Identity/Attribute-Based Signature Resilient to Hard-to-Invert Leakage under Standard Assumptions

Masahito Ishizaka[1] Kanta Matsuura[1]

Abstract: If a cryptographic scheme is resilient to some leakage, its security is guaranteed to be maintained even if secret information, e.g., the secret-key, is partially leaked. Various security models considering leakage-resilience have been proposed. Hard-to-invert leakage (HL) model, a.k.a. auxiliary (input) leakage model, proposed by Dodis et al. in STOC'09 is especially meaningful since it can deal with leakage caused by a function which information-theoretically determines the secret-key, e.g., a one-way permutation.
In this paper, we generically construct identity-based signature (IBS) and attribute-based signature (ABS) which are adaptively secure in the HL model. By instantiating them, we give the first concrete constructions of IBS and ABS which are adaptively secure in the HL model under standard assumptions, i.e., the decisional linear assumption.

Keywords: Leakage-resilience, Hard-to-invert leakage, Auxiliary (input) leakage, Identity-based signature, Attribute-based signature.

1. Introduction

Identity-Based Signature (IBS). The idea of IBS was presented by Shamir [29]. It is a digital signature where any bit-string (whose bit-length is polynomially bounded) can be used as a verification-key. Since the early years of the current century, it has been known that IBS schemes can be generically constructed from signature schemes. Various direct (or concrete) constructions of IBS were proposed in [28] and the others.

Attribute-Based Signature (ABS). In ABS systems, a set of attributes is assigned to each user. A trusted third authority is assigned a role to generate a secret-key from a set of attributes. A signer specifies a predicate satisfied by her attributes when signing a message. A verifier confirms that the signature was really generated by using a valid secret-key for a set of attributes which satisfies the predicate. ABS schemes were proposed in [24], [30] and the others.

Existential Unforgeability of IBS/ABS. Like digital signature schemes, IBS and ABS schemes are required to be existentially unforgeable. Formal definitions of them can be seen in [28] and [30]. Informally speaking, in case of ABS schemes, an ABS scheme is said to be existentially unforgeable if any PPTA adversary cannot find with a non-negligible probability a pair of a signature σ^*, a message m^* and a predicate ϕ^* which satisfies (at least) a condition where σ^* is a valid signature for (m^*, ϕ^*) even if he can adaptively use a secret-key-revelation oracle which takes a set of attributes, then returns a secret-key for it, and a signature-generation oracle which takes a set of attributes and a predicate, then returns a valid signature for them. Obviously, if the adversary is allowed to query a set of attributes satisfying ϕ^* to the secret-key-revelation oracle, he succeeds to forge a valid signature with probability 1. So, such a query must be prohibited. Thus, in the standard security definition, only PPT adversaries who are not able to get any information from any secret-key for any set of attributes which satisfies the target predicate ϕ^* are considered.

Leakage-Resilient (LR) Cryptography. Leakage-resilience (LR) is a property which guarantees that even if secret information such as secret-keys (for a set of attributes which satisfies the target predicate ϕ^* in case of ABS) is partially leaked, its security is maintained. Any scheme whose security has been proven only in a security model w/o LR is not guaranteed to be secure when such secret information is partially leaked. There are some side-channel attacks which are real threats, e.g., [18], so LR cryptographic schemes are practically more desirable than non-LR one.

In security models considering LR, a side-channel attack caused by an adversary is modeled as a polynomial time computable function[*1] $f : \{0,1\}^{|Secret|} \to \{0,1\}^*$. An adversary is allowed to arbitrarily choose a leakage-function f, query it to a leakage oracle, and learn $f(Secret)$. If we allow the adversary to choose the identity-map as f, the adversary acquires the secret-key entirely and is able to break the security model with probability 1. Hence, we have to impose a restriction on f. Several security models in which different restrictions are imposed on f have been proposed.

In bounded leakage (BL) model [1], the output bit-length of f is restricted. More concretely, only f satisfying $f : \{0,1\}^{|Secret|} \to \{0,1\}^{l(k)}$ such that $l(k) < k$ can be chosen[*2]. To make the output

[1] Institute of Industrial Science, The University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo, 153-8505, Japan.
{ishimasa, kanta}@iis.u-tokyo.ac.jp

[*1] $Secret$ denotes the secret information. $|Secret|$ denotes bit-length of $Secret$.

[*2] k denotes the minimum entropy of a secret-key sk. If a secret-key is gen-

bit-length of f unbounded, noisy leakage (NL) model [26] was invented. In NL model, only $f : \{0,1\}^{|Secret|} \rightarrow \{0,1\}^*$ such that, when we observe $f(Secret)$, the minimum entropy of the secret-key sk drops by at most $l(k) < k$ can be chosen. Any function which information-theoretically reveals the secret-key sk is excluded in each one of the two models. Thus, for instance, one-way permutation cannot be chosen in each one of the models. To remove such a restriction, hard-to-invert leakage (HL) model, a.k.a. auxiliary (input) leakage (AL) model [12], was invented. In HL model, the function f must be a hard-to-invert function. More concretely, only f such that given $f(Secret)$, no PPT algorithm can find sk with a probability larger than $\mu(k)$ can be chosen, where $\mu(\cdot)$ is a negligible function such that $\mu(k) > 2^{-k}$. Note that the larger $\mu(k)$ is, the larger the function class of f is. HL model is a generalization of BL and NL model and has a larger function class. Moreover, HL model is useful in the context of the composition [9], [35].

In each one of the above models, the secret-keys cannot be updated, so each one of the secret-keys leaks its partial information repeatedly throughout the lifetime of the cryptographic system. Meanwhile, in continual leakage (CL) model [5], [10], a situation where the secret-key can be updated periodically is considered. For the secret-key sk_t in time period t, output bit-length of the leakage function of sk_t is bounded like the BL model. The total number of times of secret-key update is unbounded, so total amount of leakage is also unbounded.

A large number of cryptographic schemes with LR were proposed. For instance, public-key encryption [1], [4], [9], [11], identity-based encrypiton [8], [20], [21], [23], [35], attribute-based encrypiton [23], [37], [38], [39], identification [2], [11], and authenticated key agreement [2], [11], were proposed.

Related Works. Digital signature schemes secure in the BL model or the CL model were proposed in [6], [17], [22], [25] and the others.

The first digital signature scheme secure in the HL model was proposed by Faust et al. [13] at Asiacrypt'12. However, their scheme has some disadvantages. Firstly, it is resilient to exponentially hard-to-invert leakage-functions, but not polynomially HL functions. Secondly, it is weakly existentially unforgeable. Recently, Ishizaka and Matsuura [19] proposed a digital signature scheme based on the scheme of Faust et al. Their scheme removed the disadvantages of the scheme in [13], thus it was proved to be resilient to polynomially HL functions and strongly existentially unforgeable. In [32], [36], other digital signature schemes with HL resilience which are secure in a new model named selective auxiliary input model were proposed.

An IBS scheme secure in the BL model was proposed by Wu et al. [34]. However, their scheme is practically undesirable since its existential unforgeability is guaranteed by a security model requiring a strong assumption which is called generic bilinear group model. Thus, no direct IBS constructions secure in the BL model or the HL model under standard (or weak) assumptions have been known. It also has not been known that the famous generic (or indirect) construction of IBS from digital signature effectively works in the BL/HL setting.

As far as we know, no concrete ABS constructions secure in the BL/HL model have been known.

Our Results. We generically construct an IBS scheme existentially unforgeable in the HL model. We also generically construct an ABS scheme whose predicate is represented as a monotone span program existentially unforgeable in the HL model and computationally signer-private. Then, we show that they can be instantiated under standard assumptions such as the decisional linear (DLIN) assumption. Each one of the instantiations of IBS scheme and ABS scheme is not only the first one secure in the HL model under standard assumptions, but also the first leakage-resilient one under standard assumptions.

In this paper, due to the strict page-limitation, we introduce only the result related to the ABS scheme, and omit the result related to the IBS scheme.

Improvements from Our Results [40] at CSS2017. In [40], the same authors proposed a generic construction of predicate signature scheme[*3] secure in the continual auxiliary leakage model or the continual hard-to-invert leakage model. The result gives us generic constructions of IBS and ABS secure in the HL model. However, the IBS and ABS have some undesirable characteristics. Firstly, (it seems that) it is hard for us to instantiate them under standard assumptions. Secondly, (it seems that) it is not easy for us to present any concrete examples of allowed leakage-functions. On the other hand, the IBS and ABS given in this work do not have such undesirable characteristics. Thus, we can instantiate them under standard assumptions such as the DLIN assumption and we can present various concrete examples of functions included in the set of leakage-functions.

Paper Organization. This paper is organized as follows. In Sect. 2, notations used in this paper is explained. Definitions of the DL assumption and the DLIN assumption are also given. Definitions of some cryptographic primitives such as attribute-based signature are also given. In Sect. 3, our generic construction of ABS scheme and proofs for its security, i.e., existential unforgeability in the HL model and computational signer-privacy are described. In Sect. 4, we show that our ABS scheme can be instantiated under the DLIN assumption.

2. Preliminaries

Notations. For $\lambda \in \mathbb{N}$, 1^λ denotes a security parameter. We say that a function $h : \mathbb{N} \rightarrow \mathbb{R}$ is negligible if for every $c \in \mathbb{N}$, there exists $x_0 \in \mathbb{N}$ such that $h(x) \leq x^{-c}$ for every $x \geq x_0$. $\mathcal{G}$ denotes a function which takes 1^λ as input and randomly generates $(p, \mathbb{G}, g)$ and outputs them, where p is a prime number whose bit-size is λ, $\mathbb{G}$ is a multiplicative cyclic group whose order is p, and g is a generator of $\mathbb{G}$. PPTA is an abbreviation of probabilistic polynomial time algorithm.

2.1 Bilinear Groups

$\mathcal{G}_{bg}$ denotes a generator of bilinear groups of prime order. $\mathcal{G}_{bg}$ takes 1^λ, where $\lambda \in \mathbb{N}$, as input, and outputs $(p, \mathbb{G}, \mathbb{G}_T, \hat{e})$, where p is a prime whose bit-length is λ, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups

erated uniformly at random, k is equivalent to bit-length of a secret-key $|sk|$.

[*3] Predicate signature is a generalization of some signature schemes such as digital signature, IBS and ABS

which have order p, and $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a map which is computable in polynomial time and satisfies the following conditions: $\forall g, h \in \mathbb{G}$, $\forall a, b \in \mathbb{Z}_p$, it holds that $\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$. If g is a generator of $\mathbb{G}$, $\hat{e}(g, g)$ becomes a generator of $\mathbb{G}_T$.

2.2 Hardness Assumptions

Discrete Logarithm (DL) Assumption. For $\lambda \in \mathbb{N}$, let $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. DL assumption holds, if for every PPTA $\mathcal{A}$, the probability $\Pr[x \leftarrow \mathcal{A}(p, \mathbb{G}, g, g^x) \mid x \xleftarrow{\mathrm{U}} \mathbb{Z}_p]$ is negligible

Decisional Linear (DLIN) Assumption [3]. For $\lambda \in \mathbb{N}$, let $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. DLIN assumption holds, if for every PPTA $\mathcal{A}$, $|\Pr[1 \leftarrow \mathcal{A}(p, \mathbb{G}, g_1, g_2, g_3, g_1^{r_1}, g_2^{r_2}, g_3^{r_1+r_2}) \mid g_1, g_2, g_3 \xleftarrow{\mathrm{U}} \mathbb{G}, r_1, r_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_p] - \Pr[1 \leftarrow \mathcal{A}(p, \mathbb{G}, g_1, g_2, g_3, g_1^{r_1}, g_2^{r_2}, g_3^{u}) \mid g_1, g_2, g_3 \xleftarrow{\mathrm{U}} \mathbb{G}, r_1, r_2, u \xleftarrow{\mathrm{U}} \mathbb{Z}_p]|$ is negligible.

2.3 Labeled Public Key Encryption

Syntax. Labeled public key encryption (LPKE) consists of three polynomial time algorithms $\{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$. Gen and Enc are probabilistic. Dec is deterministic.

$\mathsf{Gen}(1^\lambda, 1^l) \to (ek, dk)$**.** The key generation algorithm takes 1^λ as input, and outputs an encryption key ek, and a decryption key dk. Plaintext space $\mathcal{M}$, ciphetext space C, label space $\mathcal{L}$ are uniquely determined by ek.

$\mathsf{Enc}(ek, m, L) \to C$**.** The encryption algorithm takes ek, a plaintext $M \in \mathcal{M}$, and a label $L \in \mathcal{L}$ as inputs, and outpus a ciphertext C.

$\mathsf{Dec}(dk, C, L) \to M \mid \perp$**.** The decryption algorithm[*4] takes dk, a ciphetext $C \in C$, and a label $L \in \mathcal{L}$ as inputs, and outputs a plaintext M or $\perp$.

Any LPKE scheme must be correct. An LPKE scheme $\Sigma_{\mathrm{LPKE}} = \{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$ is correct, if $\forall \lambda \in \mathbb{N}$, $\forall (ek, dk) \leftarrow \mathsf{Gen}(1^\lambda)$, $\forall M \in \mathcal{M}$, $\forall L \in \mathcal{L}$, $\forall C \leftarrow \mathsf{Enc}(ek, M, L)$, $M \leftarrow \mathsf{Dec}(dk, C, L)$.

2.3.1 Ciphertext Indistinguishability

We define weak ciphertext-indistinguishability against adaptively chosen label/ciphertexts attacks (IND-wLCCA) for an LPKE scheme $\Sigma_{\mathrm{LPKE}} = \{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$. We use the following game which is played between an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$.

`Key-Generation`. $\mathcal{CH}$ runs $(ek, dk) \leftarrow \mathsf{Gen}(1^\lambda)$, and sends ek to $\mathcal{A}$.

`Query`. $\mathcal{A}$ is allowed to use the decryption oracle **`Dec`** adaptively.

`Dec`(C, L)**:** $\mathcal{A}$ queries a ciphertext $C \in C$ and a label $L \in \mathcal{L}$. $\mathcal{CH}$ returns $M \mid \perp \leftarrow \mathsf{Dec}(dk, C, L)$.

`Challenge`(M_0, M_1, L^*)**.** $\mathcal{A}$ sends two plaintexts $M_0, M_1 \in \mathcal{M}$, and a label $L^* \in \mathcal{L}$. $\mathcal{CH}$ sets $b \xleftarrow{\mathrm{U}} \{0, 1\}$, then returns $C^* \leftarrow \mathsf{Enc}(ek, M_b, L^*)$.

`Query 2.` $\mathcal{A}$ is allowed to use the decryption oracle **`Dec`** adaptively.

`Dec`(C, L)**:** $\mathcal{A}$ queries a ciphertext $C \in C$ and a label $L \in \mathcal{L}$ such that $L \neq L^*$. $\mathcal{CH}$ returns $M \mid \perp \leftarrow \mathsf{Dec}(dk, C, L)$.

`Guess`(b')**.** $\mathcal{A}$ sends $b' \in \{0, 1\}$ to $\mathcal{CH}$.

[*4] Although Dec needs the encryption-key ek as an input since ek includes information such as a prime p, a group $\mathbb{G}$, and etc., we often omit ek as the input.

Definition 1. *LPKE scheme Σ_{LPKE} is IND-wLCCA secure if for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}^{\mathit{IND\text{-}wLCCA}}_{\mathcal{A},\Sigma_{\mathrm{LPKE}}}(\lambda) = |2 \cdot \Pr[b' = b] - 1|$ is negligible.*

2.4 Non-Interactive Zero-Knowledge Proof

Syntax. Non-interactive zero-knowledge proof (NIZK) Σ_{NIZK} for a language L consists of three polynomial time algorithms $\{\mathsf{Gen}, \mathsf{Pro}, \mathsf{Ver}\}$. Gen and Pro are probabilistic and Ver is deterministic. R_L denotes the witness relation.

$\mathsf{Gen}(1^\lambda) \to crs$**.** The key-generation algorithm takes 1^λ as an input, and outputs a common reference string (CRS) crs.

$\mathsf{Pro}(crs, x, w) \to \pi$**.** The proof-generation algorithm takes the CRS crs, a statement x, and a witness w as inputs, and outputs a proof π.

$\mathsf{Ver}(crs, x, \pi) \to 1 \mid 0$**.** The proof-verification algorithm takes the CRS crs, a statement x, and a proof π as inputs, and outputs 1 or 0.

Any NIZK scheme must be correct. An NIZK scheme $\Sigma_{\mathrm{NIZK}} = \{\mathsf{Gen}, \mathsf{Pro}, \mathsf{Ver}\}$ is correct if $\forall \lambda \in \mathbb{N}$, $\forall crs \leftarrow \mathsf{Gen}(1^\lambda)$, $\forall (x, w)$ s.t. $(x, w) \in R_L$, $\forall \pi \leftarrow \mathsf{Pro}(crs, x, w)$, $1 \leftarrow \mathsf{Ver}(crs, x, \pi)$.

Definition 2. *$\Sigma_{\mathrm{NIZK}} = \{\mathsf{Gen}, \mathsf{Pro}, \mathsf{Ver}\}$ is sound if for every $\lambda \in \mathbb{N}$, every $crs \leftarrow \mathsf{Gen}(1^\lambda)$, and every PPT $\mathcal{A}$, $\Pr[\mathcal{A}(crs) \to (x, \pi)$ s.t. $[\mathsf{Ver}(crs, x, \pi) \to 1] \wedge [x \notin L]]$ is negligible.*

Definition 3. *$\Sigma_{\mathrm{NIZK}} = \{\mathsf{Gen}, \mathsf{Pro}, \mathsf{Ver}\}$ is zero-knowledge (ZK) if or every $\lambda \in \mathbb{N}$ and every PPT $\mathcal{A}$, there exists a PPT $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ s.t. $|\Pr[\mathcal{A}^{\mathcal{O}_0^{crs}(x,w)}(crs) \to 1 \mid \mathsf{Gen}(1^\lambda) \to crs] - \Pr[\mathcal{A}^{\mathcal{O}_1^{crs,td}(x,w)}(crs) \to 1 \mid S_1(1^\lambda) \to (crs, td)]|$ is negligible, where $\mathcal{O}_0^{crs}(x, w)$ returns $\mathsf{Pro}(crs, x, w)$ (resp. $\perp$), if $(x, w) \in R_L$ (resp. $(x, w) \notin R_L$), and $\mathcal{O}_1^{crs,td}(x, w)$ returns $\mathcal{S}_2(crs, x, td)$ (resp. $\perp$), if $(x, w) \in R_L$ (resp. $(x, w) \notin R_L$).*

2.5 Attribute-Based Signature (ABS)

Monotone Span Program [24]. For $n \in \mathbb{N}$, $\phi : \{0, 1\}^n \to \{0, 1\}$ denotes a monotone boolean function. A monotone span program for ϕ over a field $\mathbb{F}$ is denoted by $(\mathbf{M}, \xi)$, where $\mathbf{M} \in \mathbb{F}^{k \times t}$ and $\xi : [1, k] \to [0, n-1]$ is a deterministic function which associates each row of $\mathbf{M}$ with an element in $[0, n-1]$. For every $(x_0, \cdots, x_{n-1}) \in \{0, 1\}^n$, the monotone span program $(\mathbf{M}, \xi)$ for ϕ satisfies that $\phi(x_0, \cdots, x_{n-1}) = 1 \iff \exists \vec{v} \in \mathbb{F}^{1 \times k}$ s.t. $[\vec{v}\mathbf{M} = (1\ 0\ \cdots\ 0) \in \mathbb{F}^{1 \times t}] \bigwedge_{i \in [1,k]} [v_i \neq 0 \Rightarrow x_{\xi(i)} = 1]$.

In our ABS scheme, the universal set of attributes is denoted by $\mathcal{U} = \{0, 1\}^l$, where $l \in \mathbb{N}$. A set of attributes is denoted by $\mathbb{S} \in 2^{\mathcal{U}}$, where $2^{\mathcal{U}}$ denotes the super set of $\mathcal{U}$. Let us consider a monotone boolean function $\phi : \{0, 1\}^n \to \{0, 1\}$. We define n as 2^l and $\phi(\mathbb{S})$ as the output of the function ϕ which is given as input $\vec{x}_{\mathbb{S}} = (x_0, \cdots, x_{n-1})$ whose element x_i for the attribute $i \in [0, n-1] (= \mathcal{U})$ is set to 1 if $i \in \mathbb{S}$, and 0 otherwise.

Syntax. Attribute-based signature (ABS) consists of polynomial time algorithms $\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$. Setup, KeyGen and Sig are probabilistic, and Ver is deterministic.

$\mathsf{Setup}(1^\lambda, 1^l) \to (pk, mk)$**.** 1^λ, where $\lambda \in \mathbb{N}$, denotes a security parameter. $l \in \mathbb{N}$ denotes the bit-length of an attribute, and the universal set of attributes is denoted by $\mathcal{U} = \{0, 1\}^l$. The setup algorithm takes 1^λ and 1^l as inputs, and outputs a system public-key pk and a master-key mk. The message space $\mathcal{M}$ is uniquely determined by pk.

$\mathsf{KeyGen}(pk, mk, \mathbb{S} \in 2^{\mathcal{U}}) \rightarrow sk$**.** The key-generation algorithm takes pk, mk and a set of attributes $\mathbb{S}$ as inputs, and outputs a secret-key sk. Secret-key space $\mathcal{K}$ is defined as $\bigcup_{\mathbb{S}\in 2^{\mathcal{U}}} \bigcup_{sk\leftarrow \mathsf{KeyGen}(pk,mk,\mathbb{S})} \{sk\}$.

$\mathsf{Sig}(pk, m, \phi, sk) \rightarrow \sigma$**.** The signing algorithm takes pk, a message $m \in \mathcal{M}$, a predicate ϕ represented as a monotone span program $(\mathbf{M}, \xi)$, and a secret-key sk for a set of attribute $\mathbb{S}$ such that $\phi(\mathbb{S}) = 1$ as inputs, and outputs a signature σ.

$\mathsf{Ver}(pk, m, \phi, \sigma) \rightarrow 1 / 0$**.** The signature-verification algorithm takes pk, a message $m \in \mathcal{M}$, a predicate ϕ and a signature σ as inputs, and outputs 1 or 0.

Any ABS scheme must be correct. An ABS scheme $\Sigma_{\mathrm{ABS}} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$ is correct if $\forall \lambda \in \mathbb{N}$, $\forall l \in \mathbb{N}$, $\forall (pk, sk) \leftarrow \mathsf{Setup}(1^{\lambda}, 1^{l})$, $\forall \mathbb{S} \in 2^{\mathcal{U}}$, $\forall sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{S})$, $\forall m \in \mathcal{M}$, $\forall \phi$ s.t. $\phi(\mathbb{S}) = 1$, $\forall \sigma \leftarrow \mathsf{Sig}(pk, m, \phi, sk)$, $1 \leftarrow \mathsf{Ver}(pk, m, \phi, \sigma)$.

Ishizaka and Matsuura [19] introduced an original primitive named PKX which is related to signature schemes. A PKX scheme consists of an algorithm which generates a pair of public-key and secret-key and two secret-key-verification algorithms. We introduce such a primitive which is related to ABS schemes. An ABX scheme consists of Setup, KeyGen, a secret-key updating algorithm and two secret-key-verification algorithms whose definitions are given below.

$\mathsf{SKUpd}(pk, sk \in \mathcal{K}, \mathbb{S} \in 2^{\mathcal{U}}, \phi) \rightarrow \hat{sk}$**.** The secret-key updating algorithm takes a secret-key $sk \in \mathcal{K}$, a set of attributes $\mathbb{S} \in 2^{\mathcal{U}}$ and a predicate ϕ represented as a MSP $(\mathbf{M}, \xi)$ as inputs and outputs a secret-key $\hat{sk}$. Secret-key space $\hat{\mathcal{K}}$ is defined as $\bigcup_{\mathbb{S}\in 2^{\mathcal{U}}} \bigcup_{sk\leftarrow \mathsf{KeyGen}(pk,mk,\mathbb{S})} \bigcup_{\phi \text{ s.t. } \phi(\mathbb{S})=1} \bigcup_{\hat{sk}\leftarrow \mathsf{SKUpd}(pk,sk,\mathbb{S},\phi)} \{\hat{sk}\}$.

$\mathsf{SKVer}(pk, sk \in \hat{\mathcal{K}}, \phi) \rightarrow 1 / 0$**.** The (first) secret-key-verification algorithm takes a secret-key $sk \in \hat{\mathcal{K}}$ and a predicate ϕ represented as a MSP $(\mathbf{M}, \xi)$ as inputs and outputs 1 or 0.

$\mathsf{SKVer2}(pk, sk, sk') \rightarrow 1 / 0$**.** The second secret-key-verification algorithm takes two secret-keys $sk \in \mathcal{K} \bigcup \hat{\mathcal{K}}$ and $sk' \in \mathcal{K} \bigcup \hat{\mathcal{K}}$ as inputs and outputs 1 or 0.

We require that $\forall \lambda \in \mathbb{N}$, $\forall l \in \mathbb{N}$, $\forall (pk, mk) \leftarrow \mathsf{Setup}(1^{\lambda}, 1^{l})$, $\forall \mathbb{S} \in 2^{\mathcal{U}}$, $\forall sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{S})$, $\forall \phi = (\mathbf{M}, \xi)$ s.t. $\phi(\mathbb{S}) = 1$ and $\forall \hat{sk} \leftarrow \mathsf{SKUpd}(pk, sk, \mathbb{S}, \phi)$, it holds that $[1 \leftarrow \mathsf{SKVer}(pk, \hat{sk}, \phi)] \wedge [1 \leftarrow \mathsf{SKVer2}(pk, sk, sk)] \wedge [1 \leftarrow \mathsf{SKVer2}(pk, sk, \hat{sk})] \wedge [1 \leftarrow \mathsf{SKVer2}(pk, \hat{sk}, \hat{sk})]$.

2.5.1 Unforgeability in the HL Model for ABS

For the existing ABS schemes [24], [27], [30] in non-leakage setting, the authors discussed whether their ABS scheme satisfies weak existential unforgeability under adaptively chosen predicate/messages attack. We define weak existential unforgeability under adaptively chosen predicate/messages attack in the HL model (HL-EUF-CMA) for ABS schemes. We consider the following game for an ABS scheme $\Sigma_{\mathrm{ABS}} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$ which is played by an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$. In the game, $\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$ denotes a set of leakage-functions[*5]. The definition of the set of leakage-functions for our ABS scheme can be seen in Subsect. 3.2.

Setup. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^{\lambda}, 1^{l})$. The universal set of attributes is set as $\mathcal{U} = \{0, 1\}^{l}$. A list $\mathcal{L}_S$ is set as a set $\emptyset$.

Query. $\mathcal{A}$ is allowed to adaptively use secret-key-generation oracle **Generate**, secret-key-revelation oracle **Reveal**, and signature-generation oracle **Sign** as follows.

Generate$(\mathbb{S} \in 2^{\mathcal{U}})$**:** $\mathcal{A}$ issues $\mathbb{S} \in 2^{\mathcal{U}}$. $\mathcal{CH}$ generates $sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{S})$. If a list $\mathcal{L}_{\mathbb{S}}$ for the set of attributes has not been generated, $\mathcal{CH}$ generates it and sets it to $\{sk\}$. Else if such list $\mathcal{L}_{\mathbb{S}}$ has already been generated, $\mathcal{CH}$ sets $\mathcal{L}_{\mathbb{S}} := \mathcal{L}_{\mathbb{S}} \cup \{sk\}$.

Reveal$(\mathbb{S} \in 2^{\mathcal{U}}, i \in \mathbb{N})$**:** $\mathcal{A}$ issues $\mathbb{S} \in 2^{\mathcal{U}}$ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{\mathbb{S}}|]$. $\mathcal{CH}$ retrieves the i-th secret-key from $\mathcal{L}_{\mathbb{S}}$, then returns it.

Sign$(\mathbb{S} \in 2^{\mathcal{U}}, i \in \mathbb{N}, m \in \mathcal{M}, \phi)$**:** $\mathcal{A}$ issues $\mathbb{S} \in 2^{\mathcal{U}}$, $m \in \mathcal{M}$, a predicate ϕ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{\mathbb{S}}|]$. $\mathcal{CH}$ retrieves the i-th secret-key sk from $\mathcal{L}_{\mathbb{S}}$, then generates $\sigma \leftarrow \mathrm{SIG.Sig}(pk, m, \phi, sk)$. After that, $\mathcal{CH}$ returns σ, and sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \phi)\}$.

Leak$(\phi^*, f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}})$**.** $\mathcal{A}$ issues a predicate ϕ^* such that $\phi^*(\mathbb{S}) = 0$ for every set of attributes $\mathbb{S}$ queried to **Reveal** and a function $f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}$. $\mathcal{CH}$ returns $f(\mathcal{L}_{\phi^*})$, where the set $\mathcal{L}_{\phi^*}$ of secret-keys is set to $\bigcup_{\mathbb{S}^*\in 2^{\mathcal{U}} \text{ s.t. } \phi^*(\mathbb{S}^*)=1} \mathcal{L}_{\mathbb{S}^*}$.

Forgery$(m^* \in \mathcal{M}, \sigma^*)$**.** $\mathcal{A}$ sends a message m^* and a signature σ^*. We say that $\mathcal{A}$ wins the game if $[1 \leftarrow \mathsf{Ver}(pk, m^*, \phi^*, \sigma^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S]$. The advantage $\mathtt{Adv}^{\mathcal{F}(\lambda)-HL-EUF-CMA}_{\Sigma_{\mathrm{ABS}},\mathcal{A}}(\lambda)$ is defined as probability $\Pr[\mathcal{A}\ wins.]$.

Definition 4. *Σ_{ABS} is HL-EUF-CMA w.r.t. $\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$, if for every PPT $\mathcal{A}$, $\mathtt{Adv}^{\mathcal{F}(\lambda)-HL-EUF-CMA}_{\Sigma_{\mathrm{ABS}},\mathcal{A}}(\lambda)$ is negligible.*

2.5.2 Computational Signer-Privacy for ABS

For the existing ABS schemes [24], [27], [30], the authors discussed whether their scheme satisfies the information-theoretical signer-privacy, a.k.a. perfect privacy. In this paper, we originally define a computational signer-privacy for ABS schemes and consider whether our scheme satisfies it. For the definition, we referred to the definition of semantic security for PKE scheme given by Goldwasser and Micali [14]. For an ABS scheme $\Sigma_{\mathrm{ABS}} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$, we use two experiments given in Fig.1, where $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ denotes a PPT adversary, $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ denotes a PPT simulator, h_1 and h_2 denote polynomial time computable functions, and $O^{pk,mk}_{CSP}$ denotes an oracle which takes a set of attributes $\mathbb{S} \in 2^{\mathcal{U}}$ as input and returns $\mathsf{KeyGen}(pk, mk, \mathbb{S})$.

Definition 5. *An ABS scheme Σ_{ABS} is computationally signer-private, if $\forall \mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\exists \mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ s.t. $\forall \mathcal{D}$, $\mathtt{Adv}^{CSP}_{\Sigma_{\mathrm{ABS}},\mathcal{D},\mathcal{A},\mathcal{S},h_1,h_2}(\lambda) := |\Pr[\mathcal{D}(\mathtt{Expt}^{CSP-0}_{\Sigma_{\mathrm{ABS}},\mathcal{A},h_1,h_2}(\lambda)) \rightarrow 1] - \Pr[\mathcal{D}(\mathtt{Expt}^{CSP-1}_{\Sigma_{\mathrm{ABS}},\mathcal{S},h_1,h_2}(\lambda)) \rightarrow 1]|$ is negligible.*

2.5.3 Hard-to-Compute-Secret-Key (HtC-SK) for ABX

Ishizaka and Matsuura [19] introduced a property for PKX[*6] named Hard-to-Compute-Secret-Key (HtC-SK). Intuitively, the property says that any PPT given a secret-key sk cannot find a valid secret-key sk' such that a relation does not hold between sk and sk'. We define the property for ABX[*5]. We use the following

[*5] We simply write $\mathcal{F}(\lambda)$ to indicate $\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$ if the set of functions is obviously for Σ_{ABS}.

[*6] For primitives PKX and ABX, see [19], and the syntax of ABS in Sect. 2.5, respectively.

$\mathtt{Expt}^{\mathrm{CSP-0}}_{\Sigma_{\mathrm{ABS}},\mathcal{A},h_1,h_2}(\lambda)$:
$(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^l)$, $(\mathcal{K}^*, \phi^*, m, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_{CSP}^{pk,mk}(\mathbb{S})}(pk)$, where $m \in \mathcal{M}$ and $\mathcal{K}^* = \{\mathbb{S} | \mathbb{S} \in 2^{\mathcal{U}} \wedge \phi^*(\mathbb{S}) = 1\}$.
$\mathbb{S}^* \xleftarrow{\mathrm{U}} \mathcal{K}^*$, $sk^* \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{S}^*)$, $\sigma^* \leftarrow \mathsf{Sig}(pk, m, \phi^*, sk^*)$
$v \leftarrow \mathcal{A}_2^{\mathcal{O}_{CSP}^{pk,mk}(\mathbb{S})}(st, h_1(\mathbb{S}^*), \sigma^*)$.
If $v = h_2(\mathbb{S}^*)$, then $d := 1$. Else, then $d := 0$. Return $(d, \mathcal{K}^*)$.

$\mathtt{Expt}^{\mathrm{CSP-1}}_{\Sigma_{PS},\mathcal{S},h_1,h_2}(\lambda)$:
$(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^l)$, $(\mathcal{K}^*, \phi^*, m, st) \leftarrow \mathcal{S}_1^{\mathcal{O}_{CSP}^{pk,mk}(\mathbb{S})}(pk)$, where $m \in \mathcal{M}$ and $\mathcal{K}^* = \{\mathbb{S} | \mathbb{S} \in 2^{\mathcal{U}} \wedge \phi^*(\mathbb{S}) = 1\}$.
$\mathbb{S}^* \xleftarrow{\mathrm{U}} \mathcal{K}^*$, $sk^* \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{S}^*)$.
$v \leftarrow \mathcal{S}_2^{\mathcal{O}_{CSP}^{pk,mk}(\mathbb{S})}(st, h_1(\mathbb{S}^*))$.
If $v = h_2(\mathbb{S}^*)$, then $d := 1$. Else, then $d := 0$. Return $(d, \mathcal{K}^*)$.

Fig. 1 Experiments $\mathtt{Expt}^{\mathrm{CSP-0}}_{\Sigma_{\mathrm{ABS}}}$ and $\mathtt{Expt}^{\mathrm{CSP-1}}_{\Sigma_{\mathrm{ABS}}}$.

game played by an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$.

Setup. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^l)$. The universal set of attributes is $\mathcal{U} = \{0,1\}^l$.

Query. $\mathcal{A}$ is allowed to adaptively use secret-key-revelation oracle **Reveal** as follows.

Reveal$(\mathbb{S} \in 2^{\mathcal{U}})$**:** $\mathcal{CH}$ generates $sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{S})$, then returns the secret-key to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_{\mathbb{S}} := \mathcal{L}_{\mathbb{S}} \cup \{sk\}$.

Compute$(\phi^*, \mathbb{S}^*, sk^*)$**.** $\mathcal{CH}$ runs $s\hat{k}^* \leftarrow \mathsf{SKUpd}(pk, sk^*, \mathbb{S}^*, \phi^*)$. We say that $\mathcal{A}$ wins the game if $[1 \leftarrow \mathsf{SKVer}(pk, s\hat{k}^*, \phi^*)] \wedge [\bigwedge_{\mathbb{S}' \in 2^{\mathcal{U}}\ s.t.\ \phi^*(\mathbb{S}')=1} [\bigwedge_{sk' \in \mathcal{L}_{\mathbb{S}'}} [0 \leftarrow \mathsf{SKVer2}(pk, s\hat{k}^*, sk')]]]$. The advantage $\mathtt{Adv}^{HtC-SK}_{\Sigma_{\mathrm{ABS}},\mathcal{A}}(\lambda)$ is defined as $\Pr[\mathcal{A}\ wins.]$.

Definition 6. *Σ_{ABX} is HtC-SK, if for every PPT $\mathcal{A}$, $\mathtt{Adv}^{HtC-SK}_{\Sigma_{\mathrm{ABX}},\mathcal{A}}(\lambda)$ is negligible.*

3. Attribute-Based Signature Resilient to Hard-to-Invert Leakage

We generically construct an ABS scheme in Subsect. 3.1. We prove that it is existentially unforgeable in the HL model and computationally signer-private in Subsect. 3.2 and Subsect. 3.3, respectively.

3.1 Construction of ABS

We generically construct an ABS scheme $\Sigma_{\mathrm{ABS}} = \{\mathsf{ABS.Setup}, \mathsf{ABS.KeyGen}, \mathsf{ABS.Sig}, \mathsf{ABS.Ver}\}$ by using the following 3 building blocks: an LPKE scheme $\Sigma_{\mathrm{LPKE}} = \{\mathsf{LPKE.Gen}, \mathsf{LPKE.Enc}, \mathsf{LPKE.Dec}\}$, an NIZK scheme $\Sigma_{\mathrm{NIZK}} = \{\mathsf{NIZK.Gen}, \mathsf{NIZK.Pro}, \mathsf{NIZK.Ver}\}$, and an ABX scheme $\Sigma_{\mathrm{ABX}} = \{\mathsf{ABX.Setup}, \mathsf{ABX.KeyGen}, \mathsf{ABX.SKUpd}, \mathsf{ABX.SKVer}, \mathsf{ABX.SKVer2}\}$. Specifically, each algorithm of Σ_{ABS} is defined as follows.

$\mathsf{ABS.Setup}(1^\lambda, 1^l)$**:** Run $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$ and $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^l)$. Run $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$, where $\mathcal{S}_1$ is the first simulator which makes Σ_{NIZK} satisfy the definition of zero-knowledge.
$\mathcal{M}_L$, $\mathcal{L}_L$ and $\mathcal{C}_L$ denote the plaintext space, the label space and the ciphertext space of Σ_{LPKE}, respectively. $\mathcal{U}_X$ and $\mathcal{P}_X$ denote the universal set of attributes and the predicate space of Σ_{ABX}, respectively. $\mathcal{K}_X$ denotes $\bigcup_{\mathbb{S} \in 2^{\mathcal{U}}} \bigcup_{sk' \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{S})} \{sk'\}$. $\hat{\mathcal{K}}_X$ denotes $\bigcup_{\mathbb{S} \in 2^{\mathcal{U}}} \bigcup_{sk' \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{S})} \bigcup_{\phi\ s.t.\ \phi(\mathbb{S})=1} \bigcup_{s\hat{k}' \leftarrow \mathsf{ABX.SKUpd}(pk', sk', \mathbb{S}, \phi)} \{s\hat{k}'\}$. The universal set of attributes $\mathcal{U}$ and predicate space $\mathcal{P}$ of Σ_{ABS} are equivalent to $\mathcal{U}_X$ and $\mathcal{P}_X$, respectively. The message space $\mathcal{M}$ of Σ_{ABS} is the space satisfying $\mathcal{L}_L = \mathcal{M} \| \mathcal{P}$.
We set system public-key and master-key as $pk := (pk', ek, crs)$ and $mk := mk'$, respectively, and `Return` (pk, mk). We define the language L as $L := \{(C, m, \phi) \in \mathcal{C}_L \times \mathcal{M} \times \mathcal{P}_X \mid \exists s\hat{k}' \in \mathcal{K}_X\ s.t.\ [C \leftarrow \mathsf{LPKE.Enc}(ek, sk', m \| \phi)] \wedge [1 \leftarrow \mathsf{ABX.SKVer}(pk', sk', \phi)]\}$.

$\mathsf{ABS.KeyGen}(pk, mk, \mathbb{S} \in 2^{\mathcal{U}})$**:** mk is written as mk'. `Return` $sk' := \mathsf{ABX.KeyGen}(pk', mk', \mathbb{S})$.

$\mathsf{ABS.Sig}(pk, m \in \mathcal{M}, \phi \in \mathcal{P}, sk, \mathbb{S} \in 2^{\mathcal{U}})$**:** sk is written as sk'. Run $s\hat{k}' \leftarrow \mathsf{ABX.SKUpd}(pk', mk', sk', \mathbb{S}, \phi)$. Generate $C := \mathsf{LPKE.Enc}(ek, s\hat{k}', m \| \phi)$. Set $x := (C, m, \phi)$ and $w := s\hat{k}'$, then generate $\pi := \mathsf{NIZK.Pro}(crs, x, w)$. `Return` $\sigma := (C, \pi)$.

$\mathsf{ABS.Ver}(pk, m \in \mathcal{M}, \phi \in \mathcal{P}, \sigma)$**:** σ is parsed as (C, π). Set $x := (C, m, \phi)$, then `Return` $\mathsf{NIZK.Ver}(crs, x, \pi)$.

3.2 Proof of Unforgeability in the HL Model of ABS

Before we prove the existential unforgeability of our ABS scheme Σ_{ABS}, we define the set of leakage-functions $\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$. In the definition given below, variables $(pk', mk', ek, dk.crs, td)$ denote the variables which were generated at **Setup** in the game, ϕ^* denotes the target predicate, for a set of attributes $\mathbb{S} \in 2^{\mathcal{U}}$, $\mathcal{L}_{\mathbb{S}}$ denotes the list for $\mathbb{S}$ at **Leak** in the game, and a set $\mathcal{L}_{\hat{\mathbb{S}}}$ denotes a multiset which is generated as follows: we initialize $\mathcal{L}_{\hat{\mathbb{S}}}$ as en empty set, and then for every $\mathbb{S} \in 2^{\mathcal{U}}$ s.t. $\phi^*(\mathbb{S}) = 1$, we add $|\mathcal{L}_{\mathbb{S}}|$ number of the set of attributes $\mathbb{S}$ into $\mathcal{L}_{\hat{\mathbb{S}}}$.

Definition 7. *Set of leakage-functions $\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$ consists of every polynomial time computable probabilistic (or deterministic) function $f : \{0,1\}^{\sum_{sk'_{\hat{\mathbb{S}}} \in \mathcal{L}'_{\phi^*}} |sk'_{\hat{\mathbb{S}}}|} \to \{0,1\}^*$ which has a randomness space $\mathcal{R}$ and satisfies that for every PPT $\mathcal{B}$, the probability $\Pr[\mathcal{B}(pk', mk', ek, dk, crs, td, \phi^*, \mathcal{L}_{\hat{\mathbb{S}}}, f, f(\mathcal{L}'_{\phi^*}; r)) \to sk^* \in \hat{\mathcal{K}}_X\ s.t.\ [1 \leftarrow \mathsf{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge [\bigvee_{sk'_{\mathbb{S}^*} \in \mathcal{L}'_{\phi^*}} [1 \leftarrow \mathsf{ABX.SKVer2}(pk', sk^*, sk'_{\mathbb{S}^*})]]]$ is negligible, where $r \xleftarrow{\mathrm{R}} \mathcal{R}$ and for every $\hat{\mathbb{S}} \in \mathcal{L}_{\hat{\mathbb{S}}}$, $sk'_{\hat{\mathbb{S}}} := \mathsf{ABX.KeyGen}(pk, mk, \hat{\mathbb{S}})$ and $\mathcal{L}'_{\phi^*} := \mathcal{L}'_{\phi^*} \cup sk'_{\hat{\mathbb{S}}}$.*

The existential unforgeability of Σ_{ABS} is guaranteed by the following theorem.

Theorem 3.1. *Σ_{ABS} is HL-EUF-CMA w.r.t. $\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$, if Σ_{LPKE} is IND-wLCCA, Σ_{NIZK} is sound and ZK, and Σ_{ABX} is HtC-SK.*

Proof of Theorem 3.1. Hereafter, $q_s \in \mathbb{N}$ denotes the number of times that PPT adversary $\mathcal{A}$ uses the signing oracle **Sign**. To prove Theorem 3.1, we use multiple games $\mathtt{Game}_i$, where $i \in \{0, 1, 2, 3, 4, 4|1, \cdots, 4|q_s, 5\}$.

The game $\mathtt{Game}_0$ is the normal HL-EUF-CMA game w.r.t. Σ_{ABS} and $\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$. Specifically, $\mathtt{Game}_0$ is the following game.

Key-Generation. $\mathcal{CH}$ runs $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^l)$, $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. pk and mk are set to $pk := (pk', ek, crs)$ and $mk := mk'$, respectively. pk is given to $\mathcal{A}$. $\mathcal{L}_S$ is set to $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate**, **Reveal** and **Sign**, $\mathcal{CH}$ behaves as follows.

Generate$(\mathbb{S} \in 2^{\mathcal{U}})$**:** $\mathcal{CH}$ generates $sk' \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{S})$. If the list $\mathcal{L}_{\mathbb{S}}$ for the set of attributes has not been generated, $\mathcal{CH}$ generates it and

sets it to $\{sk'\}$. If the list $\mathcal{L}_{\mathbb{S}}$ has already been generated, $\mathcal{CH}$ sets $\mathcal{L}_{\mathbb{S}} := \mathcal{L}_{\mathbb{S}} \cup \{sk'\}$.

Reveal$(\mathbb{S} \in 2^{\mathcal{U}}, i \in \mathbb{N})$**:** $\mathcal{CH}$ retrieves the i-th secret-key from $\mathcal{L}_{\mathbb{S}}$, then returns the secret-key.

Sign$(\mathbb{S} \in 2^{\mathcal{U}}, i \in \mathbb{N}, m \in \mathcal{M}, \phi)$**:** $\mathcal{CH}$ retrieves the i-th secret-key sk' from $\mathcal{L}_{\mathbb{S}}$. $\mathcal{CH}$ runs $s\hat{k}' \leftarrow \mathsf{ABX.SKUpd}(pk', sk', \mathbb{S}, \phi)$, then generates $C := \mathsf{LPKE.Enc}(ek, s\hat{k}', m||\phi)$. $\mathcal{CH}$ sets $x := (C, m, \phi)$ and $w := sk'$, then generates $\pi := \mathsf{NIZK.Pro}(crs, x, w)$. After that, $\mathcal{CH}$ returns a signature $\sigma := (C, \pi)$ to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \phi)\}$.

Leak$(\phi^* \in \mathcal{P}, f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}})$**.** $\mathcal{CH}$ returns $f(\mathcal{L}_{\phi^*})$, where a set $\mathcal{L}_{\phi^*}$ of secret-keys is set as $\bigcup_{\mathbb{S}^* \in 2^{\mathcal{U}} \ s.t.\ \phi^*(\mathbb{S}^*)=1} \mathcal{L}_{\mathbb{S}^*}$.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{A}$ is said to win the game if $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S]$

We define the other games $\mathtt{Game}_i$, where $i \in \{1, 2, 3, 4, 4|1, \cdots, 4|q_s, 5\}$, as follows.

$\mathtt{Game}_1$ is the same as $\mathtt{Game}_0$ except that $\mathcal{CH}$ generates a common reference string crs by running $crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)$ at **Key-Generation**.

$\mathtt{Game}_2$ is the same as $\mathtt{Game}_1$ except that $\mathcal{A}$'s winning condition is changed to the following one, where $s\hat{k}^* := \mathsf{LPKE.Dec}(dk, C^*, m^*||\phi^*)$: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{ABX.SKVer}(pk', s\hat{k}^*, \phi^*)]$.

$\mathtt{Game}_3$ is the same as $\mathtt{Game}_2$ except that $\mathcal{A}$'s winning condition is changed to the following one: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{ABX.SKVer}(pk', s\hat{k}^*, \phi^*)] \wedge [\bigvee_{\mathbb{S}' \in 2^{\mathcal{U}} \ s.t.\ \phi^*(\mathbb{S}')=1} [\bigvee_{sk' \in \mathcal{L}_{\mathbb{S}'}} [1 \leftarrow \mathsf{ABX.SKVer2}(pk', s\hat{k}^*, sk')]]]$.

$\mathtt{Game}_4 (= \mathtt{Game}_{4|0})$ is the same as $\mathtt{Game}_3$ except that $\mathcal{CH}$ generates a common reference string crs by running $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ at **Key-Generation**. When replying to a query to **Sign** at **Query**, $\mathcal{CH}$ generates a proof π by running $\pi \leftarrow \mathcal{S}_2(crs, x, td)$, where $\mathcal{S}_2$ denotes the second simulator in the definition of zero-knowledge for Σ_{NIZK}.

$\mathtt{Game}_{4|i}$, where $i \in [1, q_s]$, is the same as $\mathtt{Game}_{4|0}$ except that when replying to the j-th signing oracle query, where $j \leq i$, $\mathcal{CH}$ generates the ciphertext C_j by running $C_j \leftarrow \mathsf{LPKE.Enc}(ek, 0^{|sk_\phi|}, m||\phi)$, where $|sk_\phi|$ denotes the bit-length of a secret-key for ϕ.

$\mathtt{Game}_5$ is the following game, which is played by $\mathcal{A}$ and $\mathcal{CH}$.

Key-Generation. $\mathcal{CH}$ runs $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^l)$, $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. $(pk', mk', ek, dk, crs, td)$ are sent to $\mathcal{A}$.

Leak$(\phi^*, \mathbb{S}_1^*, \cdots, \mathbb{S}_k^* \ s.t.\ \phi^*(\mathbb{S}_1^*) = \cdots = \phi^*(\mathbb{S}_k^*) = 1, f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda))$**.** $\mathcal{CH}$ computes $f(\{sk_i^*\}_{i \in [1,k]})$, where for $i \in [1, k]$, the secret-key sk_i^* is generated as $sk_i^* := \mathsf{ABX.KeyGen}(pk', mk', \mathbb{S}_i^*)$. Then $\mathcal{CH}$ sends it to $\mathcal{A}$.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). By decrypting C^*, we get $s\hat{k}^* := \mathsf{LPKE.Dec}(dk, C^*, m^*||\phi^*)$. The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{A}$ is said to win the game if $[1 \leftarrow \mathsf{ABX.SKVer}(pk', s\hat{k}^*, \phi^*)] \wedge [\bigvee_{i \in [1,k]} [1 \leftarrow \mathsf{ABX.SKVer2}(pk', s\hat{k}^*, sk_i^*)]]$.

Hereafter, for $i \in \{0, 1, 2, 3, 4, 4|1, \cdots, 4|q_s, 5\}$, W_i denotes the event where $\mathcal{A}$ wins the game $\mathtt{Game}_i$. Obviously, it holds that

$\mathtt{Expt}_0(\lambda) (= \mathtt{Expt}^{\mathsf{CSP}-0}_{\Sigma_{\mathrm{ABS}}, \mathcal{A}, h_1, h_2}(\lambda))$:
$(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^l)$
$(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$, $pk := (pk', ek, crs)$, $mk := mk'$
$(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{pk,mk}(\mathbb{S})}(pk)$,
  where $\mathcal{K}^* = \{\mathbb{S} | \mathbb{S} \in 2^{\mathcal{U}} \wedge \phi^*(\mathbb{S}) = 1\}$.
$\mathbb{S}^* \xleftarrow{\mathrm{U}} \mathcal{K}^*$, $sk^* \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{S}^*)$
$s\hat{k}^* \leftarrow \mathsf{ABX.SKUpd}(pk', sk^*, \mathbb{S}^*, \phi^*)$
$C^* \leftarrow \mathsf{LPKE.Enc}(ek, s\hat{k}^*, m^*||\phi^*)$, $x^* := (C^*, m^*, \phi^*)$, $w := s\hat{k}^*$
$\pi^* \leftarrow \mathsf{NIZK.Pro}(crs, x^*, w^*)$, $\sigma^* := (C^*, \pi^*)$
$v \leftarrow \mathcal{A}_2^{O_{CSP}^{pk,mk}(\mathbb{S})}(st, h_1(\mathbb{S}^*), \sigma^*)$
If $v = h_2(\mathbb{S}^*)$, then $d := 1$. Else, then $d := 0$. Return $(d, \mathcal{K}^*)$.

$\mathtt{Expt}_6(\lambda) (= \mathtt{Expt}^{\mathsf{CSP}-1}_{\Sigma_{\mathrm{ABS}}, \mathcal{S}, h_1, h_2}(\lambda))$:
$(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^l)$
$(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$, $pk := (pk', ek, crs)$, $mk := mk'$
$(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{S}_1(pk)$,
  where $\mathcal{K}^* = \{\mathbb{S} | \mathbb{S} \in 2^{\mathcal{U}} \wedge \phi^*(\mathbb{S}) = 1\}$.
$\mathbb{S}^* \xleftarrow{\mathrm{U}} \mathcal{K}^*$
$v \leftarrow \mathcal{S}_2(st, h_1(\mathbb{S}^*))$
If $v = h_2(\mathbb{S}^*)$, then $d := 1$. Else, then $d := 0$. Return $(d, \mathcal{K}^*)$.

Fig. 2 Experiments $\mathtt{Expt}_0$ and $\mathtt{Expt}_5$.

$\mathsf{Adv}^{\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)-HL-EUF-CMA}_{\Sigma_{\mathrm{ABS}}, \mathcal{A}}(\lambda) = \Pr[W_0] \leq \sum_{i=1}^4 |\Pr[W_{i-1}] - [W_i]| + \sum_{i=1}^{q_s} |\Pr[W_{4|i-1}] - \Pr[W_{4|i}]| + |\Pr[W_{4|q_s}] - \Pr[W_5]| + \Pr[W_5]$.

Theorem 3.1 is proven by the above inequality and the following lemmas.

Lemma 3.1. $|\Pr[W_0] - \Pr[W_1]|$ *is negligible if* Σ_{NIZK} *is ZK.*

Lemma 3.2. $|\Pr[W_1] - \Pr[W_2]|$ *is negligible if* Σ_{NIZK} *is sound.*

Lemma 3.3. $|\Pr[W_2] - \Pr[W_3]|$ *is negligible if* Σ_{ABX} *is HtC-SK.*

Lemma 3.4. $|\Pr[W_3] - \Pr[W_4]|$ *is negligible if* Σ_{NIZK} *is ZK.*

Lemma 3.5. *For every* $i \in [1, q_s]$, $|\Pr[W_{4|i-1}] - \Pr[W_{4|i}]|$ *is negligible if* Σ_{LPKE} *is IND-wLCCA.*

Lemma 3.6. $\Pr[W_{4|q_s}]$ *is negligible if* $\Pr[W_5]$ *is negligible.*

Lemma 3.7. $\Pr[W_5]$ *is negligible.*

Proof of each lemma is given in the full paper. □

3.3 Proof of Computational Signer-Privacy of ABS

Previous ABS schemes [24], [27], [30] were proven to be perfectly signer-private. On the other hand, it must be hard to prove that our ABS scheme Σ_{ABS} is perfectly signer-private since any signature on a predicate ϕ generated by Σ_{ABS} includes a ciphertext of a secret-key for the predicate. So, we prove the following theorem which guarantees that Σ_{ABS} is computationally signer-private.

Theorem 3.2. *If* Σ_{LPKE} *is IND-wLCCA and* Σ_{NIZK} *is zero-knowledge, then* Σ_{ABS} *is computationally signer-private.*

Proof of Theorem 3.2. We define six experiments $\mathtt{Expt}_i$, where $i \in \{0, 1, 2, 3, 4, 5\}$, to prove the theorem. Hereafter, $\mathcal{A}_1$ and $\mathcal{A}_2$ denote PPT adversaries, $\mathcal{S}_1$ and $\mathcal{S}_2$ denote PPT simulators, $\mathcal{B}_1$ and $\mathcal{B}_2$ denote PPT simulators which makes Σ_{NIZK} satisfy the definition of zero-knowledge, and h_1, h_2 are polynomial time computable functions.

$\mathtt{Expt}_0$, given in Fig.2, is the experiment $\mathtt{Expt}^{\mathsf{CSP}-0}_{\Sigma_{\mathrm{ABS}}, \mathcal{A}, h_1, h_2}(\lambda)$ w.r.t. PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, and functions h_1 and h_2. $\mathtt{Expt}_5$, given in Fig.2, is the experiment $\mathtt{Expt}^{\mathsf{CSP}-1}_{\Sigma_{\mathrm{ABS}}, \mathcal{S}, h_1, h_2}(\lambda)$ w.r.t. PPT simulators $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, and functions h_1 and h_2. Let us consider a PPT distinguisher $\mathcal{D}$, whose advantage is defined as follows. $\mathsf{Adv}^{\mathsf{CSP}}_{\Sigma_{\mathrm{ABS}}, \mathcal{D}, \mathcal{A}, \mathcal{S}, h}(\lambda) := |\Pr[\mathcal{D}(\mathtt{Expt}_0(\lambda)) \rightarrow 1] - \Pr[\mathcal{D}(\mathtt{Expt}_5(\lambda)) \rightarrow 1]|$.

We define the other experiments $\texttt{Expt}_i$, where $i \in \{1,2,3,4\}$. Specifically, each experiment is defined as follows.

$\texttt{Expt}_1$ is the same as $\texttt{Expt}_0$ except that the common reference string crs is generated by running $crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)$.

$\texttt{Expt}_2$ is the same as $\texttt{Expt}_1$ except that the common reference string crs is generated by running $(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$ and the NIZK-proof π^* in the signature $\sigma^* = (C^*, \pi^*)$ is generated by running $\pi^* \leftarrow \mathcal{B}_2(crs, x^*, td)$.

$\texttt{Expt}_3$ is the same as $\texttt{Expt}_2$ except that the LPKE-ciphertext C^* in the signature $\sigma^* = (C^*, \pi^*)$ is generated as a ciphertext of $\hat{sk}^\spadesuit$, where $\hat{sk}^\spadesuit$ is generated as follows: $\mathbb{S}^\spadesuit \xleftarrow{\mathrm{U}} \mathcal{K}^*$, $sk^\spadesuit \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{S}^\spadesuit)$, and then $\hat{sk}^\spadesuit \leftarrow \mathsf{ABX.SKUpd}(pk', sk^\spadesuit, \mathbb{S}^\spadesuit, \phi^*)$.

$\texttt{Expt}_4$ is basically the same as $\texttt{Expt}_3$. In the experiment, we generate not only the pair of (pk, mk), but also another pair of $(\hat{pk}, \hat{mk})$. After generating them, we use only the latter pair. For instance, the public-key $\hat{pk}$ is given to the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, and the key-revelation oracle $O_{CSP}^{\hat{pk},\hat{mk}}(\hat{\mathbb{S}})$ generates a secret-key for a set of attributes $\hat{\mathbb{S}}$ by using the master-key $\hat{mk}$ and returns it. Every variable which is dependent on the key-pair $(\hat{pk}, \hat{mk})$ is given the hat symbol ($\hat{}$).

We obtain the following inequality: $\mathsf{Adv}^{\mathsf{CSP}}_{\Sigma_{\mathsf{ABS}},\mathcal{D},\mathcal{A},\mathcal{S},h_1,h_2}(\lambda) \leq \sum_{i=1}^{5} |\Pr[\mathcal{D}(\texttt{Expt}_{i-1}(\lambda)) \to 1] - \Pr[\mathcal{D}(\texttt{Expt}_i(\lambda)) \to 1]|$.

Theorem 3.2 is proven true by the above inequality and the following 5 lemmas. □

Lemma 3.8. *If Σ_{NIZK} is ZK, then $\forall\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\forall\mathcal{D}$, $|\Pr[\mathcal{D}(\texttt{Expt}_0(\lambda)) \to 1] - \Pr[\mathcal{D}(\texttt{Expt}_1(\lambda)) \to 1]|$ is negligible.*

Lemma 3.9. *If Σ_{NIZK} is ZK, then $\forall\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\forall\mathcal{D}$, $|\Pr[\mathcal{D}(\texttt{Expt}_1(\lambda)) \to 1] - \Pr[\mathcal{D}(\texttt{Expt}_2(\lambda)) \to 1]|$ is negligible.*

Lemma 3.10. *If Σ_{LPKE} is IND-wLCCA secure, then $\forall\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\forall\mathcal{D}$, $|\Pr[\mathcal{D}(\texttt{Expt}_2(\lambda)) \to 1] - \Pr[\mathcal{D}(\texttt{Expt}_3(\lambda)) \to 1]|$ is negligible.*

Lemma 3.11. *$\forall\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\forall\mathcal{D}$, $\Pr[\mathcal{D}(\texttt{Expt}_3(\lambda)) \to 1] = \Pr[\mathcal{D}(\texttt{Expt}_4(\lambda)) \to 1]$.*

Lemma 3.12. *$\forall\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\exists\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ s.t. $\forall\mathcal{D}$, $\Pr[\mathcal{D}(\texttt{Expt}_4(\lambda)) \to 1] = \Pr[\mathcal{D}(\texttt{Expt}_5(\lambda)) \to 1]$.*

Proof of each lemma is given in the full paper.

4. Instantiation under the DLIN assumption

We show that our ABS scheme Σ_{ABS} can be instantiated under standard assumptions, i.e., the DLIN assumption.

As a concrete construction for the LPKE scheme Σ_{LPKE}, we adopt $\Pi_{\mathrm{LPKE},l,n}$ given in Fig. 3. The LPKE scheme is a modified variant of the LPKE scheme by Camenisch et al. [7] which is IND-LCCA secure[*7] under the DLIN assumption and the collision-resistance of hash function. We obtain the following theorem whose proof is given in the full paper.

Theorem 4.1. *For any $l, n \in \mathbb{N}$, $\Pi_{\mathrm{LPKE},l,n}$ is IND-wLCCA under the collision-resistance of the hash function $H_{CL} : \{0,1\}^* \to \mathbb{Z}_p$ and the DLIN assumption.*

As a concrete construction for the non-interactive zero-knowledge proof Σ_{NIZK}, we adopt the Groth-Sahai proof Π_{NIZK} in [16] whose soundness and zero-knowledge are guaranteed under the DLIN assumption.

[*7] IND-LCCA is stronger security notion than IND-wLCCA. For the details, refer to [13].

$\mathsf{LPKE.Gen}(1^\lambda, 1^l, 1^n)$: $(p, \mathbb{G}, g_0) \leftarrow \mathcal{G}(1^\lambda)$. $g_1, g_2 \xleftarrow{\mathrm{U}} \mathbb{G}$
$u_1, u_2, u_3, v_1, v_2, v_3, w_1, w_2, w_3 \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, $d_1 := g_0^{u_1} \cdot g_1^{u_2}, d_2 := g_0^{u_1} \cdot g_2^{u_3}$
$e_1 := g_0^{v_1} \cdot g_1^{v_2}, e_2 := g_0^{v_1} \cdot g_2^{v_3}, h_1 := g_0^{w_1} \cdot g_1^{w_2}, h_2 := g_0^{w_1} \cdot g_2^{w_3}$
$ek := (p, \mathbb{G}, g_0, g_1, g_2, d_1, d_2, e_1, e_2, h_1, h_2)$
$dk := (u_1, u_2, u_3, v_1, v_2, v_3, w_1, w_2, w_3)$, $\texttt{Return}(ek, dk)$

$\mathsf{LPKE.Enc}(ek, x \in \{0,1\}^l, m_1 \in \mathbb{G}, \cdots, m_n \in \mathbb{G}, L \in \{0,1\}^*)$:
For $i \in [1, l]$, the i-th bit of $x \in \{0,1\}^l$ is denoted by x_i.
For every $i \in [1, l]$, do:
$r_i, s_i \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, $\mathbf{y}_i := (y_{i,1}, y_{i,2}, y_{i,3}) := (g_0^{r_i+s_i}, g_1^{r_i}, g_2^{s_i})$
$z_i := h_1^{r_i} \cdot h_2^{s_i} \cdot g_0^{x_i}$, $t_i := H_{CL}(\mathbf{y}_i, z_i, L)$
$c_i := (d_1 \cdot e_1^{t_i})^{r_i} \cdot (d_2 \cdot e_2^{t_i})^{s_i}$, $\mathbf{c}_i := (\mathbf{y}_i, z_i, c_i)$
For every $i \in [1, n]$, do:
$r_{l+i}, s_{l+i} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, $\mathbf{y}_{l+i} := (y_{l+i,1}, y_{l+i,2}, y_{l+i,3}) := (g_0^{r_{l+i}+s_{l+i}}, g_1^{r_{l+i}}, g_2^{s_{l+i}})$
$z_{l+i} := h_1^{r_{l+i}} \cdot h_2^{s_{l+i}} \cdot m_i$, $t_{l+i} := H_{CL}(\mathbf{y}_{l+i}, z_{l+i}, L)$
$c_{l+i} := (d_1 \cdot e_1^{t_{l+i}})^{r_{l+i}} \cdot (d_2 \cdot e_2^{t_{l+i}})^{s_{l+i}}$, $\mathbf{c}_{l+i} := (\mathbf{y}_{l+i}, z_{l+i}, c_{l+i})$
$\texttt{Return}\ \mathbf{C} := \{\mathbf{c}_i\}_{i \in [1, l+n]}$

$\mathsf{LPKE.Dec}(ek, dk, \mathbf{C}, L \in \{0,1\}^*)$:
For every $i \in [1, l]$, do:
$t_i := H_{CL}(\mathbf{y}_i, z_i, L)$, $\tilde{c}_i := y_{i,1}^{u_1+t_iv_1} \cdot y_{i,2}^{u_2+t_iv_2} \cdot y_{i,3}^{u_3+t_iv_3}$
If $\tilde{c}_i \neq c_i$, then return $\perp$.
Else if $z_i/(y_{i,1}^{w_1} \cdot y_{i,2}^{w_2} \cdot y_{i,3}^{w_3}) = g_0$, then $x'_i := 1$. Else, then $x'_i := 0$.
The j-th bit of x' is set as x'_i.
For every $i \in [1, n]$, do:
$t_{l+i} := H_{CL}(\mathbf{y}_{l+i}, z_{l+i}, L)$, $\tilde{c}_{l+i} := y_{l+i,1}^{u_1+t_{l+i}v_1} \cdot y_{l+i,2}^{u_2+t_{l+i}v_2} \cdot y_{l+i,3}^{u_3+t_{l+i}v_3}$
If $\tilde{c}_{l+i} \neq c_{l+i}$, then return $\perp$.
Else, then $m'_i := z_{l+i}/(y_{l+i,1}^{w_1} \cdot y_{l+i,2}^{w_2} \cdot y_{l+i,3}^{w_3})$.
$\texttt{Return}\ (x', m'_1, \cdots, m'_n)$

Fig. 3 Construction of LPKE scheme $\Pi_{\mathrm{LPKE},l,n}$, where $l, n \in \mathbb{N}$ and $H_{CL} : \{0,1\}^* \to \mathbb{Z}_p$ is a collision-resistant hash function.

$\mathsf{ABX.Setup}(1^\lambda, 1^l, 1^n)$: $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. $g_1, \cdots, g_n \xleftarrow{\mathrm{U}} \mathbb{G}$. $\mathcal{U} := \{0,1\}^l$
$\theta \in \mathbb{N}$ denotes bit-size of an element of $\mathbb{G}$. Q denotes signature space.
$\mathcal{K}$ denotes $\bigcup_{\mathbb{S} \in 2^{\mathcal{U}}} \bigcup_{sk \leftarrow \mathsf{ABX.KeyGen}(pk,mk,\mathbb{S})} \{sk\}$.
$\hat{\mathcal{K}}$ denotes $\bigcup_{\mathbb{S} \in 2^{\mathcal{U}}} \bigcup_{sk \leftarrow \mathsf{ABX.KeyGen}(pk,mk,\mathbb{S})} \bigcup_{\phi\ s.t.\ \phi(\mathbb{S})=1}$
$\bigcup_{\hat{sk} \leftarrow \mathsf{ABX.SKUpd}(pk,sk,\mathbb{S},\phi)} \{\hat{sk}\}$. $(vk, mk) \leftarrow \mathsf{SIG.Gen}(1^\lambda, 1^{l+\theta})$.
$\texttt{Return}\ pk := (vk, \mathcal{U}, p, \mathbb{G}, g_1, \cdots, g_n)$ and mk.

$\mathsf{ABX.KeyGen}(pk, mk, \mathbb{S} \in 2^{\mathcal{U}})$:
$\mathbb{S}$ is parsed as $\{a | a \in \mathcal{U}\}$. $x_1, \cdots, x_n \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. $y := \prod_{i=1}^n g_i^{x_i}$.
For every $a \in \mathbb{S}$, $\hat{\sigma}_a \leftarrow \mathsf{SIG.Sig}(vk, mk, y \| a)$.
$\texttt{Return}\ sk := (x_1, \cdots, x_n, y, \{\hat{\sigma}_a \mid a \in \mathbb{S}\})$.

$\mathsf{ABX.SKUpd}(pk, sk \in \mathcal{K}, \mathbb{S} \in 2^{\mathcal{U}}, \phi)$:
sk is parsed as $(x_1, \cdots, x_n, y, \{\hat{\sigma}_a \mid a \in \mathbb{S}\})$.
ϕ is represented as $(\mathbf{M}, \xi)$, where $\mathbf{M} \in \mathbb{F}^{k \times t}$ and $\xi : [1, k] \to \mathcal{U}$.
$\hat{\mathbb{S}} \in 2^{\mathcal{U}}$ is defined as $\bigcup_{i \in [1,k]} \{\xi(i)\}$. For every $a \in \hat{\mathbb{S}} \setminus \mathbb{S}$, $\hat{\sigma}_a \xleftarrow{\mathrm{U}} Q$.
$\texttt{Return}\ \hat{sk} := (x_1, \cdots, x_n, y, \{\hat{\sigma}_a \mid a \in \hat{\mathbb{S}}\})$.

$\mathsf{ABX.SKVer}(pk, \hat{sk} \in \hat{\mathcal{K}}, \phi)$:
$\hat{sk}$ is for $\mathbb{S} \in 2^{\mathcal{U}}$ and is parsed as $(x_1, \cdots, x_n, y, \{\hat{\sigma}_a \mid a \in \mathbb{S}\})$.
ϕ is represented as $(\mathbf{M}, \xi)$, where $\mathbf{M} \in \mathbb{F}^{k \times t}$ and $\xi : [1, k] \to \mathcal{U}$.
$\texttt{Return}\ 1$, if $[y = \prod_{i=1}^n g_i^{x_i}] \wedge [\exists \vec{v} \in \mathbb{F}^{1 \times k}$ s.t. $[\vec{v}\mathbf{M} = (1\ 0\ \cdots\ 0) \in \mathbb{F}^{1 \times t}]$
$\bigwedge_{i \in [1,k]\ s.t.\ v_i \neq 0} [1 \leftarrow \mathsf{SIG.Ver}(vk, \hat{\sigma}_{\xi(i)}, y \| \xi(i))]\ ]$.
$\texttt{Return}\ 0$, otherwise.

$\mathsf{ABX.SKVer2}(pk, sk^*, sk')$:
sk^* is for $\mathbb{S}^* \in 2^{\mathcal{U}}$ and is parsed as $(x_1^*, \cdots, x_n^*, y^*, \{\hat{\sigma}_a^* \mid a \in \mathbb{S}^*\})$
sk' is for $\mathbb{S}' \in 2^{\mathcal{U}}$ and is parsed as $(x'_1, \cdots, x'_n, y', \{\hat{\sigma}'_b \mid b \in \mathbb{S}'\})$
$\texttt{Return}\ 1$, if $\bigwedge_{i=1}^n \left[x_i^* = x'_i\right]$. $\texttt{Return}\ 0$, otherwise.

Fig. 4 Construction of ABX scheme $\Pi_{\mathrm{ABX},l,n}$, where $l, n \in \mathbb{N}$ and $\Sigma_{\mathrm{SIG},l+\theta} = \{\mathsf{SIG.Gen}, \mathsf{SIG.Sig}, \mathsf{SIG.Ver}\}$ is a signature scheme with message space $\{0,1\}^{l+\theta}$.

As a concrete construction for the ABX scheme Σ_{ABX}, we adopt the ABX scheme $\Pi_{\mathrm{ABX},l,n}$ given in Fig. 4. For $\Pi_{\mathrm{ABS},l,n}$, we obtain Theorem 4.2 whose proof is given in the full paper.

Theorem 4.2. *For any $l, n \in \mathbb{N}$ and any EUF-CMA secure signa-*

$\underline{\mathsf{SIG.Gen}(1^\lambda, 1^n)}$: $(p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g) \leftarrow \mathcal{G}_{bg}(1^\lambda)$, $\mathcal{M} := \{0,1\}^n$
$g, h, V_0 \xleftarrow{\mathrm{U}} \mathbb{G}, a, v_1, \cdots, v_n \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, $A := h^a$. For $i \in [1, n]$, $V_i := g^{v_i}$.
Return $(vk, sk) := ((p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, h, A, V_0, V_1, \cdots, V_n), g^a)$

$\underline{\mathsf{SIG.Sig}(vk, sk, m \in \mathcal{M})}$: $r \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, $\sigma_1 := (V_0 \prod_{i=1}^n V_i^{m_i})^r \cdot g^a$, $\sigma_2 := h^r$
Return $\sigma := (\sigma_1, \sigma_2)$

$\underline{\mathsf{SIG.Ver}(vk, \sigma, m \in \mathcal{M})}$:
Return 1 if $\hat{e}(V_0 \prod_{i=1}^n V_i^{m_i}, \sigma_2)\hat{e}(g, A) = \hat{e}(\sigma_1, h)$. Return 0, otherwise.

Fig. 5 Construction of signature scheme $\bar{\Pi}_{\mathrm{SIG},n}$, where $n \in \mathbb{N}$.

ture scheme $\Sigma_{\mathrm{SIG},l+\theta}$, $\Pi_{\mathrm{ABX},l,n}$ *is HtC-SK under the DL assumption and the EUF-CMA security of* $\Sigma_{\mathrm{SIG},l+\theta}$.

We consider the signature scheme $\bar{\Pi}_{\mathrm{SIG},n}$ given in Fig. 5. Actually, it is the signature scheme proposed by Waters [33]. Waters proved that it is strongly EUF-CMA secure under the bilinear diffie-hellman assumption which is implied by the DLIN assumption. Thus,

Theorem 4.3. *For any* $n \in \mathbb{N}$, $\Pi_{\mathrm{SIG},n}$ *is EUF-CMA under the DLIN assumption.*

Let $\bar{\Pi}_{\mathrm{ABX},l,n}$ denote the ABX scheme $\Pi_{\mathrm{ABX},l,n}$ using the signature scheme $\bar{\Pi}_{\mathrm{SIG},l+\theta}$ as $\Sigma_{\mathrm{SIG},l+\theta}$. By Theorem 4.2 and Theorem 4.3, we obtain

Corollary 4.1. *For any* $l, n \in \mathbb{N}$, $\bar{\Pi}_{\mathrm{ABX},l,n}$ *is HtC-SK under the DLIN assumption.*

Acknowledgements

This work was supported by JSPS KAKENHI (Grant Number JP17KT0081).

References

[1] Akavia, A., Goldwasser, S., Vaikuntanathan, V. :Simultaneous hardcore bits and cryptography against memory attacks. In: TCC 2009, LNCS 5444, pp. 474-495, 2009.
[2] Alwen, J., Dodis, Y., Wichs, D. :Leakage-resilient public-key cryptography in the bounded-retrieval model. In: CRYPTO 2009, LNCS 5677, pp. 36-54, 2009.
[3] Boneh, D., Boyen, X., Shacham, H. :Short group signatures. In: CRYPTO 2004, LNCS 3152, pp.41-55, 2004.
[4] Brakerski, Z., Goldwasser, S. :Circular and leakage resilient public-key encryption under subgroup indistinguishability (or: quadratic residuosity strikes back). In: CRYPTO 2010, LNCS 6223, pp. 1-10, 2010.
[5] Brakerski, Z., Kalai, Y.T., Katz, J., Vaikuntanathan, V. :Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In: FOCS 2010, pp. 501-510, 2010.
[6] Boyle, E., Segev, G., Wichs, D. :Fully leakage-resilient signatures. In: EUROCRYPT 2011, LNCS 6632, pp. 89-108, 2011.
[7] Camenisch, J., Chandran, N., Shoup, V. :A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In: EUROCRYPT 2009, LNCS 5479, pp. 351-368, 2009.
[8] Chow, S.S.M, Dodis, Y., Rouselakis, Y., Waters, B. :Practical leakage-resilient identity-based encryption from simple assumptions. In: ACMCCS 2010, pp. 152-161, 2010.
[9] Dodis, Y., Goldwasser, S., Kalai, Y.T., Peikert, C., Vaikuntanathan, V. :Public-key encryption with auxiliary inputs. In: TCC 2010, LNCS 5978, pp. 361-381, 2010.
[10] Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D. :Cryptography against continuous memory attacks. In: FOCS 2010, pp. 511-520, 2010.
[11] Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D. :Efficient public-key cryptography in the presence of key leakage. In: ASIACRYPT 2010, LNCS 6477, pp. 613-631, 2010.
[12] Dodis, Y., Kalai, Y.T., Lovett, S. :On Cryptography with auxiliary input. In: STOC 2009, pp. 621-630, 2009.
[13] Faust, S., Haway, C., Nielsen, J.B., Nordholt, P.S., Zottarel, A. :Signature schemes secure against hard-to-invert leakage. In: ASIACRYPT 2012, LNCS 7658, pp.98-115, 2012.
[14] Goldwasser, M., Micali, S. :Probabilistic encryption. In: Journal of Computer and System Sciences, Vol. 28, Issue 2, pp. 270-299, 1984.
[15] Gentry, C., Peikert, C., Vaikuntanathan, V. :Trapdoors for hard lattices and new cryptographic constructions. In: STOC 2008, pp. 197-206, 2008.
[16] Groth, J., Sahai, A. :Efficient non-interactive proof systems for bilinear groups. In: EUROCRYPT 2008, LNCS 4965, pp. 415-432, 2008.
[17] Galindo, D., Vivek, S. :A practical leakage-resilient signature scheme in the generic group model. In: SAC 2012, LNCS 7707, pp. 50-65, 2012.
[18] Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W. :Lest we remember: cold boot attacks on encryption keys. In: USENIX Security Symposium, pp. 45-60, 2008.
[19] Ishizaka, M., Matsuura, K.: Strongly unforgeable signature resilient to polynomially hard-to-invert leakage under standard assumptions. In: ISC 2018, LNCS 11060, pp. ???-???, 2018. (To appear)
[20] Ishizaka, M., Matsuura, K.: Identity-based encryption resilient to the auxiliary leakage under the decisional linear assumption. In: CANS 2018, LNCS 11124, pp. ???-???, 2018. (To appear)
[21] Kurosawa, K., Phong, L.T. :Leakage resilient IBE and IPE under the DLIN assumption. In: ACNS 2013, LNCS 7954, pp. 487-501, 2013.
[22] Katz, J., Vaikuntanathan, V. :Signature schemes with bounded leakage resilience. In: ASIACRYPT 2009, LNCS 5912, pp. 703-720, 2009.
[23] Lewko, A., Rouselakis, Y., Waters, B. :Achieving leakage resilience through dual system encryption. In: TCC 2011, LNCS 6597, pp. 70-88, 2011.
[24] Maji, H.K., Prabhakaran, M., Rosulek, M. :Attribute-based signatures. In: CT-RSA 2011, LNCS 6558, pp. 376-392, 2011.
[25] Malkin, T., Teranishi, I., Vahlis, Y., Yung, M. :Signatures resilient to continual leakage on memory and computation. In: TCC 2011, LNCS 6597, pp. 89-108, 2011.
[26] Naor, M., Segev, G. :Public-key cryptosystems resilient to key leakage. In: CRYPTO 2009, LNCS 5677, pp. 18-35, 2009.
[27] Okamoto, T., Takashima, K. :Efficient attribute-based signatures for non-monotone predicates in the standard model. In: PKC 2011, LNCS 6571, pp. 35-52, 2011.
[28] Paterson, K.G., Schuldt, J.C.N. :Efficient identity-based signatures secure in the standard model. In: ACISP 2006, LNCS 4058, pp. 207-222, 2006.
[29] Shamir, A. :Identity-based cryptosystems and signature schemes. In: CRYPTO 1984, LNCS 196, pp. 47-53, 1984.
[30] Sakai, Y., Attrapadung, N., Hanaoka, G. :Attribute-based signatures for circuits from bilinear map. In: PKC 2016, LNCS 9614, pp.283-300, 2016.
[31] Steinfeld, R., Pieprzyk, J., Wang, H. :How to strengthen any weakly unforgeable signature into a strongly unforgeable signature. In: CT-RSA 2007, LNCS 4377, pp. 357-371, 2007.
[32] Wang, Y., Matsuda, T., Hanaoka, G., Tanaka, K. :Signatures resilient to uninvertible leakage. In: SCN 2016, LNCS 9841, pp. 372-390, 2016.
[33] Waters, B. :Efficient identity-based encryption without random oracles. In: EUROCRYPT 2005, LNCS 3494, pp.114-117, 2005.
[34] Wu, J.D., Tseng, Y.M., Huang, S.S. :Leakage-resilient id-based signature scheme in the generic bilinear group model. In: Security Comm. Networks, Vol. 9, Issue 17, pp. 3987-4001, 2016.
[35] Yuen, T.H., Chow, S.S.M, Zhang, Y., Yiu, S.M. :Identity-based encryption resilient to continual auxiliary leakage. In: EUROCRYPT 2012, LNCS 7232, pp. 117-134, 2012.
[36] Yuen, T.H., Yiu, S.M., Hui, L.C.K. :Fully leakage-resilient signatures with auxiliary inputs. In: ACISP 2012, LNCS 7372, pp. 294-307, 2012.
[37] Zhang, M. :New model and construction of ABE: achieving key resilient-leakage and attribute direct-revocation. In: ACISP 2014, LNCS 8544, pp. 192-208, 2014.
[38] Zhang, M., Shi, W., Wang, C., Chen, Z., Mu, Y. :Leakage-resilient attribute-based encryption with fast decryption: models, analysis and constructions. In: ISPEC 2013, LNCS 7863, pp. 75-90, 2013.
[39] Zhang, M., Wang, C., Takagi, T., Mu, Y.: Functional encryption resilient to hard-to-invert leakage. In: The Computer Journal, doi:10.1093/comjnl/bxt105, 2013.
[40] 石坂, 松浦. :Continual Auxiliary Leakage に耐性を持つ適応的安全な述語署名. In: CSS2017. (In Japanese)