Automorphic Signatures in Bilinear Groups

Georg Fuchsbauer

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

Abstract. We introduce the notion of *automorphic signatures*, which satisfy the following properties: the verification keys lie in the message space, messages and signatures consist of group elements only, and verification is done by evaluating a set of pairing-product equations. These signatures make a perfect counterpart to the powerful proof system by Groth and Sahai (Eurocrypt 2008). We provide practical instantiations of automorphic signatures under appropriate assumptions and an extension yielding blind signatures. We then use them in combination with Groth-Sahai proofs to efficiently instantiate several cryptographic primitives, such as group signatures, non-interactive anonymous credentials and anonymous proxy signatures.

1 Introduction

One of the main goals of modern cryptography is anonymity. A classical primitive ensuring user anonymity is group signatures [Cv91]: they allow members that were enrolled by a group manager to sign on behalf of a group while not revealing their identity. Another example is anonymous credentials [Cha85], by which a user can prove that she holds a certain credential, and at the same time remain anonymous. Blind signatures [Cha83] were introduced for electronic cash to prevent linking of a coin to its spender; and there are numerous other primitives ensuring user privacy.

Security of such primitives is addressed by defining a security model, which is typically first proved to be satisfiable in theory under general assumptions. Let us consider the example of *dynamic group signatures* by Bellare et al. [BSZ05]. To show feasibility of their model, they give the following generic construction: Assume the existence of a signature scheme, an encryption scheme and general zero-knowledge proofs. The group manager publishes a signature verification key and uses the corresponding signing key to issue certificates on the group members' personal verification keys. The latter produce a group signature on a message by first signing it with their personal signing key, and then by encrypting the user's certificate, her verification key, and the signature on the message. The group signature consists of these ciphertexts completed by a zero-knowledge (ZK) proof that the certificate and the signature in the plaintext are valid. The fact that a signature is a ciphertext and that the ZK proof leaks no information guarantees user anonymity.

For a long time the most efficient way to instantiate such primitives was to rely on the random-oracle heuristic [BR93]. Due to a series of criticisms starting with [CGH98], more and more practical schemes are being proposed in the standard model. In particular, groups with a bilinear map (pairing) turned out to be an attractive tool to achieve efficiency. Many of the practical instantiations use ad hoc constructions, since the generic ones—in particular zero-knowledge proofs—are by far too inefficient.

The Groth-Sahai proof system. Recently, Groth and Sahai [GS08] proposed *efficient* zero-knowledge proofs for a large class of statements over bilinear groups, which already found use in many implementations [Gro07,GL07,BCKL08,BCC^{+}09]. They start by constructing witness indistinguishable (WI) proofs of satisfiability of various types of equations. Given a witness for satisfiability, one makes *commitments* to its values and then constructs proofs asserting that the committed values satisfy the equations. As already observed by [Gro06], the most interesting and widely used type is the following: pairing-product equations (PPE) whose variables are elements of the bilinear group (cf. Sect. 2.2). A PPE equation consists of products of pairings applied to the variables and constants from the group. Since the employed commitments to the witnesses are extractable, the resulting proofs actually constitute *proofs of knowledge* as well.

To efficiently realize the BSZ-model of group signatures, Groth [Gro07] instantiates encryption and the proofs of plaintext validity with the Groth-Sahai WI proof system.[1] The extractability of the used commitments serves two purposes: first, it lets the opener extract the user's verification key and thereby trace the signer (the commitments are thus used as encryptions that can be decrypted with the extraction key); second, it enables the unforgeability of group signatures to be directly reducible to the unforgeability of the underlying signatures.

For the Groth-Sahai methodology to be applicable, Groth gives certification and signing schemes such that certificates, signature verification keys and signatures (i.e., the components that need to be hidden) are group elements whose validity is verified by evaluating PPEs.

Signatures and the Groth-Sahai Proof System. The first practical schemes which used Groth-Sahai-like proofs were the group signatures by Boyen and Waters [BW06,BW07], who independently developed their purpose-built proofs. They require weakly secure[2] signatures whose components and messages can be encrypted (committed to) and proved to be valid. To sign messages from the bilinear group, they modify the weak Boneh-Boyen signatures [BB04], which consist of a group element, and whose messages are scalars: instead of giving the scalar directly, they give it as an exponentiation of two different group generators. The security of their construction holds under the *hidden strong Diffie-Hellman assumption* (HSDH) a variant of the *strong Diffie-Hellman assumption* (SDH) [BB04].

Belenkiy et al. [BCKL08] apply the Boneh-Boyen [BB04] transformation "from weak to strong security" to the Boyen-Waters signature scheme. This way they obtain fully secure signatures, at the price of introducing a stronger assumption called "*triple Diffie-Hellman*" (TDH). Their signatures consist of group elements, yet the messages are scalars. To construct anonymous credentials, they make commitments to a message and a signature on it and prove that their content is valid using Groth-Sahai proofs. Since from these commitments only group elements can be extracted efficiently (and one would have to compute discrete logarithms to recover the scalar message), they are obliged to define f-extractability, meaning that only a function of the committed value can be extracted. This entails stronger security notions ("F-unforgeability") for the signature scheme in order to prove security of their construction.

In the abovementioned group signatures from [Gro07], this drawback is avoided by designing the key-certification scheme so that all committed values are group elements. The key certification is thus different from the signature scheme whose keys are certified. Moreover, the certificate-verification key is an element of the *target* group. As opposed to standard group signatures, in hierarchical group signatures [TW05] or anonymous proxy signatures [FP08], verification keys are not only certified once, but must also serve to certify other keys—it is therefore required that the message space contain the verification keys. If we want to apply the Groth-Sahai methodology to "anonymize" such schemes and prove unforgeability by reducing it to the security of the underlying signatures, everything has to be in the group.

We identify the all-purpose building block to efficiently instantiate more intricate primitives as the following: a practical signature scheme secure against adaptive chosen-message attacks that can sign its own verification keys; and which at the same time respects the pairing-product paradigm, that is, keys, messages and signatures consist of group elements and the signature verification relations are PPEs. We call such a scheme an *automorphic signature*, since it is able to sign its *own* keys and verification

[1] To achieve anonymity against adversaries with opening oracles as required by the BSZ-model, some extra encryption has to be added; see the discussion in Sect. 6.1.

[2] We call a signature scheme *weakly secure* if an adversary getting signatures on *random* messages is unable to produce with a new signed message.

preserves the *structure* of keys and messages, which makes it perfectly suitable to be combined with Groth-Sahai proofs.

We believe that working with group elements enables a modular approach of combining signatures with Groth-Sahai proofs, and automorphic signatures are the building block tailored to do so. As demonstrated in Sect. 6, they yield straightforward efficient implementations of generic constructions, by simply plugging in concrete schemes for generic ones.

Our Contribution

After formally defining automorphic signatures, we give two practical instantiations over bilinear groups (Sect. 4). We note that a scheme in [Gro06] based on the *decision linear assumption* [BBS04] satisfies all the properties, but should rather be regarded as a proof of concept due to its inefficiency (a signature consists of hundreds of thousands of group elements). We give practical-level efficiency under reasonable assumptions.

Recently, Fuchsbauer, Pointcheval and Vergnaud [FPV09] introduced a variant of HSDH, called *double hidden SDH* (DHSDH). We observe that—unlike HSDH—their assumption immediately yields weakly secure signatures on messages consisting of group elements and verifiable by PPEs.

We make their scheme secure against chosen-message attacks by adding some randomness analogously to the transformation of [BB04]. The security of the resulting scheme requires an additional assumption, which can be considered quite mild though (cf. Assumption 3): it is a non-parametrized non-interactive falsifiable assumption in the flavor of CDH, whose validity in the generic group model [Sho97] is immediate. Our second instantiation improves the efficiency and is secure under yet another variant of HSDH, called DAHSDH ("double *asymmetrically* hidden", cf. Assumption 2), which we also prove secure in the generic group model. Its signatures consist of 5 group elements only.

In Sect. 5, we provide extensions of our scheme: we show how to sign vectors of messages and in particular we construct *blind* automorphic signatures. In Sect. 6, we give illustrative applications of automorphic signatures: we build CCA-secure group signatures and revisit the construction of non-interactive credentials of [BCKL08]; in particular, we achieve actual message extractability and give a more efficient credential issuing protocol. We then present our main application: the first efficient instantiation of *anonymous proxy signatures* (APS) without random oracles. While the other primitives are realizable without automorphic signatures, no other efficient construction of APS is known.

Proxy signatures, which allow the delegation of signing rights, were introduced by [MUO96] and later formalized by [BPW03,SMP08]. Anonymous proxy signatures [FP08] unify group signatures and (multi-level) proxy signatures by requiring that the proxy signer and intermediate delegators remain anonymous. They enable users ("original signers") to delegate others to sign on their behalf; the latter can either sign or re-delegate to other users. Anonymity ensures that from a proxy signature one cannot tell who signed and who re-delegated; however one can be sure that the proxy signer was delegated by the original signer. As for group signatures, an algorithm to revoke anonymity to deter from misuse is also provided. Due to consecutiveness of delegation, this primitive also models hierarchical group signatures satisfying a security model generalizing the one of [BSZ05].

In our instantiation, we use automorphic signatures to construct certification chains of which one can prove knowledge using Groth-Sahai proofs. Witness indistinguishability of these proofs ensures user anonymity. We note that the delegation protocol of our instantiation is non-interactive.

In Sect. 6.4, we define the following additional functionalities for APS in order to incorporate *delegatable anonymous credentials* [BCC$^+$09] (see below):

- When re-delegating signing rights, the previous delegators remain anonymous even to the delegatee.
- We give a protocol for *blind delegation*: a user can be delegated to without revealing her identity.
- Finally we show how to sign messages on behalf of several users simultaneously.

We then give an efficient instantiation of the extended model, which illustrates the benefits of automorphic signatures: combined with the Groth-Sahai proof system, they enable straightforward practical realization of generic concepts. Note that it is precisely the lack of an efficient automorphic signature that accounts for the necessity of a strong assumption for the first instantiation of APS in [FP09], as well as for its impracticality.

APS vs. DAC. Delegatable anonymous credentials (DAC) were recently defined by Belenkiy et al. [BCC+09]. Although they are not comparable to our extended model of APS, there are substantial similarities between them: both provide mechanisms enabling users to prove possession of certain rights while remaining anonymous; and both consider re-delegation of the received rights.

The core protocol of DAC allows a user to obtain a proof of knowledge of a signature on her *secret* key, without revealing the identity of neither the signer nor the user. This imposes interactivity of the delegation process, while (non-blind) delegations for APS are non-interactive, even when previous delegators remain anonymous. (We show how to achieve delegatee anonymity at the expense of non-interactivity). Further differences are that DAC merely deal with authentication rather than signatures, and they do not provide tracing mechanisms.

2 Preliminaries

2.1 Primitives

We recall some standard concepts from the literature.

Commitments. A non-interactive commitment scheme is composed of an algorithm $\mathsf{Setup}_{\mathrm{Com}}$, outputting a *commitment key* ck, and an algorithm Com with arguments ck, a message M and randomness ρ. We require that (1) the scheme is *perfectly binding*, i.e., for a commitment $\mathbf{c}$ there exists only one M s.t.: $\mathbf{c} = \mathsf{Com}(ck, M, \rho)$ for some ρ; (2) the scheme is *computationally hiding*, in particular, there exists $\mathsf{SmSetup}_{\mathrm{Com}}$ outputting keys that are computationally indistinguishable from those output by $\mathsf{Setup}_{\mathrm{Com}}$, and which generate perfectly hiding commitments.

A commitment scheme is *extractable* if there exists an algorithm $\mathsf{ExSetup}_{\mathrm{Com}}$ outputting (ck, ek), where ck is distributed as the output of $\mathsf{Setup}_{\mathrm{Com}}$, and an algorithm $\mathsf{Extr}_{\mathrm{Com}}$ that on input the *extraction key* ek and a commitment extracts the committed value from it.

Digital Signatures. A digital signature scheme consists of the following algorithms: $\mathsf{Setup}_{\mathrm{Sig}}$ outputs public parameters pp. $\mathsf{KeyGen}_{\mathrm{Sig}}$ outputs a pair (vk, sk) of verification and signing key. $\mathsf{Sign}(sk, M)$ outputs a signature σ, which is verified by $\mathsf{Verify}_{\mathrm{Sig}}(vk, M, \sigma)$. Signatures are *existentially unforgeable under chosen-message attack* (EUF-CMA) [GMR88] if no adversary, given vk and a signing oracle for messages of its choice, can output a pair (M, σ) s.t. M was never queried and $\mathsf{Verify}(vk, M, \sigma) = 1$.

Blind Signatures. Blind signatures [Cha83,PS00] extend digital signatures by an interactive protocol $\mathsf{Issue} \leftrightarrow \mathsf{Obtain}$ between the signer and a user allowing the latter to obtain a signature on a message hidden from the signer. Okamoto [Oka06] defines the following security requisites: *Blindness:* An adversary impersonating the signer interacting with Obtain twice for messages of its choice cannot relate the resulting signatures to their issuings. *Unforgeability:* No adversary interacting $q - 1$ times with Issue can output q different messages and valid signatures on them.

Bilinear Groups. A (symmetric) *bilinear group* is a tuple $\mathcal{BG} = (p, \mathbb{G}, \mathbb{G}_T, e, G)$ where $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups[3] of prime order p, G is a generator of $\mathbb{G}$, and $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a non-degenerate bilinear map, i.e., $\forall X, Y \in \mathbb{G}\ \forall a, b \in \mathbb{Z} : e(aX, bY) = e(X, Y)^{ab}$, and $e(G, G)$ generates $\mathbb{G}_T$.

The *Decision Linear (DLIN) Assumption*, introduced by Boneh et al. [BBS04], in a bilinear group $(p, \mathbb{G}, \mathbb{G}_T, e, G)$ states that given $(\alpha G, \beta G, r\alpha G, s\beta G, tG)$ for random $\alpha, \beta, r, s \in \mathbb{Z}_p$, it is hard to decide whether $t = r + s$ or t is random.

Throughout the paper, we will assume two fixed generators G, H of $\mathbb{G}$. We call a pair $(A, B) \in \mathbb{G}^2$ a *Diffie-Hellman pair* (w.r.t. (G, H)), if there exists $a \in \mathbb{Z}_p$ such that $A = aG$ and $B = aH$. Using the

[3] Following [GS08], we denote $\mathbb{G}$ additively and $\mathbb{G}_T$ multiplicatively. We denote the neutral element of $\mathbb{G}$ by 0.

bilinear map e, such pairs are efficiently decidable by checking $e(A,H)=e(G,B)$. We let $\mathcal{DH}$ denote the set of DH pairs and implicitly assume them to be w.r.t. G and H.

2.2 Groth-Sahai Proofs for Pairing-Product Equations

We start with presenting perfectly binding extractable commitments, which are computationally hiding under the decision linear assumption, and then give an overview of Groth-Sahai proofs introduced in [GS08].

Linear Commitments. Let $\mathcal{BG}=(p,\mathbb{G},\mathbb{G}_T,e,G)$ be a bilinear group. $\mathsf{Setup}(\mathcal{BG})$ is defined as follows: choose $\alpha,\beta,r_1,r_2\leftarrow\mathbb{Z}_p$ and output $ck=(\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3)$ with

$$\mathbf{u}_1:=\big(\alpha G,0,G\big)\qquad\mathbf{u}_2:=\big(0,\beta G,G\big)\qquad\mathbf{u}_3:=\big(r_1\alpha G,r_2\beta G,(r_1+r_2)G\big)$$

$\mathsf{ExSetup}$ additionally outputs the extraction key $ek:=(\alpha,\beta)$. A commitment to a group element $X\in\mathbb{G}$ using randomness $\rho=(\rho_1,\rho_2,\rho_3)\leftarrow\mathbb{Z}_p^3$ is defined as $\mathsf{Com}\big(ck,X,\rho\big):=(0,0,X)+\sum_{i=1}^3\rho_i\mathbf{u}_i$.

Extraction is done by computing $\mathsf{Extr}((\alpha,\beta),(c_1,c_2,c_3)):=c_3-\frac{1}{\alpha}c_1-\frac{1}{\beta}c_2=X$. $\mathsf{SmSetup}$ replaces $u_{3,3}$ in ck with $(r_1+r_2-1)G$ (which is indistinguishable by DLIN) resulting in perfectly hiding commitments.

Groth-Sahai WI Proofs. We will use Groth-Sahai witness-indistinguishable (WI) proofs of *satisfiability of pairing-product equations.* A *pairing-product equation* (PPE) over variables $\mathcal{Y}_1,\ldots,\mathcal{Y}_n\in\mathbb{G}$ is an equation of the form

$$\prod_{i=1}^n e(A_i,\mathcal{Y}_i)\prod_{i=1}^n\prod_{j=1}^n e(\mathcal{Y}_i,\mathcal{Y}_j)^{\gamma_{i,j}}\;=\;t_T\;, \tag{E}$$

determined by $A_i\in\mathbb{G}$, $\gamma_{i,j}\in\mathbb{Z}_p$ and $t_T\in\mathbb{G}_T$, for $1\le i,j\le n$.

We present the DLIN-instantiation of the proof system, which uses the linear commitment scheme ($\mathsf{Setup},\mathsf{Com},\mathsf{ExSetup},\mathsf{Extr},\mathsf{SmSetup}$). The proof system for a bilinear group $\mathcal{BG}=(p,\mathbb{G},\mathbb{G}_T,e,G)$ is set up by running $\mathsf{Setup}(\mathcal{BG})$ which produces a perfectly-binding linear commitment key $ck\in\mathbb{G}^5$. Given an assignment $\mathcal{Y}_i\leftarrow X_i$, for $X_i\in\mathbb{G}$, satisfying E, one first *commits* to the values X_i by choosing randomness $\rho_i\leftarrow\mathbb{Z}_p^3$ and setting $\mathbf{c}_{X_i}:=\mathsf{Com}(ck,X_i,\rho_i)$ for all i. Running $\mathsf{Prove}_{\mathrm{GS}}(ck,E,(X_i,\rho_i)_{i=1}^n)$ generates a proof[4] $\phi\in\mathbb{G}^{3\times3}$ which asserts that the values committed in $\mathbf{c}_{X_i}$ satisfy E. A proof ϕ for equation E and commitments $(\mathbf{c}_{X_i})_{i=1}^n$ under ck is verified by $\mathsf{Verify}_{\mathrm{GS}}(ck,E,(\mathbf{c}_{X_i})_{i=1}^n,\phi)$. An honestly computed proof on commitments to values satisfying E is always accepted by $\mathsf{Verify}_{\mathrm{GS}}$.

Security. *Soundness.* Given the extraction key ek output by $\mathsf{ExSetup}$ and commitments $(\mathbf{c}_{X_i})_{i=1}^n$ s.t. $\mathsf{Verify}_{\mathrm{GS}}(ck,E,(\mathbf{c}_{X_i})_{i=1}^n,\phi)=1$ for some ϕ, algorithm Extr applied to $\mathbf{c}_{X_i}$ for all i yields a vector $(X_i)_{i=1}^n$ that satisfies E.

Witness Indistinguishability (WI). If the commitment key is replaced by ck^* output by $\mathsf{SmSetup}$ (which is indistinguishable under DLIN), then a commitment $\mathbf{c}:=\mathsf{Com}(ck^*,X,\rho)$ is perfectly hiding, i.e., given $\mathbf{c}$ then for any X there exists randomness ρ s.t. $\mathbf{c}=\mathsf{Com}(ck^*,X,\rho)$. Moreover, given values $((X_1,\rho_1),\ldots,(X_n,\rho_n))$ and $((X_1',\rho_1'),\ldots,(X_n',\rho_n'))$ such that for all i it holds that $\mathsf{Com}(ck,X_i,\rho_i)=\mathsf{Com}(ck,X_i',\rho_i')$, and $(X_1,\ldots,X_n)$ and $(X_1',\ldots,X_n')$ both satisfy E, then $\mathsf{Prove}_{\mathrm{GS}}(ck^*,E,(X_i,\rho_i)_{i=1}^n)$ and $\mathsf{Prove}_{\mathrm{GS}}(ck^*,E,(X_i',\rho_i')_{i=1}^n)$ generate the same distribution of proofs.

Examples. (1) *Proof of Two Commitments Containing the Same Value.* Let $E_{\mathrm{equal}}(X_1,X_2)$ denote the equation $e(X_1,G)\,e(X_2,G^{-1})=1$. Given two commitments $\mathbf{c}_M=\mathsf{Com}(ck,M,\rho)$ and $\mathbf{c}_N=\mathsf{Com}(ck,N,\sigma)$, $\mathsf{Prove}(ck,E_{\mathrm{equal}},(M,\rho),(N,\sigma))$ proves that $\mathbf{c}_M$ and $\mathbf{c}_N$ commit to the same value.

[4] For general PPEs, the proof is in $\mathbb{G}^{3\times3}$. If E is a *linear equation* (i.e., $\gamma_{i,j}=0$ for all i,j), then the proof reduces to 3 group elements. Note that in this context the word *proof* can either denominate "proof of satisfiability" (or language-membership)—which thus includes the commitments—or mean a proof *that the content of some given commitments satisfies a given equation.* We adopt the latter diction.

(2) *Proof of Commitments to a $\mathcal{DH}$-Pair.* Define $E_{\mathcal{DH}}(X, Y)$ as $e(X, H)\, e(Y, G^{-1}) = 1$. A proof for Equation $E_{\mathcal{DH}}$ yields a 3-element proof showing that the values in two commitments are in $\mathcal{DH}$.

Zero-Knowledge Proofs. Groth and Sahai also define algorithms $\mathsf{ZKProve}_{\mathrm{GS}}$ and $\mathsf{ZKVerify}_{\mathrm{GS}}$ (and further ones to simulate) to construct non-interactive zero-knowledge (NIZK) proofs for equations E whose right-hand side t_T is of the form $e(T_1, T_2)$ for public $T_1, T_2 \in \mathbb{G}$ (which is the case for all our equations).

Randomizing Groth-Sahai Proofs. As observed by [FP09] and [BCC$^+$09] and formalized by the latter, Groth-Sahai WI proofs can be *randomized*: There exists an algorithm $\mathsf{RdCom}_{\mathrm{GS}}$ that on input ck, a commitment $\mathbf{c}$ and fresh randomness ρ outputs a *randomization* of $\mathbf{c}$ under ρ. A proof ϕ for an equation E and commitments $(\mathbf{c}_1, \ldots, \mathbf{c}_n)$ can be *adapted* to the randomizations $\mathbf{c}'_i = \mathsf{RdCom}_{\mathrm{GS}}(ck, \mathbf{c}_i, \rho_i)$ by running $\mathsf{RdProof}_{\mathrm{GS}}(ck, E, (\mathbf{c}_i, \rho_i)_{i=1}^n, \phi)$ yielding ϕ' such that $\mathsf{Verify}_{\mathrm{GS}}(ck, E, (\mathbf{c}'_i)_{i=1}^n, \phi') = 1$. (Basically, if $\mathbf{c}_i = \mathsf{Com}_{\mathrm{GS}}(ck, X_i, \sigma_i)$ then $\mathbf{c}'_i = \mathsf{Com}_{\mathrm{GS}}(ck, X_i, \sigma_i + \rho_i)$ and ϕ' is distributed as proofs output by $\mathsf{Prove}_{\mathrm{GS}}(ck, E, (X_i, \sigma_i + \rho_i)_{i=1}^n)$; see [FPV09, Sect. 5] for a concise overview.)

3 Assumptions

We first restate the assumption from [FPV09] and then introduce two new ones.

Assumption 1 (q-DHSDH). *Given $(G, H, K, X = xG) \in \mathbb{G}^4$ and $q - 1$ tuples*

$$\big(A_i = \tfrac{1}{x+c_i}(K + v_i G),\ C_i = c_i G,\ D_i = c_i H,\ V_i = v_i G,\ W_i = v_i H\big)_{i=1}^{q-1}\ , \quad \textit{for } c_i, v_i \leftarrow \mathbb{Z}_p,$$

it is hard to output a new tuple $(A^*, C^*, D^*, V^*, W^*)$ that satisfies

$$e(A^*, X + C^*) = e(K + V^*, G) \qquad e(C^*, H) = e(G, D^*) \qquad e(V^*, H) = e(G, W^*) \qquad (1)$$

Argument. As pointed out by its inventors, under the *Knowledge-of-Exponent Assumption* (KEA) [Dam92,BP04], hardness of q-DHSDH follows from hardness of the following problem:

q-SDH-III: Given $\big(G, K, X = xG, (A_i = \frac{1}{x+c_i}(K + v_i G), c_i, v_i)_{i=1}^{q-1}\big)$, produce a new tuple (A^*, c^*, v^*) satisfying $e(A^*, X + c^* G) = e(K + v^* G, G)$.

(KEA asserts that given (G, H), from an adversary returning $(c^* G, c^* H)$ and $(v^* G, v^* H)$ one can extract c^* and v^*.) They then show that hardness of q-SDH-III is implied by hardness of q-SDH [BB04], a well-established assumption by now. □

We introduce a variant of DHSDH that enables a more efficient instantiation of automorphic signatures. An instance now contains an additional generator F, and the elements $C_i = c_i G$ are doubled by $D_i = c_i F$. This makes it possible to include an additional element $Y = xH$ in the instance (if it is given together with $c_i H$, we arrive at an easy problem; cf. Appendix A.1).

Assumption 2 (q-DAHSDH). *Given $(G, F, H, K, X = xG, Y = xH) \in \mathbb{G}^6$ and $q - 1$ tuples*

$$\big(A_i = \tfrac{1}{x+c_i}(K + v_i G),\ C_i = c_i G,\ D_i = c_i F,\ V_i = v_i G,\ W_i = v_i H\big)_{i=1}^{q-1}\ , \quad \textit{for } c_i, v_i \leftarrow \mathbb{Z}_p,$$

it is hard to output a new tuple $(A^*, C^*, D^*, V^*, W^*)$ that satisfies

$$e(A^*, X + C^*) = e(K + V^*, G) \qquad e(C^*, F) = e(G, D^*) \qquad e(V^*, H) = e(G, W^*) \qquad (2)$$

Argument. Due to the fact that we give $Y = xH$, the KEA-reduction to SDH does not apply here (given G, H, X we would have to solve CDH to compute Y for the reduction). Instead, we directly prove that the assumption holds in the generic group model [Sho97] in Appendix A.2. □

The next assumption, called HDL ("hidden discrete log"), states that given 3 generators G, H, T, it is hard to output a non-trivial tuple (mG, mH, rG, rH) such that $rT = mG$.

Assumption 3 (HDL). *Given (G, H, T), a triple of random generators of $\mathbb{G}$, it is hard to output a tuple $(M, N, R, S) \neq (0, 0, 0, 0) \in \mathbb{G}^4$ such that*

$$e(R, T) = e(M, G) \qquad e(M, H) = e(G, N) \qquad e(R, H) = e(G, S) \tag{3}$$

Argument. With overwhelming probability, we have $G, H, T \in \mathbb{G} \setminus \{0\}$. We show that under KEA, Assumption 3 is equivalent to the discrete-logarithm (DL) assumption, thus a fortiori it holds in the generic group model. Let (G, T) be a DL instance, i.e., we have to compute $t := \log_G T$. Let H be the group element for KEA. Give the adversary (G, H, T). From a successful output, by KEA, we can extract $m := \log_G M = \log_H N$ and $r := \log_G R = \log_H S$. From (3), we have $rt = m$, and $r \neq 0$ (otherwise all output elements must be 0), and therefore $t = \frac{m}{r}$. □

4 Automorphic Signatures

Definition 1. *An* automorphic signature *over a bilinear group $(p, \mathbb{G}, \mathbb{G}_T, e, G)$ is an EUF-CMA secure signature whose verification keys are contained in the message space. Moreover, the messages and signatures consist of elements of $\mathbb{G}$ and verification amounts to evaluating pairing-product equations over the verification key, the message and the signature.*

DHSDH immediately yields a weakly secure signature scheme if we consider X as the public key, (V, W) as a message in $\mathcal{DH}$ and (A, C, D) as the signature.[5] We show how to transform this into a CMA-secure signature scheme by assuming HDL. We introduce some additional randomness that lets us map a query for a message chosen by the adversary to a given tuple $(A_i, C_i, D_i, V_i, W_i)$ from a DHSDH instance. HDL then basically asserts that the adversary cannot produce a signed new message $\big((A^*, C^*, D^*, R^*, S^*), (M^*, N^*)\big)$ that maps back to a tuple from the instance (see the proof of Theorem 2).

Scheme 1 ($\mathbf{Sig}_{\mathrm{FPV}}$). Given a bilinear group, $\mathsf{Setup}_{\mathrm{FPV}}$ chooses parameters $(G, H, K, T) \leftarrow \mathbb{G}^4$, which define the message space as $\mathcal{DH} := \{(aG, aH) \,|\, a \in \mathbb{Z}_p\}$, $\mathsf{KeyGen}_{\mathrm{FPV}}$ chooses a secret key $x \leftarrow \mathbb{Z}_p$ and sets $vk = X := xG$. A message $(M, N) \in \mathcal{DH}$ is signed by $\mathsf{Sign}_{\mathrm{FPV}}(x, (M, N))$ outputting $\big(A := \frac{1}{x+c}(K + rT + M), C := cG, D := cH, R := rG, S := rH\big)$ for random $c, r \in \mathbb{Z}_p$. $\mathsf{Verify}_{\mathrm{FPV}}$ accepts a signature on a message $(M, N) \in \mathcal{DH}$ if it satisfies

$$e(A, X + C) = e(K + M, G)\, e(T, R) \qquad e(C, H) = e(G, D) \qquad e(R, H) = e(G, S)\ .$$

Theorem 1. *Under q-DHSDH and HDL, $\mathbf{Sig}_{\mathrm{FPV}}$ is existentially unforgeable against adversaries making at most $q - 1$ adaptive chosen-message queries.*

(We omit the proof as Theorem 2 is proved almost identically.) The fact that the verification keys are not in the message space is the only reason the scheme is not automorphic. However, the following hybrid scheme $\mathbf{Sig}_{2\mathrm{FPV}}$ with parameters (G, H, L, K, T) satisfies that condition:

We define a two-level scheme, using two instances of $\mathbf{Sig}_{\mathrm{FPV}}$. The first signs pairs (mG, mH), whereas the messages for the second are of the form (mG, mL) (i.e., it uses a different parameter L instead of H). The public keys of $\mathbf{Sig}_{2\mathrm{FPV}}$ are formed as (xG, xL)—and are thus messages of the second scheme—and signing a message (mG, mL) is defined as follows: choose $v \leftarrow \mathbb{Z}_p$ and first make a $\mathbf{Sig}_{\mathrm{FPV}}$-signature on the *one-time key* $(V := vG, W := vH)$. Now use v to produce a signature on (mG, mL)

[5] Note that this is not the case for the q-HSDH assumption stating that given G, H, xG and $q-1$ triples $(\frac{1}{x+c_i}G, c_iG, c_iH)$ for random $c_i \in \mathbb{Z}_p$, it is hard to produce a new triple $(\frac{1}{x+c^*}G, c^*G, c^*H)$ with $c^* \neq c_i$ for all i. We cannot regard (cG, cH) as the message, since the signer must know c in order to produce $\frac{1}{x+c}G$.

w.r.t. parameters (G, L, K, T). A $\mathbf{Sig}_{2\mathrm{FPV}}$-signature is the concatenation of the first parameter-H signature, the pair (V, W) and the parameter-L signature on the message under key (V, W).

Security of the construction follows from a simple hybrid argument: forgeries using a new one-time key are reduced to forgeries for the 1st-level scheme, whereas forgeries recycling a key from a signing query are reduced security of the 2nd-level scheme. A signature consists of 12 group elements satisfying 7 PPEs.

If we assume DAHSDH instead of DHSDH, we get the following more efficient construction, whose signatures consist of 5 group elements only.

Scheme 2 ($\mathbf{Sig}_\mathrm{A}$). $\mathsf{Setup}_\mathrm{A}$. Given $(p, \mathbb{G}, \mathbb{G}_T, e, G)$, choose additional generators $F, H, K, T \in \mathbb{G}$.
$\mathsf{KeyGen}_\mathrm{A}$. Choose $sk = x \leftarrow \mathbb{Z}_p$ and set $vk = (xG, xH)$.
Sign_A. A signature on a DH-pair (M, N), valid under public key (xG, xH), is defined as

$$\big(A := \tfrac{1}{x+c}(K + rT + M), C := cG, D := cF, R := rG, S := rH\big) \ , \quad \text{for random } c, r \leftarrow \mathbb{Z}_p$$

$\mathsf{Verify}_\mathrm{A}$. (A, C, D, R, S) is valid on a valid message (M, N) under a valid public key $vk = (X, Y)$ iff

$$e(A, X + C) = e(K + M, G)\, e(T, R) \qquad e(C, F) = e(G, D) \qquad e(R, H) = e(G, S) \qquad (4)$$

(Key (X, Y) and message (M, N) are valid if $e(X, H) = e(G, Y)$ and $e(M, H) = e(G, N)$, resp.)

Theorem 2. *Assuming q-DAHSDH and HDL,* $\mathbf{Sig}_A$ *is existentially unforgeable against adversaries making at most* $q - 1$ *adaptive chosen-message queries.*

Proof. Consider an adversary that after receiving parameters (G, F, H, K, T) and public key (X, Y) is allowed to ask for $q - 1$ signatures $(A_i, C_i, D_i, R_i, S_i)$ on messages $(M_i, N_i) \in \mathcal{DH}$ of its choice and outputs $(M, N) \in \mathcal{DH}$ that it did not query and a valid signature (A, C, D, R, S) on it.

We distinguish two kinds of forgers: An adversary is called of Type I if its output satisfies

$$\forall\, 1 \le i \le q - 1 : e(R - R_i, T) \neq e(M_i - M, G) \qquad (5)$$

Otherwise it is called of Type II. We will use the first type to break q-DAHSDH and the second type to break HDL.

Type I Let $\big(G, F, H, K, X, Y, (A_i, C_i, D_i, V_i, W_i)_{i=1}^{q-1}\big)$ be a $q-$DAHSDH challenge. It satisfies thus

$$e(A_i, X + C_i) = e(K + V_i, G) \qquad e(C_i, F) = e(G, D_i) \qquad e(V_i, H) = e(G, W_i) \qquad (6)$$

Let $\mathcal{A}$ be a forger of Type I. Choose $t \leftarrow \mathbb{Z}_p$ and give parameters $(G, F, H, K, T := tG)$ and the public key (X, Y) to $\mathcal{A}$. The i-th query for $(M_i, N_i) \in \mathcal{DH}$ is answered as $\big(A_i, C_i, D_i, R_i := \frac{1}{t}(V_i - M_i), S_i = \frac{1}{t}(W_i - N_i)\big)$. It satisfies (4) and is correctly distributed since v_i is random in the DAHSDH instance. If the adverseray produces a valid signature/message pair $((A, C, D, R, S), (M, N))$, then

$$(A, C, D, V := tR + M, W := tS + N)$$

is a solution for the DAHSDH instance, since it satisfies (1) and it is a *new* tuple: if for some i, we had $V = V_i$, then $tR + M = tR_i + M_i$, and thus $e(R, T)\, e(M, G) = e(R_i, T)\, e(M_i, G)$, which contradicts (5).

Type II Let (G, H, T) be a HDL instance; let $\mathcal{A}$ be a forger of Type II. Pick $F, K \leftarrow \mathbb{G}$ and $x \leftarrow \mathbb{Z}_p$, set $X := xG$, $Y := xH$ and give the adversary parameters (G, F, H, K, T) and public key (X, Y). Answer a signing query on $(M_i, N_i) \in \mathcal{DH}$ by returning a signature $(A_i, C_i, D_i, R_i, S_i)$ produced using x. Suppose $\mathcal{A}$ returns $((A, C, D, R, S), (M, N))$ satisfying (4) s.t. for some i: $e(R - R_i, T) = e(M_i - M, G)$. Then $(M^* := M_i - M, N^* := N_i - N, R^* := R - R_i, S^* := S - S_i)$ is a HDL solution: (M^*, N^*) and (R^*, S^*) are DH pairs, and $M^* \neq 0$, since (M, N) must be a new message. □

5 Extensions

5.1 Signatures on Bit-Strings

$\mathbf{Sig}_{\mathrm{A}}$ also serves to sign bit strings (fitting thus the standard definition of signatures). Let $\mathsf{Hash}\colon \{0,1\}^* \to \mathbb{Z}_p$ be a collision-resistant hash function. Define $\mathbf{Sig}^*_{\mathrm{A}} := (\mathsf{Setup}_{\mathrm{A}}, \mathsf{KeyGen}_{\mathrm{A}}, \mathsf{Sign}^*_{\mathrm{A}}, \mathsf{Verify}^*_{\mathrm{A}})$ with

- $\mathsf{Sign}^*_{\mathrm{A}}(sk, m) := \mathsf{Sign}_{\mathrm{A}}\big(sk, (\mathsf{Hash}(m)G, \mathsf{Hash}(m)H)\big)$
- $\mathsf{Verify}^*_{\mathrm{A}}(vk, (A, C, D, R, S), m) := \mathsf{Verify}_{\mathrm{A}}\big(vk, (A, C, D, R, S), (\mathsf{Hash}(m)G, \mathsf{Hash}(m)H)\big)$

Security against chosen-message attack follows by a straightforward reduction to security of $\mathbf{Sig}_{\mathrm{A}}$ and collision resistance of Hash.

5.2 Automorphic Signatures on Message Vectors

We give a general transform of a signature scheme whose message space contains its public-key space to one signing message vectors of arbitrary length that leaves the structure of verification invariant. The signer produces a one-time key pair (vk, sk), signs vk with her actual secret key and uses sk to sign every component of the vector. (Actually, for each component, she signs another transient key, which will sign the component and its *index* to prevent shuffling of messages. The vector's length is signed too, barring thus truncating.) Formally:

Definition 2. *Assume an efficiently computable injection* Msg *from* $\{1, \ldots, n_{max}\}$ *to the message space, where* n_{max} *is the maximum length of a message vector. Let* $\mathbf{Sig} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *be a signature scheme whose message space contains its public-key space. The* vector transform *of* $\mathbf{Sig}$ *is defined as* $\mathbf{Sig}' = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}', \mathsf{Verify}')$ *with*

$\mathsf{Sign}'_{sk}(M_1, \ldots, M_n)$ *with* $n \leq n_{max}$

- $(vk_0, sk_0) \leftarrow \mathsf{KeyGen}$; $(vk_1, sk_1) \leftarrow \mathsf{KeyGen}$; ... ; $(vk_n, sk_n) \leftarrow \mathsf{KeyGen}$;
- $\sigma := \big(vk_0, \mathsf{Sign}_{sk}(vk_0), \mathsf{Sign}_{sk_0}(\mathsf{Msg}(n)), \big(vk_i, \mathsf{Sign}_{sk_0}(vk_i), \mathsf{Sign}_{sk_i}(\mathsf{Msg}(i)), \mathsf{Sign}_{sk_i}(M_i)\big)_{i=1}^{n}\big)$.

$$\mathsf{Verify}'_{vk}\big((M_1, \ldots, M_n),\ (vk_0, \sigma_0, \rho_0, (vk_i, \sigma_i, \rho_i, \tau_i)_{i=1}^{n})\big) := \ \mathsf{Verify}_{vk}(vk_0, \sigma_0) \wedge \mathsf{Verify}_{vk_0}(\mathsf{Msg}(n), \rho_0) \wedge$$
$$\textstyle\bigwedge_{i=1}^{n} \big(\mathsf{Verify}_{vk_0}(vk_i, \sigma_i) \wedge \mathsf{Verify}_{vk_i}(\mathsf{Msg}(i), \rho_i) \wedge \mathsf{Verify}_{vk_i}(M_i, \tau_i)\big).$$

It is easily seen that if Verify is a conjunction of pairing-product equations then so is Verify'. The signatures of the transform of $\mathbf{Sig}_{\mathrm{A}}$ on a length-n vector are of size $12 + 17n$ group elements. This is however for the most general transform. If message vectors have some predefined structure (e.g., they are *signatures* as defined in Sect. 5.4), so that mixing and truncating is unlikely to result in a valid message, then *one* temporary key per message that signs all components suffices. The signature size is then $7 + 5n$ $\mathbb{G}$-elements. The following is proved in Appendix B.1.

Theorem 3. *If* $\mathbf{Sig}$ *is secure against EUF-CMA then so is* $\mathbf{Sig}'$.

5.3 Blind Automorphic Signatures

We show how to construct *blind signatures* based on the scheme $\mathbf{Sig}_{\mathrm{A}}$ (Scheme 2). Intuitively, a blind signature will be a Groth-Sahai proof of knowledge of a $\mathbf{Sig}_{\mathrm{A}}$-signature (which is precisely what we will use to achieve anonymity in our applications in Sect. 6). By witness indistinguishability of the proof system, two proofs of knowledge of different signatures on the same message (and under the same public key) are indistinguishable. Thus, it remains to ensure that the signer does not learn the message from a user that runs Obtain. The latter therefore *blinds* the message by multiplying it by some randomness and committing to it. The signer then produces a pre-signature from which, by

Let $ck \leftarrow \mathsf{Setup}_{\mathrm{GS}}$ be the common reference string. The signer holds her secret key x corresponding to public key (xG, xH). The user holds a message $(M, N) \in \mathcal{DH}$ on which he gets a blind signature.

1. **Obtain** The user chooses $\tau \leftarrow \mathbb{Z}_p$ and $\rho, \sigma \leftarrow \mathbb{Z}_p^3$, and sends the following:
   (a) $\mathbf{c}_M := \mathsf{Com}_{\mathrm{GS}}(ck, M, \rho)$, $\mathbf{c}_N := \mathsf{Com}_{\mathrm{GS}}(ck, N, \sigma)$, $\phi := \mathsf{Prove}_{\mathrm{GS}}\big(ck, E_{\mathcal{DH}}, (M, \rho), (N, \sigma)\big)$
   (b) $J := \frac{1}{\tau}(K + M)$
   (c) A zero-knowledge proof of knowledge of τ s.t. $\mathbf{c}_M$ commits to $\tau J - K$. (cf. Remark 1)
2. **Issue** If all proofs are valid, choose $c, r \leftarrow \mathbb{Z}_p$ and send:

$$A' := \tfrac{1}{x+c}(J + rT) \qquad C := cG \qquad D := cF \qquad R' := rG \qquad S' := rH$$

   The user sets $A := \tau A'$, $R := \tau R'$, $S := \tau S'$, and checks whether (A, C, D, R, S) is valid on (M, N). The blind signature is a Groth-Sahai proof of knowledge of (A, C, D, R, S) satisfying (4) for ck.

Fig. 1. Two-move blind signing protocol.

removing the randomness, the user obtains a signature of which he publishes commitments $\vec{\mathbf{c}}$ to its components and a proof π of validity.

The details of the signing protocol are given in Fig. 1. A blind signature $(\vec{\mathbf{c}}, \pi)$ on (M, N) under (X, Y) is verified by checking $\mathsf{Verify}_{\mathrm{GS}}(ck, E_{\mathsf{Ver}_\mathrm{A}}, \vec{\mathbf{c}}, \vec{\pi})$, with $E_{\mathsf{Ver}_\mathrm{A}}$ being Equation (4). Note that the scheme remains automorphic, because commitments and proofs are composed of group elements and are verified by checking PPEs.

Remark 1 (On the Proof of Knowledge). Step 1c of the protocol in Fig. 1 requires a zero-knowledge proof of knowledge of τ such that $M = \tau J - K$ where M is only given as a commitment. The proof is a proof of knowledge of *logarithms satisfying linear equations*: let $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ be the linear commitment key (cf. Sect. 2.2), then $\mathbf{c}_M = \big(\sum \rho_i u_{i,1},\ \sum \rho_i u_{i,2},\ M + \sum \rho_i u_{i,3}\big)$.

One possibility is to prove knowledge of $(\rho_1, \rho_2, \rho_3, \tau)$ such that

$$c_{M,1} = \textstyle\sum \rho_i u_{i,1} \qquad\qquad c_{M,2} = \sum \rho_i u_{i,2} \qquad\qquad c_{M,3} + K = \tau J + \sum \rho_i u_{i,3} \tag{7}$$

using interactive Schnorr-like proofs [Sch90] as exhibited in Appendix C.

A way of implementing the proof of knowledge (PoK) *without* increasing the number of rounds is by combining NIZK Groth-Sahai proofs for scalars (which are not extractable) with the bit-by-bit techniques of [FP09] (in Appendix A of the full version a NIZK PoK of a logarithm is constructed using tools from [GOS06]). We sketch our NIZK proof of knowledge of τ:

The common reference string is a key $ck' = (\mathbf{u}'_1, \mathbf{u}'_2, \mathbf{u}'_3) \leftarrow \mathsf{Setup}_{\mathrm{GS}}$. Both [GOS06] and [GS08] define a commitment to a *scalar* x as

$$\mathbf{d} \leftarrow \mathsf{ScCom}_{\mathrm{GS}}\big(ck', x, (r_1, r_2)\big) := \big(xu'_{3,1} + r_1 u'_{1,1},\ xu'_{3,2} + r_2 u'_{2,2},\ x(u'_{3,3} + G) + (r_1 + r_2)G\big)\ .$$

Let $\tau_j \in \{0, 1\}$ be such that $\tau = \sum \tau_j\, 2^j$. To guarantee extractability of τ, make commitments to its bits: $\mathbf{d}_{\tau,j} \leftarrow \mathsf{ScCom}_{\mathrm{GS}}\big(ck', \tau_j, (t_{j,1}, t_{j,2})\big)$. Then add perfectly WI proofs ξ_j that $\mathbf{d}_{\tau,j}$ commits to a bit as shown in [GOS06] (a proof consists of 6 group elements). Define $(\overline{\mathbf{d}}_\tau)_i := \sum_j 2^j\, (\mathbf{d}_{\tau,j})_i$ for $1 \le i \le 3$. Since the commitments are homomorphic, we have $\overline{\mathbf{d}}_\tau = \mathsf{ScCom}_{\mathrm{GS}}\big(ck', \tau, (\bar{t}_1, \bar{t}_2)\big)$ with $\bar{t}_i := \sum_j t_{j,i}\, 2^j$. Make commitments $\mathbf{d}_{\rho,i} := \mathsf{ScCom}_{\mathrm{GS}}\big(ck', \rho_i, (r_{i,1}, r_{i,2})\big)$ for $1 \le i \le 3$ and Groth-Sahai proofs π_1, π_2 and π_3 that the values committed in $\overline{\mathbf{d}}_\tau, \mathbf{d}_{\rho,1}, \mathbf{d}_{\rho,2}, \mathbf{d}_{\rho,3}$ satisfy the 3 linear equations in (7), respectively. The NIZK proof of knowledge is $\big((\mathbf{d}_{\tau,j}, \xi_j)_j, \mathbf{d}_{\rho,1}, \mathbf{d}_{\rho,2}, \mathbf{d}_{\rho,3}, \pi_1, \pi_2, \pi_3\big)$. It is verified by checking the ξ_j's, computing $\overline{\mathbf{d}}_\tau$ and checking the Groth-Sahai proofs π_1, π_2 and π_3.

Soundness follows from soundness of the proof systems and τ is extracted bit by bit using extraction key (α, β) for ck' (cf. Sect. 2.2): if $\tau_j = 0$ then $\mathbf{d}_{\tau,j}$ is linear and thus $(\mathbf{d}_{\tau,j})_3 = \alpha^{-1}(\mathbf{d}_{\tau,j})_1 + \beta^{-1}(\mathbf{d}_{\tau,j})_2$, which is not the case if $\tau_j = 1$. Zero-knowledge is shown as follows. Instead of running $\mathsf{Setup}_{\mathrm{GS}}$, produce a perfectly hiding key as in $\mathsf{SmSetup}$, but keep the used randomness as a simulation trapdoor *tr*. Then Groth-Sahai proofs for this type of equations ("multi-scalar multiplication equations") and the proofs ξ_j can be simulated using *tr* (cf. [GS08] and [GOS06] for details).

Remark 2 (Signing Committed Values). The core building block for P-signatures [BCKL08] is an interactive protocol allowing a user that published a commitment to obtain a signature on the committed value. By defining Step 1a of our blind-signature protocol to precede the protocol we get exactly this.

Theorem 4. *Under Assumptions 2 and 3 and DLIN, the scheme in Figure 1 is an unforgeable blind-signature scheme.*

See Appendix B.2 for a proof. We analyze the efficiency of our blind-signature scheme. If we use interactive concurrent-zero-knowledge proofs of knowledge (cf. Appendix C), we get a 5-round protocol. While using the non-interactive proof sketched above is less efficient due to the bit-by-bit techniques, it leads to a 2-round protocol. A $\mathbf{Sig}_{\mathrm{A}}$-signature consists of 5 group elements (GE) satisfying 2 linear equations and one general equation. Committing to the signature yields $5{\cdot}3$ GE and the Groth-Sahai proofs consist of $2{\cdot}3 + 9$ GE. The size of a blind signature is thus 30 GE.

5.4 Signatures on Signatures

For applications requiring automorphic signatures on *signatures*,[6] we suggest the scheme $\mathbf{Sig}_{\mathrm{FPV}}$ (Scheme 1) for the signatures to be signed, since its signatures fit the message space: a signature on a message/signature pair $\big((M,N),(A,C,D,R,S)\big)$ is a signature on the vector (cf. Sect. 5.2) of messages $\big((M,N),(C,D),(R,S)\big) \in \mathcal{DH}^3$. (Note that it is not necessary to sign A, as it is fully determined by these elements.)

6 Applications

We present various applications of automorphic signatures and their extensions. However, we merely sketch the application areas, as going into details would be beyond the scope of this paper.

6.1 Fully-Secure Group Signatures

In order to implement the model for group signatures by [BSZ05], Groth [Gro07] uses the following ingredients to achieve CCA-anonymity: the tag-based encryption scheme[7] [MRY04] $\mathbf{Enc}_{\mathrm{tb}}$ by Kiltz [Kil06] and a strong one-time signature scheme[8] $\mathbf{Sig}_{\mathrm{ot}}$.

A user produces a signature key pair $(\mathit{vk}, \mathit{sk})$ and is enrolled by the issuer who gives her a *certificate* *cert* on *vk*. Now to make a group signature on a message M, the user holding $(\mathit{cert}, \mathit{vk}, \mathit{sk})$ generates a key pair $(\mathit{vk}_{\mathrm{ot}}, \mathit{sk}_{\mathrm{ot}})$ for $\mathbf{Sig}_{\mathrm{ot}}$, makes a signature *sig* on $\mathit{vk}_{\mathrm{ot}}$ under *vk* and produces a Groth-Sahai WI proof of knowledge π of $(\mathit{cert}, \mathit{vk}, \mathit{sig})$ s.t. *cert* is a valid certificate on *vk* and *sig* is a signature on $\mathit{vk}_{\mathrm{ot}}$ valid under *vk*. She produces an $\mathbf{Enc}_{\mathrm{tb}}$-ciphertext C encrypting *sig* under tag $\mathit{vk}_{\mathrm{ot}}$ and adds a Groth-Sahai NIZK proof ζ that the encrypted value *sig* is the same as in π. Using $\mathit{sk}_{\mathrm{ot}}$, she finally

[6] A potential application is electronic cash, where a *coin* is basically a signature from the bank. In order to make double spenders traceable, one could have a user sign the coin when spending it.

[7] A tag-based encryption scheme is a public-key encryption scheme whose encryption and decryption algorithms take as additional argument a *tag*. A scheme is *selective-tag weakly CCA-secure* if an adversary outputting a tag t^* and two messages and getting an encryption of one of them under t^* cannot decide which one was encrypted—even when provided with an oracle decrypting any ciphertext for tags $t \neq t^*$.

[8] A signature scheme is *strongly one-time*, if no adversary granted a single weak chosen-message query can output a new signed message nor a new signature on the queried message. Groth uses the weak Boneh-Boyen signature from [BB04].

makes a signature sig_{ot} on $(M, vk_{\mathrm{ot}}, \pi, C, \zeta)$ and outputs the group signature $\sigma = (vk_{\mathrm{ot}}, \pi, C, \zeta, sig_{\mathrm{ot}})$. The opener holds a key enabling her to extract $(cert, vk, sig)$ from π. The key vk allows to determine the signer and sig acts as a non-frameable proof of correct tracing.

[FPV09] suggest to substitute the certified-signature scheme based on the "q-U Assumption" by one based on the more natural DHSDH. Their replacement however uses Waters signatures [Wat05] which entail a dramatic increase of the public-key size. This is avoided by using instead the two-level construction $\mathbf{Sig}_{\mathrm{2FPV}}$ given before Scheme 2 (based on DHSDH as well), which actually *is* a certified-signature scheme: certificates are signatures on user verification keys (xG, xH); messages are of the form (mG, mL) and signatures use parameters (G, L, K, T). The certificate-verification key is one element of $\mathbb{G}$ (in the construction of [Gro07], it consists of two elements of $\mathbb{G}$ and one of $\mathbb{G}_T$).

6.2 P-Signatures and Anonymous Credentials

In order to realize *non-interactive anonymous credentials*, Belenkiy et al. [BCKL08] introduce a new primitive, called *P-signature.* This primitive extends a signature scheme and a commitment scheme by the following functionalities: a protocol Issue ↔ Obtain between a signer and a user allows the latter to obtain a signature on a value the signer only knows a commitment to; the holder of a message and a signature on it can produce a commitment to the message and a proof of knowledge of the signature; and two commitments can be proved to be to the same value.

The commitments and proofs are instantiated with the Groth-Sahai methodology; the compatible signature scheme is the one discussed in Sect. 1. Our scheme $\mathbf{Sig}_{\mathrm{A}}$, combined with the modified blind-issuing protocol from Remark 2, suitably replaces their scheme and overcomes the aforementioned shortcomings: it avoids the TDH assumption, actual message/signature pairs can be extracted from the proofs (rather than a function of them) and it provides an efficient Issue ↔ Obtain-protocol (the one in [BCKL08] resorts to generic secure multiparty computation).

6.3 Anonymous Delegation of Signing Rights

Anonymous Proxy Signatures. Anonymous proxy signatures (APS) generalize group signatures in the following sense: everyone can become a group manager by delegating his signing rights to other users who can then sign in his name while remaining anonymous; moreover, received rights can be *re-delegated* consecutively. We give a brief overview of the model defined in [FP08].

Algorithm Setup establishes the public parameters. *Users* generate key pairs using KeyGen and run a protocol Reg with the *issuer* and their *opener* when joining the system. (This is essential to achieve traceability; see below.) To delegate to Bob, Alice runs Delgt on Bob's public key, which produces a *warrant* that she gives to Bob. With this warrant, Bob can either sign, or *re-delegate* to Carol, in which case Carol can again re-delegate or produce an *anonymous proxy signature* with PSign on behalf of Alice, which is verifiable by Verify on Alice's verification key.

Anonymity ensures that from a proxy signature one cannot tell who actually signed (or re-delegated), thus Bob and Carol remain anonymous. To prevent misuse, Alice's opener can revoke the anonymity of the intermediate delegators and the proxy signer. *Traceability* asserts that every valid signature can be opened to registered users and *non-frameability* guarantees that no adversary, even when colluding with the issuer, openers and other users, can produce a signature that opens to an honest user for a delegation or a signing she did not perform.

A Generic Construction. The generic construction by [FP08] proving feasibility of the model is as follows. Assume an EUF-CMA-secure signature scheme. The issuer and the users choose a signing/verification key pair each. When enrolling, a user U_i obtains a signature $cert_i$ on her verification key vk_i from the issuer. A warrant $warr_{1\to 2}$ from user U_1 to user U_2 is a signature on (vk_1, vk_2) valid under vk_1. U_2 re-delegates to U_3 by sending $warr_{1\to 2}$ and $warr_{2\to 3}$, a signature on (vk_1, vk_2, vk_3) under vk_2. Additionally, in each delegation step, the delegators' certificates are also passed on.

Given a warrant $(\mathit{warr}_{1\to 2}, \mathit{warr}_{2\to 3})$, U_3 proxy-signs a message M on behalf of U_1 as follows: first produce a signature *sig* on $(\mathit{vk}_1, \mathit{vk}_2, \mathit{vk}_3, M)$ using sk_3; then define the *plain* proxy signature as $(\mathit{warr}_{1\to 2}, \mathit{vk}_2, \mathit{cert}_2, \mathit{warr}_{2\to 3}, \mathit{vk}_3, \mathit{cert}_3, \mathit{sig})$. In general we say that a plain proxy signature $\Sigma = (\mathit{warr}_{1\to 2}, \ldots, \mathit{vk}_k, \mathit{cert}_k, \mathit{sig})$ on message M under vk_1 is valid if:

- $\forall i$: cert_i is valid under the issuer's verification key;
- $\forall i$: $\mathit{warr}_{i\to i+1}$ is a signature on $(\mathit{vk}_1, \ldots, \mathit{vk}_{i+1})$ valid under vk_i; (8)
- *sig* is a signature on $(\mathit{vk}_1, \ldots, \mathit{vk}_k, M)$ valid under vk_k.

Now to transform this into an *anonymous* proxy signature, the signer encrypts Σ under the public key of U_1's opener and adds a NIZK proof that the plaintext satisfies the above. Due to her decryption key, the opener can retrieve the plain signature and thus trace the delegators and the signer. The warrants and *sig* are non-frameable proofs of correct tracing.

Concrete Instantiations. Restricting the model to CPA-anonymity, the building blocks can be instantiated as follows: define encryption to be linear commitments (which can be "decrypted" due to extractability) and use Groth-Sahai proofs to show that the verification relations are satisfied by the committed values. For this to work however, the plain proxy signatures must fit the Groth-Sahai framework; meaning that the EUF-CMA signature scheme's verification keys, messages and signatures must be group elements satisfying pairing-product equations.

Fuchsbauer and Pointcheval [FP09] follow this overall approach using basically Groth-Sahai proofs in the *subgroup-decision* [BGN05] instantiation, which confines their scheme to one general opener (once the bilinear group is fixed there is only one key pair for encryption). The compatible EUF-CMA-secure signature scheme they construct has several shortcomings: besides being based on an unusual new assumption (the "XF-assumption"), it is far from being practical, since a public key contains several commitments to each *bit* of the corresponding secret key. Moreover, it imposes a maximum number of re-delegations in one signature that is fixed during setup.

Replacing their scheme by $\mathbf{Sig}_{\mathrm{A}}$ (Sect. 4), which meets the necessary conditions, and using the DLIN instantiation of Groth-Sahai, we circumvent all these shortcomings, getting an efficient scheme based on reasonable assumptions. Moreover, we satisfy the model of [FP08] since we can assume several openers each publishing their own commitment key (due to our DLIN-based proofs), we do not impose a limit on the number of re-delegations, and we show how to achieve CCA-anonymity (see below).

CCA-Anonymous Proxy Signatures. CCA-anonymity (i.e., anonymity against adversaries provided with an opening oracle) of Groth's group signatures sketched in Sect. 6.1 is proved as follows: modify the security game by substituting the opener's commitment key by one that results in perfectly hiding commitments and WI proofs; due to the additional encryptions contained in a group signature, opening queries can still be simulated.

We transform our APS scheme into one satisfying CCA-anonymity analogously. Suppose a proxy signer holds $W := (\mathit{vk}_1, (\mathit{warr}_i, \mathit{cert}_i, \mathit{vk}_i)_{i=2}^{k})$ and sk_k. To make a signature, she first chooses keys for a one-time signature $(\mathit{vk}_{\mathrm{ot}}, \mathit{sk}_{\mathrm{ot}}) \leftarrow \mathsf{KeyGen}_{\mathrm{ot}}$, signs $\mathit{vk}_{\mathrm{ot}}$ (instead of M) with her personal key sk_k yielding *sig*, makes commitments $\vec{\mathbf{c}}$ to the elements of W and *sig*, and adds a WI proofs ϕ_j for each equation E_j in (8)—as in the original scheme.

In addition, for $2 \le i \le k$ she computes an $\mathbf{Enc}_{\mathrm{tb}}$-encryption C_i of warr_i under tag $\mathit{vk}_{\mathrm{ot}}$ and makes a NIZK proof ζ_i that the plaintext of C_i is the value committed in $\mathbf{c}_{\mathit{warr}_i}$. Finally she sets $\mathit{sig}_{\mathrm{ot}} := \mathsf{Sign}_{\mathrm{ot}}(\mathit{sk}_{\mathrm{ot}}, (\mathit{vk}_{\mathrm{ot}}, M, \vec{\mathbf{c}}, \vec{\phi}, \vec{C}, \vec{\zeta}))$ and outputs the signature $(\mathit{vk}_{\mathrm{ot}}, \vec{\mathbf{c}}, \vec{\phi}, \vec{C}, \vec{\zeta}, \mathit{sig}_{\mathrm{ot}})$. A signature is *valid* if $\mathit{sig}_{\mathrm{ot}}$ is valid under $\mathit{vk}_{\mathrm{ot}}$, the proofs ϕ_j are valid for all j, and the proofs ζ_i and the ciphertexts C_i are valid for all i.

6.4 Extending Anonymous Proxy Signatures

We briefly sketch how to instantiate the extended model of APS discussed at the end of Sect. 1. A formal description can be found in Appendix D.

Blind Delegation. Using the variant of our blind-signature protocol discussed in Remark 2, we can define *blind delegation*: given a commitment to a verification key, a user can delegate her signing rights to the holder of that key without learning her identity.

Delegator Anonymity. Due to the modularity of Groth-Sahai proofs (for each equation its proof only depends on the commitments to the variables appearing in it), the "anonymization" of a signature need not be delayed until the proxy signing: warrants can be anonymized by the delegators already and randomized in each delegation step (which prevents linkability of signatures). However, we need to revise the way warrants are defined, since the present scheme requires knowledge of the identities of all previous delegators for their construction. We follow the general approach by [BCC$^+$09], who associate an identifier *id* to each original delegation. A warrant from the user at level i in the delegation chain to the next one is then a signature on $(\mathsf{Hash}(id \,\|\, i), vk_{i+1})$ under vk_i, where $\mathsf{Hash}\colon \{0,1\}^* \to \mathbb{G}$ is a collision-resistant hash function.[9] The hash value prevents combining different warrants and reordering the same warrant.

Consider the following situation (we simplify our exposition by omitting the certificates from the issuer): Oliver (the **o**riginal delegator), owning vk_O, delegated to Alice by giving her a signature $warr_{O\to A}$ on her key vk_A. Alice delegates to Bob sending him $warr_{A\to B}$. Bob can now delegate to Carol *without revealing Alice's identity*: He makes commitments $\mathbf{c}_{O\to A}$, $\mathbf{c}_A$ and $\mathbf{c}_{A\to B}$ to $warr_{O\to A}$, vk_A and $warr_{A\to B}$, respectively. Moreover, he makes a *trivial* commitment $\mathbf{c}_B = \mathsf{Com}_{\mathrm{GS}}(ck, vk_B, 0)$ to his own key, and the following proofs: $\phi_{O\to A}$ for $\mathbf{c}_{O\to A}$ containing a valid warrant from vk_O to the content of $\mathbf{c}_A$, and $\phi_{A\to B}$ for $\mathbf{c}_{A\to B}$ containing a valid warrant from the content of $\mathbf{c}_A$ to the content of $\mathbf{c}_B$. He sends Carol $\widetilde{warr} := (vk_O, \mathbf{c}_A, \mathbf{c}_{O\to A}, \phi_{O\to A}, \mathbf{c}_B, \mathbf{c}_{A\to B}, \phi_{A\to B})$ and a warrant $warr_{B\to C}$.

Now, Carol produces a signature on behalf of Oliver on M as follows (re-delegation works analogously): make a signature *sig* on M valid under vk_C; *randomize* the commitments and adapt the proofs in $\widetilde{warr}$, in particular, set $\mathbf{c}'_B := \mathsf{RdCom}_{\mathrm{GS}}(ck, \mathbf{c}_B, \rho_B)$; make commitments to $warr_{B\to C}$, vk_C and *sig*, and proofs of validity of $warr_{B\to C}$ and *sig*. Note that for the first proof the randomness of the related commitments—in particular $\mathbf{c}'_B$—is required. Since $\mathbf{c}_B$ was a trivial commitment, the randomness of $\mathbf{c}'_B$ is ρ_B which was chosen by Carol (cf. end of Sect. 2.2)!

Remark 3. (1) Note that delegator-anonymous delegation is compatible with blind delegation: instead of simply sending $warr_{B\to C}$, Bob runs the interactive blind-issuing protocol with Carol, upon which she obtains $warr_{B\to C}$ and continues as above.

(2) Bob could even hide *his own identity* to Carol as follows: he sends (hiding) commitments to his own key and to $warr_{B\to C}$, and in addition a trivial commitment to Carol's key and proof of validity of $warr_{B\to C}$. Carol randomizes what Bob sent her, commits to a signature on the message and proves validity. Unfortunately this does not seem to be compatible with blind delegation.

In Appendix D, we formally describe an instantiation of anonymous proxy signatures with delegator anonymity.

Multiple Original Delegators. If we allow delegation to take the form of a tree (whose leaves represent original delegators) rather than a list, we can define proxy signatures on behalf of several originators. For example, consider three original delegators O, P, Q, the first of which delegates to A who in turn re-delegates to B. B is also delegated by P and re-delegates the rights for both O and P to C. Moreover Q delegates to C. Now C can produce a signature on behalf of O, P *and* Q.

In general, we define a *multi-originator signature* (MOS) recursively: A (plain) MOS consists of a signature on the message, the signer's verification key and a list of objects *del* for the signer (which represent the delgations to her). A *del* for user U is either a warrant from an originator for U or a warrant from a user U', the verification key of U' and a list of *del*'s for U'. A (plain) signature on behalf of a set of originators is valid if the signature on the message is valid, all warrants are valid and

[9] Since *id* and i are publicly known, $\mathsf{Hash}(id \,\|\, i) \in \mathbb{G}$ will be considered as a constant in the Groth-Sahai proofs.

it contains a warrant from each of the originators. As for the single-originator case, a plain signature is anonymized by committing to its components and adding proofs of validity.

In the above example, a signature by C on behalf of O, P and Q then has the following form (we let $\psi_{U_1 \to U_2}$ denote $\mathbf{c}_{U_1 \to U_2} \,\|\, \phi_{U_1 \to U_2}$, and ψ_M denote a commitment to *sig* and a proof of validity):

$$\Big\{\psi_M, \mathbf{c}_C, \big\{\big\{\psi_{B\to C}, \mathbf{c}_B, \{\{\psi_{A\to B}, \mathbf{c}_A, \psi_{O\to A}\}, \psi_{P\to B}\}\big\}, \psi_{Q\to C}\big\}\Big\} \;.$$

7 Conclusions

We introduced the concept of automorphic signatures and gave two instantiations; the first is based on known assumptions while the second is more efficient and relies on a new assumption, which we prove to hold in the generic group model. We also gave a practical blind automorphic signature.

We then illustrated the benefits of automorphic signatures by constructing fully-secure group signatures and anonymous credentials and by giving the first efficient instantiation of anonymous proxy signatures, which satisfies additional security requirements that have not been considered so far.

Acknowledgments

The author would like to thank David Pointcheval for many invaluable discussions that led to the present paper. This work was supported by EADS, the French ANR-07-SESU-008-01 PAMPA Project and the European Commission through the IST Program under Contract ICT-2007-216646 ECRYPT II.

References

[BB04] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73. Springer, May 2004.

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

[BCC$^+$09] Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. Randomizable proofs and delegatable anonymous credentials. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 108–125. Springer, August 2009.

[BCKL08] Mira Belenkiy, Melissa Chase, Markulf Kohlweiss, and Anna Lysyanskaya. P-signatures and noninteractive anonymous credentials. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 356–374. Springer, March 2008.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 325–341. Springer, February 2005.

[BP04] Mihir Bellare and Adriana Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 273–289. Springer, August 2004.

[BPW03] Alexandra Boldyreva, Adriana Palacio, and Bogdan Warinschi. Secure proxy signature schemes for delegation of signing rights. Cryptology ePrint Archive, Report 2003/096, 2003. http://eprint.iacr.org/.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73. ACM Press, November 1993.

[BSZ05] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In Alfred Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–153. Springer, February 2005.

[BW06] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 427–444. Springer, May / June 2006.

[BW07] Xavier Boyen and Brent Waters. Full-domain subgroup hiding and constant-size group signatures. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *PKC 2007*, volume 4450 of *LNCS*, pages 1–15. Springer, April 2007.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998.

[Cha83] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1983.

[Cha85] David Chaum. Security without identification: Transaction systems to make big brother obsolete. *Commun. ACM*, 28(10):1030–1044, 1985.

[Cv91] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer, April 1991.

[Dam92] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 445–456. Springer, August 1992.

[FP08] Georg Fuchsbauer and David Pointcheval. Anonymous proxy signatures. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *SCN 08*, volume 5229 of *LNCS*, pages 201–217. Springer, September 2008.

[FP09] Georg Fuchsbauer and David Pointcheval. Proofs on encrypted values in bilinear groups and an application to anonymity of signatures. In Hovav Shacham and Brent Waters, editors, *PAIRING 2009*, volume 5671 of *LNCS*, pages 132–149. Springer, August 2009. Full version available at http://eprint.iacr.org/2008/528.

[FPV09] Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. Transferable constant-size fair e-cash. In Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors, *CANS 2009*, volume 5888 of *LNCS*, pages 226–247. Springer, December 2009. Full version available at http://eprint.iacr.org/2009/146.

[GL07] Jens Groth and Steve Lu. A non-interactive shuffle with pairing based verifiability. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 51–67. Springer, December 2007.

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

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for NIZK. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 97–111. Springer, August 2006.

[Gro06] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, December 2006.

[Gro07] Jens Groth. Fully anonymous group signatures without random oracles. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 164–180. Springer, December 2007.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, April 2008.

[Kil06] Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 581–600. Springer, March 2006.

[MRY04] Philip D. MacKenzie, Michael K. Reiter, and Ke Yang. Alternatives to non-malleability: Definitions, constructions, and applications. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 171–190. Springer, February 2004.

[MUO96] Masahiro Mambo, Keisuke Usuda, and Eiji Okamoto. Proxy signatures for delegating signing operation. In *ACM CCS 96*, pages 48–57. ACM Press, March 1996.

[Oka06] Tatsuaki Okamoto. Efficient blind and partially blind signatures without random oracles. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 80–99. Springer, March 2006.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.

[Sch90] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, August 1990.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer, May 1997.

[SMP08] Jacob C. N. Schuldt, Kanta Matsuura, and Kenneth G. Paterson. Proxy signatures secure against proxy key exposure. In Ronald Cramer, editor, *PKC 2008*, volume 4939 of *LNCS*, pages 141–161. Springer, March 2008.

[TW05] Mårten Trolin and Douglas Wikström. Hierarchical group signatures. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *ICALP 2005*, volume 3580 of *LNCS*, pages 446–458. Springer, July 2005.

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, May 2005.

A The q-DAHSDH Assumption

A.1 A Note on DAHSDH

To make $\mathbf{Sig}_{\mathrm{FPV}}$ (Scheme 1) an automorphic scheme without the detour via a transient key, one could be tempted to simply expand the public key by $Y = (\log_G X)H$ and assume a variant of DHSDH where Y is given as part of the instance.

However, this assumption is wrong, as it succumbs to the following attack: Given an instance, set $A^* := -A_1$, $C^* := -2X - C_1$, $D^* := -2Y - D_1$, $V^* := V_1$, $W^* := W_1$. Then we have $e(A^*, X + C^*) =$

$e(-A_1, -C_1) = e(K + V_1, G) = e(K + V^*, G)$. The attack comes from the fact that we can use X to build C^* since given Y, the "shadow" of X, we are able to construct the shadow of C^*.

This is what makes it indispensable to use a different basis for the shadow of C, leading to a generically secure assumption, as proved below.

A.2 Generic Security of the q-DAHSDH Assumption

We restate the assumption for convenience:

(q-DAHSDH) Given $(G, F, H, K, X = xG, Y = xH)$ and $q - 1$ tuples

$$\left(A_i = \tfrac{1}{x+c_i}(K + v_i G),\ C_i = c_i G,\ D_i = c_i F,\ V_i = v_i G,\ W_i = v_i H\right)\ ,$$

with $c_i, v_i \in \mathbb{Z}_p^*$ for $i = 1, \ldots, q-1$, it is hard to output a new tuple $(A^*, C^*, D^*, V^*, W^*)$ that satisfies

$$e(A^*, X + C^*) = e(K + V^*, G) \qquad e(C^*, F) = e(G, D^*) \qquad e(V^*, H) = e(G, W^*)\ . \tag{1}$$

Theorem 5. *The q-DAHSDH assumption holds in generic bilinear groups when q is a polynomial.*

Proof. We assume that the reader is familiar with the methodology of proofs in the generic group model and thus focus on our particular assumption. We work with the "discrete-log" representation of all group elements w.r.t. basis G. A q-DAHSDH instance is thus represented by the following rational fractions:

$$1,\ f,\ h,\ k,\ x,\ xh,\ \{a_i = \tfrac{k+v_i}{x+c_i},\ c_i,\ c_i f,\ v_i,\ v_i h\}_{i=1}^{q-1} \tag{2}$$

Considering the logarithms of the $\mathbb{G}_T$-elements in (1) w.r.t. the basis $e(G, G)$ yields

$$a^*(x + c^*) = k + v^* \qquad d^* = c^* f \qquad w^* = v^* h \tag{3}$$

In a generic group, all the adversary can do is apply the group operation to the elements of its input. We will show that the only linear combinations $(a^*, c^*, d^*, v^*, w^*)$ of elements in (2) satisfying (3) are $(a^* = a_i, c^* = c_i, d^* = c_i f, v^* = v_i, w^* = v_i h)$ for some i; which means all the adversary can do is return a quintuple from the instance. We make the following ansatz for a^*:

$$a^* = \alpha + \alpha_f f + \alpha_h h + \alpha_k k + \alpha_x x + \alpha_y xh + \sum \alpha_{a_i} \tfrac{k+v_i}{x+c_i} + \sum \alpha_{c,i} c_i + \sum \alpha_{d,i} c_i f + \sum \alpha_{v,i} v_i + \sum \alpha_{w,i} v_i h$$

Since for any c^* the adversary forms, it has to produce $c^* f$ as well, we can limit the elements used for c^* to those of which their product with f is also given: 1 and c_i (for all i). The same argument holds for v^* and elements for which we have their products with h. We set thus

$$c^* = \gamma + \sum \gamma_{c,i} c_i \qquad v^* = \mu + \mu_x x + \sum \mu_{v,i} v_i$$

which implicitly satisfies the last two equations of (3) since d^* and w^* can then be defined as

$$d^* = \gamma f + \sum \gamma_{c,i} c_i f \qquad w^* = \mu h + \mu_x xh + \sum \mu_{v,i} v_i h$$

We substitute a^*, c^*, v^* by their ansätze in the first equation of (3), that is $a^*(x + c^*) - v^* = k$. After some rearranging we get

$$(\alpha\gamma - \mu)\ 1\ +\ (\alpha_f \gamma)\ f\ +\ (\alpha_h \gamma)\ h\ +\ (\alpha + \alpha_x \gamma - \mu_x)\ x\ +\ (\alpha_h + \alpha_y \gamma)\ xh\ + \tag{4a}$$

$$\sum (\alpha_{a,i}\gamma)\ \tfrac{k+v_i}{x+c_i}\ +\ \sum (\alpha_{c,i}\gamma + \alpha\gamma_{c,i})\ c_i\ +\ \sum (\alpha_{d,i}\gamma + \alpha_f \gamma_{c,i})\ c_i f\ +\ \sum (\alpha_{w,i}\gamma)\ v_i h\ + \tag{4b}$$

$$(\alpha_f)\ xf\ +\ (\alpha_k)\ xk\ +\ (\alpha_x)\ x^2\ +\ (\alpha_y)\ x^2 h\ +\ \sum (\alpha_{v,i})\ v_i x\ +\ \sum (\alpha_{w,i})\ v_i xh\ + \tag{4c}$$

$$\sum (\alpha_{c,i} + \alpha_x \gamma_{c,i})\ c_i x\ +\ \sum (\alpha_{d,i})\, c_i xf\ +\ \sum (\alpha_h \gamma_{c,i})\, c_i h\ +\ \sum (\alpha_k \gamma_{c,i})\, c_i k\ +\ \sum (\alpha_y \gamma_{c,i})\, x c_i h\ + \tag{4d}$$

$$\sum\sum (\alpha_{c,i}\gamma_{c,j})\, c_i c_j\ +\ \sum\sum (\alpha_{d,i}\gamma_{c,j})\, c_i c_j f\ +\ \sum\sum (\alpha_{v,i}\gamma_{c,j})\, v_i c_j\ +\ \sum\sum (\alpha_{w,i}\gamma_{c,j})\, v_i c_j h\ + \tag{4e}$$

$$\underbrace{(\alpha_k \gamma)}_{=:\lambda_k}\ k\ +\ \sum\ \underbrace{(\alpha_{v,i}\gamma - \mu_{v,i})}_{=:\lambda_{v,i}}\ v_i\ +\ \sum\ \underbrace{(\alpha_{a,i})}_{=:\lambda_{xa,i}}\ \tfrac{x(k+v_i)}{x+c_i}\ +\ \sum\sum\ \underbrace{(\alpha_{a,i}\gamma_{c,j})}_{=:\lambda_{ca,i,j}}\ \tfrac{c_j(k+v_i)}{x+c_i}\ =\ k \tag{4f}$$

Comparison of coefficients[10] of the two sides of the equation shows that all coefficients in lines (4a)–(4e) must be 0, whereas for the last line we have a different situation: Adding $\frac{x(k+v_i)}{x+c_i}$ and $\frac{c_i(k+v_i)}{x+c_i}$ reduces to $k+v_i$ (but this is the only combination that reduces). We have thus

$$\lambda_{xa,i} = \lambda_{ca,i,i} \quad \text{for all } i \qquad\qquad \lambda_{ca,i,j} = 0 \quad \text{for all } i \neq j \tag{5}$$

and moreover

$$\text{coefficient of } k\text{:} \quad \sum \lambda_{xa,i} + \lambda_k = 1 \qquad\qquad \text{coefficient of } v_i\text{:} \quad \lambda_{xa,i} + \lambda_{v,i} = 0 \tag{6}$$

We now solve the equations "all coefficients in Lines (4a) to (4e) equal 0", and Equations (5) and (6) for $\big(\alpha, \alpha_f, \alpha_h, \alpha_k, \alpha_x, \alpha_y, \gamma, \mu, \mu_x, \{\alpha_{a_i}, \alpha_{c,i}, \alpha_{d,i}, \alpha_{v,i}, \alpha_{w,i}, \gamma_{c,i}, \mu_{v,i}\}\big)$:

Line (4c) and the second term in Line (4d) immediately yield: $\alpha_f = \alpha_k = \alpha_x = \alpha_y = \alpha_{v,i} = \alpha_{w,i} = \alpha_{d,i} = 0$ for all i. Now $\alpha_y = 0$ implies $\alpha_h = 0$ by the last term in (4a), and $\alpha_x = 0$ implies $\alpha_{c,i} = 0$ for all i by the first term in in (4d). The first equations in (5) and (6) give

$$\alpha_{a,i}(1-\gamma_{c,i}) = 0 \quad \text{for all } i \qquad\qquad \sum \alpha_{a,i} + \alpha_k \gamma \overset{(\alpha_k=0)}{=} \sum \alpha_{a,i} = 1 \tag{7}$$

which together imply that for some i: $\gamma_{c,i} \neq 0$. The second term in (4b) requires that for all i: $\alpha_{c,i}\gamma + \alpha\gamma_{c,i} \overset{(\alpha_{c,i}=0)}{=} \alpha\gamma_{c,i} = 0$, thus $\alpha = 0$ since some $\gamma_{c,i} \neq 0$. Combining this with $\alpha_x = 0$ and the the fourth term in (4a), we get $\mu_x = 0$. The first term in (4b) and the second equation in (7) implies $\gamma = 0$, which yields $\mu = 0$ by the first term in in (4a). The only variables not shown to be 0 so far are $\{\alpha_{a,i}, \gamma_{c,i}, \mu_{v,i}\}$.

We show that there exists exactly one index i^* such that $\alpha_{a,i^*} \neq 0$: if we had $i \neq j$ s.t. $\alpha_{a,i} \neq 0$ and $\alpha_{a,j} \neq 0$ then by the first equation in (7) we would have $\gamma_{c,i} = \gamma_{c,j} = 1$. This however contradicts the second equation in (5) which states $\alpha_{a,i}\gamma_{c,j} = 0$ for all $i \neq j$. The same equation ensures that $\gamma_j = 0$ for all $j \neq i^*$. We have $\gamma_{c,i^*} = 1$ by (7) and by the last equation of (6): $\alpha_{a,i} + \alpha_{v,i}\gamma - \mu_{v,i} \overset{(\gamma=0)}{=} \alpha_{a,i} - \mu_{v,i} = 0$, we have $\alpha_{a,i} = \mu_{v,i}$ for all i.

We proved thus that there exists a i^* such that $a^* = \frac{k+v_{i^*}}{x+c_{i^*}}$, $c^* = c_{i^*}$ and $v^* = v_{i^*}$, which means the only tuples $(A^*, C^*, D^*, V^*, W^*)$ satisfying (1) and being generically constructable from a DAHSDH instance are the tuples from that instance. □

B Security Proofs for Extensions of $\mathbf{Sig}_{\mathrm{A}}$

B.1 Proof of Theorem 3

Let q be the maximal number of the adversary's signing queries, let n_m the maximal length of all queried vectors and the output vector. Let $\vec{M}^{(i)} := (M_1^{(i)}, \ldots, M_{n_i}^{(i)})$ denote the adversary's i-th signing query, and let

$$\big(\vec{M}^* := (M_1^*, \ldots, M_{n^*}^*), \sigma^* := (vk_0^*, \sigma_0^*, (\sigma_i^*, \rho_i^*, \tau_i^*))\big)$$

be the adversary's output. Let vk be a challenge for (KeyGen, Sign, Verify). We distinguish four types of forgers:

1. $\forall i : vk_0^* \neq vk_0^{(i)}$. Set vk to be the challenge verification key and answer signing queries by choosing (vk_0, sk_0), querying vk_0 to the Sign-oracle and using sk_0 to sign the vector entries and indices. If σ^* is of Type 1, then (vk_0^*, σ_0^*) is a forgery under vk.

[10] To do straightforward comparison of coefficients, we actually would have to multiply the equation by $\prod_{i=1}^{q-1}(x+c_i)$ first. For the sake of presentation, we keep the fractions and instead introduce new equations for the cases where a linear combination leads to a fraction that cancels down.

2. $\exists i : vk_0^* = vk_0^{(i)}$ and: $\exists j\, \forall k : vk_j^* \neq vk_k^{(i)}$ or $n^* \neq n_i$. Choose $i^* \leftarrow \{1, \ldots, q\}$. Produce $(vk', sk') \leftarrow \mathsf{KeyGen}(1^k)$ and give the adversary vk' as challenge. Answer all queries as in the protocol, except for the i^*-th query: Set $vk_0^{(i^*)} := vk$, choose keys $(vk_j^{(i^*)}, sk_j^{(i^*)})$ and query signatures on $vk_j^{(i^*)}$ to the Sign-oracle. Complete the signature. If σ^* is of Type 2 and we guessed correctly ($i^* = i$) then (vk_j^*, σ_j^*) or (n^*G, ρ_0^*) is a forgery under vk.
3. $\exists i : vk_0^* = vk_0^{(i)}$, $n^* = n_i$, and $\forall j\, \exists k_j : vk_j^* = vk_{k_j}^{(i)}$ and $\exists j : M_j^* \neq M_{k_j}^{(i)}$. We choose $i^* \leftarrow \{1, \ldots, q\}$ and $j^* \leftarrow \{1, \ldots, n_m\}$ and set $vk_{j^*}^{(i^*)} := vk$. On guessing correctly ($i^* = i$ and $j^* = j$), (M_j^*, τ_j^*) is a forgery.
4. $\exists i : vk_0^* = vk_0^{(i)}$, $n^* = n_i$, and $\forall j\, \exists k_j : vk_j^* = vk_{k_j}^{(i)}$ and $\forall j : M_j^* = M_{k_j}^{(i)}$. Since $\vec{M}^*$ is a new message of equal length, there must be a j s.t. $k_j \neq j$. We set vk as for Type 3. On guessing correctly, (jG, ρ_j^*) is a forgery under vk.

□

B.2 Proof of Theorem 4

The protocol is correct: The signer sends $A' = \frac{1}{x+c}(J + rT) = \frac{1}{\tau} \cdot \frac{1}{x+c}(K + M + \tau r T)$, $R' = rG$, $S' = rH$ and the user sets $A = \tau A' = \frac{1}{x+c}(K + \tau r T + M)$, $R = \tau R' = \tau r G$, $S = \tau S' = \tau r H$, which together with C and D constitutes a valid signature.

Blindness requires the following: *If we are given two messages from the signer and run* Obtain *twice for these messages (in random order) with it, and then give the two resulting signature/message pairs, then the signer cannot relate them to their issuings.*

Let Game 0 be the original game; let Game 1 be the same game, but setting $ck \leftarrow \mathsf{SmSetup}$ (leading to perfectly WI commitments and proofs). The two games are indistinguishable by DLIN. In Game 2, we simulate the proof in Step 1c, which is indistinguishable since the proof is zero-knowledge. A signature/message pair $\big((\vec{\mathbf{c}}, \pi), (M, N)\big)$ that the adversary gets in the end perfectly hides the signature, since the commitments are under ck. For every pair $(M', N') \in \mathcal{DH}$, there exist ρ and σ that lead to $(\mathbf{c}_M, \mathbf{c}_N, \phi)$ and there exists τ leading to J. Since ζ is simulated, it only depends on $\mathbf{c}_M$ and J. Thus the adversary's view $(\mathbf{c}_M, \mathbf{c}_N, \phi, J, \zeta)$ after the first round of the protocol is independent of (M, N) in Game 2.

Unforgeability means: *After running the protocol $q - 1$ times with an honest signer, no adversary can output q different messages and valid signatures on them.*

We reduce unforgeability to the security of the underlying signature scheme (Theorem 2): We run $(ck, ek) \leftarrow \mathsf{ExSetup}$, publish ck and then run the protocol as signer with adversary $\mathcal{A}$. Whenever $\mathcal{A}$ sends $(\mathbf{c}_M, \mathbf{c}_N, \phi, J, \zeta)$, we use ek to extract (M, N) and extract τ from ζ. We query our oracle for a signature on (M, N). On receiving (A, C, D, R, S), we give the adversary $(\frac{1}{\tau}A, C, D, \frac{1}{\tau}R, \frac{1}{\tau}S)$. The adversary wins the game if after $q - 1$ issuings, it outputs q blind signatures on different messages. We extract the signature on a message which we did not query to our own oracle. By soundness of GS proofs, this is a valid signature and can thus be returned as a forgery. □

C An Interactive Instantiation of the Proof of Knowledge

In Sect. 5.3, we require the following proof of knowledge. Let $(u_{i,j})_{i,j=1,2,3} \in \mathbb{G}^9$, let $K, J \in \mathbb{G}$. Given an instance $(c_1, c_2, c_3) \in \mathbb{G}^3$, we prove knowledge of (s_1, s_2, s_3, τ) such that

$$c_1 = \sum s_i u_{i,1} \qquad\qquad c_2 = \sum s_i u_{i,2} \qquad\qquad c_3 + K = \tau J + \sum s_i u_{i,3}$$

Let ck be a perfectly binding commitment key.

- The verifier sends a commitment to a random value $k \in \mathbb{Z}_p$ under ck.

- The prover chooses $r_1, r_2, r_3, \psi \leftarrow \mathbb{Z}_p$ and sends

$$R_1 := \sum r_i u_{i,1} \qquad R_2 := \sum r_i u_{i,2} \qquad R_3 := \psi J + \sum r_i u_{i,3}$$

- The verifier opens his commitment to k
- The prover answers $(d_i := r_i - ks_i)_{i=1}^3$ and $\delta := \psi - k\tau$
- The verifier accepts the proof if

$$\sum d_i u_{i,1} = R_1 - kc_1 \qquad \sum d_i u_{i,2} = R_2 - kc_2 \qquad \delta J + \sum d_i u_{i,3} = R_3 - k(c_3 + K)$$

The proof is zero-knowledge. Given an instance (c_1, c_2, c_3), we simulate the prover. Run the protocol by picking random values R_i, and let the verifier open its commitment to k. Now rewind it to the state after committing. Choose random values $(d_1, d_2, d_3, \delta) \in \mathbb{Z}_p$ and send $R_1 := \sum d_i u_{i,1} + kc_1$, $R_2 := \sum d_i u_{i,2} + kc_2$, $R_3 := \delta J + \sum d_i u_{i,3} + k(c_3 + K)$. Finish the protocol by sending (d_1, d_2, d_3, δ).

The proof proves knowledge. To extract knowledge from the prover, we simulate the verifier. We set the commitment key ck to produce equivocable commitments and run the protocol with the prover. After finishing, we rewind the prover to its state after sending the R_i's and now open the commitment to a different value k', whereafter the prover sends values $(d'_1, d'_2, d'_3, \delta')$. Now we have $s_i = \frac{d_i - d'_i}{k' - k}$ for $1 \leq i \leq 3$, $\tau = \frac{\delta - \delta'}{k' - k}$.

Achieving concurrent zero-knowledge. If we use an *extractable* commitment (e.g., by using linear commitments to the bits of k—or sufficiently small blocks of bits) in the first phase of the protocol then we even get *concurrent* zero-knowledge: to simulate a proof we need not rewind the verifier, as we can directly extract the challenge *before* producing the R_i's.

D An Anonymous Proxy Signature Scheme with Delegator Anonymity

We formally describe an instantiation of anonymous proxy signatures with delegator anonymity as discussed in Remark 3 (2).

D.1 Building Blocks

To instantiate APS with delegator anonymity, we will use the following building blocks that were introduced in Sections 2.2 and 4, respectively.

- Linear commitments: $\mathsf{ExSetup}(\cdot)$ takes as input a bilinear group and outputs a commitment key $ck \in \mathbb{G}^5$ and an extraction key $ek \in \mathbb{Z}_p^2$. On inputs a commitment key, a value in $\mathbb{G}$, and randomness from $\mathbb{Z}_p^3$, $\mathsf{Com}(\cdot,\cdot,\cdot)$ outputs a commitment in $\mathbb{G}^3$; $\mathsf{RdCom}(\cdot,\cdot,\cdot)$ takes a commitment key, a commitment and fresh randomness, and outputs a randomized commitment to the same value; $\mathsf{Extr}(\cdot,\cdot)$ outputs the committed value on input ek and a commitment.
- Groth-Sahai proofs: $\mathsf{Prove}(\cdot,\cdot,\cdot)$ produces a proof (in $\mathbb{G}^3$ for linear equations, or $\mathbb{G}^9$ for general equations) on inputs a commitment key, the description of a PPE and a vector of pairs of committed values / randomness; on inputs the commitment key, the equation description, a vector of commitments and a proof, $\mathsf{Verify}(\cdot,\cdot,\cdot)$ outputs a value in $\{0,1\}$; $\mathsf{RdProof}(\cdot,\cdot,\cdot)$ takes as inputs a commitment key, an equation description, a vector of pairs of commitments / fresh randomness and a proof, and outputs a new proof adapted to the randomizations of the commitments.
- Automorphic signatures: let **Sig** $= (\mathsf{Setup}_{\mathrm{sig}}, \mathsf{KeyGen}_{\mathrm{sig}}, \mathsf{Sign}_{\mathrm{sig}}, \mathsf{Verify}_{\mathrm{sig}})$ denote Scheme 2 in Sect. 4. For $vk = (X, Y)$, $m = (M, N)$ and $\sigma = (A, C, D, R, S)$, let $E_{\mathrm{sig}}(vk, m, \sigma)$ denote the equations in (4) and the following two equations: $e(X, H) = e(G, Y)$ and $e(M, H) = e(G, N)$. (We implicitly assume fixed parameters (G, F, H, K, T).) Analogously, let $E'_{\mathrm{sig}}(vk, (m_1, m_2), \sigma)$ be the verification relations for a signature on a message consisting of 2 $\mathcal{DH}$-pairs (cf. Sect. 5.1).

D.2 Instantiation

$\mathsf{Setup}_{\text{aps}}(1^\lambda)$

- Generate a bilinear group $\mathcal{BG}$ for security parameter λ.
- Run $\mathsf{Setup}_{\text{sig}}(\mathcal{BG})$ to get parameters pp_{sig}.
- Run $\mathsf{KeyGen}_{\text{sig}}(pp_{\text{sig}})$ to produce a key pair (ipk, ik). Return the public parameters $pp := (pp_{\text{sig}}, ipk)$ and the issuer's key ik.

$\mathsf{Reg}_{\text{aps}}$ is a protocol between a new user, the issuer and the user's opener.

- The user runs $(vk, sk) \leftarrow \mathsf{KeyGen}_{\text{sig}}(pp_{\text{sig}})$ and produces a signature (possibly via an external PKI[11]) σ_{pki} on vk. She sends $(vk, \sigma_{\text{pki}})$ to the issuer and vk to the opener.
- The issuer checks σ_{pki}, produces $cert \leftarrow \mathsf{Sign}_{\text{sig}}(ik, vk)$, sends $cert$ to the user, and writes $(vk, \sigma_{\text{pki}})$ to its register.
- The opener runs $(ck, ek) \leftarrow \mathsf{ExSetup}(\mathcal{BG})$ and sends ck to the user. It sets the opening key as $ok := (vk, ck, ek)$.
- The user sets his public key $upk = (vk, ck)$ and his secret key $usk = (upk, sk, cert)$.

$\mathsf{Delgt}_{\text{aps}}(usk, [\boldsymbol{warr}], upk)$

- Set $k = 0$ if this is an original delegation (i.e., there is no optional argument $\boldsymbol{warr}$), otherwise let k be s.t. this is the k-th intermediate delegation. Parse usk as $\big((vk_k, ck_k), sk_k, cert_k\big)$ and upk as (vk_{k+1}, ck_{k+1}).
- If $k = 0$ then choose an identifier id, compute $warr_{0\to 1} \leftarrow \mathsf{Sign}_{\text{sig}}\big(sk_0, (\mathsf{Hash}(id \,\|\, 1), vk_1)\big)$ and return $(ck, id, vk_0, warr_{0\to 1})$.
- If $k = 1$ then do the following:
  - Parse $\boldsymbol{warr}$ as $(ck, id, vk_0, warr_{0\to 1})$.
  - Compute $warr_{1\to 2} \leftarrow \mathsf{Sign}_{\text{sig}}\big(sk_1, (\mathsf{Hash}(id \,\|\, 2), vk_2)\big)$.
  - Choose $\rho^{(v)}, \rho^{(c)}, \rho_1^{(w)}, \rho_2^{(w)} \leftarrow \mathbb{Z}_p^3$ and compute the following commitments and proofs:
    $\mathbf{c}_{warr_{0\to 1}} \leftarrow \mathsf{Com}(ck, warr_{0\to 1}, \rho_1^{(w)})$, $\mathbf{c}_{vk_1} \leftarrow \mathsf{Com}(ck, vk_1, \rho^{(v)})$, $\mathbf{c}_{cert_1} \leftarrow \mathsf{Com}(ck, cert_1, \rho^{(c)})$,
    $\mathbf{c}_{warr_{1\to 2}} \leftarrow \mathsf{Com}(ck, warr_{1\to 2}, \rho_2^{(w)})$, $\mathbf{c}_{vk_2} \leftarrow \mathsf{Com}(ck, vk_2, 0)$, (Footnote[12])
    $\phi_{cert_1} \leftarrow \mathsf{Prove}\big(ck, E_{\text{sig}}(ipk, \cdot, \cdot), ((vk_1, \rho_1^{(v)}), (cert_1, \rho^{(c)}))\big)$,
    $\phi_{warr_{0\to 1}} \leftarrow \mathsf{Prove}\big(ck, E'_{\text{sig}}\big(vk_0, (\mathsf{Hash}(id \,\|\, 1), \cdot), \cdot\big), \big((vk_1, \rho^{(v)}), (warr_{0\to 1}, \rho_1^{(w)})\big)\big)$,
    $\phi_{warr_{1\to 2}} \leftarrow \mathsf{Prove}\big(ck, E'_{\text{sig}}\big(\cdot, (\mathsf{Hash}(id \,\|\, 2), \cdot), \cdot\big), \big((vk_1, \rho^{(v)}), (vk_2, 0), (warr_{1\to 2}, \rho_2^{(w)})\big)\big)$.
  - Return $\boldsymbol{warr}' := \big(ck, id, vk_0, (\mathbf{c}_{warr_{0\to 1}}, \phi_{warr_{0\to 1}}, \mathbf{c}_{vk_1}, \mathbf{c}_{cert_1}, \phi_{cert_1}), \mathbf{c}_{warr_{1\to 2}}, \phi_{warr_{1\to 2}}, \mathbf{c}_{vk_2}\big)$.
- Otherwise, do the following:
  - Parse $\boldsymbol{warr}$ as $\big(ck, id, vk_0, (\mathbf{c}_{warr_{(i-1)\to i}}, \phi_{warr_{(i-1)\to i}}, \mathbf{c}_{vk_i}, \mathbf{c}_{cert_i}, \phi_{cert_i})_{i=1}^{k-1},$
    $\mathbf{c}_{warr_{(k-1)\to k}}, \phi_{warr_{(k-1)\to k}}, \mathbf{c}_{vk_k}\big)$.
  - Compute $warr_{k\to(k+1)} \leftarrow \mathsf{Sign}_{\text{sig}}\big(sk_k, (\mathsf{Hash}(id \,\|\, k+1), vk_{k+1})\big)$.
  - Choose randomness for commitments and randomization: Pick $\rho_i^{(v)}, \rho_i^{(c)}, \rho_i^{(w)} \leftarrow \mathbb{Z}_p^3$ for $1 \le i \le k$ and $\rho_{k+1}^{(w)} \leftarrow \mathbb{Z}_p^3$.
  - Randomize the commitments and adapt the proofs in $\boldsymbol{warr}$:
    For $1 \le i \le k$: $\mathbf{c}'_{warr_{(i-1)\to i}} \leftarrow \mathsf{RdCom}(ck, \mathbf{c}_{warr_{(i-1)\to i}}, \rho_i^{(w)})$, $\mathbf{c}'_{vk_i} \leftarrow \mathsf{RdCom}(ck, \mathbf{c}_{vk_i}, \rho_i^{(v)})$,
    $\phi'_{warr_{(i-1)\to i}} \leftarrow \mathsf{RdProof}\big(ck, E'_{\text{sig}}\big(\cdot, (\mathsf{Hash}(id \,\|\, i), \cdot), \cdot\big),$
    $\big((\mathbf{c}_{vk_{i-1}}, \rho_{i-1}^{(v)}), (\mathbf{c}_{vk_i}, \rho_i^{(v)}), (\mathbf{c}_{warr_{(i-1)\to i}}, \rho_i^{(w)})\big), \phi_{warr_{(i-1)\to i}}\big)$.
    For $1 \le i \le k-1$: $\mathbf{c}'_{cert_i} \leftarrow \mathsf{RdCom}(ck, \mathbf{c}_{cert_i}, \rho_i^{(c)})$,
    $\phi'_{cert_i} \leftarrow \mathsf{RdProof}\big(ck, E_{\text{sig}}(ipk, \cdot, \cdot), \big((\mathbf{c}_{vk_i}, \rho_i^{(v)}), (\mathbf{c}_{cert_i}, \rho_i^{(c)})\big), \phi_{cert_i}\big)$.

[11] To achieve strong notions of non-frameability, it is necessary to assume an external PKI infrastructure (cf. [BSZ05])

[12] $\mathbf{c}_{vk_2}$ is thus a *trivial* commitment.

- Compute the following commitments and proofs:
  $\mathbf{c}_{\mathrm{cert}_k} \leftarrow \mathsf{Com}(ck, cert_k, \rho_k^{(c)})$, $\mathbf{c}_{\mathrm{warr}_{k\to(k+1)}} \leftarrow \mathsf{Com}(ck, warr_{k\to(k+1)}, \rho_{k+1}^{(w)})$,
  $\mathbf{c}_{\mathrm{vk}_{k+1}} \leftarrow \mathsf{Com}(ck, vk_{k+1}, 0)$,
  $\phi_{\mathrm{cert}_k} \leftarrow \mathsf{Prove}\big(ck, E_{\mathrm{sig}}(ipk, \cdot, \cdot), ((vk_k, \rho_k^{(v)}), (cert_k, \rho_k^{(c)}))\big)$,
  $\phi_{\mathrm{warr}_{k\to(k+1)}} \leftarrow \mathsf{Prove}\big(ck, E'_{\mathrm{sig}}\big(\cdot, (\mathsf{Hash}(id \,\|\, k+1), \cdot), \cdot\big),$
  $\big((vk_k, \rho_k^{(v)}), (vk_{k+1}, 0), (warr_{k\to(k+1)}, \rho_{k+1}^{(w)})\big)\big)$.
- Return $\boldsymbol{warr}' = \big(ck, id, vk_0, (\mathbf{c}'_{\mathrm{warr}_{(i-1)\to i}}, \phi'_{\mathrm{warr}_{(i-1)\to i}}, \mathbf{c}'_{\mathrm{vk}_i}, \mathbf{c}'_{\mathrm{cert}_i}, \phi'_{\mathrm{cert}_i})_{i=1}^{k-1},$
  $(\mathbf{c}'_{\mathrm{warr}_{(k-1)\to k}}, \phi'_{\mathrm{warr}_{(k-1)\to k}}, \mathbf{c}'_{\mathrm{vk}_k}, \mathbf{c}_{\mathrm{cert}_k}, \phi_{\mathrm{cert}_k}), \mathbf{c}_{\mathrm{warr}_{k\to(k+1)}}, \phi_{\mathrm{warr}_{k\to(k+1)}}, \mathbf{c}_{\mathrm{vk}_{k+1}}\big)$.

$\mathsf{PSign}_{\mathrm{aps}}(usk, \boldsymbol{warr}, msg)$ Signing is done similarly to delegation, where the message now plays the rôle of vk_{k+1}. Since the message is public, there is no need to commit to it; moreover, ck and vk_0 are part of the verification key and need thus not be included in the signature (see (8)).

$\mathsf{Verify}_{\mathrm{aps}}(upk, msg, \Sigma)$

- Parse upk as (vk_0, ck) and parse the signature Σ as

$$\big(id, (\mathbf{c}_{\mathrm{warr}_{(i-1)\to i}}, \phi_{\mathrm{warr}_{(i-1)\to i}}, \mathbf{c}_{\mathrm{vk}_i}, \mathbf{c}_{cert_i}, \phi_{cert_i})_{i=1}^{k}, \mathbf{c}_{sig}, \phi_{sig}\big)\ . \tag{8}$$

- Return 1 if all of the following return 1, otherwise return 0.
  - $\mathsf{Verify}\big(ck, E_{\mathrm{sig}}(ipk, \cdot, \cdot), (\mathbf{c}_{\mathrm{vk}_i}, \mathbf{c}_{cert_i}), \phi_{\mathrm{cert}_i}\big)$, for $1 \le i \le k$;
  - $\mathsf{Verify}\big(ck, E'_{\mathrm{sig}}(\cdot, (\mathsf{Hash}(id \,\|\, i), \cdot), \cdot), (\mathbf{c}_{\mathrm{vk}_{i-1}}, \mathbf{c}_{\mathrm{vk}_i}, \mathbf{c}_{\mathrm{warr}_{(i-1)\to i}}), \phi_{\mathrm{warr}_{(i-1)\to i}}\big)$, for $1 \le i \le k$;
  - $\mathsf{Verify}\big(ck, E'_{\mathrm{sig}}(\cdot, (\mathsf{Hash}(id \,\|\, k+1), msg), \cdot), (\mathbf{c}_{\mathrm{vk}_k}, \mathbf{c}_{sig}), \phi_{sig}\big)$.

$\mathsf{Open}_{\mathrm{aps}}(ok, msg, \Sigma)$ Parse ok as (vk, ck, ek), parse Σ as (8) and check if it is valid. If so then set $vk_i \leftarrow \mathsf{Extr}(ek, \mathbf{c}_{\mathrm{vk}_i})$ and $warr_{(i-1)\to i} \leftarrow \mathsf{Extr}(ek, \mathbf{c}_{\mathrm{warr}_{(i-1)\to i}})$ for $1 \le i \le k$, and $sig \leftarrow \mathsf{Extr}(ek, \mathbf{c}_{sig})$. Return $\big((vk_1, \dots, vk_k), (warr_{0\to 1}, \dots, warr_{(k-1)\to k}, sig)\big)$, where the second component is the proof.