Progression-Free Sets and Sublinear Pairing-Based Non-Interactive Zero-Knowledge Arguments

Helger Lipmaa

Institute of Computer Science, University of Tartu, Estonia

Abstract. In Asiacrypt 2010, Groth constructed the only previously known sublinear-communication NIZK argument for circuit satisfiability in the common reference string model. We propose a more efficient NIZK argument. In particular, we reduce both the CRS length and the prover's computational complexity from quadratic to quasilinear. We also use a (presumably) weaker security assumption than Groth, and have tighter security reductions.
From a theoretical perspective, we achieve the mentioned efficiency gain by using a (seemingly natural, but at the end quite non-trivial) connection with the theory of progression-free sets and with additive combinatorics in general. While bilinear protocols seem to be specifically suited for using methods from additive combinatorics, we are not aware of any previous work that has made this connection. We hope that more applications of this connection will be found in the future.
Keywords. Additive combinatorics, bilinear pairings, circuit satisfiability, NIZK, progression-free sets.

1 Introduction

By using a zero-knowledge [GMR85] proof, a prover can make a verifier to accept a statement without leaking any side information. Due to the wide applications of zero-knowledge, it is of utmost importance to construct efficient zero-knowledge proofs. In particular, *non-interactive zero-knowledge* (NIZK) proofs are extremely useful, since in many applications the zero-knowledge proof is generated once but has to be verified many times by potentially different verifiers. As an example, in the setting of e-voting, the voting server has to prove that the tally is correct. It is infeasible to interact with the server every single time when verifying the correctness of the election outcome, and thus using NIZK proofs seem to be the only possible solution in this case.

It is well-known that NIZK proofs (or arguments, that is, computationally sound proofs) cannot be constructed in the plain model (i.e., without random oracles or any trusted setup assumptions). Blum, Feldman and Micali showed in [BFM88] how to construct NIZK proofs in the common reference string (CRS) model. During the last years, substantial amount of research has been done towards constructing efficient NIZK proofs (and arguments). Since the communication complexity and the verifier's computational complexity are arguably more important than the prover's computational complexity (again, an NIZK proof/argument is generated once but can be verified many times), special effort has been put in minimizing these two parameters.

In some NIZK proofs for **NP**-complete problems, see Tbl. 1, both the communication complexity and the verifier's computational complexity are sublinear. Micali proposed polylogarithmic NIZK *arguments* for all **NP**-languages. However, his CS (computationally sound) proofs [Mic94] are based on the PCP theorem (making them computationally unattractive) and on the random oracle model (ROM). Gentry [Gen09] noted that given a fully-homomorphic cryptosystem, one can construct efficient NIZK *proofs* for all **NP**-languages in the ROM. Another NIZK argument for circuit satisfiability, a **NP**-complete problem, proposed by Groth in 2009 [Gro09], is also based on the ROM. (Without relying on the ROM, this argument takes $\log_2 |C| + 5$ rounds, where $|C|$ is the circuit size. Recently, Seo [Seo11] showed how to reduce the number of rounds in Groth's argument.)

While one can design efficient protocols in the ROM [BR93], it is also known [CGH98] that some functionalities are secure in the ROM and insecure in the plain model. In particular, Goldwasser and Kalai [GK03] designed a signature scheme, built by using the Fiat-Shamir heuristics (which is predominant in the construction of NIZK arguments in the ROM), that is secure in the ROM but insecure when instantiated with any "real" hash function. As a safeguard, it is important to design efficient NIZK proofs and arguments that do not rely on the ROM.

Recently, Groth [Gro10] proposed the first sublinear-communication NIZK argument for circuit satisfiability in the CRS model. Groth's circuit satisfiability argument is based on two basic NIZK arguments, for Hadamard product (the prover knows how to open three commitments to vectors where the third vector is a Hadamard product of the first two vectors) and permutation (the prover knows how to open two commitments to vectors where the second one is permuted according to a public permutation). In particular, it inherits the length-$\Theta(|C|^2)$ CRS of those two basic arguments. The security of Groth's NIZK argument is based on two assumptions, a computational assumption (CPDH, see App. A) and a knowledge assumption (PKE, see Sect. 2).

While knowledge assumptions are unfalsifiable [Nao03], it is necessary to use some non-standard assumption due to known impossibility results [AF07,GW11]. In fact, the assumption that an NIZK argument is sound is by itself unfalsifiable, since even if an adversary outputs a false statement together with a convincing NIZK argument, it may be hard to verify that the statement is false. A concrete and well-defined knowledge assumption (that is proven secure in the generic group model) seems to offer a nice alternative to the ROM. See [Gro10] for more discussion.

Our Contributions. As our underlying observation, we note that Groth's basic arguments are constructed by making use of $\Theta(n^2)$ generators of the underlying group, where n is their input size (dimension of the vectors). Of those possible generators, only $\leq 2n$ generators are needed in the description of the arguments, while the rest of the generators are inherent to the proof technique. When constructing the NIZK argument for circuit satisfiability, this amounts to using $\Theta(|C|^2)$ generators that must all be described by the CRS. In this paper, we re-examine Groth's argument by first constructing more efficient basic arguments, and then show how to put them together in a more efficient circuit satisfiability argument. We also use more efficient asymmetric pairings (not symmetric, as in [Gro10]), but for the sake of convenience we will use symmetric notation throughout the introduction.

New Basic Arguments. We use a generalized version of Groth's knowledge commitment scheme with n generators $g^{x^{\lambda_i}}$, with $0 = \lambda_0 < \lambda_1 < \lambda_2 < \cdots < \lambda_n = \mathrm{poly}(\kappa)$, to commit to a vector $\boldsymbol{a} = (a_1, \ldots, a_n)$. More precisely, for a secret key $x \in \mathbb{Z}_p$ and a new randomizer $r_a \in \mathbb{Z}_p$, $\mathcal{C}\mathrm{om}(\mathsf{ck}; \boldsymbol{a}; r_a) = g^{r_a + \sum_{i=1}^n a_i x^{\lambda_i}} = g^{r_a} \cdot \prod_{i=1}^n g_i^{a_i}$. Here, ck is a CRS that in particular fixes a bilinear group of order p, a generator g, and group elements $g_i = g^{\lambda_i}$ for $i \in \{1, \ldots, n\}$. See Sect. 4.

For the arguments to be perfectly complete and sound (more precisely, culpably sound [GL07,GOS11], also known as co-sound, see Sect. 2), the set $\Lambda = \{\lambda_i\}$ has to satisfy certain properties. E.g., in the product argument, the prover constructs an argument π such that $e(A, B) = e(C, D) \cdot e(g, \pi)$, where e is the bilinear pairing, $A = \mathcal{C}\mathrm{om}(\mathsf{ck}; \boldsymbol{a}; r_a)$, $B = \mathcal{C}\mathrm{om}(\mathsf{ck}; \boldsymbol{b}; r_b)$, $C = \mathcal{C}\mathrm{om}(\mathsf{ck}; \boldsymbol{c}; r_c)$, and $D = \mathcal{C}\mathrm{om}(\mathsf{ck}; \mathbf{1}; 0)$. As we show in Sect. 5, the discrete logarithm of π, $(r_a + \sum_{i=1}^n a_i x^{\lambda_i})(r_b + \sum_{i=1}^n b_i x^{\lambda_i}) - (r_c + \sum_{i=1}^n c_i x^{\lambda_i})(\sum_{i=1}^n x^{\lambda_i})$, can be thought as $\pi(x)$, where $\pi(X) = \sum_{i \in [n]} (a_i b_i - c_i) X^{2\lambda_i} + \sum_{i \in \bar{\Lambda}} \pi_i X^i$ is a polynomial. Here, the powers corresponding to nonzero coefficients belong either to the set $2 \cdot \Lambda = \{2\lambda_i : i \in [n]\}$ (the coefficients $a_i b_i - c_i$) or to the set $\bar{\Lambda} = \{0\} \cup \{\lambda_i\} \cup \{\lambda_i + \lambda_j : i \neq j\}$. If the prover is honest (that is, $a_i b_i - c_i = 0$ for all i), then the coefficients $a_i b_i - c_i$ corresponding to the powers in the set $2 \cdot \Lambda$ are equal to 0.

To be certain that the last claim holds, we additionally need that $2 \cdot \Lambda \cap \bar{\Lambda} = \emptyset$. As we show in Sect. 3, Λ has to be progression-free (that is, not contain any arithmetic progressions of length 3, see Sect. 3), while it is sufficient to assume that Λ is a progression-free set of odd integers. Thus, our choice of the set Λ is optimal (up to a factor of 2 since one can construct contrived examples of suitable sets Λ that also include even integers). Essentially the same idea also works in the case of our permutation argument, though in that case we have to use an extra trick—namely, we have to use two separate subarguments, one that shows that $a_{\varrho(i)} \geq b_i$ for $i \in [n]$ and another one that shows that $a_{\varrho(i)} \leq b_i$ for $i \in [n]$—to be able to use permutation-free sets (and not rely on the Moser-de Bruijn sequence [Mos62,dB64], where $\lambda_n = \Theta(n^2)$). While in the case of the permutation argument, the set $\bar{\Lambda}$ is larger than in the case of the product argument, it suffices again for Λ to be a progression-free set of odd integers.

For the prover to be able to construct π, the set $\{g^{x^i} : i \in \bar{\Lambda}\}$ must belong to the CRS, and thus in particular we need $\bar{\Lambda}$ to be small. This can be guaranteed by upper bounding λ_n, since $\bar{\Lambda} \subset \{0, \ldots, 2\lambda_n\}$. (In the case of the permutation argument, $\bar{\Lambda} \subset \{-\lambda_n, \ldots, 3\lambda_n\}$.) On the other hand, for (culpable) soundness it is important that no element g^{x^i}, $i \in 2 \cdot \Lambda$, belongs to the CRS. (The real arguments are somewhat more complicated, see Sect. 5 and Sect. 6.) For this we show in Sect. 3 that any range $[y] = \{1, \ldots, y\}$ contains a progression-free set of odd integers of size $n = y^{1-o(1)}$, and thus one can assume that $\lambda_n = n^{1+o(1)}$. In App. B, we will give another proof of this result that, while based on

Green and Wolf's exposition [GW10] of Elkin's recent breakthrough [Elk10], provides more details and is slightly sharper. In particular, Elkin's progression-free set [Elk10] is efficiently constructible.

Thus, we obtain basic arguments that only use $O(\lambda_n)$ generators $\{g^{x^i} : i \in \bar{\Lambda}\}$. This means that the length of the CRS is $n^{1+o(1)}$ and not $\Theta(n^2)$ as in [Gro10]. In the case of both basic arguments, the verifier only has to perform $\Theta(1)$ bilinear pairings (plus $n-1$ multiplications in the case of the permutation argument that can be done offline if the permutation is fixed), and the prover has to compute π (which takes $\Theta(n^2)$ multiplications or additions in $\mathbb{Z}_p$ and $n^{1+o(1)}$ bilinear-group exponentiations). The new basic arguments are substantially more efficient than those of Groth. As in [Gro10], the prover's computation can be optimized even further by using efficient multi-exponentiation algorithms. They are also based on a weaker security assumption (power-DL instead of power-CDH), and have a tighter security reduction (due to the smaller value of λ_n). See Sect. 7 for a detailed comparison.

Circuit Satisfiability Argument. We use the basic arguments to construct a new circuit satisfiability (SAT) argument, using a technique similar to that of Groth [Gro10]. Briefly, the two basic arguments together with the trivial argument for element-wise sum (given commitments to $\boldsymbol{a}$ and $\boldsymbol{b}$, the product of the commitments commits to $\boldsymbol{a}+\boldsymbol{b}$) form basic building blocks for an NIZK "programming language". We show that in this "language", one can build an NIZK for circuit satisfiability in a small constant number of steps. The new SAT argument than [Gro10] is more efficient not only because the basic arguments are more efficient, but also since we use a slightly different "program". For the SAT argument to be sound, following [Gro10] we rely on a knowledge assumption (PKE).

The communication and computation of the new SAT argument follow directly from those of the basic arguments. In addition, we apply a few more optimizations that result in the SAT argument having shorter communication (33 group elements instead of 42). We also use a (potentially) weaker security assumption than Groth (PSDL versus CPDH, which is like using the DL assumption instead of the DH assumption) in the security proof. We note that the security parameter κ depends on $\log |C|$, and is thus not a constant value. If $|C|$ grows, then so does κ and the group size. The same comment however also applies to the prior work. Since we have tighter security reductions than [Gro10], we can use a somewhat smaller value of κ. We will not elaborate on this point.

Importantly, instead of an ad hoc choice of the set Λ in [Gro10], we use an essentially optimal one. However, thus also the obtained efficiency estimates (see Tbl. 1) are close to optimal. One can try to replace $|C|^{1+o(1)}$ with $\Theta(|C|)$ in the case of CRS length and the prover's computation, though for this a different proof technique seems to be required. Importantly, in our case, both the argument length and the verifier's computational complexity are linear: even if the prover spends a huge amount of time to compute the argument, he only has to do it once, while many verifiers can verify the short argument many times, and each verification takes linear time. In many applications (like e-voting, where one needs to verify the correctness of the tally), this is very important. Moreover, our results show that one can build an argument with constant communication complexity for every language in **NP**: one can just reduce the task at hand to a SAT instance.

Micali's computationally sound proofs [Mic94] and Gentry's NIZK on top of fully-homomorphic cryptosystems [Gen09] have (asymptotically) longer arguments. Thus, up to our best knowledge, the new argument for circuit satisfiability is more communication-efficient than any prior art random-oracle based NIZK argument for the same problem. In the case of the prover's computational complexity, the new argument is also more efficient than Micali's computationally sound proofs since the latter uses computationally unattractive PCP proofs.

Balancing. As shown by Groth [Gro10], to achieve both sublinear CRS length and communication, one can apply the same basic arguments on length-m inputs, $m < n$, n/m times in parallel. The actual balanced argument is more complex, see [Gro10] for details. We can use the same balancing technique with the new arguments. We can also balance the CRS length and the communication complexity of the new argument by either choosing $m = n^{1/3}$ or $m = n^{1/2}$. See Tbl. 1.

Perfect Zaps. In App. J, we describe zap [DN00] for SAT has the communication complexity of $|C|^{1/2+o(1)}$ group elements, while Groth's zap from [Gro10] has the communication complexity of $\Theta(|C|^{2/3})$ group elements.

Theoretical Considerations. It is interesting that in the case of both basic arguments, it was sufficient to use progression-free sets of odd integers. In the case of the permutation argument, this is not trivial at all, as we will show in Sect. 6. It is intriguing that there we will avoid the use of the much sparser Moser-de Bruijn sequence (with $\lambda_n = \Theta(n^2)$) by using a simple conjunction of two more basic subarguments. As

	CRS length	Argument length	Prover comp.	Verifier comp.	Assumption
[Gen09]	$O(1)G$	$\|w\|\operatorname{poly}(\kappa)G$	$\|C\|\operatorname{poly}(\kappa)M$	$\|C\|\operatorname{poly}(\kappa)M$	lattice + RO
[Gro09]	$O(\|C\|^{1/2})G$	$O(\|C\|^{1/2})G$	$O(\|C\|)M$	$O(\|C\|)M$	DDH + RO
[Gro10]	$\Theta(\|C\|^2)G$	$42G$	$\Theta(\|C\|^2)E$	$\Theta(\|C\|)M + \Theta(1)P$	PKE + CPDH
[Gro10]	$\Theta(\|C\|^{2/3})G$	$\Theta(\|C\|^{2/3})G$	$\Theta(\|C\|^{4/3})E$	$\Theta(\|C\|)M + \Theta(\|C\|^{2/3})P$	PKE + CPDH
This paper	$\|C\|^{1+o(1)}G$	$33G$	$\Theta(\|C\|^2)A + \|C\|^{1+o(1)}E$	$(8\|C\|+4)M + 66P$	PKE + PSDL
This paper	$\|C\|^{1/3+o(1)}G$	$\Theta(\|C\|^{2/3})G$	$\Theta(\|C\|^{4/3})A + \|C\|^{1+o(1)}E$	$\Theta(\|C\|)M + \Theta(\|C\|^{2/3})P$	PKE + PSDL
This paper	$\|C\|^{1/2+o(1)}G$	$\Theta(\|C\|^{1/2})G$	$\Theta(\|C\|^{3/2})A + \|C\|^{1+o(1)}E$	$\Theta(\|C\|)M + \Theta(\|C\|^{1/2})P$	PKE + PSDL

Table 1. Comparison of NIZK arguments with sublinear argument size. Note that the summary length of the CRS and the argument corresponds to the zap length. $|C|$ is the size of circuit, $|w|$ is the length of witness, κ is the security parameter, G is the length of group elements and $A/M/E/P$ corresponds to a single addition/multiplication/exponentiation/pairing

shown in a follow-up work [LZ11] (who construct a NIZK argument for shuffle by using basic arguments for "committed vector is $\mathbf{0}$" and "committed vector has at most 1 non-0 coefficient"), there exist other related basic arguments that make use of progression-free sets, but also in their case the way of choosing and constructing basic arguments is non-trivial.

We hope that subsequent research will help to unify such results to a common framework. While bilinear protocols, as seen both from the current paper and from [LZ11], seem to be specifically suited for using methods from additive combinatorics, we are not aware of any previous work that has made this connection. We do hope that more applications of this connection will be found in the future. Up to our knowledge, the only previous cryptographic work that has explicitly used additive combinatorics is [CLS10] but methods are completely different from ours.

After establishing basic primitives (that do not hopefully only include that basic arguments of the current paper and of [LZ11]), the next step is to design efficient NIZK "programs" for various tasks that make use of those primitives. There are many cryptographic tasks that need tailor-made NIZK arguments. We think it is an interesting research direction to establish firmer foundations for such "programs".

2 Preliminaries

Notation. Let $[n] = \{1, 2, \ldots, n\}$. Let S_n be the set of permutations from $[n]$ to $[n]$. If $y = h^x$, then let $\log_h y := x$. Let κ be the security parameter. We say that $\Lambda = (\lambda_1, \ldots, \lambda_n) \subset \mathbb{Z}$ is an (n, κ)-*nice set*, if $0 < \lambda_1 < \cdots < \lambda_i < \cdots < \lambda_n = \operatorname{poly}(\kappa)$.

Additive combinatorics. By using notation that is common in additive combinatorics [TV06], if Λ_1 and Λ_2 are subsets of some additive group ($\mathbb{Z}$ or $\mathbb{Z}_p$ within this paper), then $\Lambda_1 + \Lambda_2 = \{\lambda_1 + \lambda_2 : \lambda_1 \in \Lambda_1 \wedge \lambda_2 \in \Lambda_2\}$ is their *sum set* and $\Lambda_1 - \Lambda_2 = \{\lambda_1 - \lambda_2 : \lambda_1 \in \Lambda_1 \wedge \lambda_2 \in \Lambda_2\}$ is their *difference set*. If Λ is a set, then $k\Lambda = \{\lambda_1 + \cdots + \lambda_k : \lambda_i \in \Lambda\}$ is an *iterated sumset* while $k \cdot \Lambda = \{k\lambda : \lambda \in \Lambda\}$ is a *dilation* of Λ. We let $2\hat{}\Lambda = \{\lambda_1 + \lambda_2 : \lambda_1 \in \lambda \wedge \lambda_2 \in \Lambda \wedge \lambda_1 \neq \lambda_2\} \subseteq \Lambda + \Lambda$ to denote a *restricted sumset* [ANR95,TV06].

Polynomial factorization. Within our proofs, we need efficient (low-degree polynomial-time) polynomial factorization in $\mathbb{Z}_p$. See [vHN10] for references.

Bilinear groups. Let $\mathcal{G}_{\mathsf{bp}}(1^\kappa)$ be a bilinear group generator that outputs a description of a bilinear group $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, such that p is a κ-bit prime, $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are multiplicative cyclic groups of order p, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map (pairing) such that $\forall a, b \in \mathbb{Z}$ and $g_t \in \mathbb{G}_t$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$. If g_t generates $\mathbb{G}_t$ for $t \in \{1, 2\}$, then $e(g_1, g_2)$ generates $\mathbb{G}_T$. Moreover, deciding the membership in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, group operations, the pairing e, and sampling the generators are efficient. Finally, the descriptions of the groups and group elements each are $O(\kappa)$ bit long. Groth [Gro10] used symmetric pairings with $\mathbb{G}_1 = \mathbb{G}_2$. However, well-chosen asymmetric pairings $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ are much more efficient than the best existing symmetric pairings. As an example, one could implement an optimal (asymmetric) Ate pairing [HSV06] over a subclass of Barreto-Naehrig curves [BN05,PSNB10] very efficiently. In that case, w.l.o.g., an element of $\mathbb{G}_1/\mathbb{G}_2/\mathbb{G}_T$ can be represented in respectively 320/160/1920 bits.

Non-Interactive Zero-Knowledge. Let $\mathcal{R} = \{(C, w)\}$ be an efficiently computable binary relation such that $|w| = \text{poly}(|C|)$. Here, C is a statement, and w is a witness. Let $\mathcal{L} = \{C : \exists w, (C, w) \in \mathcal{R}\}$ be an **NP**-language. Let n be some fixed concrete input length $n = |C|$. For fixed n, we have a relation $\mathcal{R}_n$ and a language $\mathcal{L}_n$. A *non-interactive argument* for $\mathcal{R}$ consists of the next probabilistic polynomial-time algorithms: a common reference string (CRS) generator $\mathcal{G}_{\mathsf{crs}}$, a prover $\mathcal{P}$, and a verifier $\mathcal{V}$. For $\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n)$, $\mathcal{P}(\sigma; C, w)$ produces an argument π. The verifier $\mathcal{V}(\sigma; C, \pi)$ outputs either 1 (accept) or 0 (reject). If the verifier only accesses a small part σ_v of σ, we say that σ_v is the verifier's part of the CRS and we will give just σ_v as an input to $\mathcal{V}$. Sometimes, σ is considerably longer than σ_v. In the case where efficiency is not important (like the security definitions), we give the entire σ to $\mathcal{V}$.

A non-interactive argument $(\mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly complete*, if for all $n = \text{poly}(\kappa)$, all $\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n)$ and all $(C, w) \in \mathcal{R}_n$, $\mathcal{V}(\sigma; C, \mathcal{P}(\sigma; C, w)) = 1$. A non-interactive argument $(\mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *(adaptively) computationally sound*, if for all non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$ and all $n = \text{poly}(\kappa)$, the probability

$$\Pr[\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n), (C, \pi) \leftarrow \mathcal{A}(\sigma) : C \notin \mathcal{L} \wedge \mathcal{V}(\sigma; C, \pi) = 1]$$

is negligible in κ. Here, the soundness is adaptive in the sense that the adversary sees the CRS before producing the statement C. A non-interactive argument $(\mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly witness-indistinguishable*, if for all $n = \text{poly}(\kappa)$, if $\sigma \in \mathcal{G}_{\mathsf{crs}}(1^\kappa, n)$ and $((C, w_0), (C, w_1)) \in \mathcal{R}_n^2$, then the distributions $\mathcal{P}(\sigma; C, w_0)$ and $\mathcal{P}(\sigma; C, w_1)$ are equal.

A non-interactive argument $(\mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly zero-knowledge*, if there exists a polynomial-time simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, such that for all stateful interactive non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$ and $n = \text{poly}(\kappa)$,

$$\Pr\begin{bmatrix}\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n), (C, w) \leftarrow \mathcal{A}(\sigma), \\ \pi \leftarrow \mathcal{P}(\sigma; C, w) : \\ (C, w) \in \mathcal{R}_n \wedge \mathcal{A}(\pi) = 1\end{bmatrix} = \Pr\begin{bmatrix}(\sigma; \mathsf{td}) \leftarrow \mathcal{S}_1(1^\kappa, n), \\ (C, w) \leftarrow \mathcal{A}(\sigma), \pi \leftarrow \mathcal{S}_2(\sigma; C, \mathsf{td}) : \\ (C, w) \in \mathcal{R}_n \wedge \mathcal{A}(\pi) = 1\end{bmatrix} .$$

Here, td is the *simulation trapdoor*.

3 Progression-Free Sets

A set of positive integers $\Lambda = \{\lambda_1, \ldots, \lambda_n\}$ is *progression-free* [TV06], if no three elements of Λ are in an arithmetic progression, that is, $\lambda_i + \lambda_j = 2\lambda_k$ only if $i = j = k$, or equivalently, $2\hat{}\Lambda \cap 2 \cdot \Lambda = \emptyset$.

Let $r_3(n)$ denote the cardinality of the largest progression-free set that belongs to $[n]$. For any $n > 1$, the set of integers in $[n]$ that have no ternary digit equal to 2 is progression-free [ET36]. If $n = 3^k$, then there are $2^k - 1$ such integers, and thus $r_3(n) = \Omega(n^{\log_3 2}) = \Omega(n^{0.63})$. Clearly, this set can be efficiently constructed. As shown by Behrend [Beh46], this idea can be generalized to non-ternary bases, with $r_3(n) = \Omega(n/(2^{2\sqrt{2\log_2 n}} \cdot \log_2^{1/4} n))$. Behrend's result was not improved upon for a very long time. In a recent breakthrough, Elkin [Elk08,Elk10] showed that $r_3(n) = \Omega(n \cdot \log_2^{1/4} n / 2^{2\sqrt{2\log_2 n}})$. We have included a proof of Elkin's result in App. B. Our proof is closely based on [GW10] but it has sharper constant inside Ω. Moreover, our proof is much more detailed than that given in [GW10]. While both constructions employ the pigeonhole principle, Elkin's methodology can be used to compute his progression-free set in quasi-linear time $n2^{O(\sqrt{\log n})}$, see [Elk10]. On the other hand, Bourgain [Bou98] showed that $r_3(n) = O(n \cdot (\log n / \log\log n)^{1/2})$, and recently Sanders [San10] showed that $r_3(n) = O(n \cdot (\log\log n)^5 / \log n)$. Thus, according to Behrend and Elkin, the minimal y such that $r_3(y) = n$ is $y = n^{1+o(1)}$, while according to Sanders, $y = \omega(n)$.

We need all members of the progression-free subset also to be odd. For this, one could Elkin's set $\Lambda = \{\lambda_1, \ldots, \lambda_n\} \subset [y]$ of density $y^{1-o(1)}$, and then use the set $2 \cdot \Lambda + 1 = \{2\lambda_1 + 1, \ldots, 2\lambda_n + 1\}$. Clearly, if $\Lambda \in [n^{1+o(1)}]$ then also $2 \cdot \Lambda + 1 \in [n^{1+o(1)}]$. We provide an expository proof of the next result in App. B.

Theorem 1. *Let $r_3^{odd}(N)$ be the size of the largest progression-free set in $[N]$ that only consists of odd numbers. For any fixed N, there exists $y = N^{1+o(1)}$, such that $r_3^{odd}(y) = N$.*

4 Cryptographic Tools

In this section, we generalize the PKE assumption from [Gro10] and then define two new cryptographic assumptions, PDL and PSDL, and prove that the strongest of the last two assumptions, PSDL, is secure in the generic group model. After that we proceed to describe a generalization of Groth's knowledge commitment scheme from [Gro10] and prove that it is computationally binding under the PDL assumption. We note that Groth proved in from [Gro10] that his commitment scheme is computationally binding under the (potentially stronger) CPDH assumption.

Λ-Power Knowledge of Exponent Assumption (Λ-PKE). Abe and Fehr showed in [AF07] that no statistically zero-knowledge non-interactive argument for an **NP**-complete language can have a "direct black-box" security reduction to a standard cryptographic assumption unless $\mathbf{NP} \subseteq \mathbf{P}/\mathrm{poly}$. See [GW11] for related impossibility results. In fact, the soundness of NIZK arguments (for example, an argument that a perfectly hiding commitment scheme commits to 0) is often unfalsifiable by itself. To tackle this obstacle, Groth, Ostrovsky and Sahai [GOS06] (full version, [GOS11]) defined culpable soundness[1] for languages in $\mathbf{NP} \cap \mathbf{coNP}$. However, since we are interested in an **NP**-complete language, we cannot take their approach. Similarly to Groth [Gro10], we will base our NIZK argument for circuit satisfiability on Λ-PKE, an explicit knowledge assumption. This assumption, proposed by Groth [Gro10] (though only for $\Lambda = [n]$), is a generalization of the KEA assumption of Damgård [Dam91] and of the KEA3 assumption of Bellare and Palacio [BP04].

Let $t \in \{1, 2\}$. For two algorithms $\mathcal{A}$ and $X_{\mathcal{A}}$, we write $(y; z) \leftarrow (\mathcal{A}||X_{\mathcal{A}})(x)$ if $\mathcal{A}$ on input x outputs y, and $X_{\mathcal{A}}$ on the same input (including the random tape of $\mathcal{A}$) outputs z. Let Λ be an (n, κ)-nice set for some $n = \mathrm{poly}(\kappa)$. The bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is *Λ-PKE secure in group $\mathbb{G}_t$* if for any non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ there exists a non-uniform probabilistic polynomial-time extractor $X_{\mathcal{A}}$, such that

$$\Pr\left[\begin{array}{l}\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), g_t \leftarrow \mathbb{G}_t \setminus \{1\}, (\bar{\alpha}, x) \leftarrow \mathbb{Z}_p^2, \sigma \leftarrow (\mathsf{gk}; (g_t^{x^i}, g_t^{\bar{\alpha}x^i})_{i\in\{0\}\cup\Lambda}), \\ (c, \bar{c}; r, (a_i)_{i\in\Lambda}) \leftarrow (\mathcal{A}||X_{\mathcal{A}})(\sigma) : \bar{c} = c^{\bar{\alpha}} \wedge c \neq g_t^r \cdot \prod_{i\in\Lambda} g_t^{a_i x^i}\end{array}\right]$$

is negligible in κ. That is, if $\mathcal{A}$ (given access to σ that for a random $\bar{\alpha}$ contains $g_t^{x^i}$ and $g_t^{\bar{\alpha}x^i}$ only if $i \in \{0\} \cup \Lambda$) can produce c and $\bar{c}$ such that $\bar{c} = c^{\bar{\alpha}}$, then $X_{\mathcal{A}}$ (given access to σ and to the random coins of $\mathcal{A}$) can produce a tuple $(r, (a_i)_{i\in\Lambda})$ such that $c = g_t^r \cdot \prod_{i\in\Lambda} g_t^{a_i x^i}$. Groth [Gro10] proved that the Λ-PKE assumption holds in the generic group model in the case $\lambda_i = i$; his proof can be straightforwardly modified to the general case.

Λ-Power (Symmetric) Discrete Logarithm Assumption. Let Λ be an (n, κ)-nice set for some $n = \mathrm{poly}(\kappa)$. We say that a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is *(n, κ)-PDL secure in group $\mathbb{G}_t$* for $t \in \{1, 2\}$, if for any non-uniform probabilistic polynomial-time adversary $\mathcal{A}$,

$$\Pr[\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), g_t \leftarrow \mathbb{G}_t \setminus \{1\}, x \leftarrow \mathbb{Z}_p : \mathcal{A}(\mathsf{gk}; (g_t^{x^i})_{i\in\{0\}\cup\Lambda}) = x]$$

is negligible in κ. Similarly, we say that a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is *Λ-PSDL secure*, if for any non-uniform probabilistic polynomial-time adversary $\mathcal{A}$,

$$\Pr[\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}, g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}, x \leftarrow \mathbb{Z}_p : \mathcal{A}(\mathsf{gk}; (g_1^{x^i}, g_2^{x^i})_{i\in\{0\}\cup\Lambda}) = x]$$

is negligible in κ. A version of P(S)DL assumption in a non pairing-based group was defined in [GJM02]. Clearly, if the Λ-PSDL assumption is hard, then the Λ-PDL assumption is hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$. Moreover, if the bilinear group generator is CPDH secure, then it is also P(S)DL secure. Therefore, by the results of [Gro10], P(S)DL holds in the generic group model. We give a direct proof of the next result in App. D.

Theorem 2. *The Λ-PSDL assumption holds in the generic group model for any (n, κ)-nice set Λ given that $n = \mathrm{poly}(\kappa)$. Any successful adversary for Λ-PSDL requires time $\Omega(\sqrt{p/\lambda_n})$ where λ_n is the largest element of Λ.*

[1] Originally named co-soundness. The current version of [GOS11] uses the name culpable soundness, since the notion of co-soundness already has several different meanings. The definition of this security property has remained the same as in say [GL07].

Trapdoor Commitment Schemes in the CRS Model. A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ in a bilinear group consists of two probabilistic polynomial-time algorithms: a randomized CRS generation algorithm $\mathcal{G}_{\mathsf{com}}$, and a randomized commitment algorithm $\mathcal{C}\mathrm{om}$. Here, $\mathcal{G}^t_{\mathsf{com}}(1^\kappa)$, $t \in \{1,2\}$, produces a CRS ck_t, and $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r)$ outputs a commitment value A in $\mathbb{G}_t$ (or in $\mathbb{G}_t^b$ for some $b > 1$). We open $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r)$ by outputting $\boldsymbol{a}$ and r.

A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ is *computationally binding in group* $\mathbb{G}_t$, if for every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$, the probability

$$\Pr[\mathsf{ck}_t \leftarrow \mathcal{G}^t_{\mathsf{com}}(1^\kappa), (\boldsymbol{a_1}, r_1, \boldsymbol{a_2}, r_2) \leftarrow \mathcal{A}(\mathsf{ck}_t) : (\boldsymbol{a_1}, r_1) \neq (\boldsymbol{a_2}, r_2) \wedge \mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a_1}; r_1) = \mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a_2}; r_2)]$$

is negligible in κ. A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ is *perfectly hiding in group* $\mathbb{G}_t$, if for any $\mathsf{ck}_t \in \mathcal{G}^t_{\mathsf{com}}(1^\kappa)$ and any two messages a_1, a_2, the distributions $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; a_1; \cdot)$ and $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; a_2; \cdot)$ are equal. We use the next variant of the *knowledge commitment scheme* from [Gro10] with a generalized choice of generators, defined as follows:

CRS generation: Let Λ be an (n, κ)-nice set with $n = \mathrm{poly}(\kappa)$. Define $\lambda_0 = 0$. Given a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$, set $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $g_1 \leftarrow \mathbb{G}_1 \backslash \{1\}$, $g_2 \leftarrow \mathbb{G}_2 \backslash \{1\}$, and $\bar{\alpha}, x \leftarrow \mathbb{Z}_p$. Let $t \in \{1,2\}$. The CRS is $\mathsf{ck}_t \leftarrow (\mathsf{gk}; (g_{ti}, \bar{g}_{ti})_{i \in \{0,\ldots,n\}})$, where $g_{ti} = g_t^{x^{\lambda_i}}$, and $\bar{g}_{ti} = g_t^{\bar{\alpha} x^{\lambda_i}}$. Note that $g_t = g_{t0}$.

Commitment: To commit to $\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_p^n$ in group $\mathbb{G}_t$, the committing party chooses a random $r \leftarrow \mathbb{Z}_p$, and defines $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r) := (g_t^r \cdot \prod_{i=1}^n g_{ti}^{a_i},\ \bar{g}_t^r \cdot \prod_{i=1}^n \bar{g}_{ti}^{a_i})$.

Theorem 3. *Let $t \in \{1,2\}$. The knowledge commitment scheme is perfectly hiding in $\mathbb{G}_t$, and computationally binding in $\mathbb{G}_t$ under the Λ-PDL assumption in $\mathbb{G}_t$. If the Λ-PKE assumption holds in $\mathbb{G}_t$, then for any non-uniform probabilistic polynomial-time committer $\mathcal{A}$ there exists a non-uniform probabilistic polynomial-time extractor $X_{\mathcal{A}}$ that, given the input of $\mathcal{A}$ together with $\mathcal{A}$'s random coins, extracts the contents of the commitments.*

The proof of this theorem is given in App. E. Note that here and in the case of all other security reductions in this paper, the tightness of the security reduction depends on the value λ_n.

A trapdoor commitment scheme has three additional efficient algorithms, for trapdoor CRS generation $\mathsf{TdG}(1^\kappa, t) = (\sigma^*, \mathsf{td})$ with σ^* having the same distribution as $\mathcal{G}^t_{\mathsf{com}}(1^\kappa)$, randomized trapdoor commitment $\mathsf{TdCom}^t(\sigma^*, \mathsf{td}; r) = \mathcal{C}\mathrm{om}^t(\sigma^*; 0; r)$, and trapdoor opening $\mathsf{TdOpen}(\sigma^*; a; r) = r'$ such that $\mathcal{C}\mathrm{om}^t(\sigma^*; 0; r) = \mathcal{C}\mathrm{om}^t(\sigma^*; a; r')$. Clearly, the knowledge commitment scheme is also perfectly trapdoor, with the trapdoor being $\mathsf{td} = x$: after trapdoor-committing $A \leftarrow \mathcal{C}\mathrm{om}^t(\mathsf{ck}; \mathbf{0}; r) = g_t^r$ for $r \leftarrow \mathbb{Z}_p$, the committer can open it to any $(\boldsymbol{a}; r')$ where r' is chosen so that $r' + \sum a_i x^i = r$.

Adaptive $\mathcal{R}_{\mathsf{guilt}}$-Soundness. To avoid knowledge assumptions, one can alternatively relax the notion of soundness. Following [GOS06] (the full version [GOS11]) and [GL07], $\mathcal{R}_{\mathsf{guilt}}$-soundness is a weaker version of soundness, where it is required that an adversary who *knows* that $C \notin \mathcal{L}$ should not be able to produce a witness w_{guilt} such that $(C, w_{\mathsf{guilt}}) \in \mathcal{R}_{\mathsf{guilt}}$ (see [GL07,GOS11] for a longer explanation). More formally, let $\mathcal{R} = \{(C, w)\}$ and $\mathcal{L} = \{C : (\exists w)(C, w) \in \mathcal{R}\}$ be defined as earlier. Let $\mathcal{R}_{\mathsf{guilt}} = \{(C, w_{\mathsf{guilt}})\}$ be an efficiently computable binary relation with $|w| = \mathrm{poly}(|C|)$. A non-interactive argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is (adaptively) $\mathcal{R}_{\mathsf{guilt}}$*-sound*, if for all non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$, the probability

$$\Pr[\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa), (C, w_{\mathsf{guilt}}, \pi) \leftarrow \mathcal{A}(\sigma) : (C, w_{\mathsf{guilt}}) \in \mathcal{R}_{\mathsf{guilt}} \wedge \mathcal{V}(\sigma; C, \pi) = 1]$$

is negligible in κ.

For $\mathcal{R}_{\mathsf{guilt}}$-soundness to make sense for relation $\mathcal{R}$, one obviously has to define $\mathcal{R}_{\mathsf{guilt}}$ carefully. As we will see, the use of knowledge commitments, while making arguments more efficient, makes the definition of $\mathcal{R}_{\mathsf{guilt}}$ more tricky. To understand the difficulty, note that the last property of the knowledge commitment scheme stated by Thm. 3 is needed to prove the soundness of NIZK arguments that are based on this commitment scheme. If we want to restrict ourselves just to $\mathcal{R}_{\mathsf{guilt}}$-soundness (so that we will not have to rely on a knowledge assumption), we have to modify $\mathcal{R}_{\mathsf{guilt}}$ so that the $\mathcal{R}_{\mathsf{guilt}}$-witness contains openings corresponding to all used knowledge pseudo-commitments, where informally, (A, A^α) is a knowledge pseudo-commitment if and only if a secret $\alpha \in \mathbb{Z}_p$ was used while creating some elements $(g_t^{\alpha x^i})_{i \in \Lambda'}$ belonging to the common reference string, where $\Lambda' \subset \mathbb{Z}$. The opening is defined as in the case of the PKE assumption.

This obviously makes the language $\mathcal{R}_{\mathsf{guilt}}$ dependent on the construction of the NIZK argument, and thus seems to give a circular argument. Nevertheless, the next can be seen to be prudent engineering approach. First, we construct an argument *of knowledge* that is sound under the PKE assumption. This argument makes use of knowledge commitments. Second, we eliminate the PKE assumption by requesting the adversary herself (instead of the extractor) to return the openings of the commitments. By doing that, one can argue that the security proof becomes more clear since instead of the general PKE assumption, we rely on the more concrete assumption that an adversary who knows how to open *certain* commitments cannot break the argument.

5 New Hadamard Product Argument

Assume that $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ is a perfectly hiding trapdoor commitment scheme in the CRS model. In a *Hadamard product argument*, the prover aims to convince the verifier that for commitments A, B and C, he knows how to open them as $A = \mathcal{C}\mathrm{om}^1(\mathsf{ck}; \boldsymbol{a}; r_a)$, $B = \mathcal{C}\mathrm{om}^1(\mathsf{ck}; \boldsymbol{b}; r_b)$, and $C = \mathcal{C}\mathrm{om}^1(\mathsf{ck}; \boldsymbol{c}; r_c)$, such that $c_j = a_j \cdot b_j$ for $j \in [n]$. Product argument in $\mathbb{G}_2$ can be defined dually.

Groth constructed a Hadamard product argument [Gro10] with communication $\Theta(1)$, verifier's computation $\Theta(n)$, and prover's computation and the length of the common reference string (CRS) $\Theta(n^2)$. We present a more efficient argument in Prot. 1. Intuitively, the discrete logarithm on basis $h = e(g_1, g_2)$ of the left side of the first verification equation in Prot. 1 is a degree-n formal polynomial in X, "formal" version of x. That formal polynomial is spanned by $\{x^i\}_{i \in 2\cdot\Lambda \cup \bar{\Lambda}}$, where

$$\bar{\Lambda} := \{0\} \cup \Lambda \cup 2\hat{}\Lambda \ . \tag{1}$$

We need that $2 \cdot \Lambda$ and $\bar{\Lambda}$ do not intersect. The next lemma is straightforward to prove.

Lemma 1. *1) If Λ is a progression-free set of odd integers, then $2 \cdot \Lambda \cap \bar{\Lambda} = \emptyset$. 2) If $2 \cdot \Lambda \cap \bar{\Lambda} = \emptyset$, then Λ is a progression-free set.*

Moreover, since $\bar{\Lambda} \in \{0, \ldots, 2\lambda_n\}$, then by Thm. 1,

Lemma 2. *For any value n there exists a choice of Λ such that $|\bar{\Lambda}| = n^{1+o(1)}$.*

We are now ready to prove the security of the new Hadamard product argument for the knowledge commitment scheme. The (knowledge) commitments are $(A, \bar{A})$, $(B, \bar{B})$ and $(C, \bar{C})$. For efficiency reasons, we include another element B_2 to the Hadamard product language. We denote the argument in Prot. 1 by $[\![(A, \bar{A})]\!] \circ [\![(B, \bar{B}, B_2)]\!] = [\![(C, \bar{C})]\!]$. Since $(C, \bar{C})$ is always a commitment of $(a_1 b_1, \ldots, a_n b_n)$ for *some* value of r_c, we cannot claim that Prot. 1 is computationally sound (even under a knowledge assumption). Instead, we prove $\mathcal{R}^{\times}_{\mathsf{guilt}}$-soundness of this argument, where

$$\mathcal{R}^{\times}_{\mathsf{guilt}} := \left\{ \begin{array}{l} (\sigma; (A, \bar{A}, B, \bar{B}, B_2, C, \bar{C}), w^{\times}_{\mathsf{guilt}} = (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, \boldsymbol{c}, r_c, (f'_i)_{i \in \bar{\Lambda}}), (\pi, \bar{\pi})) : \\ (A, \bar{A}) = \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{a}; r_a) \wedge (B, \bar{B}) = \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{b}; r_b) \wedge B_2 = g_2^{r_b} \cdot \prod_{i=1}^{n} g_{2i}^{b_i} \wedge \\ (C, \bar{C}) = \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{c}; r_c) \wedge (\exists i \in [n])(a_i b_i \neq c_i) \wedge (\pi, \bar{\pi}) = (g_2^{\sum_{i \in \bar{\Lambda}} f'_i x^i}, \bar{g}_2^{\sum_{i \in \bar{\Lambda}} f'_i x^i}) \wedge \\ \mathcal{V}_{\times}(\sigma; (A, \bar{A}, B, \bar{B}, B_2, C, \bar{C}), (\pi, \bar{\pi})) = 1 \end{array} \right\} .$$

The definition of $\mathcal{R}^{\times}_{\mathsf{guilt}}$ is quite natural, except the part of f'_i that is needed because of our proof technique. Importantly, $\mathcal{R}^{\times}_{\mathsf{guilt}}$-soundness of the product argument is sufficient for us to later prove computational soundness of the circuit satisfiability argument. Groth [Gro10] proves culpable soundness of his basic arguments with respect to similar languages. Intuitively, $\mathcal{R}^{\times}_{\mathsf{guilt}}$ includes only f'_i for $i \in \bar{\Lambda}$ since $\bar{g}_{2i}$ belongs to the CRS only for $i \in \bar{\Lambda}$, and $a_i/b_i/c_i$ for $i \in [n]$ since $\bar{g}_{1i} = \bar{g}_1^{x^{\lambda_i}}$ belongs to the CRS only for $i \in \Lambda$. One can add additional elements g_{2i} (without corresponding elements $\bar{g}_{2i}$) to the CRS, without damaging the security.

Theorem 4. *The Hadamard product argument in Prot. 1 is perfectly complete and perfectly witness-indistinguishable. Assume that the verifier additionally checks the correctness of $(A, \bar{A})$, $(B, \bar{B}, B_2)$ and $(C, \bar{C})$ by verifying that $e(A, \bar{g}_2) = e(\bar{A}, g_2)$, $e(\bar{g}_1, B) = e(g_1, \bar{B})$, $e(g_1, B_2) = e(B, g_2)$, and $e(C, \bar{g}_2) = e(\bar{C}, g_2)$. If the bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is $\bar{\Lambda}$-PSDL secure, then Prot. 1 is adaptively $\mathcal{R}^{\times}_{\mathsf{guilt}}$-sound.*

System parameters: Let $n = \mathrm{poly}(\kappa)$. Let $\Lambda = \{\lambda_i : i \in [n]\}$ be a progression-free set of odd integers, such that $\lambda_{i+1} > \lambda_i > 0$. Denote $\lambda_0 := 0$. Let $\bar{\Lambda}$ be as in Eq. (1).

CRS generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $\alpha, x \leftarrow \mathbb{Z}_p$. Let $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$ and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. Denote $g_{ti} \leftarrow g_t^{x^i}$ and $\bar{g}_{ti} \leftarrow g_t^{\alpha x^i}$ for $t \in \{1,2\}$ and $i \in \{0\} \cup \bar{\Lambda}$. Let $D \leftarrow \prod_{i=1}^n g_{2,\lambda_i}$. The CRS is $\sigma \leftarrow (\mathsf{gk}; (g_{1i}, \bar{g}_{1i})_{i\in\{0\}\cup\Lambda}, (g_{2i}, \bar{g}_{2i})_{i\in\bar{\Lambda}}, D)$. Set $\mathsf{ck}_t \leftarrow (\mathsf{gk}; (g_{ti}, \bar{g}_{ti})_{i\in\{0\}\cup\Lambda})$ for $t \in \{1,2\}$, and let $\sigma_v \leftarrow (\mathsf{gk}; g_1, \bar{g}_1, g_2, \bar{g}_2, D)$ be the verifier's part of σ.

Common inputs: $(A, \bar{A}, B, \bar{B}, B_2, C, \bar{C})$, where $(A, \bar{A}) \leftarrow \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{a}; r_a)$, $(B, \bar{B}) \leftarrow \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{b}; r_b)$, $B_2 \leftarrow g_2^{r_b} \cdot \prod_{i=1}^n g_{2i}^{b_i}$, $(C, \bar{C}) \leftarrow \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{c}; r_c)$, such that $a_i b_i = c_i$ for $i \in [n]$.

Argument generation $\mathcal{P}_\times(\sigma; (A, \bar{A}, B, \bar{B}, B_2, C, \bar{C}), (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, \boldsymbol{c}, r_c))$**:** The prover defines

$$\mu_k \leftarrow \sum_{\substack{i,j\neq i \\ \lambda_i+\lambda_j=k}} (a_i b_j - c_i)$$

for $k \in 2\hat{}\Lambda$, and then sets $\pi \leftarrow g_{20}^{r_a r_b} \cdot \prod_{i=1}^n g_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{k\in 2\hat{}\Lambda} g_{2k}^{\mu_k}$, and $\bar{\pi} \leftarrow \bar{g}_{20}^{r_a r_b} \cdot \prod_{i=1}^n \bar{g}_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{k\in 2\hat{}\Lambda} \bar{g}_{2k}^{\mu_k}$. The prover sends $\pi^\times \leftarrow (\pi, \bar{\pi}) \in \mathbb{G}_2^2$ to the verifier as the argument.

Verification $\mathcal{V}_\times(\sigma_v; (A, \bar{A}, B, \bar{B}, B_2, C, \bar{C}), \pi^\times)$**:** $\mathcal{V}_\times$ accepts iff $e(A, B_2)/e(C, D) = e(g_1, \pi)$ and $e(g_1, \bar{\pi}) = e(\bar{g}_1, \pi)$.

Protocol 1: New Hadamard product argument $[\![(A, \bar{A})]\!] \circ [\![(B, \bar{B}, B_2)]\!] = [\![(C, \bar{C})]\!]$

Here and in Sect. 6, we do not count the correctness verification as a part of the cost of basic arguments, since it will be done externally in the circuit satisfiability argument. Note that due to the choice of ck_1, the commitment scheme is defined as in Sect. 4 with respect to the set Λ.

Proof. Let $\eta \leftarrow e(A, B_2)/e(C, D)$ and $h \leftarrow e(g_1, g_2)$. Witness-Indistinguishability: follows from the fact that there is exactly one possible argument $\pi^\times = (\pi, \bar{\pi})$ that satisfies the verification equations.

Perfect completeness. The second verification is straightforward. For the first one, note that

$$\begin{aligned}\log_h \eta =& (r_a + \sum_{i=1}^n a_i x^{\lambda_i})(r_b + \sum_{i=1}^n b_i x^{\lambda_i}) - (r_c + \sum_{i=1}^n c_i x^{\lambda_i})(\sum_{i=1}^n x^{\lambda_i}) \\ =& \sum_{i=1}^n (a_i b_i - c_i) x^{2\lambda_i} + r_a r_b + \sum_{i=1}^n (r_a b_i + r_b a_i - r_c) x^{\lambda_i} + \sum_{i=1}^n \sum_{j=1:j\neq i}^n (a_i b_j - c_i) x^{\lambda_i + \lambda_j} \ .\end{aligned}$$

Consider x as a formal variable, then $\log_h \eta$ is a formal polynomial of x. This formal polynomial is spanned by $\{x^i\}_{i\in 2\cdot\Lambda\cup\bar{\Lambda}}$. If the prover is honest, then $c_i = a_i \cdot b_i$ for $i \in [n]$, and thus $\log_h \eta$ is spanned by $\{x^i\}_{i\in\bar{\Lambda}}$. Denoting $h_i \leftarrow h^{x^i}$, $\eta = h^{r_a r_b} \cdot \prod_{i=1}^n h_{\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{i=1}^n \prod_{j=1:j\neq i}^n h_{\lambda_i+\lambda_j}^{a_i b_j - c_i}$. Thus,

$$\pi \leftarrow g_{20}^{r_a r_b} \cdot \prod_{i=1}^n g_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{i=1}^n \prod_{j=1:j\neq i}^n g_{2,\lambda_i+\lambda_j}^{a_i b_j - c_i} = g_{20}^{r_a r_b} \cdot \prod_{i=1}^n g_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{k\in 2\hat{}\Lambda} g_{2k}^{\mu_k} \ ,$$

and $\bar{\pi}$ is computed similarly. Thus, if the prover is honest, then the first verification succeeds.

Adaptive $\mathcal{R}^\times_{\mathsf{guilt}}$-soundness. Assume that $\mathcal{A}$ is an adversary that can break the adaptive $\mathcal{R}^\times_{\mathsf{guilt}}$-soundness of the Hadamard product argument. Next, we construct an adversary $\mathcal{A}'$ against the $\bar{\Lambda}$-PSDL assumption. Let $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $x \leftarrow \mathbb{Z}_p$, $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$, and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. The adversary $\mathcal{A}'$ receives $\sigma \leftarrow (\mathsf{gk}; (g_1^{x^i}, g_2^{x^i})_{i\in\bar{\Lambda}})$ as her input, and her task is to output x. She sets $\alpha \leftarrow \mathbb{Z}_p$, $\sigma' \leftarrow (\mathsf{gk}; (g_1^{x^i}, g_1^{\alpha x^i})_{i\in\{0\}\cup\Lambda}, (g_2^{x^i}, g_2^{\alpha x^i})_{i\in\bar{\Lambda}}, \prod_{i=1}^n g_2^{x^{\lambda_i}})$, and then forwards σ' to $\mathcal{A}$. Clearly, σ' follows the distribution imposed by $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$. Both $\mathcal{A}$ and $\mathcal{A}'$ set $\mathsf{ck}_t \leftarrow (\mathsf{gk}; (g_t^{x^i}, g_t^{\alpha x^i})_{i\in\{0\}\cup\Lambda})$ for $t \in \{1,2\}$. Assume that $\mathcal{A}$ returns $(inp^\times, w^\times_{\mathsf{guilt}}, \pi^\times)$ such that $\mathcal{V}(\sigma'; inp^\times, \pi^\times)$ accepts. Here, $inp^\times = (A, \bar{A}, B, \bar{B}, B_2, C, \bar{C})$ and $w^\times_{\mathsf{guilt}} = (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, \boldsymbol{c}, r_c, (f'_i)_{i\in\bar{\Lambda}})$.

Whenever $\mathcal{A}$ is successful, $(A, \bar{A}) = \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{a}; r_a)$, $(B, \bar{B}) = \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{b}; r_b)$, $B_2 = g_2^{r_b} \cdot \prod_{i=1}^n g_{2i}^{b_i}$, $(C, \bar{C}) = \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{c}; r_c)$, and for some $i \in [n]$, $c_i \neq a_i b_i$. Since $2 \cdot \Lambda \cap \bar{\Lambda} = \emptyset$, $\mathcal{A}'$ has thus expressed $\log_h \eta$ as a polynomial $f(X)$ where at least for some $i \in 2 \cdot \Lambda$, X^i has a non-zero coefficient.

On the other hand, $\mathcal{A}$ also outputs $(f'_i)_{i\in\bar{\Lambda}}$ such that $\log_h \eta = \log_{g_2} \pi = f'(x) = \sum_{i\in\bar{\Lambda}} f'_i x^i$, where all non-zero coefficients of $f'(X)$ correspond to X^i for some $i \in \bar{\Lambda}$. Since Λ is a progression-free set of odd

integers and all elements of $2\cdot\Lambda$ are distinct, then by Lem. 1, $i \notin 2\cdot\Lambda$. Therefore, all coefficients of $f'(X)$ corresponding to any X^i, $i \in 2\cdot\Lambda$, are equal to 0. Thus $f(X) = \sum_{i\in\bar{\Lambda}\cup(2\cdot\Lambda)} f_i X^i$ and $f'(X) = \sum_{i\in\bar{\Lambda}} f'_i X^i$ are different polynomials with $f(x) = f'(x) = \log_h \eta$. Therefore, $\mathcal{A}'$ has succeeded in creating a non-zero polynomial $d(X) = f(X) - f'(X)$, such that $d(X) = \sum_{i\in\bar{\Lambda}\cup(2\cdot\Lambda)} d_i X^i = 0$.

Next, $\mathcal{A}'$ can use an efficient polynomial factorization algorithm in $\mathbb{Z}_p[X]$ to efficiently compute all $< 2\lambda_n + 1$ roots of $d(X)$. For some root y, $g_1^{x^i} = g_1^{y^i}$. $\mathcal{A}'$ sets $x \leftarrow y$, thus violating the $\bar{\Lambda}$-PSDL assumption. □

Note that the tightness of the reduction depends on the value λ_n.

Theorem 5. *Let Λ be as described in Thm. 1. The communication (argument size) of Prot. 1 is two elements from $\mathbb{G}_2$. The prover's computational complexity is $\Theta(n^2)$ multiplications in $\mathbb{Z}_p$ and $n^{1+o(1)}$ exponentiations in $\mathbb{G}_2$. The verifier's computational complexity is dominated by 5 bilinear pairings. The CRS consists of $n^{1+o(1)}$ group elements, with the verifier's part of the CRS consisting of only the bilinear group description plus 5 group elements.*

The proof is given in App. F.

In the circuit satisfiability argument, all values a_i, b_i and c_i are Boolean, and thus all $n^{1+o(1)}$ values μ_k can be computed in $n(n-1) = \Theta(n^2)$ additions, by using the next simple subroutine (the server obviously also needs to use other operations like comparisons $j \neq i$, but they can be eliminated by using loop unrolling, and λ_i and λ_j can be computed by using table lookups):

1. For $k \in 2\hat{}\Lambda$ do: $\mu_k \leftarrow 0$
2. For $i = 1$ to n do:
   - If $a_i = 0$ then for $j = 1$ to n do: if $j \neq i$ then $\mu_{\lambda_i+\lambda_j} \leftarrow \mu_{\lambda_i+\lambda_j} - c_i$
   - Else for $j = 1$ to n do: if $j \neq i$ then $\mu_{\lambda_i+\lambda_j} \leftarrow \mu_{\lambda_i+\lambda_j} + b_j - c_i$

6 New Permutation Argument

In a *permutation argument*, the prover aims to convince the verifier that for given permutation $\varrho \in S_n$ and two commitments A and B, he knows how to open them as $A = \mathcal{C}\mathrm{om}^1(\mathsf{ck}; \boldsymbol{a}; r_a)$ and $B = \mathcal{C}\mathrm{om}^1(\mathsf{ck}; \boldsymbol{b}; r_b)$, such that $b_j = a_{\varrho(j)}$ for $j \in [n]$. In some cases, ϱ is a common (statement-dependent) input while in some cases ϱ does not dependent on the statement. The next argument assumes that ϱ is a part of the statement. In [Gro10], Groth constructed a permutation argument where the communication is $\Theta(1)$, the verifier's computation is $\Theta(n)$, and the prover's computation and the length of the CRS are $\Theta(n^2)$. In this section, we propose a new argument with the CRS of length $n^{1+o(1)}$. In addition, prover's concrete computation will be improved, and the argument is based on a (probably) weaker assumption.

The proposed permutation argument $\varrho([\![(A, \tilde{A})]\!]) = [\![(B, \tilde{B})]\!]$, see Prot. 2, is constructed by using (almost) the same high-level ideas as the Hadamard product argument from Sect. 5. However, the situation is more complicated in this case. To see that, consider again a verification equation of type $e(A, D)/e(B, E) = e(g_1, \pi^{\varrho})$ where $D = g_2^{\prod_{i=1}^n x^{\varphi_i}}$ and $E = g_2^{\prod_{i=1}^n x^{\psi_i}}$ for some φ_i and ψ_i. Letting $h = e(g_1, g_2)$, we get

$$\begin{aligned}
\log_h \eta_{\varrho} =& (r_a + \sum_{i=1}^n a_i x^{\lambda_i})(\sum_{i=1}^n x^{\varphi_i}) - (r_b + \sum_{i=1}^n b_i x^{\lambda_i})(\sum_{i=1}^n x^{\psi_i}) \\
=& r_a \sum_{i=1}^n x^{\varphi_i} - r_b \sum_{i=1}^n x^{\psi_i} + \sum_{i=1}^n \sum_{j=1}^n a_{\varrho(i)} x^{\lambda_{\varrho(i)}+\varphi_{\varrho(j)}} - \sum_{i=1}^n \sum_{j=1}^n b_i x^{\lambda_i+\psi_j} \\
=& r_a \sum_{i=1}^n x^{\varphi_i} - r_b \sum_{i=1}^n x^{\psi_i} + \sum_{i=1}^n a_{\varrho(i)} x^{\lambda_{\varrho(i)}+\varphi_{\varrho(i)}} - \sum_{i=1}^n b_i x^{\lambda_i+\psi_i} + \sum_{i=1}^n \sum_{\substack{j=1 \\ j\neq i}}^n a_{\varrho(i)} x^{\lambda_{\varrho(i)}+\varphi_{\varrho(j)}} - \\
& \sum_{i=1}^n \sum_{\substack{j=1 \\ j\neq i}}^n b_i x^{\lambda_i+\psi_j} \ .
\end{aligned}$$

Since we are interested in the case $a_{\varrho(i)} = b_i$, it is natural to define

$$\psi_i := \lambda_{\varrho(i)} + \varphi_{\varrho(i)} - \lambda_i \ . \tag{2}$$

Then,

$$\log_h \eta_\varrho = \sum_{i=1}^{n}(a_{\varrho(i)} - b_i)x^{\lambda_{\varrho(i)}+\varphi_{\varrho(i)}} + r_a\sum_{i=1}^{n} x^{\varphi_i} - r_b\sum_{i=1}^{n} x^{\lambda_{\varrho(i)}+\varphi_{\varrho(i)}-\lambda_i} + \sum_{i=1}^{n}\sum_{j=1:j\neq i}^{n} a_{\varrho(i)}x^{\lambda_{\varrho(i)}+\varphi_{\varrho(j)}} - \sum_{i=1}^{n}\sum_{j=1:j\neq i}^{n} b_i x^{\lambda_i+\lambda_{\varrho(j)}+\varphi_{\varrho(j)}-\lambda_j} \ .$$

If one wants to use the same methodology as in Sect. 5, one has to define φ_i so that $\widetilde{\Lambda} := \{\varphi_i\} \cup \{\lambda_{\varrho(i)} + \varphi_{\varrho(i)} - \lambda_i\} \cup \{\lambda_{\varrho(i)} + \varphi_{\varrho(j)} : i \neq j\} \cup \{\lambda_i + \lambda_{\varrho(j)} + \varphi_{\varrho(j)} - \lambda_j : i \neq j\}$ and $\{\lambda_{\varrho(i)} + \varphi_{\varrho(i)}\}$ will not intersect. To this end, it is natural to define $\varphi_i = k\lambda_i + \ell\lambda_{\varrho^{-1}(i)}$ for some k and ℓ. Most of those linear combinations do not make the task simpler, and thus k and ℓ should be chosen so that at least some of the terms in $\widetilde{\Lambda}$ would simplify. For example[2], one can choose

$$\varphi_i = \lambda_i \ , \tag{3}$$

in this case we get

$$\log_h \eta_\varrho = \sum_{i=1}^{n}(a_{\varrho(i)} - b_i)x^{2\lambda_{\varrho(i)}} + r_a\sum_{i=1}^{n} x^{\lambda_i} - r_b\sum_{i=1}^{n} x^{2\lambda_{\varrho(i)}-\lambda_i} + \sum_{i=1}^{n} a_{\varrho(i)} \cdot \sum_{j=1:j\neq i}^{n} x^{\lambda_{\varrho(i)}+\lambda_{\varrho(j)}} - \sum_{i=1}^{n} b_i \cdot \sum_{j=1:j\neq i}^{n} x^{\lambda_i+2\lambda_{\varrho(j)}-\lambda_j} \ .$$

In this case, $\widetilde{\Lambda} = \Lambda \cup \{2\lambda_{\varrho(i)} - \lambda_i\} \cup 2\hat{}\Lambda \cup \{\lambda_i + 2\lambda_{\varrho(j)} - \lambda_j : i \neq j\}$ and the second set is $2 \cdot \Lambda = \{2\lambda_{\varrho(i)}\}$. Since ϱ is unknown, we have to replace $\varrho(i)$ and $\varrho(j)$ with an arbitrary new element k, so $\widetilde{\Lambda} = \Lambda \cup \{2\lambda_k - \lambda_i\} \cup 2\hat{}\Lambda \cup \{\lambda_i + 2\lambda_k - \lambda_j : i \neq j\}$.

If we assume that Λ consists of odd integers, then $(\Lambda \cup \{2\lambda_k - \lambda_i\}) \cap 2 \cdot \Lambda = \emptyset$. If we assume that Λ is a progression-free set, then $2\hat{}\Lambda \cap \Lambda = \emptyset$. However, in the fourth case, we need that $2\lambda_\ell \neq 2\lambda_k + \lambda_i - \lambda_j$ for $i \neq j$. That is, one can uniquely represent any non-negative integer a as $a = 2\lambda_\ell + \lambda_j$. As it is known from [Mos62,dB64], the unique sequence $\Lambda = (\lambda_i)_{i\in\mathbb{Z}}$ (the *Moser-de Bruijn sequence*) that satisfies this property is the sequence of all non-negative integers that have only 0 or 1 as their radix-4 digits. Since $\lambda_n = \Theta(n^2)$, this sequence is not good enough for our purposes. Fortunately, we can overcome this problem as follows. For $i \in [n]$ and permutation ϱ, let $T(i, \varrho) := |\{j \in [n] : 2\lambda_{\varrho(i)} + \lambda_j = 2\lambda_{\varrho(j)} + \lambda_i\}|$, note that $T(i, \varrho) \geq 1$. We can then rewrite $\log_h \eta_\varrho$ as

$$\log \eta_\varrho = \sum_{i=1}^{n}(a_{\varrho(i)} - b_i \cdot T(i,\varrho))x^{2\lambda_{\varrho(i)}} + r_a\sum_{i=1}^{n} x^{\lambda_i} - r_b\sum_{i=1}^{n} x^{2\lambda_{\varrho(i)}-\lambda_i} + \sum_{i=1}^{n} a_{\varrho(i)}\sum_{\substack{j=1\\ j\neq i}}^{n} x^{\lambda_{\varrho(i)}+\lambda_{\varrho(j)}} - \sum_{i=1}^{n} b_i \sum_{\substack{j=1\\ j\neq i\\ 2\lambda_{\varrho(i)}+\lambda_j\neq\lambda_i+2\lambda_{\varrho(j)}}}^{n} x^{\lambda_i+2\lambda_{\varrho(j)}-\lambda_j} \ , \tag{4}$$

with $\widetilde{\Lambda}$ being redefined as

$$\widetilde{\Lambda} = \Lambda \cup \{2\lambda_k - \lambda_i\} \cup 2\hat{}\Lambda \cup (\{\lambda_i + 2\lambda_k - \lambda_j : i \neq j\} \setminus 2 \cdot \Lambda) \ . \tag{5}$$

Since $\widetilde{\Lambda} \cap 2 \cdot \Lambda = \emptyset$, verification of $e(A, D)/e(B, E) = e(g_1, \pi)$ (with $D = g_2^{\prod_{i=1}^{n} x^{\lambda_i}}$ and $E = E_\varrho = g_2^{\prod_{i=1}^{n} x^{2\lambda_{\varrho(i)}-\lambda_i}}$, as it follows from Eq. (2) and (3)) convinces the verifier that $a_{\varrho(i)} = T(i, \varrho) \cdot b_i$ for $i \in [n]$.

[2] We tried many other choices of φ_i, but none of them gave a simpler solution. Details are omitted due to the lack of space.

System parameters: Let $n = \text{poly}(\kappa)$. Let $\Lambda = \{\lambda_i : i \in [n]\}$ be a progression-free set of odd integers, such that $\lambda_{i+1} > \lambda_i > 0$. Denote $\lambda_0 := 0$. Let $\widetilde{\Lambda}$ be as in Eq. (5).

CRS generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $\widetilde{\alpha}, x \leftarrow \mathbb{Z}_p$. Let $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$ and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. Denote $g_{ti} \leftarrow g_t^{x^i}$ and $\tilde{g}_{ti} \leftarrow g_t^{\widetilde{\alpha} x^i}$ for $t \in \{1,2\}$ and $i \in \{-\lambda_n + 1, \ldots, 3\lambda_n\}$. Let $D \leftarrow \prod_{i=1}^n g_{2,\lambda_i}$, $\tilde{D} \leftarrow \prod_{i=1}^n \tilde{g}_{2,\lambda_i}$. The CRS is equal to $\sigma \leftarrow (\mathsf{gk}; (g_{1i}, \tilde{g}_{1i})_{i \in \{0\} \cup \Lambda}, (g_{2i}, \tilde{g}_{2i})_{i \in \{0\} \cup \widetilde{\Lambda}}, D, \tilde{D})$. Let $\mathsf{ck}_t \leftarrow (\mathsf{gk}; (g_{ti}, \tilde{g}_{ti})_{i \in \{0\} \cup \Lambda})$ for $t \in \{1,2\}$, and let $\sigma_v \leftarrow (\mathsf{gk}; g_1, \tilde{g}_1, D, (g_{2,2\lambda_i - \lambda_j})_{i,j \in [n]})$ be the verifier's part of σ.

Common inputs: $(A, \tilde{A}, B, \tilde{B})$, where $(A, \tilde{A}) \leftarrow \mathcal{C}\text{om}^1(\mathsf{ck}_1; \boldsymbol{a}; r_a)$, $(B, \tilde{B}) \leftarrow \mathcal{C}\text{om}^1(\mathsf{ck}_1; \boldsymbol{b}; r_b)$, and a permutation $\varrho \in S_n$, such that $b_j = a_{\varrho(j)}$ for $j \in [n]$.

Argument generation $\mathcal{P}_{perm}(\sigma; (A, \tilde{A}, B, \tilde{B}, \varrho), (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b))$**:** Do the following for $\psi \in \{\varrho, \varrho^{-1}\}$:

1. Let $\widetilde{\Lambda}'_\psi := 2\hat{\Lambda} \cup \{2\lambda_{\psi(j)} + \lambda_i - \lambda_j : i, j \in [n] \wedge i \neq j\} \subset \{-\lambda_n + 1, \ldots, 3\lambda_n\}$. Set
$$\mu_{\psi,k} \leftarrow \sum_{\substack{i,j \neq i \\ \lambda_i + \lambda_j = k}} a_i + \sum_{\substack{i,j \neq i \\ 2\lambda_{\psi(j)} + \lambda_i - \lambda_j = k}} b_i \quad \text{for } k \in \widetilde{\Lambda}'_\psi \ .$$
2. Let $E_\psi \leftarrow \prod_{i=1}^n g_{2,2\lambda_{\psi(i)} - \lambda_i}$ and $\tilde{E}_\psi \leftarrow \prod_{i=1}^n \tilde{g}_{2,2\lambda_{\psi(i)} - \lambda_i}$. Set $\pi^\psi \leftarrow D^{r_a} \cdot E_\psi^{r_b} \cdot \prod_{k \in \widetilde{\Lambda}'_\psi} g_{2,k}^{\mu_{\psi,k}}$, $\widetilde{\pi}^\psi \leftarrow \tilde{D}^{r_a} \cdot \tilde{E}_\psi^{r_b} \cdot \prod_{k \in \widetilde{\Lambda}'_\psi} \tilde{g}_{2,k}^{\mu_{\psi,k}}$,

Send $\pi^{perm} \leftarrow (\pi^\varrho, \widetilde{\pi}^\varrho, \pi^{\varrho^{-1}}, \widetilde{\pi}^{\varrho^{-1}}) \in \mathbb{G}_4$ to the verifier as the argument.

Verification $\mathcal{V}_{perm}(\sigma_v; (A, \tilde{A}, B, \tilde{B}, \varrho), \pi^{perm})$**:** Let $E_\varrho \leftarrow \prod_{i=1}^n g_{2,2\lambda_{\varrho(i)} - \lambda_i}$ and $E_{\varrho^{-1}} \leftarrow \prod_{i=1}^n g_{2,2\lambda_{\varrho^{-1}(i)} - \lambda_i}$.

If $e(A, D)/e(B, E_\varrho) = e(g_1, \pi^\varrho)$, $e(g_1, \widetilde{\pi}^\varrho) = e(\tilde{g}_1, \pi^\varrho)$, $e(B, D)/e(A, E_{\varrho^{-1}}) = e(g_1, \pi^{\varrho^{-1}})$ and $e(g_1, \widetilde{\pi}^{\varrho^{-1}}) = e(\tilde{g}_1, \pi^{\varrho^{-1}})$, then $\mathcal{V}_{perm}$ accepts. Otherwise, $\mathcal{V}_{perm}$ rejects.

Protocol 2: New permutation argument $\varrho([\![(A, \tilde{A})]\!]) = [\![(B, \tilde{B})]\!]$

While we do not know the value of $T(i, \varrho)$, since $T(i, \varrho) \geq 1$ the verifier is convinced that $a_{\varrho(i)} \geq b_i$ for $i \in [n]$. All that is now left to do is to execute the same argument second time but with changing the roles of A and B and for permutation ϱ^{-1} (and a different value of $E = E_{\varrho^{-1}}$). Those two arguments together show that $a_{\varrho(i)} \geq b_i$ and $a_{\varrho(i)} \leq b_i$, and thus $a_{\varrho(i)} = b_i$ for $i \in [n]$. Since $\widetilde{\Lambda} \subset \{-\lambda_n + 1, \ldots, 3\lambda_n\}$ then by Thm. 1

Lemma 3. *For any n there exists a choice of Λ such that $|\widetilde{\Lambda}| = n^{1+o(1)}$.*

We are now ready to prove the security of the new permutation argument. The culpable soundness of this argument is based on exactly the same ideas as that of the Hadamard product argument. Similarly to the Hadamard product argument, we cannot prove the computational soundness of this argument. Instead, we prove $\mathcal{R}^{perm}_{\mathsf{guilt}}$-soundness, where

$$\mathcal{R}^{perm}_{\mathsf{guilt}} := \left\{ \begin{array}{l} (\sigma; (A, \tilde{A}, B, \tilde{B}, \varrho), (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, (f'_{(\varrho,i)}, f'_{(\varrho^{-1},i)})_{i \in \{0\} \cup \widetilde{\Lambda}}), \pi^{perm} = (\pi^\varrho, \widetilde{\pi}^\varrho, \pi^{\varrho^{-1}}, \widetilde{\pi}^{\varrho^{-1}})) : \\ (A, \tilde{A}) = \mathcal{C}\text{om}^1(\mathsf{ck}_1; \boldsymbol{a}; r_a) \wedge (B, \tilde{B}) = \mathcal{C}\text{om}^1(\mathsf{ck}_1; \boldsymbol{b}; r_b) \wedge \\ (\pi^\varrho, \widetilde{\pi}^\varrho) = (g_2^{\sum_{i \in \{0\} \cup \widetilde{\Lambda}} f'_{(\varrho,i)} x^i}, \tilde{g}_2^{\sum_{i \in \{0\} \cup \widetilde{\Lambda}} f'_{(\varrho,i)} x^i}) \wedge \\ (\pi^{\varrho^{-1}}, \widetilde{\pi}^{\varrho^{-1}}) = (g_2^{\sum_{i \in \{0\} \cup \widetilde{\Lambda}} f'_{(\varrho^{-1},i)} x^i}, \tilde{g}_2^{\sum_{i \in \{0\} \cup \widetilde{\Lambda}} f'_{(\varrho^{-1},i)} x^i}) \wedge \varrho \in S_n \wedge (\exists i \in [n]) a_{\varrho(i)} \neq b_i \wedge \\ \mathcal{V}_{perm}(\sigma_v; (A, \tilde{A}, B, \tilde{B}), \pi^{perm}, \varrho) = 1 \end{array} \right\} .$$

Theorem 6. *Prot. 2 is perfectly complete and perfectly witness-indistinguishable. Assume that the verifier additionally checks the correctness of commitments A and B by verifying that $e(A, \tilde{g}_2) = e(\tilde{A}, g_2)$, and $e(B, \tilde{g}_2) = e(\tilde{B}, g_2)$. If the bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is $\widetilde{\Lambda}$-PSDL secure, then Prot. 2 is adaptively $\mathcal{R}^{perm}_{\mathsf{guilt}}$-sound.*

The proof is given in App. G.

Theorem 7. *Let Λ be as described in Thm. 1. The CRS consists of $n^{1+o(1)}$ group elements. The communication (the argument size) of the argument in Prot. 2 is 4 elements from $\mathbb{G}_2$. The prover's computational complexity is dominated by $\Theta(n^2)$ additions in $\mathbb{Z}_p$ and by $n^{1+o(1)}$ exponentiations in $\mathbb{G}_2$. The verifier's computational complexity is dominated by 10 bilinear pairings and $2n$ multiplications in bilinear groups.*

The proof is given in App. H.

7 Comparison: Basic Arguments

Efficiency. In [Gro10], Groth used basically the same commitment scheme but with $\Lambda = \{i : i \in [n]\}$. In fact, he committed to $\boldsymbol{b}$ by using a different set $\Lambda' = \{i(n+1) : i \in [n]\}$. According to our terminology, in both of Groth's basic arguments $|\bar{\Lambda}| = \Theta(n^2)$ (this is since $|\{g^{i+j(n+1)} : i \in [n] \wedge j \in [n]\}| = \Theta(n^2)$), and thus the CRS length is quadratic (since it contains $\{g^{x^i} : i \in \bar{\Lambda}\}$) but also in addition, it takes $\Theta(n^2)$ bilinear-group exponentiations to compute π. We emphasize that the difference between $\Theta(n^2)$ additions (as in the new arguments) and $\Theta(n^2)$ bilinear-group exponentiations (as in Groth's argument) is very important in practice; and in both cases, Θ hides approximately the same constant. Moreover, the choice of Λ and Λ' in [Gro10] is somewhat ad hoc. One of the contributions of the current paper is the intuitive reasoning for the choice of Λ, and a near-optimal (up to a factor of 2!) construction of Λ. (We emphasize that [Gro10] did not consider any other possible sets Λ at all.)

The fact that we use the same n generators to commit to all A, B, and C means that we can more readily reuse these commitments in different arguments, even if B is a commitment in a different group. For example, to prove that $A \in \mathbb{G}_1$ and $B \in \mathbb{G}_2$ commit to the same vector (by using the same randomizer), Groth [Gro10] had to use a separate product argument. In our case, it is just sufficient to verify that $e(A, g_2) = e(g_1, B)$. Finally, in the case of the Hadamard product argument, we made the verifier's computational complexity constant by adding 1 group element to the CRS. (In the permutation argument, one can also make the verifier's computational complexity constant by adding a few elements to the CRS. However, in the case of the permutation argument, this optimization is valid only when the permutation is previously known like the permutation ϱ in the case of the SAT argument in Sect. 8.) Due to this optimization, the verifier of the Hadamard product argument only needs access to a small part σ_v of the (potentially long) CRS; σ_v consists of the description of the bilinear group plus 5 group elements.

Asymmetric Pairings. Differently from [Gro10], we use asymmetric pairings. While at least currently, asymmetric pairings are much more efficient than symmetric pairings, it also means that we have to take additional care by modifying several aspects of the security assumptions, of the arguments and of the proofs.

Security Assumptions/Proof. By giving a somewhat clearer proof, we were able to avoid Groth's CPDH assumption, and rely on the (potentially weaker) PSDL assumption instead. Moreover, since the adversary in both security proofs has to factorize a degree λ_n polynomial, our security proofs have a tighter reduction than those in [Gro10].

8 New NIZK Argument for Circuit Satisfiability

In a *NIZK argument for circuit satisfiability* (SAT, well-known to be a **NP**-complete language), the prover and the verifier share a circuit C. The prover aims to prove in non-interactive zero-knowledge that she knows an assignment of input values that makes the circuit output 1. As in [Gro10], the SAT argument will use the Hadamard product argument, the permutation argument and a trivial argument for element-wise sum of two vectors — in our case, all operating in parallel on $2|C| + 1$ dimensional vectors, where $|C|$ is the circuit size. Those three arguments can be seen as basic operations in an NIZK "programming language". We show that a small constant number of such basic operations is sufficient for SAT. The full argument then contains additional cryptographic sugar: precise definition of the used CRS, computational/communication optimizations, etc.

To make it easier to read the following (and to construct future protocols based on this "programming language"), we give first a high-level description of the resulting argument. The first task is to express the underlying argument as a parallel composition of some addition, permutation and product arguments. These arguments may include intermediate variables (that will be committed to by the prover) and constants (that can be online committed to by both of the parties separately). When choosing the arguments, one has to keep in mind that we work in an asymmetric setting. This may mean that for some of the inputs to the circuit satisfiability argument, one has to commit to them both in $\mathbb{G}_1$ and $\mathbb{G}_2$ (and the verifier has to check that this is done correctly).

System parameters: Define Λ and $\bar{\Lambda}$ as in Prot. 1 and $\widetilde{\Lambda}$ as in Prot. 2, but in all cases with n replaced by $2|C|+1$. Permutation ϱ.

Common reference string generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Set $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Generate random $\alpha, \widetilde{\alpha}, x \leftarrow \mathbb{Z}_p$, $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$, $g_2 \leftarrow \mathbb{G}_2 \setminus \{2\}$. Denote $g_{ti} \leftarrow g_t^{x^i}$, $\bar{g}_{ti} \leftarrow g_t^{\alpha x^i}$ and $\tilde{g}_{ti} \leftarrow g_t^{\widetilde{\alpha} x^i}$ for $t \in \{1,2\}$ and $i \in [2|C|+1]$. Let $\bar{g}_t \leftarrow \bar{g}_{t0}$ and $\tilde{g}_t \leftarrow \tilde{g}_{t0}$ for $t \in \{1,2\}$. Set $(D, \bar{D}, D_2) \leftarrow (\prod_{i=1}^n g_{1,\lambda_i}, \prod_{i=1}^n \bar{g}_{1,\lambda_i}, \prod_{i=1}^n g_{2,\lambda_i})$, and $(F, \bar{F}) \leftarrow (\prod_{i=1}^{n-1} g_{1,\lambda_i} \cdot g_{1,\lambda_n}^2, \prod_{i=1}^{n-1} \bar{g}_{1,\lambda_i} \cdot \bar{g}_{1,\lambda_n}^2)$. The CRS is $\sigma \leftarrow (\mathsf{gk}; (g_{1i}, \bar{g}_{1i}, \tilde{g}_{1i})_{i\in\{0\}\cup\Lambda}, (\bar{g}_{2i})_{i\in\bar{\Lambda}}, (g_{2i}, \tilde{g}_{2i})_{i\in\{0\}\cup\widetilde{\Lambda}}, D, \bar{D}, D_2, F, \bar{F})$. Let $\mathsf{ck}_t \leftarrow (\mathsf{gk}; (g_{ti}, \bar{g}_{ti}, \tilde{g}_{ti})_{i\in\{0\}\cup\Lambda})$, $\overline{\mathsf{ck}}_t \leftarrow (\mathsf{gk}; (g_{ti}, \bar{g}_{ti})_{i\in\{0\}\cup\Lambda})$, $\widetilde{\mathsf{ck}}_t \leftarrow (\mathsf{gk}; (g_{ti}, \tilde{g}_{ti})_{i\in\{0\}\cup\Lambda})$, for $t \in \{1,2\}$, and let $\sigma_v \leftarrow (\mathsf{gk}; g_1, \bar{g}_1, \tilde{g}_1, g_2, \bar{g}_2, \widetilde{g_2}, D, (g_{2,2\lambda_i-\lambda_j})_{i,j\in[n]\wedge i\neq j})$ be the verifier's part of σ.

Common inputs: A satisfiable circuit C, and permutations τ and ζ generated based on C, such that $(\boldsymbol{L}, \boldsymbol{R}, R_{n+1}, \boldsymbol{U}, \boldsymbol{X}, X_{n+1})$ is a "satisfying assignment".

Argument generation $\mathcal{P}(\sigma; C, (\boldsymbol{L}, \boldsymbol{R}, R_{n+1}, \boldsymbol{U}, \boldsymbol{X}))$**:** Denote $\boldsymbol{Y} := (Y_1, \ldots, Y_n)$ for $Y \in \{L, R, U, X\}$. The prover does the following.

1. Set $r_1, \ldots, r_4 \leftarrow \mathbb{Z}_p$, and then compute $(\mathsf{lr}, \overline{\mathsf{lr}}, \widetilde{\mathsf{lr}}) \leftarrow \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{L}, \boldsymbol{R}, R_{n+1}; r_1)$, $\mathsf{lr}_2 \leftarrow g_2^{r_1} \cdot \prod_{i=1}^n g_{2i}^{L_i} \cdot \prod_{i=1}^{n+1} g_{2,i+n}^{R_i}$, $(\mathsf{rl}, \widetilde{\mathsf{rl}}) \leftarrow \mathcal{C}\mathrm{om}^1(\widetilde{\mathsf{ck}}_1; \boldsymbol{R}, \boldsymbol{L}, R_{n+1}; r_1)$, $(\mathsf{rz}, \overline{\mathsf{rz}}) \leftarrow \mathcal{C}\mathrm{om}^1(\overline{\mathsf{ck}}_1; \boldsymbol{R}, 0, \ldots, 0, 0; r_2)$, $(\mathsf{uz}, \overline{\mathsf{uz}}) \leftarrow \mathcal{C}\mathrm{om}^1(\overline{\mathsf{ck}}_1; \boldsymbol{U}, 0, \ldots, 0, 0; r_3)$, $(\mathsf{ux}, \overline{\mathsf{ux}}, \widetilde{\mathsf{ux}}) \leftarrow \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{U}, \boldsymbol{X}, X_{n+1}; r_4)$.
2. Create an argument π_1 for $[\![(\mathsf{lr}, \overline{\mathsf{lr}})]\!] \circ [\![(\mathsf{lr}, \overline{\mathsf{lr}}, \mathsf{lr}_2)]\!] = [\![(\mathsf{lr}, \overline{\mathsf{lr}})]\!]$. Create an argument π_2 for $\varrho([\![(\mathsf{lr}, \widetilde{\mathsf{lr}})]\!]) = [\![(\mathsf{rl}, \widetilde{\mathsf{rl}})]\!]$. Create an argument π_3 for $[\![(\mathsf{rl}, \widetilde{\mathsf{rl}})]\!] \circ [\![(D, \bar{D}, D_2)]\!] = [\![(\mathsf{rz}, \overline{\mathsf{rz}})]\!]$. Create an argument π_4 for $[\![(\mathsf{ux}, \overline{\mathsf{ux}})]\!] \circ [\![(D, \bar{D}, D_2)]\!] = [\![(\mathsf{uz}, \overline{\mathsf{uz}})]\!]$. Create an argument π_5 for $[\![(\mathsf{rz}, \overline{\mathsf{rz}})]\!] \circ [\![(\mathsf{lr}, \overline{\mathsf{lr}}, \mathsf{lr}_2)]\!] = [\![(F, \bar{F}) \cdot (\mathsf{uz}^{-1}, \overline{\mathsf{uz}}^{-1})]\!]$. Create an argument π_6 for $\tau([\![(\mathsf{lr}, \widetilde{\mathsf{lr}})]\!]) = [\![(\mathsf{lr}, \widetilde{\mathsf{lr}})]\!]$. Create an argument π_7 for $\zeta([\![(\mathsf{lr}, \widetilde{\mathsf{lr}})]\!]) = [\![(\mathsf{ux}, \widetilde{\mathsf{ux}})]\!]$.
3. Send the argument $\pi \leftarrow (\mathsf{lr}, \overline{\mathsf{lr}}, \widetilde{\mathsf{lr}}, \mathsf{lr}_2, \mathsf{rl}, \widetilde{\mathsf{rl}}, \mathsf{rz}, \overline{\mathsf{rz}}, \mathsf{uz}, \overline{\mathsf{uz}}, \mathsf{ux}, \overline{\mathsf{ux}}, \widetilde{\mathsf{ux}}, \pi_1, \ldots, \pi_7)$ to the verifier.

Verification $\mathcal{V}(\sigma_v; C, \pi)$**:** The verifier does the following:

- For $A \in \{\mathsf{lr}, \mathsf{rz}, \mathsf{uz}, \mathsf{ux}\}$ check that $e(\bar{A}, g_2) = e(A, \bar{g}_2)$.
- Check that $e(g_1, \mathsf{lr}_2) = e(\mathsf{lr}, g_2)$.
- For $A \in \{\mathsf{lr}, \mathsf{rl}, \mathsf{ux}\}$ check that $e(\tilde{A}, g_2) = e(A, \tilde{g}_2)$.
- She verifies all 7 arguments $\pi_1, \ldots, \pi_7$.

Protocol 3: New NIZK argument for SAT

The CRS is basically the union of CRS-s for the product and permutation arguments. The total argument consists of commitments to intermediate variables and of all arguments in the program of this "programming language". Finally, the verifier has to check that all commitments are internally consistent, and then verify all used arguments.

Let us now turn to the concrete case of circuit satisfiability. For the sake of simplicity, assume that the circuit C is only composed of the NAND gates. Let C have n gates. Assume that the output gate of the circuit is n, and U_n is the output of the circuit. For every gate $j \in [n]$ of C, let the input wires of its jth gate be L_j and R_j, and let U_j be one of its output wires. We also define an extra value $R_{n+1} = 1$. We let X_j be other "output" wires that correspond to some L_k or R_k that were not already covered by U_k (that is, inputs to the circuit, or duplicates of output wires). That is, $(U_1, \ldots, U_n, X_1, \ldots, X_{n+1})$ is chosen so that for some permutation ζ, $(\boldsymbol{U}, \boldsymbol{X}, X_{n+1})$ is a ζ-permutation of $(\boldsymbol{L}, \boldsymbol{R}, R_{n+1})$, where $\boldsymbol{Y} = (Y_1, \ldots, Y_n)$ for $Y \in \{L, R, U, X\}$. (See Fig. 1 in appendix for an example circuit with values R_i, L_i, U_i and X_i.)

More precisely, the prover and the verifier share the following three permutations, the first two of which completely describe the circuit C. First, $\tau \in S_{2n+1}$ is a permutation, such that for any values $L_{i_1}, \ldots, L_{i_s}, R_{j_1}, \ldots, R_{j_t}$ that correspond to the same wire, τ contains a cycle $i_1 \to i_2 \to \cdots \to i_s \to j_1 + n \to \cdots \to j_t + n \to i_1$. For unique wires i, $\tau(i) = i$. Second, $\zeta \in S_{2n+1}$ is a permutation that for every input wire (either L_i or R_{i-n}), outputs an index $j \leftarrow \zeta(i)$, such that the output wire U_j or X_{j-n} is equal to that input wire. Third, $\varrho \in S_{2n+1}$ is a permutation, with $\varrho(i) = i + n$ and $\varrho(i + n) = i$ for $i \in [n]$, and $\varrho(2n + 1) = 2n + 1$.

The argument is given by Prot. 3. We note that in every subargument used in Prot. 3, the prover and the verifier use a substring of σ as the CRS. The corresponding substrings are easy to compute, and thus in what follows, we do not mention this issue. In a few cases, instead of computing two different commitments $\mathcal{C}\mathrm{om}^t(\overline{\mathsf{ck}}_t; \boldsymbol{a}; r) = (g_t^r \cdot \prod g_{ti}^{a_i}, \bar{g}_t^r \cdot \prod \bar{g}_{ti}^{a_i})$ and $\mathcal{C}\mathrm{om}^t(\widetilde{\mathsf{ck}}_t; \boldsymbol{a}; r) = (g_t^r \cdot \prod g_{ti}^{a_i}, \tilde{g}_t^r \cdot \prod \tilde{g}_{ti}^{a_i})$, we compute a composed commitment $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r) = (g_t^r \cdot \prod g_{ti}^{a_i}, \bar{g}_t^r \prod \bar{g}_{ti}^{a_i}, \tilde{g}_t^r \cdot \prod \tilde{g}_{ti}^{a_i})$.

Theorem 8. *Let $\mathcal{G}_{\mathsf{bp}}$ be $\widetilde{\Lambda}$-PSDL secure, and Λ-PKE secure in both $\mathbb{G}_1$ and $\mathbb{G}_2$. Then Prot. 3 is a perfectly complete, computationally adaptively sound and perfectly zero-knowledge non-interactive SAT argument.*

The proof is given in App. I.

Theorem 9. *Let Λ be chosen as in Thm. 1. The CRS consists of $|C|^{1+o(1)}$ group elements. The communication (argument length) of the argument in Prot. 3 is 12 elements from $\mathbb{G}_1$ and 21 elements from $\mathbb{G}_2$. The prover's computational complexity is dominated by $\Theta(|C|^2)$ simple arithmetical operations in $\mathbb{Z}_p$ and $|C|^{1+o(1)}$ exponentiations in $\mathbb{G}$. The verifier's computational complexity is dominated by 66 bilinear pairings and $8|C|+4$ multiplications.*

Proof. This argument contains 6 commitments, 4 product arguments and 3 permutation arguments, all together $13 + 4 \cdot 2 + 3 \cdot 4 = 33$ group elements. From those group elements, 12 belong to $\mathbb{G}_1$ and $1+4\cdot 2+3\cdot 4 = 21$ belong to $\mathbb{G}_2$. The verifier has to perform $2\cdot(4+1+3) = 16$ bilinear pairings to check the correctness of all commitments, and $4 \cdot 5 + 3 \cdot 10 = 50$ bilinear pairings to check all 7 arguments. She also has to perform $2(|C|+1)$ multiplications to verify each of the two permutation arguments π_6 and π_7. (Since ϱ does not depend on the circuit, one can optimize the multiplications out in the case of π_2.) Statements about the prover's computational complexity and the length of the CRS follow directly from Thm. 5 and Thm. 7. For the prover's computational complexity note that all committed elements in product arguments are Boolean, and that we can assume that all three used permutations are simple to compute (e.g., ζ and τ can be computed by a table-lookup). □

Recalling that elements of $\mathbb{G}_1$ can be represented by 320 bits and elements of $\mathbb{G}_2$ can be represented by 160 bits, the communication (argument length) is $12 \cdot 320 + 21 \cdot 160 = 7200$ bits.

Comparison to Groth's Argument from [Gro10]. The new SAT argument inherits several properties of the new Hadamard product and permutation arguments. First, while Groth's SAT argument had CRS of length $\Theta(n^2)$, in the new argument it is $n^{1+o(1)}$. Second, we have reduced the prover's computation from $\Theta(n^2)$ exponentiations in $\mathbb{G}_t$ to $\Theta(n^2)$ (conditional) additions in $\mathbb{Z}_p$. Third, instead of Groth's CPDH assumption, the new argument relies on a potentially weaker PSDL assumption.

There are also several differences that are caused by how we compose the basic arguments. Compared to Groth's SAT argument from [Gro10], we have saved one commitment and one Hadamard product argument. Moreover, in Groth's argument, every commitment consisted of 3 group elements while every argument consisted of 2 group elements. In the new argument, most of the commitments consist of 2 group elements. Therefore, we saved 14 group elements, reducing the argument size from 42 to 33 group elements, even taking into account that the new permutation argument is more complex than that of [Gro10]. Since we use asymmetric pairings, we also have a more efficient representation of the group elements. There are a few other subtle differences with Groth's argument but they mostly affect readability.

References

[AF07] Masayuki Abe and Serge Fehr. Perfect NIZK with Adaptive Soundness. In Salil Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 118–136, Amsterdam, The Netherlands, February 21–24, 2007. Springer-Verlag.

[ANR95] Noga Alon, Melvyn B. Nathason, and Imre Ruzsa. Adding Distinct Congruence Classes Modulo a Prime. *American Mathematical Monthly*, 102(3):250–255, March 1995.

[Beh46] Felix A. Behrend. On the Sets of Integers Which Contain No Three in Arithmetic Progression. *Proceedings of the National Academy of Sciences*, 32(12):331–332, December 1946.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-Interactive Zero-Knowledge and Its Applications. In *STOC 1988*, pages 103–112, Chicago, Illinois, USA, May 2–4, 1988. ACM Press.

[BN05] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-Friendly Elliptic Curves of Prime Order. In Bart Preneel and Stafford E. Tavares, editors, *SAC 2005*, volume 3897 of *LNCS*, pages 319–331, Kingston, ON, Canada, August 11–12, 2005. Springer-Verlag.

[Bou98] Jean Bourgain. On Triples in Arithmetic Progression. *Geometric and Functional Analysis*, 9(5):968–984, 1998.

[BP04] Mihir Bellare and Adriana Palacio. The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols. In Matthew K. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 273–289, Santa Barbara, USA, August 15–19, 2004. Springer-Verlag.

[BR93] Mihir Bellare and Phillip Rogaway. Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols. In Victoria Ashby, editor, *ACM CCS 1993*, pages 62–73, Fairfax, Virginia, 3–5 November 1993. ACM Press.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The Random Oracle Methodology, Revisited. In *STOC 1998*, pages 209–218, New York, May 23–26, 1998.

[CLS10] Rafik Chaabouni, Helger Lipmaa, and Abhi Shelat. Additive Combinatorics and Discrete Logarithm Based Range Protocols. In Ron Steinfeld and Philip Hawkes, editors, *ACISP 2010*, volume 6168 of *LNCS*, pages 336–351, Sydney, Australia, July 5–7, 2010. Springer-Verlag.

[Dam91] Ivan Damgård. Towards Practical Public Key Systems Secure against Chosen Ciphertext Attacks. In Joan Feigenbaum, editor, *CRYPTO 1991*, volume 576 of *LNCS*, pages 445–456, Santa Barbara, California, USA, August 11–15, 1991. Springer-Verlag, 1992.

[dB64] Nicolaas Govert de Bruijn. Some Direct Decompositions of the Set of Integers. *Mathematics of Computation*, 18:537–546, 1964.

[DL08] Giovanni Di Crescenzo and Helger Lipmaa. Succinct NP Proofs from an Extractability Assumption. In Arnold Beckmann, Costas Dimitracopoulos, and Benedikt Löwe, editors, *Computability in Europe, CIE 2008*, volume 5028 of *LNCS*, pages 175–185, Athens, Greece, June 15–20, 2008. Springer-Verlag.

[DN00] Cynthia Dwork and Moni Naor. Zaps and Their Applications. In *FOCS 2000*, pages 283–293, Redondo Beach, California, USA, November 12–14, 2000. IEEE Computer Society Press.

[Elk08] Michael Elkin. An Improved Construction of Progression-Free Sets. Technical Report arXiv:0801.4310v1, arXiv.org, January 28, 2008. Available from http://arxiv.org/abs/0801.4310.

[Elk10] Michael Elkin. An Improved Construction of Progression-Free Sets. In Moses Charikar, editor, *SODA 2010*, pages 886–905, Austin, Texas, USA, January 17–19, 2010. SIAM.

[ET36] Paul Erdős and Paul Turán. On Some Sequences of Integers. *Journal of the London Mathematical Society*, 11(4):261–263, 1936.

[Gen09] Craig Gentry. Fully Homomorphic Encryption Using Ideal Lattices. In Michael Mitzenmacher, editor, *STOC 2009*, pages 169–178, Bethesda, Maryland, USA, May 31 — June 2, 2009. ACM Press.

[GJM02] Philippe Golle, Stanislaw Jarecki, and Ilya Mironov. Cryptographic Primitives Enforcing Communication and Storage Complexity. In Matt Blaze, editor, *FC 2002*, volume 2357 of *LNCS*, pages 120–135, Southhampton Beach, Bermuda, March 11–14, 2002. Springer-Verlag.

[GK03] Shafi Goldwasser and Yael Tauman Kalai. On the (In)security of the Fiat-Shamir Paradigm. In *FOCS 2003*, pages 102–113, Cambridge, MA, USA, October, 11–14 2003. IEEE, IEEE Computer Society Press.

[GL07] Jens Groth and Steve Lu. A Non-interactive Shuffle with Pairing Based Verifiability. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 51–67, Kuching, Malaysia, December 2–6, 2007. Springer-Verlag.

[GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The Knowledge Complexity of Interactive Proof-Systems. In *STOC 1985*, pages 291–304, Providence, Rhode Island, USA, May 6–8, 1985. ACM Press.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect Non-Interactive Zero-Knowledge for NP. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 338–359, St. Petersburg, Russia, May 28–June 1, 2006. Springer-Verlag.

[GOS11] Jens Groth, Rafail Ostrovsky, and Amit Sahai. New Techniques for Non-interactive Zero Knowledge. Full version of [GOS06]. Draft, available from the authors, March 7, 2011.

[Gro09] Jens Groth. Linear Algebra with Sub-linear Zero-Knowledge Arguments. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 192–208, Santa Barbara, California, USA, August 16–20, 2009. Springer-Verlag.

[Gro10] Jens Groth. Short Pairing-Based Non-interactive Zero-Knowledge Arguments. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 321–340, Singapore, December 5–9 2010. Springer-Verlag.

[GW10] Ben Green and Julia Wolf. A Note on Elkin's Improvement of Behrend's Construction. In David Chudnovsky and Gregory Chudnovsky, editors, *Additive Number Theory*, pages 141–144. Springer New York, 2010.

[GW11] Craig Gentry and Daniel Wichs. Separating Succinct Non-Interactive Arguments from All Falsifiable Assumptions. In Salil Vadhan, editor, *STOC 2011*, pages 99–108, San Jose, California, USA, June 6–8, 2011. ACM Press.

[HSV06] Florian Hess, Nigel P. Smart, and Frederik Vercauteren. The Eta Pairing Revisited. *IEEE Transactions on Information Theory*, 52(10):4595–4602, 2006.

[LZ11] Helger Lipmaa and Bingsheng Zhang. A More Efficient Computationally Sound Non-Interactive Zero-Knowledge Shuffle Argument. Technical Report 2011/394, International Association for Cryptologic Research, July 21, 2011. Available at http://eprint.iacr.org/2011/394.

[Mic94] Silvio Micali. CS Proofs. In Shafi Goldwasser, editor, *FOCS 1994*, pages 436–453, Los Alamitos, California, USA, November 1994. IEEE, IEEE Computer Society Press.

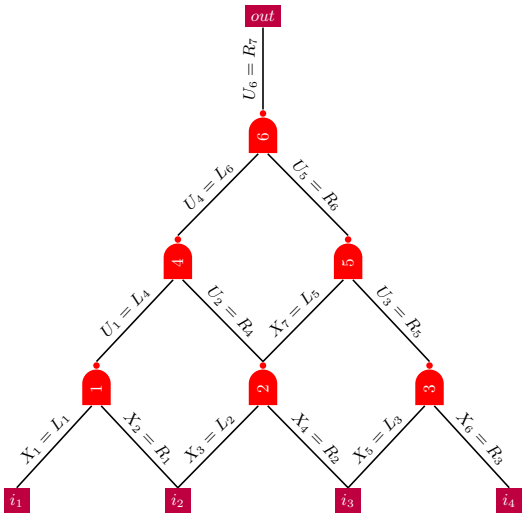

Fig. 1. Example circuit

[Mos62] Leo Moser. An Application of Generating Series. *Mathematics Magazine*, 35(1):37–38, January 1962.

[Nao03] Moni Naor. On Cryptographic Assumptions and Challenges. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 96–109, Santa Barbara, USA, August 17–21, 2003. Springer-Verlag.

[PSNB10] Geovandro C. C. F. Pereira, Marcos A. Simplício Jr, Michael Naehrig, and Paulo S. L. M. Barreto. A Family of Implementation-Friendly BN Elliptic Curves. Technical Report 2010/429, International Association for Cryptologic Research, August 3, 2010. Available at http://eprint.iacr.org/2010/429.

[San10] Tom Sanders. On Roth's Theorem on Progressions. Technical Report arXiv:1011.0104v1, arXiv.org, October 30, 2010. Available from http://arxiv.org/abs/1011.0104.

[Sch80] Jacob T. Schwartz. Fast Probabilistic Algorithms for Verification of Polynomial Identities. *Journal of the ACM*, 27(4):701–717, 1980.

[Seo11] Jae Hong Seo. Round-Efficient Sub-linear Zero-Knowledge Arguments for Linear Algebra. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 387–402, Taormina, Italy, March 6–9, 2011. Springer-Verlag.

[TV06] Terrence Tao and Van Vu. *Additive Combinatorics*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2006.

[vHN10] Mark van Hoeij and Andrew Novocin. Gradual Sub-lattice Reduction and a New Complexity for Factoring Polynomials. In Alejandro López-Ortiz, editor, *LATIN 2010*, volume 6034 of *LNCS*, pages 539–553, Oaxaca, Mexico, April 19–23, 2010. Springer-Verlag.

A CPDH Assumption

Computational Power Diffie-Hellman Assumption. Following [Gro10], we say that a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is $q(\kappa)$-*CPDH secure in group* $\mathbb{G}_t$ for $t \in \{1,2\}$, if for any non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ and for any $j \in [q]$,

$$\Pr\left[\begin{array}{l}\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), g_t \leftarrow \mathbb{G}_t \setminus \{1\}, (x, \alpha) \leftarrow \mathbb{Z}_p^2 : \\ \mathcal{A}(\mathsf{gk}; (g_t^{x^i})_{i\in\{0,\ldots,q\}}, (g_t^{\alpha x^i})_{i\in\{0,\ldots,q\}\setminus\{j\}}) = g_t^{\alpha x^j}\end{array}\right]$$

is negligible in κ. As shown in [Gro10], q-CPDH holds in generic group model for any polynomial q. Note that in [Gro10], $\mathbb{G}_1 = \mathbb{G}_2$, while here, the two groups are different, and the q-CPDH assumption is only required to be true in one of them. This does not change the security proof in the generic group model. A related assumption in a non pairing-based group was defined in [GJM02].

B Proof of Elkin's Result

Theorem 10. *For any fixed N, there exists $y = N^{1+o(1)}$, such that $r_3^{odd}(y) = N$.*

Proof. For any $S \subset \mathbb{R}^n$, let $\mathrm{vol}(S)$ be the volume of S. Let $\mathbb{T}^n = \mathbb{R}^n/\mathbb{Z}^n$ be the n-dimensional torus, we identify $\mathbb{T}^n$ with $[0,1]^n$. For each $r \leq \frac{1}{2} \cdot \sqrt{n}$, let

$$S_n(r,\delta) := \{X \in [0,1/2]^n : r - \delta \leq ||X||_2 \leq r\} \ .$$

We need the next technical lemma, proof of which is given in App. C.

Lemma 4. *For some choice of $r \sim \sqrt{n/12}$, $\mathrm{vol}(S_n(r,\delta)) \geq 4\sqrt{5}\delta/3 \cdot 2^{-n} \cdot (1 - O(n^{-1/2}))$.*

Denote $S := S_n(r,\delta)$ for this choice of r and δ. Since there is no overlap, we identify $[0,1]^n$ with $\mathbb{T}^n$, and S with the corresponding subset of $\mathbb{R}^n$.

Now, let x and y be such that $x-y, x, x+y$ lie in S. But then $2\,||x||_2^2 + 2\,||y||_2^2 = ||x+y||_2^2 + ||x-y||_2^2$, and thus

$$\begin{aligned} ||y||_2 =& \sqrt{(||x+y||_2^2 + ||x-y||_2^2 - 2\,||x||_2^2)/2} \leq \sqrt{r^2 - (r-\delta)^2} = \sqrt{2\delta r - \delta^2} \\ \leq& \sqrt{2\delta r} \ . \end{aligned}$$

The volume of n-dimensional ball $B_n(R) := \{x \in \mathbb{R}^n : ||x||_2 \leq R\}$ is

$$\mathrm{vol}(B_n(R)) = \pi^{n/2}/\Gamma(n/2+1) \cdot R^n \approx \frac{(2e\pi)^{n/2}}{\sqrt{\pi n} \cdot n^{n/2}} \cdot R^n \ ,$$

where $\Gamma(x)$ is the standard Γ function, and we have used Stirling's approximation. Thus,

$$\mathrm{vol}(B_n(\sqrt{2\delta r})) \approx \frac{(2e\pi)^{n/2}}{\sqrt{\pi n} \cdot n^{n/2}} \cdot (2\delta r)^{n/2} = \frac{1}{\sqrt{\pi n}} \cdot (4e\pi\delta r/n)^{n/2} \ .$$

The volume of the region $B \subset \mathbb{T}^n \times \mathbb{T}^n$ where all such values (x,y) belong cannot be larger than $\mathrm{vol}(S) \cdot \mathrm{vol}(B_n(\sqrt{2\delta r}))$. Since $r \leq \sqrt{n/12} + 1/(2\sqrt{5}) + O(n^{-1/2}) = \sqrt{n/12} + O(1)$, then $\mathrm{vol}(B_n(\sqrt{2\delta r})) \approx 1/\sqrt{\pi n} \cdot (4e\pi\delta r/n)^{n/2} \leq 1/\sqrt{\pi n} \cdot (4e\pi\delta(\sqrt{n/12} + O(1))/n)^{n/2} = 1/\sqrt{\pi n} \cdot (2e\pi\delta(1 + O(n^{-1/2}))/\sqrt{3n})^{n/2} = 1/\sqrt{\pi n} \cdot (2e\pi\delta/\sqrt{3n})^{n/2}(1 + O(n^{-1/2}))$.

We have now established that $\mathbb{T}^n$ has a relatively large progression-free area. We now show how to use this result to establish a similar result in $[N]$. Let $\Psi_{\theta,\alpha} : [N] \to \mathbb{T}^n$ be defined as $\Psi_{\theta,\alpha}(d) = \theta d + \alpha \mod 1$. If m is an integer, then $\Psi_{\theta,\alpha}(m)$ clearly is uniformly distributed on $\mathbb{T}^n$ as θ, α vary uniformly and independently on $\mathbb{T}^n$. Moreover, one can show [GW10] that when n and n' are distinct positive integers, then $(\Psi_{\theta,\alpha}(n), \Psi_{\theta,\alpha}(n'))$ is uniformly distributed on $\mathbb{T}^n \times \mathbb{T}^n$ as θ and α vary uniformly and independently over $\mathbb{T}^n$. Define $A_{\theta,\alpha} := \{n \in [N] : \Psi_{\theta,\alpha}(n) \in S\}$. Then due to the uniform distribution of $\Psi_{\theta,\alpha}(n)$, $\mathbb{E}_{\theta,\alpha}|A_{\theta,\alpha}| = N \cdot \mathrm{vol}(S)$.

Next, let $T(A_{\theta,\alpha})$ be the number of nontrivial 3-term arithmetic progressions in $A_{\theta,\alpha}$ and assume that N is even. Each nontrivial 3-term progression is of the form $(x-y, x, x+y)$ with $y \neq 0$. Since N is even, the possible values of y range from 1 to $N/2 - 1$. For every y, the possible values of x range from $y+1$ to $N-y$. Thus, there are $\sum_{y=1}^{N/2-1}(N-2y) = N(N-2)/4$ possible such 3-term progressions. The probability that an arbitrary such progression (x,y) belongs to $A_{\theta,\alpha}$ is $\mathrm{vol}(B)$, and thus $\mathbb{E}_{\theta,\alpha}[|T(A_{\theta,\alpha})|] = \frac{1}{4} \cdot N(N-2)\,\mathrm{vol}(B)$.

Now, for a moment, assume that

$$\frac{1}{4}(N-2)\,\mathrm{vol}(B) \leq \frac{1}{3}\,\mathrm{vol}(S) \ . \tag{6}$$

Then $\mathbb{E}[\frac{2}{3}|A_{\theta,\alpha}| - T(A_{\theta,\alpha})] = \mathbb{E}[\frac{2}{3}|A_{\theta,\alpha}|] - \mathbb{E}[T(A_{\theta,\alpha})] = \frac{2}{3} \cdot N\,\mathrm{vol}(S) - \frac{1}{4} \cdot N(N-2)\,\mathrm{vol}(B) \geq \frac{1}{3} \cdot N\,\mathrm{vol}(S)$. Clearly, thus there is a choice $A = A_{\theta,\alpha}$ for which both $T(A) \leq 2|A|/3$ and $|A| \geq \frac{1}{2} \cdot N\,\mathrm{vol}(S)$. Deleting up to 2/3 of the elements of A, we get a set of size at least $\frac{1}{6} \cdot N\,\mathrm{vol}(S)$ that is progression-free.

For Eq. (6) to hold, we must have

$$\frac{1}{3} \geq \frac{1}{4}(N-2)\,\mathrm{vol}(B_n(\sqrt{2\delta r})) \ ,$$

or

$$\frac{1}{\sqrt{\pi n}} \cdot \left(\frac{2e\pi\delta}{\sqrt{3n}}\right)^{n/2} (1 + O(n^{-1/2})) \leq \frac{4}{3(N-2)} = \frac{4}{3N} \cdot (1 - O(N^{-1})) \ ,$$

or

$$\frac{1}{\sqrt{\pi n}} \cdot \left(\frac{2e\pi\delta}{\sqrt{3n}}\right)^{n/2} \leq \frac{4}{3N} \cdot (1 - O(N^{-1}))(1 - O(n^{-1/2})) = \frac{4}{3N} \cdot (1 - O(n^{-1/2})) \ .$$

(Here, we assumed $n \ll \sqrt{N}$.) The latter is true whenever

$$\delta \leq \frac{\sqrt{3n}}{2e\pi} \cdot \left(\frac{16\pi n}{9(1 + O(n^{-1/2}))^2 N^2}\right)^{1/n} = \frac{\sqrt{3n}}{2e\pi} \cdot \left(\frac{16\pi n}{9N^2}\right)^{1/n} \cdot (1 - O(n^{-1/2})) \ ,$$

and we can take δ to be equal to the right-hand side of this inequality. But then

$$\begin{aligned}
\frac{1}{6} N \operatorname{vol}(S) =& \frac{N}{6} \cdot \frac{4\sqrt{5}\delta}{3} (1 - O(n^{-1/2})) \cdot \frac{1}{2^n} \\
=& \frac{N}{6} \cdot \frac{4\sqrt{5}(1 - O(n^{-1/2}))}{3 \cdot 2^n} \cdot \frac{\sqrt{3n}(1 - O(n^{-1/2}))}{2e\pi} \cdot \left(\frac{16\pi n}{9N^2}\right)^{1/n} \\
=& \frac{\sqrt{5}}{3\sqrt{3} \cdot e\pi} \cdot \frac{N^{1-2/n}\sqrt{n}}{2^n} \cdot \left(\frac{16\pi n}{9}\right)^{1/n} \cdot (1 - O(n^{-1/2})) \\
\geq& \frac{\sqrt{5}}{3\sqrt{3} \cdot e\pi} \cdot \frac{N^{1-2/n}\sqrt{n}}{2^n} \cdot (1 - O(n^{-1/2})) \ .
\end{aligned}$$

This is approximately maximized if $n = \sqrt{2 \log_2 N}$, then

$$\frac{1}{6} N \operatorname{vol}(S) \geq \frac{\sqrt[4]{2}\sqrt{5}}{3\sqrt{3}e\pi} \cdot \frac{N}{2^{2\sqrt{2}\sqrt{\log_2 N}}} \cdot (\log_2 N)^{1/4} \cdot (1 - O(\log^{-1/4} N)) \ .$$

Thus, $r_3(N) = \Theta(N \log^{1/4} N / 2^{2\sqrt{2 \log_2 N}})$, or $r_3(N) = N^{1-O(1)}$. This proves the theorem. □

C Proof of Lem. 4

Proof. For $i \in [n]$, let $X_i \in \mathbb{R}$ be a uniformly random variable in $[0, 1/2]$. Let $||X||_2 = \sqrt{\sum_{i=1}^n X_i}$ be the Euclidean norm of $X = (X_1, \ldots, X_n)$. Then $\mathbb{E}[X_i^2] = 2 \cdot \int_0^{1/2} x^2 dx = 1/12$, $\mathbb{E}[X_i^4] = 2 \cdot \int_0^{1/2} x^4 dx = 1/80$, $\operatorname{var}[X_i^2] = \mathbb{E}[X_i^4] - \mathbb{E}[X_i^2]^2 = 1/180$, $\mathbb{E}[||X||_2^2] = \sum_{i=1}^n \mathbb{E}[X_i^2] = n/12$, $\operatorname{var}[||X||_2^2] = \sum_{i=1}^n \operatorname{var}[X_i^2] = n/180$. By Chebyshev's inequality,

$$\Pr[|\, ||X||_2^2 - \mathbb{E}[||X||_2^2]| \geq t \cdot \sqrt{\operatorname{var}[||X||_2^2]}] \leq 1/t^2$$

for any $t > 0$. That is, $\Pr[|\, ||X||_2^2 - n/12| \geq t\sqrt{n/180}] \leq 1/t^2$. Thus, with probability at least $1 - 1/t^2$, $n/12 + t\sqrt{n/180} \geq ||X||_2^2 \geq n/12 - t\sqrt{n/180}$, and thus also

$$\sqrt{n/12 + t\sqrt{n/180}} \geq ||X||_2 \geq \sqrt{n/12 - t\sqrt{n/180}} \ .$$

If $n \to \infty$, then

$$\sqrt{n/12 \pm t\sqrt{n/180}} \to \sqrt{n/12} \pm t/(2\sqrt{15}) + O(n^{-1/2}) \ .$$

Thus, for large n, with probability $1 - 1/t^2$, $\sqrt{n/12} + t/(2\sqrt{15}) + O(n^{-1/2}) \geq ||X||_2 \geq \sqrt{n/12} - t/(2\sqrt{15}) + O(n^{-1/2})$, or equivalently, $|\, ||X||_2 - \sqrt{n/12}| \leq t/(2\sqrt{15}) + O(n^{-1/2})$.

Fix a small δ, $t/\sqrt{15} \geq \delta > 0$. Then, by the pigeonhole principle, there exists an r, such that with probability at least $\left(1 - 1/t^2\right) \cdot \delta / (t/(2\sqrt{15})(1 + O(n^{-1/2}))) = \left(1 - 1/t^2\right) \cdot 2\sqrt{15}\delta(1 - O(n^{-1/2}))/t$, $||X||_2 \in [r - \delta, r]$. This is maximized when $t = \sqrt{3}$, then the corresponding probability is $4\sqrt{5}\delta/3 \cdot (1 - O(n^{-1/2}))$. But then $\operatorname{vol}(S_n(r, \delta)) \geq 4\sqrt{5}\delta/3 \cdot 2^{-n} \cdot (1 - O(n^{-1/2}))$ as required. □

D Proof of Thm. 2

Proof. In the generic group model, an adversary only performs generic group operations (multiplications in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, bilinear pairings, and equality tests). A generic adversary produces an element of $\mathbb{Z}_p$, which depends only on gk and $(g_1^{x^i}, g_2^{x^i})_{i\in\{0\}\cup\Lambda}$. The only information that the adversary gets is when an equality (collision) between two previously computed elements of either $\mathbb{G}_1$, $\mathbb{G}_2$ or $\mathbb{G}_T$ occurs. We prove that finding even a single collision is difficult even if the adversary can compute an arbitrary group element in unit time.

Assume that the adversary can find a collision $y = y'$ in group $\mathbb{G}_1$. Then it must be the case that $y = \prod_{i\in\{0\}\cup\Lambda} g_1^{a_i x^i}$ and $y' = \prod_{i\in\{0\}\cup\Lambda} g_1^{a'_i x^i}$ for some known values of a_i and a'_i. But then also

$$\sum_{i\in\{0\}\cup\Lambda} (a_i - a'_i)x^i \equiv 0 \pmod{p} \ .$$

Since the adversary does not know the actual representations of the group elements, it will perform the same group operations independently of x. Thus a_i and a'_i are independent of x. By the Schwartz-Zippel lemma [Sch80] modulo p, the probability that $\sum_{i\in\{0\}\cup\Lambda}(a_i - a'_i)x^i \equiv 0 \pmod{p}$ is equal to λ_n/p for randomly chosen a_i and a'_i. If the adversary works in polynomial time $\tau = \mathrm{poly}(\kappa)$, it can generate at most τ such group elements. The total probability that there exists a collision between any two generated group elements is thus upper bounded by $\binom{\tau}{2} \cdot \lambda_n/p$, and thus a successful adversary requires time $\Omega(\sqrt{p/\lambda_n})$ to produce one collision.

A similar bound holds for collisions in $\mathbb{G}_2$. In the case of $\mathbb{G}_T$, the pairing enables the adversary to compute up to τ different values

$$y = e(g_1, g_2)^{\sum_{i\in\{0\}\cup\Lambda}\sum_{j\in\{0\}\cup\Lambda} a_{ij}x^{i+j}} \ ,$$

and thus we get an upper bound $\binom{\tau}{2} \cdot 2\lambda_n/p$, and thus a successful adversary requires time $\Omega(\sqrt{p/\lambda_n})$ to produce one collision. □

E Proof of Thm. 3

Proof. Perfect hiding: follows from the fact that the output of $\mathcal{C}\mathrm{om}^t$ is a random element of $\mathbb{G}_t$. Computational binding: Assume that $\mathcal{A}$ is an adversary that can break the binding property with some non-negligible probability. We construct the next adversary $\mathcal{A}'$ against the Λ-PDL assumption in group $\mathbb{G}_t$ that works with the same probability. Let $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $g_t \leftarrow \mathbb{G}_t \setminus \{1\}$, and $x \leftarrow \mathbb{Z}_p$. The adversary $\mathcal{A}'$ gets $(\mathsf{gk}; (g_t^{x^i})_{i\in\{0\}\cup\Lambda})$ as her input. She creates a random $\bar{\alpha}' \leftarrow \mathbb{Z}_p$, and sets $\mathsf{ck}_t \leftarrow (\mathsf{gk}; (g_t^{x^i}, g_t^{\bar{\alpha}' x^i})_{i\in\{0\}\cup\Lambda})$. She forwards ck_t to $\mathcal{A}$. By definition, $\mathcal{A}(\mathsf{ck}_t)$ produces a tuple $(\boldsymbol{a}, r_a, \boldsymbol{b}, r_b)$ with $(\boldsymbol{a}, r_a) \neq (\boldsymbol{b}, r_b)$, such that $g_t^{r_a} \cdot \prod_{i\in[n]} g_t^{a_i x^{\lambda_i}} = g_t^{r_b} \cdot \prod_{i\in[n]} g_t^{b_i x^{\lambda_i}}$. But then $g_t^{r_a - r_b + \sum_{i=1}^n (a_i - b_i)x^{\lambda_i}} = 1$, and thus $r_a - r_b + \sum_{i=1}^n (a_i - b_i)x^{\lambda_i} \equiv 0 \pmod{p}$. The adversary has generated a non-trivial polynomial $f(X)$ such that $f(x) = 0$. One can now use an efficient polynomial factorization algorithm in $\mathbb{Z}_p[X]$ to find all $\lambda_n + 1$ possible roots of f. For one of those roots, say y, it must be the case that $g_t^{x^{\lambda_1}} = g_t^{y^{\lambda_1}}$. Set $x \leftarrow y$. Thus, $\mathcal{A}'$ has broken the Λ-PDL assumption in group $\mathbb{G}_t$.

Extractability: By the Λ-PKE assumption in group $\mathbb{G}_t$, for every committer $\mathcal{A}$ there exists an extractor $X_{\mathcal{A}}$ that can open the commitment in group $\mathbb{G}_t$, given access to $\mathcal{A}$'s inputs and random tape. Since the commitment scheme is computationally binding, then the extracted opening has to be the same that $\mathcal{A}$ used. □

F Proof of Thm. 5

Proof. By Lem. 2, the size of the CRS is $\Theta(|\bar{\Lambda}|) = n^{1+o(1)}$. From the CRS, the verifier clearly only needs to access g_1, $\bar{g}_1$, g_2, $\bar{g}_2$ and D. Since $2\hat{}\Lambda \subseteq \bar{\Lambda}$, the statement about the prover's computational complexity follows. The prover's computational complexity comes from the fact that one can compute all values μ_k simultaneously by looping over n^2 possible values of i and j. The verifier's computational complexity follows from the description of the argument. □

G Proof of Thm. 6

Proof. Denote $\eta_\varrho := e(A,D)/e(B,E_\varrho)$ and $\eta_{\varrho^{-1}} := e(B,D)/e(A,E_{\varrho^{-1}})$. Let $h \leftarrow e(g_1,g_2)$. Witness-Indistinguishability: follows straightforwardly from the fact that there is exactly one possible argument π^{perm} that satisfies the verification equations.

Perfect completeness. The second and the fourth verifications are straightforward. For the first one, consider $\log_h \eta_\varrho$ in Eq. (4). Consider x as a formal variable, then the right-hand side (and thus also $\log_h \eta$) is a formal polynomial of x. More specifically, this formal polynomial is spanned by $\{x^i\}_{i\in 2\cdot\Lambda\cup\widetilde{\Lambda}}$. If the prover is honest, then $b_i = a_{\varrho(i)}$ for $i\in[n]$, and thus $\log_h \eta$ is a formal polynomial that is spanned by $\{x^i\}_{i\in\widetilde{\Lambda}}$. Defining

$$\pi^\varrho \leftarrow (\prod_{i=1}^{n} g_{2,\lambda_i})^{r_a} \cdot (\prod_{i=1}^{n} g_{2,2\lambda_{\varrho(i)}-\lambda_i})^{r_b} \cdot \prod_{i=1}^{n}\prod_{\substack{j=1\\ j\neq i}}^{n} (g_{2,\lambda_i+\lambda_j}^{a_i} \cdot g_{2,2\lambda_{\varrho(j)}+\lambda_i-\lambda_j}^{b_i}) = D^{r_a} \cdot E_\varrho^{r_b} \cdot \prod_{k\in\widetilde{\Lambda}'_\varrho} g_{2,k}^{\mu_{\varrho,k}} \ ,$$

we see that the first verification equation holds. The third verification can be checked dually.

Adaptive $\mathcal{R}_{\mathsf{guilt}}^{perm}$-soundness. Assume that $\mathcal{A}$ is an adversary that can break adaptive $\mathcal{R}_{\mathsf{guilt}}^{perm}$-soundness of the permutation argument. Next, we construct an adversary $\mathcal{A}'$ against the $\widetilde{\Lambda}$-PSDL assumption. Let $\mathsf{gk}\leftarrow\mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $x\leftarrow\mathbb{Z}_p$, $g_1\leftarrow\mathbb{G}_1\setminus\{1\}$, and $g_2\leftarrow\mathbb{G}_2\setminus\{1\}$. The adversary $\mathcal{A}'$ receives $\sigma\leftarrow(\mathsf{gk};(g_1^{x^i},g_2^{x^i})_{i\in\{0\}\cup\widetilde{\Lambda}})$ as her input, and her task is to output x. She sets $\widetilde{\alpha}\leftarrow\mathbb{Z}_p$, and

$$\sigma'\leftarrow(\mathsf{gk};(g_1^{x^i},g_1^{\widetilde{\alpha}x^i})_{i\in\{0\}\cup\Lambda},(g_2^{x^i},g_2^{\widetilde{\alpha}x^i})_{i\in\{0\}\cup\widetilde{\Lambda}},\prod_{i=1}^{n} g_2^{x^{\lambda_i}},\prod_{i=1}^{n}\tilde{g}_2^{x^{\lambda_i}}) \ ,$$

and forwards σ' to $\mathcal{A}$. Clearly, σ' follows the distribution imposed by $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$. Both parties also set

$$\mathsf{ck}_t\leftarrow(\mathsf{gk};(g_t^{x^i},g_t^{\widetilde{\alpha}x^i})_{i\in\{0\}\cup\Lambda})$$

for $t\in\{1,2\}$. Assume that $\mathcal{A}$ returns $((A,\tilde{A},B,\tilde{B},\varrho),w_{\mathsf{guilt}}^{perm},\pi^{perm})$ such that $\mathcal{V}(\sigma';(A,\tilde{A},B,\tilde{B},\varrho),\pi^{perm})$ accepts. Here, $w_{\mathsf{guilt}}^{perm}=(\boldsymbol{a},r_a,\boldsymbol{b},r_b,(f'_{(\varrho,i)},f'_{(\varrho^{-1},i)})_{i\in\{0\}\cup\widetilde{\Lambda}})$.

Since $\mathcal{A}$ is successful, $(A,\tilde{A})=\mathcal{C}\mathrm{om}^1(\mathsf{ck}_1;\boldsymbol{a};r_a)$, $(B,\tilde{B})=\mathcal{C}\mathrm{om}^2(\mathsf{ck}_2;\boldsymbol{b};r_b)$, and for some $i\in[n]$, $b_i\neq a_{\varrho(i)}$. W.l.o.g., assume that $a_{\varrho(i)}<b_i$. (The opposite case can be handled dually.) Since $2\cdot\Lambda\cap\widetilde{\Lambda}=\emptyset$, $\mathcal{A}'$ has thus expressed $\log_h\eta_\varrho$ as a polynomial $f(x)$ where at least for some $i\in 2\cdot\Lambda$, x^i has a non-zero coefficient.

On the other hand, $\mathcal{A}$ also outputs $(f'_{(\varrho,i)})_{i\in\{0\}\cup\widetilde{\Lambda}}$ with $\log_h\eta_\varrho=\log_{g_2}\pi=f'_\varrho(x)=\sum_{i\in\{0\}\cup\widetilde{\Lambda}}f'_{(\varrho,i)}x^i$, where all non-zero coefficients of $f'_\varrho(X)$ correspond to X^i for some $i\in\{0\}\cup\widetilde{\Lambda}$. Since Λ is a progression-free set of odd integers and all elements of $2\cdot\Lambda$ are distinct, then by the discussion in the beginning of Sect. 6, $i\notin 2\cdot\Lambda$. Therefore, all coefficients of $f'_\varrho(X)$ corresponding to any X^i, $i\in 2\cdot\Lambda$, are equal to 0. Thus

$$f(X)\cdot X^{\lambda_n}=\sum_{i\in\{0\}\cup\widetilde{\Lambda}\cup(2\cdot\Lambda)} f_i X^{i+\lambda_n}$$

and

$$f'_\varrho(X)=\sum_{i=\{0\}\cup\widetilde{\Lambda}} f'_{(\varrho,i)}X^{i+\lambda_n}$$

are different polynomials with $f(x)=f'_\varrho(x)=\log_h\eta_\varrho$. Therefore, $\mathcal{A}'$ has succeeded in creating a nonzero polynomial $d(X)=f(X)\cdot X^{\lambda_n}-f'_\varrho(X)$ such that $d(x)=\sum_{i\in\{0\}\cup\widetilde{\Lambda}}d_i x^i=0$.

Next, $\mathcal{A}'$ can use an efficient polynomial factorization algorithm [vHN10] in $\mathbb{Z}_p[X]$ to efficiently compute all $\leq 4\lambda_n+1$ roots of $d(x)$. For some root y, $g_1^{x^i}=g_1^{y^i}$. $\mathcal{A}'$ sets $x\leftarrow y$, thus violating the $\widetilde{\Lambda}$-PSDL assumption. □

H Proof of Thm. 7

Proof. It is clear that with this choice of Λ, by Lem. 3, the size of the CRS is $\Theta(|\widetilde{\Lambda}|) = n^{1+o(1)}$. Since $\widetilde{\Lambda}'_\varrho \subseteq \widetilde{\Lambda}$, the statement about the prover's computational complexity follows from the fact that one can compute all values $\mu_{\psi,k}$, where $\psi \in \{\varrho, \varrho^{-1}\}$, simultaneously by looping over n^2 possible values of i and j, as follows:

1. For $k \in 2\hat{}\Lambda$ do: $\mu_{\psi,k} \leftarrow 0$
2. For $i = 1$ to n do:
   - For $j = 1$ to n do: if $j \neq i$ then $\mu_{\psi,\lambda_i+\lambda_j} \leftarrow \mu_{\psi,\lambda_i+\lambda_j} + a_i$
3. For $j = 1$ to n do:
   - Set $t \leftarrow 2\lambda_{\varrho(j)} - \lambda_j$
   - For $i = 1$ to n do: if $j \neq i$ then $\mu_{\psi,t+\lambda_i} \leftarrow \mu_{\psi,t+\lambda_i} + b_i$

Here, one has to do $\Theta(n^2)$ additions and $\Theta(n)$ evaluations of ϱ or ϱ^{-1} (say, by using table-lookup). The verifier's computational complexity follows from the description of the argument. □

I Proof of Thm. 8

Proof. Perfect completeness: follows from that of the product and permutation arguments.

Adaptive computational soundness: Let $\mathcal{A}$ be a non-uniform probabilistic polynomial-time adversary that creates a circuit C and an accepting NIZK argument π. By the Λ-PKE assumption, there exists a non-uniform probabilistic polynomial-time extractor $X_{\mathcal{A}}$ that, running on the same input and seeing $\mathcal{A}$'s random tape, extracts all openings. From the culpable soundness of the product and permutation arguments and by the $\widetilde{\Lambda}$-PSDL assumption, it follows that the corresponding relations are satisfied between the opened values. Moreover, by the $\widetilde{\Lambda}$-PSDL assumption, the opened values belong to corresponding sets $\bar{\Lambda}$ and $\widetilde{\Lambda}$. Let $(\boldsymbol{L}, \boldsymbol{R}, R_{n+1})$ be the opening of $(\mathsf{lr}, \overline{\mathsf{lr}})$, where $\boldsymbol{L} = (L_1, \ldots, L_n)$ and $\boldsymbol{R} = (R_1, \ldots, R_n)$, and let $(U_1, \ldots, U_n, X_1, \ldots, X_n, X_{n+1})$ be the opening of $(\mathsf{ux}, \overline{\mathsf{ux}})$. We now analyze the effect of every subargument in Prot. 3. See Fig. 1 for a visual reference.

Successful verification of $e(g_1, \mathsf{lr}_2) = e(\mathsf{lr}, g_2)$ shows that lr_2 is correctly formed, and thus the first (product) argument shows that $L_i, R_i \in \{0, 1\}$. The second (permutation) argument shows that $(\mathsf{rl}, \widetilde{\mathsf{rl}})$ commits to $(\boldsymbol{R}, \boldsymbol{L}, R_{n+1})$. The third (product) argument shows that $(\mathsf{rz}, \overline{\mathsf{rz}})$ commits to $(R, 0, \ldots, 0, 0)$ and is thus consistent with the opening of $(\mathsf{lr}, \overline{\mathsf{lr}})$. The fourth argument shows that $(\mathsf{uz}, \overline{\mathsf{uz}})$ commits to $(U_1, \ldots, U_{n-1}, U_n, 0, \ldots, 0, 0)$ with $U_n = 0$, which is consistent with the opening of $(\mathsf{ux}, \overline{\mathsf{ux}})$.

The fifth argument shows that the NAND gates are followed. That is, $\neg(L_i \wedge R_i) = U_i$ for $i \in [n-1]$. It also shows that the circuit outputs 0. Really, since $(\mathsf{uz}, \overline{\mathsf{uz}})$ commits to $(U_1, \ldots, U_n, 0, \ldots, 0, 0)$, then $(E, \bar{E}) \cdot (\mathsf{uz}^{-1}, \overline{\mathsf{uz}}^{-1})$ commits to $(1 - U_1, \ldots, 1 - U_{n-1}, 2 - U_n, 0, \ldots, 0, 0)$. The product argument verifies exactly if $L_i \cdot R_i = 1 - U_i$ for $i \in [n-1]$ and $L_n \cdot R_n = 2 - U_n$. Since $L_n, R_n \in \{0, 1\}$, the latter can only be true if $U_n = 1$.

The sixth argument shows that if $i_1, \ldots, i_s, j_1 + n, \ldots, j_t + n$ correspond to the same wire, then $L_{i_1} = \cdots = L_{i_s} = R_{j_1} = \cdots = R_{j_t}$, i.e., the values are internally consistent with the wires. As an example, in Fig. 1, the permutation τ contains cycles $R_1 \to L_2 \to R_1$, $R_2 \to L_3 \to R_2$, and $R_4 \to L_5 \to R_4$. The seventh argument shows that the "input wires" and "output" wires are consistent. In Fig. 1, $\zeta(L_1) = X_1$, $\zeta(R_1) = X_2$, etc.

Perfect zero-knowledge: the simulator creates a correctly formed CRS together with a simulation trapdoor, so that the trapdoor commitments can be opened to any values. When simulating the argument, the simulator creates $(\mathsf{lr}, \overline{\mathsf{lr}}, \widetilde{\mathsf{lr}}, \mathsf{lr}_2)$, $(\mathsf{rl}, \widetilde{\mathsf{rl}})$, $(\mathsf{rz}, \overline{\mathsf{rz}})$, $(\mathsf{uz}, \overline{\mathsf{uz}})$ and $(\mathsf{ux}, \overline{\mathsf{ux}})$ as commitments to $(0, \ldots, 0)$. Due to the trapdoor, the simulator can simulate all product and permutation arguments. More precisely, he uses $L_i = R_i = U_i = 1$ to simulate all product and permutation arguments, except in the case of π_5 where he uses $U_i = 0$ instead. (Obviously, also $(\mathsf{rz}, \overline{\mathsf{rz}})$ and $(\mathsf{uz}, \overline{\mathsf{uz}})$ commit to $(1, \ldots, 1, 0, \ldots, 0)$.) To show that this argument π'' simulates the real argument π, note that π is perfectly indistinguishable from the simulated NIZK argument π' where one makes trapdoor commitments but opens them to *real* witnesses L_i, R_i when making product and permutation arguments. On the other hand, also π' and π'' are perfectly indistinguishable, and thus so are π and π''. □

J Perfect Zaps

The presented NIZK argument is in the CRS model, and thus requires some trusted third party to construct the CRS. If this is not possible, then one can instead construct a sublinear-size zap [DN00] (a 2-move publicly-verifiable witness-indistinguishable argument, where the verifier's first message can be reused polynomially many times), just by letting the verifier's first message to be equal to the CRS, and then letting the prover to verify the well-formedness of the CRS and then send his argument. It is straightforward to see that all CRSs constructed in the current paper are publicly verifiable. This zap will be perfectly witness-indistinguishable. Due to Groth's balancing technique [Gro10], the new zap for circuit satisfiability has the communication complexity of $|C|^{1/2+o(1)}$ group elements, while Groth's zap from [Gro10] has the communication complexity of $\Theta(|C|^{2/3})$ group elements. Note that Di Crescenzo and Lipmaa [DL08] constructed a 2-message perfectly complete and computationally sound polylogarithmic argument for **NP** under a knowledge assumption (and without random oracles), but their argument is not witness-indistinguishable.