A More Efficient Computationally Sound Non-Interactive Zero-Knowledge Shuffle Argument

Helger Lipmaa[1] and Bingsheng Zhang[2]

[1] University of Tartu, Estonia
[2] State University of New York at Buffalo, USA

Abstract. We propose a new non-interactive (perfect) zero-knowledge (NIZK) shuffle argument that, when compared the only previously known efficient NIZK shuffle argument by Groth and Lu, has a small constant factor times smaller computation and communication, and is based on more standard computational assumptions. Differently from Groth and Lu who only prove the co-soundness of their argument under purely computational assumptions, we prove computational soundness under a necessary knowledge assumption. We also present a general transformation that results in a shuffle argument that has a quadratically smaller common reference string (CRS) and a small constant factor times times longer argument than the original shuffle.
Keywords. Bilinear pairings, cryptographic shuffle, non-interactive zero-knowledge, progression-free sets.

1 Introduction

In a shuffle argument, the prover proves that two tuples of randomized ciphertexts encrypt the same multiset of plaintexts. Such an argument is needed in e-voting and anonymous broadcast. In the case of e-voting, shuffles are used to destroy the relation between the voters and their ballots. There, the voters first encrypt their ballots. The ciphertexts are then sequentially shuffled by several independent mix servers, where every server also produces a zero-knowledge shuffle argument. At the end, all shuffle arguments are verified and the final ciphertexts are threshold-decrypted. If all arguments are accepted, then the shuffle is correct. Moreover, as long as one mix server is honest, the shuffle remains private (that is, one cannot relate the voters and their ballots).

A lot of research has been conducted in the area of constructing secure and efficient shuffle arguments, with recent work resulting in shuffles that have sublinear communication and very competitive computational complexity. However, it is also important that the shuffle argument is non-interactive, due to the fact that non-interactive arguments are transferable (create once, verify many times without interacting with the prover). This is especially important in e-voting, where the correctness of e-voting (and thus of the shuffle) should be verifiable in years to come. Practically all previous shuffle arguments are interactive, and can only be made non-interactive by using the Fiat-Shamir heuristic, that is, in the random oracle model. For example, Groth and Ishai [13], Groth [11], and Bayer and Groth [2] have constructed shuffle arguments with communication $\Theta(n^{2/3})$, $\Theta(n^{1/2})$, and $\Theta(n^{1/2})$ respectively, where n is the number of ciphertexts. Unfortunately, they make use of the Schwartz-Zippel lemma that

	$\lvert$CRS$\rvert$	Comm.	$\mathcal{P}$'s comp.	$\mathcal{V}$'s comp.	Pairing	Sound	Assumption
[14]	$2n+8$	$15n+120$	$51n+246$	$75n+282$	Sym.	Co-	PPA+SPA+DLIN
Sect. 5	$7n+6$	$6n+11$	$17n+16$	$28n+18$	Asym.	Sound	PKE+PSDL+DLIN
[19]	$7\sqrt{n}+6$	$30n+33\sqrt{n}$	$63n+48\sqrt{n}$	$84n+54\sqrt{n}$	Asym.	Sound	PKE+PSDL+DLIN

Table 1. Brief comparison of existing (not random-oracle based) and new (two last ones) NIZK shuffle arguments. Here, the communication complexity and the CRS length are given in group elements, prover's computation is given in exponentiations, and verifier's computation is given in (symmetric or asymmetric) bilinear pairings

requires the verifier to first provide a random input. The only known way to make the Schwartz-Zippel lemma based arguments non-interactive is to use the random oracle model. Unfortunately, it is well-known that there are protocols that are secure in the random oracle model but not in the plain model. Even if there are no similar distinguishing attacks against any of the existing shuffle arguments, it is prudent to design alternative non-interactive shuffle arguments that are not based on random oracle model.

The only known (not random-oracle based) efficient non-interactive zero-knowledge (NIZK) shuffle argument (for the BBS cryptosystem [3]) was proposed by Groth and Lu in [14]. The security of the Groth-Lu argument is based on the common reference string model and on two new computational assumptions, the permutation pairing assumption (PPA, see [14]) and the simultaneous pairing assumption (SPA). While Groth and Lu proved that their assumptions are secure in the generic group model, one can argue that their assumptions are specifically constructed so as the concrete shuffle argument will be co-sound [16] (see [14] and Sect. 2 for discussions on co-soundness). It is therefore interesting to construct a shuffle argument from "more standard" assumptions. Moreover, their shuffle argument has a relatively large computational complexity and communication complexity. (See Tbl. 1 for a comparison.)

We construct a new non-interactive shuffle argument that has better communication and is based on more standard computational security assumptions than the Groth-Lu argument. Full comparison between the Groth-Lu and the new argument is given later. Recall that permutation matrix is a Boolean matrix that has exactly one 1 in every row and column. From a very high-level point of view, following [9] and subsequent papers, we let the prover to commit to a permutation matrix and then present an efficient permutation matrix argument (given commitments commit to a permutation matrix). We then prove that the plaintext vector corresponding to the output ciphertext vector is equal to the product of this matrix and the plaintext vector corresponding to the input ciphertext vector, and thus is correctly formed. Both parts are involved. In particular, coming up with a characterization of permutation matrices that allows for an efficient cryptographic implementation was not a trivial task.

Terelius and Wikström [23] constructed an interactive permutation matrix argument based on the fact that a matrix is a permutation matrix iff its every column sums to 1 and its every row has exactly one non-zero element. To verify that the committed matrix satisfies these properties, they used the Schwartz-Zippel lemma with the verifier sending a random vector to the prover. This introduces interaction (or the use of a random oracle). We do not know how to prove efficiently in NIZK that a commitment commits to a unit vector; how to construct such an *efficient* argument is an interesting open problem. We propose a superficially similar permutation matrix argument that is based on the (related) fact that a matrix is a permutation matrix exactly if every column sums to 1 and every row has *at most* one non-zero element. However, we do not explicitly use the Schwartz-Zippel lemma, and this makes it possible for us to create a NIZK argument without using the random oracle model.

Cryptographically, the new permutation matrix argument is based on recent techniques of Groth [12] and Lipmaa [18] who proposed an NIZK argument for circuit satisfiability based on two subarguments, for Hadamard — that is, entry-wise — product and permutation. (The same basic arguments were then used in [4] to construct an efficient non-interactive range proof.) Unfortunately, in their subarguments, the prover has quadratic (or quasilinear $O(n2^{2\sqrt{2\log_2 n}})$, if one only counts the group operations) computational complexity. This is not acceptable in our case, and therefore *we do not use any of the arguments that were constructed in [12, 18]*.

We propose 2 new basic arguments (a zero argument, see Sect. 3.1, and a 1-sparsity argument, see Sect. 3.2), and then combine them in Sect. 3.3 to form a permutation matrix argument. The zero argument (the prover can open the given commitment to the zero tuple) can be interpreted as a knowledge of the discrete logarithm argument, and is a special case of Groth's restriction argument from [12]. On the other hand, the 1-sparsity argument (the prover can open the given commitment to a tuple $\boldsymbol{a} = (a_1, \ldots, a_n)$, where at most one coordinate a_i is non-zero) is conceptually new.

Like the basic arguments of [18], the new 1-sparsity argument relies on the existence of a dense progression-free set. However, the costs of the 1-sparsity argument do not depend explicitly on the size of the used progression-free sets. Briefly, in [18] and the new 1-sparsity argument, the discrete logarithm of the non-interactive argument is equal to the sum of two polynomials $F_{con}(x)$ and $F_\pi(x)$, where x is the secret key. The first polynomial F_{con} has exactly one monomial per constraint that a honest prover has to satisfy. The number of constraints is linear (for any i, $a_i \cdot b_i = c_i$) in [18] and quadratic (for any two different coefficients a_i and a_j, $a_i \cdot a_j = 0$) in the new 1-sparsity argument. The second polynomial consists of monomials (a quasilinear number $O(n2^{2\sqrt{2\log_2 n}})$ in [18] and a linear number in the new 1-sparsity argument) that have to be computed by a honest prover during the argument, and this is the main reason why both the CRS length and the prover's computational complexity are lower in the 1-sparsity argument compared to the arguments in [18]. We find this to be an interesting result by itself, leading to an obvious question whether similar arguments (that have a superlinear number of constraints and a linear number of spurious monomials) can be used as an underlying engine to construct other interesting NIZK proofs.

In Sect. 5, we combine the permutation matrix argument with a knowledge version of the BBS [3] cryptosystem to obtain an efficient NIZK shuffle argument. Informally, by the KE assumption [6], in the knowledge BBS cryptosystem (defined in Sect. 4) the ciphertext creator knows both the used plaintext and the randomizer. Since it is usually not required that the ciphertext creator also knows the randomizer, the knowledge BBS cryptosystem satisfies a stronger than usual version of plaintext-awareness. While this version of plaintext-awareness has not been considered in the literature before, it is also satisfied by the Damgård's Elgamal cryptosystem from [6].

According to [1], only languages in $\mathbf{P/poly}$ can have direct black-box *perfect* NIZK arguments.[1] Since all known constructions of NIZK arguments use direct black-box reductions, one can argue that the "natural" definition of soundness is not the right definition of soundness for perfect NIZK arguments, see [14] for more discussion. To overcome the impossibility results of [1], Groth and Lu [14] proved co-soundness [14, 16] of their argument under purely computational assumptions.

Our subarguments (the zero argument, the 1-sparsity argument, and the permutation matrix argument) are not computationally sound since their languages are based on a perfectly hiding commitment scheme, see Sect. 3. Instead, we prove that these arguments satisfy a weak version of soundness [12, 18] under purely computational assumptions. We could use a similar definition of the weak soundness of the shuffle argument and prove that the new shuffle argument is (weakly) sound by using only standard computational assumptions. Instead (mostly since computational soundness is a considerably more standard security requirement), we prove computational soundness of the shuffle argument under a (known) knowledge assumption. This is also the reason why we need to use the *knowledge* BBS cryptosystem.

Apart from the knowledge assumption, the security of the new shuffle argument is based on the DLIN assumption [3] (which is required for the CPA-security of the BBS cryptosystem), and on the power symmetric discrete logarithm (PSDL, see Sect. 2) assumption from [18]. The PSDL assumption is much more standard(-looking) than the SPA and PPA assumptions from [14].

Tbl. 1 provides a comparison between [14] and the new shuffle argument. Since it was not stated in [14], we have calculated ourselves[2] the computational complexity of the Groth-Lu argument. As seen from Tbl. 1, the new argument is computationally about 2.5 to 3 times more efficient and communication-wise about 2 times more efficient, if one just counts the number of exponentiations (in the case of the prover's computation), pairings (verifier's computation), or group elements (communication). In addition, the new argument uses asymmetric pairings $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, while [14] uses symmetric pairings with $\mathbb{G}_1 = \mathbb{G}_2$. This means in particular that the difference in efficiency is larger than seen from Tbl. 1. First, asymmetric pairings themselves are much more efficient than symmetric pairings. Second, if asymmetric pairings were used in the Groth-Lu shuffle, one would have to communicate two different versions (one in group $\mathbb{G}_1$ and another one in group $\mathbb{G}_2$) of some of the group elements.

The main drawback of the new shuffle argument is that its soundness relies additionally on a knowledge assumption. However, a non-standard assumption is necessary to achieve perfect zero-knowledge [1]. Differently from the random oracle assumption that is known to be false in general, knowledge assumptions are just known to be non-

[1] It is not necessary to have a perfect NIZK argument for a shuffle (one could instead construct a computational NIZK proof), but the techniques of both [14] and especially of the current paper are better suited to construct *efficient* perfect NIZK arguments. We leave it as an open question to construct a computational NIZK proof for shuffle with a comparable efficiency.

[2] Our calculations are based on the Groth-Sahai proofs [17] that were published after the Groth-Lu shuffle argument. The calculations may be slightly imprecise.

falsifiable and thus might be true for any practical purposes. (In comparison, the Groth-Lu argument was proven to be co-sound, which is a weaker version of computational soundness, under purely computational assumptions.)

Moreover, the Groth-Lu shuffle uses the BBS cryptosystem (where one ciphertext is 3 group elements), while we use the new knowledge BBS cryptosystem (6 group elements). This difference is small compared to the reduction in the argument size. The use of knowledge BBS cryptosystem corresponds to adding a proof of knowledge of the plaintexts (and the randomizers) by the voters. However, it means that in the proof of soundness, we show security only against (white-box) adversaries who have access to the secret coins of all voters and mixservers. It is a reasonable compromise, comparable to the case in interactive (or Fiat-Shamir heuristic based) shuffles where the ballots are accompanied by a proof of knowledge of the ballot, from which either the adversary of the simulator can obtain the actual votes, but without the use of a random oracle, see Sect. 5 for more discussion. As we note there, our soundness definition follows that of [14], but the mentioned issues are due to the use of a knowledge assumption. We hope that the current work will motivate more research on clarifying such issues.

Another drawback of our scheme as compared to [14] is that it uses a lifted cryptosystem, and thus can be only used to shuffle small plaintexts. This is fine in applications like e-voting (where the plaintext is a candidate number). Many of the existing e-voting schemes are based on (lifted) Elgamal and thus require the plaintexts to be small. We note that significant speedups can be achieved in both cases by using efficient multi-exponentiation algorithms and thus for a meaningful computational comparison, one should implement the shuffle arguments.

In the full version [19], we show that one can transform both the Groth-Lu argument and the new argument, by using the Clos network [5], to have a CRS of size $\Theta(\sqrt{n})$ while increasing the communication and computation by a small constant factor. This version of the new argument is computationally/communication-wise only slightly less efficient than the Groth-Lu argument but has a quadratically smaller CRS, see Tbl. 1. This transformation can be applied to any shuffle argument that has linear communication and computation, and a CRS of length $f(n) = \Omega(1)$. We pose it as an open problem to construct (may be using similar techniques) an NIZK shuffle argument where both the CRS and the communication are sublinear.

Due to the lack of space, some proofs are only given in the full version [19].

2 Preliminaries

Let $[n] = \{1, 2, \ldots, n\}$. If $y = h^x$, then let $\log_h y := x$. To help readability in cases like $g_2^{r_i + x^{\lambda_{\psi^{-1}(i)}}}$, we sometimes write $\exp(h, x)$ instead of h^x. Let κ be the security parameter. PPT denotes probabilistic polynomial time. For a tuple of integers $\Lambda = (\lambda_1, \ldots, \lambda_n)$ with $\lambda_i < \lambda_{i+1}$, let $(a_i)_{i \in \Lambda} = (a_{\lambda_1}, \ldots, a_{\lambda_n})$. We sometimes denote $(a_i)_{i \in [n]}$ as $\boldsymbol{a}$. We say that $\Lambda = (\lambda_1, \ldots, \lambda_n) \subset \mathbb{Z}$ is an (n, κ)-*nice tuple*, if $0 < \lambda_1 < \cdots < \lambda_i < \cdots < \lambda_n = \mathrm{poly}(\kappa)$. Let S_n be the set of permutations from $[n]$ to $[n]$.

By using notation that is common in additive combinatorics [22], if Λ_1 and Λ_2 are subsets of some additive group ($\mathbb{Z}$ or $\mathbb{Z}_p$ within this paper), then $\Lambda_1 + \Lambda_2 = \{\lambda_1 + \lambda_2 : \lambda_1 \in \Lambda_1 \wedge \lambda_2 \in \Lambda_2\}$ is their *sum set* and $\Lambda_1 - \Lambda_2 =$

$\{\lambda_1 - \lambda_2 : \lambda_1 \in \Lambda_1 \wedge \lambda_2 \in \Lambda_2\}$ is their *difference set*. In particular, if Λ is a set, then $k\Lambda = \{\sum_{i=1}^{k} \lambda_i : \lambda_i \in \Lambda\}$ is an *iterated sumset*. On the other hand, $k \cdot \Lambda = \{k\lambda : \lambda \in \Lambda\}$ is a *dilation* of Λ. We also let $2\hat{}\Lambda = \{\lambda_1 + \lambda_2 : \lambda_1 \in \Lambda \wedge \lambda_2 \in \Lambda \wedge \lambda_1 \neq \lambda_2\} \subseteq \Lambda + \Lambda$ to denote a *restricted sumset*.

A set $\Lambda = \{\lambda_1, \dots, \lambda_n\}$ of integers is *progression-free* [22], if no three elements of Λ are in arithmetic progression, that is, $\lambda_i + \lambda_j = 2\lambda_k$ only if $i = j = k$. Let $r_3(N)$ denote the cardinality of the largest progression-free set that belongs to $[N]$. Recently, Elkin [7] showed that $r_3(N) = \Omega((N \cdot \log^{1/4} N)/2^{2\sqrt{2 \log_2 N}})$. On the other hand, it is known that $r_3(N) = O(N(\log \log N)^5 / \log N)$ [21]. Thus, according to [21], the minimal N such that $r_3(N) = n$ is $\omega(n)$, while according to Elkin, $N = O(n 2^{2\sqrt{2 \log_2 n}}) = n^{1+o(1)}$. Thus, for any fixed $n > 0$, there exists $N = n^{1+o(1)}$, such that $[N]$ contains an n-element progression-free subset [18].

While the efficiency of arguments from [18] directly depended on the choice of the progression-free set, in our case the only thing dependent on this choice is the tightness of most of our security reductions; see the definition of PSDL below, or the proofs of Thm. 2, Thm. 4 and Thm. 5. Due to this, one may opt to use a less dense (but easy to construct) progression-free set. As an example, Erdős and Turán [8] defined a set $T(n)$ of all integers up to n that have no number 2 in their ternary presentation. Clearly, $|T(n)| \approx n^{\log_3 2} \approx n^{0.63}$ and $T(n)$ is progression-free. One can obtain a dense set of progression-free odd positive integers by mapping every a in $T(n)$ to $2a + 1$.

A *bilinear group generator* $\mathcal{G}_{\mathsf{bp}}(1^\kappa)$ outputs $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$ 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, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map (pairing), and $g_t \leftarrow \mathbb{G}_t \setminus \{1\}$ is a random generator of $\mathbb{G}_t$ for $t \in \{1, 2\}$. Additionally, it is required that (a) $\forall a, b \in \mathbb{Z}$, $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$, (b) $\hat{e}(g_1, g_2)$ generates $\mathbb{G}_T$, and (c) it is efficient to decide the membership in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, the group operations and the pairing $\hat{e}$ are efficiently computable, generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ are efficiently sampleable, and the descriptions of the groups and group elements each are $O(\kappa)$ bit long. One can represent an element of $\mathbb{G}_1$/$\mathbb{G}_2$/$\mathbb{G}_T$ in respectively 512/256/3072 bits, by using an optimal (asymmetric) Ate pairing over a subclass of Barreto-Naehrig curves.

A public-key cryptosystem $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{D}\mathsf{ec})$ is a tuple of efficient algorithms, where $\mathcal{G}_{\mathsf{bp}}$ is a bilinear group generator that outputs gk, $\mathcal{G}_{\mathsf{pkc}}(\mathsf{gk})$ generates a secret/public key pair $(\mathsf{sk}, \mathsf{pk})$, randomized encryption algorithm $\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu; r)$ produces a ciphertext c, and deterministic decryption algorithm $\mathcal{D}\mathsf{ec}_{\mathsf{sk}}(c)$ produces a plaintext μ. It is required that for all $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $(\mathsf{sk}, \mathsf{pk}) \in \mathcal{G}_{\mathsf{pkc}}(\mathsf{gk})$ and for all valid μ and r, $\mathcal{D}\mathsf{ec}_{\mathsf{sk}}(\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu; r)) = \mu$. Assume that the randomizer space $\mathcal{R}$ is efficiently sampleable. A public-key cryptosystem $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{D}\mathsf{ec})$ is *CPA-secure*, if for all stateful non-uniform PPT adversaries $\mathcal{A}$, the following probability is negligible in κ:

$$\left| \Pr \left[\begin{matrix} \mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{G}_{\mathsf{pkc}}(\mathsf{gk}), (\mu_0, \mu_1) \leftarrow \mathcal{A}(\mathsf{pk}), \\ b \leftarrow \{0, 1\}, r \leftarrow \mathcal{R} : \mathcal{A}(\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_b; r)) = b \end{matrix} \right] - \frac{1}{2} \right| \enspace .$$

Let Λ be an (n, κ)-nice tuple for $n = \mathrm{poly}(\kappa)$. A bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is *Λ-PSDL secure* [18], if for any non-uniform PPT adversary $\mathcal{A}$,

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

is negligible in κ. (Note that $\mathcal{A}$ also has access to $g_t^{x^0}$ since it belongs to gk.) A version of PSDL assumption in a non pairing-based group was defined in [10]. Lipmaa [18] proved that the Λ-PSDL assumption holds in the generic group model for any (n, κ)-nice tuple Λ given that $n = \mathrm{poly}(\kappa)$. More precisely, any successful generic adversary for Λ-PSDL requires time $\Omega(\sqrt{p/\lambda_n})$ where λ_n is the largest element of Λ. Thus, the choice of the actual security parameter depends on λ_n and thus also on Λ.

Let $\mathcal{G}_{\mathsf{bp}}$ be a bilinear group generator, and let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $R = \{(\mathsf{gk}; C, w)\}$ be an efficiently computable group-specific binary relation such that $|w| = \mathrm{poly}(|C|)$. Here, C is a statement, and w is a witness. Let $L = \{(\mathsf{gk}; C) : (\exists w)\, (\mathsf{gk}; C, w) \in R\}$ be a group-specific $\mathbf{NP}$-language. Shuffle (see Sect. 5) has a natural corresponding group-specific language, since one proves a relation between elements of the same group.

A *non-interactive argument* for R consists of the following PPT algorithms: a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$, a common reference string (CRS) generator $\mathcal{G}_{\mathsf{crs}}$, a prover $\mathcal{P}$, and a verifier $\mathcal{V}$. For $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$ and $\mathsf{crs} \leftarrow \mathcal{G}_{\mathsf{crs}}(\mathsf{gk})$, $\mathcal{P}(\mathsf{gk}, \mathsf{crs}; C, w)$ produces an argument π. The verifier $\mathcal{V}(\mathsf{gk}, \mathsf{crs}; C, \pi)$ outputs either 1 (accept) or 0 (reject). If the verifier only accesses a small part crs_v of crs, we say that crs_v is the verifier's part of the CRS and we will give just crs_v as an input to $\mathcal{V}$. When efficiency is not important (e.g., in the security definitions), we give the entire crs to $\mathcal{V}$.

An argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly complete*, if for all $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, all $\mathsf{crs} \leftarrow \mathcal{G}_{\mathsf{crs}}(\mathsf{gk})$ and all (C, w) such that $(\mathsf{gk}; C, w) \in R$, $\mathcal{V}(\mathsf{gk}, \mathsf{crs}; C, \mathcal{P}(\mathsf{gk}, \mathsf{crs}; C, w)) = 1$. An argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *adaptively computationally sound*, if for all non-uniform PPT adversaries $\mathcal{A}$, the probability

$$\Pr\left[\begin{array}{l}\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), \mathsf{crs} \leftarrow \mathcal{G}_{\mathsf{crs}}(\mathsf{gk}), (C, \pi) \leftarrow \mathcal{A}(\mathsf{gk}, \mathsf{crs}) : \\ (\mathsf{gk}; C) \notin L \wedge \mathcal{V}(\mathsf{gk}, \mathsf{crs}; C, \pi) = 1\end{array}\right]$$

is negligible in κ. The soundness is adaptive in the sense that the adversary sees the CRS before producing the statement C. An argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly witness-indistinguishable*, if for all $\mathsf{gk} \in \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $\mathsf{crs} \in \mathcal{G}_{\mathsf{crs}}(\mathsf{gk})$ and $((\mathsf{gk}; C, w_0), (\mathsf{gk}; C, w_1)) \in R^2$, the distributions $\mathcal{P}(\mathsf{gk}, \mathsf{crs}; C, w_0)$ and $\mathcal{P}(\mathsf{gk}, \mathsf{crs}; C, w_1)$ are equal. An argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly zero-knowledge*, if there exists a PPT simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, such that for all stateful interactive non-uniform PPT adversaries $\mathcal{A}$,

$$\Pr\left[\begin{array}{l}\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), \mathsf{crs} \leftarrow \mathcal{G}_{\mathsf{crs}}(\mathsf{gk}), \\ (C, w) \leftarrow \mathcal{A}(\mathsf{gk}, \mathsf{crs}), \\ \pi \leftarrow \mathcal{P}(\mathsf{gk}, \mathsf{crs}; C, w) : \\ (\mathsf{gk}; C, w) \in R \wedge \mathcal{A}(\pi) = 1\end{array}\right] = \Pr\left[\begin{array}{l}\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), (\mathsf{crs}, \mathsf{td}) \leftarrow \mathcal{S}_1(\mathsf{gk}), \\ (C, w) \leftarrow \mathcal{A}(\mathsf{gk}, \mathsf{crs}), \\ \pi \leftarrow \mathcal{S}_2(\mathsf{gk}, \mathsf{crs}, \mathsf{td}; C) : \\ (\mathsf{gk}; C, w) \in R \wedge \mathcal{A}(\pi) = 1\end{array}\right] .$$

Here, td is the *simulation trapdoor*.

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. We will use a weaker version of soundness in the subarguments, but in the case of the shuffle argument, we will prove soundness. Similarly to [12, 18], we will base the soundness of that argument on an explicit knowledge assumption.

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 tuple for some $n = \mathrm{poly}(\kappa)$. Consider $t \in \{1,2\}$. The bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is *Λ-PKE secure in group* $\mathbb{G}_t$ if for any non-uniform PPT adversary $\mathcal{A}$ there exists a non-uniform PPT extractor $X_{\mathcal{A}}$, such that

$$\Pr\left[\begin{array}{l}\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), (\alpha, x) \leftarrow \mathbb{Z}_p^2, \\ \mathsf{crs} \leftarrow (g_t^\alpha, (g_t^{x^\ell}, g_t^{\alpha x^\ell})_{\ell \in \Lambda}), (c, \hat{c}; (a_\ell)_{\ell \in \{0\} \cup \Lambda}) \leftarrow (\mathcal{A}||X_{\mathcal{A}})(\mathsf{gk}; \mathsf{crs}) : \\ \hat{c} = c^\alpha \wedge c \neq \displaystyle\prod_{\ell \in \{0\} \cup \Lambda} g_t^{a_\ell x^\ell}\end{array}\right]$$

is negligible in κ. Note that the element a_0 is output since g_t belongs to the CRS, and thus the adversary has access to $(g_t^{x^\ell}, g_t^{\alpha x^\ell})$ for $\ell \in \{0\} \cup \Lambda$. Groth [12] proved that the Λ-PKE assumption holds in the generic group model in the case $\Lambda = [n]$; his proof can be straightforwardly modified to the general case. We later need the special case where $\Lambda = \emptyset$, that is, the CRS contains only g_t^α, and the extractor returns a_0 such that $c = g_t^{a_0}$. This *KE assumption (in a bilinear group)* is similar to Damgård's KE assumption [6], except that it is made in a bilinear group setting.

A (tuple) commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ consists of two PPT algorithms: a randomized CRS generation algorithm $\mathcal{G}_{\mathsf{com}}$, and a randomized commitment algorithm $\mathcal{C}\mathrm{om}$. Here, $\mathcal{G}_{\mathsf{com}}^t(1^\kappa, n)$, $t \in \{1,2\}$, produces a CRS ck_t, and $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r)$, with $\boldsymbol{a} = (a_1, \dots, a_n)$, outputs a commitment value $A \in \mathbb{G}_t$. Within this paper, we open a commitment $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r)$ by publishing the values $\boldsymbol{a}$ and r.

A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ is *computationally binding in group* $\mathbb{G}_t$, if for every non-uniform PPT adversary $\mathcal{A}$ and positive integer $n = \mathrm{poly}(\kappa)$, the probability

$$\Pr\left[\begin{array}{l}\mathsf{ck}_t \leftarrow \mathcal{G}_{\mathsf{com}}^t(1^\kappa, n), (\boldsymbol{a_1}, r_1, \boldsymbol{a_2}, r_2) \leftarrow \mathcal{A}(\mathsf{ck}_t) : \\ (\boldsymbol{a_1}, r_1) \neq (\boldsymbol{a_2}, r_2) \wedge \mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a_1}; r_1) = \mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a_2}; r_2)\end{array}\right]$$

is negligible in κ. A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ is *perfectly hiding in group* $\mathbb{G}_t$, if for any positive integer $n = \mathrm{poly}(\kappa)$ and $\mathsf{ck}_t \in \mathcal{G}_{\mathsf{com}}^t(1^\kappa, n)$ and any two messages $\boldsymbol{a_1}$ and $\boldsymbol{a_2}$, the distributions $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a_1}; \cdot)$ and $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a_2}; \cdot)$ are equal. We use the following variant of the *knowledge commitment scheme* from [12] as modified by Lipmaa [18]:

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

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

Let $t = 1$. Fix a commitment key ck_1 that in particular specifies $g_2, \hat{g}_2 \in \mathbb{G}_2$. A commitment $(A, \hat{A}) \in \mathbb{G}_1^2$ is *valid*, if $e(A, \hat{g}_2) = e(\hat{A}, g_2)$. The case of $t = 2$ is dual.

As shown in [18], the knowledge commitment scheme in group $\mathbb{G}_t$ is perfectly hiding, and computationally binding under the Λ-PSDL assumption in group $\mathbb{G}_t$. If the Λ-PKE assumption holds in group $\mathbb{G}_t$, then for any non-uniform PPT algorithm $\mathcal{A}$, that outputs some valid knowledge commitments there exists a non-uniform PPT 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 these commitments.

A trapdoor commitment scheme has 3 additional efficient algorithms: (a) A trapdoor CRS generation algorithm inputs t, n and 1^κ and outputs a CRS ck^* (that has the same distribution as $\mathcal{G}^t_{\mathsf{com}}(1^\kappa, n)$) and a trapdoor td, (b) a randomized trapdoor commitment that takes ck^* and a randomizer r as inputs and outputs the value $\mathcal{C}\mathrm{om}^t(\mathsf{ck}^*; \mathbf{0}; r)$, and (c) a trapdoor opening algorithm that takes ck^*, td, $\boldsymbol{a}$ and r as an input and outputs an r', s.t. $\mathcal{C}\mathrm{om}^t(\mathsf{ck}^*; \mathbf{0}; r) = \mathcal{C}\mathrm{om}^t(\mathsf{ck}^*; \boldsymbol{a}; r')$. The knowledge commitment scheme is trapdoor, with the trapdoor being $\mathsf{td} = x$: after trapdoor-committing $A \leftarrow \mathcal{C}\mathrm{om}^t(\mathsf{ck}; \mathbf{0}; r) = g_t^r$ for $r \leftarrow \mathbb{Z}_p$, the committer can open it to $(\boldsymbol{a}; r - \sum_{i=1}^n a_i x^{\lambda_i})$ for any $\boldsymbol{a}$ [12, 18].

To avoid knowledge assumptions, one can relax the notion of soundness. Following [16] and [14], R_{co}-soundness is a weaker version of soundness, where it is required that an adversary who *knows* that $(\mathsf{gk}; C) \notin L$ should not be able to produce a witness w_{co} such that $(\mathsf{gk}; C, w_{co}) \in R_{co}$ (see [14] or [16] for a longer explanation). More formally, let $R = \{(\mathsf{gk}; C, w)\}$ and $L = \{(\mathsf{gk}; C) : (\exists w)(\mathsf{gk}; C, w) \in R\}$ be defined as earlier. Let $R_{co} = \{(\mathsf{gk}; C, w_{co})\}$ be an efficiently computable binary relation. An argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is (adaptively) *R_{co}-sound*, if for all non-uniform PPT adversaries $\mathcal{A}$, the following probability is negligible in κ:

$$\Pr\left[\begin{matrix}\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), \mathsf{crs} \leftarrow \mathcal{G}_{\mathsf{crs}}(\mathsf{gk}), (C, w_{co}, \pi) \leftarrow \mathcal{A}(\mathsf{gk}, \mathsf{crs}) : \\ (\mathsf{gk}; C, w_{co}) \in R_{co} \wedge \mathcal{V}(\mathsf{gk}, \mathsf{crs}; C, \pi) = 1\end{matrix}\right] .$$

In [12], Groth proposed efficient NIZK arguments that he proved to be sound under the power computational Diffie-Hellman assumption and the PKE assumption. Groth's arguments were later made more efficient by Lipmaa [18], who also showed that one can use somewhat weaker security assumptions (PSDL instead of PCDH). Groth [12] and Lipmaa [18] proposed two basic arguments (for Hadamard product and permutation). In both cases, Lipmaa showed that by using results about progression-free sets one can construct a set Λ_2 with $|\Lambda_2| = O(n 2^{2\sqrt{2 \log_2 n}}) = n^{1+o(1)}$. Together with a trivial Hadamard sum argument, one obtains a complete set of arguments that can be used to construct NIZK arguments for any NP language. (See [12, 18] for discussion.) However, this is always not the most efficient way to obtain a NIZK argument for a concrete language. In Sect. 3 we define new basic arguments that enable us to construct a very efficient permutation matrix argument and thus also a very efficient shuffle argument.

3 New Subarguments

In this section we present some subarguments that are required to construct the final shuffle argument. However, we expect them to have independent applications and thus we will handle each of them separately.

CRS generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $\mathring{\alpha} \leftarrow \mathbb{Z}_p$. Denote $\mathring{g}_t \leftarrow g_t^{\mathring{\alpha}}$ for $t \in \{1, 2\}$. The CRS is $\mathsf{crs} \leftarrow (\mathring{g}_1, \mathring{g}_2)$. The commitment key is $\mathsf{ck}_2 \leftarrow (\mathsf{gk}; \mathring{g}_2)$, and the verifier's part of the CRS is $\mathsf{crs}_v \leftarrow \mathring{g}_1$.
Common input: $A_2 \leftarrow g_2^r \in \mathbb{G}_2$.
Argument generation $\mathcal{P}_0(\mathsf{gk}, \mathsf{crs}; A_2, r)$**:** The prover defines $\mathring{A}_2 \leftarrow \mathring{g}_2^r$, and sends $\pi \leftarrow \mathring{A}_2 \in \mathbb{G}_2$ to $\mathcal{V}$ as the argument.
Verification $\mathcal{V}_0(\mathsf{gk}, \mathsf{crs}_v; A_2, \pi = \mathring{A}_2)$**:** The verifier accepts if $\hat{e}(\mathring{g}_1, A_2) = \hat{e}(g_1, \mathring{A}_2)$.

Protocol 1: New zero argument in group $\mathbb{G}_2$

3.1 New Zero Argument

In a zero argument, the prover aims to convince the verifier that he knows how to open knowledge commitment $A_t \in \mathbb{G}_t$ to the all-zero message tuple $\mathbf{0} = (0, \dots, 0)$. Alternatively, one aims to prove the knowledge of the discrete logarithm of A_t, that is, that $A_t = g_t^r$ for some r. By using the homomorphic properties of the knowledge commitment scheme, the prover can use the zero argument to show that A_t can be opened to an arbitrary constant.

This argument can be derived from [12, 18]. Intuitively, we set (only for this argument) $n = 0$ and show that $A = A_2$ is a commitment to a length-0 tuple. For this, we only have to include to the CRS the elements $\mathring{g}_1$ and $\mathring{g}_2$. (The case $t = 1$ can be handled dually.) The following theorem is basically a tautology, since the KE assumption states that the prover knows r. However, since any $(A_2, \mathring{A}_2)$, where $\mathring{A}_2 = A_2^{\mathring{\alpha}}$, is a commitment of $\mathbf{0}$ (and thus, $(\mathsf{gk}; A_2) \in L$) for *some* r, we cannot claim that Prot. 1 is computationally sound (even under a knowledge assumption). Instead, analogously to [12, 18], we prove a weaker version of soundness (which is however sufficient to achieve soundness of the shuffle argument). Note that the last statement of the theorem basically says that no efficient adversary can output an input to the product argument together with an accepting argument and openings to all commitments and all other pairs of type $(y, \bar{y})$ that are present in the argument, such that $a_i b_i \neq c_i$ for some i.

Theorem 1. *The non-interactive zero argument in Prot. 1 is perfectly complete, perfectly zero-knowledge. Any non-uniform probabilistic-polynomial time adversary has a negligible chance of returning an input $inp^0 = A_2$ and a satisfying argument $\pi^0 = \mathring{A}_2$ together with a opening witness $w^0 = (\boldsymbol{a}, r)$, such that $(A_2, \mathring{A}_2) = \mathcal{C}om^2(\mathsf{ck}_2; \boldsymbol{a}; r)$, $\boldsymbol{a} \neq 0$ but the verification $\mathcal{V}_0(\mathsf{gk}, \mathsf{crs}; A_2, \mathring{A}_2)$ accepts.*

Proof. PERFECT COMPLETENESS is straightforward, since $\hat{e}(\mathring{g}_1, A_2) = \hat{e}(g_1^{\mathring{\alpha}}, A_2) = \hat{e}(g_1, A_2^{\mathring{\alpha}}) = \hat{e}(g_1, \mathring{A}_2)$. PERFECT ZERO-KNOWLEDGE: we construct the following simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$. The simulator $\mathcal{S}_1$ generates first $\mathsf{td} = \mathring{\alpha} \leftarrow \mathbb{Z}_p$, and then $\mathsf{crs} \leftarrow \left(\mathring{g}_1 \leftarrow g_1^{\mathring{\alpha}}, \mathring{g}_2 \leftarrow g_2^{\mathring{\alpha}}\right)$, and saves td. Since the simulator $\mathcal{S}_2$ later knows $\mathring{\alpha}$, it can compute a satisfying argument $\mathring{A}_2$ as $\mathring{A}_2 \leftarrow A_2^{\mathring{\alpha}}$. Clearly, $\mathring{A}_2$ has the same distribution as in the real argument.

WEAKER VERSION OF SOUNDNESS: assume that there exists an adversary $\mathcal{A}$ that can break the last statement of the theorem. That is, $\mathcal{A}$ can create $(A_2, (\boldsymbol{a}, r), \mathring{A}_2)$ such that $(A_2, \mathring{A}_2) = \mathcal{C}om^2(\boldsymbol{a}; r)$, $\boldsymbol{a} \neq \mathbf{0}$, and $\hat{e}(\mathring{g}_1, A_2) = \hat{e}(g_1, \mathring{A}_2)$. But then $(A_2, \mathring{A}_2) = (g_2^r \cdot \prod_{i=1}^n g_2^{a_i x^{\lambda_i}}, \mathring{g}_2^r \cdot \prod_{i=1}^n \mathring{g}_2^{a_i x^{\lambda_i}})$ with $\lambda_I \neq 0$ for some $I \in [n]$. Since

$(\mathsf{gk}, \mathsf{crs})$ contains $\mathring{g}_2^{x^\ell}$ only for $\ell \in \{0\}$, the adversary has thus broken the $\emptyset$-PSDL assumption. But the $\emptyset$-PSDL assumption is straightforwardly true, since then the input of the adversary does not depend on x at all. Thus, the argument in Prot. 1 satisfies the last statement of the theorem. $\square$

The fact that the weaker version of soundness of this argument does not require any (non-trivial) assumption is, while somewhat surprising, also a logical consequence of CRS including $\mathring{g}_2^{x^\ell}$ only for $\ell \neq 0$. In fact, if the CRS contained $\mathring{g}_2^{x_\ell}$ for some other value of ℓ then the argument would not be sound under any (reasonable) computational assumption. The proof of the following lemma is straightforward.

Lemma 1. *The CRS length in Prot. 1 is* 1 *element from the group* $\mathbb{G}_1$ *and* 1 *element from the group* $\mathbb{G}_2$*. The argument size in Prot. 1 is* 1 *element from the group* $\mathbb{G}_2$*. Prover's computational complexity is dominated by* 1 *exponentiation. The verifier's computational complexity is dominated by* 2 *bilinear pairings.*

3.2 New 1-Sparsity Argument

Assume that $A_2 \in \mathbb{G}_2$. A vector $\boldsymbol{a} \in \mathbb{Z}_p^n$ is *k-sparse*, if it has at most k non-zero coefficients. In a 1-*sparsity argument* in $\mathbb{G}_2$, the prover aims to convince the verifier that he knows an opening $A_2 = g_2^r \cdot \prod_{i=1}^n g_{2,\lambda_i}^{a_i}$ such that $\boldsymbol{a}$ is 1-sparse, that is, there exists $I \in [n]$ such that for $i \neq I$, $a_i = 0$, while a_I can take any value, including 0. Alternatively, since $\mathbb{Z}_p$ has no zero divisors, this means that the prover aims to convince the verifier that $a_i a_j = 0$ for every $i, j \in [n]$ such that $i \neq j$. (Note that the zero argument can seen as a 0-sparsity argument.) A new 1-sparsity argument is depicted by Prot. 2; 1-sparsity argument in $\mathbb{G}_1$ is defined dually.

Intuitively, the new 1-sparsity argument is constructed by following the same main ideas as the basic arguments (for Hadamard product and permutation) from [18]. That is, we start with a verification equation $\hat{e}(A_1, A_2) = \hat{e}(g_1, F)$, where the discrete logarithm of the left-hand side, see Eq. (1), is a sum of two polynomials $F_{con}(x)$ and $F_\pi(x)$, where x is the secret key. In this case, $F_{con}(x)$ has $n(n-1)$ monomials (with coefficients $a_i a_j$ with $i \neq j$) that all vanish exactly if the prover is honest. On the other hand, the polynomial $F_\pi(x)$ has only $2n+1$ monomials. Therefore, a honest prover can compute the argument given $2n+1$ pairs $(g_{2\ell}, \bar{g}_{2\ell})$. Moreover, the prover can construct F by using 10 exponentiations. For comparison, in the basic arguments (the Hadamard product argument and the permutation argument) of [18], the polynomial $F_{con}(x)$ had n monomials, and the polynomial $F_\pi(x)$ had $O(n 2^{2\sqrt{2 \log_2 n}}) = n^{1+o(1)}$ monomials. Thus, the CRS had $O(n 2^{2\sqrt{2 \log_2 n}}) = n^{1+o(1)}$ group elements and the prover's computational complexity was dominated by $O(n 2^{2\sqrt{2 \log_2 n}}) = n^{1+o(1)}$ exponentiations.

Similarly to the zero argument, we cannot prove the computational soundness of this argument, since for every $\boldsymbol{a}$, there exists r such that $A_2 = g_2^r \prod_{i \in [n]} g_2^{a_i x^{\lambda_i}}$. Instead, following [12, 18], we prove a weaker version of knowledge. Intuitively, the theorem statement includes f'_ℓ only for $\ell \in \bar{\Lambda}$ (resp., a_ℓ for $\ell \in \Lambda$ together with r) since $\bar{g}_{2\ell}$ (resp., $\bar{g}_{1\ell}$) belongs to the CRS only for $\ell \in \bar{\Lambda}$ (resp., $\ell \in \{0\} \cup \Lambda$).

Theorem 2. *The* 1*-sparsity argument in Prot. 2 is perfectly complete and perfectly witness-indistinguishable. Let* $\varLambda$ *be a progression-free set of odd positive integers. If the* $\mathcal{G}_{\mathsf{bp}}$ *is* $\bar{\varLambda}$*-PSDL secure, then any non-uniform PPT adversary has negligible chance of outputting* $inp^{spa} \leftarrow (A_2, \bar{A}_2)$ *and a satisfying argument* $\pi^{spa} \leftarrow (A_1, \bar{A}_1, F, \bar{F})$ *together with an opening witness* $w^{spa} \leftarrow ((a_\ell)_{\ell \in \varLambda}, r, (f'_\ell)_{\ell \in \bar{\varLambda}})$*, such that* $(A_2, \bar{A}_2) = \mathcal{C}om^2(\mathsf{ck}_2; \boldsymbol{a}; r)$*,* $(F, \bar{F}) = (g_2^{\sum_{\ell \in \bar{\varLambda}} f'_\ell x_\ell}, \bar{g}_2^{\sum_{\ell \in \bar{\varLambda}} f'_\ell x_\ell})$*, for some* $i \neq j \in [n]$*,* $a_i a_j \neq 0$*, and the verification* $\mathcal{V}_{spa}(\mathsf{gk}, \mathsf{crs}; (A_2, \bar{A}_2), \pi^{spa})$ *accepts.*

The (weak) soundness reduction is tight, except that it requires to factor a polynomial of degree $2\lambda_n = \max\{i \in \bar{\varLambda}\}$.

Proof. Let $\eta \leftarrow \hat{e}(A_1, A_2)$ and $h \leftarrow \hat{e}(g_1, g_2)$. PERFECT WITNESS-INDISTINGUISHABILITY: since satisfying argument π^{spa} is uniquely determined, all witnesses result in the same argument, and thus this argument is witness-indistinguishable.

PERFECT COMPLETENESS. All verifications but the last one are straightforward. For the last verification $\hat{e}(A_1, A_2) = \hat{e}(g_1, F)$, note that $\log_h \eta = (r + \sum_{i=1}^n a_i x^{\lambda_i})(r + \sum_{j=1}^n a_j x^{\lambda_j}) = F_{con}(x) + F_\pi(x)$, where

$$F_{con}(x) = \underbrace{\sum_{i=1}^{n} \sum_{j=1: j \neq i}^{n} a_i a_j x^{\lambda_i + \lambda_j}}_{\delta \in 2 \hat{\cdot} \varLambda} \text{ and } F_\pi(x) = \underbrace{r^2 + 2r \sum_{i=1}^{n} a_i x^{\lambda_i} + \sum_{i=1}^{n} a_i^2 x^{2\lambda_i}}_{\delta \in \bar{\varLambda}} \quad . \tag{1}$$

Thus, $\log_h \eta$ is equal to a sum of x^δ for $\delta \in 2 \hat{\cdot} \varLambda$ and $\delta \in \bar{\varLambda}$. If the prover is honest, then $a_i a_j = 0$ for $i \neq j$, and thus $\log_h \eta$ is a formal polynomial that has non-zero monomials γx^δ with only $\delta \in \bar{\varLambda}$. Since then $a_i = 0$ for $i \neq I$, we have $\log_h \eta = r^2 + 2r a_I x^{\lambda_I} + a_I^2 x^{2\lambda_I} = \log_{g_2} F$. Thus, if the prover is honest, then the third verification succeeds.

WEAKER VERSION OF SOUNDNESS: Assume that $\mathcal{A}$ is an adversary that can break the last statement of the theorem. Next, we construct an adversary $\mathcal{A}'$ against the $\bar{\varLambda}$-PSDL assumption. Let $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$ and $x \leftarrow \mathbb{Z}_p$. The adversary $\mathcal{A}'$ receives $\mathsf{crs} \leftarrow (\mathsf{gk}; (g_1^{x^\ell}, g_2^{x^\ell})_{\ell \in \bar{\varLambda}})$ as her input, and her task is to output x. She sets $\bar{\alpha} \leftarrow \mathbb{Z}_p$, $\mathsf{crs}' \leftarrow (\bar{g}_1, \bar{g}_2, (g_1^{x^\ell}, g_1^{\bar{\alpha} x^\ell})_{\ell \in \varLambda}, (g_2^{x^\ell}, g_2^{\bar{\alpha} x^\ell})_{\ell \in \varLambda \cup (2 \cdot \varLambda)})$, and then forwards crs' to $\mathcal{A}$. Clearly, crs' follows the distribution imposed by $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$. Denote $\mathsf{ck}_2 \leftarrow (\mathsf{gk}; \bar{g}_2, (g_2^{x^\ell}, g_2^{\bar{\alpha} x^\ell})_{\ell \in \varLambda})$. According to the last statement of the theorem, $\mathcal{A}(\mathsf{gk}; \mathsf{crs}')$ returns $((A_2, \bar{A}_2), w^{spa} = ((a_\ell)_{\ell \in \varLambda}, r, (f'_\ell)_{\ell \in \bar{\varLambda}}), \pi^{spa} = (A_1, \bar{A}_1, F, \bar{F}))$.

Assume that $\mathcal{A}$ was successful, that is, for some $i, j \in [n]$ and $i \neq j$, $a_i a_j \neq 0$. Since $(A_2, \bar{A}_2) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{a}; r)$ and $\mathcal{V}_{spa}(\mathsf{gk}, \mathsf{crs}'; (A_2, \bar{A}_2), \pi^{spa}) = 1$, $\mathcal{A}'$ has expressed $\log_h \eta = \log_{g_2} F$ as a polynomial $f(x)$, where at least for some $\ell \in 2 \hat{\cdot} \varLambda$, x^ℓ has a non-zero coefficient.

On the other hand, $\log_{g_2} F = \sum_{\ell \in \bar{\varLambda}} f'_\ell x^\ell = f'(x)$. Since $\varLambda$ is a progression-free set of odd positive integers, then $2 \hat{\cdot} \varLambda \cap \bar{\varLambda} = \emptyset$ and thus if $\ell \in \bar{\varLambda}$ then $\ell \notin 2 \hat{\cdot} \varLambda$. Therefore, all coefficients of $f'(x)$ corresponding to any x^ℓ, $\ell \in 2 \hat{\cdot} \varLambda$, are equal to 0. Thus $f(X) = \sum f_\ell X^\ell$ and $f'(X) = \sum_{\ell \in \bar{\varLambda}} f'_\ell X^\ell$ are different polynomials with

$$f(x) = f'(x) = \log_{g_2} F \quad .$$

System parameters: Let $n = \mathrm{poly}(\kappa)$. Let $\Lambda = \{\lambda_i : i \in [n]\}$ be an (n, κ)-nice progression-free set of odd positive integers. Denote $\lambda_0 := 0$. Let $\bar{\Lambda} = \{0\} \cup \Lambda \cup (2 \cdot \Lambda)$.

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

Common input: $(A_2, \bar{A}_2) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{a}; r) = (g_2^r \cdot g_{2,\lambda_I}^{a_I}, \bar{g}_2^r \cdot \bar{g}_{2,\lambda_I}^{a_I}) \in \mathbb{G}_2^2$, with $I \in [n]$.

Argument generation $\mathcal{P}_{spa}(\mathsf{gk}, \mathsf{crs}; (A_2, \bar{A}_2), (\boldsymbol{a}, r))$**:** The prover defines $A_1 \leftarrow g_1^r \cdot g_{1,\lambda_I}^{a_I}$, $\bar{A}_1 \leftarrow \bar{g}_1^r \cdot \bar{g}_{1,\lambda_I}^{a_I}$, $F \leftarrow g_2^{r^2} \cdot g_{2,\lambda_I}^{2ra_I} \cdot g_{2,2\lambda_I}^{a_I^2}$, and $\bar{F} \leftarrow \bar{g}_2^{r^2} \cdot \bar{g}_{2,\lambda_I}^{2ra_I} \cdot \bar{g}_{2,2\lambda_I}^{a_I^2}$. The prover sends $\pi^{spa} \leftarrow (A_1, \bar{A}_1, F, \bar{F}) \in \mathbb{G}_1^2 \times \mathbb{G}_2^2$ to the verifier as the argument.

Verification $\mathcal{V}_{spa}(\mathsf{gk}, \mathsf{crs}_v; (A_2, \bar{A}_2), \pi^{spa})$**:** $\mathcal{V}_{spa}$ accepts iff $\hat{e}(A_1, g_2) = \hat{e}(g_1, A_2)$, $\hat{e}(\bar{A}_1, g_2) = \hat{e}(A_1, \bar{g}_2)$, $\hat{e}(g_1, \bar{A}_2) = \hat{e}(\bar{g}_1, A_2)$, $\hat{e}(g_1, \bar{F}) = \hat{e}(\bar{g}_1, F)$, and $\hat{e}(A_1, A_2) = \hat{e}(g_1, F)$.

Protocol 2: New 1-sparsity argument

Therefore, $\mathcal{A}'$ has succeeded in creating a non-zero polynomial $d = f - f'$, such that $d(x) = \sum_{\ell \in \bar{\Lambda}} d_\ell x^\ell = 0$.

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

The 1-sparsity argument is not perfectly zero-knowledge. The problem is that the simulator knows $\mathsf{td} = (\bar{\alpha}, x)$, but given td and $(A_2, \bar{A}_2)$ she will not be able to generate π^{spa}. E.g., she has to compute $A_1 = g_1^r \cdot g_1^{a_I x^{\lambda_I}}$ based on $A_2 = g_2^r \cdot g_2^{a_I x^{\lambda_I}}$ and x, but without knowing r, I or a_I. This seems to be impossible without knowing an efficient isomorphism $\mathbb{G}_1 \to \mathbb{G}_2$. Computing F and $\bar{F}$ is even more difficult, since in this case the simulator does not even know the corresponding elements in $\mathbb{G}_1$. Technically, the problem is that due to the knowledge of the trapdoor, the simulator can, knowing one opening $(\boldsymbol{a}, r)$, produce an opening $(\boldsymbol{a'}, r')$ to any other $\boldsymbol{a'}$. However, here she does not know any openings. For the same reason, the permutation matrix argument of Sect. 3.3 will not be zero-knowledge. On the other hand, in the final shuffle argument of Sect. 5, the simulator creates all commitments by herself and can thus properly simulate the argument. By the same reason, the subarguments of [12, 18] are not zero-knowledge but their final argument (for circuit satisfiability) is.

Theorem 3. *Consider Prot. 2. The CRS consists of* $2n + 1$ *elements of* $\mathbb{G}_1$ *and* $4n + 1$ *elements of* $\mathbb{G}_2$*, with the verifier's part of the CRS consisting of only* 1 *element of* $\mathbb{G}_1$ *and* 1 *element of* $\mathbb{G}_2$*. The communication complexity (argument size) of the argument in Prot. 2 is* 2 *elements from* $\mathbb{G}_1$ *and* 2 *elements from* $\mathbb{G}_2$*. Prover's computational complexity is dominated by* 10 *exponentiations. Verifier's computational complexity is dominated by* 10 *bilinear pairings.*

3.3 New Permutation Matrix Argument

In this section, we will design a new *permutation matrix argument* where the prover aims to convince the verifier that he knows a permutation matrix P such that $(c_{2i}, \bar{c}_{2i}) \in$

Setup: let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$.
Common reference string $\mathcal{G}_{\mathsf{crs}}(\mathsf{gk})$**:** Let $\bar{\alpha}, \mathring{\alpha}, x \leftarrow \mathbb{Z}_p$, $\bar{g}_t \leftarrow g_t^{\bar{\alpha}}$, $\mathring{g}_t \leftarrow g_t^{\mathring{\alpha}}$, $g_{t\ell} \leftarrow g_t^{x^\ell}$, and $\bar{g}_{t\ell} \leftarrow \bar{g}_t^{x^\ell}$. Let $D \leftarrow \prod_{i=1}^n g_{2,\lambda_i}$. Let $\mathsf{crs} \leftarrow (\bar{g}_1, \bar{g}_2, \mathring{g}_1, \mathring{g}_2, (g_{1\ell}, \bar{g}_{1\ell})_{\ell \in \Lambda}, (g_{2\ell}, \bar{g}_{2\ell})_{\ell \in \Lambda \cup (2\cdot\Lambda)}, D)$, $\mathsf{ck}_2 = (\mathsf{gk}; \bar{g}_2, (g_{2\ell}, \bar{g}_{2\ell})_{\ell \in \Lambda})$, $\mathring{\mathsf{ck}}_2 = (\mathsf{gk}; g_2, \mathring{g}_2)$, and $\mathsf{crs}_v = (\bar{g}_1, \bar{g}_2, \mathring{g}_1)$.
Common input: $(c_{2i}, \bar{c}_{2i}) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{P_i}; r_i) = (g_2^{r_i} \cdot g_{2,\lambda_{\psi(i)}}, \bar{g}_2^{r_i} \cdot \bar{g}_{2,\lambda_{\psi(i)}})$ for $i \in [n]$.
Argument Generation $\mathcal{P}_{pm}(\mathsf{gk}, \mathsf{crs}; (\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}}), (P, \boldsymbol{r}))$**:** Construct a zero argument $\pi^0 \leftarrow \mathring{g}_2^{\sum_{i=1}^n r_i}$ that $(\prod_{i=1}^n c_{2i})/D$ commits to $\mathbf{0}$. For $i \in [n]$, construct a 1-sparsity argument $\pi_i^{spa} = (c_{1i}, \bar{c}_{1i}, F_i, \bar{F}_i)$ that $(c_{2i}, \bar{c}_{2i})$ commits to a 1-sparse row. Send $\pi^{pm} \leftarrow (\pi^0, \boldsymbol{\pi^{spa}})$ to the verifier.
Verification $\mathcal{V}_{pm}(\mathsf{gk}, \mathsf{crs}_v; (\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}}); \pi^{pm})$**:** The verifier checks $n+1$ arguments $(\pi^0, \boldsymbol{\pi^{spa}})$.

Protocol 3: New permutation matrix argument in group $\mathbb{G}_2$ with $P = P_\psi$

$\mathbb{G}_2^2$ are knowledge commitments to P's rows. Recall that a permutation matrix is a Boolean matrix with exactly one 1 in every row and column: if ψ is a permutation then the corresponding permutation matrix P_ψ is such that $(P_\psi)_{ij} = 1$ iff $j = \psi(i)$. Thus $(P_{\psi^{-1}})_{ij} = 1$ iff $i = \psi(j)$. We base our argument on the following lemma.

Lemma 2. *An $n \times n$ matrix P is a permutation matrix if and only if the following two conditions hold: (a) the sum of elements in any single column is equal to* 1*, and (b) no row has more than* 1 *non-zero elements.*

Proof. First, assume that P is a permutation matrix. Then every column has exactly one non-zero element (namely, with value 1), and thus both claims hold. Second, assume that (a) and (b) are true. Due to (a), every column must have at least one non-zero element, and thus the matrix has at least n non-zero elements. Due to (b), no row has more than 1 non-zero elements, and thus the matrix has at most n non-zero elements. Thus the matrix has exactly n non-zero elements, one in each column. Due to (a), all non-zero elements are equal to 1, and thus P is a permutation matrix. □

We now use the 1-sparsity argument and the zero argument to show that the committed matrix satisfies the claims of Lem. 2. Therefore, by Lem. 2, P is a permutation matrix. Following [12, 18] and similarly to the case of the zero and 1-sparsity arguments, we prove that the permutation argument satisfies a "weaker" version of soundness.

Theorem 4. *The argument in Prot. 3 is a perfectly complete and perfectly witness-indistinguishable permutation matrix argument. Let Λ be a progression-free set of odd positive integers. If the $\bar{\Lambda}$-PSDL assumption holds, then any non-uniform PPT adversary has a negligible chance in outputting an input $inp^{pm} \leftarrow (\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}})$ and a satisfying argument $\pi^{pm} \leftarrow \left(\pi^0, (c_{1i}, \bar{c}_{1i}, F_i, \bar{F}_i)_{i \in [n]}\right)$ together with an opening witness $w^{pm} \leftarrow \left((a_i)_{i \in \Lambda}, r_a, (\boldsymbol{P_i}, r_i, (f'_{ij})_{j \in \bar{\Lambda}})_{i \in [n]}\right)$, such that $\left(\left(\prod_{i=1}^n c_{2i}\right)/D, \pi^0\right) = \mathcal{C}om^2(\mathsf{ck}_2; \boldsymbol{a}; r_a)$, $(\forall i \in [n])(c_{2i}, \bar{c}_{2i}) = \mathcal{C}om^2(\mathsf{ck}_2; \boldsymbol{P_i}; r_i)$, $(\forall i \in [n]) \log_{g_2} F_i = \sum_{j \in \bar{\Lambda}} f'_{ij} x^j$, $(\boldsymbol{a} \neq \mathbf{0} \vee (\exists i \in [n])\boldsymbol{P_i}$ is not 1-sparse), and the verification $\mathcal{V}_{pm}(\mathsf{gk}, \mathsf{crs}; (\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}}), \pi^{pm})$ accepts.*

Proof. PERFECT COMPLETENESS: follows from the completeness of the 1-sparsity and zero arguments and from Lem. 2, if we note that $\prod_{i=1}^{n} c_{2i}/D = g_2^{\sum_{i=1}^{n} r_i}$, and thus $(\prod_{i=1}^{n} c_{2i}/D, \pi^0)$ commits to $\mathbf{0}$ iff every column of P sums to 1.

WEAKER VERSION OF SOUNDNESS: Let $\mathcal{A}$ be a non-uniform PPT adversary that creates $(\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}})$, an opening witness $((a_\ell)_{\ell\in\Lambda}, r_a, (P_i, r_i, (f'_{ij})_{j\in\bar{\Lambda}})_{i\in[n]})$, and an accepting NIZK argument π^{spa}.

Since the zero argument is (weakly) sound, verification of the argument π^0 shows that every column of P sums to 1. Here the witness is $w^0 = (\boldsymbol{a}, r_a)$ with $\boldsymbol{a} = \sum_{i=1}^{n} \boldsymbol{P_i} - \mathbf{1}$. By the $\bar{\Lambda}$-PSDL assumption, the 1-sparsity assumption is (weakly) sound. Therefore, verification of the arguments $\boldsymbol{\pi^{spa}}$ shows that every row of P has exactly one 1 (here the witness is $w_i^{spa} = (P_i, r_i, (f'_{ij})_{j\in\bar{\Lambda}})$). Therefore, by Lem. 2 and by the (weak) soundness of the 1-sparsity and zero arguments, P is a permutation matrix.

PERFECT WITNESS-INDISTINGUISHABILITY: since satisfying argument π^{pm} is uniquely determined, all witnesses result in the same argument, and therefore the permutation matrix argument is witness-indistinguishable. $\square$

Lemma 3. *Consider Prot. 3. The CRS consists of* $2n+2$ *elements of* $\mathbb{G}_1$ *and* $5n+4$ *elements of* $\mathbb{G}_2$*. The verifier's part of the CRS consists of* 2 *elements of* $\mathbb{G}_1$ *and of* 2 *elements of* $\mathbb{G}_2$*. The communication complexity is* $2n$ *elements of* $\mathbb{G}_1$ *and* $2n+1$ *elements of* $\mathbb{G}_2$*. The prover's computational complexity is dominated by* $10n+1$ *exponentiations. The verifier's computational complexity is dominated by* $10n+2$ *pairings.*

4 Knowledge BBS Cryptosystem

Boneh, Boyen and Shacham [3] proposed the BBS cryptosystem $\Pi = (\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{D}\mathsf{ec})$. We will use a (publicly verifiable) "knowledge" version of this cryptosystem so that according to the KE (that is, the $\emptyset$-PKE) assumption, the party who produces a valid ciphertext must know both the plaintext and the randomizer. We give a definition for group $\mathbb{G}_1$, the knowledge BBS cryptosystem for group $\mathbb{G}_2$ can be defined dually.

Setup (1^κ)**:** Let $\mathsf{gk} \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$.

Key Generation $\mathcal{G}_{\mathsf{pkc}}(\mathsf{gk})$**:** Set $(\tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3) \leftarrow \mathbb{Z}_p^3$, $\tilde{g}_1 \leftarrow g_1^{\tilde{\alpha}_3}$, $\tilde{g}_2^{(1)} \leftarrow g_2^{\tilde{\alpha}_1}$, $\tilde{g}_2^{(2)} \leftarrow g_2^{\tilde{\alpha}_2}$, $\tilde{g}_2^{(3)} \leftarrow g_2^{\tilde{\alpha}_3}$. The secret key is $\mathsf{sk} := (\mathsf{sk}_1, \mathsf{sk}_2) \leftarrow (\mathbb{Z}_p^*)^2$, and the public key is $\mathsf{pk} \leftarrow (\mathsf{gk}; \tilde{g}_1, \tilde{g}_2^{(1)}, \tilde{g}_2^{(2)}, \tilde{g}_2^{(3)}, f, \tilde{f}, h, \tilde{h})$, where $f = g_1^{1/\mathsf{sk}_1}$, $\tilde{f} = f^{\tilde{\alpha}_1}$, $h = g_1^{1/\mathsf{sk}_2}$, and $\tilde{h} = h^{\tilde{\alpha}_2}$.

Encryption $\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu; \sigma, \tau)$**:** To encrypt a message $\mu \in \mathbb{Z}_p$ with randomizer $(\sigma, \tau) \in \mathbb{Z}_p^2$, output the ciphertext $\mathfrak{u} = (\mathfrak{u}_1, \mathfrak{u}_2, \mathfrak{u}_3, \tilde{\mathfrak{u}}_1, \tilde{\mathfrak{u}}_2, \tilde{\mathfrak{u}}_3)$, where $\mathfrak{u}_1 = f^\sigma$, $\mathfrak{u}_2 = h^\tau$, $\mathfrak{u}_3 = g_1^{\mu+\sigma+\tau}$, $\tilde{\mathfrak{u}}_1 = \tilde{f}^\sigma$, and $\tilde{\mathfrak{u}}_2 = \tilde{h}^\tau$, and $\tilde{\mathfrak{u}}_3 = \tilde{g}_1^{\mu+\sigma+\tau}$.

Decryption $\mathcal{D}\mathsf{ec}_{\mathsf{sk}}(\mathfrak{u}_1, \mathfrak{u}_2, \mathfrak{u}_3, \tilde{\mathfrak{u}}_1, \tilde{\mathfrak{u}}_2, \tilde{\mathfrak{u}}_3)$**:** if $\hat{e}(\mathfrak{u}_1, \tilde{g}_2^{(1)}) = \hat{e}(\tilde{\mathfrak{u}}_1, g_2)$, $\hat{e}(\mathfrak{u}_2, \tilde{g}_2^{(2)}) = \hat{e}(\tilde{\mathfrak{u}}_2, g_2)$ and $\hat{e}(\mathfrak{u}_3, \tilde{g}_2^{(3)}) = \hat{e}(\tilde{\mathfrak{u}}_3, g_2)$, then return the discrete logarithm of $g_1^\mu \leftarrow \mathfrak{u}_3/(\mathfrak{u}_1^{\mathsf{sk}_1}\mathfrak{u}_2^{\mathsf{sk}_2})$. Otherwise, return $\perp$.

Since $\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_1; \sigma_1, \tau_1) \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_2; \sigma_2, \tau_2) = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_1 + \mu_2; \sigma_1 + \sigma_2, \tau_1 + \tau_2)$, the knowledge BBS cryptosystem is additively homomorphic (with respect to element-wise multiplication of the ciphertexts). In particular, one can re-encrypt (that is, blind) a ciphertext efficiently: if σ_2 and τ_2 are random, then $\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu; \sigma_1, \tau_1) \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(0; \sigma_2, \tau_2) = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu; \sigma_1 + \sigma_2, \tau_1 + \tau_2)$ is a random encryption of μ, independently of σ_1 and τ_1.

The cryptosystem has to be lifted (i.e., the value μ be in exponent) for the soundness proof of the new shuffle argument in Sect. 5 to go through; see there for a discussion. Thus, to decrypt, one has to compute discrete logarithms. Since this the latter is intractable, in real applications one has to assume that μ is small. Consider for example the e-voting scenario where μ is the number of the candidate (usually a small number).

One can now use one of the following approaches. First, discard the ballots if the ciphertext does not decrypt. (This can be checked publicly.) Second, use a (non-interactive) range proof [20, 4] (in the e-voting scenario, range proofs are only given by the voters and not by the voting servers, and thus the range proof can be relatively less efficient compared to the shuffle argument) to guarantee that the ballots are correctly formed. In this case, invalid ballots can be removed from the system before starting to shuffle (saving thus valuable time otherwise wasted to shuffle invalid ciphertexts). Both approaches have their benefits, and either one can be used depending on the application.

The inclusion of $\tilde{\mathfrak{u}}_3$ to the ciphertext is required because of our proof technique. Without it, the extractor in the proof of of the soundness of the new shuffle argument can extract μ only if μ is small. Thus, security would not be guaranteed against an adversary who chooses $\mathfrak{u}_3$ without actually knowing the element μ.

It is easy to see that the knowledge BBS cryptosystem, like the original BBS cryptosystem, is CPA-secure under the DLIN assumption (see Sect. A for the definition of the latter).

5 New Shuffle Argument

Let $\Pi = (\mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{D}\mathsf{ec})$ be an additively homomorphic cryptosystem. Assume that $\mathfrak{u}_i$ and $\mathfrak{u}'_i$ are valid ciphertexts of Π. We say that $(\mathfrak{u}'_1, \dots, \mathfrak{u}'_n)$ is a *shuffle* of $(\mathfrak{u}_1, \dots, \mathfrak{u}_n)$ iff there exists a permutation $\psi \in S_n$ and randomizers $r_1, \dots, r_n$ such that $\mathfrak{u}'_i = \mathfrak{u}_{\psi(i)} \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(0; r_i)$ for $i \in [n]$. (In the case of the knowledge BBS cryptosystem, $r_i = (\sigma_i, \tau_i)$.) In a shuffle argument, the prover aims to convince the verifier in zero-knowledge that given $(\mathsf{pk}, (\mathfrak{u}_i, \mathfrak{u}'_i)_{i \in [n]})$, he knows a permutation $\psi \in S_n$ and randomizers r_i such that $\mathfrak{u}'_i = \mathfrak{u}_{\psi(i)} \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(0; r_i)$ for $i \in [n]$. More precisely, we define the group-specific binary relation R^{sh} exactly as in [14]: $R^{sh} := \{((p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g_1, g_2), (\mathsf{pk}, \{\mathfrak{u}_i\}, \{\mathfrak{u}'_i\}), (\psi, \{r_i\})) : \psi \in S_n \wedge \left(\forall i : \mathfrak{u}'_i = \mathfrak{u}_{\psi(i)} \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(0; r_i)\right)\}$ Note that both according to the corresponding computational soundness definition and the Groth-Lu co-soundness definition (see App. B), the adversary picks not only the final ciphertexts $\mathfrak{u}'_i$ but also the initial ciphertexts $\mathfrak{u}_i$.

In a real life application of the shuffle argument, the adversary (e.g., a malicious mix server) usually gets the ciphertexts $\mathfrak{u}_i$ from a third party (from voters, or from another mix server), and thus does not know their discrete logarithms. However, in such a case we can still prove soundness of the full e-voting system (including the voters and all mix servers) if we give the adversary access to secret coins of all relevant parties. The use of knowledge BBS guarantees that the encrypters (voters) know the plaintexts and the randomizers, and thus the use of knowledge BBS can be seen as a white-box non-interactive knowledge argument. This corresponds to the case in several interactive (or Fiat-Shamir heuristic based) shuffles, where the ballots are accompanied by a proof of knowledge of the actual vote, from what the (black-box) simulator obtains the actual plaintexts necessary to complete the simulation. We thus think that soundness in our model is relevant, and corresponds to the established cryptographic practice with a twist. We leave the question of whether this model is necessary in applications like e-voting (where initial ciphertexts are not provided by the mixservers), and when co-soundness is undesired, as an interesting open problem. Using the Groth-Lu co-soundness definition avoids this issue, since in that case the adversary does not have access to the random coins of the participants.

We note that Groth and Lu made in addition a similar assumption in [14] where they prove co-soundness against adversaries who also output and thus know the secret key of the cryptosystem. (See App. B for a precise definition.) Thus, the adversary can decrypt all the ciphertexts, and thus knows the plaintexts (but does not have to know the randomizers). As argued in [14], this is reasonable in the setting of mixnet where the servers can usually threshold-decrypt all the results. Their approach is however not applicable in our case, since the knowledge of the secret key enables the adversary to obtain the plaintexts and the randomizers in exponents, while to prove the soundness in Thm. 5 the adversary has to know the plaintexts and the randomizers themselves.

Next, we construct an efficient shuffle argument that works with the knowledge BBS cryptosystem of Sect. 4. Assume that the ciphertexts $(\mathfrak{u}_{i1}, \mathfrak{u}_{i2}, \mathfrak{u}_{i3}, \tilde{\mathfrak{u}}_{i1}, \tilde{\mathfrak{u}}_{i2}, \tilde{\mathfrak{u}}_{i3})$, where $i \in [n]$, are created as in Sect. 4. The shuffled ciphertexts with permutation $\psi \in S_n$ and randomizers $(\sigma'_i, \tau'_i)_{i\in[n]}$ are $\mathfrak{u}'_i = (\mathfrak{u}'_{i1}, \mathfrak{u}'_{i2}, \mathfrak{u}'_{i3}, \tilde{\mathfrak{u}}'_{i1}, \tilde{\mathfrak{u}}'_{i2}, \tilde{\mathfrak{u}}'_{i3}) = \mathfrak{u}_{\psi(i)} \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(0; \sigma'_i, \tau'_i) = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_{\psi(i)}; \sigma_{\psi(i)} + \sigma'_i, \tau_{\psi(i)} + \tau'_i)$. Let $P = P_{\psi^{-1}}$ denote the permutation matrix corresponding to the permutation ψ^{-1}.

The new shuffle argument is described in Prot. 4. Here, the prover first constructs a permutation matrix and a permutation matrix argument π^{pm}. After that, he shows that the plaintext vector of $\mathfrak{u}'_i$ is equal to the product of this permutation matrix and the plaintext vector of $\mathfrak{u}_i$. Importantly, we can prove the adaptive computational soundness of the shuffle argument. This is since while in the previous arguments one only relied on (perfectly hiding) knowledge commitment scheme and thus any commitment could commit at the same time to the correct value (for example, to a permutation matrix) and to an incorrect value (for example, to an all-zero matrix), here the group-dependent language contains statements about a public-key cryptosystem where any ciphertext can be uniquely decrypted. Thus, it makes sense to state that $(\mathsf{pk}, (\mathfrak{u}_i, \mathfrak{u}'_i)_{i\in[n]})$ is *not a shuffle*. To prove computational soundness, we need to rely on the PKE assumption. It is also nice to have a shuffle argument that satisfies a standard security notion.

Theorem 5. *Prot. 4 is a non-interactive perfectly complete and perfectly zero-knowledge shuffle argument of the knowledge BBS ciphertexts. Assume that μ is sufficiently small so that $\log_{g_1} g_1^{\mu}$ can be computed in polynomial time. If the Λ-PSDL, the DLIN, the KE (in group $\mathbb{G}_1$), and the $\bar{\Lambda}$-PKE (in group $\mathbb{G}_2$) assumptions hold, then the argument is also adaptively computationally sound.*

We recall that $\emptyset$-PKE is equal to the KE assumption (in the same bilinear group). Thus, if $\bar{\Lambda}$-PKE is hard then also Λ-PKE and KE are hard (in the same group).

Common reference string: Similarly to the permutation matrix argument, let $\bar{\alpha}, \mathring{\alpha}, x \leftarrow \mathbb{Z}_p$, $\bar{g}_t \leftarrow g_t^{\bar{\alpha}}$, $\mathring{g}_t \leftarrow g_t^{\mathring{\alpha}}$, $g_{t\ell} \leftarrow g_t^{x^\ell}$, and $\bar{g}_{t\ell} \leftarrow \bar{g}_t^{x^\ell}$. Let $D \leftarrow \prod_{i=1}^n g_{2,\lambda_i}$. In addition, let $\mathsf{sk}_1, \mathsf{sk}_2 \leftarrow \mathbb{Z}_p^*$ and $\tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3 \leftarrow \mathbb{Z}_p$. Let $f \leftarrow g_1^{1/\mathsf{sk}_1}$, $h \leftarrow g_1^{1/\mathsf{sk}_2}$, $\tilde{f} \leftarrow f^{\tilde{\alpha}_1}$, $\tilde{h} \leftarrow h^{\tilde{\alpha}_2}$, $\tilde{g}_1 \leftarrow g_1^{\tilde{\alpha}_3}$, $\tilde{g}_2^{(1)} \leftarrow g_2^{\tilde{\alpha}_1}$, $\tilde{g}_2^{(2)} \leftarrow g_2^{\tilde{\alpha}_2}$, and $\tilde{g}_2^{(3)} \leftarrow g_2^{\tilde{\alpha}_3}$.

The CRS is $\mathsf{crs} := (\bar{g}_1, \bar{g}_2, \mathring{g}_1, \mathring{g}_2, (g_{1\ell}, \bar{g}_{1\ell})_{\ell \in \Lambda}, (g_{2\ell}, \bar{g}_{2\ell})_{\ell \in \Lambda \cup (2\cdot\Lambda)}, D)$. The commitment keys are $\mathsf{ck}_t \leftarrow (\mathsf{gk}; \bar{g}_t, (g_{t\ell}, \bar{g}_{t\ell})_{\ell\in\Lambda})$ and $\mathring{\mathsf{ck}}_2 \leftarrow (\mathsf{gk}; \mathring{g}_2)$. The public key is $\mathsf{pk} = (\mathsf{gk}; \tilde{g}_1, \tilde{g}_2^{(1)}, \tilde{g}_2^{(2)}, \tilde{g}_2^{(3)}, f, \tilde{f}, h, \tilde{h})$, and the secret key is $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)$.

Common input: $(\mathsf{pk}, (\mathfrak{u}_i, \mathfrak{u}'_i)_{i\in[n]})$, where $\mathfrak{u}_i = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_i; \sigma_i, \tau_i) \in \mathbb{G}_1^3$ and $\mathfrak{u}'_i = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_{\psi(i)}; \sigma_{\psi(i)} + \sigma'_i, \tau_{\psi(i)} + \tau'_i) \in \mathbb{G}_1^3$.

Argument $\mathcal{P}_{sh}(\mathsf{gk}, \mathsf{crs}; (\mathsf{pk}, (\mathfrak{u}_i, \mathfrak{u}'_i)_{i\in[n]}), (\psi, (\sigma'_i, \tau'_i)_{i\in[n]}))$**:** the prover does the following.

1. Let $P = P_{\psi^{-1}}$ be the $n \times n$ permutation matrix corresponding to the permutation ψ^{-1}.
2. For $i \in [n]$, let $r_i \leftarrow \mathbb{Z}_p$ and $(c_{2i}, \bar{c}_{2i}) \leftarrow \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{P_i}; r_i) = (g_2^{r_i} \cdot g_{2,\lambda_{\psi^{-1}(i)}}, \bar{g}_2^{r_i} \cdot \bar{g}_{2,\lambda_{\psi^{-1}(i)}})$.
3. Generate a permutation matrix argument π^{pm} for inputs $(\boldsymbol{c_2}, \boldsymbol{\bar{c}_2})$.
4. Set $(R_\sigma, R_\tau) \leftarrow \mathbb{Z}_p^2$, $(c_\sigma, \bar{c}_\sigma) \leftarrow \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \sigma'_1, \ldots, \sigma'_n; R_\sigma)$, and $(c_\tau, \bar{c}_\tau) \leftarrow \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \tau'_1, \ldots, \tau'_n; R_\tau)$.
5. Compute $(\mathfrak{u}_\sigma, \tilde{\mathfrak{u}}_\sigma) \leftarrow (f^{R_\sigma} \cdot \prod_{i=1}^n \mathfrak{u}_{i1}^{r_i}, \tilde{f}^{R_\sigma} \cdot \prod_{i=1}^n \tilde{\mathfrak{u}}_{i1}^{r_i})$, $(\mathfrak{u}_\tau, \tilde{\mathfrak{u}}_\tau) \leftarrow (h^{R_\tau} \cdot \prod_{i=1}^n \mathfrak{u}_{i2}^{r_i}, \tilde{h}^{R_\tau} \cdot \prod_{i=1}^n \tilde{\mathfrak{u}}_{i2}^{r_i})$, $(\mathfrak{u}_\mu, \tilde{\mathfrak{u}}_\mu) \leftarrow (g_1^{R_\sigma + R_\tau} \cdot \prod_{i=1}^n \mathfrak{u}_{i3}^{r_i}, \tilde{g}_1^{R_\sigma + R_\tau} \cdot \prod_{i=1}^n \tilde{\mathfrak{u}}_{i3}^{r_i})$.
6. The argument is

$$\pi^{sh} \leftarrow ((c_{2i}, \bar{c}_{2i})_{i\in[n]}, \pi^{pm}, c_\sigma, \bar{c}_\sigma, c_\tau, \bar{c}_\tau, \mathfrak{u}_\sigma, \tilde{\mathfrak{u}}_\sigma, \mathfrak{u}_\tau, \tilde{\mathfrak{u}}_\tau, \mathfrak{u}_\mu, \tilde{\mathfrak{u}}_\mu) \ . \tag{2}$$

Verification $\mathcal{V}_{sh}(\mathsf{gk}, \mathsf{crs}; (\mathsf{pk}, (\mathfrak{u}_i, \mathfrak{u}'_i)_{i\in[n]}), \pi^{sh})$**:** the verifier does the following.

1. Check that $\hat{e}(\bar{g}_1, c_\sigma) = \hat{e}(g_1, \bar{c}_\sigma)$ and $\hat{e}(\bar{g}_1, c_\tau) = \hat{e}(g_1, \bar{c}_\tau)$.
2. Check that $\hat{e}(\mathfrak{u}_\sigma, \tilde{g}_2^{(1)}) = \hat{e}(\tilde{\mathfrak{u}}_\sigma, g_2)$, $\hat{e}(\mathfrak{u}_\tau, \tilde{g}_2^{(2)}) = \hat{e}(\tilde{\mathfrak{u}}_\tau, g_2)$, and $\hat{e}(\mathfrak{u}_\mu, \tilde{g}_2^{(3)}) = \hat{e}(\tilde{\mathfrak{u}}_\mu, g_2)$.
3. For $i \in [n]$, check that $\hat{e}(\mathfrak{u}_{i1}, \tilde{g}_2^{(1)}) = \hat{e}(\tilde{\mathfrak{u}}_{i1}, g_2)$, $\hat{e}(\mathfrak{u}_{i2}, \tilde{g}_2^{(2)}) = \hat{e}(\tilde{\mathfrak{u}}_{i2}, g_2)$, $\hat{e}(\mathfrak{u}_{i3}, \tilde{g}_2^{(3)}) = \hat{e}(\tilde{\mathfrak{u}}_{i3}, g_2)$, $\hat{e}(\mathfrak{u}'_{i1}, \tilde{g}_2^{(1)}) = \hat{e}(\tilde{\mathfrak{u}}'_{i1}, g_2)$, $\hat{e}(\mathfrak{u}'_{i2}, \tilde{g}_2^{(2)}) = \hat{e}(\tilde{\mathfrak{u}}'_{i2}, g_2)$, and $\hat{e}(\mathfrak{u}'_{i3}, \tilde{g}_2^{(3)}) = \hat{e}(\tilde{\mathfrak{u}}'_{i3}, g_2)$.
4. Check the permutation matrix argument π^{pm}.
5. Check that the following three equations hold:
   (a) $\hat{e}(f, c_\sigma) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i1}, c_{2i}) = \hat{e}(\mathfrak{u}_\sigma, g_2) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i1}, g_{2,\lambda_i})$,
   (b) $\hat{e}(h, c_\tau) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i2}, c_{2i}) = \hat{e}(\mathfrak{u}_\tau, g_2) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i2}, g_{2,\lambda_i})$, and
   (c) $\hat{e}(g_1, c_\sigma c_\tau) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i3}, c_{2i}) = \hat{e}(\mathfrak{u}_\mu, g_2) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i3}, g_{2,\lambda_i})$.

Protocol 4: New shuffle argument

Proof. PERFECT COMPLETENESS: To verify the proof, the verifier first checks the consistency of the commitments, ciphertexts and the permutation matrix argument; here one needs that the permutation matrix argument is perfectly complete. Assume that the prover is honest. The verification equation in step 5a holds since

$$\begin{aligned}\hat{e}(f, c_\sigma) \cdot \prod_{i=1}^{n} \hat{e}(\mathfrak{u}_{i1}, c_{2i}) =& \hat{e}(f, g_2^{R_\sigma} \cdot \prod_{i=1}^{n} g_{2,\lambda_i}^{\sigma'_i}) \cdot \prod_{i=1}^{n} (\hat{e}(\mathfrak{u}_{i1}, g_2^{r_i}) \cdot \hat{e}(f^{\sigma_i}, g_{2,\lambda_{\psi^{-1}(i)}})) \\ =& \hat{e}(f^{R_\sigma} \cdot \prod_{i=1}^{n} \mathfrak{u}_{i1}^{r_i}, g_2) \cdot \prod_{i=1}^{n} \hat{e}(f^{\sigma_{\psi(i)} + \sigma'_i}, g_{2,\lambda_i}) \\ =& \hat{e}(\mathfrak{u}_\sigma, g_2) \cdot \prod_{i=1}^{n} \hat{e}(\mathfrak{u}'_{i1}, g_{2,\lambda_i}) \enspace .\end{aligned}$$

The equations in steps 5b and 5c can be verified similarly.

ADAPTIVE COMPUTATIONAL SOUNDNESS: Let $\mathcal{A}$ be a non-uniform PPT adversary that, given gk and a crs, creates a statement $(\mathsf{pk} = (\mathsf{gk}; \tilde{g}_1, \tilde{g}_2^{(1)}, \tilde{g}_2^{(2)}, \tilde{g}_2^{(3)}, f, \tilde{f}, h, \tilde{h}), (\mathfrak{u}_i, \mathfrak{u}'_i)_{i \in [n]})$ and an accepting NIZK argument π^{sh} (as in Eq. (2) in Prot. 4), such that the plaintext vector $(\mathfrak{u}'_i)_{i \in [n]}$ is not a permutation of the plaintext vector $(\mathfrak{u}_i)_{i \in [n]}$. Assume that the DLIN assumption holds in $\mathbb{G}_1$, the KE assumption holds in $\mathbb{G}_1$ and $\bar{\Lambda}$-PKE (and thus also Λ-PKE and KE) assumption holds in $\mathbb{G}_2$. We now construct an adversary $\mathcal{A}'$ that breaks the Λ-PSDL assumption.

Recall that π^{pm} contains values π^0 and $\pi_i^{spa} = (c_{1i}, \bar{c}_{1i}, F_i, \bar{F}_i)$. By applying the relevant knowledge assumption, we can postulate the existence of the following non-uniform PPT knowledge extractors that, with all but a negligible probability, return certain values:

- By the KE assumption in group $\mathbb{G}_1$, there exists a knowledge extractor that, given $(\mathfrak{u}_{ij}, \tilde{\mathfrak{u}}_{ij}, \mathfrak{u}'_{ij}, \tilde{\mathfrak{u}}'_{ij})_{j \in [3]}$ and access to $\mathcal{A}$'s random coins, returns the values μ_i, σ_i, τ_i, μ'_i, σ'_i and τ'_i, such that $\mathfrak{u}_i = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_i; \sigma_i, \tau_i)$ and $\mathfrak{u}'_i = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu'_i; \sigma'_i, \tau'_i)$. Note that it might be the case that $\mu'_i \neq \mu_{\varrho(i)}$.
- By the Λ-PKE assumption in group $\mathbb{G}_2$, there exists a knowledge extractor that, given $(c_\sigma, \bar{c}_\sigma, c_\tau, \bar{c}_\tau)$ and access to $\mathcal{A}$'s random coins, returns openings $(\boldsymbol{\sigma}^*, R_\sigma)$ and $(\boldsymbol{\tau}^*, R_\tau)$, such that $(c_\sigma, \bar{c}_\sigma) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{\sigma}^*; R_\sigma)$ and $(c_\tau, \bar{c}_\tau) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{\tau}^*; R_\tau)$. It does not have to hold that $\sigma'_i = \sigma_{\psi(i)} + \sigma^*_i$ and $\tau'_i = \tau_{\psi(i)} + \tau^*_i$ for $i \in [n]$.
- By the KE assumption in group $\mathbb{G}_1$, there exists a knowledge extractor that, given $(\mathfrak{u}_\sigma, \tilde{\mathfrak{u}}_\sigma, \mathfrak{u}_\tau, \tilde{\mathfrak{u}}_\tau, \mathfrak{u}_\mu, \tilde{\mathfrak{u}}_\mu)$ and access to $\mathcal{A}$'s random coins, returns openings $(v_\sigma, v_\tau, v_\mu)$, such that $(\mathfrak{u}_\sigma, \tilde{\mathfrak{u}}_\sigma) = (f^{v_\sigma}, \tilde{f}^{v_\sigma})$, $(\mathfrak{u}_\tau, \tilde{\mathfrak{u}}_\tau) = (h^{v_\tau}, \tilde{h}^{v_\tau})$, and $(\mathfrak{u}_\mu, \tilde{\mathfrak{u}}_\mu) = (g_1^{v_\mu}, \tilde{g}_1^{v_\mu})$. (Thus, it is not necessary that the adversary created the values $\mathfrak{u}_\sigma$, $\mathfrak{u}_\tau$ and $\mathfrak{u}_\mu$ correctly, it is just needed that she knows their discrete logarithms.)
- By the KE assumption in group $\mathbb{G}_2$, there exists a knowledge extractor that, given $((\prod_{i=1}^{n} c_{2i})/D, \pi^0)$ and access to $\mathcal{A}$'s random coins, returns an opening $((a_i)_{i \in [n]}, r_a)$, such that $((\prod_{i=1}^{n} c_{2i})/D, \pi^0) = \mathcal{C}\mathrm{om}^2(\mathring{\mathsf{ck}}_2; \boldsymbol{a}; r_a)$.

- By the Λ-PKE assumption in group $\mathbb{G}_2$, for every $i \in [n]$ there exists a knowledge extractor that, given $(c_{2i}, \bar{c}_{21})$ and access to $\mathcal{A}$'s random coins, returns an opening $((P_{ij})_{j\in[n]}, r_i)$ such that $(c_{2i}, \bar{c}_{2i}) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{P_i}; r_i)$.
- By the $\bar{\Lambda}$-PKE assumption in group $\mathbb{G}_2$, for every i there exists a knowledge extractor that, given $(F_i, \bar{F}_i)$ and access to $\mathcal{A}$'s random coins, returns openings $(f'_{ij})_{j\in\bar{\Lambda}}$ such that $\log_{g_2} F_i = \sum_{j\in\bar{\Lambda}} f'_{ij} x^j$.

The probability that any of these extractors fails is negligible, in this case we can abort. In the following, we will assume that all extractors succeeded.

Let $\mathfrak{a}$ be $\mathcal{A}$'s output. Based on $\mathcal{A}$ and the last three type of extractors, we can build an adversary $\mathcal{A}'$ that returns $\mathfrak{a}$ together with $((a_i)_{i\in[n]}, r_a, (\boldsymbol{P_i}, r_i, (f'_{ij})_{j\in\bar{\Lambda}})_{i\in[n]})$. Since the permutation matrix argument is (weakly) sound (as defined in the last statement of Thm. 4) and π^{pm} verifies, we have that $\boldsymbol{c_2} = (c_{2i})_{i\in[n]}$ commits to a permutation matrix. Thus, there exists $\psi \in S_n$ such that for every $i \in [n]$, $c_{2i} = \exp(g_2, r_i + x^{\lambda_{(\psi^{-1}(i))}})$.

Assume now that the equation in step 5a holds. Then

$$\begin{aligned}
\hat{e}(\mathfrak{u}_\sigma, g_2) =&\hat{e}(f, c_\sigma) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i1}, c_{2i}) / \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i1}, g_{2,\lambda_i}) \\
=&\hat{e}(f, g_2^{R_\sigma + \sum_{i=1}^n \sigma_i^* x^{\lambda_i}}) \cdot \prod_{i=1}^n \hat{e}(f^{\sigma_i}, g_2^{r_i + x^{\lambda_{\psi^{-1}(i)}}}) / \prod_{i=1}^n \hat{e}(f^{\sigma'_i}, g_2^{x^{\lambda_i}}) \\
=&\hat{e}(f^{R_\sigma + \sum_{i=1}^n \sigma_i r_i + \sum_{i=1}^n (\sigma_{\psi(i)} + \sigma_i^* - \sigma'_i) x^{\lambda_i}}, g_2) \enspace .
\end{aligned}$$

Since $\mathfrak{u}_\sigma = f^{v_\sigma}$, $\sum_{i=1}^n (\sigma_{\psi(i)} + \sigma_i^* - \sigma'_i) x^{\lambda_i} + R_\sigma + \sum_{i=1}^n \sigma_i r_i - v_\sigma = 0$. If $\sigma'_i \neq \sigma_{\psi(i)} + \sigma_i^*$ for some $i \in [n]$, then the adversary has succeeded in creating a non-trivial polynomial $f^*(X) = \sum_{i=1}^n f_i^* X^{\lambda_i} + f_0^*$, with $f_i^* = \sigma_{\psi(i)} + \sigma_i^* - \sigma'_i$ and $f_0^* = R_\sigma + \sum_{i=1}^n \sigma_i r_i - v_\sigma$, such that $f^*(x) = 0$. By using an efficient polynomial factorization algorithm, one can now find all $\lambda_n + 1$ roots of $f^*(X)$. For one of those roots, say y, we have $g_2^y = g_2^x$. $\mathcal{A}'$ can now return $y = x$. Since $(\mathsf{gk}, \mathsf{crs})$ only contains f^{x^ℓ} for $\ell = 0$, the adversary has thus broken the $\emptyset$-PSDL assumption, an assumption that is true unconditionally since the adversary's input does not depend on x at all. Thus, $\sigma'_i = \sigma_{\psi(i)} + \sigma_i^*$ for $i \in [n]$.

Analogously, by the verification in step 5b, $\sum_{i=1}^n (\tau_{\psi(i)} + \tau_i^* - \tau'_i) x^{\lambda_i} + R_\tau + \sum_{i=1}^n \tau_i r_i - v_\tau = 0$, and thus, $\tau'_i = \tau_{\psi(i)} + \tau_i^*$ for all $i \in [n]$.

Finally, by the verification in step 5c,

$$\begin{aligned}
\hat{e}(\mathfrak{u}_\mu, g_2) =&\hat{e}(g_1, c_\sigma c_\tau) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i3}, c_{2i}) / \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i3}, g_{2,\lambda_i}) \\
=&\hat{e}(g_1, g_2^{R_\sigma + R_\tau + \sum_{i=1}^n (\sigma_i^* + \tau_i^*) x^{\lambda_i}}) \cdot \\
&\prod_{i=1}^n \hat{e}(g_1^{\mu_i + \sigma_i + \tau_i}, \exp(g_2, r_i + x^{\lambda_{\psi^{-1}(i)}})) / \prod_{i=1}^n \hat{e}(g_1^{\mu'_i + \sigma'_i + \tau'_i}, g_2^{x^{\lambda_i}}) \enspace .
\end{aligned}$$

Thus,

$$\begin{aligned}\log_{g_1} \mathfrak{u}_\mu =& R_\sigma + R_\tau + \sum_{i=1}^{n}(\sigma_i^* + \tau_i^*)x^{\lambda_i} + \sum_{i=1}^{n}(\mu_i + \sigma_i + \tau_i)(r_i + x^{\lambda_{\psi^{-1}(i)}}) - \\ & \sum_{i=1}^{n}(\mu_i' + \sigma_i' + \tau_i')x^{\lambda_i} \\ =& R_\sigma + R_\tau + \sum_{i=1}^{n}(\mu_i + \sigma_i + \tau_i)r_i + \\ & \sum_{i=1}^{n}(\mu_{\psi(i)} - \mu_i' + \sigma_{\psi(i)} + \sigma_i^* - \sigma_i' + \tau_{\psi(i)} + \tau_i^* - \tau_i')x^{\lambda_i} \\ =& R_\sigma + R_\tau + \sum_{i=1}^{n}(\mu_i + \sigma_i + \tau_i)r_i + \sum_{i=1}^{n}(\mu_{\psi(i)} - \mu_i')x^{\lambda_i} \quad .\end{aligned}$$

If $\mu_i' \neq \mu_{\psi(i)}$ for some $i \in [n]$, then the adversary has succeeded in creating a non-trivial polynomial $f^*(X) = \sum_{i=1}^n f_i^* X^{\lambda_i} + f_0^*$, with $f_i^* = \sum_{i=1}^n(\mu_{\psi(i)} - \mu_i')$ and $f_0^* = R_\sigma + R_\tau + \sum_{i=1}^n(\mu_i + \sigma_i + \tau_i)r_i - v_\mu$, such that $f^*(x) = 0$. By using an efficient polynomial factorization algorithm, one can now find all $\lambda_n + 1$ roots of f^*. For one of those roots, say y, we have $g_2^y = g_2^x$. Since $(\mathsf{gk}, \mathsf{crs})$ only contains $g_1^{x^\ell}$ for $\ell \in \Lambda$, the adversary has thus broken the Λ-PSDL assumption. Therefore, due to the Λ-PSDL assumption, $\mu_i' = \mu_{\psi(i)}$ for $i \in [n]$.[3]

Thus, $\mathfrak{u}_{i1}' = f^{\sigma_{\psi(i)} + \sigma_i^*}$, $\mathfrak{u}_{i2}' = h^{\tau_{\psi(i)} + \tau_i^*}$, $\mathfrak{u}_{i3}' = g_1^{\mu_{\psi(i)} + \sigma_{\psi(i)} + \sigma_i^* + \tau_{\psi(i)} + \tau_i^*}$ and similarly for elements $\tilde{\mathfrak{u}}_{ij}'$, and therefore, $\{\mathfrak{u}_i'\}$ is indeed a correct shuffle of $\{\mathfrak{u}_i\}$.

PERFECT ZERO-KNOWLEDGE: We construct a simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ as follows. First, $\mathcal{S}_1$ generates random $\mathring{a}, \bar{\alpha}, x \leftarrow \mathbb{Z}_q$, and sets $\mathsf{td} \leftarrow (\mathring{a}, \bar{\alpha}, x)$. He then creates crs as in Prot. 4, and stores td. The construction of $\mathcal{S}_2$ is given in Prot. 5. Next, we give an analysis of the simulated proof. Note that c_σ, c_τ and c_{2i} are independent and random variables in $\mathbb{G}$, exactly as in the real run of the protocol. With respect to those variables, we define $\mathfrak{u}_\sigma$, $\mathfrak{u}_\tau$ and $\mathfrak{u}_\mu$ so that they satisfy the verification equations. Thus, we are now only left to show that the verification equations in steps 5a, 5b and 5c hold.

Clearly, π^{pm} is simulated correctly, since $\hat{e}\left(\mathring{g}_1, (\prod_{i=1}^n c_{2i})/D\right) = \hat{e}(g_1, \pi^0)$, $\hat{e}(c_{1i}, g_2) = \hat{e}(g_1, c_{2i})$, $\hat{e}(\bar{c}_{1i}, g_2) = \hat{e}(c_{1i}, \bar{g}_2)$, $\hat{e}(g_1, \bar{c}_{2i}) = \hat{e}(\bar{g}_1, c_{2i})$, $\hat{e}\left(g_1, \bar{F}_i\right) = \hat{e}(\bar{g}_1, F_i)$, and $\hat{e}(c_{1i}, c_{2i}) = \hat{e}(g_1^{z_i}, g_2^{z_i}) = \hat{e}\left(g_1, g_2^{z_i^2}\right) = \hat{e}(g_1, F_i)$.

[3] For the argument in this paragraph to go through, we need the knowledge BBS cryptosystem to be lifted and the plaintexts to be small. Otherwise, the adversary will not know the coefficients of $f'(X)$, and thus one could not use a polynomial factorization algorithm to break the Λ-PSDL assumption. Thus, a crafty adversary might be able to break soundness by choosing g_1^μ from which she cannot compute μ.

Inputs: gk **and CRS as in Prot. 4, trapdoor** $\mathsf{td} = (\mathring{\alpha}, \bar{\alpha}, x)$**, and** $(\mathsf{pk}, (\mathfrak{u}_i, \mathfrak{u}'_i)_{i \in [n]})$
Output: π^{sh}
Simulation:

1. Pick random $z_i, r_{i1}, r_{i2} \leftarrow \mathbb{Z}_p$ for $i \in [n]$.
2. Set $c_\sigma \leftarrow \prod_{i=1}^n g_2^{r_{i1}}$, $c_\tau \leftarrow \prod_{i=1}^n g_2^{r_{i2}}$, $c_{2i} \leftarrow g_2^{z_i}$ and $\bar{c}_{2i} \leftarrow \bar{g}_2^{z_i}$ for $i \in [n]$.
3. Set $(\mathfrak{u}_\sigma, \tilde{\mathfrak{u}}_\sigma) \leftarrow (\prod_{i=1}^n (f^{r_{i1}} \cdot \mathfrak{u}_{i1}^{z_i} \cdot (\mathfrak{u}'_{i1})^{-x^{\lambda_i}}), \prod_{i=1}^n (\tilde{f}^{r_{i1}} \cdot \tilde{\mathfrak{u}}_{i1}^{z_i} \cdot (\tilde{\mathfrak{u}}'_{i1})^{-x^{\lambda_i}}))$, $(\mathfrak{u}_\tau, \tilde{\mathfrak{u}}_\tau) \leftarrow (\prod_{i=1}^n (h^{r_{i2}} \cdot \mathfrak{u}_{i2}^{z_i} \cdot (\mathfrak{u}'_{i2})^{-x^{\lambda_i}}), \prod_{i=1}^n (\tilde{h}^{r_{i2}} \cdot \tilde{\mathfrak{u}}_{i2}^{z_i} \cdot (\tilde{\mathfrak{u}}'_{i2})^{-x^{\lambda_i}}))$, $(\mathfrak{u}_\mu, \tilde{\mathfrak{u}}_\mu) \leftarrow (\prod_{i=1}^n (g_1^{r_{i1}+r_{i2}} \cdot \mathfrak{u}_{i3}^{z_i} \cdot (\mathfrak{u}'_{i3})^{-x^{\lambda_i}}), \prod_{i=1}^n (\tilde{g}_1^{r_{i1}+r_{i2}} \cdot \tilde{\mathfrak{u}}_{i3}^{z_i} \cdot (\tilde{\mathfrak{u}}'_{i3})^{-x^{\lambda_i}}))$.
4. Complete the remaining part of the proof.
5. Simulate π^{pm} by using the trapdoor opening of commitments as follows:
   (a) Let $\pi^0 \leftarrow ((\prod_{i=1}^n c_{2i})/D)^{\mathring{\alpha}}$.
   (b) Let π_i^{spa} be a 1-sparsity argument that $(c_{2i}, \bar{c}_{2i})$ commits to a 1-sparse vector. That is, $\pi_i^{spa} = (c_{1i}, \bar{c}_{1i}, F_i, \bar{F}_i)$ for $c_{1i} \leftarrow g_1^{z_i}$, $\bar{c}_{1i} \leftarrow \bar{g}_1^{z_i}$, $F_i \leftarrow g_2^{z_i^2}$, $\bar{F}_i \leftarrow \bar{g}_2^{z_i^2}$.
   (c) Let $\pi^{pm} \leftarrow (\pi^0, \boldsymbol{\pi^{spa}})$.
6. Set $\pi^{sh} \leftarrow ((c_{2i}, \bar{c}_{2i})_{i \in [n]}, \pi^{pm}, c_\sigma, \bar{c}_\sigma, c_\tau, \bar{c}_\tau, \mathfrak{u}_\sigma, \tilde{\mathfrak{u}}_\sigma, \mathfrak{u}_\tau, \tilde{\mathfrak{u}}_\tau, \mathfrak{u}_\mu, \tilde{\mathfrak{u}}_\mu)$.

Protocol 5: Simulator $\mathcal{S}_2$: construction

Finally, we have

$$\begin{aligned}\hat{e}(f, c_\sigma) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i1}, c_{2i}) =& \hat{e}(f, \prod_{i=1}^n g_2^{r_{i1}}) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i1}, g_2^{z_i}) = \hat{e}(\prod_{i=1}^n f^{r_{i1}} \cdot \prod_{i=1}^n \mathfrak{u}_{i1}^{z_i}, g_2) \\ =& \hat{e}(\prod_{i=1}^n (f^{r_{i1}} \mathfrak{u}_{i1}^{z_i} (\mathfrak{u}'_{i1})^{-x^{\lambda_i}}), g_2) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i1}, g_{2,\lambda_i}) \\ =& \hat{e}(\mathfrak{u}_\sigma, g_2) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i1}, g_{2,\lambda_i}) \enspace .\end{aligned}$$

Similarly, $\hat{e}(h, c_\tau) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i2}, c_{2i}) = \hat{e}(\mathfrak{u}_\tau, g_2) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i2}, g_{2,\lambda_i})$ and $\hat{e}(g_1, c_\sigma c_\tau) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}_{i3}, c_{2i}) = \hat{e}(\mathfrak{u}_\mu, g_2) \cdot \prod_{i=1}^n \hat{e}(\mathfrak{u}'_{i3}, g_{2,\lambda_i})$. Thus all three verification equations hold, and therefore the simulator has succeeded in generating an argument that has the same distribution as the real argument. □

Theorem 6. *Consider Prot. 4. The CRS consists of* $2n + 2$ *elements of* $\mathbb{G}_1$ *and* $5n + 4$ *elements of* $\mathbb{G}_2$*, in total* $7n+6$ *group elements. The communication complexity is* $2n+6$ *elements of* $\mathbb{G}_1$ *and* $4n+5$ *elements of* $\mathbb{G}_2$*, in total* $6n+11$ *group elements. The prover's computational complexity is dominated by* $17n + 16$ *exponentiations. The verifier's computational complexity is dominated by* $28n + 18$ *pairings.*

We note that in a mix server-like application where several shuffles are done sequentially, one can get somewhat smaller amortized cost. Namely, the output ciphertext $\mathfrak{u}'_i$ of one shuffle is equal to the input ciphertext $\mathfrak{u}_i$ of the following shuffle. Therefore, in step 3, one only has to check the correctness of the ciphertexts $\mathfrak{u}'_i$ in the case of the very last shuffle. This means that the verifier's amortized computational complexity is dominated by $22n + 18$ pairings (that is, one has thus saved $6n$ pairings).

Acknowledgments. We would like to thank Jens Groth for insightful comments. The work was done while the second author was working at the University of Tartu. The authors were supported by Estonian Science Foundation, grant #9303, and European Union through the European Regional Development Fund.

References

1. Abe, M., Fehr, S.: Perfect NIZK with Adaptive Soundness. In: Vadhan, S. (ed.) TCC 2007. LNCS, vol. 4392, pp. 118–136. Springer, Heidelberg, Amsterdam, The Netherlands (Feb 21–24, 2007)
2. Bayer, S., Groth, J.: Efficient Zero-Knowledge Argument for Correctness of a Shuffle. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 263–280. Springer, Heidelberg, Cambridge, UK (Apr 15–19, 2012)
3. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M.K. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg, Santa Barbara, USA (Aug 15–19, 2004)
4. Chaabouni, R., Lipmaa, H., Zhang, B.: A Non-Interactive Range Proof with Constant Communication. In: Keromytis, A. (ed.) FC 2012. LNCS, vol. ?, pp. ?–? Springer, Heidelberg, Bonaire, The Netherlands (February 27–March 2, 2012)
5. Clos, C.: A Study of Non-Blocking Switching Networks. Bell System Technical Journal 32(2), 406–424 (Mar 1953)
6. Damgård, I.: Towards Practical Public Key Systems Secure against Chosen Ciphertext Attacks. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 445–456. Springer, Heidelberg, 1992, Santa Barbara, California, USA (Aug 11–15, 1991)
7. Elkin, M.: An Improved Construction of Progression-Free Sets. Israeli Journal of Mathematics 184, 93–128 (2011)
8. Erdős, P., Turán, P.: On Some Sequences of Integers. Journal of the London Mathematical Society 11(4), 261–263 (1936)
9. Furukawa, J., Sako, K.: An Efficient Scheme for Proving a Shuffle. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 368–387. Springer, Heidelberg, Santa Barbara, USA (Aug 19–23, 2001)
10. Golle, P., Jarecki, S., Mironov, I.: Cryptographic Primitives Enforcing Communication and Storage Complexity. In: Blaze, M. (ed.) FC 2002. LNCS, vol. 2357, pp. 120–135. Springer, Heidelberg, Southhampton Beach, Bermuda (Mar 11–14, 2002)
11. Groth, J.: Linear Algebra with Sub-linear Zero-Knowledge Arguments. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 192–208. Springer, Heidelberg, Santa Barbara, California, USA (Aug 16–20, 2009)
12. Groth, J.: Short Pairing-Based Non-interactive Zero-Knowledge Arguments. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 321–340. Springer, Heidelberg, Singapore (Dec 5–9 2010)
13. Groth, J., Ishai, Y.: Sub-linear Zero-Knowledge Argument for Correctness of a Shuffle. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 379–396. Springer, Heidelberg, Istanbul, Turkey (Apr 13–17, 2008)
14. Groth, J., Lu, S.: A Non-interactive Shuffle with Pairing Based Verifiability. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 51–67. Springer, Heidelberg, Kuching, Malaysia (Dec 2–6, 2007)
15. Groth, J., Ostrovsky, R., Sahai, A.: Perfect Non-Interactive Zero-Knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 338–359. Springer, Heidelberg, St. Petersburg, Russia (May 28–June 1, 2006)

16. Groth, J., Ostrovsky, R., Sahai, A.: New Techniques for Non-interactive Zero Knowledge (Mar 7, 2011), full version of [15]. Draft, available from the authors
17. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg, Istanbul, Turkey (Apr 13–17, 2008)
18. Lipmaa, H.: Progression-Free Sets and Sublinear Pairing-Based Non-Interactive Zero-Knowledge Arguments. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 169–189. Springer, Heidelberg, Taormina, Italy (Mar 18–21, 2012)
19. Lipmaa, H., Zhang, B.: A More Efficient Computationally Sound Non-Interactive Zero-Knowledge Shuffle Argument. Tech. Rep. 2011/394, International Association for Cryptologic Research (Jul 21, 2011), available at http://eprint.iacr.org/2011/394
20. Rial, A., Kohlweiss, M., Preneel, B.: Universally Composable Adaptive Priced Oblivious Transfer. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 231–247. Springer, Heidelberg, Palo Alto, CA, USA (Aug 12–14, 2009)
21. Sanders, T.: On Roth's Theorem on Progressions. Annals of Mathematics 174(1), 619–636 (Jul 2011)
22. Tao, T., Vu, V.: Additive Combinatorics. Cambridge Studies in Advanced Mathematics, Cambridge University Press (2006)
23. Terelius, B., Wikström, D.: Proofs of Restricted Shuffles. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 100–113. Springer, Heidelberg, Stellenbosch, South Africa (May 3–6, 2010)

A Decisional Linear Assumption

We say that a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is DLIN (decisional linear) secure [3] in group $\mathbb{G}_t$, for $t \in \{1, 2\}$, if for all non-uniform polynomial time adversaries $\mathcal{A}$, the following probability is negligible in κ:

$$\left| \Pr \begin{bmatrix} \mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), \\ (f, h) \leftarrow (\mathbb{G}_t^*)^2, (\sigma, \tau) \leftarrow \mathbb{Z}_p^2 : \\ \mathcal{A}(\mathsf{gk}; f, h, f^\sigma, h^\tau, g_t^{\sigma+\tau}) = 1 \end{bmatrix} - \Pr \begin{bmatrix} \mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), \\ (f, h) \leftarrow (\mathbb{G}_t^*)^2, (\sigma, \tau, z) \leftarrow \mathbb{Z}_p^3 : \\ \mathcal{A}(\mathsf{gk}; f, h, f^\sigma, h^\tau, g_t^{z}) = 1 \end{bmatrix} \right| .$$

B Groth-Lu Co-Soundness Definition

The Groth-Lu shuffle argument is proven to be R_{co}^{sh}-sound with respect to the next language [14] (here, as in [14], we assume the setting of symmetric pairings $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and like [14] we give the definition with respect to the BBS cryptosystem only):

$$R_{co}^{sh} := \left\{ \begin{array}{l} ((p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g), (f, h, \{\mathfrak{u}_i\}, \{\mathfrak{u}'_i\}), \mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)) : (x, y) \in (\mathbb{Z}_p^*)^2 \wedge \\ f = g^{\mathsf{sk}_1} \wedge h = g^{\mathsf{sk}_2} \wedge \left(\forall \psi \in S_n \exists i : \mathcal{D}\mathsf{ec}_{\mathsf{sk}}(\mathfrak{u}'_i) \neq \mathcal{D}\mathsf{ec}_{\mathsf{sk}}(\mathfrak{u}_{\psi(i)})\right) \end{array} \right\} .$$

That is, the adversary is required to return not only a non-shuffle $(\{\mathfrak{u}_i\}, \{\mathfrak{u}'_i\})$, but also a secret key sk that makes it possible to verify efficiently that $(\{\mathfrak{u}_i\}, \{\mathfrak{u}'_i\})$ is really not a shuffle. As argued in [14], this definition of R_{co}^{sh} makes sense in practice, since there is always some coalition of the parties who knows the secret key. See [14] for more.