A Shuffle Argument Secure in the Generic Model

Prastudy Fauzi, Helger Lipmaa, and Michał Zając

University of Tartu, Estonia

Abstract. Implementation details of the Asiacrypt 2016 paper

1 Preliminaries

Let S_n be the symmetric group on n elements. For a (Laurent) polynomial or a rational function f and its monomial μ, denote by $\mathrm{coeff}_\mu(f)$ the coefficient of μ in f. We write $f(\kappa) \approx_\kappa g(\kappa)$, if $f(\kappa) - g(\kappa)$ is negligible as a function of κ.

Bilinear Maps. Let κ be the security parameter. Let q be a prime of length $O(\kappa)$ bits. Assume we use a secure bilinear group generator $\mathsf{genbp}(1^\kappa)$ that returns $\mathsf{gk} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e})$, where $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ are three multiplicative groups of order q, and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Within this paper, we denote the elements of $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ as in $\mathfrak{g}_1$ (i.e., by using the Fraktur typeface). It is required that $\hat{e}$ is bilinear (i.e., $\hat{e}(\mathfrak{g}_1^a, \mathfrak{g}_2^b) = \hat{e}(\mathfrak{g}_1, \mathfrak{g}_2)^{ab}$), efficiently computable, and non-degenerate. We define $\hat{e}((\mathfrak{A}_1, \mathfrak{A}_2, \mathfrak{A}_3), \mathfrak{B}) = (\hat{e}(\mathfrak{A}_1, \mathfrak{B}), \hat{e}(\mathfrak{A}_2, \mathfrak{B}), \hat{e}(\mathfrak{A}_3, \mathfrak{B}))$ and $\hat{e}(\mathfrak{B}, (\mathfrak{A}_1, \mathfrak{A}_2, \mathfrak{A}_3)) = (\hat{e}(\mathfrak{B}, \mathfrak{A}_1), \hat{e}(\mathfrak{B}, \mathfrak{A}_2), \hat{e}(\mathfrak{B}, \mathfrak{A}_3))$. Assume that $\mathfrak{g}_i$ is a generator of $\mathbb{G}_i$ for $i \in \{1, 2\}$, and set $\mathfrak{g}_T \leftarrow \hat{e}(\mathfrak{g}_1, \mathfrak{g}_2)$.

For $\kappa = 128$, the current recommendation is to use an optimal (asymmetric) Ate pairing over a subclass of Barreto-Naehrig curves. In that case, at security level of $\kappa = 128$, an element of $\mathbb{G}_1/\mathbb{G}_2/\mathbb{G}_T$ can be represented in respectively 256/512/3072 bits.

Cryptosystems. A public-key cryptosystem Π is a triple $(\mathsf{genpkc}, \mathsf{enc}, \mathsf{dec})$ of efficient algorithms. The key generation algorithm $\mathsf{genpkc}(1^\kappa)$ returns a fresh public and secret key pair $(\mathsf{pk}, \mathsf{sk})$. The encryption algorithm $\mathsf{enc}_{\mathsf{pk}}(m; r)$, given a public key pk, a message m, and a randomizer r (from some randomizer space $\mathcal{R}$), returns a ciphertext. The decryption algorithm $\mathsf{dec}_{\mathsf{sk}}(c)$, given a secret key sk and a ciphertext c, returns a plaintext m. It is required that for each $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{genpkc}(1^\kappa)$ and each m, r, it holds that $\mathsf{dec}_{\mathsf{sk}}(\mathsf{enc}_{\mathsf{pk}}(m; r)) = m$. Informally, Π is *IND-CPA secure*, if the distributions of ciphertexts corresponding to any two plaintexts are computationally indistinguishable.

We will use the ILin cryptosystem from [3]; it is distinguished from other well-known cryptosystems like the BBS cryptosystem [1] by having shorter secret and public keys. Consider group $\mathbb{G}_k$, $k \in \{1, 2\}$. In this cryptosytem, where the secret key is $\mathsf{sk} = \gamma \leftarrow_r \mathbb{Z}_q \setminus \{0, -1\}$, the public key is $\mathsf{pk}_k \leftarrow (\mathfrak{g}_k, \mathfrak{h}_k) = (\mathfrak{g}_k, \mathfrak{g}_k^\gamma)$, and the encryption of a small $m \in \mathbb{Z}_q$ is

$$\mathsf{enc}_{\mathsf{pk}_k}(m; \boldsymbol{s}) := (\mathfrak{h}_k^{s_1}, (\mathfrak{g}_k \mathfrak{h}_k)^{s_2}, \mathfrak{g}_k^m \mathfrak{g}_k^{s_1 + s_2})$$

for $\boldsymbol{s} \leftarrow_r \mathbb{Z}_q^{1 \times 2}$. Denote $\mathfrak{P}_{k1} := (\mathfrak{h}_k, \mathfrak{1}_k, \mathfrak{g}_k)$ and $\mathfrak{P}_{k2} := (\mathfrak{1}_k, \mathfrak{g}_k \mathfrak{h}_k, \mathfrak{g}_k)$, thus $\mathsf{enc}_{\mathsf{pk}_k}(m; \boldsymbol{s}) = (\mathfrak{1}_k, \mathfrak{1}_k, \mathfrak{g}_k^m) \cdot \mathfrak{P}_{k1}^{s_1} \mathfrak{P}_{k_2}^{s_2}$. Given $\mathfrak{v} \in \mathbb{G}_k^3$, the decryption sets

$$\mathsf{dec}_{\mathsf{sk}}(\mathfrak{v}) := \log_{\mathfrak{g}_k}(\mathfrak{v}_3 \mathfrak{v}_2^{-1/(\gamma+1)} \mathfrak{v}_1^{-1/\gamma}) \ ,$$

Decryption succeeds since $\mathfrak{v}_3 \mathfrak{v}_2^{-1/(\gamma+1)} \mathfrak{v}_1^{-1/\gamma} = \mathfrak{g}_k^m \mathfrak{g}_k^{s_1+s_2} \cdot (\mathfrak{g}_k \mathfrak{h}_k)^{-s_2/(\gamma+1)} \cdot \mathfrak{h}_k^{-s_1/\gamma} = \mathfrak{g}_k^m \mathfrak{g}_k^{s_1+s_2} \cdot \mathfrak{g}_k^{-s_2/(\gamma+1)} \mathfrak{g}_k^{-s_2 \cdot \gamma/(\gamma+1)} \cdot \mathfrak{g}_k^{-s_1} = \mathfrak{g}_k^m$. This cryptosystem is CPA-secure under the 2-Incremental Linear (2-ILin) assumption, see [3]. The ILin cryptosystem is *blindable*, $\mathsf{enc}_{\mathsf{pk}_k}(m; \boldsymbol{s}) \cdot \mathsf{enc}_{\mathsf{pk}_k}(0; \boldsymbol{s}') = \mathsf{enc}_{\mathsf{pk}}(m; \boldsymbol{s} + \boldsymbol{s}')$.

We use a variant of the ILin cryptosystem where each plaintext is encrypted twice, in group $\mathbb{G}_1$ and in $\mathbb{G}_2$ (but by using the same secret key an the same randomizer $\boldsymbol{s}$ in both). For technical reasons (relevant to the shuffle argument but not to the lLin cryptosystem), in group $\mathbb{G}_1$ we will use an auxiliary generator $\hat{\mathfrak{g}}_1 = \mathfrak{g}_1^{\varrho/\beta}$ instead of $\mathfrak{g}_1$, for $(\varrho, \beta) \leftarrow_r (\mathbb{Z}_q \setminus \{0\})^2$; both encryption and decryption are done as before but just using the secret key $\mathsf{sk} = (\varrho, \beta, \gamma)$ and the public key $\mathsf{pk}_1 = (\hat{\mathfrak{g}}_1, \mathfrak{h}_1 = \hat{\mathfrak{g}}_1^{\gamma})$; this also redefines $\mathfrak{P}_{k1}$. That is, $\mathsf{enc}_{\mathsf{pk}}(m; \boldsymbol{s}) = (\mathsf{enc}_{\mathsf{pk}_1}(m; \boldsymbol{s}), \mathsf{enc}_{\mathsf{pk}_2}(m; \boldsymbol{s}))$, where $\mathsf{pk}_1 = (\hat{\mathfrak{g}}_1, \mathfrak{h}_1 = \hat{\mathfrak{g}}_1^{\gamma})$, and $\mathsf{pk}_2 = (\mathfrak{g}_2, \mathfrak{h}_2 = \mathfrak{g}_2^{\gamma})$, and $\mathsf{dec}_{\mathsf{sk}}(\boldsymbol{\mathfrak{v}}) := \log_{\hat{\mathfrak{g}}_1}(\mathfrak{v}_3 \mathfrak{v}_2^{-1/(\gamma+1)} \mathfrak{v}_1^{-1/\gamma}) = \log_{\mathfrak{g}_1}(\mathfrak{v}_3 \mathfrak{v}_2^{-1/(\gamma+1)} \mathfrak{v}_1^{-1/\gamma})/(\varrho/\beta)$ for $\boldsymbol{\mathfrak{v}} \in \mathbb{G}_1^3$. We call this the *validity-enhanced* lLin cryptosystem.

In this case we denote the ciphertext in group k by $\boldsymbol{\mathfrak{v}}_k$, and its jth component by $\mathfrak{v}_{kj}$. In the case when we have many ciphertexts, we denote the ith ciphertext by $\boldsymbol{\mathfrak{v}}_i$ and the jth component of the ith ciphertext in group k by $\mathfrak{v}_{ikj}$.

2 Shuffle Argument

Let $\mathit{\Pi} = (\mathsf{genpkc}, \mathsf{enc}, \mathsf{dec})$ be an additively homomorphic cryptosystem with randomizer space R; we assume henceworth that one uses the validity-enhanced lLin cryptosystem. Assume that $\boldsymbol{\mathfrak{v}}_i$ and $\boldsymbol{\mathfrak{v}}'_i$ are valid ciphertexts of $\mathit{\Pi}$. In a shuffle argument, the prover aims to convince the verifier in zero-knowledge that given $(\mathsf{pk}, (\boldsymbol{\mathfrak{v}}_i, \boldsymbol{\mathfrak{v}}'_i)_{i=1}^n)$, he knows a permutation $\sigma \in S_n$ and randomizers s_{ij}, $i \in [1 \mathinner{..} n]$ and $j \in [1 \mathinner{..} 2]$, such that $\boldsymbol{\mathfrak{v}}'_i = \boldsymbol{\mathfrak{v}}_{\sigma(i)} \cdot \mathsf{enc}_{\mathsf{pk}}(0; \boldsymbol{s}_i)$ for $i \in [1 \mathinner{..} n]$. More precisely, we define the group-specific binary relation $\mathcal{R}_{sh,n}$ exactly as in [7, 8]:

$$\mathcal{R}_{sh,n} := \left(\begin{array}{l} (\mathsf{gk}, (\mathsf{pk}, \boldsymbol{\mathfrak{v}}_i, \boldsymbol{\mathfrak{v}}'_i)_{i=1}^n), (\sigma, \boldsymbol{s})) : \\ \sigma \in S_n \wedge \boldsymbol{s} \in R^{n \times 2} \wedge \left(\forall i : \boldsymbol{\mathfrak{v}}'_i = \boldsymbol{\mathfrak{v}}_{\sigma(i)} \cdot \mathsf{enc}_{\mathsf{pk}}(0; \boldsymbol{s}_i)\right) \end{array} \right) \enspace .$$

See Prot. 1 for the full description of the new shuffle argument.

We note that in the real mix-net, (γ, ϱ, β) is handled differently (in particular, γ — and possibly ϱ/β — will be known to the decrypting party while (ϱ, β) does not have to be known to anybody) than the real trapdoor (χ, α) that enables one to simulate the argument and thus cannot be known to anybody. Moreover, $(\mathfrak{g}_1, \mathfrak{g}_2)^{\sum P_i(\chi)}$ is in the CRS only to optimize computation.

3 Permutation Matrix Argument

3.1 New 1-Sparsity Argument

In a 1-sparsity argument [8], the prover aims to convince the verifier that he knows how to open a commitment $\mathfrak{A}_1$ to $(\boldsymbol{a}, r)$, such that *at most* one coefficient a_I is non-zero. If, in addition, $a_I = 1$, then we have a unit vector argument [4]. A 1-sparsity argument can be constructed by using square span programs [2], an especially efficient variant of the quadratic span programs of [5]. We prove its security in the GBGM and therefore use a technique similar to that of [6], and this introduces some complications as we will demonstrate below. While we start using ideas behind the unit vector argument of [4], we only obtain a 1-sparsity argument. Then, in Sect. 3, we show how to obtain an efficient permutation matrix argument from it.

Clearly, $\boldsymbol{a} \in \mathbb{Z}_q^n$ is a unit vector iff the following $n + 1$ conditions hold [4]:

- $a_i \in \{0, 1\}$ for $i \in [1 \mathinner{..} n]$ (i.e., $\boldsymbol{a}$ is Boolean), and
- $\sum_{i=1}^n a_i = 1$.

Let $\{0, 2\}^{n+1}$ denote the set of $(n+1)$-dimensional vectors where every coefficient is from $\{0, 2\}$, let $\circ$ denote the Hadamard (entry-wise) product of two vectors, let $V := \left(\begin{smallmatrix} 2 \cdot I_{n \times n} \\ \boldsymbol{1}_n^\top \end{smallmatrix} \right) \in \mathbb{Z}_q^{(n+1) \times n}$ and $\boldsymbol{b} := \left(\begin{smallmatrix} \boldsymbol{0}_n \\ 1 \end{smallmatrix} \right) \in \mathbb{Z}_q^{n+1}$. Clearly, the above $n + 1$ conditions hold iff $V\boldsymbol{a} + \boldsymbol{b} \in \{0, 2\}^{n+1}$, i.e.,

$$(V\boldsymbol{a} + \boldsymbol{b} - \boldsymbol{1}_{n+1}) \circ (V\boldsymbol{a} + \boldsymbol{b} - \boldsymbol{1}_{n+1}) = \boldsymbol{1}_{n+1} \enspace . \tag{1}$$

Let ω_i, $i \in [1 \mathinner{..} n + 1]$ be $n + 1$ different values. Let

$$Z(X) := \prod_{i=1}^{n+1} (X - \omega_i)$$

$\mathsf{gencrs}(1^\kappa, n \in \mathrm{poly}(\kappa))$**:** Call $\mathsf{gk} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}) \leftarrow \mathsf{genbp}(1^\kappa)$. Let $P_i(X)$ for $i \in [0 \mathinner{..} n]$ be polynomials, chosen in Sect. 3. Set $\boldsymbol{\chi} = (\chi, \alpha, \varrho, \beta, \gamma) \leftarrow_r \mathbb{Z}_q^2 \times (\mathbb{Z}_q \setminus \{0\})^2 \times (\mathbb{Z}_q \setminus \{0, -1\})$. Let enc be the ILin cryptosystem with the secret key γ, and let $(\mathsf{pk}_1, \mathsf{pk}_2)$ be its public key. Set

$$\mathsf{crs} \leftarrow \begin{pmatrix} \mathsf{gk}, (\mathfrak{g}_1^{P_i(\chi)})_{i=1}^n, \mathfrak{g}_1^\varrho, \mathfrak{g}_1^{\alpha + P_0(\chi)}, \mathfrak{g}_1^{P_0(\chi)}, (\mathfrak{g}_1^{((P_i(\chi)+P_0(\chi))^2-1)/\varrho})_{i=1}^n, \mathsf{pk}_1 = (\hat{\mathfrak{g}}_1 = \mathfrak{g}_1^{\varrho/\beta}, \mathfrak{h}_1 = \hat{\mathfrak{g}}_1^\gamma), \\ (\mathfrak{g}_2^{P_i(\chi)})_{i=1}^n, \mathfrak{g}_2^\varrho, \mathfrak{g}_2^{-\alpha+P_0(\chi)}, \mathsf{pk}_2 = (\mathfrak{g}_2, \mathfrak{h}_2 = \mathfrak{g}_2^\gamma), \mathfrak{g}_2^\beta, \hat{e}(\mathfrak{g}_1, \mathfrak{g}_2)^{1-\alpha^2}, (\mathfrak{g}_1, \mathfrak{g}_2)^{\sum_{i=1}^n P_i(\chi)} \end{pmatrix} .$$

and $\mathsf{td} \leftarrow (\chi, \varrho)$. Return $(\mathsf{crs}, \mathsf{td})$.

$\mathsf{pro}(\mathsf{crs}; \mathfrak{v} \in (\mathbb{G}_1 \times \mathbb{G}_2)^{3n}; \sigma \in S_n, \boldsymbol{s} \in \mathbb{Z}_q^{n \times 2})$**:**

1. `Commitment function`
   `Input`: permutation σ, CRS elements $((\mathfrak{g}_1^{P_i(\chi)})_{i=1}^n$, $\mathfrak{g}_1^\varrho$, $(\mathfrak{g}_2^{P_i(\chi)})_{i=1}^n$, $\mathfrak{g}_2^\varrho)$.
   (a) For $i = 1$ to $n-1$:
       Set $r_i \leftarrow_r \mathbb{Z}_q$. Set $(\mathfrak{A}_{i1}, \mathfrak{A}_{i2}) \leftarrow (\mathfrak{g}_1, \mathfrak{g}_2)^{P_{\sigma^{-1}(i)}(\chi) + r_i \varrho}$.
   (b) Set $r_n \leftarrow -\sum_{i=1}^{n-1} r_i$.
   (c) Set $(\mathfrak{A}_{n1}, \mathfrak{A}_{n2}) \leftarrow (\mathfrak{g}_1, \mathfrak{g}_2)^{\sum_{i=1}^n P_i(\chi)} / \prod_{i=1}^{n-1} (\mathfrak{A}_{i1}, \mathfrak{A}_{i2})$.
2. `Sparsity, for permutation matrix`
   `Input`: permutation σ, elements $(\mathfrak{A}_{i1})_{i=1}^n$ from commitment,
   CRS elements $(\mathfrak{g}_1^{P_0(\chi)}$, $\mathfrak{g}_1^\varrho$, $(\mathfrak{g}_1^{((P_i(\chi)+P_0(\chi))^2-1)/\varrho})_{i=1}^n)$.
   (a) For $i = 1$ to n:
       Set $\pi_{\mathsf{1sp}:i} \leftarrow (\mathfrak{A}_{i1} \mathfrak{g}_1^{P_0(\chi)})^{2r_i} (\mathfrak{g}_1^\varrho)^{-r_i^2} \mathfrak{g}_1^{((P_{\sigma^{-1}(i)}(\chi)+P_0(\chi))^2-1)/\varrho}$.
3. `Shuffling function`
   `Input`: permutation σ, original ciphertexts $\mathfrak{v}$, randomness $\boldsymbol{s}$ for shuffling, public keys $\mathsf{pk}_1, \mathsf{pk}_2$.
   (a) For $i = 1$ to n: Set $(\mathfrak{v}'_{i1}, \mathfrak{v}'_{i2}) \leftarrow (\mathfrak{v}_{\sigma(i)1}, \mathfrak{v}_{\sigma(i)2}) \cdot (\mathsf{enc}_{\mathsf{pk}_1}(0; \boldsymbol{s}_i), \mathsf{enc}_{\mathsf{pk}_2}(0; \boldsymbol{s}_i))$.
4. `Consistency function`
   `Input`: original ciphertexts $\mathfrak{v}$, randomness $\boldsymbol{r}$ used in commitment, CRS values $((\mathfrak{g}_2^{P_i(\chi)})_{i=1}^n$, $\mathfrak{g}_2^\varrho)$.
   (a) For $k = 1$ to 2: Set $r_{s:k} \leftarrow_r \mathbb{Z}_q$. Set $\pi_{\mathsf{c1}:k} \leftarrow \mathfrak{g}_2^{\sum_{i=1}^n s_{ik} P_i(\chi) + r_{s:k} \varrho}$.
   (b) $$(\boldsymbol{\pi}_{\mathsf{c2}:1}, \boldsymbol{\pi}_{\mathsf{c2}:2}) \leftarrow \prod_{i=1}^n (\mathfrak{v}_{i1}, \mathfrak{v}_{i2})^{r_i} \cdot (\mathsf{enc}_{\mathsf{pk}_1}(0; \boldsymbol{r}_s), \mathsf{enc}_{\mathsf{pk}_2}(0; \boldsymbol{r}_s)).$$
5. Return $\pi_{sh} \leftarrow (\mathfrak{v}', (\mathfrak{A}_{i1}, \mathfrak{A}_{i2})_{i=1}^{n-1}, (\pi_{\mathsf{1sp}:i})_{i=1}^n, \pi_{\mathsf{c1}:1}, \pi_{\mathsf{c1}:2}, \boldsymbol{\pi}_{\mathsf{c2}:1}, \boldsymbol{\pi}_{\mathsf{c2}:2})$.

$\mathsf{ver}(\mathsf{crs}; \mathfrak{v}; \mathfrak{v}', (\mathfrak{A}_{i1}, \mathfrak{A}_{i2})_{i=1}^{n-1}, (\pi_{\mathsf{1sp}:i})_{i=1}^n, \pi_{\mathsf{c1}:1}, \pi_{\mathsf{c1}:2}, \boldsymbol{\pi}_{\mathsf{c2}:1}, \boldsymbol{\pi}_{\mathsf{c2}:2})$**:**

1. Set $(\mathfrak{A}_{n1}, \mathfrak{A}_{n2}) \leftarrow (\mathfrak{g}_1, \mathfrak{g}_2)^{\sum_{i=1}^n P_i(\chi)} / \prod_{i=1}^{n-1} (\mathfrak{A}_{i1}, \mathfrak{A}_{i2})$.
2. Set $(p_{1i}, p_{2j}, p_{3ij}, p_{4j})_{i \in [1 \mathinner{..} n], j \in [1 \mathinner{..} 3]} \leftarrow_r \mathbb{Z}_q^{4n+6}$.
3. Check that `/* Permutation matrix: */`
   $$\prod_{i=1}^n \hat{e}\left((\mathfrak{A}_{i1} \mathfrak{g}_1^{\alpha + P_0(\chi)})^{p_{1i}}, \mathfrak{A}_{i2} \mathfrak{g}_2^{-\alpha + P_0(\chi)} \right) = \hat{e}\left(\prod_{i=1}^n \pi_{\mathsf{1sp}:i}^{p_{1i}}, \mathfrak{g}_2^\varrho \right) \cdot \hat{e}(\mathfrak{g}_1, \mathfrak{g}_2)^{(1-\alpha^2) \sum_{i=1}^n p_{1i}}.$$
4. Check that `/* Validity: */`
   $$\hat{e}\left(\mathfrak{g}_1^\varrho, \prod_{j=1}^3 \pi_{\mathsf{c2}:2j}^{p_{2j}} \cdot \prod_{i=1}^n \prod_{j=1}^3 (\mathfrak{v}'_{i2j})^{p_{3ij}} \right) = \hat{e}\left(\prod_{j=1}^3 \pi_{\mathsf{c2}:1j}^{p_{2j}} \cdot \prod_{i=1}^n \prod_{j=1}^3 (\mathfrak{v}'_{i1j})^{p_{3ij}}, \mathfrak{g}_2^\beta \right).$$
5. Set $$\mathfrak{R} \leftarrow \hat{e}\left(\hat{\mathfrak{g}}_1, \pi_{\mathsf{c1}:2}^{p_{42}} (\pi_{\mathsf{c1}:1} \pi_{\mathsf{c1}:2})^{p_{43}} \right) \cdot \hat{e}\left(\mathfrak{h}_1, \pi_{\mathsf{c1}:1}^{p_{41}} \pi_{\mathsf{c1}:2}^{p_{42}} \right) / \hat{e}\left(\prod_{j=1}^3 \pi_{\mathsf{c2}:1j}^{p_{4j}}, \mathfrak{g}_2^\varrho \right).$$
6. Check that `/* Consistency: */`
   $$\prod_{i=1}^n \hat{e}\left(\prod_{j=1}^3 (\mathfrak{v}'_{i1j})^{p_{4j}}, \mathfrak{g}_2^{P_i(\chi)} \right) / \prod_{i=1}^n \hat{e}\left(\prod_{j=1}^3 \mathfrak{v}_{i1j}^{p_{4j}}, \mathfrak{A}_{i2} \right) = \mathfrak{R}.$$

Protocol 1: The new shuffle argument

be the unique degree $n+1$ monic polynomial, such that $Z(\omega_i) = 0$ for all $i \in [1 \mathinner{..} n+1]$. Let the ith Lagrange basis polynomial

$$\ell_i(X) := \prod_{j \in [1 \mathinner{..} n+1], j \neq i} ((X - \omega_j)/(\omega_i - \omega_j))$$

be the unique degree n polynomial, s.t. $\ell_i(\omega_i) = 1$ and $\ell_i(\omega_j) = 0$ for $j \neq i$.

For $i \in [1 \mathinner{..} n]$, let $P_i(X)$ be the polynomial that interpolates the ith column of the matrix V. That is,

$$P_i(X) = 2\ell_i(X) + \ell_{n+1}(X)$$

for $i \in [1 \mathinner{..} n]$. Let

$$P_0(X) = \ell_{n+1}(X) - 1$$

be the polynomial that interpolates $\boldsymbol{b} - \mathbf{1}_{n+1}$. It can be shown that $\{P_i(X)\}_{i=0}^{n}$ is linearly independent.

We arrive at the polynomial $Q(X) = (\sum_{i=1}^{n} a_i P_i(X) + P_0(X))^2 - 1 = (P_I(X) + P_0(X))^2 - 1$ (here, we used the fact that $\boldsymbol{a} = \boldsymbol{e_I}$ for some $I \in [1 \mathinner{..} n]$), such that $\boldsymbol{a}$ is a unit vector iff $Z(X) \mid Q(X)$. However, since we replace $Z(X)$ in their commitment scheme with X_ϱ, the statement changes slightly: $\boldsymbol{a}$ is a unit vector iff $Q_{wi}(X, X_\varrho) = Q(X)/X_\varrho$ is a valid polynomial (i.e., $\pi_{\mathsf{1sp}} = g_1^{Q_{wi}(\chi, \varrho)}$ can be computed from the CRS).

As in [6], to obtain privacy, we now add randomness $A_\varrho X_\varrho$ to $Q(X)$, arriving at the degree $2n$ polynomial

$$Q_{wi}(X, X_\varrho) = (P_I(X) + P_0(X) + A_\varrho X_\varrho)^2 - 1 \ . \tag{2}$$

Here, X_ϱ is a special independent random variable, and $A_\varrho \leftarrow_r \mathbb{Z}_q$. This means that we will use an instantiation of the polynomial commitment scheme with $P_i(X)$ defined as in the current subsection.

3.2 Permutation Matrix Argument

Assume we explicitly compute $\mathfrak{A}_{n1} = \mathfrak{g}_1^{\sum_{i=1}^{n} P_i(\chi)} / \prod_{j=1}^{n-1} \mathfrak{A}_{j1}$ as in Prot. 1, and then apply the 1-sparsity argument to each $\mathfrak{A}_{i1}$, $i \in [1 \mathinner{..} n]$. Then, as in [8], we get that $(\mathfrak{A}_{11}, \dots, \mathfrak{A}_{n1})$ commits to a permutation matrix.

4 Validity Argument

The shuffle argument employs validity arguments for $(\boldsymbol{\pi}_{\mathsf{c2:1}}, \boldsymbol{\pi}_{\mathsf{c2:2}})$ and for each $(\boldsymbol{\mathfrak{v}}'_{i1}, \boldsymbol{\mathfrak{v}}'_{i2})$. We outline this argument for $(\boldsymbol{\pi}_{\mathsf{c2:1}}, \boldsymbol{\pi}_{\mathsf{c2:2}})$, the argument is the same for $(\boldsymbol{\mathfrak{v}}'_{i1}, \boldsymbol{\mathfrak{v}}'_{i2})$. More precisely, in the validity argument for $(\boldsymbol{\pi}_{\mathsf{c2:1}}, \boldsymbol{\pi}_{\mathsf{c2:2}})$, the verifier checks that $\hat{e}(\mathfrak{g}_1^{\varrho}, \pi_{\mathsf{c2:2}j}) = \hat{e}(\pi_{\mathsf{c2:1}j}, \mathfrak{g}_2^{\beta})$ for $j \in [1 \mathinner{..} 3]$. Thus, for

$$\mathcal{V}_{val:j}(\boldsymbol{X}) = E_{1j}(\boldsymbol{X}) X_\beta - X_\varrho E_{2j}(\boldsymbol{X}) \ ,$$

this argument guarantees that in the GBGM, $\mathcal{V}_{val:j}(\boldsymbol{X}) = 0$ for $j \in [1 \mathinner{..} 3]$.

Essentially, this guarantees that $(\boldsymbol{\mathfrak{v}}'_{i1}, \boldsymbol{\mathfrak{v}}'_{i2})$ decrypt into the same combination of specific CRS values. In the case of an honest prover, these ciphertexts are encryptions of the same plaintext $m_i \in \mathbb{Z}_q$.

5 Consistency Argument

We call the subargument of Prot. 1, where the verifier only executes the last verification, the *consistency argument*. Intuitively, the consistency argument guarantees that the ciphertexts have been permuted by using the same permutation according to which the elements $\mathfrak{g}_k^{P_i(\chi)}$ were permuted inside the commitments $\mathfrak{A}_{i1}$.

Let $\boldsymbol{\mathfrak{v}}$ be the original ciphertexts, where $\mathfrak{v}_i = \mathsf{enc}_{\mathsf{pk}}(m_i; u_i)$. If the ciphertexts $\boldsymbol{\mathfrak{v}'}$ were permuted using some permutation ψ but not re-randomized (so for $i \in [1 \mathinner{..} n]$, $t_i = 0$ and $\mathfrak{v}'_i = \mathfrak{v}_{\psi(i)}$), we can easily verify that $\mathfrak{v}'_i = \mathfrak{v}_{\psi(i)}$ by checking that

$$\prod_{i=1}^{n} \hat{e}(\mathfrak{v}'_i, g_2^{P_i(\chi)}) = \prod_{i=1}^{n} \hat{e}(\mathfrak{v}_i, g_2^{P_{\psi^{-1}(i)}(\chi)}) \ .$$

Note that this holds iff the equation

$$\sum_{i=1}^{n} P_i(\chi) u_{\psi(i)} = \sum_{i=1}^{n} P_{\psi^{-1}(i)}(\chi) u_i$$

also holds.

However, the verifier can find the permutation ψ by comparing $\boldsymbol{z}$ and $\boldsymbol{z}'$. Also, to check this verification equation, the verifier needs to know the values $g_2^{P_{\psi^{-1}(i)}(\chi)}$.

To ensure privacy, $\boldsymbol{\mathfrak{v}}'$ needs to be both permuted and randomized (i.e. $\mathfrak{v}'_i = \mathfrak{v}_{\psi(i)} \cdot \mathsf{enc}_{\mathsf{pk}}(1; t_i)$ for some randomness value t_i), and we need to replace $g_2^{P_{\psi^{-1}(i)}(\chi)}$ with the value $\mathfrak{A}_{i2}$, where

$$(\mathfrak{A}_{i1}, \mathfrak{A}_{i2}) = (g_1, g_2)^{r_i P_0(\chi) + P_{\psi^{-1}(i)}(\chi)}$$

is a commitment of $\boldsymbol{e}_{\psi^{-1}(i)}$, the $\psi^{-1}(i)$-th unit vector. This adds an extra term E in the verification equation, such that the check becomes

$$\prod_{i=1}^{n} \hat{e}(\mathfrak{v}'_i, g_2^{P_i(\chi)}) = \prod_{i=1}^{n} \hat{e}(\mathfrak{v}_i, \mathfrak{A}_{i2}) \cdot E. \qquad (3)$$

Note that E depends on the CRS and the randomness values $\boldsymbol{t}$ used in shuffling, and $\boldsymbol{r}$ used in committing the permutation matrix.

References

1. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: CRYPTO 2004. LNCS, vol. 3152, pp. 41–55
2. Danezis, G., Fournet, C., Groth, J., Kohlweiss, M.: Square Span Programs with Applications to Succinct NIZK Arguments. In: ASIACRYPT 2014 (1). LNCS, vol. 8873, pp. 532–550
3. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.L.: An Algebraic Framework for Diffie-Hellman Assumptions. In: CRYPTO (2) 2013. LNCS, vol. 8043, pp. 129–147
4. Fauzi, P., Lipmaa, H.: Efficient Culpably Sound NIZK Shuffle Argument without Random Oracles. In: CT-RSA 2016. LNCS, vol. 9610, pp. 200–216
5. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic Span Programs and NIZKs without PCPs. In: EUROCRYPT 2013. LNCS, vol. 7881, pp. 626–645
6. Groth, J.: On the Size of Pairing-based Non-interactive Arguments. In: EUROCRYPT 2016. LNCS, vol. 9666, pp. 305–326
7. Groth, J., Lu, S.: A Non-interactive Shuffle with Pairing Based Verifiability. In: ASIACRYPT 2007. LNCS, vol. 4833, pp. 51–67
8. Lipmaa, H., Zhang, B.: A More Efficient Computationally Sound Non-Interactive Zero-Knowledge Shuffle Argument. Journal of Computer Security **21**(5) (2013) pp. 685–719