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

Helger Lipmaa[1,2]

[1] Cybernetica AS, Estonia
[2] Tallinn University, Estonia

Abstract. In Asiacrypt 2010, Groth constructed the most efficient known (sublinear) non-interactive zero-knowledge (NIZK) argument for circuit satisfiability in the common reference string model. Let n be the circuit size. We propose a related NIZK argument that reduces the common reference string length from $\Theta(n^2)$ to $n^{1+o(1)}$, the prover's computation from $\Theta(n^2)$ bilinear-group exponentiations to $\Theta(n^2)$ additions and $n^{1+o(1)}$ bilinear-group exponentiations, the verifier's computation from $\Theta(n)$ pairings to 51 pairings, and the argument size from 42 group elements to 28 group elements. We also construct a perfect zap for circuit satisfiability with communication complexity $n^{1/2+o(1)}$. We achieve this by using a connection to the theory of progression-free sets, and this may have independent applications. Moreover, we use a slightly weaker security assumption than Groth.

Keywords. Bilinear pairings, circuit satisfiability, non-interactive zero-knowledge, progression-free sets.

1 Introduction

One of the central concepts in cryptography is zero-knowledge [GMR85]. By using the power of randomness and (usually) interaction, zero-knowledge proofs make it possible for the prover to convince the verifier in the truth of a statement without leaking any side information. Due to the wide applications of zero-knowledge in diverse areas of cryptography, 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 needs to prove that the tally is correct. It is infeasible to interact with the server every single time when somebody wants to verify 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, a NIZK proof/argument is generated once but can be verified many times), special effort has been put in minimizing these two parameters.

Previous Work. In some NIZK proofs for $\mathbf{NP}$-complete problems, see Tbl. 1, both the communication complexity and verifier's computational complexity are sublinear in a well-defined sense. Already in 1994, Micali [Mic94] proposed polylogarithmic NIZK *arguments* for all $\mathbf{NP}$-languages. However, his CS (computationally-sound) proofs are based on the PCP theorem (making them computationally unattractive) and on random oracles. Gentry [Gen09] noted that given a fully-homomorphic cryptosystem, one can construct very efficient NIZK *proofs* for all $\mathbf{NP}$-languages in the random oracle model. Another NIZK argument for a $\mathbf{NP}$-complete problem, circuit satisfiability, proposed by Groth in [Gro09], is also based on the random oracle model. (Without a random oracle, this argument takes $\log_2 |C| + 5$ rounds, where $|C|$ is the circuit size. In PKC 2011, Seo [Seo11] showed how to slightly reduce the number of rounds in Groth's argument.) While the random oracle model [BR93] makes it possible to design efficient protocols, it is also well-known, see e.g. [CGH98,BP04], that there exist functionalities that are secure in the random oracle model 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 random oracle model), that is secure in the random oracle model but

* Fifth eprint version, June 19, 2011. Includes many readability updates. The most noteworthy new result: the prover's computational complexity in the new SAT argument is $\Theta(n^2)$ additions (not $\Theta(n^2)$ multiplications, as claimed before).

insecure when instantiated with any "real" hash function. Thus, as a safeguard, it is important to design efficient NIZK proofs and arguments that do not rely on the random oracle model.

In Asiacrypt 2010, 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 (show that given three commitments commit to vectors where the third vector is a Hadamard product of the first two vectors) and permutation (the two commitments commit 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, related to the non pairing-based PCDH assumption from [GJM02] — see App. A for a brief recap) and a knowledge assumption (PKE). See Sect. 2. In particular, while knowledge assumptions are criticized for being unfalsifiable, the use of a non-standard assumption is necessary due to an impossibility result of Abe and Fehr [AF07]. In fact, the assumption that a NIZK argument is sound itself seems to be 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 full-blown random oracle model. We refer to [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 the input size. Of those possible generators, only $\leq 2n$ generators are really needed in the description of the arguments, while the rest of the generators are inherit 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 asymmetric pairings (not symmetric, as in [Gro10]) but for the sake of convenience we will use symmetric notation throughout the introduction.

New Basic Arguments. In the new basic arguments, we use a generalized version of Groth's knowledge commitment scheme with n generators $g^{x^{\lambda_i}}$, with some $0 = \lambda_0 < \lambda_1 < \lambda_2 < \cdots < \lambda_n = \mathrm{poly}(\kappa)$, to commit to a tuple $\boldsymbol{a} = (a_1, \ldots, a_n)$. More precisely, for a new randomness r_a, $\mathcal{C}\mathrm{om}(\sigma^{\mathsf{c}}; \boldsymbol{a}; r_a) = g^{r_a + \sum_{i=1}^n a_i x^{\lambda_i}} = g^{r_a} \cdot \prod_{i=1}^n g_i^{a_i}$. Here, σ^{c} is a commitment key, i.e., a CRS that in particular fixes the bilinear group, the generator g and group elements $g_i = g^{\lambda_i}$ for $i \in \{0, 1, \ldots, q\}$. See Sect. 3.

For the arguments to be perfectly complete and computationally sound, the set $\Lambda = \{\lambda_i\}$ has to satisfy certain properties. In the product argument, the prover has to construct an argument π such that $e(A, B) = e(C, D) \cdot e(g, \pi)$, where e is the bilinear pairing, $A = \mathcal{C}\mathrm{om}(\sigma^{\mathsf{c}}; \boldsymbol{a}; r_a)$, $B = \mathcal{C}\mathrm{om}(\sigma^{\mathsf{c}}; \boldsymbol{b}; r_b)$, $C = \mathcal{C}\mathrm{om}(\sigma^{\mathsf{c}}; \boldsymbol{c}; r_c)$, and $D = \mathcal{C}\mathrm{om}(\sigma^{\mathsf{c}}; \mathbf{1}; 0)$. As we show in Sect. 4, 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 written as a formal polynomial $\pi(x) = \sum_{i \in [n]} (a_i b_i - c_i) x^{2\lambda_i} + \sum_{i \in \Lambda_2} \pi_i x^i$ of x, with the powers corresponding to nonzero coefficients belonging either to the set $\Lambda_1 = \{2\lambda_i\}$ (the coefficients $a_i b_i - c_i$) or to the set $\Lambda_2 = \{0\} \cup \{\lambda_i\} \cup \{\lambda_i + \lambda_j : i \neq j\}$. If the prover is honest, then the coefficients $a_i b_i - c_i$ corresponding to the powers in the set Λ_1 are equal to 0.

To really be certain of this, we additionally need that $\Lambda_1 \cap \Lambda_2 = \emptyset$. As we show in Sect. 2.1, Λ has to be progression-free (that is, not contain any arithmetic progressions of length 3, see Sect. 2.1), 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. While in the case of the permutation argument, the set Λ_2 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 $\{g^{x^i} : i \in \Lambda_2\}$ has to belong to the CRS, and thus in particular we need Λ_2 to be small. This can be achieved by upper bounding λ_n, since $\Lambda_2 \subset \{0, \ldots, 2\lambda_n\}$. (In the case of the permutation argument, $\Lambda_2 \subset \{-\lambda_n, \ldots, 3\lambda_n\}$.) On the other hand, for soundness it is important that no element g^{x^i}, $i \in \Lambda_1$, belongs to the CRS. (The real arguments are somewhat more complicated, see Sect. 4 and Sect. 5.) For this we show in Sect. 2.1 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 $\lambda_n = n^{1+o(1)}$. For the sake of completeness, 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.

Thus, we obtain basic arguments where only $\Theta(\lambda_n)$ generators $\{g^{x^i} : i \in \Lambda_2\}$ are used in the argument generation. In the case of both basic arguments, the verifier only has to perform one check of type $e(A, B) = e(C, D) \cdot \pi$, the prover has to compute π (which takes $\Theta(n^2)$ additions in $\mathbb{Z}_p$ and $n^{1+o(1)}$ bilinear-group exponentiations). As

in [Gro10], the prover's computation can be optimized even further by using efficient multiexponentiation algorithms. For the arguments to be sound, following [Gro10] we make a knowledge assumption (PKE) that basically states that if for some random α, $(g^{x^i}, g^{\alpha x^i})$ is in the CRS exactly if $i \in \Lambda_2$ and the prover outputs c and c^α, then the prover must know how to represent c as $g^{\sum_{i\in\Lambda_2} a_i x^i}$ for some a_i.

Comparison: Basic Arguments. In the case of [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 $|\Lambda_2| = \Theta(n^2)$ (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 $\{g^{x^i} : i \in \Lambda_2\}$) 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; moreover, in both cases Θ hides 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.)

In addition, we prove the adaptive co-soundness of the basic arguments by using a (potentially) weaker security assumption PSDL (power symmetric discrete logarithm assumption, see Sect. 3), as opposed to Groth's CPDH (power computational Diffie-Hellman assumption). Moreover, as opposed to [Gro10] that only proved adaptive co-soundness [GOS06], we prove both the computational adaptive soundness and the adaptive co-soundness of the basic arguments. Almost the same proof suffices in both cases, but in the case of soundness one also has to make the PKE assumption. Finally, we note that verifier's computation can be made constant[1] by adding up to 2 group elements to the CRS. In particular, due to this optimization, the verifier 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 or 6 group elements. To emphasize this, we give an explicit definition of the verifier's part of CRS in Sect. 2.

Circuit Satisfiability Argument. We use the new basic arguments to construct the new NIZK argument for circuit satisfiability (SAT) by using a technique similar to that of Groth [Gro10]. Briefly, one can see the two basic arguments together with the trivial argument for element-wise sum (given commitments to $\boldsymbol{a}$ and $\boldsymbol{b}$, the prover and the verifier can both multiply the commitments to get a commitment to $\boldsymbol{a}+\boldsymbol{b}$) as basic building blocks for a NIZK "programming language". We will show that in this "language", one can build a NIZK for circuit satisfiability in a small constant number of steps. The new SAT argument is more efficient not only because the basic arguments are more efficient, but also since we use a slightly different "program". To optimize the resulting NIZK argument, we need to add some cryptographic sugar (e.g., define the required CRS).

Comparison: SAT Argument. As in the Groth's SAT argument of [Gro10], the communication and computational complexity of the new SAT argument directly follow from those of the basic arguments. In addition, we apply a few more optimizations that result in the SAT argument having shorter communication (28 group elements instead of 42), and the verifier having a constant computational complexity (instead of $|C|$). We also use a (potentially) weaker security assumption than Groth (PSDL versus CPDH) in the security proof.

As we mentioned above, one substantial difference with [Gro10] is that instead of an ad hoc choice of the set Λ, we use an essentially optimal one. However, this also means that the obtained efficiency estimates (see Tbl. 1) are quite close to optimal. One could still hope to replace $|C|^{1+o(1)}$ with $\Theta(|C|)$ in the case of CRS length and the prover's computation, but for this one seems to require a completely different proof technique. It is also important that in our case, the argument length and the verifier's computation are constant. Therefore, even if the prover has to spend an enormous amount of time to compute the argument, he only has to do it once. In the applications with potentially many verifiers but only one prover who has to compute the argument relatively rarely (like e-voting), this is really important. Moreover, our results show that one can build an argument with constant communication and constant verifier's computation for every language in $\mathbf{NP}$: one can just reduce the task at hand to a SAT instance.

We emphasize that Micali's computationally-sound proofs [Mic94] and Gentry's NIZK on top of fully-homomorphic cryptosystems [Gen09] have (asymptotically) longer arguments and are less efficient to verify. Thus, up to our best knowledge, the new argument for circuit satisfiability is more communication-efficient and computation-efficient for the verifier than any prior art random-oracle based NIZK argument for the same prob-

[1] 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. Thus, when we claim something to be constant within this paper, we mean a "constant number of group elements/operations" and we take into the account the fact that the size of the group depends on $|C|$. The same comment however also applies to most of the prior work.

	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	$\|C\|^{1+o(1)}G$	$28G$	$\Theta(\|C\|^2)A + \|C\|^{1+o(1)}E$	$51P$	PKE + PSDL
This paper	$\|C\|^{1/3+o(1)}G$	$\Theta(\|C\|^{2/3})G$	$\Theta(\|C\|^{4/3})A + \|C\|^{1+o(1)}E$	$\Theta(\|C\|^{2/3})P$	PKE + PSDL
This paper	$\|C\|^{1/2+o(1)}G$	$\Theta(\|C\|^{1/2})G$	$\Theta(\|C\|^{3/2})A + \|C\|^{1+o(1)}E$	$\Theta(\|C\|^{1/2})P$	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 security parameter, G is the length of group elements and A/M/E/P corresponds to a single addition/multiplication/exponentiation/pairing. For verifier, we count the total (offline + online) computation

lem. 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 for a full comparison.

Perfect Zaps. Finally, in Sect. 7, we construct a new zap [DN00] 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.

2 Preliminaries

Notation. Let $[n] = \{1, 2, \ldots, n\}$. If $y = h^x$ then $\log_h y := x$. Let κ be the security parameter.

Polynomial factorization. Within our security 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 $\mathsf{parm} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$ 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 $g_i \in \mathbb{G}_i$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$. If g_i generates $\mathbb{G}_i$, then $e(g_1, g_2)$ generates $\mathbb{G}_T$. Moreover, it is efficient to decide the membership in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, group operations and the pairing e are efficiently computable, generators are efficiently sampleable, and 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 $R = \{(C, w)\}$ be an efficiently computable binary relation such that $|w| = \operatorname{poly}(|C|)$. Here, C is a statement, and w is a witness. Let $L = \{C : \exists w, (C, w) \in R\}$ be an $\mathbf{NP}$-language. Let n be some fixed concrete input length $n = |C|$. For fixed n, we have a relation R_n and a language L_n. A *non-interactive argument* for 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 $\mathcal{V}$. In the arguments to come, σ is considerably longer than σ_v. In the case where efficiency is not important (like 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 $n = \operatorname{poly}(\kappa)$, all $\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n)$ and all $(C, w) \in R_n$, $\mathcal{V}(\sigma, C, \mathcal{P}(\sigma, C, w)) = 1$. A non-interactive argument $(\mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *computationally (adaptively) sound*, if for all non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$ and all $n = \operatorname{poly}(\kappa)$, $\Pr[\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n), (C, \pi) \leftarrow \mathcal{A}(\sigma) : C \notin L \wedge \mathcal{V}(\sigma, C, \pi) = 1]$ is negligible. Here, 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 (given that there are several possible witnesses) it is impossible to tell which witness the prover used. That is, for all $n = \mathrm{poly}(\kappa)$, if $\sigma \in \mathcal{G}_{\mathsf{crs}}(1^\kappa, n)$ and $(C, w_0), (C, w_1) \in R_n$, then the distributions $\mathcal{P}(\sigma, C, w_0)$ and $\mathcal{P}(\sigma, C, w_0)$ 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 $S = (S_1, S_2)$, such that for all stateful interactive non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$ and $n = \mathrm{poly}(\kappa)$,

$$\begin{aligned}|\Pr[\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n), (C, w) \leftarrow \mathcal{A}(\sigma), \pi \leftarrow \mathcal{P}(\sigma, C, w) : (C, w) \in R_n \wedge \mathcal{A}(\pi) = 1] - \\ \Pr[(\sigma, \mathsf{td}) \leftarrow S_1(1^\kappa, n), (C, w) \leftarrow \mathcal{A}(\sigma), \pi \leftarrow S_2(\sigma, C, \mathsf{td}) : (C, w) \in R_n \wedge \mathcal{A}(\pi) = 1]|\end{aligned}$$

is negligible in κ. Here, td is the *simulation trapdoor*.

Adaptive Co-Soundness. Co-soundness is a weaker version of soundness, where it is required that an adversary who *knows* that $C \notin L$ should not be able to produce a convincing argument that $C \in L$ (see [GL07] or the full version of [GOS06]). Thus, the adversary should not know that he succeeded in cheating. The natural way of formalizing is to require that no efficient adversary should be able to output at the same time a $C \notin L$, a witness $\bar{w}$ of the fact that $C \notin L$, and an argument that convinces the verifier that $C \in L$. Clearly, this definition only makes sense if $L \in \mathbf{NP} \cap \mathbf{coNP}$.

More formally, let $R = \{(C, w)\}$ and $L = \{C : (\exists w)(C, w) \in R\}$ be defined as earlier, with $L \in \mathbf{NP} \cap \mathbf{coNP}$. Let $\bar{R} = \{(C, \bar{w})\}$ be an efficiently computable binary relation, and let $\bar{L} = \{C : (\exists \bar{w})(C, \bar{w}) \in \bar{R}\}$ be equal to the complement of L. Similarly to above, we define a language $\bar{L}_n$ and a relation $\bar{R}_n$. A non-interactive argument $(\mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is (adaptively) *co-sound*, if for all non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$ and all $n = \mathrm{poly}(\kappa)$, $\Pr[\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n), (C, \bar{w}, \pi) \leftarrow \mathcal{A}(\sigma) : (C, \bar{w}) \in \bar{R}_n \wedge \mathcal{V}(\sigma, C, \pi) = 1]$ is negligible in κ. While circuit satisfiability is a $\mathbf{NP}$-complete language and thus does not belong to $\mathbf{NP} \cap \mathbf{coNP}$, our basic arguments (for Hadamard product and permutation) are for languages in $\mathbf{NP} \cap \mathbf{coNP}$. For the sake of completeness, we will prove that the corresponding basic arguments are both co-sound (under a computational assumption) and computationally sound (if we additionally make a knowledge assumption).

2.1 Progression-Free Sets

A set of positive integers $\Lambda = \{\lambda_1, \ldots, \lambda_n\}$ is *progression-free* [TV06], if no three of the numbers are in arithmetic progression, so that $\lambda_i + \lambda_j = 2\lambda_k$ only if $i = j = k$. Let $r_3(n)$ denote the cardinality of the largest progression-free set that belongs to $[n]$.

It is well-known that for any $n > 1$, the set of integers in $[n]$ that have no ternary digit equal to 2 is progression-free. If $n = 3^k$, then there are $2^k - 1$ such integers, and thus $r_3(n) = \Omega(n^{\log_3 2})$. This set can also be efficiently constructed. As shown by Behrend [Beh46], this idea can be generalized to non-ternary bases, with $r_3(n) = \Omega(n/(2^{2\sqrt{2\log_2 n}} \cdot \log_2^{1/4} n))$. Recently, Elkin [Elk10] improved upon Behrend's construction, by showing that $r_3(n) = \Omega(n \cdot \log_2^{1/4} n / 2^{2\sqrt{2\log_2 n}})$. For the sake of completeness, we have included a proof of the Elkin's result in App. B. Our proof is based on [GW10] but it has sharper constant inside the $o(1)$. 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 the latter progression-free set in quasi-linear time $n2^{O(\sqrt{\log n})}$ [Elk10]. On the other hand, Bourgain [Bou98] showed that $r_3(n) = O(n \cdot (\log n / \log\log n)^{1/2})$, and very 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, $y = n / \log^{1-o(1)} n$.

We need all members of the progression-free subset also to be odd. For this, one could take the set $\Lambda = \{\lambda_1, \ldots, \lambda_n\}$, existence of which is guaranteed by [Elk08], and use the set $2\Lambda + 1 = \{2\lambda_1, \ldots, 2\lambda_n + 1\}$ instead. Clearly, if $\Lambda \in [n^{1+o(1)}]$ then also $2\Lambda + 1 \in [n^{1+o(1)}]$. Let $r_3^{odd}(N)$ be the size of the largest progression-free set in $[N]$ that only consists of odd numbers. For the sake of completeness, we provide an expository proof of the next result in App. B.

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

3 Cryptographic Tools

In this section we redefine PKE and then define two new cryptographic assumptions, PDL and PSDL (all three for "nice sets" Λ), and prove that the stronger assumption 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. Here and in what follows we need the next definition. Let $n = \mathrm{poly}(\kappa)$. We say that a set Λ is (n, κ)-*nice* if $\Lambda = \{\lambda_i : i \in \{1, \ldots, n\} \wedge 0 < \lambda_1 < \lambda_2 < \cdots < \lambda_n = \mathrm{poly}(\kappa)\}$.

Λ-Power Knowledge of Exponent Assumption (Λ-PKE). Abe and Fehr showed in [AF07] that no statistically zero-knowledge non-interactive argument for an $\mathbf{NP}$-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 computationally binding commitment scheme commits to 0) seems to be an unfalsifiable assumption in general. To tackle this obstacle, Groth, Ostrovsky and Sahai [GOS06] (full version) defined co-soundness for languages in $\mathbf{NP} \cap \mathbf{coNP}$. However, since we are interested in an $\mathbf{NP}$-complete language, we cannot take their approach. Thus, similarly to Groth [Gro10], we will base our NIZK argument for circuit satisfiability on Λ-PKE on 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].

Fix $t \in \{1, 2\}$. 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 $X_{\mathcal{A}}$,

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

is negligible in κ. That is, if $\mathcal{A}$ (given access to σ that for a random α contains $g_t^{x^i}$ and $g_t^{\alpha x^i}$ only if $i \in \{0\} \cup \Lambda$) can produce c and $\hat{c}$ such that $\hat{c} = c^\alpha$, then $X_{\mathcal{A}}$ (given access to σ and to the random coins of $\mathcal{A}$) can produce a tuple $(a_i)_{i \in \{0\} \cup \Lambda}$ such that $c = \prod_{i \in \{0\} \cup \Lambda} g^{a_i x^{\lambda_i}}$. 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 $\mathcal{A}$, $\Pr[\mathsf{parm} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), g_t \leftarrow \mathbb{G}_t \setminus \{1\}, x \leftarrow \mathbb{Z}_p : \mathcal{A}(\mathsf{parm}; (g_t^{x^i})_{i \in \{0\} \cup \Lambda}) = x]$ 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 $\mathcal{A}$,

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

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. Next, we give a direct proof.

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

Proof. In the generic group model, the adversary only performs generic group operations (multiplications in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, bilinear pairings, and equality tests). Here, the generic adversary produces an element of $\mathbb{Z}_p$, which depends only on parm and $(g_1^{x^i}, g_2^{x^i})_{i \in \Lambda}$. 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 elements in unit time.

Assume that the adversary can find a collision $y = y'$ in say group $\mathbb{G}_1$. Then it must be the case that $y = \prod_{i \in \Lambda} g_1^{a_i x^i}$ and $y' = \prod_{i \in \Lambda} g_1^{a'_i x^i}$ for some known values of a_i and a'_i. But then also $\sum_{i \in \Lambda} (a_i - a'_i) x^i \equiv 0 \pmod p$. 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 modulo p, the probability that $\sum_{i \in \Lambda} (a_i - a'_i) x^i \equiv 0 \pmod p$ is equal to λ_n / p for randomly chosen a_i and

a'_i. If the adversary works in polynomial time $t = \text{poly}(\kappa)$, it can generate at most t such group elements. The total probability that there exists a collision between any two generated group elements is thus upper bounded by $\binom{t}{2} \cdot \lambda_n/p$, and thus a successful adversary requires time $\Omega(\sqrt{p/\lambda_n})$ to produce one collision.

A similar bound holds for collusions in $\mathbb{G}_2$. In the case of $\mathbb{G}_T$, the pairing enables the adversary to compute up to t different values $y = e(g_1, g_2)^{\sum_{i\in\Lambda}\sum_{j\in\Lambda} a_{ij}x^{i+j}}$, and thus we get an upper bound $\binom{t}{2} \cdot 2\lambda_n/p$, and thus a successful adversary requires time $\Omega(\sqrt{p/\lambda_n})$ to produce one collision. □

Trapdoor Commitment Schemes in the CRS Model. An (asymmetric) commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\text{om})$ 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 $\mathcal{C}\text{om}$. Here, $\mathcal{G}_{\mathsf{com}}(1^\kappa, t)$, $t \in \{1, 2\}$, produces a CRS σ_t^{c}, and $\mathcal{C}\text{om}^t(\sigma_t^{\mathsf{c}}; \boldsymbol{a}; r)$ outputs a commitment value A in $\mathbb{G}_t$. A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\text{om})$ is *computationally binding in group* $\mathbb{G}_t$, if for every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$, the probability $\Pr[\sigma_t^{\mathsf{c}} \leftarrow \mathcal{G}_{\mathsf{com}}(1^\kappa, t), (\boldsymbol{a_1}, r_1, \boldsymbol{a_2}, r_2) \leftarrow \mathcal{A}(\sigma_t^{\mathsf{c}}) : (\boldsymbol{a_1}, r_1) \neq (\boldsymbol{a_2}, r_2) \wedge \mathcal{C}\text{om}^t(\sigma_t^{\mathsf{c}}; \boldsymbol{a_1}; r_1) = \mathcal{C}\text{om}^t(\sigma_t^{\mathsf{c}}; \boldsymbol{a_2}; r_2)]$ is negligible. A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\text{om})$ is *perfectly hiding in group* $\mathbb{G}_t$, if for any $\sigma_t^{\mathsf{c}} \in \mathcal{G}_{\mathsf{com}}(1^\kappa, t)$ and any two messages a_1, a_2, the distributions $\mathcal{C}\text{om}^t(\sigma_t^{\mathsf{c}}; a_1; \cdot)$ and $\mathcal{C}\text{om}^t(\sigma_t^{\mathsf{c}}; a_2; \cdot)$ are equal.

We use a variant of the *knowledge commitment scheme* from [Gro10] with a generalized choice of generators, defined as follows:

CRS generation: Let Λ be a (n, κ)-nice set with $n = \text{poly}(\kappa)$. Define $\lambda_0 = 0$. Given a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$, set $\mathsf{parm} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ be generators, and choose random $\alpha, \beta, x \leftarrow \mathbb{Z}_p$. Fix $t \in \{1, 2\}$. The CRS is $\sigma_t^{\mathsf{c}} \leftarrow (\mathsf{parm}; (g_{t,i}, \hat{g}_{t,i}, \tilde{g}_{t,i})_{i\in\{0,\ldots,n\}})$, where $g_{t,i} = g_t^{x^{\lambda_i}}$, $\hat{g}_{t,i} = g_t^{\alpha x^{\lambda_i}}$, and $\tilde{g}_{t,i} = g_t^{\beta x^{\lambda_i}}$. Note that, for example, $g_t = g_{t,0}$.

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 $\mathcal{C}\text{om}^t(\sigma_t^{\mathsf{c}}; \boldsymbol{a}; r) := (g_t^r \cdot \prod_{i=1}^n g_{t,i}^{a_i},\ \hat{g}_t^r \cdot \prod_{i=1}^n \hat{g}_{t,i}^{a_i},\ \tilde{g}_t^r \cdot \prod_{i=1}^n \tilde{g}_{t,i}^{a_i})$.

Thus, for $f(x) := r + \sum_{i=1}^n a_i x^{\lambda_i}$, $\mathcal{C}\text{om}^t(\sigma_t^{\mathsf{c}}; \boldsymbol{a}; r) = (g_t^{f(x)}, g_t^{\alpha f(x)}, g_t^{\beta f(x)})$. In subsequent Hadamard product and permutation arguments, we will only need two of the three components, while in the argument for circuit satisfiability, we use either two or all three.

Theorem 3. *Let $t \in \{1, 2\}$. The knowledge commitment scheme is statistically hiding in group $\mathbb{G}_t$, and computationally binding in group $\mathbb{G}_t$ under the n-PDL assumption in group $\mathbb{G}_t$. If the Λ-PKE assumption holds in group $\mathbb{G}_t$, 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 as an input the input of $\mathcal{A}$ together with $\mathcal{A}$'s random coins, extracts the contents of the commitments.*

Proof. HIDING: follows from the fact that the output of $\mathcal{C}\text{om}^t$ is a random element of $\mathbb{G}_t$. 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 n-PDL assumption in group $\mathbb{G}_t$ that works with the same probability. Let $\mathsf{parm} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $g_t \leftarrow \mathbb{G}_t \setminus \{1\}$, and $x \leftarrow \mathbb{Z}_p$. The adversary $\mathcal{A}'$ gets $(\mathsf{parm}; (g_t^{x^i})_{i\in\{0\}\cup\Lambda})$ as her input. She creates random $\alpha', \beta' \leftarrow \mathbb{Z}_p$, and sets $\sigma_t^{\mathsf{c}} \leftarrow (\mathsf{parm}; (g_t^{x^i}, g_t^{\alpha x^i}, g_t^{\beta x^i})_{i\in\{0\}\cup\Lambda})$. She forwards σ_t^{c} to $\mathcal{A}$. By definition, $\mathcal{A}(\sigma^{\mathsf{c}})$ 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 $g_{t,\lambda_1} = g_t^{y^{\lambda_1}}$. Set $x \leftarrow y$. Thus, $\mathcal{A}'$ has broken the n-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. (This can be achieved only using 2 out of 3 components of the commitment.) Since the commitment scheme is computationally binding, then the extracted opening has to be the same that $\mathcal{A}$ used. □

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

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 $\Lambda_1 = (2\lambda_i)_{i\in[n]}$, and Λ_2 be as in Eq. (1), and $\Lambda_3 = \{\lambda_i + \lambda_j : i, j \in [n] \wedge i \neq j\}$.

CRS generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Let $\mathsf{parm} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $\alpha, \beta, x \leftarrow \mathbb{Z}_p$. Let $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$ and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. Denote $g_{t,i} \leftarrow g_t^{x^i}$, $\hat{g}_{t,i} \leftarrow g_t^{\alpha x^i}$, and $\tilde{g}_{t,i} \leftarrow g_t^{\beta x^i}$ for $t \in \{1,2\}$ and $i \in [2\lambda_n]$. Let $D \leftarrow \prod_{i=1}^n g_{2,\lambda_i}$. The CRS is $\sigma \leftarrow (\mathsf{parm}; (g_{1,i}, \hat{g}_{1,i}, \tilde{g}_{1,i}, \tilde{g}_{2,i})_{i\in\{0\}\cup\Lambda}, (g_{2,i}, \hat{g}_{2,i})_{i\in\Lambda_2}, D)$. Set $\sigma_t^{\mathsf{c}} \leftarrow (\mathsf{parm}; (g_{t,i}, \hat{g}_{t,i}, \tilde{g}_{t,i})_{i\in\{0\}\cup\Lambda})$ for $t \in \{1,2\}$, and let $\sigma_v \leftarrow (\mathsf{parm}; g_1, \hat{g}_1, g_2, \hat{g}_2, D)$ be the verifier's part of σ.

Common inputs: Commitments $A = (A_0, A_1, A_2) \leftarrow \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{a}; r_a)$, $B = (B_0, B_1, B_2) \leftarrow \mathcal{C}\mathrm{om}^2(\sigma_2^{\mathsf{c}}; \boldsymbol{b}; r_b)$, $C = (C_0, C_1, C_2) \leftarrow \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{c}; r_c)$.

Argument generation $\mathcal{P}(\sigma; (A, B, C), (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, \boldsymbol{c}, r_c))$**:** The prover defines $p_k \leftarrow \sum_{i,j\neq i:\lambda_i+\lambda_j=k}(a_i b_j - c_i)$ for $k \in \Lambda_3$, and then sets $\pi \leftarrow g_{2,0}^{r_a r_b} \cdot \prod_{i=1}^n g_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{k\in\Lambda_3} g_{2,k}^{p_k}$, and $\hat{\pi} \leftarrow \hat{g}_{2,0}^{r_a r_b} \cdot \prod_{i=1}^n \hat{g}_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{k\in\Lambda_3} \hat{g}_{2,k}^{p_k}$. The prover sends $(\pi, \hat{\pi}) \in \mathbb{G}_2^2$ to the verifier as the argument.

Verification $\mathcal{V}(\sigma_v; (A, B, C), (\pi, \hat{\pi}))$**:** $\mathcal{V}$ accepts iff $e(A_0, B_0)/e(C_0, D) = e(g_1, \pi)$ and $e(g_1, \hat{\pi}) = e(\hat{g}_1, \pi)$.

Protocol 1: New argument for Hadamard product

4 New Product Argument

In a *Hadamard product argument*, the prover aims to convince the verifier that for given three commitments A, B and C, he knows how to open them as $A = \mathcal{C}\mathrm{om}^1(\sigma^{\mathsf{c}}; \boldsymbol{a}; r_a)$, $B = \mathcal{C}\mathrm{om}^2(\sigma^{\mathsf{c}}; \boldsymbol{b}; r_b)$, and $C = \mathcal{C}\mathrm{om}^1(\sigma^{\mathsf{c}}; \boldsymbol{c}; r_c)$, such that $c_j = a_j \cdot b_j$ for $j \in [n]$. We denote this non-interactive argument by $[\![A]\!] \circ [\![B]\!] = [\![C]\!]$; note that the order of A and B is important. The case where $A, B, C \in \mathbb{G}_1$ will be only slightly less efficient, however, we will not need it in this paper. In [Gro10], Groth constructed a Hadamard product argument where the communication and the verifier's online computation are $\Theta(1)$, the verifier's offline computation is $\Theta(n)$, and the prover's computation and the length of the common reference string (CRS) are $\Theta(n^2)$.

In this section, we propose a more efficient argument, see 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. That formal polynomial has a monomial γx^δ with non-zero γ exactly if either $\delta \in \Lambda_1 := \{2\lambda_i : i \in [n]\}$, or $\delta \in \Lambda_2$, where

$$\Lambda_2 := \{0\} \cup \Lambda \cup \{\lambda_i + \lambda_j : i, j \in [n] \wedge i \neq j\} \ . \tag{1}$$

We need that Λ_1 and Λ_2 do not intersect. The next lemma is straightforward to prove.

Lemma 1. *1) If Λ is a progression-free set of odd integers, then $\Lambda_1 \cap \Lambda_2 = \emptyset$. 2) If $\Lambda_1 \cap \Lambda_2 = \emptyset$, then Λ is a progression-free set.*

Moreover, since $\Lambda_2 \in \{0, \ldots, \lambda_{n-1} + \lambda_n\}$, then by Thm. 1,

Lemma 2. *Fix n. There exists a choice of Λ, such that $|\Lambda_2| = n^{1+o(1)}$.*

We are now ready to prove the security of the new Hadamard product argument. Let $\eta := e(A_0, B_0)/e(C_0, D)$. Very briefly, the (co-)soundness of this argument is based on the next facts: 1) $\log_h \eta$ is a polynomial in x with only powers from $\Lambda_1 \cup \Lambda_2$ and the prover is honest iff $\log_h \eta$ is a polynomial in x with powers from Λ_2, 2) π only contains powers from Λ_2, 3) σ contains generators $g_2^{\alpha x^\delta}$ only for $\delta \in \Lambda_2$, 4) $\Lambda_1 \cap \Lambda_2 = \emptyset$ (by Lem. 1), 5) $\hat{\pi}$ is computed exactly like π but by using generators $g_2^{\alpha x^i}$ instead of $g_2^{x^i}$, and 6) by the Λ-PSDL assumption and the Λ-PKE assumption (only required for soundness) in groups $\mathbb{G}_1$ and $\mathbb{G}_2$. We note that one can add additional elements $g_{2,i}$ and $\tilde{g}_{2,i}$ (without adding corresponding elements $\hat{g}_{2,i}$) 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 commitments A, B and C by verifying that $e(A_0, \hat{g}_2) = e(A_1, g_2)$, $e(\hat{g}_1, B_0) = e(g_1, B_1)$, and $e(C_0, \hat{g}_2) = e(C_1, g_2)$. If the bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is Λ_2-PSDL secure, then Prot. 1 is adaptively co-sound. If $\mathcal{G}_{\mathsf{bp}}$ is additionally Λ-PKE secure in both $\mathbb{G}_1$ and $\mathbb{G}_2$, then Prot. 1 is computationally adaptively sound.*

Here and in Sect. 5, 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 σ^{c}, the commitment scheme is defined as in Sect. 3 with respect to the set Λ.

Proof. Let $\eta \leftarrow e(A_0, B_0)/e(C_0, D)$ and $h \leftarrow e(g_1, g_2)$. WITNESS-INDISTINGUISHABILITY: follows from the fact that there is exactly one possible argument $(\pi, \hat{\pi})$ that satisfies the verification equations.

PERFECT COMPLETENESS. The second and the third verifications are straightforward. For the first one, note that

$$\begin{aligned}\log_h \eta =&(r_a + \sum_{i=1}^{n} a_i x^{\lambda_i})(r_b + \sum_{i=1}^{n} b_i x^{\lambda_i}) - (r_c + \sum_{i=1}^{n} c_i x^{\lambda_i})(\sum_{i=1}^{n} x^{\lambda_i}) \\ =&\underbrace{\sum_{i=1}^{n}(a_i b_i - c_i)x^{2\lambda_i}}_{\delta \in \Lambda_1} + \underbrace{r_a r_b + \sum_{i=1}^{n}(r_a b_i + r_b a_i - r_c)x^{\lambda_i} + \sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n}(a_i b_j - c_i)x^{\lambda_i+\lambda_j}}_{\delta \in \Lambda_2}.\end{aligned}$$

Thus, the right-hand side of $\log_h \eta$ is equal to a sum of some powers x^δ of x for $\delta \in \Lambda_1$ and $\delta \in \Lambda_2$. If the prover is honest, then $c_i = a_i \cdot b_i$ for $i \in [n]$, and thus $\log_h \eta$ is a formal polynomial with only monomials γx^δ with $\delta \in \Lambda_2$. Thus, denoting $h_i \leftarrow h^{x^i}$, $\eta = h^{r_a r_b} \cdot \prod_{i=1}^{n} h_{\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{i=1}^{n}\prod_{j=1:j\neq i}^{n} h_{\lambda_i+\lambda_j}^{a_i b_j - c_i}$. Therefore, $\pi \leftarrow g_{2,0}^{r_a r_b} \cdot \prod_{i=1}^{n} g_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{i=1}^{n}\prod_{j=1:j\neq i}^{n} g_{2,\lambda_i+\lambda_j}^{a_i b_j - c_i} = g_{2,0}^{r_a r_b} \cdot \prod_{i=1}^{n} g_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{k\in\Lambda_3} g_{2,k}^{p_k}$, and $\hat{\pi}$ is computed similarly. Thus, if the prover is honest, then the first verification succeeds.

COMPUTATIONAL (CO-)SOUNDNESS. Assume that $\mathcal{A}$ is an adversary, that can break the adaptive soundness of the Hadamard product argument. That is, given a $\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa)$, $\mathcal{A}(\sigma)$ outputs $((A, B, C), (\pi, \hat{\pi}))$, such that $\mathcal{V}(\sigma; (A, B, C), (\pi, \hat{\pi}))$ accepts. For soundness only, if the Λ-PKE assumption is not true in either $\mathbb{G}_1$ or $\mathbb{G}_2$, then we are done, and thus we can assume that the Λ-PKE assumption holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$. For co-soundness, we do not need the PKE assumption.

Next, we construct an adversary $\mathcal{A}'$ against the Λ_2-PSDL assumption. Let $\mathsf{parm} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $x \leftarrow \mathbb{Z}_p$, $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$, and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. The adversary $\mathcal{A}'$ receives $\sigma \leftarrow (\mathsf{parm}; (g_1^{x^i}, g_2^{x^i})_{i\in\{0\}\cup\Lambda_2})$ as her input, and her task is to output x. She sets $\alpha, \beta \leftarrow \mathbb{Z}_p$, $\sigma' \leftarrow (\mathsf{parm}; (g_1^{x^i}, g_1^{\alpha x^i}, g_1^{\beta x^i}, g_2^{\beta x^i})_{i\in\{0\}\cup\Lambda}, (g_2^{x^i}, g_2^{\alpha x^i})_{i\in\Lambda_2}, \prod_{i=1}^{n} g_2^{x^{\lambda_i}})$, 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 $\sigma_t^{\mathsf{c}} \leftarrow (\mathsf{parm}; (g_t^{x^i}, g_t^{\alpha x^i}, g_t^{\beta x^i})_{i\in\{0\}\cup\Lambda})$ for $t \in \{1, 2\}$. According to the previous paragraph, $\mathcal{A}$ returns $((A, B, C), (\pi, \hat{\pi}))$ such that $\mathcal{V}(\sigma'; (A, B, C), (\pi, \hat{\pi}))$ accepts.

If we prove soundness, then by the Λ-PKE assumption in groups $\mathbb{G}_1$ and $\mathbb{G}_2$, there exists an efficient extractor $X_{\mathcal{A}}$, that on input σ' (and $\mathcal{A}$'s random coins) produces openings $(\boldsymbol{a}; r_a)$, $(\boldsymbol{b}; r_b)$ and $(\boldsymbol{c}; r_c)$ such that $A = \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{a}; r_a)$, $B = \mathcal{C}\mathrm{om}^2(\sigma_2^{\mathsf{c}}; \boldsymbol{b}; r_b)$ and $C = \mathcal{C}\mathrm{om}^1(\sigma_2^{\mathsf{c}}; \boldsymbol{c}; r_c)$. If we prove co-soundness, then the PKE assumption is not necessary, since $\mathcal{A}'$ herself will output such openings. Assume that $\mathcal{A}$ was successful, that is, for some $j \in [n]$, $c_j \neq a_j \cdot b_j$. Since $\Lambda_1 \cap \Lambda_2 = \emptyset$, $\mathcal{A}'$ has thus expressed $\log_h \eta$ as a polynomial $f(x)$ where at least for some $i \in \Lambda_1$, x^i has a non-zero coefficient.

On the other hand, $\mathcal{A}$ also outputs $(\pi, \hat{\pi})$. Thus, if we prove soundness, then by the Λ-PKE assumption in group $\mathbb{G}_2$, there exists another extractor $X'_{\mathcal{A}}$, that given access to σ' and to $\mathcal{A}$'s coins, outputs values $(\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, \boldsymbol{c}, r_c)$ (*not* equal to those output by $X_{\mathcal{A}}$), such that $\log_h \eta = \log_{g_2} \pi = f'(x)$, where all non-zero coefficients of $f'(x)$ correspond to x^j for some $j \in \Lambda_2$. If we prove co-soundness, then the PKE assumption is not necessary since $\mathcal{A}$ herself will output such opening. Now, since Λ is a progression-free set, then by Lem. 1, $j \notin \Lambda_1$. Therefore, all coefficients of $f'(x)$ corresponding to any x^i, $i \in \Lambda_1$, are equal to 0. Thus $f(X) = \sum_{i\in\{0\}\cup\Lambda_2} f_i X^i$ and $f'(X) = \sum_{i\in\{0\}\cup\Lambda_2} f'_i X^i$ are different polynomials with $f(x) = f'(x) = \log_h \eta$. Therefore, $\mathcal{A}'$ has succeeded in creating a non-zero polynomial d, such that $d(x) = \sum_{i\in\{0\}\cup\Lambda_2} d_i x^i = 0$.

Next, $\mathcal{A}'$ can use an efficient polynomial factorization algorithm in $\mathbb{Z}_p[X]$ to efficiently compute all $|\Lambda_2| + 1$ 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 Λ_2-PSDL assumption. □

Theorem 5. *Let Λ be as described in Thm. 1. The communication (argument size) of the argument in Prot. 1 is two elements from $\mathbb{G}_2$. Prover's computational complexity is $\Theta(n^2)$ multiplications in $\mathbb{Z}_p$ and $n^{1+o(1)}$ exponentiations in $\mathbb{G}_2$. 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.*

Proof. It is clear that with this choice of Λ, by Lem. 2, the size of the CRS is $\Theta(|\Lambda_2|) = n^{1+o(1)}$. From the CRS, the verifier clearly only needs to access g_1, $\hat{g}_1$, g_2, $\hat{g}_2$ and D. Since $\Lambda_3 \subseteq \Lambda_2$, the statement about prover's computational complexity follows. Prover's computational complexity comes from the fact that one can compute all values p_k simultaneously by looping over n^2 possible values of i and j. Verifier's computational complexity follows from the description of the argument. □

Comparison to [Gro10]. There are several differences between this argument, and between the corresponding product argument in [Gro10] that we already outlined in the introduction. In addition, note that in our application (NIZK argument for circuit satisfiability) all values a_i, b_i and c_i are Boolean, and thus all $n^{1+o(1)}$ values p_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 table lookups):

1. For $k \in \Lambda_3$ do: $p_k \leftarrow 0$
2. For $i = 1$ to n do:
   - If $a_i = 0$ then do:
     (a) For $j = 1$ to n do: if $j \neq i$ then $p_{\lambda_i+\lambda_j} \leftarrow p_{\lambda_i+\lambda_j} - c_i$
   - Else do: // if $a_i = 1$
     (a) For $j = 1$ to n do: if $j \neq i$ then $p_{\lambda_i+\lambda_j} \leftarrow p_{\lambda_i+\lambda_j} + b_j - c_i$

Second (though this is a minor point), Groth claimed that his product argument is (adaptively) sound only under a computationally assumption. He achieved it by explicitly assuming that $\mathcal{A}$ can output the openings of the commitments it produces. Thus, actually, Groth's argument only provides co-soundness, which is possible since the Hadamard product language belongs to $\mathbf{NP} \cap \mathbf{coNP}$. However, since we need soundness (and not co-soundness) of this argument later, we prefer to rely on the explicit knowledge assumption that provides $\mathcal{A}'$ with an extractor that outputs the openings. As a drawback, this means that the verifier has to check the correctness of commitments; however, in Groth's argument of circuit satisfiability this is anyways necessary. (For the sake of completeness, we also proved co-soundness without using the PKE assumption.)

Third, 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 argument and of the proof. Fourth, 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. Fifth, by giving a somewhat clearer proof, we were able to avoid Groth's CPDH assumption, and relied on the (potentially weaker) PSDL assumption instead. Sixth, we have added D to the CRS, which means that verifier's computation is constant. In Groth's argument, it is assumed that the verifier computes D.

5 New Permutation Argument

In a *permutation argument*, the prover aims to convince the verifier that for given permutation $\varrho : [n] \to [n]$ and two commitments A and B, he knows how to open them as $A = \mathcal{C}\mathrm{om}^1(\sigma^{\mathsf{c}}; \boldsymbol{a}; r_a)$ and $B = \mathcal{C}\mathrm{om}^1(\sigma^{\mathsf{c}}; \boldsymbol{b}; r_b)$, such that $b_j = a_{\varrho(j)}$ for $j \in [n]$. Note that ϱ can be either as a common input to the prover and the verifier, or generated by the prover as a part of the argument, and sent publicly to the verifier. In our applications, ϱ is a common input. We denote this non-interactive argument by $\varrho([\![A]\!]) = [\![B]\!]$. In [Gro10], Groth constructed a Hadamard permutation argument where the communication and the verifier's online computation are $\Theta(1)$, the verifier's offline 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 that is closely related to that of Groth in [Gro10], but with the CRS of length $n^{1+o(1)}$. In addition, prover's concrete computation will be improved, and it is based on a (probably) weaker assumption. (A more thorough comparison is provided at the end of this section.)

The proposed permutation argument, see Prot. 2, follows the same outline that was described in the beginning of Sect. 4. The logarithm of the left side of the first verification equation in Prot. 2 is a degree-n formal polynomial in x. The set of powers of x that are present in this formal polynomial are divided into the following two types: either 1) $\delta \in \Lambda_1 = \{2\lambda_i : i \in [n]\}$, or 2) $\delta \in \Lambda \cup \{\lambda_{\varrho(i)} - \lambda_i : i \in [n]\} \cup \{\lambda_i + \lambda_j : i, j \in [n] \wedge i \neq j\} \cup \{2\lambda_{\varrho(j)} + \lambda_i - \lambda_j : i, j \in [n] \wedge i \neq j\} \subseteq \Lambda'_2$, where

$$\Lambda'_2 := \Lambda \cup \{\lambda_i - \lambda_j : i, j \in [n]\} \cup \{\lambda_i + \lambda_j : i, j \in [n] \wedge i \neq j\} \cup \{2\lambda_k + \lambda_i - \lambda_j : i, j, k \in [n] \wedge i \neq j\} \quad (2)$$

for $i, j \in [n]$, $i \neq j$ and ϱ being an arbitrary permutation. Note that $0 \in \Lambda'_2$, and also that $\Lambda_2 \subseteq \Lambda'_2$.

Lemma 3. *1) If Λ is a progression-free set of odd integers, then $\Lambda_1 \cap \Lambda'_2 = \emptyset$. 2) If $\Lambda_1 \cap \Lambda'_2 = \emptyset$, then Λ is a progression-free set.*

(A short proof is given in App. D.) Moreover, since $\Lambda'_2 \subset \{-\lambda_n + 1, \ldots, 3\lambda_n\}$ then from Thm. 1 we get

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 $\Lambda_1 = (2\lambda_i)_{i\in[n]}$, and Λ'_2 be as in Eq. (2). Let $\Lambda'_3 := \{\lambda_i + \lambda_i : i, j \in [n] \wedge i \neq j\} \cup \{2\lambda_{\varrho(j)} + \lambda_i - \lambda_j : i, j \in [n] \wedge i \neq j\} \subset \{-\lambda_n + 1, \ldots, 3\lambda_n\}$.

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

Common inputs: Commitments $A = (A_0, A_1, A_2) \leftarrow \mathcal{C}\mathrm{om}^1(\sigma^{\mathsf{c}}; \boldsymbol{a}; r_a)$, $B = (B_0, B_1, B_2) \leftarrow \mathcal{C}\mathrm{om}^1(\sigma^{\mathsf{c}}; \boldsymbol{b}; r_b)$, and a permutation $\varrho : [n] \to [n]$.

Argument generation $\mathcal{P}(\sigma; (A, B), (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b))$: $\mathcal{P}$ sets $p_k \leftarrow \sum_{i,j\neq i:\lambda_i+\lambda_j=k} a_i + \sum_{i,j\neq i:2\lambda_{\varrho(j)}+\lambda_i-\lambda_j=k} b_i$ for $k \in \Lambda'_3$. He sets $\pi \leftarrow D^{r_a} \cdot E^{r_b} \cdot \prod_{k\in\Lambda'_3} g_{2,k}^{p_k}$ and $\tilde{\pi} \leftarrow \tilde{D}^{r_a} \cdot \tilde{E}^{r_b} \cdot \prod_{k\in\Lambda'_3} \tilde{g}_{2,k}^{p_k}$. The prover sends $(\pi, \tilde{\pi}) \in \mathbb{G}_2$ to the verifier as the argument.

Verification $\mathcal{V}(\sigma_v; (A, B), (\pi, \tilde{\pi}))$: If $e(A_0, D)/e(B_0, E) = e(g_1, \pi)$ and $e(g_1, \tilde{\pi}) = e(\tilde{g}_1, \pi)$, then $\mathcal{V}$ accepts. Otherwise, $\mathcal{V}$ rejects.

Protocol 2: New argument for permutation

Lemma 4. *Fix n. There exists a choice of Λ, such that $|\Lambda'_2| = n^{1+o(1)}$.*

We are now ready to prove the security of the new permutation argument. The soundness of this argument is based on exactly the same ideas as that of the Hadamard product argument.

Theorem 6. *The permutation argument in 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 $e(A_0, \tilde{g}_2) = e(A_2, g_2)$, and $e(B_0, \tilde{g}_2) = e(B_2, g_2)$. If the bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is Λ'_2-PSDL secure, then Prot. 2 is adaptively co-sound. If $\mathcal{G}_{\mathsf{bp}}$ is additionally Λ-PKE secure in both $\mathbb{G}_1$ and $\mathbb{G}_2$, then Prot. 2 is computationally adaptively sound.*

The security proof is similar to the proof of Thm. 4, and is given in App. E.

Theorem 7. *Let Λ be as described in Thm. 1. The communication (the argument size) of the argument in Prot. 2 is two elements from $\mathbb{G}_2$. Prover's computational complexity is dominated by $\Theta(n^2)$ additions in $\mathbb{Z}_p$ and by $n^{1+o(1)}$ exponentiations in $\mathbb{G}_2$. Verifier's computational complexity is dominated by 5 bilinear pairings. The CRS consists of $n^{1+o(1)}$ group elements, and the verifier's part of the CRS consists of the bilinear group description plus 6 group elements.*

Proof of this theorem is given in App. F.

Comparison to [Gro10]. The relation between this product argument and the one of Groth [Gro10] is similar to the relation between the new Hadamard product argument and the one of Groth. See the comparison between the latter in Sect. 4.

6 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 $2n + 1$ dimensional vectors, where n is the circuit size. Those three arguments can be seen as basic operations in a NIZK "programming language". Following [Gro10], 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 committed to online by both of the parties separately). When choosing the arguments, one has to keep in mind that we work in asymmetric setting, and thus for example the second input to the Hadamard product argument comes from a different group than the first and the third inputs. 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). The CRS is basically the union of CRSs for the product and permutation arguments, though it contains separate "E_j^{ϱ}" elements for every used permutation ϱ. The total argument itself consists of commitments to intermediate variables and of all arguments in the program of this "programming language". We note that formally, every commitment is a triple of the group elements, but if this commitment is used either only in permutation arguments or only in product arguments, then only 2 out of the 3 group elements have to be forwarded to the verifier. 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 is only composed of the NAND gates. Let the circuit 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 the resulting circuit 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 gate $R_{n+1} = 1$. 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, (U, X) is a ζ-permutation of (L, R). (See Fig. 1 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:

- $\tau : [2n+1] \to [2n+1]$ 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, $\tau(i) = i$.
- $\zeta : [2n+1] \to [2n+1]$ 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.
- $\varrho : [2n+1] \to [2n+1]$ 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.

Theorem 8. *Let the bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ be Λ_2'-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 argument for SAT.*

Proof. PERFECT COMPLETENESS: follows from that of the product and permutation arguments.

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 soundness of the product and permutation arguments and by the Λ_2'-PSDL assumption, it follows that the corresponding relations are satisfied between the opened values. Moreover, by the Λ_2'-PSDL assumption, the opened values belong to corresponding sets Λ_2 and Λ_2'. Let (L, R, R_{n+1}) be the opening of LR, where $L = (L_1, \ldots, L_n)$ and $R = (R_1, \ldots, R_n)$, and let $(U_1, \ldots, U_n, X_1, \ldots, X_n, X_{n+1})$ be the opening of UX. We now analyze the effect of every subargument in Prot. 3. See Fig. 1 for a visual reference.

Successful verification of $e(g_1, \mathsf{LR}_0') = e(\mathsf{LR}_0, g_2)$ shows that LR' commits to (L, R, R_{n+1}) in $\mathbb{G}_2$, and thus the first (product) argument shows that $L_i, R_i \in \{0, 1\}$. The second (permutation) argument shows that RL commits to (R, L, R_{n+1}). The third (product) argument shows that RZ commits to $(R, 0, \ldots, 0, 0)$ and is thus consistent with the opening of LR. The fourth argument shows that 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 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 UZ commits to $(U_1, \ldots, U_n, 0, \ldots, 0, 0)$, then $E \cdot \mathsf{UZ}^{-1}$ 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.

System parameters: Define Λ and Λ_2 as in Prot. 1 and Λ_2' as in Prot. 2, but in all cases with n replaced by $2|C|+1$.
Common inputs: Permutations τ, ζ and ϱ.
Common reference string generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$: Set $\mathsf{parm} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Generate random $\alpha, \beta \leftarrow \mathbb{Z}_p$, $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$, $g_2 \leftarrow \mathbb{G}_2 \setminus \{2\}$. Denote $g_{t,i} \leftarrow g_t^{x^i}$, $\hat{g}_{t,i} \leftarrow g_t^{\alpha x^i}$ and $\tilde{g}_{t,i} \leftarrow g_t^{\beta x^i}$ for $t \in \{1,2\}$ and $i \in [2|C|+1]$. Let $\hat{g}_t \leftarrow \hat{g}_{t,0}$ and $\tilde{g}_t \leftarrow \tilde{g}_{t,0}$. Set $(D_0, D_1, D_2) \leftarrow (\prod_{i=1}^n g_{2,\lambda_i}, \prod_{i=1}^n \hat{g}_{2,\lambda_i}, \prod_{i=1}^n \tilde{g}_{2,\lambda_i})$, $(E_0^\zeta, E_2^\zeta) \leftarrow (\prod_{i=1}^n g_{2,2\lambda_{\zeta(i)}-\lambda_i}, \prod_{i=1}^n \tilde{g}_{2,2\lambda_{\zeta(i)}-\lambda_i})$, $(E_0^\tau, E_2^\tau) \leftarrow (\prod_{i=1}^n g_{2,2\lambda_{\tau(i)}-\lambda_i}, \prod_{i=1}^n \tilde{g}_{2,2\lambda_{\tau(i)}-\lambda_i})$, $(E_0^\varrho, E_2^\varrho) \leftarrow (\prod_{i=1}^n g_{2,2\lambda_{\varrho(i)}-\lambda_i}, \prod_{i=1}^n \tilde{g}_{2,2\lambda_{\varrho(i)}-\lambda_i})$, and $(F_0, F_1, F_2) \leftarrow (\prod_{i=1}^{n-1} g_{1,\lambda_i} \cdot g_{1,\lambda_n}^2, \prod_{i=1}^{n-1} \hat{g}_{1,\lambda_i} \cdot \hat{g}_{1,\lambda_n}^2, \prod_{i=1}^{n-1} \tilde{g}_{1,\lambda_i} \cdot \tilde{g}_{1,\lambda_n}^2)$. The CRS is $\sigma \leftarrow (\mathsf{parm}; (g_{1,i}, \hat{g}_{1,i}, \tilde{g}_{1,i})_{i\in\{0\}\cup\Lambda}, (\hat{g}_{2,i})_{i\in\Lambda_2}, (g_{2,i}, \tilde{g}_{2,i})_{i\in\Lambda_2'}, D_0, D_1, D_2, E_0^\zeta, E_2^\zeta, E_0^\tau, E_2^\tau, E_0^\varrho, E_2^\varrho, F_0, F_1, F_2)$. Let $\sigma_t^{\mathsf{c}} \leftarrow (\mathsf{parm}; (g_{t,i}, \hat{g}_{t,i}, \tilde{g}_{t,i})_{i\in\{0\}\cup\Lambda})$ for $t \in \{1,2\}$, and let $\sigma_v \leftarrow (\mathsf{parm}; g_1, \hat{g}_1, \tilde{g}_1, g_2, \hat{g}_2, \tilde{g}_2, D_0, E_0^\zeta, E_0^\tau, E_0^\varrho)$ be the verifier's part of σ.
Argument generation $\mathcal{P}(\sigma; (n, \tau, \zeta))$: Denote $\boldsymbol{Y} := (Y_1, \ldots, Y_n)$ for $Y \in \{L, R, U, X\}$. The prover does the following things.

1. He sets $r_1, \ldots, r_5 \leftarrow \mathbb{Z}_p$, and then computes $\mathsf{LR} \leftarrow \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{L}, \boldsymbol{R}, R_{n+1}; r_1)$, $\mathsf{RL} \leftarrow \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{R}, \boldsymbol{L}, R_{n+1}; r_1)$, $\mathsf{RZ} \leftarrow \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{R}, 0, \ldots, 0, 0; r_2)$, $\mathsf{UZ} \leftarrow \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{U}, 0, \ldots, 0, 0; r_3)$, $\mathsf{UX} \leftarrow \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{U}, \boldsymbol{X}, X_{n+1}; r_4)$, $\mathsf{LR}' \leftarrow \mathcal{C}\mathrm{om}^2(\sigma_2^{\mathsf{c}}; \boldsymbol{L}, \boldsymbol{R}, R_{n+1}; r_5)$. Note that every commitment is a triple of form $A = (A_0, A_1, A_2)$.
2. $\mathcal{P}$ creates an argument π_1 for $[\![\mathsf{LR}]\!] \circ [\![\mathsf{LR}']\!] = [\![\mathsf{LR}]\!]$.
3. $\mathcal{P}$ creates an argument π_2 for $\varrho([\![\mathsf{LR}]\!]) = [\![\mathsf{RL}]\!]$.
4. $\mathcal{P}$ creates an argument π_3 for $[\![\mathsf{RL}]\!] \circ [\![(D_0, D_1, D_2)]\!] = [\![\mathsf{RZ}]\!]$.
5. $\mathcal{P}$ creates an argument π_4 for $[\![\mathsf{UX}]\!] \circ [\![(D_0, D_1, D_2)]\!] = [\![\mathsf{UZ}]\!]$.
6. $\mathcal{P}$ creates an argument π_5 for $[\![\mathsf{RZ}]\!] \circ [\![\mathsf{LR}']\!] = [\![(F_0, F_1, F_2) \cdot \mathsf{UZ}^{-1}]\!]$.
7. $\mathcal{P}$ creates an argument π_6 for $\tau([\![\mathsf{LR}]\!]) = [\![\mathsf{LR}]\!]$.
8. $\mathcal{P}$ creates an argument π_7 for $\zeta([\![\mathsf{LR}]\!]) = [\![\mathsf{UX}]\!]$.
9. He sends $\pi \leftarrow (\mathsf{LR}_0, \mathsf{LR}_1, \mathsf{LR}_2, \mathsf{LR}_0', \mathsf{LR}_1', \mathsf{RL}_0, \mathsf{RL}_2, \mathsf{RZ}_0, \mathsf{RZ}_1, \mathsf{UZ}_0, \mathsf{UZ}_1, \mathsf{UX}_0, \mathsf{UX}_1, \mathsf{UX}_2, \pi_1, \ldots, \pi_7)$ to the verifier.

Verification $\mathcal{V}(\sigma_v; (n, \tau, \zeta); \pi)$: The verifier does the following things:

- For $A \in \{\mathsf{LR}, \mathsf{RZ}, \mathsf{UZ}, \mathsf{UX}\}$ check that $e(A_1, g_2) = e(A_0, \hat{g}_2)$.
- Check that $e(g_1, \mathsf{LR}_1') = e(\hat{g}_1, \mathsf{LR}_0')$.
- For $A \in \{\mathsf{LR}, \mathsf{RL}, \mathsf{UX}\}$ check that $e(A_2, g_2) = e(A_0, \tilde{g}_2)$.
- She verifies that $e(\hat{g}_1, \mathsf{LR}_0') = e(\mathsf{LR}_0, \hat{g}_2)$.
- She verifies all 7 arguments.

Protocol 3: New NIZK argument for SAT

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 an argument, the simulator creates LR, LR', RL, RZ, UZ and 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 the one on step 6 where he uses $U_i = 0$ instead. (Obviously, also RZ and 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''. □

Theorem 9. *Let Λ be chosen as in Thm. 1. The communication (argument length) of the argument in Prot. 3 is* 12 *elements from $\mathbb{G}_1$ and* 16 *elements from $\mathbb{G}_2$. Prover's computational complexity is dominated by $\Theta(|C|^2)$ simple arithmetical operations in $\mathbb{Z}_p$ and $|C|^{1+o(1)}$ exponentiations in $\mathbb{G}$. Verifier's computational complexity is dominated by* 51 *bilinear pairings. The CRS consists of $|C|^{1+o(1)}$ group elements, and the verifier's part of the CRS consists of the description of the bilinear group plus* 10 *group elements.*

Proof. This argument contains 6 commitments and 7 arguments, all together $14+7\cdot 2 = 28$ group elements. From those group elements, 12 belong to $\mathbb{G}_1$ and $2+7\cdot 2 = 16$ belong to $\mathbb{G}_2$. Verifier has to perform $2\cdot(4+1+3) = 16$ bilinear pairings to check the correctness of all commitments, and $7 \cdot 5 = 35$ bilinear pairings to check all 7 arguments. (To check that $e(\hat{g}_1, \mathsf{LR}'_0) = e(\mathsf{LR}_0, \hat{g}_2)$, she can reuse previous computation.) Statements about prover's computational complexity and the length of (the verifier's part of) the CRS follow directly from Thm. 5 and Thm. 7. For 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). □

Finally, the communication can be optimized even more by noticing that the group elements A_2, for $A \in \{\mathsf{LR}, \mathsf{LR}', \mathsf{RZ}, \mathsf{UZ}, \mathsf{UX}\}$, and the group elements A_1, for $A \in \{\mathsf{LR}, \mathsf{RL}, \mathsf{UX}\}$ are never used and thus do not have to be transferred. Then, the communication only consists of 8 elements of $\mathbb{G}_1$ and 16 elements of $\mathbb{G}_2$. 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 6400 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 prover's computation from $\Theta(n^2)$ exponentiations in $\mathbb{G}_t$ to $\Theta(n^2)$ (conditional) additions in $\mathbb{Z}_p$. Third, since we have added elements $D_0,\ldots,F_2$ to the CRS, the verifier's computation is constant instead of linear in Groth's argument. Fourth, 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, 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, it is the other way around. Therefore, we saved 14 group elements, reducing the argument size from 42 to 28 group elements. 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.

7 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 $\mathbf{NP}$ under a knowledge assumption (and without random oracles), but their argument is not witness-indistinguishable.

Acknowledgments. The author was supported by Estonian Science Foundation, grant #8058, and European Union through the European Regional Development Fund.

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.

[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.

[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.

[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.

[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.

[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.

[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.

[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.

[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 $j \in [q]$, $\Pr[\mathsf{parm} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), g_t \leftarrow \mathbb{G}_t \setminus \{1\}, (x,\alpha) \leftarrow \mathbb{Z}_p^2 : \mathcal{A}(\mathsf{parm}; (g_t^{x^i})_{i\in\{0,\ldots,q\}}, (g_t^{\alpha x^i})_{i\in\{0,\ldots,q\}\setminus\{j\}}) = g_t^{\alpha x^j}]$ 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 $r \le \frac{1}{2}\cdot\sqrt{n}$, let

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

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

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

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

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

The volume of n-dimensional ball $B_n(R) := \{x \in \mathbb{R}^n : ||x||_2 \le R\}$ 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 $r \le \sqrt{n/12} + 1/(2\sqrt{5}) + O(n^{-1/2}) = \sqrt{n/12} + O(1)$, then $\mathrm{vol}(B_n(\sqrt{2\delta r})) \approx 1/\sqrt{\pi n}\cdot$

$(4e\pi\delta r/n)^{n/2} \le 1/\sqrt{\pi n}\cdot(4e\pi\delta(\sqrt{n/12}+O(1))/n)^{n/2} = 1/\sqrt{\pi n}\cdot(2e\pi\delta(1+O(n^{-1/2}))/\sqrt{3n})^{n/2} = 1/\sqrt{\pi n}\cdot(2e\pi\delta/\sqrt{3n})^{n/2}(1+O(n^{-1/2}))$.

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

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

Then $\mathbb{E}[\frac{2}{3}|A_{\theta,\alpha}| - T(A_{\theta,\alpha})] = \mathbb{E}[\frac{2}{3}|A_{\theta,\alpha}|] - \mathbb{E}[T(A_{\theta,\alpha})] = \frac{2}{3}\cdot N\,\mathrm{vol}(S) - \frac{1}{4}\cdot N(N-2)\,\mathrm{vol}(B) \ge \frac{1}{3}\cdot N\,\mathrm{vol}(S)$. Clearly, thus there is a choice $A = A_{\theta,\alpha}$ for which both $T(A) \le 2|A|/3$ and $|A| \ge \frac{1}{2}\cdot N\,\mathrm{vol}(S)$. 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. (3) to hold, we must have

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

or

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

or

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

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

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

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

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

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

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

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. 5

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 $\mathbb{E}[X_i^2] = 2 \cdot \int_0^{1/2} x^2 dx = 1/12$, $\mathbb{E}[X_i^4] = 2 \cdot \int_0^{1/2} x^4 dx = 1/80$, $\mathrm{var}[X_i^2] = \mathbb{E}[X_i^4] - \mathbb{E}[X_i^2]^2 = 1/180$, $\mathbb{E}[||X||_2^2] = \sum_{i=1}^n \mathbb{E}[X_i^2] = n/12$, $\mathrm{var}[||X||_2^2] = \sum_{i=1}^n \mathrm{var}[X_i^2] = n/180$. By Chebyshev's inequality,

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

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 $\mathrm{vol}(S_n(r, \delta)) \geq 4\sqrt{5}\delta/3 \cdot 2^{-n} \cdot (1 - O(n^{-1/2}))$ as required. □

D Proof of Lem. 3

Proof. FIRST CLAIM: Let $t \in \Lambda_1$, that is, $t = 2\lambda_\ell$ for some $\ell \in [n]$, and assume that $t \in \Lambda_1 \cap \Lambda_2'$. Then $t \notin \Lambda$ since t is not odd, and $t \notin \{\lambda_i + \lambda_j : i \neq j\}$ since Λ is progression-free. Assume now that $t = 2\lambda_k + \lambda_i - \lambda_j$ for $i \neq j$, that is, $2(\lambda_\ell - \lambda_k) = \lambda_i - \lambda_j$ for $i \neq j$. But since Λ is a set of odd integers, $4 \mid 2(\lambda_\ell - \lambda_k)$ while $4 \nmid (\lambda_i - \lambda_j)$, a contradiction. SECOND CLAIM: If Λ is not a progression-free set, then for some $i \neq j$ and k, there exists $t := 2\lambda_k = \lambda_i + \lambda_j$, and thus $t \in \Lambda_1 \cap \Lambda_2'$. □

E Proof of Thm. 6

Proof. Denote $\eta := e(A_0, D)/e(B_0, E)$. Let $h \leftarrow e(g_1, g_2)$. WITNESS-INDISTINGUISHABILITY: follows straightforwardly from the fact that there is exactly one possible argument $(\pi, \tilde{\pi})$ that satisfies the verification equations.

PERFECT COMPLETENESS. The second verification is straightforward. For the first one, note that

$$\begin{aligned}\log_h \eta &= (r_a + \sum_{i=1}^n a_i x^{\lambda_i})(\sum_{i=1}^n x^{\lambda_i}) - (r_b + \sum_{i=1}^n b_i x^{\lambda_i})(\sum_{i=1}^n x^{2\lambda_{\varrho(i)} - \lambda_i}) \\ &= \underbrace{\sum_{i=1}^n (a_{\varrho(i)} - b_i) x^{2\lambda_{\varrho(i)}}}_{\delta \in \Lambda_1} + \underbrace{r_a \sum_{i=1}^n x^{\lambda_i} - r_b \sum_{i=1}^n x^{\lambda_{\varrho(i)} - \lambda_i} + \sum_{i=1}^n \sum_{\substack{j=1 \\ j \neq i}}^n a_i x^{\lambda_i + \lambda_j} - \sum_{i=1}^n \sum_{\substack{j=1 \\ j \neq i}}^n b_i x^{2\lambda_{\varrho(j)} + \lambda_i - \lambda_j}}_{\delta \in \Lambda_2'} \ .\end{aligned}$$

Thus, the right-hand side is equal to a sum of some powers x^δ of x for $\delta \in \Lambda_1$ and $\delta \in \Lambda_2'$. If the prover is honest, then $b_i = a_{\varrho(i)}$ for $i \in [n]$, and thus $\log_h \eta$ is a formal polynomial with only monomials γx^δ with $\delta \in \Lambda_2$. Defining $\pi = (\prod_{i=1}^n g_{2,\lambda_i})^{r_a} \cdot (\prod_{i=1}^n g_{2,\lambda_{\varrho(i)} - \lambda_i})^{r_b} \cdot \prod_{i=1}^n \prod_{\substack{j=1 \\ j \neq i}}^n (g_{2,\lambda_i + \lambda_j}^{a_i} \cdot g_{2,2\lambda_{\varrho(j)} + \lambda_i - \lambda_j}^{b_i}) = D^{r_a} \cdot E^{r_b} \cdot \prod_{k \in \Lambda_3'} g_{2,k}^{p_k}$, we see that the first verification equation holds.

COMPUTATIONAL (CO-)SOUNDNESS. Assume that $\mathcal{A}$ is an adversary, that can break the adaptive soundness of the Hadamard product argument. That is, given a $\sigma \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa)$, $\mathcal{A}(\sigma)$ outputs $((A, B, C), (\pi, \hat{\pi}))$, such that

$\mathcal{V}(\sigma; (A, B, C), (\pi, \hat{\pi}))$ accepts. For soundness only, if the Λ-PKE assumption is not true in either $\mathbb{G}_1$ or $\mathbb{G}_2$, then we are done and thus we can assume that the Λ-PKE assumption holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$. For co-soundness, we do not need the PKE assumption.

Next, we construct an adversary $\mathcal{A}'$ against the Λ'_2-PSDL assumption. Let $\mathsf{parm} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $x \leftarrow \mathbb{Z}_p$, $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$, and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. The adversary $\mathcal{A}'$ receives $\sigma \leftarrow (\mathsf{parm}; (g_1^{x^i}, g_2^{x^i})_{i \in \{0\} \cup \Lambda'_2})$ as her input, and her task is to output x. She sets $\alpha, \beta \leftarrow \mathbb{Z}_p$, and $\sigma' \leftarrow (\mathsf{parm}; (g_1^{x^i}, g_1^{\alpha x^i}, g_1^{\beta x^i}, g_2^{\alpha x^i})_{i \in \{0\} \cup \Lambda}, (g_2^{x^i}, g_2^{\beta x^i})_{i \in \Lambda'_2}, \prod_{i=1}^n g_2^{x^{\lambda_i}}, \prod_{i=1}^n \tilde{g}_2^{x^{\lambda_i}}, \prod_{i=1}^n g_2^{x^{2\lambda_{\varrho(i)} - \lambda_i}}, \prod_{i=1}^n \tilde{g}_2^{x^{2\lambda_{\varrho(i)} - \lambda_i}})$, and forwards σ' to $\mathcal{A}$. Clearly, σ' follows the distribution imposed by $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$. Both parties also set $\sigma_t^{\mathsf{c}} \leftarrow (\mathsf{parm}; (g_t^{x^i}, g_t^{\alpha x^i})_{i \in \{0\} \cup \Lambda})$ for $t \in \{1, 2\}$. According to the previous paragraph, $\mathcal{A}$ returns $((A, B, C), (\pi, \hat{\pi}))$ such that $\mathcal{V}(\sigma'; (A, B, C), (\pi, \hat{\pi}))$ accepts.

If we prove soundness, then by the Λ-PKE assumption in group $\mathbb{G}_1$, there exists an efficient extractor $X_{\mathcal{A}}$, that on input σ' (and $\mathcal{A}$'s random coins) produces openings $(\boldsymbol{a}; r_a)$ and $(\boldsymbol{b}; r_b)$, such that $A = \mathcal{C}\mathrm{om}^1(\sigma_1^{\mathsf{c}}; \boldsymbol{a}; r_a)$ and $B = \mathcal{C}\mathrm{om}^2(\sigma_2^{\mathsf{c}}; \boldsymbol{b}; r_b)$. If we prove co-soundness, then the PKE assumption is not necessary, since $\mathcal{A}'$ herself will output such openings. Assume that $\mathcal{A}$ was successful, that is, for some $j \in [n]$, $b_j \neq a_{\varrho(j)}$. Since $\Lambda_1 \cap \Lambda'_2 = \emptyset$, $\mathcal{A}'$ has thus expressed $\log_h \eta$ as a polynomial $f(x)$ where at least for some $i \in \Lambda_1$, x^i has a non-zero coefficient.

On the other hand, $\mathcal{A}$ also outputs $(\pi, \hat{\pi})$. Thus, if we prove soundness, then by the Λ-PKE assumption in group $\mathbb{G}_2$, there exists another extractor $X'_{\mathcal{A}}$, that given access to σ' and $\mathcal{A}$'s coins, outputs values $(\boldsymbol{a}, r_a, \boldsymbol{b}, r_b)$ (*not* equal to those output by $X_{\mathcal{A}}$), such that $\log_h \eta = \log_{g_2} \pi = f'(x)$, where all non-zero coefficients of $f'(x)$ correspond to x^j for some $j \in \Lambda_2$. If we prove co-soundness, then the PKE assumption is not necessary since $\mathcal{A}$ herself will output such opening. Now, since Λ is a progression-free set, then by Lem. 3, $j \notin \Lambda_1$. Therefore, all coefficients of $f'(x)$ corresponding to any x^i, $i \in \Lambda_1$, are equal to 0. Thus $f(X) \cdot X^{\lambda_n} = \sum_{i \in \Lambda'_2} f_i X^{i + \lambda_n}$ and $f'(X) = \sum_{i = \Lambda'_2} f'_i X^{i + \lambda_n}$ are different polynomials with $f(x) = f'(x) = \log_h \eta$. Therefore, $\mathcal{A}'$ has succeeded in creating a nonzero polynomial d such that $d(x) = \sum_{i \in \Lambda'_2} d_i x^{i + \lambda_n} = 0$.

Next, $\mathcal{A}'$ can use an efficient polynomial factorization algorithm [vHN10] in $\mathbb{Z}_p[X]$ to efficiently compute all $|\Lambda'_2| + 1$ 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 Λ'_2-PSDL assumption. □

F Proof of Thm. 7

Proof. It is clear that with this choice of Λ, by Lem. 4, the size of the CRS is $\Theta(|\Lambda'_2|) = n^{1 + o(1)}$. From the CRS, the verifier clearly only needs to access g_1, $\tilde{g}_1$, g_2, $\tilde{g}_2$, D and E. Since $\Lambda'_3 \subseteq \Lambda'_2$, the statement about prover's computational complexity follows. Provers computational complexity comes from the fact that one can compute all values p_k simultaneously by looping over n^2 possible values of i and j, as follows:

1. For $k \in \Lambda_3$ do: $p_k \leftarrow 0$
2. For $i = 1$ to n do:
   - For $j = 1$ to n do: if $j \neq i$ then $p_{\lambda_i + \lambda_j} \leftarrow p_{\lambda_i + \lambda_j} + a_i$
3. For $j = 1$ to n do:
   - Set $t \leftarrow 2\lambda_{\rho(j)} - \lambda_j$
   - For $i = 1$ to n do: if $j \neq i$ then $p_{t + \lambda_i} \leftarrow p_{t + \lambda_i} + b_i$

Here, one has to do $\Theta(n^2)$ additions and $\Theta(n)$ computations of ϱ (say, by using table-lookup). Verifier's computational complexity follows from the description of the argument. □

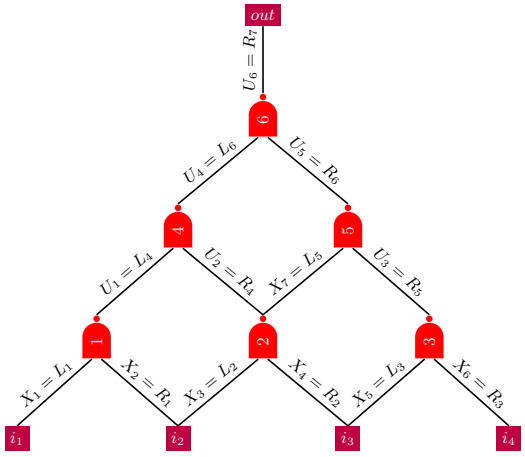


Fig. 1. Example circuit