Encrypting Proofs on Pairings
and Its Application to Anonymity for Signatures

Georg Fuchsbauer David Pointcheval

École normale supérieure, LIENS - CNRS - INRIA, Paris, France
http://www.di.ens.fr/{~fuchsbau,~pointche}

Abstract

We give a generic methodology to unlinkably anonymize cryptographic schemes in bilinear groups using the Boneh-Goh-Nissim cryptosystem and NIZK proofs in the line of Groth, Ostrovsky and Sahai. We illustrate our techniques by presenting the first instantiation of anonymous proxy signatures, a recent primitive unifying the functionalities and strong security notions of group and proxy signatures. To construct our scheme, we introduce various efficient NIZK and witness-indistinguishable proofs, and a relaxed version of simulation soundness.

Keywords: NIZK, bilinear groups, anonymity, unlinkability, group signatures, non-frameability.

1 Introduction

One of the major concerns of modern cryptography is anonymity. Group signatures [CvH91] for example allow members to sign on behalf of a group while remaining anonymous. Other concepts to which anonymity is central are hierarchical group signatures [TW05], identity escrow [KP98] and anonymous credentials [Cha85], to mention only a few. The main issue of these concepts is to demonstrate that a user is entitled to perform a respective task, while not revealing anything about his identity. Zero-knowledge proofs provide the means to do so: prove something without leaking any further information. In particular, non-interactive zero-knowledge (NIZK) proofs [BFM88] have enjoyed numerous applications to achieve anonymity.

Substantial progress has been made in recent years in making NIZK proofs efficient and thus applicable to practical schemes: Groth et al. [GOS06b] show how to efficiently non-interactively prove that a BGN-ciphertext [BGN05] (cf. Sect. 2) encrypts 0 or 1. Although conceived for purely theoretical purposes, their techniques were used by Boyen and Waters [BW06] to construct compact group signatures, which they improve in [BW07].

In a different line of research—which has been unified with the one based on BGN in [GS08]—, Groth et al. [GOS06a] based NIZK proofs on a commitment scheme building on *linear encryption* [BBS04]. The latter is an extension of ElGamal encryption to bilinear groups[1] and is semantically secure under the *decisional linear assumption* (DLIN). Keys for GOS-commitments are basically linear encryptions of either 0 or 1, while the encrypted value determines whether the resulting commitments are perfectly hiding or perfectly binding. Since both types of keys are indistinguishable by DLIN, they inherit a computational version of the other's property from one another.

[1] The decisional Diffie-Hellman assumption (DDH), on which ElGamal relies, does not hold in symmetric bilinear groups.

This scheme has given rise to a multitude of practical NIZK proof systems (see e.g. the full version of [Gro06] for an impressive demonstration of its power), practical implementations of fully secure group signatures [Gro07] without random oracles [BR93], as well as the introduction of new primitives such as non-interactive anonymous credentials in [BCKL08].

Our Contributions. All the above analyses required ad-hoc security proofs. When extending anonymity to more complex protocols, the security proofs easily become too intricate—unless one manages to provide a generic way to anonymize a large class of proofs. Such a generic anonymization is our first contribution; in Sect. 3, we generalize the ideas of [BW06, BW07] to BGN-encrypt proofs (and in particular signatures) and prove validity of the encrypted values, for the following category of schemes: The relations checked by the verification algorithm are equations consisting exclusively of products of pairings (as is the case for most signature schemes in bilinear groups such as Boneh-Boyen's short signatures [BB04] or Waters' scheme [Wat05]).

We give a methodology to construct proofs demonstrating that encrypted values satisfy certain relations, and show that these proofs do not leak information on the plaintexts, nor relations about the plaintexts—providing thus anonymity (unlinkability and untraceability). Moreover, given ciphertexts and a corresponding proof, then without knowledge of the plaintexts, one can *re-encrypt* (or re-randomize) the ciphertexts and adapt the proof to the new encryptions. In particular, re-randomizations of two sets of ciphertexts and proofs are indistinguishable. This yields a generic method to anonymize schemes in an unlinkable way, such as group signatures ("full anonymity" of the schemes in [BW06] and [BW07] is an immediate consequence of our results), fair contract signing, or verifiable encryption, as shown in Sect. 3.2. Since we use encryption to achieve anonymity, the decryption key provides a trapdoor to revoke anonymity in case of abuse.

In order to illustrate our methodology and to demonstrate its power, our second contribution is the first concrete implementation of *anonymous proxy signatures* in the standard model. This primitive was recently introduced by Fuchsbauer and Pointcheval [FP08], who while giving practical applications merely prove theoretical feasibility. It merges group signatures with proxy signatures [MUO96], generalizing the strong security notions of both (in particular, [BMW03, BSZ05] for group signatures and [BPW03] for proxy signatures). Proxy signatures allow consecutive delegation of signing rights and publicly provide the chain of delegators with the signed document. Now anonymity for this primitive means that the delegation chain remains hidden: While nobody knows who actually signed, anyone can verify that the proxy signer was indeed entitled to do so. As usual, traceability deters from misuse, by making anonymity revocable.

We slightly simplify the model from [FP08], in that we consider only one general opener (instead of having each user choose his own) and anonymity against adversaries without opening oracles (a.k.a. CPA-anonymity [BBS04]). Furthermore, we introduce a maximal number of possible delegations. We emphasize that this variant still directly yields *dynamic hierarchical group signatures* that satisfy *non-frameability* (while [BW07] only consider the static and non-hierarchical case).

Before presenting our full scheme in Sect. 5, we mainly focus on constructing a (non-anonymous) scheme for consecutive signature delegations (Sect. 4) to which our methodology can then readily be applied. Its main building block is a signature scheme secure against existential forgeability under chosen message attacks (EUF-CMA) [GMR88], capable of signing public keys for the scheme itself. The security of the scheme relies on a new assumption presented in Sect. 4.3. The scheme uses a zero-knowledge proof of knowledge [DP92], which we introduce in Sect. 4.2 and of which we give an instantiation in Sect. 6. In order to achieve the strong security notions, we design the proof system to satisfy *weak simulation soundness*, a relaxation of the concept introduced by Sahai [Sah99].

2 Preliminaries

We briefly recapitulate the employed concepts from the literature. A *bilinear group* is a tuple $(n, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot), g)$ where $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups of order n and g is a generator of $\mathbb{G}$. Furthermore, $e(\cdot,\cdot)$ is a non-degenerate bilinear map $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, i.e. $\forall\, u, v \in \mathbb{G}\ \forall a, b \in \mathbb{Z}$: $e(u^a, v^b) = e(u, v)^{ab}$ and $e(g, g)$ is a generator of $\mathbb{G}_T$.

The Subgroup Decision Assumption. Let the group order $|\mathbb{G}| = n = pq$, where p, q are prime. The *subgroup decision assumption* (SD) states that no p.p.t. adversary not knowing the factorization of n can with non-negligible probability distinguish a random element of $\mathbb{G}$ from a random element of $\mathbb{G}_q$, the subgroup of order q.

BGN-Encryption. The subgroup decision assumption implies semantic security of the encryption scheme from [BGN05]: The public key is the bilinear group (not revealing the factors of its order) and an element $h \in \mathbb{G}_q$. The secret key is q. To encrypt a message $m \in \{0, \ldots, T\}$, with $T < p$, choose $r \leftarrow \mathbb{Z}_n$ and compute the ciphertext $C := g^m h^r$. Since h is of order q, we have $C^q = (g^m h^r)^q = (g^q)^m$, so m can be recovered by computing $\log_{g^q} C^q = m$.

The Decisional Linear Assumption. Let $(p, \mathbb{G}, \mathbb{G}_T, e)$ be a bilinear group; let f, h, g be generators of $\mathbb{G}$. We call a triple $(c_1, c_2, c_3) \in \mathbb{G}^3$ *linear* w.r.t. to the basis (f, h, g) iff $c_1 = f^r,\ c_2 = h^s,\ c_3 = g^{r+s}$ for some $r, s \in \mathbb{Z}_p$. The *decisional linear assumption* states that no p.p.t. adversary can distinguish random linear triples w.r.t. a random basis from random triples; that is, given $(g, g^x, g^y, g^{xr}, g^{ys})$ for random x, y, r, s, it is hard to distinguish g^{r+s} from a uniformly random element in $\mathbb{G}$.

Linear Encryption. [BBS04] introduced an encryption scheme whose security relies on the above assumption: Choose a secret key $(x, y) \leftarrow (\mathbb{Z}_p^*)^2$ and publish $pk := (f := g^x, h := g^y, g)$. To encrypt a message $m \in \mathbb{G}$, choose $r, s \leftarrow \mathbb{Z}_p$ and compute $\mathsf{Enc}(pk, m; (r, s)) := (f^r, h^s, mg^{r+s})$. Any ciphertext (u, v, w) can then be decrypted by computing $u^{-x^{-1}} v^{-y^{-1}} w = g^{-r} g^{-s} m g^{r+s} = m$.

GOS-Commitments. Based on linear encryption, [GOS06a] gave the following homomorphic commitment scheme: The commitment key is a public key for linear encryption and a triple (u, v, w) which is an encryption of either 1 or the group generator (i.e., $(f^{r_u}, h^{s_v}, g^{r_u+s_v})$ or $(f^{r_u}, h^{s_v}, g^{r_u+s_v+1})$ for random $r_u, s_v \in \mathbb{Z}_p$). The first leads to a perfectly hiding key, while the latter constitutes a perfectly binding key. Now, $\mathsf{Com}((f, h, g, u, v, w), m;\ (r, s)) := (u^m f^r,\ v^m h^s,\ w^m g^{r+s})$ is a commitment to $m \in \mathbb{Z}_p$ for random r, s.

3 The Leak-Tightness Lemma

In [BW07], Boyen and Waters use the following strategy to construct efficient group signatures without random oracles: First, they construct two-level hierarchical signatures (i.e., *certified signatures*) that satisfy unforgeability ("traceability"), such that signatures consist of group elements only and can be verified by checking *pairing product equations* (cf. Lemma 1). They then convert the scheme into a group signature scheme, ensuring anonymity by BGN-encrypting the signature components and adding proofs for the plaintexts satisfying the verification equations. Considerable effort is dedicated to showing that their specific proofs do not leak information on the plaintexts.

In fact, as shown by the following lemma, proofs of this kind generally do not leak any additional information on the encrypted values. We first state the—somewhat technical—results and clarify their relevance in the subsequent discussion.

Lemma 1 (Leak tightness)**.** *Let $(n, \mathbb{G}, \mathbb{G}_T, e, g)$ be a bilinear group, for $1 \leq j \leq \ell$, let $a_j, b_j \in \mathbb{G}$. Let $(X_i)_{i=1}^m$ satisfy a* **pairing product equation** $E_{(a_j,b_j)_j}$ *that is*

$$E_{(a_j,b_j)_j}(X_1, \dots, X_m): \quad \prod_{j=1}^{\ell} e\big(a_j \prod_{i=1}^m X_i^{\delta_{j,i}}, b_j \prod_{i=1}^m X_i^{\varepsilon_{j,i}}\big) = \mathbf{1} \qquad \text{with } \delta_{j,i}, \varepsilon_{j,i} \in \mathbb{Z}_n.$$

1. *Let $H \in \mathbb{G}$, $(\rho_i) \in \mathbb{Z}_n^{\,m}$. For $\widetilde{X}_i := X_i H^{\rho_i}$, we have*

$$\prod_j e\big(a_j \prod_i \widetilde{X}_i^{\delta_{j,i}}, b_j \prod_i \widetilde{X}_i^{\varepsilon_{j,i}}\big) = e\big(H, P_E\big((X_i), (\rho_i)\big)\big) \tag{$\tilde{E}$}$$

with $P_E\big((X_i), (\rho_i)\big) := \prod_j \big((a_j \prod_i X_i^{\delta_{j,i}})^{\sum \varepsilon_{j,i}\rho_i} (b_j \prod_i X_i^{\varepsilon_{j,i}})^{\sum \delta_{j,i}\rho_i} H^{(\sum \delta_{j,i}\rho_i)(\sum \varepsilon_{j,i}\rho_i)}\big)$.

2. *Given (X_i) and (X'_i) both satisfying E, and (ρ_i), (ρ'_i), s.t. for all i, $X_i H^{\rho_i} = X'_i H^{\rho'_i}$, then*

$$P_E\big((X_i), (\rho_i)\big) = P_E\big((X'_i), (\rho'_i)\big).$$

3. *Let $|G| = pq$, let $a_j, b_j, X_i \in \mathbb{G}_p$; $c_j, d_j, Y_i \in \mathbb{G}_q$ for all i, j. If (X_i) satisfy $E_{(a_j,b_j)_j}$ and (Y_i) satisfy $E_{(c_j,d_j)_j}$, then $(X_i Y_i)$ satisfy $E_{(a_j c_j, b_j d_j)_j}$.*

4. *Let furthermore $H \in \mathbb{G}_q$ and $\theta \in \mathbb{N}$ be such that $\theta \equiv 1 \pmod p$ and $\theta \equiv 0 \pmod q$. If $(\widetilde{X}_i) \in \mathbb{G}$ satisfy $\tilde{E}_{(a_j c_j, b_j d_j)_j}$ for some P_E, then $(\widetilde{X}_i^{\theta})$ satisfy $E_{(a_j,b_j)_j}$.*

See Appendix C.1 for the proof. We give a brief description of the lemma's content: Let (X_i) be group elements satisfying relation E; e.g., (X_i) are the components of a digital signature and E is the verification relation. Then, after BGN-encrypting (X_i) to $(\widetilde{X}_i)$, (1) states how to construct the proof P_E that satisfies $\widetilde{E}$.

Now (4) ensures that if $(\widetilde{X}_i)$ satisfy $\widetilde{E}$ in $\mathbb{G}$ for some P_E, then their projections $(\widetilde{X}_i^{\theta})$ into $\mathbb{G}_p$ satisfy E in $\mathbb{G}_p$, a fact that is used to reduce a forgery in the "anonymized" scheme in $\mathbb{G}$ to a forgery in the underlying scheme in $\mathbb{G}_p$, [BW06] for example translate a forged group signature to a forgery of a certified signature. If we have equations $E_{(a_j,b_j)_j}$ in $\mathbb{G}_p$ and $E_{(c_j,d_j)_j}$ in $\mathbb{G}_q$, and satisfying values (X_i), (Y_i), respectively, then their products satisfy equation $E_{(a_j c_j,\, b_j d_j)_j}$ in $\mathbb{G}$ due to (3), which we will use in our simulations.

The main result is (2): Assume $H \in \mathbb{G}$, rather than in $\mathbb{G}_q$, which is indistinguishable by the subgroup decision (SD) assumption. In this case the $(\widetilde{X}_i)$ are perfectly random: Given "encryptions" $(\widetilde{X}_i)$, then for any potential plaintexts (X_i), there exists randomness $(\rho_i := \log_H(\widetilde{X}_i / X_i))$ leading to $(\widetilde{X}_i)$. However, by (2) any such pair of plaintexts/randomness leads to the same proof P_E, thus the proof leaks no more information on the plaintext than the ciphertext already does.

Remark 2 ("Unlinkably re-randomizing" randomized values)**.** Consider (X_i) satisfying E, but with right-hand side $e(H, P')$ instead of 1; again, let $\widetilde{X}_i := X_i H^{\rho_i}$ for all i. Then $(\widetilde{X}_i)$ satisfy $\widetilde{E}$ with right-hand side $e(H, P' \cdot P_E((X_i), (\rho_i))$.

So, given a proof P for randomized $(\widetilde{X}_i)$ satisfying $\widetilde{E}$, we can re-randomize the $(\widetilde{X}_i)$ using fresh ρ'_i and adapt the proof by setting $P_{\text{new}} := P \cdot P_E((\widetilde{X}), (\rho'_i))$. If $((\widetilde{X}_i), P)$ and $((\widetilde{Y}_i), P')$ both satisfy $\widetilde{E}$, then their re-randomizations are indistinguishable by SD and Lemma 1(2).

3.1 The Waters Signature Scheme

We review the scheme from [Wat05] to sign messages $M = (M_1, \dots, M_m) \in \{0,1\}^m$.

Setup Choose a bilinear group $(n, \mathbb{G}, \mathbb{G}_T, e, g)$. The parameters are $g_2 \leftarrow \mathbb{G}^*$ and a vector $\vec{u} := (u_0, u_1, \dots, u_m) \leftarrow \mathbb{G}^{m+1}$. Choose a secret key $x \leftarrow \mathbb{Z}_m$, define the public key as $X := g^x$. For convenience, we define the following hash function $\mathcal{F}(M) := \prod_{i=1}^m u_i^{M_i}$.

Signing Choose $r \leftarrow \mathbb{Z}_p$ and define the signature as $\sigma := (g_2^x (u_0 \mathcal{F}(M))^r,\ g^{-r})$.

Verification Signature σ is accepted for message M if $e(\sigma_1, g)\, e(u_0 \mathcal{F}(M), \sigma_2) = e(g_2, X)$.

Security Existential unforgeability under chosen message attack (EUF-CMA) follows from hardness of the computational Diffie-Hellman assumption (CDH) in the underlying group.

3.2 Applying Lemma 1 to Construct Verifiable Encryption

To exemplify our techniques, we construct a verifiable encryption scheme [BGLS03], which we only sketch due to space limitations. Suppose, we want to encrypt a signature and prove that the plaintext satisfies the signature verification relation. This can be done thanks to Lemma 1, if the verification procedure consists merely of verifying pairing product equations, as is the case for Waters' scheme. Moreover, if the signatures are EUF-CMA, then a similar property holds for encryption/proof pairs: Even after querying such pairs for messages of his choice, no adversary can come up with a valid pair for a new message.

We construct scheme $\mathcal{ES}$ for encrypted signatures: Given a plain signature in scheme $\mathcal{S}$, independently BGN-encrypt all elements and add a proof P_E for each verification equation E, as defined in Lemma 1(1). Indistinguishability of the hidden elements follows from the SD assumption combined with (2): Replacing $H \in \mathbb{G}_q$ by a random element from the *entire* group $\mathbb{G}$ is indistinguishable by SD. Now the encryptions are perfectly random and the proofs do not reveal any information either; every hypothesis (X_i) on the plaintexts of $(\widetilde{X}_i)$ leads to exactly the same proof.

Unforgeability of $\mathcal{ES}$ on the other hand is inherited from scheme $\mathcal{S}$ defined in subgroup $\mathbb{G}_p$: Lemma 1(3) allows us to simulate all oracle queries and (4) lets us transform a forgery in $\mathcal{ES}$ to a forgery in $\mathcal{S}$; more precisely: Given an adversary $\mathcal{A}$ against $\mathcal{ES}$, we construct $\mathcal{B}$ against $\mathcal{S}$ as follows: After receiving the parameters of $\mathcal{S}$ in $\mathbb{G}_p$, $\mathcal{B}$ produces parameters and the public key for a twin scheme $\mathcal{TS}$ in subgroup $\mathbb{G}_q$ (knowing thus all secret information). Then $\mathcal{B}$ constructs scheme $\mathcal{ES}$ with parameters in $\mathbb{G}$ which are just the ones from $\mathcal{S}$ multiplied with their counterparts in $\mathcal{TS}$.

Whenever $\mathcal{A}$ performs an oracle query, $\mathcal{B}$ separates all involved group elements (if any) into their p-components (by raising them to the θth power as in (4)) and their q-components by raising them to the power of θ_q, with $\theta_q \equiv 0 \pmod p$ and $\theta_q \equiv 1 \pmod q$. The p-parts are submitted to $\mathcal{B}$'s own oracle, while the action on the q-parts can be performed by $\mathcal{B}$ himself. The two results are then combined to a solution in $\mathbb{G}$ by multiplying them. (3) guarantees that the product satisfies the equations in group $\mathbb{G}$, since both components satisfy the equations in their respective subgroups. Finally, a forgery returned by $\mathcal{A}$ can be translated to one for $\mathcal{S}$, again via (4).

To illustrate the power of our methodology, we give an instantiation of "anonymous proxy signatures". We first construct a (non-anonymous) delegation scheme whose verification relations satisfy the requirements of Lemma 1. To instantiate the generic concept of such a scheme, the most important tool is the following: an EUF-CMA-secure signature scheme, where the messages to be signed are vectors of public keys of the scheme itself. This is the main difference to previous certified-signature schemes (on which group signatures build), where the certification and the signature itself are not based on the same mechanism, excluding thus consecutive delegation. In order to motivate our proceeding we briefly review the notions from [FP08] in the next section.

3.3 Definition and Security of Anonymous Proxy Signatures

In an anonymous proxy signature scheme, there are the following protagonists: The *issuer* enrolls users to make them traceable, the *users* who can delegate and sign on behalf of other users, and the *opener* who can trace the hidden delegators and the signer from a signature in case of misuse.

The scheme consists of 7 algorithms: Setup produces the public parameters, the issuer's secret key and the opening key. Algorithm UKGen is run by the users in order to produce a key pair, the public key of which is registered by the issuer running Enroll. Users can delegate their signing rights by producing a *warrant* with Del—which also provides the possibility to re-delegate. Now given a warrant, users can "proxy sign" messages running PSig, whereas the resulting signatures are verifiable via PVer using the first ("original") delegator's public key only. Algorithm Open allows the opener to reveal the delegators and the signer.

We overview the required security notions and refer to Appendix B for the rigorous definitions:

Anonymity. The experiment for anonymity is the following: Consider an adversary getting the issuer's key and who in a first phase returns an original delegator's public key, two pairs consisting of a warrant and a secret key, and a message. Now, flip a random bit and depending on the outcome give the adversary a signature produced using either the first or the second warrant/secret-key pair. Then as long as both warrants result from the same number of delegations and both lead to valid signatures, the adversary cannot decide the value of the flipped bit.

Traceability. No adversary, after enrolling arbitrarily many users via an Enroll-oracle, can produce a signature which cannot be opened. Thus, every valid signature can be traced to registered users.

Non-Frameability. No adversary, even when colluding with the issuer and the opener, can frame honest users. More precisely, give the adversary *all* keys returned by Setup, and oracles to create honest users and ask delegations and signatures of them—or adaptively corrupt them by asking their secret key. Then the adversary is not able to produce a valid signature whose opening yields an honest user for a delegation or a signing he has not been queried for.

4 A Consecutive Signature-Delegation Scheme

4.1 Overview

A Generic Construction. The issuer and each user create a key pair for an EUF-CMA-secure signature scheme. To enroll a user, the issuer signs her public key, creating thus a *certificate* sent to the user. If user U_1 wants to delegate U_2, she sends her a signature on her own and U_2's public key, called *warrant*. To re-delegate to U_3, U_2 sends her her certificate $cert_2$, the warrant $warr_{1\to2}$ received from U_1, and $warr_{1\to2\to3}$, a signature on (pk_1, pk_2, pk_3), the user's public keys. Now to sign a message M on behalf of U_1, U_3 produces a signature σ on (pk_1, pk_2, pk_3, M). The (non-anonymous) proxy signature is $\Sigma := (warr_{1\to2}, pk_2, cert_2, warr_{1\to2\to3}, pk_3, cert_3, \sigma)$.

Remark 3 (Delegating for specific *tasks* only)**.** The scheme can easily be extended, so that delegation of signing rights can be done for specific tasks only—as proposed by [FP08]— as follows: When delegating, sign $(pk_1, \ldots, pk_i, task)$ rather than the public keys only; likewise for proxy signing. The verification procedure takes the *task* tag as additional argument and the verification relations are adapted respectively.

Instantiation. We instantiate the generic scheme by choosing Waters' signature scheme (cf. Sect. 3.1) as EUF-CMA-secure scheme, which supports the hierarchical nature of the messages to be signed. Unfortunately, at the same time, this limits us to a fixed maximal number of delegations.

The messages in the Waters scheme are bit-strings, while we need to sign vectors of public keys (i.e., group elements) for the scheme itself. We solve this shortcoming in the following way: Instead of signing public keys, we sign the bits of the *private* keys—which the signer should obviously not learn. We take thus advantage of the fact that Waters signatures can be computed and verified without knowledge of the message if its hash value $F = \mathcal{F}(M) = \prod_{i=1}^{m} u_i^{M_i}$ is given instead. On the other hand, the assumption we introduce in Sect. 4.3 guarantees that the hash value hides enough information about the secret key. In particular, it states that the public key and the secret key's hash look unrelated.

The private key's hash value can be precomputed by its owner and then be signed directly by the delegator. We define thus the following two functions[2]

$$\begin{aligned} \mathsf{FSig}(x, F) &:= (\bar{g}^x (\bar{u}F)^r, g^{-r}) \;\text{ for random }\; r \leftarrow \mathbb{Z}_p \\ \mathsf{FVer}(X, F, (\sigma_1, \sigma_2)) = 1 &\;\text{ iff }\; e(\sigma_1, g)\, e(\bar{u}F, \sigma_2) = e(\bar{g}, X) \end{aligned}$$

Now we need to add a NIZK proof of consistency of the hash with the corresponding public key, which we discuss in the next section. Anticipating, we note that the secret key must be extractable from such a proof, so we can reduce unforgeability of delegations (i.e., non-frameability) of our scheme to security of Waters signatures. We emphasize the fact that verifying the NIZK proof must exclusively consist of checking pairing product equations to be compatible with the Leak-Tightness Lemma.

4.2 ZK Proof of Equality of Logarithm and Hash Preimage

As mentioned above, in order to prove consistency of a public key $X = g^x$ with its hash value $F = \mathcal{F}(x)$, in Sect. 6 we construct a zero-knowledge proof system $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$ for NP-relation

$$R_{\mathrm{X}\leftrightarrow\mathrm{F}} := \big\{((X, F), x) \,\big|\, X = g^x, F = \mathcal{F}(x)\big\}$$

The language $\mathcal{L}_{\mathrm{X}\leftrightarrow\mathrm{F}}$ defined by it is then indistinguishable from $\mathbb{G}^2$ by the XF-assumption given in the next section. We require $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$ to have the following properties:

- Verification of a proof consists of checking pairing product equations.
- The proof is a proof of knowledge at the same time, i.e., we can extract witness x. Furthermore, extraction must be efficient and consequently cannot rely on rewinding techniques.
- We can simulate proofs for any (possibility false) statements $(g^{x_1}, \mathcal{F}(x_2))$ without knowledge of (x_1, x_2)
- Even after seeing a simulated proof of a *random* (not necessarily true) statement, no adversary can produce a proof for a false statement; in addition, from every valid proof, the witness can still be extracted. This property, defined below, is a relaxation of the standard notion of simulation soundness where it is the adversary that chooses the statement to be simulated.

[2]Note that $\mathsf{FSig}, \mathsf{FVer}$ do *not* constitute a secure signature scheme on their own; a successful forgery must include the message's bits in order to be reducible to CDH.

Definition 4 (Weak Simulation Soundness)**.** A NIZK proof of knowledge[3] $\Pi = (\mathsf{K}, \mathsf{P}, \mathsf{V}, \mathsf{Sim}_1, \mathsf{Sim}_2, \mathsf{Ext})$ for NP-language $\mathcal{L}$ is **weakly simulation sound** if for every p.p.t. adversary $\mathcal{A}$ the following probability is negligible in the security parameter λ:

$$\Pr\big[(crs, tr, ek) \leftarrow \mathsf{Sim}_1(1^\lambda);\ y \leftarrow \mathcal{L} \cup \overline{\mathcal{L}};\ \pi \leftarrow \mathsf{Sim}_2(tr, y);\ (y^*, \pi^*) \leftarrow \mathcal{A}(crs, (y, \pi));\\ w^* \leftarrow \mathsf{Ext}(ek, (y^*, \pi^*))\ :\ y^* \neq y \wedge (y^*, w^*) \notin R_{\mathcal{L}} \wedge \mathsf{V}(crs, y^*, \pi^*) = 1\big]$$

Weak simulation soundness (WSS) is implied by the following strengthening of zero-knowledge, where the adversary trying to distinguish between a real and a simulated proof is provided with an extraction oracle.

Definition 5 (Extraction Zero Knowledge)**.** A proof of knowledge $\Pi = (\mathsf{K}, \mathsf{P}, \mathsf{V}, \mathsf{Sim}_1, \mathsf{Sim}_2, \mathsf{Ext})$ is **extraction zero knowledge** if for every p.p.t. adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ we have:

$$\big|\ \Pr\big[\mathbf{Exp}^{\text{zk}}_{\Pi,\mathcal{A}}(\lambda) = 1\big] - \Pr\big[\mathbf{Exp}^{\text{zk-S}}_{\Pi,\mathcal{A}}(\lambda) = 1\big]\ \big| \ = \ \text{negl}(\lambda)$$

with

$\mathbf{Exp}^{\text{zk}}_{\Pi,\mathcal{A}}(\lambda)$
- $(crs, ek) \leftarrow \mathsf{K}(1^\lambda)$
- $(y, w, st) \leftarrow \mathcal{A}_1(crs\ :\ \mathsf{Ext}(ek, \cdot, \cdot))$
- $\pi \leftarrow \mathsf{P}(crs, y, w)$
- $b \leftarrow \mathcal{A}_2(st, \pi\ :\ \mathsf{Ext}(ek, \cdot, \cdot))$

$\mathbf{Exp}^{\text{zk-S}}_{\Pi,\mathcal{A}}(\lambda)$
- $(crs, ek, tr) \leftarrow \mathsf{Sim}_1(1^\lambda)$
- $(y, w, st) \leftarrow \mathcal{A}_1(crs\ :\ \mathsf{Ext}(ek, \cdot, \cdot))$
- $\pi \leftarrow \mathsf{Sim}_2(crs, tr, y)$
- $b \leftarrow \mathcal{A}_2(st, \pi\ :\ \mathsf{Ext}(ek, \cdot, \cdot))$

Claim 6 (EZK implies WSS)**.** *Let $\mathcal{L}$ be a language which no p.p.t. adversary can decide with non-negligible probability; let Π be an extraction-zero-knowledge proof of knowledge for $\mathcal{L}$. Then Π is weakly simulation sound.*

Proof. Consider the following game:

$$\text{GAME 0}\quad (crs, ek) \leftarrow \mathsf{K}(1^\lambda);\ (y, w) \leftarrow R_{\mathcal{L}};\ \pi \leftarrow \mathsf{P}(crs, y, w);\ (y^*, \pi^*) \leftarrow \mathcal{A}(crs, (y, \pi));\\ w^* \leftarrow \mathsf{Ext}(ek, (y^*, \pi^*));\ \text{return 1 iff } y^* \neq y \wedge (y^*, w^*) \notin R_{\mathcal{L}} \wedge \mathsf{V}(crs, y^*, \pi^*) = 1$$

Soundness of Π implies that $\mathcal{A}$ can win Game 0 with at most negligible probability. Now define Game 1 replacing K and P by Sim_1 and Sim_2, respectively. Games 0 and 1 are indistinguishable by EZK, since a distinguisher can perfectly simulate the games because of its extraction oracle. Finally, a distinguisher between Game 1 and the WSS game would contradict the assumption on $\mathcal{L}$ (neither game uses the witness w). ☐

4.3 The XF-Assumption

The XF-assumption basically states that for someone seeing a public key $X = g^x$ without knowing the secret key x, the hash $\mathcal{F}(x)$ of the latter looks random. We will utilize this when reducing non-frameability of our delegation scheme to unforgeability of Waters signatures, where we will have to produce hashes corresponding to unknown secret keys. Proof system $\Pi_{\text{X}\leftrightarrow\text{F}}$ allows us to simulate the consistency proofs, but however, replacing an element of $\mathcal{L}_{\text{X}\leftrightarrow\text{F}}$ by one outside the language must be indistinguishable to guarantee simulation.

[3] K generates the common reference string *crs*, P produces proofs that are verified via V. Simulator Sim_1 produces a *crs*, a trapdoor *tr* which allows Sim_2 to simulate proofs, and an extraction key *ek*, used by Ext to extract the witness.

Moreover, having to simulate hash values for all delegation levels (cf. Sect. 4.4 for the details), we will generalize our assumption: Given $X = g^{x_0}$ and Λ hash values $F_i = \mathcal{F}_i(x_i)$, for different hash functions $\mathcal{F}_i$, it is hard to tell whether all x_i's are equal. Intuitively, the assumption states that values F_i do not reveal more information about x than X.

Definition 7 ((Λ, m)–XF-Assumption)**.** Let $\Lambda, m \in \mathbb{N}$, $(n, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda)$ be a bilinear group, let $((u_{i,j})_{j=1}^m)_{i=1}^\Lambda \in \mathbb{G}^{\Lambda\times m}$. We define the $i^{\mathbf{th}}$ **hash** of $(x_1, \ldots, x_m) \in \{0,1\}^m$:

$$\mathcal{F}_i(x_1, \ldots, x_m) := \prod_{j=1}^m u_{i,j}^{x_j}$$

We say the (Λ, m)–**XF-Assumption** holds for $\mathcal{G}$ if it is difficult to distinguish the NP-language

$$\mathcal{L}_{\mathrm{X}\leftrightarrow\mathrm{F}} := \Big\{(X, (F_i)_{i=1}^\Lambda) \in \mathbb{G}^{\Lambda+1} \;\Big|\; \exists\, x := (x_1, \ldots, x_m) \in \{0,1\}^m : X = g^{\sum x_i 2^{i-1}} \wedge \bigwedge\nolimits_{i=1}^{\Lambda} F_i = \mathcal{F}_i(x)\Big\}$$

from $\mathbb{G}^{\Lambda+1}$, that is, for all p.p.t. adversaries $\mathcal{A}$, the following function is negligible in λ:

$$\begin{aligned} \Big|\ \Pr\big[&(n, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda);\ \vec{u} \leftarrow \mathbb{G}^{\Lambda\times m};\ x \leftarrow \{0,1\}^m\ : \\ &\qquad \mathcal{A}\big(n, \mathbb{G}, \mathbb{G}_T, e, g, \vec{u},\ g^{\sum x_i 2^{i-1}},\ \textstyle\prod u_{1,i}^{x_i},\ \ldots,\ \prod u_{\Lambda,i}^{x_i}\big) = 1\big] \\ -\ \Pr\big[&(n, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda);\ \vec{u} \leftarrow \mathbb{G}^{\Lambda\times m};\ X, F_1, \ldots, F_\Lambda \leftarrow \mathbb{G}\ : \\ &\qquad \mathcal{A}\big(n, \mathbb{G}, \mathbb{G}_T, e, g, \vec{u}, X, F_1, \ldots, F_\Lambda\big) = 1\big]\ \Big| \end{aligned}$$

Note that the assumption satisfies Naor's falsifiability criterion [Nao03]. We give some more intuition on the assumption.

Comparison to DDH and DLIN. Consider the $(1, m)$–XF-Assumption in a group $\mathbb{G}$ with $2^{\lambda-1} \le |\mathbb{G}| < 2^\lambda$, and $m = \lambda - 1$: Given $(g, u_1, \ldots, u_m, X, F)$, decide whether there exist $x_i \in \mathcal{S} := \{0,1\}$, s.t. $X = g^{\sum x_i 2^{i-1}}$ and $F = \prod u_i^{x_i}$.

If we set $m = 1$ and $\mathcal{S} = \mathbb{Z}_{2^\lambda}$, we end up with DDH—which is easy in bilinear groups; however, case $m = 2$, $\mathcal{S} = \mathbb{Z}_{2^{\lambda/2}}$ (i.e., $X = g^{x_1 + x_2 2^{\lambda/2}} \overset{?}{\Rightarrow} F = u_1^{x_1} u_2^{x_2}$) is implied by a variant of DLIN, where r, s are randomly chosen from a smaller set $\mathcal{S}$: An instance $(Y = g^y, Z = g^z, R = g^{yr}, S = g^{zs}, T \in \{g^{r+s}, g^t\})$ of DLIN with $r, s \in \mathcal{S}$ can be decided by running the XF-decider on $(u_1 = Y, u_2 = Z, X = T, F = R \cdot S^{2^{\lambda/2}})$. Now, if we continue to increase m while at the same time reducing the set of possible values for x_i, we end up with the XF-Assumption.

Relation to the DL Problem with Auxiliary Information. Consider the problem of computing $x = \log X$ on input $(X, F) \in \mathcal{L}$, i.e., additionally to instance X, a hash value of the logarithm is given. Suppose, there exists an algorithm $\mathcal{A}$ that on input $(\vec{u}, X, F)$ decides whether $F = \prod u_i^{x_i}$ for $x := \log X$. Then we can construct an algorithm $\mathcal{B}$ that given $(X, F) \in \mathcal{L}_{\vec{u}}$ computes $x = \log X$: For $1 \le i \le m$, choose random u_i^* and run $\mathcal{A}$ on $\big(U_i := (u_1, \ldots, u_{i-1}, u_i^*, u_{i+1}, \ldots, u_m), X, F\big)$. If $x_i = 0$, then $(X, F) \in \mathcal{L}_{U_i}$, whereas this is only the case with negligible probability if $x_i = 1$. $\mathcal{B}$ can thus extract x bit by bit.

4.4 Implementation of the Delegation Scheme $\mathcal{DS}$

Based on the ideas from Sect. 4.1, we give implementations of the algorithms introduced in Sect. 3.3:

$\mathsf{Setup}(1^\lambda, \Lambda)$ (λ is the security parameter, while $\Lambda - 1$ is the maximum number of delegations)

- Choose a bilinear group $gPar := (p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}(1^\lambda)$.

– Define m, the maximal length of messages to be signed, with $m := \lambda - 1$.
– Choose Waters parameters to sign messages consisting of $\Lambda \cdot m$ bits:

$$sPar := (\bar{g}, \bar{u}, (u_{i,1}, \ldots, u_{i,m})_{i=1}^{\Lambda}) \leftarrow \mathbb{G}^{\Lambda m+2}$$

– For $1 \leq i \leq \Lambda$, choose crs_i, a common reference string for $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$ for parameters $(u_{i,j})_{j=1}^{m}$.

The issuer chooses an issuing key $ik := \omega \leftarrow \mathbb{Z}_p$ and defines $\Omega := g^{\omega}$. The public parameters are

$$pp := \big(gPar, sPar, (crs_i)_{i=1}^{\Lambda},\ \Omega\big)$$

$\mathsf{UKGen}(pp)$ Choose $x \leftarrow \mathbb{Z}_{2^m}$ and set $X := g^x$. Define the public key $pk := (X, (F_i, P_i)_{i=1}^{\Lambda})$, where $F_i := \mathcal{F}_i(x)$, is the i^{th} hash (Definition 7) and $P_i := \mathsf{P}_{\mathrm{X}\leftrightarrow\mathrm{F}}(crs_i, (X, F_i), x)$ is a proof for X and F_i containing the same x.

$\mathsf{Enroll}(pp, ik, pk)$ Parse pk as $(X, (F_i, P_i)_{i=1}^{\Lambda})$.

(1) Check all proofs P_i; if one is invalid, return $\perp$.
(2) $cert_i := \mathsf{FSig}(\omega, F_i)$ for $1 \leq i \leq \Lambda$.
(3) Add $(X, (F_i, P_i, cert_i)_{i=1}^{\Lambda})$ to $UList$ and return $(cert_i)_{i=1}^{\Lambda}$.

The user defines his secret key as $sk = (X, (F_i, P_i, cert_i)_{i=1}^{\Lambda}, x)$.

$\mathsf{Del}(pp, sk_i, [warr_{\to i}], pk_{i+1})$ Let the user holding sk_i be the i^{th} user in the delegation chain.

(1) Parse sk_i as $(X_i, (F_{i,j}, P_{i,j}, cert_{i,j})_{j=1}^{\Lambda}, x_i)$ and pk_{i+1} as $(X_{i+1}, (F_{i+1,j}, P_{i+1,j})_{j=1}^{\Lambda})$.
Parse $warr_{\to i} = \big((X_j, F_{j,j}, P_{j,j}, cert_{j,j}, \sigma_j)_{j=1}^{i-1}, (X'_i, F'_{i,i}, P'_{i,i})\big)$,
in case $i = 1$, define $warr_{\to 1} := (X_1, F_{1,1}, P_{1,1})$
(2) If one of the proofs in $warr_{\to i}$ or pk_{i+1} is invalid or if $(X'_i, F'_{i,i}, P'_{i,i}) \neq (X_i, F_{i,i}, P_{i,i})$
return $\perp$.
(3) Define $\sigma_i \leftarrow \mathsf{FSig}(x_i, F_{1,1} \cdots F_{i,i} \cdot F_{i+1,i+1})$.
Return $warr_{\to i+1} := warr_{\to i} \,\|\, (cert_{i,i}, \sigma_i, (X_{i+1}, F_{i+1,i+1}, P_{i+1,i+1}))$.

$\mathsf{PSig}(pp, sk_k, warr_{\to k}, M)$ Let $sk_k = (X_k, (F_{k,j}, P_{k,j}, cert_{k,j})_{j=1}^{\Lambda}, x_k)$;
let $warr_{\to k} = \big((X_j, F_{j,j}, P_{j,j}, cert_{j,j}, \sigma_j)_{j=1}^{k-1}, (X'_k, F'_{k,k}, P'_{k,k})\big)$.

(1) If one of the proofs in $warr_{\to k}$ is invalid or if $(X'_k, F'_{k,k}, P'_{k,k}) \neq (X_k, F_{k,k}, P_{k,k})$
return $\perp$
(2) Define $\sigma_k := \mathsf{FSig}(x_k, F_{1,1} \cdots F_{k,k} \cdot \mathcal{F}_{\Lambda}(M))$.
(3) The proxy signature is $\Sigma := \big(\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=2}^{k}\big)$

$\mathsf{PVer}(pp, pk, M, \Sigma)$ Let $pk = (X_1, F_{1,1}, P_{1,1}, \ldots)$, let $\Sigma = \big(\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=2}^{k}\big)$. Return 0 if any of the following returns 0, otherwise return 1.

(1) $\mathsf{V}_{\mathrm{X}\leftrightarrow\mathrm{F}}(crs_i, (X_i, F_{i,i}), P_{i,i})$, for $1 \leq i \leq k$,
(2) $\mathsf{FVer}(\Omega, F_{i,i}, cert_{i,i})$, for $2 \leq i \leq k$,
(3) $\mathsf{FVer}(X_i, F_{1,1} \cdots F_{i+1,i+1}, \sigma_i)$, for $1 \leq i < k$,
$\mathsf{FVer}(X_k, F_{1,1} \cdots F_{k,k} \cdot \mathcal{F}_{\Lambda}(M), \sigma_k)$.

$\mathsf{Open}(pp, pk, M, \Sigma, UList)$ If Σ is valid, parse it as $\big(\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=2}^k\big)$. If for all i, $X_i \in UList$, return $(X_2, \ldots, X_k)$, otherwise return $\perp$.

Claim 8. *Scheme $\mathcal{DS}$ is non-frameable*

We give an overview of the proof and refer to Appendix C.3 for the rigorous proof. Our strategy is to reduce a "framing" proxy signature to a forgery of a Waters signature: An EUF-CMA adversary $\mathcal{B}$ against Waters' scheme receives public key X from its environment and sets out to simulate the non-frameability game for adversary $\mathcal{A}$ against $\mathcal{DS}$, while setting X as the public key of a random honest user U^*. Now to do so, it must simulate the hash values (F_i) corresponding to X. We define thus a sequence of indistinguishable games: The first game is the original non-frameability game. In the next one, we simulate the ZK-proofs (P_i) in the public key of U^*. In the third game, relying on the XF assumption, we substitute the (F_i) by random values. Now the last game can be simulated by $\mathcal{B}$, given the fact that the signatures required to answer Del and PSig queries can be forwarded to $\mathcal{B}$'s own signing oracle. If $\mathcal{A}$ wins the non-frameability game by framing U^*, then the signature output by $\mathcal{A}$ contains a Waters forgery.

However, to win the EUF-CMA game, $\mathcal{B}$ is required to return the bits of the message rather than its hash value—in fact, $\mathcal{B}$'s oracle queries also require bits. This is why we need $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$ to be an extractable proof system; moreover, extraction must be possible even after having simulated proofs—which is the reason for $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$ to be weakly simulation sound.

Claim 9. *Scheme $\mathcal{DS}$ is traceable*

Proof. The claim follows by a reduction to unforgeability of the Waters signature scheme for messages of length $\Lambda \cdot m$ using the following fact:

> Let $\mathbf{0}^i$ denote a string of $i \cdot m$ zeroes. Then for any i^*, a signature on $(\mathbf{0}^{i^*-1} \,\|\, x \,\|\, \mathbf{0}^{\Lambda-i^*})$ w.r.t. parameters $((u_{i,j})_{j=1}^m)_{i=1}^\Lambda$ is a signature on x w.r.t. parameters $(u_{i^*,j})_{j=1}^\Lambda$.

The simulator sets Ω to the public key it is challenged on and deals with $\mathsf{Enroll}(X, (F_i, P_i))$ queries the in following way:

- If one of the P_i is invalid, return $\perp$, otherwise extract x from one of them.
- To produce $cert_i$, query a signature on the message $(\mathbf{0}^{i-1} \,\|\, x \,\|\, \mathbf{0}^{\Lambda-i})$.

Finally, open the signature returned by the adversary to $X_2, \ldots, X_k$. If for some i, $X_i \notin UList$, return $cert_i$ from the signature, together with the extracted bits. $\square$

5 The Anonymous Delegation Scheme

Now using the techniques discussed in Sect. 3, we can convert the scheme $\mathcal{DS}$ from the last section into an anonymous proxy signature scheme $\mathcal{APS}$. We give the necessary modifications to $\mathcal{DS}$:

$\mathsf{Setup}(1^\lambda, \Lambda)$ Choose a bilinear group of composite order $(p, q, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}_c(1^\lambda)$ and define $gPar := (n = pq, \mathbb{G}, \mathbb{G}_T, e, g)$. Add $H \leftarrow \mathbb{G}_q$, a subgroup element for BGN-encryptions, to pp and additionally output the opening key $ok := q$.

$\mathsf{Enroll}(pp, ik, (X, \ldots))$ The issuer accepts the request only if $X^q \neq (X')^q$ for all $X' \in UList$—otherwise the set of possible ciphertexts of X and X' would be equal, making correct tracing impossible.

$\mathsf{PSig}(pp, sk_k, warr_{\to k}, M)$ After producing $\Sigma := \big(\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=2}^k\big)$, BGN-encrypt all elements of Σ using H and add one proof π (cf. Lemma 1) per pairing product equation to be satisfied in PVer. The blinded signature is $\widetilde{\Sigma} := \big(\widetilde{\sigma}_1, (\widetilde{X}_i, \widetilde{F}_{i,i}, \widetilde{P}_{i,i}, \widetilde{cert}_{i,i}, \widetilde{\sigma}_i)_{i=2}^k, (\pi_i)\big)$.

$\mathsf{PVer}(pp, pk, M, \widetilde{\Sigma})$ instead of verifying the pairing product equations directly, verify the proofs (π_i) on the encrypted values.

$\mathsf{Open}(pp, ok, M, \widetilde{\Sigma}, \mathit{UList})$ If $\widetilde{\Sigma}$ passes verification, do the following for $2 \leq i \leq \Lambda$: if $\widetilde{X}_i^q = (X')^q$ for some $X' \in \mathit{UList}$, then set $X_i := X'$, otherwise return $\perp$. Finally, return $(X_2, \ldots, X_k)$.

Anonymity. Consider two "plain" proxy signatures Σ_1 and Σ_2 that are both valid under the same public key and result from the same number of delegations (and consequently contain the same number of group elements). If we blind both signatures, then they are indistinguishable: The ciphertexts are indistinguishable by a hybrid argument on semantic security of BGN and the added (π_i) do not leak any information on the cleartexts by Lemma 1(2). As a consequence, $\mathcal{APS}$ satisfies anonymity as defined in Sect. 3.3.

Traceability and Non-Frameability. Traceability as well as non-frameability follow by a reduction to the respective notions for $\mathcal{DS}$ in the subgroup $\mathbb{G}_p$. Given an adversary $\mathcal{A}$ against $\mathcal{APS}$, we construct $\mathcal{B}$ against $\mathcal{DS}$: After receiving $pp_{\mathcal{DS}}$, $\mathcal{B}$ constructs $pp_{\mathcal{APS}}$ by first creating parameters pp', ik' for a new instance of $\mathcal{DS}$ in group $\mathbb{G}_q$, and then multiplying all parameters from $pp_{\mathcal{DS}}$ with the new ones, resulting thus in correctly distributed parameters in $\mathbb{G}$, e.g., $g \in pp$ and $g' \in pp'$ yield $\overline{g} := gg' \in \mathbb{G}$. Finally, $\mathcal{B}$ adds $H \in \mathbb{G}_q$ to $pp_{\mathcal{APS}}$. $\mathcal{A}$'s oracle queries are dealt with in the following way:

PK **queries** Run the PK oracle for $\mathcal{DS}$ to get $pk := (X, (F_i, P_i))$, then choose a secret key $x' \in \{0,1\}^m$ and compute $X' := (g')^{x'}$ and $F'_i := \prod (u'_{i,j})^{x'_j}$ for $1 \leq i \leq \Lambda$, as well as the corresponding proofs w.r.t. parameters pp'. Let the result be pk' and define $(\overline{X}, (\overline{F}_i, \overline{P}_i))$ by multiplying all components of pk with the respective ones of pk'.

First, note that due to Lemma 1(3), all proofs $\overline{P}_i$ satisfy all pairing product equations of $\mathsf{V}_{\mathrm{X}\leftrightarrow\mathrm{F}}$. Second, $(\overline{X}, (\overline{F}_i))$ is indistinguishable from a correctly computed one by the XF-Assumption.[4]

Enroll, Del, PSig **queries** Answering these queries basically consists of simulating $\mathsf{FSig}(y, F_1 \cdots F_k)$ for some $y, F_1, \ldots, F_k$. Define θ_p, θ_q such that $\theta_p \equiv_p 1$, $\theta_p \equiv_q 0$, $\theta_q \equiv_p 0$, $\theta_q \equiv_q 1$. If $F = \prod_{i=1}^m (u_i u'_i)^{x_i}$, then $F^{\theta_p} = \prod u_i^{x_i} \in \mathbb{G}_p$ and $F^{\theta_q} = \prod (u'_i)^{x_i} \in \mathbb{G}_q$. Now, $\mathcal{B}$ submits ${F_1}^{\theta_p} \cdots {F_k}^{\theta_p}$ to its own oracle to get σ and—knowing all secret keys for the q-components—computes σ' in $\mathbb{G}_q$ on his own. Finally, $\mathcal{B}$ returns $\overline{\sigma} = \sigma \cdot \sigma'$ which is a valid signature according to Lemma 1(3).

When $\mathcal{A}$ eventually returns (pk, M, Σ), $\mathcal{B}$ "translates" the result back to $\mathbb{G}_p$ by raising everything to the power of θ_p and outputs it. It follows from Lemma 1(4) that $\mathcal{B}$'s output passes verification. If $\mathcal{A}$ wins its game then so does $\mathcal{B}$:

Traceability If $\mathcal{A}$ wins the game, then for some i, $\widetilde{X}_i^q \neq (X')^q$ for all $X' \in \mathit{UList}_{\mathcal{APS}}$, which implies $\widetilde{X}_i^{\theta_p} \neq (X')^{\theta_p}$ for all X'. On the other hand we have $\mathit{UList}_{\mathcal{DS}} = \{X^{\theta_p} \mid X \in \mathit{UList}_{\mathcal{APS}}\}$. Together, this implies $\widetilde{X}_i^{\theta_p} \notin \mathit{UList}_{\mathcal{DS}}$, the condition for $\mathcal{A}$ winning the game.

[4]An element $(g^x, (\mathcal{F}_i(x))) \in \mathcal{L}_{\mathbb{G}}$ is indistinguishable from a random element in $\mathbb{G}^{\Lambda+1}$ by the XF-Assumption in $\mathbb{G}$. Now the latter is indistinguishable from elements $(g^x (g')^{x'_1}, (\mathcal{F}_i(x)\mathcal{F}'_i(x'_2))$ in $\mathcal{L}_{\mathbb{G}_p} \cdot \mathbb{G}_q^{\Lambda+1}$ by the XF-Assumption in $\mathbb{G}_p$, whereas the one in $\mathbb{G}_q$ guarantees indistinguishability of $\mathcal{L}_{\mathbb{G}_p} \cdot \mathbb{G}_q^{\Lambda+1}$ from $\mathcal{L}_{\mathbb{G}_p} \cdot \mathcal{L}_{\mathbb{G}_q}$.

Non-frameability Analogously: $\mathcal{A}$ wins the game if in the returned signature, there is one delegation step it has not queried. Since we compare "openings" of the signature and the warrants, the argument works as for traceability.

6 The Proof of Equality of Exponent and Hash Preimage

In order to construct $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$, as introduced in Sect. 4.2, we will use the following proof systems, detailed in Appendix A.

$\Pi_{\mathbf{1L}}$ A perfect WI (witness indistinguishable) proof system similar to the one from [GOS06a]: Given two triples, prove that at least one of them is linear. We generalize their method, so linearity holds with respect to different bases.

$\Pi_{\mathbf{b,eq}}$ From Π_{1L} we directly derive a proof of the following: Given a GOS-commitment to x and a linear encryption to $g^{x'}$, prove that $x, x' \in \{0,1\}$ and $x = x'$.

$\Pi_{\mathbf{cX}}$ Given a vector of GOS-commitments to bits $(c_i)_{i=1}^m$ and $X \in \mathbb{G}$, Π_{cX} is a NIZK proof for the committed values being the bits of $\log X$.

$\Pi_{\mathbf{cF}}$ Given a vector of commitments to bits $(c_i)_{i=1}^m$ and $F \in \mathbb{G}$, Π_{cF} is a NIZK proof for the committed values being a Waters-hash preimage of F, i.e., if c_i commits to x_i for all i, then $F = \mathcal{F}(x_1, \ldots, x_m)$.

$\Pi_{\mathbf{G}}$ Given $(pk, pk', d, d', ck, c, v)$, Π_{G} is a WI proof for either d and d' being linear encryptions of the same message under pk, pk', resp., or c being a commitment to v under ck.

Furthermore, we will use a one-time signature scheme $\mathcal{S}_{\mathrm{ots}} = (\mathsf{KGen}_{\mathrm{ots}}, \mathsf{Sig}_{\mathrm{ots}}, \mathsf{Ver}_{\mathrm{ots}})$ (cf. [Gro06] for an implementation). All verification procedures of the above systems consist exclusively of checking pairing product equations. Before going into details, we give an overview of our construction:

Let $((X,F),x) \in R_{\mathrm{X}\leftrightarrow\mathrm{F}}$, i.e., $X = g^x$ and $F = \mathcal{F}(x)$. Aiming for an extractable proof, we first produce vectors of commitments $\vec{c_X}$ and $\vec{c_F}$ to the bits of x and prove consistency with X and F via Π_{cX} and Π_{cF}, resp. The proofs can be simulated by replacing the commitment keys for c_X and c_F by perfectly hiding keys. However, to achieve extraction-zero knowledge (EZK)—and thus weak simulation soundness—, we must extract from proofs queried to the oracle, even after replacing the CRS by a simulated one. We thus add linear encryptions d'_i and d''_i under public keys pk', pk'' of the bits in $(c_{X\,i})$ and $(c_{F\,i})$, resp., and prove that we did so via $\Pi_{\mathrm{b,eq}}$. At the same time this proves that $c_{X\,i}, c_{F\,i}$ are commitments to bits and that d'_i, d''_i are encryptions of either g^0 or g^1.

The latter enables us to ensure equality of the plaintexts in d'_i and d''_i for all i at once, by proving $d'_P := \prod (d'_i)^{2^{i-1}}$ and $d''_P := \prod (d''_i)^{2^{i-1}}$ decrypt to the same plaintext. However, this proof must contain some kind of trapdoor, because in the proof of EZK, d'_i and d''_i might contain different plaintexts. To do so, we use Groth's trick to achieve RCA-secure encryptions in [Gro06]:

Add a commitment c_G under key ck_G of a signature verification key vk_G to the CRS of $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$ and require the prover to choose a one-time signature key pair (vk, sk), and to add vk and a signature on (X,F) to the proof. The proof for $\vec{d'}$ and $\vec{d''}$ being consistent is a Π_{G} proof of $(pk', pk'', d', d'', ck_G, c_G, vk)$. Now we can (one-time) simulate proofs by choosing $vk := vk_G$ and using the corresponding signing key which is unknown to the adversary.

6.1 The Proof System $\Pi_{\mathbf{X}\leftrightarrow\mathbf{F}} = (\mathsf{K}_{\mathbf{X}\leftrightarrow\mathbf{F}}, \mathsf{P}_{\mathbf{X}\leftrightarrow\mathbf{F}}, \mathsf{V}_{\mathbf{X}\leftrightarrow\mathbf{F}}, \mathsf{Sim}_{\mathbf{X}\leftrightarrow\mathbf{F}}, \mathsf{Ext}_{\mathbf{X}\leftrightarrow\mathbf{F}})$

Reference String Generation $\mathsf{K}_{\mathrm{X}\leftrightarrow\mathrm{F}}(p, \mathbb{G}, \mathbb{G}_T, e, g, \vec{u})$

- choose $\phi, \psi, \xi_1', \xi_2', \xi_1'', \xi_2'', r_a, s_b, r_v, s_w, r_G \leftarrow \mathbb{Z}_p$
- $(vk_G, sk_G) \leftarrow \mathsf{KGen}_{\text{ots}}(1^\lambda);\ ck_G \leftarrow \mathsf{KGen}_{\text{Com,binding}}(1^\lambda);\ c_G \leftarrow \mathsf{Com}(ck_G, vk_G;\ r_G)$
- $crs_G' \leftarrow \mathsf{K}_G(1^\lambda)$
- define

$$\begin{array}{llllll} & f := g^\phi & h := g^\psi & z := g^{(r_a+s_b+1)^{-1}} & a := f^{r_a} & b := h^{s_b} \\ & f' := z^{\xi_1'} & h' := z^{\xi_2'} & f'' := z^{\xi_1''} & h'' := z^{\xi_2''} & \\ \text{for } 1 \le i \le m & f_i := u_i^\phi & h_i := u_i^\psi & z_i := u_i^{(r_v+s_w+1)^{-1}} & v_i := u_i^{r_v} & w_i := u_i^{s_w} \end{array}$$

- $ck_X := (f, h, z, a, b, g),\ ck_{Fi} := (f_i, h_i, z_i, v_i, w_i, u_i),\ pk' := (f', h', z),\ pk'' := (f'', h'', z).$
  $crs_G := (ck_G, c_G, crs_G')$

$$crs := (ck_X, \vec{ck}_F, pk', pk'', crs_G) \quad tr := (\phi, \psi, r_a, s_b, r_v, s_w, vk_G, sk_G, r_G) \quad ek := (\xi_1', \xi_2', \xi_1'', \xi_2'')$$

Proof $\mathsf{P}_{\text{X}\leftrightarrow\text{F}}(crs, (X, F), x)$ – for all $1 \le i \le m$ do: choose $r_{Xi}, s_{Xi}, r_{Fi}, s_{Fi}, r_i', s_i', r_i'', s_i'' \leftarrow \mathbb{Z}_p$

$$\begin{array}{ll} c_{Xi} := \mathsf{Com}(ck_X, x_i;\ r_{Xi}, s_{Xi}) & c_{Fi} := \mathsf{Com}(ck_{Fi}, x_i;\ r_{Fi}, s_{Fi}) \\ d_i' := \mathsf{Enc}(pk', g^{x_i};\ r_i', s_i') & \pi_i' := \mathsf{P}_{\text{b,eq}}\big((ck_X, pk'), (c_{Xi}, d_i'), (r_{Xi}, s_{Xi}, r_i', s_i')\big) \\ d_i'' := \mathsf{Enc}(pk'', g^{x_i};\ r_i'', s_i'') & \pi_i'' := \mathsf{P}_{\text{b,eq}}\big((ck_{Fi}, pk''), (c_{Fi}, d_i''), (r_{Fi}, s_{Fi}, r_i'', s_i'')\big) \end{array}$$

- $\pi_X := \mathsf{P}_{\text{cX}}\big(ck_X, (X, \vec{c}_X), (x, \vec{r}_X, \vec{s}_X)\big)$
- $\pi_F := \mathsf{P}_{\text{cF}}\big(ck_F, (F, \vec{c}_F), (x, \vec{r}_F, \vec{s}_F)\big)$
- $(vk, sk) \leftarrow \mathsf{K}_{\text{ots}}(1^\lambda);\quad \sigma := \mathsf{Sig}_{\text{ots}}(sk, (X, F))$
- $d_P' := \prod (d_i')^{2^{i-1}};\ r_P' := \sum r_i' 2^{i-1};\ s_P' := \sum s_i' 2^{i-1};$
  $d_P'' := \prod (d_i'')^{2^{i-1}};\ r_P'' := \sum r_i'' 2^{i-1};\ s_P'' := \sum s_i'' 2^{i-1}$
- $\pi_G := \mathsf{P}_{\text{G}}\big(crs_G', (pk', pk'', d_P', d_P'', ck_G, c_G, vk), (X, r_P', s_P', r_P'', s_P'')\big)$ ([5])

The proof for (X, F) is $\pi := \big(\pi_X, \vec{c}_X, \vec{\pi}', \vec{d}', \pi_G, \vec{d}'', \vec{\pi}'', \vec{c}_F, \pi_F, vk, \sigma\big)$

Verification $\mathsf{V}_{\text{X}\leftrightarrow\text{F}}(crs, (X, F), \pi)$ To verify a proof, verify σ on (X, F) under vk and verify the proofs π_X, π_F, π_G and π_i', π_i'' for all $1 \le i \le m$.

Extraction $\mathsf{Ext}_{\text{X}\leftrightarrow\text{F}}(ek, (X, F), \pi)$ Let $ek := (\xi', \xi'')$, $\pi := \big(\pi_X, \vec{c}_X, \vec{\pi}', \vec{d}', \pi_G, \vec{d}'', \vec{\pi}'', \vec{c}_F, \pi_F, vk, \sigma\big)$ If π is valid, extract bits x_i using either ξ' on d_i' or ξ'' on d_i'' for all i.

Simulation $\mathsf{Sim}_{\text{X}\leftrightarrow\text{F},1}$ works as $\mathsf{K}_{\text{X}\leftrightarrow\text{F}}$ except for outputting tr and replacing ck_X and all ck_{Fi} by perfectly hiding commitment keys by setting $z := g^{(r_a+s_b)^{-1}}$ and $z_i := u_i^{(r_v+s_w)^{-1}}$

$\mathsf{Sim}_{\text{X}\leftrightarrow\text{F},2}$ – replace c_{Xi} and c_{Fi} by commitments to 0, d_i', d_i'' by encryptions of g^0.
- $\pi_X := \mathsf{Sim}_{\text{cX}}((\phi, \psi, r_a, r_b), (X, \vec{c}_X)),\quad \pi_i' := \mathsf{Sim}_{\text{b,eq}}\big((ck_X, pk', r_a, s_b), (c_{Xi}, d_i'), (r_{Xi}, s_{Xi})\big)$
- $\pi_F := \mathsf{Sim}_{cF}((\phi, \psi, r_v, s_w), (F, \vec{c}_F)),\quad \pi_i'' := \mathsf{Sim}_{\text{b,eq}}\big((ck_{Fi}, pk'', r_v, s_w), (c_{Fi}, d_i''), (r_{Fi}, s_{Fi})\big)$
- $(vk, sk) := (vk_G, sk_G),\quad \sigma := \mathsf{Sig}_{\text{ots}}(sk_G, (X, F)),$
  $\pi_G := \mathsf{P}_G\big(crs_G, (pk', pk'', \prod (d_i')^{2^{i-1}}, \prod (d_i'')^{2^{i-1}}, ck_G, c_G, vk_G), (vk_G, r_G)\big)$

[5] Due to the homomorphic property of linear encryption, $d_i' = \mathsf{Enc}(x_i;\ r_i', s_i')$ implies $d_P' = \mathsf{Enc}(\prod g^{x_i 2^{i-1}};\ r_P', s_P')$.

Soundness / Extraction. Let π be a valid proof for (X, F). First of all, soundness of π'_i guarantees that if c_{Xi} is a commitment to x'_i, then $x'_i \in \{0,1\}$ and furthermore d'_i encrypts $g^{x'_i}$. Now π_X ensures that $X = g^{\sum x'_i 2^{i-1}}$. An analogous argument applied to $\vec{\pi}''$ and π_F yields that if c_{Fi} commits to x''_i, then d''_i encrypts $g^{x''_i}$, with $x''_i \in \{0,1\}$, and $F = \mathcal{F}(x''_1, \ldots, x''_m)$. Since the commitment c_G is computationally hiding, it is only with negligible probability that vk is the committed value. Soundness of π_G ensures thus that the values encrypted in $\prod (d'_i)^{2^{i-1}}$ and $\prod (d''_i)^{2^{i-1}}$ are the same, i.e., $g^{\sum x'_i 2^{i-1}} = g^{\sum x''_i 2^{i-1}}$, which, given the fact that $x'_i, x''_i \in \{0,1\}$, means $x'_i = x''_i$ for all i. Thus, $(X, F) \in \mathcal{L}_{\mathrm{X}\leftrightarrow\mathrm{F}}$ and the bits extracted by Ext are the bits of the witness.

Extraction Zero Knowledge.

Theorem 10. $\Pi_{X\leftrightarrow F}$ *is an extraction-zero-knowledge proof of knowledge and language membership.*

See Appendix C.2 for a proof sketch.

Acknowledgments

This work was partially funded by EADS and by the French ANR-07-SESU-008-01 PAMPA Project.

References

[ACJT00] G. Ateniese, J. Camenisch, M. Joye, G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. *CRYPTO '00* LCNS 1880, pp. 255–270, Springer Verlag, 2000.

[BCKL08] M. Belenkiy, M. Chase, M. Kohlweiss, A. Lysyanskaya. Non-interactive anonymous credentials. *TCC '08* LCNS 4948, pp. 356–374. Springer Verlag, 2008.

[BMW03] M. Bellare, D. Micciancio, B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. *EUROCRYPT '03*, LNCS 2656, pp. 614–629. Springer Verlag, 2003.

[BR93] M. Bellare, P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. *ACM Conference on Computer and Communications Security '93*, pp. 62–73, ACM, 1993.

[BSZ05] M. Bellare, H. Shi, C. Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA 2005*, LNCS 3376, pp. 136–153. Springer Verlag, 2005.

[BFM88] M. Blum, P. Feldman, S. Micali. Non-interactive zero-knowledge and its applications. *STOC '88*, pp. 103–112, ACM, 1988.

[BPW03] A. Boldyreva, A. Palacio, B. Warinschi. Secure proxy signature schemes for delegation of signing rights. *IACR ePrint Archive:* Report 2003/096, 2003.

[BB04] D. Boneh, X. Boyen. Short signatures without random oracles. *EUROCRYPT '04*, LCNS 3027, pp. 56–73, Springer Verlag, 2004.

[BBS04] D. Boneh, X. Boyen, H. Shacham. Short group signatures. *CRYPTO '04*, LNCS 3152, pp. 43–55, Springer Verlag, 2004

[BGLS03] D. Boneh, C. Gentry, B. Lynn, H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. *EUROCRYPT '03*, LNCS 2656, pp. 416–432, Springer Verlag, 2003.

[BGN05] D. Boneh, E.-J. Goh, K. Nissim. Evaluating 2-DNF formulas on ciphertexts. *TCC '05*, LCNS 3378, pp. 325–341, Springer Verlag, 2005

[BW06] X. Boyen, B. Waters. Compact group signatures without random oracles. *EUROCRYPT '06*, LNCS 4004, pp. 427–444. Springer Verlag, 2006

[BW07] X. Boyen, B. Waters. Full-domain subgroup hiding and constant-size group signatures. *PKC '07*, LNCS 4450, pp. 1–15. Springer Verlag, 2007.

[Cha85] D. Chaum. Security without identication: transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10), pp. 1030–1044, 1985.

[CvH91] D. Chaum, E. van Heyst. Group signatures. *EUROCRYPT '91*, LNCS 547, pp. 257–265. Springer Verlag, 1991.

[DP92] A. De Santis, G. Persiano. Zero-knowledge proofs of knowledge without interaction. *FOCS '92*, pp. 427–436, IEEE Computer Society, 1992.

[FP08] G. Fuchsbauer, D. Pointcheval. Anonymous proxy signatures. *SCN '08*, LNCS 5229, pp. 201–217. Spinger Verlag, 2008

[GMR88] S. Goldwasser, S. Micali, R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing,* 17(2):281–308, 1988.

[GOS06a] J. Groth, R. Ostrovsky, A. Sahai. Non-interactive zaps and new techniques for NIZK. *CRYPTO '06*, LNCS 4117, pp. 97–111, Springer Verlag, 2006.

[GOS06b] J. Groth, R. Ostrovsky, A. Sahai. Perfect non-interactive zero knowledge for NP. *EUROCRYPT '06*, LNCS 4004, pp. 339–358, Springer Verlag, 2006.

[Gro07] J. Groth. Fully anonymous group signatures without random oracles. *ASIACRYPT '07*, LNCS 4833, pp. 164-180, Springer Verlag, 2007

[Gro06] J. Groth. Simulation-sound nizk proofs for a practical language and constant size group signatures. *ASIACRYPT '06*, LCNS 4284, pp. 444–459, Springer Verlag, 2006

[GS08] J. Groth, A. Sahai. Efficient non-interactive proof systems for bilinear groups. *EUROCRYPT '08*, LCNS 4965, pp. 415–432, Springer Verlag, 2008.

[KP98] J. Kilian, E. Petrank. Identity escrow. *CRYPTO '98*: LCNS 1462, pp. 169–185, Springer Verlag, 1998.

[MUO96] M. Mambo, K. Usuda, E. Okamoto. Proxy signatures for delegating signing operation. *Proceedings of the 3rd ACM Conference on Computer and Communications Security (CCS)*. ACM, 1996.

[Nao03] M. Naor. On cryptographic assumptions and challenges. *CRYPTO '03*, LCNS 2729, pp. 96–109, Springer Verlag, 2003.

[Sah99] A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. *FOCS '99*, pp. 543–553, IEEE Computer Society, 1999.

[TW05] M. Trolin, D. Wikström. Hierarchical group signatures. *Automata, Languages and Programming, 32nd International Colloquium (ICALP'05),* LNCS 3580, pp. 446–458, Springer Verlag, 2005.

[Wat05] B. Waters. Efficient identity-based encryption without random oracles. *EUROCRYPT '05* LNCS 3494, pp. 114–127, Springer Verlag, 2005

A Tools

We give the proof systems introduced in Sect. 6.

A.1 $\Pi_{\mathbf{1L}}$, a Perfectly WI Proof of Linearity of One of Two Triples w.r.t. Different Bases

[GOS06a] give a witness indistinguishable (WI) proof for one of two tuples being linear. We extend their ideas to construct a proof system where linearity holds w.r.t. *different* bases.

Let $(n, \mathbb{G}, \mathbb{G}_T, e)$ be a bilinear group, $f, h, g, \bar{f}, \bar{h}, \bar{g}$ be generators. Given two triples $c = (c_1, c_2, c_3)$ and $\bar{c} = (\bar{c}_1, \bar{c}_2, \bar{c}_3)$, we prove that either c is linear w.r.t. (f, h, g) or $\bar{c}$ is linear w.r.t. $(\bar{f}, \bar{h}, \bar{g})$.

Proof. In case $c = (f^r, h^s, g^{r+s})$, let $\bar{r}, \bar{s} := 0$, in case $\bar{c} = (\bar{f}^{\bar{r}}, \bar{h}^{\bar{s}}, \bar{g}^{\bar{r}+\bar{s}})$, let $r, s := 0$. Define

$$\begin{array}{lllll}
\pi_{11} := c_1^{\bar{r}} f^{t_{11}} & \pi_{12} := c_1^{\bar{s}} f^{t_{12}} & \bar{\pi}_{11} := \bar{c}_1^{r} \bar{f}^{-t_{11}} & \bar{\pi}_{12} := \bar{c}_2^{r} \bar{h}^{-t_{12}} & \bar{\pi}_{13} := \bar{c}_3^{r} \bar{g}^{-t_{11}-t_{12}} \\
\pi_{21} := c_2^{\bar{r}} h^{t_{21}} & \pi_{22} := c_2^{\bar{s}} h^{t_{22}} & \bar{\pi}_{21} := \bar{c}_1^{s} \bar{f}^{-t_{21}} & \bar{\pi}_{22} := \bar{c}_2^{s} \bar{h}^{-t_{22}} & \bar{\pi}_{23} := \bar{c}_3^{s} \bar{g}^{-t_{21}-t_{22}} \\
\pi_{31} := c_3^{\bar{r}} g^{t_{11}+t_{21}} & \pi_{32} := c_3^{\bar{s}} g^{t_{12}+t_{22}} & & &
\end{array}$$

for random $t_{11}, t_{12}, t_{21}, t_{22}$.

Verification. Check the following relations:

$$\begin{array}{rrr}
e(f, \bar{\pi}_{11}) e(\pi_{11}, \bar{f}) = e(c_1, \bar{c}_1) & e(f, \bar{\pi}_{12}) e(\pi_{12}, \bar{h}) = e(c_1, \bar{c}_2) & e(f, \bar{\pi}_{13}) e(\pi_{11}\pi_{12}, \bar{g}) = e(c_1, \bar{c}_3) \\
e(h, \bar{\pi}_{21}) e(\pi_{21}, \bar{f}) = e(c_2, \bar{c}_1) & e(h, \bar{\pi}_{22}) e(\pi_{22}, \bar{h}) = e(c_2, \bar{c}_2) & e(h, \bar{\pi}_{23}) e(\pi_{21}\pi_{22}, \bar{g}) = e(c_2, \bar{c}_3) \\
e(g, \bar{\pi}_{11}\bar{\pi}_{21}) e(\pi_{31}, \bar{f}) = e(c_3, \bar{c}_1) & e(g, \bar{\pi}_{12}\bar{\pi}_{22}) e(\pi_{32}, \bar{h}) = e(c_3, \bar{c}_2) & e(g, \bar{\pi}_{13}\bar{\pi}_{23}) e(\pi_{31}\pi_{32}, \bar{g}) = e(c_3, \bar{c}_3)
\end{array}$$

Perfect Completeness. We content ourselves to show satisfaction of the relations "in the corner", for the rest works analogously. Note that the last equality in each line follows from the fact that either $r, s = 0$ or $\bar{r}, \bar{s} = 0$

$$\begin{aligned}
&e(f, \bar{\pi}_{11}) e(\pi_{11}, \bar{f}) = e(f, \bar{c}_1^{r}) e(f, \bar{f}^{-t_{11}}) e(c_1^{\bar{r}}, \bar{f}) e(f^{t_{11}}, \bar{f}) = e(f^r, \bar{c}_1) e(c_1, \bar{f}^{\bar{r}}) = e(c_1, \bar{c}_1) \\
&e(f, \bar{\pi}_{13}) e(\pi_{11}\pi_{12}, \bar{g}) = e(f, \bar{c}_3^{r}) e(f, \bar{g}^{-t_{11}-t_{12}}) e(c_1^{\bar{r}+\bar{s}}, \bar{g}) e(f^{t_{11}+t_{12}}, \bar{g}) = e(f^r, \bar{c}_3) e(c_1, \bar{g}^{\bar{r}+\bar{s}}) = e(c_1, \bar{c}_3) \\
&e(g, \bar{\pi}_{11}\bar{\pi}_{21}) e(\pi_{31}, \bar{f}) = e(g, \bar{c}_1^{r+s}) e(g, \bar{f}^{-t_{11}-t_{21}}) e(c_3^{\bar{r}}, \bar{f}) e(g^{t_{11}+t_{21}}, \bar{f}) = e(g^{r+s}, \bar{c}_1) e(c_3, \bar{f}^{\bar{r}}) = e(c_3, \bar{c}_1) \\
&e(g, \bar{\pi}_{13}\bar{\pi}_{23}) e(\pi_{31}\pi_{32}, \bar{g}) = e(g, \bar{c}_3^{r+s}) e(g, \bar{g}^{-t_{11}-t_{12}-t_{21}-t_{22}}) e(c_3^{\bar{r}+\bar{s}}, \bar{g}) e(g^{t_{11}+t_{12}+t_{21}+t_{22}}, \bar{g}) = e(c_3, \bar{c}_3)
\end{aligned}$$

Perfect Soundness. Define

$$\begin{array}{llll}
\gamma_1 := \log_f c_1 & \bar{\gamma}_1 := \log_{\bar{f}} \bar{c}_1 & m_{1j} := \log_f \pi_{1j} & \bar{m}_{i1} := \log_{\bar{f}} \bar{\pi}_{i1} \\
\gamma_2 := \log_h c_2 & \bar{\gamma}_2 := \log_{\bar{h}} \bar{c}_2 & m_{2j} := \log_h \pi_{2j} & \bar{m}_{i2} := \log_{\bar{h}} \bar{\pi}_{i2} \\
\gamma_3 := \log_g c_3 & \bar{\gamma}_3 := \log_{\bar{g}} \bar{c}_3 & m_{3j} := \log_g \pi_{3j} & \bar{m}_{i3} := \log_{\bar{g}} \bar{\pi}_{i3}
\end{array}$$

Then $e(f, \bar{f})^{m_{11}+\bar{m}_{11}} = e(f, \bar{\pi}_{11}) e(\pi_{11}, \bar{f}) = e(c_1, \bar{c}_1) = e(f, \bar{f})^{\gamma_1\bar{\gamma}_1}$, thus $m_{11} + \bar{m}_{11} = \gamma_1\bar{\gamma}_1$. For all verification relations we get thus:

$$\begin{array}{rrr}
\bar{m}_{11} + m_{11} = \gamma_1\bar{\gamma}_1 & \bar{m}_{12} + m_{12} = \gamma_1\bar{\gamma}_2 & \bar{m}_{13} + m_{11} + m_{12} = \gamma_1\bar{\gamma}_3 \\
\bar{m}_{21} + m_{21} = \gamma_2\bar{\gamma}_1 & \bar{m}_{22} + m_{22} = \gamma_2\bar{\gamma}_2 & \bar{m}_{23} + m_{21} + m_{22} = \gamma_2\bar{\gamma}_3 \\
\bar{m}_{11} + \bar{m}_{21} + m_{31} = \gamma_3\bar{\gamma}_1 & \bar{m}_{12} + \bar{m}_{22} + m_{32} = \gamma_3\bar{\gamma}_2 & \bar{m}_{13} + \bar{m}_{23} + m_{31} + m_{32} = \gamma_3\bar{\gamma}_3
\end{array}$$

Now, these relations imply that either $\gamma_3 = \gamma_1 + \gamma_2$ or $\bar{\gamma}_3 = \bar{\gamma}_1 + \bar{\gamma}_2$, since

$$\begin{aligned}
(\gamma_1 + \gamma_2 - \gamma_3)(\bar{\gamma}_1 + \bar{\gamma}_2 - \bar{\gamma}_3) \; &= \; \gamma_1\bar{\gamma}_1 + \gamma_1\bar{\gamma}_2 - \gamma_1\bar{\gamma}_3 \; + \; \gamma_2\bar{\gamma}_1 + \gamma_2\bar{\gamma}_2 - \gamma_2\bar{\gamma}_3 \; - \; \gamma_3\bar{\gamma}_1 - \gamma_3\bar{\gamma}_2 + \gamma_3\bar{\gamma}_3 \\
&= \; \bar{m}_{11} + \bar{m}_{12} - \bar{m}_{13} \; + \; \bar{m}_{21} + \bar{m}_{22} - \bar{m}_{23} \; - \; (\bar{m}_{11} + \bar{m}_{21}) - (\bar{m}_{12} + \bar{m}_{22}) + \bar{m}_{13} + \bar{m}_{23} \; = \; 0
\end{aligned}$$

Perfect Witness-Indistinguishability. Suppose, both c and $\bar{c}$ are linear (otherwise, there is only one witness and we're done). Let π_{ij} be proofs formed with witness (r, s) (and $\bar{r} = \bar{s} = 0$) and randomness t_{ij}, and let π'_{ij} be proofs formed via witness $(\bar{r}, \bar{s})$ and the following (equally distributed) randomness:

$$\begin{array}{ll}
t'_{11} := t_{11} - r\bar{r} & t'_{12} := t_{12} - r\bar{s} \\
t'_{21} := t_{21} - s\bar{r} & t'_{22} := t_{22} - s\bar{s}
\end{array}$$

Then we get (we dispense with the cases $\pi_{2j}, \bar{\pi}_{2j}$ as they work analogously to $\pi_{1j}, \bar{\pi}_{1j}$)

$$\begin{array}{lll}
\pi_{11} = f^{t_{11}} = f^{r\bar{r}+t'_{11}} = \bar{\pi}_{11} & \pi_{12} = f^{t_{12}} = f^{r\bar{s}+t'_{12}} = \bar{\pi}_{12} & \bar{\pi}_{11} = \bar{f}^{\bar{r}r-t_{11}} = \bar{f}^{-t'_{11}} = \bar{\pi}'_{11} \\
& \bar{\pi}_{12} = \bar{f}^{\bar{s}r-t_{12}} = \bar{f}^{-t'_{12}} = \bar{\pi}'_{12} & \bar{\pi}_{13} = \bar{g}^{(\bar{r}+\bar{s})r-t_{11}-t_{12}} = \bar{g}^{-t'_{11}-t'_{12}} = \bar{\pi}'_{13} \\
\pi_{31} = g^{t_{11}+t_{21}} = g^{r\bar{r}+t'_{11}+s\bar{r}+t'_{21}} = \pi'_{31} & & \pi_{32} = g^{t_{12}+t_{22}} = g^{r\bar{s}+t'_{12}+s\bar{s}+t'_{22}} = \pi'_{32}
\end{array}$$

A.2 $\Pi_{\mathbf{b,eq}}$, Proof for a Commitment and a Ciphertext Containing the Same Bit

Using the WI proofs of the previous section, we can easily give a proof of the following: Given $c = \mathsf{Com}(ck, x\,;\; r, s)$ and $d = \mathsf{Enc}(pk, g^{x'};\; r', s')$, with $ck = (v, w, u, f, h, z)$ and $pk = (f', h', z')$, prove that $x = x'$ and $x \in \{0,1\}$.

$\mathsf{P}_{\mathrm{b,eq}}\big(crs, (c,d), (x, r_0, s_0, r_1, s_1)\big) := (\pi_1, \pi_2)$ with

- $\pi_1 := \mathsf{P}_{1L}\big((f,h,z,f',h',z'),\ ((c_1,c_2,c_3),(d_1,d_2,d_3g^{-1})),\ (r_x,s_x)\big)$
- $\pi_2 := \mathsf{P}_{1L}\big((f,h,z,f',h',z'),\ ((c_1v^{-1},c_2w^{-1},c_3u^{-1}),(d_1,d_2,d_3)),(r_{1-x},s_{1-x})\big)$

Now depending on whether ck is binding or hiding, we get either soundness or simulation:

ck **binding** Suppose both proofs pass verification. (1) if c is linear, $(c_1v^{-1}, c_2w^{-1}, c_3u^{-1})$ is non-linear, so by π_2, d must be linear; c is thus a commitment to 0, and d an encryption of g^0. (2) if on the other hand (d_1, d_2, d_3g^{-1}) is linear, d is not, thus $(c_1v^{-1}, c_2w^{-1}, c_3u^{-1})$ is linear, again by π_2; thus, c is a commitment to 1, and d an encryption of g^1.

ck **hiding** Now if the commitment key is perfectly hiding and given r_v, s_w such that $v = f^{r_v}, w = h^{s_w}$, the proof can be simulated given the randomness in c only: let c be a commitment to 0 using (r,s), then $\mathsf{Sim}_{\mathrm{b,eq}}\big((crs, r_v, s_w), (c,d), (r_0,s_0)\big) := (\pi_1,\pi_2)$ where π_1 is constructed using (r,s) and π_2 using $(r - r_v, s - s_w)$.

A.3 $\Pi_{\mathbf{cX}}$, Proof for Commitments to the Bits of a Logarithm

Let $(n, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot), g)$ be a bilinear group, let (a,b,g) be a binding commitment key for base (f,h,e). Let $X = g^x$, $(c_i)_{i=0}^{m-1}$ be commitments to $x_i \in \{0,1\}$. We prove that $x = \sum x_i 2^i$.

CRS generation choose $(n, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot), g) \leftarrow \mathcal{G}(1^\lambda)$; $f, h \leftarrow \mathbb{G}^*$; $r_a, s_b \leftarrow \mathbb{Z}_p$; $a := f^{r_a}$, $b = h^{s_b}$, $e := g^{(r_a+s_b+1)^{-1}}$. Define $ck := (f,h,e,a,b,g)$; return $crs := (n, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot), ck)$.

Proof The witnesses are $(x_i, r_i, s_i)_{i=0}^{m-1}$ s.t. $c_i = \mathsf{Com}(ck, x_i;\, r_i, s_i)$; let $x := \sum x_i 2^i$. The proof is (A,B,L) with $A := a^x$, $B := b^x$, $L := e^{\sum r_i 2^i}$.

Verification given $crs = (n, \mathbb{G}, \mathbb{G}_t, e(\cdot,\cdot), a, b, g, f, h, e)$, statement $(X, (c_i))$ and proof (A,B,L)

- check $e(A,g) \stackrel{?}{=} e(a,X)$ and $e(B,g) \stackrel{?}{=} e(b,X)$ (V1)
- check $e(A^{-1}\prod c_{i,1}^{2^i}, e) \stackrel{?}{=} e(f,L)$ and $e(B^{-1}\prod c_{i,2}^{2^i}, e) \stackrel{?}{=} e(h, X^{-1}L^{-1}\prod c_{i,3}^{2^i})$ (V2)

Completeness (V1): trivial.

$$\text{(V2)}:\quad e(A^{-1}\textstyle\prod c_{i,1}^{2^i}, e) = e(a^{-x}a^{\sum x_i 2^i} f^{\sum r_i 2^i}, e) = e(f^{\sum r_i 2^i}, e) = e(f, L)$$

$$e(B^{-1}\textstyle\prod c_{i,2}^{2^i}, e) = e(b^{-x}b^{\sum x_i 2^i} h^{\sum s_i 2^i}, e) = e(h, e^{\sum s_i 2^i}) =$$
$$= e(h, g^{-x}e^{-\sum r_i 2^i} g^{\sum x_i 2^i} e^{\sum (r_i+s_i)2^i}) = e(h, X^{-1}L^{-1}\textstyle\prod c_{i,3}^{2^i})$$

Soundness Let $x = \log_g X$ and (A,B,L) a proof that passes verification. From (V1) we have $A = a^x$ and $B = b^x$. Let $c_i = \mathsf{Com}(\alpha_i;\, r_i, s_i)$; define $\alpha = \sum \alpha_i 2^i$ and consider (V3). From

$$e(f,L) \;=\; e(A^{-1}\textstyle\prod c_{i,1}^{2^i}, e) = e(a^{-x}a^{\sum \alpha_i 2^i} f^{\sum r_i 2^i}, e) = e(f^{r_a(\alpha - x) + \sum r_i 2^i}, e)$$

we get $L = e^{r_a(\alpha-x)+\sum r_i 2^i}$. Analogously we have

$$e(B^{-1}\textstyle\prod c_{i,2}^{2^i}, e) = e(b^{-x}b^{\sum \alpha_i 2^i} h^{\sum s_i 2^i}, e) = e(h^{s_b(\alpha-x)+\sum s_i 2^i}, e) \tag{1}$$

and on the other hand (let $g = e^t$),

$$e(h, X^{-1}L^{-1}\prod c_{i,3}^{2^i}) = e(h, g^{-x}e^{-r_a(\alpha-x)-\sum r_i 2^i} g^\alpha e^{\sum(r_i+s_i)2^i}) = e(h, e^{t(\alpha-x)-r_a(\alpha-x)+\sum s_i 2^i}) \qquad (2)$$

Since by (V3), the leftmost terms in (1) and (2) are equivalent, considering the exponents, we get $(r_a + s_b - t)(\alpha - x) = 0$, thus $\alpha = x$, otherwise (a, b, g) were not a binding commitment key.

Zero Knowledge – simulated CRS: choose $\phi, \psi, r_a, s_b \leftarrow \mathbb{Z}_p$ and define $f := g^\phi$, $h := g^\psi$, $a := f^{r_a}$, $b := h^{s_b}$, $e := g^{(r_a+s_b)^{-1}}$. The trapdoor is $tr := (\phi, \psi, r_a, s_b)$. Since (a, b, g) in the simulated CRS is linear w.r.t (f, h, e), while in the real one it is not, they are indistinguishable by the decisional linear assumption.

– simulated proof: given X, (c_i); let $x := \log_g X$ be unknown.

  – define $A := X^{\phi r_a}$ and $B := X^{\psi s_b}$. (perfect simulations since $a = g^{\phi r_a}, b = g^{\psi s_b}$.)
  – define $L := \left(\prod c_{i,1}^{2^i\phi^{-1}} X^{-r_a}\right)^{(r_a+s_b)^{-1}}$.

To see that The simulation is perfect, let r_i, s_i s.t. $c_{i,1} = f^{r_i}, c_{i,2} = h^{s_i}$. Setting $\tilde{r}_i := r_i - r_a x_i$ and $\tilde{s}_i := s_i - s_b x_i$, we have $c_i = (f^{r_a x_i + \tilde{r}_i}, h^{s_b x_i + \tilde{s}_i}, e^{(r_a+s_b)x_i + \tilde{r}_i + \tilde{s}_i}) = (a^{x_i} f^{\tilde{r}_i}, b^{x_i} h^{\tilde{s}_i}, g^{x_i} e^{\tilde{r}_i + \tilde{s}_i})$ and

$$L = \left(\prod c_{i,1}^{2^i\phi^{-1}} X^{-r_a}\right)^{(r_a+s_b)^{-1}} = (f^{\sum r_i 2^i})^{\phi^{-1}(r_a+s_b)^{-1}} (g^{\sum x_i 2^i})^{-r_a(r_a+s_b)^{-1}} = e^{\sum r_i 2^i} e^{-r_a \sum x_i 2^i} = e^{\sum (r_i - r_a x_i) 2^i} = e^{\sum \tilde{r}_i 2^i}$$

Remark 11. Proof system Π_{cX} can be used to construct a proof of knowledge of logarithm: given statement X and witness $x = \sum_{i=0}^{m-1} x_i 2^i$ s.t. $X = g^x$, produce a *crs* for Π_{cX}. Next, for all $0 \le i < m$, define $c_i := \mathsf{Com}(crs, x_i; r_i, s_i)$ and prove that the committed values are in $\{0, 1\}$ via $\mathsf{P}_{1\mathrm{L}}(\ldots, (c_i, (c_{i1}a^{-1}, c_{i2}b^{-1}, c_{i3}g^{-1})), \ldots)$. Finally, run $\mathsf{P}_{\mathrm{cX}}(crs, (X, (c_i)), \ldots)$. Now from each c_i, we can extract the committed bit using extraction key $(\log_e f, \log_e h)$.

A.4 Π_{cF}, Proof for Commitments to the Bits of a Hash Preimage

Let $(n, \mathbb{G}, \mathbb{G}_T, e, g)$ be a bilinear group, let $(u_i)_{i=1}^m$ be a basis for the W-signature scheme, $(v_i, w_i, u_i, f_i, h_i, z_i)_{i=1}^m$ be a special binding commitment keys as constructed below. Given a hash value F and commitments $(c_i)_{i=1}^m$ to $x_i \in \{0, 1\}$, we show that the c_i are commitments to the bits of a preimage of F.

Common Reference String given $(n, \mathbb{G}, \mathbb{G}_T, e)$ and $(u_i)_{i=1}^m \in \mathbb{G}^\lambda$. Choose $\phi, \psi, r_v, s_w \leftarrow \mathbb{Z}_p$ and define for $1 \le i \le m$: $f_i := u_i^\phi$, $h_i := u_i^\psi$, $v_i := f_i^{r_v}$, $w_i := h_i^{s_w}$, $z_i := u_i^{(r_v+s_w+1)^{-1}}$. The common reference string is $crs := (n, \mathbb{G}, \mathbb{G}_T, e, (ck_i)_{i=1}^m)$ with $ck_i := (f_i, h_i, z_i, v_i, w_i, u_i)$.

Proof let the witness be $x_1, \ldots, x_m \in \{0, 1\}$, s.t. $F = \mathcal{F}(x_1, \ldots x_m)$ and (r_i, s_i), the randomness used for c_i (i.e., $c_i = \mathsf{Com}(ck_i, x_i; (r_i, s_i))$). The proof is $\pi = (V, W, Z)$ with $V := \prod v_i^{x_i}$, $W := \prod w_i^{x_i}$ and $Z := \prod z_i^{r_i}$.

Verification given *crs*, statement $(F, (c_i))$ and proof (V, W, Z), return 1 if the following hold

– $e(V, u_1) \stackrel{?}{=} e(F, v_1)$ and $e(W, u_1) \stackrel{?}{=} e(F, w_1)$ (V1)
– $e(V^{-1} \prod c_{i,1}, z_1) \stackrel{?}{=} e(f_1, Z)$ and $e(W^{-1}\prod c_{i,2}, z_1) \stackrel{?}{=} e(h_1, F^{-1} Z^{-1} \prod c_{i,3})$ (V2)

Simulation – simulated CRS: As the real CRS, except that $z_i := u_i^{(r_v+s_w)^{-1}}$.

– simulated proof: given F, (c_i), define $V := F^{r_v}$, $W := F^{r_w}$ and $Z := \left(\prod c_{i,1}^{\phi^{-1}} F^{-r_v}\right)^{(r_v+s_w)^{-1}}$

Note that Π_{cX} is a special case of Π_{cF}, setting $u_i := g^{2^{i-1}}$, since $\mathcal{F}(x) = \prod u_i^{x_i} = g^{\sum x_i 2^{i-1}} = X$.

A.5 Π_{G}, Groth's WI Proof of Two Encryptions Having the Same Plaintext or a Commitment Containing a Specific Value

Due to limited space we will use the following WI proof as a black box (cf. the full version of [Gro06] for an implementation that respects the pairing-product-equation paradigm): Given two encryptions d, d' under public keys for linear encryption pk, pk', resp., a commitment c under commitment key ck, and v, a vector of group elements. Then Π_{G} enables to give a WI proof for "either d and d' contain the same plaintext or c is a commitment to v".

The statement $(pk, pk', d, d', ck, c, v)$ can be proven using either witness (m, r, r') s.t. $d = \mathsf{Enc}(pk, m;\ r)$ and $d' = \mathsf{Enc}(pk, m;\ r')$ or witness (v, r) s.t. $c = \mathsf{Com}(ck, v;\ r)$.

B Security Definitions for Anonymous Proxy Signatures

$\mathbf{Exp}^{\text{anon-b}}_{\mathcal{PS},\mathcal{A}}(\lambda, \Lambda)$

- $(pp, ik, ok) \leftarrow \mathsf{Setup}(1^\lambda, \Lambda)$
- $(st, pk, (sk^0, warr^0), (sk^1, warr^1), M) \leftarrow \mathcal{A}_1(pp, ik)$
- for $c = 0\,..\,1$
  - $\Sigma^c \leftarrow \mathsf{PSig}(pp, sk^c, warr^c, M)$
  - if $\mathsf{PVer}(pp, pk, M, \Sigma^0) = 0$, return 0
  - $(pk_2^c, \ldots, pk_{k_c}^c) \leftarrow \mathsf{Open}(pp, ok, M, \Sigma^c)$
- if $k_0 \neq k_1$, return 0
- $d \leftarrow \mathcal{A}_2(st, \Sigma^b)$
- return d

$\mathbf{Exp}^{\text{trace}}_{\mathcal{DS},\mathcal{A}}(\lambda, \Lambda)$

- $(pp, ik, ok) \leftarrow \mathsf{Setup}(1^\lambda, \Lambda)$
- $(pk, M, \Sigma) \leftarrow \mathcal{A}(pp, ok : \mathsf{Enroll})$
- if $\mathsf{PVer}(pp, pk, M, \Sigma) = 1$ and $\mathsf{Open}(pp, ok, M, \Sigma) = \perp$ return 1
  - return 0

$\mathbf{Exp}^{\text{n-frame}}_{\mathcal{DS},\mathcal{A}}(\lambda, \Lambda)$

- $(pp, ik, ok) \leftarrow \mathsf{Setup}(1^\lambda, \Lambda);\ HU := \emptyset$
- $(pk_1, M, \Sigma) \leftarrow \mathcal{A}(pp, ik, ok : \mathsf{PK}, \mathsf{SK}, \mathsf{Del}, \mathsf{PSig})$
- if $\mathsf{PVer}(pp, pk, M, \Sigma) = 0$ or $\mathsf{Open}(pp, ok, M, \Sigma) = \perp$, return 0
- let $(pk_2, \ldots, pk_k) = \mathsf{Open}(pp, ok, M, \Sigma)$
- if for some $1 \leq i < k$, $pk_i \in HU$ and no query $\mathsf{Del}(pk_i, warr, pk_{i+1})$
  with $\mathsf{Open}(warr) = (pk_1, \ldots, pk_i)$, return 1
- if $pk_k \in HU$ and no query $\mathsf{PSig}(pk_k, warr, M)$
  with $\mathsf{Open}(warr) = (pk_1, \ldots, pk_k)$, return 1
- return 0

Figure 1: Experiments for anonymity, traceability and non-frameability

We review the slightly adapted security notions from [FP08]. In particular, we do not consider delegation for specific tasks (although this can be easily included in our scheme, cf. Remark 3), we content ourselves to having only one general opener and our version of anonymity is CPA rather than CCA-2 as in their model.

Traceability and non-frameability are defined as any adversary being able to win games $\mathbf{Exp}^{\text{trace}}$, $\mathbf{Exp}^{\text{n-frame}}$, resp., with at most negligible probability. Anonymity holds if no adversary can distinguish

$\mathbf{Exp}^{\text{anon-0}}$ from $\mathbf{Exp}^{\text{anon-1}}$ (see Fig. 1 for the definitions of the experiments).

In the experiments, the adversary disposes over a subset of the following oracles:

$\mathsf{Enroll}(pk)$ Enroll pk using the issuer's key.

PK Create a random public/private key pair (pk, sk) and add it to HU, the list of honest users; output pk.

$\mathsf{SK}(pk)$ If pk is in HU, delete the entry and return the corresponding sk.

$\mathsf{Del}(pk, warr, pk')$ If pk is in HU, look up the corresponding sk and return $\mathsf{Del}(pp, sk, warr, pk')$. Otherwise return $\perp$.

$\mathsf{PSig}(pk, warr, M)$ If pk is in HU, look up the corresponding sk and return $\mathsf{PSig}(pp, sk, warr, M)$. Otherwise return $\perp$.

Definition 12 (Anonymity)**.** A signature delegation scheme $\mathcal{DS}$ is *anonymous* if for any p.p.t. adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the following is negligible in λ:

$$\left|\Pr\left[\mathbf{Exp}^{\text{anon-1}}_{\mathcal{PS},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{anon-0}}_{\mathcal{PS},\mathcal{A}}(\lambda) = 1\right]\right|$$

Definition 13 (Traceability)**.** A signature delegation scheme $\mathcal{DS}$ is *traceable* if for any p.p.t. adversary $\mathcal{A}$, $\Pr\left[\mathbf{Exp}^{\text{trace}}_{\mathcal{PS},\mathcal{A}}(\lambda) = 1\right]$ is negligible in λ.

Definition 14 (Non-frameability)**.** A signature delegation scheme $\mathcal{DS}$ is *non-frameable* if for any p.p.t. adversary $\mathcal{A}$, $\Pr\left[\mathbf{Exp}^{\text{n-frame}}_{\mathcal{PS},\mathcal{A}}(\lambda) = 1\right]$ is negligible in λ.

C Proofs

C.1 Proof of Lemma 1

Proof of Lemma 1(1)*.* Follows from bilinearity of $e(\cdot,\cdot)$.

Proof of Lemma 1(3)*.* Consider $E_{(a_j c_j, b_j d_j)_j}$:

$$\begin{aligned}&\prod e\big(a_j c_j \textstyle\prod (X_i Y_i)^{\delta_{j,i}}, b_j d_j \prod (X_i Y_i)^{\varepsilon_{j,i}}\big) = \\ &\quad\prod e\big(a_j \textstyle\prod X_i^{\delta_{j,i}}, b_j \prod X_i^{\varepsilon_{j,i}}\big) e\big(a_j \prod X_i^{\delta_{j,i}}, d_j \prod Y_i^{\varepsilon_{j,i}}\big) e\big(c_j \prod Y_i^{\delta_{j,i}}, b_j \prod X_i^{\varepsilon_{j,i}}\big) e\big(c_j \prod Y_i^{\delta_{j,i}}, d_j \prod Y_i^{\varepsilon_{j,i}}\big) = \\ &\qquad\qquad\prod e\big(a_j \textstyle\prod X_i^{\delta_{j,i}}, b_j \prod X_i^{\varepsilon_{j,i}}\big) \prod e\big(c_j \prod Y_i^{\delta_{j,i}}, d_j \prod Y_i^{\varepsilon_{j,i}}\big) = 1\end{aligned}$$

where the second equation holds, because the two pairings in the middle of the second line are pairings of elements of $\mathbb{G}_p$ with elements of $\mathbb{G}_q$ and are therefore 1.

Proof of Lemma 1(4)*.* Let $(\widetilde{X}_i)$ satisfy $\tilde{E}_{(a_j c_j, b_j d_j)_j}$. H is of order q and consequently so is the right hand side of the equation, thus raising $(\widetilde{E})$ to the power of θ^2 yields

$$1 = e\big(H^\theta, P_E^\theta\big) = \textstyle\prod_j e\big((a_j c_j)^\theta \prod_i \widetilde{X}_i^{\theta\delta_{j,i}}, (b_j d_j)^\theta \prod_j \widetilde{X}_i^{\theta\varepsilon_{j,i}}\big) = \prod_j e\big(a_j \prod_i (\widetilde{X}_i^\theta)^{\delta_{j,i}}, b_j \prod_j (\widetilde{X}_i^\theta)^{\varepsilon_{j,i}}\big)$$

Proof of Lemma 1(2)*.* Let g be a generator of $\mathbb{G}$. For all i, define: $x_i := \log_g X_i$, $x'_i := \log_g X'_i$, $\alpha_i := \log_g a_i$, $\beta_i := \log_g b_i$ and $\kappa := \log_g H$. Note that for all i, $X_i H^{\rho_i} = X'_i H^{\rho'_i}$ implies

$$x'_i - x_i = \kappa(\rho_i - \rho'_i). \tag{3}$$

We show that $\Delta := \log_g P_E((X_i), (\rho_i)) - \log_g P_E((X'_i), (\rho'_i)) = 0$.

$$\begin{aligned}\Delta \ = \ \textstyle\sum_j \Big(&(\alpha_j + \sum_i \delta_{j,i} x_i)(\sum_i \varepsilon_{j,i}\rho_i) + (\beta_j + \sum_i \varepsilon_{j,i} x_i)(\sum_i \delta_{j,i}\rho_i) + \kappa(\sum_i \delta_{j,i}\rho_i)(\sum_i \varepsilon_{j,i}\rho_i) \\ &- (\alpha_j + \textstyle\sum_i \delta_{j,i} x'_i)(\sum_i \varepsilon_{j,i}\rho'_i) - (\beta_j + \sum_i \varepsilon_{j,i} x'_i)(\sum_i \delta_{j,i}\rho'_i) - \kappa(\sum_i \delta_{j,i}\rho'_i)(\sum_i \varepsilon_{j,i}\rho'_i)\Big) \ = \ A + B\end{aligned}$$

with $A := \sum\Big(\alpha_j \sum_i \varepsilon_{j,i}(\rho_i - \rho'_i) + \beta_j \sum_i \delta_{j,i}(\rho_i - \rho'_i)\Big)$ and

$B := \sum\Big(\sum_i \delta_{j,i}(x_i + \kappa\rho_i)\sum_i \varepsilon_{j,i}\rho_i - \sum_i \delta_{j,i}(x'_i + \kappa\rho'_i)\sum_i \varepsilon_{j,i}\rho'_i + \sum_i \varepsilon_{j,i}x_i \sum_i \delta_{j,i}\rho_i - \sum_i \varepsilon_{j,i}x'_i \sum_i \delta_{j,i}\rho'_i\Big)$.

Considering the logarithms of $E((X_i))$ and $E((X'_i))$ we get

$$\sum(\alpha_j + \sum_i \delta_{j,i}x_i)(\beta_j + \sum_i \varepsilon_{j,i}x_i) = 0 = \sum(\alpha_j + \sum_i \delta_{j,i}x'_i)(\beta_j + \sum_i \varepsilon_{j,i}x'_i)$$

which, subtracting the left-hand from the right-hand side, yields

$$\sum\Big(\alpha_j \sum_i \varepsilon_{j,i}(x'_i - x_i) + \beta_j \sum_i \delta_{j,i}(x'_i - x_i) + \sum_i \delta_{j,i}x'_i \sum_i \varepsilon_{j,i}x'_i - \sum_i \delta_{j,i}x_i \sum_i \varepsilon_{j,i}x_i\Big) = 0$$

$$\text{and from (3):}\ \sum\Big(\kappa\big(\alpha_j \sum_i \varepsilon_{j,i}(\rho_i - \rho'_i) + \beta_j \sum_i \delta_{j,i}(\rho_i - \rho'_i)\big) + \sum_i \delta_{j,i}x'_i \sum_i \varepsilon_{j,i}x'_i - \sum_i \delta_{j,i}x_i \sum_i \varepsilon_{j,i}x_i\Big) = 0$$

thus $\quad A = \frac{1}{\kappa}\sum\big(\sum_i \delta_{j,i}x_i \sum_i \varepsilon_{j,i}x_i - \sum_i \delta_{j,i}x'_i \sum_i \varepsilon_{j,i}x'_i\big)$.

On the other hand, since $x_i + \kappa\rho_i = x'_i + \kappa\rho'_i$, (note that we subtract and add $\sum \delta_{j,i}\rho_i \sum \varepsilon_{j,i}x'_i$)

$$\begin{aligned}
B &= \sum\Big(\sum_i \delta_{j,i}(x_i + \kappa\rho_i)\sum_i \varepsilon_{j,i}(\rho_i - \rho'_i) - \sum_i \delta_{j,i}\rho_i \sum_i \varepsilon_{j,i}(x'_i - x_i) + \sum_i \varepsilon_{j,i}x'_i \sum_i \delta_{j,i}(\rho_i - \rho'_i)\Big) \\
&= \sum\Big(\frac{1}{\kappa}\sum_i \delta_{j,i}(x_i + \kappa\rho_i)\sum_i \varepsilon_{j,i}(x'_i - x_i) - \sum_i \delta_{j,i}\rho_i \sum_i \varepsilon_{j,i}(x'_i - x_i) + \frac{1}{\kappa}\sum_i \varepsilon_{j,i}x'_i \sum_i \delta_{j,i}(x'_i - x_i)\Big) \\
&= \frac{1}{\kappa}\sum\Big(\sum_i \delta_{j,i}x_i \sum_i \varepsilon_{j,i}(x'_i - x_i) + \sum_i \varepsilon_{j,i}x'_i \sum_i \delta_{j,i}(x'_i - x_i)\Big) \\
&= \frac{1}{\kappa}\sum\Big(-\sum_i \delta_{j,i}x_i \sum_i \varepsilon_{j,i}x_i + \sum_i \varepsilon_{j,i}x'_i \sum_i \delta_{j,i}x'_i\Big) \;=\; -A
\end{aligned}$$

□

C.2 Proof sketch of Theorem 10

We define a sequence of games:

GAME 0 := $\mathbf{Exp}^{\mathrm{zk}}_{\Pi,\mathcal{A}}$, where Ext queries are answered using ξ'. In addition, whenever $\mathcal{A}_2$ makes a valid query involving (X, F), the statement output by $\mathcal{A}_1$ together with a witness x, we simply output x. Since x is the only witness, Game 0 and $\mathbf{Exp}^{\mathrm{zk}}_{\Pi,\mathcal{A}}$ are indistinguishable by correctness of extraction.

GAME 1 Define crs to be a simulated reference string (triples (a, b, g) and (v_i, w_i, u_i) become linear w.r.t. (f, h, e) and (f_i, h_i, e_i), respectively. Indistinguishability of Games 0 and 1 is implied by the decisional linear assumption.

GAME 2 In $\mathsf{P}_{\mathrm{X}\leftrightarrow\mathrm{F}}(crs, (X, F))$, set $(vk, sk) := (vk_G, sk_G)$, the pair from tr, instead of running $\mathsf{K}_{\mathrm{ots}}$. Games 1 and 2 are indistinguishable by the computational hiding property of c_G.

GAME 3 Compute π by $\mathsf{Sim}_{\mathrm{X}\leftrightarrow\mathrm{F},2}$, except that d'_i, d''_i remain encryptions of x_i. The probabilities of Game 2 and 3 are equivalent, since all we did is compute the WI proofs using different witnesses: Commitments $c_{X\,i}$ and $c_{F\,i}$ are already linear in Game 2 due to the simulated CRS. Proofs π_X, π_F, π'_i and π''_i are computed using different witnesses.

GAME 4 We replace encryptions d''_i by encryptions of 0. Games 3 and 4 are indistinguishable by a hybrid argument on semantic security of encryptions. (Note that the distinguisher can perfectly simulate both games, since we use ξ' to extract and because the WI proofs do not use the witnesses of d''_i.)

GAME 5 The Ext-oracle uses ξ'' instead of ξ'. The only difference between Games 4 and 5 would be if the adversary queried some $((X^*, F^*), \pi^*)$ with $(X^*, F^*) \neq (X, F)$ and π^* containing $(d'_i)^*$ encrypting x'_i and $(d''_i)^*$ encrypting $x''_i \neq x'_i$ for some i. However, in this case $\prod((d'_i)^*)^{2^{i-1}}$ and $\prod((d''_i)^*)^{2^{i-1}}$ contain different ciphertexts. Now soundness of π^*_G implies that $vk^* = vk_G$, while on the other hand σ^* is a signature on $(X^*, F^*) \neq (X, F)$ (the statement output by $\mathcal{A}_1$), which is valid w.r.t. vk_G. The distance between Games 4 and 5 is thus upper-bounded by the advantage of a forger against the one-time signature scheme

GAME 6 Replace encryptions d'_i by encryptions to 0. Game 5 and 6 are indistinguishable analogously to Game 3 and 4. Game 6 is indistinguishable from $\mathbf{Exp}^{\text{zk-S}}_{\Pi,\mathcal{A}}$.

C.3 Proof of Claim 8

Relying on the security of Waters signatures, we give an abstracted version of its proof, which in addition enables us to use some of its components in our proof: Unforgeability of Waters signatures is shown by specifying three algorithms SimPars, SimSig and Extract in order to employ an EUF-CMA-adversary $\mathcal{A}$ against a scheme for m-bit messages making ℓ signing queries to solve a CDH-instance (X, Y) as follows:

$$(sPar, tr) \leftarrow \mathsf{SimPar}(X, Y, m, \ell)$$
$$(M, \sigma) \leftarrow \mathcal{A}(sPar, X : \mathsf{SimSig}(tr, \cdot))$$
$$Z \leftarrow \mathsf{Extract}(tr, M, \sigma)$$

Some probability analysis then shows that if the adversary never queried M, the above experiment returns a CDH-solution with probability $\varepsilon' \geq \frac{\varepsilon}{4m\ell}$, where ε is the adversary's advantage. In particular, the probability that all SimSig queries are answered correctly is $\frac{1}{2}$, whereas the probability that Extract actually transforms a valid forgery to a CDH-solution is $\frac{1}{2m\ell}$.

Let $\mathcal{A}$ be an adversary against non-frameability of $\mathcal{DS}$ making at most κ PK-queries and at most ℓ Del and PSig (together) queries per user. We define a series of games, starting with the original experiment:

Game 0: The Real Game

$\mathbf{Exp}^{\text{euf-nf}}_{\mathcal{DS},\mathcal{A}}(\lambda, \Lambda)^{(0)}$

Setup:
- $gPar := (p, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot), g) \leftarrow \mathcal{G}(1^\lambda)$
- $sPar := (\bar{g}, \bar{u}, (u_{i,j})_{1\leq i\leq\Lambda, 1\leq j\leq m}) \leftarrow \mathbb{G}^{\Lambda m+2}$
- for $i = 1 \ldots \Lambda$: $crs_i \leftarrow \mathsf{K}_{\mathrm{X}\leftrightarrow\mathrm{F}}(gPar, (u_{i,j})_{j=1}^m)$
- $ik := \omega \leftarrow \mathbb{Z}_p$; $\Omega := g^\omega$; $pp := (gPar, sPar, \vec{crs}, \Omega)$; $HU := \emptyset$

$\big((X_1, F_{1,1}, P_{1,1} \ldots), M, (\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=1}^k)\big)$
$\qquad \leftarrow \mathcal{A}\big(pp, \omega : \mathsf{PK}^{(0)}, \mathsf{SK}^{(0)}, \mathsf{Del}^{(0)}, \mathsf{PSig}^{(0)}\big)$
if $\mathsf{PVer}((X_1, F_{1,1}, P_{1,1} \ldots), M, (\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=1}^k)) = 0$, return 0
if $\exists\, 1 \leq i < k$: $X_i \in HU$ and no query $\mathsf{Del}\big(X_i, ((X_j \ldots)_{j=1}^{i-1}, X_i \ldots), X_{i+1}\big)$
  return 1
if $X_k \in HU$ and no query $\mathsf{PSig}\big(X_k, ((X_j \ldots)_{j=1}^{k-1}, X_k \ldots), M\big)$
  return 1
return 0

with the following oracles:

$\mathsf{PK}^{(0)}()$

$x \leftarrow \mathbb{Z}_{2^m};\ X := g^x$
for $i = 1 \ldots \Lambda$
  $F_i := \mathcal{F}_i(x)$
  $P_i \leftarrow \mathsf{P}_{\mathrm{X}\leftrightarrow\mathrm{F}}(crs_i, (X, F_i), x)$
  $cert_i \leftarrow \mathsf{FSig}(\omega, F_i)$
add $(X, (F_i, P_i, cert_i), x)$ to HU
return $(X, (F_i, P_i))$

$\mathsf{SK}^{(0)}(X)$

if $\exists\, x$: $(X, \ldots, x) \in HU$
  delete the entry
  return x
return $\perp$

$\mathsf{Del}^{(0)}(X, warr, (X', F', P'))$

if $\exists\, i_X :\ HU[i_X] = (X \ldots)$
  return $\mathsf{Del}(HU[i_X], warr, (X', F', P'))$
return $\perp$

$\mathsf{PSig}^{(0)}(X, warr, M)$

if $\exists\, i_X :\ HU[i_X] = (X \ldots)$
  return $\mathsf{PSig}(HU[i_X], warr, M)$
return $\perp$

Game 1: Choosing Target X^* and Double-Checking the Proofs

$\mathbf{Exp}^{\text{euf-nf}}_{\mathcal{DS},\mathcal{A}}(\lambda, \Lambda)^{(1)}$

$gPar := (p, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot), g) \leftarrow \mathcal{G}(1^\lambda)$
$i_K^* \leftarrow \{1, \ldots, \kappa\};\ i_K := 0$
$x^* \leftarrow \mathbb{Z}_{2^m};\ X^* := g^{x^*};$ for $i = 1 \ldots \Lambda :\ F_i^* := \mathcal{F}_i(x^*)$
$sPar := (\bar{g}, \bar{u}, (u_{i,j})_{1 \le i \le \Lambda, 1 \le j \le m}) \leftarrow \mathbb{G}^{\Lambda m + 2}$
for $i = 1 \ldots \Lambda :\ crs_i \leftarrow \mathsf{K}_{\mathrm{X}\leftrightarrow\mathrm{F}}(gPar, (u_{i,j})_{j=1}^m)$
$\omega \leftarrow \mathbb{Z}_p;\ \Omega := g^\omega;\ pp := (gPar, sPar, \vec{crs}, \Omega);\ HU := \emptyset$
$\big((X_1, F_{1,1}, P_{1,1} \ldots), M, (\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=1}^k)\big)$
  $\leftarrow \mathcal{A}\big(pp, \omega\ :\ \mathsf{PK}^{(1)}, \mathsf{SK}^{(1)}, \mathsf{Del}^{(1)}, \mathsf{PSig}^{(1)}\big)$
if $\mathsf{PVer}^*((X_1, F_{1,1}, P_{1,1} \ldots), M, (\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=1}^k)) = 0$, return 0
if $\exists\, 1 \le i < k$: $X_i = X^*$ and no query $\mathsf{Del}\big(X^*, ((X_j \ldots)_{j=1}^{i-1}, X^* \ldots), X_{i+1}\big)$
  return 1
if $X_k = X^*$ and no query $\mathsf{PSig}\big(X^*, ((X_j \ldots)_{j=1}^{k-1}, X^* \ldots), M\big)$
  return 1
return 0

$\mathsf{PK}^{(1)}()$

if $i_K = i_K^*$
  for $i = 1 \ldots \Lambda$
    $P_i^* \leftarrow \mathsf{P}_{\mathrm{X}\leftrightarrow\mathrm{F}}(crs_i, (X^*, F_i^*), x^*)$
    $cert_i^* \leftarrow \mathsf{FSig}(\omega, F_i^*)$
  define $sk^* := (X^*, (F_i^*, P_i^*), x^*)$
  $i_K := i_K + 1$; return $(X^*, (F_i^*, P_i^*))$
else $i_K := i_K + 1$; return $\mathsf{PK}^{(0)}()$

$\mathsf{SK}^{(1)}(X)$

if $X = X^*$ ABORT GAME
  return $\mathsf{SK}^{(0)}(X)$

$\mathsf{Del}^{(1)}(X, warr, (X', F', P'))$

if $X = X^*$
  return $\mathsf{Del}^*(x^*, warr, (X', F', P'))$
return $\mathsf{Del}^{(0)}(X, warr, (X', F', P'))$

$\mathsf{PSig}^{(1)}(X, warr, M)$

if $X = X^*$
  return $\mathsf{PSig}^*(x^*, warr, M)$
return $\mathsf{PSig}^{(0)}(X, warr, M)$

with PVer^*, Del^* and PSig^* working as their non-starred version, except that $\mathsf{V}_{\mathrm{X}\leftrightarrow\mathrm{F}}$ is replaced by

$\mathsf{V}^*_{\mathrm{X}\leftrightarrow\mathrm{F}}(crs_i, (X, F), P)$

1 if $\mathsf{V}_{\mathrm{X}\leftrightarrow\mathrm{F}}(crs_i, (X, F), P) = 0$, return 0
2 if $(X, F) = (X^*, F_i^*)$, return 1
3 $x \leftarrow \mathsf{Ext}_{\mathrm{X}\leftrightarrow\mathrm{F}}(ek_i, (X, F), P)$
4 if $X \neq g^x$ or $F \neq \mathcal{F}_i(x)$, return 0
5 return 0

Game 0 $\curvearrowright$ Game 1 First of all, soundness of $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$ ensures that replacing $\mathsf{V}_{\mathrm{X}\leftrightarrow\mathrm{F}}$ by $\mathsf{V}^*_{\mathrm{X}\leftrightarrow\mathrm{F}}$ only results in a negligible change, since a correct witness can be extracted from a valid proof with overwhelming probability. Second, since the choice of i_K is random and independent of the rest of the game, an adversary winning Game 0 with probability ε wins Game 1 with probability $\frac{1}{\kappa}\varepsilon$.

Game 2: Simulating the Proofs for X^*

$\mathbf{Exp}^{\text{euf-nf}}_{\mathcal{DS},\mathcal{A}}(\lambda, \Lambda)^{(2)}$

⋮
3 for $i = 1 \dots \Lambda :\ (crs_i, tr_i, ek_i) \leftarrow \mathsf{Sim}_{\mathrm{X}\leftrightarrow\mathrm{F},1}(gPar, (u_{i,j})_{j=1}^m)$
⋮
$\dots \leftarrow \mathcal{A}\big(pp, \omega : \mathsf{PK}^{(2)}, \mathsf{SK}^{(1)}, \mathsf{Del}^{(1)}, \mathsf{PSig}^{(1)}\big)$

$\mathsf{PK}^{(2)}()$

⋮
3 $P_i^* \leftarrow \mathsf{Sim}_{\mathrm{X}\leftrightarrow\mathrm{F},2}(tr_i, (X^*, F_i^*))$
⋮

Game 1 $\curvearrowright$ Game 2 Games $\mathbf{Exp}^{(1)}$ and $\mathbf{Exp}^{(2)}$ are indistinguishable by a hybrid argument on extraction zero knowledge of $\Pi_{\mathrm{X}\leftrightarrow\mathrm{F}}$. Consider hybrid game number i: After receiving crs_i, the distinguisher $\mathcal{D}$ simulates $\mathbf{Exp}^{(b)}$ using its extraction oracle for Line 3 of $\mathsf{V}^*_{\mathrm{X}\leftrightarrow\mathrm{F}}$. When eventually Line 3 of $\mathsf{PK}^{(b)}$ is reached, $\mathcal{D}$ outputs (X^*, F_i^*) and receives P_i^* from the overlying experiment.

Game 3: Simulating Signatures by X^*

$\mathbf{Exp}^{\text{euf-nf}}_{\mathcal{DS},\mathcal{A}}(\lambda, \Lambda)^{(3)}$

⋮
2 $Y^* \leftarrow \mathbb{G};\ \big(sPar := (\bar{g}, \bar{u}, (u_{i,j})_{1\le i\le\Lambda, 1\le j\le m}), tr_S\big) \leftarrow \mathsf{SimPar}(X^*, Y^*, \ell)$
⋮
$\dots \leftarrow \mathcal{A}\big(pp, \omega : \mathsf{PK}^{(2)}, \mathsf{SK}^{(1)}, \mathsf{Del}^{(3)}, \mathsf{PSig}^{(3)}\big)$

$\mathsf{Del}^{(3)}(X, warr, (X', F', P'))$

1 if $X = X^*$
2 return $\mathsf{Del}^*(pp, sk^*, warr, (X', F', P'))$, replacing (3) by
(3') for $j = 1 \dots i+1 :\ x_j \leftarrow \mathsf{Ext}(ek_j, (X_j, F_{j,j}), P_{j,j})$
$\sigma_i := \mathsf{SimSig}(tr_S, x_1 \| \dots \| x_{i+1} \| \mathbf{0}^{\Lambda-i-1})$
3 else return $\mathsf{Del}^{(0)}(X, warr, (X', F', P'))$

$\mathsf{PSig}^{(3)}(X, \mathit{warr}, M)$

```
1    if X = X*
2          return PSig*(pp, sk*, warr, M), replacing (2) by
                 (2') for i = 1 ... k :  x_i ← Ext(ek_i, (X_i, F_{i,i}), P_{i,i})
                      σ_k := SimSig(tr_S, x_1 ‖ ... ‖ x_k ‖ 0^{Λ−k−1} ‖ M)
3    else return PSig^(0)(X, warr, M)
```

Game 2 $\curvearrowright$ Game 3 First of all, note that due to the additional checks in PSig^* introduced in Game 2, extracting and signing yields the same as signing the hash values directly. From the security proof for Waters signatures follows that the parameters created by SimPar are perfectly indistinguishable from actual ones and that the simulation is perfect at least half of the time. Since in addition, failure of simulation is independent from the adversary's view, an adversary winning Game 2 with probability ε wins Game 3 with probability at least $\frac{1}{2}\varepsilon$.

Game 4: Beyond $\mathcal{L}_{\mathbf{X}\leftrightarrow\mathbf{F}}$

$\mathbf{Exp}^{\text{euf-nf}}_{\mathcal{DS},\mathcal{A}}(\lambda, \Lambda)^{(4)}$

```
⋮
1.2  x', x* ← Z_{2^m};  X* := g^{x'};  for  i = 1 ... Λ :  F*_i := F_i(x*)
⋮
```

Game 3 $\curvearrowright$ Game 4 Note that the private key x^* in the original game is used to compute (1) the F_i^*, (2) the proofs P_i^* and (3) the signatures σ_i. From Game 2 on, we compute P_i^* without x^* and from Game 3 on the signatures as well. Thus, a distinguisher between Games 3 and 4 yields a distinguisher for $\mathcal{L}_{\mathrm{X}\leftrightarrow\mathrm{F}}$, since after plugging in challenge $(X, (F_i))$, the games can be simulated without knowledge of x^*.

The CDH-Adversary

So far, we showed that any adversary having non-negligible advantage in winning Game 0 also wins Game 4 with non-negligible probability. If the experiment returns 1 in Line 8 then the fact that the signature returned by $\mathcal{A}$ passes PVer^* implies that σ_i is a signature on $F_{1,1} \cdots F_{i+1,i+1}$, thus actually a Waters signature of $x_1 \| \ldots \| x_{i+1}$, the bits of the logarithms of $X_1, \ldots, X_{i+1}$. Now the condition in Line 7 guarantees that this very message has never been submitted to SimSig. (Note that verifying all proofs $P_{j,j}$ with $\mathsf{V}^*_{\mathrm{X}\leftrightarrow\mathrm{F}}$ ensures that the bits in X_j and $F_{j,j}$ are the same.) An analogous argumentation holds if the experiment returns 1 in Line 10. Thus, to win the game $\mathcal{A}$ must in fact forge a Waters signature, which means we can use it to define a CDH-adversary by simulating Game 4 and instead of merely returning 1 extracting the bits from the proofs $P_{j,j}$ and feeding it to $\mathsf{Extract}$.

$\mathcal{A}^{\mathrm{CDH}}(X^*, Y^*)$

$i_K^* \leftarrow \{1, \ldots, n\};\ i_K := 0$

$x^* \leftarrow \mathbb{Z}_{2^m};$ for $i = 1 \ldots \Lambda:\ F_i^* := \mathcal{F}_i(x^*)$

$\big(sPar := (\bar{g}, \bar{u}, (u_{i,j})_{1\le i\le\Lambda, 1\le j\le m}), tr_S\big) \leftarrow \mathsf{SimPar}(X^*, Y^*, \ell)$

for $i = 1 \ldots \Lambda:\ (crs_i, tr_i, ek_i) \leftarrow \mathsf{Sim}_{\mathrm{X}\leftrightarrow\mathrm{F},\,1}(gPar, (u_{i,j})_{j=1}^m)$

$\omega \leftarrow \mathbb{Z}_p;\ \Omega := g^\omega;\ pp := (gPar, sPar, \vec{crs}, \Omega);\ HU := \emptyset$

$\big((X_1, F_{1,1}, P_{1,1} \ldots), M, (\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=1}^k)\big)$

$\leftarrow \mathcal{A}\big(pp, \omega : \mathsf{PK}^{(2)}, \mathsf{SK}^{(1)}, \mathsf{Del}^{(3)}, \mathsf{PSig}^{(3)}\big)$

if $\mathsf{PVer}^*((X_1, F_{1,1}, P_{1,1} \ldots), M, (\sigma_1, (X_i, F_{i,i}, P_{i,i}, cert_{i,i}, \sigma_i)_{i=1}^k)\big) = 0$, return 0

if $\exists\, 1 \le i < k$: $X_i = X^*$ and no query $\mathsf{Del}\big(X^*, ((X_j \ldots)_{j=1}^{i-1}, X^* \ldots), X_{i+1}\big)$

for $j = 1 \ldots i+1:\ x_j \leftarrow \mathsf{Ext}(ek_j, (X_j, F_{j,j}), P_{j,j})$

return $\mathsf{Extract}(tr_S, x_1 \,\|\, \ldots \,\|\, x_{i+1} \,\|\, \mathbf{0}^{\Lambda-i-1}, \sigma_i)$

if $X_k = X^*$ and no query $\mathsf{PSig}\big(X^*, ((X_j \ldots)_{j=1}^{k-1}, X^* \ldots), M\big)$

for $i = 1 \ldots k:\ x_i \leftarrow \mathsf{Ext}(ek_i, (X_i, F_{i,i}), P_{i,i})$

return $\mathsf{Extract}(tr_S, x_1 \,\|\, \ldots \,\|\, x_k \,\|\, \mathbf{0}^{\Lambda-k-1} \,\|\, M, \sigma_k)$

return 0

Game 4 $\curvearrowright$ CDH Game 4 is won if and only if the adversary manages to produce a Waters forgery. Now Waters' security proof guarantees that if all oracle queries were correctly simulated and the adversary returns a valid forgery then Extract computes a CDH solution from it with probability $\frac{1}{2m\ell}$. It follows that if $\mathcal{A}$ wins Game 4 with probability ε, then $\mathcal{A}^{\mathrm{CDH}}$ produces a CDH-instance with probability at least $\frac{1}{2m\ell}\varepsilon$.

All in all, we have thus shown that given an adversary $\mathcal{A}$ breaking non-frameability of $\mathcal{DS}$, we can construct an algorithm solving the CDH-problem with non-negligible probability.