DE GRUYTER OPEN

BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **15**, No 3

Sofia • 2015

Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.1515/cait-2015-0039

Attribute-Based Parallel Key-Insulated Signature

Jianhong Chen[1], *Kun Yu*[1], *Yu Long*[2], *Kefei Chen*[3]

[1]*Faculty of Computer Engineering, Huaiyin Institute of Technology, Huaian 223003, Jiangsu, China*
[2]*Dept. of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China*
[3]*School of Science, Hangzhou Normal University, Hangzhou 310036, Zhejiang, China*
Emails: jianhong_chen_cis@163.com Varguard@163.com longyu@sjtu.edu.cn kfchen@sjtu.edu.cn

Abstract: *To deal with the key-exposure protection problem in attribute-based signature systems, we extend the parallel key-insulated mechanism to attribute-based signature scenarios, and then introduce the primitive of an Attribute-Based Parallel Key-Insulated Signature* (*ABPKIS*)*. After formalizing the definition and security notions for ABPKIS, a concrete ABPKIS scheme is presented. The security of our proposed ABPKIS scheme can be proved on a standard model. According to our knowledge, this is the first ABPKIS scheme up to now. Moreover, this is also the first concrete attribute-based key-insulated signature construction supporting multi-helpers.*

Keywords: *Parallel key-insulated, attribute-based signature, key-exposure.*

1. Introduction

To protect private keys, D o d i s et al. [2] introduced the key insulation mechanism in 2002. In a key-insulated cryptosystem, the helper keys are kept in a helper (a physically-secure, but computationally-limited device) while the temporary private keys are stored in a powerful but insecure device where cryptographic computations occur. The temporary private keys are refreshed at discrete times via interaction between the user and the helper, and the public key remains constant during the lifetime of the system. To enhance the security and flexibility of the key insulation mechanism, the idea of parallel key insulation [3, 6, 8, 9, 10] was put

forward. In a parallel key-insulated cryptosystem, distinct independent helpers are alternatively used in key update operations.

Attribute-Based Signature (ABS) [4, 5, 7] is a kind of signature that manifests a pretension to the attributes that the underlying signer owns. Attribute-based signatures have many applications, such as attribute based messaging systems and anonymous authentication. In 2014 C h e n et al. [1] proposed an Attribute-Based Key-Insulated Signature (ABKIS), which is a crucial technique for protecting signing keys in ABS systems. Nevertheless, in Chen et al. scheme, where only one helper is used, if both the user's helper key and some of his temporary signing keys are exposed, the security of the signature system will be entirely lost.

To further enhance the security of the system by allowing frequent key-updates without increasing the risk of helper key exposure, we extend the parallel key-insulated mechanism to attribute-based signature scenarios and give an Attribute-Based Parallel Key-Insulated Signature (ABPKIS) scheme which is provably secure in the standard model. Our proposed scheme is strongly key-insulated, and even if one of a user's helper keys and some of his temporary signing keys are exposed, it is still impossible for an adversary to obtain all of this user's temporary signing keys. To the best of our knowledge, this is the first ABPKIS scheme up to now. Further on, this is also the first concrete attribute-based key-insulated signature construction supporting multi-helpers.

2. Preliminaries

Throughout this paper, we let Z_p denote the set $\{0, 1, 2,\ldots, p-1\}$ and Z_p^* denote $Z_p \backslash \{0\}$. For a finite set S, $x \xleftarrow{U} S$ means choosing an element x from S with a uniform distribution.

2.1. Bilinear pairings

Our ABPKIS scheme uses a bilinear map (pairing), $\hat{e} : G_1 \times G_1 \rightarrow G_2$, where G_1 is a multiplicative group with a prime order p and G_2 is also a multiplicative group with a prime order p. The pairing $\hat{e}$ satisfies the following conditions:

- *Bilinear*: For all $g_1, g_2 \in G_1$ and for all $a, b \in Z_p^*$, we have $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$.
- *Non-degenerate*: There exist $g_1, g_2 \in G_1$ such that $\hat{e}(g_1, g_2) \neq 1$.
- *Computable*: There is an efficient algorithm to compute (g_1, g_2) for all $g_1, g_2 \in G_1$.

2.2. Computational Diffie-Hellman assumptions

Definition 1. The CDH problem in G_1 is, given $g, g^a, g^b \in G_1$, to compute g^{ab} (for unknown randomly chosen $a, b \in Z_p^*$).

Definition 2. We say that the (t, ε)-CDH assumption holds in a group G_1 if no algorithm running in time t can solve the CDH problem in G_1 with probability at least ε.

3. Model of ABPKIS

3.1. Definition

An ABPKIS scheme consists of six algorithms:

- Setup(κ, l, d): Given a security parameter κ, the length l of some universe U, of size $|U|$ and a threshold value d, the PKG runs this algorithm to output a master key (msk) and public parameters (cp).
- KeyGen(msk, cp, ω): Given the user's identity $\omega \subseteq U$ as a set representing the user's attributes, the public parameters cp and the master key msk, the PKG runs this algorithm to output an initial private key $TK_{\omega,0}$ and two helper keys ($HK_{\omega,0}$, $HK_{\omega,1}$) that correspond to ω.
- HelperUpt(cp, t, ω, HK_{ω}): Given the public parameters cp, the period index t, an identity $\omega \subseteq U$ and its helper key HK_{ω}, the helper runs this algorithm to output the key-update information for ω for a period t, $UI_{\omega,t}$.
- UserUpt(cp, t, ω, $TK_{\omega,t-1}$,$UI_{\omega,t}$, $TK_{\omega,t}$): Given the public parameters cp, the period index t, an identity $\omega \subseteq U$, the key-update information $UI_{\omega,t}$, and the temporary private key $TK_{\omega,t-1}$ that corresponds to ω and $\tilde{t}$–1, the user with identity ω runs this algorithm to output the temporary private key $TK_{\omega,t}$ that corresponds to ω and t.
- Sign(cp, t, m, $TK_{\omega,t}$): Given the public parameters cp, a period index t, a message m and a temporary private key $TK_{\omega,t}$, this algorithm outputs a signature (t, σ) with regard to the period index t, the attribute set $\omega' \subseteq \omega$ and message m.
- Verify(cp, m, ω', (t, σ)): Given the public parameters cp, a message m, an attribute set ω' and a signature (t, σ) with regard to the period t, an attribute set ω' and message m, this algorithm outputs 1 if (t, σ) is a valid signature and 0 otherwise.

3.2. Security notions for ABPKIS

For convenience, we give the definition of a restricted identity as below: a restricted identity ω^* satisfies $\alpha \subseteq \omega^*$, where α is the challenge identity.

3.2.1. Key-insulated security

The key-insulated security notion captures the intuition that, if an adversary does not compromise the helper key for a given identity (i.e., an attribute set), then the exposure of any of the private keys does not enable an adversary to forge a valid signature for the non-exposed time periods.

Formally, for an ABPKIS scheme, its key-insulated security can be defined via the following game of existential unforgeability against a chosen identity and an adaptive chosen message attack under key-exposure (UF-ID&KE-CMA) between an adversary $\mathscr{A}$ and a challenger $\mathscr{C}$.

- **Init.** The adversary declares the identity α, where $|\alpha| < d$ and d is the threshold and the time period index t^* that he wishes to be challenged upon.

- **Setup.** The challenger $\mathscr{C}$ runs the Algorithm Setup and tells the adversary $\mathscr{A}$ the public parameters.
- **Query Phase.** The adversary $\mathscr{A}$ adaptively issues a set of queries as below:

– *Key Generation queries* $\langle\omega\rangle$: $\mathscr{C}$ first runs the algorithm KeyGen to obtain the initial private key $TK_{\omega,0}$ and the helper key HK_ω that corresponds to the identity ω. It then sends these results to the adversary.

– *Helper Key queries* $\langle\omega\rangle$: $\mathscr{C}$ runs the algorithm KeyGen to generate HK_ω and sends it to the adversary.

– *Temporary Private Key queries* $\langle\omega, t\rangle$: $\mathscr{C}$ runs the algorithm UserUpt to obtain the temporary private key for the identity ω and period index t. It then sends the result to $\mathscr{A}$.

– *Signing queries* $\langle\omega, t, m\rangle$: $\mathscr{C}$ runs the algorithm Sign(cp, t, m, $TK_{\omega,t}$) to generate a signature (t, σ). Then, $\mathscr{C}$ returns (t, σ) to $\mathscr{A}$.

- **Output.** Finally, $\mathscr{A}$ outputs an identity α, a period index t^* and a corresponding signature (t^*, σ^*).

In the above game, it is also mandated that the following conditions are simultaneously satisfied: ① **Verify**(cp, (t^*, σ^*), m^*, ω') =1; ② $\mathscr{A}$ is disallowed to issue key generation queries for any restricted identity; ③ $\mathscr{A}$ is disallowed to issue temporary private key queries for any restricted identity and the challenged time period t^*; ④ $\mathscr{A}$ is disallowed to issue signing queries for any restricted identity, the challenged time period t^* and message m^*.

3.2.2. Strongly key-insulated security

The strongly key-insulated security for ABPKIS systems says that, even if one of the user's helper keys and some of his temporary signing keys are exposed, it is still impossible for an adversary to obtain all of this user's temporary signing keys.

Formally, for an ABPKIS scheme, its strongly key-insulated security can be defined via the following *strongly*-UF-ID&KE-CMA game between an adversary $\mathscr{A}$ and a challenger $\mathscr{C}$:

- **Init.** The same as a UF-ID&KE-CMA game.
- **Setup.** The same as a UF-ID&KE-CMA game.
- **Query Phase.** The adversary $\mathscr{A}$ adaptively issues a set of queries, such as those given below:

– *Key Generation queries* $\langle\omega\rangle$: The same as a UF-ID&KE-CMA game.

– *Helper Key queries* $\langle\omega\rangle$: $\mathscr{C}$ runs algorithm KeyGen to generate HK_ω and sends it to the adversary.

– *Signing queries* $\langle\omega, t, m\rangle$: The same as a UF-ID&KE-CMA game.

- **Output.** Finally, $\mathscr{A}$ outputs an identity α, a period index t^* and a corresponding signature (t^*, σ^*).

In the above game, it is also mandated that the following conditions are

simultaneously satisfied: ① **Verify**(cp, (t^*, σ^*), m^*, ω') =1; ② $\mathscr{A}$ is disallowed to issue key generation queries for any restricted identity; ③ $\mathscr{A}$ is disallowed to issue signing queries for any restricted identity, the challenged time period t^* and message m^*.

3.2.3. Anonymity

An ABPKIS scheme satisfies the anonymity requirement if no adversary $\mathscr{A}$ can win the following ANONY-ABPKIS game between $\mathscr{A}$ and a challenger $\mathscr{C}$ with a non-negligible advantage:

- $\mathscr{C}$ runs the algorithm **Setup** to generate a master key msk and public parameters cp and sends them to $\mathscr{A}$.
- $\mathscr{A}$ can use the master key msk to generate temporary private keys and signatures.
- $\mathscr{A}$ will next submit a challenge period index t^*, a message m^*, two identities (α_1, α_2) and a challenge identity α, where $\alpha \subseteq (\alpha_1 \cap \alpha_2)$ and $|\alpha| \le d$.
- Assume that $\mathscr{A}$ has issued temporary private key queries $\langle \alpha_1, t^* \rangle$ and $\langle \alpha_2, t^* \rangle$. Let $\mathrm{TK}_{\alpha_1, t^*}$ and $\mathrm{TK}_{\alpha_2, t^*}$ be temporary private keys for (α_1, t^*) and (α_2, t^*), respectively.
- $\mathscr{C}$ flips a random coin, b, computes a signature (t^*, σ^*) =Sign(cp, t^*, m^*, $\mathrm{TK}_{\alpha_b, t^*}$) and sends it to $\mathscr{A}$.
- Finally, $\mathscr{A}$ outputs a guess b' of b by judging whether (t^*, σ^*) is generated from $\mathrm{TK}_{\alpha_1, t^*}$ or $\mathrm{TK}_{\alpha_2, t^*}$.

4. Model of ABPKIS

4.1. Definition

We use a Pseudo-Random Function (PRF) [1] F, such that given a κ-bit seed (index) s and a κ-bit argument (input) x, it outputs a κ-bit string $\mathrm{Fs}(x)$, and defines the Lagrange coefficient

$$\Delta_{i,S}(x) = \prod_{\substack{j \in S \\ j \neq i}} \frac{x-j}{i-j}$$

for $i \in Z_p$ and a set S of elements in Z_p. The proposed ABPKIS scheme consists of the following algorithms:

Setup(κ, l, d): Given a security parameter κ, the length l of some universe U of size $|U|$, and a threshold value d, the PKG works as follows: ① Define the universe U. For simplicity, we can take the first l elements of Z_p^* to be the universe, specifically, the integers 1, …, l (mod p); ② Let G_1 and G_2 be two groups with

prime order p of size, let g be a generator of G_1, let $\hat{e}: G_1 \times G_1 \to G_2$ denotes the bilinear map; ③ Let $H_1: \{0,1\}^* \to \{0,1\}^{n_w}$, $H_2: \{0,1\}^* \to \{0,1\}^{n_m}$ be two collision-resistant hash functions with $n_w, n_m \in Z$; ④ Pick $y \xleftarrow{U} Z_p$ and $g_2 \xleftarrow{U} G_1$, set $g_1 = g^y$; ⑤ Choose a default set of $d-1$ attributes from Z_p, $\Omega = \{l+1, l+2, \ldots, l+d-1\}$; ⑥ Pick a random $(l+d-1)$-length vector $\vec{H} = (h_i)$ whose elements are randomly chosen from G_1; ⑦ Pick $w' \xleftarrow{U} G_1$ and a random n_w-length vector $\vec{W} = (w_i)$ whose elements are randomly chosen from G_1; ⑧ Pick $m' \xleftarrow{U} G_1$ and a random n_m-length vector $\vec{M} = (m_i)$ whose elements are randomly chosen from G_1; ⑨ Output the public parameters and the master secret key as cp = (G_1, G_2, $\hat{e}$, g, g_1, g_2, w', m', $\vec{H}$, $\vec{W}$, $\vec{M}$, Ω), msk = y. For convenience, we define two functions L_1 and L_2, such that $L_1(S_1)=w'\prod_{i\in S_1} w_i$, $L_2(S_2)=m'\prod_{i\in S_2} m_i$. In addition, for a given time period t and a given message m, we hereafter use W_t and $\mathcal{M}_m$ to denote the following sets: $W_t = \{i | a[i] = 1,\ a = H_1(t) \subseteq \{1, \ldots, n_w\}\}$, $\mathcal{M}_m = \{j | b[j] = 1,\ b = H_2(m) \subseteq \subseteq \{1, \ldots, n_m\}\}$.

KeyGen(msk, cp, ω): To generate the helper key and the initial private key for an identity $\omega \subseteq U$, the PKG works as follows: (1) Pick a helper key $\mathrm{HK}_{\omega,0}$, $\mathrm{HK}_{\omega,1} \xleftarrow{U} \{0,1\}^{\kappa}$; (2) A $d-1$ degree polynomial q is randomly chosen such that $q(0) = y$; (3) Generate a new attribute set $\hat{\omega} = \omega \cup \Omega$; (4) For all $i \in \hat{\omega}$, pick $r_i \xleftarrow{U} Z_p$, compute $k_{i,0} = F_{\mathrm{HK}_{\omega,0}}(0 \,\|\, i)$ and $k_{i,-1} = F_{\mathrm{HK}_{\omega,1}}(-1 \,\|\, i)$ (Note that if the length of the input for F is less than κ, then we can add some "0"s as the prefix to meet the length requirement); (5) Define the initial private key as

(1) $$\mathrm{TK}_{\omega,0} = \{(d_{i1,0}, d_{i2,0}, d_{i3,0}, d_{i4,0})\}_{i\in\hat{\omega}} =$$
$$= \{(g_2^{q(i)}(g_1 h_i)^{r_i} L_1(W_{-1})^{k_{i,-1}} L_1(W_0)^{k_{i,0}}, g^{k_{i,-1}}, g^{k_{i,0}}, g^{r_i})\}_{i\in\hat{\omega}}.$$

HelperUpt(cp, t, ω, HK_{ω}): Given the public parameters cp, the period indices t and t', an identity $\omega \subseteq U$, and its helper key HK_{ω}, for each $i \in \hat{\omega}$, where $\hat{\omega} = \omega \cup \Omega$, this algorithm computes $k_{i,t} = F_{\mathrm{HK}_{\omega}}(t \| i)$ and $k_{i,t-2} = F_{\mathrm{HK}_{\omega}}(t-2 \| i)$. Then this algorithm defines and returns the key-update information as follows:

$$\mathrm{UI}_{\omega,t} = \{(\mathrm{UI}_{i1,t}, \mathrm{UI}_{i2,t})\}_{i\in\hat{\omega}} = \left\{\left(\frac{L_1(W_t)^{k_{i,t}}}{L_1(W_{t-2})^{k_{i,t-2}}}, g^{k_{i,t}}\right)\right\}_{i\in\hat{\omega}}.$$

UserUpt(cp, t, ω, $\mathrm{UI}_{\omega,t}$, $\mathrm{TK}_{\omega,t}$): This algorithm is executed by the user with identity $\omega \subseteq U$ and works as follows: (1) Parse the temporary private key for the identity ω and period $t-1$ as $\mathrm{TK}_{\omega,\,t-1} = \{(d_{i1,\,t-1}, d_{i2,\,t-1}, d_{i3,\,t-1}, d_{i4,\,t-1})\}_{i\in\hat{\omega}}$; (2) Parse the key-update information for the identity ω for t as

$\mathrm{UI}_{\omega,t} = \{(\mathrm{UI}_{i1,t}, \mathrm{UI}_{i2,t})\}_{i\in\hat{\omega}\ i\in\hat{\omega}}$; (3) Set the temporary private key for the identity ω and period t $\mathrm{TK}_{\omega,t} = \{(d_{i1,t}, d_{i2,t}, d_{i3,t}, d_{i4,t}\}_{i\in\omega} = \{(\ d_{i1,t-1} \cdot \mathrm{UI}_{i1,t}, d_{i3,t-1}, \mathrm{UI}_{i2,t}, d_{i4,t-1})\}_{\ i\in\hat{\omega}}$; (4) Delete $\mathrm{TK}_{\omega,t-1}$ and $\mathrm{UI}_{\omega,t}$; (5) Return $\mathrm{TK}_{\omega,t}$. Note that if let $\hat{i} = t \bmod 2$ and $\hat{j} = (t-1) \bmod 2$, then $\mathrm{TK}_{\omega,t}$ is always set to be

$$\mathrm{TK}_{\omega,t} = \{(g_2^{q(i)}(g_1h_i)^{r_i} L_1(W_{t-1})^{k_{i,t-1}} L_1(W_t)^{k_{i,t}}, g^{k_{i,t-1}}, g^{k_{i,t}}, g^{r_i})\}_{i\in\hat{\omega}}, \tag{2}$$

where $k_{i,t-1} = F_{\mathrm{HK}_{\omega,\hat{j}}}(t-1\|i)$ and $k_{i,t} = F_{\mathrm{HK}_{\omega,\hat{j}}}(t\|i)$.

Sign(cp, t, m, $\mathrm{TK}_{\omega,t}$): Suppose that the signer has a temporary private key for the attribute set ω and time period t. In period t, the signer can produce the signature on m with the attribute set $\omega' = \{i_1, i_2, \ldots, i_k\} \subseteq \omega$, where $1 \le k \le d$, as follows: (1) Parse $\mathrm{TK}_{\omega,t}$ as $\mathrm{TK}_{\omega,t} = \{(d_{i1,t}, d_{i2,t}, d_{i3,t}, d_{i4,t}\}_{i\in\omega}$; (2) Select a $d-k$ default attribute subset $\Omega' = \{i_{k+1}, i_{k+2}, \ldots, i_d\} \subseteq \Omega$. Then, pick $r_1', r_2', \ldots, r_d', s_1, s_2, \ldots, s_d, c_{1,t-1}, c_{2,t-1}, \ldots, c_{d,t-1}, c_{1,t}, c_{2,t}, \ldots, c_{d,t} \xleftarrow{U} Z_p$, choose a $d-1$ degree polynomial $q'(x)$ such that $q'(0) = 0$; (3) For each $v \in \{1, \ldots, d\}$, compute $\sigma_{v1} = d_{i_v 1,t} g_2^{q'(i_v)} (g_1 h_{i_v})^{r_v'} L_1(W_{t-1})^{c_{v,t-1}} \quad L_1(W_t)^{c_{v,t}} L_2(M_m)^{s_v}$, $\sigma_{v2} = d_{i_v 4,t} g^{r_v'}$, $\sigma_{v3} = g^{s_v}$, $\sigma_{v4} = d_{i_v 2,t-1} g^{c_{v,t-1}}$, $\sigma_{v5} = d_{i_v 2,t} g^{c_{v,t}}$; (4) Output the signature $(t, \sigma) = (t, \{(\sigma_{v1}, \sigma_{v2}, \sigma_{v3}, \sigma_{v4}, \sigma_{v5})\}_{1\le v\le d})$. Note that if we let $\overline{r}_{i_v} = r_{i_v} + r_v'$, $\overline{c}_{v,t-1} = k_{i_v,t-1} + c_{v,t-1}$, $\overline{c}_{v,t} = k_{i_v,t} + c_{v,t}$, choose a $d-1$ default attribute subset $\overline{q}$ such that for each $1 \le v \le d$, $\overline{q}(i_v) = q(i_v) + q'(i_v)$ (note that $\overline{q}(0) = q(0) + q'(0) = y$), the signature (t, σ) is always set to be

$$(t, \{(g_2^{\overline{q}(i_v)}(g_1h_{i_v})^{\overline{r}_{i_v}} L_1(W_{t-1})^{\overline{c}_{v,t-1}} L_1(W_t)^{\overline{c}_{v,t}} L_2(M_m)^{s_v}, g^{\overline{r}_{i_v}}, g^{s_v}, g^{\overline{c}_{v,t-1}}, g^{\overline{c}_{v,t}})\}_{1\le v\le d}). \tag{3}$$

This result can be seen from the following:

$$(\sigma_{v1}, \sigma_{v2}, \sigma_{v3}, \sigma_{v4}, \sigma_{v5}) =$$

$$= (d_{i_v,t}^{\langle 1\rangle} g_2^{q'(i_v)} (g_1h_{i_v})^{r_v'} L_1(W_{t-1})^{c_{v,t-1}} L_1(W_t)^{c_{v,t}} L_2(M_m)^{s_v}, d_{i_v,t}^{\langle 4\rangle} g^{r_v'}, g^{s_v}, d_{i_v,t}^{\langle 2\rangle} g^{\overline{c}_{v,t-1}}, d_{i_v,t}^{\langle 3\rangle} g^{\overline{c}_{v,t}}) =$$

$$= (g_2^{q(i_v)} (g_1h_{i_v})^{r_{i_v}} L_1(W_{t-1})^{k_{i_v,t-1}} L_1(W_t)^{k_{i_v,t}} g_2^{q'(i_v)} (g_1h_{i_v})^{r_v'} L_1(W_{t-1})^{c_{v,t-1}} L_1(W_t)^{c_{v,t}} \times$$

$$\times L_2(M_m)^{s_v}, g^{r_{i_v}} g^{r_v'}, g^{s_v}, g^{k_{i_v,t-1}} g^{\overline{c}_{v,t-1}}, g^{k_{i_v,t}} g^{\overline{c}_{v,t}}) =$$

$$= (g_2^{q(i_v)+q'(i_v)} (g_1h_{i_v})^{r_{i_v}+r_v'} L_1(W_{t-1})^{k_{i_v,t-1}+c_{v,t-1}} L_1(W_t)^{k_{i_v,t}+c_{t,v}} L_2(M_m)^{s_v}, g^{r_{i_v}+r_v'}, g^{s_v}, g^{k_{i_v,t-1}+c_{v,t-1}},$$

$$g^{k_{i_v,t}+c_{v,t}}) =$$

$$= (g_2^{\overline{q}(i_v)} (g_1h_{i_v})^{\overline{r}_{i_v}} L_1(W_{t-1})^{\overline{c}_{v,t-1}} L_1(W_t)^{\overline{c}_{v,t}} L_2(M_m)^{s_v}, g^{\overline{r}_{i_v}}, g^{s_v}, g^{\overline{c}_{v,t-1}}, g^{\overline{c}_{v,t}}).$$

Verify(cp, m, ω', (t, σ)): Let $S = \{i_1, \ldots, i_d\}$. Given the signature (t, σ) for period t, the attribute set $\omega' = \{i_1, \ldots, i_k\}$ and a message m with the default attribute set $\Omega' = \{i_{k+1}, i_{k+2}, \ldots, i_d\}$, the verifier accepts (t, σ) iff the following equality holds:

$$\prod_{v=1}^{d} \left(\frac{\hat{e}(g,\sigma_{v1})}{\hat{e}(g_1h_{i_v},\sigma_{v2})\hat{e}(L_2(M_m),\sigma_{v3})\hat{e}(L_1(W_{t-1}),\sigma_{v4})\hat{e}(L_1(W_t),\sigma_{v5})}\right)^{\Delta_{i_v,S}(0)} = \hat{e}(g_1, g_2).$$

The consistency of this scheme can be explained as follows:

$$\prod_{v=1}^{d}\left(\frac{\hat{e}(g,\sigma_{v1})}{\hat{e}(g_1 h_{i_v},\sigma_{v2})\hat{e}(L_2(M_m),\sigma_{v3})\hat{e}(L_1(W_{t-1}),\sigma_{v4})\hat{e}(L_1(W_t),\sigma_{v5})}\right)^{\Delta_{i_v,S}(0)} =$$

$$=\prod_{v=1}^{d}\left(\frac{\hat{e}(g,g_2^{\overline{q}(i_v)}(g_1 h_{i_v})^{\overline{r}_{i_v}} L_1(W_{t-1})^{\overline{c}_{v,t-1}} L_1(W_t)^{\overline{c}_{v,t}} L_2(M_m)^{s_v})}{\hat{e}(g_1 h_{i_v},g^{\overline{r}_{i_v}})\hat{e}(L_2(M_m),g^{s_v})\hat{e}(L_1(W_{t-1}),g^{\overline{c}_{v,t-1}})\hat{e}(L_1(W_t),g^{\overline{c}_{v,t}})}\right)^{\Delta_{i_v,S}(0)} =$$

$$=\prod_{v=1}^{d}\left(\hat{e}(g,g_2^{\overline{q}(i_v)})\right)^{\Delta_{i_v,S}(0)} = \hat{e}(g,g_2)^{\overline{q}(0)} = \hat{e}(g,g_2)^{y} = \hat{e}(g_1,g_2).$$

4.2. Security

Theorem 1. The proposed ABPKIS scheme is key-insulated in the Selective-ID model, assuming that the CDH assumption holds in group G_1 and the hash function H is collision-resistant. Concretely, if there exists a (T,ε)-UF-ID&KE-CMA adversary $\mathscr{A}$ against our scheme, asking at most q_k (q_t, q_s, respectively) queries to the oracle of key generation queries (temporary private key queries, signature queries, respectively), then there exists an efficient algorithm $\mathscr{B}$ that can solve the (T', ε')-CDH assumption in group G_1 with $T' \le T + O((q_k + q_t + q_s d)t_e + (n_w(q_k + q_t) + +(n_w + n_m)q_s d)t_m)$, $\varepsilon' \ge \dfrac{\varepsilon}{27(n_w+1)^2(n_m+1)(q_t+q_s)^2 q_s\binom{d-k}{d-1}}$, where t_e and t_m denote the running time of an exponentiation and a multiplication in group G_1, respectively, where d is the threshold value, and k is the length of the challenge identity.

P r o o f: Suppose that $\mathscr{B}$ is given a tuple $(g, g^a, g^b) \in G_1^3$ for some unknown $a, b \in Z_p^*$. The task of $\mathscr{B}$ is to compute g^{ab}. $\mathscr{B}$ flips a fair coin $\mathscr{COIN} \in \{1, 2\}$. If $\mathscr{COIN} = 1$, $\mathscr{B}$ plays **Game 1** with $\mathscr{A}$, else he plays **Game 2**.

Game 1. Define the universe U of l elements as $\{1, \ldots, l\}$. For simplicity, let the default set of d–1 attributes be $\Omega = \{l+1, l+2, \ldots, l+d-1\}$. During the initial phase, $\mathscr{B}$ receives the challenge period index t^* and the challenge identity α (an n elements set of members of Z_p^*), where $|\alpha| = k < d$, and d is the threshold value.

Setup. $\mathscr{B}$ generates the public parameters for $\mathscr{A}$ as follows: (1) Choose a random subset $\Omega^* \subseteq \Omega$ with $|\Omega^*| = d - k$; (2) Set $l_w = 2(q_t + q_s)$, $l_m = 2q_s$, and randomly choose two integers k_w and k_m with $0 \le k_w \le n_w$ and $0 \le k_m \le n_m$. We here assume that $l_w(n_w + 1) < p$ and $l_m(n_m + 1) < p$; (3) Randomly choose the following integers:

$$x' \xleftarrow{U} Z_{l_w}, z' \xleftarrow{U} Z_{l_m}, y', u' \xleftarrow{U} Z_p; \beta_i \xleftarrow{U} Z_p, i = 1, \ldots, l + d - 1; \hat{x}_j \xleftarrow{U} Z_{l_w}, j = 1, \ldots, n_w;$$

$$\hat{z}_\tau \xleftarrow{U} Z_{l_m}, \tau = 1, \ldots, n_m; \hat{y}_j \xleftarrow{U} Z_p, j = 1, \ldots, n_w; \hat{u}_\tau \xleftarrow{U} Z_p, \tau = 1, \ldots, n_m;$$

(4) Define a set of public parameters: $g_1 = g^a$, $g_2 = g^b$, $w' = g_2^{-l_w k_w + x'} g^{y'}$,

$m' = g_1^{-l_m k_m + z'} g^{u'}$; $\vec{H} = (h_i)$, $i = 1, \ldots, l+d-1$, with $h_i = g_1^{-1} g^{\beta_i}$ for $\forall\ i \in \alpha \cup \Omega^*$ and $h_i = g^{\beta_i}$ for $\forall\ i \notin \alpha \cup \Omega^*$; $\vec{W} = (w_j)$, with $w_j = g_2^{\hat{x}_j} g^{\hat{y}_j}$, $j = 1, \ldots, n_w$; $\vec{M} = (m_\tau)$, with $m_\tau = g_1^{\hat{z}_\tau} g^{\hat{u}_\tau}$, $\tau = 1, \ldots, n_m$. Finally, give $\mathscr{A}$ the above public parameters. Note that the distribution of these public parameters is identical to the real construction. To make the notation easy to follow, we define four functions J_1, J_2, K_1 and K_2, such that

$$K_1(S_1) = -l_w k_w + x' + \sum_{j \in S_1} \hat{x}_j,\ J_1(S_1) = y' + \sum_{j \in S_1} \hat{y}_j \sum_{j \in S_1} \hat{x}_j,$$
$$K_2(S_2) = -l_m k_m + z' + \sum_{\tau \in S_2} \hat{x}_\tau,\ J_2(S_2) = u' + \sum_{\tau \in S_2} \hat{u}_\tau,$$

where $S_1 \subseteq \{1, \ldots, n_w\}$ and $S_2 \subseteq \{1, \ldots, n_m\}$. Note that the following equalities always hold: $g_1^{K_1(S_1)} g^{J_1(S_1)} = L_1(S_1)$, $g_1^{K_2(S_2)} g^{J_2(S_2)} = L_2(S_2)$. According to Equations (1) and (2), a given user's initial private key and all of his temporary private keys share the same exponent r_i and $q(i)$. To embody these implicit relations, $\mathscr{B}$ forms two lists, called $\mathrm{R}^{\mathrm{list}}$ and $\mathrm{S}^{\mathrm{list}}$, which are initially empty. For easy explanation, an algorithm, called RQuery(ω, i) is defined such that: for a given input $\langle \omega, i \rangle$, if $\mathrm{R}^{\mathrm{list}}$ contains a tuple $(\omega, i, \hat{r})$, then return $\hat{r}$; otherwise, pick $\hat{r} \overset{U}{\leftarrow} Z_p$, add $(\omega, i, \hat{r})$ to $\mathrm{R}^{\mathrm{list}}$, and return $\hat{r}$. In addition, an algorithm called SQuery(ω, i) is defined such that: for a given input $\langle \omega, i \rangle$, if $\mathrm{S}^{\mathrm{list}}$ contains a tuple $(\omega, i, \hat{s})$, then return $\hat{s}$; otherwise, pick $\hat{s} \overset{U}{\leftarrow} Z_p$, add $(\omega, i, \hat{s})$ to $\mathrm{S}^{\mathrm{list}}$, and return $\hat{s}$.

Query Phase. $\mathscr{B}$ answers a series of oracle queries for $\mathscr{A}$ in the following way:

Help key queries: $\mathscr{B}$ maintains a list $\mathrm{HK}^{\mathrm{list}}$ that is initially empty. Suppose that $\mathscr{A}$ requests a helper key and an initial private key for the identity ω. $\mathscr{B}$ first checks whether $\mathrm{HK}^{\mathrm{list}}$ has contained a tuple for this input. If yes, the predefined value is returned to $\mathscr{A}$. Otherwise, it picks $\mathrm{HK}_\omega \overset{U}{\leftarrow} \{0, 1\}^\kappa$. Next, it adds the tuple $(\omega, \mathrm{HK}_\omega)$ to the list $\mathrm{HK}^{\mathrm{list}}$ and returns HK_ω to $\mathscr{A}$. $\mathscr{B}$ first checks whether $\mathrm{HK}^{\mathrm{list}}$ has contained a tuple for this input. If yes, the predefined value is returned to $\mathscr{A}$. Otherwise, it picks $\mathrm{HK}_\omega \overset{U}{\leftarrow} \{0, 1\}^\kappa$. Next, it adds the tuple $(\omega, \mathrm{HK}_\omega)$ to the list $\mathrm{HK}^{\mathrm{list}}$ and returns HK_ω to $\mathscr{A}$.

Key generation queries: We define three sets Γ, Γ', S, as follows: (1) Let $\Gamma = \omega \cap \alpha$; (2) Let Γ' be any set such that $\Gamma' \subset \alpha$ and $|\Gamma'| = d-1$; and (3) Let $S = \Gamma' \cup \{0\}$. Next, we define the initial private key components as follows: (1) A $d-1$ degree polynomial q is randomly chosen such that $q(0) = a$; (2) Generate a new attribute set $\hat{\omega} = \omega \cup \Omega$, and for each $i \in \hat{\omega}$, compute $k_{i,0} = F_{\mathrm{HK}_\omega}(0 \| i)$; (3) For each $i \in \Gamma'$, compute $r_i = \mathrm{RQuery}(\omega, i)$ and $s_i = \mathrm{SQuery}(\omega, i)$, let $q(i) = s_i$, and according to (1), set the initial private key components to be $(g_2^{s_i} (g_1 h_i)^{r_i} L_1(W_0)^{k_{i,0}}, g^{k_{i,0}}, g^{r_i})$; (4) For each $i \in \hat{\omega} - \Gamma'$, compute $r_i' = \mathrm{RQuery}(\omega, i)$, and set the initial private key

components to be $(g_2^{\sum_{j\in\Gamma'} q(j)\cdot\Delta_{j,S}(i)}\ g_2^{-\beta_i\cdot\Delta_{0,S}(i)}\ (g_1h_i)^{r_i'}\ L_1(W_0)^{k_{i,0}}, g^{k_{i,0}}, g_2^{-\Delta_{0,S}(i)}g^{r_i'})$.

Temporary private key queries: We require that $\mathscr{A}$ only queries on the restricted identity ω^* where $\alpha\subseteq\omega^*$. Consider that $\mathscr{A}$ receives a temporary private key query $\langle\omega, t\rangle$. Upon receiving a temporary private key query $\langle\omega, t\rangle$, $\mathscr{B}$ outputs "failure" and aborts if $K_1(W_t) \equiv 0 \bmod p$ holds (denote this event by **E1**). Otherwise, we define three sets Γ, Γ', S in the same way as *Key generation queries*. Next, we define the temporary private key components as follows. ① A d–1 degree polynomial q is randomly chosen such that $q(0) = a$; ② Generate a new attribute set $\hat{\omega} = \omega\cup\Omega$; ③ For each $i \in \Gamma'$, compute $r_i = \mathrm{RQuery}(\omega, i)$ and $s_i = \mathrm{SQuery}(\omega, i)$, pick $k_{i,t} \xleftarrow{U} Z_p^*$, let $q(i) = s_i$, and according to Equation (2), set the temporary private key components to be

$$(g_2^{q(i)}(g_1h_i)^{r_i}L_1(W_{t-1})^{k_{i,t-1}}L_1(W_t)^{k_{i,t}}, g^{k_{i-1,t}}, g^{k_{i,t}}, g^{r_i});$$

④ For each $i \in \hat{\omega}-\Gamma'$, compute $r_i = \mathrm{RQuery}(\omega, i)$, pick $k'_{i,t} \xleftarrow{U} Z_p^*$, and set the temporary private key components to be

$$(g_2^{\sum_{j\in\Gamma'} q(j)\cdot\Delta_{j,S}(i)}\ g_2^{-\frac{J_1(W_t)\cdot\Delta_{0,S}(i)}{K_1(W_t)}}\ (g_1h_i)^{r_i}\ L_1(W_{t-1})^{k_{i,t-1}}\ L_1(W_t)^{k'_{i,t}}, g^{k_{i,t-1}}, g_2^{-\frac{\Delta_{0,S}(i)}{K_1(W_t)}}g^{k'_{i,t}}, g^{r_i}).$$

Signing queries: Suppose $\mathscr{B}$ receives a signature query $\langle\omega, t, m\rangle$. If $\alpha\not\subseteq\omega$, then $\mathscr{B}$ can generate a simulated temporary private key for $\langle\omega, t\rangle$ to be a *Temporary private key query* and obtain a signature for $\langle\omega, t\rangle$ on message m, normally. If $(\alpha\subseteq\omega)\wedge(K_1(W_t) \equiv K_2(\mathcal{M}_m) \equiv 0 \bmod p)$ holds, then $\mathscr{B}$ outputs "failure" and aborts (denote this event by **E3**). Otherwise, $\mathscr{B}$ constructs the signature for $\mathscr{A}$ according to two cases

Case 1. If $(\alpha\subseteq\omega)\wedge(K_1(W_t) \not\equiv 0 \bmod p) \wedge (K_2(\mathcal{M}_m) \equiv 0 \bmod p)$, then $\mathscr{B}$ chooses a random (d-$|\omega|$)-element subset Γ' from Γ. Assume that $\omega\cup\Gamma = \{i_1, \ldots, i_d\}$. A d–1 degree polynomial $\overline{q}$ is randomly chosen such that $\overline{q}(0) = a$. For each $k \in \{1,\ldots, d-1\}$, pick τ_k, $\overline{r}_{i_k}, \overline{c}_{k,t}, s_k \xleftarrow{U} Z_p$, set $\overline{q}(i_k) = \tau_k$, and according to (3), set the signature components to be $(g_2^{\overline{q}(i_v)}(g_1h_{i_v})^{\overline{r}_{i_v}}L_1(W_{t-1})^{\overline{c}_{v,t-1}}L_1(W_t)^{\overline{c}_{v,t}}L_2(M_m)^{s_v}, g^{\overline{r}_{i_v}}, g^{s_v}, g^{\overline{c}_{v,t-1}}, g^{\overline{c}_{v,t}})$. For $k = d$, pick $\overline{r}_{i_d}$, $\overline{c}_{d,t-1}, \overline{c}'_{d,t}$, $s_d \xleftarrow{U} Z_p$, and set the signature components to be

$$\left((g_1h_{i_d})^{\overline{r}_{i_d}}\ g_2^{\sum_{i=1}^{d-1}\Delta_{i_d,S}(i_k)\overline{q}(i_k)}\ g_2^{-\frac{J_1(W_t)\Delta_{i_d,S}(0)}{K_1(W_t)}}\ L_1(W_{t-1})^{\overline{c}_{v,t-1}}\ L_1(W_t)^{\overline{c}'_{d,t}}\ L_2(M_m)^{s_d},\ g^{\overline{r}_{i_k}}, g^{s_d}, g^{\overline{c}_{v,t-1}},\right.$$

$$\left. g^{\overline{c}'_{d,t}-\frac{b\cdot\Delta_{i_d,S}(0)}{K_1(W_t)}}\right).$$

Case 2. If $(\alpha \subseteq \omega) \wedge (K_1(W_t) \equiv 0 \bmod p) \wedge (K_2(\mathcal{M}_m) \not\equiv 0 \bmod p)$, $\mathcal{B}$ chooses a random $(d-|\omega|)$-element subset Γ' from Γ. Assume that $\omega \cup \Gamma' = \{i_1, \ldots, i_d\}$. A $d-1$ degree polynomial $\overline{q}$ is randomly chosen such that $\overline{q}(0) = a$. For each $k \in \{1,\ldots, d-1\}$, pick τ_k, $\overline{r}_{i_k}, \overline{c}_{k,t}$, $s_k \overset{U}{\leftarrow} Z_p$, set $\overline{q}(i_k) = \tau_k$, and according to (3), set the signature components to be

$$(g_2^{\overline{q}(i_v)}(g_1 h_{i_v})^{\overline{r}_{i_v}} L_1(W_{t-1})^{\overline{c}_{v,t-1}} L_1(W_t)^{\overline{c}_{v,t}} L_2(M_m)^{s_v}, g^{\overline{r}_{i_v}}, g^{s_v}, g^{\overline{c}_{v,t-1}}, g^{\overline{c}_{v,t}}\ g^{\overline{c}_{k,t}}).$$

For $k = d$, pick $\overline{r}_{i_d}$, $\overline{c}_{d,t}$, $s'_d \overset{U}{\leftarrow} \mathbb{Z}_p$, and set the signature components to be

$$\left((g_1 h_{i_d})^{\overline{r}_{i_d}} g_2^{\sum_{i=1}^{d-1} \Delta_{i_d,S}(i_k)\overline{q}(i_k)} g_2^{-\frac{J_2(M_m)\Delta_{i_d,S}(0)}{K_2(M_m)}} L_1(W_{t-1})^{\overline{c}_{v,t-1}} L_1(W_t)^{\overline{c}_{d,t}}\ L_2(M_m)^{s'_d},\ g^{\overline{r}_{i_k}},\right.$$

$$\left. g^{s'_d - \frac{\Delta_{i_d,S}(0)}{K_2(M)}}, g^{\overline{c}_{v,t-1}}, g^{\overline{c}_{d,t}} \right).$$

Output. Finally, the adversary outputs a forged signature $(t^*, \sigma^*) = (t^*, \{\sigma_v^{\langle 1\rangle *},\ \sigma_v^{\langle 2\rangle *}, \sigma_v^{\langle 3\rangle *}, \sigma^{\langle 4\rangle *}, \sigma^{\langle 5\rangle *}\}_{1\le v\le d})$ on message m^* for α and t^* with a default attribute subset $\overline{\Omega}^*$. If $(K_1(W_{t^*-1}) \equiv K_1(W_{t^*}) \equiv K_2(\mathcal{M}_{m\ *}) \equiv 0 \bmod p) \wedge \wedge(\overline{\Omega}^* = \Omega^*)$ does not hold, then $\mathcal{B}$ outputs "failure" and aborts (denote this event by **E4**). Otherwise, we have

$$\prod_{v=1}^{d}\left(\frac{\hat{e}(g,\sigma_v^{\langle 1\rangle *})}{\hat{e}(g_1 h_{i_v},\sigma_v^{\langle 2\rangle *})\hat{e}(L_2(M_{m\ *}),\sigma_v^{\langle 3\rangle *})\hat{e}(L_1(W_{t\ *-1}),\sigma_v^{\langle 4\rangle *}\hat{e}(L_1(W_{t\ *}),\sigma_v^{\langle 5\rangle *})}\right)^{\Delta_{i_v,S}(0)} =$$

$$=\prod_{v=1}^{d}\left(\frac{\hat{e}(g,\sigma_v^{\langle 1\rangle *})}{\hat{e}(g_1 g_1^{-1} g^{\beta_{i_v}},\sigma_v^{\langle 2\rangle *})\hat{e}(g_1^{K_2(M_{m\ *})} g^{J_2(M_{m\ *})},\sigma_v^{\langle 3\rangle *})\hat{e}(g_1^{K_1(W_{t\ *-1}^*)} g^{J_1(W_{t\ *-1}^*)},\sigma_v^{\langle 4\rangle *})\hat{e}(g_1^{K_1(W_{t\ *}^*)} g^{J_1(W_{t\ *}^*)},\sigma_v^{\langle 5\rangle *})}\right)^{\Delta_{i_v,S}(0)} =$$

$$=\prod_{v=1}^{d}\left(\frac{\hat{e}(g,\sigma_v^{\langle 1\rangle *})}{\hat{e}(g^{\beta_{i_v}},\sigma_v^{\langle 2\rangle *})\hat{e}(g^{J_2(M_{m\ *}^*)},\sigma_v^{\langle 3\rangle *})\hat{e}(g^{J_1(W_{t\ *-1}^*)},\sigma_v^{\langle 4\rangle *})\hat{e}(g^{J_1(W_{t\ *}^*)},\sigma_v^{\langle 5\rangle *})}\right)^{\Delta_{i_v,S}(0)} =$$

$$=\prod_{v=1}^{d}\left(\frac{\hat{e}(g,\sigma_v^{\langle 1\rangle *})}{\hat{e}(g,\sigma_v^{\langle 2\rangle * \beta_{i_v}})\hat{e}(g,\sigma_v^{\langle 3\rangle * J_2(M_{m\ *}^*)})\hat{e}(g,\sigma_v^{\langle 4\rangle * J_1(W_{t\ *-1}^*)})\hat{e}(g,\sigma_v^{\langle 5\rangle * J_1(W_{t\ *}^*)})}\right)^{\Delta_{i_v,S}(0)} =$$

$$=\prod_{v=1}^{d}\hat{e}\left(\frac{\sigma_v^{\langle 1\rangle *}}{\sigma_v^{\langle 2\rangle * \beta_{i_v}}\sigma_v^{\langle 3\rangle * J_2(M_{m\ *}^*)}\sigma_v^{\langle 4\rangle * J_1(W_{t\ *-1}^*)}\sigma_v^{\langle 5\rangle * J_1(W_{t\ *}^*)}}, g\right)^{\Delta_{i_v,S}(0)} =$$

$$=\hat{e}\left(\prod_{v=1}^{d}\left(\frac{\sigma_v^{\langle 1\rangle *}}{\sigma_v^{\langle 2\rangle * \beta_{i_v}}\sigma_v^{\langle 3\rangle * J_2(M_{m\ *}^*)}\sigma_v^{\langle 4\rangle * J_1(W_{t\ *-1}^*)}\sigma_v^{\langle 5\rangle * J_1(W_{t\ *}^*)}}\right)^{\Delta_{i_v,S}(0)}, g\right),$$

and

$$\prod_{v=1}^{d}\left(\frac{\hat{e}(g,\sigma_v^{\langle 1\rangle *})}{\hat{e}(g_1 h_{i_v},\sigma_v^{\langle 2\rangle *})\hat{e}(L_2(M_{m^*}),\sigma_v^{\langle 3\rangle *})\hat{e}(L_1(W_{t^*-1}),\sigma_v^{\langle 4\rangle *}\hat{e}(L_1(W_{t^*}),\sigma_v^{\langle 5\rangle *})}\right)^{\Delta_{i_v,S}(0)} =$$

$$=\prod_{v=1}^{d}\left(\frac{\hat{e}(g,g_2^{\overline{q}(i_v)}(g_1 h_{i_v})^{\overline{f}_{i_v}} L_1(W_{t^*-1})^{\overline{c}_{v,t^*-1}} L_1(W_{t^*})^{\overline{c}_{v,t^*}} L_2(M_{m^*})^{s_v})}{\hat{e}(g_1 h_{i_v},g^{\overline{f}_{i_v}})\hat{e}(L_2(M_{m^*}),g^{s_v})\hat{e}(L_1(W_{t^*-1}),g^{\overline{c}_{v,t^*-1}})\hat{e}(L_1(W_{t^*}),g^{\overline{c}_{v,t^*}})}\right)^{\Delta_{i_v,S}(0)} =$$

$$=\prod_{v=1}^{d}\hat{e}(g,g_2^{\overline{q}(i_v)})^{\Delta_{i_v,S}(0)}=\hat{e}(g,g_2)^{\prod_{v=1}^{d}\overline{q}(i_v)\cdot\Delta_{i_v,S}(0)}=\hat{e}(g,g_2)^{\overline{q}(0)}=\hat{e}(g,g^b)^a=\hat{e}(g^{ab},g).$$

$$\hat{e}\left(\prod_{v=1}^{d}\left(\frac{\sigma_v^{\langle 1\rangle *}}{\sigma_v^{\langle 2\rangle *\beta_{i_v}}\sigma_v^{\langle 3\rangle *J_2(M_{m^*})}\sigma_v^{\langle 4\rangle *J_1(W_{t^*-1})}\sigma_v^{\langle 5\rangle *J_1(W_{t^*})}}\right)^{\Delta_{i_v,S}(0)},g\right)=\hat{e}(g^{ab},g).$$

Then, $\mathscr{B}$ can successfully compute g^{ab} as

$$\prod_{v=1}^{d}\left(\frac{\sigma_v^{\langle 1\rangle *}}{\sigma_v^{\langle 2\rangle *\beta_{i_v}}\sigma_v^{\langle 3\rangle *J_2(M_{m^*})}\sigma_v^{\langle 4\rangle *J_1(W_{t^*-1})}\sigma_v^{\langle 5\rangle *J_1(W_{t^*})}}\right)^{\Delta_{i_v,S}(0)}=g^{ab}.$$

Game 2. In this game, $\mathscr{B}$ acts as a challenger expecting that $\mathscr{A}$ will corrupt exactly one of the helper keys on the challenged identity. $\mathscr{B}$ picks $\eta \overset{U}{\leftarrow} \{0, 1\}$ and bets on that $\mathscr{A}$ queries on the η-th helper. We assume $\eta = 1$, the case of $\eta = 0$ can be handled in a similar manner). $\mathscr{B}$ provides the simulation of Setup, *Key generation queries*, *Help key queries* and *Signing queries* for $\mathscr{A}$ in the same way as **Game 1.** $\mathscr{B}$ provides *Temporary Private Key queries* for $\mathscr{A}$ as follows:

Temporary Private Key queries: we explain how to deal with the case of an even t (the case of an odd t can be handled in a similar manner). Since $\mathscr{A}$ does not know $HK_{\omega,0,}$ $\mathscr{B}$ can compute $k_{i,t-1}$ by KQuery. The other steps are the same as *Temporary Private Key queries* of **Game 1**.

We can see that $\mathscr{B}$'s running time is bounded by $T' \le T + O((q_k + q_t + q_s d)t_e +$ $+ (n_w(q_k + q_t) + (n_w + n_m)q_s d)t_m)$. The probability analysis is similar to [4]. The advantage of $\mathscr{B}$ can be bounded by $\varepsilon' \ge \dfrac{\varepsilon}{27(n_w+1)^2(n_m+1)(q_t+q_s)^2 q_s\binom{d-k}{d-1}}$.

Theorem 2. The proposed ABPKIS scheme is key-insulated in the Selective-ID model, assuming that the CDH assumption holds in group G_1 and the hash function H is collision-resistant. Concretely, if there exists a (T,ε)-strongly-UF-ID&KE-CMA adversary $\mathscr{A}$ against our scheme, asking at most q_k (q_h, q_s respectively) queries to the oracle of key generation queries (helper key queries, signature queries respectively), then there exists an efficient algorithm $\mathscr{B}$ that can solve the (T', ε')-CDH assumption in group G_1 with $T' \le T + O((q_k + q_s d)t_e + (n_w q_k +$ $+ (n_w + n_m)q_s d)t_m)$, $\varepsilon' \ge \dfrac{\varepsilon}{27(n_w+1)^2(n_m+1)(q_t+q_s)^2 q_s^2\binom{d-k}{d-1}}$, where t_e, t_m, d, k denote the same quantities as in Theorem 1.

P r o o f: The proof is the same as that of Theorem 1 except that: the temporary private key queries are no longer provided to $\mathscr{A}$.

Theorem 3. The proposed ABPKIS scheme satisfies anonymity.

P r o o f; First, the challenger $\mathscr{C}$ runs the algorithm Setup to obtain the public parameters *cp* and the master secret key *msk* = *y*. $\mathscr{C}$ also gives *cp* and *msk* = *y* to the adversary $\mathscr{A}$. After these interactions, the adversary outputs two identities ω_1^* and ω_2^*, where $\overline{\omega}^* = \omega_1^* \cap \omega_2^*$. Note that the temporary private key for each user should include the (d–1)-element default attribute set Γ. Let $\widehat{\omega}_b^* = \omega_b^* \cup \Omega$ for $b \in \{0, 1\}$. Assume that $\mathscr{C}$ or $\mathscr{A}$ has generated the temporary private key $\mathrm{TK}_{\omega_1^*,t^*} = \{(d^1_{i1,t^*}, d^1_{i2,t^*}, d^1_{i3,t^*}, d^1_{i4,t^*})\}_{i\in\omega_1^*}$ for (ω_1^*, t^*) and $\mathrm{TK}_{\omega_2^*,t^*} = \{(d^2_{i1,t^*}, d^2_{i2,t^*}, d^2_{i3,t^*}, d^2_{i4,t^*})\}_{i\in w_2^*}$ for (ω_2^*, t^*). For each $i \in \widehat{\omega}_\theta^*$, let

$$(d^\theta_{i1,t^*}, d^\theta_{i2,t^*}, d^\theta_{i3,t^*}, d^\theta_{i4,t^*}) = (g_2^{q_\theta(i)} (g_1 h_i)^{r_i^\theta} L_1(W_{t^*-1})^{k^\theta_{i,t^*-1}} L_1(W_{t^*})^{k^\theta_{i,t^*}}, g^{k^\theta_{i,t^*-1}}, g^{k^\theta_{i,t^*}}, g^{r_i^\theta}),$$

where $\theta \in \{0, 1\}$, $r_i^\theta, k^\theta_{i,t^*}, k^\theta_{i,t^*-1} \xleftarrow{U} Z_p$, q_θ is a (d–1) degree polynomial such that $q_\theta(0) = y$.

Then $\mathscr{A}$ outputs the period index t^*, message m^* and a k-element subset $\omega^* = \{i_1, \ldots, i_k\} \subseteq \overline{\omega}^*$, where $|\omega^*| \le d$. $\mathscr{A}$ asks $\mathscr{C}$ to generate a signature on message m^* with respect to ω^* and t^* from either $\mathrm{TK}_{\omega_1^*,t^*}$ or $\mathrm{TK}_{\omega_2^*,t^*}$. $\mathscr{C}$ picks $b \xleftarrow{U} \{0, 1\}$ and a (d–k)-element subset $\Omega' = \{i_{k+1}, i_{k+2}, \ldots, i_d\} \subseteq \Omega$. Then $\mathscr{C}$ runs algorithm Sign(cp, t^*, m^*, $\mathrm{TK}_{\omega_b^*,t^*}$) to output a signature

$$(t^*, \{(d^b_{i_v 1,t^*} g_2^{q'(i_v)} (g_1 h_{i_v})^{r'_v} L_1(W_{t^*-1})^{c_{v,t^*-1}} L_1(W_{t^*})^{c_{v,t^*}} L_2(M_m)^{s_v}, d^b_{i_v 4,t} g^{r'_v}, g^{s_v},$$
$$d^b_{i_v 2,t^*} g^{c_{v,t^*-1}}, d^b_{i_v 3,t^*} g^{c_{v,t^*}})\}_{v\in\omega^*\cup\Omega'}),$$

where $\mathrm{TK}_{\omega_b^*,t^*} = \{(d^b_{i0,t^*}, d^b_{i1,t^*}, d^b_{i2,t^*}, d^b_{i3,t^*})\}_{i\in\omega_b^*}$, r'_v, s_v, c_{v,t^*-1}, $c_{v,t^*} \xleftarrow{U} Z_p$, q' is a d–1 degree polynomial function with $q'(0) = 0$. The signature (t^*, σ^*) could be generated from either $\mathrm{TK}_{\omega_1^*,t^*}$ or $\mathrm{TK}_{\omega_2^*,t^*}$. If $b = 1$, the signature (t^*, σ^*) from $\mathrm{TK}_{\omega_1^*,t^*}$ is

$$(t^*, \{(d^1_{i_v 1,t^*} g_2^{q'(i_v)} (g_1 h_{i_v})^{r'_v} L_1(W_{t^*-1})^{c_{v,t^*-1}} L_1(W_{t^*})^{c_{v,t^*}} L_2(M_m)^{s_v}, d^1_{i_v 4,t} g^{r'_v}, g^{s_v},$$
$$d^1_{i_v 2,t^*} g^{c_{v,t^*-1}}, d^1_{i_v 3,t^*} g^{c_{v,t^*}})\}_{v\in\omega^*\cup\Omega'}).$$

We prove that this signature could be generated from $\mathrm{TK}_{\omega_2^*,t^*}$ in four steps.

(1) From the construction of a temporary private key, we have

$$\frac{d^1_{i1,t^*}}{d^2_{i1,t^*}} = \frac{g_2^{q1(i)} (g_1 h_i)^{r_i^1} L_1(\mathcal{W}_{t^*})^{k^1_{i,t^*}}}{g_2^{q2(i)} (g_1 h_i)^{r_i^2} L_1(\mathcal{W}_{t^*})^{k^2_{i,t^*}}} = g_2^{q_1(i)-q_2(i)} (g_1 h_i)^{r_i^1 - r_i^2} L_1(\mathcal{W}_t^*)^{k^1_{i,t^*} - k^2_{i,t^*}},$$

$$\frac{d^1_{i2,t^*}}{d^2_{i2,t^*}} = \frac{g^{k^1_{i,t^*}}}{g^{k^2_{i,t^*}}} = g^{k^1_{i,t^*} - k^2_{i,t^*}}, \quad \frac{d^1_{i3,t^*}}{d^2_{i3,t^*}} = \frac{g^{r^1_i}}{g^{r^2_i}} = g^{r^1_i - r^2_i}.$$

(2) A d–1 degree polynomial $\overline{q}$ is randomly chosen such that $\overline{q}$ (0) = 0.

(3) Then we have

$$(d^1_{i_v 1,t^*}\, g_2^{q'(i_v)}\, (g_1 h_{i_v})^{r'_v}\, L_1(W_{t^*-1})^{c_{v,t^*-1}} L_1(W_{t^*})^{c_{v,t^*}}\, L_2(M_m)^{s_v}, d^1_{i_v 4,t}\, g^{r'_v}, g^{s_v},$$

$$d^1_{i_v 2,t^*}\, g^{c_{v,t^*-1}}, d^1_{i_v 3,t^*}\, g^{c_{v,t^*}})=$$

$$=(d^2_{i_v 1,t^*}\, g_2^{q_1(i_v)-q_2(i_v)}\, (g_1 h_{i_v})^{r^1_{i_v}-r^2_{i_v}}\, L_1(W_{t^*-1})^{k^1_{i_v,t^*-1}-k^2_{i_v,t^*-1}}\, L_1(W_{t^*})^{k^1_{i_v,t^*}-k^2_{i_v,t^*}}\, g_2^{q'(i_v)}\, (g_1 h_{i_v})^{r'_v} \times$$

$$\times L_1(W_{t^*-1})^{c_{v,t^*-1}}\, L_1(W_{t^*})^{c_{v,t^*}}\, L_2(M_m)^{s_v}, d^2_{i_v 4,t} g^{r^1_{i_v}-r^2_{i_v}} g^{r'_v}, g^{s_v}, d^2_{i_v 2,t} g^{k^1_{i_v,t^*-1}-k^2_{i_v,t^*-1}} g^{c_{v,t^*-1}},$$

$$d^2_{i_v 3,t} g^{k^1_{i_v,t^*}-k^2_{i_v,t^*}} g^{c_{v,t^*}})=$$

$$=(d^2_{i_v 1,t^*}\, g_2^{q'(i_v)}\, (g_1 h_{i_v})^{r'_v}\, L_1(W_{t^*-1})^{c_{v,t^*-1}} L_1(W_{t^*})^{c_{v,t^*}}\, L_2(M_m)^{s_v}, d^2_{i_v 4,t}\, g^{r'_v}, g^{s_v},$$

$$d^2_{i_v 2,t^*}\, g^{c_{v,t^*-1}}, d^2_{i_v 3,t^*}\, g^{c_{v,t^*}}).$$

(4) A d–1 degree polynomial q'' is randomly chosen such that $q''(x) = q_1(x) - - q_2(x) + q''(x)$. Then we have $q''(0) = 0$, $q''(i_v) = q_1(i_v) - q_2(i_v) + q'(i_v)$. Let $r''_v = r^1_{i_v} - - r^2_{i_v} + r'_v, c'_{v,t^*-1} = k^1_{i_v,t^*-1} - k^2_{i_v,t^*-1} + c_{v,t^*-1}, c'_{v,t^*} = k^1_{i_v,t^*} - k^2_{i_v,t^*} + c_{v,t^*}$. Then, the signature (t^*, σ^*) could be rewritten as

$$(t^*, \{(d^2_{i_v 1,t^*}\, g_2^{q'(i_v)}\, (g_1 h_{i_v})^{r'_v}\, L_1(W_{t^*-1})^{c_{v,t^*-1}} L_1(W_{t^*})^{c_{v,t^*}}\, L_2(M_m)^{s_v}, d^2_{i_v 4,t}\, g^{r'_v}, g^{s_v},$$

$$d^2_{i_v 2,t^*}\, g^{c_{v,t^*-1}}, d^2_{i_v 3,t^*}\, g^{c_{v,t^*}})\}_{v \in \omega^* \cup \Omega'}),$$

which is a valid signature generated from $\mathrm{TK}_{\omega_2^*,t^*}$.

Similarly, a signature (t^*, σ^*) from $\mathrm{TK}_{\omega_2^*,t^*}$ can also be generated from $\mathrm{TK}_{\omega_1^*,t^*}$. From the proof, it has been shown that the proposed ABPKIS scheme satisfies unconditional anonymity.

5. Conclusion

We introduce the notion of an Attribute-Based Parallel Key Insulated Signature (ABPKIS) and describe a construction that is based on an Attribute-Based Signature (ABS).

Acknowledgements: This work is supported by NSFC under Grant No 61133014.

References

1. Chen, J., Y. Long, K. Chen, G. J. Hook. Attribute-Based Key-Insulated Signature and Its Applications. – Information Sciences, Vol. **275**, 2014, pp. 57-67.
2. Dodis, Y., J. Katz, S. Xu, M. Yung. Key-Insulated Public-Key Cryptosystem. – In: Proc. of 21th Annual International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt), 28 April-2 May 2002.
3. Dodis, J. K., S. Xu, M. Yung. Key-Insulated Public-Key Cryptosystem. – In: Proc. of 9th International Conference on Theory and Practice in Public-Key Cryptography (PKC), 24-26 April 2006.
4. Li, J., K. Kim. Hidden Attribute-Based Signatures without Anonymity Revocation. – Information Sciences, Vol. **180**, 2010, No 9, pp. 1681-1689.
5. Li, J., M. H. Au, W. Susilo, D. Xie, K. Ren. Attribute-Based Signature and Its Applications. – In: Proc. of 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS), 13-16 April 2010.
6. Libert, B., J. Quisquater, M. Young. Parallel Key-Insulated Public Key Encryption without Random Oracles. – In: Proc. of 10th International Conference on Theory and Practice in Public-Key Cryptography (PKC), 16-20 April 2007.
7. Shahandashti, S. F., R. Safavi-Naini. Threshold Attribute-Based Signatures and Their Application to Anonymous Credential Systems. – In: Proc. of 2rd International Conference on Cryptology in Africa (AFRICACRYPT), 21-25 June 2009.
8. Weng, J., S. Liu, K. Chen, C. Ma. Identity-Based Parallel Key-Insulated Encryption without Random Oracles: Security Notions and Construction. – In: Proc. of 7th International Conference on Cryptology in India (INDOCRYPT), 11-16 December 2006.
9. Weng, S. J., S. Liu, K. Chen, X. Li. Identity-Based Parallel Key-Insulated Signature: Framework and Construction. – Journal of Research and Practice in Information Technology, Vol. **40**, 2008, No 1, pp. 55-68.
10. . Weng, X. Li, K. Chen, S. Liu. Identity-Based Parallel Key-Insulated Signature without Random Oracles. – Journal of Information Science and Engineering, Vol. **24**, 2008, No 4, pp. 1143-1157.