Updatable NIZKs from Non-Interactive Zaps

Karim Baghery[1] and Navid Ghaedi Bardeh[2]

[1] imec-COSIC, KU Leuven, Leuven, Belgium
[2] Norwegian University of Science and Technology, Trondheim, Norway.
karim.baghery@kuleuven.be, navid.ghaedibardeh@gmail.com

Abstract. In ASIACRYPT 2016, Bellare, Fuchsbauer, and Scafuro studied the security of NIZK arguments under subverted Structured Reference String (SRS) and presented some positive and negative results. In their best positive result, they showed that by defining an SRS as a tuple of knowledge assumption in bilinear groups (e.g. g^a, g^b, g^{ab}), and then using a Non-Interactive (NI) zap to prove that either there is a witness for the statement x or one knows the trapdoor of SRS (e.g. a or b), one can build NIZK arguments that can achieve soundness and *subversion zero-knowledge* (zero-knowledge without trusting a third party; Sub-ZK). In this paper, we expand their idea and use NI zaps (of knowledge) to build NIZK arguments (of knowledge) with *updatable*, *universal*, and *succinct* SRS. To this end, we first show that their proposed sound and Sub-ZK NIZK argument can also achieve *updatable* soundness, which is a more desired notion than the plain soundness. Updatable soundness allows the verifier to update the SRS one time and bypass the need for a trusted third party. Then, we show that using a similar OR language, given a NI zap (of knowledge) and a *key-updatable* signature scheme, one can build NIZK arguments that can achieve Sub-ZK and *updatable* simulation soundness (resp. *updatable* simulation extractability). The proposed constructions are the first NIZK arguments that have updatable and succinct SRS, and do not require a random oracle. Our instantiations show that in the resulting NIZK arguments the computational cost for the parties to verify/update the SRS is negligible, namely, a few exponentiations and pairing checks. The run times of the prover and verifier, as well as the size of the proof, are asymptotically the same as those of the underlying NI zap.

Keywords: Non-interactive Zaps, Non-interactive Zap of Knowledge, NIZK, Subversion ZK, Updatable SRS Model, Simulation Extractability

1 Introduction

Let $\mathbf{R_L}$ be an NP relation which defines the language $\mathbf{L}$ of all statements x for which there exists a witness w s.t. $(\mathsf{x}, \mathsf{w}) \in \mathbf{R_L}$. A Non-Interactive Zero-Knowledge (NIZK) argument [24, 10] for $\mathbf{R_L}$ allows a party P (called prover), knowing w, to non-interactively prove the truth of a statement x without leaking extra information about the witness w. As the basic requirements, a NIZK argument is expected to satisfy, (i) <u>*Completeness*</u>, guaranteeing that an honest

P will convince an honest V with probability 1. (ii) *Soundness* (SND), ensuring that no malicious P can convince (honest) V of a false statement, except with negligible probability. (iii) *Zero-Knowledge* (ZK), guaranteeing that the verifier V learns nothing beyond the truth of statement x from the proof. In [18], Feige and Shamir proposed (iv) *Witness Indistinguishability* (WI), as a relaxation of ZK, which only guarantees that V cannot distinguish which witness was used by the prover to generate the proof.

To achieve SND and ZK at the same time, NIZK arguments [10] rely on the existence of a common reference string. The later and more efficient constructions need a Structured Reference String (SRS) which is supposed to be generated by a Trusted Third Party (TTP) and shared with P and V. Finding a TTP to generate the SRS can be a serious concern in practical applications where parties mutually distrust each other. In [17], Dwork and Naor presented a two-round proof system for NP, so called zap, that *does not require* a TTP while achieves WI and SND. A zap is a two-round protocol which starts by sending a message from V to the P, and finishes by sending the proof from P to V. Later in [29], Groth, Ostrovsky and Sahai presented a Non-Interactive (NI) zap under standard assumptions, where P sends the proof in one round to V. In comparison with NIZK arguments, NI zaps come with a weaker security guarantee, namely WI and SND, while they do not need a trusted setup phase. Recently, in [20], Fuchsbauer and Orru showed that under knowledge of exponent assumption [14, 7], the NI zap of Groth, Ostrovsky and Sahai, can be extended to achieve WI and (v) *Knowledge Soundness* (KS), which guarantees that no malicious P can convince (honest) V, unless he *knows* a witness w for the statement x, s.t. $(\mathsf{x}, \mathsf{w}) \in \mathbf{R_L}$. Their construction is the first NI Zap of Knowledge (NI-ZoK). Zaps and NI zaps for NP are shown to be extremely useful in the design of various cryptographic primitives.

Subversion-Resistant and Updatable NIZK Arguments. In 2016, Bellare, Fuchsbauer and Scafuro [7] studied achievable security in NIZK arguments in the face of a subverted SRS. They first presented a stronger variation of standard notions, so called Subversion-SND (Sub-SND), Subversion-WI (Sub-WI), Subversion-ZK (Sub-ZK), that respectively imply SND, WI, ZK even if the SRS is generated by an adversary $\mathcal{A}$. For instance, (vi) Sub-ZK implies that the NIZK argument guarantees ZK even if $\mathcal{A}$ sets up the SRS. Given the new definitions, they presented some negative and positive results for building subversion-resistant NIZK arguments. As the best positive result, they showed that using an OR-based language one can use a NI zap (e.g. [29]) along with an SRS (e.g. a tuple of a knowledge assumption in bilinear groups [14, 7], which can be verified publicly) and build a NIZK argument that satisfies SND and Sub-ZK. The resulting NIZK argument will have a *universal* and *succinct* SRS. A universal and succinct SRS allows one to prove knowledge of a witness for different languages, and it additionally enables more efficient SRS verification to achieve Sub-ZK. Two follow-up works [1, 19] studied achieving Sub-ZK in zero-knowledge Succinct Non-interactive ARguments of Knowledge (zk-SNARKs) [31, 9, 26]. In Sub-ZK NIZK arguments (of knowledge), the prover does not need to trust a third party, while the verifier still has to trust the SRS generator. In 2018, Groth et al. [27] introduced *updatable* SRS model that allows both P and V to update a *universal* SRS and bypass the need for a TTP. Unlike a language-dependent SRS, a universal SRS can be used for various languages. They defined new notions (vii) *Updatable KS*, (U-KS), (viii) *Updatable SND*, (U-SND), and (ix) *Updatable ZK*, (U-ZK), which each implies the standard notion, as long as the initial SRS generation or one of (the follow up) SRS updates is done by an honest party. Then, Groth et al. [27] presented the first zk-SNARK that can achieve Sub-ZK and U-KS. In such zk-SNARKs, the prover verifies the final SRS, and the verifier one-time updates the SRS and they avoid trusting a third party. The follow-up works in this direction [13, 12, 33] are more efficient. In 2019, Baghery [4] showed that Sub-ZK and KS SNARKs can be lifted to achieve Sub-ZK and (x) *Simulation Extractability* (SE) (a.k.a. *Simulation Knowledge Soundness*), which implies an $\mathcal{A}$ cannot convince (honest) V, even if he has seen polynomially many simulated proofs, unless he *knows* a witness w for the statement x. Recent studies achieve Sub-ZK and SE in quasi-adaptive NIZK arguments [3] and ad-hoc construction of zk-SNARKs [5]. A follow-up work [2] showed that Sub-ZK and U-KS zk-SNARKs can be lifted to achieve Sub-ZK and (xi) *Updatable SE* (U-SE) which ensures that the protocol satisfy SE as long as the initial SRS generation or one of the SRS updates is done honestly. A recent study [22], shows that some of the zk-SNARKs with updatable SRS, can also achieve U-SE [30, 21, 13]. TIRAMISU allows one to lift such constructions to achieve *black-box* extractability [6], in the updatable SRS model.

Our Contribution. The core of our contribution is to show that the technique used by Bellare et al. [7] for building a Sub-ZK and NIZK argument, can also be expanded to build NIZK arguments with updatable SRS. Namely, one can use NI zap (resp. NI-ZoK) arguments along with an OR language and build Sub-ZK NIZK arguments (resp. of knowledge) with an updatable SRS, that can satisfy U-X, where $X \in \{\text{SND}, \text{KS}, \text{Simulation Soundness}, \text{SE}\}$.

To this end, we first propose an *SRS updating* and an *SRS verification* algorithms for the Sub-ZK and SND NIZK argument of Bellare et al. [7], and then show that under the same assumptions used in the security proof of their scheme, namely, the Diffie-Hellman Knowledge-of-Exponent (DH-KE) [7], the Computational Diffie-Hellman (CDH), and the Decision Linear (DLin) [11] assumptions, their scheme can also achieve U-SND. This results in the first NIZK argument with *updatable*, *universal*, and *succinct* SRS that does not exploit a random oracle. By instantiating Bellare et al.'s [7] construction with Fuchsbauer and Orru's NI-ZoK [20], our results lead to a universal and updatable NIZK argument of knowledge that can satisfy Sub-ZK and U-KS.

After that, we generalize Bellare et al.'s idea and show that given a NI zap argument (resp. NI-ZoK) and a *key-updatable* signature scheme [2], one can define an OR language, namely,

$$((\mathsf{x}, \mathsf{w}) \in \mathbf{R_L}) \ \cup \ (\text{I know the } \mathsf{sk} \text{ associated with the signature's updatable } \mathsf{pk}),$$

where $(\mathsf{sk}, \mathsf{pk})$ are a pair of secret and public keys, and build NIZK arguments that can achieve Sub-ZK and *Updatable* Simulation Soundness, U-SS, (resp. U-

SE). Our resulting Sub-ZK and U-SS NIZK arguments are the first constructions that can achieve these notions. But, Abdolmaleki et al. [2] also built Sub-ZK and U-SE zk-SNARKs using a similar OR language. However, as stated in [[2], theorems 2, and 3], they require a NIZK argument that satisfies Sub-ZK and (U-)KS, which are stronger security requirements than what a NI zap (or NI-ZoK) achieves. In nutshell, we require weaker security guarantees from the input proof system, but, on the other side, we obtain Sub-ZK and U-SE NIZK arguments with succinct SRS and linear proofs.

Key Insights. Our key insights to generalize the OR-based technique of Bellare et al. that uses NI zaps have been the following: First, since NI zap (or NI-ZoK) arguments do not have an SRS, therefore once we define an OR-based language based on them, the SRS of the resulting NIZK argument only contains the elements that are added from the second clause in the OR language. Consequently, to simulate the proofs in the resulting NIZK argument, one uses the trapdoors of the SRS that come from the second clause of the OR relation. The second key point is that by using an *updatable* SRS in the second clause, one can construct NIZK arguments with *universal* and *updatable* SRS. This allows us to build the first NIZK arguments with succinct (constant group elements), updatable and universal SRS. The third insight is that if the updatable SRS in the second clause of the OR language be the public key of a (key-updatable) signature scheme [2], then we can construct Sub-ZK NIZK arguments that can achieve U-SS or U-SE.

It is worth mentioning that we mainly used the NI zap of Groth, Ostrovsky and Sahai [29], and the NI-ZoK of Fuchsbauer and Orru [20]. However, our constructions and techniques can be instantiated and adapted with other NI zaps, e.g. [32]. As we discuss later, in all the resulting NIZK arguments, to bypass the need for a TTP, the computational cost for the prover/verifier is a few numbers of exponentiation and pairing operations.

The rest of the paper is organized as follows; Sec. 2 presents necessary preliminaries for the paper. In Sec. 3, we show that the Sub-ZK and SND NIZK argument of Bellare, Fuchsbauer and Scafuro [7], can also achieve U-SND. In Sec. 4, we show how one can build NIZK arguments with updatable, universal, and succinct SRS using NI zaps and key-updatable signature schemes. Finally, we conclude the paper in Sec. 5.

2 Preliminaries

Let λ be the security parameter and $\mathsf{negl}(\lambda)$ denotes a negligible function. We use $x \leftarrow_\$ X$ to denote x sampled uniformly according to the distribution X. PPT stands for probabilistic polynomial-time. All adversaries are assumed to be state-full. For an algorithm $\mathcal{A}$, let $\mathrm{im}(\mathcal{A})$ be the image of $\mathcal{A}$, i.e., the set of valid outputs of $\mathcal{A}$. Moreover, assume $\mathsf{RND}(\mathcal{A})$ denotes the random tape of $\mathcal{A}$, and $r \leftarrow_\$ \mathsf{RND}(\mathcal{A})$ denotes sampling of a randomizer r of sufficient length for $\mathcal{A}$'s needs. By $y \leftarrow \mathcal{A}(x; r)$ we mean given an input x and a randomizer r, $\mathcal{A}$ outputs y. For algorithms $\mathcal{A}$ and $\mathsf{Ext}_\mathcal{A}$, we write $(y \parallel y') \leftarrow (\mathcal{A} \parallel \mathsf{Ext}_\mathcal{A})(x; r)$ as a shorthand for "$y \leftarrow \mathcal{A}(x; r)$, $y' \leftarrow \mathsf{Ext}_\mathcal{A}(x; r)$".

In pairing-based groups, we use additive notation together with the bracket notation, i.e., in group $\mathbb{G}_\mu$, $[a]_\mu = a\,[1]_\mu$, where $[1]_\mu$ is a fixed generator of $\mathbb{G}_\mu$. A *bilinear group generator* $\mathsf{BGgen}(1^\lambda)$ returns $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, [1]_1, [1]_2)$, where p (a large prime) is the order of cyclic abelian groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$. Finally, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficient non-degenerate bilinear pairing, s.t. $\hat{e}([a]_1, [b]_2) = [ab]_T$. Denote $[a]_1 \bullet [b]_2 = \hat{e}([a]_1, [b]_2)$.

We adopt the definition of NI zap and NI-ZoK arguments from [29, 20], and the definition of standard, subversion-resistant and updatable NIZK arguments from [26, 7, 1, 27, 2], and in few cases we define a natural extension of current definitions that we achieve in our proposed schemes. Let $\mathcal{R}$ be a relation generator, such that $\mathcal{R}(1^\lambda)$ returns a polynomial-time decidable binary relation $\mathbf{R} = \{(\mathsf{x}, \mathsf{w})\}$, where x is the statement and w is the corresponding witness. Let $\mathbf{L_R} = \{\mathsf{x} : \exists\, \mathsf{w} \mid (\mathsf{x}, \mathsf{w}) \in \mathbf{R}\}$ be an **NP**-language including all the statements which there exist corresponding witnesses in relation $\mathbf{R}$.

2.1 NI Zap and NI-ZoK Arguments

Definition 1 (NI Zap). *A pair of PPT algorithms* (P, V) *is a non-interactive zap (a.k.a. NIWI) for an* **NP** *relation* $\mathbf{R}$ *if it satisfies,*

1. Completeness*: For all* $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}$*,* $\Pr[\pi \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{x}, \mathsf{w}) : \mathsf{V}(\mathbf{R}, \mathsf{x}, \pi) = 1] = 1$.
2. Soundness*: there exists a negligible function* $\mathsf{negl}(\lambda)$*, such that for every* $\mathsf{x} \notin \mathbf{L_R}$ *and* $\pi \in \{0,1\}^\star$*:* $\Pr[\mathsf{V}(\mathbf{R}, \mathsf{x}, \pi) = 1] \leq \mathsf{negl}(\lambda)$*.*
3. Witness Indistinguishable (WI)*: for any sequence* $I = \{(\mathsf{x}, \mathsf{w}_1, \mathsf{w}_2) : (\mathsf{x}, \mathsf{w}_1) \in \mathbf{R} \wedge (\mathsf{x}, \mathsf{w}_2) \in \mathbf{R}\}$*:*

$$\{\pi_1 \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{x}, \mathsf{w}_1) : \pi_1\}_{(\mathsf{x}, \mathsf{w}_1, \mathsf{w}_2) \in I} \approx_c \{\pi_2 \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{x}, \mathsf{w}_2) : \pi_2\}_{(\mathsf{x}, \mathsf{w}_1, \mathsf{w}_2) \in I}.$$

NI-ZoK arguments are NI zaps that additionally satisfy KS, defined as bellow:

Definition 2 (Knowledge Soundness (KS)). *A NI argument* Ψ_{NIZK} *is* KS *for* $\mathcal{R}$*, if for any PPT* $\mathcal{A}$*, there exists a PPT* $\mathsf{Ext}_\mathcal{A}$ *s.t. for all* λ*,*

$$\Pr\left[\begin{array}{l}\mathbf{R} \leftarrow \mathcal{R}(1^\lambda), \mathsf{srs} \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}), r \leftarrow\!\!\$\ \mathsf{RND}(\mathcal{A}), ((\mathsf{x}, \pi) \parallel \mathsf{w}) \leftarrow ...\\ ...(\mathcal{A} \parallel \mathsf{Ext}_\mathcal{A})(\mathbf{R}, \mathsf{srs}; r) : (\mathsf{x}, \mathsf{w}) \notin \mathbf{R} \wedge \mathsf{V}(\mathbf{R}, \mathsf{srs}, \mathsf{x}, \pi) = 1\end{array}\right] = \mathsf{negl}(\lambda),$$

2.2 NIZKs with Trusted, Subverted, and Updatable SRS

A *NIZK argument* Ψ_{NIZK} with updatable SRS for $\mathcal{R}$ consists of PPT algorithms $(\mathsf{K}_{\mathsf{srs}}, \mathsf{SU}, \mathsf{SV}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$, such that:

- $(\mathsf{srs}_0, \Pi_{\mathsf{srs}_0}) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R})$: Given $\mathbf{R}$, $\mathsf{K}_{\mathsf{srs}}$ sample the trapdoor ts and then use it to generate srs_0 along with Π_{srs_0} as a proof of its well-formedness.
- $(\mathsf{srs}_i, \Pi_{\mathsf{srs}_i}) \leftarrow \mathsf{SU}(\mathbf{R}, \mathsf{srs}_{i-1}, \{\Pi_{\mathsf{srs}_j}\}_{j=0}^{i-1})$: SU return the pair of $(\mathsf{srs}_i, \Pi_{\mathsf{srs}_i})$, where srs_i is the updated SRS and Π_{srs_i} is a proof for correct updating.
- $(\perp/1) \leftarrow \mathsf{SV}(\mathbf{R}, \mathsf{srs}_i, \{\Pi_{\mathsf{srs}_j}\}_{j=0}^{i})$: Given a potentially updated srs_i, and $\{\Pi_{\mathsf{srs}_j}\}_{j=0}^{i}$, SV return either $\perp$ (if srs_i is incorrectly formed or updated) or 1.
- $(\pi/\perp) \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{srs}_i, \mathsf{x}, \mathsf{w})$: Given the tuple of $(\mathbf{R}, \mathsf{srs}_i, \mathsf{x}, \mathsf{w})$, such that $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}$, P output an argument π. Otherwise, it returns $\perp$.

– $(0/1) \leftarrow \mathsf{V}(\mathbf{R}, \mathsf{srs}_i, \mathsf{x}, \pi)$: Given $(\mathbf{R}, \mathsf{srs}_i, \mathsf{x}, \pi)$, V verify the proof π and return either 0 (reject) or 1 (accept).
– $\pi \leftarrow \mathsf{Sim}(\mathbf{R}, \mathsf{srs}_i, \mathsf{ts}_i, \mathsf{x})$: Given $(\mathbf{R}, \mathsf{srs}_i, \mathsf{ts}_i, \mathsf{x})$, where ts_i is the (simulation) trapdoor of the updated SRS, namely srs_i, Sim output a simulated π.

In the standard SRS model, a NIZK argument for $\mathcal{R}$ has a tuple of algorithms $(\mathsf{K}_{\mathsf{srs}}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$ (and $\mathsf{K}_{\mathsf{srs}}$ does not return the Π_{srs_0}), while subversion-resistant constructions [7, 1] additionally have an SV algorithm which is used to verify the well-formedness of the SRS elements to achieve Sub-ZK [7]. But as listed above, in the *updatable* SRS model, a NIZK argument additionally has an SU algorithm that allows the parties to update the SRS and add their own private shares to the SRS generation. Note that in the latest case, the algorithms $\mathsf{K}_{\mathsf{srs}}, \mathsf{SU}$ and SV do not necessarily need $\mathbf{R}$, and they just deduce 1^λ and the length of the SRS from it. In the standard case, Ψ_{NIZK} is expected to satisfy *completeness*, *ZK* and *soundness*, while in the case of subverted setup, Ψ_{NIZK} is expected to achieve *Sub-WI*, *Sub-ZK* and *Sub-SND*. Finally, in the *updatable* SRS model, Ψ_{NIZK} can achieve *updatable completeness*, *U-ZK*, *U-SND*, *U-SE*, *U-SS*, and *U-SE*.

Next, we recall the security definitions of Sub-WI, Sub-ZK, U-SND, U-SS and U-SE used or achieved by our presented constructions, and refer to App. A for the definitions for NIZK arguments in the trusted SRS model.

Definition 3 (Sub-WI). *A NI argument Ψ is* Sub-WI for $\mathcal{R}$*, if for any PPT* Sub*, for all λ, all $\mathbf{R} \in \operatorname{im}(\mathcal{R}(1^\lambda))$, and for any PPT $\mathcal{A}$, one has $\varepsilon_0 \approx_\lambda \varepsilon_1$, where*

$$\varepsilon_b = \Pr\left[\begin{matrix} r \leftarrow_\$ \mathsf{RND}(\mathsf{Sub}), (\mathsf{srs}, \xi_{\mathsf{Sub}}) \leftarrow \mathsf{Sub}(\mathbf{R}; r) : \\ \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\mathbf{R}, \mathsf{srs}, \xi_{\mathsf{Sub}}) = 1 \end{matrix}\right] .$$

Here, ξ_{Sub} is auxiliary information generated by subvertor Sub*, and the oracle $\mathsf{O}_b(\mathsf{x}, \mathsf{w}_b)$ returns $\perp$ (reject) if $(\mathsf{x}, \mathsf{w}_b) \notin \mathbf{R}$, and otherwise it returns $\mathsf{P}(\mathbf{R}, \mathsf{srs}, \mathsf{x}, \mathsf{w}_b)$.*

Definition 4 (Sub-ZK). *A NI argument Ψ is* computationally Sub-ZK for $\mathcal{R}$*, if for any PPT subvertor* Sub *there exists a PPT extractor $\mathsf{Ext}_{\mathsf{Sub}}$, s.t. for all λ, all $\mathbf{R} \in \operatorname{im}(\mathcal{R}(1^\lambda))$, and for any PPT $\mathcal{A}$, one has $\varepsilon_0 \approx_\lambda \varepsilon_1$, where*

$$\varepsilon_b = \Pr\left[\begin{matrix} r \leftarrow_\$ \mathsf{RND}(\mathsf{Sub}), ((\mathsf{srs}, \Pi_{\mathsf{srs}}, \xi_{\mathsf{Sub}}) \parallel \mathsf{ts}) \leftarrow (\mathsf{Sub} \parallel \mathsf{Ext}_{\mathsf{Sub}})(\mathbf{R}; r) : \\ \mathsf{SV}(\mathbf{R}, \mathsf{srs}, \Pi_{\mathsf{srs}}) = 1 \wedge \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\mathbf{R}, \mathsf{srs}, \mathsf{ts}, \xi_{\mathsf{Sub}}) = 1 \end{matrix}\right] .$$

Here, ξ_{Sub} is auxiliary information generated by subvertor Sub*, and the oracle $\mathsf{O}_0(\mathsf{x}, \mathsf{w})$ returns $\perp$ (reject) if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R}$, and otherwise it returns $\mathsf{P}(\mathbf{R}, \mathsf{srs}, \mathsf{x}, \mathsf{w})$. Similarly, $\mathsf{O}_1(\mathsf{x}, \mathsf{w})$ returns $\perp$ (reject) if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R}$, and otherwise it returns $\mathsf{Sim}(\mathbf{R}, \mathsf{srs}, \mathsf{ts}, \mathsf{x})$. Ψ is* perfectly Sub-ZK for $\mathcal{R}$ *if one requires that $\varepsilon_0 = \varepsilon_1$.*

Definition 5 (U-SND). *A NI argument Ψ_{NIZK} is* updatable sound for $\mathcal{R}$*, if for every PPT $\mathcal{A}$, any PPT* Sub*, for all $\mathbf{R} \in \operatorname{im}(\mathcal{R}(1^\lambda))$, and for all λ,*

$$\Pr\left[\begin{matrix} (\mathsf{srs}_0, \Pi_{\mathsf{srs}_0}) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}), (\{\mathsf{srs}_j, \Pi_{\mathsf{srs}_j}\}_{j=1}^i, \xi_{\mathsf{Sub}}) \leftarrow \mathsf{Sub}(\mathsf{srs}_0, \Pi_{\mathsf{srs}_0}), \\ \mathsf{SV}(\mathsf{srs}_i, \{\Pi_{\mathsf{srs}_j}\}_{j=0}^i) = 1, (\mathsf{x}, \pi) \leftarrow \mathcal{A}(\mathbf{R}, \mathsf{srs}_i, \xi_{\mathsf{Sub}}) : \mathsf{x} \notin \mathbf{L} \wedge \mathsf{V}(\mathbf{R}, \mathsf{srs}_i, \mathsf{x}, \pi) = 1 \end{matrix}\right] \approx_\lambda 0 .$$

Here $\mathsf{RND}(\mathcal{A}) = \mathsf{RND}(\mathsf{Sub})$, *and* Π_{srs} *is a proof for correctness of SRS generation or updating process. In the definition,* ξ_{Sub} *can be seen as an auxiliary information provided by* Sub *to* $\mathcal{A}$.

Definition 6 (U-SS). *A NI argument* Ψ_{NIZK} *is* updatable simulation soundness for $\mathcal{R}$, *if for all* $\mathbf{R} \in \mathrm{im}(\mathcal{R}(1^\lambda))$, *for any PPT* Sub, *and every PPT* $\mathcal{A}$, *for all* λ,

$$\Pr\left[\begin{array}{l}((\mathsf{srs}_0, \Pi_{\mathsf{srs}_0}) \,\|\, \mathsf{ts}_0 := \mathsf{ts}'_0) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}), r_s \leftarrow\!\$\ \mathsf{RND}(\mathsf{Sub}),\\ ((\{\mathsf{srs}_j, \Pi_{\mathsf{srs}_j}\}_{j=1}^{i}, \xi_{\mathsf{Sub}}) \,\|\, \{\mathsf{ts}'_j\}_{j=1}^{i}) \leftarrow (\mathsf{Sub} \,\|\, \mathsf{Ext}_{\mathsf{Sub}})(\mathsf{srs}_0, \Pi_{\mathsf{srs}_0}, r_s),\\ \mathsf{SV}(\mathsf{srs}_i, \{\Pi_{\mathsf{srs}_j}\}_{j=0}^{i}) = 1, (\mathsf{x}, \pi) \leftarrow \mathcal{A}^{\mathsf{O}(\mathsf{ts}_i,\ldots)}(\mathbf{R}, \mathsf{srs}_i, \xi_{\mathsf{Sub}}) :\\ (\mathsf{x}, \pi) \notin Q \wedge \mathsf{x} \notin \mathbf{L} \wedge \mathsf{V}(\mathbf{R}, \mathsf{srs}_i, \mathsf{x}, \pi) = 1\end{array}\right] = \mathsf{negl}(\lambda),$$

where Π_{srs} *is a proof for correctness of SRS generation/updating,* ts_i *is the simulation trapdoor associated with the final SRS that can be computed using* $\{\mathsf{ts}'_j\}_{j=0}^{i}$, *and* Q *is the set of simulated statement-proof pairs returned by* $\mathsf{O}(.)$.

Definition 7 (U-SE). *A NI argument* Ψ_{NIZK} *is* updatable simulation-extractable (with non-black-box extraction) for $\mathcal{R}$, *if for every PPT* $\mathcal{A}$ *and* Sub, *for all* $\mathbf{R} \in \mathrm{im}(\mathcal{R}(1^\lambda))$, *the following probability is* $\mathsf{negl}(\lambda)$,

$$\Pr\left[\begin{array}{l}((\mathsf{srs}_0, \Pi_{\mathsf{srs}_0}) \,\|\, \mathsf{ts}_0 := \mathsf{ts}'_0) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}), r_s \leftarrow\!\$\ \mathsf{RND}(\mathsf{Sub}),\\ ((\{\mathsf{srs}_j, \Pi_{\mathsf{srs}_j}\}_{j=1}^{i}, \xi_{\mathsf{Sub}}) \,\|\, \{\mathsf{ts}'_j\}_{j=1}^{i}) \leftarrow (\mathsf{Sub} \,\|\, \mathsf{Ext}_{\mathsf{Sub}})(\mathsf{srs}_0, \Pi_{\mathsf{srs}_0}, r_s),\\ \mathsf{SV}(\mathsf{srs}_i, \{\Pi_{\mathsf{srs}_j}\}_{j=0}^{i}) = 1, r_{\mathcal{A}} \leftarrow\!\$\ \mathsf{RND}(\mathcal{A}), (\mathsf{x}, \pi) \leftarrow \mathcal{A}^{\mathsf{O}(\mathsf{ts}_i,\ldots)}(\mathbf{R}, \mathsf{srs}_i, \xi_{\mathsf{Sub}}; r_{\mathcal{A}}),\\ \mathsf{w} \leftarrow \mathsf{Ext}_{\mathcal{A}}(\mathbf{R}, \mathsf{srs}_i, \mathsf{Ext}_{\mathsf{Sub}}, \pi) : (\mathsf{x}, \pi) \notin Q \wedge (\mathsf{x}, \mathsf{w}) \notin \mathbf{R} \wedge \mathsf{V}(\mathbf{R}, \mathsf{srs}_i, \mathsf{x}, \pi) = 1\end{array}\right],$$

where $\mathsf{Ext}_{\mathsf{Sub}}$ *in a PPT extractor,* Π_{srs} *is a proof for correctness of SRS generation/updating, and* ts_i *is the simulation trapdoor associated with the final SRS that can be computed using* $\{\mathsf{ts}'_j\}_{j=0}^{i}$. *Here,* $\mathsf{RND}(\mathcal{A}) = \mathsf{RND}(\mathsf{Sub})$ *and* Q *is the set of the statement and simulated proofs returned by* $\mathsf{O}(.)$.

2.3 Assumptions

Definition 8 (Diffie-Hellman Knowledge of Exponent Assumption (DH-KEA) [7]). *We say that a bilinear group generator* BGGen *is* DH-KE secure *for relation set* $\mathcal{R}$ *if for any* λ, $\mathbf{R} \in \mathrm{im}(\mathcal{R}(1^\lambda))$, *and PPT adversary* $\mathcal{A}$ *there exists a PPT extractor* $\mathsf{Ext}_{\mathcal{A}}$, *such that, the following probability is* $\mathsf{negl}(\lambda)$,

$$\Pr\left[\begin{array}{l}(p, \mathbb{G}, \mathbb{G}_T, \hat{e}, [1]_1) \leftarrow \mathsf{BGGen}(1^\lambda), r \leftarrow\!\$\ \mathsf{RND}(\mathcal{A}), ([k]_1, [l]_1) \leftarrow \mathbb{G}, ([s]_1, [t]_1, [st]_1\\ \,\|\, x, y) \leftarrow (\mathcal{A} \,\|\, \mathsf{Ext}_{\mathcal{A}})(\mathbf{R}, [k]_1, [l]_1; r) : [s]_1 \bullet [t]_1 = [1]_1 \bullet [st]_1 \wedge x \neq s \wedge y \neq t\end{array}\right].$$

Definition 9 (Computational Diffie-Hellman (CDH) Assumption). *We say that the decisional Diffie-Hellman assumption holds for a group generator* GGen *if for any PPT adversary* $\mathcal{A}$ *the following probability is* $\mathsf{negl}(\lambda)$,

$$\Pr\left[(p, \mathbb{G}, g) \leftarrow \mathsf{GGen}(1^\lambda), x, y \leftarrow \mathbb{Z}_p, h \leftarrow \mathcal{A}(\lambda, g^x, g^y) : h = g^{xy}\quad\right].$$

Definition 10 (Decisional Linear (dLin) Assumption [11]). *We say that the decisional linear assumption holds for a group generator* GGen *if for any PPT adversary* $\mathcal{A}$ *the following probability is* $\mathsf{negl}(\lambda)$*,*

$$\Pr\left[\begin{matrix}(p, \mathbb{G}, g) \leftarrow \mathsf{GGen}(1^\lambda), u, v, s, t, \xi \leftarrow \mathbb{Z}_p, b \leftarrow \{0, 1\} \\ b' \leftarrow \mathcal{A}(1^\lambda, g^u, g^v, g^{us}, g^{vt}, g^{s+t+b\xi}) : b = b'\end{matrix}\right].$$

Definition 11 (Bilinear Diffie-Hellman Knowledge of Exponent (BDH-KE) Assumption [1]). *We say* BGgen *is* BDH-KE secure *for relation set* $\mathcal{R}$ *if for any* λ*,* $\mathbf{R} \in \mathrm{im}(\mathcal{R}(1^\lambda))$*, and PPT adversary* $\mathcal{A}$*, there exists a PPT extractor* $\mathsf{Ext}_\mathcal{A}$*, such that, the following probability is* $\mathsf{negl}(\lambda)$*,*

$$\Pr\left[\begin{matrix}(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, [1]_1, [1]_2) \leftarrow \mathsf{BGgen}(1^\lambda), r \leftarrow\!\!\$\ \mathsf{RND}(\mathcal{A}), ([\alpha_1]_1, [\alpha_2]_2 \\ \| a) \leftarrow (\mathcal{A} \| \mathsf{Ext}_\mathcal{A})(\mathbf{R}, r) : [\alpha_1]_1 \bullet [1]_2 = [1]_1 \bullet [\alpha_2]_2 \wedge a \neq \alpha_1\end{matrix}\right].$$

2.4 Key Updatable, One-time, and Key Homomorphic Signatures

Some of our constructions exploit key-updatable signature schemes that allow parties to update the secret and public keys and inject their shares into the keys. We adopt their definition from [2], which we review next.

An updatable signature scheme Σ is a tuple $\Sigma = (\mathsf{KG}, \mathsf{Upk}, \mathsf{Vpk}, \mathsf{Sign}, \mathsf{SigVerify})$ of PPT algorithms, which are defined as follows.

- $\mathsf{KG}(1^\lambda)$: Given 1^λ, return a signing key sk and a verification key pk with associated message space $\mathcal{M}$.
- $\mathsf{Upk}(\mathsf{pk}_{i-1})$: Given a pk_{i-1}, output an updated verification key pk_i with the associated secret updating key $\bar{\mathsf{sk}}_i$, and a proof Π_i.
- $\mathsf{Vpk}(\mathsf{pk}_i, \mathsf{pk}_{i-1}, \Pi_i)$: Given a verification key pk_{i-1}, an updated verification key pk_i, and the proof Π_i, check if pk_i has been updated correctly.
- $\mathsf{Sign}(\mathsf{sk}_i, m)$: Given sk_i and a message $m \in \mathcal{M}$, output a signature σ_i.
- $\mathsf{SigVerify}(\mathsf{pk}_i, m, \sigma_i)$: Given a verification key pk_i, a message $m \in \mathcal{M}$, and a signature σ_i, output a bit $b \in \{0, 1\}$.

Definition 12 (Updatable Correctness). *An updatable signature scheme* Σ *is correct, if*

$$\Pr\left[\begin{matrix}(\mathsf{sk}_{i-1}, \mathsf{pk}_{i-1}, \Pi_{i-1}) \leftarrow \mathsf{KG}(1^\lambda), (\bar{\mathsf{sk}}_i, \mathsf{pk}_i, \Pi_i) \leftarrow \mathsf{Upk}(\mathsf{pk}_{i-1}), \\ \mathsf{Vpk}(\mathsf{pk}_i, \mathsf{pk}_{i-1}, \Pi_i) = 1 : \mathsf{SigVerify}(\mathsf{pk}_{i-1}, m, \mathsf{Sign}(\mathsf{sk}_{i-1}, m)) = 1 \\ \wedge \mathsf{SigVerify}(\mathsf{pk}_i, m, \mathsf{Sign}(\mathsf{sk}_{i-1} + \bar{\mathsf{sk}}_i, m)) = 1\end{matrix}\right] = 1 .$$

Definition 13 (Updatable EUF-CMA). *A signature scheme* Σ *is updatable EUF-CMA secure, if for all PPT subverter* Sub*, there exists a PPT extractor* $\mathsf{Ext}_\mathsf{Sub}$ *such that for all* λ*, and all PPT adversaries* $\mathcal{A}$

$$\Pr\left[\begin{matrix}(\mathsf{sk}_{i-1}, \mathsf{pk}_{i-1}, \Pi_{i-1}) \leftarrow \mathsf{KG}(1^\lambda), w_\mathsf{Sub} \leftarrow\!\!\$\ \mathsf{RND}(\mathsf{Sub}), \\ (\mathsf{pk}_i, \Pi_i, \xi_\mathsf{Sub}) \leftarrow \mathsf{Sub}(pk_{i-1}; w_\mathsf{Sub}), \bar{\mathsf{sk}}_i \leftarrow \mathsf{Ext}_\mathsf{Sub}(\mathsf{pk}_{i-1}; w_\mathsf{Sub}) \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk}_{i-1} + \bar{\mathsf{sk}}_i, \cdot)}(\mathsf{pk}_i, \xi_\mathsf{Sub}) : \mathsf{Vpk}(\mathsf{pk}_i, \mathsf{pk}_{i-1}, \Pi_i) = 1 \\ \wedge \mathsf{pk}_i = \mathsf{pk}_{i-1} \cdot \mu(\bar{\mathsf{sk}}_i) \wedge \mathsf{SigVerify}(\mathsf{pk}, m^*, \sigma^*) = 1 \wedge m^* \notin \mathcal{Q}^\mathsf{Sign}\end{matrix}\right] = \mathsf{negl}(\lambda),$$

where μ is an efficiently computable map from the secret key to the public key space, and $\mathcal{Q}^{\mathsf{Sign}}$ is the list of queries and responses to the signing oracle.

We also require one-time signature schemes in our construction.

Definition 14 (Strong One-Time Signature Scheme). *A strong one-time signature scheme Σ_{OT} is a signature scheme Σ which satisfies the following unforgeability notion: for all PPT adversaries $\mathcal{A}$*

$$\Pr\left[\begin{matrix}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KG}(1^\lambda), (m^*,\sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},.)}(\mathsf{pk}) \\ \mathsf{SigVerify}(\mathsf{pk}, m^*, \sigma^*) = 1 \wedge (m^*,\sigma^*) \notin \mathcal{Q}^{\mathsf{Sign}}\end{matrix}\right] = \mathsf{negl}(\lambda),$$

where the oracle $\mathsf{Sign}(\mathsf{sk}; m) := \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$ can only be called once.

We now recall the notion of key-homomorphic signatures [16]. We assume that the secret key space $\mathbb{H}$ and the public key space $\mathbb{E}$ are groups with two operations that we will denote by $+$ and $\cdot$ respectively, by convention.

Definition 15 (Secret Key to Public Key Homomorphism). *A signature scheme $\Sigma = (\mathsf{KG}, \mathsf{Sign}, \mathsf{SigVerify})$ provides a secret key to public key homomorphism if there is an efficiently computable homomorphism $\mu : \mathbb{H} \to \mathbb{E}$ such that for all key pairs $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KG}\left(1^\lambda\right)$ one has $\mathsf{pk} = \mu(\mathsf{sk})$.*

Definition 16 (Key-Homomorphic Signatures). *A signature scheme is called key-homomorphic, if it provides a secret key to public key homomorphism and an additional PPT algorithm Adapt, defined as:*
$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$*: Given a public key pk, a message m, a signature σ, and a shift amount Δ outputs a public key pk' and a signature σ', such that for all $\Delta \in \mathbb{H}$ and all $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KG}(1^\lambda)$, all messages $m \in \mathcal{M}$ and all $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$ and $(\mathsf{pk}', \sigma') \leftarrow \mathsf{Adapt}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m)$ it holds that*

$$Pr[\mathsf{SigVerify}(\mathsf{pk}', m, \sigma') = 1] = 1 \wedge \mathsf{pk}' = \mu(\Delta) \cdot \mathsf{pk}.$$

Definition 17 (Adaptability of Signatures). *A key-homomorphic signature scheme provides adaptability of signatures, if for every $\lambda \in \mathbb{N}$ and every message $m \in \mathcal{M}$, it holds that $[(\mathsf{sk},\mathsf{pk}), \mathsf{Adapt}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m), \Delta)]$, where $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KG}(1^\lambda), \Delta \leftarrow \mathbb{H}$, and $[(\mathsf{sk}, \mu(sk)), (\mu(sk) \cdot \mu(\Delta), \mathsf{Sign}(\mathsf{sk} + \Delta, m))]$, where $\mathsf{sk} \leftarrow \mathbb{H}, \Delta \leftarrow \mathbb{H}$, are identically distributed.*

3 Bellare et al.'s Sub-ZK NIZK Can Achieve U-SND

In this section, we show that the technique proposed by Bellare, Fuchsbauer and Scafuro [7] to build Sub-ZK NIZK arguments can be expanded to build NIZK arguments with updatable and universal SRS. In order to build a Sub-ZK and sound NIZK argument from NI zaps, they defined an SRS as a tuple of knowledge assumption, namely $([x]_1, [y]_1, [xy]_1)$, and let a proof for a statement x prove that either there is a witness for x or one knows x or y. They proved knowledge by adding a ciphertext C (of linear encryption) and using a perfectly sound NI zap to prove that either $\mathsf{x} \in \mathbf{L}$ or C encrypts x or y. Their construction uses NI zap of Groth, Ostrovsky and Sahai [29].

In the resulting NIZK argument, ZK is achieved since by encrypting the trapdoor x (or y) proofs can be simulated, and by IND-CPA of C and WI of the NI zap, they are indistinguishable from the real ones. Soundness is achieved since by soundness of the NI zap, a proof for a wrong statement must contain encryption of x or y, which is hard to obtain from an honestly generated SRS. They defined the SRS, as a tuple of a knowledge assumption in bilinear groups, which is *publicly verifiable* by paring, such that they can extract either x or y from an SRS subverter, and simulate the proofs without the need for a trusted third party, consequently achieving Sub-ZK. Nowadays, the mentioned technique is used in all Sub-ZK NIZK arguments, e.g. [1, 19, 27, 4, 2]. In nutshell, to achieve Sub-ZK, one first check the well-formedness of SRS elements by pairing equations, and then relying on a knowledge assumption extracts the simulation trapdoors from the SRS subverter and provides to the simulation algorithm constructed in the proof of (standard) ZK. In Bellare, Fuchsbauer and Scafuro's construction [7], another important point to achieve Sub-ZK was that the keys of the encryption scheme were generated with the prover, rather than the subverter. Therefore a subverter or verifier does not have access to the secret key to decrypt C and learn the witness, but the reduction allows non-black-box extraction under a knowledge assumption. We refer to the original work for further details [7].

Expanding Bellare, Fuchsbauer and Scafuro's OR Technique to Achieve U-SND. The folklore OR technique, introduced by [8], is shown to be a prominent technique for designing various cryptographic primitives. As reviewed above, Bellare, Fuchsbauer and Scafuro [7] showed that using such techniques with a NI zap and a *trapdoor-extractable* SRS, e.g. tuple of a knowledge assumption, one can build NIZK argument that can achieve Sub-ZK and SND. As a part of our contribution, we noticed that using NI zaps (resp. NI-ZoKs) with a similar OR language but with an *updatable* and trapdoor-extractable SRS, we can build Sub-ZK NIZK arguments (of knowledge) with *universal* and *updatable* SRS.

The universality of the resulting NIZK argument is inherited from the input NI zap. Particular about their construction [7], one can see that given the SRS, which includes the tuple of DH-KE assumption, namely $([x]_1, [y]_1, [xy]_1)$, a prover needs to prove that either there is a witness for x or he knows x or y, by sending a ciphertext C that encrypts x or y. In other words, the SRS is generated independently of the statement to be proved, and the same SRS can be used to prove different statements defined in the first clause. Note that in the resulting NIZK argument $[t_i]_1$ is used to prove the soundness of scheme under CDH assumption, and to achieve Sub-ZK, we only need to extract one of the simulation trapdoors x_i or y_i from the Sub (a malicious SRS generator).

For updatability, next we describe an updating algorithm for the Sub-ZK NIZK argument of Bellare, Fuchsbauer and Scafuro [7] that allows one (e.g. a verifier) to update the SRS and bypass the need for a trusted setup by injecting his share to the generation of SRS. After updating the SRS, the updater needs

SRS Generator, $\mathsf{K}_{\mathsf{srs}}(\mathbf{R_L})$: $(p, \mathbb{G}_1, \mathbb{G}_T, \hat{e}, [1]_1) \leftarrow \mathsf{BGgen}(1^\lambda)$; Sample $x_0, y_0, t_0 \leftarrow\$ \mathbb{Z}_p$, where x_0, y_0 are the simulation trapdoors; Set the string $\mathsf{srs}_0 := ([t_0]_1, [x_0]_1, [y_0]_1, [x_0y_0]_1)$ and the well-formedness proof $\pi_0 := (([x_0]_1, [y_0]_1, [x_0y_0]_1), ([x_0]_1, [y_0]_1, [x_0y_0]_1))$; Return (srs_0, π_0).

SRS Update, $\mathsf{SU}(\mathbf{R_L}, \mathsf{srs}_{i-1}, \{\pi_j\}_{j=0}^{i-1})$: Given (a potentially updated) srs_{i-1}, parse $\mathsf{srs}_{i-1} := ([t_{i-1}]_1, [x_{i-1}]_1, [y_{i-1}]_1, [x_{i-1}y_{i-1}]_1)$; Sample $t'_i, x'_i, y'_i \leftarrow\$ \mathbb{Z}_p$; Set $[t_i]_1 := t'_i[t_{i-1}]_1$, $[x_i]_1 := x'_i[x_{i-1}]_1$, $[y_i]_1 := y'_i[y_{i-1}]_1$, $[x_iy_i]_1 := x'_iy'_i[x_{i-1}y_{i-1}]_1$; Set the string $\mathsf{srs}_i := ([t_i]_1, [x_i]_1, [y_i]_1, [x_iy_i]_1)$ and the proof $\pi_i := (([x_i]_1, [y_i]_1, [x_iy_i]_1), ([x'_i]_1, [y'_i]_1, [x'_iy'_i]_1))$; Return (srs_i, π_i).

SRS Verification, $\mathsf{SV}(\mathsf{srs}_i, \{\pi_j\}_{j=0}^{i})$: Given $(\mathsf{srs}_i, \{\pi_j\}_{j=0}^{i})$, in order to verify the well-formedness of srs_i, act as follows:

- If $i = 0$: parse $\mathsf{srs}_0 := ([t_0]_1, [x_0]_1, [y_0]_1, [x_0y_0]_1)$ and $\pi_0 := (([x_0]_1, [y_0]_1, [x_0y_0]_1), ([x_0]_1, [y_0]_1, [x_0y_0]_1))$ and do the following:
  1. Check if $[x_0]_1 \bullet [y_0]_1 = [1]_1 \bullet [x_0y_0]_1$;
  2. Check if $([t_0]_1 \neq 1) \cap ([t_0]_1 \in \mathbb{G}_1)$;
- If $i \geq 1$: parse $\mathsf{srs}_i := ([t_i]_1, [x_i]_1, [y_i]_1, [x_iy_i]_1)$ and $\{\pi_j := (([x_j]_1, [y_j]_1, [x_jy_j]_1), ([x'_j]_1, [y'_j]_1, [x'_jy'_j]_1))\}_{j=0}^{i}$ and do the following:
  1. Check if $[x_0]_1 = [x'_0]_1$, $[y_0]_1 = [y'_0]_1$, and $[x_0y_0]_1 = [x'_0y'_0]_1$;
  2. Check if $[x'_j]_1 \bullet [y'_j]_1 = [x'_jy'_j]_1 \bullet [1]_1$ for $j = 0, 1, \cdots, i$;
  3. Check if $[x_j]_1 \bullet [1]_1 = [x_{j-1}]_1 \bullet [x'_j]_1$ for $j = 1, 2, \cdots, i$;
  4. Check if $[y_j]_1 \bullet [1]_1 = [y_{j-1}]_1 \bullet [y'_j]_1$ for $j = 1, 2, \cdots, i$;
  5. Check if $[x_jy_j]_1 \bullet [1]_1 = [x_{j-1}y_{j-1}]_1 \bullet [x'_jy'_j]_1$ for $j = 1, 2, \cdots, i$;
  6. Check if $[x_i]_1 \bullet [y_i]_1 = [1]_1 \bullet [x_iy_i]_1$;
  7. Check if $([t_i]_1 \neq 1) \cap ([t_i]_1 \in \mathbb{G}_1)$;

return 1 if all the checks pass (srs_i is well-formed), otherwise $\perp$.

Fig. 1. The SRS generation $\mathsf{K}_{\mathsf{srs}}$, SRS updating SU, and SRS verification SV algorithms for the Sub-ZK NIZK argument of Bellare, Fuchsbauer and Scafuro [7].

to return a proof to show that the SRS updating was done honestly and the updated SRS is well-formed. As we discuss later, verifying this proof is necessary to achieve Sub-ZK and U-SND. Fig. 1 describes the SRS generation $\mathsf{K}_{\mathsf{srs}}$, SRS verification SV, and our proposed SRS updating SU algorithms for the Sub-ZK NIZK argument of [7]. For the description of the proof generation and proof verification algorithms we refer to the original paper [7].

Efficiency. From the description of algorithms (in Fig. 1) one can see that the SRS generation algorithm requires 4 exponentiations in $\mathbb{G}_1$. Similarly, to update an SRS the SU algorithm requires 4 exponentiations in $\mathbb{G}_1$. The SU algorithm also requires 3 additional exponentiations in $\mathbb{G}_1$ to generate a well-formedness proof π_i. Finally, to verify an i-time updated SRS srs_i, $i \geq 1$, the SRS verification algorithm SV needs to compute $8i + 4$ pairings, which mostly are batchable.

Security Proof. Bellare, Fuchsbauer and Scafuro [7], showed that their NIZK argument satisfies SND under DH-KE and Computational Diffie-Hellman (CDH) assumptions; and Sub-ZK under DH-KEA and DLin assumption. Next, we show that their construction can also achieve U-SND. To this end, first, we recall a corollary from [23, Lemma 6], and then give a lemma which is used in the security proof. The lemma proves that in their construction even when given an honestly generated SRS as input, updaters need to know their injected share to the (simulation) trapdoor. In this way security against the updater is linked to an honest SRS.

Corollary 1. *In the updatable SRS model, single adversarial updates imply full updatable security [23, Lemma 6].*

Lemma 1 (Trapdoor Extraction from Updated SRS). *Considering the algorithms in Fig. 1, suppose that there exists a PPT adversary $\mathcal{A}$ such that given $(\mathsf{srs}_0, \pi_0) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R_L})$, $\mathcal{A}$ returns the updated SRS (srs_i, π_i), where $\mathsf{SV}(\mathsf{srs}_i, \{\pi_j\}_{j=0}^{1}) = 1$ with non-negligible probability. Then, the DH-KE assumption implies that there exists a PPT extractor $\mathsf{Ext}_{\mathcal{A}}$ that, given the randomness of $\mathcal{A}$ as input, outputs x'_1 or y'_1 such that $[x_1]_1 = x'_1[x_0]$, $[y_1]_1 = y'_1[y_0]$, and $[x_1y_1]_1 = x'_1y'_1[x_0y_0]$.*

Proof. Considering Corollary 1, we consider the case that $\mathcal{A}$ makes only one update to the SRS, but similar to [23, Lemma 6], it can be generalized. Parse π_0 as containing $([x_0]_1, [y_0]_1, [x_0y_0]_1)$ and parse srs_0 as containing $\mathsf{srs}_0 := ([t_0]_1, [x_0]_1, [y_0]_1, [x_0y_0]_1)$. We consider an adversary $\mathcal{A}$, that given srs_0 returns an updated SRS, srs_1, where it contains $([t_1]_1, [x_1]_1, [y_1]_1, [x_1y_1]_1)$ and its well-formedness proof π_1 contains $(([t_1]_1, [x_1]_1, [y_1]_1, [x_1y_1]_1), ([x'_1]_1, [y'_1]_1, [x'_1y'_1]_1))$. If the SRS verification accepts the updated SRS, namely if $\mathsf{SV}(\mathsf{srs}_i, \{\pi_j\}_{j=0}^{1}) = 1$, then the following equations hold,

$$[x'_0]_1 \bullet [y'_0]_1 = [1]_1 \bullet [x'_0y'_0]_1\,, \quad [x'_1]_1 \bullet [y'_1]_1 = [1]_1 \bullet [x'_1y'_1]_1\,, \quad [x_1]_1 \bullet [1]_1 = [x_0]_1 \bullet [x'_1]_1\,,$$

$$[y_1]_1 \bullet [1]_1 = [y_0]_1 \bullet [y'_1]_1\,, \; [x_1y_1]_1 \bullet [1]_1 = [x_0y_0]_1 \bullet [x'_1y'_1]_1\,, \; [x_1]_1 \bullet [y_1]_1 = [1]_1 \bullet [x_1y_1]_1\,,$$

By the equations, $[x'_0]_1 \bullet [y'_0]_1 = [1]_1 \bullet [x'_0y'_0]_1$, under the DH-KE assumption, there exists an extraction algorithm that can extract either x'_0 or y'_0. Similarly, by the equations $[x'_1]_1 \bullet [y'_1]_1 = [1]_1 \bullet [x'_1y'_1]_1$, and $[x_1]_1 \bullet [y_1]_1 = [x_1y_1]_1 \bullet [1]_1$ under the DH-KE assumption, there exists an extraction algorithm $\mathsf{Ext}_{\mathcal{A}}$ that given the randomness of $\mathcal{A}$, can extract x_0 or y_0, x'_1 or y'_1, and x_1 or y_1. The other equations check the consistency and the well-formedness of SRS and imply that $x_1 = x_0x'_1 = x'_0x'_1$ and $y_1 = y_0y'_1 = y'_0y'_1$. □

Given the constructed SU and SV algorithms (in Fig. 1), the following theorem shows that their construction can also achieve U-SND under the same assumption used in the security proof of the protocol.

Theorem 1. *Given the* SU *and* SV *algorithms described in Fig. 1, the Sub-ZK and SND NIZK argument proposed by Bellare, Fuchsbauer and Scafuro [7] additionally satisfies U-SND assuming the DH-KE and CDH assumptions hold.*

Proof. Considering Corollary 1, to prove this it suffices to prove security in the case that $\mathcal{A}$ makes only one update to the SRS. Imagine we have a PPT adversary $\mathcal{A}$ that given honestly sampled $(\mathsf{srs}_0, \pi_0) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R_L}, 1^\lambda)$, returns an updated SRS (srs_i, π_i) (along with a well-formedness proof) and a pair of (x, π) that get accepted; i.e., such that $\mathsf{SV}(\mathsf{srs}_i, \{\pi_j\}_{j=0}^{i}) = 1$, and also the verifier of NIZK argument accepts π as a valid proof for the statement x, $\mathsf{V}(\mathbf{R_L}, \mathsf{srs}_i, \mathsf{x}, \pi) = 1$. By Lemma 1, because the updated SRS verifies, there exists an extractor $\mathsf{Ext}_{\mathcal{A}}$ that outputs the SRS trapdoors τ, such that running the SU algorithm with trapdoors τ returns (srs_i, π_i).

The rest of proof is the same as the proof of soundness, which is given in [7]. □

Remark 1. The above theorem showed that the Sub-ZK NIZK argument of Bellare, Fuchsbauer and Scafuro [7] can also achieve U-SND. It is worth mentioning that their original construction was instantiated with the NI zap of Groth, Ostrovsky and Sahai [29] that is proven to achieve (Sub-)WI and (Sub-)SND. By instantiating their construction with the NI-ZoK of Fuchsbauer and Orru [20], which is a variant of [29], one can obtain a Sub-ZK NIZK argument of *knowledge* that can satisfy KS. Then, using our proposed SU and SV algorithms, one will obtain a NIZK argument that will satisfy Sub-ZK and U-KS.

4 Sub-ZK and U-{SS, SE} NIZKs from NI Zaps

In Sec. 3, we showed that the Sub-ZK NIZK argument of Bellare, Fuchsbauer and Scafuro [7], built with a NI zap argument and a trapdoor-extractable SRS, can also achieve U-SND. Moreover, it can be instantiated with a NI-ZoK [20] to build a Sub-ZK and U-KS NIZK argument. The resulting NIZK arguments are universal and have a succinct updatable SRS, which allows both the prover and verifier to efficiently update and bypass the need for a TTP. However, in practice one may require updatable NIZK arguments with even stronger security guarantees than U-SND and U-KS, so-called Updatable Simulation Soundness (U-SS) [6] and U-SE [2]. SS and SE imply SND and KS, respectively, even if the adversary sees a polynomial number of simulated proofs. Both notions guarantee the non-malleability of proofs [15] which is necessary for various applications. As in U-SND and U-KS, in the U-SS and U-SE, the adversary is also allowed to update the SRS, while the notions ensure that SS and SE are achieved as long as the initial SRS generation or one of SRS updates is done by an honest party.

As mentioned in the introduction, one of our key insights is that once we use NI zaps and define an OR language, then the NI zap does not have an SRS, and if the SRS of the second clause be the public key of a key-updatable signature scheme, then we can achieve even stronger security notions in the resulting Sub-ZK NIZK argument. In this section, we show that Bellare, Fuchsbauer and Scafuro's approach can be expanded to build an OR-based compiler that uses NI zaps and allows one to build Sub-ZK NIZK arguments with *succinct*, *universal* and *updatable* SRS that can satisfy U-SS or U-SE. To this end, we revise the OR-based compiler of Derler and Slamanig [16], which is presented to build SE NIZK arguments with *black-box* extractability, and present a new variant that can be used to build Sub-ZK NIZK arguments with succinct, universal and updatable SRS that can achieve U-SS or U-SE with *non-black-box* extraction. In [2], Abdolmaleki, Ramacher and Slamanig also presented a new variant of

Derler and Slamanig's compiler [16], so-called LAMASSU, that allows one to lift any Sub-ZK and U-KS zk-SNARK to a Sub-ZK and U-SE one. LAMASSU is particularly designed for updatable zk-SNARKs, and as discussed in ([2], Theorem 4), it requires the input zk-SNARK to satisfy U-ZK which is a stronger requirement than (sub-)WI that NI zaps achieve, and we require it in the new compiler. Considering the negative result that Sub-ZK and black-box extractability cannot be achieved at the same time [7, 6], Sub-ZK and U-SE with non-black-box extraction is the best achievable if we aim to retain Sub-ZK [6].

In Derler and Slamanig's OR-based construction [16], they use a EUF-CMA secure adaptable key-homomorphic signature scheme Σ and a strongly unforgeable one-time signature (sOTS) scheme Σ_{OT} to achieve non-malleability in the input KS NIZK argument Π, which results in a SE NIZK argument, but under a trusted setup. A key point about their construction is that the prover does not need to prove that a signature verifies under the verification key given in the SRS, as done in Groth's constructions [25]. Instead, the prover needs to prove that he knows the secret key of the public key given in the SRS, which he has used to adapt signatures produced with Σ_{OT}, to the ones valid under the public key given in the SRS. Next, we present a variant of their construction that given a NI zap, an sOTS scheme, and a *key-updatable* signature scheme allows one to build NIZK arguments with universal and updatable SRS. Key-updatable signature schemes are an extension of key-homomorphic signatures, where their keys are updatable, and they can guarantee unforgeability as long as at least one of the key updates or the key generation will be done honestly [2].

For the compiler, let $\mathbf{L}$ be an arbitrary NP-language $\mathbf{L} = \{x \mid \exists \mathsf{w} : \mathbf{R_L}(\mathsf{x}, \mathsf{w}) = 1\}$, for which we aim to construct a Sub-ZK and U-SS (or U-SE) NIZK, and let $\mathbf{L}'$ be the language for lifted construction, such that, $((\mathsf{x}, \mathsf{cpk}, \mathsf{pk}), (\mathsf{w}, \mathsf{csk} - \mathsf{sk})) \in \mathbf{R_{L'}}$, iff:

$$(\mathsf{x}, \mathsf{w}) \in \mathbf{R_L} \vee \mathsf{cpk} = \mathsf{pk} \cdot \mu(\mathsf{csk} - \mathsf{sk}),$$

where μ is an efficiently computable map, $(\mathsf{cpk}, \mathsf{csk})$ is a key pair for a key-updatable signature scheme, $(\mathsf{sk}, \mathsf{pk})$ is a key pair for an sOTS scheme. Let $\Sigma_{OT} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{SigVerify})$ be an sOTS scheme, $\Sigma = (\mathsf{KG}, \mathsf{Upk}, \mathsf{Vpk}, \mathsf{Sign}, \mathsf{SigVerify})$ be a key-updatable signature scheme, and $\Psi := (\mathsf{P}, \mathsf{V})$ be a NI zap (or NI-ZoK) arguments. Then, Fig. 2 presents the resulting NIZK argument for $\mathbf{L}$ with PPT algorithms $(\mathsf{K}'_{\mathsf{srs}}, \mathsf{SU}', \mathsf{SV}', \mathsf{P}', \mathsf{V}', \mathsf{Sim}')$, where we later show that it can achieve Sub-ZK and U-SS or U-SE, depending on the input NI zap.

As it can be seen in Fig. 2, every time P' uses Σ to certify the public key of a fresh key pair $(\mathsf{sk}, \mathsf{pk})$ of the Σ_{OT} sOTS scheme. It uses sk_{OT} to sign the proof and achieve non-malleability. By adaptability of the key-updatable signature scheme Σ, the prover can use fresh keys of Σ for each proof. Due to OR construction of $\mathbf{R_{L'}}$, one needs to know the secret key csk to shift such signatures to the valid ones under cpk given in the SRS. On the other side, honest P can use his witness w to generate valid proof, without knowledge of the csk. However, given csk, one can simulate the proofs and pass the verification using the second clause of OR relation. Next, we show how one can exploit NI zaps along with the described construction (in Fig. 2) and build universal and updatable NIZK arguments.

SRS Generation, $(\mathsf{srs}'_0, \Pi'_{\mathsf{srs}_0}) \leftarrow \mathsf{K}'_{\mathsf{srs}}(\mathbf{R_L})$**:** Run the key generator of the key-updatable signature scheme $(\mathsf{csk}_0, \mathsf{cpk}_0, \Pi_{\mathsf{cpk}_0}) \leftarrow \Sigma.\mathsf{KG}(1^\lambda)$; Set $\mathsf{srs}'_0 := \mathsf{cpk}_0$, $\Pi'_{\mathsf{srs}_0} := \Pi_{\mathsf{cpk}_0}$, and $\mathsf{ts}_0 = \mathsf{csk}_0$; Return $(\mathsf{srs}'_0, \Pi'_{\mathsf{srs}_0})$.

SRS Updating, $(\mathsf{srs}'_i, \Pi'_{\mathsf{srs}_i}) \leftarrow \mathsf{SU}'(\mathbf{R_L}, \mathsf{srs}'_{i-1}, \{\Pi'_{\mathsf{srs}_j}\}_{j=0}^{i-1})$**:** Parse $\mathsf{srs}'_{i-1} := \mathsf{cpk}_{i-1}$; Run $(\mathsf{csk}_i, \mathsf{cpk}_i, \Pi_{\mathsf{cpk}_i}) \leftarrow \Sigma.\mathsf{KU}(\mathsf{cpk}_{i-1})$; Set $(\mathsf{srs}'_i \parallel \Pi'_{\mathsf{srs}_i}) := (\mathsf{cpk}_i \parallel \Pi_{\mathsf{cpk}_i})$, where Π'_{srs_i} is the well-formedness proof; Return $(\mathsf{srs}'_i, \Pi'_{\mathsf{srs}_i})$.

SRS Verify, $(\perp/1) \leftarrow \mathsf{SV}'(\mathbf{R_L}, \mathsf{srs}'_i, \{\Pi'_{\mathsf{srs}_j}\}_{j=0}^{i})$**:** Given $\mathsf{srs}'_i := (\perp, \mathsf{cpk}_i)$, and $\{\Pi'_{\mathsf{srs}_j} := \Pi_{\mathsf{cpk}_j}\}_{j=0}^{i}$ act as follows: if $\{\Sigma.\mathsf{KV}(\mathsf{cpk}_{j-1}, \mathsf{cpk}_j, \Pi_{\mathsf{cpk}_j}) = 1\}_{j=1}^{i}$ return 1 (i.e., the srs'_i is well-formed), otherwise $\perp$.

Prover, $(\pi'/\perp) \leftarrow \mathsf{P}'(\mathbf{R_L}, \mathsf{srs}'_i, \mathsf{x}, \mathsf{w})$**:** Parse $\mathsf{srs}'_i := \mathsf{cpk}_i$; Return $\perp$ if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R_L}$; generate $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KG}(1^\lambda)$ and $(\mathsf{sk}_{OT}, \mathsf{pk}_{OT}) \leftarrow \Sigma_{OT}.\mathsf{KG}(1^\lambda)$; generate $\pi \leftarrow \mathsf{P}(\mathcal{R}_\mathbf{L}, \perp, (\mathsf{x}, \mathsf{cpk}_i, \mathsf{pk}), (\mathsf{w}, \perp))$; sign $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}_{OT})$ and $\sigma_{OT} \leftarrow \Sigma_{OT}.\mathsf{Sign}(\mathsf{sk}_{OT}, (\mathsf{x}, \mathsf{pk}, \pi, \sigma))$; and return $\pi' := (\pi, \mathsf{pk}, \sigma, \mathsf{pk}_{OT}, \sigma_{OT})$.

Verifier, $(0/1) \leftarrow \mathsf{V}'(\mathbf{R_L}, \mathsf{srs}'_i, \mathsf{x}, \pi')$**:** Parse $\mathsf{srs}'_i := \mathsf{cpk}_i$ and $\pi' := (\pi, \mathsf{pk}, \sigma, \mathsf{pk}_{OT}, \sigma_{OT})$; return 1 if the following holds and 0 otherwise: $\mathsf{V}(\mathbf{R_L}, \perp, (\mathsf{x}, \mathsf{cpk}_i, \mathsf{pk}), \pi) = 1 \wedge \Sigma.\mathsf{SigVerify}(\mathsf{pk}, \mathsf{pk}_{OT}, \sigma_{OT}) = 1 \wedge \Sigma_{OT}.\mathsf{SigVerify}(\mathsf{pk}_{OT}, (\mathsf{x}, \mathsf{pk}, \pi, \sigma), \sigma_{OT}) = 1$.

Simulator, $(\pi') \leftarrow \mathsf{Sim}'(\mathbf{R_L}, \mathsf{srs}'_i, \mathsf{x}, \mathsf{ts}'_i)$**:** Parse $\mathsf{srs}'_i := \mathsf{cpk}_i$ and $\mathsf{ts}'_i := \mathsf{ts}_i$; generate $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KG}(1^\lambda)$ and $(\mathsf{sk}_{OT}, \mathsf{pk}_{OT}) \leftarrow \Sigma_{OT}.\mathsf{KG}(1^\lambda)$; generate $\pi \leftarrow \mathsf{P}(\mathcal{R}_\mathbf{L}, \perp, (\mathsf{x}, \mathsf{cpk}_i, \mathsf{pk}), (\perp, \mathsf{csk}_i - \mathsf{sk}))$; sign $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}_{OT})$ and $\sigma_{OT} \leftarrow \Sigma_{OT}.\mathsf{Sign}(\mathsf{sk}_{OT}, (\mathsf{x}, \mathsf{pk}, \pi, \sigma))$; and return $\pi' := (\pi, \mathsf{pk}, \sigma, \mathsf{pk}_{OT}, \sigma_{OT})$.

Fig. 2. A construction for building Sub-ZK and U-SE (or U-SS) NIZK arguments from a NI zap (or NI-ZoK) with PPT algorithms (P, V).

Theorem 2. *Let Ψ be a NI-ZoK, which satisfies completeness, (Sub-)KS and (Sub)-WI. Let Σ be a EUF-CMA secure key-updatable and adaptable signature scheme, and Σ_{OT} be a strongly unforgeable one-time signature scheme. Then, the universal and updatable NIZK argument constructed in Fig. 2 satisfies (i) completeness, (ii) Sub-ZK, and (iii) U-SE.*

Proof. The completeness is straight forward from the description of the resulting NIZK argument given in Fig. 2.

Subversion Zero-Knowledge (Sub-ZK). Considering the fact that NI zap arguments do not have an SRS, the SRS of the resulting NIZK argument is just the public key of the key-updatable signature scheme σ. Suppose that there exists a PPT subvertor $\mathcal{A}$ that outputs a srs' and $\Pi_{\mathsf{srs}'}$ such that $\{\mathsf{SV}'(\mathbf{R_L}, \mathsf{srs}'_i, \Pi'_{\mathsf{srs}_j}) = 1\}_{j=0}^{i}$. From the definition of key-updatable signatures, it then implies that there exists a PPT extractor $\mathsf{Ext}_\mathcal{A}$ that given the source code of the adversary and under a proper knowledge assumption (e.g. BDH-KE in our case, Def. 11), it can extract the simulation trapdoor $\mathsf{ts}_i := \mathsf{csk}_i$ associated with the updated SRS srs'_i. Then, given the SRS trapdoor ts_i, we can run the simulator algorithm Sim' described in Fig. 2, and simulate the proofs. This concludes the proof of Sub-ZK.

Updatable Simulation Extractability (U-SE). The proof of U-SE is a modified version of the proof of Theorem 4 in [2], where for each game change based on the ZK property of the zk-SNARK, we now exploit the (Sub-)WI property of the NI-ZoK argument. For the steps relying on the knowledge soundness of zk-SNARK, we rely on the knowledge soundness of NI-ZoK. For the sake of completeness, we present the proof below.

Considering Corollary 1, we prove the theorem in the case that Sub makes only a single update to the SRS after an honest setup. One could also consider the case that Sub generate the SRS, and then we do a single honest update on it. From the definition of a secure key updatable signature, Def. 13, it is possible to use $\mathsf{Ext}_{\mathsf{Sub}}$ and extract the subverter's contribution to the trapdoor when it updates the SRS. Note that in order to collapse all the honest updates into an honest setup, we assume that the trapdoor contributions of setup and update commute. Due to this fact, we use a chain of honest updates, and a single honest setup interchangeably. As we mentioned earlier, NI zaps, do not have an SRS, and we use csk_i of Σ to simulate the proofs.

Relying on the updatable EUF-CMA of Σ, if Sub (or $\mathcal{A}$) return an updated SRS $\mathsf{srs}_i := \mathsf{cpk}_i$ and the corresponding proof Π_i, then (relying on a knowledge assumption, e.g. BDH-KE in our case, given in Def. 11) there exists a PPT extractor $\mathsf{Ext}_{\mathsf{Sub}}$, that given the randomness of Sub as input, outputs $\mathsf{ts}_i := \mathsf{csk}_i$. It is important to note that in the U-SE (given in Def. 7), in addition to accessing the simulation oracle $\mathsf{O}(\mathsf{ts}_i, ...)$ that returns simulated proofs, an adversary $\mathcal{A}$ (or subverter Sub) is allowed to update or generate the initial SRS, which is examined below in experiment $\mathsf{Exp.\ 2}$. Following the Def. 13, we use the subverter Sub for updating the SRS and the adversary $\mathcal{A}$ against SE. However, Sub and $\mathcal{A}$ can communicate and $\mathsf{RND}(\mathsf{Sub}) = \mathsf{RND}(\mathcal{A})$, or can be the same entity. The definition of U-SE and the simulation oracle are presented in Def. 7 and Fig. 2, respectively. Next, we write some consecutive games, starting from the real experiment, and highlight the changes, and show that they all are indistinguishable from each other.

$\mathsf{Exp.\ 0}$. This is the original experiment given in Def. 7 and Fig. 2.

$\mathsf{Exp.\ 1}$. This experiment is the same as $\mathsf{Exp.\ 0}$., but Sim' uses the simulation trapdoor $\mathsf{ts}_i := \mathsf{csk}_i$, which is extracted by $\mathsf{Ext}_{\mathsf{Sub}}$, and generate the simulated proof π as $\pi \leftarrow \mathsf{P}(\mathcal{R}_{\mathbf{L}}, \perp, (\mathsf{x}, \mathsf{cpk}_i, \mathsf{pk}), (\perp, \mathsf{csk}_i - \mathsf{sk}))$ (shown in Fig. 2).

Winning Condition. Let Q be the set of (x, π) pairs, and let T be the set of verification keys generated by $\mathsf{O}(.)$. The experiment returns 1 iff: $(\mathsf{x}, \pi) \notin Q \wedge \mathsf{V}(\mathbf{R_L}, \mathsf{srs}_i, \mathsf{x}, \pi) = 1 \wedge \mathsf{pk}_{OT} \notin T \wedge \mathsf{cpk}_i = \mathsf{pk} \cdot \mu(\mathsf{csk} - \mathsf{sk})$.

$\mathsf{Exp.\ 0.} \rightarrow \mathsf{Exp.\ 1.}$: If the underlying OT signature is strongly unforgeable, and that the underlying NI-ZoK is knowledge sound and WI, then we have $\Pr[\mathsf{Exp.\ 0.}] \leq \Pr[\mathsf{Exp.\ 1.}] + \mathsf{negl}(\lambda)$. The reason is that if $(\mathsf{x}, \pi) \notin Q$ and pk_{OT} has been generated by $\mathsf{O}(.)$, then the $(\mathsf{x}, \pi, \mathsf{pk})$ is a valid message/signature pair. Hence by the unforgeability of the σ_{OT}, we know that the case $(\mathsf{x}, \pi) \notin Q$ and pk_{OT} generated by $\mathsf{O}(.)$, happens with probability $\mathsf{negl}(\lambda)$, which allows us to focus on $\mathsf{pk}_{OT} \notin T$.

The extracted w is unique for all valid witnesses. Further, if some witness is valid for $\mathbf{L}$ and that $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R_L}$, we know it must be the case that due to the WI property of the NI-ZoK and the property of the updating procedure that if

SV' outputs 1, then there exists an extractor that extracts the $\mathsf{ts}_i := \mathsf{csk}_i$ (i.e., the trapdoor extraction for the key-updatable signature scheme implies that it is possible to extract the trapdoor when the adversary generates the $\mathsf{srs}_i := \mathsf{cpk}_i$ itself). Given, $\mathsf{ts}_i := \mathsf{csk}_i$, one can use Sim' (given in Fig. 2) and simulate the proofs in a way that no PPT adversary can distinguish them.

Exp. 2. This experiment is the same as Exp. 1., but the only difference is that Sub generates the srs and then we have an honest update.

Exp. 1. $\rightarrow$ Exp. 2.: From the property of the updating procedure in key-updatable signature schemes, we know that if SV' outputs 1, then there exists an $\mathsf{Ext}_{\mathsf{Sub}}$ that can extract the $\mathsf{ts}_i := \mathsf{csk}_i$ (relying on a knowledge assumption). Considering that and also from the WI property of the NI-ZoK argument we have $\Pr[\mathsf{Exp.\ 0.}] \leq \Pr[\mathsf{Exp.\ 1.}] + \mathsf{negl}(\lambda)$.

Exp. 3. This experiment is the same as Exp. 2., but the $\Delta \leftarrow\!\!\$\ \mathbb{H}$ is replaced in $\mathsf{cpk} = \mu(\Delta) \cdot \mathsf{pk}$, where $\Delta \leftarrow\!\!\$\ \mathbb{H}$ and $\mathsf{srs}_i := (\mathsf{cpk} \cdot \mu(\Delta))$ and $\mathsf{ts}_i := \mathsf{csk}_i$.

Winning Condition. Let Q be the set of (x, π) pairs, and let T be the set of verification keys generated by the $\mathsf{O}(.)$. The experiment returns 1 iff: $(\mathsf{x}, \pi) \notin Q \wedge \mathsf{V}(\mathcal{R}_{\mathbf{L}}, \mathsf{srs}_i, \mathsf{x}, \pi) = 1 \wedge \mathsf{pk}_{OT} \notin T \wedge \mathsf{cpk}_i \cdot \mu(\Delta) = \mathsf{pk} \cdot \mu(\Delta) \cdot \mu(\mathsf{csk}_i - \mathsf{sk})$, where relying on the adaptable and updatable EUF-CMA property of the underlying key-updatable signature scheme Σ, it is $\mathsf{negl}(\lambda)$. □

Theorem 3. *Let Ψ be a NI zap, which satisfies (Sub-)SND and (Sub)-WI. Let Σ be a EUF-CMA secure key-updatable and adaptable signature scheme, and Σ_{OT} be a strongly unforgeable one-time signature scheme. Then, the universal and updatable NIZK argument constructed in Fig. 2 satisfies (i) completeness, (ii) Sub-ZK, and (iii) U-SS.*

Proof. The proofs of completeness and Sub-ZK are the same as in the proof of Theorem 2. The proof of U-SS is analogous to the proof of U-SE in the same theorem, which again can be considered as a minimally modified version of the proof of Theorem 4 in [2], where for each game change based on the ZK property of the zk-SNARK, we now use the (Sub-)WI property of the NI zap, and for the steps relying on the knowledge soundness of the zk-SNARK, we rely on the soundness of NI zap, but only achieve U-SS instead of U-SE. □

Instantiations. Next, we instantiate the construction presented in Fig. 2 and obtain Sub-ZK NIZK arguments (resp. of knowledge) with an updatable SRS, that can satisfy U-SS and U-SE.

Considering Theorem 2, to obtain a Sub-ZK and U-SE NIZK argument of knowledge, we instantiate the NI-ZoK Ψ with the one proposed in [20], and the key-updatable signature scheme Σ with the one proposed in [2], and the strongly unforgeable one-time signature scheme Σ_{OT} with Groth's scheme [25].

Similarly, considering Theorem 3, to build a Sub-ZK and U-SS NIZK argument, we instantiate the NI zap Ψ with the one proposed in [29], and again the key-updatable signature scheme Σ from [2], and the one-time signature scheme Σ_{OT} from [25]. In both cases, the resulting NIZK argument will have an *updatable* SRS containing only two group elements, which are the public key of the key-updatable signature scheme, which can be updated and verified efficiently.

5 Conclusion

In this paper, we first showed that the Sub-ZK NIZK argument proposed by Bellare et al. [7], can also satisfy U-SND, which reduces the trust in the third parties even more. By using our proposed SRS updating algorithm, the verifier can update the SRS one time and then bypass the need for a TTP.

Then, we show that one can expand their idea and construct a general construction based on NI zaps that can be used to build universal and updatable NIZK arguments in a modular way. To this end, we presented a compiler based on the construction of Derler and Slamanig [16], which allows one to use NI zaps and build NIZK arguments with succinct, universal, and updatable SRS, that can achieve Sub-ZK, U-SND, U-KS, U-SS, and U-SE. We instantiated our compiler with some NI zaps [29, 20] and presented some universal and updatable NIZK arguments that all have succinct SRS.

Due to the currently available options, we instantiated our scheme with the DL-based NI zaps, key-updatable signatures, and one-time secure signatures. Building the mentioned primitives from Post-Quantum (PQ) secure assumptions can be an interesting future research question, as they would allow for the building of universal and updatable NIZK arguments from the PQ assumptions.

Acknowledgement. We thank the anonymous reviewers for their valuable comments and suggestions. The authors also thank Chaya Ganesh for shepherding the final version of the paper. This work has been supported in part by the Defense Advanced Research Projects Agency (DARPA) under contract No. HR001120C0085, and by CyberSecurity Research Flanders with reference number VR20192203, and by the Norwegian Research Council. Navid Ghaedi Bardeh has been supported by Norwegian Research Council project number 288545.

References

1. B. Abdolmaleki, K. Baghery, H. Lipmaa, and M. Zajac. A subversion-resistant SNARK. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 3–33. Springer, Heidelberg, Dec. 2017. 2, 5, 6, 8, 10
2. B. Abdolmaleki, S. Ramacher, and D. Slamanig. Lift-and-Shift: Obtaining simulation extractable subversion and updatable SNARKs generically. In *ACM SIGSAC Conference on Computer and Communications Security, CCS 2020, avalable on: http://eprint.iacr.org/2020/062*, 2020. 3, 4, 5, 8, 10, 13, 14, 15, 17
3. B. Abdolmaleki and D. Slamanig. Subversion-resistant quasi-adaptive NIZK and applications to modular zk-snarks. In M. Conti, M. Stevens, and S. Krenn, editors, *Cryptology and Network Security - 20th International Conference, CANS 2021, Vienna, Austria, December 13-15, 2021, Proceedings*, volume 13099 of *Lecture Notes in Computer Science*, pages 492–512. Springer, 2021. 3
4. K. Baghery. Subversion-resistant simulation (knowledge) sound NIZKs. In M. Albrecht, editor, *17th IMA International Conference on Cryptography and Coding*, volume 11929 of *LNCS*, pages 42–63. Springer, Heidelberg, Dec. 2019. 3, 10

5. K. Baghery, Z. Pindado, and C. Ràfols. Simulation extractable versions of groth's zk-SNARK revisited. In S. Krenn, H. Shulman, and S. Vaudenay, editors, *CANS 20*, volume 12579 of *LNCS*, pages 453–461. Springer, Heidelberg, Dec. 2020. 3
6. K. Baghery and M. Sedaghat. Tiramisu: Black-box simulation extractable nizks in the updatable CRS model. In M. Conti, M. Stevens, and S. Krenn, editors, *Cryptology and Network Security - 20th International Conference, CANS 2021, Vienna, Austria, December 13-15, 2021, Proceedings*, volume 13099 of *Lecture Notes in Computer Science*, pages 531–551. Springer, 2021. 3, 13, 14
7. M. Bellare, G. Fuchsbauer, and A. Scafuro. NIZKs with an untrusted CRS: Security in the face of parameter subversion. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 777–804. Springer, Heidelberg, Dec. 2016. 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 18
8. M. Bellare and S. Goldwasser. New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 194–211. Springer, Heidelberg, Aug. 1990. 10
9. E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct non-interactive arguments for a von neumann architecture. Cryptology ePrint Archive, Report 2013/879, 2013. `https://eprint.iacr.org/2013/879`. 2
10. M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 103–112. ACM, 1988. 1, 2
11. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Heidelberg, Aug. 2004. 3, 8
12. M. Campanelli, A. Faonio, D. Fiore, A. Querol, and H. Rodríguez. Lunar: A toolbox for more efficient universal and updatable zkSNARKs and commit-and-prove extensions. In M. Tibouchi and H. Wang, editors, *ASIACRYPT 2021, Part III*, volume 13092 of *LNCS*, pages 3–33. Springer, Heidelberg, Dec. 2021. 3
13. A. Chiesa, Y. Hu, M. Maller, P. Mishra, N. Vesely, and N. P. Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In A. Canteaut and Y. Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 738–768. Springer, Heidelberg, May 2020. 3
14. I. Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 445–456. Springer, Heidelberg, Aug. 1992. 2
15. A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. Robust non-interactive zero knowledge. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 566–598. Springer, Heidelberg, Aug. 2001. 13
16. D. Derler and D. Slamanig. Key-homomorphic signatures and applications to multiparty signatures. Cryptology ePrint Archive, Report 2016/792, 2016. `https://eprint.iacr.org/2016/792`. 9, 13, 14, 18
17. C. Dwork and M. Naor. Zaps and their applications. In *41st FOCS*, pages 283–293. IEEE Computer Society Press, Nov. 2000. 2
18. U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *22nd ACM STOC*, pages 416–426. ACM Press, May 1990. 2
19. G. Fuchsbauer. Subversion-zero-knowledge SNARKs. In M. Abdalla and R. Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 315–347. Springer, Heidelberg, Mar. 2018. 2, 10

20. G. Fuchsbauer and M. Orrù. Non-interactive zaps of knowledge. In B. Preneel and F. Vercauteren, editors, *ACNS 18*, volume 10892 of *LNCS*, pages 44–62. Springer, Heidelberg, July 2018. 2, 3, 4, 5, 13, 17, 18

21. A. Gabizon, Z. J. Williamson, and O. Ciobotaru. PLONK: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019. `https://eprint.iacr.org/2019/953`. 3

22. C. Ganesh, H. Khoshakhlagh, M. Kohlweiss, A. Nitulescu, and M. Zajac. What makes Fiat–Shamir zkSNARKs (updatable srs) simulation extractable? Cryptology ePrint Archive, Report 2021/511, 2021. `https://eprint.iacr.org/2021/511`. 3

23. S. Garg, M. Mahmoody, D. Masny, and I. Meckler. On the round complexity of OT extension. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 545–574. Springer, Heidelberg, Aug. 2018. 11, 12

24. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989. 1

25. J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Heidelberg, Dec. 2006. 14, 17, 21

26. J. Groth. On the size of pairing-based non-interactive arguments. In M. Fischlin and J.-S. Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Heidelberg, May 2016. 2, 5, 21

27. J. Groth, M. Kohlweiss, M. Maller, S. Meiklejohn, and I. Miers. Updatable and universal common reference strings with applications to zk-SNARKs. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 698–728. Springer, Heidelberg, Aug. 2018. 3, 5, 10

28. J. Groth and M. Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 581–612. Springer, Heidelberg, Aug. 2017. 21

29. J. Groth, R. Ostrovsky, and A. Sahai. Non-interactive zaps and new techniques for NIZK. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 97–111. Springer, Heidelberg, Aug. 2006. 2, 4, 5, 10, 13, 17, 18

30. M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In L. Cavallaro, J. Kinder, X. Wang, and J. Katz, editors, *ACM CCS 2019*, pages 2111–2128. ACM Press, Nov. 2019. 3

31. B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013. 2

32. C. Ràfols. Stretching groth-sahai: NIZK proofs of partial satisfiability. In Y. Dodis and J. B. Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 247–276. Springer, Heidelberg, Mar. 2015. 4

33. C. Ràfols and A. Zapico. An algebraic framework for universal and updatable SNARKs. In T. Malkin and C. Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 774–804, Virtual Event, Aug. 2021. Springer, Heidelberg. 3

A Security Requirements of NIZK Arguments

In the standard SRS model, a NIZK argument Ψ_{NIZK} for $\mathcal{R}$ can satisfy the following security definitions. Note that in the security requirements of standard NIZK argument, where the setup phase is trusted.

Definition 18 (Perfect Completeness [26]). *A non-interactive argument Ψ_{NIZK} is* perfectly complete for $\mathcal{R}$*, if for all λ, all $\mathbf{R} \in \mathrm{im}(\mathcal{R}(1^\lambda))$, and $(\mathsf{x}, \mathsf{w}) \in \mathbf{R}$,*

$$\Pr\left[\mathsf{srs} \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}) : \mathsf{V}(\mathbf{R}, \mathsf{srs}, \mathsf{x}, \mathsf{P}(\mathbf{R}, \mathsf{srs}, \mathsf{x}, \mathsf{w})) = 1\right] = 1 \ .$$

Definition 19 (Statistically Zero-Knowledge [26]). *A non-interactive argument Ψ_{NIZK} is* statistically ZK for $\mathcal{R}$*, if for all λ, all $\mathbf{R} \in \mathrm{im}(\mathcal{R}(1^\lambda))$, and for all NUPPT $\mathcal{A}$, $\varepsilon_0^{unb} \approx_\lambda \varepsilon_1^{unb}$, where*

$$\varepsilon_b = \Pr[(\mathsf{srs} \parallel \mathsf{ts}) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}) : \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\mathbf{R}, \mathsf{srs}) = 1] \ .$$

Here, the oracle $\mathsf{O}_0(\mathsf{x}, \mathsf{w})$ returns $\perp$ (reject) if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R}$, and otherwise it returns $\mathsf{P}(\mathbf{R}, \mathsf{srs}_{\mathsf{P}}, \mathsf{x}, \mathsf{w})$. Similarly, $\mathsf{O}_1(\mathsf{x}, \mathsf{w})$ returns $\perp$ (reject) if $(\mathsf{x}, \mathsf{w}) \notin \mathbf{R}$, and otherwise it returns $\mathsf{Sim}(\mathbf{R}, \mathsf{srs}, \mathsf{x}, \mathsf{ts})$. Ψ_{NIZK} is perfect ZK for $\mathcal{R}$ *if one requires that $\varepsilon_0 = \varepsilon_1$.*

Definition 20 (Soundness). *A non-interactive argument Ψ_{NIZK} is computationally (adaptively)* sound for $\mathcal{R}$*, if for every NUPPT $\mathcal{A}$, for all λ,*

$$\Pr\left[\begin{array}{l}\mathbf{R} \leftarrow \mathcal{R}(1^\lambda), (\mathsf{srs} \parallel \mathsf{ts}) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}), \\ (\mathsf{x}, \pi) \leftarrow \mathcal{A}(\mathbf{R}, \mathsf{srs}) : \mathsf{x} \notin \mathbf{L} \wedge \mathsf{V}(\mathbf{R}, \mathsf{srs}_{\mathsf{V}}, \mathsf{x}, \pi) = 1\end{array}\right] \approx_\lambda 0 \ .$$

Next, we recall some stronger notions of NIZK arguments that usually are needed in the case one requires a stronger security guarantees.

Definition 21 (Simulation Soundness [25]). *A non-interactive argument Ψ_{NIZK} is* simulation sound for $\mathcal{R}$ *if for all NUPPT $\mathcal{A}$, and all λ,*

$$\Pr\left[\begin{array}{l}\mathbf{R} \leftarrow \mathcal{R}(1^\lambda), (\mathsf{srs} \parallel \mathsf{ts}) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}), (\mathsf{x}, \pi) \leftarrow \mathcal{A}^{\mathsf{O}(\cdot)}(\mathbf{R}, \mathsf{srs}) : \\ (\mathsf{x}, \pi) \notin Q \wedge \mathsf{x} \notin \mathbf{L} \wedge \mathsf{V}(\mathbf{R}, \mathsf{srs}_{\mathsf{V}}, \mathsf{x}, \pi) = 1\end{array}\right] \approx_\lambda 0 \ .$$

Here, Q is the set of simulated statement-proof pairs generated by adversary's queries to O, that returns simulated proofs.

Definition 22 (Non-Black-Box Simulation Extractability [28]). *A non-interactive argument Ψ_{NIZK} is* non-black-box simulation-extractable for $\mathcal{R}$*, if for any NUPPT $\mathcal{A}$, there exists a NUPPT extractor $\mathsf{Ext}_{\mathcal{A}}$ s.t. for all λ,*

$$\Pr\left[\begin{array}{l}\mathbf{R} \leftarrow \mathcal{R}(1^\lambda), (\mathsf{srs} \parallel \mathsf{ts}) \leftarrow \mathsf{K}_{\mathsf{srs}}(\mathbf{R}), \\ r \leftarrow\!\!\$\ \mathsf{RND}(\mathcal{A}), ((\mathsf{x}, \pi) \parallel \mathsf{w}) \leftarrow (\mathcal{A}^{\mathsf{O}(\cdot)} \parallel \mathsf{Ext}_{\mathcal{A}})(\mathbf{R}, \mathsf{srs}; r) : \\ (\mathsf{x}, \pi) \notin Q \wedge (\mathsf{x}, \mathsf{w}) \notin \mathbf{R} \wedge \mathsf{V}(\mathbf{R}, \mathsf{srs}_{\mathsf{V}}, \mathsf{x}, \pi) = 1\end{array}\right] \approx_\lambda 0 \ .$$

Here, Q is the set of simulated statement-proof pairs generated by adversary's queries to O that returns simulated proofs. One may notice that *non-black-box simulation extractability* implies *simulation soundness* (given in Def. 21) [25].