Short Pairing-based Non-interactive Zero-Knowledge Arguments

Jens Groth*

University College London
j.groth@ucl.ac.uk

April 17, 2026

Abstract. We construct non-interactive zero-knowledge arguments for circuit satisfiability with perfect completeness, perfect zero knowledge and computational soundness. The non-interactive zero-knowledge arguments have sublinear size and very efficient public verification. Their size can even be reduced to a constant number of group elements if we allow the common reference string to be large. Our constructions rely on groups with pairings, and security is based on two new cryptographic assumptions; we do not use the Fiat-Shamir heuristic or random oracles.

Keywords: Sublinear-size non-interactive zero-knowledge arguments, pairing-based cryptography, power knowledge of exponent assumption, computational power Diffie-Hellman assumption.

Postscript: Published at ASIACRYPT 2010, this is a polished version. We have added a postscript to the introduction to put the work in context with later developments and an appendix on how to use the construction as a preprocessing SNARK.

1 Introduction

Zero-knowledge proofs introduced by Goldwasser, Micali and Rackoff [GMR89] are fundamental building blocks in cryptography and are used in numerous protocols. Zero-knowledge proofs enable a prover to convince a verifier of the truth of a statement without leaking any other information. The central properties are captured in the notions of completeness, soundness and zero knowledge.

Completeness: The prover can convince the verifier if the prover knows a witness testifying to the truth of the statement.

Soundness: A malicious prover cannot convince the verifier if the statement is false. We distinguish between computational soundness that protects against polynomial time cheating provers and statistical or perfect soundness where even an unbounded prover cannot convince the verifier of a false statement. We will call computationally sound proofs for *arguments*.

Zero knowledge: A malicious verifier learns nothing except that the statement is true. We distinguish between computational zero knowledge, where a polynomial time verifier learns nothing from the proof, and statistical or perfect zero knowledge, where even a verifier with unlimited resources learns nothing from the proof.

The first zero-knowledge proofs relied on interaction between the prover and the verifier. Many cryptographic tasks are carried out off-line, though; for instance, signing or encrypting messages. For these tasks it is desirable to have *non-interactive* zero-knowledge (NIZK) proofs, where there is no interaction and a proof consists of a single message from the prover

* Supported by Engineering and Physical Sciences Research Council grant number EP/G013829/1.

to the verifier. Unfortunately, only languages in BPP have NIZK proofs in the plain model without any setup [Ore87,GO94,GK96]. However, Blum, Feldman and Micali [BFM88] introduced NIZK proofs in the *common reference string model*, where both the prover and verifier have access to a common reference string generated in a trusted way. Such NIZK proofs have many applications, ranging from early chosen-ciphertext attack secure public-key cryptosystems [DDN00,Sah01] to recent advanced signature schemes [CGS07,BW06]. For this reason there has been a great deal of research into the underlying assumptions [FLS99,BCNP04,GO14], the efficiency [Dam92,DDP02,KP98,Gro10], and the security guarantees offered by NIZK proofs [DP92,Sah01,DDO+02].

NIZK proofs based on standard cryptographic assumptions used to be inefficient and not useful in practice. To get around this inefficiency, applied cryptographers have relied on the so-called Fiat-Shamir heuristic for transforming public-coin interactive zero-knowledge proofs into NIZK arguments by using a cryptographic hash function to compute the verifier's challenges. The Fiat-Shamir heuristic can give very efficient NIZK arguments that are secure in the random oracle model [BR93], where the cryptographic hash function is modeled as a random function. It is for instance possible to use the Fiat-Shamir heuristic to transform sublinear-size interactive public-coin zero-knowledge arguments [Kil92] into sublinear-size non-interactive zero-knowledge arguments [Mic00]. Unfortunately, there are several examples of protocols that are secure in the random oracle model, but do not have any secure standard-model instantiation no matter which hash function is used [CGH98,CGH04,MRH04,BBP04,Nie02]. Particularly relevant here is Goldwasser and Kalai's [GK03] demonstration of a signature scheme built from a public-coin identification scheme that is secure in the random oracle model but insecure in real life. While it is possible that the Fiat-Shamir heuristic is secure for "natural" protocols, it is worthwhile to investigate alternative approaches.

Another way to get around the inefficiency of traditional NIZK proofs is to use non-interactive designated-verifier proofs. In a designated-verifier proof, the proof is not publicly verifiable; it can only be verified by a designated verifier. Damgård, Fazio and Nicolosi [DFN06] gave an efficient linear-size non-interactive designated-verifier proof for circuit satisfiability based on an assumption related to Paillier encryption. Designated-verifier proofs suffice for some applications, for instance Cramer and Shoup's chosen-ciphertext attack secure public-key cryptosystem [CS98]. In many other cases, the lack of public verifiability is problematic, though. When there is only one designated verifier, it is for instance not possible to use them to construct advanced digital signatures, such as ring signatures and group signatures, since here non-repudiation relies on *public* verifiability.

Recent work on NIZK proofs has used bilinear groups to improve efficiency. Groth, Ostrovsky and Sahai [GOS06b,GOS06a] gave NIZK proofs for circuit satisfiability where the proof consists of $O(N)$ group elements for a circuit with N gates. Their NIZK proofs have the property that they can be set up to give either perfect soundness and computational zero knowledge, or alternatively computational soundness and perfect zero knowledge. Work by Boyen, Waters, Groth and Sahai [BW06,BW07,Gro06,GS08] has explored how to build efficient NIZK proofs that are directly applicable in bilinear groups instead of going through circuit satisfiability. In some special cases, for instance in the ring signature of Chandran, Groth and Sahai [CGS07], these techniques lead to sublinear-size NIZK proofs, but in general the number of group elements in an NIZK proof grows linearly in the size of the statement. Abe and Fehr [AF07] gave a construction based on commitments instead of encryptions, but since there is a commitment for each wire they also get linear growth in the size of the circuit.

Looking at the NP-complete problem of circuit satisfiability, the reason the NIZK proofs grow linearly in the circuit size is that they encrypt the value of each wire in the circuit. Gentry's new fully homomorphic cryptosystem [Gen09] can reduce the NIZK proof to being linear in the size of the witness: the prover encrypts the inputs to the circuit and uses the homomorphic properties of the cryptosystem to compute the output of the circuit. The prover then gives NIZK proofs for the input ciphertexts being valid and the output ciphertext containing 1. Fully homomorphic encryption only helps when the circuit has a small witness, though; if the circuit has a linear number of input wires the resulting NIZK proof will also be linear in the circuit size.

1.1 Our Contribution

Micali's CS proofs [Mic00] indicated the possibility of sublinear-size NIZK arguments, but despite more than a decade of research the Fiat-Shamir heuristic is the only known strategy for constructing sublinear-size NIZK arguments. Our goal is to introduce a new type of sublinear-size NIZK arguments where security does not rely on the random oracle model.

We construct NIZK arguments for circuit satisfiability with perfect completeness, computational soundness and perfect zero knowledge (see Section 2 for definitions). The arguments are short and very efficient to verify, though the prover uses a superlinear number of group operations. Our starting point is an argument with a constant number of group elements but a large common reference string. We then show how to reduce the size of the common reference string at the cost of increasing the size of the argument. More generally, for circuits with $N = \frac{1}{2}n^d$ gates, where d is constant, we obtain a common reference string of size $O(n^2)$ and an argument of size $O(n^{d-1})$, so both quantities can be made sublinear in the circuit size.

The soundness of our NIZK argument relies on the q-computational power Diffie-Hellman and the q-power knowledge of exponent assumptions (see Section 3). The q-CPDH assumption is an ordinary computational intractability assumption, whereas q-PKE is a knowledge-of-exponent assumption. Knowledge-of-exponent assumptions have been criticized for being unfalsifiable [Nao03], but the use of a non-standard assumption may be unavoidable since Abe and Fehr [AF07] have demonstrated that no statistical zero-knowledge NIZK argument for an NP-complete language can have a "direct black-box" security reduction to a standard cryptographic assumption unless NP $\subseteq$ P/poly.[1][2]

Table 1 gives a quick comparison to other NIZK proofs and arguments for circuit satisfiability, where N is the number of gates in the circuit, k is a security parameter, G stands for the size of a group element, M and E are the costs of respectively multiplications and exponentiations, and P is the cost of a pairing in a bilinear group (see Section 3).

Compared to other pairing-based NIZK arguments, our arguments are smaller and faster to verify. The prover uses a superlinear number of multiplications and the computational cost may grow beyond a linear number of exponentiations. The public verifiability means that the NIZK arguments are transferable though; they can be copied and distributed to many different

[1] Abe and Fehr do not rule out the existence of statistical NIZK arguments with non-adaptive soundness, where the adversary chooses the statement obliviously of the common reference string. Since the common reference string is public it is more natural to define soundness adaptively though, indeed we do not know of any practical applications of NIZK arguments with non-adaptive soundness.

[2] The very assumption that an NIZK argument is sound seems to be unfalsifiable as well, since even if an adversary outputs a false statement and a convincing NIZK argument it may be hard to verify that the statement is false. Groth, Ostrovsky and Sahai [GOS06b] circumvented this problem by defining co-soundness for languages in NP $\cap$ coNP, which is falsifiable since the adversary can produce a coNP-witness certifying that the statement is false.

	CRS size	Proof size	Prover comp.	Verifier comp.	Sound	ZK	Assumption
Groth [Gro10]	$\tilde{O}(N)$ G	$\tilde{O}(N)$ G	$\tilde{O}(N)$ E	$\tilde{O}(N)$ M	stat.	comp.	trapdoor perm.
Groth [Gro10]	$\tilde{O}(N)$ bits	$\tilde{O}(N)$ bits	$\tilde{O}(N)$ M	$\tilde{O}(N)$ M	stat.	comp.	Naccache-Stern
Gentry [Gen09]	$O(1)$ G	$\|w\|k^{O(1)}$ G	$Nk^{O(1)}$ M	$Nk^{O(1)}$ M	stat.	comp.	lattice-based
Groth-Ostrovsky-Sahai	$O(1)$ G	$O(N)$ G	$O(N)$ E	$O(N)$ P	perf.	comp.	subgroup decision or
[GOS06b,GOS06a]	$O(1)$ G	$O(N)$ G	$O(N)$ E	$O(N)$ E	comp.	perf.	decision linear
Abe-Fehr [AF07]	$O(1)$ G	$O(N)$ G	$O(N)$ E	$O(N)$ E	comp.	perf.	knowledge of expo.
Groth [Gro09]	$O(N^{\frac{1}{2}})$ G	$O(N^{\frac{1}{2}})$ G	$O(N)$ M	$O(N)$ M	comp.	perf.	random oracle
This paper	$O(N^2)$ G	$O(1)$ G	$O(N^2)$ M	$O(N)$ M	comp.	perf.	PKE and CPDH
This paper	$O(N^{\frac{2}{3}})$ G	$O(N^{\frac{2}{3}})$ G	$O(N^{\frac{4}{3}})$ M	$O(N)$ M	comp.	perf.	PKE and CPDH

Table 1. Comparison of NIZK proofs and arguments.

entities that can do their own independent verification. The prover only pays a one-time cost for computing the NIZK argument, while all verifiers enjoy the benefits of low transmission bandwidth and efficient verification.

NIZK arguments based on the Fiat-Shamir heuristic are more efficient than our NIZK arguments. These are interactive zero-knowledge arguments (and interaction seems to help) that rely on a cryptographic hash-function to compute the verifier's challenges. The security proofs for soundness rely on the assumption that the hash-function can be modeled as a source of random coins though, which seems to imply at least some type of knowledge extraction assumption that a hash-value can only be obtained if the adversary knows the input beforehand. Furthermore, the cryptographic hash-function is deterministic and therefore it is clearly false that it is a random function. In contrast, the q-PKE assumption is as far as we know true. Even though the random oracle assumption is false and even though there are examples of the Fiat-Shamir heuristic leading to insecure arguments [GK03], it may be that a particular NIZK argument based on the Fiat-Shamir heuristic is sound. However, we believe it is worthwhile to investigate alternatives to the Fiat-Shamir heuristic.

Perfect Zaps. The common reference string model assumes a trusted setup for generating common reference strings and making them available to the prover and verifier. In case no such setup is available[3] we can still get a sublinear-size 2-move publicly verifiable witness-indistinguishable argument where the verifier's first message can be reused many times, a so-called Zap [DN00], as follows: The verifier generates a common reference string. The prover verifies that the common reference string is well-formed (our common reference string is not a random bit-string, but it does have a certain structure that makes it possible to verify that it is well-formed) and can now make arbitrarily many Zaps using the verifier's initial message as the common reference string. Since our NIZK argument has perfect zero knowledge, the Zaps will be perfectly witness-indistinguishable.

1.2 Outline of Our NIZK Argument

We will construct NIZK arguments for the existence of an input to a binary circuit C making it output 1. At a loss of a constant factor, we may assume C is built entirely from NAND-gates. Furthermore, if we label the output wire a we may add a self-loop to the circuit consisting of a

[3] We remark that even if the common reference string is adversarially chosen, the sublinearity of our NIZK arguments imposes an information theoretic upper bound on how much information can be leaked.

NAND-gate $a = \neg(a \wedge b)$ forcing a to be 1. This reduces the challenge to prove that there is an assignment of truth-values to the wires that respect all the N NAND-gates in the circuit.

The NIZK argument relies on length-reducing commitments where we can commit to n values in a finite field $\mathbb{Z}_p$ using only a constant number of group elements. The commitment scheme should be homomorphic, which means that we can combine two commitments to get a new commitment containing the entry-wise sum of their values. We will also use non-interactive arguments consisting of a constant number of group elements for proving the following properties about committed values:

Entry-wise product: Commitments c, d, v contain values $a_1, \ldots, a_n, b_1, \ldots, b_n$ and $u_1, \ldots, u_n$ that satisfy $u_i = a_i b_i$ for all i.

Permutation: Commitments c, d contain values $a_1, \ldots, a_n$ and $b_1, \ldots, b_n$ that satisfy $b_i = a_{\rho(i)}$ for all i, where ρ is a publicly known permutation of n elements.

Let us sketch how homomorphic commitments combined with these two types of non-interactive arguments give us a constant-size NIZK argument for circuit satisfiability when $n = 2N$. The construction has four steps: encode the wire values in commitments, prove that they are bits, prove that they are consistent with the circuit wiring, and finally prove that the NAND constraints hold. The prover gets as a witness for the satisfiability of the circuit $a_1, \ldots, a_N$ and $b_1, \ldots, b_N$ such that a_i, b_i are the inputs to gate i and all the values are consistent with the wires and respect the NAND-gates. We will use the convention that 0 corresponds to false and 1 corresponds to true, so if u_i is the output of gate i we have $1 - u_i = a_i b_i$.

The prover makes commitments to the $2N$-tuples

$$(a_1, \ldots, a_N, b_1, \ldots, b_N) \qquad (b_1, \ldots, b_N, 0, \ldots, 0) \qquad (u_1, \ldots, u_N, 0, \ldots, 0).$$

The prover gives an entry-wise product argument with both c, d and v being the commitment to $(a_1, \ldots, a_N, b_1, \ldots, b_N)$ to show $a_i = a_i^2$ and $b_i = b_i^2$ for all i. This is the bitness step, showing that $a_1, \ldots, a_N, b_1, \ldots, b_N \in \{0, 1\}$ are appropriate truth values.

An output of one NAND-gate may be the input of other NAND-gates, which means the corresponding values $a_{i_1}, \ldots, a_{i_\ell}, b_{j_1}, \ldots, b_{j_m}$ have to have the same assignment. The prover picks a permutation ρ that contains cycles $i_1 \to i_2 \to \ldots \to i_\ell \to j_1 + N \to j_2 + N \to \ldots \to j_m + N \to i_1$ for all such sets of values that have to be consistent and gives a permutation argument on the commitment to $(a_1, \ldots, a_N, b_1, \ldots, b_N)$. This shows for each set of values corresponding to the same output wire that $a_{i_2} = a_{i_1}, \ldots, b_{j_1} = a_{i_\ell}, \ldots, b_{j_m} = b_{j_{m-1}}$ so the values $(a_1, \ldots, a_N, b_1, \ldots, b_N)$ are consistent with the wiring of the circuit.

The prover then uses additional commitments and further entry-wise product and permutation arguments to show that the other committed values $(b_1, \ldots, b_N, 0, \ldots, 0)$ and $(u_1, \ldots, u_N, 0, \ldots, 0)$ are consistent with the wiring of the circuit and the values $(a_1, \ldots, a_N, b_1, \ldots, b_N)$; we refer to Section 8 for the details.

Finally, the prover uses the entry-wise product argument to show that the entry-wise product of $(a_1, \ldots, a_N, b_1, \ldots, b_N)$ and $(b_1, \ldots, b_N, 0, \ldots, 0)$ is $(1-u_1, \ldots, 1-u_N, 0, \ldots, 0)$, so all the values respect the NAND gates. The latter commitment to $(1 - u_1, \ldots, 1 - u_N, 0, \ldots, 0)$ can be created easily from the commitment to $(u_1, \ldots, u_N, 0, \ldots, 0)$ using the homomorphic property of the commitment scheme.

This outline shows how to get a constant-size NIZK argument for circuit satisfiability, but to enable the entry-wise product arguments and the permutation arguments the common reference string has size $O(N^2)$ group elements. In Section 9 we reduce the common reference string

size by using commitments to n elements where $n < N$. Once n is smaller than $2N$, however, we need permutation arguments that span multiple commitments. Using permutation network techniques [Clo53], we build such large permutations from many smaller permutations.

The technical contribution of this paper is the construction of an appropriate commitment scheme with corresponding non-interactive entry-wise product and permutation arguments. The commitment scheme is a variant of the Pedersen commitment scheme, where the commitment key is of the form $(g, g^x, \ldots, g^{x^q})$. A commitment to $a_1, \ldots, a_q$ is a single group element computed as $g^r \prod_{i=1}^{q} (g^{x^i})^{a_i}$.

The nice thing about this commitment scheme is that the discrete logarithm is a simple polynomial $r + \sum_{i=1}^{q} a_i x^i$. When we pair two commitments with each other we get a product of two polynomials in the exponent. By taking appropriate linear combinations over products of polynomials, we can express entry-wise products and permutations as equations over the coefficients of these polynomials. The q-CPDH assumption then allows us to conclude that these coefficients are identical and therefore the committed values satisfy an entry-wise multiplication relationship or a permutation relationship to each other.

When pairing commitments (equivalent to multiplying polynomials in the exponent) there will be various cross-terms. The role of the non-interactive arguments will be to cancel out these terms. Usually, a single group element paired with g suffices to cancel out all the cross-terms, so the non-interactive arguments for entry-wise products and permutations are highly efficient themselves.

To prove that our NIZK argument is sound, we need to reason about the coefficient of these polynomials. However, a cheating prover might create a commitment without knowing an opening of it. This is where the q-PKE assumption comes in handy: the prover gives non-interactive arguments demonstrating that it "knows" the openings of the commitments. By this we mean that there is an extractor that given the same input as the prover can reconstruct the commitments together with the openings of the commitments.

1.3 Postscript: Subsequent Related Work

Terminology has changed since the first version: today it would be more natural to speak of succinct proofs rather than short proofs, and refer to the construction as a succinct non-interactive argument of knowledge (SNARK), a term that was only coined later [BCCT12]. If we had to choose the title again, a natural alternative might therefore be "Pairing-based SNARKs" or "Pairing-based zk-SNARKs," to emphasize the zero-knowledge property as well.

The performance of our SNARK depends on how compactly we can encode the constraints. Our construction uses a straightforward encoding where constant-size proofs for circuits with N gates require common reference string size and prover effort $O(N^2)$. Lipmaa [Lip12] found a beautiful connection to progression-free sets from additive combinatorics, reducing the common reference string size and prover effort to $O(N^{\log 3})$ or even lower while staying within the same general coefficient-comparison paradigm.

The SNARK in this paper proves satisfiability of an arbitrary circuit. Gennaro et al. [GGPR13] drew attention to a different task: proving satisfiability of a fixed circuit. In that setting, statements specify public inputs on some wires, while witnesses provide the remaining input values needed to make the circuit satisfiable. The advantage is that the common reference string can be preprocessed for the specific circuit. When this preprocessing is amortized over many proofs, the verifier's work can become sublinear in the circuit size. In contrast, when the

statement itself contains an arbitrary circuit, the verifier must in general at least read its description and can therefore not be sublinear. While this paper focused on the arbitrary-circuit setting, there is a twist to the story: preprocessing for a fixed circuit is trivial in our protocol, so while we did not consider this question it is already a preprocessing SNARK with sublinear verification; see Appendix B.

Gennaro et al. [GGPR13] also introduced key concepts such as quadratic span programs and quadratic arithmetic programs, and achieved linear proving cost. Technically, the crucial shift is from encoding constraints as identities between coefficients of products of polynomials to checking constraints by point evaluation. This gain in efficiency comes at the cost of a more complex common reference string, containing group elements of the form $g^{\mathrm{poly}(x,y,z,\ldots)}$ for low-degree multivariate polynomials in the exponent. In contrast, the group elements in our common reference string are of the form g^{x^i} or $g^{x^i y^j}$, i.e., with low-degree monomials in the exponent.

Later work on updatable common reference strings returned to this monomial structure. An updatable common reference string [GKM+18] is designed so that any skeptical outsider can rerandomize it before use. After rerandomizing the CRS, the new user can fully trust it. Since rerandomization updates are publicly verifiable, previous users can also trust the new CRS as much as they trusted the previous version. Groth et al. [GKM+18] observed that monomial exponents are especially well suited for this purpose: for example, a term g^{x^i} can be updated as $(g^{x^i})^{y^i} = g^{(xy)^i}$, and pairings can be used to verify that the updates are consistent throughout the common reference string. This gives our SNARK a useful combination of properties: universality, because the same setup can support many circuits; preprocessability, because it can be specialized to a fixed circuit; and updatability, because trust can be accumulated through public rerandomization. Subsequent constructions achieved smaller CRS size and lower proving costs.

2 Definitions

Let R be an efficiently computable binary relation. For pairs $(C, w) \in R$ we call C the statement and w the witness. Let L be the NP-language consisting of statements with witnesses in R. When we restrict ourselves to statements of size N, we write respectively L_N and R_N.

A non-interactive argument for a relation R consists of a common reference string generator algorithm K, a prover algorithm P and a verifier algorithm V that run in probabilistic polynomial time. The common reference string generator takes as input a security parameter k and may also take additional inputs and produces a common reference string σ. In our case, the additional input to the key generation algorithm may be a value $N \in \mathbb{N}$ specifying the size of statements we are interested in. The prover takes as input (σ, C, w) and produces an argument π. The verifier takes as input (σ, C, π) and outputs 1 if the argument is acceptable and 0 if rejecting the argument. We call (K, P, V) an argument for R if it has the completeness and soundness property described below.

Perfect completeness. Completeness captures the notion that an honest prover should be able to convince an honest verifier if the statement is true. For all adversaries $\mathcal{A}$ and $N = k^{O(1)}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k, N); (C, w) \leftarrow \mathcal{A}(\sigma); \pi \leftarrow P(\sigma, C, w) : V(\sigma, C, \pi) = 1 \text{ if } (C, w) \in R_N\right] = 1.$$

Computational soundness. Soundness captures the notion that it should be infeasible for an adversary to come up with an accepting argument for a false statement. For all non-uniform

polynomial time adversaries $\mathcal{A}$ and $N = k^{O(1)}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k, N); (C, \pi) \leftarrow \mathcal{A}(\sigma) : C \notin L \text{ and } V(\sigma, C, \pi) = 1\right] \approx 0.$$

PERFECT WITNESS-INDISTINGUISHABILITY. We say a non-interactive argument (K, P, V) is perfectly witness-indistinguishable if it is impossible to tell which witness the prover used when there are many possible witnesses. For all stateful interactive adversaries $\mathcal{A}$ and $N = k^{O(1)}$ we have

$$\begin{aligned}
&\Pr\Big[\sigma \leftarrow K(1^k, N); (C, w_0, w_1) \leftarrow \mathcal{A}(\sigma); \\
&\qquad \pi \leftarrow P(\sigma, C, w_0) : (C, w_0), (C, w_1) \in R_N \text{ and } \mathcal{A}(\pi) = 1\Big] \\
&= \Pr\Big[\sigma \leftarrow K(1^k, N); (C, w_0, w_1) \leftarrow \mathcal{A}(\sigma); \\
&\qquad \pi \leftarrow P(\sigma, C, w_1) : (C, w_0), (C, w_1) \in R_N \text{ and } \mathcal{A}(\pi) = 1\Big].
\end{aligned}$$

PERFECT ZERO KNOWLEDGE. An argument is zero knowledge if it does not leak any information besides the truth of the statement. We say a non-interactive argument (K, P, V) has perfect zero knowledge if there exists a polynomial time simulator $S = (S_1, S_2)$ with the following zero-knowledge property. S_1 outputs a simulated common reference string and a simulation trapdoor. S_2 takes the common reference string, the simulation trapdoor and a statement as input and produces a simulated argument. For all stateful interactive adversaries $\mathcal{A}$ and $N = k^{O(1)}$ we require

$$\begin{aligned}
&\Pr\left[\sigma \leftarrow K(1^k, N); (C, w) \leftarrow \mathcal{A}(\sigma); \pi \leftarrow P(\sigma, C, w) : (C, w) \in R_N \text{ and } \mathcal{A}(\pi) = 1\right] \\
&= \Pr\left[(\sigma, \tau) \leftarrow S_1(1^k, N); (C, w) \leftarrow \mathcal{A}(\sigma); \pi \leftarrow S_2(\sigma, \tau, C) : (C, w) \in R_N \text{ and } \mathcal{A}(\pi) = 1\right].
\end{aligned}$$

3 Bilinear Groups

NOTATION. Given two functions $f, g : \mathbb{N} \to [0, 1]$ we write $f(k) \approx g(k)$ when $|f(k) - g(k)| = O(k^{-c})$ for every constant $c > 0$. We say that f is *negligible* when $f(k) \approx 0$ and that it is *overwhelming* when $f(k) \approx 1$.

We write $y = A(x; r)$ when the algorithm A on input x and randomness r outputs y. We write $y \leftarrow A(x)$ for the process of picking randomness r at random and setting $y = A(x; r)$. We also write $y \leftarrow S$ for sampling y uniformly at random from the set S. We will assume it is possible to sample uniformly at random from sets such as $\mathbb{Z}_p$. We define $[n]$ to be the set $\{1, 2, \ldots, n\}$.

BILINEAR GROUPS. Let $\mathcal{G}$ take a security parameter k written in unary as input and output a description of a bilinear group $(p, G, G_T, e) \leftarrow \mathcal{G}(1^k)$ such that

1. p is a k-bit prime.
2. G, G_T are cyclic groups of order p.
3. $e : G \times G \to G_T$ is a bilinear map (pairing) such that for every generator g of G and all $a, b \in \mathbb{Z}_p$ we have $e(g^a, g^b) = e(g, g)^{ab}$.
4. If g generates G then $e(g, g)$ generates G_T.

5. Membership in G, G_T can be efficiently decided, group operations and the pairing e are efficiently computable, generators are efficiently sampleable, and the descriptions of the groups and group elements each have size $O(k)$ bits.

The security of our NIZK arguments is based on two new assumptions, which we call the q-power knowledge of exponent assumption and the q-computational power Diffie-Hellman assumption.

THE q-POWER KNOWLEDGE OF EXPONENT ASSUMPTION. The knowledge of exponent (KEA) assumption introduced by Damgård [Dam91] says that given g, g^α it is infeasible to create $c, \hat{c}$ such that $\hat{c} = c^\alpha$ without knowing a such that $c = g^a$ and $\hat{c} = (g^\alpha)^a$. Bellare and Palacio [BP04] extended this to the KEA3 assumption, which says that given $g, g^x, g^\alpha, g^{\alpha x}$ it is infeasible to create $c, \hat{c}$ such that $\hat{c} = c^\alpha$ without knowing a_0, a_1 such that $c = g^{a_0}(g^x)^{a_1}$ and $\hat{c} = (g^\alpha)^{a_0}(g^{\alpha x})^{a_1}$. This assumption has also been used in bilinear groups by Abe and Fehr [AF07], who called it the extended knowledge of exponent assumption.

The q-power knowledge of exponent assumption is a generalization of KEA and KEA3. It says that given $(g, g^x, \ldots, g^{x^q}, g^\alpha, g^{\alpha x}, \ldots, g^{\alpha x^q})$ it is infeasible to create $c, \hat{c}$ such that $\hat{c} = c^\alpha$ without knowing $a_0, \ldots, a_q$ such that $c = \prod_{i=0}^{q}(g^{x^i})^{a_i}$ and $\hat{c} = \prod_{i=0}^{q}(g^{\alpha x^i})^{a_i}$.

We will now give the formal definition of the q-power knowledge of exponent assumption. Following Abe and Fehr [AF07] we write $(y; z) \leftarrow (\mathcal{A} \parallel \mathcal{X}_\mathcal{A})(x)$ when $\mathcal{A}$ on input x outputs y and $\mathcal{X}_\mathcal{A}$ on the same input, including the random tape of $\mathcal{A}$, outputs z.

Definition 1 (q-PKE). *The $q(k)$-power knowledge of exponent assumption holds for $\mathcal{G}$ if for every non-uniform probabilistic polynomial time adversary $\mathcal{A}$ there exists a non-uniform probabilistic polynomial time extractor $\mathcal{X}_\mathcal{A}$ such that*

$$\begin{aligned}\Pr\Big[&(p, G, G_T, e) \leftarrow \mathcal{G}(1^k)\ ;\ g \leftarrow G \setminus \{1\}\ ;\ \alpha, x \leftarrow \mathbb{Z}_p^*\ ;\\ &\sigma = (p, G, G_T, e, g, g^x, \ldots, g^{x^q}, g^\alpha, g^{\alpha x}, \ldots, g^{\alpha x^q})\ ;\\ &(c, \hat{c}\ ;\ a_0, \ldots, a_q) \leftarrow (\mathcal{A} \parallel \mathcal{X}_\mathcal{A})(\sigma)\ :\\ &\hat{c} = c^\alpha \wedge c \neq \prod_{i=0}^{q} g^{a_i x^i}\Big]\ \approx\ 0.\end{aligned}$$

We give a heuristic argument for believing in the q-PKE assumption by showing that it holds in the generic group model in Appendix A.

THE q-COMPUTATIONAL POWER DIFFIE-HELLMAN ASSUMPTION. The computational Diffie-Hellman (CDH) assumption says that given g, g^β, g^x it is infeasible to compute $g^{\beta x}$. The q-computational power Diffie-Hellman assumption is a generalization of the CDH assumption which says that, given $(g, g^x, \ldots, g^{x^q}, g^\beta, g^{\beta x}, \ldots, g^{\beta x^q})$ except for one missing element $g^{\beta x^j}$, it is hard to compute the missing element.

Definition 2 (q-CPDH). *The $q(k)$-computational power Diffie-Hellman assumption holds for $\mathcal{G}$ if for all $j \in \{0, \ldots, q\}$ and all non-uniform probabilistic polynomial time adversaries $\mathcal{A}$ we have*

$$\begin{aligned}\Pr\Big[&(p, G, G_T, e) \leftarrow \mathcal{G}(1^k)\ ;\ g \leftarrow G \setminus \{1\}\ ;\ \beta, x \leftarrow \mathbb{Z}_p^*\ ;\ y \leftarrow \mathcal{A}(p, G, G_T, e, g, g^x, \ldots, g^{x^q},\\ &g^\beta, g^{\beta x}, \ldots, g^{\beta x^{j-1}}, g^{\beta x^{j+1}}, \ldots, g^{\beta x^q})\ :\ y = g^{\beta x^j}\Big]\ \approx\ 0.\end{aligned}$$

We give a heuristic argument for believing in the q-CPDH assumption by showing that it holds in the generic group model in Appendix A.

4 Knowledge Commitment

We will use a variant of the Pedersen commitment scheme in our NIZK proof where we commit to $a_1, \ldots, a_q$ as $c = g^r \prod_{i \in [q]} g_i^{a_i}$. In the security proof of our NIZK argument for circuit satisfiability we will need to extract the committed values $a_1, \ldots, a_q$, but the commitment scheme itself is perfectly hiding and does not reveal them. For this reason, we require the prover to create a related commitment $\hat{c} = \hat{g}^r \prod_{i \in [q]} \hat{g}_i^{a_i}$ and rely on the q-PKE assumption to extract the committed values. We call $(c, \hat{c})$ a knowledge commitment, since the prover cannot make a valid commitment without "knowing" the committed values.

Key generation: Pick $gk = (p, G, G_T, e) \leftarrow \mathcal{G}(1^k) \;\; g \leftarrow G \setminus \{1\} \; ; \; x, \alpha \leftarrow \mathbb{Z}_p^*$. The commitment key is $ck = (gk, g, g_1, \ldots, g_q, \hat{g}, \hat{g}_1 \ldots, \hat{g}_q) = (gk, g, g^x, \ldots, g^{x^q}, g^\alpha, g^{\alpha x}, \ldots, g^{\alpha x^q})$ and the trapdoor key is $tk = x$.

Commitment: To commit to $a_1, \ldots, a_q$ pick $r \leftarrow \mathbb{Z}_p$ and compute the knowledge commitment $(c, \hat{c})$ as

$$c = g^r \prod_{i \in [q]} g_i^{a_i} \qquad \hat{c} = \hat{g}^r \prod_{i \in [q]} \hat{g}_i^{a_i}.$$

Given $(c, \hat{c}) \in G^2$ we can verify that it is well-formed by checking $e(g, \hat{c}) = e(c, \hat{g})$.

Trapdoor commitment: To make a trapdoor commitment sample trapdoor randomness $t \leftarrow \mathbb{Z}_p$ and compute the knowledge commitment $(c, \hat{c})$ as $c = g^t \; ; \; \hat{c} = \hat{g}^t$.

Trapdoor opening: The trapdoor opening algorithm on messages $a_1, \ldots, a_q \in \mathbb{Z}_p$ returns the randomizer $r = t - \sum_{i \in [q]} a_i x^i$. The trapdoor opening satisfies $c = g^r \prod_{i \in [q]} g_i^{a_i}$ and $\hat{c} = \hat{g}^r \prod_{i \in [q]} \hat{g}_i^{a_i}$.

The commitment scheme has properties similar to those of standard Pedersen commitments, as the following theorem shows.

Theorem 1. *The commitment scheme is perfectly trapdoor and therefore perfectly hiding. Assuming the q-CPDH assumption holds, it is computationally binding. Assuming the q-PKE assumption holds, for any non-uniform probabilistic polynomial time committer $\mathcal{A}$ there exists a non-uniform probabilistic polynomial time extractor $\mathcal{X}_\mathcal{A}$ that computes the contents of the commitment given the input of $\mathcal{A}$, including any random coins.*

Proof. To see that the commitment is perfectly trapdoor and perfectly hiding, consider an arbitrary q-tuple of messages $(a_1, \ldots, a_q)$. Since g is a generator of G, both the real commitment algorithm and the trapdoor commitment algorithm generate a random group element $c \in G$ and the other part is given by $\hat{c} = c^\alpha$. Given c and $a_1, \ldots, a_q$ the randomizer is determined uniquely, so real commitments and their openings have the same distribution as trapdoor commitments and trapdoor openings.

To prove that the commitment scheme is binding, suppose a non-uniform probabilistic polynomial time adversary creates two openings $(r, a_1, \ldots, a_q)$ and $(s, b_1, \ldots, b_q)$ of the same commitment, i.e., $g^r \prod_{i \in [q]} g_i^{a_i} = g^s \prod_{i \in [q]} g_i^{b_i}$. By the homomorphic property of the commitment scheme we have $g^{r-s} \prod_{i \in [q]} g_i^{a_i - b_i} = 1$, which implies $r - s + \sum_{i \in [q]} (a_i - b_i) x^i = 0$. By Lemma 1 there is negligible probability of finding a non-trivial linear combination of $1, x, \ldots, x^q$, so except with negligible probability we have $r = s, a_1 = b_1, \ldots, a_q = b_q$.

The existence of a non-uniform probabilistic polynomial time knowledge extractor $\mathcal{X}_\mathcal{A}$ that extracts the contents of the knowledge commitment made by a non-uniform probabilistic polynomial time committer $\mathcal{A}$ follows directly from the q-PKE assumption. □

4.1 Restriction Argument

Consider a subset $S \subset [q]$ and a commitment c. We will need an argument for the opening $r, a_1, \ldots, a_q$ being such that the indices of non-zero values are restricted to S. In other words, we need an argument for the commitment being of the form $c = g^r \prod_{i \in S} g_i^{a_i}$. The argument will take the form $\pi = h^r \prod_{i \in S} h_i^{a_i}$, which intuitively corresponds to an additional argument of knowledge with respect to a small base $(h, \{h_i\}_{i \in S})$.

Setup: $gk \leftarrow \mathcal{G}(1^k)$; $ck \leftarrow K_{\text{commit}}(gk)$.
Common reference string: Given (ck, S) as input pick at random $\beta \leftarrow \mathbb{Z}_p^*$ and compute the common reference string as $\sigma = (h, \{h_i\}_{i \in S}) = (g^\beta, \{g_i^\beta\}_{i \in S})$.
Statement: A valid knowledge commitment $(c, \hat{c})$.
Prover's witness: Opening $r, \{a_i\}_{i \in S}$ so $c = g^r \prod_{i \in S} g_i^{a_i}$ and $\hat{c} = \hat{g}^r \prod_{i \in S} \hat{g}_i^{a_i}$.
Argument: Compute the argument as $\pi = h^r \prod_{i \in S} h_i^{a_i}$.
Verification: Output 1 if and only if $e(c, h) = e(g, \pi)$.

Theorem 2. *The restriction argument is perfectly complete and perfectly witness-indistinguishable. Assume the q-CPDH assumption holds. Then every non-uniform probabilistic polynomial time adversary has negligible probability of outputting an opening $(r, a_1, \ldots, a_q)$ and an acceptable restriction argument π for the corresponding commitment such that $a_i \neq 0$ for some $i \notin S$.*

Observe that we phrase the soundness of the restriction argument as the inability to find an opening of the commitment that violates the restriction. Since the commitment scheme is perfectly hiding, we cannot exclude the existence of openings that violate the restriction. However, if the commitment is a knowledge commitment (Theorem 1), then the opening we extract from the committer must respect the restriction.

Proof. Perfect completeness follows from the fact that an honestly generated proof satisfies $\pi = c^\beta$, which implies that the verification succeeds. Since g, h are generators of G there is for any commitment c a unique acceptable argument π. Since all valid witnesses result in this unique acceptable argument, we have perfect witness-indistinguishability.

It remains to argue that there is negligible probability of producing an opening $(r, a_1, \ldots, a_q)$ of a commitment c together with an acceptable proof π, where for some $i \notin S$ we have $a_i \neq 0$. Assume for contradiction that $\mathcal{A}$ is a non-uniform probabilistic polynomial time adversary with $\epsilon(k)$ chance of breaking this notion of soundness. We will use it to construct a non-uniform probabilistic polynomial time algorithm $\mathcal{B}$ that breaks the q-CPDH assumption with probability $\epsilon(k)/q$ when $q = \text{poly}(k)$.

Let us pick $j \in [q] \setminus S$ at random and give the q-CPDH challenge $(p, G, G_T, e, g, g^x, \ldots, g^{x^q}, g^\beta, g^{\beta x}, \ldots, g^{\beta x^{j-1}}, g^{\beta x^{j+1}}, \ldots, g^{\beta x^q})$ to $\mathcal{B}$. Now $\mathcal{B}$ picks at random $\alpha \leftarrow \mathbb{Z}_p^*$ and hands $(p, G, G_T, e, g, g^x, \ldots, g^{x^q}, g^\alpha, g^{\alpha x}, \ldots, g^{\alpha x^q}, g^\beta, \{g^{\beta x^i}\}_{i \in S})$ to $\mathcal{A}$. This looks like a normal commitment key and common reference string, so with probability $\epsilon(k)$ the adversary $\mathcal{A}$ returns an opening $(r, a_1, \ldots, a_q)$ and an accepting argument π, with at least one $i \notin S$ for which $a_i \neq 0$. We chose j at random so there is at least $1/q$ chance for $a_j \neq 0$. Since the argument is accepting we have $e(c, h) = e(g, \pi)$, which means $e(g^r \prod_{i \in [q]} g^{a_i x^i}, g^\beta) = e(g, \pi)$. By the bilinearity of e this implies $\pi = g^{r\beta} \prod_{i \in [q]} g^{a_i \beta x^i}$, which in turn means $g^{-a_j \beta x^j} = \pi^{-1} g^{r\beta} \prod_{i \in [q] \setminus \{j\}} g^{a_i \beta x^i}$. With $a_j \neq 0$ we get $g^{\beta x^j} = (\pi^{-1} (g^\beta)^r \prod_{i \in [q] \setminus \{j\}} (g^{\beta x^i})^{a_i})^{-1/a_j}$, which breaks the q-CPDH challenge. □

5 Common Reference String

We will now describe how to generate the common reference string for our NIZK argument. The common reference string will consist of a knowledge commitment key ck for $q = n^2 + 3n - 2$ values together with three common reference strings for restriction to the sets

$$\tilde{S} = \{1, 2, \ldots, n\} \qquad \bar{S} = \{(n+1), 2(n+1), \ldots, n(n+1)\} \qquad \dot{S} = \{\ell \in [q] \mid \ell \neq 0 \bmod n+2\}.$$

The zero-knowledge simulation of the argument will use the same type of common reference string, and the simulation trapdoor for our NIZK argument will be the trapdoor for the knowledge commitment.

Common Reference String Generation:

On input 1^k and n do

1. Generate a bilinear group $(p, G, G_T, e) \leftarrow \mathcal{G}(1^k)$ and set $gk = (p, G, G_T, e)$.
2. Pick $g \leftarrow G \setminus \{1\}$; $x, \alpha \leftarrow \mathbb{Z}_p^*$ and compute

$$ck = (gk, g, g_1, \ldots, g_q, \hat{g}, \hat{g}_1, \ldots, \hat{g}_q) = (gk, g, g^x, \ldots, g^{x^{n^2+3n-2}}, g^\alpha, g^{\alpha x}, \ldots, g^{\alpha x^{n^2+3n-2}}).$$

3. Generate $\tilde{\sigma} \leftarrow K_{\text{restrict}}(ck, \tilde{S})$ where $\tilde{S} = \{1, 2, \ldots, n\}$.
4. Generate $\bar{\sigma} \leftarrow K_{\text{restrict}}(ck, \bar{S})$ where $\bar{S} = \{(n+1), 2(n+1), \ldots, n(n+1)\}$.
5. Generate $\dot{\sigma} \leftarrow K_{\text{restrict}}(ck, \dot{S})$ where $\dot{S} = \{\ell \in [q] \mid \ell \neq 0 \bmod n+2\}$.

The common reference string is $\sigma = (ck, \tilde{\sigma}, \bar{\sigma}, \dot{\sigma})$. The simulation trapdoor is $tk = x$.

Given a common reference string, it is hard to find a non-trivial linear combination of $1, x, \ldots, x^q$, since such a relation would let us recover x by polynomial factorization over $\mathbb{Z}_p[X]$.

Lemma 1. *If the q-CPDH assumption holds for $\mathcal{G}$ with $q = n^2+3n-2$, a non-uniform probabilistic polynomial time adversary has negligible chance of finding a non-trivial linear combination $(a_0, \ldots, a_q)$ such that $\sum_{i=0}^q a_i x^i = 0$ given a random common reference string σ.*

Proof. Suppose $\mathcal{A}$ is a non-uniform probabilistic polynomial time algorithm that, given a common reference string, has probability $\epsilon(k)$ of finding $(a_0, \ldots, a_q)$ such that $\sum_{i=0}^q a_i x^i = 0$. We will construct an adversary $\mathcal{B}$ for the q-CPDH assumption with success probability $\epsilon(k)$. On a CPDH challenge $(p, G, G_T, e, g, \ldots, g^{x^q}, g^\beta, g^{\beta x}, \ldots, g^{\beta x^{j-1}}, g^{\beta x^{j+1}}, \ldots, g^{\beta x^q})$, $\mathcal{B}$ ignores the latter half and uses $\mathcal{A}$ to compute x given $(p, G, G_T, e, g, g_1, \ldots, g_q) = (p, G, G_T, e, g, g^x, \ldots, g^{x^q})$. Once we have x, it is easy to compute the solution to the q-CPDH problem as $y = g^{\beta x^j} = (g^\beta)^{x^j}$.

$\mathcal{B}$ picks at random $\hat{\alpha}, \tilde{\beta}, \bar{\beta}, \dot{\beta} \leftarrow \mathbb{Z}_p^*$ and computes a common reference string as follows

$$\hat{g} = g^{\hat{\alpha}} \qquad \tilde{h} = g^{\tilde{\beta}} \qquad \bar{h} = g^{\bar{\beta}} \qquad \dot{h} = g^{\dot{\beta}}$$

$$\forall i \in [q] : \hat{g}_i = g_i^{\hat{\alpha}} \qquad \forall i \in \tilde{S} : \tilde{h}_i = g_i^{\tilde{\beta}} \qquad \forall i \in \bar{S} : \bar{h}_i = g_i^{\bar{\beta}} \qquad \forall i \in \dot{S} : \dot{h}_i = g_i^{\dot{\beta}}$$

$\mathcal{B}$ gives the common reference string to $\mathcal{A}$, and since it has the same distribution as a real common reference string, $\mathcal{A}$ returns a non-trivial linear combination $(a_0, \ldots, a_q)$ such that $\sum_{i=0}^q a_i x^i = 0$ with probability $\epsilon(k)$. Using a polynomial factorization algorithm for $\mathbb{Z}_p[X]$, we can efficiently find the up to q roots of the polynomial. It is then easy to try each of them until we find the one satisfying $g_1 = g^x$. □

Verifying the common reference string. The common reference string described above has a particular mathematical structure and we do not know of an extraction procedure that can generate it from a public string of *random* bits. However, provided we can verify that (p, G, G_T, e) does describe a bilinear group, it is also possible to verify that σ is a well-formed common reference string. First, we check that all group elements in σ are non-trivial. This demonstrates that the secret exponents $x, \alpha, \tilde{\beta}, \bar{\beta}, \dot{\beta}$ are non-zero. Next, we use the pairing operation to verify the structure of the common reference string. We check

$$\forall i \in [q-1] : e(g, g_{i+1}) = e(g_1, g_i) \qquad \forall i \in [q] : e(g, \hat{g}_i) = e(\hat{g}, g_i)$$

$$\forall i \in \tilde{S} : e(g, \tilde{h}_i) = e(\tilde{h}, g_i) \qquad \forall i \in \bar{S} : e(g, \bar{h}_i) = e(\bar{h}, g_i) \qquad \forall i \in \dot{S} : e(g, \dot{h}_i) = e(\dot{h}, g_i).$$

By verifying the common reference string, the prover can be assured that the argument is perfectly witness-indistinguishable. This means that even if the common reference string is generated by an untrusted source such as the verifier, we obtain a 2-move argument with perfect witness-indistinguishability, also known as a Zap [DN00]. In the first move, the verifier sends a common reference string, and the prover can then give many publicly verifiable arguments, i.e. second moves, for different statements using the same common reference string.

6 Product Argument

Consider three commitments

$$c = g^r \prod_{i \in [n]} g_i^{a_i} \qquad d = g^s \prod_{j \in [n]} g_{j(n+1)}^{b_j} \qquad v = g^t \prod_{i \in [n]} g_i^{u_i} \qquad \forall i \in [n] : u_i = a_i b_i.$$

With the corresponding restriction arguments, $\hat{c}, \tilde{c}, \hat{d}, \bar{d}, \hat{v}, \tilde{v}$ we can assume the committer knows openings to values $a_1, \ldots, a_n, b_1, \ldots, b_n$ and $u_1, \ldots, u_n$. We will give an argument $(\pi, \hat{\pi}, \dot{\pi})$ consisting of three group elements for the committed values satisfying $u_1 = a_1 b_1, \ldots, u_n = a_n b_n$.

In order to explain the intuition behind the argument, let us consider the following toy example $c = \prod_{i \in [n]} g_i^{a_i}$ and $d = \prod_{j \in [n]} g_{j(n+1)}^{b_j}$, where we want to show $a_1 b_1 = 0, \ldots, a_n b_n = 0$. The discrete logarithms of the two commitments are $\sum_{i \in [n]} a_i x^i$ and $\sum_{j \in [n]} b_j x^{j(n+1)}$, and the discrete logarithm of $e(c, d)$ is

$$\left(\sum_{i \in [n]} a_i x^i \right) \cdot \left(\sum_{j \in [n]} b_j x^{j(n+1)} \right) = \sum_{i \in [n]} \sum_{j \in [n]} a_i b_j x^{j(n+1)+i} = \sum_{i \in [n]} a_i b_i x^{i(n+2)} + \sum_{i \in [n]} \sum_{j \in [n] \setminus \{i\}} a_i b_j x^{j(n+1)+i}.$$

In the final sum, the left term contains the coefficients $a_1 b_1, \ldots, a_n b_n$ that are supposed to be 0; however, the right term complicates matters. The argument π will be constructed so that it cancels out the latter term.

Notice that the left term isolates the coefficients of $x^{n+2}, \ldots, x^{n(n+2)}$, while the right term does not contain any such coefficients. By giving an appropriate restriction argument, the prover can guarantee that it only cancels out the right term without interfering with the left term containing $x^{n+2}, \ldots, x^{n(n+2)}$. The prover computes $\pi = \prod_{i \in [n]} \prod_{j \in [n] \setminus \{i\}} g_{j(n+1)+i}^{a_i b_j}$ and gives corresponding $\hat{\pi}, \dot{\pi}$ values demonstrating that it knows an opening $(z, \{z_\ell\}_{\ell \in \dot{S}})$ of π restricted to $\dot{S}$. The verifier checks

$$e(c, d) = e(g, \pi).$$

Let us now argue soundness: given $\pi, \hat{\pi}, \dot{\pi}$ such that $e(c,d) = e(g,\pi)$, the verifier can be assured that $a_1 b_1 = 0, \ldots, a_n b_n = 0$. Taking discrete logarithms, the verification equation tells us that

$$\sum_{i\in[n]} a_i b_i x^{i(n+2)} + \sum_{i\in[n]} \sum_{j\in[n]\setminus\{i\}} a_i b_j x^{j(n+1)+i} = z + \sum_{\ell\in\dot{S}} z_\ell x^\ell.$$

Recall that $\dot{S} = \{\ell \in [n^2+3n-2] \mid \ell \neq 0 \bmod n+2\}$, so the argument π will not contain any coefficients of the form $x^{n+2}, \ldots, x^{n(n+2)}$. This means the coefficients of $x^{n+2}, \ldots, x^{n(n+2)}$ are $a_1 b_1, \ldots, a_n b_n$. If there is an i such that $a_i b_i \neq 0$, then we have a non-trivial polynomial equation in x. By Lemma 1, this would allow us to recover x, thereby breaking the q-CPDH assumption.

In the general case we want to give an argument for $a_i b_i = u_i$ instead of just $a_i b_i = 0$. However, if we evaluate $e(v, \prod_{j\in[n]} g_{j(n+1)})$, we can view the latter as a commitment to $(1,1,\ldots,1)$, and we get the products $u_1 \cdot 1, \ldots, u_n \cdot 1$ as coefficients of $x^{n+2}, \ldots, x^{n(n+2)}$. If $u_1 = a_1 b_1, \ldots, u_n = a_n b_n$, the two pairings $e(c,d)$ and $e(v, \prod_{j\in[n]} g_{j(n+1)})$ therefore have the same coefficients of $x^{n+2}, \ldots, x^{n(n+2)}$; otherwise the coefficients are different. As in the toy example above, we may choose π so that it cancels out all the other terms. Due to the restriction to $\dot{S}$, the argument will not have any $x^{n+2}, \ldots, x^{n(n+2)}$ terms, and we therefore get soundness. In the general case, the commitments also have randomizers, and we choose π so that it cancels out these terms as well. We give the full argument below.

Statement: Commitments $c, d, v \in G$.
Prover's witness: Openings $r, a_1, \ldots, a_n$ and $s, b_1, \ldots, b_n$ and $t, u_1, \ldots, u_n$ so

$$c = g^r \prod_{i\in[n]} g_i^{a_i} \quad \text{and} \quad d = g^s \prod_{j\in[n]} g_{j(n+1)}^{b_j}$$

$$v = g^t \prod_{i\in[n]} g_i^{u_i} \quad \text{and} \quad \forall i \in [n] : u_i = a_i b_i.$$

Argument: Compute the argument $(\pi, \hat{\pi}, \dot{\pi})$ as

$$\begin{aligned}
\pi &= g^{rs} \prod_{i\in[n]} g_i^{a_i s} \prod_{j\in[n]} g_{j(n+1)}^{b_j r - t} \\
&\quad \cdot \prod_{i\in[n]} \prod_{j\in[n]\setminus\{i\}} g_{j(n+1)+i}^{a_i b_j - u_i} \\
\hat{\pi} &= \hat{g}^{rs} \prod_{i\in[n]} \hat{g}_i^{a_i s} \prod_{j\in[n]} \hat{g}_{j(n+1)}^{b_j r - t} \\
&\quad \cdot \prod_{i\in[n]} \prod_{j\in[n]\setminus\{i\}} \hat{g}_{j(n+1)+i}^{a_i b_j - u_i} \\
\dot{\pi} &= \dot{h}^{rs} \prod_{i\in[n]} \dot{h}_i^{a_i s} \prod_{j\in[n]} \dot{h}_{j(n+1)}^{b_j r - t} \\
&\quad \cdot \prod_{i\in[n]} \prod_{j\in[n]\setminus\{i\}} \dot{h}_{j(n+1)+i}^{a_i b_j - u_i}
\end{aligned}$$

Verification: Output 1 if and only if

$$e(g,\hat{\pi}) = e(\pi,\hat{g}) \qquad \wedge \qquad e(g,\dot{\pi}) = e(\pi,\dot{h}) \qquad \wedge \qquad e(c,d) = e(v, \prod_{j\in[n]} g_{j(n+1)}) e(g,\pi).$$

Theorem 3. *The product argument has perfect completeness and perfect witness-indistinguishability. Assume the q-CPDH assumption holds. Then a non-uniform probabilistic polynomial time adversary has negligible chance of outputting commitments (c, d, v) and an accepting argument π with corresponding openings of the commitments and the argument such that for some $i \in [n]$ we have $a_i b_i \neq u_i$.*

Proof. Straightforward computation shows that the argument is perfectly complete. To see that the argument is perfectly witness-indistinguishable, observe that given c, d, v there is exactly one acceptable argument π satisfying the verification equation. Given π, the other parts $\hat{\pi}$ and $\dot{\pi}$ are also determined uniquely by the verification equations because $\hat{g}$ and $\dot{h}$ are generators of G. Hence any valid opening of c, d, v with $a_1 b_1 = u_1, \ldots, a_n b_n = u_n$ yields the same argument $(\pi, \hat{\pi}, \dot{\pi})$.

We will now prove that the argument satisfies the soundness condition given in the theorem. Suppose there is a non-uniform probabilistic polynomial time adversary $\mathcal{A}$ that has more than negligible chance of finding openings $r, a_1, \ldots, a_n, s, b_1, \ldots, b_n, t, u_1, \ldots, u_n$ and $z, \{z_\ell\}_{\ell \in \dot{S}}$ such that

$$c = g^r \prod_{i \in [n]} g_i^{a_i} \qquad d = g^s \prod_{j \in [n]} g_{j(n+1)}^{b_j} \qquad v = g^t \prod_{i \in [n]} g_i^{u_i} \qquad \pi = g^z \prod_{\ell \in \dot{S}} g_\ell^{z_\ell},$$

$$\exists i \in [n] : a_i b_i \neq u_i \qquad \wedge \qquad e(c, d) = e(v, \prod_{j \in [n]} g_{j(n+1)}) e(g, \pi).$$

Then we have

$$\begin{aligned}
\log_{e(g,g)} e(c, d) &= (r + \sum_{i \in [n]} a_i x^i)(s + \sum_{j \in [n]} b_j x^{j(n+1)}) \\
&= rs + s \sum_{i \in [n]} a_i x^i + r \sum_{j \in [n]} b_j x^{j(n+1)} + \sum_{i \in [n]} \sum_{j \in [n]} a_i b_j x^{j(n+1)+i} \\
\log_{e(g,g)} e(v, \prod_{j \in [n]} g_{j(n+1)}) &= (t + \sum_{i \in [n]} u_i x^i)(\sum_{j \in [n]} x^{j(n+1)}) = t \sum_{j \in [n]} x^{j(n+1)} + \sum_{i \in [n]} \sum_{j \in [n]} u_i x^{j(n+1)+i} \\
\log_{e(g,g)} e(g, \pi) &= z + \sum_{\ell \in \dot{S}} z_\ell x^\ell
\end{aligned}$$

Since $e(c, d) = e(v, \prod_{j \in [n]} g_{j(n+1)}) e(g, \pi)$ the discrete logarithms satisfy

$$\begin{aligned}
&rs + s \sum_{i \in [n]} a_i x^i + r \sum_{j \in [n]} b_j x^{j(n+1)} + \sum_{i \in [n]} \sum_{j \in [n]} a_i b_j x^{j(n+1)+i} \\
&= t \sum_{j \in [n]} x^{j(n+1)} + \sum_{i \in [n]} \sum_{j \in [n]} u_i x^{j(n+1)+i} + z + \sum_{\ell \in \dot{S}} z_\ell x^\ell.
\end{aligned}$$

Recall that $\dot{S}$ does not contain $n+2, 2(n+2), \ldots, n(n+2)$. Moreover, among the terms $x^{j(n+1)+i}$ the exponent $i(n+2)$ occurs only when $j = i$. We therefore see that for all $i \in [n]$ the coefficients of $x^{i(n+2)}$ on each side of the equality are respectively $a_i b_i$ and u_i. If $a_i b_i \neq u_i$ for some $i \in [n]$ this gives us a non-trivial linear combination of $1, x, \ldots, x^{n^2+3n-2}$ and by Lemma 1 a breach of the q-CPDH assumption. □

The product argument has two commitments with restriction to $\tilde{S}$ and one commitment restricted to $\bar{S}$. It is quite easy to translate commitments back and forth between $\tilde{S}$ and $\bar{S}$,

though. If we have two commitments v and d restricted to respectively $\tilde{S}$ and $\bar{S}$, we can give a product argument for the values in v being the product of the values in $c = \prod_{i\in[n]} g_i$ and d. Since c is a commitment to $(1,\ldots,1)$ this proves that v and d contain the same values.

The product argument makes it possible to prove that the committed values in a commitment c are bits. If we give a product argument for c containing the entry-wise product of the values in c and d, where d contains the same values as c, then we have that the values satisfy $a_i = a_i^2$, which implies $a_i \in \{0,1\}$.

7 Permutation Argument

Consider two commitments and a permutation

$$c = g^r \prod_{i\in[n]} g_i^{a_i} \qquad d = g^s \prod_{i\in[n]} g_i^{b_i} \qquad \rho \in S_n \qquad \forall i \in [n] :\ b_i = a_{\rho(i)}.$$

We will now give an argument for the committed values satisfying $b_i = a_{\rho(i)}$, where $\rho \in S_n$ is a publicly known permutation.

The idea behind the permutation argument is to show

$$\sum_{i\in[n]} a_i x^{i(n+2)} = \sum_{i\in[n]} b_i x^{\rho(i)(n+2)}.$$

By Lemma 1, this implies $b_i = a_{\rho(i)}$ for all $i \in [n]$.

To get the desired linear combination we compute $e(c, \prod_{j\in[n]} g_{j(n+1)})$ and $e(d, \prod_{j\in[n]} g_{\rho(j)(n+2)-j})$. They have discrete logarithms

$$\begin{aligned}(r + \sum_{i\in[n]} a_i x^i) \sum_{j\in[n]} x^{j(n+1)} &= r \sum_{j\in[n]} x^{j(n+1)} + \sum_{i\in[n]} a_i x^{i(n+2)} \\ &\quad + \sum_{i\in[n]} \sum_{j\in[n]\setminus\{i\}} a_i x^{j(n+1)+i} \\ (s + \sum_{i\in[n]} b_i x^i) \sum_{j\in[n]} x^{\rho(j)(n+2)-j} &= s \sum_{j\in[n]} x^{\rho(j)(n+2)-j} + \sum_{i\in[n]} b_i x^{\rho(i)(n+2)} \\ &\quad + \sum_{i\in[n]} \sum_{j\in[n]\setminus\{i\}} b_i x^{\rho(j)(n+2)+i-j}\end{aligned}$$

We have the desired sums $\sum_{i\in[n]} a_i x^{i(n+2)}$ and $\sum_{i\in[n]} b_i x^{\rho(i)(n+2)}$, but due to the extra terms it is not the case that $e(c, \prod_{j\in[n]} g_{j(n+1)}) = e(d, \prod_{j\in[n]} g_{\rho(j)(n+2)-j})$.

The prover constructs an argument π that cancels out the extra terms, and the verifier checks that

$$e(c, \prod_{j\in[n]} g_{j(n+1)}) = e(d, \prod_{j\in[n]} g_{\rho(j)(n+2)-j}) e(g, \pi).$$

The prover also gives restriction arguments $\hat{\pi}, \dot{\pi}$ so that the verifier is guaranteed that π does not contain any $x^{n+2}, \ldots, x^{n(n+2)}$ terms. Soundness then follows because the verification equation gives us $\sum_{i\in[n]} a_i x^{i(n+2)} = \sum_{i\in[n]} b_i x^{\rho(i)(n+2)}$ when π is free of $x^{n+2}, \ldots, x^{n(n+2)}$ terms.

Statement: Commitments $c, d \in G$ and permutation $\rho \in S_n$.

Prover's witness: Openings $r, a_1, \ldots, a_n \in \mathbb{Z}_p$ and $s, b_1, \ldots, b_n \in \mathbb{Z}_p$ so

$$c = g^r \prod_{i\in[n]} g_i^{a_i} \quad \text{and} \quad d = g^s \prod_{i\in[n]} g_i^{b_i} \quad \text{and} \quad \forall i \in [n] : b_i = a_{\rho(i)}.$$

Argument: Compute the argument as

$$\begin{aligned}
\pi &= \prod_{j\in[n]} g_{j(n+1)}^{r} g_{\rho(j)(n+2)-j}^{-s} \\
&\qquad \cdot \prod_{i\in[n]} \prod_{j\in[n]\setminus\{i\}} g_{j(n+1)+i}^{a_i} g_{\rho(j)(n+2)+i-j}^{-b_i} \\
\hat{\pi} &= \prod_{j\in[n]} \hat{g}_{j(n+1)}^{r} \hat{g}_{\rho(j)(n+2)-j}^{-s} \\
&\qquad \cdot \prod_{i\in[n]} \prod_{j\in[n]\setminus\{i\}} \hat{g}_{j(n+1)+i}^{a_i} \hat{g}_{\rho(j)(n+2)+i-j}^{-b_i} \\
\dot{\pi} &= \prod_{j\in[n]} \dot{h}_{j(n+1)}^{r} \dot{h}_{\rho(j)(n+2)-j}^{-s} \\
&\qquad \cdot \prod_{i\in[n]} \prod_{j\in[n]\setminus\{i\}} \dot{h}_{j(n+1)+i}^{a_i} \dot{h}_{\rho(j)(n+2)+i-j}^{-b_i}
\end{aligned}$$

Verification: Output 1 if and only if

$$e(g, \hat{\pi}) = e(\pi, \hat{g}) \quad \wedge \quad e(g, \dot{\pi}) = e(\pi, \dot{h}) \quad \wedge \quad e(c, \prod_{j\in[n]} g_{j(n+1)}) = e(d, \prod_{j\in[n]} g_{\rho(j)(n+2)-j}) e(g, \pi).$$

Theorem 4. *The permutation argument has perfect completeness and perfect witness-indistinguishability. Assume the q-CPDH assumption holds. Then a non-uniform probabilistic polynomial time adversary has negligible chance of outputting a permutation ρ, commitments (c, d) and an acceptable argument $(\pi, \hat{\pi}, \dot{\pi})$ with corresponding openings of the commitments and the argument such that for some $i \in [n]$ we have $b_i \neq a_{\rho(i)}$.*

Proof. Straightforward computation shows that the argument has perfect completeness. To see that the argument is perfectly witness-indistinguishable, observe that given c, d, ρ there is a unique acceptable argument π satisfying the verification equation. Given π, the other parts $\hat{\pi}$ and $\dot{\pi}$ are also determined uniquely by the verification equations since $\hat{g}$ and $\dot{h}$ are generators of G. Hence any witness in the form of openings of c and d with $b_i = a_{\rho(i)}$ for all $i \in [n]$ yields the same argument $(\pi, \hat{\pi}, \dot{\pi})$.

Consider now a non-uniform probabilistic polynomial time adversary $\mathcal{A}$ that outputs $\rho \in S_n$ and $r, a_1, \ldots, a_n, s, b_1, \ldots, b_n$ and $z, \{z_\ell\}_{\ell\in\dot{S}}$ such that

$$c = g^r \prod_{i\in[n]} g_i^{a_i} \quad d = g^s \prod_{i\in[n]} g_i^{b_i} \quad \pi = g^z \prod_{\ell\in\dot{S}} g_\ell^{z_\ell} \quad e(c, \prod_{j\in[n]} g_{j(n+1)}) = e(d, \prod_{j\in[n]} g_{\rho(j)(n+2)-j}) e(g, \pi).$$

Computing the discrete logarithms of the verification equation we get

$$\begin{aligned}
\log_{e(g,g)} e(c, \prod_{j\in[n]} g_{j(n+1)}) &= (r + \sum_{i\in[n]} a_i x^i)(\sum_{j\in[n]} x^{j(n+1)}) \\
&= r \sum_{j\in[n]} x^{j(n+1)} + \sum_{i\in[n]} \sum_{j\in[n]} a_i x^{j(n+1)+i} \\
\log_{e(g,g)} e(d, \prod_{j\in[n]} g_{\rho(j)(n+2)-j}) &= (s + \sum_{i\in[n]} b_i x^i)(\sum_{j\in[n]} x^{\rho(j)(n+2)-j}) \\
&= s \sum_{j\in[n]} x^{\rho(j)(n+2)-j} + \sum_{i\in[n]} \sum_{j\in[n]} b_i x^{\rho(j)(n+2)+i-j} \\
\log_{e(g,g)} e(g, \pi) &= z + \sum_{\ell\in\dot{S}} z_\ell x^\ell
\end{aligned}$$

The verification equation $e(c, \prod_{j\in[n]} g_{j(n+1)}) = e(d, \prod_{j\in[n]} g_{\rho(j)(n+2)-j}) e(g, \pi)$ therefore gives us

$$\begin{aligned}
& r \sum_{j\in[n]} x^{j(n+1)} + \sum_{i\in[n]} \sum_{j\in[n]} a_i x^{j(n+1)+i} \\
= s & \sum_{j\in[n]} x^{\rho(j)(n+2)-j} + \sum_{i\in[n]} \sum_{j\in[n]} b_i x^{\rho(j)(n+2)+i-j} + z + \sum_{\ell\in\dot{S}} z_\ell x^\ell.
\end{aligned}$$

Recall that $\dot{S}$ does not contain $n+2, 2(n+2), \ldots, n(n+2)$. This means π does not have any $x^{n+2}, \ldots, x^{n(n+2)}$ terms. Moreover, since ρ is a permutation, the exponent $\rho(i)(n+2)$ is reached on each side only by the intended signal terms. Looking at the term $x^{\rho(i)(n+2)}$ in the polynomial equation, the coefficients on each side of the equality are therefore respectively $a_{\rho(i)}$ and b_i. Lemma 1 therefore gives us negligible probability for $b_i \neq a_{\rho(i)}$ for some $i \in [n]$. □

8 Constant Size NIZK Argument for Circuit Satisfiability

We will now give an NIZK argument for the satisfiability of a binary circuit C, which consists of a constant number of group elements but has a large common reference string. Without loss of generality we assume that the circuit consists of NAND-gates. Let a be the output wire of the circuit. By adding an extra self-looping NAND gate $a = \neg(a \wedge b)$ we force a to be true, so we can reduce the satisfiability problem to demonstrating that there is a truth-value assignment to the wires such that C is internally consistent with all the NAND-gates. We now give the full NIZK argument for circuit satisfiability that was outlined in the introduction.

At a high level, the construction combines the ingredients developed in the previous sections in a modular way. Knowledge commitments together with restriction arguments ensure that the prover is committed to well-formed vectors with openings in the intended index sets. Product arguments are used to enforce bitness and the NAND relations, while permutation arguments are used to enforce consistency with the circuit wiring. The final argument has constant size because each of these components has constant size.

Common reference string: Generate common reference string $\sigma = (ck, \tilde{\sigma}, \bar{\sigma}, \dot{\sigma})$ with $n = 2N$.
Statement: A circuit C with N NAND-gates, where we want to prove the wires can be assigned values such that the circuit is internally consistent.

Witness: $2N$ input values $a_1, \ldots, a_N, b_1, \ldots, b_N \in \{0,1\}$ for the N gates that are consistent with the wires in the circuit and respect the NAND-gates. Define $u_1, \ldots, u_N$ to be the values of the output wires, and let $r_1, \ldots, r_N$ be the remaining values in $(a_1, \ldots, a_N, b_1, \ldots, b_N)$, i.e., either inputs to the circuit or duplicates of NAND-gate output wires appearing multiple times as inputs to other NAND-gates.

Argument:

1. Create restricted knowledge commitment $(c_{a\|b}, \hat{c}_{a\|b}, \tilde{c}_{a\|b})$ to $(a_1, \ldots, a_N, b_1, \ldots, b_N)$.
2. Create restricted knowledge commitment $(d_{a\|b}, \hat{d}_{a\|b}, \bar{d}_{a\|b})$ to $(a_1, \ldots, a_N, b_1, \ldots, b_N)$.
3. Create restricted knowledge commitment $(c_{b\|a}, \hat{c}_{b\|a}, \tilde{c}_{b\|a})$ to $(b_1, \ldots, b_N, a_1, \ldots, a_N)$.
4. Create restricted knowledge commitment $(c_{b\|0}, \hat{c}_{b\|0}, \tilde{c}_{b\|0})$ to $(b_1, \ldots, b_N, 0, \ldots, 0)$.
5. Create restricted knowledge commitment $(c_{u\|r}, \hat{c}_{u\|r}, \tilde{c}_{u\|r})$ to $(u_1, \ldots, u_N, r_1, \ldots, r_N)$.
6. Create restricted knowledge commitment $(c_{u\|0}, \hat{c}_{u\|0}, \tilde{c}_{u\|0})$ to $(u_1, \ldots, u_N, 0, \ldots, 0)$.
7. Show that $c_{a\|b}$ and $d_{a\|b}$ contain the same values by giving a product argument for $c_{a\|b}$ containing the entry-wise product of the values in $\prod_{i=1}^{2N} g_i$ (a commitment to $(1, \ldots, 1, 1, \ldots, 1)$) and $d_{a\|b}$.
8. Show that $a_1, \ldots, a_N, b_1, \ldots, b_N \in \{0,1\}$ by giving a product argument for $c_{a\|b}$ containing the entry-wise product of the values in $c_{a\|b}$ and $d_{a\|b}$.
9. Show that the values are internally consistent with the wires. The values $a_{i_1}, \ldots, a_{i_\ell}, b_{j_1}, \ldots, b_{j_m}$ may, for instance, all correspond to the same wire. Pick a permutation $\rho \in S_{2N}$ such that it contains cycles of the form $i_1 \to i_2 \to \ldots \to i_\ell \to j_1 + N \to j_2 + N \to \ldots \to j_m + N \to i_1$ for all sets of values corresponding to the same wire. Give a permutation argument for $c_{a\|b}$ containing the ρ-permutation of the values in $c_{a\|b}$. For each such set, this shows $a_{i_2} = a_{i_1}, \ldots, b_{j_1} = a_{i_\ell}, \ldots, b_{j_m} = b_{j_{m-1}}$. Hence the values must be the same.
10. Give a permutation argument for $c_{u\|r}$ and $c_{a\|b}$ showing that the output values $(u_1, \ldots, u_N)$ are consistent with the input values $(a_1, \ldots, a_N, b_1, \ldots, b_N)$. The $(r_1, \ldots, r_N)$ values are the remaining N values in $(a_1, \ldots, a_N, b_1, \ldots, b_N)$ that correspond either to duplicates of an output wire or to input wires of the circuit.
11. Give a permutation argument for $c_{b\|a}$ containing the swap of the values in $c_{a\|b}$.
12. Give a product argument for $c_{b\|0}$ containing the entry-wise product of the values in $c_{b\|a}$ and $\prod_{j=1}^{N} g_{j(n+1)}$ (a commitment to $(1, \ldots, 1, 0, \ldots, 0)$).
13. Give a product argument for $c_{u\|0}$ containing the entry-wise product of the values in $c_{u\|r}$ and $\prod_{j=1}^{N} g_{j(n+1)}$ (a commitment to $(1, \ldots, 1, 0, \ldots, 0)$).
14. Show that the NAND-gates are respected by giving a product argument for $c_{u\|0}^{-1} \prod_{i=1}^{N} g_i$ (a commitment to $(1 - u_1, \ldots, 1 - u_N, 0, \ldots, 0)$) containing the entry-wise product of the values in $c_{b\|0}$ and $d_{a\|b}$.

The argument consists of the 6 knowledge commitments with corresponding restriction arguments, the 5 product arguments and the 3 permutation arguments given above. The total size is 42 group elements.

Verification: Accept the argument if and only if the 6 knowledge commitments are well-formed, their corresponding restriction arguments are acceptable, the 5 product arguments are acceptable and the 3 permutation arguments are acceptable.

Theorem 5. *The NIZK argument for circuit satisfiability is perfectly complete and has perfect zero knowledge. If the q-PKE and q-CPDH assumptions hold with $q = 4N^2 + 6N - 2$, then the NIZK argument is computationally sound.*

Proof. Perfect completeness follows from the perfect completeness of the restriction, product and permutation arguments.

The zero-knowledge simulator works as follows. The common reference string is generated correctly, but the simulation trapdoor x makes it possible to create trapdoor commitments that can be opened to any set of values. The simulator creates trapdoor commitments $c_{a\|b}, d_{a\|b}, c_{b\|a}, c_{b\|0}, c_{u\|r}, c_{u\|0}$. Since trapdoor commitments are distributed identically to commitments to $(0,\ldots,0,0,\ldots,0)$, we can give the corresponding knowledge and restriction arguments. By trapdoor opening to $a_i = 1, b_i = 1, u_i = 1, r_i = 1$ for all $i \in [N]$, the simulator can give arguments 7-13: the four product arguments before the NAND check and the three permutation arguments. With this opening, all committed vectors are either $(1,\ldots,1,1,\ldots,1)$ or $(1,\ldots,1,0,\ldots,0)$, so each of these seven checks is satisfied. By trapdoor opening instead to $a_i = 1, b_i = 1, u_i = 0$ for all $i \in [N]$, it can give the last product argument, which checks the NAND relation.

Let us now argue that this perfectly simulates a real argument. Consider a hybrid between a real NIZK argument and a simulated NIZK argument, where we make trapdoor commitments but open them to a real witness $(a_1,\ldots,a_N,b_1,\ldots,b_N)$ when making the product and permutation arguments. Since the commitments are perfectly trapdoor, the hybrid is perfectly indistinguishable from a real NIZK argument. At the same time, since the arguments are perfectly witness-indistinguishable, the hybrid and the simulated NIZK argument are also perfectly indistinguishable.

It remains to show that the argument is sound. Consider a non-uniform probabilistic polynomial time adversary $\mathcal{A}$ that creates a circuit C and an accepting NIZK argument π. By the q-PKE assumption, this implies the existence of a non-uniform probabilistic polynomial time extractor $\mathcal{X}_{\mathcal{A}}$ that, running on the same input, extracts openings of the commitments and the arguments.

The restriction arguments give us, by the q-CPDH assumption, that the extracted openings are restricted to respectively $\tilde{S}, \bar{S}$ and $\dot{S}$. The product and permutation arguments then give us, again by the q-CPDH assumption, that the openings satisfy the corresponding relations between the committed values.

The first argument shows that $c_{a\|b}$ and $d_{a\|b}$ contain the same values, and the second argument then implies that the extracted values $a_1,\ldots,a_N,b_1,\ldots,b_N$ lie in $\{0,1\}$. The third argument shows that the values are consistent with the wires. The fourth argument shows that the output wires $(u_1,\ldots,u_N)$ have the values corresponding to $(a_1,\ldots,a_N,b_1,\ldots,b_N)$. The following four arguments show that for all $i \in [N]$ we have $1 - u_i = a_i b_i$, which means the values respect the NAND-gates. Since the values are consistent with the wires and respect the NAND-gates, the circuit is satisfiable. □

Arithmetic circuits. It is possible to adjust our NIZK argument to handle arithmetic circuits consisting of addition and multiplication gates over $\mathbb{Z}_p$. The commitment scheme is homomorphic so if we multiply two commitments we get the sum of their values, which can be used to handle the addition gates. The multiplication gates can be handled with our product arguments.

9 Reducing the Common Reference String

In the last section, we constructed constant-size NIZK arguments. The common reference string, however, grows quadratically in the size of the circuit. If the NIZK argument is only used a few times, the cost of setting up the common reference string may be prohibitive. In this section,

we outline how to reduce the size of the common reference string in return for increasing the size of the argument. If the circuit has $2N = n^d$ wires for some constant $d \geq 1$, we can get a common reference string with $O(n^2)$ group elements and an NIZK argument with $O(n^{d-1})$ group elements. If we choose $d = 3$, the CRS and the NIZK argument each have size $O(N^{2/3})$, making both components sub-linear in the circuit size.

The intuition is that we can reduce the common reference string if we accept that each commitment holds fewer values. If we have a circuit with n^d wires and a common reference string with $q = n^2 + 3n - 2 = O(n^2)$, the set $\tilde{S}$ permits commitments to n elements at a time. Each commitment is still a constant number of group elements, but now we need n^{d-1} commitments to commit to all the $2N = n^d$ input values to the gates. The product and permutation arguments are also of constant size, but they only work on commitments to n values. In our NIZK argument, the product arguments can simply be used on each of the n^{d-1} triples of commitments containing n values each, so there is no problem here. The ordinary permutation argument is not sufficient, however, because we need to permute $2N = n^d$ committed values spread across n^{d-1} commitments. The goal in this section is therefore to build a permutation argument for two n^{d-1}-tuples of commitments to a total of $2N = n^d$ values each. This permutation argument consists of $O(n^{d-1})$ group elements and uses the existing CRS of size $O(n^2)$.

9.1 Permutation Argument Spanning Multiple Commitments

Consider two sets of n commitments $c_1, \ldots, c_n, d_1, \ldots, d_n$ to values $a_{11}, \ldots, a_{nn}$ and $b_{11}, \ldots, b_{nn}$. We will use a Clos-network [Clo53] to give an argument for the two sets of committed values being permutations of each other for some publicly known permutation $\rho \in S_{n^2}$. The idea in a Clos network is to build large permutations from smaller permutations. Consider a permutation $\rho \in S_{n^2}$. First we divide the elements into n blocks of n elements and permute the elements within each block. Next, we distribute the elements in each block evenly on n other blocks giving us a new set of n blocks each containing one element from each of the previous blocks. We permute the elements in each block again. Once again, we distribute the elements in each block evenly on n new blocks. Finally, we permute the elements within the last blocks to get the elements permuted in the desired order. The permutations in the Clos network vary depending on ρ, whereas the distributions between blocks are fixed and independent of ρ.

To give a permutation argument for $\{c_i\}_{i\in[n]}, \{d_i\}_{i\in[n]}$ containing the same values permuted according to $\rho \in S_{n^2}$, the prover builds a Clos network for the permutation ρ. It constructs 4 sets of n intermediate commitments $\{c'_i\}_{i\in[n]}, \{v_i\}_{i\in[n]}, \{v'_i\}_{i\in[n]}, \{d'_i\}_{i\in[n]}$ together with arguments of knowledge and restriction arguments. Each commitment contains a block of n values in the middle stages of the Clos network. It uses the permutation argument from Section 7 to show that for all $i \in [n]$ the pairs of commitments $(c_i, c'_i), (d_i, d'_i)$ and (v_i, v'_i) contain the same elements in permuted order. The remaining problem is to give an argument for having dispersed the values between $\{c'_i\}_{i\in[n]}$ and $\{v_j\}_{j\in[n]}$ such that for each c'_i the values have been dispersed to n different v_j's, and likewise to give a dispersion argument for having spread the values in $\{v'_i\}_{i\in[n]}$ to $\{d'_j\}_{j\in[n]}$ such that for each v'_i the n committed values have been dispersed to n different d'_js. We present a dispersion argument in Section 9.2, which uses the existing CRS consisting of $O(n^2)$ group elements and has argument size $O(n)$. Counting the cost of all the commitments, within-block permutation arguments, and the dispersion arguments, we get a total size of $O(n)$ group elements for proving that two sets of n commitments to n values each have been permuted according to a publicly known permutation $\rho \in S_{n^2}$.

Once we have a permutation argument for n^2 values spread over n commitments, we can recursively get permutation arguments for larger permutations. For any constant d the cost of a permutation argument for n^d elements spread over two sets of n^{d-1} commitments is $O(n^{d-1})$ group elements.

9.2 Dispersion Argument

Consider matrices

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \qquad B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{pmatrix}.$$

We can view commitments $c_1, \ldots, c_n$ given by $c_j = g^{r_j} \prod_{i \in [n]} g_i^{a_{ij}}$ as commitments to the columns of A. Similarly, we can view $d_1, \ldots, d_n$ given by $d_i = g^{s_i} \prod_{j \in [n]} g_{j(n+1)}^{b_{ij}}$ as commitments to the rows of B. We give an argument for demonstrating that $c_1, \ldots, c_n$ and $d_1, \ldots, d_n$ contain the columns and the rows of the same $n \times n$ matrix, i.e., $A = B$. This means that for each c_j the n committed values have been distributed to n different commitments $d_1, \ldots, d_n$.

To get some intuition for the construction consider first the simple case where all the randomizers are 0. We then have

$$\prod_{j \in [n]} e(c_j, g_{j(n+1)}) = \prod_{i \in [n]} e(g_i, d_i).$$

Taking discrete logarithms on both sides of the equation we get

$$\sum_{j \in [n]} \sum_{i \in [n]} a_{ij} x^{j(n+1)+i} = \sum_{i \in [n]} \sum_{j \in [n]} b_{ij} x^{j(n+1)+i},$$

which by Lemma 1 implies $a_{ij} = b_{ij}$ for all $i, j \in [n]$. Due to the randomizers this verification equation will not hold in general though. The prover therefore constructs an argument $(\pi_L, \pi_R, \hat{\pi}_L, \hat{\pi}_R, \bar{\pi}_L, \tilde{\pi}_R)$ consisting of six group elements such that the cross-terms arising from the randomizers cancel out.

Statement: Commitments $c_1, \ldots, c_n, d_1, \ldots, d_n \in G$.
Prover's witness: Openings $r_1, \ldots, r_n, a_{11}, \ldots, a_{nn}, s_1, \ldots, s_n, b_{11}, \ldots, b_{nn} \in \mathbb{Z}_p$ so

$$\forall j \in [n] : c_j = g^{r_j} \prod_{i \in [n]} g_i^{a_{ij}} \qquad \forall i \in [n] : d_i = g^{s_i} \prod_{j \in [n]} g_{j(n+1)}^{b_{ij}} \qquad \forall i, j \in [n] : a_{ij} = b_{ij}.$$

Argument: Pick $t \leftarrow \mathbb{Z}_p$ at random and compute the argument $(\pi_L, \pi_R, \hat{\pi}_L, \hat{\pi}_R, \bar{\pi}_L, \tilde{\pi}_R)$ as

$$\pi_L = g^t \prod_{j \in [n]} g_{j(n+1)}^{-r_j} \qquad \pi_R = g^t \prod_{i \in [n]} g_i^{-s_i}$$

$$\hat{\pi}_L = \hat{g}^t \prod_{j \in [n]} \hat{g}_{j(n+1)}^{-r_j} \qquad \hat{\pi}_R = \hat{g}^t \prod_{i \in [n]} \hat{g}_i^{-s_i}$$

$$\bar{\pi}_L = \bar{h}^t \prod_{j \in [n]} \bar{h}_{j(n+1)}^{-r_j} \qquad \tilde{\pi}_R = \tilde{h}^t \prod_{i \in [n]} \tilde{h}_i^{-s_i}$$

Verification: Output 1 if and only if

$$e(g,\hat{\pi}_R) = e(\pi_R,\hat{g}) \qquad e(g,\tilde{\pi}_R) = e(\pi_R,\tilde{h}) \qquad e(g,\hat{\pi}_L) = e(\pi_L,\hat{g}) \qquad e(g,\bar{\pi}_L) = e(\pi_L,\bar{h})$$

$$e(g,\pi_L)\prod_{j\in[n]} e(c_j, g_{j(n+1)}) = e(g,\pi_R)\prod_{i\in[n]} e(g_i,d_i).$$

Theorem 6. *The dispersion argument is perfectly complete and perfectly witness-indistinguishable. Assume the q-CPDH assumption holds. Then a non-uniform probabilistic polynomial time adversary has negligible chance of producing commitments $c_1,\ldots,c_n,d_1,\ldots,d_n$ and an accepting argument $(\pi_L,\pi_R,\hat{\pi}_L,\hat{\pi}_R,\bar{\pi}_L,\tilde{\pi}_R)$ with corresponding openings of the commitments and the argument such that $c_1,\ldots,c_n$ and $d_1,\ldots,d_n$ are commitments to two different matrices.*

Proof. Perfect completeness follows by straightforward computation. For perfect witness-indistinguishability, observe that the g^t factor makes π_L uniformly random in G no matter what the prover's witness is. Given π_L, the main verification equation determines π_R uniquely. Given π_L and π_R, the remaining parts of the argument $\hat{\pi}_L,\hat{\pi}_R,\bar{\pi}_L,\tilde{\pi}_R$ are determined uniquely by the other verification equations. Therefore all openings with identical matrices lead to arguments with the same probability distribution.

Consider now a non-uniform probabilistic polynomial time adversary that outputs $r_1,\ldots,r_n,a_{11},\ldots,a_{nn},s_1,\ldots,s_n,b_{11},\ldots,b_{nn},y,y_1,\ldots,y_n,z,z_1,\ldots,z_n$ such that if we define

$$c_j = g^{r_j}\prod_{i\in[n]} g_i^{a_{ij}} \qquad d_i = g^{s_i}\prod_{j\in[n]} g_{j(n+1)}^{b_{ij}}$$

$$\pi_L = g^{y}\prod_{j\in[n]} g_{j(n+1)}^{y_j} \qquad \pi_R = g^{z}\prod_{i\in[n]} g_i^{z_i}$$

then we have

$$e(g,\pi_L)\prod_{j\in[n]} e(c_j, g_{j(n+1)}) = e(g,\pi_R)\prod_{i\in[n]} e(g_i,d_i).$$

Taking the discrete logarithm with respect to $e(g,g)$ on both sides of the equation we get

$$\begin{aligned} &y + \sum_{j\in[n]} y_j x^{j(n+1)} + \sum_{j\in[n]} r_j x^{j(n+1)} + \sum_{j\in[n]}\sum_{i\in[n]} a_{ij}x^{j(n+1)+i} \\ &= z + \sum_{i\in[n]} z_i x^i + \sum_{i\in[n]} s_i x^i + \sum_{i\in[n]}\sum_{j\in[n]} b_{ij}x^{j(n+1)+i} \end{aligned}$$

The randomizer and argument terms contribute only degrees of the form $j(n+1)$ or i, so for all $i,j\in[n]$ the coefficients of $x^{j(n+1)+i}$ on each side of the equality are respectively a_{ij} and b_{ij}. By Lemma 1 the adversary will have negligible success probability unless $a_{ij}=b_{ij}$ for all $i,j\in[n]$. □

10 Conclusion

We have constructed a pairing-based NIZK argument for circuit satisfiability, which is the first sublinear-size NIZK argument for circuit satisfiability that does not rely on the random oracle

model. The argument has perfect completeness and perfect zero knowledge and is computationally sound assuming the q-PKE and the q-CPDH assumptions hold.

More generally, for circuits with $N = \frac{1}{2}n^d$ gates and constant $d \in \mathbb{N}$, the common reference string has size $O(n^2)$ group elements and the NIZK argument has size $O(n^{d-1})$ group elements. This yields two particularly natural operating points. With $d = 1$ we obtain a constant-size NIZK argument at the cost of a common reference string of size $O(N^2)$ group elements. With $d = 3$ we obtain a more balanced regime where both the common reference string and the NIZK argument have size $O(N^{\frac{2}{3}})$ group elements.

These two operating points capture the main message of the paper: constant-size proofs are possible in the standard model, and a natural parameter tradeoff yields simultaneously sub-linear common reference strings and proofs.

Acknowledgment

We would like to thank Amit Sahai for helpful comments and Craig Gentry for a correction of an earlier version of the paper.

References

[AF07] Masayuki Abe and Serge Fehr. Perfect NIZK with adaptive soundness. In *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 118–136, 2007.

[BBP04] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An uninstantiable random-oracle-model scheme for a hybrid encryption problem. In *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 171–188, 2004.

[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 *Innovations in Theoretical Computer Science*, pages 326–349, 2012.

[BCNP04] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally composable protocols with relaxed set-up assumptions. In *FOCS*, pages 186–195, 2004.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *STOC*, pages 103–112, 1988.

[BP04] Mihir Bellare and Adriana Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In *CRYPTO*, Lecture Notes in Computer Science, pages 273–289. Springer, 2004.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS*, pages 62–73, 1993.

[BW06] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 427–444, 2006.

[BW07] Xavier Boyen and Brent Waters. Full-domain subgroup hiding and constant-size group signatures. In *PKC*, volume 4450 of *Lecture Notes in Computer Science*, pages 1–15, 2007.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. In *STOC*, pages 209–218, 1998.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. On the random-oracle methodology as applied to length-restricted signature schemes. In *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 40–57, 2004.

[CGS07] Nishanth Chandran, Jens Groth, and Amit Sahai. Ring signatures of sub-linear size without random oracles. In *ICALP*, volume 4596 of *Lecture Notes in Computer Science*, pages 423–434, 2007.

[Clo53] Charles Clos. A study of non-blocking switching networks. *Bell System Technical Journal*, 32(2):406–424, 1953.

[CS98] Ronald Cramer and Victor Shoup. A practical public-key encryption schemes secure against adaptive chosen ciphertext attack. In *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25, 1998.

[Dam91] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456, 1991.

[Dam92] Ivan Damgård. Non-interactive circuit based proofs and non-interactive perfect zero-knowledge with preprocessing. In *EUROCRYPT*, volume 658 of *Lecture Notes in Computer Science*, pages 341–355, 1992.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.

[DDO+02] Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. Robust non-interactive zero knowledge. In *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 566–598, 2002.

[DDP02] Alfredo De Santis, Giovanni Di Crescenzo, and Giuseppe Persiano. Randomness-optimal characterization of two NP proof systems. In *RANDOM*, volume 2483 of *Lecture Notes in Computer Science*, pages 179–193, 2002.

[DFN06] Ivan Damgård, Nelly Fazio, and Antonio Nicolosi. Non-interactive zero-knowledge from homomorphic encryption. In *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 41–59, 2006.

[DN00] Cynthia Dwork and Moni Naor. Zaps and their applications. In *FOCS*, pages 283–293, 2000.

[DP92] Alfredo De Santis and Giuseppe Persiano. Zero-knowledge proofs of knowledge without interaction. In *FOCS*, pages 427–436, 1992.

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs under general assumptions. *SIAM Journal on Computing*, 29(1):1–28, 1999.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct nizks without pcps. In *EUROCRYPT*, volume 7881 of *Lecture Notes in Computer Science*, pages 626–645, 2013.

[GK96] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.

[GK03] Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the Fiat-Shamir paradigm. In *FOCS*, pages 102–113, 2003.

[GKM+18] Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-SNARKs. In *CRYPTO*, volume 10993 of *Lecture Notes in Computer Science*, pages 698–728. Springer, 2018.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proofs. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, 1994.

[GO14] Jens Groth and Rafail Ostrovsky. Cryptography in the multi-string model. *J. Cryptology*, 27(3):506–543, 2014.

[GOS06a] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for NIZK. In *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 97–111, 2006.

[GOS06b] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero-knowledge for NP. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 339–358, 2006.

[Gro06] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT*, volume 4248 of *Lecture Notes in Computer Science*, pages 444–459, 2006.

[Gro09] Jens Groth. Linear algebra with sub-linear zero-knowledge arguments. In *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 192–208, 2009.

[Gro10] Jens Groth. Short non-interactive zero-knowledge proofs. In *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 341–358, 2010.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432, 2008.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments. In *STOC*, pages 723–732, 1992.

[KP98] Joe Kilian and Erez Petrank. An efficient noninteractive zero-knowledge proof system for NP with general assumptions. *Journal of Cryptology*, 11(1):1–27, 1998.

[Lip12] Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In *TCC*, volume 7194 of *Lecture Notes in Computer Science*, pages 169–189, 2012.

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000.

[MRH04] Ueli M. Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39, 2004.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 96–109, 2003.

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126, 2002.

[Ore87] Yair Oren. On the cunning power of cheating verifiers: Some observations about zero knowledge proofs. In *FOCS*, pages 462–471, 1987.

[Sah01] Amit Sahai. Non-malleable non-interactive zero-knowledge and adaptive chosen-ciphertext security. In *FOCS*, pages 543–553, 2001.

A Heuristic Arguments for the Assumptions

Theorem 7. *The $q(k)$-PKE assumption holds in the generic group model for $q(k) = \mathrm{poly}(k)$.*

Sketch of proof. In the generic group model, the adversary $\mathcal{A}$ acts obliviously to the actual representation of the group elements and only performs generic group operations such as multiplying elements in G and G_T, pairing elements in G, and comparing elements to see if they are identical.

When $\mathcal{A}$ only does generic group operations it can only produce new elements in G by multiplying existing group elements together. The adversary is given input elements $g, g^x, \ldots, g^{x^q}, g^\alpha, g^{\alpha x}, \ldots, g^{\alpha x^q}$ and outputs $c, \hat{c}$. The adversary can only win if $\hat{c} = c^\alpha$, so assume in the following that this is the case. The extractor $\mathcal{X}_\mathcal{A}$ can keep track of all the multiplications and therefore write the outputs $c, \hat{c}$ in the form

$$c = g^{a_0}(g^\alpha)^{b_0} \prod_{i\in[q]} (g^{x^i})^{a_i}(g^{\alpha x^i})^{b_i} \qquad \hat{c} = g^{u_0}(g^\alpha)^{v_0} \prod_{i\in[q]} (g^{x^i})^{u_i}(g^{\alpha x^i})^{v_i},$$

for known $a_0, b_0, u_0, v_0, \ldots, a_q, b_q, u_q, v_q$. If $b_i = u_i = 0$ and $a_i = v_i$ for all $i = 0, \ldots, q$, then the extractor outputs $a_0, \ldots, a_q$ such that $c = \prod_{i=0}^q g^{a_i x^i}$, which is permitted in the q-PKE problem. We will now argue that other unpermitted solutions do not occur. In other words, there is negligible probability for $\hat{c} = c^\alpha$ if $a_i \neq v_i$ or $b_i \neq 0$ or $u_i \neq 0$ for some i.

Taking discrete logarithms the condition $\hat{c} = c^\alpha$ says

$$\sum_{i=0}^q u_i x^i + \sum_{i=0}^q v_i \alpha x^i = \alpha(\sum_{i=0}^q a_i x^i + \sum_{i=0}^q b_i \alpha x^i) = \sum_{i=0}^q a_i \alpha x^i + \sum_{i=0}^q b_i \alpha^2 x^i.$$

With $a_i \neq v_i$ or $b_i \neq 0$ or $u_i \neq 0$ the two multivariate polynomials in α and x are different formal polynomials. Since the adversary is oblivious to the actual representation of the group elements it will do the same group operations no matter the actual values of α and x; so the values $a_0, \ldots, v_q$ are generated (almost[4]) independently of α and x. By the Schwartz-Zippel lemma there is negligible probability for the two polynomials evaluating to the same when we choose α, x randomly, and therefore negligible probability for $\hat{c} = c^\alpha$. □

Theorem 8. *The $q(k)$-CPDH assumption holds in the generic group model for $q(k) = \mathrm{poly}(k)$.*

Sketch of proof. In the generic group model, the adversary $\mathcal{A}$ acts obliviously to the actual representation of the group elements and only performs generic group operations such as multiplying elements in G and G_T, pairing elements in G, and comparing elements to see if they are identical.

[4] A generic group adversary may learn a little bit about α and x by comparing group elements. However, the information leak is negligible, so we skip this part in the proof.

When $\mathcal{A}$ only does generic group operations it can only produce new elements in G by multiplying existing group elements together. Keeping track of the group elements we get that it outputs

$$y = \prod_{i=0}^{q} (g^{x^i})^{a_i} \prod_{i \neq j} (g^{\beta x^i})^{b_i}$$

for known $a_0, b_0, \ldots, a_q, b_q$. Taking discrete logarithms on the condition $y = g^{\beta x^j}$ we get

$$\sum_{i=0}^{q} a_i x^i + \sum_{i \neq j} b_i \beta x^i = \beta x^j.$$

This can be seen as an equation involving two different formal polynomials over β and x.

Since the adversary is oblivious to the actual representation of the group elements it will do the same group operations no matter the actual values of β and x; so the values $a_0, b_0, \ldots, a_q, b_q$ are generated (almost[5]) independently of β and x. By the Schwartz-Zippel lemma there is negligible probability for the two different polynomials evaluating to the same when we choose β, x randomly, and therefore negligible probability for $y = g^{\beta x^j}$. □

B Addendum: Fixed Circuits and Preprocessing

Suppose the circuit C is fixed, and suppose as in Section 8 that it has N NAND-gates. Assume furthermore that the circuit has ℓ public input wires and m private input wires. A statement is therefore a public input instance $\mathbf{x} = (x_1, \ldots, x_\ell) \in \{0,1\}^\ell$, and a witness is a private input vector $\mathbf{w} \in \{0,1\}^m$ such that $C(\mathbf{x}, \mathbf{w}) = 1$. We explain in this addendum that the construction from Section 8 with minor tweaks is a preprocessing SNARK for this setting.

Without loss of generality, let the public input wires be numbered $1, \ldots, \ell$. We then reserve the positions $r_1, \ldots, r_\ell$ inside the commitment $c_{u\|r}$ for these public inputs, so that $r_i = x_i$ for all $i \in [\ell]$. The remaining values $r_{\ell+1}, \ldots, r_N$ contain the private inputs together with the same auxiliary values as in Section 8, namely unused input slots and duplicates of output wires appearing several times as inputs to later gates.

There are two separate simplifications in the fixed-circuit setting. The first concerns the verifier's work in the subproofs. In Section 8, each subproof requires a constant number of pairings, but also the computation of an n-term product. In the case of the product arguments, these products can be precomputed and included as part of the CRS, so the verifier does not need to spend the effort online. The same goes for the permutation argument verifying a swap: the relevant verifier-side product can be added as a precomputed group element in the CRS. The only remaining issue is the permutation arguments for the wiring of the circuit and for matching $(u_1, \ldots, u_N, r_1, \ldots, r_N)$ with $(a_1, \ldots, a_N, b_1, \ldots, b_N)$. These permutations depend on the wiring of the circuit, so they cannot be precomputed in a way that works for all circuits. The reason is that the permutation verifier encodes the relevant permutation ρ as a product $\prod_{j \in [n]} g_{\rho(j)(n+2)-j}$. If the circuit is fixed, however, these products too can be precomputed during preprocessing.

Accordingly, the preprocessing algorithm for the fixed circuit C generates the ordinary common reference string σ from Section 8 and augments it with the circuit-specific data needed by the verifier:

[5] A generic group adversary may learn a little bit about β and x by comparing group elements. However, the information leak is negligible, so we skip this part in the proof.

1. The three verifier-side grand products corresponding to the permutations used in the proof, namely the wiring permutation on $(a_1, \ldots, a_N, b_1, \ldots, b_N)$, the permutation matching $(u_1, \ldots, u_N, r_1, \ldots, r_N)$ with $(a_1, \ldots, a_N, b_1, \ldots, b_N)$, and the swap permutation between $(a_1, \ldots, a_N, b_1, \ldots, b_N)$ and $(b_1, \ldots, b_N, a_1, \ldots, a_N)$.
2. The fixed commitments that appear in the product arguments, for instance the commitment to $(1, \ldots, 1, 1, \ldots, 1)$ and the commitment to $(1, \ldots, 1, 0, \ldots, 0)$.
3. A third change to the CRS, to be used later, is a modified knowledge-commitment and restriction key for the support set

$$\tilde{S}_{\text{rest}} = \{1, \ldots, N\} \cup \{N + \ell + 1, \ldots, 2N\},$$

which excludes the positions reserved for the public inputs. Equivalently, on the $\hat{g}$-side we omit the powers $\hat{g}_{N+1}, \ldots, \hat{g}_{N+\ell}$ corresponding to the public coordinates.

All of this data depends only on the fixed circuit and can therefore be computed once during preprocessing.

The second simplification concerns the public input itself. In the general construction, the prover commits to a full wire-value assignment in $c_{u\|r}$. In the fixed-circuit setting, however, the verifier needs to ensure that the designated wires match the public input. An easy way to do this is to split the commitment into two parts that can be computed separately. The verifier computes a commitment

$$c_{\mathbf{x}} = \prod_{j=1}^{\ell} g_{N+j}^{x_j}$$

to the public-input wires. The prover creates a restricted knowledge commitment $(c_{\text{rest}}, \hat{c}_{\text{rest}})$ to the remaining values, namely a vector of the form

$$(u_1, \ldots, u_N, 0, \ldots, 0, r_{\ell+1}, \ldots, r_N),$$

and proves with respect to the restriction set $\tilde{S}_{\text{rest}}$ that the public-input coordinates are all 0. The full commitment used when verifying the product and permutation arguments from Section 8 is then reconstructed by homomorphism as

$$c_{u\|r} = c_{\mathbf{x}} c_{\text{rest}}.$$

Since the public commitment occupies exactly the coordinates $N+1, \ldots, N+\ell$ and the prover's commitment is restricted away from these coordinates, the combined commitment has the designated public values in the correct slots and the prover cannot modify them. The point of $\hat{c}_{\text{rest}}$ is only to certify knowledge of the complementary coordinates; there is no need for a corresponding $\hat{c}_{\mathbf{x}}$, because the verifier computed the public part itself.

After this substitution, the prover gives exactly the same permutation arguments and product arguments as in Section 8; the only difference is that wherever Section 8 used $c_{u\|r}$ and $\hat{c}_{u\|r}$, the verifier now uses the combined commitments above.

The verifier no longer needs to read the circuit description online. The circuit-dependent grand products for the permutation arguments, as well as the fixed commitments used in the product arguments, have already been absorbed into the preprocessed common reference string. The only online computation depending on the statement is the formation of $c_{\mathbf{x}}$, costing fewer than ℓ group multiplications.

Soundness follows from the same extraction argument as in Section 8. By the q-PKE assumption we extract the opening of $(c_{\mathrm{rest}}, \hat{c}_{\mathrm{rest}})$ and of the other prover-generated commitments and arguments. Since $(c_{\mathrm{rest}}, \hat{c}_{\mathrm{rest}})$ is restricted to $\tilde{S}_{\mathrm{rest}}$, it has zero in all public-input coordinates. Therefore, after multiplying by the verifier-computed public commitment $c_{\mathbf{x}}$, the extracted opening of $c_{u\|r}$ has public-input coordinates exactly equal to $\mathbf{x}$. The remaining arguments then imply, exactly as in Section 8, that the extracted values describe a satisfying assignment for the fixed circuit. Hence the extracted assignment consists of a witness $\mathbf{w}$ for the public instance $\mathbf{x}$.

Zero knowledge is proved exactly as before. The simulator forms the public commitment $c_{\mathbf{x}}$ from the statement and uses trapdoor commitments for the prover-generated part $(c_{\mathrm{rest}}, \hat{c}_{\mathrm{rest}})$ and the remaining commitments. Since the commitments are perfectly trapdoor and the product and permutation arguments are perfectly witness-indistinguishable, the simulation is perfect. We therefore obtain a preprocessing SNARK for the fixed circuit C with the same constant-size proof as in Section 8 and verifier work $O(\ell)$ plus a constant number of pairing evaluations. In particular, when the public input is shorter than the circuit description, the online verification is sublinear in the circuit size.