq n$. For example, the indexed relation of satisfiable arithmetic circuits consists of the triples where $\mathtt{i}$ is the description of an arithmetic circuit, $\mathtt{x}$ is a partial assignment to its input wires, and $\mathtt{w}$ is an appropriate assignment to the remaining wires such that the circuit outputs 0.

We implicitly assume all PPT algorithms described in this work run in time polynomial in the security parameter λ. Furthermore, we assume there is an efficient algorithm GGen which, on input 1^λ, generates a bilinear group $\mathcal{G} \coloneqq (\mathbb{F}_p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \mathsf{g}_1, \mathsf{g}_2, \mathsf{g}_T)$ where:

- $\mathbb{F}_p$ is a prime field of super-polynomial size $|\mathbb{F}_p| = \lambda^{\omega(1)}$, which admits certain primitive n-th roots of unity (restricting the factorization of $p-1$).
- $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are groups of prime order p with a pairing $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$.
- $\mathsf{g}_1 \in \mathbb{G}_1, \mathsf{g}_2 \in \mathbb{G}_2$ are uniformly chosen generators, and $\mathsf{g}_T = e(\mathsf{g}_1, \mathsf{g}_2)$.

When instantiating a cryptographic scheme over $\mathcal{G}$, we assume implicitly that $\mathsf{GGen}(1^\lambda)$ is run during the setup phase and its output is publicly available.

2.1 zk-SNARKs

In the following, we define universal, preprocessing zk-SNARKs, an example of which is the PLONK system we focus on in this work. Our definitions are inspired by [CBBZ23, Sec. 2.1] and [CHM+20, Sec. 7].

Definition 1 (zk-SNARK). *A zk-SNARK for an indexed relation $\mathcal{R}$ is a tuple of four PPT algorithms* $(\mathsf{Setup}, \mathsf{Preproc}, \mathsf{Prove}, \mathsf{Verify})$ *with the following syntax:*

- $\mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n)$*, given the security parameter 1^λ and a size bound $n \in \mathbb{N}$ on indices in $\mathcal{R}$, returns a structured reference string* srs*.*
- $(\mathsf{pp}, \mathsf{vp}) \coloneqq \mathsf{Preproc}(\mathsf{srs}, \mathtt{i})$*, given the* srs *(with implicit size bound n) and an index $\mathtt{i}$ of size $\leq n$, returns prover and verifier parameters* pp *and* vp*.*
- $\pi \leftarrow \mathsf{Prove}(\mathsf{pp}, \mathtt{x}, \mathtt{w})$*, given* pp *(with implicit index $\mathtt{i}$), a statement $\mathtt{x}$, and a witness $\mathtt{w}$, returns a proof π for $(\mathtt{i}, \mathtt{x}, \mathtt{w}) \in \mathcal{R}$.*
- $b \coloneqq \mathsf{Verify}(\mathsf{vp}, \mathtt{x}, \pi)$*, given* vp*, a statement $\mathtt{x}$, and a proof π, returns a bit $b \in \{0, 1\}$, with 1 meaning accept and 0 reject.*

This definition captures *universal* zk-SNARKs, because the structured reference string srs generated by Setup supports proofs of any $(\mathtt{i}, \mathtt{x}, \mathtt{w}) \in \mathcal{R}$ up to a selected size bound n, as opposed to an index-specific reference string.

Moreover, we define *preprocessing* zk-SNARKs because Prove and Verify are not directly run on the instance $(\mathtt{i}, \mathtt{x}) \in \mathcal{L}_\mathcal{R}$. Instead, there is a non-interactive preprocessing phase where anyone can use $\mathsf{Preproc}$ to publicly compute prover and verifier parameters pp, vp for a given index $\mathtt{i}$. This allows to compress the srs and $\mathtt{i}$ significantly, enabling the prover and, especially, the verifier to check different statements $\mathtt{x}$ in subsequent online phases more efficiently.

Furthermore, zk-SNARKs have to satisfy the following properties.

Definition 2 (Completeness). *For all* $n = \mathsf{poly}(\lambda)$ *and all* $(\mathtt{i}, \mathtt{x}, \mathtt{w}) \in \mathcal{R}_n$*:*

$$\Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (\mathsf{pp}, \mathsf{vp}) \coloneqq \mathsf{Preproc}(\mathsf{srs}, \mathtt{i}); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, \mathtt{x}, \mathtt{w}); \\ b \coloneqq \mathsf{Verify}(\mathsf{vp}, \mathtt{x}, \pi): \\ \\ b = 1 \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

We define *statistical* completeness, which is the notion achieved by PLONK (see Section 2.2). For *perfect* completeness, the above probability must be one.

Definition 3 (Knowledge Soundness). *For all* $n = \mathsf{poly}(\lambda)$ *and every PPT adversary* $\mathcal{A}$*, there exists a PPT extractor* $\mathcal{E}$ *such that:*

$$\Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ \big((\mathtt{i}, \mathtt{x}, \pi); \mathtt{w}\big) \leftarrow (\mathcal{A} \parallel \mathcal{E})(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) \coloneqq \mathsf{Preproc}(\mathsf{srs}, \mathtt{i}); \\ b \coloneqq \mathsf{Verify}(\mathsf{vp}, \mathtt{x}, \pi): \\ \\ b = 1 \wedge (\mathtt{i}, \mathtt{x}, \mathtt{w}) \notin \mathcal{R}_n \end{array}\right] \leq \mathsf{negl}(\lambda).$$

This is an adaptive definition, where the malicious prover $\mathcal{A}$ can choose the statement $(\mathtt{i}, \mathtt{x}) \in \mathcal{L}_\mathcal{R}$ based on the structured reference string srs. Note that the extractor $\mathcal{E}$ is given the adversary's randomness, which allows it to rewind $\mathcal{A}$.

Definition 4 (Succinctness). *The proof size is sublinear in the size of the proved statement* $(\mathtt{i}, \mathtt{x}, \mathtt{w}) \in \mathcal{R}_n$*:* $|\pi| = o(|\mathtt{i}| + |\mathtt{x}| + |\mathtt{w}|)$.

Definition 5 (Zero Knowledge). *There exists a PPT simulator* $\mathcal{S}$ *such that for all* $n = \mathsf{poly}(\lambda)$ *and all adversaries* $\mathcal{A}$*:*

$$\left|\Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (\mathtt{i}, \mathtt{x}, \mathtt{w}, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) \coloneqq \mathsf{Preproc}(\mathsf{srs}, \mathtt{i}); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, \mathtt{x}, \mathtt{w}); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi): \\ \\ b = 1 \wedge (\mathtt{i}, \mathtt{x}, \mathtt{w}) \in \mathcal{R}_n \end{array}\right] - \Pr\left[\begin{array}{c} (\mathsf{srs}, \tau) \leftarrow \mathcal{S}(1^\lambda, n); \\ (\mathtt{i}, \mathtt{x}, \mathtt{w}, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) \coloneqq \mathsf{Preproc}(\mathsf{srs}, \mathtt{i}); \\ \pi \leftarrow \mathcal{S}(\tau, \mathsf{pp}, \mathtt{x}); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi): \\ \\ b = 1 \wedge (\mathtt{i}, \mathtt{x}, \mathtt{w}) \in \mathcal{R}_n \end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

This captures *statistical* zero knowledge. The weaker notion of *computational* zero knowledge only considers PPT adversaries, while the stronger notion of *perfect* zero knowledge requires the two probabilities to be equal.

Definition 6 (Witness Indistinguishability). *For all* $n = \mathsf{poly}(\lambda)$ *and all adversaries* $\mathcal{A}$*:*

$$\left|\Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (\mathtt{i}, \mathtt{x}, \mathtt{w}_0, \mathtt{w}_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) \coloneqq \mathsf{Preproc}(\mathsf{srs}, \mathtt{i}); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, \mathtt{x}, \mathtt{w}_0); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi): \\ \\ b = 1 \wedge (\mathtt{i}, \mathtt{x}, \mathtt{w}_0) \in \mathcal{R}_n \\ \wedge\, (\mathtt{i}, \mathtt{x}, \mathtt{w}_1) \in \mathcal{R}_n \end{array}\right] - \Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (\mathtt{i}, \mathtt{x}, \mathtt{w}_0, \mathtt{w}_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) \coloneqq \mathsf{Preproc}(\mathsf{srs}, \mathtt{i}); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, \mathtt{x}, \mathtt{w}_1); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi): \\ \\ b = 1 \wedge (\mathtt{i}, \mathtt{x}, \mathtt{w}_0) \in \mathcal{R}_n \\ \wedge\, (\mathtt{i}, \mathtt{x}, \mathtt{w}_1) \in \mathcal{R}_n \end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

Zero knowledge implies witness indistinguishability [FS90] (for a proof adapted to our setting, see Appendix A.1). The notion we define here is again *statistical*.

2.2 The PLONK construction

We give a condensed summary of PLONK and its underlying constraint system. For more details, refer to the original paper by Gabizon et al. [GWC19a, Sec. 8].

Let $\mathcal{C}$ be an arithmetic circuit over $\mathbb{F}_p$ with fan-in two, unlimited fan-out, n gates, and ℓ public inputs (which are also counted as gates). For simplicity, assume the public inputs are associated with the first ℓ gates. Let $(w_i)_{i\in[3n]} \in \mathbb{F}_p^{3n}$ be a (redundant) wire assignment, where w_i, w_{n+i}, and w_{2n+i} represent the values assigned to the left input, right input, and output wire of the i-th gate, respectively. Using the PLONK constraint system [GWC19a, Sec. 6], $\mathcal{C}$ is then modeled as a combination of gate and copy constraints as follows:

1. The *gate constraints* ensure correctness of the individual gate operations. Concretely, the i-th gate is associated with a constraint of the form

$$s_{\mathsf{L}i} \cdot w_i + s_{\mathsf{R}i} \cdot w_{n+i} + s_{\mathsf{O}i} \cdot w_{2n+i} + s_{\mathsf{M}i} \cdot w_i \cdot w_{n+i} + s_{\mathsf{C}i} = x_i,$$

   where $x_i \in \mathbb{F}_p$ is a public input or 0 if $i > \ell$, and the values of the selector vectors $\mathbf{s_L}, \mathbf{s_R}, \mathbf{s_O}, \mathbf{s_M}, \mathbf{s_C} \in \mathbb{F}_p^n$ depend on the gate type as specified in Table 1.

Table 1. Selector vector assignment of PLONK.

Gate type	$s_{\mathsf{L}i}$	$s_{\mathsf{R}i}$	$s_{\mathsf{O}i}$	$s_{\mathsf{M}i}$	$s_{\mathsf{C}i}$
Addition	1	1	-1	0	0
Multiplication	0	0	-1	1	0
Constant	1	0	0	0	$-c$
Public input	1	0	0	0	0

2. The *copy constraints* enforce that gate outputs are propagated correctly, e.g., if output wire w_{2n+i} is the same as input wire w_j, then both must be assigned the same value. This is achieved by choosing a permutation $\sigma\colon [3n] \to [3n]$ with appropriate cycles such that the statement "$w_i = w_{\sigma(i)}$ for all $i \in [3n]$" captures all the values in $(w_i)_{i\in[3n]}$ that must be equal. In the previous example, σ would contain the cycle $(2n+i, j)$ such that $w_{2n+i} = w_j$.

In combination, this results in the indexed relation $\mathcal{R}_{\text{PLONK}} :=$

$$\left\{ (\mathtt{i}, \mathtt{x}, \mathtt{w}) \;\middle|\; \begin{array}{l} \mathtt{i} = \big((\mathbf{s_L}, \mathbf{s_R}, \mathbf{s_O}, \mathbf{s_M}, \mathbf{s_C}) \in \mathbb{F}_p^{5n},\ \sigma\colon [3n] \to [3n]\big), \\ \mathtt{x} = (x_1, \ldots, x_\ell) \in \mathbb{F}_p^{\ell},\ \mathtt{w} = (w_1, \ldots, w_{3n}) \in \mathbb{F}_p^{3n}, \\ \forall i \in [\ell]\colon s_{\mathsf{L}i} w_i + s_{\mathsf{R}i} w_{n+i} + s_{\mathsf{O}i} w_{2n+i} + s_{\mathsf{M}i} w_i w_{n+i} + s_{\mathsf{C}i} = x_i, \\ \forall i \in (\ell, n]\colon s_{\mathsf{L}i} w_i + s_{\mathsf{R}i} w_{n+i} + s_{\mathsf{O}i} w_{2n+i} + s_{\mathsf{M}i} w_i w_{n+i} + s_{\mathsf{C}i} = 0, \\ \forall i \in [3n]\colon w_i = w_{\sigma(i)} \end{array} \right\},$$

where the index $\mathtt{i}$ is the arithmetic circuit $\mathcal{C}$, the statement $\mathtt{x}$ its public inputs, and the witness $\mathtt{w}$ a consistent wire assignment to $\mathcal{C}$.

Let ω be a primitive n-th root of unity in $\mathbb{F}_p$ and define $H \coloneqq \langle \omega \rangle$ as the subgroup generated by ω. The core of the PLONK proof system is to express a statement $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}_{\text{PLONK}}$ as an equivalent polynomial division of the form

$$\frac{\mathsf{gates}(X) + \alpha \cdot \mathsf{copy}_1(X) + \alpha^2 \cdot \mathsf{copy}_2(X)}{Z_H(X)},$$

where $Z_H(X) \coloneqq X^n - 1$ is the vanishing polynomial over H, the polynomials $\mathsf{gates}(X)$, $\mathsf{copy}_1(X)$, $\mathsf{copy}_2(X)$ depend on the given statement, and $\alpha \in \mathbb{F}_p$ is a randomly chosen challenge from the verifier. Leveraging the Schwartz–Zippel lemma, the prover then convinces the verifier that this divisibility holds at a random evaluation challenge $\delta \in \mathbb{F}_p$ selected by the verifier.

To obtain non-interactive proofs, PLONK applies the Fiat–Shamir transformation [FS87]. For this purpose, let $\mathcal{H}\colon \{0,1\}^* \to \mathbb{F}_p$ be a hash function modeled as a random oracle [BR93], and let inputs denote all the public information, i.e., the SRS, the common preprocessed input, and all the public circuit inputs $(x_i)_{i \in [\ell]}$. Each verifier challenge is then computed by evaluating $\mathcal{H}$ on the concatenation of inputs and all the proof elements output by the prover up to that point. Note that including all of this information in inputs is crucial to prevent certain attacks on knowledge soundness [Mil22].

In the following, let $\mathcal{L}_i \in \mathbb{F}_p^{(<n)}[X]$ denote the i-th Lagrange basis polynomial over H, i.e., $\mathcal{L}_i(\omega^i) = 1$ and $\mathcal{L}_i(\omega^j) = 0$ for all $j \neq i \in [n]$. Moreover, $k_1, k_2 \in \mathbb{F}_p^*$ are chosen such that H, $k_1 H$, $k_2 H$ are distinct cosets of H, and the permutation $\sigma\colon [3n] \to [3n]$ is extended onto $H' \coloneqq H \cup k_1 H \cup k_2 H$ by defining the wire permutation $\sigma^*\colon [3n] \to H'$ as

$$\sigma^*(i) \coloneqq \begin{cases} \omega^{\sigma(i)} & \text{if } \sigma(i) \in [n] \\ k_1 \cdot \omega^{\sigma(i)} & \text{if } \sigma(i) \in (n, 2n] \\ k_2 \cdot \omega^{\sigma(i)} & \text{if } \sigma(i) \in (2n, 3n] \end{cases}.$$

We now give a detailed description of the PLONK protocol in Construction 1, including our fix of step 5 of the Prove algorithm highlighted in blue.

Construction 1: The PLONK zk-SNARK.

$\underline{\mathsf{Setup}(1^\lambda, n)}$: Choose random $\tau \leftarrow \mathbb{F}_p$ and output

$$\mathsf{srs} \coloneqq \left(\mathsf{g}_1, \mathsf{g}_1^\tau, \mathsf{g}_1^{\tau^2}, \ldots, \mathsf{g}_1^{\tau^{n+2}}, \mathsf{g}_2, \mathsf{g}_2^\tau\right) \in \mathbb{G}_1^{n+3} \times \mathbb{G}_2^2.$$

$\underline{\mathsf{Preproc}(\mathsf{srs}, \mathbb{i})}$: Given the circuit size n, the wire permutation σ^*, and the gate constraints $(s_{\mathsf{L}i}, s_{\mathsf{R}i}, s_{\mathsf{O}i}, s_{\mathsf{M}i}, s_{\mathsf{C}i})_{i \in [n]}$, compute the polynomials

$$S_{\mathsf{L}}(X) \coloneqq \sum\nolimits_{i \in [n]} s_{\mathsf{L}i} \mathcal{L}_i(X), \qquad S_{\mathsf{R}}(X) \coloneqq \sum\nolimits_{i \in [n]} s_{\mathsf{R}i} \mathcal{L}_i(X),$$
$$S_{\mathsf{O}}(X) \coloneqq \sum\nolimits_{i \in [n]} s_{\mathsf{O}i} \mathcal{L}_i(X), \qquad S_{\mathsf{M}}(X) \coloneqq \sum\nolimits_{i \in [n]} s_{\mathsf{M}i} \mathcal{L}_i(X),$$

$$S_{\mathsf{C}}(X) \coloneqq \sum\nolimits_{i\in[n]} s_{\mathsf{C}i}\mathcal{L}_i(X), \qquad S_{\sigma,1}(X) \coloneqq \sum\nolimits_{i\in[n]} \sigma^*(i)\mathcal{L}_i(X),$$
$$S_{\sigma,2}(X) \coloneqq \sum\nolimits_{i\in[n]} \sigma^*(n+i)\mathcal{L}_i(X), \quad S_{\sigma,3}(X) \coloneqq \sum\nolimits_{i\in[n]} \sigma^*(2n+i)\mathcal{L}_i(X),$$

and their KZG commitments with respect to srs

$$c_{S_{\mathsf{L}}} \coloneqq \mathsf{g}_1^{S_{\mathsf{L}}(\tau)}, \quad c_{S_{\mathsf{R}}} \coloneqq \mathsf{g}_1^{S_{\mathsf{R}}(\tau)}, \quad c_{S_{\mathsf{O}}} \coloneqq \mathsf{g}_1^{S_{\mathsf{O}}(\tau)}, \quad c_{S_{\mathsf{M}}} \coloneqq \mathsf{g}_1^{S_{\mathsf{M}}(\tau)},$$
$$c_{S_{\mathsf{C}}} \coloneqq \mathsf{g}_1^{S_{\mathsf{C}}(\tau)}, \quad c_{S_{\sigma,1}} \coloneqq \mathsf{g}_1^{S_{\sigma,1}(\tau)}, \quad c_{S_{\sigma,2}} \coloneqq \mathsf{g}_1^{S_{\sigma,2}(\tau)}, \quad c_{S_{\sigma,3}} \coloneqq \mathsf{g}_1^{S_{\sigma,3}(\tau)}.$$

Output the prover and verifier parameters

$$\mathsf{pp} \coloneqq \big(\mathsf{srs}, n, \sigma^*, S_{\mathsf{L}}, S_{\mathsf{R}}, S_{\mathsf{O}}, S_{\mathsf{M}}, S_{\mathsf{C}}, S_{\sigma,1}, S_{\sigma,2}, S_{\sigma,3}\big),$$
$$\mathsf{vp} \coloneqq \big(\mathsf{g}_1, \mathsf{g}_2, \mathsf{g}_2^{\tau}, n, c_{S_{\mathsf{L}}}, c_{S_{\mathsf{R}}}, c_{S_{\mathsf{O}}}, c_{S_{\mathsf{M}}}, c_{S_{\mathsf{C}}}, c_{S_{\sigma,1}}, c_{S_{\sigma,2}}, c_{S_{\sigma,3}}\big).$$

$\underline{\mathsf{Prove}\big(\mathsf{pp}, (x_i)_{i\in[\ell]}, (w_i)_{i\in[3n]}\big)}$:

1. Choose random blinding scalars $\rho_1, \ldots, \rho_6 \leftarrow \mathbb{F}_p$ and compute the wire polynomials

$$A(X) \coloneqq (\rho_1 X + \rho_2)Z_H(X) + \sum\nolimits_{i\in[n]} w_i\mathcal{L}_i(X),$$
$$B(X) \coloneqq (\rho_3 X + \rho_4)Z_H(X) + \sum\nolimits_{i\in[n]} w_{n+i}\mathcal{L}_i(X),$$
$$C(X) \coloneqq (\rho_5 X + \rho_6)Z_H(X) + \sum\nolimits_{i\in[n]} w_{2n+i}\mathcal{L}_i(X).$$

   The first part of the proof are the commitments

$$c_A \coloneqq \mathsf{g}_1^{A(\tau)}, \; c_B \coloneqq \mathsf{g}_1^{B(\tau)}, \; c_C \coloneqq \mathsf{g}_1^{C(\tau)}.$$

2. Compute the permutation challenges $\beta, \gamma \in \mathbb{F}_p$ as

$$\beta \coloneqq \mathcal{H}(\mathsf{inputs}, c_A, c_B, c_C, 0), \; \gamma \coloneqq \mathcal{H}(\mathsf{inputs}, c_A, c_B, c_C, 1).$$

3. Define the polynomials

$$f(X) \coloneqq \big(A(X)+\beta S_{\mathrm{id},1}(X)+\gamma\big)\big(B(X)+\beta S_{\mathrm{id},2}(X)+\gamma\big)\big(C(X)+\beta S_{\mathrm{id},3}(X)+\gamma\big),$$
$$g(X) \coloneqq \big(A(X)+\beta S_{\sigma,1}(X)+\gamma\big)\big(B(X)+\beta S_{\sigma,2}(X)+\gamma\big)\big(C(X)+\beta S_{\sigma,3}(X)+\gamma\big).$$

   Choose random blinding scalars $\rho_7, \rho_8, \rho_9 \leftarrow \mathbb{F}_p$ and compute the permutation check polynomial

$$\Phi(X) \coloneqq (\rho_7 X^2 + \rho_8 X + \rho_9)Z_H(X) + \sum_{i\in[n]} \mathcal{L}_i(X) \prod_{j=1}^{i-1} \frac{f(\omega^j)}{g(\omega^j)},$$

where $\frac{f(\omega^j)}{g(\omega^j)} = \frac{(w_j+\beta\omega^j+\gamma)(w_{n+j}+\beta k_1\omega^j+\gamma)(w_{2n+j}+\beta k_2\omega^j+\gamma)}{(w_j+\beta\sigma^*(j)+\gamma)(w_{n+j}+\beta\sigma^*(n+j)+\gamma)(w_{2n+j}+\beta\sigma^*(2n+j)+\gamma)}$.

The second part of the proof is the commitment

$$c_\Phi \coloneqq \mathsf{g}_1^{\Phi(\tau)}.$$

4. Compute the quotient challenge $\alpha \in \mathbb{F}_p$ as

$$\alpha \coloneqq \mathcal{H}(\mathsf{inputs}, c_A, c_B, c_C, c_\Phi).$$

5. Define the polynomials

$$\begin{aligned}
\mathsf{gates}(X) &\coloneqq \begin{pmatrix} S_\mathsf{L}(X)A(X) + S_\mathsf{R}(X)B(X) + S_\mathsf{O}(X)C(X) \\ + S_\mathsf{M}(X)A(X)B(X) + S_\mathsf{C}(X) + S_\mathsf{PI}(X) \end{pmatrix}, \\
\mathsf{copy}_1(X) &\coloneqq \Phi(X)f(X) - \Phi(\omega X)g(X), \\
\mathsf{copy}_2(X) &\coloneqq \mathcal{L}_1(X)(\Phi(X) - 1).
\end{aligned}$$

Compute the quotient polynomial of degree $\leq 3n+5$

$$T(X) \coloneqq \frac{\mathsf{gates}(X) + \alpha \cdot \mathsf{copy}_1(X) + \alpha^2 \cdot \mathsf{copy}_2(X)}{Z_H(X)},$$

and split it into 3 unique polynomials $T_1', T_2', T_3' \in \mathbb{F}_p^{(\leq n+1)}[X]$ such that

$$T(X) = T_1'(X) + X^{n+2} \cdot T_2'(X) + X^{2n+4} \cdot T_3'(X).$$

Then, choose random blinding scalars $\rho_{10}, \rho_{11} \leftarrow \mathbb{F}_p$ and define

$$\begin{aligned}
T_1(X) &\coloneqq T_1'(X) + \rho_{10}X^{n+2}, \\
T_2(X) &\coloneqq T_2'(X) - \rho_{10} + \rho_{11}X^{n+2}, \\
T_3(X) &\coloneqq T_3'(X) - \rho_{11}.
\end{aligned}$$

Note that these polynomials still satisfy

$$T(X) = T_1(X) + X^{n+2} \cdot T_2(X) + X^{2n+4} \cdot T_3(X).$$

The third part of the proof are the commitments

$$c_{T_1} \coloneqq \mathsf{g}_1^{T_1(\tau)}, \; c_{T_2} \coloneqq \mathsf{g}_1^{T_2(\tau)}, \; c_{T_3} \coloneqq \mathsf{g}_1^{T_3(\tau)}.$$

6. Compute the evaluation challenge $\delta \in \mathbb{F}_p$ as

$$\delta \coloneqq \mathcal{H}(\mathsf{inputs}, c_A, c_B, c_C, c_\Phi, c_{T_1}, c_{T_2}, c_{T_3}).$$

7. The fourth part of the proof are the evaluations

$$\begin{aligned} a_\delta &\coloneqq A(\delta), & b_\delta &\coloneqq B(\delta), & c_\delta &\coloneqq C(\delta), \\ s_{\sigma_1,\delta} &\coloneqq S_{\sigma,1}(\delta), & s_{\sigma_2,\delta} &\coloneqq S_{\sigma,2}(\delta), & \phi_{\delta\omega} &\coloneqq \Phi(\delta\omega). \end{aligned}$$

8. Compute the opening challenge $\varepsilon \in \mathbb{F}_p$ as

$$\varepsilon \coloneqq \mathcal{H}(\mathsf{inputs}, c_A, c_B, c_C, c_\Phi, c_{T_1}, c_{T_2}, c_{T_3}, a_\delta, b_\delta, c_\delta, s_{\sigma_1,\delta}, s_{\sigma_2,\delta}, \phi_{\delta\omega}).$$

9. Define the linearized polynomials

$$\begin{aligned} L_{\mathsf{gates}}(X) &\coloneqq \begin{pmatrix} S_{\mathsf{L}}(X)A(\delta) + S_{\mathsf{R}}(X)B(\delta) + S_{\mathsf{O}}(X)C(\delta) \\ + S_{\mathsf{M}}(X)A(\delta)B(\delta) + S_{\mathsf{C}}(X) + S_{\mathsf{PI}}(\delta) \end{pmatrix}, \\ L_{\mathsf{copy}_1}(X) &\coloneqq \begin{pmatrix} \Phi(X)(A(\delta)+\beta\delta+\gamma)(B(\delta)+\beta k_1\delta+\gamma)(C(\delta)+\beta k_2\delta+\gamma) \\ -\Phi(\delta\omega)(A(\delta)+\beta S_{\sigma,1}(\delta)+\gamma)(B(\delta)+\beta S_{\sigma,2}(\delta)+\gamma)(C(\delta)+\beta S_{\sigma,3}(X)+\gamma) \end{pmatrix}, \\ L_{\mathsf{copy}_2}(X) &\coloneqq \mathcal{L}_1(\delta)(\Phi(X)-1), \\ L_{\mathsf{division}}(X) &\coloneqq Z_H(\delta)\big[T_1(X) + \delta^{n+2}T_2(X) + \delta^{2n+4}T_3(X)\big]. \end{aligned}$$

Compute the linearization polynomial

$$L(X) \coloneqq L_{\mathsf{gates}}(X) + \alpha \cdot L_{\mathsf{copy}_1}(X) + \alpha^2 \cdot L_{\mathsf{copy}_2}(X) - L_{\mathsf{division}}(X),$$

and the aggregated quotient polynomials

$$\begin{aligned} Q_1(X) &\coloneqq \frac{1}{X-\delta}\begin{pmatrix} L(X)+\varepsilon\big[A(X)-A(\delta)\big]+\varepsilon^2\big[B(X)-B(\delta)\big]+\varepsilon^3\big[C(X)-C(\delta)\big] \\ +\varepsilon^4\big[S_{\sigma,1}(X)-S_{\sigma,1}(\delta)\big]+\varepsilon^5\big[S_{\sigma,2}(X)-S_{\sigma,2}(\delta)\big] \end{pmatrix}, \\ Q_2(X) &\coloneqq \frac{\Phi(X)-\Phi(\delta\omega)}{X-\delta\omega}. \end{aligned}$$

The fifth part of the proof are the commitments

$$\pi_1 \coloneqq \mathsf{g}_1^{Q_1(\tau)},\ \pi_2 \coloneqq \mathsf{g}_1^{Q_2(\tau)}.$$

Output the full proof

$$\pi_{\mathrm{PLONK}} \coloneqq \begin{pmatrix} c_A, c_B, c_C, c_\Phi, c_{T_1}, c_{T_2}, c_{T_3}, \pi_1, \pi_2, \\ a_\delta, b_\delta, c_\delta, s_{\sigma_1,\delta}, s_{\sigma_2,\delta}, \phi_{\delta\omega} \end{pmatrix} \in \mathbb{G}_1^9 \times \mathbb{F}_p^6.$$

$\underline{\mathsf{Verify}\big(\mathsf{vp}, (x_i)_{i\in[\ell]}, \pi_{\mathrm{PLONK}}\big)}$: Compute the challenges $\beta, \gamma, \alpha, \delta, \varepsilon \in \mathbb{F}_p$ the same way as the prover, and the multipoint evaluation challenge $\zeta \in \mathbb{F}_p$ as

$$\zeta \coloneqq \mathcal{H}(\mathsf{inputs}, c_A, c_B, c_C, c_\Phi, c_{T_1}, c_{T_2}, c_{T_3}, a_\delta, b_\delta, c_\delta, s_{\sigma_1,\delta}, s_{\sigma_2,\delta}, \phi_{\delta\omega}, \pi_1, \pi_2).$$

Compute the commitments

$$
\begin{aligned}
c_{L_{\mathsf{gates}}} &\coloneqq {c_{S_{\mathsf{L}}}}^{a_\delta} \cdot {c_{S_{\mathsf{R}}}}^{b_\delta} \cdot {c_{S_{\mathsf{O}}}}^{c_\delta} \cdot {c_{S_{\mathsf{M}}}}^{a_\delta b_\delta} \cdot c_{S_{\mathsf{C}}} \cdot \mathsf{g}_1^{S_{\mathsf{PI}}(\delta)}, \\
c_{L_{\mathsf{copy}_1}} &\coloneqq \frac{{c_\Phi}^{(a_\delta+\beta\delta+\gamma)(b_\delta+\beta k_1\delta+\gamma)(c_\delta+\beta k_2\delta+\gamma)}}{\left(c_{S_{\sigma,3}}{}^{\beta} \cdot \mathsf{g}_1^{c_\delta+\gamma}\right)^{\phi_{\delta\omega}(a_\delta+\beta s_{\sigma_1,\delta}+\gamma)(b_\delta+\beta s_{\sigma_2,\delta}+\gamma)}}, \\
c_{L_{\mathsf{copy}_2}} &\coloneqq \left(c_\Phi \cdot \mathsf{g}_1^{-1}\right)^{\mathcal{L}_1(\delta)}, \\
c_{L_{\mathsf{division}}} &\coloneqq \left(c_{T_1} \cdot {c_{T_2}}^{\delta^{n+2}} \cdot {c_{T_3}}^{\delta^{2n+4}}\right)^{Z_H(\delta)}, \\
c_L &\coloneqq c_{L_{\mathsf{gates}}} \cdot (c_{L_{\mathsf{copy}_1}})^{\alpha} \cdot (c_{L_{\mathsf{copy}_2}})^{\alpha^2} \cdot (c_{L_{\mathsf{division}}})^{-1},
\end{aligned}
$$

where $Z_H(\delta) = \delta^n - 1,\ \mathcal{L}_i(\delta) = \dfrac{\omega^i(\delta^n-1)}{n(\delta-\omega^i)},\ S_{\mathsf{PI}}(\delta) = \displaystyle\sum_{i\in[\ell]} -x_i\mathcal{L}_i(\delta).$

Then, compute

$$
C_1 \coloneqq c_L \cdot \left(\frac{c_A}{\mathsf{g}_1^{a_\delta}}\right)^{\varepsilon} \cdot \left(\frac{c_B}{\mathsf{g}_1^{b_\delta}}\right)^{\varepsilon^2} \cdot \left(\frac{c_C}{\mathsf{g}_1^{c_\delta}}\right)^{\varepsilon^3} \cdot \left(\frac{c_{S_{\sigma,1}}}{\mathsf{g}_1^{s_{\sigma_1,\delta}}}\right)^{\varepsilon^4} \cdot \left(\frac{c_{S_{\sigma,2}}}{\mathsf{g}_1^{s_{\sigma_2,\delta}}}\right)^{\varepsilon^5},\ C_2 \coloneqq \frac{c_\Phi}{\mathsf{g}_1^{\phi_{\delta\omega}}},
$$

and accept iff

$$
e\Big(C_1 \cdot {\pi_1}^{\delta} \cdot \big(C_2 \cdot {\pi_2}^{\delta\omega}\big)^{\zeta}, \mathsf{g}_2\Big) \stackrel{?}{=} e\Big(\pi_1 \cdot {\pi_2}^{\zeta}, \mathsf{g}_2^{\tau}\Big).
$$

Statistical completeness. Note that in step 3 of PLONK's prover algorithm, it is possible that $\frac{f(\omega^j)}{g(\omega^j)}$ is undefined due to $g(\omega^j) = 0$ for some $j \in [n]$. In this case, the protocol has to be aborted. Fortunately, this can only happen with negligible probability $\leq 3n/p$ over the choice of the permutation challenge $\gamma \in \mathbb{F}_p$, i.e, PLONK is statistically complete.

Vulnerability in a previous version of PLONK. Since the authors of PLONK never constructed a simulator to formally prove that PLONK is zero-knowledge (cf. [GWC19a, Sec. 8]), our attempt to do so led to the discovery of a vulnerability in their original implementation of step 5 of the Prove algorithm [GWC19b].

In this step, the prover decomposes the computed quotient polynomial $T(X)$ of degree $\leq 3n+5$ into three lower-degree polynomials as an optimization that keeps the SRS size minimal with respect to the circuit size n and the degree of the other polynomials committed to in a PLONK proof. In the original specification, the prover directly commits to the polynomials $T_1', T_2', T_3' \in \mathbb{F}_p^{(\leq n+1)}[X]$ (instead of the randomized polynomials $T_1, T_2, T_3 \in \mathbb{F}_p^{(\leq n+2)}[X]$ used in the fixed version). Even though a simulator is able to compute the correct evaluation of $T(\tau)$, it does not have enough information to compute the correct values $T_1'(\tau), T_2'(\tau), T_3'(\tau)$ satisfying $T(\tau) = T_1'(\tau) + \tau^{n+2} \cdot T_2'(\tau) + \tau^{2n+4} \cdot T_3'(\tau)$. While there are p^2 possible

triples in $\mathbb{F}_p^3$ satisfying this equation, only one of them corresponds to the correct distribution of $T_1'(\tau), T_2'(\tau), T_3'(\tau)$ in the execution of the real protocol.

We fixed this issue in collaboration with the authors of PLONK by randomizing these three polynomials in a way that preserves how they combine to T while enabling a correct simulation. Essentially, they are randomized such that the values of $T_1(\tau), T_2(\tau), T_3(\tau)$ are distributed uniformly in $\mathbb{F}_p^3$ conditioned on satisfying $T(\tau) = T_1(\tau) + \tau^{n+2} \cdot T_2(\tau) + \tau^{2n+4} \cdot T_3(\tau)$. We discuss this fix in more detail in the next section, where we present our simulator.

3 PLONK is statistically zero-knowledge

As our main result we prove the following theorem by constructing a simulator for PLONK and arguing that it has the correct output distribution.

Theorem 1. *The patched version of PLONK (Construction 1) is statistically zero-knowledge.*

To show this, we will rely on the following lemma (for a proof, see Appendix A.2), which explains why the prover's witness polynomials are randomized by adding the product of a random blinding polynomial and Z_H.

Lemma 1. *Let $S \subset \mathbb{F}_p$ and $Z_S(X) \coloneqq \prod_{a \in S}(X - a)$. Fix a polynomial $f \in \mathbb{F}_p[X]$ and any distinct values $x_1, \ldots, x_k \in \mathbb{F}_p \setminus S$. Then the following distribution is uniform in $\mathbb{F}_p^k$:*

1. *Choose a random polynomial $\rho \leftarrow \mathbb{F}_p^{(\leq k-1)}[X]$ of degree $k-1$ and define*

$$\tilde{f}(X) \coloneqq f(X) + Z_S(X)\rho(X).$$

2. *Output $\big(\tilde{f}(x_1), \ldots, \tilde{f}(x_k)\big) \in \mathbb{F}_p^k$.*

As a consequence, the number of independent and uniform evaluations of a polynomial randomized in this way depends on the degree of the used blinding polynomial $\rho(X)$, i.e., a constant blinding polynomial enables the simulation of a single evaluation, a linear polynomial of two, etc. This also explains why the wire polynomials A, B, C are randomized using linear blinding polynomials, while the permutation check polynomial Φ uses a quadratic blinding polynomial: A, B, C, and Φ have to account for their commitments (which correspond to evaluations at τ) as well as the openings at the evaluation challenge δ (respectively $\delta\omega$ in the case of Φ), but Φ additionally has to compensate for the commitment to the quotient polynomial T (or its decomposition into T_1, T_2, T_3), since the value of $T(\tau)$ depends on $\Phi(\tau\omega)$ (cf. step 5 of Prove in Construction 1).

An important condition of the lemma is that the evaluation points come from $\mathbb{F}_p \setminus H$, which means that both the SRS trapdoor τ and the evaluation challenge δ must not be in H. Since $H \coloneqq \langle\omega\rangle = \{\omega^1, \omega^2, \ldots, \omega^n\}$ with $n = \mathsf{poly}(\lambda)$, we have $|H| \,/\, |\mathbb{F}_p| = \mathsf{negl}(\lambda)$, i.e., we can ignore this case when constructing a simulator for PLONK and still obtain statistical zero knowledge. Also, since

H is a multiplicative order-n subgroup of $\mathbb{F}_p$, this ensures that the remaining evaluation points $\tau\omega$ and $\delta\omega$ come from $\mathbb{F}_p \setminus H$ as well.

In summary, as long as $\tau, \delta \in \mathbb{F}_p \setminus H$, the commitments to the witness polynomials A, B, C, Φ using evaluations at τ as well as the evaluations of these polynomials at the points δ and $\delta\omega$ are distributed independently and uniformly in $\mathbb{F}_p$, hiding any information about the prover's witness.

The simulator. We present our simulator $\mathcal{S}$ for PLONK in Construction 2. We will assume that both the SRS trapdoor τ and the evaluation challenge δ come from $\mathbb{F}_p \setminus H$. In this case, $\mathcal{S}$ perfectly simulates PLONK when ignoring the aborts caused by denominators being zero in the computation of the permutation check polynomial Φ, as well as the cases $\delta = \tau$ or $\delta\omega = \tau$, which is why we attain *statistical* zero knowledge. Furthermore, we only give the second part of the simulator from Definition 5, implicitly assuming it runs the Setup algorithm to create a well-formed srs with a uniform trapdoor $\tau \in \mathbb{F}_p \setminus H$ as its first output.

Construction 2: Simulator for PLONK.

$\underline{\mathcal{S}\big(\tau, \mathsf{pp}, (x_i)_{i\in[\ell]}\big)}$:

1. Choose random $a_\tau, b_\tau, c_\tau \leftarrow \mathbb{F}_p$ and compute the commitments
$$c_A \coloneqq \mathsf{g}_1^{a_\tau},\ c_B \coloneqq \mathsf{g}_1^{b_\tau},\ c_C \coloneqq \mathsf{g}_1^{c_\tau}.$$
2. Compute the permutation challenges $\beta, \gamma \in \mathbb{F}_p$.
3. Choose random $\phi_\tau \leftarrow \mathbb{F}_p$ and compute the commitment $c_\Phi \coloneqq \mathsf{g}_1^{\phi_\tau}$.
4. Compute the quotient challenge $\alpha \in \mathbb{F}_p$.
5. Choose random $\phi_{\tau\omega} \leftarrow \mathbb{F}_p$ (if $\tau = 0$, set $\phi_{\tau\omega} \coloneqq \phi_\tau$ instead) and compute the evaluation of the quotient polynomial T at τ as
$$t_\tau \coloneqq \frac{1}{Z_H(\tau)}\left(\begin{array}{c}\big[S_{\mathsf{L}}(\tau)a_\tau + S_{\mathsf{R}}(\tau)b_\tau + S_{\mathsf{O}}(\tau)c_\tau + S_{\mathsf{M}}(\tau)a_\tau b_\tau + S_{\mathsf{C}}(\tau) + S_{\mathsf{PI}}(\tau)\big] \\ +\alpha\Big[\begin{array}{c}\phi_\tau(a_\tau+\beta\tau+\gamma)(b_\tau+\beta k_1\tau+\gamma)(c_\tau+\beta k_2\tau+\gamma) \\ -\phi_{\tau\omega}(a_\tau+\beta S_{\sigma,1}(\tau)+\gamma)(b_\tau+\beta S_{\sigma,2}(\tau)+\gamma)(c_\tau+\beta S_{\sigma,3}(\tau)+\gamma)\end{array}\Big] \\ +\alpha^2\big[\mathcal{L}_1(\tau)(\phi_\tau - 1)\big]\end{array}\right).$$
Choose random $t_2, t_3 \leftarrow \mathbb{F}_p$ and compute $t_1 \coloneqq t_\tau - \tau^{n+2}\cdot t_2 - \tau^{2n+4}\cdot t_3$ such that $t_\tau = t_1 + \tau^{n+2}\cdot t_2 + \tau^{2n+4}\cdot t_3$. Compute the commitments
$$c_{T_1} \coloneqq \mathsf{g}_1^{t_1},\ c_{T_2} \coloneqq \mathsf{g}_1^{t_2},\ c_{T_3} \coloneqq \mathsf{g}_1^{t_3}.$$
6. Compute the evaluation challenge $\delta \in \mathbb{F}_p$.
7. If $\delta = \tau$ or $\delta\omega = \tau$, abort. Otherwise, choose random $a_\delta, b_\delta, c_\delta, \phi_{\delta\omega} \leftarrow \mathbb{F}_p$ and compute the evaluations $s_{\sigma_1,\delta} \coloneqq S_{\sigma,1}(\delta)$, $s_{\sigma_2,\delta} \coloneqq S_{\sigma,2}(\delta)$.
8. Compute the opening challenge $\varepsilon \in \mathbb{F}_p$.

9. Compute the evaluation of the linearization polynomial L at τ as

$$l_\tau \coloneqq \begin{pmatrix} \left[a_\delta S_{\mathsf{L}}(\tau)+b_\delta S_{\mathsf{R}}(\tau)+c_\delta S_{\mathsf{O}}(\tau)+a_\delta b_\delta S_{\mathsf{M}}(\tau)+S_{\mathsf{C}}(\tau)+S_{\mathsf{PI}}(\delta)\right] \\ +\alpha\Big[\begin{matrix}\phi_\tau(a_\delta+\beta\delta+\gamma)(b_\delta+\beta k_1\delta+\gamma)(c_\delta+\beta k_2\delta+\gamma) \\ -\phi_{\delta\omega}(a_\delta+\beta S_{\sigma,1}(\delta)+\gamma)(b_\delta+\beta S_{\sigma,2}(\delta)+\gamma)(c_\delta+\beta S_{\sigma,3}(\tau)+\gamma)\end{matrix}\Big] \\ +\alpha^2\left[\mathcal{L}_1(\delta)(\phi_\tau-1)\right]-Z_H(\delta)\left[t_1+\delta^{n+2}t_2+\delta^{2n+4}t_3\right] \end{pmatrix}.$$

Then, compute the opening proofs as

$$\pi_1 \coloneqq \mathsf{g}_1^{\frac{1}{\tau-\delta}\left(\begin{matrix} l_\tau+\varepsilon(a_\tau-a_\delta)+\varepsilon^2(b_\tau-b_\delta)+\varepsilon^3(c_\tau-c_\delta) \\ +\varepsilon^4(S_{\sigma,1}(\tau)-S_{\sigma,1}(\delta))+\varepsilon^5(S_{\sigma,2}(\tau)-S_{\sigma,2}(\delta))\end{matrix}\right)}, \ \pi_2 \coloneqq \mathsf{g}_1^{\frac{\phi_\tau-\phi_{\delta\omega}}{\tau-\delta\omega}}.$$

Output the simulated proof

$$\tilde{\pi}_{\text{PLONK}} \coloneqq \begin{pmatrix} c_A, c_B, c_C, c_\Phi, c_{T_1}, c_{T_2}, c_{T_3}, \pi_1, \pi_2, \\ a_\delta, b_\delta, c_\delta, s_{\sigma_1,\delta}, s_{\sigma_2,\delta}, \phi_{\delta\omega} \end{pmatrix} \in \mathbb{G}_1^9 \times \mathbb{F}_p^6.$$

Proof. Let us argue why $\mathcal{S}$ perfectly simulates PLONK when ignoring the aborts caused by $\Phi \notin \mathbb{F}_p[X]$, i.e., the permutation check polynomial being undefined as discussed in Section 2.2, as well as the cases $\delta = \tau$ or $\delta\omega = \tau$. Specifically, we will show that under these conditions all the elements in a real PLONK proof π_{PLONK} are either uniformly random or determined by the verifier equations, and that a proof $\tilde{\pi}_{\text{PLONK}}$ generated by our simulator $\mathcal{S}$ has exactly the same distribution.

To this end, we begin by analyzing the distribution of all the elements contained in a real PLONK proof

$$\pi_{\text{PLONK}} \coloneqq \begin{pmatrix} \mathsf{g}_1^{A(\tau)}, \mathsf{g}_1^{B(\tau)}, \mathsf{g}_1^{C(\tau)}, \mathsf{g}_1^{\Phi(\tau)}, \mathsf{g}_1^{T_1(\tau)}, \mathsf{g}_1^{T_2(\tau)}, \mathsf{g}_1^{T_3(\tau)}, \mathsf{g}_1^{Q_1(\tau)}, \mathsf{g}_1^{Q_2(\tau)}, \\ A(\delta), B(\delta), C(\delta), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta), \Phi(\delta\omega) \end{pmatrix}.$$

Note that $S_{\sigma,1}(\delta), S_{\sigma,2}(\delta)$ are the evaluations of the public polynomials $S_{\sigma,1}, S_{\sigma,1}$ at the evaluation challenge δ, and hence trivially correctly simulated. Moreover, $\mathsf{g}_1^{A(\tau)}, \mathsf{g}_1^{B(\tau)}, \mathsf{g}_1^{C(\tau)}, \mathsf{g}_1^{\Phi(\tau)}, A(\delta), B(\delta), C(\delta), \Phi(\delta\omega)$ are just uniform group and field elements due to the randomization of the polynomials A, B, C, Φ according to Lemma 1 and our assumption that $\tau, \delta \in \mathbb{F}_p \setminus H$. This is exactly how $\mathcal{S}$ chooses $a_\tau, b_\tau, c_\tau, \phi_\tau \in \mathbb{F}_p$ in steps 1 and 3 to produce the commitments to the polynomials A, B, C, Φ. The same is true in step 7, where $\mathcal{S}$ again outputs uniform values $a_\delta, b_\delta, c_\delta, \phi_{\delta\omega} \in \mathbb{F}_p$ as the evaluations $A(\delta), B(\delta), C(\delta), \Phi(\delta\omega)$.

The commitment to the quotient polynomial $\mathsf{g}_1^{T(\tau)}$ in step 5 can be simulated directly, as $T(\tau)$ is a deterministic function in τ with all the necessary inputs known to the simulator (cf. the computation of t_τ in step 5 of $\mathcal{S}$). On the contrary, simulating the commitments to the decomposed polynomials T_1, T_2, T_3 satisfying

$$T(X) = T_1(X) + X^{n+2} \cdot T_2(X) + X^{2n+4} \cdot T_3(X)$$

is more delicate. As explained in the previous section, this is exactly where the specification of PLONK prior to our fix, which instead commits to the deterministic polynomials T_1', T_2', T_3', leaks information about the prover's witness polyno-

mials, and thus cannot be perfectly simulated. Instead, let us argue why $\mathcal{S}$ correctly simulates the commitments to the now randomized polynomials T_1, T_2, T_3 in step 5. By the randomness of $\rho_{10}, \rho_{11} \in \mathbb{F}_p$, both $T_2(\tau) = T'_2(\tau) - \rho_{10} + \rho_{11} \cdot \tau^{n+2}$ and $T_3(\tau) = T'_3(\tau) - \rho_{11}$ are distributed independently and uniformly in $\mathbb{F}_p$. The value of $T_1(\tau) = T'_1(\tau) + \rho_{10} \cdot \tau^{n+2}$ is then uniquely determined by the equality $T(\tau) = T_1(\tau) + \tau^{n+2} \cdot T_2(\tau) + \tau^{2n+4} \cdot T_3(\tau)$, which is precisely how $\mathcal{S}$ chooses t_1, t_2, t_3 to compute its commitments $c_{T_1} \coloneqq \mathsf{g}_1^{t_1}$, $c_{T_2} \coloneqq \mathsf{g}_1^{t_2}$, $c_{T_3} \coloneqq \mathsf{g}_1^{t_3}$.

Lastly, $\mathcal{S}$ also correctly simulates the opening proofs $\mathsf{g}_1^{Q_1(\tau)}, \mathsf{g}_1^{Q_2(\tau)}$ in step 9 when $\delta \neq \tau$ and $\delta\omega \neq \tau$, since both of them are deterministic functions in τ in this case, with all the necessary inputs known to the simulator. For example, see how $\mathcal{S}$ computes the value l_τ, which is the evaluation of the linearization polynomial $L(\tau)$ required for $\mathsf{g}_1^{Q_1(\tau)}$.

Having established that $\mathcal{S}$ perfectly simulates PLONK conditioned on the event $\tau, \delta \in \mathbb{F}_p \setminus H$, $\tau \neq \delta$, $\tau \neq \delta\omega$, and Φ being well-defined, i.e., $\Phi \in \mathbb{F}_p[X]$, we can turn this into a formal proof of statistical zero knowledge as defined in Definition 5. Let $\mathtt{F}$ denote the complementary event in which our simulator $\mathcal{S}$ is unable to produce correctly distributed proofs. Recalling from Section 2.2 that the abort probability due to Φ being undefined is at most $3n/p$, we get

$$\begin{aligned}\Pr[\mathtt{F}] &= \Pr\big[\tau \in H \vee \delta \in H \vee \tau = \delta \vee \tau = \delta\omega \vee \Phi \notin \mathbb{F}_p[X]\big] \\ &\leq \frac{n + n + 1 + 1 + 3n}{p} = \frac{5n+2}{p}.\end{aligned}$$

With this, we can show that the difference of the two probabilities in Definition 5 is negligible as well. Using $\mathtt{I}_\mathcal{P}$ and $\mathtt{I}_\mathcal{S}$ to denote the respective two events from the definition, i.e., the adversary interacting with the honest prover $\mathcal{P}$ or with our simulator $\mathcal{S}$, we have $\Pr[\mathtt{I}_\mathcal{P} \mid \bar{\mathtt{F}}] = \Pr[\mathtt{I}_\mathcal{S} \mid \bar{\mathtt{F}}]$ by the arguments laid out above, and thus $\Pr[\mathtt{I}_\mathcal{P} \wedge \bar{\mathtt{F}}] = \Pr[\mathtt{I}_\mathcal{S} \wedge \bar{\mathtt{F}}]$. From this we get

$$\begin{aligned}\big|\Pr[\mathtt{I}_\mathcal{P}] - \Pr[\mathtt{I}_\mathcal{S}]\big| &= \big|\big(\Pr[\mathtt{I}_\mathcal{P} \wedge \mathtt{F}] + \Pr[\mathtt{I}_\mathcal{P} \wedge \bar{\mathtt{F}}]\big) - \big(\Pr[\mathtt{I}_\mathcal{S} \wedge \mathtt{F}] + \Pr[\mathtt{I}_\mathcal{S} \wedge \bar{\mathtt{F}}]\big)\big| \\ &= \big|\Pr[\mathtt{I}_\mathcal{P} \wedge \mathtt{F}] - \Pr[\mathtt{I}_\mathcal{S} \wedge \mathtt{F}]\big| \leq \frac{5n+2}{p} = \mathsf{negl}(\lambda).\end{aligned}$$

The final inequality follows from the fact that both $\Pr[\mathtt{I}_\mathcal{P} \wedge \mathtt{F}]$ and $\Pr[\mathtt{I}_\mathcal{S} \wedge \mathtt{F}]$ are at most $\Pr[\mathtt{F}]$, and so their difference cannot be any larger. This finishes the proof of PLONK's statistical zero knowledge. □

4 Old PLONK is not statistically zero-knowledge

We demonstrate that the previous version of PLONK [GWC19b], where the prover directly commits to the polynomials $T'_1, T'_2, T'_3 \in \mathbb{F}_p^{(\leq n+1)}[X]$ instead of the randomized $T_1, T_2, T_3 \in \mathbb{F}_p^{(\leq n+2)}[X]$ (cf. step 5 of Prove in Construction 1), is not statistically zero-knowledge, aligning with our intuition of not being able to find a way to simulate its proofs.

Theorem 2. *The previous version of PLONK is not statistically zero-knowledge.*

We will prove this claim by showing that this old version of PLONK does not even satisfy statistical witness indistinguishability, which by the discussion following Definition 6 gives the result.

Proposition 1. *The previous version of PLONK is not statistically witness-indistinguishable.*

Proof. We construct an adversary $\mathcal{A}$ (see Construction 3) that breaks witness indistinguishability in the case of $n = 1$, concretely, for the statement of multiplying any two values $w_1, w_2 \in \mathbb{F}_p$ such that $w_3 = w_1 \cdot w_2 \pmod p$. This corresponds to a circuit with a single multiplication gate and no public inputs as depicted in Figure 1. In the PLONK constraint system this is expressed as the statement output by $\mathcal{A}$ in step 1.

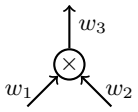

Fig. 1. A single multiplication gate with private values $w_1, w_2, w_3 \in \mathbb{F}_p$.

Since the adversary is unbounded, we will assume that whenever it gets a value of the form g_1^x, it implicitly recovers $x \in \mathbb{F}_p$, e.g., when seeing a KZG [KZG10] commitment $\mathsf{g}_1^{f(\tau)}$ to a polynomial $f \in \mathbb{F}_p[X]$, it learns $f(\tau)$.

Construction 3: Adversary breaking statistical witness indistinguishability of old PLONK.

$\mathcal{A}(\mathsf{srs})$:

1. Choose all distinct $w_1, w_2, w_3, w_1', w_2', w_3' \in \mathbb{F}_p$ such that $w_1 \cdot w_2 = w_3$ and $w_1' \cdot w_2' = w_3'$. Define the identity permutation $\sigma := (1)(2)(3)$, $\mathtt{i} := \big((0, 0, -1, 1, 0), \sigma\big)$, $\mathtt{x} := \emptyset$, $\mathtt{w}_0 := (w_1, w_2, w_3)$, $\mathtt{w}_1 := (w_1', w_2', w_3')$, and output $(\mathtt{i}, \mathtt{x}, \mathtt{w}_0, \mathtt{w}_1)$.
2. Receive the proof
$$\pi_{\text{PLONK}} := \left(\begin{matrix}\mathsf{g}_1^{A(\tau)}, \mathsf{g}_1^{B(\tau)}, \mathsf{g}_1^{C(\tau)}, \mathsf{g}_1^{\Phi(\tau)}, \mathsf{g}_1^{T_1'(\tau)}, \mathsf{g}_1^{T_2'(\tau)}, \mathsf{g}_1^{T_3'(\tau)}, \pi_1, \pi_2, \\ A(\delta), B(\delta), C(\delta), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta), \Phi(\delta\omega)\end{matrix}\right).$$
3. Compute the following values over $\mathbb{F}_p$ as
$$\rho_1^* := \big[(A(\tau) - w_1)(\tau - 1)^{-1} - (A(\delta) - w_1)(\delta - 1)^{-1}\big](\tau - \delta)^{-1},$$
$$\rho_3^* := \big[(B(\tau) - w_2)(\tau - 1)^{-1} - (B(\delta) - w_2)(\delta - 1)^{-1}\big](\tau - \delta)^{-1}.$$
If any of the inverses is undefined, or if $\rho_1^* \cdot \rho_3^* = T_2'(\tau)$, output 0; otherwise, output 1.

For this particular statement, the parameters involved in the computation of a PLONK proof simplify to:

$$
\begin{aligned}
&\mathsf{srs} = \left(\mathsf{g}_1, \mathsf{g}_1^{\tau}, \mathsf{g}_1^{\tau^2}, \mathsf{g}_1^{\tau^3}, \mathsf{g}_2, \mathsf{g}_2^{\tau}\right),\ \omega = 1,\ \mathcal{L}_1(X) = 1,\ Z_H(X) = X - 1,\\
&S_{\mathsf{L}}(X) = 0,\ S_{\mathsf{R}}(X) = 0,\ S_{\mathsf{O}}(X) = -1,\ S_{\mathsf{M}}(X) = 1,\ S_{\mathsf{C}}(X) = 0,\ S_{\mathsf{PI}}(X) = 0,\\
&S_{\mathrm{id},1}(X) = S_{\sigma,1}(X) = 1,\ S_{\mathrm{id},2}(X) = S_{\sigma,2}(X) = k_1,\ S_{\mathrm{id},3}(X) = S_{\sigma,3}(X) = k_2,\\
&A(X) = (\rho_1 X + \rho_2)Z_H(X) + w_1,\\
&B(X) = (\rho_3 X + \rho_4)Z_H(X) + w_2,\\
&C(X) = (\rho_5 X + \rho_6)Z_H(X) + w_3,\\
&f(X) = g(X) = (A(X) + \beta + \gamma)(B(X) + \beta k_1 + \gamma)(C(X) + \beta k_2 + \gamma),\\
&\Phi(X) = (\rho_7 X^2 + \rho_8 X + \rho_9)Z_H(X) + 1.
\end{aligned}
$$

The quotient polynomial is computed as

$$
T(X) \coloneqq \frac{\mathsf{gates}(X) + \alpha \cdot \mathsf{copy}_1(X) + \alpha^2 \cdot \mathsf{copy}_2(X)}{Z_H(X)},
$$

$$
\begin{aligned}
\text{where } \mathsf{gates}(X) &\coloneqq \begin{pmatrix} S_{\mathsf{L}}(X)A(X) + S_{\mathsf{R}}(X)B(X) + S_{\mathsf{O}}(X)C(X) \\ + S_{\mathsf{M}}(X)A(X)B(X) + S_{\mathsf{C}}(X) + S_{\mathsf{PI}}(X) \end{pmatrix} = A(X)B(X) - C(X),\\
\mathsf{copy}_1(X) &\coloneqq \Phi(X)f(X) - \Phi(X\omega)g(X) = 0,\\
\mathsf{copy}_2(X) &\coloneqq \mathcal{L}_1(X)(\Phi(X) - 1) = (\rho_7 X^2 + \rho_8 X + \rho_9)Z_H(X).
\end{aligned}
$$

Concretely, this simplifies to:

$$
\begin{aligned}
T(X) = {} & (\rho_1\rho_3)X^3 + (\rho_1\rho_4 + \rho_2\rho_3 - \rho_1\rho_3 + \alpha^2\rho_7)X^2\\
& + (w_1\rho_3 + w_2\rho_1 + \rho_2\rho_4 - \rho_1\rho_4 - \rho_2\rho_3 - \rho_5 + \alpha^2\rho_8)X\\
& + w_1\rho_4 + w_2\rho_2 - \rho_2\rho_4 - \rho_6 + \alpha^2\rho_9
\end{aligned}
$$

Recall that $T_1', T_2', T_3' \in \mathbb{F}_p^{(\leq n+1)}[X]$ and that $n + 1 = 2$ here. Thus, we get

$$
T_1'(X) = T(X) - (\rho_1\rho_3)X^3, \quad T_2'(X) = \rho_1\rho_3, \quad T_3'(X) = 0.
$$

Note, in particular, that the commitment to T_2' leaks the product of the blinding scalars ρ_1, ρ_3. We will show how the adversary can recover the correct values of ρ_1, ρ_3 assuming it chooses the correct witness for this computation. Finally, the adversary just has to check if the resulting product matches the given value $T_2'(\tau)$ to distinguish the two witnesses.

Assume the proof was computed using the first witness $\mathbb{w}_0 = (w_1, w_2, w_3)$, i.e., $A(X) = (\rho_1 X + \rho_2)(X - 1) + w_1$. Then, with the values $\tau, \delta, A(\tau), A(\delta)$ all known to $\mathcal{A}$, it can solve the following system of 2 linear equations in the 2 unknowns ρ_1, ρ_2 over $\mathbb{F}_p$:

$$
\begin{aligned}
\tau\rho_1 + \rho_2 &= (A(\tau) - w_1)(\tau - 1)^{-1}\\
\delta\rho_1 + \rho_2 &= (A(\delta) - w_1)(\delta - 1)^{-1}
\end{aligned}
$$

Concretely, $\mathcal{A}$ computes $(A(\tau)-w_1)(\tau-1)^{-1}-(A(\delta)-w_1)(\delta-1)^{-1}=(\tau-\delta)\rho_1$, and then multiplies by $(\tau-\delta)^{-1}$ to obtain ρ_1. Note that the probability that any of these inverses does not exist is negligible. Using the values $\tau, \delta, B(\tau), B(\delta)$, the adversary can recover ρ_3 in an analogous fashion.

Let $\mathtt{I}_{\mathcal{A},\mathbb{w}_0}$ and $\mathtt{I}_{\mathcal{A},\mathbb{w}_1}$ denote the two events inside of the probabilities in Definition 6, i.e., $\mathcal{A}$ interacting with proofs generated under the witness $\mathbb{w}_0$ or $\mathbb{w}_1$, respectively. From the arguments laid out above, it follows that $\Pr[\mathtt{I}_{\mathcal{A},\mathbb{w}_0}]=0$. So in order for $\mathcal{A}$ to break PLONK's witness indistinguishability, it suffices to show that $\Pr[\mathtt{I}_{\mathcal{A},\mathbb{w}_1}]$ is non-negligible. We will argue that the only two events making $\mathcal{A}$ output 0 in this case are negligible, concretely:

1. At least one of the elements $(\tau-1),(\delta-1),(\tau-\delta)$ is not invertible.
2. It holds that $\rho_1^*\cdot\rho_3^*=T_2'(\tau)$.

The former can be trivially bounded by $3/p$, while the latter requires some more analysis. Assume the former event does not occur, i.e., $(\tau-1),(\delta-1),(\tau-\delta)$ are non-zero. Then we want to bound the probability that $\rho_1^*\cdot\rho_3^*=\rho_1\cdot\rho_3$, where ρ_1^*,ρ_3^* are computed according to step 3 of Construction 3 and ρ_1,ρ_3 are the correct blinding scalars, i.e., can be computed the same way as ρ_1^*,ρ_3^* but with w_1,w_2 replaced by w_1',w_2'. After doing some simplification, i.e., multiplying both sides of $\rho_1^*\cdot\rho_3^*=\rho_1\cdot\rho_3$ by $(\tau-\delta)^2(\tau-1)(\delta-1)$, we can look at this as a polynomial equation in δ over $\mathbb{F}_p$. One can verify (by expanding the respective expressions) that the highest occurring degree of δ in this equation is at most 4. Since $\delta\in\mathbb{F}_p\setminus\{1,\tau\}$, we can apply the Schwartz–Zippel lemma to see that the probability of $\rho_1^*\cdot\rho_3^*=T_2'(\tau)$ is at most $4/(p-2)$ over the choice of δ. Altogether, we get

$$\Pr[\mathtt{I}_{\mathcal{A},\mathbb{w}_1}]\geq 1-\frac{3}{p}-\frac{4}{p-2}\geq 1-\frac{7}{p-2},$$

which is clearly non-negligible, finishing the proof. □

The general case. In general, it is possible for an unbounded adversary to recover the values of all the blinding scalars $\rho_1,\ldots,\rho_9$ used to mask the polynomials A,B,C,Φ in a similar fashion as in the special case $n=1$ above. The values of $\rho_1,\ldots,\rho_6$ can be obtained by solving the 3 systems of linear equations resulting from the given evaluations $\{A(\tau),A(\delta)\},\{B(\tau),B(\delta)\},\{C(\tau),C(\delta)\}$, respectively. The values of ρ_7,ρ_8,ρ_9 can be obtained using the given evaluations $\Phi(\tau),\Phi(\delta\omega)$, as well as recovering the evaluation $\Phi(\tau\omega)$, which is used to compute $\mathsf{copy}_1(X)\coloneqq\Phi(X)f(X)-\Phi(\omega X)g(X)$ in step 5 of Prove in Construction 1. More precisely, we compute $T(\tau)$ as $T_1'(\tau)+\tau^{n+2}T_2'(\tau)+\tau^{2n+4}T_3'(\tau)$, and then apply some basic arithmetic involving the given values $A(\tau),B(\tau),C(\tau),\Phi(\tau)$ to recover $\Phi(\tau\omega)$. This then results in a fully determined system of 3 linear equations with 3 unknowns. Finally, it is again possible to compute candidate values for $T_1'(\tau),T_2'(\tau),T_3'(\tau)$ and check (with high probability) which of the two witnesses was used to compute the proof.

5 Conclusion

This work highlights the importance of rigorous security proofs in the design of cryptographic schemes. In the case of PLONK, the lack thereof led to a vulnerability in its zero-knowledge implementation, which we discovered and fixed as a result of this work. Moreover, we proved that the resulting specification of PLONK achieves statistical zero knowledge.

On the contrary, we showed that the old specification does not satisfy this claimed notion of zero knowledge. We leave it for future work to prove or disprove whether the previous version of PLONK achieves computational zero knowledge.

Acknowledgments. This work was funded by the Vienna Science and Technology Fund (WWTF) [10.47379/VRG18002] and by the Austrian Science Fund (FWF) [10.55776/F8515-N]. I would like to thank my doctoral supervisor Georg Fuchsbauer for helpful comments. Further, I thank Ariel Gabizon for discussing the found vulnerability in PLONK's zero-knowledge implementation and possible fixes.

References

BBB+18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334, 2018. https://doi.org/10.1109/SP.2018.00020.

BCCT12. Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 326–349. Association for Computing Machinery, 2012. https://doi.org/10.1145/2090236.2090263.

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 62–73. Association for Computing Machinery, 1993. https://doi.org/10.1145/168588.168596.

BSBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Paper 2018/046, 2018. https://eprint.iacr.org/2018/046.

CBBZ23. Binyi Chen, Benedikt Bünz, Dan Boneh, and Zhenfei Zhang. Hyperplonk: Plonk with linear-time prover and high-degree custom gates. In *Advances in Cryptology – EUROCRYPT 2023*, volume 14005 of *LNCS*, pages 499–530. Springer, 2023. https://doi.org/10.1007/978-3-031-30617-4_17.

CHM+20. Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zksnarks with universal and updatable srs. In *Advances in Cryptology – EUROCRYPT 2020*, volume 12105 of *LNCS*, pages 738–768. Springer, 2020. https://doi.org/10.1007/978-3-030-45721-1_26.

Ele22. Electric Coin Company. The halo2 book, 2022. https://zcash.github.io/halo2/index.html.

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology – CRYPTO '86*, volume 263 of *LNCS*, pages 186–194. Springer, 1987. https://doi.org/10.1007/3-540-47721-7_12.

FS90. Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, STOC '90, pages 416–426. Association for Computing Machinery, 1990. https://doi.org/10.1145/100216.100272.

GMR85. Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304. Association for Computing Machinery, 1985. https://doi.org/10.1145/22145.22178.

Gro16. Jens Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology – EUROCRYPT 2016*, volume 9666 of *LNCS*, pages 305–326. Springer, 2016. https://doi.org/10.1007/978-3-662-49896-5_11.

GW20. Ariel Gabizon and Zachary J. Williamson. plookup: A simplified polynomial protocol for lookup tables. Cryptology ePrint Archive, Paper 2020/315, 2020. https://eprint.iacr.org/2020/315.

GW22. Ariel Gabizon and Zachary J. Williamson. Proposal: The turbo-plonk program syntax for specifying snark programs, 2022. https://docs.zkproof.org/pages/standards/accepted-workshop3/proposal-turbo_plonk.pdf.

GWC19a. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Paper 2019/953, 2019. https://eprint.iacr.org/2019/953.

GWC19b. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Paper 2019/953, version 20220429:112734, 2019. https://eprint.iacr.org/archive/2019/953/20220429:112734.

HBHW22. Daira Hopwood, Sean Bowe, Taylor Hornby, and Nathan Wilcox. Zcash protocol specification. Technical report, Electric Coin Company, 2022. https://zips.z.cash/protocol/protocol.pdf.

KZG10. Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 177–194. Springer, 2010. https://doi.org/10.1007/978-3-642-17373-8_11.

MBKM19. Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. 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 '19, pages 2111–2128. Association for Computing Machinery, 2019. https://doi.org/10.1145/3319535.3339817.

Mil22. Jim Miller. The frozen heart vulnerability in plonk, 2022. https://blog.trailofbits.com/2022/04/18/the-frozen-heart-vulnerability-in-plonk/.

PFM+22. Luke Pearson, Joshua Fitzgerald, Héctor Masip, Marta Bellés-Muñoz, and Jose Luis Muñoz-Tapia. Plonkup: Reconciling plonk with plookup. Cryptology ePrint Archive, Paper 2022/086, 2022. https://eprint.iacr.org/2022/086.

A Proofs

A.1 ZK implies WI

We prove that zero knowledge, as formalized in Definition 5, implies witness indistinguishability, as formalized in Definition 6, for zk-SNARKs.

Proof. Assume $\Pi = (\mathsf{Setup}, \mathsf{Preproc}, \mathsf{Prove}, \mathsf{Verify})$ is zero-knowledge. Let $\mathcal{A}$ be an arbitrary adversary against the witness indistinguishability of Π. Without loss of generality, assume that $\mathtt{w}_0 \neq \mathtt{w}_1$. Construct the two adversaries $\mathcal{A}_0, \mathcal{A}_1$ against zero knowledge of Π that behave exactly as $\mathcal{A}$, but instead of returning $(\mathtt{i}, \mathtt{x}, \mathtt{w}_0, \mathtt{w}_1, \mathsf{st})$ as their first output, $\mathcal{A}_0$ only returns $(\mathtt{i}, \mathtt{x}, \mathtt{w}_0, \mathsf{st})$, and $\mathcal{A}_1$ only returns $(\mathtt{i}, \mathtt{x}, \mathtt{w}_1, \mathsf{st})$.

For $i \in \{0, 1\}$, let $\mathtt{I}_{\mathcal{A}_i,\mathcal{P}}$ and $\mathtt{I}_{\mathcal{A}_i,\mathcal{S}}$ denote the two events from the definition of zero knowledge, in which $\mathcal{A}_i$ is interacting with the honest prover $\mathcal{P}$ or the simulator $\mathcal{S}$, respectively. Then we have

$$\left|\Pr[\mathtt{I}_{\mathcal{A}_0,\mathcal{P}}] - \Pr[\mathtt{I}_{\mathcal{A}_0,\mathcal{S}}]\right| \leq \mathsf{negl}(\lambda), \tag{1}$$

$$\left|\Pr[\mathtt{I}_{\mathcal{A}_1,\mathcal{P}}] - \Pr[\mathtt{I}_{\mathcal{A}_1,\mathcal{S}}]\right| \leq \mathsf{negl}(\lambda), \tag{2}$$

which together implies

$$\left|\Pr[\mathtt{I}_{\mathcal{A}_0,\mathcal{P}}] - \Pr[\mathtt{I}_{\mathcal{A}_1,\mathcal{P}}]\right| \leq \mathsf{negl}(\lambda). \tag{3}$$

Note that this expression is equivalent to $\mathcal{A}$'s advantage in breaking the witness indistinguishability of Π, which is therefore also negligible. □

A.2 Proof of Lemma 1

We give a proof of Lemma 1, which is used to blind the prover's witness polynomials in PLONK in a way that reveals no information about the witness.

Proof. For all $i \in [k]$, we have

$$\tilde{f}(x_i) = f(x_i) + Z_S(x_i)\rho(x_i),$$

where the values $f(x_i), Z_S(x_i)$ are fixed and $Z_S(x_i) \neq 0$ (due to $x_i \in \mathbb{F}_p \setminus S$). Since the product of any fixed $a \in \mathbb{F}_p^*$ and random $b \in \mathbb{F}_p$ is uniform in $\mathbb{F}_p$, all we need to show is that the values $\rho(x_1), \ldots, \rho(x_k)$ are distributed independently and uniformly in $\mathbb{F}_p$, which is a well-known claim for any random degree-$(k-1)$ polynomial such as $\rho \in \mathbb{F}_p^{(\leq k-1)}[X]$. One way to see this, is by fixing any distinct $x_1, \ldots, x_k \in \mathbb{F}_p$ and observing that for any choice of $y_1, \ldots, y_k \in \mathbb{F}_p$ there is a unique degree-$(k-1)$ polynomial interpolating the points $(x_1, y_1), \ldots, (x_k, y_k)$. Formally, there are p^k distinct degree-$(k-1)$ polynomials over $\mathbb{F}_p$, which corresponds to the number of choices for $y_1, \ldots, y_k \in \mathbb{F}_p$. Furthermore, there cannot be any two distinct polynomials $f_1 \not\equiv f_2 \in \mathbb{F}_p^{(\leq k-1)}[X]$ interpolating the same set of points $(x_1, y_1), \ldots, (x_k, y_k)$, since otherwise the non-zero, degree-$(k-1)$ polynomial $f_1 - f_2$ would have at least k roots, which is a contradiction. □