Attribute Based Key-Insulated Signatures with Message Recovery

Y. Sreenivasa Rao(✉) and Ratna Dutta

Indian Institute of Technology Kharagpur,
Kharagpur 721302, India
{ysrao,ratna}@maths.iitkgp.ernet.in

Abstract. In order to minimize the impact of secret signing key exposure in attribute based signature scenario, we design two attribute based key-insulated signature (ABKIS) with *message recovery* schemes for *expressive* linear secret-sharing scheme (LSSS)-realizable access structures utilizing only *4 bilinear pairing operations* in verification process and making the *message-signature length constant.* The first scheme deals with small universes of attributes while the second construction supports large universe of attributes. The signing key is computed according to LSSS access structure over signer's attributes, and is later updated at discrete time periods with the help of a physically secure but computationally limited device, called helper, without changing the access structure. A signing key for some time period is used to sign every message during that time period. The original message is not required to be transmitted with the signature, however, it can be recovered during verification procedure. The size of signing key in the proposed schemes is quadratic in number of attributes. The (strong) key-insulated security of our ABKIS primitives is reduced to the classical computational Diffie Hellman Exponent problem in selective attribute set and random oracle model. We also show that both the proposed signature constructions provide signer privacy.

Keywords: Attribute based signature · Key-insulation · Message recovery · Linear secret-sharing scheme · Constant message-signature length

1 Introduction

There exist two types of digital signature designs: (1) signature schemes in which the original message of the signature is required to be transmitted together with the signature, and (2) signature schemes with message recovery in which the message is embedded in a signature and can be recovered from the signature during verification process. The latter approach reduces the total size of the original message and the appended signature, hence interesting for applications where bandwidth is at prime concern. Message recovery signatures are also appropriate for applications in which small messages should be signed.

Attribute Based Signature (ABS) schemes are broadly categorized as (i) *key-policy* ABS [6,12]: the signing key is associated with an access structure and a message is signed with an attribute set satisfying the access structure, and (ii) *signature-policy* ABS [10,11]: the signing key is tagged with an attribute set and a message is signed with an access structure (or predicate) satisfied by the attribute set.

Maji *et al.* [10,11] proposed signature-policy ABS schemes with formal security definitions of existential unforgeability and signer privacy. The signature leaks no other attribute information about the original signer except the fact that a single user with some set of attributes satisfying the predicate has generated the signature. A formal definition and security model for threshold key-policy ABS are presented in [6,12] together with schemes that support small as well as large attribute universe. In small universe schemes, the set of attributes used in the system is fixed during system setup. On the contrary, in large universe constructions, the size of the attribute universe can be exponentially large in the security parameter as the attribute parameters are dynamically computed after the system setup. The *signer privacy* for key-policy ABS of [6,12] ensures that the signature of a message for the attribute set W reveals nothing about the access structure $\mathbb{A}$ of the signer except the fact that $\mathbb{A}$ is satisfied by W. Specifically, even a computationally unbounded adversary cannot recognize the access structure or the signing key used to compute the given signature. ABS has found several applications like attribute based messaging, attribute based authentication and trust-negotiation, leaking secrets, etc.

The protection of secret key (*decryption key* in case of encryption whereas *signing key* in case of digital signature) is pivotal to all security guarantees in any cryptosystem. Standard notions of security cannot protect the system once its secret key gets compromised. In practice, we cannot assume the secret keys are kept perfectly secure at all times. For instance, an adversary might learn secret information from a physically compromised device or from a malicious user. A variety of frameworks have been proposed in an attempt to mitigate the potential damages caused by secret key exposure, including forward security [2], key-insulated cryptography [5] and intrusion resilience [7]. We focus here on key-insulation mechanism in the context of digital signature design. In this framework, a physically secure but computationally limited device, called *helper*, is involved. Initially, the key generation authority creates a helper key and initial signing key for every user. The helper key is stored in helper whereas the initial signing key is issued to respective user. The lifetime of the system is divided into several discrete time periods. The user can update his initial signing key at regular time periods with the help of helper without further access to the key generation authority. This greatly reduces unnecessary burden on the authority. The updated signing keys are called temporary signing keys. In every time period, the user can obtain the temporary signing key for the current time period by combining the temporary signing key of the previous time period with the partial key returned by the helper. The partial key is computed using helper key. Note also that any temporary signing key can be updated to either future time periods or past time periods according to user choice. The public key remains same throughout the lifetime. A temporary signing key for some time period is used to sign every message during that time period. The signatures are labeled with the time period during which they were generated. The exposure of signing keys for some time periods do not enable an adversary to create signing keys for non-exposed time periods as long as the helper is not compromised (it is assumed that helper is physically secure). Thus this framework can minimize the damage caused by signing key exposure.

Recently, Chen *et al.* [4] added key-insulation mechanism to the key-policy ABS of [9] and constructed an *attribute based key-insulated signature* (ABKIS) scheme in the key-policy setting. The ordinary ABS schemes are not appropriate for certain applications. For instance, consider a scenario in which Alice, who is a bank manager, wishes to take a vacation and wants to delegate her subordinate Bob to complete her daily business. Clearly, the mere ABS schemes are not enough to meet such requirements. Using ABKIS frame work, Alice can compute the temporary signing keys correspond to the time periods of her vacation and send those keys to Bob. Consequently, Bob can sign on behalf of Alice during the designated time periods.

There are several limitations of [4]: (a) The scheme handles only threshold policies, i.e., the signing key is computed according to threshold policy. The signing and verification attribute sets are essentially the same, i.e., the scheme reveals the attributes that the signer used to generate the signature. This may not desirable for certain applications like secret-leaking environments [11]. However, the scheme preserves signer privacy, a legitimate signer is indistinguishable among all the users whose access structures accepting the attribute set specified in the signature verification. (b) The size of the signature and the number of pairings required in verification are proportional to the number of underlying attributes. Specifically, the signature includes 4 group elements for each involved attribute and verification process requires 4 pairing operations for each required attribute, which could be prohibitively costly. (c) The scheme works for small universes of attributes. We address each of these limitations in this work.

Wang *et al.* [13] proposed a threshold policy ABS scheme with message recovery based on [8] and an identity based message recovery signature [15]. The size of the signature and the number of pairings during verification grow linearly with the number of involved attributes, in [13]. There is no ABKIS with message recovery proposed so far. Note that one can extend the ABKIS of [4] to support message recovery mechanism using the framework of [15], but the above mentioned limitations (a), (b) and (c) remain unchanged. To the best of our knowledge, [4] is the only ABKIS scheme (without message recovery) available in the literature.

1.1 Our Contribution

In this work, we consider designing key-insulated signatures with message recovery in the attribute based setting to achieve the following three main goals (i) *secret key security:* to mitigate the damage caused by the signing key exposure, (ii) *communication efficiency:* to realize constant message-signature size and (iii) *expressive access policies:* to exploit as expressive access structures as possible.

Table 1. Comparative summary of ABKIS schemes.

Scheme	Signature Size	Verification cost		Access structure	Security	Assumption	Attribute Universe
		Exp	Pairings				
[4]	$4 \cdot \delta$	δ	$4 \cdot \delta$	threshold policy	selective	cDH	small
SU-ABKIS	4	-	4	LSSS-realizable	selective	cDHE	small
LU-ABKIS	4	δ	4	LSSS-realizable	selective	cDHE	large

We propose the *first* attribute based key-insulated signature, ABKIS, schemes with *message recovery* for *expressive* linear secret-sharing scheme (LSSS)-realizable access structures in the key-policy framework. Our designs feature constant number of pairing computations for signature verification and constant message-signature length. In these constructions, the signing key is computed according to LSSS-realizable access structure over signer's attributes. We present two variants of these schemes, the first for small attribute universe and the second for large universe setting.

Small Universe Construction. We construct an ABKIS scheme with message recovery for small universes of attributes, referred as SU-ABKIS, by adapting the message recovery technique of [15], in which the whole message is embedded in a signature and it can be recovered by the verification algorithm. The signer can update his signing key to any time period without help of the authority and without changing his access structure by interacting with a helper device (it holds another secret key called helper key which is user specific). The signature consists of 4 group elements and a fixed length string (the string is appeared due to message recovery mechanism), and the signature verification can be done within 4 pairing executions. These measures are independent of the number of required attributes.

Large Universe Construction. We also present another ABKIS scheme with message recovery that can deal with large universe of attributes, referred as LU-ABKIS. The LU-ABKIS preserves the same functionality as that of SU-ABKIS except the size of public parameters is linear to the maximum number of attributes used to sign a message, as opposed to the size of attribute universe in small universe construction. And, verification needs δ exponentiations in addition to 4 pairings, δ is the number of attributes used in signing process. In our LU-ABKIS, the actual attribute set used to sign a message is hidden inside a larger attribute set and the verifier is provided with the latter one to verify the signature. As a result, the verifier cannot recognize the actual set of attributes utilized in the signing process (see *Remark* 6 in Sect. 3.1 for details). In contrast, the ABKIS of [4] needs to reveal the original signing attribute set to verify the signature. This means the signing and verification attribute sets are equal. In sum, our LU-ABKIS construction overcomes all the limitations (a), (b) and (c) of [4] mentioned in the previous section (see Table 1).

Both the proposed signature schemes protect signer privacy in the sense that the distribution of the signature is independent of the signing key used to compute it and hence no one can get any information about the signer's access structure or attributes involved in the generation of a signature. Consequently, a legitimate signer is indistinguishable among all the users whose access structures accepting the attribute set specified in the signature verification. The signing key size in both the proposed constructions is quadratic in number of involved attributes. However, the number of required bilinear pairing evaluations is independent of this size. The key-insulated and strong key-insulated security of our ABKIS schemes are analyzed in selective attribute set and random oracle model under the computational Diffie Hellman Exponent (cDHE) assumption.

Table 1 presents comparative summary of our schemes against ABKIS (without message recovery) scheme [4].

2 Preliminaries

We use the following notations in the rest of the paper.

$x\|\|y$	: concatenation of two strings x and y
$\oplus$	: X-OR operation in binary system
$^{\ell}[\![y]\!]$	: first ℓ bits from left
$[\![y]\!]^{\ell}$	: first ℓ bits from right
$[n]$	: $\{1, 2, \dots, n\}$, for any positive integer n
$s \leftarrow_R S$	: operation of picking an element s uniformly at random from the set S
$\vec{a}\vec{b}$	: $a_1b_1 + \cdots + a_jb_j$, if $\vec{a} = (a_1, \dots, a_j), \vec{b} = (b_1, \dots, b_j) \in \mathbb{Z}_p^j$
$r\vec{a}$	: $(ra_1, ra_2, \dots, ra_j)$, where $r \in \mathbb{Z}_p$ and $\vec{a} = (a_1, \dots, a_j)$

Definition 1. We use multiplicative cyclic groups $\mathbb{G}, \mathbb{G}_0$ of prime order p with an efficiently computable mapping $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_0$ such that $e(u^a, v^b) = e(u, v)^{ab}$, $\forall\ u, v \in \mathbb{G},\ a, b \in \mathbb{Z}_p$ and $e(u, v) \neq 1_{\mathbb{G}_0}$ whenever $u, v \neq 1_{\mathbb{G}}$. Here $1_{\mathbb{G}}$ (or $1_{\mathbb{G}_0}$) is identity element in $\mathbb{G}$ (or $\mathbb{G}_0$). We denote the bilinear group parameters as $\Sigma = (p, \mathbb{G}, \mathbb{G}_0, e)$ in the remainder of the paper. □

Definition 2. Given the instance $(g, g^a, \dots, g^{a^q}, g^{a^{q+2}}, \dots, g^{a^{2q}}) \in \mathbb{G}^{2q}$, where $a \leftarrow_R \mathbb{Z}_p$, a random generator $g \leftarrow_R \mathbb{G}$, the computational q-Diffie-Hellman Exponent (q-DHE) problem is to compute $g^{a^{q+1}}$. □

Definition 3. Let U be the universe of attributes and $\mathcal{P}(U) = \{S : S \subset U\}$. Let $\mathcal{P}(U)^* = \mathcal{P}(U) \setminus \{\emptyset\}$. Every non-empty subset $\mathbb{A}$ of $\mathcal{P}(U)^*$ is called an *access structure.* Any access structure $\mathbb{A}$ satisfying the condition [$A \subseteq B$ and $A \in \mathbb{A}$ implies $B \in \mathbb{A}$] is called a *monotone access structure.* An attribute set C *satisfies* the (monotone) access structure $\mathbb{A}$ (in other words, $\mathbb{A}$ *accepts* C) if and only if $C \in \mathbb{A}$. In this case, C is called an *authorized* set in $\mathbb{A}$. Note that C *does not* satisfy $\mathbb{A}$ if and only if $C \notin \mathbb{A}$. □

Definition 4. Let $\mathbb{A}$ be a monotone access structure. A linear secret sharing scheme (LSSS) for $\mathbb{A}$ over a field $\mathbb{Z}_p$ is an $\ell \times k$ matrix $\mathbb{M}$ with entries in $\mathbb{Z}_p$, along with a row labeling function ρ which associates each row i of $\mathbb{M}$ with an attribute $\rho(i)$ in $\mathbb{A}$, that consists of the following two polynomial time algorithms:

Share $(\mathbb{M}, \rho, \alpha)$ takes as input $\mathbb{M}, \rho$ and a secret $\alpha \in \mathbb{Z}_p$ to be shared. It samples $a_2, a_3, \dots, a_k \leftarrow_R \mathbb{Z}_p$ and sets $\vec{v} = (\alpha, a_2, a_3, \dots, a_k) \in \mathbb{Z}_p^k$. It outputs a set $\{\lambda_{\rho(i)} : \lambda_{\rho(i)} = \vec{M}_i \vec{v}\}_{i \in [\ell]}$ of ℓ shares, where $\vec{M}_i \in \mathbb{Z}_p^k$ is ith row of the matrix $\mathbb{M}$. The share $\lambda_{\rho(i)}$ belongs to the attribute $\rho(i)$.

$\mathsf{Recover}$ $(\mathbb{M}, \rho, L)$ takes as input $\mathbb{M}, \rho$ and an authorized attribute set $L \in \mathbb{A}$. It outputs a set of constants $\{\omega_i\}_{i \in I} \subset \mathbb{Z}_p$, where $I = \{i \in [\ell] : \rho(i) \in L\}$, satisfying $\sum_{i \in I} \omega_i \vec{M}_i = (1, 0, \dots, 0)$, i.e., $\sum_{i \in I} \omega_i \lambda_{\rho(i)} = \alpha$. □

Remark 1. Note that the constants $\{\omega_i\}_{i \in I}$ can be computed in time polynomial in the size of the matrix $\mathbb{M}$ using Gaussian elimination. We denote $\mathbb{A}$ by the LSSS $(\mathbb{M}, \rho)$ and is called LSSS-realizable access structure. □

2.1 ABKIS with Message Recovery Template

Apart from the Trusted Authority (TA), this model employs another device called *helper* which is physically secure but computationally limited. The TA creates a helper key and initial signing key for every user. The initial signing key is computed according to some access structure over user attributes. An ABKIS with message recovery is a set of the following six algorithms.

$\mathsf{Setup}(\kappa)$. The TA takes as input a security parameter κ and generates the system public parameters params and system master key MK. The public parameters params include a description of the attribute universe U, time period index space T and the message space $\mathcal{M}$. Here params are publicly available to all users and MK is kept secret by TA.

$\mathsf{KeyGen}(\mathsf{params}, \mathsf{MK}, \mathbb{A})$. On input $\mathsf{params}, \mathsf{MK}$ and an access structure $\mathbb{A}$ over a set of attributes $L_{\mathbb{A}} \subset U$, the TA computes a helper key $\mathsf{HK}_{\mathbb{A}}$ and an initial signing key $\mathsf{SK}_{\mathbb{A}}^0$ associated with $\mathbb{A}$. The helper key $\mathsf{HK}_{\mathbb{A}}$ is stored in the helper and the user holds the initial signing key $\mathsf{SK}_{\mathbb{A}}^0$.

$\mathsf{HelperUpdate}(\mathsf{params}, \mathsf{HK}_{\mathbb{A}}, t_1, t_2)$. This algorithm is executed by the helper for the user holding an access structure $\mathbb{A}$ with the input $\mathsf{params}, \mathsf{HK}_{\mathbb{A}}$, time period indices t_1, t_2 and returns an update key $\mathsf{UK}_{\mathbb{A}}^{t_1 \rightarrow t_2}$ for $\mathbb{A}$ from the time period t_1 to time period t_2.

$\mathsf{UserUpdate}(\mathsf{params}, \mathsf{SK}_{\mathbb{A}}^{t_1}, t_1, t_2, \mathsf{UK}_{\mathbb{A}}^{t_1 \rightarrow t_2})$. Given public parameters params, temporary signing key $\mathsf{SK}_{\mathbb{A}}^{t_1}$ associated with $\mathbb{A}$ for time period t_1, time period indices t_1, t_2 and update key $\mathsf{UK}_{\mathbb{A}}^{t_1 \rightarrow t_2}$, the user performs this algorithm and obtains the temporary signing key $\mathsf{SK}_{\mathbb{A}}^{t_2}$ associated with $\mathbb{A}$ for time period t_2. The user erases the temporary signing key $\mathsf{SK}_{\mathbb{A}}^{t_1}$ for the time period t_1.

$\mathsf{Sign}(\mathsf{params}, \mathsf{msg}, t, \mathsf{SK}_{\mathbb{A}}^t, W)$. To generate a signature on a message $\mathsf{msg} \in \mathcal{M}$ with respect to an attribute set $W \in \mathbb{A}$ at time period t, the signer runs this algorithm with the input $\mathsf{params}, \mathsf{msg}, t, \mathsf{SK}_{\mathbb{A}}^t, W$ and outputs a signature $\langle t, \Gamma \rangle$ consisting of the time period t and a signature Γ.

$\mathsf{Verify}(\mathsf{params}, \langle t, \Gamma \rangle, W)$. Given a candidate signature $\langle t, \Gamma \rangle$ and an attribute set W, this algorithm outputs 1 if Γ is a valid signature with respect to the attribute set W in time period t. Otherwise, it returns 0.

Remark 2. In the above ABKIS, one can update the temporary signing key $\mathsf{SK}_{\mathbb{A}}^{t_1}$ to $\mathsf{SK}_{\mathbb{A}}^{t_2}$ in one step for any time periods t_1, t_2. This mechanism is called *random-access key updates.* Neither forward secure systems [2] nor intrusion resilience systems [7] support this functionality. □

2.2 Security Definitions for ABKIS with Message Recovery

Following [4,14], we formalize here the key-insulated and strong key-insulated security, and signer privacy which are the typical security requirements for ABKIS schemes.

Key-insulated and strong key-insulated security. In key-insulated security notion, the exposure of any of the temporary signing keys for some time periods do not enable an adversary to create a valid signature for the non-exposed time periods as long as the corresponding helper keys are not compromised. On the other hand, in *strong* key-insulated security model, the exposure of helper keys do not enable an adversary to generate a valid signature for any time period as long as none of the temporary signing keys are compromised. These security notions are defined based on existential unforgeability under selective attribute set and time period index, and adaptive chosen message attack framework, in which the adversary needs to submit an attribute set and time period index pair before obtaining the system public parameters and the forgery must be created for that pair on any message. Formally, we define these security notions as follows.

Definition 5. (Key-insulated Security). An ABKIS scheme with message recovery is said to be $(\mathcal{T}, q_{\mathsf{KG}}, q_{\mathsf{TSK}}, q_{\mathsf{Sign}}, \epsilon)$-key-insulated if for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ running in time at most $\mathcal{T}$ that makes at most q_{KG} key generation queries, q_{TSK} temporary signing key queries and q_{Sign} signature queries, the adversary's advantage $Adv_{\mathcal{A}}^{\mathsf{key\text{-}ins}} = \Pr[\mathcal{A} \text{ wins}] \leq \epsilon$ in the following game (between a challenger $\mathcal{C}$ and the adversary $\mathcal{A}$):

1: First $\mathcal{C}$ selects a security parameter κ, and specifies an attribute universe U and time period index space T. Then $\mathcal{A}$ outputs an attribute set $W^* \subset U$ and a time period index $t^* \in T$.
2: $\mathcal{C}$ performs Setup algorithm and sends params to $\mathcal{A}$.
3: $\mathcal{A}$ is given access to the following three oracles.

- Key generation oracle $\mathcal{O}_{\mathsf{KG}}(\mathbb{A})$: when $\mathcal{A}$ submits an access structure $\mathbb{A}$ with the restriction $W^* \notin \mathbb{A}$, $\mathcal{C}$ runs KeyGen algorithm and returns a helper key, initial signing key pair $(\mathsf{HK}_{\mathbb{A}}, \mathsf{SK}_{\mathbb{A}}^0)$ to $\mathcal{A}$.
- Temporary signing key oracle $\mathcal{O}_{\mathsf{TSK}}(\mathbb{A}, t)$: When $\mathcal{A}$ submits an access structure $\mathbb{A}$ and time period index t with the condition $t \neq t^*$, $\mathcal{C}$ computes the temporary signing key $\mathsf{SK}_{\mathbb{A}}^t$ for the time period t by executing $\mathsf{HelperUpdate}, \mathsf{UserUpdate}$ algorithms and sends it to $\mathcal{A}$.
- Signature oracle $\mathcal{O}_{\mathsf{Sign}}(\mathsf{msg}, W, t)$: On receiving a message $\mathsf{msg} \in \mathcal{M}$, an attribute set $W \subset U$ and a time period index $t \in T$ from $\mathcal{A}$, $\mathcal{C}$ chooses

an access structure $\mathbb{A}$ such that $W \in \mathbb{A}$, computes the temporary signing key $\mathsf{SK}_{\mathbb{A}}^{t}$ and forwards the signature $\langle t, \Gamma \rangle$ to $\mathcal{A}$ that is returned by $\mathsf{Sign}(\mathsf{params}, \mathsf{msg}, t, \mathsf{SK}_{\mathbb{A}}^{t}, W)$.

4: At the end $\mathcal{A}$ outputs $(W^*, t^*, \mathsf{msg}^*, \Gamma^*)$.

We say that $\mathcal{A}$ succeeds if $\mathsf{Verify}(\mathsf{params}, \langle t^*, \Gamma^* \rangle, W^*) = 1$ and $(\mathsf{msg}^*, W^*, t^*)$ was never queried to the signature oracle. □

Remark 3. In the foregoing definition, the adversary can obtain the helper key and initial signing key for any access structure which is *not* satisfied by the challenged attribute set W^* by issuing key generation queries. So, he can further compute temporary signing key of that access structure for any time period of his choice including t^*, without accessing temporary signing key oracle because he knows the corresponding helper key.

On the other hand, the temporary signing key oracle is actually meant for the access structures accepting W^*. Adversary can get temporary signing key of an access structure accepting W^* for any time period except t^* by querying the temporary signing key oracle. The helper keys of such access structures are not known to adversary. Wlog, we can assume that adversary issues only temporary signing key queries for the access structures accepting W^*. □

In the strong key-insulated security notion defined below, the adversary is prohibited to request temporary signing keys and hence the challenge time period index t^* is not necessary to submit before receiving the system public parameters.

Definition 6. (Strong key-insulated Security). An ABKIS scheme with message recovery is said to be $(\mathcal{T}, q_{\mathsf{KG}}, q_{\mathsf{HK}}, q_{\mathsf{Sign}}, \epsilon)$-strong-key-insulated if for any PPT adversary $\mathcal{A}$ running in time at most $\mathcal{T}$ that makes at most q_{KG} key generation queries, q_{HK} helper key queries and q_{Sign} signature queries, the adversary's advantage $Adv_{\mathcal{A}}^{\mathsf{strong\text{-}key\text{-}ins}} = \Pr[\mathcal{A} \text{ wins}] \le \epsilon$ in the following game (between a challenger $\mathcal{C}$ and $\mathcal{A}$):

1: First $\mathcal{C}$ selects a security parameter κ, and specifies an attribute universe U. Then $\mathcal{A}$ outputs an attribute set $W^* \subset U$.
2: $\mathcal{C}$ performs Setup algorithm and sends params to $\mathcal{A}$.
3: $\mathcal{A}$ is given access to the following three oracles.

- Key generation oracle $\mathcal{O}_{\mathsf{KG}}(\mathbb{A})$: Same as Definition 5.
- Helper key oracle $\mathcal{O}_{\mathsf{HK}}(\mathbb{A})$: when $\mathcal{A}$ submits an access structure $\mathbb{A}$, $\mathcal{C}$ runs KeyGen algorithm, obtains the helper key $\mathsf{HK}_{\mathbb{A}}$ and returns it to $\mathcal{A}$.
- Signature oracle $\mathcal{O}_{\mathsf{Sign}}(\mathsf{msg}, W, t)$: Same as Definition 5.

4: At the end $\mathcal{A}$ outputs $(W^*, t^*, \mathsf{msg}^*, \Gamma^*)$.

We say that $\mathcal{A}$ succeeds if $\mathsf{Verify}(\mathsf{params}, \langle t^*, \Gamma^* \rangle, W^*) = 1$ and $(\mathsf{msg}^*, W^*, t^*)$ was never queried to the signature oracle. □

Remark 4. In Definition 6, temporary signing key queries are not explicitly provided for the adversary. However, the adversary can derive temporary signing keys for the access structures $\mathbb{A}$ such that $W^* \notin \mathbb{A}$ through key generation oracle $\mathcal{O}_{\mathsf{KG}}(\cdot)$. Hence the adversary can compromise a temporary signing key in any time period for the access structures not accepting the challenged attribute set W^*. On the other hand, he can obtain helper key of any access structure independent of such restriction by querying helper key oracle $\mathcal{O}_{\mathsf{HK}}(\cdot)$. $\square$

Signer Privacy. This property ensures that the distribution of a signature is independent of the temporary signing key that is used to generate it. This means that one cannot get any information about the access structure $\mathbb{A}$ held by the signer from a signature for the attribute set W in time period t, other than the fact that W satisfies $\mathbb{A}$. Precisely, the actual signer is indistinguishable from all the users whose access structures accepting the attribute set specified in the signature. Formally, this security notion is defined as follows.

Definition 7. An ABKIS scheme is said to provide signer privacy if for any message $\mathsf{msg} \in \mathcal{M}$, all $\langle \mathsf{params}, \mathsf{MK} \rangle \leftarrow \mathsf{Setup}(\kappa)$, all signing access structures $\mathbb{A}$ and $\mathbb{A}'$, all time periods t, all signing temporary signing keys $\mathsf{SK}^t_{\mathbb{A}}$ and $\mathsf{SK}^t_{\mathbb{A}'}$, all attribute sets W such that W satisfies both $\mathbb{A}$ and $\mathbb{A}'$, the distributions of $\mathsf{Sign}(\mathsf{params}, \mathsf{msg}, t, \mathsf{SK}^t_{\mathbb{A}}, W)$ and $\mathsf{Sign}(\mathsf{params}, \mathsf{msg}, t, \mathsf{SK}^t_{\mathbb{A}'}, W)$ are equal. $\square$

3 Proposed ABKIS with Message Recovery

3.1 ABKIS Scheme with Message Recovery for Small Universe

In this section, we present an ABKIS scheme with message recovery for small universes of attributes, referred as SU-ABKIS, that supports fixed length messages. We adapt the key updating technique used in [3] where the keys are randomized by means of pseudorandom function (PRF). A simple randomness is not adequate because the same attribute might held by several users and a user may derive the random exponents for other time periods from the current time period. Hence the randomness introduced in the key update process must be the time period as well as the attribute specific. To this end, we use a PRF family $\mathcal{F} = \{\mathsf{PRF}_s : s \leftarrow_R \{0,1\}^\kappa\}$, where $\mathsf{PRF}_s : \{0,1\}^{2\kappa} \rightarrow \{0,1\}^\kappa$ such that given a κ-bit seed s and 2κ-bit input ip, the PRF PRF_s outputs κ-bit string $\mathsf{PRF}_s(ip)$. We treat each time period index as an element in $\mathbb{Z}_p^*$, i.e., $T = \mathbb{Z}_p^*$ and the message space is $\mathcal{M} = \{0,1\}^{\ell_1}$. Let $U = \{1, 2, \ldots, n\} = [n]$ be the attribute universe. Our SU-ABKIS is composed of the following six algorithms.

$\mathsf{Setup}(\kappa)$: The TA carries out the following steps.

1. Choose $\Sigma = (p, \mathbb{G}, \mathbb{G}_0, e)$ according to the security parameter κ.
2. Select four collision resistant hash functions $H : \{0,1\}^* \rightarrow \mathbb{G}$, $H_1 : \{0,1\}^{\ell_1} \rightarrow \{0,1\}^{\ell_2}$, $H_2 : \{0,1\}^{\ell_2} \rightarrow \{0,1\}^{\ell_1}$ and $H' : \{0,1\}^* \rightarrow \{0,1\}^{\ell_1+\ell_2}$, where $\ell_1, \ell_2 \in \mathbb{N}$, the set of all natural numbers.
3. Sample $\alpha \leftarrow_R \mathbb{Z}_p$, a random generator $g \leftarrow_R \mathbb{G}$ and set $Y = e(g,g)^\alpha$.
4. Pick $w_1, w_2 \leftarrow_R \mathbb{G}$. Define a function $F : \mathbb{Z}_p \rightarrow \mathbb{G}$ by $F(t) = w_1 w_2^t$.

5. Select $v \leftarrow_R \mathbb{G}$. For each attribute $k \in U = [n]$, choose $v_k \leftarrow_R \mathbb{G}$.
6. The system master key is $\mathsf{MK} = \alpha$ and the system public parameters are published as $\mathsf{params} = \langle \Sigma, g, Y, v, w_1, w_2, \{v_k\}_{k\in[n]}, H, H_1, H_2, H', F, U = [n], T = \mathbb{Z}_p^*, \mathcal{M} = \{0,1\}^{\ell_1} \rangle$.

$\mathsf{KeyGen}(\mathsf{params}, \mathsf{MK}, (\mathbb{M}, \rho))$: To compute a helper key $\mathsf{HK}_{(\mathbb{M},\rho)}$ and an initial signing key $\mathsf{SK}^0_{(\mathbb{M},\rho)}$ associated with an access structure $(\mathbb{M}, \rho)$, the TA performs as follows. Each ith row $\vec{M}_i$ of the LSSS matrix $\mathbb{M}$ of size $\nu \times \tau$ is associated with an attribute $\rho(i) \in [n]$. We assume that the time period index is 0 for initial signing key.

1. Pick $\mathsf{hk}_{(\mathbb{M},\rho)} \leftarrow_R \{0,1\}^\kappa$ and choose $\mathsf{PRF}_{\mathsf{hk}_{(\mathbb{M},\rho)}} \in \mathcal{F}$.
2. Sample $z_2, \ldots, z_\tau \leftarrow_R \mathbb{Z}_p$ and set $\vec{v} = (\alpha, z_2, \ldots, z_\tau)$.
3. For each row $i \in [\nu]$,
   - compute $\lambda_{\rho(i)} = \vec{M}_i \vec{v}$ and $\gamma_{i,0} = \mathsf{PRF}_{\mathsf{hk}_{(\mathbb{M},\rho)}}(0||\rho(i))$
     (if the length of each input string for PRF is less than κ, then we add required number of 0 s on the left side of the string to make 2κ-bit long),
   - pick $r_i \leftarrow_R \mathbb{Z}_p$ and set
     $d^0_{i,1} = g^{r_i}, d^0_{i,2} = g^{\gamma_{i,0}}, d^0_{i,3} = g^{\lambda_{\rho(i)}}(vv_{\rho(i)})^{r_i} F(0)^{\gamma_{i,0}},\ d^0_{i,4} = \{d^0_{i,4,k} = v_k^{r_i}\}_{k\in[n]\setminus\{\rho(i)\}}$.
4. The initial signing key is $\mathsf{SK}^0_{(\mathbb{M},\rho)} = \langle (\mathbb{M}, \rho), \{d^0_{i,1}, d^0_{i,2}, d^0_{i,3}, d^0_{i,4} : i \in [\nu]\} \rangle$.
5. The helper key is $\mathsf{HK}_{(\mathbb{M},\rho)} = \langle \rho, \mathsf{hk}_{(\mathbb{M},\rho)} \rangle$.

$\mathsf{HelperUpdate}(\mathsf{params}, \mathsf{HK}_{(\mathbb{M},\rho)}, t_1, t_2)$: To construct the update key $\mathsf{UK}^{t_1\to t_2}_{(\mathbb{M},\rho)}$ for $(\mathbb{M}, \rho)$ from the time period t_1 to time period t_2, the helper carries out the following steps.

1. For each row $i \in [\nu]$,
   - compute $\gamma_{i,t_1} = \mathsf{PRF}_{\mathsf{hk}_{(\mathbb{M},\rho)}}(t_1||\rho(i))$, $\gamma_{i,t_2} = \mathsf{PRF}_{\mathsf{hk}_{(\mathbb{M},\rho)}}(t_2||\rho(i))$,
   - set $\mathsf{UK}^{t_1\to t_2}_{i,1} = F(t_2)^{\gamma_{i,t_2}} \cdot F(t_1)^{-\gamma_{i,t_1}}$, $\mathsf{UK}^{t_1\to t_2}_{i,2} = g^{\gamma_{i,t_2}}$.
2. The update key is $\mathsf{UK}^{t_1\to t_2}_{(\mathbb{M},\rho)} = \left\{\mathsf{UK}^{t_1\to t_2}_{i,1}, \mathsf{UK}^{t_1\to t_2}_{i,2} : i \in [\nu]\right\}$.

$\mathsf{UserUpdate}(\mathsf{params}, \mathsf{SK}^{t_1}_{(\mathbb{M},\rho)}, t_1, t_2, \mathsf{UK}^{t_1\to t_2}_{(\mathbb{M},\rho)})$: The user computes the temporary signing key $\mathsf{SK}^{t_2}_{(\mathbb{M},\rho)}$ for time period t_2 from the temporary signing key $\mathsf{SK}^{t_1}_{(\mathbb{M},\rho)}$ for time period t_1 by using update key $\mathsf{UK}^{t_1\to t_2}_{(\mathbb{M},\rho)}$ as follows.

1. Parse the temporary signing key for $(\mathbb{M}, \rho)$ at time period t_1 as $\mathsf{SK}^{t_1}_{(\mathbb{M},\rho)} = \langle (\mathbb{M}, \rho), \{d^{t_1}_{i,1}, d^{t_1}_{i,2}, d^{t_1}_{i,3}, d^{t_1}_{i,4} : i \in [\nu]\} \rangle$, where $d^{t_1}_{i,1} = g^{r_i}, d^{t_1}_{i,2} = g^{\gamma_{i,t_1}}$, $d^{t_1}_{i,3} = g^{\lambda_{\rho(i)}}(vv_{\rho(i)})^{r_i} F(t_1)^{\gamma_{i,t_1}}$, $d^{t_1}_{i,4} = \{d^{t_1}_{i,4,k} = v_k^{r_i}\}_{k\in[n]\setminus\{\rho(i)\}}$.
2. The update key $\mathsf{UK}^{t_1\to t_2}_{(\mathbb{M},\rho)}$ is parsed as $\mathsf{UK}^{t_1\to t_2}_{(\mathbb{M},\rho)} = \left\{\mathsf{UK}^{t_1\to t_2}_{i,1}, \mathsf{UK}^{t_1\to t_2}_{i,2} : i \in [\nu]\right\}$.
3. The user sets the temporary signing key for $(\mathbb{M}, \rho)$ and time period t_2 as $\mathsf{SK}^{t_2}_{(\mathbb{M},\rho)} = \langle (\mathbb{M}, \rho), \{d^{t_2}_{i,1}, d^{t_2}_{i,2}, d^{t_2}_{i,3}, d^{t_2}_{i,4} : i \in [\nu]\} \rangle$, where $d^{t_2}_{i,1} = d^{t_1}_{i,1}, d^{t_2}_{i,2} = \mathsf{UK}^{t_1\to t_2}_{i,2}$, $d^{t_2}_{i,3} = d^{t_1}_{i,3} \cdot \mathsf{UK}^{t_1\to t_2}_{i,1}$ and $d^{t_2}_{i,4,k} = d^{t_1}_{i,4,k}, \forall k \in [n] \setminus \{\rho(i)\}$.
4. The temporary signing key $\mathsf{SK}^{t_1}_{(\mathbb{M},\rho)}$ for the time period t_1 will be discarded.

Note: At time period t_2, it can be seen that $d_{i,1}^{t_2} = g^{r_i}, d_{i,2}^{t_2} = g^{\gamma_{i,t_2}}, d_{i,3}^{t_2} = g^{\lambda_{\rho(i)}}(vv_{\rho(i)})^{r_i} F(t_2)^{\gamma_{i,t_2}}$, $d_{i,4}^{t_2} = \{d_{i,4,k}^{t_2} = v_k^{r_i}\}_{k \in [n] \setminus \{\rho(i)\}}$.

$\mathsf{Sign}(\mathsf{params}, \mathsf{msg}, t, \mathsf{SK}^t_{(\mathbb{M},\rho)}, W)$: The signer with a temporary signing key $\mathsf{SK}^t_{(\mathbb{M},\rho)}$ for LSSS access structure $(\mathbb{M}, \rho)$ and time period t generates the signature on a message $\mathsf{msg} \in \{0,1\}^{\ell_1}$ for the attribute set W satisfying $(\mathbb{M}, \rho)$ as follows. The temporary signing key $\mathsf{SK}^t_{(\mathbb{M},\rho)}$ for the time period t is parsed as $\mathsf{SK}^t_{(\mathbb{M},\rho)} = \big\langle(\mathbb{M}, \rho), \{d_{i,1}^t, d_{i,2}^t, d_{i,3}^t, d_{i,4}^t : i \in [\nu]\}\big\rangle$.

1. Compute $\beta = H'(t||W||Y), m = H_1(\mathsf{msg})||(H_2(H_1(\mathsf{msg})) \oplus \mathsf{msg})$ and $c = \beta \oplus m$.
2. Obtain a set of constants $\{\omega_i \in \mathbb{Z}_p : i \in I\} \leftarrow \mathsf{Recover}(\mathbb{M}, \rho, W)$, where $I = \{i \in [\nu] : \rho(i) \in W\}$, satisfying $\sum_{i \in I} \omega_i \vec{M}_i = (1, 0, \ldots, 0)$. This is possible since W satisfies $(\mathbb{M}, \rho)$.
3. Choose $z_2', \ldots, z_\tau' \leftarrow_R \mathbb{Z}_p$ and define $\vec{z} = (0, z_2', \ldots, z_\tau')$.
4. For each $i \in [\nu]$, pick $r_i', b_{i,t} \leftarrow_R \mathbb{Z}_p$ and compute $\widehat{d}_{i,1}^t = d_{i,1}^t \cdot g^{r_i'}, \widehat{d}_{i,2}^t = d_{i,2}^t \cdot g^{b_{i,t}}$, $\widehat{d}_{i,3}^t = d_{i,3}^t \cdot g^{\vec{M}_i \vec{z}}(vv_{\rho(i)})^{r_i'} F(t)^{b_{i,t}}, \widehat{d}_{i,4}^t = \{\widehat{d}_{i,4,k}^t = d_{i,4,k}^t \cdot v_k^{r_i'}\}_{k \in [n] \setminus \{\rho(i)\}}$.
5. Sample $\theta, \xi, \zeta \leftarrow_R \mathbb{Z}_p$ and set $\sigma = g^\theta, \sigma_1 = g^\xi \prod\limits_{i \in I} \left(\widehat{d}_{i,1}^t\right)^{\omega_i}, \sigma_2 = g^\zeta \prod\limits_{i \in I} \left(\widehat{d}_{i,2}^t\right)^{\omega_i}$,
$$\sigma_3 = \prod_{i \in I} \Big(\widehat{d}_{i,3}^t \prod_{k \in W, k \neq \rho(i)} \widehat{d}_{i,4,k}^t\Big)^{\omega_i} \cdot \big(v \prod_{k \in W} v_k\big)^\xi \cdot H(c||\sigma||\sigma_1||\sigma_2||t||W)^\theta \cdot F(t)^\zeta.$$
6. The signature is $\Gamma = \langle c, \sigma, \sigma_1, \sigma_2, \sigma_3\rangle$ and finally outputs $\langle t, \Gamma\rangle$.

$\mathsf{Verify}(\mathsf{params}, \langle t, \Gamma\rangle, W)$: Given a signature $\langle t, \Gamma = \langle c, \sigma, \sigma_1, \sigma_2, \sigma_3\rangle\rangle$ and an attribute set W, the verifier performs in the following way.

1. Check validity of the following equation
$$\frac{e(\sigma_3, g)}{e(v \prod_{k \in W} v_k, \sigma_1) \cdot e(H(c||\sigma||\sigma_1||\sigma_2||t||W), \sigma) \cdot e(F(t), \sigma_2)} \overset{?}{=} Y. \qquad (1)$$
If it is not valid, output 0. Otherwise, proceed as follows.
2. Compute $\beta = H'(t||W||Y)$ and $m = c \oplus \beta$.
3. Recover the message $\mathsf{msg} = [\![m]\!]^{\ell_1} \oplus H_2({}^{\ell_2}[\![m]\!])$.
4. Return 1 if ${}^{\ell_2}[\![m]\!] = H_1(\mathsf{msg})$ and accept Γ as a valid signature of the message msg in time period t. In this case, the message $\mathsf{msg} \in \{0,1\}^{\ell_1}$ is recovered. Otherwise, output 0.

Remark 5. We can modify the above construction for messages of fixed length ℓ_1 to deal with messages of any length, i.e., $\mathsf{msg} \in \{0,1\}^*$. To sign a message of length less than ℓ_1, one can pad spaces after the message until ℓ_1 and can use the foregoing scheme. If message size is larger than ℓ_1, a part of the message of length ℓ_1 is embed in a signature and can be recovered in verification phase and the other message segment is sent together with the signature. Specifically, divide the message msg into two parts as $\mathsf{msg} = \mathsf{msg}_1||\mathsf{msg}_2$ such that size of msg_1 is ℓ_1. In the corresponding scheme, replace msg with msg_1, compute $\beta = H'(t||W||Y||\mathsf{msg}_2)$ instead of $\beta = H'(t||W||Y)$ and everything else remains the same. In verification phase, recover msg_1 and pad with the available msg_2 as $\mathsf{msg}_1||\mathsf{msg}_2$ in order to obtain msg. □

Remark 6. In signing process, W satisfies $(\mathbb{M}, \rho)$ means there exists an attribute set $\widehat{W} \subset W \bigcap L_{(\mathbb{M},\rho)}$, $L_{(\mathbb{M},\rho)}$ is the set of attributes assigned to the rows of the matrix $\mathbb{M}$, such that the rows of $\mathbb{M}$ corresponding to the attributes of $\widehat{W}$ (the row set I in the construction) spans the vector $(1, 0, \ldots, 0)$. The signer computes the signature using the signing key components corresponding to the attributes appeared in $\widehat{W}$. The set $\widehat{W}$ is completely hidden in the process (i.e., hidden in W) and the verifier is given the attribute set W. In verifier or adversary point of view, the signer can use any subset of W. Consequently, the verifier cannot identify the subset of signer attributes which has originally been used to generate a signature from the set $\mathcal{P}(W)^*$ of all non-empty subsets of W. But, he can verify the signature using the whole attribute set W. In contrast, the signing and verification attribute sets are same in existing ABKIS [4], i.e., the signer needs to expose the attributes used to create a signature. □

3.2 Large Universe ABKIS Scheme with Message Recovery

We present here an ABKIS scheme with message recovery for large universes of attributes, which we denote as LU-ABKIS. In this construction $U = \mathbb{Z}_p^*$ be the attribute universe. We impose a bound, say N, on the size of attribute set used in signing algorithm. Note that one can extend our small universe ABKIS construction to support large universe by using a collision resistant hash function to compute the attribute values after system setup. However, the signer needs to reveal his attributes used to create a signature as in ABKIS scheme of [4], which is not desirable. To preserve signer attribute anonymity as well as to realize constant-size signature like our small universe ABKIS scheme, we change the signing key generation process using the technique of [1].

$\mathsf{Setup}(\kappa)$: Given a security parameter κ, the TA executes the following steps.

1: Choose suitable $\Sigma = (p, \mathbb{G}, \mathbb{G}_0, e)$ according to κ.
2: Select four collision resistant hash functions $H : \{0,1\}^* \to \mathbb{G}$, $H_1 : \{0,1\}^{\ell_1} \to \{0,1\}^{\ell_2}$, $H_2 : \{0,1\}^{\ell_2} \to \{0,1\}^{\ell_1}$ and $H' : \{0,1\}^* \to \{0,1\}^{\ell_1+\ell_2}$, where $\ell_1, \ell_2 \in \mathbb{N}$.
3: Choose a PRF family $\mathcal{F} = \{\mathsf{PRF}_s : s \leftarrow_R \{0,1\}^\kappa\}$, where $\mathsf{PRF}_s : \{0,1\}^{2\kappa} \to \{0,1\}^\kappa$.
4: Sample $\alpha \leftarrow_R \mathbb{Z}_p$, a random generator $g \leftarrow_R \mathbb{G}$ and set $Y = e(g,g)^\alpha$.
5: Pick $w_1, w_2 \leftarrow_R \mathbb{G}$. Define a function $F : \mathbb{Z}_p \to \mathbb{G}$ by $F(t) = w_1 w_2^t$.
6: Pick $V_0, V_1, \ldots, V_N \leftarrow_R \mathbb{G}$. Where N is a bound on the size of signing attribute set used in signing algorithm below.
7: The system master key is $\mathsf{MK} = \alpha$ and the system public parameters are $\mathsf{params} = \langle \Sigma, N, g, Y, w_1, w_2, V_0, \{V_k\}_{k\in[N]}, H, H_1, H_2, H', F, U = \mathbb{Z}_p^*, T = \mathbb{Z}_p^*, \mathcal{M} = \{0,1\}^{\ell_1}\rangle$.

$\mathsf{KeyGen}(\mathsf{params}, \mathsf{MK}, (\mathbb{M}, \rho))$: The TA performs as follows. Each ith row $\vec{M}_i$ of the matrix $\mathbb{M}$ of size $\nu \times \tau$ is associated with an attribute $\rho(i) \in \mathbb{Z}_p^*$.

1: Pick $\mathsf{hk}_{(\mathbb{M},\rho)} \leftarrow_R \{0,1\}^\kappa$ and choose $\mathsf{PRF}_{\mathsf{hk}_{(\mathbb{M},\rho)}} \in \mathcal{F}$.
2: Sample $z_2, \ldots, z_\tau \leftarrow_R \mathbb{Z}_p$ and set $\vec{v} = (\alpha, z_2, \ldots, z_\tau)$.

3: For each row $i \in [\nu]$, compute $\lambda_{\rho(i)} = \vec{M}_i \vec{v}$ and $\gamma_{i,0} = \mathsf{PRF}_{\mathsf{hk}_{(\mathbb{M},\rho)}}(0||\rho(i))$. Pick $r_i \leftarrow_R \mathbb{Z}_p$ and set $d^0_{i,1} = g^{r_i}, d^0_{i,2} = g^{\gamma_{i,0}}, d^0_{i,3} = g^{\lambda_{\rho(i)}} V_0^{r_i} F(0)^{\gamma_{i,0}}$, $d^0_{i,4} = \big\{d^0_{i,4,k} = \big(V_1^{-\rho(i)^{k-1}} V_k\big)^{r_i}\big\}_{k=2}^{N}$.

4: The initial signing key of $(\mathbb{M},\rho)$ is $\mathsf{SK}^0_{(\mathbb{M},\rho)} = \big\langle(\mathbb{M},\rho), \{d^0_{i,1}, d^0_{i,2}, d^0_{i,3}, d^0_{i,4}\}_{i\in[\nu]}\big\rangle$.

5: The helper key is $\mathsf{HK}_{(\mathbb{M},\rho)} = \langle \rho, \mathsf{hk}_{(\mathbb{M},\rho)}\rangle$.

$\mathsf{HelperUpdate}(\mathsf{params}, \mathsf{HK}_{(\mathbb{M},\rho)}, t_1, t_2)$: Same as SU-ABKIS construction.
$\mathsf{UserUpdate}(\mathsf{params}, \mathsf{SK}^{t_1}_{(\mathbb{M},\rho)}, t_1, t_2, \mathsf{UK}^{t_1\to t_2}_{(\mathbb{M},\rho)})$: Same as SU-ABKIS construction.
$\mathsf{Sign}(\mathsf{params}, \mathsf{msg}, t, \mathsf{SK}^t_{(\mathbb{M},\rho)}, W)$: Here $\mathsf{msg} \in \{0,1\}^{\ell_1}$ and W satisfies $(\mathbb{M},\rho)$

The signing key $\mathsf{SK}^t_{(\mathbb{M},\rho)}$ is parsed as $\mathsf{SK}^t_{(\mathbb{M},\rho)} = \big\langle(\mathbb{M},\rho), \{d^t_{i,1}, d^t_{i,2}, d^t_{i,3}, d^t_{i,4} : i \in [\nu]\}\big\rangle$. Note that $|W| < N$.

1: Compute $\beta = H'(t||W||Y), m = H_1(\mathsf{msg})||(H_2(H_1(\mathsf{msg})) \oplus \mathsf{msg})$ and $c = \beta \oplus m$.

2: Choose $z'_2, \ldots, z'_\tau \leftarrow_R \mathbb{Z}_p$ and define $\vec{z} = (0, z'_2, \ldots, z'_\tau)$.

3: For each row $i \in [\nu]$, pick $r'_i, b_{i,t} \leftarrow_R \mathbb{Z}_p$ and compute $\widehat{d}^t_{i,1} = d^t_{i,1} \cdot g^{r'_i}, \widehat{d}^t_{i,2} = d^t_{i,2} \cdot g^{b_{i,t}}$, $\widehat{d}^t_{i,3} = d^t_{i,3} \cdot g^{\vec{M}_i \vec{z}} V_0^{r'_i} F(t)^{b_{i,t}}, \widehat{d}^t_{i,4} = \big\{\widehat{d}^t_{i,4,k} = d^t_{i,4,k} \cdot (V_1^{-\rho(i)^{k-1}} V_k)^{r'_i}\big\}_{k=2}^{N}$.

4: Obtain a set of constants $\{\omega_i \in \mathbb{Z}_p : i \in I\} \leftarrow \mathsf{Recover}(\mathbb{M}, \rho, W)$, where $I = \{i \in [\nu] : \rho(i) \in W\}$, satisfying $\sum_{i\in I} \omega_i \vec{M}_i = (1, 0, \ldots, 0)$. This is possible because W satisfies $(\mathbb{M},\rho)$.

5: Compute $(y_1, y_2, \ldots, y_N)$ such that
$P_W(X) = \prod_{w\in W}(X - w) = \sum_{j=1}^{|W|+1} y_j \cdot X^{j-1}$. Note that if $|W| + 1 < N$, then set $y_{|W|+2} = \cdots = y_N = 0$. So, $P_W(X) = \sum_{j=1}^{N} y_j \cdot X^{j-1}$.

6: Sample $\theta, \xi, \zeta \leftarrow_R \mathbb{Z}_p$ and set $\sigma = g^\theta, \sigma_1 = g^\xi \prod_{i\in I} \big(\widehat{d}^t_{i,1}\big)^{\omega_i}, \sigma_2 = g^\zeta \prod_{i\in I} \big(\widehat{d}^t_{i,2}\big)^{\omega_i}$,

$$\sigma_3 = \prod_{i\in I} \Big(\widehat{d}^t_{i,3} \prod_{k=2}^{N} \big(\widehat{d}^t_{i,4,k}\big)^{y_k}\Big)^{\omega_i} \cdot \big(V_0 \prod_{k\in[N]} V_k^{y_k}\big)^\xi \cdot H(c||\sigma||\sigma_1||\sigma_2||t||W)^\theta \cdot F(t)^\zeta.$$

7: The signature is $\Gamma = \langle c, \sigma, \sigma_1, \sigma_2, \sigma_3\rangle$ and output $\langle t, \Gamma\rangle$.

$\mathsf{Verify}(\mathsf{params}, \langle t, \Gamma\rangle, W)$: Given a signature $\langle t, \Gamma = \langle c, \sigma, \sigma_1, \sigma_2, \sigma_3\rangle\rangle$ and an attribute set W with $|W| < N$, the verifier executes the following steps.

1: Compute $(y_1, y_2, \ldots, y_N)$ such that
$P_W(X) = \prod_{w\in W}(X - w) = \sum_{k\in[N]} y_k \cdot X^{k-1}$ as above.

2: Check validity of the following equation

$$\frac{e(\sigma_3, g)}{e\big(V_0 \prod_{k\in[N]} V_k^{y_k}, \sigma_1\big) \cdot e\big(H(c||\sigma||\sigma_1||\sigma_2||t||W), \sigma\big) \cdot e(F(t), \sigma_2)} \stackrel{?}{=} Y. \quad (2)$$

If it is not valid, output 0. Otherwise, proceed as in a manner similar to that of our small universe construction SU-ABKIS.

3.3 Security Proof of SU-ABKIS and LU-ABKIS

Theorem 1. *Suppose the attribute universe U has q attributes (resp., $N = q$ is a bound on the size of signing attribute set used in signing phase), the hash function H is modeled as random oracle, the hash functions H_1, H_2, H' are collision resistant and $\mathcal{F} = \{\mathsf{PRF}_s : s \leftarrow_R \{0,1\}^\kappa\}$ is PRF family. Assume the computational q-DHE problem is $(\mathcal{T}', \epsilon)$-hard in $\mathbb{G}$. Then, our SU-ABKIS (resp., LU-ABKIS) scheme is $(\mathcal{T}, q_{\mathsf{KG}}, q_{\mathsf{TSK}}, q_{\mathsf{Sign}}, \epsilon)$-key-insulated in the random oracle model, where $\mathcal{T} = \mathcal{T}' - \mathcal{O}\big(q^2(q_{\mathsf{KG}} + q_{\mathsf{TSK}}) + q_{\mathsf{Sign}} + q_{\mathsf{H}}\big)\mathcal{T}_{exp} - 2\mathcal{T}_{pair}$. Here, q_{H} is number of H hash queries allowed during simulation, $\mathcal{T}_{exp}$ is cost of one exponentiation and $\mathcal{T}_{pair}$ is cost of one pairing computation.*

Theorem 2. *Suppose the attribute universe U has q attributes (resp., $N = q$ is a bound on the size of signing attribute set used in signing phase), the hash function H is modeled as random oracle, the hash functions H_1, H_2, H' are collision resistant and $\mathcal{F} = \{\mathsf{PRF}_s : s \leftarrow_R \{0,1\}^\kappa\}$ is PRF family. Assume the computational q-DHE problem is $(\mathcal{T}', \epsilon)$-hard in $\mathbb{G}$. Then, our SU-ABKIS (resp., LU-ABKIS) scheme is $(\mathcal{T}, q_{\mathsf{KG}}, q_{\mathsf{HK}}, q_{\mathsf{Sign}}, \epsilon)$-strong-key-insulated in the random oracle model, where $\mathcal{T} = \mathcal{T}' - \mathcal{O}\big(q^2 \cdot q_{\mathsf{KG}} + q_{\mathsf{Sign}} + q_{\mathsf{H}}\big)\mathcal{T}_{exp} - 2\mathcal{T}_{pair}$.*

Theorem 3. *Both the proposed SU-ABKIS and LU-ABKIS schemes provide signer privacy.*

Due to page restriction, the proofs will be given in full version of the paper.

4 Some Possible Extensions

Constructions Without Random Oracles. We can further extend our ABKIS schemes to realize security reduction in standard model as follows. The hash function H is redefined as $H : \{0,1\}^* \to \{0,1\}^\varrho$. Pick $\varpi_0, \varpi_1, \ldots, \varpi_\varrho \leftarrow_R \mathbb{G}$ and define $\widehat{F} : \{0,1\}^\varrho \to \mathbb{G}$ by $\widehat{F}(x_1, \ldots, x_\varrho) = \varpi_0 \varpi_1^{x_1} \cdots \varpi_\varrho^{x_\varrho}$. In signature component σ_3, replace $H(c||\sigma||\sigma_1||\sigma_2||t||W)^\theta$ with $\widehat{F}\big(H(c||\sigma_1||\sigma_2||t||W)\big)^\theta$ and change the verification tests accordingly.

Schemes Supporting Negative Attributes. We can extend our constructions to support *negative* attributes by treating the negation of an attribute as a separate attribute. This doubles the total number of attributes used in the system. However, the resulting schemes attains the same efficiency as that of our monotone access structure primitives.

5 Conclusion

We presented the *first* ABKIS schemes with message recovery for expressive LSSS-realizable access structures that feature constant number of pairing computations required for signature verification process and constant message-signature length. The (strong) key-insulated security of ABKIS schemes are reduced to the computational q-DHE problem in selective attribute set and random oracle model. Both the proposed constructions provide signer privacy. The signing key size in our constructions is quadratic in number of involved attributes.

Acknowledgement. The authors would like to thank the anonymous reviewers of this paper for their valuable comments and suggestions.

References

1. Attrapadung, N., Herranz, J., Laguillaumie, F., Libert, B., de Panafieu, E., Rfols, C.: Attribute-based encryption schemes with constant-size ciphertexts. Theo. Comput. Sci. **422**, 15–38 (2012)
2. Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 255–271. Springer, Heidelberg (2003)
3. Chen, J.H., Wang, Y.T., Chen, K.F.: Attribute-based key-insulated encryption. J. Inf. Sci. Eng. **27**(2), 437–449 (2011)
4. Chen, J., Long, Y., Chen, K., Guo, J.: Attribute-based key-insulated signature and its applications. Inf. Sci. **275**, 57–67 (2014)
5. Dodis, Y., Katz, J., Xu, S., Yung, M.: Key-Insulated Public Key Cryptosystems. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 65–75. Springer, Heidelberg (2002)
6. Gagné, M., Narayan, S., Safavi-Naini, R.: Short pairing-efficient threshold-attribute-based signature. In: Abdalla, M., Lange, T. (eds.) Pairing 2012. LNCS, vol. 7708, pp. 295–313. Springer, Heidelberg (2013)
7. Itkis, G., Reyzin, L.: SiBIR: signer-base intrusion-resilient signatures. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 499–599. Springer, Heidelberg (2002)
8. Li, J., Au, M.H., Susilo, W., Xie, D., Ren, K.: Attribute-based signature and its applications. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2010, pp. 60–69. ACM (2010)
9. Li, J., Kim, K.: Hidden attribute-based signatures without anonymity revocation. Inf. Sci. **180**(9), 1681–1689 (2010)
10. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-based signatures: achieving attribute-privacy and collusion-resistance. IACR Cryptology ePrint Archive 2008, 328 (2008)
11. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-based signatures. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 376–392. Springer, Heidelberg (2011)
12. Shahandashti, S.F., Safavi-Naini, R.: Threshold attribute-based signatures and their application to anonymous credential systems. In: Preneel, B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 198–216. Springer, Heidelberg (2009)
13. Wang, K., Mu, Y., Susilo, W., Guo, F.: Attribute-based signature with message recovery. In: Huang, X., Zhou, J. (eds.) ISPEC 2014. LNCS, vol. 8434, pp. 433–447. Springer, Heidelberg (2014)
14. Weng, J., Liu, S., Chen, K., Li, X.: Identity-based key-insulated signature with secure key-updates. In: Lipmaa, H., Yung, M., Lin, D. (eds.) Inscrypt 2006. LNCS, vol. 4318, pp. 13–26. Springer, Heidelberg (2006)
15. Zhang, F., Susilo, W., Mu, Y.: Identity-based partial message recovery signatures (or how to shorten id-based signatures). In: S. Patrick, A., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, pp. 45–56. Springer, Heidelberg (2005)