Zero-Knowledge Proxy Re-Identification Revisited

Xavier Bultel and Pascal Lafourcade

LIMOS, Université d'Auvergne

Abstract. *Zero-knowledge proxy re-identification* (ZK-PRI) has been introduced by Blaze *et al.* in 1998 together with two other well known primitives of re-cryptography, namely *proxy re-encryption* (PRE) and *proxy re-signature* (PRS). A ZK-PRI allows a proxy to transform an identification protocol for Alice into an identification protocol for Bob using a *re-proof* key. PRE and PRS have been largely studied in the last decade, but surprisingly, no results about ZK-PRI have been published since the pioneer paper of Blaze *et al.*. We first show the insecurity of this scheme: just by observing the communications Alice can deduce Bob's secret key. Then we give (i) definitions of the different families of ZK-PRI (bidirectional/unidirectional and interactive/non-interactive) (ii) a formal security model for these primitives and (iii) a concrete construction for each family. Moreover, we show that ZK-PRI can be used to manage the acces policy to several services that require a public key authentication.

1 Introduction

Proxy Re-Encryption (PRE), Proxy Re-Signature (PRS) and Zero-Knowledge Proxy Re-Identification (ZK-PRI) schemes are three cryptographic primitives that allow *a delegator* (Bob) to delegate his decryption, his signature or his identification right to *a delegate* (Alice). The proxy does not just help Alice to decrypt, sign or identify instead of Bob without Bob's secret key: in PRE, the proxy transforms (*i.e.*, he *re-encrypts*) a message encrypted for Bob into another one encrypted for Alice using a *re-encryption* key. In PRS, the proxy transforms (*i.e.*, he *re-signs*) a message signed by Alice into a message signed with Bob's secret key using a *re-signature* key. In ZK-PRI, the proxy transforms (*i.e.*, he *re-identifies*) an identification protocol for Alice into another one for Bob using a *re-proof* key. A naïve solution is to give the secret of Bob to the proxy, however it requires a fully trusted proxy. In proxy re-cryptography, we usually consider a *semi-trusted* proxy that does not collude with Alice and that cannot act alone as Alice or Bob.

PRE, PRS and ZK-PRI have been introduced by Blaze *et al.* in [5]. In [15] Ivan *et al.* revisit the proxy re-cryptography by introducing the notion of bidirectional and unidirectional proxies: in unidirectional proxies, the delegator Bob is able to compute the re-key alone, *i.e.* using only his secret key and the public key of Alice. Authors give an unidirectional scheme for both PRE and PRS but they do not study the case of unidirectionality in ZK-PRI. Unidirectional PRE and PRS have led to numerous publications [1, 2, 5, 7, 14, 15, 19–21, 24]. However, surprisingly, ZK-PRI has not received the same attention. To the best of our knowledge, the scheme given in [5] is the only one of the literature. This scheme works as follows: Let g be a generator of a group of prime order p, Alice has her secret a and Bob has his secret b. Alice interacts with a proxy who knows a re-proof key $\mathsf{rk} = a/b$, and the proxy interacts with a verifier who knows the public key of Bob g^b. Alice chooses a random element $k \in \mathbb{Z}_p^*$ and sends $K = g^k$ to the proxy, who forwards it to the verifier. The verifier chooses a challenge $c \in \{0, 1\}$ and sends it to the proxy who forwards it to Alice. If $c = 1$ then Alice sends $z = k/a$ to the proxy, who computes and sends $z' = z \cdot \mathsf{rk} = k/b$ to the verifier. Else, Alice sends $z = k$ and the proxy sends $z' = z$ to the verifier. If $c = 1$ the verifier checks that $K = (g^b)^{z'}$. Else, it checks $K = g^{z'}$. The protocol is repeated several times. This scheme has the following security flaws:

- if Alice and the verifier collude, then they can recover the secret key of Bob as follows: the verifier sends the challenge $c = 1$ and sends z' to Alice, then she can compute the Bob's secret as follows $k/z' = b$.
- if Alice observes the communication, then she can also recover the secret key of Bob as follows: if the verifier sends the challenge $c = 1$, then by observing the communication Alice learns $z' = k/b$. Then Alice knowing k can easily recover b the secret key of Bob by computing $k/z' = b$.

Contributions: We revisit the concept of ZK-PRI by formally defining bidirectional, unidirectional, interactive, and non-interactive ZK-PRI schemes. Our second contribution is four secure ZK-PRI schemes:

- $\mathsf{B.RE\Pi}$, a bidirectional and interactive ZK-PRI that does not have the weakness of [5]. In this scheme, the delagator and the delegate only use Schnorr's identification protocol. Thus, it can be easily deployed in existing systems that use authentication with this protocol. This scheme is *multi-hop* (*i.e.* it is possible to successively transform an identification transcript several times using several different re-proof keys).
- $\mathsf{U.RE\Pi}$, an unidirectional and interactive ZK-PRI scheme.
- $\mathsf{NiB.RE\Pi}$, a bidirectional and non-interactive ZK-PRI scheme.
- $\mathsf{NiU.RE\Pi}$, an unidirectional and non-interactive ZK-PRI scheme.

Our security model is stronger than the initial model introduced in [5], since it allows the delegate to collude with the verifier. In Appentix A we sum up and compare the properties and the efficiency of our four schemes. Using our two non-interactive schemes $\mathsf{NiB.RE\Pi}$ and $\mathsf{NiU.RE\Pi}$, we naturally propose two proxy re-signature schemes respectively bidirectional and unidirectional.

Applications: ZK-PRI scheme can be used to provide a practical mechanism of authentication delegation. For instance, Bob manipulates sensitive data in his work and he must often authenticate himself using his secret key. During his holidays, he has to delegate his access rights to Alice. However, Bob does not want to reveal his secret to Alice, and he wants to be able to cancel the delegation when he wants. To solve this problem, he uses a ZK-PRI which allows Alice to authenticate on behalf of Bob using his secret key during Bob's holidays. Thus, Alice, using the same algorithm and her secret, is able to authenticate herself under Bob's identity as long as she has access to the proxy. Until now the only solution for solving this problem was to use the bidirectional scheme of Blaze *et al.* [5], which is insecure.

Another application of ZK-PRI is access control management. For instance a company has subscribed to several services like email, cloud storage or software. Each time, an authentication mechanism is required for all these services and it uses always the same login and password. Using a ZK-PRI, it is possible to manage which person of the company has access to which services using their own login and password: depending on his access rights, the proxy accepts or not to help a user to authenticate for a given service. The main advantage of this solution is that the company can manage its access policy alone, without giving new authentication keys to the manager of the services, and without changing the secret/public keys of its employees. Each user just needs to know his own authentication material. It allows the delegator to not distribute several keys per application and per user, similarly as in the proxy-based distributed encrypted storage by Atenise *et al.* in [1].

Related work: In the late 80's, Okamoto and Ohta introduce the notion of *divertible zero-knowledge interactive proofs* [22]. This notion is pretty close to ZK-PRI: a ZKP between a prover P and a verifier V is *divertible* when a third party W can impersonate V (resp. P) during the protocol such that P (resp. V) cannot distinguish if he interacts with V (resp. P) or W. Then W just *randomizes* but does not transform a proof of a secret to the proof of another one. Thus, W has no *re-key* and cannot be used to delegate the proving ability. ZK-PRI can be viewed as an extension of divertible zero-knowledge proofs.

To achieve similar properties as ZK-PRI, we can use a *two-party computation* [17] (TPC): the delegator shares his secret over the delegate and the proxy. Using TPC they can compute together values that allow them to identify as the delegator. However, this generic solution is not efficient, and the delegate does not use the same procedure to identify itself or to identify as the delegator.

In distributed and threshold zero-knowledge proofs [18] a secret is shared into n shares, and these shares are distributed to n parties such that a threshold number t of shares allows someone to recover the secret. Then t parties are able to prove the knowledge of the secret together. This primitive can be used for distributed zero-knowledge identification (DZKI). ZK-PRI can be viewed as a particular case of DZKI since the secret of the delegator and the re-proof key allow to compute the delegator's secret key, and the delegate and the proxy interact together to identify as the delegator. However, these schemes do not really *transform* an identification transcript into another one for two different public keys. Moreover, unidirectional proxy requires that the re-proof key is computed from the public key of the delegate and the secret of the delegator, and the delegate and the delegator must have a way to identify himself alone. The delegate (resp. verifier) uses the same protocol when he interacts directly with the verifier (resp. delegator) and when he interacts with the proxy. Actually, the differences between DZKI and ZK-PRI are analogous to the differences between proxy re-encryptions and threshold encryptions, and between proxy re-signatures and threshold signatures.

To the best of our knowledge, there exists neither formal definition nor concrete scheme of unidirectional ZK-PRI and non-interactive ZK-PRI. Finally, note that ZK-PRI should not be confused with *proxy zero-knowledge proof* defined in [16]. In this primitive, the proxy helps Alice to perform a proof of knowledge of the secret of the delegator named Bob in order to identify as Bob, but the proxy does not transform the proof of Alice's secret knowledge into a proof of Bob's secret knowledge. Indeed, in ZK-PRI, Alice must be able to identify herself to use the proxy, and she uses the same identification protocol to identify herself (interacting with the verifier) and to identify as Bob (interacting with the proxy). Again, it is the same difference as between proxy encryption and PRE, and between proxy signature and PRS. Moreover, ZK-PRI should not be confused with *homomorphic proxy re-authenticators* [8]: this primitive is a kind of proxy re-signature that offer some additional verifiability guarantees, thus it is not a kind of ZK-PRI, and it does not focus on zero-knowledge properties.

2 Background

Notations: We denote by $r \xleftarrow{\$} S$ the random draw of r into the uniform distribution on S. Let X be a *Probabilistic Polynomial-Time* algorithm (PPT), we denote by $z \leftarrow X(x;r)$ the result z of the execution of X on input x and on the random tape r. When it is clear from the context, we omit the parameter r and simply use $X(x)$. A *protocol* involves at least two entities that are modeled by PPT algorithms. We denote by $\mathsf{P}\langle X(x); Y(y)\rangle$ the execution of the protocol P between X and Y using respectively inputs x and y. Moreover, $\mathtt{out}_X(\mathsf{P}\langle X(x); Y(y)\rangle)$ returns the output of the entity X at the end of the execution of the protocol P. We also denote by $\mathtt{view}_X(\mathsf{P}\langle X(x); Y(y)\rangle)$ all the values sent and received by X throughout the execution of the protocol P. We note that `out` and `view` can be used by several entities, for example, the function $\mathtt{out}_{X,Z}(\mathsf{P}\langle X(x); Y(y); Z(z)\rangle)$ returns the couple (o_X, o_Z) such that o_X (resp.

o_Z) is the output of the entity X (resp. Z) at the end of the protocol P execution. By convention, we use the symbol $*$ to denote a dishonest user in a protocol: by example, $\mathsf{P}\langle X(x); Y^*(y)\rangle$ denotes that X honestly runs the protocol but Y^* does not. If several dishonest entities *collude* then each of theses entities has access to all information known by all other dishonest entities. When it is not precised, we assume that dishonest entities do not collude.

Definition 1 (DL **[6]).** *Let $\mathbb{G}$ be a multiplicative group of prime order p, and g be a group element. Given an instance $h \in \mathbb{G}$ where $x \xleftarrow{\$} \mathbb{Z}_p^*$ and $h = g^x$, the* discrete logarithm problem in $(\mathbb{G}, p, g)$ *(*DL*) is to compute x. The* DL assumption *states that there exists no PPT algorithm that solves* DL *with non-negligible probability.*

Definition 2 (FAPI2 **[11]).** *Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be three groups of prime order p, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a non-degenerate bilinear pairing and $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ be two group elements. Given an instance $h \in \mathbb{G}_T$ where $X \xleftarrow{\$} \mathbb{G}_2$ and $e(g_1, X) = h$, the* fixed argument pairing inversion 2 problem *(*FAPI2*) in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$ is to compute X. The* FAPI2 assumption *states that there exists no PPT algorithm that solves* FAPI2 *with non-negligible probability.*

Definition 3 (BDLV)**.** *Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be 3 groups of prime order p, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a type 2 non-degenerate bilinear pairing and $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ be two generators of $\mathbb{G}_1$ and $\mathbb{G}_2$. Given an instance (h_1, h_2) where $x \xleftarrow{\$} \mathbb{Z}_p^*$, $h_1 = g_1^x$ and $h_2 = g_2^{1/x}$, the* bilinear discrete logarithm variant *(*BDLV*) problem in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$ is to compute x. In appendix B, we prove that* BDLV *is hard to solve under under the discrete logarithm hypothesis.*

Definition 4 (ZKI [13]). *An* Identification scheme *is a tuple of three algorithms* (Set, Gen, Proof) *where* Set(λ) *is an algorithm that returns a setup value $\mathscr{S}$,* Gen($\mathscr{S}$) *is an algorithm that returns a pair of public/private keys* (pk, sk)*, and* Prove *is a protocol between the* prover P(sk) *and the* verifier V(pk) *where V outputs a bit b. Such a scheme is a* Zero-Knowledge Identification *scheme (ZKI) if it satisfies the three following properties for any $\mathscr{S} \leftarrow$* Set(λ) *and* (pk, sk) $\leftarrow$ Gen($\mathscr{S}$)*:*

Completeness: *If P knows* sk*, then he is able to convince V, i.e. V outputs 1:*
$\Pr\left[b \leftarrow \mathtt{out}_V(\mathsf{Proof}\langle P(\mathsf{sk}); V(\mathsf{pk})\rangle) : b = 1\right] = 1$

Soundness: *If a dishonest prover P^* does not know* sk*, then he is not able to convince V except with negligible probability:*
$\Pr\left[b \leftarrow \mathtt{out}_V(\mathsf{Proof}\langle P^*(\mathsf{pk}); V(\mathsf{pk})\rangle) : b = 1\right] \leq \epsilon(\lambda)$ *where ϵ is a negligible function.*

Zero-knowledge: *A dishonest verifier V^* learns* nothing *about* sk *except* pk *during the protocol, i.e. there exists a PPT algorithm* `Sim`*, called the* simulator*, such that for any bit-string α:*

$\mathsf{Pr}\left[\alpha_* \leftarrow \mathtt{view}_{V^*}(\mathsf{Proof}\langle P(\mathsf{sk}); V^*(\mathsf{pk})\rangle) : \alpha = \alpha_*\right]$
$= \mathsf{Pr}\left[\alpha_* \leftarrow \mathit{Sim}(\mathsf{pk}) : \alpha = \alpha_*\right]$

Honest-verifier ZKI *(HZKI) is a weaker notion of ZKI which is restricted to case where the verifier is* honest*, i.e. V correctly runs the protocol.*

Definition 5 (NIZKI). *A* non-interactive identification *scheme is a tuple of algorithms* $(\mathsf{Set}, \mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ *such that* Set *and* Gen *are defined as in the interactive case,* $\mathsf{Prove}(\mathsf{pk}, \mathsf{sk})$ *outputs a proof* π *and* $\mathsf{Verify}(\mathsf{pk}, \pi)$ *outputs a bit b. A* non-interactive zero-knowledge identification *scheme (NIZKI) verifies the following properties for any* $\mathscr{S} \leftarrow \mathsf{Set}(\lambda)$ *and* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathscr{S})$*:*

Completeness: $\mathsf{Verify}(\mathsf{pk}, \mathsf{Prove}(\mathsf{pk}, \mathsf{sk})) = 1$

Soundness: *For any PPT adversary $\mathcal{A}$:*
$\mathsf{Pr}\left[\pi \leftarrow \mathcal{A}^{\mathsf{Prove}(\mathsf{pk},\mathsf{sk})}(\mathsf{pk}); b \leftarrow \mathsf{Verify}(\mathsf{pk}, \pi) : b = 1\right] \leq \epsilon(\lambda)$ *where ϵ is a negligible function and π has not be generated by the oracle* $\mathsf{Prove}(\mathsf{pk}, \mathsf{sk})$.

Zero-knowledge: *A proof π leaks* nothing *about* sk *except* pk*, i.e. there exists a PPT simulator* $\mathtt{Sim}$ *such that for any bit-string α:*
$\mathsf{Pr}\left[\alpha_* \leftarrow \mathsf{Prove}(\mathsf{pk}, \mathsf{sk}) : \alpha = \alpha_*\right] = \mathsf{Pr}\left[\alpha_* \leftarrow \mathit{Sim}(\mathsf{pk}) : \alpha = \alpha^*\right]$

We consider ZKIs based on DL and related problems. The Schnorr's protocol in Fig. 1 is an honest verifier ZKI based on DL, *i.e.* $\mathsf{pk} = g^{\mathsf{sk}}$ where g is a generator of a prime order group. It is known to be honest-verifier zero-knowledge but not zero-knowledge [23]. A simple method to transform it in a zero-knowledge protocol is to pick the challenge c in $\{0, 1\}$ instead of to pick it in $\mathbb{Z}_p^*$. However, it is necessary to repeat this protocol λ times for a chosen security parameter λ.

As Schnorr's protocol, several ZKIs have three exchanges between the prover and the verifier: a commitment, a challenge, and a response. Such protocols are called *sigma protocols*. To transform a sigma protocol into a NIZKI using the Fiat-Shamir heuristic [9], it suffices to use the digest of a hash function on the commitment as challenge. The hash function allows us to generate a challenge that cannot be known by the prover before his commitment. For example, using the hash function H, this transformation on Schnorr's protocol gives the following NIZKI algorithms:

$$\begin{array}{lcl}
\textbf{Prover } P & & \textbf{Verifier } V \\
\mathsf{sk} & & \mathsf{pk} = g^{\mathsf{sk}} \\
\hline
r \overset{\$}{\leftarrow} \mathbb{Z}_p^* & & \\
R = g^r & \xrightarrow{R} & c \overset{\$}{\leftarrow} \mathbb{Z}_p^* \\
\alpha = r + \mathsf{sk} \cdot c & \xleftarrow{c} & \\
 & \xrightarrow{\alpha} & \text{Check that:} \\
 & & g^{\alpha} \overset{?}{=} R \cdot \mathsf{pk}^{c}
\end{array}$$

Fig. 1. Schnorr Protocol [23], denoted Π_1.

$\mathsf{Prove}(\mathsf{pk}, \mathsf{sk})$: pick $r \overset{\$}{\leftarrow} \mathbb{Z}_p^*$, compute $R = g^r$, $c = H(R)$ and $\alpha = r + \mathsf{sk} \cdot c$. Output $\pi = (R, \alpha)$.

$\mathsf{Verify}(\mathsf{pk}, \pi)$: Compute $c = H(R)$. If $g^{\alpha} \overset{?}{=} R \cdot \mathsf{pk}^c$ then output 1, else 0.

Note that this version of the Fiat-Shamir transformation is not sound against a malicious prover who can select his problem instance adaptively [4]. However, the security of this transformation is sufficient for our identification protocols.

3 Definitions

A ZK-PRI is composed of two ZKI schemes and a protocol RProof. This protocol allows the proxy to transform a transcript of the first ZKI protocol into a transcript of the second one for two different public keys. The proxy has a re-proof key given by the algorithm RGen. We classify ZK-PRI as follows:
Interactive/Non-interactive: a ZK-PRI can be *interactive* or *non-interactive* according to the interaction between the prover and the verifier of the ZKPs.
Bidirectional/Unidirectional: a ZK-PRI is *bidirectional* if the re-proof key that delegates Bob's rights to Alice is used by the proxy to delegate Alice's rights to Bob, and is *unidirectional* if the delegator generates the re-proof key alone.

We extend the zero-knowledge property for our primitive: informally, a PRI is zero-knowledge when a collusion between the proxy and the verifier learns nothing about the secret key of Alice and a collusion between Alice and the verifier learns nothing about the re-proof key. Thus the scheme of Blaze *et al.* [5] is not secure in our model since a collusion between Alice and the verifier learns the secret key of Bob and can use it to deduce the re-proof key. We also extend the soundness for ZK-PRI: in addition to the soundness of the two ZKI schemes, it must be hard for the proxy to identify alone as Alice or Bob, *i.e.* using only the re-proof key and their public keys.

Definition 6 (ZK-PRI). *A ZK-PRI is a tuple of algorithms/protocols* $(\mathsf{Set}, \mathsf{Gen}_1, \mathsf{Gen}_2, \mathsf{Proof}_1, \mathsf{Proof}_2, \mathsf{RGen}, \mathsf{RProof})$ *such that for* $i \in \{1, 2\}$*:*
$\mathsf{Set}(\lambda)$: *It returns a setup* $\mathscr{S}$*.*
$\mathsf{Gen}_i(\mathscr{S})$: *It returns a a pair of public/private keys* $(\mathsf{pk}_i, \mathsf{sk}_i)$*. Optionally, it returns an intermediate secret* w_i*.*
Proof_i: *It is a protocol between the* prover $P(\mathsf{sk}_i)$ *and the* verifier $V(\mathsf{pk}_i)$ *where* V *outputs a bit* b*.*
$\mathsf{RGen}(\mathsf{sk}_1, w_1, \mathsf{sk}_2, w_2)$: *It returns a re-proof key* rk.
RProof: *let a delegate* $A(\mathsf{sk}_1)$*, a proxy* $P(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2)$ *and a verifier* $V(\mathsf{pk}_2)$ *be three entities,* RProof *is a protocol such that* A *runs* Proof_1 *with* P *and* P *runs* Proof_2 *with* V*. Note that* A *and* V *never interact during the protocol.*
We denote by I_i *the ZKI scheme* $(\mathsf{Set}, \mathsf{Gen}_i, \mathsf{Proof}_i)$*. A ZK-PRI satisfies the following properties for any* $\mathscr{S} \leftarrow \mathsf{Gen}(\lambda)$*,* $(\mathsf{pk}_i, \mathsf{sk}_i, w_i) \leftarrow \mathsf{Gen}_i(\mathscr{S})$ *and* $rk \leftarrow \mathsf{RGen}(\mathsf{sk}_1, w_1, \mathsf{sk}_2, w_2)$*:*
Completeness: (i) I_1 *and* I_2 *are complete and* (ii) *For honest* A*,* P *and* V*:*

$$\Pr\left[b \leftarrow \mathtt{out}_V(\mathsf{RProof}\langle A(s_1); P(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2); V(\mathsf{pk}_2)\rangle) : b = 1\right] = 1$$

Soundness: (i) *I_1 and I_2 are sound and* (ii) *For any dishonest proxy P^* and $i \in \{1,2\}$:*
$\mathsf{Pr}\left[b \leftarrow \mathtt{out}_V(\mathsf{Proof}_i\langle P^*(\mathsf{rk},\mathsf{pk}_1,\mathsf{pk}_2); V(\mathsf{pk}_i)\rangle) : b = 1\right] \leq \epsilon(\lambda)$ *where ϵ is a negligible function.*

Zero knowledge: (i) *I_1 and I_2 are zero-knowledge.*
(ii) *For any dishonest verifier V^* and any dishonest proxy P^*, there exists a PPT simulator $\mathtt{Sim}_1$ such that for any α:*
$\mathsf{Pr}[\alpha_* \leftarrow \mathtt{view}_{P^*,V^*}(\mathsf{RProof}\langle A(\mathsf{sk}_1); P^*(\mathsf{rk},\mathsf{pk}_1,\mathsf{pk}_2); V^*(\mathsf{rk},\mathsf{pk}_1,\mathsf{pk}_2)\rangle) : \alpha = \alpha_*] = \mathsf{Pr}[\alpha_* \leftarrow \mathtt{Sim}_1(\mathsf{rk},\mathsf{pk}_1,\mathsf{pk}_2) : \alpha = \alpha_*]$
(iii) *For any verifier V^* and any dishonest delegate A^*, there exists a PPT simulator $\mathtt{Sim}_2$ such that for any α:*
$\mathsf{Pr}[\alpha_* \leftarrow \mathtt{view}_{A^*,V^*}(\mathsf{RProof}\langle A^*(\mathsf{sk}_1,\mathsf{pk}_1,\mathsf{pk}_2); P(\mathsf{rk},\mathsf{pk}_1,\mathsf{pk}_2);$
$V^*(\mathsf{sk}_1,\mathsf{pk}_1,\mathsf{pk}_2)\rangle) : \alpha = \alpha_*] = \mathsf{Pr}[\alpha_* \leftarrow \mathtt{Sim}_2(\mathsf{sk}_1,\mathsf{pk}_1,\mathsf{pk}_2) : \alpha = \alpha_*]$

We say that a ZK-PRI is bidirectional *if there exists a polynomial time algorithm that allows to compute* $\mathsf{rk}' = \mathsf{RGen}(\mathsf{sk}_2, w_2, \mathsf{sk}_1, w_1)$ *from* rk. *We say that a ZK-PRI is* unidirectional *when the delegator is able to compute the re-proof key alone using his secret key and the public key of the delegate. In this case, we can replace the* RGen *algorithm by:*
$\mathsf{RGen}(\mathsf{pk}_1, \mathsf{sk}_2, w_2)$: *It returns the re-proof key* rk.
Finally honest-ZK-PRI *is a weaker notion of ZK-PRI where all entities correctly run the protocols for the zero-knowledge property.*

Definition 7 (NIZK-PRI). *A NIZK-PRI is a tuple of algorithms* $(\mathsf{Set}, \mathsf{Gen}_1, \mathsf{Gen}_2, \mathsf{Prove}_1, \mathsf{Prove}_2, \mathsf{Verify}_1, \mathsf{Verify}_2, \mathsf{RGen}, \mathsf{RProof})$ *such that* Set, Gen_1, Gen_2 *and* RGen *are defined as in the interactive case and for $i \in \{1,2\}$:*
$\mathsf{Prove}_i(\mathsf{pk}_i, \mathsf{sk}_i)$**:** *It returns a proof π_i.*
$\mathsf{Verify}_i(\mathsf{pk}_i, \pi_i)$**:** *It returns a bit b.*
$\mathsf{RProve}(\mathsf{rk}, \pi_1)$: *It returns a proof π_2 (compatible with* Verify_2*).*
We denote by I_i the NIZKI scheme $(\mathsf{Set}, \mathsf{Gen}_i, \mathsf{Prove}_i, \mathsf{Verify}_i)$. *A NIZK-PRI satisfies the following properties for any $\mathscr{S} \leftarrow \mathsf{Gen}(\lambda)$, $(\mathsf{pk}_i, \mathsf{sk}_i, w_i) \leftarrow \mathsf{Gen}_i(\mathscr{S})$ and $rk \leftarrow \mathsf{RGen}(\mathsf{sk}_1, w_1, \mathsf{sk}_2, w_2)$:*

Completeness: (i) *I_1 and I_2 are complete.*
(ii) $\mathsf{Verify}_2(\mathsf{pk}_2, \mathsf{RProve}(\mathsf{rk}, \mathsf{Prove}_1(\mathsf{pk}_1, \mathsf{sk}_1))) = 1$

Soundness: (i) *I_1 and I_2 are sound* (ii) *for any PPT adversary $\mathcal{A}$, $\forall i \in \{1,2\}$:*
$\mathsf{Pr}\left[\pi \leftarrow \mathcal{A}^{\mathsf{Prove}_i(\mathsf{pk}_i,\mathsf{sk}_i)}(\mathsf{rk},\mathsf{pk}_1,\mathsf{pk}_2); b \leftarrow \mathsf{Verify}_i(\mathsf{pk}_i, \pi) : b = 1\right] \leq \epsilon(\lambda)$ *where ϵ is a negligible function and π was not generated by the oracle* $\mathsf{Prove}_i(\mathsf{pk}_i, \mathsf{sk}_i)$.

Zero knowledge: (i) *I_1 and I_2 are zero-knowledge.*
(ii) *There exists a PPT simulator $\mathtt{Sim}_1$ such that for any bitstring α:*
$$\mathsf{Pr}\left[\alpha_* \leftarrow \mathsf{Prove}_1(\mathsf{pk}_1, \mathsf{sk}_1) : \alpha = \alpha_*\right] = \mathsf{Pr}\left[\alpha_* \leftarrow \mathtt{Sim}_1(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2) : \alpha = \alpha_*\right]$$

$$\begin{array}{lclcl}
\textbf{Prover } A & & \textbf{Proxy } P & & \textbf{Verifier } V \\
\mathsf{sk}_1 & & \mathsf{rk} = \mathsf{sk}_2/\mathsf{sk}_1; \mathsf{pk}_1 = g^{\mathsf{sk}_1}; & & \mathsf{pk}_2 = g^{\mathsf{sk}_2} \\
 & & \mathsf{pk}_2 = g^{\mathsf{sk}_2} & & \\
\hline
r \overset{\$}{\leftarrow} \mathbb{Z}_p^*; R = g^r & \xrightarrow{R} & s \overset{\$}{\leftarrow} \mathbb{Z}_p^*; S = R^{\mathsf{rk}} \cdot g^s & \xrightarrow{S} & c \overset{\$}{\leftarrow} \mathbb{Z}_p^* \\
\alpha = r + \mathsf{sk}_1 \cdot c & \xleftarrow{c} & & \xleftarrow{c} & \\
 & \xrightarrow{\alpha} & \text{If } g^\alpha \overset{?}{=} R \cdot \mathsf{pk}_1^c & & \\
 & & \text{Then } \beta = s + \mathsf{rk} \cdot \alpha & \xrightarrow{\beta} & \text{Check that:} \\
 & & \text{Else abort} & & g^\beta \overset{?}{=} S \cdot \mathsf{pk}_2^c
\end{array}$$

Fig. 2. Protocol $\Pi_{1\to 1}$ used for $\mathsf{B.RE\Pi}$.

(iii)*For any PPT adversary $\mathcal{A}$, there exists a PPT simulator $\mathtt{Sim}_2$ such that for any bitstring α:*

$\mathsf{Pr}\left[\pi_1 \leftarrow \mathcal{A}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{pk}_2); \alpha_* \leftarrow \mathsf{RProve}(\mathsf{rk}, \pi_1) : \alpha = \alpha_*\right]$
$= \mathsf{Pr}\left[\alpha_* \leftarrow \mathtt{Sim}_2(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{pk}_2) : \alpha = \alpha_*\right]$

The definition of unidirectional NIZK-PRI is similar as the interactive case.

In Appendix C, we give a stronger soundness definition for both ZKI and ZK-PRI that we call *strong soundness*. It is closely related to the soundness property of *proofs of knowledge* [13]: informally, a scheme is *strongly sound* when there exists a PPT *extractor* such that for any prover P that has a non-negligible probability to identify, the extractor has a non-negligible probability to guess the secret key sk using the view of P during the Proof protocol.

4 Proxy re-ZKP constructions

We describe the four following ZK-PRI: $\mathsf{B.RE\Pi}$ a multi-hop bidirectional scheme, $\mathsf{U.RE\Pi}$ an unidirectional scheme, $\mathsf{NiB.RE\Pi}$ a bidirectional scheme and $\mathsf{NiU.RE\Pi}$ an unidirectional scheme. The proofs of the theorems of this section are given respectively in Appendix D, E, F and G.

4.1 Bidirectional ZK-PRI ($\mathsf{B.RE\Pi}$)

Building blocks: For the scheme $\mathsf{B.RE\Pi}$, we use Schnorr's protocol (Π_1) as first and second ZKI protocol.

Construction: Let $(\mathbb{G}, p, g)$ be a prime order group. We designate Alice as delegate and Bob as delegator. Alice (resp. Bob) knows sk_1 such that $\mathsf{pk}_1 = g^{\mathsf{sk}_1}$ (resp. sk_2 such that $\mathsf{pk}_2 = g^{\mathsf{sk}_2}$), and the re-key is $\mathsf{rk} = \mathsf{sk}_2/\mathsf{sk}_1$. The scheme is bidirectional since the proxy can compute $\mathsf{rk}^{-1} = 1/\mathsf{rk}$. In the re-proof protocol (Fig. 2), Alice runs a Schnorr's protocol as prover with the proxy, and the proxy runs a Schnorr's protocol as prover with a verifier. Alice sends her commitment $R = g^r$, the proxy picks s in $\mathbb{Z}_p^*$ and computes his commitment as follows: $S = R^{\mathsf{rk}} \cdot g^s$. Note that since s is randomly chosen, S comes from the uniform distribution on $\mathbb{G}$. The proxy receives the challenge c and forwards it to Alice who responds $\alpha = r + \mathsf{sk}_1 \cdot c$. Since $S = g^{(r \cdot \frac{\mathsf{sk}_2}{\mathsf{sk}_1} + s)}$, the proxy is able to compute $\beta = s + \mathsf{rk} \cdot \alpha = s + \frac{\mathsf{sk}_2}{\mathsf{sk}_1} \cdot (r + \mathsf{sk}_1 \cdot c) = (r \cdot \frac{\mathsf{sk}_2}{\mathsf{sk}_1} + s) + \mathsf{sk}_2 \cdot c$, which is the correct response to the challenge c to identify as the owner of pk_2 using the commitment S.

Scheme 1 $\mathsf{B.RE\Pi} = (\mathsf{B.Set}, \mathsf{B.Gen}_1, \mathsf{B.Gen}_2, \Pi_1, \Pi_1, \mathsf{B.RGen}, \Pi_{1\to1})$ *is a ZK-PRI such that* $\Pi_{1\to1}$ *is the protocol given in Fig. 2 and for* $i \in \{1,2\}$*:*
$\mathsf{B.Set}(\lambda)$: *It returns a prime order group setup* $\mathscr{S} = (\mathbb{G}, p, g)$
$\mathsf{B.Gen}_i(\mathscr{S})$: *It picks* $\mathsf{sk}_i \xleftarrow{\$} \mathbb{Z}_p^*$ *and outputs* sk_i *and* $\mathsf{pk}_i = g^{\mathsf{sk}_i}$.
$\mathsf{B.RGen}(\mathsf{sk}_1, \perp, \mathsf{sk}_2, \perp)$: *It returns* $\mathsf{rk} = \mathsf{sk}_2/\mathsf{sk}_1$.

The security properties of this scheme are given by the following theorem.

Theorem 2. *The* $\mathsf{B.RE\Pi}$ *scheme is bidirectional, complete, sound and honest zero-knowledge under the* DL *assumption.*

4.2 Unidirectional ZK-PRI ($\mathsf{U.RE\Pi}$)

Building blocks: We introduce the two following ZKI protocols:

Protocol $\widetilde{\Pi}_1$: The public key is an instance of BDLV in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$ and the private key is the corresponding solution. It is the same protocol as Π_1, except that the verifier knows the value $g_2^{1/\mathsf{sk}}$ in addition to g_1^{sk}.

Protocol Π_2 (Fig. 3): In this protocol, the public key is an instance $\mathsf{pk} = e(g_1, \mathsf{sk})$ of FAPI2 in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$ and the private key sk is the corresponding solution. The protocol Π_2 is built using the same methodology as the Schnorr's protocol.

Construction: The scheme $\mathsf{U.RE\Pi}$ transforms a transcript of $\widetilde{\Pi}_1$ into a transcript of Π_2. The setup is a bilinear setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$, sk_1 and $\mathsf{sk}_2 = g_2^{w_2}$ are the respective secrets of Alice and Bob, and the re-key is $\mathsf{rk} = g_2^{w_2/\mathsf{sk}_1}$. Note that Bob has to know the value of w_2

Prover P		**Verifier** V
sk		$\mathsf{pk} = e(g_1, \mathsf{sk})$
$s \xleftarrow{\$} \mathbb{Z}_p^*$		$c \xleftarrow{\$} \mathbb{Z}_p^*$
$S = e(g_1, g_2)^s$	$\xrightarrow{S}$	
$\beta = g_2^s \cdot \mathsf{sk}^c$	$\xleftarrow{c}$	
	$\xrightarrow{\beta}$	Check that:
		$e(g_1, \beta) \overset{?}{=} S \cdot \mathsf{pk}^c$

Fig. 3. Protocol Π_2 used for $\mathsf{U.RE\Pi}$.

Prover A		**Proxy** P		**Verifier** V
sk_1		$\mathsf{pk}_1 = (h_1, h_2) = (g_1^{\mathsf{sk}_1}, g_2^{1/\mathsf{sk}_1})$; $\mathsf{pk}_2 = e(g_1, g_2^{w_2})$; $\mathsf{rk} = g_2^{w_2/\mathsf{sk}_1}$		$\mathsf{pk}_2 = e(g_1, g_2^{w_2})$
$r \xleftarrow{\$} \mathbb{Z}_p^*$		$s \xleftarrow{\$} \mathbb{Z}_p^*$		$c \xleftarrow{\$} \mathbb{Z}_p^*$
$R = g_1^r$	$\xrightarrow{R}$	$S = e(R, \mathsf{rk}) \cdot e(g_1, g_2)^s$	$\xrightarrow{S}$	
$\alpha = r + \mathsf{sk}_1 \cdot c$	$\xleftarrow{c}$		$\xleftarrow{c}$	
	$\xrightarrow{\alpha}$	If $g_1^\alpha \overset{?}{=} R \cdot h_1^c$		
		Then $\beta = g_2^s \cdot \mathsf{rk}^\alpha$	$\xrightarrow{\beta}$	Check that:
		Else abort		$e(g_1, \beta) \overset{?}{=} S \cdot \mathsf{pk}_2^c$

Fig. 4. Protocol $\Pi_{1\to 2}$ used for U.REΠ.

to compute the re-key $\mathsf{rk} = (g_2^{1/\mathsf{sk}_1})^{w_2}$. Alice sends her commitment g^r, Bob picks s in $\mathbb{Z}_p^*$ and computes his commitment as follows: $S = e(R, \mathsf{rk}) \cdot e(g_1, g_2)^s$. The proxy receives the challenge c and forwards it to Alice who responds $\alpha = r + \mathsf{sk}_1 \cdot c$. Since $S = e(g_1, g_2)^{(r \cdot \frac{w_2}{\mathsf{sk}_1} + s)}$, the proxy is able to compute $\beta = g_2^s \cdot \mathsf{rk}^\alpha = g_2^{s + \frac{w_2}{\mathsf{sk}_1} \cdot (r + \mathsf{sk}_1 \cdot c)} = g_2^{(r \cdot \frac{w_2}{\mathsf{sk}_1} + s) + w_2 \cdot c} = g_2^{(r \cdot \frac{w_2}{\mathsf{sk}_1} + s)} \cdot \mathsf{sk}_2^c$, which is the correct response to the challenge c for the public key $\mathsf{pk}_2 = e(g_1, \mathsf{sk}_2)$.

Scheme 2 U.REΠ $= (\mathsf{U.Set}, \mathsf{U.Gen}_1, \mathsf{U.Gen}_2, \widetilde{\Pi}_1, \Pi_2, \mathsf{U.RGen}, \Pi_{1\to 2})$ *is a ZK-PRI such that* $\Pi_{1\to 2}$ *is the protocol described in Fig 4 and:*

$\mathsf{U.Set}(\lambda)$: *It returns a bilinear map setup* $\mathscr{S} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$

$\mathsf{U.Gen}_1(\mathscr{S})$: *It picks* $\mathsf{sk}_1 \xleftarrow{\$} \mathbb{Z}_p^*$*, sets* $h_1 = g_1^{\mathsf{sk}_1}$ *and* $h_2 = g_2^{1/\mathsf{sk}_1}$ *and returns* sk_1 *and* $\mathsf{pk}_1 = (h_1, h_2)$.

$\mathsf{U.Gen}_2(\mathscr{S})$: *It picks* $w_2 \xleftarrow{\$} \mathbb{Z}_p^*$ *and returns* $\mathsf{sk}_2 = g_2^{w_2}$, $\mathsf{pk}_2 = e(g_1, \mathsf{sk}_2)$ *and the intermediate secret* w_2.

$\mathsf{U.RGen}(\mathsf{pk}_1, \mathsf{sk}_2, w_2)$: *It returns* $\mathsf{rk} = h_2^{w_2}$ *using* $\mathsf{pk}_1 = (h_1, h_2)$.

The security properties of this scheme are given by the following theorem.

Theorem 3. U.REΠ *is unidirectional, complete, sound and honest zero-knowledge under the* CDH*, the* DL *and the* FAPI2 *assumptions.*

Remark 1. B.REΠ and U.REΠ are honest zero-knowledge. As in the Schnorr's protocol, it is possible to design two fully zero-knowledge ZK-PRI by forcing the verifier to choose the challenge c in $\{0, 1\}$ instead of $\mathbb{Z}_p^*$. However, these protocols would be less practical than B.REΠ and U.REΠ since they would have to be repeated k times (for a chosen security parameter k). Note that there exists constant round protocols for perfect zero-knowledge proof [12]; the design of

$$
\begin{array}{lcl}
\textbf{Prover } P & & \textbf{Verifier } V \\
\mathsf{sk} & & \mathsf{pk} = g^{\mathsf{sk}} \\
\hline
r, u \xleftarrow{\$} \mathbb{Z}_p^* & & \\
R = g^r; U = g^u & \xrightarrow{(R,U)} & c, d \xleftarrow{\$} \mathbb{Z}_p^* \\
z = r + \mathsf{sk} \cdot d & \xleftarrow{(c,d)} & \\
\mu = u + r \cdot c & \xrightarrow{(z,\mu)} & \text{Check that: } g^{\mu} \overset{?}{=} U \cdot R^c \\
 & & \text{and } g^z \overset{?}{=} R \cdot \mathsf{pk}^d
\end{array}
$$

Fig. 5. Protocol Π_3 used in $\mathsf{NiB.RE\Pi}$.

a ZK-PRI that is constant round and perfectly zero-knowledge is still an open problem.

4.3 Bidirectional NIZK-PRI ($\mathsf{NiB.RE\Pi}$)

Building blocks: We introduce the two following ZKP protocols:

Protocol Π_3 (Fig. 5): This protocol is a modified version of Schnorr's protocol where the prover must know the discrete logarithm of his commitment R. Thus, the prover uses *a second* Schnorr's protocol to prove it. To do that, the prover sends a second commitment $U = g^u$ and receives a second challenge d. The prover uses c to prove the knowledge of r using the commitment U and uses d to identify himself using his secret sk and the commitment R.

Protocol Π_4 (Fig. 6): This protocol is a ZKI where $\mathsf{pk} = g^{\mathsf{sk}}$ is a DL instance in $(\mathbb{G}, p, g)$. It is based on Π_3, but it requires that the commitment is built in a particular way. More precisely, the prover sends a first *classical* commitment $R = g^r$ and receives the challenge d. As in Π_3, the prover must prove the knowledge of the discrete logarithm of his commitment R. The prover chooses two random values t and s in $\mathbb{Z}_p^*$ and computes $S = R^t \cdot g^s$. It sends the *second* commitment S and proves that he knows the two values t and s to the verifier. Then he identifies himself using his secret sk in a Schnorr's way using the commitment S. Note that the challenge is known by the prover before the commitment S. However, the proof works because the prover knows r the discrete logarithm of R (committed before the reception of the challenge d): it can compute the value $r \cdot t + s$ (which is the discrete logarithm of S) and succeed the proof.

Using the Fiat-Shamir heuristic, we build the two non-interactive proofs $\mathsf{NI.\Pi_3}$ and $\mathsf{NI.\Pi_4}$ from the interactive proofs Π_3 and Π_4 as follows.

Scheme 3 $\mathsf{NI.\Pi_3} = (\Pi_3.\mathsf{Set}, \Pi_3.\mathsf{Gen}, \Pi_3.\mathsf{Prove}, \Pi_3.\mathsf{Verify})$ *is a NIZKI where:*

$$
\begin{array}{lcl}
\textbf{Prover } P & & \textbf{Verifier } V \\
\mathsf{sk} & & \mathsf{pk} = g^{\mathsf{sk}} \\
\hline
a, b, r, s, t, u \xleftarrow{\$} \mathbb{Z}_p^* & & c, d, f \xleftarrow{\$} \mathbb{Z}_p^* \\
R = g^r ; U = g^u & \xrightarrow{\quad (R,U) \quad} & \\
\mu = u + r \cdot c & \xleftarrow{\quad (c,d) \quad} & \\
S = g^{r \cdot t + s} & \xrightarrow{\quad \mu \quad} & \text{Check that:} \\
A = R^a \; ; B = g^b & \xrightarrow{\quad (A,B,S) \quad} & g^\mu \overset{?}{=} U \cdot R^c \\
\alpha = a + t \cdot f & \xleftarrow{\quad f \quad} & \\
\beta = b + s \cdot f & & \\
\theta = s + r \cdot t + \mathsf{sk} \cdot d & \xrightarrow{\quad (\alpha,\beta,\theta) \quad} & \text{Check that: } g^\theta \overset{?}{=} S \cdot \mathsf{pk}^d \\
& & \text{and } R^\alpha \cdot g^\beta \overset{?}{=} A \cdot B \cdot S^f
\end{array}
$$

Fig. 6. Protocol Π_4 used in $\mathsf{NiB.RE\Pi}$.

$\Pi_3.\mathsf{Set}(\lambda)$: *It returns a prime order group setup* $\mathscr{S} = (\mathbb{G}, p, g, H)$ *where* $H : \{0,1\}^* \to \mathbb{Z}_p^*$ *is a hash function.*

$\Pi_3.\mathsf{Gen}(\mathscr{S})$: *It picks* $\mathsf{sk} \xleftarrow{\$} \mathbb{Z}_p^*$ *and outputs* sk *and* $\mathsf{pk} = g^{\mathsf{sk}}$.

$\Pi_3.\mathsf{Prove}(\mathsf{pk}, \mathsf{sk})$: *It picks* $r \xleftarrow{\$} \mathbb{Z}_p^*$ *and* $u \xleftarrow{\$} \mathbb{Z}_p^*$, *computes* $R = g^r$, $U = g^u$, $c = H(R, U, 0)$, $d = H(R, U, 1)$, $z = r + \mathsf{sk} \cdot c$ *and* $\mu = u + r \cdot c$, *and outputs* $\pi = (R, U, z, \mu)$.

$\Pi_3.\mathsf{Verify}(\mathsf{pk}, \pi)$: *It computes* $c = H(R, U, 0)$ *and* $d = H(R, U, 1)$. *If* $g^\mu \overset{?}{=} U \cdot R^c$ *and* $g^z \overset{?}{=} R \cdot \mathsf{pk}^d$ *then it outputs* 1, *else* 0.

Scheme 4 $\mathsf{NI}.\Pi_4 = (\Pi_4.\mathsf{Set}, \Pi_4.\mathsf{Gen}, \Pi_4.\mathsf{Prove}, \Pi_4.\mathsf{Verify})$ *is a NIZKI where* $\Pi_4.\mathsf{Set} = \Pi_3.\mathsf{Set}$ *and* $\Pi_4.\mathsf{Gen} = \Pi_3.\mathsf{Gen}$ *and:*

$\Pi_4.\mathsf{Prove}(\mathsf{pk}, \mathsf{sk})$: *This algorithm picks* r, s, t, u, a *and* b *in the uniform distribution on* $\mathbb{Z}_p^*$. *It computes:*

$$
\begin{array}{lll}
R = g^r & f = H(R, U, A, B, S) & S = g^{r \cdot t} \cdot g^s \\
c = H(R, U, 0) & \beta = b + s \cdot f & B = g^b \\
\mu = u + r \cdot c & U = g^u & \alpha = a + t \cdot f \\
A = R^a & d = H(R, U, 1) & \theta = s + r \cdot t + \mathsf{sk} \cdot d
\end{array}
$$

It outputs $\pi = (R, U, \mu, S, A, B, \alpha, \beta, \theta)$.

$\Pi_4.\mathsf{Verify}(h, \pi)$: *It computes the values* $c = H(R, U, 0)$, $d = H(R, U, 1)$ *and* $f = H(R, U, A, B, S)$. *If* $(g^\mu \overset{?}{=} U \cdot R^c)$, $(R^\alpha \cdot g^\beta \overset{?}{=} A \cdot B \cdot S^f)$ *and* $(g^\theta \overset{?}{=} S \cdot \mathsf{pk}^d)$ *then it outputs* 1, *else* 0.

Construction: We first remark that it is not possible to use the Fiat-Shamir transformation on our previous interactive bidirectional ZK-PRI: indeed, it is not possible to use as challenge the hash of the delegate's commitment R since the proxy uses another commitment S. Then we need to use ZKI schemes such that *the same* commitment is used to compute the challenge for both the delegate and the proxy. Note that in $\mathsf{NI}.\Pi_3$ and $\mathsf{NI}.\Pi_4$, the challenge d is computed from *the same* commitment pair (R, U). In $\mathsf{NI}.\Pi_4$ some other values are committed during the protocol but the last computation of the proof uses the challenge d. Thus we can transform a $\mathsf{NI}.\Pi_3$ proof into a $\mathsf{NI}.\Pi_4$ one. The re-prove method is similar to the $\Pi_{1\to1}$ except that the proxy must prove that he *correctly* constructs his commitment S from the delegate commitment R. Forcing the proxy to prove that S totally depends on the commitment R implies that the commitment must be known after R is committed but can be revealed before that S is committed.

Scheme 5 $\mathsf{NiB.RE}\Pi = (\Pi_3.\mathsf{Set}, \Pi_3.\mathsf{Gen}, \Pi_4.\mathsf{Gen}, \Pi_3.\mathsf{Prove}, \Pi_4.\mathsf{Prove}, \Pi_3.\mathsf{Verify}, \Pi_4.\mathsf{Verify}, \mathsf{B.RGen}, \mathsf{NiB.RProve})$ *is a NIZK-PRI where* $\mathsf{B.RGen}$ *is defined in Scheme 1,* $\Pi_3.\mathsf{Gen}$*,* $\Pi_3.\mathsf{Prove}$ *and* $\Pi_3.\mathsf{Verify}$ *are defined in Scheme 3,* $\Pi_4.\mathsf{Gen}$*,* $\Pi_4.\mathsf{Prove}$ *and* $\Pi_4.\mathsf{Verify}$ *are defined in Scheme 4 and:*

$\mathsf{NiB.RProve}(\mathsf{rk}, \pi_1)$: *Using* $\pi_1 = (R, U, z, \mu)$*, this algorithm picks* s, a *and* b *in the uniform distribution on* $\mathbb{Z}_p^*$ *and computes* $S = R^{\mathsf{rk}} \cdot g^s$, $A = R^a$, $B = g^b$, $f = H(R, U, A, B, S)$, $\alpha = a + \mathsf{rk} \cdot f$, $\beta = b + s \cdot f$ *and* $\theta = s + z \cdot \mathsf{rk}$. *It outputs* $\pi_2 = (R, U, \mu, A, B, S, \alpha, \beta, \theta)$.

The security properties of this scheme are given by the following theorem.

Theorem 4. $\mathsf{NiB.RE}\Pi$ *scheme is bidirectional, complete, sound and zero-knowledge in the random oracle model under the* DL *assumption.*

4.4 Unidirectional NIZK-PRI (NiU.REΠ)

Building blocks: We introduce the two following ZKI protocols:

Protocol $\widetilde{\Pi}_3$: pk is an instance of BDLV in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$. It is exactly the same protocol as Π_3 except that the verifier knows the value $g_2^{1/\mathsf{sk}}$ in addition to g_1^{sk}.

Protocol Π_5 (Fig. 7): This protocol is an adaptation of Π_4 for the $\mathsf{FAPI2}$ problem.

Using the Fiat-Shamir heuristic, we build non-interactive ZKIs $\mathsf{NI}.\widetilde{\Pi}_3$ and $\mathsf{NI}.\Pi_5$ from the respective interactive ZKIs $\widetilde{\Pi}_3$ and Π_5 as follows.

Scheme 6 $\mathsf{NI}.\widetilde{\Pi}_3 = (\widetilde{\Pi}_3.\mathsf{Set}, \widetilde{\Pi}_3.\mathsf{Gen}, \widetilde{\Pi}_3.\mathsf{Prove}, \widetilde{\Pi}_3.\mathsf{Verify})$ *is a NIZKI where:*

$$\begin{array}{lcl}
\textbf{Prover } P & & \textbf{Verifier } V \\
\mathsf{sk} & & \mathsf{pk} = e(g_1, \mathsf{sk}) \\
\hline
a, b, r, s, t, u \overset{\$}{\leftarrow} \mathbb{Z}_p^* & & c, d, f \overset{\$}{\leftarrow} \mathbb{Z}_p^* \\
R = g_1^r \; ; U = g_1^u & \xrightarrow{(R,U)} & \\
\mu = u + r \cdot c & \xleftarrow{(c,d)} & \\
S = e(g_1, g_2)^{r \cdot t + s} & \xrightarrow{\mu} & \text{Check that:} \\
A = e(R, g_2)^a \; ; B = e(g_1, g_2)^b & \xrightarrow{(A,B,S)} & g_1^\mu \overset{?}{=} U \cdot R^c \\
\alpha = g_2^{a + t \cdot f} & \xleftarrow{f} & \\
\beta = g_2^{b + s \cdot f} & & \\
\theta = g_2^{s + r \cdot t} \cdot \mathsf{sk}^d & \xrightarrow{(\alpha,\beta,\theta)} & \text{Check that: } e(g_1, \theta) \overset{?}{=} S \cdot \mathsf{pk}^d \\
 & & \text{and } e(R, \alpha) \cdot e(g_1, \beta) \overset{?}{=} A \cdot B \cdot S^f
\end{array}$$

Fig. 7. Protocol Π_5 used in $\mathsf{NiU.RE\Pi}$.

$\widetilde{\Pi}_3.\mathsf{Set}(\lambda)$: *It returns a bilinear map setup* $\mathscr{S} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e, H)$ *where* $H : \{0,1\}^* \to \mathbb{Z}_p^*$ *is a hash function.*

$\widetilde{\Pi}_3.\mathsf{Gen}(\mathscr{S})$: *It picks* $\mathsf{sk} \overset{\$}{\leftarrow} \mathbb{Z}_p^*$*, computes* $h_1 = g_1^{\mathsf{sk}}$ *and* $h_2 = g_2^{1/\mathsf{sk}}$ *and outputs* sk *and* $\mathsf{pk} = (h_1, h_2)$.

$\widetilde{\Pi}_3.\mathsf{Prove}(\mathsf{pk}, \mathsf{sk})$: *It picks* $r \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ *and* $u \overset{\$}{\leftarrow} \mathbb{Z}_p^*$*, computes* $R = g_1^r$*,* $U = g_1^u$*,* $c = H(R, U, 0)$*,* $d = H(R, U, 1)$*,* $z = r + \mathsf{sk} \cdot c$ *and* $\mu = u + r \cdot c$*, and outputs* $\pi = (R, U, z, \mu)$.

$\widetilde{\Pi}_3.\mathsf{Verify}(\mathsf{pk}, \pi)$: *Using* $\mathsf{pk} = (h_1, h_2)$*, it computes* $c = H(R, U, 0)$ *and* $d = H(R, U, 1)$*. It returns* 1 *iff* $g_1^\mu \overset{?}{=} U \cdot R^c$ *and* $g_1^z \overset{?}{=} R \cdot h_1^d$.

Scheme 7 $\mathsf{NI.\Pi}_5 = (\Pi_5.\mathsf{Set}, \Pi_5.\mathsf{Gen}, \Pi_5.\mathsf{Prove}, \Pi_5.\mathsf{Verify})$ *is a NIZKI where* $\Pi_5.\mathsf{Set} = \widetilde{\Pi}_3.\mathsf{Set}$ *and:*

$\Pi_5.\mathsf{Gen}(\mathscr{S})$: *It picks* $w \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ *and returns* $\mathsf{sk} = g_2^w$ *and* $\mathsf{pk} = e(g_1, \mathsf{sk})$.

$\Pi_5.\mathsf{Prove}(\mathsf{pk}, \mathsf{sk})$: *This algorithm picks* r, s, t, u, a *and* b *in* $\mathbb{Z}_p^*$ *and computes:*

$$\begin{array}{lll}
R = g_1^r & f = H(R, U, A, B, S) & S = e(g_1, g_2)^{r \cdot t + s} \\
c = H(R, U, 0) & \beta = g_2^{b + s \cdot f} & B = e(g_1, g_2)^b \\
\mu = u + r \cdot c & U = g_1^u & \alpha = g_2^{a + t \cdot f} \\
A = e(R, g_2)^a & d = H(R, U, 1) & \theta = g_2^{s + r \cdot t} \cdot \mathsf{sk}^d
\end{array}$$

It outputs $\pi = (R, U, \mu, S, A, B, \alpha, \beta, \theta)$.

$\Pi_5.\mathsf{Verify}(\mathsf{pk}, \pi)$: *This algorithm computes* $c = H(R, U, 0)$*,* $d = H(R, U, 1)$ *and* $f = H(R, U, A, B, S)$*. If* $(g_1^\mu \overset{?}{=} U \cdot R^c)$*,* $(e(R, \alpha) \cdot e(g_1, \beta) \overset{?}{=} A \cdot B \cdot S^f)$ *and* $(e(g_1, \theta) \overset{?}{=} S \cdot \mathsf{pk}^d)$ *then it outputs* 1*, else* 0.

Construction: The last scheme $\mathsf{NiU.RE\Pi}$ is unidirectional and non-interactive. It is obtained by merging the design of $\mathsf{NiB.RE\Pi}$ for the non-interactivity and the design of $\mathsf{U.RE\Pi}$ for the unidirectionality.

Scheme 8 $\mathsf{NiU.RE\Pi} = (\Pi_5.\mathsf{Set}, \mathsf{U.Gen}_1, \mathsf{U.Gen}_2, \widetilde{\Pi}_3.\mathsf{Prove}, \Pi_5.\mathsf{Prove}, \widetilde{\Pi}_3.\mathsf{Verify}, \Pi_5.\mathsf{Verify}, \mathsf{U.RGen}, \mathsf{NiU.RProve})$ *is a NIZK-PRI scheme where* $\widetilde{\Pi}_3.\mathsf{Prove}$ *and* $\widetilde{\Pi}_3.\mathsf{Verify}$ *are defined in Scheme 6,* $\Pi_5.\mathsf{Set}$*,* $\Pi_5.\mathsf{Prove}$ *and* $\Pi_5.\mathsf{Verify}$ *are defined in Scheme 7 and* $\mathsf{U.Gen}_1$*,* $\mathsf{U.Gen}_2$ *and* $\mathsf{U.RGen}$ *are defined in Scheme 2 and:*

$\mathsf{NiU.RProve}(\mathsf{rk}, \pi_1)$: *Using* $\pi_1 = (R, U, z, \mu)$*, this algorithm picks* s, a *and* b *in the uniform distribution on* $\mathbb{Z}_p^*$ *and computes* $S = e(R, \mathsf{rk}) \cdot e(g_1, g_2)^s$*,* $A = e(R, g_2)^a$*,* $B = e(g_1, g_2)^b$*,* $f = H(R, U, A, B, S)$*,* $\alpha = g_2^a \cdot \mathsf{rk}^f$*,* $\beta = g_2^{b+s\cdot f}$ *and* $\theta = g_2^s \cdot \mathsf{rk}^z$*. It outputs the proof* $\pi_2 = (R, U, \mu, A, B, S, \alpha, \beta, \theta)$*.*

The security properties of this scheme are given by the following theorem.

Theorem 5. $\mathsf{NiU.RE\Pi}$ *is unidirectional, complete, sound and zero-knowledge in the random oracle model under the* DL*, the* CDH *and the* $\mathsf{FAPI2}$ *assumptions.*

4.5 Two Proxy Re-Signature Sechemes

The non-interactive Schnorr's protocol can be used to design a digital signature scheme as follows. The signer who knows the secret key sk for the public key $\mathsf{pk} = g^{\mathsf{sk}}$ generates a non-interactive proof $(g^r, r + \mathsf{sk} \cdot H(g^r, m))$ using the message m in addition to g^r to generate the challenge $c = H(g^r, m)$. Since such a proof cannot be performed by anybody who does not know the secret sk, this method allows us to produce digital signatures which are unforgeable for anybody who does not know the secret key sk. Applying the same method to $\mathsf{NiB.RE\Pi}$ and $\mathsf{NiU.RE\Pi}$, we can construct two new proxy re-signature schemes respectively bidirectional and unidirectional. For this purpose the signer and the proxy add the signed message m in the hash of each challenge in both $\mathsf{NiB.RE\Pi}$ and $\mathsf{NiU.RE\Pi}$.

5 Conclusion

In this paper, we define the bidirectional/unidirectional and interactive/non-interactive ZK-PRI. We design several proxies re-ZKP for number theory-based cryptographic problems: a multi-hop bidirectional interactive ZK-PRI, a single-hop unidirectional interactive ZK-PRI, a single-hop bidirectional non-interactive ZK-PRI and a single-hop unidirectional non-interactive ZK-PRI. We also design two new proxy re-signature schemes (bidirectional and unidirectional) from our non-interactive proxy re-ZKP schemes. We leave as an open problem the design of multi-hop unidirectional schemes and multi-hop non-interactive schemes. In the future, it should be interesting to investigate the case of ZK-PRI based on other cryptographic problems, or NP-complete problems.

References

1. G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. In *NDSS 2005*. The Internet Society, Feb. 2005.
2. G. Ateniese and S. Hohenberger. Proxy re-signatures: New definitions, algorithms, and applications. In V. Atluri, C. Meadows, and A. Juels, editors, *ACM CCS 05*, pages 310–319. ACM Press, Nov. 2005.
3. F. Bao, R. H. Deng, and H. Zhu. Variations of diffie-hellman problem. In *ICICS*, volume 6280, pages 301–312, 2003.
4. D. Bernhard, O. Pereira, and B. Warinschi. How not to prove yourself: Pitfalls of the Fiat-Shamir heuristic and applications to Helios. In X. Wang and K. Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 626–643. Springer, Dec. 2012.
5. M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In K. Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 127–144. Springer, May / June 1998.
6. D. Boneh. The decision Diffie-Hellman problem. In *Third Algorithmic Number Theory Symposium (ANTS)*, volume 1423 of *LNCS*. Springer, 1998. Invited paper.
7. S. S. M. Chow and R. C.-W. Phan. Proxy re-signatures in the standard model. In T.-C. Wu, C.-L. Lei, V. Rijmen, and D.-T. Lee, editors, *ISC 2008*, volume 5222 of *LNCS*, pages 260–276. Springer, Sept. 2008.
8. D. Derler, S. Ramacher, and D. Slamanig. Homomorphic proxy re-authenticators and applications to verifiable multi-user data aggregation. Cryptology ePrint Archive, Report 2017/086, 2017. `http://eprint.iacr.org/2017/086.pdf`.
9. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Aug. 1987.
10. S. Galbraith, K. Paterson, and N. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, Sept. 2008.
11. S. D. Galbraith, F. Hess, and F. Vercauteren. Aspects of pairing inversion. In *Information Theory, IEEE Transactions*, volume 54, pages 5719 – 5728. IEEE, 2008.
12. O. Goldreich and A. Kahan. How to construct constant-round zero-knowledge proof systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.
13. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.
14. M. Green and G. Ateniese. Identity-based proxy re-encryption. In J. Katz and M. Yung, editors, *ACNS 07*, volume 4521 of *LNCS*, pages 288–306. Springer, June 2007.
15. A. Ivan and Y. Dodis. Proxy cryptography revisited. In *NDSS 2003*. The Internet Society, Feb. 2003.
16. H. Jannati1, M. Salmasizadeh, A. J. Mohajeri, and A. Moradi. Introducing proxy zero-knowledge proof and utilization in anonymous credential systems. In *Security and Communication Networks*, volume 6, page 161172, 2013.
17. J. Katz and R. Ostrovsky. Round-optimal secure two-party computation. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 335–354. Springer, Aug. 2004.

18. M. Keller, G. L. Mikkelsen, and A. Rupp. Efficient threshold zero-knowledge with applications to user-centric protocols. In A. Smith, editor, *ICITS 12*, volume 7412 of *LNCS*, pages 147–166. Springer, Aug. 2012.
19. B. Libert and D. Vergnaud. Multi-use unidirectional proxy re-signatures. In P. Ning, P. F. Syverson, and S. Jha, editors, *ACM CCS 08*, pages 511–520. ACM Press, Oct. 2008.
20. B. Libert and D. Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. In R. Cramer, editor, *PKC 2008*, volume 4939 of *LNCS*, pages 360–379. Springer, Mar. 2008.
21. S. Luo, J. Hu, and Z. Chen. Ciphertext policy attribute-based proxy re-encryption. In M. Soriano, S. Qing, and J. López, editors, *ICICS 10*, volume 6476 of *LNCS*, pages 401–415. Springer, Dec. 2010.
22. T. Okamoto and K. Ohta. Divertible zero knowledge interactive proofs and commutative random self-reducibility. In J.-J. Quisquater and J. Vandewalle, editors, *EUROCRYPT'89*, volume 434 of *LNCS*, pages 134–148. Springer, Apr. 1990.
23. C.-P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, Aug. 1990.
24. J. Shao, M. Feng, B. Zhu, Z. Cao, and P. Liu. The security model of unidirectional proxy re-signature with private re-signature key. In R. Steinfeld and P. Hawkes, editors, *ACISP 10*, volume 6168 of *LNCS*, pages 216–232. Springer, July 2010.

Scheme	Uni.	Non-in.	Multi.	P. free	P_1	P_2	V_1	V_2	Proxy
Blaze *et al.* [5]	no	no	**yes**	**yes**	$1e$	$1e$	$1e$	$1e$	0
B.REΠ	no	no	**yes**	**yes**	$1e$	$1e$	$2e$	$2e$	$4e$
U.REΠ	**yes**	no	no	no	$1e$	$1p+3e$	$2e$	$1p+1e$	$2p+5e$
NiB.REΠ	no	**yes**	no	**yes**	$2e$	$5e$	$4e$	$7e$	$4e$
NiU.REΠ	**yes**	**yes**	no	no	$2e$	$2p+9e$	$4e$	$3p+4e$	$3p+8e$

Fig. 8. Comparison of our schemes.

A Security and Efficiency Comparison

The table in Fig. 8 compares the properties and the efficiency of the insecure scheme of Blaze *et al.* [5] and our four schemes. The first columns sum up the properties of each scheme, namely *unidirectional*, *non-interactive*, *multi-hop* and *pairing free*. The last columns sum up the computation cost for each entities: the delegate P_1, the delegator P_2, and the respective verifiers V_1 and V_2. The last column gives the computation cost for the proxy. We only evaluate the number of exponentiations and pairing computations. We denote by e (resp. p) the computation time of an exponentiation (resp. a pairing computation). All our schemes can be used for the applications that we propose in Introduction. However, each of them allows a different compromise between security and efficiency.
Security: Unidirectional schemes are more secure than bidirectional schemes. Indeed, since the bidirectional re-proof key generation requires the secret keys of both the delegate and the delegator, this kind of proxy requires a fully trusted key manager. Moreover, if the proxy and the delegator collude then they can deduce the secret key of the delegate. In a practical scenario, the delegate could refuse to reveal his secret key to the trusted re-proof key manager. In unidirectional schemes, the delegator computes the re-proof key with the public key of the delegate, then the secret key of the delegate is protected.
Efficency: The efficiency is evaluated by two different properties: the number of interactions and the number of pairing computations. Thus, our two non-interactive schemes are optimal for the number of interactions (only one interaction), and our two bidirectional schemes are pairing free.

B Proof that BDLV is hard

Lemma 1. BDLV *is hard to solve under the* DL *hypothesis.*

Proof. Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be three groups of prime order p, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a type 2 non-degenerate bilinear pairing and $g_2 \in \mathbb{G}_2$ be a generator. Suppose that there exists a polynomial time algorithm $\mathcal{A}$ that solves the problem BDLV in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$ for any g_1 chosen in the uniform distribution on $\mathbb{G}_1$. We construct the polynomial time algorithm $\mathcal{B}$ that solves DL in $(\mathbb{G}_2, p, g_2)$. $\mathcal{B}$ receives $h_2 = g_2^{x'}$ as input. Since e is a type 2 pairing there exists a computationally efficient morphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$ [10]. $\mathcal{B}$ picks $a \in \mathbb{Z}_p^*$ and computes $h_1 = \phi(g_2^a)$ and

$g_1 = \phi(h_2^a) = \phi((g_2^a)^{x'}) = h_1^{x'}$. Thus, $g_1^{1/x'} = h_1$, and (h_1, h_2) is an instance of BDLV in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$ with regards to the solution $x = 1/x'$. $\mathcal{B}$ runs $x'' \leftarrow \mathcal{A}(h_1, h_2)$ and returns $1/x''$. Note that $1/x'' = 1/x = x'$ iff $\mathcal{A}$ solves the instance (h_1, h_2), then $\mathcal{B}$ solves DL in $(\mathbb{G}_2, p, g_2)$ with the same complexity and the same probability that $\mathcal{A}$ solves BDLV in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$. Note that this proof requires the hypothesis that e is a type 2 pairing. To the best of our knowledge, there is neither known reduction to DL in $(\mathbb{G}_2, p, g_2)$ where e is a type 3 pairing, nor known polynomial time algorithm that solves BDLV. $\square$

C Strong soundness.

We present stronger soundness notions for ZKI and ZK-PRI. First, we introduce the notion of *knowledge extractor* used for proofs of knowledge [13]. A knowledge extractor is an algorithm that extracts the secret key sk from the view of any prover who is able to identify with non-negligible probability. In this definition, the input of the prover is not specified.

Definition 8 (Knowledge extractor).

– *Let $I = (\mathsf{Set}, \mathsf{Gen}, \mathsf{Proof})$ be a ZKI scheme. A* knowledge extractor *$\mathcal{E}$ of I is a PPT algorithm such that for any $\mathscr{S} \leftarrow \mathsf{Set}(\lambda)$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathscr{S})$, any PPT prover P, any bit-string in and any non-negligible function ϵ_1 such that:*
$\Pr\left[b \leftarrow \mathtt{out}_V(\mathsf{Proof}\langle P(\mathsf{in}); V(\mathsf{pk})\rangle) : b = 1\right] \geq \epsilon_1(\lambda)$
then there exists non-negligible function ϵ_2 such that:
$\Pr\left[v \leftarrow \mathtt{view}_P(\mathsf{Proof}\langle P(\mathsf{in}); V(\mathsf{pk})\rangle); s \leftarrow \mathcal{E}(v) : s = \mathsf{sk}\right] \geq \epsilon_2(\lambda)$

– *Let $N = (\mathsf{Set}, \mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ be a NIZKI scheme. A* knowledge extractor *$\mathcal{E}$ of N is a PPT algorithm such that for any $\mathscr{S} \leftarrow \mathsf{Set}(\lambda)$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathscr{S})$, any PPT algorithm $\mathcal{A}$, any bit-string in and any non-negligible function ϵ_1 such that:*
$\Pr\left[\pi \leftarrow \mathcal{A}^{\mathsf{Prove}(\mathsf{pk},\mathsf{sk})}(\mathsf{in}); b \leftarrow \mathsf{Verify}(\mathsf{pk}, \pi) : b = 1\right] \geq \epsilon_1(\lambda)$
where π was not produced by the oracle $\mathsf{Prove}(\mathsf{pk}, \mathsf{sk})$, then there exists a non-negligible function ϵ_2 such that:
$\Pr\left[v \leftarrow \mathtt{view}_{\mathcal{A}}(\mathcal{A}^{\mathsf{Prove}(\mathsf{pk},\mathsf{sk})}(\mathsf{in})); s \leftarrow \mathcal{E}(v) : s = \mathsf{sk}\right] \geq \epsilon_2(\lambda)$

We present the *secret security*. A ZKI is secret secure if it is hard to compute a secret key from the corresponding public key. A ZK-PRI is *secret secure* if it is hard to compute the delegate or the delegator secret key using the corresponding public keys and the re-proof key.

Definition 9 (Secret Security).

– *We say that a ZKI (or NIZKI) is secret secure when for $\mathscr{S} \leftarrow \mathsf{Gen}(\lambda)$ and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathscr{S})$ we have, for all PPT adversaries $\mathcal{A}$:*
$\Pr\left[s \leftarrow \mathcal{A}(\mathsf{pk}) : s = \mathsf{sk}\right] \leq \epsilon(\lambda)$ *where ϵ is negligible.*

- *We say that a bidirectional (resp. unidirectional) ZK-PRI is* secret secure *when for any* $\mathscr{S} \leftarrow \mathsf{Gen}(\lambda)$*, any* $(\mathsf{pk}_i, \mathsf{sk}_i, w_i) \leftarrow \mathsf{Gen}_i(\mathscr{S})$ *for* $i \in \{1,2\}$ *and any* $rk \leftarrow \mathsf{RGen}(\mathsf{sk}_1, w_1, \mathsf{sk}_2, w_2)$ *(resp.* $rk \leftarrow \mathsf{RGen}(\mathsf{pk}_1, \mathsf{sk}_2, w_2)$*), we have, for all PPT adversaries* $\mathcal{A}$*:* $Pr[\mathsf{sk} \leftarrow \mathcal{A}(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2) : \mathsf{sk}_1 = \mathsf{sk} \vee \mathsf{sk}_2 = \mathsf{sk}] \leq \epsilon(\lambda)$ *where* ϵ *is negligible.*

A ZKI is *strongly sound* when it is secret secure and it admits a knowledge extractor. A ZK-PRI is strongly sound when it is secret secure and the two corresponding ZKI are strong sound.

Definition 10 (Strong soundness).
- *We say that a ZKI (or NIZKI) denoted* I *is* strongly sound *when* (i) *there exists a PPT extractor* $\mathcal{E}$ *for* I *and* (ii) I *is secret secure.*
- *Let* $P = (\mathsf{Set}, \mathsf{Gen}_1, \mathsf{Gen}_2, \mathsf{Proof}_1, \mathsf{Proof}_2, \mathsf{RGen}, \mathsf{RProof})$ *be a ZK-PRI. We denote by* I_i *for* $i \in \{1,2\}$ *the ZKI scheme* $(\mathsf{Set}, \mathsf{Gen}_i, \mathsf{Proof}_i)$*. We say that* P *is* strongly sound *when* (i) I_1 *and* I_2 *are both strongly sound and* (ii) P *is secret secure.*
- *Let* $Q = (\mathsf{Set}, \mathsf{Gen}_1, \mathsf{Gen}_2, \mathsf{Prove}_1, \mathsf{Prove}_2, \mathsf{Verify}_1 \mathsf{Verify}_2, \mathsf{RGen}, \mathsf{RProof})$ *be a NIZK-PRI. We denote by* I_i *for* $i \in \{1,2\}$ *the NIZKI scheme* $(\mathsf{Set}, \mathsf{Gen}_i, \mathsf{Prove}_i, \mathsf{Verify}_i)$*. We say that* P *is* strongly sound *when* (i) I_1 *and* I_2 *are both strongly sound and* (ii) Q *is secret secure.*

We show that the strong soundness implies the soundness for both ZKI and ZK-PRI. The idea is that, when a scheme is strong sound, someone who does not know the exact value of the secret key cannot identify. Thus the scheme is *sound even if* the adversary has some additional information that does not trivially leak the secret key.

Lemma 2. *A strongly sound ZKI (or NIZKI) is sound.*

Proof. We give the proof for interactive ZKI, the non-interactive case can be proven in a similar way. Suppose that there exists a ZKI denoted $I = (\mathsf{Set}, \mathsf{Gen}, \mathsf{Proof})$ that is strongly sound but not sound, we have the following hypothesis:
1. There exists a knowledge extractor $\mathcal{E}$ for I.
2. I is secret secure.
3. I is not sound.

Using Hypothesis 3, we deduce that there exists a PPT prover P^* such that for any $\mathscr{S} \leftarrow \mathsf{Set}(\lambda)$ and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathscr{S})$:
$\mathsf{Pr}\left[b \leftarrow \mathtt{out}_V(\mathsf{Proof}\langle P(\mathsf{pk}); V(\mathsf{pk})\rangle) : b = 1\right] \geq \epsilon_1(\lambda)$
where ϵ_1 is not negligible. Using Hypothesis 1, we deduce that:
$\mathsf{Pr}\left[v \leftarrow \mathtt{view}_P(\mathsf{Proof}\langle P^*(\mathsf{pk}); V(\mathsf{pk})\rangle); s \leftarrow \mathcal{E}(v) : s = \mathsf{sk}\right] \geq \epsilon_2(\lambda)$
where ϵ_2 is not negligible. We show how to build a PPT adversary $\mathcal{A}$ that breaks the secret security of I with non-negligible probability using P^* and $\mathcal{E}$:
- $\mathcal{A}$ receives pk as input.

- $\mathcal{A}$ runs $v \leftarrow \texttt{view}_P(\mathsf{Proof}\langle P^*(\mathsf{pk}); V(\mathsf{pk})\rangle)$ and $s \leftarrow \mathcal{E}(v)$.
- $\mathcal{A}$ returns s.

Then $s = \mathsf{sk}$ with non-negligible probability, which contradicts Hypothesis 2. □

Lemma 3. *A strongly sound ZK-PRI (or NIZK-PRI) is sound.*

Proof. We give the proof for bidirectional interactive ZK-PRI, the unidirectional and non-interactive case can be proven in a similar way. Suppose that there exists a ZK-PRI denoted $P = (\mathsf{Set}, \mathsf{Gen}_1, \mathsf{Gen}_2, \mathsf{Proof}_1, \mathsf{Proof}_2, \mathsf{RGen}, \mathsf{RProof})$ that is strongly sound but not sound. We set $I_i = (\mathsf{Set}, \mathsf{Gen}_i, \mathsf{Proof}_i)$ for $i \in \{1, 2\}$. We have the following hypothesis:

1. I_i is strongly sound for $i \in \{1, 2\}$ (there exists a knowledge extractor $\mathcal{E}_i$ for I_i and I_i is secret secure).
2. P is secret secure.
3. P is not sound.

Not that I_i is sound for $i \in \{1, 2\}$ from Hypothesis 1 and Lemma 2. Since P is not sound, we deduce that there exists a PPT proxy P^* such that for $\mathscr{S} \leftarrow \mathsf{Gen}(\lambda)$, $(\mathsf{pk}_i, \mathsf{sk}_i, w_i) \leftarrow \mathsf{Gen}_i(\mathscr{S})$ for $i \in \{1, 2\}$ and $rk \leftarrow \mathsf{RGen}(\mathsf{sk}_1, w_1, \mathsf{sk}_2, w_2)$:
$\mathsf{Pr}\left[b \leftarrow \texttt{out}_V(\mathsf{Proof}_i\langle P^*(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2); V(\mathsf{pk}_i)\rangle) : b = 1\right] \geq \epsilon_1(\lambda)$
where ϵ_1 is non-negligible. Using the knowledge extractor $\mathcal{E}_i$, we have:
$\mathsf{Pr}\left[v \leftarrow \texttt{view}_P(\mathsf{Proof}\langle P^*(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2); V(\mathsf{pk})\rangle); s \leftarrow \mathcal{E}(v) : s = \mathsf{sk}\right] \geq \epsilon_2(\lambda)$
where ϵ_2 is non-negligible. We show how to build a PPT adversary $\mathcal{A}$ that breaks the secret security of P with non-negligible probability using P^* and $\mathcal{E}_i$:

- $\mathcal{A}$ receives $(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2)$ as input.
- $\mathcal{A}$ runs $v \leftarrow \texttt{view}_P(\mathsf{Proof}\langle P^*(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2); V(\mathsf{pk})\rangle)$ and $s \leftarrow \mathcal{E}(v)$.
- $\mathcal{A}$ returns s.

Then $s = \mathsf{sk}_i$ with non-negligible probability, which contradicts Hypothesis 2. □

D Security proofs of $\mathsf{B.RE\Pi}$

Theorem 2. *The* $\mathsf{B.RE\Pi}$ *scheme is bidirectional, complete, sound and honest zero-knowledge under the* DL *assumption.*

Proof (Theorem 2). We denote by I_i for $i \in \{1, 2\}$ the ZKI scheme ($\mathsf{B.Set}, \mathsf{B.Gen}_i$, Π_1). Note that these two ZKIs are the Schnorr's protocol that is complete, strongly sound and honest-verifier zero-knowledge [23]. To prove that $\mathsf{B.RE\Pi}$ is sound, we prove that it is strongly sound (Lemma 3), *i.e.* we prove that I_i is strongly sound for $i \in \{1, 2\}$ and that $\mathsf{B.RE\Pi}$ is secret secure.
Bidirectional: The scheme is bidirectional since for any re-proof keys rk and rk' such that $\mathsf{rk} \leftarrow \mathsf{RGen}(\mathsf{sk}_1, \perp, \mathsf{sk}_2, \perp)$ and $\mathsf{rk}' \leftarrow \mathsf{RGen}(\mathsf{sk}_2, \perp, \mathsf{sk}_1, \perp)$, $\mathsf{rk} = \mathsf{sk}_2/\mathsf{sk}_1$ and $1/\mathsf{rk} = \mathsf{sk}_1/\mathsf{sk}_2 = \mathsf{rk}'$.
Completeness: (i) I_1 and I_2 are complete. (ii) Suppose that a delegate Alice using

the secret sk_1, a proxy knowing the appropriate re-proof key $\mathsf{rk} = \mathsf{sk}_2/\mathsf{sk}_1$ and a verifier knowing $\mathsf{pk}_2 = g^{\mathsf{sk}_2}$ honestly run the protocol $\Pi_{1\to 1}$. Then Alice sends $R = g^r$ to the proxy, the proxy sends $S = R^{\mathsf{rk}} \cdot g^s$ to the verifier, the verifier generates the challenge c, Alice computes $\alpha = r + \mathsf{sk}_1 \cdot c$ and the proxy computes $\beta = s + \mathsf{rk} \cdot \alpha$. Then $g^\beta = g^{s+\mathsf{rk}\cdot\alpha} = g^s \cdot g^{\mathsf{rk}\cdot r} \cdot g^{\frac{\mathsf{sk}_2}{\mathsf{sk}_1}\cdot \mathsf{sk}_1 \cdot c} = S \cdot (g^{\mathsf{sk}_2})^c = S \cdot {\mathsf{pk}_2}^c$ and the verifier outputs 1.
Honest zero-knowledge: (i) I_1 and I_2 are honest-verifier zero-knowledge. (ii) Since I_1 is honest-verifier zero-knowledge, there exists a simulator $\mathtt{Sim}$ such that the outputs of Π_1 follow the same probability distribution that the outputs of $\mathtt{Sim}$. We show how to build the simulator $\mathtt{Sim}_1(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2)$: this algorithm runs $\mathtt{Sim}$ to generate the transcript (R, c, α), picks $s \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $S = R^{\mathsf{rk}} \cdot g^s$ and $\beta = s + \mathsf{rk} \cdot \alpha$. $\mathtt{Sim}_1$ outputs $\tau = (R, S, c, \alpha, \beta)$. Knowing rk, the simulator perfectly simulates the proxy behavior and the outputs of $\mathtt{Sim}_1$ follow the same distribution as the real protocol $\Pi_{1\to 1}$ (Fig. 2). (iii) We show how to build the simulator $\mathtt{Sim}_2(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{pk}_2)$: it picks $r \xleftarrow{\$} \mathbb{Z}_p^*$, $c \xleftarrow{\$} \mathbb{Z}_p^*$ and $\beta \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $R = g^r$, $\alpha = r + \mathsf{sk}_1 \cdot c$ and $S = \frac{g^\beta}{\mathsf{pk}_2^c}$. It returns the transcript $\tau = (R, S, c, \alpha, \beta)$. As τ is the transcript of a valid proof and β comes from the uniform distribution on $\mathbb{Z}_p^*$, the outputs of $\mathtt{Sim}_2$ follow the same distribution as the real protocol $\Pi_{1\to 1}$.
Secret secure: Suppose that there exists a PPT adversary $\mathcal{A}$ that breaks the secret security of $\mathsf{B.RE\Pi}$ with non-negligible probability $\epsilon(\lambda)$. We show how to build a PPT algorithm $\mathcal{B}$ that breaks DL in $(\mathbb{G}, p, g)$ with non-negligible probability $\epsilon(\lambda)/2$. $\mathcal{B}$ receives $\mathsf{pk}_0 = g^{\mathsf{sk}_0}$ for unknown sk_0. It picks $\mathsf{rk}_0 \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $\mathsf{pk}_1 = \mathsf{pk}_0^{\mathsf{rk}_0}$ and $\mathsf{rk}_1 = 1/\mathsf{rk}_0$. It picks $b \xleftarrow{\$} \{0, 1\}$, runs $\mathsf{sk} \leftarrow \mathcal{A}(\mathsf{rk}_b, \mathsf{pk}_b, \mathsf{pk}_{1-b})$ and returns sk. We set $\mathsf{rk}_0 = \mathsf{sk}_1/\mathsf{sk}_0$, then $\mathsf{pk}_1 = g^{\mathsf{sk}_1}$ and $\mathsf{rk}_1 = \mathsf{sk}_0/\mathsf{sk}_1$. If $\mathcal{A}$ wins his experiment then it returns sk_0 or sk_1 with non-negligible probability. Then $\Pr[\mathsf{sk}_0 \leftarrow \mathcal{B}(\mathsf{pk}_0)] = \epsilon(\lambda)/2$.
Soundness: $\mathsf{B.RE\Pi}$ is strongly sound since I_1 and I_2 are strongly sound and $\mathsf{B.RE\Pi}$ is secret secure.

□

E Security proofs of U.REΠ

Definition 11 (DCDH [3]). *Let $\mathbb{G}$ be a multiplicative group of prime order p and $g \in \mathbb{G}$ be a generator. Given an instance $h = (g^a, g^b)$ for unknown $a, b \xleftarrow{\$} \mathbb{Z}_p^*$, the* divisible computational Diffie-Hellman (DCDH) problem *in $(\mathbb{G}, p, g)$ is to compute $g^{a/b}$. The* divisible computational Diffie-Hellman hypothesis *states that there exists no polynomial time algorithm that solves* DCDH *with non-negligible probability. This problem is equivalent to the computational Diffie-Hellman problem [3].*

Lemma 4. *ZKIs schemes $(U.\mathsf{Set}, U.\mathsf{Gen}_1, \widetilde{\Pi}_1)$ and $(U.\mathsf{Set}, U.\mathsf{Gen}_2, \Pi_2)$ (Fig. 3) are complete, strongly sound, and honest-verifier zero-knowledge under the respective assumptions* DL *and* $\mathsf{FAPI2}$.

Proof. The proofs of the properties of $(\mathsf{U.Set}, \mathsf{U.Gen}_1, \widetilde{\Pi}_1)$ are implied by the proofs of Schnorr's protocol. The proof of $(\mathsf{U.Set}, \mathsf{U.Gen}_2, \Pi_2)$ is done in five steps:
Secret security: Since the public key is an instance of BDLV and the secret key is the corresponding solution, breaking the secret security of this ZKI is equivalent to solve the BDLV problem which is hard under the DL assumption (Lemma 1).
Completeness: Suppose that a prover P using the secret sk and a verifier knowing $\mathsf{pk} = e(g_1, \mathsf{sk})$ honestly run the protocol. P sends $S = e(g_1, g_2)^s$, receives c and computes $\beta = g_2^s \cdot \mathsf{sk}^c$. Then $e(g_1, \beta) = e(g_1, g_2^s \cdot \mathsf{sk}^c) = e(g_1, g_2^s) \cdot e(g_1, \mathsf{sk})^c = S \cdot \mathsf{pk}^c$.
Knowledge extractor: We consider a prover who is able to convince a verifier $V(\mathsf{pk})$ for several challenges with non-negligible probability, and we build a PPT knowledge extractor for this ZKI. Supposing that P is able to respond to two different challenges c_0 and c_1 for the same commitment S, then P is able to compute β_0 and β_1 such that $e(g_1, \beta_0) = S \cdot \mathsf{pk}^{c_0}$ and $e(g_1, \beta_1) = S \cdot \mathsf{pk}^{c_1}$. We have $(S, c_0, c_1 \beta_0, \beta_1) \in v$ where v is the view of P during the session. We build the following extractor $\mathcal{E}(v)$: this extractor computes $\mathsf{sk} = (\beta_1/\beta_0)^{1/(c_1-c_0)}$. We show that sk is the secret key corresponding to pk since:
$e(g_1, \mathsf{sk}) = e(g_1, \frac{\beta_1}{\beta_0})^{1/(c_1-c_0)} = \frac{e(g_1,\beta_1)}{e(g_1,\beta_0)}^{1/(c_1-c_0)} = \left(\frac{S \cdot \mathsf{pk}^{c_1}}{S \cdot \mathsf{pk}^{c_0}}\right)^{1/(c_1-c_0)} = \mathsf{pk}.$
Strong soundness: The ZKI is secret secure and it admits a knowledge extractor, then it is strongly sound (Lemma 2).
Honest zero-knowledge: We construct the following simulator $\mathtt{Sim}(\mathsf{pk})$. It picks $\beta \xleftarrow{\$} \mathbb{G}_2$ and $c \xleftarrow{\$} \mathbb{Z}_p^*$, computes $S = e(g_1, \beta)/\mathsf{pk}^c$ and the outputs $\tau = (S, c, \beta)$. As τ is the transcript of a valid proof and β and c comes from the uniform distribution on $\mathbb{Z}_p^*$, outputs of $\mathtt{Sim}$ follow the same distribution as the real protocol. □

Theorem 3. $\mathsf{U.RE\Pi}$ *is unidirectional, complete, sound and honest zero-knowledge under the* CDH*, the* DL *and the* $\mathsf{FAPI2}$ *assumptions.*

Proof (Theorem 3). We denote by I_1 and I_2 the two ZKI schemes $I_1 = (\mathsf{U.Set}, \mathsf{U.Gen}_1, \widetilde{\Pi}_1)$ and $I_2 = (\mathsf{U.Set}, \mathsf{U.Gen}_2, \Pi_2)$. To prove that $\mathsf{U.RE\Pi}$ is sound, we prove that it is strongly sound (Lemma 3), *i.e.* we prove that I_i is strongly sound for $i \in \{1, 2\}$ and that $\mathsf{U.RE\Pi}$ is secret secure.
Completeness: (i) I_1 and I_2 are complete (Lemma 4). (ii) Suppose that a delegate Alice knowing the secret sk_1, a proxy knowing the appropriate re-proof key $\mathsf{rk} = \mathsf{sk}_2^{1/\mathsf{sk}_1}$ and a verifier knowing $\mathsf{pk}_2 = e(g_1, \mathsf{sk}_2)$ honestly run the protocol RProof. Alice sends $R = g_2^r$ to the proxy, the proxy sends $S = e(R, \mathsf{rk}) \cdot e(g_1, g_2)^s$ to the verifier, the verifier generates the challenge c. Alice computes $\alpha = r + \mathsf{sk}_1 \cdot c$ and the proxy computes $\beta = g_2^s \cdot \mathsf{rk}^\alpha$. Then $e(g_1, \beta) = e(g_1, g_2)^s \cdot e(g_1, \mathsf{rk})^\alpha = e(g_1, g_2)^s \cdot e(g_1^r, \mathsf{rk}) \cdot e(g_1, \mathsf{sk}_2)^c = S \cdot e(g_1, \mathsf{sk}_2)^c = S \cdot \mathsf{pk}_2^c$ and the verifier outputs 1.
Honest zero-knowledge: (i) I_1 and I_2 are both honest-verifier zero-knowledge

(Lemma 4). (ii) Since I_1 is honest-verifier zero-knowledge, there exists a simulator $\mathtt{Sim}$ such that the outputs of $\widetilde{\Pi}_1$ follow the same probability distribution that the outputs of $\mathtt{Sim}$. We show how to build the algorithm $\mathtt{Sim}_1(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2)$: this simulator runs $\mathtt{Sim}(\mathsf{pk}_1)$ to generate the transcript (R, c, α), picks $s \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $S = e(R, \mathsf{rk}) \cdot e(g_1, g_2)^s$ and $\beta = g_2^s \cdot \mathsf{rk}^\alpha$. $\mathtt{Sim}_1$ outputs $\tau = (R, S, c, \alpha, \beta)$. Knowing rk, the simulator perfectly simulates the proxy behavior and the outputs of $\mathtt{Sim}_1$ follow the same distribution as the real protocol $\Pi_{1\to 2}$ (Fig. 4). (iii) We show how to build $\mathtt{Sim}_2(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{pk}_2)$: the simulator picks $r \xleftarrow{\$} \mathbb{Z}_p^*$, $c \xleftarrow{\$} \mathbb{Z}_p^*$ and $\beta \xleftarrow{\$} \mathbb{G}_2$, and computes $R = g_1^r$, $\alpha = r + \mathsf{sk}_1 \cdot c$ and $S = e(g_1, \beta)/\mathsf{pk}_2^c$. It returns the transcript $\tau = (R, S, c, \alpha, \beta)$. As τ is the transcript of a valid re-proof and β comes from the uniform distribution on $\mathbb{G}_2$, the outputs of $\mathtt{Sim}_2$ follow the same distribution that the real protocol $\Pi_{1\to 2}$ (Fig. 4).
Secret secure: Suppose that there exists an adversary $\mathcal{A}$ that breaks the secret security. We first show that the probability that $\mathcal{A}$ outputs the delegate's secret key sk_1 is negligible. We then show that the probability that $\mathcal{A}$ outputs the delegator's secret key sk_2 is also negligible, and we conclude that $\mathcal{A}$ has a negligible probability to break the secret security of the scheme, which contradicts our hypothesis. Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be three groups of prime order p and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a type 2 non-degenerate bilinear pairing:

– We suppose that there exists a polynomial time algorithm $\mathcal{A}$ that computes the delegate's secret key sk_1 from his public key pk_1, a re-proof key rk and the public key of the corresponding delegator pk_2. Then we show how to construct an algorithm $\mathcal{B}$ that breaks BDLV. $\mathcal{B}$ receives the challenge (h_1, h_2) that is an instance of BDLV in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$. It sets $\mathsf{pk}_1 = (h_1, h_2)$, $(\mathsf{pk}_2, \mathsf{sk}_2, w_2) \leftarrow \mathsf{Gen}_i(\mathscr{S})$ and $rk \leftarrow \mathsf{RGen}(\mathsf{pk}_1, \mathsf{sk}_2, w_2,)$. Then $\mathcal{B}$ runs $\mathsf{sk} \leftarrow \mathcal{A}(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2)$ and outputs sk. Thus if $\mathcal{A}$ wins his experiment then $h_1 = g_1^{\mathsf{sk}}$ and $h_2 = g_2^{1/\mathsf{sk}}$, and $\mathcal{B}$ outputs the correct answer. Finally, $\mathcal{B}$ breaks BDLV with non-negligible probability.
– We suppose that there exists a polynomial time algorithm $\mathcal{A}$ that computes the delegator's secret key sk_2 from his public key pk_2, a re-proof key rk and the public key of the corresponding delegate pk_1. Then we show how to construct an algorithm $\mathcal{B}$ that breaks DCDH. $\mathcal{B}$ receives the challenge (A, B) that is an instance of DCDH in $(\mathbb{G}_2, p, g_2)$. Since e is a type 2 pairing there exists a morphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$. Then $\mathcal{B}$ picks $a \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $g_1 = \phi(B^a)$ and $h_1 = \phi(g_2^a)$. He computes the delegate public key $\mathsf{pk}_1 = (h_1, B)$, the re-proof key $\mathsf{rk} = A$ and the delegator public key $\mathsf{pk}_2 = e(h_1, \mathsf{rk})$. Then $\mathcal{B}$ runs $C \leftarrow \mathcal{A}(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2)$ and outputs C. There exist sk_1 and w_2 such that $B = g_2^{1/\mathsf{sk}_1}$ and $A = g_2^{w_2/\mathsf{sk}_1}$. We observe that $h_1 = \phi(g_2^a) = \phi(B^{a\cdot\mathsf{sk}_1}) = \phi(B^a)^{\mathsf{sk}_1} = g_1^{\mathsf{sk}_1}$, $\mathsf{rk} = g_2^{w_2/\mathsf{sk}_1}$ and $e(h_1, \mathsf{rk}) = e(g_1, g_2)^{\mathsf{sk}_1 \cdot \frac{w_2}{\mathsf{sk}_1}} = e(g_1, g_2^{w_2})$. Therefore $(\mathsf{pk}_1, \mathsf{pk}_2, \mathsf{rk})$ is a valid challenge for the experiment of $\mathcal{A}$. Then, if $\mathcal{A}$ wins the experiment, $C = g_2^{\mathsf{sk}_1 \cdot \frac{w_2}{\mathsf{sk}_1}} = g_2^{w_2}$ and $\mathcal{B}$ wins the attack.

We conclude that $\mathcal{A}$ have a non-negligible probability to break the secret security of the scheme, which contradicts our hypothesis.
Soundness: $\mathsf{U.RE\Pi}$ is strongly sound since I_1 and I_2 are strongly sound (Lemma 4) and $\mathsf{U.RE\Pi}$ is secret secure. $\square$

F Security proofs of NiB.REΠ

Lemma 5. *The ZKI scheme* $(\Pi_3.\mathsf{Set}, \Pi_3.\mathsf{Gen}, \Pi_3)$ *(Fig. 5) is complete, strongly sound, and honest-verifier zero-knowledge under the* DL *assumption.*

Proof. **Completeness:** Suppose that a prover P using the secret sk and a verifier knowing $\mathsf{pk} = g^{\mathsf{sk}}$ honestly run the protocol. P sends $R = g^r$ and $U = g^u$ to the verifier, the verifier generates the challenge (c, d), P computes $z = r + \mathsf{sk} \cdot d$ and $\mu = u + r \cdot c$. Then $g^z = g^r \cdot g^{\mathsf{sk} \cdot d} = R \cdot \mathsf{pk}^d$ and $g^\mu = g^u \cdot g^{r \cdot c} = U \cdot R^c$ and the verifier outputs 1.
Secret security: Since the public key is an instance of DL and the secret key is the corresponding solution, breaking the secret security of this ZKI is equivalent to break the DL assumption.
Knowledge extractor: We consider a prover who is able to convince a verifier $V(\mathsf{pk})$ for several challenges with non-negligible probability, and we build a PPT knowledge extractor for this ZKI. Supposing that P is able to respond to two different challenges (c_0, d_0) and (c_1, d_1) for the same commitment (R, U). Then P is able to compute z_0 and z_1 such that $g^{z_0} = R \cdot \mathsf{pk}^{d_0}$ and $g^{z_1} = R \cdot \mathsf{pk}^{d_1}$. We have $(R, U, c_0, c_1, z_0, z_1) \in v$ where v is the view of P during the session. We build the following extractor $\mathcal{E}(v)$: this extractor computes $\mathsf{sk} = \frac{z_1 - z_0}{d_1 - d_0}$. We show that sk is the secret key corresponding to pk since:
$g^{\mathsf{sk}} = g^{(z_1 - z_0)/(d_1 - d_0)} = \left(\frac{g^{z_1}}{g^{z_0}}\right)^{1/(d_1 - d_0)} = \left(\frac{R \cdot \mathsf{pk}^{d_1}}{R \cdot \mathsf{pk}^{d_0}}\right)^{1/(d_1 - d_0)} = \mathsf{pk}.$
Strong soundness: The scheme is secret secure and it admits a knowledge extractor, then it is strongly sound (Lemma 2).
Honest zero-knowledge: We construct the following simulator $\mathtt{Sim}(\mathsf{pk})$. It picks $z \xleftarrow{\$} \mathbb{Z}_p^*$, $\mu \xleftarrow{\$} \mathbb{Z}_p^*$, $c \xleftarrow{\$} \mathbb{Z}_p^*$ and $d \xleftarrow{\$} \mathbb{Z}_p^*$, and computes $R = g^z / \mathsf{pk}^d$ and $U = g^\mu / R^c$. It outputs $\tau = (R, U, c, d, z, \mu)$. As τ is the transcript of a valid proof and z and μ come from the uniform distribution on $\mathbb{Z}_p^*$, the outputs of $\mathtt{Sim}$ follow the same distribution as the real protocol. $\square$

Lemma 6. *The ZKI scheme* $(\Pi_3.\mathsf{Set}, \Pi_4.\mathsf{Gen}, \Pi_4)$ *(Fig. 6) is complete, strongly sound, and honest-verifier zero-knowledge under the* DL *assumption.*

Proof. **Completeness:** Suppose that a prover P knowing the secret sk and a verifier knowing $\mathsf{pk} = g^{\mathsf{sk}}$ honestly runs the protocol. Then, using the three challenges c, d and f, P generates the following values during the protocol: $a, b, r, s, t, u \xleftarrow{\$} \mathbb{Z}_p^*$, $R = g^r$, $U = g^u$, $\mu = u + r \cdot c$, $S = g^{r \cdot t + s}$, $A = R^a$, $B = g^b$, $\alpha = a + t \cdot f$,

$\beta = b + s \cdot f$ and $\theta = s + r \cdot t + \mathsf{sk} \cdot d$. Then, $g^{\mu} = g^{u} \cdot g^{r \cdot c} = U \cdot R^{c}$, $R^{\alpha} \cdot g^{\beta} = R^{a} \cdot g^{b} \cdot R^{t \cdot f} \cdot g^{s \cdot f} = A \cdot B \cdot S^{f}$ and $g^{\theta} = g^{s+r \cdot t} \cdot g^{y \cdot d} = S \cdot \mathsf{pk}^{d}$. Then the verifier outputs 1.

Secret security: As in Lemma 5.

Knowledge extractor: We consider a prover who is able to convince a verifier $V(\mathsf{pk})$ for several challenges with non-negligible probability, and we build a PPT knowledge extractor for this ZKI. Supposing that P is able to perform the proof for any two different first challenges (c_0, d_0) and (c_1, d_1) using the same commitments R and U and:

- any two second challenges $f_{0,0}$ and $f_{0,1}$ using the same first challenge (c_0, d_0) and the same commitments (R, U, S_0, A_0, B_0);
- any two second challenges $f_{1,0}$ and $f_{1,1}$ using the same first challenge (c_1, d_1) and the same commitments (R, U, S_1, A_1, B_1).

Then P is able to compute μ_0, μ_1, $\alpha_{0,0}$, $\beta_{0,0}$, $\theta_{0,0}$, $\alpha_{0,1}$, $\beta_{0,1}$, $\theta_{0,1}$, $\alpha_{1,0}$, $\beta_{1,0}$, $\theta_{1,0}$, $\alpha_{1,1}$, $\beta_{1,1}$ and $\theta_{1,1}$ such that:

$$g^{\mu_0} = U \cdot R^{c_0} \tag{1}$$
$$g^{\mu_1} = U \cdot R^{c_1} \tag{2}$$
$$g^{\theta_{0,0}} = S_0 \cdot \mathsf{pk}^{d_0} \tag{3}$$
$$g^{\theta_{0,1}} = S_0 \cdot \mathsf{pk}^{d_0} \tag{4}$$
$$g^{\theta_{1,0}} = S_1 \cdot \mathsf{pk}^{d_1} \tag{5}$$
$$g^{\theta_{1,1}} = S_1 \cdot \mathsf{pk}^{d_1} \tag{6}$$
$$R^{\alpha_{0,0}} \cdot g^{\beta_{0,0}} = A_0 \cdot B_0 \cdot S_0^{f_{0,0}} \tag{7}$$
$$R^{\alpha_{0,1}} \cdot g^{\beta_{0,1}} = A_0 \cdot B_0 \cdot S_0^{f_{0,1}} \tag{8}$$
$$R^{\alpha_{1,0}} \cdot g^{\beta_{1,0}} = A_1 \cdot B_1 \cdot S_1^{f_{1,0}} \tag{9}$$
$$R^{\alpha_{1,1}} \cdot g^{\beta_{1,1}} = A_1 \cdot B_1 \cdot S_1^{f_{1,1}} \tag{10}$$

We have $(R$, U, S_0, A_0, B_0, $\alpha_{0,0}$, $\beta_{0,0}$, $\theta_{0,0}$, $\alpha_{0,1}$, $\beta_{0,1}$, $\theta_{0,1}$, S_1, A_1, B_1, $\alpha_{1,0}$, $\beta_{1,0}$, $\theta_{1,0}$, $\alpha_{1,1}$, $\beta_{1,1}$, $\theta_{1,1}) \in v$ where v is the view of P during the session. We build the following extractor $\mathcal{E}(v)$: first it sets $r = \frac{\mu_0 - \mu_1}{c_0 - c_1}$. Using equations (1) and (2), we have $g^{r} = g^{\frac{\mu_0 - \mu_1}{c_0 - c_1}} = \left(\frac{U \cdot R^{c_0}}{U \cdot R^{c_1}}\right)^{\frac{1}{c_0 - c_1}} = R$. Using equations (7), (8), (9) and (10), we have $R^{\frac{\alpha_{0,0} - \alpha_{0,1}}{f_{0,0} - f_{0,1}}} \cdot g^{\frac{\beta_{0,0} - \beta_{0,1}}{f_{0,0} - f_{0,1}}} = S_0$ and $R^{\frac{\alpha_{1,0} - \alpha_{1,1}}{f_{1,0} - f_{1,1}}} \cdot g^{\frac{\beta_{1,0} - \beta_{1,1}}{f_{1,0} - f_{1,1}}} = S_1$. Then, $\mathcal{E}$ computes $s_0 = r \cdot \frac{\alpha_{0,0} - \alpha_{0,1}}{f_{0,0} - f_{0,1}} + \frac{\beta_{0,0} - \beta_{0,1}}{f_{0,0} - f_{0,1}}$ and $s_1 = r \cdot \frac{\alpha_{1,0} - \alpha_{1,1}}{f_{1,0} - f_{1,1}} + \frac{\beta_{1,0} - \beta_{1,1}}{f_{1,0} - f_{1,1}}$. We have $g^{s_0} = S_0$ and $g^{s_1} = S_1$. Equations (3), (4), (5), (6) imply that $\theta_{0,0} = \theta_{0,1}$. We set $\theta_0 = \theta_{0,0} = \theta_{0,1}$. Similarly, $\theta_1 = \theta_{1,0} = \theta_{1,1}$. We deduce that $g^{\frac{\theta_0 - \theta_1}{d_0 - d_1}} = \left(\frac{S_0}{S_1}\right)^{\frac{1}{d_0 - d_1}} \cdot \mathsf{pk}$. Finally, the extractor $\mathcal{E}$ returns the value $\mathsf{sk} = \frac{\theta_0 - \theta_1}{d_0 - d_1} + \frac{s_1 - s_0}{d_0 - d_1}$. We show that sk is the secret key corresponding to pk since $g^{\mathsf{sk}} = g^{\frac{\theta_0 - \theta_1}{d_0 - d_1} + \frac{s_1 - s_0}{d_0 - d_1}} = \left(\frac{S_0}{S_1}\right)^{\frac{1}{d_0 - d_1}} \cdot \mathsf{pk} \cdot \left(\frac{S_1}{S_0}\right)^{\frac{1}{d_0 - d_1}} = \mathsf{pk}$

Strong soundness: The scheme is secret secure and it admits a knowledge extractor, then it is strongly sound (Lemma 2).

Honest zero-knowledge: We construct the following simulator $\mathtt{Sim}(\mathsf{pk})$. It picks $c, d, f, \mu, \alpha, \beta, \theta \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ and $R, A \overset{\$}{\leftarrow} \mathbb{G}$. Then it computes $S = g^{\theta}/\mathsf{pk}^{d}$, $U = g^{\mu}/R^{c}$ and $B = (R^{\alpha} \cdot g^{\beta})/(A \cdot S^{f})$. It outputs the transcript $\tau = (R, U, c, d, f, \mu, S, A, B, \alpha, \beta, \theta)$. As τ is the transcript of a valid proof and μ, α, β and θ come from the uniform distribution on $\mathbb{Z}_p^*$, outputs of $\mathtt{Sim}$ follow the same distribution as the real protocol. □

Lemma 7. $\mathsf{NI}.\Pi_3$ *and* $\mathsf{NI}.\Pi_4$ *are non-interactive, complete, sound, and zero-knowledge under the* DL *assumption in the random oracle model.*

Proof. This lemma is a direct implication of Lemma 5 and Lemma 6. □

Theorem 4. $\mathsf{NiB.RE\Pi}$ *scheme is bidirectional, complete, sound and zero-knowledge in the random oracle model under the* DL *assumption.*

Proof (Theorem 4). Bidirectionality and secret security proofs are similar to the scheme $\mathsf{B.RE\Pi}$ (proof of Theorem 2). We set $I_1 = (\Pi_3.\mathsf{Set}, \Pi_3.\mathsf{Gen}, \Pi_3.\mathsf{Prove}, \Pi_3.\mathsf{Verify})$ and $I_2 = (\Pi_3.\mathsf{Set}, \Pi_4.\mathsf{Gen}, \Pi_4.\mathsf{Prove}, \Pi_4.\mathsf{Verify})$. Note that $I_1 = \mathsf{NI}.\Pi_3$ and $I_2 = \mathsf{NI}.\Pi_4$.
Completeness: (i) I_1 and I_2 are complete according to the Lemma 7. (ii) Let π_1 be a proof of $\mathsf{NI}.\Pi_3$ outputted by $\Pi_3.\mathsf{Prove}(\mathsf{pk}_1, \mathsf{sk}_1)$ which is computed as follows: we pick $r \xleftarrow{\$} \mathbb{Z}_p^*$ and $u \xleftarrow{\$} \mathbb{Z}_p^*$, we compute $R = g^r$, $U = g^u$, $c = H(R, U, 0)$, $d = H(R, U, 1)$, $z = r + \mathsf{sk}_1 \cdot c$ and $\mu = u + r \cdot c$. Finally, $\pi_1 = (R, U, z, \mu)$. Using $\mathsf{rk} = \mathsf{sk}_2/\mathsf{sk}_1$, the algorithm $\mathsf{NiB.RProve}(\mathsf{rk}, \pi_1)$ computes π_2 as follows: it picks s, a and b in the uniform distribution on $\mathbb{Z}_p^*$ and it computes $S = R^{\mathsf{rk}} \cdot g^s$, $A = R^a$, $B = g^b$, $f = H(R, U, A, B, S)$, $\alpha = a + \mathsf{rk} \cdot f$, $\beta = b + s \cdot f$ and $\theta = s + z \cdot \mathsf{rk}$. Then $\pi_2 = (R, U, \mu, A, B, S, \alpha, \beta, \theta)$. Thus, using $\mathsf{pk}_2 = g^{\mathsf{sk}_2}$, we have: $g^\mu = g^u \cdot g^{r \cdot c} = U \cdot R^c$; $R^\alpha \cdot g^\beta = R^a \cdot g^b \cdot R^{\mathsf{rk} \cdot f} \cdot g^{s \cdot f} = A \cdot B \cdot S^f$; $g^\theta = g^{s + r \cdot \mathsf{rk}} \cdot g^{\mathsf{sk}_2 \cdot d} = S \cdot \mathsf{pk}_2^d$. Then the $\Pi_4.\mathsf{Verify}(\mathsf{pk}_2, \pi_2)$ outputs 1.
Soundness: $\mathsf{U.RE\Pi}$ is strongly sound since I_1 and I_2 are strongly sound (Lemma 4) and $\mathsf{U.RE\Pi}$ is secret secure.
Zero-knowledge: (i) $\mathsf{NI}.\Pi_3$ and $\mathsf{NI}.\Pi_4$ are zero-knowledge (Lemma 7). (ii) Since $\mathsf{NI}.\Pi_3$ is zero-knowledge, there exist a simulator $\mathtt{Sim}$ such that the outputs of $\Pi_3.\mathsf{Prove}$ follow the same probability distribution that the outputs of $\mathtt{Sim}$. We show how to build $\mathtt{Sim}_1(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2)$: this simulator runs $\mathtt{Sim}(\mathsf{pk}_1)$ to generate the proof π_1 and computes a second proof $\pi_2 = \mathsf{NiB.RProve}(\mathsf{rk}, \pi_1)$. Then $\mathtt{Sim}_1$ outputs $\tau = (\pi_1, \pi_2)$. Knowing rk, the simulator perfectly simulates the proxy behavior and the outputs of $\mathtt{Sim}_1$ follow the same distribution as the real algorithms Prove_1 and RProof. (iii) We show how to build $\mathtt{Sim}_2(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{pk}_2)$: this simulator picks $z, c, d, f, \mu, \alpha, \beta, \theta \xleftarrow{\$} \mathbb{Z}_p^*$ and $A \xleftarrow{\$} \mathbb{G}$. Then, using the delegate instance pk_1 and the delegator instance pk_2, it computes $R = g^z/\mathsf{pk}_1^d$, $S = g^\theta/\mathsf{pk}_2^d$, $U = g^\mu/R^c$ and $B = (R^\alpha \cdot g^\beta)/(A \cdot S^f)$. It outputs the transcript $\tau = (\pi_1, \pi_2)$ such that $\pi_1 = (R, U, c, d, \mu, z)$ and $\pi_2 = (R, U, c, d, \mu, z, S, A, B, f, \alpha, \beta, \theta)$. As τ is the transcript of a valid re-proof and μ, α, β, and θ come from the uniform distribution on $\mathbb{Z}_p^*$, the outputs of $\mathtt{Sim}_2$ follow the same distribution as the real algorithms Prove_1 and RProof. □

G Security proofs of $\mathsf{NiU.RE\Pi}$

Lemma 8. *The ZKI schemes* $(\Pi_5.\mathsf{Set}, U.\mathsf{Gen}_1, \widetilde{\Pi}_3)$ *and* $(\Pi_5.\mathsf{Set}, U.\mathsf{Gen}_2, \Pi_5)$ *(Fig. 7) are complete, strongly sound, and honest zero-knowledge under the respective assumptions* BDLV *and* $\mathsf{FAPI2}$.

Proof. The proofs for $(\Pi_5.\mathsf{Set}, \mathsf{U.Gen}_1, \widetilde{\Pi}_3)$ are similar to the proofs given in Lemma 5. We show that $(\Pi_5.\mathsf{Set}, \mathsf{U.Gen}_2, \Pi_5)$ is complete, strongly sound (it admits a knowledge extractor and it is secret secure), and honest-verifier zero-knowledge:
Completeness: Suppose that a prover $P(\mathsf{sk})$ and a $V(\mathsf{pk})$ where $\mathsf{pk} = e(g_1, \mathsf{sk})$ honestly run the protocol. Then, using the three challenges c, d and f, P generates the following values during the protocol: $a, b, r, s, t, u \xleftarrow{\$} \mathbb{Z}_p^*$, $R = g_1^r$, $U = g_1^u$, $\mu = u + r \cdot c$, $S = e(g_1, g_2)^{r \cdot t + s}$, $A = e(R, g_2)^a$, $B = e(g_1, g_2)^b$, $\alpha = g_2^{a + t \cdot f}$, $\beta = g_2^{b + s \cdot f}$ and $\theta = g_2^{s + r \cdot t} \cdot \mathsf{sk}^d$. Then, $g_1^\mu = g_1^u \cdot g_1^{r \cdot c} = U \cdot R^c$, $e(R, \alpha) \cdot e(g_1, \beta) = e(R, g_2)^a \cdot e(R, g_2)^{t \cdot f} \cdot e(g_1, g_2)^b \cdot e(g_1, g_2)^{s \cdot f} = A \cdot B \cdot S^f$ $e(g_1, \theta) = e(g_1, g_2)^{s + r \cdot t} \cdot e(g_1, \mathsf{sk})^d = S \cdot \mathsf{pk}^d$. Then the verifier outputs 1.
Secret security: Since the public key is an instance of BDLV and the secret key is the corresponding solution, breaking the secret security of this ZKI is equivalent to break the BDLV assumption.
Knowledge extractor: We consider a prover who is able to convince a verifier $V(\mathsf{pk})$ for several challenges with non-negligible probability, and we build a PPT knowledge extractor for this ZKI. Supposing that P is able to perform the proof for any two different first challenges (c_0, d_0) and (c_1, d_1) using the same commitments R and U and:

- any two second challenges $f_{0,0}$ and $f_{0,1}$ using the same first challenge (c_0, d_0) and the same commitments (R, U, S_0, A_0, B_0);
- any two second challenges $f_{1,0}$ and $f_{1,1}$ using the same first challenge (c_1, d_1) and the same commitments (R, U, S_1, A_1, B_1).

Then P is able to compute μ_0, μ_1, $\alpha_{0,0}$, $\beta_{0,0}$, $\theta_{0,0}$, $\alpha_{0,1}$, $\beta_{0,1}$, $\theta_{0,1}$, $\alpha_{1,0}$, $\beta_{1,0}$, $\theta_{1,0}$, $\alpha_{1,1}$, $\beta_{1,1}$ and $\theta_{1,1}$ such that:

$$g_1^{\mu_0} = U \cdot R^{c_0} \quad (11) \qquad e(g_1, \theta_{1,1}) = S_1 \cdot \mathsf{pk}^{d_1} \quad (16)$$
$$g_1^{\mu_1} = U \cdot R^{c_1} \quad (12) \qquad e(R, \alpha_{0,0}) \cdot e(g_1, \beta_{0,0}) = A_0 \cdot B_0 \cdot S_0^{f_{0,0}} \quad (17)$$
$$e(g_1, \theta_{0,0}) = S_0 \cdot \mathsf{pk}^{d_0} \quad (13) \qquad e(R, \alpha_{0,1}) \cdot e(g_1, \beta_{0,1}) = A_0 \cdot B_0 \cdot S_0^{f_{0,1}} \quad (18)$$
$$e(g_1, \theta_{0,1}) = S_0 \cdot \mathsf{pk}^{d_0} \quad (14) \qquad e(R, \alpha_{1,0}) \cdot e(g_1, \beta_{1,0}) = A_1 \cdot B_1 \cdot S_1^{f_{1,0}} \quad (19)$$
$$e(g_1, \theta_{1,0}) = S_1 \cdot \mathsf{pk}^{d_1} \quad (15) \qquad e(R, \alpha_{1,1}) \cdot e(g_1, \beta_{1,1}) = A_1 \cdot B_1 \cdot S_1^{f_{1,1}} \quad (20)$$

We have $(R, U, S_0, A_0, B_0, \alpha_{0,0}, \beta_{0,0}, \theta_{0,0}, \alpha_{0,1}, \beta_{0,1}, \theta_{0,1}, S_1, A_1, B_1, \alpha_{1,0}, \beta_{1,0}, \theta_{1,0}, \alpha_{1,1}, \beta_{1,1}, \theta_{1,1}) \in v$ where v is the view of P during the session. We build the following extractor $\mathcal{E}(v)$: first it sets $r = \frac{\mu_0 - \mu_1}{c_0 - c_1}$. Using equations (11) and (12), we have: $g_1^r = g_1^{\frac{\mu_0 - \mu_1}{c_0 - c_1}} = \left(\frac{U \cdot R^{c_0}}{U \cdot R^{c_1}}\right)^{\frac{1}{c_0 - c_1}} = R$. Using equations (17), (18), (19) and

(20), we have: $\left(e\left(R, \frac{\alpha_{0,0}}{\alpha_{0,1}}\right) \cdot e\left(g_1, \frac{\beta_{0,0}}{\beta_{0,1}}\right)\right)^{\frac{1}{f_{0,0}-f_{0,1}}} = \left(\frac{A_0 \cdot B_0 \cdot S_0^{f_{0,0}}}{A_0 \cdot B_0 \cdot S_0^{f_{0,1}}}\right)^{\frac{1}{f_{0,0}-f_{0,1}}} =$ S_0 and $\left(e\left(R, \frac{\alpha_{1,0}}{\alpha_{1,1}}\right) \cdot e\left(g_1, \frac{\beta_{1,0}}{\beta_{1,1}}\right)\right)^{\frac{1}{f_{1,0}-f_{1,1}}} = \left(\frac{A_1 \cdot B_1 \cdot S_1^{f_{1,0}}}{A_1 \cdot B_1 \cdot S_1^{f_{1,1}}}\right)^{\frac{1}{f_{1,0}-f_{1,1}}} = S_1$. Then, $\mathcal{E}$ sets $s_0 = \left(\frac{\alpha_{0,0}}{\alpha_{0,1}}\right)^{\frac{r}{f_{0,0}-f_{0,1}}} \cdot \left(\frac{\beta_{0,0}}{\beta_{0,1}}\right)^{\frac{1}{f_{0,0}-f_{0,1}}}$ and $s_1 = \left(\frac{\alpha_{1,0}}{\alpha_{1,1}}\right)^{\frac{r}{f_{1,0}-f_{1,1}}} \cdot \left(\frac{\beta_{1,0}}{\beta_{1,1}}\right)^{\frac{1}{f_{1,0}-f_{1,1}}}$. We have $e(g_1, s_0) = S_0$ and $e(g_1, s_1) = S_1$. Equations (13), (14), (15), (16) imply that $\theta_{0,0} = \theta_{0,1}$. We set $\theta_0 = \theta_{0,0} = \theta_{0,1}$. Similarly, $\theta_1 = \theta_{1,0} = \theta_{1,1}$. We have: $e\left(g_1, \left(\frac{\theta_0}{\theta_1}\right)^{\frac{1}{d_0-d_1}}\right) = \left(\frac{S_0}{S_1}\right)^{\frac{1}{d_0-d_1}} \cdot \mathsf{pk}$. Finally, $\mathcal{E}(v)$ computes and returns $\mathsf{sk} = \left(\frac{\theta_0 \cdot s_1}{\theta_1 \cdot s_0}\right)^{\frac{1}{d_0-d_1}}$. We show that sk is the secret key corresponding to pk since: $e(g_1, \mathsf{sk}) = e\left(g_1, \left(\frac{\theta_0 \cdot s_1}{\theta_1 \cdot s_0}\right)\right)^{\frac{1}{d_0-d_1}} = \left(\frac{e(g_1,\theta_0) \cdot e(g_1,s_1)}{e(g_1,\theta_1) \cdot e(g_1,s_0)}\right)^{\frac{1}{d_0-d_1}} = \left(\frac{S_0 \cdot \mathsf{pk}^{d_0} \cdot S_1}{S_1 \cdot \mathsf{pk}^{d_1} \cdot S_0}\right)^{\frac{1}{d_0-d_1}}$ $= \mathsf{pk}$.

Strong soundness: The scheme is secret secure and it admits a knowledge extractor, then it is strongly sound (Lemma 2).

Honest zero-knowledge: We construct the following simulator $\mathtt{Sim}(\mathsf{pk})$. It picks $c, d, f, \mu, \alpha, \beta, \theta \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ and $R, A \overset{\$}{\leftarrow} \mathbb{G}$. Then it computes $S = e(g_1, \theta)/\mathsf{pk}^d$, $U = g_1^{\mu}/R^c$ and $B = (e(R, \alpha) \cdot e(g_1, \beta))/(A \cdot S^f)$. It outputs $\tau = (R, U, c, d, f, \mu, S, A, B, \alpha, \beta, \theta)$. As τ is the transcript of a valid proof and μ, α, β and θ come from uniform distributions, outputs of $\mathtt{Sim}$ follow the same distribution as the real protocol. $\square$

Lemma 9. $\mathsf{NI}.\widetilde{\Pi}_3$ *and* $\mathsf{NI}.\Pi_5$ *are non-interactive, complete, sound, and zero-knowledge under the respective assumptions* BDLV *and* $\mathsf{FAPI2}$ *in the random oracle model.*

Proof. This lemma is a direct implication of Lemma 8. $\square$

Theorem 5. $\mathsf{NiU.RE\Pi}$ *is unidirectional, complete, sound and zero-knowledge in the random oracle model under the* DL*, the* CDH *and the* $\mathsf{FAPI2}$ *assumptions.*

Proof (Theorem 5). The scheme is unidirectional by construction. Secret security proof is similar to the scheme $\mathsf{U.RE\Pi}$ (proof of Theorem 3) since it uses the same key generation algorithms. We set the two ZKI schemes $I_1 = (\Pi_5.\mathsf{Set}, \mathsf{U.Gen}_1, \widetilde{\Pi}_3.\mathsf{Prove}, \widetilde{\Pi}_3.\mathsf{Verify})$ and $I_2 = (\Pi_5.\mathsf{Set}, \mathsf{U.Gen}_2, \Pi_5.\mathsf{Prove}, \Pi_5.\mathsf{Verify})$. Note that $I_1 = \mathsf{NI}.\widetilde{\Pi}_3$ and $I_2 = \mathsf{NI}.\Pi_5$.

Completeness: (i) I_1 and I_2 are complete (Lemma 9). (ii) Let π_1 be a proof of $\mathsf{NI}.\Pi_3$ which is outputted by $\widetilde{\Pi}_3.\mathsf{Prove}(\mathsf{pk}_1, \mathsf{sk}_1)$ computed as follows: we pick $r \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ and $u \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ and compute $R = g_1^r$, $U = g_1^u$, $c = H(R, U, 0)$, $d = H(R, U, 1)$, $z = r + \mathsf{sk}_1 \cdot d$ and $\mu = u + r \cdot c$. Finally, $\pi = (R, U, z, \mu)$. Using $\mathsf{rk} = \mathsf{sk}_2^{1/\mathsf{sk}_1}$, the algorithm $\mathsf{NiU.RProve}(\mathsf{rk}, \pi_1)$ computes π_2 as follows: it picks s, a and b in the

uniform distribution on $\mathbb{Z}_p^*$ and computes $S = e(R, \mathsf{rk}) \cdot e(g_1, g_2)^s$, $A = e(R, g_2)^a$, $B = e(g_1, g_2)^b$, $f = H(R, U, A, B, S)$, $\alpha = g_2^a \cdot \mathsf{rk}^f$, $\beta = g_2^{b+s\cdot f}$ and $\theta = g_2^s \cdot \mathsf{rk}^z$. It outputs $\pi_2 = (R, U, \mu, A, B, S, \alpha, \beta, \theta)$. Thus, using $\mathsf{pk}_2 = e(g_1, \mathsf{sk}_2)$, we have: $g_1^\mu = g_1^u \cdot g_1^{r\cdot c} = U \cdot R^c$; $e(R, \alpha) \cdot e(g_1, \beta) = e(R, g_2)^a \cdot e(g_1, g_2)^b \cdot e(R, \mathsf{rk})^f \cdot e(g_1, g_2)^{s\cdot f} = A \cdot B \cdot S^f$; $e(g_1, \theta) = e(g_1, g_2)^s \cdot e(g_1, \mathsf{rk})^z = e(g_1, g_2)^s \cdot e(g_1, \mathsf{rk})^r \cdot e(g_1, \mathsf{sk}_2)^c = S \cdot \mathsf{pk}_2^d$. Then $\Pi_5.\mathsf{Verify}(\mathsf{pk}_2, \pi_2)$ outputs 1.
Soundness: $\mathsf{U.RE\Pi}$ is strongly sound since I_1 and I_2 are strongly sound (Lemma 4) and $\mathsf{U.RE\Pi}$ is secret secure.
Zero-knowledge: (i) $\mathsf{NI.}\widetilde{\Pi}_3$ and $\mathsf{NI.\Pi}_5$ are zero-knowledge (Lemma 9). (ii) Since $\mathsf{NI.}\widetilde{\Pi}_3$ is zero-knowledge, there exists a simulator $\mathtt{Sim}$ such that outputs of $\widetilde{\Pi}_3.\mathsf{Prove}$ follow the same probability distribution that the outputs of $\mathtt{Sim}$. We show how to build $\mathtt{Sim}_1(\mathsf{rk}, \mathsf{pk}_1, \mathsf{pk}_2)$: this simulator runs $\mathtt{Sim}$ to generate the proof π_1 and computes a second proof $\pi_2 = \mathsf{NiU.RProve}(\mathsf{rk}, \pi_1)$. $\mathtt{Sim}_1$ outputs $\tau = (\pi_1, \pi_2)$. Knowing rk, the simulator perfectly simulates the proxy behavior and the outputs of $\mathtt{Sim}_1$ follow the same distribution as the real algorithms Prove_1 and RProof. (iii) We show how to build $\mathtt{Sim}_2(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{pk}_2)$: this simulator picks $z, c, d, f, \mu, \alpha, \beta, \theta \xleftarrow{\$} \mathbb{Z}_p^*$ and $A \xleftarrow{\$} \mathbb{G}$. Then, using the delegate instance pk_1 and the delegator instance pk_2, it computes $R = g_1^z/\mathsf{pk}_1^d$, $S = e(g_1, \theta)/\mathsf{pk}_2^d$, $U = g_1^\mu/R^c$ and $B = (e(R, \alpha) \cdot e(g_1, \beta))/(A \cdot S^f)$. It outputs $\tau = (\pi_1, \pi_2)$ using the two non-interactive proofs $\pi_1 = (R, U, c, d, \mu, z)$ and $\pi_2 = (R, U, c, d, \mu, z, S, A, B, f, \alpha, \beta, \theta)$. As τ is the transcript of a valid re-proof and μ, α, β and θ come from uniform distributions, the outputs of $\mathtt{Sim}_2$ follow the same distribution as the real algorithms Prove_1 and RProof. $\square$