# A Non-Interactive Range Proof with Constant Communication*

Rafik Chaabouni[1,2], Helger Lipmaa[1], and Bingsheng Zhang[1]

[1] Institute of Computer Science, University of Tartu, Estonia
[2] Security and Cryptography Laboratory, EPFL, Switzerland

**Abstract.** In a range proof, the prover convinces the verifier in zero-knowledge that he has encrypted or committed to a value $a \in [0, H]$ where $H$ is a public constant. Most of the previous non-interactive range proofs have been proven secure in the random oracle model, that is, heuristically. We show that one of the few previous non-interactive range proofs in the common reference string (CRS) model, proposed by Yuen et al. in COCOON 2009, is insecure. We then construct a secure non-interactive range proof that works in the CRS model. The new range proof can have (by different instantiations of the parameters) either very short communication (14 080 bits) and verifier's computation (81 pairings), short combined CRS length and communication ($\log^{1/2+o(1)} H$ group elements), or very efficient prover's computation ($\Theta(\log H)$ exponentiations).
**Keywords.** NIZK, pairings, progression-free sets, range proof.

## 1 Introduction

In a range proof, the prover convinces the verifier in zero-knowledge that he has encrypted or committed to a value $a \in [0, H]$, where $H$ is a public constant. Range proofs are needed in a wide variety of cryptographic protocols, like e-voting [CGS97,DJ01] (to show that a ballot corresponds to a valid candidate), e-auctions [LAN02], anonymous credentials, e-cash [CHL05], or any other protocol that needs for its correctness that the inputs are from a valid range. Given the need for range proofs in a large variety of protocols, it is not surprising that there is a large amount of research on this topic.

Most of the existing efficient range proofs fall in one of the next two categories. The first category uses a classical result of Lagrange that every non-negative integer is a sum of four squares [Lip03,Gro04,YHM+09]. However, in this case the underlying group has to be of unknown order which seriously limits the available cryptographic techniques. In particular, all known *secure* Lagrange's theorem based range proofs are based on operations in $\mathbb{Z}_n^*$ for a hard-to-factor $n$. Since to achieve 128-bit security level, $n$ must be at least 3072 bits long, arithmetic in $\mathbb{Z}_n^*$ is relatively slow. One also has to compute the four squares of the Lagrange's theorem which is inefficient by itself. Furthermore, this means that it is not known how to instantiate such schemes with bilinear groups. (This is exemplified by the fact that we break the range proof of [YHM+09] where the Lagrange theorem is used in the bilinear setting with known group order.)

Due to such considerations, one usually considers the second approach. There, one uses the fact that $a \in [0, H]$, if and only if for some well chosen coefficients $G_i$, there exist $b_i \in [0, u-1]$ such that $a = \sum_{i=1}^{n} G_i b_i$. Here, $u \ll H$ and $n$ is also small. One then proves separately for every $b_i$ that $b_i \in [0, u-1]$, and uses additively homomorphic properties of the used commitment scheme to verify that $a = \sum_{i=1}^{n} G_i b_i$. The goal is to minimize the communication of that type of range proofs.

Clearly, $a \in [0, 2^d - 1]$ iff $a = \sum_{i=1}^{d} 2^{i-1} b_i$ and $b_i \in \{0, 1\}$. Then one can prove that $a \in [0, H]$ for arbitrary $H$ by showing that both $a$ and $H - a$ belong to $[0, 2^{\lfloor \log_2 H \rfloor + 1} - 1]$. Showing that $b_i \in \{0, 1\}$ is straightforward, e.g., by using an AND of two $\Sigma$-protocols [CDS94]. This means that one has to execute two basic range proofs for $[0, 2^d - 1]$. Lipmaa, Asokan and Niemi showed in [LAN02] that by choosing the coefficients $G_i$ cleverly, one obtains a simpler result that $a \in [0, H]$, for any $H > 1$, iff $a = \sum_{i=1}^{\lfloor \log_2 H \rfloor + 1} G_i b_i$ and $b_i \in \{0, 1\}$.

In [CCs08], the authors considered the general case $u \geq 2$, following the fact that $a \in [0, u^d - 1]$ iff $a = \sum_{i=1}^{d} u^i b_i$ and $b_i \in [0, u-1]$. They show that $b_i \in [0, u-1]$ by letting the verifier to sign every integer in $[0, u-1]$, and then letting the prover to prove that he knows the signature on committed $b_i$. One can show that $a \in [0, H]$ for general $H$ by using again an AND of two $\Sigma$-protocols. Nontrivially generalizing [LAN02] (by using methods from additive combinatorics), Chaabouni, Lipmaa and shelat [CLs10]

* Full preproceedings version, January 27, 2012. Final full version might differ.

showed that there exist (efficiently computable) coefficients $G_i$ such that $(u-1)a \in (u-1)\cdot[0,H]$ iff $a = \sum_{i=1}^{\lceil \log_u((u-1)\cdot H+1)\rceil} G_i b_i$ for some $b_i \in [0, u-1]$. The CLS range proof has communication complexity of $\Theta(\log_u H + u)$ group elements, which obtains minimal value $\Theta(\log H/\log\log H)$ if $u \approx \log H/\log\log H$. (See [Gro11] for recent related work.)

Usually, it is desired that the range proof is non-interactive. For example, in the e-voting scenario, range proof is a part of the vote validity proof that is verified by various parties without any active participation of the voter. Most of the previous non-interactive range proofs first construct a $\Sigma$-protocol which is then made non-interactive in the random oracle model by using the Fiat-Shamir heuristic. While the random oracle model allows to construct efficient protocols, it is also known that there exist protocols that are secure in the random oracle models and insecure in the plain model.

Motivated by this, [CHS04,YHM+09,RKP09] have proposed non-interactive range proofs without random oracles. The range proof from [CHS04] is of mainly theoretical value. The range proof from [YHM+09] uses Lagrange's theorem, but we will demonstrate an attack on it. The range proof from [RKP09] combines the range proof of [CCs08] with the Groth-Sahai non-interactive zero-knowledge (NIZK) proofs [GS08] and P-signatures. The [RKP09] range proof is not claimed to be zero-knowledge (only NIWI, that is, non-interactive witness-indistinguishable).

We first show that the protocol from [YHM+09] is insecure. The main idea of the attack comes from using Pedersen commitments in a group of known order. In this case, using Lagrange's theorem to prove that a non-negative number is the sum of four squares fails. We can only conclude that the sum of four squares is computed modulo the group order. Hence an attacker can prove that any number is "non-negative" and completely break the protocol in [YHM+09]. See Sect. 4 for more information.

We then construct a new NIZK range proof (for an encrypted $a$ — if one needs $a$ to be committed, one can use the same cryptosystem as a perfectly binding commitment) that works in the common-reference string model. We do this by using recent NIZK arguments by Groth and Lipmaa [Gro10,Lip12]. We also use the additive combinatorics results from [CLs10], that is, we base a range proof $a \in [0, H]$ on the fact that $(u-1)a \in (u-1)\cdot[0,H]$ iff $a = \sum_{i=1}^{n} G_i b_i$ and $b_i \in [0, u-1]$, where $G_i$ are as defined in [CLs10]. However, differently from [CLs10], we prove that $b_i \in [0, u-1]$ by proving (by a recursive use of the method from [LAN02,CLs10]) that $b_i = \sum_{j=0}^{n_v} G'_j b'_{ji}$ with $b'_{ji} \in [0,1]$. Here, $n_v := \lfloor \log_2(u-1)\rfloor$. By using the commitment scheme of [Gro10,Lip12] that enables to succinctly commit to a vector $(b_1, \dots, b_n)$, and the Hadamard product argument of [Gro10,Lip12], we can do all $n_v + 1$ small range proofs in parallel. In addition, in Sect. 5 we construct a new non-interactive argument that a knowledge-commited value is equal to a BBS-encrypted [BBS04] value. (Due to the use of knowledge assumptions, this proof is computationally more efficient than the one constructed by using Groth-Sahai proofs [GS08].) The new range proof does not rely on the random oracle model or use any proofs of knowledge of signatures.

The complexity of the new protocol is described in Tbl. 1. Setting $u = 2$ results in a constant argument length (but CRS of $\Theta((\log H)^{1+o(1)})$ group elements). By using an efficient variation of Barreto-Naehrig curves (where the group elements are either 256 or 512 bits), the communication drops to 14 080 bits. The range proof of [RKP09] does not allow for constant communication. Moreover, if $u = 2$ then the communication is even smaller than that of the known range proofs based on the Lagrange's theorem like [Lip03]. We note that constant communication is achieved since the new range proof uses permutation arguments only for permutations that do not depend on the statement. On the other hand, setting $u = H$ results in summatory CRS and argument length of $\log^{1/2+o(1)} H$, and setting $u = 2^{\sqrt{\log H}}$ results in prover's computational complexity dominated by $\Theta(\log H)$ exponentiations. The previous non-interactive range proofs did not allow for such a flexibility.

This is the full version that corresponds to the publication at FC 2012.

One can obtain a zap [DN00,Gro10] (that is, a 2-message public-coin witness-indistinguishable proof) from the NIZK range proof by first letting the verifier create and send a CRS to the prover, and then letting the prover to send the range proof to the verifier. This zap works in the standard model (without needing a CRS since it is generated on run) and has total communication $\log^{1/2+o(1)} H$ in the case $u = H$.

## 2 Preliminaries

Let $[L, H] = \{L, L+1, \dots, H-1, H\}$ and $[H] = [1, H]$. Let $S_n$ be the set of permutations from $[n]$ to $[n]$. By $\boldsymbol{a}$, we denote the vector $\boldsymbol{a} = (a_1, \dots, a_n)$. If $A$ is a value, then $x \leftarrow A$ means that $x$ is set to $A$. If $A$ is a set, then $x \leftarrow A$ means that $x$ is picked uniformly and randomly from $A$. If $y = h^x$, then let

| | CRS length | Argument length | Prover comp. | Verifier comp. |
|---|---|---|---|---|
| [RKP09] | $\Theta(1)$ | $\Theta(h)$ | $\Theta(h)$ | $\Theta(h)$ |
| [RKP09] | $\Theta(\frac{h}{\log h})$ | $\Theta(\frac{h}{\log h})$ | $\Theta(\frac{h}{\log h})$ | $\Theta(\frac{h}{\log h})$ |
| This paper | | | | |
| General | $n^{1+\varepsilon}$ | $5n_v + 40$ | $\Theta(n^2 n_v)\mathrm{M} + \Theta(n^{1+o(1)} n_v)\mathrm{E}$ | $(9n_v + 81)$ P |
| $u = 2$ | $h^{1+\varepsilon}$ | 40 | $\Theta(h^2)\mathrm{M} + h^{1+\varepsilon}\mathrm{E}$ | 81 P |
| $u = 2^{\sqrt{h}}$ | $h^{1/2+\varepsilon}$ | $\approx 5\sqrt{h} + 40$ | $\Theta(h^{3/2})\mathrm{M} + h^{1+\varepsilon}\mathrm{E}$ | $\approx (9\sqrt{h} + 81)$ P |
| $u = H$ | $\Theta(1)$ | $\approx 5h + 40$ | $\Theta(h)\mathrm{E}$ | $\approx (9h + 81)$ P |

**Table 1.** Comparison of NIZK arguments for range proof. Here, M/E/P means the number of multiplications, exponentiations and pairings. Communication is given in group elements. Here, $n_v = \lfloor \log(u-1) \rfloor$, $n \approx \log H / \log u$ and $\varepsilon = o(1)$, and the basis of all logarithms is 2. To fit in page margins, in this table only, we write $h = \log_2 H$.

$\log_h y := x$. Let $\kappa$ be the security parameter. We abbreviate probabilistic polynomial-time as PPT. We say that $\Lambda = (\lambda_1, \dots, \lambda_n) \subset \mathbb{Z}$ is an $(n, \kappa)$-*nice tuple*, if $0 < \lambda_1 < \cdots < \lambda_i < \cdots < \lambda_n = \mathrm{poly}(\kappa)$.

By using notation from additive combinatorics [TV06], if $\Lambda_1$ and $\Lambda_2$ are subsets of some additive group ($\mathbb{Z}$ or $\mathbb{Z}_p$ within this paper), then

$$\Lambda_1 + \Lambda_2 = \{\lambda_1 + \lambda_2 : \lambda_1 \in \Lambda_1 \wedge \lambda_2 \in \Lambda_2\}$$

is their *sum set* and

$$\Lambda_1 - \Lambda_2 = \{\lambda_1 - \lambda_2 : \lambda_1 \in \Lambda_1 \wedge \lambda_2 \in \Lambda_2\}$$

is their *difference set*. If $\Lambda$ is a set, then

$$k\Lambda = \{\lambda_1 + \cdots + \lambda_k : \lambda_i \in \Lambda\}$$

is an *iterated sumset*, and

$$k \cdot \Lambda = \{k\lambda : \lambda \in \Lambda\}$$

is a *dilation* of $\Lambda$. Let

$$2\hat{}\Lambda = \{\lambda_1 + \lambda_2 : \lambda_1 \in \Lambda \wedge \lambda_2 \in \Lambda \wedge \lambda_1 \neq \lambda_2\} \subseteq \Lambda + \Lambda$$

denote a *restricted sumset* [TV06].

A set $\{\lambda_1, \dots, \lambda_n\} \subset \mathbb{Z}^+$ is *progression-free* [TV06], if no three of the numbers are in arithmetic progression, so that $\lambda_i + \lambda_j = 2\lambda_k$ only if $i = j = k$. Let $r_3(N)$ denote the cardinality of the largest progression-free set that belongs to $[N]$. Recently, Elkin [Elk11] showed that

$$r_3(N) = \Omega((N \cdot \log_2^{1/4} N)/2^{2\sqrt{2 \log_2 N}}) \ .$$

It is also known that $r_3(N) = O(N(\log\log N)^5/\log N)$ [San11]. Thus, the minimal $N$ such that $r_3(N) = n$ is $\omega(n)$, while according to Elkin, $N = n^{1+o(1)}$.

**Fact 1 (Lipmaa [Lip12])** *For any fixed $n > 0$, there exists $N = n^{1+o(1)}$, such that $[N]$ contains a progression-free subset $\Lambda$ of odd integers of cardinality $n$.*

**Bilinear Groups.** Let $\mathcal{G}_{\mathsf{bp}}(1^\kappa)$ be a bilinear group generator that outputs a description of a bilinear group $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$ such that $p$ is a $\kappa$-bit prime, $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are multiplicative cyclic groups of order $p$, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map (pairing) such that $\forall a, b \in \mathbb{Z}$, $t \in \{1, 2\}$ and $g_t \in \mathbb{G}_t$, $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$. If $g_t$ generates $\mathbb{G}_t$ for $t \in \{1, 2\}$, then $\hat{e}(g_1, g_2)$ generates $\mathbb{G}_T$. Moreover, it is efficient to decide the membership in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, group operations and the pairing $\hat{e}$ are efficiently computable, generators are efficiently sampleable, and the descriptions of the groups and group elements each are $O(\kappa)$ bit long. One can implement an optimal (asymmetric) Ate pairing [HSV06] over a subclass of Barreto-Naehrig curves [BN05,PSNB11] very efficiently. In that case, at security level of 128-bits, an element of $\mathbb{G}_1/\mathbb{G}_2/\mathbb{G}_T$ can be represented in respectively 256/512/3072 bits.

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 PPT adversaries $\mathcal{A}$, the next probability is negligible in $\kappa$:

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

Let $\Lambda$ be an $(n,\kappa)$-nice tuple for some $n = \mathrm{poly}(\kappa)$. We say that a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is *$\Lambda$-PSDL secure*, if for any non-uniform PPT adversary $\mathcal{A}$,

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

is negligible in $\kappa$. Let $\Lambda$ be an $(n,\kappa)$-nice tuple. According to [Lip12], any successful generic adversary for $\Lambda$-PSDL requires time $\Omega(\sqrt{p/\lambda_n})$ where $p$ is the group order and $\lambda_n$ is the largest element of $\Lambda$.

The soundness of NIZK arguments (for example, an argument that a computationally binding commitment scheme commits to 0) seems to be an unfalsifiable assumption in general. We will use a weaker version of soundness in the case of subarguments, but in the case of the range proof, we will prove soundness. Similarly to [Gro10,Lip12], we will base the soundness of that argument on an explicit knowledge assumption.

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

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

is negligible in $\kappa$. Groth [Gro10] proved that the $[n]$-PKE assumption holds in the generic group model; his proof can be modified to the general case.

**BBS Cryptosystem.** A public-key cryptosystem $(\mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{E}\mathsf{nc})$ is a triple of efficient algorithms, key generation, encryption, and decryption. It is required that for any $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{G}_{\mathsf{pkc}}(1^\kappa)$ and any valid $m$ and randomizer $r$, one has $\mathcal{D}\mathsf{ec}_{\mathsf{sk}}(\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(m; r)) = m$. A cryptosystem is IND-CPA secure, if for any $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{G}_{\mathsf{pkc}}(1^\kappa)$ and any two messages $m_0$ and $m_1$, the distributions $\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(m_0; \cdot)$ and $\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(m_1; \cdot)$ are computationally indistinguishable. In the *lifted BBS cryptosystem* [BBS04] (in group $\mathbb{G}_1$), the system parameters are equal to $(\mathsf{gk}; g_1)$, where $\mathsf{gk} \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$ and $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$. The secret key $\mathsf{sk}$ is $(\mathsf{sk}_1, \mathsf{sk}_2) \leftarrow (\mathbb{Z}_p^*)^2$, the public key $\mathsf{pk}$ is $(f, h) \leftarrow (g_1^{1/\mathsf{sk}_1}, g_1^{1/\mathsf{sk}_2})$. One encrypts $a \in \mathbb{Z}_p$ as

$$\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(\mathsf{ck}_1; a; r_f, r_h) \leftarrow (c_g, c_f, c_h) = (g_1^{r_f + r_h + a}, f^{r_f}, h^{r_h}) \ ,$$

where $(r_f, r_h) \leftarrow \mathbb{Z}_p^2$. One decrypts $(c_g, c_f, c_h)$ by returning the discrete logarithm of $c_g / (c_f^{\mathsf{sk}_1} c_h^{\mathsf{sk}_2})$. The BBS cryptosystem is IND-CPA secure under the DLIN assumption.

**Commitment Schemes in the CRS Model.** A (batch) commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathsf{om})$ in a bilinear group consists of two PPT algorithms: a randomized CRS generation algorithm $\mathcal{G}_{\mathsf{com}}$, and a randomized commitment algorithm $\mathcal{C}\mathsf{om}$. Here, $\mathcal{G}_{\mathsf{com}}^t(1^\kappa, n)$, $t \in \{1,2\}$, produces a CRS $\mathsf{ck}_t$, and $\mathcal{C}\mathsf{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r)$, with $\boldsymbol{a} = (a_1, \ldots, a_n)$, outputs a commitment value $A$ in $\mathbb{G}_t$ (or $\mathbb{G}_t^b$ for $b > 1$). We assume that a commitment $\mathcal{C}\mathsf{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r)$ is opened by revealing $(\boldsymbol{a}, r)$.

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

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

is negligible in $\kappa$. A commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathsf{om})$ is *perfectly hiding in group* $\mathbb{G}_t$, if for any positive integer $n = \mathrm{poly}(\kappa)$ and $\mathsf{ck}_t \in \mathcal{G}^t_{\mathsf{com}}(1^\kappa, n)$ and any two messages $\boldsymbol{a_1}, \boldsymbol{a_2}$, the distributions $\mathcal{C}\mathsf{om}^t(\mathsf{ck}_t; \boldsymbol{a_1}; \cdot)$ and $\mathcal{C}\mathsf{om}^t(\mathsf{ck}_t; \boldsymbol{a_2}; \cdot)$ are equal.

A trapdoor commitment scheme has three additional efficient algorithms: (a) A trapdoor CRS generation algorithm inputs $t$, $n$ and $1^\kappa$ and outputs a CRS $\mathsf{ck}^*$ (that has the same distribution as $\mathcal{G}^t_{\mathsf{com}}(1^\kappa, n)$) and a trapdoor $\mathsf{td}$, (b) a randomized trapdoor commitment that takes $\mathsf{ck}^*$ and a randomizer $r$ as inputs and outputs the value $\mathcal{C}\mathsf{om}^t(\mathsf{ck}^*; \boldsymbol{0}; r)$, and (c) a trapdoor opening algorithm that takes $\mathsf{ck}^*$, $\mathsf{td}$, $\boldsymbol{a}$ and $r$ as an input and outputs an $r'$ such that $\mathcal{C}\mathsf{om}^t(\mathsf{ck}^*; \boldsymbol{0}; r) = \mathcal{C}\mathsf{om}^t(\mathsf{ck}^*; \boldsymbol{a}; r')$.

An *extractable commitment scheme* [Di 02,ACP09] is a commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathsf{om})$ with an additional extractor $(\mathsf{Extr}_1, \mathsf{Extr}_2)$ such that: $\mathsf{Extr}^t_1(1^\kappa)$ creates a CRS $\mathsf{ck}^*$ (indistinguishable from the real CRS $\mathsf{ck}$) and a trapdoor $\mathsf{td}$, and $\mathsf{Extr}_2(\mathsf{ck}^*, \mathsf{td}; A)$ returns $(a; r)$ such that $A = \mathcal{C}\mathsf{om}(\mathsf{ck}; a; r)$, given that $A$ is a valid commitment. An extractable commitment scheme can only be computationally hiding.

We use the *knowledge commitment scheme*, defined in [Lip12], as follows.

**CRS generation:** Let $\Lambda$ be a $(n, \kappa)$-nice tuple with $n = \mathrm{poly}(\kappa)$. Define $\lambda_0 = 0$. Given a bilinear group generator $\mathcal{G}_{\mathsf{bp}}$, set $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ be generators, and choose random $\hat{\alpha}, x \leftarrow \mathbb{Z}_p$. Fix $t \in \{1, 2\}$. The CRS is $\mathsf{ck}_t \leftarrow (\mathsf{gk}; (g_{t,\lambda_i}, \hat{g}_{t,\lambda_i})_{i \in \{0,\ldots,n\}})$, where $g_{t,\lambda_i} = g_t^{x^{\lambda_i}}$, and $\hat{g}_{t,\lambda_i} = g_t^{\hat{\alpha} x^{\lambda_i}}$.

**Commitment:** To commit to $\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_p^n$, one chooses a random $r \leftarrow \mathbb{Z}_p$, and computes

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

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

According to [Lip12], the knowledge commitment scheme is statistically hiding in group $\mathbb{G}_t$, and computationally binding in group $\mathbb{G}_t$ under the $\Lambda$-PSDL assumption in group $\mathbb{G}_t$. If the $\Lambda$-PKE assumption holds in group $\mathbb{G}_t$, then for any non-uniform PPT algorithm $\mathcal{A}$, that outputs some valid knowledge commitments, there exists a non-uniform PPT extractor $X_{\mathcal{A}}$ that, given as an input the input of $\mathcal{A}$ together with $\mathcal{A}$'s random coins, extracts the contents of these commitments. The knowledge commitment scheme is also trapdoor, with the trapdoor being $\mathsf{td} = x$: after trapdoor-committing $A \leftarrow \mathcal{C}\mathsf{om}^t(\mathsf{ck}; \boldsymbol{0}; r) = g_t^r$ for $r \leftarrow \mathbb{Z}_p$, the committer can open it to $(\boldsymbol{a}; r - \sum_{i=1}^n a_i x^{\lambda_i})$ for any $\boldsymbol{a}$.

**Non-Interactive Zero-Knowledge.** Let $\mathcal{R} = \{(C, w)\}$ be an efficiently computable binary relation such that $|w| = \mathrm{poly}(|C|)$. Here, $C$ is a statement, and $w$ is a witness. Let $\mathcal{L} = \{C : \exists w, (C, w) \in \mathcal{R}\}$ be an **NP**-language. Let $n = |C|$ be a fixed input length. For fixed $n$, we have a relation $\mathcal{R}_n$ and a language $\mathcal{L}_n$. A *non-interactive argument* for $\mathcal{R}$ consists of the next PPT algorithms: a common reference string (CRS) generator $\mathcal{G}_{\mathsf{crs}}$, a prover $\mathcal{P}$, and a verifier $\mathcal{V}$. For $\mathsf{crs} \leftarrow \mathcal{G}_{\mathsf{crs}}(1^\kappa, n)$, $\mathcal{P}(\mathsf{crs}; C, w)$ produces an argument $\psi$. The verifier $\mathcal{V}(\mathsf{crs}; C, \psi)$ outputs either 1 (accept) or 0 (reject).

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

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

is negligible in $\kappa$.

A non-interactive argument $(\mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly witness-indistinguishable*, if (given that there are several possible witnesses) it is impossible to tell which witness the prover used. That is, for all $n = \mathrm{poly}(\kappa)$, if $\mathsf{crs} \in \mathcal{G}_{\mathsf{crs}}(1^\kappa, n)$ and $((C, w_0), (C, w_1)) \in \mathcal{R}_n^2$, then the distributions $\mathcal{P}(\mathsf{crs}; C, w_0)$ and $\mathcal{P}(\mathsf{crs}; C, w_1)$ are equal. A non-interactive argument $(\mathcal{G}_{\mathsf{crs}}, \mathcal{P}, \mathcal{V})$ is *perfectly zero-knowledge*, if there exists a polynomial-time simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, such that for all stateful interactive non-uniform PPT adversaries $\mathcal{A}$ and $n = \mathrm{poly}(\kappa)$,

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

Here, $\mathsf{td}$ is the *simulation trapdoor*.

**System parameters:** Let $n = \mathrm{poly}(\kappa)$. Let $\Lambda = \{\lambda_i : i \in [n]\}$ be a progression-free set of odd integers, such that $\lambda_{i+1} > \lambda_i > 0$. Denote $\lambda_0 := 0$. Let $\hat{\Lambda} := \{0\} \cup \Lambda \cup 2\hat{\ }\Lambda$.
**CRS generation** $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $\hat{\alpha}, x \leftarrow \mathbb{Z}_p$. Let $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$ and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. Denote $g_{t\ell} \leftarrow g_t^{x^\ell}$ and $\hat{g}_{t\ell} \leftarrow g_t^{\hat{\alpha} x^\ell}$ for $t \in \{1, 2\}$ and $\ell \in \{0\} \cup \hat{\Lambda}$. Let $D \leftarrow \prod_{i=1}^n g_{2,\lambda_i}$. The CRS is $\mathsf{crs} \leftarrow (\mathsf{gk}; (g_{1\ell}, \hat{g}_{1\ell})_{\ell \in \{0\} \cup \Lambda}, (g_{2\ell}, \hat{g}_{2\ell})_{\ell \in \hat{\Lambda}}, D)$. Let $\widehat{\mathsf{ck}}_1 \leftarrow (\mathsf{gk}; (g_{1\ell}, \hat{g}_{1\ell})_{\ell \in \{0\} \cup \Lambda})$.
**Common inputs:** $(A, \hat{A}, B, \hat{B}, B_2, C, \hat{C})$, where $(A, \hat{A}) \leftarrow \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{a}; r_a)$, $(B, \hat{B}) \leftarrow \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{b}; r_b)$, $B_2 \leftarrow g_2^{r_b} \cdot \prod_{i=1}^n g_{2,\lambda_i}^{b_i}$, $(C, \hat{C}) \leftarrow \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{c}; r_c)$, s.t. $a_i b_i = c_i$ for $i \in [n]$.
**Argument generation** $\mathcal{P}_\times(\mathsf{crs}; (A, \hat{A}, B, \hat{B}, B_2, C, \hat{C}), (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, \boldsymbol{c}, r_c))$**:** Let $I_1(\ell) := \{(i, j) : i, j \in [n] \wedge j \neq i \wedge \lambda_i + \lambda_j = \ell\}$. For $\ell \in 2\hat{\ }\Lambda$, the prover sets $\mu_\ell \leftarrow \sum_{(i,j) \in I_1(\ell)} (a_i b_j - c_i)$. He sets $\psi \leftarrow g_2^{r_a r_b} \cdot \prod_{i=1}^n g_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{\ell \in 2\hat{\ }\Lambda} g_{2\ell}^{\mu_\ell}$, and $\hat{\psi} \leftarrow \hat{g}_2^{r_a r_b} \cdot \prod_{i=1}^n \hat{g}_{2,\lambda_i}^{r_a b_i + r_b a_i - r_c} \cdot \prod_{\ell \in 2\hat{\ }\Lambda} \hat{g}_{2\ell}^{\mu_\ell}$. He sends $\psi^\times \leftarrow (\psi, \hat{\psi}) \in \mathbb{G}_2^2$ to the verifier as the argument.
**Verification** $\mathcal{V}_\times(\mathsf{crs}; (A, \hat{A}, B, \hat{B}, B_2, C, \hat{C}), \psi^\times)$**:** accept iff $\hat{e}(A, B_2)/\hat{e}(C, D) = \hat{e}(g_1, \psi)$ and $\hat{e}(g_1, \hat{\psi}) = \hat{e}(\hat{g}_1, \psi)$.

**Protocol 1:** Hadamard product argument $[\![(A, \hat{A})]\!] \circ [\![(B, \hat{B}, B_2)]\!] = [\![(C, \hat{C})]\!]$ from [Lip12]

# 3 Groth-Lipmaa Arguments

In this section, we describe two of our building-blocks, an Hadamard product argument and a (known) permutation argument. In both cases, Groth [Gro10] proposed efficient (weakly) sound and non-interactive witness-indistinguishable (NIWI) arguments that were further refined by Lipmaa [Lip12], who used the theory of progression-free sets to optimize Groth's arguments. Since [Lip12] is very new, we will give here a full description of Lipmaa's NIWI arguments. We refer to [Lip12] (and its full version, [Lip11]) for details.

## 3.1 Hadamard Product Argument

Assume that $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathsf{om})$ is the knowledge commitment scheme. Recall that an Hadamard product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is equal to their entrywise product vector $\boldsymbol{c}$, that is, $c_j = a_j \cdot b_j$ for $j \in [n]$. In an *Hadamard product argument*, the prover aims to convince the verifier that for given three commitments $(A, \hat{A})$, $(B, \hat{B})$ and $(C, \hat{C})$, he knows how to open them as $(A, \hat{A}) = \mathcal{C}\mathsf{om}^1(\mathsf{ck}; \boldsymbol{a}; r_a)$, $(B, \hat{B}) = \mathcal{C}\mathsf{om}^1(\mathsf{ck}; \boldsymbol{b}; r_b)$, and $(C, \hat{C}) = \mathcal{C}\mathsf{om}^1(\mathsf{ck}; \boldsymbol{c}; r_c)$, such that $c_j = a_j \cdot b_j$ for $j \in [n]$. Prot. 1 has a full description of Lipmaa's Hadamard product argument $[\![(A, \hat{A})]\!] \circ [\![(B, \hat{B}, B_2)]\!] = [\![(C, \hat{C})]\!]$, where $B_2$ is the equivalent of $B$ in $\mathbb{G}_2$: $B_2 \leftarrow g_2^{r_b} \cdot \prod_{i=1}^n g_{2,\lambda_i}^{b_i}$.

**Fact 2 (Lipmaa [Lip12])** *The above Hadamard product argument is perfectly complete and perfectly witness-indistinguishable. If the bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is $\hat{\Lambda}$-PSDL secure, then a non-uniform PPT adversary has negligible chance of outputting $inp^\times \leftarrow (A, \hat{A}, B, \hat{B}, B_2, C, \hat{C})$ and an accepting argument $\psi^\times \leftarrow (\psi, \hat{\psi})$ together with opening witness $w^\times \leftarrow (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, \boldsymbol{c}, r_c, (f'_s)_{s \in \hat{\Lambda}})$ such that*

- $(A, \hat{A}) = \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{a}; r_a)$,
- $(B, \hat{B}) = \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{b}; r_b)$,
- $B_2 = g_2^{r_b} \cdot \prod_{i=1}^n g_{2i}^{b_i}$,
- $(C, \hat{C}) = \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{c}; r_c)$,
- $(\psi, \hat{\psi}) = (g_2^{\sum_{s \in \hat{\Lambda}} f'_s x^s}, \hat{g}_2^{\sum_{s \in \hat{\Lambda}} f'_s x^s})$,
- *and for some* $i \in [n]$, $a_i b_i \neq c_i$.

For the product argument to be useful in more complex arguments, we must also assume that the verifier there additionally verifies that $\hat{e}(A, \hat{g}_2) = \hat{e}(\hat{A}, g_2)$, $\hat{e}(B, \hat{g}_2) = \hat{e}(\hat{B}, g_2)$, $\hat{e}(g_1, B_2) = \hat{e}(B, g_2)$, and $\hat{e}(C, \hat{g}_2) = \hat{e}(\hat{C}, g_2)$. Note that $(f'_s)_{s \in \hat{\Lambda}}$ is the opening of $(\psi, \hat{\psi})$.

**Fact 3 (Lipmaa [Lip12])** *For any $n > 0$ and $y = n^{1+o(1)}$, let $\Lambda \subset [y]$ be a progression-free set of odd integers as guaranteed by Fact 1, such that $|\Lambda| = n$. The communication (argument size) of the Hadamard product argument is 2 elements from $\mathbb{G}_2$. The prover's computational complexity is $\Theta(n^2)$ scalar multiplications in $\mathbb{Z}_p$ and $n^{1+o(1)}$ exponentiations in $\mathbb{G}_2$. The verifier's computational complexity is dominated by 5 bilinear pairings. The CRS consists of $n^{1+o(1)}$ group elements.*

**System parameters:** Same as in Prot. 1, but let

$$\tilde{\Lambda} := \Lambda \cup \{2\lambda_k - \lambda_j\}_{i,k\in[n]} \cup 2\hat{\ }\Lambda \cup (\{2\lambda_k + \lambda_i - \lambda_j\}_{i,j,k\in[n]\wedge i\neq j} \setminus 2\cdot\Lambda) \enspace .$$

**CRS generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$:** Let $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Let $\hat{\alpha}, \tilde{\alpha}, x \leftarrow \mathbb{Z}_p$. Let $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$ and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. Let $\hat{g}_t \leftarrow \hat{g}_t^{\hat{\alpha}}$ and $\tilde{g}_t \leftarrow \tilde{g}_t^{\tilde{\alpha}}$ for $t \in \{1,2\}$. Denote $g_{t\ell} \leftarrow g_t^{x^\ell}$, $\hat{g}_{t\ell} \leftarrow \hat{g}_t^{x^\ell}$, and $\tilde{g}_{t\ell} \leftarrow \tilde{g}_t^{x^\ell}$ for $t \in \{1,2\}$ and $\ell \in \{0\} \cup \tilde{\Lambda}$. Let $(D, \tilde{D}) \leftarrow (\prod_{i=1}^n g_{2,\lambda_i}, \prod_{i=1}^n \tilde{g}_{2,\lambda_i})$. The CRS is

$$\mathsf{crs} \leftarrow (\mathsf{gk}; (g_{1\ell}, \hat{g}_{1\ell}, \tilde{g}_{1\ell})_{\ell\in\{0\}\cup\Lambda}, (g_{2\ell})_{\ell\in\{0\}\cup\tilde{\Lambda}}, (\hat{g}_{2\ell})_{\ell\in\hat{\Lambda}}, (\tilde{g}_{2\ell})_{\ell\in\tilde{\Lambda}}, D, \tilde{D}) \enspace .$$

Let $\widehat{\mathsf{ck}}_1 \leftarrow (\mathsf{gk}; (g_{1\ell}, \hat{g}_{1\ell})_{\ell\in\{0\}\cup\Lambda})$, $\widetilde{\mathsf{ck}}_1 \leftarrow (\mathsf{gk}; (g_{1\ell}, \tilde{g}_{1\ell})_{\ell\in\{0\}\cup\Lambda})$.

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

**Argument generation $\mathcal{P}_{\mathbf{perm}}(\mathsf{crs}; (A, \tilde{A}, B, \hat{B}, \tilde{B}, \varrho), (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b))$:**

1. Let $(T^*, \hat{T}^*, T_2^*) \leftarrow (\prod_{i=1}^n g_{1,\lambda_i}^{T_\Lambda(\varrho^{-1}(i),\varrho)}, \prod_{i=1}^n \hat{g}_{1,\lambda_i}^{T_\Lambda(\varrho^{-1}(i),\varrho)}, \prod_{i=1}^n g_{2,\lambda_i}^{T_\Lambda(\varrho^{-1}(i),\varrho)})$.
2. Let $r_{a^*} \leftarrow \mathbb{Z}_p$, $(A^*, \hat{A}^*) \leftarrow \mathcal{C}\mathsf{om}_1(\widehat{\mathsf{ck}}_1; T_\Lambda(\varrho^{-1}(1), \varrho)\cdot a_1, \dots, T_\Lambda(\varrho^{-1}(n), \varrho)\cdot a_n; r_{a^*})$. Create an argument $\psi^\times$ for $[\![(A, \hat{A})]\!] \circ [\![(T^*, \hat{T}^*, T_2^*)]\!] = [\![(A^*, \hat{A}^*)]\!]$.
3. Let $\tilde{\Lambda}'_\varrho := 2\hat{\ }\Lambda \cup (\{2\lambda_{\varrho(j)} + \lambda_i - \lambda_j : i, j \in [n] \wedge i \neq j\} \setminus 2\cdot\Lambda) \subset \{-\lambda_n + 1, \dots, 3\lambda_n\}$.
4. For $\ell \in \tilde{\Lambda}'_\varrho$, $I_1(\ell)$ as in Prot. 1, and $I_2(\ell) := \{(i,j) : i, j \in [n] \wedge j \neq i \wedge 2\lambda_{\varrho(i)} + \lambda_j \neq \lambda_i + 2\lambda_{\varrho(j)} \wedge 2\lambda_{\varrho(j)} + \lambda_i - \lambda_j = \ell\}$, set
$$\mu_{\varrho,\ell} \leftarrow \sum_{(i,j)\in I_1(\ell)} a_i^* - \sum_{(i,j)\in I_2(\ell)} b_i \enspace .$$
5. Let $(E_\varrho, \tilde{E}_\varrho) \leftarrow (\prod_{i=1}^n g_{2,2\lambda_{\varrho(i)}-\lambda_i}, \prod_{i=1}^n \tilde{g}_{2,2\lambda_{\varrho(i)}-\lambda_i})$.
6. Let $\psi^\varrho \leftarrow D^{r_a^*} \cdot E_\varrho^{-r_b} \cdot \prod_{\ell\in\tilde{\Lambda}'_\varrho} g_{2\ell}^{\mu_{\varrho,\ell}}$, $\tilde{\psi}^\varrho \leftarrow \tilde{D}^{r_a^*} \cdot \tilde{E}_\varrho^{-r_b} \cdot \prod_{\ell\in\tilde{\Lambda}'_\varrho} \tilde{g}_{2\ell}^{\mu_{\varrho,\ell}}$,

Send $\psi^{\mathrm{perm}} \leftarrow (A^*, \hat{A}^*, \psi^\times, \psi^\varrho, \tilde{\psi}^\varrho) \in \mathbb{G}_1^2 \times \mathbb{G}_2^4$ to the verifier as the argument.

**Verification $\mathcal{V}_{\mathbf{perm}}(\mathsf{crs}; (A, \tilde{A}, B, \hat{B}, \tilde{B}, \varrho), \psi^{\mathbf{perm}})$:** Let $E_\varrho$ and $(T^*, \hat{T}^*, T_2^*)$ be computed as in $\mathcal{P}_{\mathrm{perm}}$. If $\psi^\times$ verifies, $\hat{e}(A^*, D)/\hat{e}(B, E_\varrho) = \hat{e}(g_1, \psi^\varrho)$, $\hat{e}(A^*, \hat{g}_2) = \hat{e}(\hat{A}^*, g_2)$, and $\hat{e}(g_1, \tilde{\psi}^\varrho) = \hat{e}(\tilde{g}_1, \psi^\varrho)$, then $\mathcal{V}_{\mathrm{perm}}$ accepts. Otherwise, $\mathcal{V}_{\mathrm{perm}}$ rejects.

**Protocol 2:** Permutation argument $\varrho([\![(A, \tilde{A})]\!]) = [\![(B, \tilde{B})]\!]$ from [Lip12]

Finally, as noted in [Lip12], if $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ are Boolean vectors then the prover's computational complexity is $\Theta(n^2)$ scalar additions in $\mathbb{Z}_p$ and $n^{1+o(1)}$ exponentiations in $\mathbb{G}$.

### 3.2 Permutation Argument

In a *permutation argument*, the prover aims to convince the verifier that for given permutation $\varrho \in S_n$ and two commitments $(A, \tilde{A})$ and $(B, \tilde{B})$, he knows how to open them as $(A, \tilde{A}) = \mathcal{C}\mathsf{om}^1(\mathsf{ck}; \boldsymbol{a}; r_a)$ and $(B, \tilde{B}) = \mathcal{C}\mathsf{om}^1(\mathsf{ck}; \boldsymbol{b}; r_b)$, such that $b_j = a_{\varrho(j)}$ for $j \in [n]$. We denote this non-interactive argument by $\varrho([\![(A, \tilde{A})]\!]) = [\![(B, \tilde{B}, B_2)]\!]$, where $B_2$ is again the equivalent of $B$ in $\mathbb{G}_2$. As in the case of the Hadamard product argument, we describe a version of the argument due to [Lip12]. See Prot. 2.

Let $T_\Lambda(i, \varrho) := |\{j \in [n] : 2\lambda_{\varrho(i)} + \lambda_j = 2\lambda_{\varrho(j)} + \lambda_i\}|$, clearly $T_\Lambda(i, \varrho) \geq 1$. One proves that $a_{\varrho(i)} = b_i$ for $i \in [n]$ by using a subargument that shows that for separately committed $a_i^*$, $a_{\varrho(i)}^* = T_\Lambda(i, \varrho) \cdot b_i$ for $i \in [n]$. Showing in addition that $a_i^* = T_\Lambda(\varrho^{-1}(i), \varrho) \cdot a_i$ (which is equivalent to $a_{\varrho(i)}^* = T_\Lambda(i, \varrho) \cdot a_{\varrho(i)}$), one obtains that $a_{\varrho(i)} = b_i$ for $i \in [n]$. We only consider the case where $\varrho$ is fixed and thus the element $E_\varrho$ can be put to the CRS. We also use the fact that $\hat{\Lambda} \cup \tilde{\Lambda} = \{0\} \cup \tilde{\Lambda}$, where $\tilde{\Lambda}$ is defined in Prot. 2.

We denote the full permutation argument by $\varrho([\![(A, \tilde{A})]\!]) = [\![(B, \hat{B}, \tilde{B})]\!]$.

**Fact 4 (Lipmaa [Lip12])** *The above permutation argument is perfectly complete and perfectly witness-indistinguishable. If the bilinear group generator $\mathcal{G}_{\mathsf{bp}}$ is $\tilde{\Lambda}$-PSDL secure, then a non-uniform PPT adversary has negligible chance of outputting $inp^{perm} \leftarrow (A, \tilde{A}, B, \hat{B}, \tilde{B}, \varrho)$ and an accepting argument $\psi^{perm} \leftarrow (A^*, \hat{A}^*, \psi^\times, \hat{\psi}^\times, \psi^\varrho, \tilde{\psi}^\varrho)$ together with a witness*

$$w^{perm} \leftarrow (\boldsymbol{a}, r_a, \boldsymbol{b}, r_b, \boldsymbol{a^*}, r_{a^*}, (f'_{(\times,\ell)})_{\ell\in\hat{\Lambda}}, (f'_{(\varrho,\ell)})_{\ell\in\tilde{\Lambda}}) \enspace ,$$

*such that*

– $(A, \tilde{A}) = \mathcal{C}\mathsf{om}^1(\widetilde{\mathsf{ck}}_1; \boldsymbol{a}; r_a)$,
– $(B, \hat{B}) = \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{b}; r_b)$,
– $(B, \tilde{B}) = \mathcal{C}\mathsf{om}^1(\widetilde{\mathsf{ck}}_1; \boldsymbol{b}; r_b)$,
– $(A^*, \hat{A}^*) = \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{a^*}; r_{a^*})$,
– $(\psi^\times, \hat{\psi}^\times) = (g_2^{\sum_{\ell \in \hat{\Lambda}} f'_{(\times,\ell)}}, \hat{g}_2^{\sum_{\ell \in \hat{\Lambda}} f'_{(\times,\ell)}})$,
– $(\psi^\varrho, \hat{\psi}^\varrho) = (g_2^{\sum_{\ell \in \tilde{\Lambda}} f'_{(\varrho,\ell)}}, \tilde{g}_2^{\sum_{\ell \in \tilde{\Lambda}} f'_{(\varrho,\ell)}})$,
– $a_i^* = T_\Lambda(\varrho^{-1}(i), \varrho) \cdot a_i$ *(for* $i \in [n]$*), and*
– *for some* $i \in [n]$*,* $a_{\varrho(i)} \neq b_i$.

For the permutation argument to be useful in more complex arguments, we must also assume that the verifier there verifies that $\hat{e}(\tilde{A}, g_2) = \hat{e}(A, \tilde{g}_2)$, $\hat{e}(\hat{B}, g_2) = \hat{e}(B, \hat{g}_2)$, and $\hat{e}(\tilde{B}, g_2) = \hat{e}(B, \tilde{g}_2)$.

**Fact 5 (Lipmaa [Lip12])** *The permutation argument has a common reference string of length* $n^{1+o(1)}$ *and communication of* 4 *group elements. The prover's computational complexity is* $\Theta(n^2)$ *scalar additions in* $\mathbb{Z}_p$ *and* $n^{1+o(1)}$ *exponentiations in* $\mathbb{G}$*. The verifier's computational complexity is dominated by* 12 *bilinear pairings.*

## 4 Breaking the COCOON 2009 Range Proof

In [YHM+09], the authors proposed a non-interactive range proof. In what follows, we show that their argument is not secure.

Their goal is to prove that a committed secret $w$ is in some range $[a, b]$. To do so they prove that both $w - a$ and $b - w$ are non-negative by making use of Lagrange theorem stating that any non-negative integer can be decomposed as the sum of four squares. Hence,

$$w - a = \sum_{j=1}^{4} w_{1j}^2 \quad \text{and} \quad b - w = \sum_{j=1}^{4} w_{2j}^2 \ , \tag{1}$$

for some $w_{ij}$. The range proof of [YHM+09] is based on (symmetric) bilinear groups of composite order, that is, on bilinear groups $(n, \mathbb{G}, \mathbb{G}_T, \hat{e})$, where $n = pq$. To commit to a message $w$, the committer picks a random[1] $r \in \mathbb{Z}_q$ and computes $C = g^w u^r$, where $g$ is a random generator of $\mathbb{G}$ (of order $n$), and $u$ is a random generator of subgroup $\mathbb{G}_q$ (of order $q$). Given $C$, $w$ is uniquely determined in $\mathbb{Z}_p$, as $C^q = g^{wq}$.

In their range proof, the prover finds the witnesses $w_{ij}$ in Eq. (1) and outputs a proof

$$\psi = (\{C_{1j}, C_{2j}\}_{j \in [4]}, C_w, \varphi_1, \varphi_2) \ ,$$

where

$$\begin{aligned} C_w &\equiv g^w u^{r_w} \in \mathbb{G} \ , \\ C_{ij} &\equiv g^{w_{ij}} u^{r_{ij}} \in \mathbb{G} \text{ for } i \in [2] \text{ and } j \in [4] \ , \\ \varphi_1 &\equiv g^{-r_w + 2\sum_{j=1}^4 r_{1j} w_{1j}} \cdot u^{\sum_{j=1}^4 r_{1j}^2} \in \mathbb{G} \ , \\ \varphi_2 &\equiv g^{r_w + 2\sum_{j=1}^4 r_{2j} w_{2j}} \cdot u^{\sum_{j=1}^4 r_{2j}^2} \in \mathbb{G} \ . \end{aligned}$$

The verifier checks if

$$\hat{e}(g^a C_w^{-1}, g) \prod_{j=1}^{4} e(C_{1j}, C_{1j}) = \hat{e}(u, \varphi_1)$$

and

$$\hat{e}(C_w g^{-b}, g) \prod_{j=1}^{4} e(C_{2j}, C_{2j}) = \hat{e}(u, \varphi_2) \ .$$

Now assume that a malicious prover $P^\star$ picks an integer $w^* \in \{0, \ldots, p-1\} \setminus [a, b]$. We have that either $w^* - a$ or $b - w^*$ is negative as an integer. Suppose $b - w^* < 0$, then $P^\star$ chooses $\{w_{2j}^*\}_{j \in [4]}$

[1] In [YHM+09], the scheme uses $r \in \mathbb{Z}_n$ to facilitate their security proof (crs switching).

such that $n + (b - w^*) = \sum_{j=1}^4 (w^*_{2j})^2$, sets $C_w \leftarrow g^{w^*} u^{r_w}$, $C_{2j} \leftarrow g^{w^*_{2j}} u^{r_{2j}}$, $\varphi_1$ as above, and $\varphi_2 \leftarrow g^{r_w + 2\cdot\sum_{j=1}^4 r_{2j} w^*_{2j}} \cdot u^{\sum_{j=1}^4 r_{2j}^2}$. Let $u = g^\alpha$ for some $\alpha$. It is easy to see that the second verification equation still holds:

$$\begin{aligned}\hat{e}(C_w g^{-b}, g)\prod_{j=1}^4 \hat{e}(C_{2j}, C_{2j}) =& \hat{e}(g,g)^{(w^*-b)+\alpha r_w + \sum_{j=1}^4 (w^*_{2j}+\alpha r_{2j})^2} \\ =& \hat{e}(g,g)^{(w^*-b)+\alpha r_w + \sum_{j=1}^4 (w^*_{2j})^2 + \sum_{j=1}^4 \alpha^2 r_{2j}^2 + 2\sum_{j=1}^4 \alpha r_{2j} w^*_{2j}} \\ =& \hat{e}(g,g)^{\alpha\cdot(r_w + 2\sum_{j=1}^4 r_{2j} w^*_{2j} + \alpha\cdot\sum_{j=1}^4 r_{2j}^2)} = \hat{e}(u, \varphi_2) \enspace .\end{aligned}$$

We have successfully constructed a polynomial time adversary who can always break the scheme. Therefore, the NIZK range proof in [YHM+09] is not sound.

## 5 New Subargument for Correct Encryption

In the new range proof of Sect. 6, we need a subargument that if $(A_c, \bar{A}_c)$ is a knowledge-commitment of some $a$ (with $n = 1$ and some randomness $r$), and $(A_g, A_f, A_h)$ is a BBS ciphertext of some $a'$, then $a = a'$. That is, $A_c = g_1^r g_{1,\lambda_1}^a$ and $(A_g, A_f, A_h) = (g_1^{r_f + r_h + a}, f^{r_f}, h^{r_h})$ for randomness $(r_f, r_h)$ and public key $(f, h)$. (The generator $g_{1,\lambda_1}$ is required in Sect. 6.)

We will construct this argument in the current section, by combining ideas from [GS08] and [Gro10,Lip12]. Intuitively, for every multi-exponentiation $h_1^{a_1} \dots h_m^{a_m} = t$ that we want to prove, we write down a verification equation $\hat{e}(h_1, \mathcal{C}\mathsf{om}(a_1)) \cdot \dots \cdot \hat{e}(h_m, \mathcal{C}\mathsf{om}(a_m)) = \hat{e}(\psi, g_2)\hat{e}(t, \mathcal{C}\mathsf{om}(1))$, where $\psi$ "compensates" for the fact that $\mathcal{C}\mathsf{om}(a_m)$ are probabilistic commitments. In addition, we use knowledge commitments (though for small values 0 or 1 of $n$) so that one can extract all committed values. Since the argument uses three committed values ($a$, $r_f$ and $r_h$) and three equations, according to Fig. 6 of [GS07] (the full version of [GS08]), the corresponding pure Groth-Sahai argument will have length of 15 group elements. Our combination argument has the same length, but is computationally more efficient.

**System parameters:** An $(n, \kappa)$-nice tuple $\Lambda = (\lambda_1, \dots, \lambda_n)$.
**Common reference string generation $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$:** Set

$$\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, \hat{e}) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa) \enspace .$$

Generate random $\alpha_g, \alpha_f, \alpha_h, \bar{\alpha}, \alpha_{g/c}, x \leftarrow \mathbb{Z}_p$. Let $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$ and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. Denote $g_{1,\lambda_1} \leftarrow g_1^{x^{\lambda_1}}$, $g_{2,\lambda_1} \leftarrow g_2^{x^{\lambda_1}}$, $\mathring{g}_1 \leftarrow g_1^{\alpha_g}$, $\mathring{g}_2 \leftarrow g_2^{\alpha_g}$, $\bar{g}_1 \leftarrow g_1^{\bar{\alpha}}$, $\bar{g}_{1,\lambda_1} \leftarrow g_{1,\lambda_1}^{\bar{\alpha}}$, $\bar{g}_2 \leftarrow g_2^{\bar{\alpha}}$, $\bar{g}_{2,\lambda_1} \leftarrow g_{2,\lambda_1}^{\bar{\alpha}}$, $\mathring{g}_{1,g/c} \leftarrow g_1^{\alpha_{g/c}\cdot(1-x^{\lambda_1})}$, $\mathring{g}_{2,g/c} \leftarrow g_2^{\alpha_{g/c}\cdot(1-x^{\lambda_1})}$, $\mathring{g}_{1,f} \leftarrow g_1^{\alpha_f}$, $\mathring{g}_{2,f} \leftarrow g_2^{\alpha_f}$, $\mathring{g}_{1,h} \leftarrow g_1^{\alpha_h}$, and $\mathring{g}_{2,h} \leftarrow g_2^{\alpha_h}$. The common reference string is

$$\mathsf{crs} \leftarrow (\mathsf{gk}; g_1, g_{1,\lambda_1}, g_2, g_{2,\lambda_1}, \mathring{g}_1, \mathring{g}_2, \bar{g}_1, \bar{g}_{1,\lambda_1}, \bar{g}_2, \bar{g}_{2,\lambda_1}, \mathring{g}_{1,g/c}, \mathring{g}_{2,g/c}, \mathring{g}_{1,f}, \mathring{g}_{2,f}, \mathring{g}_{1,h}, \mathring{g}_{2,h}) \enspace .$$

A third party also creates $\mathsf{sk} := (\mathsf{sk}_1, \mathsf{sk}_2) \leftarrow (\mathbb{Z}_p^*)^2$, and sets

$$\mathsf{pk} := (f, h, \mathring{f}, \mathring{h}) \leftarrow (g_1^{1/\mathsf{sk}_1}, g_1^{1/\mathsf{sk}_2}, \mathring{g}_{1,f}^{1/\mathsf{sk}_1}, \mathring{g}_{1,h}^{1/\mathsf{sk}_2}) \enspace .$$

**Common inputs:** $(\mathsf{crs}; \mathsf{pk}, A_g, A_f, A_h, A_c)$, where $\mathsf{pk} = (f, h, \mathring{f}, \mathring{h})$,

$$(A_g, A_f, A_h) = (g_1^{r_f + r_h + a}, f^{r_f}, h^{r_h}) \enspace ,$$

and $A_c = g_1^{r_f + r_h} g_{1,\lambda_1}^a$.
**Argument $\mathcal{P}(\mathsf{crs}; (A_f, A_g, A_h, A_c), (a, r_f, r_h))$:** let $\bar{A}_c \leftarrow \bar{g}_1^{r_f + r_h} \bar{g}_{1,\lambda_1}^a$,

$$(\mathring{A}_f, \mathring{A}_f, \mathring{A}_h) \leftarrow (\mathring{g}_1^{r_f + r_h + a}, \mathring{f}^{r_f}, \mathring{h}^{r_h}) \enspace ,$$

$\mathring{A}_{g/c} \leftarrow \mathring{g}_{1,g/c}^a$. Let $R_f, R_h \leftarrow \mathbb{Z}_p$. Let $(C_f, \bar{C}_f) \leftarrow (g_2^{R_f} g_{2,\lambda_1}^{r_f}, \bar{g}_2^{R_f} \bar{g}_{2,\lambda_1}^{r_f})$,

$$(C_h, \bar{C}_h) \leftarrow (g_2^{R_h} g_{2,\lambda_1}^{r_h}, \bar{g}_2^{R_h} \bar{g}_{2,\lambda_1}^{r_h}) \in \mathbb{G}_2^2 \enspace .$$

Let

$$(\psi_g, \mathring{\psi}_g) \leftarrow (g_1^{r+R_f+R_h}, \mathring{g}_1^{r+R_f+R_h}) \in \mathbb{G}_1^2 \enspace ,$$

$(\psi_f, \mathring{\psi}_f) \leftarrow (f^{R_f}, \mathring{f}^{R_f}) \in \mathbb{G}_1^2$, $(\psi_h, \mathring{\psi}_h) \leftarrow (h^{R_h}, \mathring{h}^{R_h}) \in \mathbb{G}_1^2$.

Send $\psi^{ce} \leftarrow (\mathring{A}_g, \mathring{A}_f, \mathring{A}_h, \mathring{A}_c, \psi_g, \mathring{\psi}_g, C_f, \bar{C}_f, \psi_f, \mathring{\psi}_f, C_h, \bar{C}_h, \psi_h, \mathring{\psi}_h, \mathring{A}_{g/c})$ to the verifier.

**Verification $\mathcal{V}(\mathsf{crs}; (A_f, A_g, A_h, A_c), \psi^{ce})$:** Verify that $\hat{e}(\mathring{f}, g_2) = \hat{e}(f, \mathring{g}_{2,f})$, $\hat{e}(\mathring{h}, g_2) = \hat{e}(h, \mathring{g}_{2,h})$, $\hat{e}(A_g, \mathring{g}_2) = \hat{e}(\mathring{A}_g, g_2)$, $\hat{e}(A_f, \mathring{g}_{2,f}) = \hat{e}(\mathring{A}_f, g_2)$, $\hat{e}(A_h, \mathring{g}_{2,h}) = \hat{e}(\mathring{A}_h, g_2)$, $\hat{e}(A_c, \bar{g}_2) = \hat{e}(\bar{A}_c, g_2)$, $\hat{e}(\psi_g, \mathring{g}_2) = \hat{e}(\mathring{\psi}_g, g_2)$, $\hat{e}(\psi_f, \mathring{g}_{2,f}) = \hat{e}(\mathring{\psi}_f, g_2)$, $\hat{e}(\psi_h, \mathring{g}_{2,h}) = \hat{e}(\mathring{\psi}_h, g_2)$, $\hat{e}(\bar{g}_1, C_f) = \hat{e}(g_1, \bar{C}_f)$, $\hat{e}(\bar{g}_1, C_h) = \hat{e}(g_1, \bar{C}_h)$, and $\hat{e}(A_g/A_c, \mathring{g}_{2,g/c}) = \hat{e}(\mathring{A}_{g/c}, g_2)$.

Verify that $\hat{e}(f, C_f) = \hat{e}(\psi_f, g_2) \cdot \hat{e}(A_f, g_{2,\lambda_1})$, $\hat{e}(h, C_h) = \hat{e}(\psi_h, g_2) \cdot \hat{e}(A_h, g_{2,\lambda_1})$, and $\hat{e}(g_1, C_f C_h) = \hat{e}(\psi_g A_c^{-1}, g_2) \cdot \hat{e}(A_g, g_{2,\lambda_1})$.

**Theorem 1.** *The argument of this subsection is a perfectly argument that for some $a, r_f, r_h \in \mathbb{Z}_p$, $A_c = g_1^r g_{1,\lambda_1}^a$ and $(A_g, A_f, A_h) = (g^{r_f+r_h+a}, f^{r_f}, h^{r_h})$. If the $\{\lambda_1\}$-PSDL assumption and the $\{\lambda_1\}$-PKE assumption (in both $\mathbb{G}_1$ and $\mathbb{G}_2$) hold, then this argument is computationally sound. If the DLIN assumption holds in group $\mathbb{G}_1$, then this argument is computationally zero-knowledge.*

Clearly, this argument has CRS of length $\Theta(1)$, its argument consists of 13 elements of $\mathbb{G}_1$ and 2 elements of $\mathbb{G}_2$. The prover's computational complexity is dominated by 20 exponentiations. The verifier's computational complexity is dominated by 33 pairings.

## 6 New Range Proof

In the next range proof, the prover has an encrypted $a \in \mathbb{Z}_p$, and he aims to convince the verifier that $a \in [0, H]$. We will use the lifted BBS cryptosystem $(\mathcal{G}_{\mathsf{pkc}}, \mathcal{E}\mathsf{nc}, \mathcal{D}\mathsf{ec})$ that can be thought of as a perfectly binding commitment scheme if decryption is not necessary. Since we are interested in obtaining a sublinear argument, we will also use the (computationally binding) knowledge commitment scheme $(\mathcal{G}_{\mathsf{com}}, \mathcal{C}\mathsf{om})$. We use the following result that was stated for $u = 2$ in [LAN02] and for general $u$ in [CLs10].

**Fact 6** *Let $H > 0$ and $u > 1$. Let $\ell(u, H) \le \log_u(H+1)$ be defined as in [CLs10]. Then $a \in [0, H]$ if and only if for some $b_i \in [0, u-1]$,*

$$(u-1)a = \sum_{i=1}^{\ell(u,(u-1)H)} G_i b_i \enspace ,$$

*where $G_i \in \mathbb{Z}$ are values defined in [CLs10]. That is, $(u-1) \cdot [0, H] = \sum_{i=1}^{\ell(u,(u-1)H)} G_i \cdot [0, u-1]$. In particular, $[0, H] = \sum_{i=0}^{\lfloor \log_2 H \rfloor} \lfloor (H + 2^i)/2^{i+1} \rfloor \cdot [0, 1]$.*

The precise values of $\ell(u, H)$ and $G_i$ are not important in the next description. It suffices to know that they can be efficiently evaluated. We note that

$$G_i = \lfloor H/u^{i+1} \rfloor + \lfloor (H_i + (\sum_{j=0}^{i-1} H_j \mod (u-1)) + 1)/u \rfloor \enspace ,$$

where $H = \sum 2^i H_i$ [CLs10].

The basic idea of the next range proof is as follows. Choose a $u > 1$, and let $n = \ell(u, (u-1)H)$. According to Fact 6, $a \in [H]$ iff for $G_i$ computed as in Fact 6, one has $(u-1)a = \sum_{i=1}^n G_i b_i$ for some $b_i \in [u-1]$. The prover shows by using a parallel version of range proof from [LAN02] that for $i \in [n]$, $b_i \in [0, u-1]$. The latter is done by writing $b_i$ as $b_i = \sum_{j=0}^{\lfloor \log_2(u-1) \rfloor} G'_j b'_{ji}$ (by again using Fact 6) and then showing that $b'_{ji} \in [0, 1]$ by using an Hadamard product arguments from [Lip12]. This will be achieved with commitments on $(b'_{j1}, \dots, b'_{jn})$ for $j \in [\lfloor \log_2(u-1) \rfloor]$.

The prover then commits to the vector $(c_1, \dots, c_n)$, where $c_j = \sum_{i=j}^n G_i b_i$, and shows that the values $c_j$ are correctly computed by using a small constant number of Hadamard product and permutation arguments. More precisely, he commits to $(G_1 b_1, \dots, G_n b_n)$ (and shows this has been done correctly), then to $(c_2, \ldots, c_n, c_1)$ (and shows this was done correctly), then to $(c_2, \ldots, c_n, 0)$ (and shows this was done correctly), and then shows that

$$(c_1, \ldots, c_n) = (G_1 b_1, \ldots, G_n b_n) + (c_2, \ldots, c_n, 0) \enspace .$$

Thus, the verifier is convinced that $c_j = \sum_{i=j}^{n} G_i b_i$. But then by Fact 6, $c_1 = \sum_{i=1}^{n} G_i b_i \in (u-1) \cdot [H]$, and thus the prover has to show, by using a single product argument, that $(A_c^{u-1}, \hat{A}_c^{u-1})$ commits to $(c_1, 0, \ldots, 0)$ and that $(A_g, A_f, A_h)$ is a lifted BBS encryption of $A$ with randomizer $(r_f, r_h)$ where $r = r_f + r_h$.

As in [Lip12], in a few cases, instead of computing two different commitments $\mathcal{C}\mathsf{om}^t(\widehat{\mathsf{ck}}_t; \boldsymbol{a}; r) = (g_t^r \cdot \prod g_{t,\lambda_i}^{a_i}, \hat{g}_t^r \cdot \prod \hat{g}_{t\lambda_i}^{a_i})$ and $\mathcal{C}\mathsf{om}^t(\widetilde{\mathsf{ck}}_t; \boldsymbol{a}; r) = (g_t^r \cdot \prod g_{t,\lambda_i}^{a_i}, \tilde{g}_t^r \cdot \prod \tilde{g}_{t,\lambda_i}^{a_i})$, we compute a composed commitment

$$\mathcal{C}\mathsf{om}^t(\mathsf{ck}_t; \boldsymbol{a}; r) = (g_t^r \cdot \prod g_{t,\lambda_i}^{a_i}, \hat{g}_t^r \prod \hat{g}_{t,\lambda_i}^{a_i}, \tilde{g}_t^r \cdot \prod \tilde{g}_{t,\lambda_i}^{a_i}) \enspace .$$

The common input to both parties is equal to a BBS encryption $(A_g, A_f, A_h)$ of $a$, accompanied by a knowledge component $\hat{A}$ such that $(A, \hat{A})$ is at the same time a knowledge commitment to $a$.

**Theorem 2.** *Let $u > 1$. Let $H = \mathrm{poly}(\kappa)$ and $n = \ell(u, (u-1)H)$ where $\ell$ is defined as in Fact 6. Let $\Lambda = \{\lambda_i\}_{i \in [n]}$ be an $(n, \kappa)$-nice tuple. Denote $\lambda_0 := 0$. Let $\widehat{\Lambda} := \{0\} \cup \Lambda \cup 2\hat{}\Lambda$, and $\tilde{\Lambda}$ as in Sect. 3.2. Let $\mathsf{rot} \in S_n$ be a permutation, where $\mathsf{rot}(i) = i - 1$ if $i > 1$, and $\mathsf{rot}(1) = n$. Define $G_i$ as in Fact 6. The argument in Prot. 3 is perfectly complete. If $\mathcal{G}_{\mathsf{bp}}$ is $\Lambda$-PKE secure and DLIN secure in $\mathbb{G}_1$, then the argument in Prot. 3 is computationally zero-knowledge. If $\mathcal{G}_{\mathsf{bp}}$ is $\tilde{\Lambda}$-PSDL secure and $\Lambda$-PKE secure in both $\mathbb{G}_1$ and $\mathbb{G}_2$, then the argument in Prot. 3 is computationally sound.*

This argument is computationally zero-knowledge because $(A_c, \hat{c}_c)$ that was provided by a prover and not generated during the argument. To achieve zero-knowledge, one must be able to open $(A_c, \hat{c})$ given only the CRS trapdoor. That is, one has to use an extractable commitment scheme [Di 02,ACP09]. It is easy to see that the knowledge commitment scheme is extractable, however, extractability is only achieved under the PKE assumption. The use of a cryptosystem also makes achieving perfect zero-knowledge impossible.

**Theorem 3.** *Let $u > 1$. Let $\Lambda$ be as in Fact 1 and let $n = \ell(u, (u-1)H) \le \lfloor \log_u((u-1)H + 1) \rfloor \approx \log H / \log u + 1$, where $\ell(\cdot, \cdot)$ is defined as in Fact 6. Let $n_v = \lceil \log_2(u-1) \rceil$. Assume that we use the Hadamard product argument and the permutation argument from Sect. 3. The range proof in Prot. 3 has a length-$n^{1+o(1)}$ common reference string, communication of $2n_v + 25$ elements from $\mathbb{G}_1$ and $3n_v + 15$ elements from $\mathbb{G}_2$, the prover's computational complexity of $\Theta(n^2 n_v)$ scalar multiplications in $\mathbb{Z}_p$ and $n^{1+o(1)} n_v$ exponentiations in $\mathbb{G}_1$ or $\mathbb{G}_2$. The verifier's computational complexity is dominated by $9n_v + 81$ pairings.*

The communication complexity is minimized when $n_v$ (and thus $u$) is as small as possible, that is, $u = 2$. Then $n_v = \lfloor \log_2 1 \rfloor = 0$. In this case the communication consists of 12 elements from $\mathbb{G}_1$ and 13 elements from $\mathbb{G}_2$. The same choice $u = 2$ is also optimal for verifier's computational complexity (81 pairings). As noted before, at the security level of $2^{128}$, elements of $\mathbb{G}_1$ can be represented in 256 bits, and elements of $\mathbb{G}_2$ in 512 bits. Thus, at this security level, if $u = 2$ then the communication is $25 \cdot 256 + 25 \cdot 512 = 14\,080$ bits, that is, only about 4 to 5 times longer than the current recommended length of a $2^{128}$-secure RSA modulus. Therefore, the communication of the new range proof is even smaller than that of Lagrange theorem based arguments like [Lip03].

The optimal prover's computational complexity is achieved when the number of exponentiations, $n^{1+o(1)} \cdot n_v = (\log H / \log u)^{1+o(1)} \cdot \lfloor \log_2(u-1) \rfloor$, is minimized. This happens if $u = H$, then the prover's computation is dominated by $\Theta(\log H)$ scalar multiplications and exponentiations. Moreover, in this case the CRS length $n^{1+o(1)}$ is constant. Finally, we might want the summatory length of the CRS and the communication to be minimal, that is, $n^{1+o(1)} + \Theta(n_v)$. Considering $n \approx \log_u H$ and $n_v \approx \log_2 u$, we get that the sum is $(\log H / \log u)^{1+o(1)} + \Theta(\log u)$. One can approximately minimize the latter by choosing $u = e^{\sqrt{\ln H}}$. Then the summatory length is $\log^{1/2+o(1)} H$. (In this case, it would make sense to change the role of groups $\mathbb{G}_1$ and $\mathbb{G}_2$ to get better efficiency.) The efficiency of the new range proof in all three cases is given in Tbl. 1.

**System parameters:** $H, G_i, n, u$, $n_v := \lfloor \log_2(u-1) \rfloor$, and $G'_j := \lfloor (u+2^j)/2^{j+1} \rfloor$.

**Common reference string generation** $\mathcal{G}_{\mathsf{crs}}(1^\kappa)$**:** Set $\mathsf{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{\mathsf{bp}}(1^\kappa)$. Generate random $\widehat{\alpha}, \tilde{\alpha}, \alpha_g, \alpha_f, \alpha_h, \bar{\alpha}, \alpha_{g/c}, x \leftarrow \mathbb{Z}_p$. Let $g_1 \leftarrow \mathbb{G}_1 \setminus \{1\}$ and $g_2 \leftarrow \mathbb{G}_2 \setminus \{1\}$. Denote $g_{ts} \leftarrow g_t^{x^s}$, $\hat{g}_{ts} \leftarrow g_t^{\widehat{\alpha} x^s}$, $\tilde{g}_{ts} \leftarrow g_t^{\tilde{\alpha} x^s}$, $\mathring{g}_1 \leftarrow g_1^{\alpha_g}$, $\mathring{g}_2 \leftarrow g_2^{\alpha_g}$, $\bar{g}_1 \leftarrow g_1^{\bar{\alpha}}$, $\bar{g}_{1,\lambda_1} \leftarrow g_{1,\lambda_1}^{\bar{\alpha}}$, $\bar{g}_2 \leftarrow g_2^{\bar{\alpha}}$, $\bar{g}_{2,\lambda_1} \leftarrow g_{2,\lambda_1}^{\bar{\alpha}}$, $\mathring{g}_{1,g/c} \leftarrow g_1^{\alpha_{g/c} \cdot (1 - x^{\lambda_1})}$, $\mathring{g}_{2,g/c} \leftarrow g_2^{\alpha_{g/c} \cdot (1 - x^{\lambda_1})}$, $\mathring{g}_{1,f} \leftarrow g_1^{\alpha_f}$, $\mathring{g}_{2,f} \leftarrow g_2^{\alpha_f}$, $\mathring{g}_{1,h} \leftarrow g_1^{\alpha_h}$, and $\mathring{g}_{2,h} \leftarrow g_2^{\alpha_h}$. Set $D \leftarrow \prod_{=1}^n g_{1,\lambda_i}$, $E_{\mathsf{rot}} \leftarrow \prod_{i=1}^n g_{2,2\lambda_{\mathsf{rot}(i)} - \lambda_i}$, and $\widetilde{E}_{\mathsf{rot}} \leftarrow E_{\mathsf{rot}}^{\tilde{\alpha}}$. The common reference string is

$$\mathsf{crs} \leftarrow (\mathsf{gk}; (g_{1,s}, \hat{g}_{1,s}, \tilde{g}_{1,s})_{s \in \{0\} \cup \Lambda}, g_2, (\hat{g}_{2,s})_{s \in \widehat{\Lambda}}, (g_{2,s}, \tilde{g}_{2,s})_{s \in \tilde{\Lambda}}, D, E_{\mathsf{rot}}, \widetilde{E}_{\mathsf{rot}}) \ .$$

Set $\mathsf{ck}_1 \leftarrow (\mathsf{gk}; (g_{1s}, \hat{g}_{1s}, \tilde{g}_{1s})_{s \in \{0\} \cup \Lambda})$, $\widehat{\mathsf{ck}}_1 \leftarrow (\mathsf{gk}; (g_{1s}, \hat{g}_{1s})_{s \in \{0\} \cup \Lambda})$ and $\widetilde{\mathsf{ck}}_1 \leftarrow (\mathsf{gk}; (g_{1s}, \tilde{g}_{1s})_{s \in \{0\} \cup \Lambda})$. The prover creates a secret key $\mathsf{sk} := (\mathsf{sk}_1, \mathsf{sk}_2) \leftarrow \mathbb{Z}_p^2$, and sets $\mathsf{pk} \leftarrow (f, h, \mathring{f}, \mathring{h}) \leftarrow (g_1^{1/\mathsf{sk}_1}, g_1^{1/\mathsf{sk}_2} \mathring{g}_{1,f}^{1/\mathsf{sk}_1}, \mathring{g}_{1,h}^{1/\mathsf{sk}_2})$. Here, $\mathcal{E}\mathsf{nc}_{\mathsf{pk}}(m; (r_f, r_h)) := (g_1^{r_f + r_h + m}, f^{r_f}, h^{r_h})$.

**Common inputs:** $(\mathsf{pk}, A_g, A_f, A_h, A_c, \hat{A}_c)$, where $(A_g, A_f, A_h) = (g_1^{r+a}, f^{r_f}, h^{r_h})$ and $(A_c, \hat{A}_c) = g_1^r g_{1,\lambda_1}^a, \hat{g}_1^r \hat{g}_{1,\lambda_1}^a)$, for $r = r_f + r_h$.

**Argument** $\mathcal{P}(\mathsf{crs}; (\mathsf{pk}, A_g, A_f, A_h, A_c, \hat{A}_c), (a, r_f, r_h))$**:** The prover does the following:

1. Compute $(b_1, \dots, b_n) \in \mathbb{Z}_u^n$ such that $(u-1)a = \sum_{i=1}^n G_i b_i$.
2. For $i \in [n]$ do: compute $(b'_{0i}, \dots, b'_{n_v,i}) \in \mathbb{Z}_2^{n_v+1}$ such that $b_i = \sum_{j=0}^{n_v} G'_j \cdot b'_{ji}$.
3. For $j \in [0, n_v]$ do:
   - Let $r_j \leftarrow \mathbb{Z}_p$, $(B'_j, \hat{B}'_j) \leftarrow \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; b'_{j1}, \dots, b'_{jn}; r_j)$, $B'_{j2} \leftarrow g_2^{r_j} \cdot \prod_{i=1}^n g_{2,\lambda_i}^{b'_{ji}}$.
   - Create an argument $(\psi'_j, \hat{\psi}'_j)$ for $[\![(B'_j, \hat{B}'_j)]\!] \circ [\![(B'_j, \hat{B}'_j, B'_{j2})]\!] = [\![(B'_j, \hat{B}'_j)]\!]$.
4. For $i \in [n]$, let $c_i \leftarrow \sum_{k=i}^n G_k b_k$.
5. Set $r'_0, r'_1, r'_2 \leftarrow \mathbb{Z}_p$, $(B^\dagger, \hat{B}^\dagger) \leftarrow \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; G_1 b_1, \dots, G_n b_n; r'_0)$, $(C, \hat{C}, \tilde{C}) \leftarrow \mathcal{C}\mathsf{om}^1(\mathsf{ck}_1; \boldsymbol{c}; r'_1)$, and $(C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}}, \tilde{C}_{\mathsf{rot}}) \leftarrow \mathcal{C}\mathsf{om}^1(\mathsf{ck}_1; c_2, \dots, c_{n-1}, c_n, c_1; r'_2)$.
6. Create an argument $(\psi_1^\times, \hat{\psi}_1^\times)$ for $[\![(\prod_{j=0}^{n_v} (B'_j)^{G'_j}, \prod_{j=0}^{n_v} (\hat{B}'_j)^{G'_j})]\!] \circ [\![(\mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; G_1, \dots, G_n; 0), \prod_{i=1}^n g_{2,\lambda_i}^{G_i})]\!] = [\![(B^\dagger, \hat{B}^\dagger)]\!]$.
7. Create an argument $(A^*, \hat{A}^*, \psi_2^\times, \hat{\psi}_2^\times, \psi_2^{\mathsf{rot}}, \hat{\psi}_2^{\mathsf{rot}})$ for $\mathsf{rot}([\![(C, \tilde{C})]\!]) = [\![(C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}}, \tilde{C}_{\mathsf{rot}})]\!]$.
8. Create an argument $(\psi_3^\times, \hat{\psi}_3^\times)$ for $[\![(C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}})]\!] \circ [\![(\mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; 1, 1, \dots, 1, 0; 0), \prod_{i=1}^{n-1} g_{2,\lambda_i})]\!] = [\![(C/B^\dagger, \hat{C}/\hat{B}^\dagger)]\!]$.
9. Create an argument $(\psi_4^\times, \hat{\psi}_4^\times)$ for $[\![(C, \hat{C})]\!] \circ [\![(\mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; 1, 0, \dots, 0, 0; 0), g_{2,\lambda_1})]\!] = [\![(A_c^{u-1}, \hat{A}_c^{u-1})]\!]$.
10. Create an argument $\psi_5^{ce}$ that $A_c$ commits to the same value that $(A_g, A_f, A_h)$ encrypts.
11. Send

$$\begin{aligned} \psi \leftarrow & ((B'_j, \hat{B}'_j, B'_{j2}, \psi'_j, \hat{\psi}'_j)_{j \in [0, n_v]}, (B^\dagger, \hat{B}^\dagger), (C, \hat{C}, \tilde{C}), (C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}}, \tilde{C}_{\mathsf{rot}}), (\psi_1^\times, \hat{\psi}_1^\times), \\ & (A^*, \hat{A}^*, \psi_2^\times, \hat{\psi}_2^\times, \psi_2^{\mathsf{rot}}, \hat{\psi}_2^{\mathsf{rot}}), (\psi_3^\times, \hat{\psi}_3^\times), (\psi_4^\times, \hat{\psi}_4^\times), \psi_5^{ce}) \end{aligned}$$

    to $\mathcal{V}$.

**Verification** $\mathcal{V}(\mathsf{crs}; (\mathsf{pk}, A_g, A_f, A_h, A_c, \hat{A}_c), \psi)$**:** $\mathcal{V}$ does the following.

1. For $j \in [0, n_v]$ do:
   (a) Check that $\hat{e}(B'_j, g_2) = \hat{e}(g_1, B'_{j2})$ and $\hat{e}(B'_j, \hat{g}_2) = \hat{e}(\hat{B}'_j, g_2)$.
   (b) Verify $(\psi'_j, \hat{\psi}'_j)$ for inputs as specified above.
2. For $K \in \{A_c, B^\dagger, C, C_{\mathsf{rot}}\}$: check that $\hat{e}(K, \hat{g}_2) = \hat{e}(\hat{K}, g_2)$.
3. For $K \in \{C, C_{\mathsf{rot}}\}$: check that $\hat{e}(K, \tilde{g}_2) = \hat{e}(\tilde{K}, g_2)$.
4. Verify the arguments $(\psi_1^\times, \hat{\psi}_1^\times), (A^*, \hat{A}^*, \psi_2^\times, \hat{\psi}_2^\times, \psi_2^{\mathsf{rot}}, \hat{\psi}_2^{\mathsf{rot}}), (\psi_3^\times, \hat{\psi}_3^\times), (\psi_4^\times, \hat{\psi}_4^\times), \psi_5^{ce}$ for inputs as specified above.

**Protocol 3:** The new range proof for some $u > 1$.

**Acknowledgments.** The first author was supported by European Social Fund's Doctoral Studies and Internationalization Programme DoRa. The second and the third authors were supported by Estonian Science Foundation, grant #8058, and European Union through the European Regional Development Fund.

## References


ACP09. Michel Abdalla, Céline Chevalier, and David Pointcheval. Smooth Projective Hashing for Conditionally Extractable Commitments. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 671–689, Santa Barbara, California, USA, August 16–20, 2009. Springer, Heidelberg. 2, 6

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, Heidelberg. 1, 2, 2

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, Heidelberg. 2

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, Heidelberg. 1

CDS94. Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In Yvo G. Desmedt, editor, *CRYPTO 1994*, volume 839 of *LNCS*, pages 174–187, Santa Barbara, USA, August 21–25 1994. Springer, Heidelberg. 1

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, Heidelberg. 1

CHL05. Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact E-Cash. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 302–321, Aarhus, Denmark, May 22–26, 2005. Springer, Heidelberg. 1

CHS04. Giovanni Di Crescenzo, Javier Herranz, and Germán Sáez. Reducing Server Trust in Private Proxy Auctions. In *TrustBus 2004*, volume 3184 of *LNCS*, pages 80–89, Zaragoza, Spain, August 30 – September 1, 2004. Springer, Heidelberg. 1

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, Heidelberg. 1, 6, 6

Di 02. Giovanni Di Crescenzo. Equivocable And Extractable Commitment Schemes. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *Security in Communication Networks, 3rd International Conference, SCN 2002*, volume 2576 of *LNCS*, pages 74–87, Amalfi, Italy, September 11–13, 2002. Springer Verlag. 2, 6

DJ01. Ivan Damgård and Mads Jurik. A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In Kwangjo Kim, editor, *PKC 2001*, volume 1992 of *LNCS*, pages 119–136, Cheju Island, Korea, February 13–15, 2001. Springer, Heidelberg. 1

DN00. Cynthia Dwork and Moni Naor. Zaps and Their Applications. In *FOCS 2000*, pages 283–293, Redondo Beach, California, USA, November 12–14, 2000. IEEE Computer Society Press. 1

Elk11. Michael Elkin. An Improved Construction of Progression-Free Sets. *Israeli Journal of Mathematics*, 184:93–128, 2011. 2

Gro04. Jens Groth. *Honest Verifier Zero-Knowledge Arguments Applied.* PhD thesis, University of Århus, Denmark, October 2004. 1

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, Heidelberg. 1, 2, 3, 5

Gro11. Jens Groth. Efficient Zero-Knowledge Arguments from Two-Tiered Homomorphic Commitments. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 431–448, Seoul, South Korea, December 4–8, 2011. Springer, Heidelberg. 1

GS07. Jens Groth and Amit Sahai. Efficient Non-Interactive Proof Systems for Bilinear Groups. Technical Report 2007/155, International Association for Cryptologic Research, April 27, 2007. Available at http://eprint.iacr.org/2007/155 (version 20100222:192509), retrieved in December, 2011. 5

GS08. Jens Groth and Amit Sahai. Efficient Non-interactive Proof Systems for Bilinear Groups. In Nigel Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432, Istanbul, Turkey, April 13–17, 2008. Springer, Heidelberg. 1, 5

HSV06. Florian Hess, Nigel P. Smart, and Frederik Vercauteren. The Eta Pairing Revisited. *IEEE Transactions on Information Theory*, 52(10):4595–4602, 2006. 2
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. 1, 6, 6
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, Heidelberg. 1, 6
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. 3
Lip12. Helger Lipmaa. Progression-Free Sets and Sublinear Pairing-Based Non-Interactive Zero-Knowledge Arguments. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages ?–?, Taormina, Italy, March 18–21, 2012. Springer, Heidelberg. 1, 1, 2, 2, 1, 3, 2, 3, 2, 3.1, 3.2, 4, 5, 5, 6
PSNB11. C. C. F. Pereira Geovandro, Marcos A. Simplício Jr., Michael Naehrig, and Paulo S. L. M. Barreto. A Family of Implementation-Friendly BN Elliptic Curves. *Journal of Systems and Software*, 84(8):1319–1326, 2011. 2
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, Heidelberg. 1
San11. Tom Sanders. On Roth's Theorem on Progressions. *Annals of Mathematics*, 174(1):619–636, July 2011. 2
TV06. Terrence Tao and Van Vu. *Additive Combinatorics.* Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2006. 2
YHM+09. Tsz Hon Yuen, Qiong Huang, Yi Mu, Willy Susilo, Duncan S. Wong, and Guomin Yang. Efficient Non-interactive Range Proof. In Hung Q. Ngo, editor, *COCOON 2009*, volume 5609 of *LNCS*, pages 138–147, Niagara Falls, NY, USA, July 13–15, 2009. Springer, Heidelberg. 1, 4, 4, 1

## A Proof of Thm. 1

*Proof.* Perfect completeness: correctness verifications are straightforward. Clearly,

$$\hat{e}(f, C_f) = \hat{e}(f, g_2^{R_f} g_{2,\lambda_1}^{r_f}) = \hat{e}(f, g_2^{R_f}) \cdot \hat{e}(f, g_{2,\lambda_1}^{r_f}) = \hat{e}(f^{R_f}, g_2) \cdot \hat{e}(f^{r_f}, g_{2,\lambda_1}) = \hat{e}(\psi_f, g_2) \cdot \hat{e}(A_f, g_{2,\lambda_1}) \enspace .$$

Analogously, $\hat{e}(h, C_h) = \hat{e}(\psi_h, g_2) \cdot \hat{e}(A_h, g_{2,\lambda_1})$. Finally,

$$\begin{aligned}
\hat{e}(A_c\psi_g^{-1}, g_2) \cdot \hat{e}(g_1, C_f C_h) =& \hat{e}(g_1^r g_{1,\lambda_1}^a \cdot g_1^{-r-R_f-R_h}, g_2) \cdot \hat{e}(g_1, g_2^{R_f+R_h}) \cdot \hat{e}(g_1, g_{2,\lambda_1}^{r_f+r_h}) \\
=& \hat{e}(g_{1,\lambda_1}^a \cdot g_1^{-R_f-R_h}, g_2) \cdot \hat{e}(g_1^{R_f+R_h}, g_2) \cdot \hat{e}(g_1^{r_f+r_h}, g_{2,\lambda_1}) \\
=& \hat{e}(g_1^a, g_{2,\lambda_1}) \cdot \hat{e}(g_1^{r_f+r_h}, g_{2,\lambda_1}) = \hat{e}(g_1^{r_f+r_h+a}, g_{2,\lambda_1}) \enspace .
\end{aligned}$$

Computational Soundness: By the $\{\lambda_1\}$-PKE assumption in $\mathbb{G}_1/\mathbb{G}_2$, one can open the next values:

$$\begin{aligned}
(A_c, \bar{A}_c) =& (g_1^r g_{1,\lambda_1}^a, \bar{g}_1^r \bar{g}_{1,\lambda_1}^a) \enspace , \\
(A_g/A_c, \mathring{A}_{g/c}) =& ((g_1 g_{1,\lambda_1}^{-1})^{a'}, \mathring{g}_{1,g/c}^{a'}) \enspace , \\
(A_g, \mathring{A}_g) =& (g_1^{a''}, \mathring{g}_1^{a''}) \enspace , \\
(A_f, \mathring{A}_f) =& (f^{r_f}, \mathring{f}^{r_f}) \enspace , \\
(A_h, \mathring{A}_h) =& (h^{r_h}, \mathring{h}^{r_h}) \enspace , \\
(C_f, \bar{C}_f) =& (g_2^{R_f} g_{2,\lambda_1}^{r'_f}, \bar{g}_2^{R_f} \bar{g}_{2,\lambda_1}^{r'_f}) \enspace , \\
(C_h, \bar{C}_h) =& (g_2^{R_h} g_{2,\lambda_1}^{r'_h}, \bar{g}_2^{R_h} \bar{g}_{2,\lambda_1}^{r'_h}) \enspace , \\
(\psi_g, \mathring{\psi}_g) =& (g_1^{r''_a}, \mathring{g}_1^{r''_a}) \enspace , \\
(\psi_f, \mathring{\psi}_f) =& (g_1^{r''_f}, \mathring{g}_{1,f}^{r''_f}) \enspace , \text{and} \\
(\psi_h, \mathring{\psi}_h) =& (g_1^{r''_h}, \mathring{g}_{1,h}^{r''_h}) \enspace .
\end{aligned}$$

Since $A_c = g_1^r g_{1,\lambda_1}^a$, $A_g = g_1^{a''}$ and $A_g/A_c = (g_1 g_{1,\lambda_1}^{-1})^{a'}$, we have that $g_1^{a''} = g_1^{r+a'} g_{1,\lambda_1}^{a-a'}$. Thus, if $a \neq a'$, one can compute $x^{\lambda_1} \leftarrow (a'' - r - a')/(a - a')$, and from this compute $x$ and thus break the $\{\lambda_1\}$-PSDL assumption. (To verify whether $x$ is the correct root, one can check that $g_1^{x^{\lambda_1}} = g_{1,\lambda_1}$.) Thus $a = a'$, and thus also $a'' = r + a$ and $A_g = g_1^{r+a}$.

Due to $C_f = g_2^{R_f} g_{2,\lambda_1}^{r'_f}$, $\psi_f = g_1^{r''_f}$, $A_f = f^{r_f}$ and $\hat{e}(f, C_f) = \hat{e}(\psi_f, g_2) \cdot \hat{e}(A_f, g_{2,\lambda_1})$, we have

$$\hat{e}(f, g_2^{R_f} g_{2,\lambda_1}^{r'_f}) = \hat{e}(g_1^{r''_f}, g_2) \hat{e}(f^{r_f}, g_2^{x^{\lambda_1}})$$

for unknown $x$. Taking the discrete logarithm of the both sides of the last equation, we get that $R_f/\mathsf{sk}_1 + r'_f x^{\lambda_1}/\mathsf{sk}_1 = r''_f + r_f x^{\lambda_1}/\mathsf{sk}_1$, or $(r_f - r'_f) x^{\lambda_1} = R_f - r''_f \cdot \mathsf{sk}_1$. Thus, if $r_f \neq r'_f$, then we can compute $x^{\lambda_1}$, and find from this $x$, and thus break the $\{\lambda_1\}$-PSDL assumption. Thus, $r_f = r'_f$ and therefore also $C_f = g_2^{R_f} g_{2,\lambda_1}^{r_f}$. Moreover, $\psi_f = g_1^{r''_f} = f^{R_f}$.

Analogously, we get that $r_h = r'_h$ and therefore $C_h = g_1^{R_h} g_{1,\lambda_1}^{r_h}$ and $\psi_h = h^{R_h}$.

Due to $C_f = g_2^{R_f} g_{2,\lambda_1}^{r_f}$, $C_h = g_1^{R_h} g_{1,\lambda_1}^{r_h}$, $\psi_g = g_1^{r''_a}$, $A_c = g_1^r g_{1,\lambda_1}^a$, $A_g = g_1^{r+a}$ and $\hat{e}(g_1, C_f C_h) = \hat{e}(\psi_g A_c^{-1}, g_2) \cdot \hat{e}(A_g, g_{2,\lambda_1})$, we have

$$\hat{e}(g_1, g_2^{r+R_f+R_h+(r_f+r_h)x^{\lambda_1}}) = \hat{e}(g_1^{r''_a} g_1^{-r} g_{1,\lambda_1}^{-a}, g_2) \cdot \hat{e}(g_1^{r+a}, g_{2,\lambda_1}) = \hat{e}(g_1^{r''_a - r + r x^{\lambda_1}}, g_2)$$

for unknown $x$. Taking the discrete logarithm of both sides of the last equation, we get $r + R_f + R_h + (r_f + r_h) x^{\lambda_1} = r''_a - r + r x^{\lambda_1}$. Again, if $r_f + r_h \neq r$, then one can compute $x^{\lambda_1}$ and thus also $x$. Thus, $r = r_f + r_h$, and thus also $r''_a = r + R_f + R_h$. This means that $A_c = g_1^{r_f + r_h} g_{1,\lambda_1}^a$ and $(A_f, A_g, A_h) = (g_1^{r_f+r_h+a}, f^{r_f}, h^{r_h})$.

Computational Zero-knowledge: we construct the next simulator $(\mathcal{S}_1, \mathcal{S}_2)$. $\mathcal{S}_1$ creates a CRS according to the protocol together with a trapdoor $\mathsf{td} = (\alpha_g, \alpha_f, \alpha_h, \bar{\alpha}, \alpha_{g,c}, x)$. On input $\mathsf{td}$, $\mathcal{S}_2$ creates $z_f, z_h \leftarrow \mathbb{Z}_p$. He then sets $C_f \leftarrow g_2^{z_f}$, $\psi_f \leftarrow f^{z_f}/A_f^{x^{\lambda_1}}$, $C_h \leftarrow g_2^{z_h}$, $\psi_h \leftarrow h^{z_h}/A_h^{x^{\lambda_1}}$, and $\psi_g \leftarrow g_1^{z_f+z_h}/A_g^{x^{\lambda_1}}$. He creates the knowledge elements $(\mathring{A}_g, \mathring{A}_f, \mathring{A}_h, \mathring{A}_c, \mathring{\psi}_g, \bar{C}_f, \mathring{\psi}_f, \bar{C}_h, \mathring{\psi}_h, \mathring{A}_{g/c})$ by using the trapdoor. For example, $\mathring{A}_{g/c} \leftarrow (A_g/A_c)^{\alpha_{g/c}}$. One can now check that the verification succeeds. For example,

$$\begin{aligned}\hat{e}(\psi_f, g_2)\hat{e}(A_f, g_{2,\lambda_1}) =& \hat{e}(f^{z_f}/A_f^{x^{\lambda_1}}, g_2) \cdot \hat{e}(A_f, g_{2,\lambda_1}) = \hat{e}(f^{z_f}, g_2)/\hat{e}(A_f^{x^{\lambda_1}}, g_2)\hat{e}(A_f, g_{2,\lambda_1}) \\ =& \hat{e}(f^{z_f}, g_2) = \hat{e}(f, C_f) \ ,\end{aligned}$$

and finally,

$$\hat{e}(A_c \psi_g^{-1}, g_2) \cdot \hat{e}(g_1, C_f C_h) = \hat{e}(g_1^{-z_f - z_h} A_g^{x^{\lambda_1}} A_c, g_2) \cdot \hat{e}(g_1, g_2^{z_f + z_h}) = \hat{e}(A_g, g_{2,\lambda_1}) \ .$$

If the DLIN assumption is true, then $(A_g, A_f, A_h)$ is indistinguishable from an encryption of $0 \in [0, H]$, and thus the whole argument is computationally knowledge. ◻

## B Proof of Thm. 2

*Proof.* Perfect completeness: Recall that in the case of the product arguments, the inputs of $\mathcal{P}$ are $(A, \hat{A}, B, \hat{B}, B_2, C, \hat{C})$. Within this proof we say that $(B, \hat{B}, B_2)$ (assuming $B_2$ is correctly defined, that is, $\hat{e}(B, g_2) = \hat{e}(g_1, B_2)$) commits to the same values as $(B, \hat{B})$.

The pairing verifications (for example, that $\hat{e}(K, \hat{g}_2) = \hat{e}(\hat{K}, g_2)$) hold by construction of the protocol. Since $(B'_j, \hat{B}'_j)$ commits to $(b'_{j1}, \ldots, b'_{jn})$ for binary $b'_{ji}$ then the argument $(\psi'_j, \hat{\psi}'_j)$ verifies.

Note that $(\prod_{j=0}^{n_v} (B'_j)^{G'_j}, \prod_{j=0}^{n_v} (\hat{B}'_j)^{G'_j})$ commits to $(b_1, \ldots, b_n)$. Thus argument $(\psi_1^\times, \hat{\psi}_1^\times)$ verifies. Since $(C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}})$ commits to a rotation of $(C, \hat{C})$, then $(A^*, \hat{A}^*, \psi_2^\times, \hat{\psi}_2^\times, \psi_2^{\mathsf{rot}}, \hat{\psi}_2^{\mathsf{rot}})$ verifies. Since $(C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}})$ commits to $(0, c_1, \ldots, c_{n-1})$ and $(C/B^\dagger, \hat{C}/\hat{B}^\dagger)$ commits to

$$(c_1 - G_1 b_1, c_2 - G_2 b_2, \ldots, c_n - G_n b_n) = (0, c_1, \ldots, c_{n-1}) \ ,$$

then $(\psi_3^\times, \hat{\psi}_3^\times)$ verifies. Finally, since $(u-1)a = \sum_{i=1}^n G_i b_i$ and $c_n = \sum_{i=1}^n G_i b_i$, then $(\psi_4^\times, \hat{\psi}_4^\times)$ verifies.

COMPUTATIONAL SOUNDNESS: let $\mathcal{A}$ be a non-uniform PPT adversary who creates a statement $(\mathsf{pk}, A_g, A_f, A_h, A_c, \hat{A}_c)$ and an accepting range proof $\psi$. By the DLIN assumption, the BBS cryptosystem is IND-CPA secure, and thus the adversary obtains no information from $(A_g, A_f, A_h)$. By the $\Lambda$-PKE assumption, there exists a non-uniform PPT extractor $X_{\mathcal{A}}$ that, running on the same inputs and seeing $\mathcal{A}$'s random tape, extracts the following openings:

$$
\begin{aligned}
(A_c, \hat{A}_c) =& \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{a}; r) \ , \\
(B'_j, \hat{B}'_j) =& \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{b'_j}; r_j) \text{for } j \in [0, n_v] \ , \\
(B^\dagger, \hat{B}^\dagger) =& \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{b}^\dagger; r'_0) \ , \\
(C, \hat{C}) =& \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{c}; r'_1) \ , \\
(C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}}) =& \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{c}_{\mathsf{rot}}; r'_2) \ , \\
(\psi_1^\times, \hat{\psi}_1^\times) =& (\prod_{s \in \hat{\Lambda}} g_{2s}^{f'_{(\times 1, s)}}, \prod_{s \in \hat{\Lambda}} \hat{g}_{2s}^{f'_{(\times 1, s)}}) \ , \\
(A^*, \hat{A}^*) =& \mathcal{C}\mathsf{om}^1(\widehat{\mathsf{ck}}_1; \boldsymbol{a}^*; r_{a^*}) \ , \\
(\psi_2^\times, \hat{\psi}_2^\times) =& (\prod_{s \in \hat{\Lambda}} g_{2s}^{f'_{(\times 2, s)}}, \prod_{s \in \hat{\Lambda}} \hat{g}_{2s}^{f'_{(\times 2, s)}}) \ , \\
(\psi_2^{\mathsf{rot}}, \hat{\psi}_2^{\mathsf{rot}}) =& (\prod_{s \in \tilde{\Lambda}} g_{2s}^{f'_{(\mathsf{rot2}, s)}}, \prod_{s \in \tilde{\Lambda}} \tilde{g}_{2s}^{f'_{(\mathsf{rot2}, s)}}) \ , \\
(\psi_3^\times, \hat{\psi}_3^\times) =& (\prod_{s \in \hat{\Lambda}} g_{2s}^{f'_{(\times 3, s)}}, \prod_{s \in \hat{\Lambda}} \hat{g}_{2s}^{f'_{(\times 3, s)}}) \ , \text{and} \\
(\psi_4^\times, \hat{\psi}_4^\times) =& (\prod_{s \in \hat{\Lambda}} g_{2s}^{f'_{(\times 4, s)}}, \prod_{s \in \hat{\Lambda}} \hat{g}_{2s}^{f'_{(\times 4, s)}}) \ .
\end{aligned}
$$

It will also create the openings that correspond to $\psi_5^{ce}$. If any of the openings fails, we are done. Since $\tilde{\Lambda}$-PSDL assumption is supposed to hold, all the following is true. (If it is not true, one can efficiently test it, and thus we have broken the PSDL assumption.)

Since $\hat{e}(B'_j, g_2) = \hat{e}(g_1, B'_{j2})$ for $j \in [0, n_v]$, then $(B_{j1}, \hat{B}_{j1}, B_{j2})$ commits to $\boldsymbol{b'_j}$. Therefore, due to the $\hat{\Lambda}$-PSDL assumption, the fact that the adversary knows the openings of $(B'_j, \hat{B}'_j)$ and $(\psi'_j, \hat{\psi}'_j)$, and the last statement of Fact 2, since $(\psi'_j, \hat{\psi}'_j)$ verifies, then $b'_{ji} \in \{0, 1\}$ for all $j \in [0, n_v]$ and $i \in [1, n]$. Thus, by Fact 6, $\boldsymbol{b} = (b_1, \ldots, b_n) := (\sum_{j=0}^{n_v} G'_j b'_{j1}, \ldots, \sum_{j=0}^{n_v} G'_j b'_{jn}) \in [0, u-1]^n$, and thus $(\prod_{j=0}^{n_v} (B'_j)^{G'_j}, \prod_{j=0}^{n_v} (\hat{B}'_j)^{G'_j})$ commits to $\boldsymbol{b}$ with $b_i \in [0, u-1]$.

Due to the $\hat{\Lambda}$-PSDL assumption, the fact that the adversary knows the openings of $(B'_j, \hat{B}'_j)$, $(B^\dagger, \hat{B}^\dagger)$ and $(\psi_1^\times, \hat{\psi}_1^\times)$, and the last statement of Fact 2, since $(\psi_1^\times, \hat{\psi}_1^\times)$ verifies, then $b_i^\dagger = G_i b_i$. Due to the $\tilde{\Lambda}$-PSDL assumption, the fact that the adversary knows the openings of $(C, \tilde{C})$, $(C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}})$ and $(A^*, \hat{A}^*, \psi_2^\times, \hat{\psi}_2^\times, \psi_2^{\mathsf{rot}}, \hat{\psi}_2^{\mathsf{rot}})$, and the last statement of Fact 2, since $(A^*, \hat{A}^*, \psi_2^\times, \hat{\psi}_2^\times, \psi_2^{\mathsf{rot}}, \hat{\psi}_2^{\mathsf{rot}})$ verifies, then $c_{\mathsf{rot},1} = c_n$ and $c_{\mathsf{rot},i+1} = c_i$ for $i \geq 1$.

Due to the $\hat{\Lambda}$-PSDL assumption, the fact that the adversary knows the openings of $(C_{\mathsf{rot}}, \tilde{C}_{\mathsf{rot}})$, $(C, \hat{C})$, $(B^\dagger, \hat{B}^\dagger)$, and $(\psi_3^\times, \hat{\psi}_3^\times)$, and the last statement of Fact 2, since $(\psi_3^\times, \hat{\psi}_3^\times)$ verifies, then $c_1 - G_1 b_1 = 0$ and $c_i - G_i b_i = c_{\mathsf{rot},i} = c_{i-1}$ for $i > 1$. Therefore, $c_1 = G_1 b_1$, $c_2 = G_2 b_2 + G_1 b_1$, and by induction $c_i = \sum_{j=1}^n G_i b_i$ for $i \geq 1$. In particular, $c_n = \sum_{i=1}^n G_i b_i$ for $b_i \in [0, u-1]$.

Due to the $\hat{\Lambda}$-PSDL assumption, the fact that the adversary knows the openings of $(C, \hat{C})$, $(A_c, \hat{A}_c)$, and $(\psi_4^\times, \hat{\psi}_4^\times)$, and the last statement of Fact 2, since $(\psi_4^\times, \hat{\psi}_4^\times)$ verifies, then $(A_c, \hat{A}_c) = (g_1^r g_{1,\lambda_1}^a, \hat{g}_1^r \hat{g}_{1,\lambda_1}^a)$ commits to $(a, 0, \ldots, 0)$ such that $(u-1)a = \sum_{i=1}^n G_i b_i$ for $b_i \in [0, u-1]$, and therefore by Fact 6, $a \in [0, H]$.

Due to the $\{\lambda_1\}$-PSDL assumption and since $\psi_5^{ce}$ verifies, then $(A_g, A_f, A_h)$ encrypts $a \in [0, H]$.

COMPUTATIONAL ZERO-KNOWLEDGE: we construct the following simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$. First, $\mathcal{S}_1$ creates a correctly formed common reference string together with a simulation trapdoor $\mathsf{td} = (\hat{\alpha}, \tilde{\alpha}, \ldots, x)$. After that, the prover creates a statement $inp^r := (\mathsf{pk}, A_g, A_f, A_h, A_c, \hat{A}_c)$ and sends it to the simulator. Second, $\mathcal{S}_2(\mathsf{crs}; inp^r; \mathsf{td})$ uses a knowledge extractor to extract $(\boldsymbol{a}, r)$ from the prover's random coins and

$(A_c, \hat{A}_c)$. Since we are only interested in the case of a honest prover, we have that $\boldsymbol{a} = (a, 0, \dots, 0)$ with $a \in [0, H]$. Thus, using the fact that the knowledge commitment scheme is also trapdoor, the simulator computes $r'' \leftarrow ax^{\lambda_n} + r$; clearly $A = g_1^{r''}$. Since both $r$ and $r''$ are uniformly random, $r''$ does not leak any information on the prover's input. After that, the simulator creates all commitments $(B'_j, \hat{B}'_j, B'_{j2})_{j \in [0, n_v]}$, $(B^\dagger, \hat{B}^\dagger)$, $(C, \hat{C}, \tilde{C})$ and $(C_{\mathsf{rot}}, \hat{C}_{\mathsf{rot}}, \tilde{C}_{\mathsf{rot}})$ as in the argument, but replacing $\boldsymbol{a}$ with 0 and $r$ with $r''$. (Note that all the mentioned commitments just commit to $\mathbf{0}$.) Thus, the simulator can simulate all product and permutation arguments and the argument of Sect. 5. Clearly, this simulated argument $\psi^{sim}$ is perfectly indistinguishable from the real argument $\psi$. $\square$

## C Proof of Thm. 3

*Proof.* The communication complexity: $n_v + 1$ tuples $(B'_j, \hat{B}'_j, B'_{j2}, \psi_j)$ (each has 2 elements of $\mathbb{G}_1$ and 3 elements of $\mathbb{G}_2$), and then 8 extra elements from $\mathbb{G}_1$, 3 Hadamard product arguments (2 elements from $\mathbb{G}_2$ each), 1 permutation argument (2 elements from $\mathbb{G}_1$ and 4 elements from $\mathbb{G}_2$), and argument $\psi^{ce}$ (13 elements from $\mathbb{G}_1$ and 2 elements from $\mathbb{G}_2$). In total, thus $2(n_v + 1) + 8 + 2 + 13 = 2n_v + 25$ elements from $\mathbb{G}_1$ and $3(n_v + 1) + 3 \cdot 2 + 4 + 2 = 3n_v + 15$ elements from $\mathbb{G}_2$.

The prover's computational complexity is dominated by $(n_v + 1) + 3 = n_v + 4$ Hadamard product arguments and 1 permutation argument ($\Theta(n^2)$ scalar multiplications and bilinear-group $n^{1+o(1)}$ exponentiations each), that is in total $\Theta(n^2 \cdot n_v) = \Theta(n^2 \cdot \log u)$ scalar multiplications and $n^{1+o(1)} \log u$ exponentiations.

The verifier's computational complexity is dominated by verifying $n_v + 4$ Hadamard product arguments (5 pairings each), 1 permutation argument (12 pairings), and the argument $\psi^{ce}$ (33 pairings). In addition, the verifier performs $2 \cdot (2(n_v + 1) + 6) = 4n_v + 16$ pairings. The total number of pairings is thus $9n_v + 81$. The rest follows. $\square$