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

Helger Lipmaa and Bingsheng Zhang

University of Tartu, Estonia

Abstract. We propose a new pairing-based non-interactive perfectly zero-knowledge shuffle argument that has smaller communication and is based on more standard computational cryptographic assumptions than the only previously known efficient non-interactive zero-knowledge shuffle argument by Groth and Lu. Differently from Groth and Lu who only prove the culpable soundness (a weaker version of computational soundness) of their argument, we provide a proof of the computational soundness. Due to well-known impossibility results this means that we also have to use a knowledge assumption. We first construct an efficient permutation matrix argument by using recent non-interactive zero-knowledge techniques of Groth and Lipmaa, and then use it to construct a non-interactive shuffle argument for a knowledge version of the Boneh-Boyen-Shacham cryptosystem.

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. A shuffle argument is needed in applications like e-voting and anonymous broadcast. In the case of e-voting, shuffles are used to destroy the relation between the voters and their ballots. As a concrete example, the voters 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 verify, 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). As a completely different application, we point out simulatable oblivious transfer [KNP10,KNP11], where the use of shuffle makes it possible for the client to query database elements by using permuted indexes.

A lot of research has been conducted in the area of constructing secure and efficient shuffle arguments (see [Cha81,Nef01,FS01,Gro03] for some classic papers), 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). Practically all previous shuffle arguments are interactive, and can only be made non-interactive by using a random oracle. For example, Groth and Ishai [GI08] and Groth [Gro09] have constructed shuffle arguments with communication $\Theta(n^{2/3})$ and $\Theta(n^{1/2})$ respectively, where n is the number of ciphertexts. However, they make use of the Schwartz-Zippel lemma [Sch80] that 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 [CGH98,GK03] that there are protocols that are secure in the random oracle model but not in the plain model, given any instantiation of the random oracle. 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 was proposed by Groth and Lu in [GL07]. Their argument shuffles the ciphertexts of the BBS cryptosystem [BBS04]. The security of the Groth-Lu argument is based on two new computational assumptions, the permutation pairing assumption (PPA, see App. A) and the simultaneous pairing assumption (SPA). While Groth and Lu proved that their assumptions are secure in the generic group

* First eprint version was published on July 21, 2011. In this second eprint version from August 20, 2011, we include proper discussion of culpable soundness, and correct several small mistakes.

	|CRS|	Comm.	$\mathcal{P}$'s comp.	$\mathcal{V}$'s comp.	Pairing	Soundness	Assumption
[GL07]	$2n+8$	$15n+120$	$\Theta(n)$	$\Theta(n)$	Sym.	Culp.	PPA + SPA + DLIN
Sect. 5	$7n+14$	$6n+11$	$17n+16$	$28n+18$	Asym.	Sound	PKE + PSDL + DLIN

Table 1. Brief comparison of existing (not random-oracle based) 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

model, one can argue that their assumptions are specifically constructed so as the concrete shuffle argument will be culpably sound [GOS11] (also called co-sound[1], see [GL07] and Sect. 2 of the current paper). It is therefore interesting to construct a shuffle argument from "more standard" assumptions. Moreover, their shuffle argument has a relatively large communication complexity of $15n + 120$ symmetric bilinear group elements. (See Tbl. 1 for a comparison.)

Our Contributions. 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 is given later.) Recall that a permutation matrix is a Boolean matrix that has exactly one 1 in every row and column. From a very high-level point of view, 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). Second, we prove that the plaintext vector corresponding to the output ciphertext tuple is equal to the product of this matrix and the plaintext vector corresponding to the input ciphertext tuple, and thus is correctly formed. Both parts are involved. In particular, coming up with a characterization of permutation matrices that allows an efficient cryptographic implementation was not an easy task.

In [TW10], Terelius and Wikström constructed an interactive permutation matrix argument based on the fact that a matrix is a permutation matrix exactly if 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 need to use a random oracle). Unfortunately, we do not know how to prove efficiently in NIZK that a commitment commits to a unit vector. We propose a superficially similar permutation matrix argument. It 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, which 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 the recent techniques of Groth [Gro10] and Lipmaa [Lip11] who proposed a perfectly NIZK argument for circuit satisfiability based on two subarguments (for Hadamard product and permutation). The problem with using their techniques directly is that in their subarguments, the prover has quadratic computational complexity; this is not acceptable in our case. Instead, we propose two new basic arguments (a zero argument, see Sect. 3.1, and a 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. On the other hand, the sparsity argument (the prover can open the given commitment to a tuple where at most one coordinate is non-zero) is conceptually new, at least up to our knowledge. Interestingly, like the basic arguments of [Lip11], the new sparsity argument relies on the existence of a dense progression-free set.

In Sect. 5, we combine the new permutation matrix argument with a knowledge version of the BBS [BBS04] cryptosystem to obtain an efficient non-interactive zero-knowledge shuffle argument. Informally, by the KE assumption [Dam91], 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

[1] The current version of [GOS11] uses the name culpable soundness, since the notion of co-soundness already has several different meanings. The definition of culpable soundness in [GOS11] and in the current paper is the same as the definition of co-soundness in say [GL07].

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 [Dam91].

In a shuffle argument, a part of the witness is the permutation $\psi : [n] \to [n]$ that can be described in $\Theta(n \log n)$ bits. Thus, all communication-$\Theta(n)$ shuffle arguments are sublinear in the witness size. To overcome the impossibility result of [AF07,GW11] (that basically state that one cannot construct computationally sound and perfectly zero-knowledge sublinear arguments based on any non-falsifiable assumptions), Groth and Lu [GL07] proved culpable soundness [GL07,GOS11] of their argument under purely computational assumptions.

Our basic arguments (the zero argument, the sparsity argument, and the permutation matrix argument) cannot be proven to be computationally sound since their "languages" are based on a perfectly hiding commitment scheme, see Sect. 3. Therefore, we prove that these arguments are culpably sound under purely computational assumptions. Unfortunately, the way knowledge commitments are used in [Gro10,Lip11], while being useful in constructing very efficient arguments, makes the definition of culpable soundness dependent on the construction of the concrete argument. (See [Lip11] for more discussion.) This can be seen say in Sect. 3.3, where the definition of the culpable soundness of the permutation matrix argument already looks somewhat convoluted. We could use a similar definition of the culpable soundness of the shuffle argument and prove that the new shuffle argument is culpably sound by using only standard computational assumptions. Instead (mostly since the definition of computational soundness is more standard), we chose to prove computational soundness of the shuffle argument under a previously known knowledge assumption. In particular, 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 [BBS04] (which is required for the CPA-security of the BBS cryptosystem), and on the power symmetric discrete logarithm (PSDL, see Sect. 2) assumption from [Lip11]. The PSDL assumption is much more standard(-looking) than the SPA and PPA assumptions from [GL07].

Tbl. 1 provides a brief comparison between the Groth-Lu shuffle argument and the new shuffle argument. We have omitted the precise computational complexity of the Groth-Lu argument (it has not been stated in [GL07], but it seems to be higher than that of the new shuffle argument). Significant speedups can be achieved in both cases by using efficient multi-exponentiation algorithms. The new protocol uses asymmetric pairings $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, while [GL07] 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. On the other hand, 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 however is quite small compared to the achieved reduction in the argument size. Another drawback of our scheme as compared to [GL07] is that it uses a lifted cryptosystem, and thus can be only used to shuffle small plaintexts. This is however fine in our imagined applications like e-voting (where the plaintext is a candidate number) or oblivious transfer (where the plaintext is an index to the database). Many of the existing e-voting schemes (for example, [CGS97]) are based on (lifted) Elgamal and thus require the plaintexts to be small. For a meaningful computational comparison, one should implement the shuffle arguments.

2 Preliminaries

Notation. 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 also sometimes write $\exp(h, x)$ instead of h^x. Let κ be the security parameter. By using notation that is common in additive combinatorics [TV06], if Λ is a set, then let $2 \cdot \Lambda = \{2\lambda : \lambda \in \Lambda\}$ and $2\Lambda = \{\lambda + \lambda' : \lambda, \lambda' \in \Lambda\}$. Thus, for example,

$$2\Lambda \setminus 2 \cdot \Lambda = \{\lambda + \lambda' : \lambda, \lambda' \in \Lambda \wedge \lambda \neq \lambda'\} \quad .$$

We say that $\Lambda = (\lambda_1, \ldots, \lambda_n) \subset \mathbb{Z}$ is an (n, κ)-*nice set*, if $0 < \lambda_1 < \cdots < \lambda_i < \cdots < \lambda_n = \mathrm{poly}(\kappa)$. Let S_n be the set of permutations from $[n]$ to $[n]$. For a set 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 $\boldsymbol{a}$.

Progression-Free Sets. A set $\Lambda = \lambda_1, \ldots, \lambda_n$ of positive integers is *progression-free* [TV06], 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 [Elk10] showed that $r_3(n) = \Omega((n \cdot \log_2^{1/4} n)/2^{2\sqrt{2\log_2 n}})$. On the other hand, it is known from [San10] that $r_3(n) = O(n(\log\log n)^5/\log n)$. Thus, according to Sanders [San10], the minimal y such that $r_3(y) = n$ is $\omega(n)$, while according to Elkin, $y = n^{1+o(1)}$. According to [Lip11], for any fixed $n > 0$, there exists $y = n^{1+o(1)}$, such that y contains a progression-free subset Λ of odd integers, with $|\Lambda| = n$.

Now, while the efficiency of arguments from [Lip11] directly depend 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, like that defined by Erdős and Turán [ET36]. This set is defined to be the set $T(n)$ of all integers up to n that have no number 2 in their ternary presentation. Clearly, $|T(n)| \approx n^{\log_3 2} \approx n^{0.63}$. One can then obtain a dense set of progression-free odd integers by mapping every a in $T(n)$ to $2a + 1$.

Bilinear groups. Let $\mathcal{G}_{\mathsf{bp}}(1^\kappa)$ be a bilinear group generator that outputs a description of a bilinear group $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$ 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, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ 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) $\forall a, b \in \mathbb{Z}$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$, (b) $e(g_1, g_2)$ 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 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 320/160/1920 bits, by using an optimal (asymmetric) Ate pairing [HSV06] over a subclass of Barreto-Naehrig curves [BN05,PSNB10].

Decisional Linear Assumption (DLIN). We say that a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is DLIN (decisional linear) secure [BBS04] in group $\mathbb{G}_t$, for $t \in \{1, 2\}$, if for all non-uniform polynomial time adversaries $\mathcal{A}$,

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

is negligible in κ.

Λ-Power Symmetric Discrete Logarithm Assumption. Let Λ be an (n, κ)-nice set for some $n = \mathrm{poly}(\kappa)$. We say that a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is *Λ-PSDL secure* [Lip11], if for any non-uniform probabilistic polynomial-time adversary $\mathcal{A}$,

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

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 [GJM02]. Lipmaa [Lip11] proved that the Λ-PSDL assumption holds in the generic group model for any (n, κ)-nice set 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.

Non-Interactive Zero-Knowledge for Group-Specific Languages. Let $\mathcal{G}_{\mathsf{bp}}$ be a bilinear group generator, and let $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa) = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ be a bilinear group. 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) **NP**-language. Shuffle (see Sect. 5 for its formal definition) has a natural a group-specific language, since one proves a relation between elements of the same bilinear group.

A *non-interactive argument* for R consists of the next probabilistic polynomial-time 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 $\mathcal{V}$. In the arguments to come, crs is considerably longer than crs_v. In the case where efficiency is not important (like in the security definitions), we give the entire crs to $\mathcal{V}$.

A non-interactive 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 $(\mathsf{gk}; C, w) \in R$,

$$\mathcal{V}(\mathsf{gk}, \mathsf{crs}; C, \mathcal{P}(\mathsf{gk}, \mathsf{crs}; C, w)) = 1 \ .$$

A non-interactive argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *adaptively computationally sound*, if for all non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$, the probability

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

is negligible in κ. The soundness is adaptive in the sense that the adversary sees the CRS before producing the statement C.

A non-interactive argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly witness-indistinguishable*, if it is impossible to tell which witness was used by the prover. That is, if $\mathsf{gk} \in \mathcal{G}_{\mathsf{bp}}(1^\kappa)$, $\mathsf{crs} \in \mathcal{G}_{\mathsf{crs}}(\mathsf{gk})$ and $((\mathsf{gk}; C, w_0), (\mathsf{gk}; C, w_1)) \in R^2$, then $\mathcal{P}(\mathsf{gk}, \mathsf{crs}; C, w_0) = \mathcal{P}(\mathsf{gk}, \mathsf{crs}; C, w_0)$ as a distribution.

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

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

Here, td is the *simulation trapdoor*.

Λ-Power Knowledge of Exponent Assumption (Λ-PKE). Abe and Fehr showed in [AF07] that no statistically zero-knowledge non-interactive argument for an **NP**-complete language can have a "direct black-box" security reduction to a standard cryptographic assumption unless $\mathbf{NP} \subseteq \mathbf{P}/\mathrm{poly}$. See also [GW11]. In fact, the soundness of NIZK arguments (for example, of the argument that a computationally binding commitment scheme commits to 0) seems to be an unfalsifiable assumption in general.

Similarly to [Gro10,Lip11], we will base the soundness of our NIZK arguments on Λ-PKE, an explicit knowledge assumption. This assumption, proposed by Groth [Gro10] for $\Lambda = [n]$, and then generalized to arbitrary Λ by Lipmaa in [Lip11], is a generalization of the KE assumption of Damgård [Dam91] and of the KEA3 assumption of Bellare and Palacio [BP04a].

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

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

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 $(g_t^{x^i}, g_t^{\alpha x^i})$ for $i \in \{0\} \cup \Lambda$.

Groth [Gro10] proved that the Λ-PKE assumption holds in the generic group model in the case $\lambda_i = i$; his proof can be straightforwardly modified to the general case. 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 original KE assumption [Dam91], except that it is made in a bilinear group setting.

Trapdoor Commitment Schemes in the CRS Model. A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ in a bilinear group consists of two probabilistic polynomial-time algorithms: a randomized CRS generation algorithm $\mathcal{G}_{\mathsf{com}}$, and a randomized commitment algorithm $\mathcal{C}\mathrm{om}$. Here, $\mathcal{G}^t_{\mathsf{com}}(1^\kappa)$, $t \in \{1,2\}$, produces a CRS ck_t, and $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r)$ outputs a commitment value $A \in \mathbb{G}_t$. A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathrm{om})$ is *computationally binding in group* $\mathbb{G}_t$, if for every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$, the probability

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

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 $\mathsf{ck}_t \in \mathcal{G}^t_{\mathsf{com}}(1^\kappa)$ 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 next variant of the *knowledge commitment scheme* from [Gro10] as modified by Lipmaa [Lip11]:

CRS generation $\mathcal{G}^t_{\mathsf{com}}(1^\kappa)$: Let Λ be an (n, κ)-nice set with $n = \mathrm{poly}(\kappa)$. Define $\lambda_0 = 0$. Given a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$, set $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. 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 $\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_p^n$ in group $\mathbb{G}_t$, the committing party chooses a random $r \leftarrow \mathbb{Z}_p$, and defines

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

Thus, for $f(x) := r + \sum_{i=1}^n a_i x^{\lambda_i}$, $\mathcal{C}\mathrm{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r) = (g_t^{f(x)}, g_t^{\alpha f(x)})$.

As shown in [Lip11], the knowledge commitment scheme in group $\mathbb{G}_t$ is statistically 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 probabilistic polynomial-time committer $\mathcal{A}$ there exists a non-uniform probabilistic polynomial-time extractor $X_{\mathcal{A}}$ that, given as an input the input of $\mathcal{A}$ together with $\mathcal{A}$'s random coins, extracts the contents of the commitments.

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

Adaptive R_{guilt}-Soundness. To avoid knowledge assumptions, one can alternatively relax the notion of soundness. Following [GOS06] (the full version), R_{guilt}-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_{guilt} such that $(\mathsf{gk}; C, w_{\mathsf{guilt}}) \in R_{\mathsf{guilt}}$ (see [GL07] or [GOS11] 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 $R_{\mathsf{guilt}} = \{(\mathsf{gk}; C, w_{\mathsf{guilt}})\}$ be an efficiently computable binary relation. A non-interactive argument $(\mathcal{G}_{\mathsf{bp}}, \mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is (adaptively) R_{guilt}-*sound*, if for all non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$, the probability

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

is negligible in κ.

For the notion of R_{guilt}-soundness to make sense for relation R, one obviously has to define R_{guilt} very carefully. As it comes out, the use of knowledge commitments, while making arguments more efficient, makes the definition of R_{guilt} more tricky. To understand the difficulty, note that the last property of the knowledge commitment scheme as defined on page 6 (under the PKE assumption, for every adversarial

committer $\mathcal{A}$ there exists an extractor $X_{\mathcal{A}}$ that can open the commitment, given access to $\mathcal{A}$'s secret coins and inputs) is needed to prove the soundness of NIZK arguments that are based on this commitment scheme. If we want to prove R_{guilt}-soundness (so that we will not have to rely on a knowledge assumption), we have to modify R_{guilt} so that the R_{guilt}-witness contains openings corresponding to all used knowledge pseudo-commitments. Here, (A, A^{α}) is a knowledge pseudo-commitment if and only if secret $\alpha \leftarrow \mathbb{Z}_p$ was used while creating some elements $(g_t^{\alpha x^{\lambda_i}})_{i \in \Lambda'}$ belonging to the common reference string, where $\Lambda' \subset \mathbb{Z}$. The opening of such a pseudo-commitment is defined as in the case of the PKE assumption.

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

Groth-Lipmaa Arguments. In [Gro10], Groth proposed several 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 [Lip11], who also showed that one can use somewhat weaker security assumptions (PSDL instead of PCDH).

Groth [Gro10] and Lipmaa [Lip11] 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 $|\Lambda_2| = n^{1+o(1)}$. Clearly, together with a trivial Hadamard addition argument, one obtains a complete set of arguments that can be used to construct NIZK arguments for any NP language. (See [Gro10,Lip11] 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.

Public-key cryptosystem. A public-key cryptosystem $(\mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{D}\mathsf{ec})$ is a tuple of efficient algorithms, where $\mathcal{G}_{\mathsf{pkc}}(1^{\kappa})$ 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 $(\mathsf{sk}, \mathsf{pk}) \in \mathcal{G}_{\mathsf{pkc}}(1^{\kappa})$ and for all valid μ and r, $\mathcal{D}\mathsf{ec}_{\mathsf{sk}}(\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu; r)) = \mu$. Since we work in a bilinear group, we assume that $\mathcal{G}_{\mathsf{bp}}$ is first used to produce gk, and then $\mathcal{G}_{\mathsf{pkc}}$ gets gk as an input.

Assume that the randomizer space $\mathcal{R}$ is efficiently sampleable. A public-key cryptosystem $(\mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{D}\mathsf{ec})$ is *CPA-secure*, if for all stateful non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$,

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

is negligible in κ.

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.

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 $\mathbf{0} = (0, \ldots, 0)$. Alternatively, one aims to prove the knowledge of the discrete logarithm of A_t, that is, that $A_t = g_t^r$ for some 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.

The idea of this argument can be derived from [Gro10,Lip11]. 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.)

CRS generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. 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 $\mathring{A}_2 \leftarrow \mathring{g}_2^r$. The prover sends $\pi \leftarrow \mathring{A}_2 \in \mathbb{G}_2$ to the verifier as the argument.

Verification $\mathcal{V}_0(\mathsf{gk}, \mathsf{crs}_v; A_2, \pi = \mathring{A}_2)$**:** The verifier accepts if $e(\mathring{g}_1, A_2) = e(g_1, \mathring{A}_2)$.

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

The next theorem is basically a tautology, since the KE assumption states that the prover knows r. However, since any $(A_2, \mathring{A}_2)$ is a commitment of $\mathbf{0}$ (and thus, $(\mathsf{gk}; A_2, \mathring{A}_2) \in L$) for *some* r, we cannot claim that Prot. 1 is computationally sound (even under a knowledge assumption). Therefore, we prove R^0_{guilt}-soundness of this argument, where

$$R^0_{\mathsf{guilt}} := \left\{ \begin{array}{l} (\mathsf{gk}, \mathsf{crs}; A_2, w^0_{\mathsf{guilt}} = (\boldsymbol{a}, r), \mathring{A}_2) : (A_2, \mathring{A}_2) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{a}; r) \wedge \\ (\boldsymbol{a} \neq \mathbf{0}) \wedge \mathcal{V}_0(\mathsf{gk}, \mathsf{crs}; A_2, \mathring{A}_2) = 1 \end{array} \right\} \ .$$

Theorem 1. *The non-interactive zero argument in Prot. 1 is perfectly complete, perfectly zero-knowledge and adaptively* R^0_{guilt}*-sound.*

Proof (Sketch). Perfect completeness is straightforward. Perfect zero-knowledge: since the adversary knows $\mathring{\alpha}$, it can compute $\mathring{A}_2 \leftarrow A_2^{\mathring{\alpha}}$.

Adaptive R^0_{guilt}-soundness: assume that there exists an adversary $\mathcal{A}$ that can break the R^0_{guilt}-soundness of this argument. That is, $\mathcal{A}$ can create $(A_2, (\boldsymbol{a}, r), \mathring{A}_2)$ such that $(A_2, \mathring{A}_2) = \mathcal{C}\mathrm{om}^2(\boldsymbol{a}; r)$, $\boldsymbol{a} \neq \mathbf{0}$, and $e(\mathring{g}_1, A_2) = e(g_1, \mathring{A}_2)$. 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 $\mathring{g}_2^{x_i}$ only for $i \in \{0\}$, 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 is unconditionally adaptively R^0_{guilt}-sound. □

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.*

Proof. Straightforward. □

3.2 New Sparsity Argument

Assume that $A_2 \in \mathbb{G}_2$. In a *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 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 $i \neq j$. A new sparsity argument is depicted by Prot. 2. Sparsity argument in $\mathbb{G}_1$ is defined dually.

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, we prove R^{spa}_{guilt}-soundness, where

$$R^{spa}_{\mathsf{guilt}} := \left\{ \begin{array}{l} (\mathsf{gk}, \mathsf{crs}; (A_2, \bar{A}_2), ((a_i)_{i \in \Lambda}, r, (f'_i)_{i \in \bar{\Lambda}}), \pi^{spa} = (A_1, \bar{A}_1, F, \bar{F})) : \\ (A_2, \bar{A}_2) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{a}; r) \wedge (\exists i, j \in [n])(i \neq j \wedge a_i a_j \neq 0) \wedge \\ \log_{g_2} F = \sum_{i \in \bar{\Lambda}} f'_i x^i \wedge \mathcal{V}_{spa}(\mathsf{gk}, \mathsf{crs}; (A_2, \bar{A}_2), \pi^{spa}) = 1 \end{array} \right\} \ .$$

Here, $\bar{\Lambda}$ is as defined in Prot. 2.

We note that the definition of R^{spa}_{guilt} is quite natural, except the part of f'_i that is needed because of the concrete proof technique. Note that both [Gro10] and [Lip11] prove the culpable soundness of their basic arguments with respect to similar languages. Intuitively, R^{spa}_{guilt} includes only f'_i for $i \in \bar{\Lambda}$ (resp., a_i for $i \in \Lambda$) since $\bar{g}_{2i}$ (resp., $\bar{g}_{1i}$) belongs to the CRS only for $i \in \bar{\Lambda}$ (resp., $i \in \Lambda$).

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

CRS generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $\bar{\alpha}, x \leftarrow \mathbb{Z}_p$. Denote $\bar{g}_t \leftarrow g_t^{\bar{\alpha}}$, $g_{ti} \leftarrow g_t^{x^i}$ and $\bar{g}_{ti} \leftarrow g_t^{\bar{\alpha}x^i}$ for $t \in \{1, 2\}$ and $i \in \bar{\Lambda}$. The CRS is

$$\mathsf{crs} \leftarrow (\bar{g}_1, \bar{g}_2, (g_{1i}, \bar{g}_{1i})_{i\in\Lambda}, (g_{2i}, \bar{g}_{2i})_{i\in\Lambda\cup(2\cdot\Lambda)}) \ .$$

Set $\mathsf{ck}_t \leftarrow (\mathsf{gk}; \bar{g}_t, (g_{ti}, \bar{g}_{ti})_{i\in\Lambda})$ for $t \in \{1, 2\}$, and let $\mathsf{crs}_v \leftarrow (\bar{g}_1, \bar{g}_2)$ be the verifier's part of crs.

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

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

Protocol 2: New sparsity argument

Theorem 2. *The sparsity argument in Prot. 2 is perfectly complete and perfectly witness-indistinguishable. Let Λ be a progression-free set of odd integers. If the bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is $\bar{\Lambda}$-PSDL secure, then Prot. 2 is adaptively R^{spa}_{guilt}-sound.*

Proof. Let $\eta \leftarrow e(A_1, A_2)$ and $h \leftarrow e(g_1, g_2)$. Witness-Indistinguishability: follows from the fact that there is exactly one possible argument π^{spa} that satisfies the verification equations.

Perfect completeness. All verifications but the last one are straightforward. For the last verification $e(A_1, A_2) = e(g_1, F)$, note that

$$\log_h \eta = (r + \sum_{i=1}^{n} a_i x^{\lambda_i})(r + \sum_{j=1}^{n} a_j x^{\lambda_j}) = \underbrace{\sum_{i=1}^{n} \sum_{j=1:j\neq i}^{n} a_i a_j x^{\lambda_i+\lambda_j}}_{\delta\in 2\Lambda\setminus 2\cdot\Lambda} + \underbrace{r^2 + 2r\sum_{i=1}^{n} a_i x^{\lambda_i} + \sum_{i=1}^{n} a_i^2 x^{2\lambda_i}}_{\delta\in\bar{\Lambda}} \ .$$

Thus, $\log_h \eta$ is equal to a sum of some powers x^δ of x for $\delta \in 2\Lambda \setminus 2 \cdot \Lambda$ and $\delta \in \bar{\Lambda}$. 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 $\delta \in \bar{\Lambda}$. Since then $a_i = 0$ for $i \neq I$, we get that $\log_h \eta = r^2 + 2ra_I x^{\lambda_I} + a_I^2 x^{2\lambda_I} = \log_{g_2} F$. Thus, if the prover is honest, then the third verification succeeds.

Adaptive R^{spa}_{guilt}-soundness: Assume that $\mathcal{A}$ is an adversary that can break the adaptive R^{spa}_{guilt}-soundness of the sparsity argument. Next, we construct an adversary $\mathcal{A}'$ against the $\bar{\Lambda}$-PSDL assumption. Let $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$ and $x \leftarrow \mathbb{Z}_p$. The adversary $\mathcal{A}'$ receives $\mathsf{crs} \leftarrow (\mathsf{gk}; (g_1^{x^i}, g_2^{x^i})_{i\in\bar{\Lambda}})$ as her input, and her task is to output x. She sets $\bar{\alpha} \leftarrow \mathbb{Z}_p$, $\mathsf{crs}' \leftarrow (\bar{g}_1, \bar{g}_2, (g_1^{x^i}, g_1^{\bar{\alpha}x^i})_{i\in\Lambda}, (g_2^{x^i}, g_2^{\bar{\alpha}x^i})_{i\in\Lambda\cup(2\cdot\Lambda)})$, and then forwards crs' to $\mathcal{A}$. Clearly, crs' follows the distribution imposed by $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$. Both $\mathcal{A}$ and $\mathcal{A}'$ set $\mathsf{ck}_t \leftarrow (\mathsf{gk}; \bar{g}_t, (g_t^{x^i}, g_t^{\bar{\alpha}x^i})_{i\in\Lambda})$ for $t \in \{1, 2\}$. According to the definition of R^{spa}_{guilt}, $\mathcal{A}(\mathsf{gk}; \mathsf{crs}')$ returns

$$((A_2, \bar{A}_2), w^{spa}_{\mathsf{guilt}} = ((a_i)_{i\in\Lambda}, r, (f'_i)_{i\in\bar{\Lambda}}), \pi^{spa} = (A_1, \bar{A}_1, F, \bar{F})) \ .$$

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 $i \in 2\Lambda \setminus 2 \cdot \Lambda$, x^i has a non-zero coefficient.

On the other hand, by the definition of the language R^{spa}_{guilt}, $\log_{g_2} F = \sum_{i\in\bar{\Lambda}} f'_i x^i = f'(x)$. Since Λ is a progression-free set of odd integers, then $(2\Lambda\setminus 2\cdot\Lambda) \cap \bar{\Lambda} = \emptyset$ and thus if $i \in \bar{\Lambda}$ then $i \notin 2\Lambda\setminus 2\cdot\Lambda$. Therefore,

all coefficients of $f'(x)$ corresponding to any x^i, $i \in 2\Lambda \setminus 2 \cdot \Lambda$, are equal to 0. Thus $f(X) = \sum f_i X^i$ and $f'(X) = \sum_{i \in \bar{\Lambda}} f'_i X^i$ are different polynomials with $f(x) = f'(x) = \log_{g_2} F$. Therefore, $\mathcal{A}'$ has succeeded in creating a non-zero polynomial $d = f - f'$, such that $d(x) = \sum_{i \in \bar{\Lambda}} d_i x^i = 0$.

Next, $\mathcal{A}'$ can use an efficient polynomial factorization [vHN10] algorithm in $\mathbb{Z}_p[X]$ to efficiently compute all $2\lambda_n + 1$ roots of $d(x)$. For some root y, $g_1^{x^i} = g_1^{y^i}$. $\mathcal{A}'$ sets $x \leftarrow y$, thus violating the $\bar{\Lambda}$-PSDL assumption. □

Note that the efficiency of the culpable soundness reduction depends on $2\lambda_n$, the largest element of $\bar{\Lambda}$.

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.*

Proof. It is clear that the size of the CRS is $\Theta(\sharp\bar{\Lambda}) = \Theta(n)$. The statement about the length of the common reference string follows from the fact that $\Lambda \cap (2 \cdot \Lambda) = \emptyset$. From the CRS, the verifier clearly only needs to access $\bar{g}_1$ and $\bar{g}_2$. The statements about prover's and verifier's computational complexity follow straightforwardly. □

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 $(c_{2i}, \bar{c}_{2i}) \in \mathbb{G}_2^2$ are knowledge commitments to P's rows. We assume that 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 next lemma.

Lemma 2. *An* $n \times n$ *matrix* P *is a permutation matrix if and only if the next 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 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.

Similarly to the case of the zero and sparsity arguments, we prove that Prot. 3 is R^{pm}_{guilt}-sound, where R^{pm}_{guilt} is equal to

$$\left\{ \begin{array}{l} \big(\mathsf{gk}, \mathsf{crs}; (\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}}), ((a_i)_{i\in\Lambda}, r_a, (\boldsymbol{P_i}, r_i, (f'_{ij})_{j\in\bar{\Lambda}})_{i\in[n]}), \pi^{pm} = (\pi^0, (c_{1i}, \bar{c}_{1i}, F_i, \bar{F}_i)_{i\in[n]} \quad) : \\ \left(\left(\prod_{i=1}^{n} c_{2i}\right)/D, \pi^0\right) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{a}; r_a) \wedge (\forall i \in [n])(c_{2i}, \bar{c}_{2i}) = \mathcal{C}\mathrm{om}^2(\mathsf{ck}_2; \boldsymbol{P_i}; r_i) \wedge \\ (\forall i \in [n]) \log_{g_2} F_i = \sum_{j\in\bar{\Lambda}} f'_{ij} x^j \wedge (\boldsymbol{a} \neq \boldsymbol{0} \vee (\exists i \in [n]) \boldsymbol{P_i} \text{ is not sparse}) \wedge \\ \mathcal{V}_{pm}(\mathsf{gk}, \mathsf{crs}; (\boldsymbol{c_2}, \hat{\boldsymbol{c}}_{\boldsymbol{2}}), \pi^{pm}) = 1 \end{array} \right\}.$$

Here, D is as defined in Prot. 3.

Again, while the definition of R^{pm}_{guilt} is argument-dependent (in particular, it gives us a clue that the permutation matrix argument consists of a zero argument and of a sparsity argument), this definition is also quite natural. Namely, if we make the Λ-PKE assumption, we will have an extractor that—given access to the common input and to the adversary's random coins—returns the witness $((a_i)_{i\in\Lambda}, r_a, (P_i, r_i, (f'_{ij})_{j\in\bar{\Lambda}})_{i\in[n]})$.

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

Common reference string $\mathcal{G}_{\mathsf{crs}}(\mathsf{gk})$**:** Let $\bar{\alpha}, \mathring{\alpha}, x \leftarrow \mathbb{Z}_p$, $\bar{g}_t \leftarrow g_t^{\bar{\alpha}}$, $\mathring{g}_t \leftarrow g_t^{\mathring{\alpha}}$, $g_{ti} \leftarrow g_t^{x^i}$, and $\bar{g}_{ti} \leftarrow \bar{g}_t^{x^i}$. Let $D \leftarrow \prod_{i=1}^n g_{2,\lambda_i}$ and $\mathring{D} \leftarrow \prod_{i=1}^n \mathring{g}_{2,\lambda_i}$. Let

$$\mathsf{crs} \leftarrow (\bar{g}_1, \bar{g}_2, \mathring{g}_1, \mathring{g}_2, (g_{1i}, \bar{g}_{1i})_{i\in\Lambda}, (g_{2i}, \bar{g}_{2i})_{i\in\Lambda\cup(2\cdot\Lambda)}, (\mathring{g}_{2i})_{i\in\Lambda}, D, \mathring{D}) \ ,$$

$\mathsf{ck}_t = (\mathsf{gk}; \bar{g}_t, (g_{ti}, \bar{g}_{ti})_{i\in\Lambda})$, $\mathring{\mathsf{ck}}_2 = (\mathsf{gk}; g_2, \mathring{g}_2)$, and $\mathsf{crs}_v = (\bar{g}_1, \bar{g}_2, \mathring{g}_1, \mathring{g}_2)$.

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 $\mathcal{P}_{pm}(\mathsf{gk}, \mathsf{crs}; (\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}}), (P, \boldsymbol{r}))$**:** Let $\mathring{c}_{2i} = \mathring{g}_2^{r_i} \cdot \mathring{g}_{2,\lambda_{\psi(i)}}$ for $i \in [n]$. Construct a zero argument $\pi^0 = (\prod_{i=1}^n \mathring{c}_{2i})/\mathring{D}$ for $(\prod_{i=1}^n c_{2i})/D$. For $i \in [n]$, construct a sparsity argument $\pi_i^{spa} = (c_{1i}, \bar{c}_{1i}, F_i, \bar{F}_i)$ that $(c_{2i}, \bar{c}_{2i})$ commits to a 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 $P = P_\psi$

Theorem 4. *The argument in Prot. 3 is a perfectly complete and perfectly zero-knowledge permutation matrix argument. Let Λ be a progression-free set of odd integers. If the $\bar{\Lambda}$-PSDL assumption holds, then Prot. 3 is adaptively R^{pm}_{guilt}-sound.*

Proof. Perfect Completeness: follows from the completeness of the sparsity and zero arguments and from Lem. 2. For this we note that $(D, \mathring{D}) = \mathcal{C}\mathrm{om}^2(\mathring{\mathsf{ck}}_2; \mathbf{1}; 0)$ and thus $(\prod_{i=1}^n c_{2i}/D, \prod_{i=1}^n \mathring{c}_{2i}/\mathring{D})$ commits to $\mathbf{0}$ if and only if every column of P sums to 1.

Adaptive R^{pm}_{guilt}-Soundness: Let $\mathcal{A}$ be a non-uniform probabilistic polynomial-time adversary that creates $(\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}})$, witness $((a_i)_{i\in\Lambda}, r_a, (P_i, r_i, (f'_{ij})_{j\in\bar{\Lambda}})_{i\in[n]})$, and an accepting NIZK argument π.

Since the zero argument is R^0_{guilt}-sound, verification of the argument π^0 shows that every column of P sums to 1. Here the witness is $w^0_{\mathsf{guilt}} = (\boldsymbol{a}, r_a)$ with $\boldsymbol{a} = \sum_{i=1}^n \boldsymbol{P_i} - \mathbf{1}$. By the $\bar{\Lambda}$-PSDL assumption, the sparsity assumption is R^{spa}_{guilt}-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 culpable soundness of the sparsity and zero arguments, P is a permutation matrix.

Perfect Zero-Knowledge: the simulator creates a correctly formed CRS together with a simulation trapdoor, so that the trapdoor commitments can be opened to any values. Due to the trapdoor, the simulator can simulate the zero and sparsity arguments. More precisely, he uses P equal to the identity matrix to simulate both the zero and the sparsity arguments. To show that this argument π'' simulates the real argument π^{pm}, note that π^{pm} is perfectly indistinguishable from the simulated NIZK argument π' where one makes trapdoor commitments but opens them to *real* witnesses when making product and permutation arguments. On the other hand, also π' and π'' are perfectly indistinguishable, and thus so are π^{pm} and π''. □

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.*

Proof. Straightforward. For example, in a single sparsity argument, the prover's (resp., the verifier's) computational complexity is dominated by 10 exponentiations (resp., pairings). In the case of the zero argument, the prover's (resp., the verifier's) computational complexity is dominated by 1 exponentiation (resp., 2 pairings). □

4 Knowledge BBS Cryptosystem

Boneh, Boyen and Shacham [BBS04] proposed a cryptosystem $\Pi = (\mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{D}\mathsf{ec})$ that we will call the BBS cryptosystem. We will use a "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 $\mathsf{gk} \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$.

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 $u = (u_1, u_2, u_3, \tilde u_1, \tilde u_2, \tilde u_3)$, where $u_1 = f^\sigma$, $u_2 = h^\tau$, $u_3 = g_1^{\mu+\sigma+\tau}$, $\tilde u_1 = \tilde f^\sigma$, and $\tilde u_2 = \tilde h^\tau$, and $\tilde u_3 = \tilde g_1^{\mu+\sigma+\tau}$.

Decryption $\mathcal{D}\mathsf{ec}_{\mathsf{sk}}(u_1, u_2, u_3, \tilde u_1, \tilde u_2, \tilde u_3)$**:** if $e(u_1, \tilde g_2^{(1)}) = e(\tilde u_1, g_2)$, $e(u_2, \tilde g_2^{(2)}) = e(\tilde u_2, g_2)$ and $e(u_3, \tilde g_2^{(3)}) = e(\tilde u_3, g_2)$, then return the discrete logarithm of $g_1^\mu \leftarrow u_3/(u_1^{\mathsf{sk}_1} u_2^{\mathsf{sk}_2})$. 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.

We need the cryptosystem to be lifted (that is, 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 is assumed to be 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 either

- use a range proof [Bou00,LAN02,Lip03,CCS08,RKP09,CLS10] (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 votes are correctly formed. This approach is prudent in case case, since invalid ballots can be removed from the system before starting to shuffle (saving thus valuable time otherwise wasted to shuffle invalid ciphertexts), or
- discard the ballots if the ciphertext does not decrypt.

Both approaches have their benefits, and either one can be used depending on the application.

The inclusion of $\tilde u_3$ 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. Therefore, security would not be guaranteed against an adversary who chooses u_3 without actually knowing the element μ.

PRA1-Security. It is easy to see that the knowledge BBS cryptosystem, like the original BBS cryptosystem, is CPA-secure under the DLIN assumption. Moreover, under a knowledge (KE, that is, $\emptyset$-PKE) assumption the encrypting party knows the tuple (μ, σ, τ): there exists an extractor that returns σ (resp., τ or μ), given access to $(u_1, \tilde u_1)$ (resp., $(u_2, \tilde u_2)$ or $(u_3, \tilde u_3)$) and the encrypter's random coins. For this it is required that values $\tilde\alpha_1$, $\tilde\alpha_2$ and $\tilde\alpha_3$ are chosen independently at random.

Thus, under the KE assumption in group $\mathbb{G}_t$, the knowledge-BBS cryptosystem satisfies *PRA1-security*, a version of plaintext-awareness (more precisely, PA1-security as defined in [BP04b]), where the extractor extracts both the plaintext and the randomizer. We will not give a formal definition of the PRA1-security in this paper.

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 u_i and u_i' are valid ciphertexts of Π. We say that $(u_1', \ldots, u_n')$ is a *shuffle* of $(u_1, \ldots, u_n)$ iff there exists a permutation $\psi \in S_n$ and randomizers $r_1, \ldots, r_n$ such that $u_i' = u_{\psi(i)} \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(0; r_i)$ 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 $(\mathsf{pk}, (u_i, u_i')_{i\in[n]})$, he knows a permutation $\psi \in S_n$ and randomizers r_i such that $u_i' = u_{\psi(i)} \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(0; r_i)$ for $i \in [n]$.

In this section, we construct an efficient shuffle argument that works with the knowledge BBS cryptosystem of Sect. 4; the argument can be generalized to other scenarios. Assume that the ciphertexts

Common reference string: Similarly to the permutation matrix argument, let $\bar{\alpha}, \mathring{\alpha}, x \leftarrow \mathbb{Z}_p$, $\bar{g}_t \leftarrow g_t^{\bar{\alpha}}$, $\mathring{g}_t \leftarrow g_t^{\mathring{\alpha}}$, $g_{ti} \leftarrow g_t^{x^i}$, and $\bar{g}_{ti} \leftarrow \bar{g}_t^{x^i}$. Let $D \leftarrow \prod_{i=1}^n g_{2,\lambda_i}$ and $\mathring{D} \leftarrow \prod_{i=1}^n \mathring{g}_{2,\lambda_i}$.
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

$$\mathsf{crs} := (\bar{g}_1, \bar{g}_2, \mathring{g}_1, \mathring{g}_2, (g_{1i}, \bar{g}_{1i})_{i\in\Lambda}, (g_{2i}, \bar{g}_{2i})_{i\in\Lambda\cup(2\cdot\Lambda)}, (\mathring{g}_{2i})_{i\in\Lambda}, D, \mathring{D}, \tilde{g}_1, \tilde{g}_2^{(1)}, \tilde{g}_2^{(3)}, \tilde{g}_2^{(2)}, f, \tilde{f}, h, \tilde{h}) \ .$$

The commitment keys are $\mathsf{ck}_t \leftarrow (\mathsf{gk}; \bar{g}_t, (g_{ti}, \bar{g}_{ti})_{i\in\Lambda})$ 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: $(\mathsf{pk}, (u_i, u_i')_{i\in[n]})$, where $u_i = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_i; \sigma_i, \tau_i)$ and $u_i' = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_{\psi(i)}; \sigma_{\psi(i)} + \sigma_i', \tau_{\psi(i)} + \tau_i')$.
Argument $\mathcal{P}_{sh}(\mathsf{gk}, \mathsf{crs}; (\mathsf{pk}, (u_i, u_i')_{i\in[n]}), (\psi, (\sigma_i', \tau_i')_{i\in[n]}))$**:** 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 $(\boldsymbol{c_2}, \bar{\boldsymbol{c}}_{\boldsymbol{2}})$.
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 $(u_\sigma, \tilde{u}_\sigma) \leftarrow (f^{R_\sigma} \cdot \prod_{i=1}^n u_{i1}^{r_i}, \tilde{f}^{R_\sigma} \cdot \prod_{i=1}^n \tilde{u}_{i1}^{r_i})$, $(u_\tau, \tilde{u}_\tau) \leftarrow (h^{R_\tau} \cdot \prod_{i=1}^n u_{i2}^{r_i}, \tilde{h}^{R_\tau} \cdot \prod_{i=1}^n \tilde{u}_{i2}^{r_i})$, $(u_\mu, \tilde{u}_\mu) \leftarrow (g_1^{R_\sigma+R_\tau} \cdot \prod_{i=1}^n u_{i3}^{r_i}, \tilde{g}_1^{R_\sigma+R_\tau} \cdot \prod_{i=1}^n \tilde{u}_{i3}^{r_i})$.
6. The argument is

$$\pi^{sh} \leftarrow ((c_{2i}, \bar{c}_{2i})_{i\in[n]}, \pi^{pm}, c_\sigma, \bar{c}_\sigma, c_\tau, \bar{c}_\tau, u_\sigma, \tilde{u}_\sigma, u_\tau, \tilde{u}_\tau, u_\mu, \tilde{u}_\mu) \ . \tag{1}$$

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

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

Protocol 4: New shuffle argument

$(u_{i1}, u_{i2}, u_{i3}, \tilde{u}_{i1}, \tilde{u}_{i2}, \tilde{u}_{i3})$, where $i \in [n]$, are created as in Sect. 4. Then, the shuffled ciphertexts with permutation $\psi \in S_n$ and randomizers $(\sigma_i', \tau_i')_{i\in[n]}$ are

$$u_i' = (u_{i1}', u_{i2}', u_{i3}', \tilde{u}_{i1}', \tilde{u}_{i2}', \tilde{u}_{i3}') = u_{\psi(i)} \cdot \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(0; \sigma_i', \tau_i') = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_{\psi(i)}; \sigma_{\psi(i)} + \sigma_i', \tau_{\psi(i)} + \tau_i') \ .$$

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 u_i' is equal to the product of this permutation matrix and the plaintext vector of 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 $(\mathsf{pk}, (u_i, u_i')_{i\in[n]})$ 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.

In a real life application, the adversary (for example, a malicious mix server) can get the ciphertexts 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 mixnet (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 knowledge BBS can be seen as a white-box non-interactive knowledge argument. One could

instead opt to use some somewhat less efficient but black-box non-interactive knowledge argument. In this case, the adversary would not have to have access to voters' secret coins, and one could just use the lifted BBS cryptosystem instead of the knowledge BBS cryptosystem.

We note that Groth and Lu made a similar, though somewhat weaker, assumption in [GL07] where they prove culpable soundness against adversaries who also output and thus know the secret key of the cryptosystem. Thus, the adversary can decrypt all the ciphertexts, and thus knows the plaintexts (but does not have to know the randomizers). As argued in [GL07], this is reasonable in the setting of mixnet where the servers can usually threshold-decrypt all the results.

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).

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. We now have to show that if the prover is honest, then the three last verification equations hold.

Assume that the prover is honest. The verification equation in step 5a holds since

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

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

Adaptive computational soundness: Let $\mathcal{A}$ be a non-uniform probabilistic polynomial-time adversary that, given gk and a crs, creates a statement $(\mathsf{pk} = (\mathsf{gk}; \tilde{g}_1, \tilde{g}_2^{(1)}, \tilde{g}_2^{(2)}, \tilde{g}_2^{(3)}, f, \tilde{f}, h, \tilde{h}), (u_i, u'_i)_{i\in[n]})$ and an accepting NIZK argument π^{sh} (as in Eq. (1) in Prot. 4), such that the plaintext vector $(u'_i)_{i\in[n]}$ is not a permutation of the plaintext vector $(u_i)_{i\in[n]}$.

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 next non-uniform probabilistic polynomial-time knowledge extractors that return certain values:

- By the KE assumption in group $\mathbb{G}_1$, there exists a knowledge extractor that, given $(u_{ij}, \tilde{u}_{ij}, u'_{ij}, \tilde{u}'_{ij})_{j\in[3]}$ and access to $\mathcal{A}$'s random coins, returns with all but a negligible probability values μ_i, σ_i, τ_i, μ'_i, σ'_i and τ'_i, such that $u_i = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu_i; \sigma_i, \tau_i)$ and $u'_i = \mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mu'_i; \sigma'_i, \tau'_i)$. Note that it might be the case that $\mu_i \neq \mu'_i$.
- 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

  $$(\boldsymbol{\sigma}^*, R_\sigma) \text{ and } (\boldsymbol{\tau}^*, R_\tau) \ ,$$

  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 $(u_\sigma, \tilde{u}_\sigma, u_\tau, \tilde{u}_\tau, u_\mu, \tilde{u}_\mu)$ and access to $\mathcal{A}$'s random coins, returns openings

  $$(v_\sigma, v_\tau, v_\mu) \ ,$$

  such that $(u_\sigma, \tilde{u}_\sigma) = (f^{v_\sigma}, \tilde{f}^{v_\sigma})$, $(u_\tau, \tilde{u}_\tau) = (h^{v_\tau}, \tilde{h}^{v_\tau})$, and $(u_\mu, \tilde{u}_\mu) = (g_1^{v_\mu}, \tilde{g}_1^{v_\mu})$. (Thus, it is not necessary that the adversary created the values u_σ, u_τ and u_μ 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

$$((a_i)_{i\in\Lambda}, r_a)$$

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 $(c_{1i}, \bar{c}_{i1})$ and access to $\mathcal{A}$'s random coins, returns an opening

$$((P_{ij})_{j\in\Lambda}, r_i)$$

such that $(c_{1i}, \bar{c}_{i1}) = \mathcal{C}\mathrm{om}^1(\mathsf{ck}_1; \boldsymbol{P_i}; r_i)$.
– 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 $\log_{g_2} F_i = \sum_{j\in\bar{\Lambda}} f'_{ij} x^j$.

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 $((a_i)_{i\in\Lambda}, r_a, (\boldsymbol{P_i}, r_i, (f'_{ij})_{j\in\bar{\Lambda}})_{i\in[n]})$. Since the permutation matrix argument is R^{pm}_{guilt}-sound 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

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

Since $u_\sigma = f^{v_\sigma}$, $\sum_{i=1}^n (\sigma_{\psi(i)} + \sigma_i^* - \sigma'_i) x^{\lambda_i} + R_\sigma + \sum_{i=1}^n \sigma_i r_i - v_\sigma = 0$. 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^i} for $i = 0$, 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,

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

Thus,

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

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 $g_1^{x^i}$ for $i \in \Lambda$, the adversary has thus broken the Λ-PSDL assumption. Therefore, due to the Λ-PSDL assumption, $\mu'_i = \mu_{\psi(i)}$ for $i \in [n]$.[2]

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

Zero-knowledge: The simulator construction 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 u_σ, u_τ and u_μ 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.

Inputs: gk and CRS as in Prot. 4 and $(\mathsf{pk}, (u_i, u'_i)_{i \in [n]})$
Output: π^{sh}
Simulation:

1. Pick random $z_i, r_{i1}, r_{i2} \leftarrow \mathbb{Z}_p$.
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 $\bar{c}_{2i} \leftarrow \bar{g}_2^{z_i}$.
3. Set

$$(u_\sigma, \tilde{u}_\sigma) \leftarrow \left(\prod_{i=1}^n \left(f^{r_{i1}} \cdot u_{i1}^{z_i} \cdot (u'_{i1})^{-x^{\lambda_i}} \right), \prod_{i=1}^n \left(\tilde{f}^{r_{i1}} \cdot \tilde{u}_{i1}^{z_i} \cdot (\tilde{u}'_{i1})^{-x^{\lambda_i}} \right) \right) ,$$

$$(u_\tau, \tilde{u}_\tau) \leftarrow \left(\prod_{i=1}^n \left(h^{r_{i2}} \cdot u_{i2}^{z_i} \cdot (u'_{i2})^{-x^{\lambda_i}} \right), \prod_{i=1}^n \left(\tilde{h}^{r_{i2}} \cdot \tilde{u}_{i2}^{z_i} \cdot (\tilde{u}'_{i2})^{-x^{\lambda_i}} \right) \right) ,$$

$$(u_\mu, \tilde{u}_\mu) \leftarrow \left(\prod_{i=1}^n \left(g_1^{r_{i1}+r_{i2}} \cdot u_{i3}^{z_i} \cdot (u'_{i3})^{-x^{\lambda_i}} \right), \prod_{i=1}^n \left(\tilde{g}_1^{r_{i1}+r_{i2}} \cdot \tilde{u}_{i3}^{z_i} \cdot (\tilde{u}'_{i3})^{-x^{\lambda_i}} \right) \right) .$$

4. Complete the remaining part of the proof and simulate π^{pm} by using the trapdoor opening of commitments.
5. Set $\pi^{sh} \leftarrow ((c_{2i}, \bar{c}_{2i})_{i \in [n]}, \pi^{pm}, c_\sigma, \bar{c}_\sigma, c_\tau, \bar{c}_\tau, u_\sigma, \tilde{u}_\sigma, u_\tau, \tilde{u}_\tau, u_\mu, \tilde{u}_\mu)$.

Protocol 5: Simulator construction

We have

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

Similarly, $e(h, c_\tau) \cdot \prod_{i=1}^n e(u_{i2}, c_{2i}) = e(u_\tau, g_2) \cdot \prod_{i=1}^n e(u'_{i2}, g_{2i})$ and $e(g_1, c_\sigma c_\tau) \cdot \prod_{i=1}^n e(u_{i3}, c_{2i}) = e(u_\mu, g_2) \cdot \prod_{i=1}^n e(u'_{i3}, g_{2i})$. 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 $2n + 7$ elements of $\mathbb{G}_1$ and $5n + 7$ elements of $\mathbb{G}_2$, in total $7n + 14$ 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.*

[2] 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.

Proof. The communication complexity: $|\pi^{pm}|$; in addition, 6 elements from $\mathbb{G}_1$ and $2n+4$ elements from $\mathbb{G}_2$. The prover's computational complexity follows from that of the permutation matrix argument, to which the shuffle argument proper adds $7n + 15$ exponentiations. Finally, the shuffle argument proper adds $18n + 16$ bilinear pairings to the verifier's computational complexity of the permutation matrix argument. The rest is straightforward. □

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 u'_i of one shuffle is equal to the input ciphertext u_i of the next shuffle. Therefore, in step 3, one only has to check the correctness of the ciphertexts 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 authors were supported by Estonian Science Foundation, grant #8058, and European Union through the European Regional Development Fund.

References

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

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short Group Signatures. In Matthew K. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55, Santa Barbara, USA, August 15–19, 2004. Springer-Verlag.

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

[Bou00] Fabrice Boudot. Efficient Proofs That a Committed Number Lies in an Interval. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 431–444, Bruges, Belgium, May 14–18, 2000. Springer-Verlag.

[BP04a] Mihir Bellare and Adriana Palacio. The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols. In Matthew K. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 273–289, Santa Barbara, USA, August 15–19, 2004. Springer-Verlag.

[BP04b] Mihir Bellare and Adriana Palacio. Towards Plaintext-Aware Public-Key Encryption without Random Oracles. In Pil Joong Lee, editor, *ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 48–62, Jeju Island, Korea, December 5-9 2004. Springer-Verlag.

[CCS08] Jan Camenisch, Rafik Chaabouni, and Abhi Shelat. Efficient Protocols for Set Membership and Range Proofs. In Josef Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 234–252, Melbourne, Australia, December 7–11, 2008. Springer-Verlag.

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

[CGS97] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. In Walter Fumy, editor, *EUROCRYPT 1997*, volume 1233 of *LNCS*, pages 103–118, Konstanz, Germany, 11–15 May 1997. Springer-Verlag.

[Cha81] David Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

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

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

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

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

[FS01] Jun Furukawa and Kazue Sako. An Efficient Scheme for Proving a Shuffle. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 368–387, Santa Barbara, USA, 19–23 August 2001. Springer-Verlag.

[GI08] Jens Groth and Yuval Ishai. Sub-linear Zero-Knowledge Argument for Correctness of a Shuffle. In Nigel Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 379–396, Istanbul, Turkey, April 13–17, 2008. Springer-Verlag.

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

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

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

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

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

[Gro03] Jens Groth. A Verifiable Secret Shuffle of Homomorphic Encryptions. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 145–160, Miami, Florida, USA, January 6–8, 2003. Springer-Verlag.

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

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

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

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

[KNP10] Kaoru Kurosawa, Ryo Nojima, and Le Trieu Phong. Efficiency-Improved Fully Simulatable Adaptive OT under the DDH Assumption. In Juan Garay, editor, *SCN 2010*, volume 6280 of *LNCS*, pages 172–181, Amalfi, Italy, September 13–15, 2010. Springer Verlag.

[KNP11] Kaoru Kurosawa, Ryo Nojima, and Le Trieu Phong. Generic Fully Simulatable Adaptive Oblivious Transfer. In Javier Lopez and Gene Tsudik, editors, *ACNS 2011*, volume 6715 of *LNCS*, pages 274–291, Nerja, Spain, June 7–10, 2011. Springer-Verlag.

[LAN02] Helger Lipmaa, N. Asokan, and Valtteri Niemi. Secure Vickrey Auctions without Threshold Trust. In Matt Blaze, editor, *FC 2002*, volume 2357 of *LNCS*, pages 87–101, Southhampton Beach, Bermuda, March 11–14, 2002. Springer-Verlag.

[Lip03] Helger Lipmaa. On Diophantine Complexity and Statistical Zero-Knowledge Arguments. In Chi Sung Laih, editor, *ASIACRYPT 2003*, volume 2894 of *LNCS*, pages 398–415, Taipei, Taiwan, November 30–December 4, 2003. Springer-Verlag.

[Lip11] Helger Lipmaa. Progression-Free Sets and Sublinear Pairing-Based Non-Interactive Zero-Knowledge Arguments. Technical Report 2011/009, International Association for Cryptologic Research, January 5, 2011. Available at http://eprint.iacr.org/2011/009.

[Nef01] C. Andrew Neff. A Verifiable Secret Shuffle and Its Application to E-Voting. In *ACM CCS 2001*, pages 116–125, Philadelphia, Pennsylvania, USA, November 6–8 2001. ACM Press.

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

[RKP09] Alfredo Rial, Markulf Kohlweiss, and Bart Preneel. Universally Composable Adaptive Priced Oblivious Transfer. In Hovav Shacham and Brent Waters, editors, *Pairing 2009*, volume 5671 of *LNCS*, pages 231–247, Palo Alto, CA, USA, August 12–14, 2009. Springer-Verlag.

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

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

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

[TW10] Björn Terelius and Douglas Wikström. Proofs of Restricted Shuffles. In Daniel J. Bernstein and Tanja Lange, editors, *AFRICACRYPT 2010*, volume 6055 of *LNCS*, pages 100–113, Stellenbosch, South Africa, May 3–6, 2010. Springer-Verlag.

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

A PPA Assumption from [GL07]

Purely for comparison reasons, we will state next the definition of the permutation pairing assumption from [GL07]. Since the original definition was given in the symmetric setting, we assume here that $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$, and $g = g_1 = g_2$.

Definition 1 (PPA Assumption [GL07]). *The permutation pairing assumption holds for $\mathcal{G}_{\mathsf{bp}}$, if for all non-uniform probabilistic polynomial-time adversaries, the probability*

$$\Pr\left[\begin{array}{l}\mathsf{gk} := (p, \mathbb{G}, \mathbb{G}, \mathbb{G}_T, g, g) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa), (x_1, \ldots, x_n) \leftarrow \mathbb{Z}_p^n, (a_i, b_i)_{i\in[n]} \leftarrow \mathcal{A}(\mathsf{gk}, (g^{x_i}, g^{x_i^2})_{i\in[n]}) : \\ \prod_{i=1}^{n} a_i g^{-x_i} = 1 \wedge \prod_{i=1}^{n} b_i g^{-x_i^2} = 1 \wedge (\forall i \in [n])\, e(a_i, a_i) = e(g, b_i) \wedge \\ (a_i)_{i\in[n]} \textit{ is not a permutation of } (g^{x_i})_{i\in[n]}\end{array}\right]$$

is negligible in κ.