Key-and-Argument-Updatable QA-NIZKs

Sunday 31st March, 2019, 16:27

Helger Lipmaa

University of Tartu, Estonia

Abstract. There are several new efficient approaches to decrease the trust in the CRS creators in the case of non-interactive zero knowledge (NIZK) in the CRS model. Recently, Groth *et al.* (CRYPTO 2018) defined the notion of NIZK with updatable CRS (*updatable NIZK*) and described an updatable SNARK. We consider the same problem in the case of QA-NIZKs. While doing it, we define an important new property: we require that after updating the CRS, one should be able to update a previously generated argument to a new argument that is valid with the new CRS. We propose a general definitional framework for *key-and-argument-updatable QA-NIZKs.* After that, we describe a key-and-argument-updatable version of the most efficient known QA-NIZK for linear subspaces by Kiltz and Wee. Importantly, for obtaining soundness it suffices to update a universal public key that just consists of a matrix drawn from a KerMDH-hard distribution and thus can be shared by any pairing-based application that relies on the same hardness assumption. After specializing the universal public key to concrete language parameter, one can use the proposed key-and-argument updating algorithms to continue updating to strengthen the soundness guarantee.

Keywords: BPK model, CRS model, QA-NIZK, subversion security, updatable CRS, updatable proof

1 Introduction

SNARKs. Zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARKs, [Gro10,Lip12,GGPR13,Gro16,GM17]) have become one of the most widely researched and deployed zero-knowledge argument systems, in particular because of their applicability in verifiable computation [PHGR13] and anonymous cryptocurrencies [DFKP13,BCG^{+}14]. However, due to a well-known impossibility result [GW11], the soundness of SNARKs can only be based on non-falsifiable assumptions [Nao03].

Moreover, a new security concern has arisen recently. Most of the existing pairing-based zk-SNARKs are defined in the common reference string (CRS, [BFM88]) model assuming the existence of a trusted third party that samples a CRS from the correct distribution and does not leak any trapdoors. The existence of such a trusted third party is often a too strong assumption.

Recently, several efficient approaches have been proposed to decrease trust in the CRS creation, like the use of multi-party CRS generation [BCG+15,BGG17,BGM17] and the notion of subversion-resistant zero-knowledge (S-ZK) SNARKs [BFS16,ABLZ17,Fuc18]. An S-ZK SNARK guarantees to the prover P the zero-knowledge property even if the CRS was maliciously created, as long as P checks that a public algorithm $\mathsf{V}_{\mathsf{crs}}$ accepts the CRS. Existing S-ZK SNARKs use a non-falsifiable assumption to extract from a $\mathsf{V}_{\mathsf{crs}}$-accepted CRS its trapdoor td. After that, one simulates CRS by using td. Since one cannot at the same time achieve subversion-soundness and S-ZK [BFS16], these SNARKs only achieve the usual (knowledge-)soundness property.

Very recently, Groth *et al.* [GKM+18] proposed the notion of updatable CRS and showed how to implement it in the case of SNARKs. The main idea behind it is that given a CRS based on some trapdoor td, one can update the CRS to a new CRS crs' based on some trapdoor td'. Updating can be repeated many times, obtaining a sequence

$$\mathsf{crs}_0 \to \mathsf{crs}_1 \to \mathsf{crs}_2 \to \ldots \to \mathsf{crs}_n$$

of CRSs, updated by some parties $\mathcal{P}_1, \ldots, \mathcal{P}_n$. The SNARK will be sound (and the CRS will be correctly distributed) if at least one of the CRS updaters was honest; this allows one to get soundness (if at least one updater was honest) and zero-knowledge (without any assumption on the updaters). At some moment the prover will create an argument. The verifier accepts only when she trusts some updater *at the moment of argument creation*. Groth *et al.* [GKM+18] constructed an updatable SNARK with a quadratically-long universal CRS (valid for all circuits of the given size) and linearly-long specialized CRS (constructed from the universal CRS when a circuit is fixed and actually used to create an argument).

As a drawback, the CRS updates after an argument has been created cannot be taken to account since the argument itself is not updatable; in particular, it means that the verifier has to signal to the prover that she is ready to trust the argument created at some moment (since the CRS at that moment has been updated by a verifier-trusted party).

QA-NIZKs. Starting from the seminal work of Jutla and Roy [JR13], QA-NIZKs have been a (quite different) alternative research direction as compared to SNARKs with quite different applications and quite different underlying techniques. As SNARKs, QA-NIZKs offer succinct arguments and super-efficient verification. On the negative side, efficient QA-NIZKs are known only for a much smaller class of languages like the language of linear subspaces [JR13,LPJY14,JR14,KW15,ABP15,LPJY15] and some related languages, including the language of quadratic relations [GHR15] and shuffles [GR16] (although see [DGP+19]). From the positive side, these QA-NIZKs are based on very standard assumptions like MDDH [EHK+13] (e.g., DDH) and Ker-MDH [MRV16] (e.g., CDH). QA-NIZKs have applications in the construction of efficient cryptographic primitives (like KDM-CCA2-secure encryption, IND-

CCA2-secure IBE, and UC-secure commitments, [JR13], an linearly homomorphic structure-preserving signatures [KW15]) based on standard assumptions.

Motivated by related research on SNARKs, Abdolmaleki *et al.* [ALSZ18] recently studied subversion-resistant QA-NIZKs. They showed that S-ZK in the CRS model is essentially equal to the previously known notion of non-uniform zero-knowledge (or, no-auxiliary-string non-black-box zero knowledge, [Wee07]) in the significantly weaker Bare Public Key (BPK, [CGGM00,MR01]) model. Due to known impossibility results [Wee07], this provides a simple proof that non-falsifiable assumptions are needed to achieve nonuniform zero-knowledge in the BPK model. Abdolmaleki *et al.* [ALSZ18] proposed also an efficient $\mathsf{V}_{\mathsf{crs}}$ algorithm for the most efficient known QA-NIZK Π'_{as} for linear subspaces by Kiltz and Wee [KW15] and proved that the resulting construction achieves nonuniform zero-knowledge in the BPK model (i.e., is S-ZK secure).

We note that the proof methods of [ALSZ18] are quite nontrivial. In particularly, Abdolmaleki *et al.* proved nonuniform zero-knowledge under a new tautological $\mathsf{KW\text{-}KE}$ knowledge assumption, and then show that $\mathsf{KW\text{-}KE}$ is secure in the subversion generic bilinear group model of [BFS16,ABLZ17] (S-GBGM, named GBGM with hashing in [BFS16]). Especially the latter proof was quite sophisticated.

Our Contributions. We define updatable QA-NIZK by roughly following the definitional guidelines of [GKM$^+$18] for updatable SNARKs. However, we make two significant changes to the model itself. The second change (the ability to update also the argument) is especially important, allowing for new applications.

Firstly, since QA-NIZK security definitions are different compared to SNARKs' (with the language parameter ϱ having an important and distinct role; moreover, QA-NIZKs do not provide knowledge-soundness), we redefine updatable versions of completeness, soundness, and zero knowledge correspondingly. We add to them the natural requirement of hiding (updated key and fresh key are indistinguishable). We will follow the framework of [ALSZ18] by relying on nonuniform QA-NIZK in the BPK model. According to [ALSZ18], the prover and the verifier of a QA-NIZK argument share a honestly generated language parameter ϱ together with a (possibly malformed) verifier's public key PK. We add the key-updating and update-verification algorithms with the corresponding security requirements: key-update completeness, key-update hiding, strong key-update hiding, key-update soundness, and key-update zero knowledge.

Secondly and more importantly, we add to the QA-NIZK the ability of updating the argument. That is, given a PK and an argument π constructed while using PK, one can then update PK (to a new key PK') and π to a new valid argument π' (corresponding to PK'). There are two different ways to update the argument. First, the honest argument updater must know the witness (but no secret information about the key update). Second, the argument-update simulator must know some secret key-update trapdoor (but he does not have to know the witness). As in the case of creating arguments, it is required that these two different ways of updating are indistinguishable.

Argument-updating has various interesting applications. Knowing the key-update trapdoor, the key-updater can simulate the argument-update and this means that we will not get soundness unless at least one of the argument-creators or argument-updaters does not collaborate with the corresponding key-creator or key-updater. One can obtain different trust models by handling the updating process differently. For example, it can guarantee additional anonymity to the honest argument-updater since we would not know which of the argument-updaters actually knows the witness. On the other hand, if there exists at least one update such that we can trust the corresponding key-updater and argument-updater not to trust, we are guaranteed that one of the argument-updater actually "knows" the witness and thus the statement is true.

We will give precise security definitions for *key-and-argument-updatable* QA-NIZKs. In particular, we require such QA-NIZKs to satisfy argument-update completeness, argument-update hiding, strong argument-update hiding, argument-update soundness, argument-update zero knowledge. We will prove that argument-update soundness and argument-update zero knowledge follow from simpler security requirements and thus do not have to be proven anew in the case of each new QA-NIZK.

We show that the provided security definitions can implemented, by proposing an updatable version $\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ of the Kiltz-Wee QA-NIZK Π'_{as} for linear subspace [KW15]. Our construction uses crucially the fact that all operations (like the public key generation and proving) in Π'_{as} consist of linear operations only. Because of this, the new update-related algorithms are relatively simple, but still non-trivial. For example, we use additive updates for some elements in the public key and multiplicative updates for some other elements. Interestingly, we update the argument π by adding to it a (honestly created or simulated) argument when using the "difference" $\widehat{\textsc{pk}} = \textsc{pk}' - \textsc{pk}$ of the new and the old public key ($\textsc{pk}'$ and $\textsc{pk}$) as a one-time public key. This is why we can update arguments without the knowledge of either the witness or the trapdoor of either $\textsc{pk}$ or $\textsc{pk}'$. Instead, it suffices to use the trapdoor corresponding to the concrete update.

We prove that $\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ satisfies all defined security properties, and in particular, that — like the (non-updatable) nonuniform QA-NIZK of [ALSZ18] — it is sound either under the KerMDH assumption [MRV16] (for some values of the system parameters) or the $\mathsf{SKerMDH}$ assumption [GHR15] (for some other values of the system parameters) and nonuniform argument-update zero-knowledge under the $\mathsf{KW\text{-}KE}$ assumption of [ALSZ18].

Updatable Universal Public Key. The goal of updatability is to protect soundness in the case $\textsc{pk}$ may be subverted, since nonuniform zero knowledge (i.e., Sub-ZK) can be obtained by running the public algorithm $\mathsf{V_{pk}}$ [ABLZ17]. In Π'_{as}, soundness is guaranteed by one of the elements of $\textsc{pk}$ (namely, $[\bar{\boldsymbol{A}}]_2$, see Fig. 2) coming from a KerMDH-hard distribution, and another element $[\boldsymbol{C}]_2$ being correctly computed from $[\bar{\boldsymbol{A}}]_2$. Since the latter can be verified by the public-key verification algorithm, it suffices to only update $[\bar{\boldsymbol{A}}]_2$. Then, $[\bar{\boldsymbol{A}}]_2$ will be a "universal public key" [GKM+18] for all possible language parameters in all applications that rely on the concrete KerMDH *assumption*.

Importantly, one is here not even restricted to Π'_{as} or even QA-NIZK: *any* application that relies on KerMDH and where it suffices to know $[\bar{\boldsymbol{A}}]_2$ (instead of $[\boldsymbol{A}]_2$) can use the same matrix $[\bar{\boldsymbol{A}}]_2$. A standard example is the 1-Lin [BBS04,EHK+13] distribution $\mathcal{L}_1 = \{\boldsymbol{A} = \left(\begin{smallmatrix} a \\ 1 \end{smallmatrix}\right) : a \leftarrow_\$ \mathbb{Z}_p\}$. We emphasize that the possibility to rely just on $[\bar{\boldsymbol{A}}]_2$ is a major difference with updatable SNARKs [GKM+18] where the universal key is quite complex and each universal key of length $\Theta(n^2)$ (see [MBKM19] for recent optimizations) can only be used for circuits of size $\leq n$. See Section 6 for more information.

2 Preliminaries

We denote empty string by ϵ. Let PPT denote probabilistic polynomial-time and let $\lambda \in \mathbb{N}$ be the security parameter. All adversaries will be stateful. For an algorithm $\mathcal{A}$, let range($\mathcal{A}$) be the range of $\mathcal{A}$, i.e., the set of of valid outputs of $\mathcal{A}$, let $\mathsf{RND}(\mathcal{A})$ denote the random tape of $\mathcal{A}$, and let $r \leftarrow_\$ \mathsf{RND}(\mathcal{A})$ denote the uniformly at random choice of the randomizer r from $\mathsf{RND}(\mathcal{A})$. By $y \leftarrow \mathcal{A}(\mathsf{x}; r)$ we denote the fact that $\mathcal{A}$, given an input x and a randomizer r, outputs y. When we use this notation then r represents the full random tape of $\mathcal{A}$. We denote by $\mathsf{negl}(\lambda)$ an arbitrary negligible function, and by $\mathsf{poly}(\lambda)$ an arbitrary polynomial function. We write $a \approx_\lambda b$ if $|a - b| = \mathsf{negl}(\lambda)$.

Bilinear pairings. A bilinear group generator $\mathsf{Pgen}(1^\lambda)$ returns $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e})$, where $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are three additive cyclic groups of prime order p, and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a non-degenerate efficiently computable bilinear pairing. We require the bilinear pairing to be Type-3 [GPS08], i.e., we assume that there is no efficient isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. We use the bracket notation of [EHK+13], e.g., we write $[a]_\iota$ to denote $a g_\iota$ where $a \in \mathbb{Z}_p$ and g_ι is a fixed generator of $\mathbb{G}_\iota$. We denote $\hat{e}([a]_1, [b]_2)$ as $[a]_1 \bullet [b]_2$. We use the bracket notation freely together with matrix notation, e.g., $\boldsymbol{AB} = \boldsymbol{C}$ iff $[\boldsymbol{A}]_1 \bullet [\boldsymbol{B}]_2 = [\boldsymbol{C}]_T$.

Bare Public Key (BPK) Model. In the BPK model [CGGM00,MR01], parties have access to a public file F, a polynomial-size collection of records (id, PK_{id}), where id is a string identifying a party (e.g., a verifier), and PK_{id} is her (alleged) public key. In a typical zero-knowledge protocol in the BPK model, a key-owning party $\mathcal{P}_{id}$ works in two stages. In stage one (the *key-generation stage*), on input a security parameter 1^λ and randomizer r, $\mathcal{P}_{id}$ outputs a public key PK_{id} and stores the corresponding secret key SK_{id}. We assume the *no-auxiliary-string BPK* model where from this it follows that $\mathcal{P}_{id}$ actually created PK_{id}. After that, F will include (id, PK_{id}). In stage two, each party has access to F, while $\mathcal{P}_{id}$ has possibly access to SK_{id} (however, the latter will be not required by us). It is commonly assumed that only the verifier of a NIZK argument system in the BPK model has a public key [MR01].

Matrix Diffie-Hellman Assumptions. Kernel Matrix Diffie-Hellman Assumption (KerMDH) is a well-known assumption family formally introduced in [MRV16] and say, used in [KW15] to show soundness of the newly proposed QA-NIZK argument system for linear subspaces. The KerMDH assumption states that for a matrix $\boldsymbol{A}$ from some well-defined distribution it is difficult to find a representation of a vector that belongs to the kernel of $\boldsymbol{A}^\top$ provided that the matrix is given in exponents only, i.e., as $[\boldsymbol{A}]_\iota$.

For fixed p, denote by $\mathcal{D}_{n_2 k_2}$ a probability distribution over matrices in $\mathbb{Z}_p^{n_2 \times k_2}$, where $n_2 > k_2$. We assume that $\mathcal{D}_{n_2 k_2}$ outputs matrices $\boldsymbol{A}$ where the upper $k_2 \times k_2$ submatrix $\bar{\boldsymbol{A}}$ is always invertible. Let $\bar{\mathcal{D}}_{n_2 k_2}$ that outputs $\bar{\boldsymbol{A}}$, where $\boldsymbol{A}$ is sampled from $\mathcal{D}_{n_2 k_2}$. When $n_2 = k_2 + 1$, we denote $\mathcal{D}_{k_2} = \mathcal{D}_{n_2 k_2}$.

$\mathcal{D}_{n_2 k_2}$-$\mathsf{KerMDH}_{\mathbb{G}_\iota}$ [MRV16] holds relative to Pgen, if $\forall$ PPT $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{kermdh}}_{\mathcal{A},\mathcal{D}_{n_2 k_2},\iota,\mathsf{Pgen}}(\lambda) \approx_\lambda 0$ where $\mathsf{Adv}^{\mathrm{kermdh}}_{\mathcal{A},\mathcal{D}_{n_2 k_2},\iota,\mathsf{Pgen}}(\lambda) :=$

$$\Pr\left[\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \boldsymbol{A} \leftarrow_\$ \mathcal{D}_{n_2 k_2}; [\boldsymbol{c}]_{3-\iota} \leftarrow \mathcal{A}(\mathsf{p}, [\boldsymbol{A}]_\iota) : \boldsymbol{A}^\top \boldsymbol{c} = \boldsymbol{0}_{k_2} \wedge \boldsymbol{c} \neq \boldsymbol{0}_{n_2}\right] .$$

$\mathcal{D}_{n_2 k_2}$-$\mathsf{SKerMDH}$ [GHR15] holds relative to Pgen, if $\forall$ PPT $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{skermdh}}_{\mathcal{A},\mathcal{D}_{n_2 k_2},\mathsf{Pgen}}(\lambda) \approx_\lambda 0$, where $\mathsf{Adv}^{\mathrm{skermdh}}_{\mathcal{A},\mathcal{D}_{n_2 k_2},\mathsf{Pgen}}(\lambda) :=$

$$\Pr\begin{bmatrix}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \boldsymbol{A} \leftarrow_\$ \mathcal{D}_{n_2 k_2}; ([\boldsymbol{c}_1]_1, [\boldsymbol{c}_2]_2) \leftarrow \mathcal{A}(\mathsf{p}, [\boldsymbol{A}]_1, [\boldsymbol{A}]_2) : \\ \boldsymbol{A}^\top(\boldsymbol{c}_1 - \boldsymbol{c}_2) = \boldsymbol{0}_{k_2} \wedge \boldsymbol{c}_1 - \boldsymbol{c}_2 \neq \boldsymbol{0}_{n_2}\end{bmatrix} .$$

According to Lem. 1 of [GHR15], if $\mathcal{D}_{n_2 k_2}$-KerMDH holds in generic symmetric bilinear groups then $\mathcal{D}_{n_2 k_2}$-$\mathsf{SKerMDH}$ holds in generic asymmetric bilinear groups. KerMDH holds also for Type-I pairings, where $\mathbb{G}_1 = \mathbb{G}_2$. However, in such case we need $k_2 \geq 2$, which affects efficiency of the arguments relying on it.

$\mathcal{D}_{n_2 k_2}$-$\mathsf{wKerMDH}_{\mathbb{G}_1}$ [ALSZ18] holds relative to Pgen, if for any PPT $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{wkermdh}}_{\mathcal{A},\mathcal{D}_{n_2 k_2},\iota,\mathsf{Pgen}}(\lambda) \approx_\lambda 0$, where $\mathsf{Adv}^{\mathrm{wkermdh}}_{\mathcal{A},\mathcal{D}_{n_2 k_2},\iota,\mathsf{Pgen}}(\lambda) :=$

$$\Pr\left[\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \boldsymbol{A} \leftarrow_\$ \mathcal{D}_{n_2 k_2}; \boldsymbol{c} \leftarrow \mathcal{A}(\mathsf{p}, [\boldsymbol{A}]_\iota) : \boldsymbol{A}^\top \boldsymbol{c} = \boldsymbol{0}_{k_2} \wedge \boldsymbol{c} \neq \boldsymbol{0}_{n_2}\right] .$$

Clearly, $\mathsf{wKerMDH}$ is not stronger (and is potentially weaker) than KerMDH for the same distribution, [ALSZ18]. To illustrate this, the hardness of CDH is a KerMDH assumption, while the hardness of DL is a $\mathsf{wKerMDH}$ assumption.

Nonuniform QA-NIZK in the BPK Model. The original QA-NIZK security definitions, [JR13], were given in the CRS model. The following description of QA-NIZKs in the BPK model is taken from [ALSZ18], and we refer to [ALSZ18] for additional discussions. Since black-box [MR01,APV05] and even auxiliary-input non-black-box [Wee07] NIZK for non-trivial languages in the BPK model is impossible we will give an explicit definition of no-auxiliary-string non-black-box NIZK (or, more precisely, *nonuniform NIZK* [Wee07]). As explained in [ALSZ18], nonuniform NIZK in the BPK model is the same as S-ZK [BFS16] in the CRS model.

As in [BFS16], we will implicitly assume that the system parameters p are generated deterministically from λ; in particular, the choice of p cannot be subverted. A QA-NIZK argument system enables to prove membership in a language defined by a relation $\mathbf{R}_\varrho = \{(\mathsf{x}, \mathsf{w})\}$, which in turn is completely determined by a parameter ϱ sampled from a distribution $\mathcal{D}_\mathsf{p}$.

It is usually assumed that samples from $\mathcal{D}_\mathsf{p}$ are generated by a trusted third party (TTP), see [JR13] for a discussion. For example, in the case of the language $\mathcal{L} = ([1]_1, [x]_1, [y]_1, [xy]_1)$ of DDH tuples, $[x]_1$ is created by the TTP. Instead of TTP, one can have a protocol participant who has self-interest in choosing ϱ securely and not leak corresponding secret. As noted by Jutla and Roy [JR13], one needs to assume that $\mathcal{D}_\mathsf{p}$ is "reasonable"; for example, it should not be the case that all languages $\mathcal{L}_\varrho$ for $\varrho \in \mathcal{D}_\mathsf{p}$ are easy to decide. Thus, in the proof of zero knowledge, we will assume that $\mathcal{D}_\mathsf{p}$ works as a black box and one cannot obtain from it any secret keys. See [ALSZ18] for a thorough discussion.

We will assume implicitly that ϱ contains p and thus not include p as an argument to algorithms that also input ϱ. A distribution $\mathcal{D}_\mathsf{p}$ on $\mathcal{L}_\varrho$ is *witness-sampleable* [JR13] if there exists a PPT algorithm $\mathcal{D}'_\mathsf{p}$ that samples $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}_\varrho$ such that ϱ is distributed according to $\mathcal{D}_\mathsf{p}$, and membership of x in *the parameter language* $\mathcal{L}_\varrho$ can be verified in PPT given w.

While the verifier's public key $\textsc{pk}$ may depend on ϱ (however, we assume that ϱ was not created by the verifier), following [ABLZ17], we require the zero-knowledge simulator to be a single (non-black-box) PPT algorithm that works for the whole collection of relations $\mathbf{R}_\mathsf{p} = \{\mathbf{R}_\varrho\}_{\varrho \in \mathrm{Supp}(\mathcal{D}_\mathsf{p})}$; that is, one usually requires *uniform simulation* (see [JR13] for a discussion). We however accompany the universal simulator with an adversary-dependent extractor. The simulator is not allowed to create new ϱ but has to operate with one given to it as an input.

A tuple of PPT algorithms $\Pi = (\mathsf{Pgen}, \mathsf{K_{bpk}}, \mathsf{V_{pk}}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$ is a *nonuniform zero knowledge QA-NIZK argument system* in the BPK model for a set of witness-relations $\mathbf{R}_\mathsf{p} = \{\mathbf{R}_\varrho\}_{\varrho \in \mathrm{Supp}(\mathcal{D}_\mathsf{p})}$ with ϱ sampled from a distribution $\mathcal{D}_\mathsf{p}$ over associated parameter language $\mathcal{L}_\mathsf{p}$, if the following properties (i-iii) hold. Here, Pgen is the parameter generation algorithm, $\mathsf{K_{bpk}}$ is the public key generation algorithm, $\mathsf{V_{pk}}$ is the public key verification algorithm, P is the prover, V is the verifier, and Sim is the simulator.

(i) **Perfect Completeness:** $\forall \lambda$, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_\mathsf{p}$, and $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}_\varrho$,

$$\Pr\left[\begin{matrix}(\textsc{pk}, \textsc{sk}) \leftarrow \mathsf{K_{bpk}}(\varrho); \pi \leftarrow \mathsf{P}(\varrho, \textsc{pk}; \mathsf{x}, \mathsf{w}) : \\ \mathsf{V_{pk}}(\varrho, \textsc{pk}) = 1 \wedge \mathsf{V}(\varrho, \textsc{pk}; \mathsf{x}, \pi) = 1\end{matrix}\right] = 1 \ .$$

(ii) **Computational Quasi-Adaptive Soundness:** $\forall$ PPT $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{snd}}_{\mathcal{A},\Pi}(\lambda) \approx_\lambda 0$, where $\mathsf{Adv}^{\mathrm{snd}}_{\mathcal{A},\Pi}(\lambda) :=$

$$\Pr\left[\begin{matrix}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \varrho \leftarrow_\$ \mathcal{D}_\mathsf{p}; (\textsc{pk}, \textsc{sk}) \leftarrow \mathsf{K_{bpk}}(\varrho); (\mathsf{x}, \pi) \leftarrow \mathcal{A}(\varrho, \textsc{pk}) : \\ \mathsf{V}(\varrho, \textsc{pk}; \mathsf{x}, \pi) = 1 \wedge \neg(\exists \mathsf{w} : \mathbf{R}_\varrho(\mathsf{x}, \mathsf{w}))\end{matrix}\right] \ .$$

```
$\mathsf{Main}_{\mathsf{Z},\mathcal{A}}(\varrho)$
$r \leftarrow_{\$} \mathsf{RND}(\mathsf{Z})$;
$(\mathrm{PK}, \mathsf{aux}_{\mathsf{Z}}) \leftarrow \mathsf{Z}(\varrho; r)$;
$\mathrm{SK} \leftarrow \mathsf{Ext}_{\mathsf{Z}}(\varrho; r)$;
$b \leftarrow_{\$} \{0,1\}$;
$b' \leftarrow \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\varrho; \mathrm{PK}, \mathsf{aux}_{\mathsf{Z}})$;
**return** $\mathsf{V}_{\mathsf{pk}}(\varrho; \mathrm{PK}) = 1 \wedge b' = b$;
```

```
$\mathsf{O}_0(\mathsf{x}, \mathsf{w})$:
**if** $(\mathsf{x}, \mathsf{w}) \not\in \mathbf{R}_\varrho$ **then return** $\perp$;
**else return** $\pi \leftarrow \mathsf{P}(\varrho, \mathrm{PK}; \mathsf{x}, \mathsf{w})$; **fi**

$\mathsf{O}_1(\mathsf{x}, \mathsf{w})$:
**if** $(\mathsf{x}, \mathsf{w}) \not\in \mathbf{R}_\varrho$ **then return** $\perp$;
**else return** $\pi_{\mathsf{Sim}} \leftarrow \mathsf{Sim}(\varrho, \mathrm{PK}, \mathrm{SK}; \mathsf{x})$; **fi**
```

Fig. 1. Experiment $\mathsf{Exp}^{\mathsf{nuzk}}_{\mathsf{Z},\mathcal{A}}(\varrho)$ for $b \in \{0,1\}$

(iii) **Statistical Nonuniform Zero Knowledge:** for any PPT subverter Z there exists a PPT $\mathsf{Ext}_{\mathsf{Z}}$, s.t. $\forall \lambda$, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, and computationally unbounded $\mathcal{A}$, $|\mathsf{Exp}^{\mathsf{nuzk}}_{\mathsf{Z},\mathcal{A}}(\varrho) - \frac{1}{2}| \approx_\lambda 0$, where $\mathsf{Exp}^{\mathsf{nuzk}}_{\mathsf{Z},\mathcal{A}}(\varrho)$ is depicted in Fig. 1.

The extractor defined by a knowledge assumption never works with probability 1. However, if it works then in our constructions the simulation will be perfect. For the sake of simplicity, we will not formalize this as perfect zero knowledge. (One reason for this is that is that differently from [ABLZ17], the secret key extracted by $\mathsf{Ext}_{\mathsf{Z}}$ is not unique in our case, see discussion in [ALSZ18].)

The existence of V_{pk} is not needed in the CRS model, assuming the CRS creator is trusted by the prover, and thus V_{pk} was not included in the prior art QA-NIZK definitions. Since soundness is proved in the case PK is chosen correctly (by the verifier or a trusted third party, trusted by her), V does not need to execute V_{pk}. However, P must run V_{pk} and the simulator is only required to correctly simulate π in the case V_{pk} accepts PK.

Kiltz-Wee QA-NIZK in the BPK model. Kiltz and Wee [KW15] described a very efficient QA-NIZK Π'_{as} for linear subspaces. Abdolmaleki *et al.* [ALSZ18] modified Π'_{as} and proved that the resulting QA-NIZK Π_{bpk} is nonuniform NIZK in the BPK model, assuming a novel KW-KE knowledge assumption. In addition, [ALSZ18] proved that the KW-KE assumption holds in the subversion generic bilinear group model of [BFS16,ABLZ17] (S-GBGM, named GBGM with hashing in [BFS16]). Since S-GBGM is a more realistic model than GBGM, KW-KE holds also in the GBGM. The soundness of Π'_{as} holds in the BPK model under a suitable KerMDH assumption for any $k_2 \geq 1$; one obtain optimal efficiency when $k_2 = 1$. Fig. 2 describes the nonuniform QA-NIZK argument system Π_{bpk} from [ALSZ18]. Here, as observed in [ALSZ18], the correctness of $[\boldsymbol{P}]_1$ is needed to guarantee zero knowledge and $[\bar{\boldsymbol{A}}, \boldsymbol{C}]_2$ is needed to guarantee soundness [ALSZ18]. Apart from restating Π'_{as} by using the terminology of the BPK model, Π_{bpk} differs from Π'_{as} only by having an additional entry $\mathrm{PK}_{\mathsf{V}_{\mathsf{pk}}}$ in the PK and by including the V_{pk} algorithm. For the sake of completeness, we will next state the security assumptions and main security results of [ALSZ18].

```
isinvertible($[\bar{\boldsymbol{A}}]_2, \textsc{pk}_{\mathsf{Vpk}}$) // $\bar{\boldsymbol{A}} = (a_{ij})$
    if $k_2 = 1$ then check $\textsc{pk}_{\mathsf{Vpk}} = \epsilon \wedge [a_{11}]_2 \neq [0]_2$;
    else check $\textsc{pk}_{\mathsf{Vpk}} = [a_{11}, a_{12}]_1 \in \mathbb{G}_1^{1\times 2} \wedge [a_{11}]_1 \bullet [1]_2 = [1]_1 \bullet [a_{11}]_2 \wedge$
        $[a_{12}]_1 \bullet [1]_2 = [1]_1 \bullet [a_{12}]_2 \wedge [a_{11}]_1 \bullet [a_{22}]_2 - [a_{12}]_1 \bullet [a_{21}]_2 \neq [0]_T$; fi
```

```
$\mathsf{K}_{\mathsf{bpk}}([\boldsymbol{M}]_1 \in \mathbb{G}_1^{n_1 \times k_1})$: $\boldsymbol{A} \leftarrow_{\$} \mathcal{D}_{k_2}$; $\boldsymbol{K} \leftarrow_{\$} \mathbb{Z}_p^{n_1 \times k_2}$; $\boldsymbol{C} \leftarrow \boldsymbol{K}\bar{\boldsymbol{A}} \in \mathbb{Z}_p^{n_1 \times k_2}$; $[\boldsymbol{P}]_1 \leftarrow$
    $[\boldsymbol{M}]_1^\top \boldsymbol{K} \in \mathbb{G}_1^{k_1 \times k_2}$;
    if $k_2 = 1$ then $\textsc{pk}_{\mathsf{Vpk}} \leftarrow \epsilon$; elseif $k_2 = 2$ then $\textsc{pk}_{\mathsf{Vpk}} \leftarrow [a_{11}, a_{12}]_1$; fi ;
    $\textsc{pk}_{\mathsf{zk}} \leftarrow [\boldsymbol{P}]_1$; $\textsc{pk}_{\mathsf{snd}} \leftarrow [\bar{\boldsymbol{A}}, \boldsymbol{C}]_2$; $\textsc{pk} \leftarrow (\textsc{pk}_{\mathsf{zk}}, \textsc{pk}_{\mathsf{snd}}, \textsc{pk}_{\mathsf{Vpk}})$; $\textsc{sk} \leftarrow \boldsymbol{K}$;
    return $(\textsc{pk}, \textsc{sk})$;
$\mathsf{P}([\boldsymbol{M}]_1, \textsc{pk}, [\boldsymbol{y}]_1, \mathbf{w})$: return $[\boldsymbol{\pi}]_1 \leftarrow [\boldsymbol{P}]_1^\top \mathbf{w} \in \mathbb{G}_1^{k_2}$;
$\mathsf{Sim}([\boldsymbol{M}]_1, \textsc{pk}, \textsc{sk}; [\boldsymbol{y}]_1)$: // sk is extracted from Z by using a knowledge assumption;
    return $[\boldsymbol{\pi}_{\mathsf{Sim}}]_1 \leftarrow \boldsymbol{K}^\top [\boldsymbol{y}]_1 \in \mathbb{G}_1^{k_2}$;
$\mathsf{V}([\boldsymbol{M}]_1, \textsc{pk}; [\boldsymbol{y}]_1, [\boldsymbol{\pi}]_1)$ : check that $[\boldsymbol{y}]_1^\top \bullet [\boldsymbol{C}]_2 = [\boldsymbol{\pi}]_1^\top \bullet [\bar{\boldsymbol{A}}]_2$; // $\in \mathbb{G}_T^{1 \times k_2}$
$\mathsf{V}_{\mathsf{pk}}([\boldsymbol{M}]_1, \textsc{pk})$: Return 1 only if the following checks all succeed:
        $\textsc{pk} = (\textsc{pk}_{\mathsf{zk}}, \textsc{pk}_{\mathsf{snd}}, \textsc{pk}_{\mathsf{Vpk}}) \wedge \textsc{pk}_{\mathsf{zk}} = [\boldsymbol{P}]_1 \wedge \textsc{pk}_{\mathsf{snd}} = [\bar{\boldsymbol{A}}, \boldsymbol{C}]_2$;
        $[\boldsymbol{M}]_1 \in \mathbb{G}_1^{n_1 \times k_1} \wedge [\boldsymbol{P}]_1 \in \mathbb{G}_1^{k_1 \times k_2} \wedge [\bar{\boldsymbol{A}}]_2 \in \mathbb{G}_2^{k_2 \times k_2} \wedge [\boldsymbol{C}]_2 \in \mathbb{G}_2^{n_1 \times k_2}$;
    $(*)$ $[\boldsymbol{M}]_1^\top \bullet [\boldsymbol{C}]_2 = [\boldsymbol{P}]_1 \bullet [\bar{\boldsymbol{A}}]_2$;
        isinvertible($[\bar{\boldsymbol{A}}]_2, \textsc{pk}_{\mathsf{Vpk}}$);
```

Fig. 2. Top: auxiliary algorithm isinvertible for $k_2 \in \{1, 2\}$. Bottom: nonuniform QA-NIZK Π_{bpk} [ALSZ18] for $[\boldsymbol{y}]_1 = [\boldsymbol{M}]_1\mathbf{w}$ in the BPK model, where $k_2 \in \{1, 2\}$

The following assumption is tautological to the Kiltz-Wee QA-NIZK Π'_{as}. Here, $\textsc{pk}_{\mathsf{Vpk}}$ denotes the part of $\textsc{pk}$ added to $\textsc{pk}$ only to make V_{pk} efficient. As explained in [ALSZ18], $\textsc{pk}_{\mathsf{Vpk}} = \epsilon$ if $k_2 = 1$; in general, $\textsc{pk}_{\mathsf{Vpk}}$ contains information needed to efficiently check that $\bar{\boldsymbol{A}}$ is invertible..

Definition 1. *Fix* $k_2 \in \{1, 2\}$ *and* $n_1 > k_1 \geq 1$. *Let* V_{pk} *be as in Fig. 2. Then* $(\mathcal{D}_{\mathsf{p}}, k_2)$-$\mathsf{KW}$-$\mathsf{KE}_{\mathbb{G}_1}$ *(resp.,* $\boxed{(\mathcal{D}_{\mathsf{p}}, k_2)\text{-}\mathsf{SKW}\text{-}\mathsf{KE}_{\mathbb{G}_1}}$*) holds relative to* Pgen *if for any* $\mathsf{p} \in \mathrm{im}(\mathsf{Pgen}(1^\lambda))$ *and PPT adversary* $\mathcal{A}$*, there exists a PPT extractor* $\mathsf{Ext}_{\mathcal{A}}$*, s.t.*

$$\Pr\left[\begin{array}{l}[\boldsymbol{M}]_1 \leftarrow_{\$} \mathcal{D}_{\mathsf{p}}; r \leftarrow_{\$} \mathsf{RND}(\mathcal{A}); \textsc{pk} \leftarrow \mathcal{A}([\boldsymbol{M}]_1; r);\\ \boldsymbol{K} \leftarrow \mathsf{Ext}_{\mathcal{A}}([\boldsymbol{M}]_1; r) : \textsc{pk} = ([\boldsymbol{P}]_1, [\bar{\boldsymbol{A}}, \boldsymbol{C}]_2, \textsc{pk}_{\mathsf{Vpk}}) \wedge\\ \mathsf{V}_{\mathsf{pk}}([\boldsymbol{M}]_1, \textsc{pk}) = 1 \wedge (\boldsymbol{P} \neq \boldsymbol{M}^\top \boldsymbol{K} \boxed{\vee \boldsymbol{C} \neq \boldsymbol{K}\bar{\boldsymbol{A}}})\end{array}\right] \approx_\lambda 0 \enspace ,$$

Here, the $\boxed{\textit{boxed}}$ *part is only present in the definition of* SKW-KE.

Theorem 1 (S-GBGM Security of KW-KE and SKW-KE, [ALSZ18]). *Let* $k_2 \in \{1, 2\}$ *and* $k_2/p = \mathsf{negl}(\lambda)$. *Then*

(i) the $(\mathcal{D}_{\mathsf{p}}, k_2)$-$\mathsf{KW}$-$\mathsf{KE}_{\mathbb{G}_1}$ *assumption holds in the S-GBGM.*

(ii) assuming that the $\mathcal{D}_{\mathsf{p}}$*-*$\mathsf{wKerMDH}_{\mathbb{G}_1}$ *assumption holds in the S-GBGM (thus,* $\varrho = [\boldsymbol{M}]_1$ *has been chosen from a* $\mathsf{wKerMDH}_{\mathbb{G}_1}$*-hard distribution) against* $\tau(\lambda)$*-time generic adversaries, the* $(\mathcal{D}_{\mathsf{p}}, k_2)$*-*$\mathsf{SKW\text{-}KE}_{\mathbb{G}_1}$ *assumption holds in the S-GBGM against* $O(\tau(\lambda))$*-time generic adversaries.*

Theorem 2 (Security of Π_{bpk}, [ALSZ18]). *Let* Π_{bpk} *be the QA-NIZK argument system for linear subspaces from Fig. 2. Let* $k_2 \in \{1, 2\}$*. The following statements hold in the BPK model.*

(i) Π_{bpk} *is perfectly complete.*
(ii) If the $(\mathcal{D}_{\mathsf{p}}, k_2)$*-*$\mathsf{SKW\text{-}KE}_{\mathbb{G}_1}$ *assumption holds relative to* Pgen *(and thus also assuming* $\mathcal{D}_{\mathsf{p}}$*-*wKerMDH*, i.e., that* $[\boldsymbol{M}]_1$ *comes from a* wKerMDH*-hard distribution) then* Π_{bpk} *is statistically nonuniform zero knowledge.*
(iii) If the $(\mathcal{D}_{\mathsf{p}}, k_2)$*-*$\mathsf{KW\text{-}KE}_{\mathbb{G}_1}$ *assumption holds relative to* Pgen *then* Π_{bpk} *is statistically nonuniform zero knowledge.*
(iv) Let $k_2 = 1$ *(resp.,* $k_2 = 2$*). If the* $\mathcal{D}_{k_2}$*-*KerMDH *(resp.,* $\mathcal{D}_{k_2}$*-*SKerMDH*) assumption holds relative to* Pgen *then* Π_{bpk} *is computationally quasi-adaptively sound.*

In the case (i), the stronger $\mathsf{SKW\text{-}KE}_{\mathbb{G}_1}$ assumption guarantees that the whole public key PK, and not only the part $\mathrm{PK}_{\mathsf{zk}}$ of PK needed to achieve zero-knowledge, is correctly constructed. To prove zero-knowledge, the case (ii) and thus $\mathsf{KW\text{-}KE}_{\mathbb{G}_1}$ is sufficient.

3 Key-and-Argument-Updatable QA-NIZK: Definitions

Following [ALSZ18]), we will consider QA-NIZK in the BPK model and thus with a public-key updating (and not CRS-updating algorithm like in [GKM$^+$18]!). In addition, we allow updating of a previously created argument to the new public key, obtaining what we will call a *key-and-argument-updatable* QA-NIZK. As in the case of [GKM$^+$18], the updatable public key PK and the corresponding secret key will be "shared" by more than one party. Thus, executing multiple updates of PK by independent parties means that PK is not created by a single verifier. To achieve soundness it suffices that V (or an entity trusted by her) was one of the parties involved in the creation or updating of the public key. It even suffices if at least one key-updater (up to the currently last available updated proof) not collaborate with the corresponding proof-updater.

This moves us out from designated-verifier arguments, typical for the BPK model, to (somewhat-)transferable arguments. In particular, the CRS model corresponds to the case where PK belongs to a universally trusted third party (TTP); updating the public key of TTP decreases trust requirements in the TTP. For example, the PK could originally belong to the TTP, then updated by one verifier, and then by another interested verifier.

New Algorithms. An (argument-)updatable nonuniform QA-NIZK has the following additional PPT algorithms on top of $(\mathsf{Pgen}, \mathsf{K}_{\mathsf{bpk}}, \mathsf{V}_{\mathsf{pk}}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$:

$\mathsf{K}_{\mathsf{upd}}(\varrho, \mathbf{PK})$**:** a randomized *key updater* algorithm that, given an old PK, generates a new updated public key PK', and returns $(\mathrm{PK}', \widehat{\mathrm{SK}})$ where $\widehat{\mathrm{SK}}$ is some trapdoor about the PK-update.

$\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathbf{PK}, \mathbf{PK}')$**:** a deterministic *key-update verifier* algorithm that, given both an old PK and an updated PK', verifies that PK' is a correct update of PK.

$\mathsf{P}_{\mathsf{upd}}(\varrho, \mathbf{PK}, \mathbf{PK}'; \mathsf{x}, \mathsf{w}, \pi)$**:** a possibly randomized *argument-updater* algorithm that, given $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}_\varrho$, an argument π with old public key PK and the updated public key PK', outputs an argument π' that corresponds to the updated public key PK'.

$\mathsf{P}_{\mathsf{upd}}$ must be executable without the knowledge of either SK, SK' (the secret key corresponding to PK'), or any trapdoor about the update. This means $\mathsf{P}_{\mathsf{upd}}$ can be used to update either a (prover-generated or a simulated) argument but only honestly, i.e., when knowing the witness.

$\mathsf{Sim}_{\mathsf{upd}}(\varrho, \widehat{\mathbf{SK}}; \mathsf{x}, \pi)$**:** a (randomized) *argument-update simulator* algorithm that, given an argument π with an old public key PK and a PK-update trapdoor from π to π', outputs an argument π' with an updated public key PK'.

$\mathsf{Sim}_{\mathsf{upd}}$ must be executable without the knowledge of either w, SK, or SK' (the secret keys corresponding to PK and PK', respectively). This means $\mathsf{Sim}_{\mathsf{upd}}$ can be used to update a (prover-generated or a simulated) argument but only when knowing some trapdoor $\widehat{\mathrm{SK}}$ about the key-update. $\mathsf{Sim}_{\mathsf{upd}}$ can have more inputs (like PK), but in our constructions we do need them.

$\mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathbf{PK}, \mathbf{PK}'; \mathsf{x}, \pi, \pi')$**:** a deterministic *argument-update verifier* algorithm that verifies that π' is a correct (updated either by $\mathsf{P}_{\mathsf{upd}}$ or $\mathsf{Sim}_{\mathsf{upd}}$ on correct inputs) update of π when the public key was updated from PK to PK'.

New Security Requirements. We have a number of new additional security requirements that accompany the new algorithms. They include requirements that guarantee that the standard definitions of completeness, (computational) soundness, and (statistical) zero-knowledge also hold after the key or the key-and-argument updates. We complete them with the various hiding requirements that guarantee that an updated key (and argument) are indistinguishable from freshly generated key (and argument), assuming that either the pre-update key (and argument) were honestly created or the update was honest. While hiding is a natural security objective by itself, we will see that it allows us to get general reductions between soundness (and key/argument-update soundness) and nonuniform soundness (and key/argument-update zero knowledge).

We will consider two versions of argument-update soundness. Argument-update soundness (I) holds when PK was created honestly but the updater is malicious, and argument-update soundness (II) holds when PK was created maliciously but the updater is honest. Argument-update soundness (I) (resp., (II)) is defined in the case when PK and π were created honestly (resp., updated honestly), since it is impossible to get both subversion-soundness (which corresponds to the case both the PK creator and the updater are malicious) and zero knowledge, [BFS16]. However, we want to still guarantee soundness even in the case when only one of the the key and argument updaters was honest but various verification algorithms accept the updates of other updaters.

In the case of key-update zero knowledge, we are interested in the case when only the public key has been updated but the update could have been done maliciously. Since the argument is not updated, we require that an argument and simulated argument given with the updated key (where both the old key and the key-update verify) are indistinguishable.

Similarly, in the case of argument-update zero knowledge, the key update does not depend on the witness and may be done maliciously (possibly not by the prover). However, the argument is updated by the prover who uses the witness but does not have know the key-update trapdoor $\widehat{\mathrm{SK}}$. This motivates the use of $\mathsf{K}_{\mathsf{upd}}$ and $\mathsf{Sim}_{\mathsf{upd}}$ in the update process in $\mathsf{Exp}^{\mathsf{auzk}}_{\mathsf{Z},\mathcal{A}}(\varrho)$. Thus, key-update zero knowledge and argument-update zero knowledge are different notions and have to be handled separately.

All security notions will be given for a single update. However, all notions can be composed over several updates, and the corresponding properties can then be proved by using standard hybrid arguments. We will omit further discussion.

Key-update completeness: for any λ, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho)$, $(\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK})$, it holds that $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1$.
Moreover, if $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1$ then $\mathsf{V}_{\mathsf{pk}}(\varrho; \mathrm{PK}) = 1$ iff $\mathsf{V}_{\mathsf{pk}}(\varrho; \mathrm{PK}') = 1$.

Argument-update completeness: for any λ, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho)$, $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}_\varrho$, $\pi \leftarrow \mathsf{P}(\varrho, \mathrm{PK}; \mathsf{x}, \mathsf{w})$, $(\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK})$, $\pi' \leftarrow \mathsf{P}_{\mathsf{upd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \mathsf{w}, \pi)$, it holds that $\mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi, \pi') = 1$.
Moreover, if $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1$ and $\mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi, \pi') = 1$ then $\mathsf{V}(\varrho, \mathrm{PK}; \mathsf{x}, \pi) = 1$ iff $\mathsf{V}(\varrho, \mathrm{PK}'; \mathsf{x}, \pi') = 1$.

Simulator-update completeness: $\forall \lambda$, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho)$, $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}_\varrho$, $\pi_{\mathsf{Sim}} \leftarrow \mathsf{Sim}(\varrho, \mathrm{PK}, \mathrm{SK}; \mathsf{x})$, $(\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK})$, $\pi' \leftarrow \mathsf{Sim}_{\mathsf{upd}}(\varrho, \widehat{\mathrm{SK}}; \mathsf{x}, \pi)$, it holds that $\mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi, \pi') = 1$.
Moreover, if $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1$ and $\mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi, \pi') = 1$ then $\mathsf{V}(\varrho, \mathrm{PK}; \mathsf{x}, \pi) = 1$ iff $\mathsf{V}(\varrho, \mathrm{PK}'; \mathsf{x}, \pi') = 1$.

Key-update hiding: for any λ, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, and $\varrho \in \mathcal{D}_{\mathsf{p}}$, if $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho)$ and $(\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK})$, then $\mathrm{PK}' \approx_\lambda \mathsf{K}_{\mathsf{bpk}}(\varrho)$.

Strong key-update hiding: for any λ, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, and $(\mathrm{PK}, \mathrm{PK}')$: $\mathrm{PK}' \approx_\lambda \mathsf{K}_{\mathsf{bpk}}(\varrho)$ holds if either

1. the old public key was honestly generated and the key-update verifies: $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho)$ and $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1$, or
2. the old public key verifies and the key-update was honest: $\mathsf{V}_{\mathsf{pk}}(\varrho, \mathrm{PK}) = 1$ and $(\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK})$.

Argument-update hiding: for any λ, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho)$, $(\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK})$, $\pi \leftarrow \mathsf{P}(\varrho, \mathrm{PK}; \mathsf{x}, \mathsf{w})$, $\pi_{\mathsf{Sim}} \leftarrow \mathsf{Sim}(\varrho, \mathrm{PK}, \mathrm{SK}; \mathsf{x})$, $\pi' \leftarrow \mathsf{P}_{\mathsf{upd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \mathsf{w}, \pi)$, $\pi'_{\mathsf{Sim}} \leftarrow \mathsf{Sim}_{\mathsf{upd}}(\varrho, \widehat{\mathrm{SK}}; \mathsf{x}, \pi)$, it holds that $\pi' \approx_\lambda \mathsf{P}(\varrho, \mathrm{PK}'; \mathsf{x}, \mathsf{w})$ and $\pi'_{\mathsf{Sim}} \approx_\lambda \mathsf{Sim}(\varrho, \mathrm{PK}', \mathrm{SK}'; \mathsf{x})$.

Strong argument-update hiding: $\forall \lambda$, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}_\varrho$, and $(\mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi, \pi')$: $\mathrm{PK}' \approx_\lambda \mathsf{K}_{\mathsf{bpk}}(\varrho)$, $\pi' \approx_\lambda \mathsf{P}(\varrho, \mathrm{PK}'; \mathsf{x}, \mathsf{w})$, and $\pi'_{\mathsf{Sim}} \approx_\lambda \mathsf{Sim}(\varrho, \mathrm{PK}', \mathrm{SK}'; \mathsf{x})$ hold if either

1. the old public key and argument were honestly generated and the updates verify: $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho)$, $\pi \leftarrow \mathsf{P}(\varrho, \mathrm{PK}; \mathsf{x}, \mathsf{w})$, $\pi_{\mathsf{Sim}} \leftarrow$

$\mathsf{Sim}(\varrho, \mathrm{PK}, \mathrm{SK}; \mathsf{x})$, $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1$, $\mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi, \pi') = 1$, and $\mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi_{\mathsf{Sim}}, \pi'_{\mathsf{Sim}}) = 1$, or

2. the old public key and argument verify and the updates were honestly generated: $\mathsf{V}_{\mathsf{pk}}(\varrho, \mathrm{PK}) = 1$, $\mathsf{V}(\varrho, \mathrm{PK}; \mathsf{x}, \pi) = 1$, $\mathsf{V}(\varrho, \mathrm{PK}; \mathsf{x}, \pi_{\mathsf{Sim}}) = 1$, $(\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK})$, $\pi' \leftarrow \mathsf{P}_{\mathsf{upd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \mathsf{w}, \pi)$, and $\pi'_{\mathsf{Sim}} \leftarrow \mathsf{Sim}_{\mathsf{upd}}(\varrho, \widehat{\mathrm{SK}}; \mathsf{x}, \pi_{\mathsf{Sim}})$.

(Computational quasi-adaptive) key-update soundness (I): $\mathsf{Adv}^{\mathrm{sndku1}}_{\mathcal{A},\Pi}(\lambda) \approx_\lambda 0$, where $\mathsf{Adv}^{\mathrm{sndku1}}_{\mathcal{A},\Pi}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \varrho \leftarrow_\$ \mathcal{D}_{\mathsf{p}}; (\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho); (\mathrm{PK}', \mathsf{x}, \pi') \leftarrow \mathcal{A}(\varrho, \mathrm{PK}) : \\ \mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1 \wedge \mathsf{V}(\varrho, \mathrm{PK}'; \mathsf{x}, \pi') = 1 \wedge \neg(\exists \mathsf{w} : \mathbf{R}_\varrho(\mathsf{x}, \mathsf{w}))\end{array}\right] .$$

(Computational quasi-adaptive) key-update soundness (II): $\mathsf{Adv}^{\mathrm{sndku2}}_{\mathcal{A},\Pi}(\lambda) \approx_\lambda 0$, where $\mathsf{Adv}^{\mathrm{sndku2}}_{\mathcal{A},\Pi}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \varrho \leftarrow_\$ \mathcal{D}_{\mathsf{p}}; \mathrm{PK} \leftarrow \mathcal{A}(\varrho); (\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK}); \\ \pi' \leftarrow \mathcal{A}(\mathrm{PK}'); \mathsf{V}_{\mathsf{pk}}(\varrho, \mathrm{PK}) = 1 \wedge \mathsf{V}(\varrho, \mathrm{PK}'; \mathsf{x}, \pi') = 1 \wedge \neg(\exists \mathsf{w} : \mathbf{R}_\varrho(\mathsf{x}, \mathsf{w}))\end{array}\right] .$$

(Computational quasi-adaptive) key-update soundness: iff both key-update soundness (I) and key-update soundness (II) hold.

(Computational quasi-adaptive) argument-update soundness (I): $\mathsf{Adv}^{\mathrm{sndpu1}}_{\mathcal{A},\Pi}(\lambda) \approx_\lambda 0$, where $\mathsf{Adv}^{\mathrm{sndpu1}}_{\mathcal{A},\Pi}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \varrho \leftarrow_\$ \mathcal{D}_{\mathsf{p}}; (\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{K}_{\mathsf{bpk}}(\varrho); \\ (\mathrm{PK}', \mathsf{x}, \pi, \pi') \leftarrow \mathcal{A}(\varrho, \mathrm{PK}) : \mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1 \wedge \\ \mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi, \pi') = 1 \wedge \mathsf{V}(\varrho, \mathrm{PK}; \mathsf{x}, \pi) = 1 \wedge \neg(\exists \mathsf{w} : \mathbf{R}_\varrho(\mathsf{x}, \mathsf{w}))\end{array}\right] .$$

(Computational quasi-adaptive) argument-update soundness (II): $\mathsf{Adv}^{\mathrm{sndpu2}}_{\mathcal{A},\Pi}(\lambda) \approx_\lambda 0$, where $\mathsf{Adv}^{\mathrm{sndpu2}}_{\mathcal{A},\Pi}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \varrho \leftarrow_\$ \mathcal{D}_{\mathsf{p}}; (\mathrm{PK}, \pi) \leftarrow \mathcal{A}(\varrho); (\mathrm{PK}', \widehat{\mathrm{SK}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \mathrm{PK}); \\ \pi' \leftarrow \mathsf{P}_{\mathsf{upd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \mathsf{w}, \pi); \mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1 \wedge \\ \mathsf{V}_{\mathsf{Pupd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \pi, \pi') = 1 \wedge \mathsf{V}(\varrho, \mathrm{PK}; \mathsf{x}, \pi) = 1 \wedge \neg(\exists \mathsf{w} : \mathbf{R}_\varrho(\mathsf{x}, \mathsf{w}))\end{array}\right] .$$

(Computational quasi-adaptive) argument-update soundness: iff both argument-update soundness (I) and argument-update soundness (II) hold.

(Statistical nonuniform) key-update zero knowledge: for any PPT subverter Z there exists a PPT $\mathsf{Ext}_{\mathsf{Z}}$, such that for any λ, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, and computationally unbounded $\mathcal{A}$, $|\mathsf{Exp}^{\mathsf{kuzk}}_{\mathsf{Z},\mathcal{A}}(\varrho) - \frac{1}{2}| \approx_\lambda 0$, where $\mathsf{Exp}^{\mathsf{kuzk}}_{\mathsf{Z},\mathcal{A}}(\varrho)$ is depicted in Fig. 3.

(Statistical nonuniform) argument-update zero knowledge: for any PPT subverter Z there exists a PPT $\mathsf{Ext}_{\mathsf{Z}}$, such that for any λ, $\mathsf{p} \in \mathsf{Pgen}(1^\lambda)$, $\varrho \in \mathcal{D}_{\mathsf{p}}$, and computationally unbounded $\mathcal{A}$, $|\mathsf{Exp}^{\mathsf{auzk}}_{\mathsf{Z},\mathcal{A}}(\varrho) - \frac{1}{2}| \approx_\lambda 0$, where $\mathsf{Exp}^{\mathsf{auzk}}_{\mathsf{Z},\mathcal{A}}(\varrho)$ is depicted in Fig. 4.

```
$\mathsf{Main}_{\mathsf{Z},\mathcal{A}}(\varrho)$
$r \leftarrow_{\$} \mathsf{RND}(\mathsf{Z})$;
$(\mathrm{PK}, \mathrm{PK}', \mathsf{aux}_{\mathsf{Z}}) \leftarrow \mathsf{Z}(\varrho; r)$;
$\mathrm{SK}' \leftarrow \mathsf{Ext}_{\mathsf{Z}}(\varrho; r)$;
$b \leftarrow_{\$} \{0, 1\}$;
$b' \leftarrow \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\varrho; \mathrm{PK}, \mathrm{PK}', \mathsf{aux}_{\mathsf{Z}})$;
return $\mathsf{V}_{\mathsf{pk}}(\varrho; \mathrm{PK}) = 1 \wedge$
    $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1 \wedge b' = b$;

$\mathsf{O}_0(\mathsf{x}, \mathsf{w})$:
if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R}_\varrho$ then return $\perp$;
else $\pi' \leftarrow \mathsf{P}(\varrho, \mathrm{PK}'; \mathsf{x}, \mathsf{w})$; return $\pi'$; fi

$\mathsf{O}_1(\mathsf{x}, \mathsf{w})$:
if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R}_\varrho$ then return $\perp$;
else $\pi'_{\mathsf{Sim}} \leftarrow \mathsf{Sim}(\varrho, \mathrm{PK}', \mathrm{SK}'; \mathsf{x})$; return $\pi'_{\mathsf{Sim}}$; fi
```

Fig. 3. Experiment $\mathsf{Exp}^{\mathsf{kuzk}}_{\mathsf{Z},\mathcal{A}}(\varrho)$ for $b \in \{0, 1\}$

```
$\mathsf{Main}_{\mathsf{Z},\mathcal{A}}(\varrho)$
$r \leftarrow_{\$} \mathsf{RND}(\mathsf{Z})$;
$(\mathrm{PK}, \mathrm{PK}', \mathsf{aux}_{\mathsf{Z}}) \leftarrow \mathsf{Z}(\varrho; r)$;
$\widehat{\mathrm{SK}} \leftarrow \mathsf{Ext}_{\mathsf{Z}}(\varrho; r)$;
$b \leftarrow_{\$} \{0, 1\}$;
$b' \leftarrow \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\varrho; \mathrm{PK}, \mathrm{PK}', \mathsf{aux}_{\mathsf{Z}})$;
return $\mathsf{V}_{\mathsf{pk}}(\varrho; \mathrm{PK}) = 1 \wedge$
    $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \mathrm{PK}, \mathrm{PK}') = 1 \wedge$
    $b' = b$;

$\mathsf{O}_0(\mathsf{x}, \mathsf{w})$:
if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R}_\varrho$ then return $\perp$;
else $\pi \leftarrow \mathsf{P}(\varrho, \mathrm{PK}; \mathsf{x}, \mathsf{w})$;
    $\pi' \leftarrow \mathsf{P}_{\mathsf{upd}}(\varrho, \mathrm{PK}, \mathrm{PK}'; \mathsf{x}, \mathsf{w}, \pi)$; return $(\pi, \pi')$; fi

$\mathsf{O}_1(\mathsf{x}, \mathsf{w})$:
if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R}_\varrho$ then return $\perp$;
else $\pi_{\mathsf{Sim}} \leftarrow \mathsf{Sim}(\varrho, \mathrm{PK}, \mathrm{SK}; \mathsf{x})$;
    $\pi'_{\mathsf{Sim}} \leftarrow \mathsf{Sim}_{\mathsf{upd}}(\varrho, \widehat{\mathrm{SK}}; \mathsf{x}, \pi_{\mathsf{Sim}})$;
    return $(\pi_{\mathsf{Sim}}, \pi'_{\mathsf{Sim}})$; fi
```

Fig. 4. Experiment $\mathsf{Exp}^{\mathsf{auzk}}_{\mathsf{Z},\mathcal{A}}(\varrho)$ for $b \in \{0, 1\}$

Table 1. Relations between security requirements due to Lemmas 1 to 4

Requirement	Follows from
Key-update soundness	key-update complete, sound, strong key-update hiding
Argument-update soundness	argument-update complete, sound, strong key-update hiding, strong argument-update hiding
Key-update zero knowledge	key-update complete, nonuniform zero-knowledge
Argument-update zero knowledge	key-update complete, nonuniform zero-knowledge

We will next show that key-update soundness, argument-update soundness, key-update zero knowledge, argument-update zero knowledge follow from simpler security requirements. This means that in the case of a concrete updatable QA-NIZK, it will suffice to prove computational soundness, nonuniform zero-knowledge, (key-update and argument-update) completeness and strong (key-update and argument-update) hiding. Dependency between security properties is summarized in Table 1.

Lemma 1. *Assume Π is a sound and strongly key-update hiding non-interactive argument system.*

(i) Π is key-update sound (I).
(ii) If Π is additionally key-update complete, then Π is key-update sound (II).

Proof. **(i: key-update sound (I))** By strong key-update hiding, PK' comes from the correct distribution. Thus, by soundness, it is computationally hard to come up with an acceptable argument π' unless x belongs to the language.

(ii: key-update sound (II)) From key-update completeness it follows that in the definition of key-update soundness (II) we can replace the condition $\mathsf{V}_{\mathsf{pk}}(\varrho, \text{PK}) = 1$ with the condition $\mathsf{V}_{\mathsf{pk}}(\varrho, \text{PK}') = 1$. Because of the strong key-update hiding, PK' is indistinguishable from an honestly generated PK'. From soundness, we now obtain key-update soundness (II). $\square$

Lemma 2. *Assume Π is a sound non-interactive argument system.*

(i) Π is argument-update sound (I).
(ii) If Π is also argument-update complete, strongly key-update hiding, and strongly argument-update hiding, then Π is argument-update sound (II).

Proof. **(i: argument-update soundness (I))** Let $\mathcal{A}$ be an adversary against argument-update soundness. We construct the following adversary $\mathcal{B}$ against soundness:

```
$\mathcal{B}(\varrho, \text{PK})$
$(\text{PK}'; \mathsf{x}, \pi, \pi') \leftarrow \mathcal{A}(\varrho, \text{PK});$ return $(\mathsf{x}, \pi);$
```

Clearly, if $\mathcal{A}$ is successful then V accepts π with honestly chosen PK but $\mathsf{x} \notin \mathcal{L}_\varrho$. Thus, $\mathcal{B}$ breaks soundness.

(ii: argument-update soundness (II)) From argument-update completeness it follows that in the definition of argument-update soundness (II) we can replace the condition $\mathsf{V}(\varrho, \text{PK}; \mathsf{x}, \pi) = 1$ with the condition $\mathsf{V}(\varrho, \text{PK}'; \mathsf{x}, \pi') = 1$. Because of the strong key-update hiding, PK' is indistinguishable from an honestly generated PK', and from the strong argument-update hiding we get that π' is indistinguishable from an honestly generated argument given PK'. From soundness, we now obtain argument-update soundness. $\square$

Lemma 3. *Assume Π is a key-update complete and nonuniform zero-knowledge non-interactive argument system. Then Π is key-update zero knowledge.*

Proof. Due to key-update completeness, we have $\mathsf{V}_{\mathsf{pk}}(\varrho; \text{PK}') = 1$. Then, by the nonuniform zero knowledge property, there exist an extractor $\mathsf{Ext}_{\mathsf{Z}}$ that extracts SK' such that $\mathsf{Sim}(\varrho, \text{PK}', \text{SK}'; \mathsf{x}) \approx_\lambda \mathsf{P}(\varrho, \text{PK}'; \mathsf{x}, \mathsf{w})$. This proves the claim. $\square$

Lemma 4. *Assume that $\widehat{\text{SK}}$ is efficiently computable from* SK *and* SK'. *Assume Π is a key-update complete, simulator-update complete, and nonuniform zero-knowledge non-interactive argument system. Then Π is argument-update zero knowledge.*

Proof (Sketch.). By nonuniform zero knowledge, there exists an extractor $\mathsf{Ext}_{\mathsf{Z}_1}$ that extracts $\textsc{sk}$, such that $\pi \leftarrow \mathsf{P}(\varrho, \textsc{pk}; \mathsf{x}, \mathsf{w})$ and $\pi_{\mathsf{Sim}} \leftarrow \mathsf{Sim}(\varrho, \textsc{pk}, \textsc{sk}; \mathsf{x})$ are indistinguishable. By the key-update completeness, $\mathsf{V}_{\mathsf{pk}}(\varrho; \textsc{pk}') = 1$ and thus, by nonuniform zero knowledge, there exists an extractor $\mathsf{Ext}_{\mathsf{Z}_2}$ that extracts $\textsc{sk}'$ such that π' and $\pi''_{\mathsf{Sim}} \leftarrow \mathsf{Sim}(\varrho, \textsc{pk}, \textsc{sk}; \mathsf{x})$ are indistinguishable. By the simulator-update completeness, $\pi''_{\mathsf{Sim}} \approx_\lambda \pi'_{\mathsf{Sim}}$. Thus, the joint distributions (π, π') and $(\pi_{\mathsf{Sim}}, \pi'_{\mathsf{Sim}})$ are indistinguishable. $\square$

Handling Multiple Updates. All security notions given above are for a single update but they can be generalized for many updates, by using standard hybrid arguments. We will omit further details and point to [GKM$^+$18] for more discussion.

4 Updatable Kiltz-Wee QA-NIZK

Intuitively, the public key of Π'_{as} consists of (bracketed) matrices, and thus one could hope to construct a quite simple updating process where all $\textsc{pk}$ elements are updated additively. In such a case, an updater would create a "difference" public key $\widehat{\textsc{pk}}$ (by choosing the trapdoor privately) and update $\textsc{pk}$ by adding $\widehat{\textsc{pk}}$ component-wise to it, $\textsc{pk}' \leftarrow \textsc{pk} + \widehat{\textsc{pk}}$. However, this simple idea does not work since in the case of additive updating (see Fig. 5 for notation), when we define $\bar{\boldsymbol{A}}' \leftarrow \bar{\boldsymbol{A}} + \widehat{\boldsymbol{A}}$, we need to compute $[\boldsymbol{C}']_2 = [\boldsymbol{K}'\bar{\boldsymbol{A}}']_2 = [(\boldsymbol{K} + \widehat{\boldsymbol{K}})(\bar{\boldsymbol{A}} + \widehat{\boldsymbol{A}})]_2 = [\boldsymbol{C}]_2 + [\boldsymbol{K}]_2\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{K}}([\bar{\boldsymbol{A}}]_2 + [\widehat{\boldsymbol{A}}]_2)$. The latter is not possible since $[\boldsymbol{K}]_2$ is not public and thus we have chosen to update the square matrix $\bar{\boldsymbol{A}}$ multiplicatively, $\bar{\boldsymbol{A}}' \leftarrow \bar{\boldsymbol{A}}\widehat{\boldsymbol{A}}$. On the other hand, note that we cannot update $[\boldsymbol{K}]_2$ multiplicatively since $[\boldsymbol{K}]_2$ is (usually) not a square matrix. We depict an updatable version of Π'_{as} in Fig. 5.

Note that (i) $\mathsf{P}_{\mathsf{upd}}$ updates a QA-NIZK argument $[\boldsymbol{\pi}]_1$ by adding to it a honest argument $[\widehat{\boldsymbol{\pi}}]_1$ under the "difference" public key $\widehat{\textsc{pk}}$, given the witness $\mathbf{w}$. Thus, $\mathsf{P}_{\mathsf{upd}}$ can be ran by a party who knows $\mathbf{w}$. (ii) $\mathsf{Sim}_{\mathsf{upd}}$ updates the existing QA-NIZK argument $[\boldsymbol{\pi}]_1$ by adding to it a simulation $[\widehat{\boldsymbol{\pi}}_{\mathsf{Sim}}]_1$ of the argument given $\widehat{\textsc{sk}} = \widehat{\boldsymbol{K}}$ (that is known to the key-updater) as the trapdoor. Thus, $\mathsf{Sim}_{\mathsf{upd}}$ can be ran by the key-updater. Thus, to be sure that at least one update was made by a party who knows the witness, one should make sure that at least one key-updater will not collude with the argument-updater of the same round.

5 Security of $\Pi^{\mathsf{upd}}_{\mathsf{kw}}$

Lemma 5. $\Pi^{\mathsf{upd}}_{\mathsf{kw}}$ *is (i) key-update complete, (ii) argument-update complete, and (iii) simulator-update complete.*

Proof. **(i: Key-update completeness)** We need to show that for $(\textsc{pk}', \widehat{\textsc{sk}}) \leftarrow \mathsf{K}_{\mathsf{upd}}(\varrho, \textsc{pk})$, $\mathsf{V}_{\mathsf{Kupd}}(\varrho, \textsc{pk}, \textsc{pk}') = 1$.

```
$\mathsf{K_{bpk}}$, $\mathsf{P}$, $\mathsf{Sim}$, $\mathsf{V}$, $\mathsf{V_{pk}}$: exactly as in Fig. 2.
------------------------------------------------------------------------
$\mathsf{K_{upd}}([\boldsymbol{M}]_1, \mathbf{PK})$: // Updates $\mathrm{PK} = ([\boldsymbol{P}]_1, [\bar{\boldsymbol{A}}, \boldsymbol{C}]_2)$ to $\mathrm{PK}' = ([\boldsymbol{P}']_1, [\bar{\boldsymbol{A}}', \boldsymbol{C}']_2)$
    $\widehat{\boldsymbol{A}} \leftarrow_{\$} \bar{\mathcal{D}}_{k_2}$; $[\bar{\boldsymbol{A}}']_2 \leftarrow [\bar{\boldsymbol{A}}]_2 \widehat{\boldsymbol{A}}$; $\widehat{\boldsymbol{K}} \leftarrow_{\$} \mathbb{Z}_p^{n_1 \times k_2}$;
    $[\widehat{\boldsymbol{C}}]_2 \leftarrow \widehat{\boldsymbol{K}} [\bar{\boldsymbol{A}}']_2$; $[\boldsymbol{C}']_2 \leftarrow [\boldsymbol{C}]_2 \widehat{\boldsymbol{A}} + [\widehat{\boldsymbol{C}}]_2$;
    $[\widehat{\boldsymbol{P}}]_1 \leftarrow [\boldsymbol{M}]_1^\top \widehat{\boldsymbol{K}}$; $[\boldsymbol{P}']_1 \leftarrow [\boldsymbol{P}]_1 + [\widehat{\boldsymbol{P}}]_1$;
    $\mathrm{PK}_{\mathsf{upd}} \leftarrow ([\widehat{\boldsymbol{A}}]_1, [\widehat{\boldsymbol{A}}, \widehat{\boldsymbol{C}}]_2)$; $\mathrm{PK}' \leftarrow ([\boldsymbol{P}']_1, [\bar{\boldsymbol{A}}', \boldsymbol{C}']_2, \mathrm{PK}_{\mathsf{upd}})$; $\widehat{\mathrm{SK}} \leftarrow \widehat{\boldsymbol{K}}$;
    return $(\mathrm{PK}', \widehat{\mathrm{SK}})$;
------------------------------------------------------------------------
$\mathsf{V_{Kupd}}([\boldsymbol{M}]_1, \mathbf{PK}, \mathbf{PK}')$: $[\widehat{\boldsymbol{P}}]_1 \leftarrow [\boldsymbol{P}' - \boldsymbol{P}]_1$; $\widehat{\mathrm{PK}} \leftarrow ([\widehat{\boldsymbol{P}}]_1, [\bar{\boldsymbol{A}}', \widehat{\boldsymbol{C}}]_2)$;
    if $\mathsf{isinvertible}([\bar{\boldsymbol{A}}]_1, \mathrm{PK}_{\mathsf{V_{pk}}}) \wedge \mathsf{isinvertible}([\widehat{\boldsymbol{A}}]_1, \mathrm{PK}_{\mathsf{V_{pk}}}) \wedge$
      $[\bar{\boldsymbol{A}}']_1 \bullet [1]_2 = [\bar{\boldsymbol{A}}]_1 \bullet [\widehat{\boldsymbol{A}}]_2 \wedge [\widehat{\boldsymbol{A}}]_1 \bullet [1]_2 = [1]_1 \bullet [\widehat{\boldsymbol{A}}]_2 \wedge$
      $[1]_1 \bullet [\boldsymbol{C}']_2 = [\boldsymbol{C}]_2 \bullet [\widehat{\boldsymbol{A}}]_1 + [1]_1 \bullet [\widehat{\boldsymbol{C}}]_2 \wedge \mathsf{V_{pk}}([\boldsymbol{M}]_1, \widehat{\mathrm{PK}}) = 1$
    then return 1 else return 0 fi
------------------------------------------------------------------------
$\mathsf{P_{upd}}([\boldsymbol{M}]_1, \mathbf{PK}; [\boldsymbol{y}]_1, [\mathbf{w}]_1, [\boldsymbol{\pi}]_1)$: $[\widehat{\boldsymbol{\pi}}]_1 \leftarrow [\widehat{\boldsymbol{P}}]_1^\top \mathbf{w}$; return $[\boldsymbol{\pi}']_1 \leftarrow [\boldsymbol{\pi}]_1 + [\widehat{\boldsymbol{\pi}}]_1$;
------------------------------------------------------------------------
$\mathsf{Sim_{upd}}([\boldsymbol{M}]_1, \widehat{\mathbf{SK}}; [\boldsymbol{y}]_1, [\boldsymbol{\pi}]_1)$: $[\boldsymbol{\pi}_{\mathsf{Sim}}]_1 \leftarrow \widehat{\boldsymbol{K}}^\top [\boldsymbol{y}]_1$; return $[\boldsymbol{\pi}']_1 \leftarrow [\boldsymbol{\pi}]_1 + [\widehat{\boldsymbol{\pi}}_{\mathsf{Sim}}]_1$;
------------------------------------------------------------------------
$\mathsf{V_{Pupd}}([\boldsymbol{M}]_1, \mathbf{PK}, \mathbf{PK}'; [\boldsymbol{y}]_1, [\boldsymbol{\pi}]_1, [\boldsymbol{\pi}']_1)$:
    $[\widehat{\boldsymbol{P}}]_1 \leftarrow [\boldsymbol{P}' - \boldsymbol{P}]_1$; $\widehat{\mathrm{PK}} \leftarrow ([\widehat{\boldsymbol{P}}]_1, [\bar{\boldsymbol{A}}', \widehat{\boldsymbol{C}}]_2)$; $[\widehat{\boldsymbol{\pi}}]_1 \leftarrow [\boldsymbol{\pi}' - \boldsymbol{\pi}]_1$;
    if $\mathsf{V}([\boldsymbol{M}]_1, \widehat{\mathrm{PK}}; [\boldsymbol{y}]_1; [\widehat{\boldsymbol{\pi}}]_1) = 1$ then return 1 else return 0 fi
```

Fig. 5. Variant $\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ of Kiltz-Wee QA-NIZK for $[\boldsymbol{y}]_1 = [\boldsymbol{M}]_1 \mathbf{w}$ in the S-ZK model. Here, $k_2 \in \{1, 2\}$.

Really, PK' is defined by $\bar{\boldsymbol{A}}' = \bar{\boldsymbol{A}}\widehat{\boldsymbol{A}}$, $\boldsymbol{C}' = \boldsymbol{C}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{K}}\bar{\boldsymbol{A}}'$, and $\boldsymbol{P}' = \boldsymbol{P} + \boldsymbol{M}^\top \widehat{\boldsymbol{K}}$. Thus, the first two verification equations in $\mathsf{V_{Kupd}}$ hold by the definition of $\bar{\boldsymbol{A}}$ and $\widehat{\boldsymbol{A}}$, and the next three ones hold trivially. Let $\widetilde{\mathrm{PK}} \leftarrow ([\boldsymbol{P}]_1, [\bar{\boldsymbol{A}}, \boldsymbol{C}]_2)$ and $\widetilde{\mathrm{PK}}' \leftarrow ([\boldsymbol{P}']_1, [\bar{\boldsymbol{A}}', \boldsymbol{C}']_2)$. We get $\mathsf{V_{pk}}([\boldsymbol{M}]_1, \widetilde{\mathrm{PK}}') = 1$ from

$$\begin{aligned} \boldsymbol{M}^\top \boldsymbol{C}' - \boldsymbol{P}'\bar{\boldsymbol{A}}' &= \boldsymbol{M}^\top(\boldsymbol{C}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{K}}\bar{\boldsymbol{A}}') - (\boldsymbol{P} + \boldsymbol{M}^\top\widehat{\boldsymbol{K}})\bar{\boldsymbol{A}}\widehat{\boldsymbol{A}} \\ &= \left(\boldsymbol{M}^\top\boldsymbol{C} - \boldsymbol{P}\bar{\boldsymbol{A}}\right)\widehat{\boldsymbol{A}} + \boldsymbol{M}^\top\widehat{\boldsymbol{K}}\left(\bar{\boldsymbol{A}}' - \bar{\boldsymbol{A}}\widehat{\boldsymbol{A}}\right) = \boldsymbol{0} \end{aligned}$$

since $\mathsf{V_{pk}}([\boldsymbol{M}]_1, \widetilde{\mathrm{PK}}) = 1$ (and thus $\boldsymbol{M}^\top\boldsymbol{C} = \boldsymbol{P}\bar{\boldsymbol{A}}$) and $\bar{\boldsymbol{A}}' = \bar{\boldsymbol{A}}\widehat{\boldsymbol{A}}$.

On the other hand, if $\mathsf{V_{Kupd}}([\boldsymbol{M}]_1; \mathrm{PK}, \mathrm{PK}') = 1$ then

$$\begin{aligned} \boldsymbol{0} = \boldsymbol{M}^\top\widehat{\boldsymbol{C}} - \widehat{\boldsymbol{P}}\bar{\boldsymbol{A}}' &= \boldsymbol{M}^\top(\boldsymbol{C}' - \boldsymbol{C}\widehat{\boldsymbol{A}}) - (\boldsymbol{P}' - \boldsymbol{P})\bar{\boldsymbol{A}}' \\ &= (\boldsymbol{M}^\top\boldsymbol{C}' - \boldsymbol{P}'\bar{\boldsymbol{A}}') - (\boldsymbol{M}^\top\boldsymbol{C} - \boldsymbol{P}\bar{\boldsymbol{A}})\widehat{\boldsymbol{A}} \end{aligned}$$

and thus $\mathsf{V_{pk}}([\boldsymbol{M}]_1; \mathrm{PK}') = 1$ iff $\mathsf{V_{pk}}([\boldsymbol{M}]_1; \mathrm{PK}) = 1$.

(ii: Argument-update completeness) Clearly,

$$\boldsymbol{y}^\top\widehat{\boldsymbol{C}} - \widehat{\boldsymbol{\pi}}^\top\bar{\boldsymbol{A}}' = \boldsymbol{y}^\top\widehat{\boldsymbol{K}}\bar{\boldsymbol{A}}' - (\widehat{\boldsymbol{P}}^\top\mathbf{w})^\top\bar{\boldsymbol{A}}' = \mathbf{w}^\top\boldsymbol{M}^\top\widehat{\boldsymbol{K}}\bar{\boldsymbol{A}}' - \mathbf{w}^\top\boldsymbol{M}^\top\widehat{\boldsymbol{K}}\bar{\boldsymbol{A}}' = \boldsymbol{0}$$

and thus $\mathsf{V}([\boldsymbol{M}]_1, \widehat{\mathrm{PK}}; [\boldsymbol{y}]_1; [\widehat{\boldsymbol{\pi}}]_1) = 1$. On the other hand,

$$\boldsymbol{y}^\top\widehat{\boldsymbol{C}} - \widehat{\boldsymbol{\pi}}^\top\bar{\boldsymbol{A}}' = \boldsymbol{y}^\top(\boldsymbol{C}' - \boldsymbol{C}\widehat{\boldsymbol{A}}) - (\boldsymbol{\pi}' - \boldsymbol{\pi})^\top\bar{\boldsymbol{A}}\widehat{\boldsymbol{A}}$$

$$= \left(\boldsymbol{y}^\top \boldsymbol{C}' - \boldsymbol{\pi}'^\top \bar{\boldsymbol{A}}'\right) - \left(\boldsymbol{y}^\top \boldsymbol{C} - \boldsymbol{\pi}^\top \bar{\boldsymbol{A}}\right) \widehat{\boldsymbol{A}}$$

and thus if $\mathsf{V}_{\mathsf{Pupd}}$ accepts then $\mathsf{V}([\boldsymbol{M}]_1, \mathrm{PK}'; [\boldsymbol{y}]_1, [\boldsymbol{\pi}']_1) = 1$ iff $\mathsf{V}([\boldsymbol{M}]_1, \mathrm{PK}; [\boldsymbol{y}]_1, [\boldsymbol{\pi}]_1) = 1$.

(iii: Simulator-update completeness) Clearly,

$$\boldsymbol{y}^\top \widehat{\boldsymbol{C}} - \widehat{\boldsymbol{\pi}}^\top \bar{\boldsymbol{A}}' = \boldsymbol{y}^\top \widehat{\boldsymbol{K}} \bar{\boldsymbol{A}}' - (\widehat{\boldsymbol{K}}^\top \boldsymbol{y})^\top \bar{\boldsymbol{A}}' = \boldsymbol{0}$$

and thus $\mathsf{V}([\boldsymbol{M}]_1, \widehat{\mathrm{PK}}; [\boldsymbol{y}]_1; [\widehat{\boldsymbol{\pi}}]_1) = 1$. The proof that if $\mathsf{V}_{\mathsf{Pupd}}$ accepts then $\mathsf{V}([\boldsymbol{M}]_1, \mathrm{PK}'; [\boldsymbol{y}]_1, [\boldsymbol{\pi}']_1) = 1$ iff $\mathsf{V}([\boldsymbol{M}]_1, \mathrm{PK}; [\boldsymbol{y}]_1, [\boldsymbol{\pi}]_1) = 1$ is the same as in the case (ii). □

Lemma 6 (Key-update hiding and argument-update hiding). *Assume that $\boldsymbol{K}, \widehat{\boldsymbol{K}} \sim \mathcal{D}_{\boldsymbol{K}}$ and $\bar{\boldsymbol{A}}, \widehat{\boldsymbol{A}} \sim \mathcal{D}_{\bar{\boldsymbol{A}}}$, where the distributions $\mathcal{D}_{\boldsymbol{K}}$ and $\mathcal{D}_{\bar{\boldsymbol{A}}}$ satisfy the following conditions: for independent random variables X_1 and X_2,*

- *if $X_i \sim \mathcal{D}_{\boldsymbol{K}}$ for both i then $X_1 + X_2 \sim \mathcal{D}_{\boldsymbol{K}}$. (Thus, $\mathcal{D}^{*2}_{\boldsymbol{K}} = \mathcal{D}_{\boldsymbol{K}}$, where $\mathcal{D}^{*s}$ is the sth convolution power of $\mathcal{D}$. That is, $\mathcal{D}_{\boldsymbol{K}}$ is a stable distribution.)*
- *if $X_i \sim \mathcal{D}_{\bar{\boldsymbol{A}}}$ for both i then $X_1 \cdot X_2 \sim \mathcal{D}_{\bar{\boldsymbol{A}}}$.*

Then, $\Pi^{\mathsf{upd}}_{\mathsf{kw}}$ is

(i) key-update hiding and
(ii) (assuming perfect simulation) argument-update hiding.

One can generalize that requirement when one allows scaling factors. For example, when we define $\boldsymbol{K}' \leftarrow (\boldsymbol{K} + \widehat{\boldsymbol{K}})/2$, one can consider distributions of $\mathcal{D}_{\boldsymbol{K}}$ of matrices where each matrix entry is either uniformly random or a constant.

Remark 1. Recall the following definition [Kle08, Def. 14.46]. Let $I \subset [0, \infty)$ be a semigroup. A family $\nu = (\nu_t : t \in I)$ of probability distributions on $\mathbb{R}^d$ is called a *convolution semigroup* if $\nu_{s+t} = \nu_s * \nu_t$ holds for all $s, t \in I$. Thus, in Lemma 6, we need a convolution semigroup consisting of a single element. However, if we apply scaling factors, we can use more general convolution semigroups.

Proof. (i) Since PK is honestly created, $\boldsymbol{C} = \boldsymbol{K}\bar{\boldsymbol{A}}$ and thus $\boldsymbol{C}' = \boldsymbol{C}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{K}}\bar{\boldsymbol{A}}' = \boldsymbol{K}\bar{\boldsymbol{A}}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{K}}\bar{\boldsymbol{A}}' = (\boldsymbol{K} + \widehat{\boldsymbol{K}})\bar{\boldsymbol{A}}' = \boldsymbol{K}'\bar{\boldsymbol{A}}'$. Similarly, $\boldsymbol{P} = \boldsymbol{M}^\top \boldsymbol{K}$ and $\boldsymbol{P}' = \boldsymbol{P} + \boldsymbol{M}^\top \widehat{\boldsymbol{K}} = \boldsymbol{M}^\top(\boldsymbol{K} + \widehat{\boldsymbol{K}}) = \boldsymbol{M}^\top \boldsymbol{K}'$. Due to the assumption on $\mathcal{D}_{\bar{\boldsymbol{A}}}$ and $\mathcal{D}_{\boldsymbol{K}}$, PK and PK$'$ come from the same distribution.

(ii) We already know PK and PK$'$ come from the same distribution. Assume that $[\boldsymbol{y}]_1 = [\boldsymbol{M}]_1 \mathbf{w}$. Due to the perfect simulation,

$$\boldsymbol{\pi} = \mathsf{Sim}([\boldsymbol{M}]_1, \mathrm{PK}; [\boldsymbol{y}]_1, [\boldsymbol{\pi}]_1) = \boldsymbol{K}^\top \boldsymbol{y} \ .$$

Thus,

$$\begin{aligned}\boldsymbol{\pi}' =& \boldsymbol{K}^\top \boldsymbol{y} + \widehat{\boldsymbol{K}}^\top \boldsymbol{y} = \boldsymbol{K}'^\top \boldsymbol{y} = \mathsf{Sim}([\boldsymbol{M}]_1, \mathrm{PK}'; [\boldsymbol{y}]_1, [\boldsymbol{\pi}']_1) \\ =& \mathsf{P}([\boldsymbol{M}]_1, \mathrm{PK}', [\boldsymbol{y}]_1, \mathbf{w}) \ .\end{aligned}$$

□

Theorem 3 (Strong key-update hiding and strong argument-update hiding). *Assume that $\boldsymbol{K}, \widehat{\boldsymbol{K}} \sim \mathcal{D}_{\boldsymbol{K}}$ and $\bar{\boldsymbol{A}}, \widehat{\boldsymbol{A}} \sim \mathcal{D}_{\bar{\boldsymbol{A}}}$, where the distributions satisfy the following conditions: for independent random variables X_1 and X_2,*

- *if $X_i \sim \mathcal{D}_{\boldsymbol{K}}$ for at least one i then $X_1 + X_2 \sim \mathcal{D}_{\boldsymbol{K}}$. (Thus, the convolution of $\mathcal{D}_{\boldsymbol{K}}$ with any other distribution — over the support of $\mathcal{D}_{\boldsymbol{K}}$ — is $\mathcal{D}_{\boldsymbol{K}}$, or $\mathcal{D}_{\boldsymbol{K}}$ is belongs to an ideal of a convolution semigroup.)*
- *if $X_i \sim \mathcal{D}_{\bar{\boldsymbol{A}}}$ for at least one i then $X_1 \cdot X_2 \sim \mathcal{D}_{\bar{\boldsymbol{A}}}$.*

Then, $\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ is

(i) strong key-update hiding and
(ii) (assuming perfect simulation) strong argument-update hiding.

Proof. We will prove (i) and (ii) together in two different cases: (1) when PK (and the argument) was honestly created, and (2) when PK$'$ (and the argument) was honestly updated.

(1: PK / π were honestly created and the updates verify) Since PK is honestly created, $\boldsymbol{C} = \boldsymbol{K}\bar{\boldsymbol{A}}$ and $\boldsymbol{P} = \boldsymbol{M}^\top \boldsymbol{K}$. Since $\mathsf{V}_{\mathsf{Kupd}}$ accepts, we have $\bar{\boldsymbol{A}}' = \bar{\boldsymbol{A}}\widehat{\boldsymbol{A}}$ (thus, by the assumption on $\mathcal{D}_{\bar{\boldsymbol{A}}}$, $\bar{\boldsymbol{A}}'$ comes from the correct distribution), $\boldsymbol{C}' = \boldsymbol{C}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{C}}$, and $\boldsymbol{M}^\top \widehat{\boldsymbol{C}} = \widehat{\boldsymbol{P}}\bar{\boldsymbol{A}}'$ where $\widehat{\boldsymbol{P}} = \boldsymbol{P}' - \boldsymbol{P}$. Thus,

$$\boldsymbol{C}' = \boldsymbol{C}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{C}} = \boldsymbol{K}\bar{\boldsymbol{A}}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{C}} = \boldsymbol{K}\bar{\boldsymbol{A}}' + \widehat{\boldsymbol{C}} \ .$$

Define implicitly $\boldsymbol{K}' := \boldsymbol{C}'\bar{\boldsymbol{A}}'^{-1} = (\boldsymbol{K}\bar{\boldsymbol{A}}' + \widehat{\boldsymbol{C}})\bar{\boldsymbol{A}}'^{-1} = \boldsymbol{K} + \widehat{\boldsymbol{C}}\bar{\boldsymbol{A}}'^{-1}$ (note that $\bar{\boldsymbol{A}}'$ is invertible). Thus, obviously, $\boldsymbol{C}' = \boldsymbol{K}'\bar{\boldsymbol{A}}'$. On the other hand,

$$\boldsymbol{M}^\top \boldsymbol{K}' = \boldsymbol{M}^\top(\boldsymbol{K} + \widehat{\boldsymbol{C}}\bar{\boldsymbol{A}}'^{-1}) = \boldsymbol{P} + \boldsymbol{M}^\top \widehat{\boldsymbol{C}}\bar{\boldsymbol{A}}'^{-1} = \boldsymbol{P} + \widehat{\boldsymbol{P}}\bar{\boldsymbol{A}}'\bar{\boldsymbol{A}}'^{-1} = \boldsymbol{P} + \widehat{\boldsymbol{P}} = \boldsymbol{P}'$$

and thus $\boldsymbol{P}' = \boldsymbol{M}^\top \boldsymbol{K}'$. To show that PK and PK$'$ come from the same distribution, we now only need to show that $\boldsymbol{K}' = \boldsymbol{K} + \widehat{\boldsymbol{C}}\bar{\boldsymbol{A}}'^{-1}$ comes from the same distribution as $\boldsymbol{K}$. This holds under the assumption that $\mathcal{D}_{\boldsymbol{K}}$ is an ideal of a convolution semigroup.

Now, consider the argument $[\boldsymbol{\pi}']_1$. We have, in addition to equations above, that

- since $[\boldsymbol{\pi}]_1$ is honestly created: $\boldsymbol{\pi} = \boldsymbol{P}^\top \mathbf{w}$.
- since $[\boldsymbol{\pi}']_1$ verifies: $\boldsymbol{y}^\top \widehat{\boldsymbol{C}} = \widehat{\boldsymbol{\pi}}\bar{\boldsymbol{A}}$.

Due to completeness, $\boldsymbol{y}^\top \boldsymbol{C} = \boldsymbol{\pi}^\top \bar{\boldsymbol{A}}$. Thus,

$$\begin{aligned}\boldsymbol{y}^\top \boldsymbol{C}' - \boldsymbol{\pi}'\bar{\boldsymbol{A}}' =& \boldsymbol{y}^\top(\boldsymbol{C}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{C}}) - (\boldsymbol{\pi} + \widehat{\boldsymbol{\pi}})\bar{\boldsymbol{A}}' \\ =& \underbrace{(\boldsymbol{y}^\top \boldsymbol{C} - \boldsymbol{\pi}\bar{\boldsymbol{A}})}_{=\boldsymbol{0}}\widehat{\boldsymbol{A}} + \underbrace{(\boldsymbol{y}^\top \widehat{\boldsymbol{C}} - \widehat{\boldsymbol{\pi}}\bar{\boldsymbol{A}}')}_{=\boldsymbol{0}}\end{aligned}$$

and thus $\boldsymbol{C}'$ verifies. But then clearly, $\boldsymbol{\pi}' = \boldsymbol{y}^\top \boldsymbol{C}'\bar{\boldsymbol{A}}'^{-1}$ is the unique correct proof for $\boldsymbol{y} \in \mathrm{ColSpace}(\boldsymbol{M})$ when using the public key PK$'$.

(2: PK verifies and the update was honestly done) We have the following equations:

- since PK verifies: $\boldsymbol{M}^\top \boldsymbol{C} = \boldsymbol{P}\bar{\boldsymbol{A}}$,
- since PK$'$ was honestly updated: for correctly distributed $\widehat{\boldsymbol{A}}$ and $\widehat{\boldsymbol{K}}$, $\bar{\boldsymbol{A}}' = \bar{\boldsymbol{A}}\widehat{\boldsymbol{A}}$, $\widehat{\boldsymbol{C}} = \widehat{\boldsymbol{K}}\bar{\boldsymbol{A}}'$, $\boldsymbol{C}' = \boldsymbol{C}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{C}}$, $\widehat{\boldsymbol{P}} = \boldsymbol{M}^\top\widehat{\boldsymbol{K}}$, $\boldsymbol{P}' = \boldsymbol{P} + \widehat{\boldsymbol{P}}$.

Due to the assumption on $\mathcal{D}_{\bar{\boldsymbol{A}}}$, $\bar{\boldsymbol{A}}'$ comes from the correct distribution. Define implicitly $\boldsymbol{P} := \boldsymbol{M}^\top\boldsymbol{C}\bar{\boldsymbol{A}}^{-1}$ (note that $\bar{\boldsymbol{A}}$ is invertible) and $\boldsymbol{K} := \boldsymbol{C}\bar{\boldsymbol{A}}^{-1}$. Then,

$$\boldsymbol{K}' = \boldsymbol{K} + \widehat{\boldsymbol{K}} = \boldsymbol{K} + \widehat{\boldsymbol{C}}\bar{\boldsymbol{A}}'^{-1} = (\boldsymbol{C}\widehat{\boldsymbol{A}} + \widehat{\boldsymbol{C}})\bar{\boldsymbol{A}}'^{-1} = \boldsymbol{C}'\bar{\boldsymbol{A}}'^{-1} \quad .$$

Next, $\boldsymbol{K}' = \boldsymbol{K} + \widehat{\boldsymbol{K}}$ has the same distribution as $\widehat{\boldsymbol{K}}$ by the assumption on $\mathcal{D}_{\boldsymbol{K}}$. Because both $\boldsymbol{K}'$ and $\bar{\boldsymbol{A}}'$ have correct distributions, also $\boldsymbol{C}' = \boldsymbol{K}'\bar{\boldsymbol{A}}'$ has correct distribution.

Next, obviously $\boldsymbol{P}' = \boldsymbol{M}^\top(\boldsymbol{C}\bar{\boldsymbol{A}}^{-1} + \widehat{\boldsymbol{K}}) = \boldsymbol{M}^\top(\boldsymbol{K} + \boldsymbol{K}' - \boldsymbol{K}) = \boldsymbol{M}^\top\boldsymbol{K}'$ has correct distribution. This proves strong key-updating.

In the case of strong argument-updating, additionally, the following holds:

- the original argument verifies: $\boldsymbol{y}^\top\boldsymbol{C} = \boldsymbol{\pi}^\top\bar{\boldsymbol{A}}$,
- $[\boldsymbol{\pi}]_1$ was updated honestly: $\boldsymbol{\pi}' = \boldsymbol{\pi} + \widehat{\boldsymbol{K}}^\top\boldsymbol{y}$ for honestly distributed $\widehat{\boldsymbol{K}} \sim \mathcal{D}_{\boldsymbol{K}}$.

From this we get that $\boldsymbol{\pi} = (\boldsymbol{y}^\top\boldsymbol{C}\bar{\boldsymbol{A}}^{-1})^\top = (\boldsymbol{y}^\top\boldsymbol{K})^\top = \boldsymbol{K}^\top\boldsymbol{y}$. Thus,

$$\boldsymbol{\pi}' = \boldsymbol{\pi} + \widehat{\boldsymbol{K}}^\top\boldsymbol{y} = \boldsymbol{K}^\top\boldsymbol{y} + \widehat{\boldsymbol{K}}^\top\boldsymbol{y} = (\boldsymbol{K} + \widehat{\boldsymbol{K}})^\top\boldsymbol{y} = \boldsymbol{K}'^\top\boldsymbol{y}$$

which is equal to the simulated argument of $\boldsymbol{y} = \boldsymbol{M}\mathbf{w}$ (when using the public key PK$'$) and by perfect simulation, thus also to the real argument (when using PK$'$). □

Example 1 (Of the required distributions). $\mathcal{D}_{\boldsymbol{K}}$ is the uniform distribution over $k_2 \times k_2$ matrices over $\mathbb{Z}_p$ and $\mathcal{D}_{\bar{\boldsymbol{A}}}$ is the uniform distribution over $k_2 \times k_2$ invertible matrices over $\mathbb{Z}_p$ where as the sanity check, one checks that both matrices $[\bar{\boldsymbol{A}}]_1$ and $[\widehat{\boldsymbol{A}}]_1$ are invertible.[1] As mentioned before, in the actual instantiation of Π'_{as} (at least in the most efficient case $k_2 = 1$) both $\mathcal{D}_{\boldsymbol{K}}$ and $\mathcal{D}_{\bar{\boldsymbol{A}}}$ are equal to the uniform distribution over $\mathbb{Z}_p$ and thus satisfy the required properties. □

Theorem 4. *Let $k_2 = 1$ (resp., $k_2 = 2$). If the $\mathcal{D}_{k_2}$-*KerMDH *(resp., $\mathcal{D}_{k_2}$-*SKerMDH*) assumption holds relative to* Pgen *then $\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ is*

(i) computationally quasi-adaptively argument-update sound (I), and
(ii) assuming that the preconditions of Theorem 3 are fulfilled, also computationally quasi-adaptively argument-update sound (II)

in the BPK model.

We emphasize that argument-update soundness (II) follows only when the update has been done by a honest prover (who knows the witness and does not know the key-update secret key $\widehat{\text{SK}}$).

[1] Intuitively, we require the family ν of probability distributions to be an ideal of a convolution semigroup: $\nu_s * \mu = \nu_t$, for some t, for any element μ of the semigroup.

Proof. **(i: argument-update soundness (I))** follows from Theorem 2 (Π_{bpk} is computationally quasi-adaptively sound under the KerMDH / $\mathsf{SKerMDH}$ assumption) and Lemma 2 (any sound argument system is also argument-update sound (I)).

(i: argument-update soundness (II)) follows from Theorem 2 (Π_{bpk} is computationally quasi-adaptively sound under the KerMDH / $\mathsf{SKerMDH}$ assumption), Lemma 2 (any sound, argument-update complete, strongly-key-update hiding, and strongly argument-update hiding argument system is also argument-update sound (II)), Lemma 5 ($\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ is argument-update complete), and Theorem 3 ($\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ is strongly key-update hiding and strongly argument-update hiding). □

The proof of the following theorem closely follows the proof of nonuniform zero knowledge of Π_{bpk} in [ALSZ18], except that it deals additionally with the key-updating part; the latter part of the proof relies on the key-update completeness (Lemma 5).

Interestingly, we rely on $\mathsf{SKW\text{-}KE}$ and $\mathsf{KW\text{-}KE}$ that are tautological knowledge assumptions for Π_{bpk}, but not for $\Pi_{\mathsf{kw}}^{\mathsf{upd}}$. This gives more credence to these assumptions as something of independent interest.

Lemma 7. *Let $\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ be the updatable QA-NIZK argument system for linear subspaces from Fig. 5. Let $k_2 \in \{1, 2\}$. The following statements hold in the BPK model.*

(i) If the $(\mathcal{D}_{\mathsf{p}}, k_2)$-$\mathsf{SKW\text{-}KE}_{\mathbb{G}_1}$ assumption holds relative to Pgen (and thus also assuming $\mathcal{D}_{\mathsf{p}}$-$\mathsf{wKerMDH}$, i.e., that $[\boldsymbol{M}]_1$ comes from a $\mathsf{wKerMDH}$-hard distribution) then Π_{bpk} is statistically nonuniform key-update zero knowledge.

(ii) If the $(\mathcal{D}_{\mathsf{p}}, k_2)$-$\mathsf{KW\text{-}KE}_{\mathbb{G}_1}$ assumption holds relative to Pgen then Π_{bpk} is statistically nonuniform key-update zero knowledge.

Proof. The proof follows from Lemma 5 ($\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ is key-update complete), Theorem 2 (Π_{bpk} is nonuniform zero-knowledge), and Lemma 3 (key-update zero knowledge follows from key-update completeness and nonuniform zero-knowledge). □

Lemma 8. *Let $\Pi_{\mathsf{kw}}^{\mathsf{upd}}$ be the updatable QA-NIZK argument system for linear subspaces from Fig. 5. Let $k_2 \in \{1, 2\}$. The following statements hold in the BPK model.*

(i) If the $(\mathcal{D}_{\mathsf{p}}, k_2)$-$\mathsf{SKW\text{-}KE}_{\mathbb{G}_1}$ assumption holds relative to Pgen (and thus also assuming $\mathcal{D}_{\mathsf{p}}$-$\mathsf{wKerMDH}$, i.e., that $[\boldsymbol{M}]_1$ comes from a $\mathsf{wKerMDH}$-hard distribution) then Π_{bpk} is statistically nonuniform argument-update zero knowledge.

(ii) If the $(\mathcal{D}_{\mathsf{p}}, k_2)$-$\mathsf{KW\text{-}KE}_{\mathbb{G}_1}$ assumption holds relative to Pgen then Π_{bpk} is statistically nonuniform argument-update zero knowledge.

Proof. By Lemma 4, argument-update zero knowledge follows from key-update completeness, simulator-update completeness (both proven in Lemma 5) and nonuniform zero knowledge (proven in Theorem 2). The result now follows from the precise security statements in Theorem 2. □

6 Discussion

Why updating $\boldsymbol{M}$ might be difficult. In certain applications, one might also be interested in updating $\boldsymbol{M}$ (e.g., if $[\boldsymbol{M}]_1$ is a public key of some trusted third party). Assume $[\boldsymbol{M}]_1$ is updated, then $[\boldsymbol{P}']_1 - [\boldsymbol{P}]_1 = [\boldsymbol{M}'^\top \boldsymbol{K}']_1 - [\boldsymbol{M}^\top \boldsymbol{K}_{i-1}]_1 = [\boldsymbol{M}'^\top(\boldsymbol{K}_{i-1} + \widehat{\boldsymbol{K}})]_1 - [\boldsymbol{M}_{i-1}^\top \boldsymbol{K}_{i-1}]_1 = [\widehat{\boldsymbol{M}}^\top \boldsymbol{K}_{i-1}]_1 + [\boldsymbol{M}'^\top]_1 \widehat{\boldsymbol{K}}$ which can be computed assuming the party knows either $\widehat{\boldsymbol{M}'}^\top$ or $\boldsymbol{K}_{i-1}$, or if all previous parties help him to compute $[\widehat{\boldsymbol{M}'}^\top \boldsymbol{K}_{i-1}]_1$. Since neither possibility seems realistic, one cannot probably update the language parameter. In the case of some applications it might be allowed for the updater to know it: for example in the case of the use of QA-NIZKs in UC commitments [FLM11], $\boldsymbol{M}$ would correspond to the public key of an IND-CPA-secure cryptosystem and the updater could update it additively.

Updatable Universal Public Key. The goal of updatability is to protect soundness in the case PK may be subverted, since nonuniform zero knowledge (i.e., Sub-ZK) can be obtained by running the public algorithm V_{pk} [ABLZ17]. In Π'_{as}, soundness is guaranteed by one of the elements of PK (namely, $[\bar{\boldsymbol{A}}]_2$, see Fig. 2) coming from a KerMDH-hard distribution[2], and another element $[\boldsymbol{C}]_2$ being correctly computed from $[\bar{\boldsymbol{A}}]_2$. Since the latter can be verified by V_{pk}, to obtain soundness it suffices to update $[\bar{\boldsymbol{A}}]_2$. (The procedure of this is the same as creating $[\bar{\boldsymbol{A}}']_2$ by $\mathsf{K}_{\mathsf{upd}}$ in Fig. 5, and verifying this update consists of the first four verification equations in $\mathsf{V}_{\mathsf{Kupd}}$.) Then, $[\bar{\boldsymbol{A}}]_2$ will be a "universal public key" [GKM+18] for all possible language parameters in all applications that trust the concrete KerMDH *assumption.*

Importantly, one is here not restricted to Π'_{as} or even QA-NIZK: *any* application that relies on KerMDH-hardness of $\mathcal{D}_{\bar{\boldsymbol{A}}}$, where it suffices to know $[\bar{\boldsymbol{A}}]_2$ (instead of $[\boldsymbol{A}]_2$), and where $\mathcal{D}_{\bar{\boldsymbol{A}}}$ satisfies the conditions of Theorem 3, can use the same matrix $[\bar{\boldsymbol{A}}]_2$. A standard example is the 1-Lin [BBS04,EHK+13] distribution $\mathcal{L}_1 = \{\boldsymbol{A} = \left(\begin{smallmatrix} a \\ 1 \end{smallmatrix}\right) : a \leftarrow_\$ \mathbb{Z}_p\}$. After potentially many updates of $[\bar{\boldsymbol{A}}]_2$, one can create the whole public key PK corresponding to a concrete language parameter. Thereafter, one can continue updating $[\bar{\boldsymbol{A}}]_2$ together with all known public keys and arguments that use (the same version of) $[\bar{\boldsymbol{A}}]_2$ for soundness. Such one-phase updating can be formalized like in [GKM+18], adding to it QA-NIZK and argument-update specifics, and is out of the scope of the current paper.

We emphasize that the possibility to rely just on $[\bar{\boldsymbol{A}}]_2$ is a major difference with updatable SNARKs [GKM+18] where the universal key is quite complex and each universal key of length $\Theta(n^2)$ (see [MBKM19] for recent optimizations) can only be used for circuits of size $\leq n$.

[2] Technically, $\boldsymbol{A}$ comes from a KerMDH-hard distribution, not $\bar{\boldsymbol{A}}$. However, in the case of some distributions (like DLIN-related distributions), $\boldsymbol{A}$ has an extra constant column compared to $\bar{\boldsymbol{A}}$. The knowledge of $[\boldsymbol{A}]_2$ and $[\bar{\boldsymbol{A}}]_2$ is equivalent in such a case.

Witness-sampleability. The QA-NIZK Π'_{as} from [KW15] requires the distribution of $[\boldsymbol{M}]_1$ to be witness-sampleable. We leave it as an open problem to construct an updatable QA-NIZK that does not have this requirement. We note that [ALSZ18] only proposed nonuniform version of Π'_{as}.

Acknowledgment. We would like to thank Dario Fiore and Markulf Kohlweiss for useful comments. Helger Lipmaa was supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 780477 (project PRIViLEDGE), and by the Estonian Research Council grant (PRG49).

References

ABLZ17. Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, and Michal Zajac. A subversion-resistant SNARK. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 3–33. Springer, Heidelberg, December 2017.

ABP15. Michel Abdalla, Fabrice Benhamouda, and David Pointcheval. Disjunctions for hash proof systems: New constructions and applications. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 69–100. Springer, Heidelberg, April 2015.

ALSZ18. Behzad Abdolmaleki, Helger Lipmaa, Janno Siim, and Michał Zając. QANIZK in the BPK Model. Technical Report 2018/877, IACR, September 18, 2018. Available from `https://eprint.iacr.org/2018/877`, updated version from 19 Feb 2019.

APV05. Joël Alwen, Giuseppe Persiano, and Ivan Visconti. Impossibility and feasibility results for zero knowledge with public keys. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 135–151. Springer, Heidelberg, August 2005.

BBS04. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Heidelberg, August 2004.

BCG$^+$14. Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014.

BCG$^+$15. Eli Ben-Sasson, Alessandro Chiesa, Matthew Green, Eran Tromer, and Madars Virza. Secure sampling of public parameters for succinct zero knowledge proofs. In *2015 IEEE Symposium on Security and Privacy*, pages 287–304. IEEE Computer Society Press, May 2015.

BFM88. Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pages 103–112. ACM Press, May 1988.

BFS16. Mihir Bellare, Georg Fuchsbauer, and Alessandra Scafuro. NIZKs with an untrusted CRS: Security in the face of parameter subversion. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 777–804. Springer, Heidelberg, December 2016.

BGG17. Sean Bowe, Ariel Gabizon, and Matthew D. Green. A multi-party protocol for constructing the public parameters of the pinocchio zk-SNARK. Cryptology ePrint Archive, Report 2017/602, 2017. `http://eprint.iacr.org/2017/602`.

BGM17. Sean Bowe, Ariel Gabizon, and Ian Miers. Scalable multi-party computation for zk-SNARK parameters in the random beacon model. Cryptology ePrint Archive, Report 2017/1050, 2017. `http://eprint.iacr.org/2017/1050`.

CGGM00. Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In *32nd ACM STOC*, pages 235–244. ACM Press, May 2000.

DFKP13. George Danezis, Cédric Fournet, Markulf Kohlweiss, and Bryan Parno. Pinocchio coin: building zerocoin from a succinct pairing-based proof system. pages 27–30, Berlin, Germany, November 4, 2013. ACM.

DGP^{+}19. Vanesa Daza, Alonso González, Zaira Pindado, Carla Ràfols, and Javier Silva. Shorter Quadratic QA-NIZK Proofs. In Dongdai Lin and Kazue Sako, editors, *PKC 2019*, volume ? of *LNCS*, pages ?–?, Beijing, China, April 14–17, 2019. Springer, Cham.

EHK^{+}13. Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, August 2013.

FLM11. Marc Fischlin, Benoît Libert, and Mark Manulis. Non-interactive and re-usable universally composable string commitments with adaptive security. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 468–485. Springer, Heidelberg, December 2011.

Fuc18. Georg Fuchsbauer. Subversion-zero-knowledge SNARKs. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 315–347. Springer, Heidelberg, March 2018.

GGPR13. Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 626–645. Springer, Heidelberg, May 2013.

GHR15. Alonso González, Alejandro Hevia, and Carla Ràfols. QA-NIZK arguments in asymmetric groups: New tools and new constructions. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 605–629. Springer, Heidelberg, November / December 2015.

GKM^{+}18. Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-SNARKs. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 698–728. Springer, Heidelberg, August 2018.

GM17. Jens Groth and Mary Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 581–612. Springer, Heidelberg, August 2017.

GPS08. Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for Cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

GR16. Alonso González and Carla Ràfols. New techniques for non-interactive shuffle and range arguments. In Mark Manulis, Ahmad-Reza Sadeghi, and

Steve Schneider, editors, *ACNS 16*, volume 9696 of *LNCS*, pages 427–444. Springer, Heidelberg, June 2016.

Gro10. Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 321–340. Springer, Heidelberg, December 2010.

Gro16. Jens Groth. On the size of pairing-based non-interactive arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Heidelberg, May 2016.

GW11. Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 99–108. ACM Press, June 2011.

JR13. Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2013.

JR14. Charanjit S. Jutla and Arnab Roy. Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 295–312. Springer, Heidelberg, August 2014.

Kle08. Achim Klenke. *Probability Theory: A Comprehensive Course*. Universitext. Springer, 1 edition, 2008.

KW15. Eike Kiltz and Hoeteck Wee. Quasi-adaptive NIZK for linear subspaces revisited. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 101–128. Springer, Heidelberg, April 2015.

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 169–189. Springer, Heidelberg, March 2012.

LPJY14. Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 514–532. Springer, Heidelberg, May 2014.

LPJY15. Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Compactly hiding linear spans - tightly secure constant-size simulation-sound QA-NIZK proofs and applications. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 681–707. Springer, Heidelberg, November / December 2015.

MBKM19. Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-Knowledge SNARKs from Linear-Size Universal and Updateable Structured Reference Strings. Technical Report 2019/099, IACR, January 30, 2019. Available from `https://eprint.iacr.org/2019/099`, last check version from Feb 7, 2019.

MR01. Silvio Micali and Leonid Reyzin. Soundness in the public-key model. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 542–565. Springer, Heidelberg, August 2001.

MRV16. Paz Morillo, Carla Ràfols, and Jorge Luis Villar. The kernel matrix Diffie-Hellman assumption. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 729–758. Springer, Heidelberg, December 2016.

Nao03. Moni Naor. On cryptographic assumptions and challenges (invited talk). In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 96–109. Springer, Heidelberg, August 2003.

PHGR13. Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013.

Wee07. Hoeteck Wee. Lower bounds for non-interactive zero-knowledge. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 103–117. Springer, Heidelberg, February 2007.