Advance Access publication on 11 November 2013
doi:10.1093/comjnl/bxt132

Efficient Identity-Based Proxy Signature in the Standard Model

KE GU[1,2]* WEIJIA JIA[3] AND CHUNLIN JIANG[1]

[1]*School of Information Science and Engineering, Central South University, Changsha 410083, China*
[2]*School of Computer & Communication Engineering, Changsha University of Science & Technology, Changsha 410004, China*
[3]*Department of Computer Science, City University of Hong Kong, Hong Kong, China*
**Corresponding author: gk4572@163.com*

Presently, many identity-based proxy signature (IBPS) schemes have been proposed, but most of them are constructed in the random oracle model. Also, the proposed security model for IBPS is not enough complete according to Boldyreva's work. Cao and Cao proposed an IBPS scheme in the standard model. However, their scheme is not secure because of not resisting the attack of delegator and requires more computation cost. In this paper, we present a framework for IBPS and show a detailed security model for IBPS. Under our framework, we present an efficient IBPS scheme in the standard model. Compared with other IBPS schemes, the proposed scheme has more complete security and is more efficient.

Keywords: proxy signature; identity-based cryptography; identity-based proxy signature; security model; efficiency

Received 24 July 2012; revised 23 August 2013
Handling editor: Jong Hyuk Park

1. INTRODUCTION

Proxy signature (PS) is a practical cryptographic primitive [1], which is a variant of ordinary signature [2–4]. Compared with ordinary signature schemes, PS schemes have four security properties [1, 5], which are unforgeability, non-repudiation, strong identifiability and prevention of misuse. A provable security model for PS was first proposed by Boldyreva *et al.* [1]. Then many researchers and scholars proposed some improved security models for PS [6–9], which are based on Boldyreva's model [1]. Boldyreva *et al.* [1] described what the security standards for PS are, and proposed a provable security model for PS which is based on register key model. Although their security model provides a strict standard for proving security of a PS scheme, the concept of warrant[1] is not introduced into their model. Because the warrant is considered the important information involving signing rights, their security model is not complete. Malkin *et al.* [7] proposed a provable security model for hierarchical PS. Although the concept of warrant is introduced into their security model [7], their model is very complicated and is not suitable for proving the security of a PS scheme. In Schuldt *et al.* [8], further strengthen the security model for hierarchical PS by considering exposure arbitrary proxy signing keys (not just self-delegated proxy keys). However, their security model is also very complicated. In [9–11], the security models for PS defined three types of attacks. However, the three types of attacks lack the logical relationship, so whether the three types of attacks can completely cover all attacks still needs to be studied in PS schemes. Also, the scheme proposed by Zhang and Mao [11] is not secure because of not resisting the attack of original signer by substituting public-key. Fuchsbauer *et al.* [6] proposed a generalized provable security model for group signature and PS, which defines the security of signing rights based on the mixed concept of group signature and PS. But their model is still not complete. Recently, Boldyreva *et al.* summarized related work about the provable security models for PS [1, 7, 8, 12] in [5]. Boldyreva *et al.* [5] proposed a more accurate security model for PS. In their security model, the security of a PS scheme needs to be analyzed in four situations, and generating self-PS is considered a weak secure situation [1, 5]. However, Boldyreva *et al.* did not consider exposure arbitrary proxy signing keys in their security model. Thus, Schuldt's work [8] is not introduced to their security

[1]Warrant involves some delegating information from delegator to proxy signer, which includes signing rights, signing policies and so on; in this paper, we consider that delegator may generate his own warrant by himself.

model. Furthermore, the proposed security models are based on public key cryptography and do not involve the concept of identity. With the development of identity-based cryptography, it is necessary to research and develop PS based on identity-based cryptography.

Identity-based cryptography is another cryptographic primitive, which was first introduced by Shamir. In identity-based cryptography, a user's public key is obtained from his/her public identity, such as name, IP address and email address. Then, the user's private key is distributed from a private key generator (PKG). The main target of application of identity-based cryptography is to simplify key management and remove public key certificates. Owing to the contributions of [3, 4, 13, 14], a rapid development of identity-based cryptography has taken place. Boneh and Franklin [13] proposed an identity-based encryption scheme in the random oracle model. Waters [4] proposed an efficient identity-based encryption scheme in the standard model. Then, based on their works, many researchers proposed more and more identity-based signature (IBS) schemes in the random oracle model or standard model [3, 15–17]. Also, with these IBS schemes, a lot of variants, such as identity-based proxy signature (IBPS) scheme [18–24] and identity-based ring signature scheme [25, 26], have also been proposed. Thus, IBPS is a special identity-based cryptography, which is a combination of PS and identity-based cryptography. Comparing with PS based on public key cryptography, IBPS can simplify key management and be more easily used for many applications, such as distributed systems, grid computing, mobile agent applications, distributed shared object systems, global distribution networks and mobile communications. Then we focus on IBPS in this paper.

Compared with public-key-based PS, the security requirements of IBPS have the following:

> *Unforgeability*: A valid IBPS signature must be signed by a proxy signer with a warrant delegated by a delegator. Therefore, no poly-time adversary can produce a valid IBPS signature on any identities and messages when the adversary may adaptively be permitted to choose identities and messages after executing key oracle and signature oracle[2] (see the security model of Section 5 for more details).
> *Non-repudiation*: A valid IBPS signature for a message must be approved by the delegator and proxy signer. It means that the delegator and proxy signer cannot deny signing the message.
> *Strong identifiability*: A valid IBPS signature for a message must reveal the identities of the delegator and proxy signer. Although a user's public key is replaced by his/her public identity in IBPS, such as name, IP address or email address, the IBPS signature must identify who the delegator and the proxy signer are (whose identities can satisfy the IBPS signature).
> *Prevention of misuse*: The security property is the same as that of public-key-based PS. It means that any valid IBPS signature must be generated by a valid proxy signer on a valid warrant.

Because IBPS has more advantages than public-key-based PS in key management, IBPS is a natural choice for many earlier mentioned applications. Thus, our motivation for developing such IBPS scheme also comes from the requirements of the applications.

Presently, many public-key-based PS schemes [6, 10–12, 27–45] are proposed in the random oracle model or standard model, which are based on various mathematical problem assumptions, such as computational Diffie–Hellman (CDH) problem [9, 10, 28, 34, 36, 42], factoring problem [40], discrete logarithm problem [27, 29, 30, 35] and extended k-plus problem [39]. Also, based on identity-based cryptography, many variants of public-key-based PS are proposed [18–24, 46–49]. The first IBPS scheme was proposed in [24], but it lacked a formal security model. Then the first formal security model for IBPS was proposed in [23]. Unfortunately, the security model is not complete because of lacking the security analysis for self-proxy signing. Gu and Zhu [19] provided a general security model for IBPS. However, their model is still not complete. Hence, based on the works of [19, 23, 24], more and more IBPS schemes are proposed [18, 20–22]. In [15, 20, 21], the security models are not complete enough according to Boldyreva's work (the work of [21] considered exposure arbitrary proxy signing keys). In [22], the security models defined three types of attacks. However, the three types of attacks lack the logical relationship. Furthermore, the proposed schemes [18–24] are still constructed in the random oracle model. Cao and Cao [47] proposed an IBPS scheme in the standard model. However, their security model is not complete. Thus, their scheme is not secure because of not resisting the attack of delegator. Also, their scheme requires more computation cost. Additionally, to make secure PS schemes against quantum analysis, Kim *et al.* [49] proposed a provably secure IBPS scheme based on the lattice problems in the adaptive security model. Because of requiring many computations based on matrix, their scheme is not efficient under the current framework of computer.

The rest of this paper is organized as follows. In Section 2, we introduce our contributions; in Section 3, we review the bilinear pairings and complexity assumptions on which we build; in Section 4, we show a framework for IBPS; in Section 5, we set up the security model for IBPS; in Section 6, we propose an IBPS scheme in the standard model and in Section 7, we

[2] In the attacking experiments for public-key-based proxy signature, adversary cannot execute key oracle; we call this unforgeability 'no fully secure unforgeability'. In the attacking experiments for identity-based proxy signature, adversary may be permitted to execute key oracle, we call this unforgeability 'fully secure unforgeability' (adaptive-identity unforgeability). In this paper, the security of the proposed scheme meets the requirement of adaptive-identity unforgeability.

analyze the correctness, efficiency and security of the proposed scheme. Finally, we draw our conclusions in Section 8.

2. OUR CONTRIBUTIONS

In this paper, we present an efficient IBPS scheme in the standard model. Compared with other IBPS schemes proposed by [18–24] in the random oracle model, the proposed scheme is constructed in the standard model and is efficient. Furthermore, compared with Cao's scheme [47] in the standard model, the proposed scheme is secure and decreases the computation cost. In this paper, our contributions are as follows:

(i) We present a fully secure (adaptive-identity unforgeable) IBPS scheme in the standard model. No poly-time adversary can produce a valid IBPS signature on any identities and messages when the adversary may adaptively be permitted to choose identities and messages after executing key oracle and signature oracle.

(ii) We present a framework for IBPS, and show a detailed security model for IBPS. Compared with the security models for IBPS [19, 23, 24], we introduce Boldyreva's model [5] and Schuldt's work [8] to our security model. We further strengthen our security model by considering self-proxy signing and exposure arbitrary proxy signing keys. Under our security model, the proposed IBPS scheme is proved to be secure in the standard model, and has a security reduction to the simple standard assumption (CDH assumption).

Remark 2.1. Wen *et al.* [21] considered exposure arbitrary proxy signing keys. However, the proposed security model [21] still lacks the security analysis for self-proxy signing according to Boldyreva's work [5].

(iii) The proposed IBPS scheme is efficient by reducing the amount of computations and communications. Compared with the public-key-based PS scheme [9] and the IBPS scheme [47] in the standard model, the proposed IBPS scheme further reduces the computation cost; and compared with another IBPS scheme [20] in the random oracle model, the proposed IBPS scheme also has some advantages (the comparisons of the four schemes are given in Appendix 1).

3. PRELIMINARIES

3.1. Bilinear maps

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be groups of prime order q and g be a generator of $\mathbb{G}_1$. We say $\mathbb{G}_2$ has an admissible bilinear map e: $\mathbb{G}_1 \times \mathrm{G}_1 \to \mathbb{G}_2$, into $\mathbb{G}_2$ if the following two conditions hold. The map is bilinear; for all a, b, we have $e(g^a, g^b) = e(g, g)^{ab}$. The map is non-degenerate; we must have that $e(g, g) \neq 1$.

3.2. CDH assumption

DEFINITION 3.1 (CDH PROBLEM). *Let $\mathbb{G}_1$ be a group of prime order q and g be a generator of $\mathbb{G}_1$; for all $(g, g^a, g^b) \in \mathbb{G}_1$, with $a, b \in \mathbb{Z}_q$, the CDH problem is to compute g^{ab}.*

DEFINITION 3.2. *The (t, ε)-CDH assumption holds if not-time algorithm can solve the CDH problem with probability at least ε.*

4. A FRAMEWORK FOR IBPS

IBPS is an extension of IBS. An IBPS scheme involves an IBS scheme for standard signing [1, 5]. In the section, we present a formal definition of IBPS, which is based on those of Gu and Zhu [19], Xu *et al.* [23] and Zhang and Kim [24]. Let $\mathbb{A}$ be the universe of possible identities, we set $\mathrm{ID} \subseteq \mathbb{A}$ as the identity of user.

DEFINITION 4.1 (IBPS SCHEME). *Let* **IBPS** $=$ (**IBS**, ***IDelegate***, ***IProxyKeyGen***, ***IProxySign***, ***IProxyVerify***) *be an IBPS scheme on $\mathbb{A}$, where* **IBS** $=$ (***Setup***, ***KeyGen***, ***ISign***, ***IVerify***) *is an* IBS *scheme and $\mathbb{A}$ is the universe of possible identities. In* **IBPS**, *all algorithms are described as follows:*

Setup: The randomized algorithm run by PKG inputs a security parameter 1^k, and then outputs all system parameters IPK.

KeyGen: The randomized algorithm run by PKG inputs (IPK, $\mathrm{ID}_i \subseteq \mathbb{A}$), and then outputs a private key sk_{ID_i}, where ID_i is the identity of user i and sk_{ID_i} is the private key of user i.

Remark 4.1. In this paper, to make our descriptions simpler, we set $i \in$ {delegator, proxy_signer}. Then, the ID/private key pair of delegator is $(\mathrm{ID}_{de}, sk_{\mathrm{ID}_{de}})$, the ID/private key pair of proxy signer is $(\mathrm{ID}_{pr}, sk_{\mathrm{ID}_{pr}})$.

ISign: *The randomized algorithm is a standard* IBS *algorithm. Signer needs to sign a message $\mathcal{M} \in \{0,1\}^*$. The algorithm run by a signer inputs (IPK, sk_{ID_i}, $\mathcal{M}$), and then outputs a standard signature σ, where $\sigma \in \{0,1\}^* \cup \{\perp\}$, sk_{ID_i} is the private key of signer and ID_i is the corresponding identity.*

IVerify: *The signature receivers verify a standard signature σ. The deterministic algorithm run by a signature verifier inputs (IPK, $\mathcal{M}$, ID_i, σ), and then outputs the Boolean value, accept or reject.*

IDelegate: *The randomized algorithm run by a delegator inputs (IPK, $sk_{\mathrm{ID}_{de}}$, w), where $w \subseteq \{0,1\}^*$ is the warrant of the delegator, then the algorithm outputs a delegation δ for a proxy signer.*

IProxyKeyGen: *The randomized algorithm run by the proxy signer generates a proxy signing key. The algorithm inputs (IPK, δ, $sk_{\mathrm{ID}_{pr}}$), and then outputs (w, psk_{pd}), where psk_{pd} is a proxy signing key in w.*

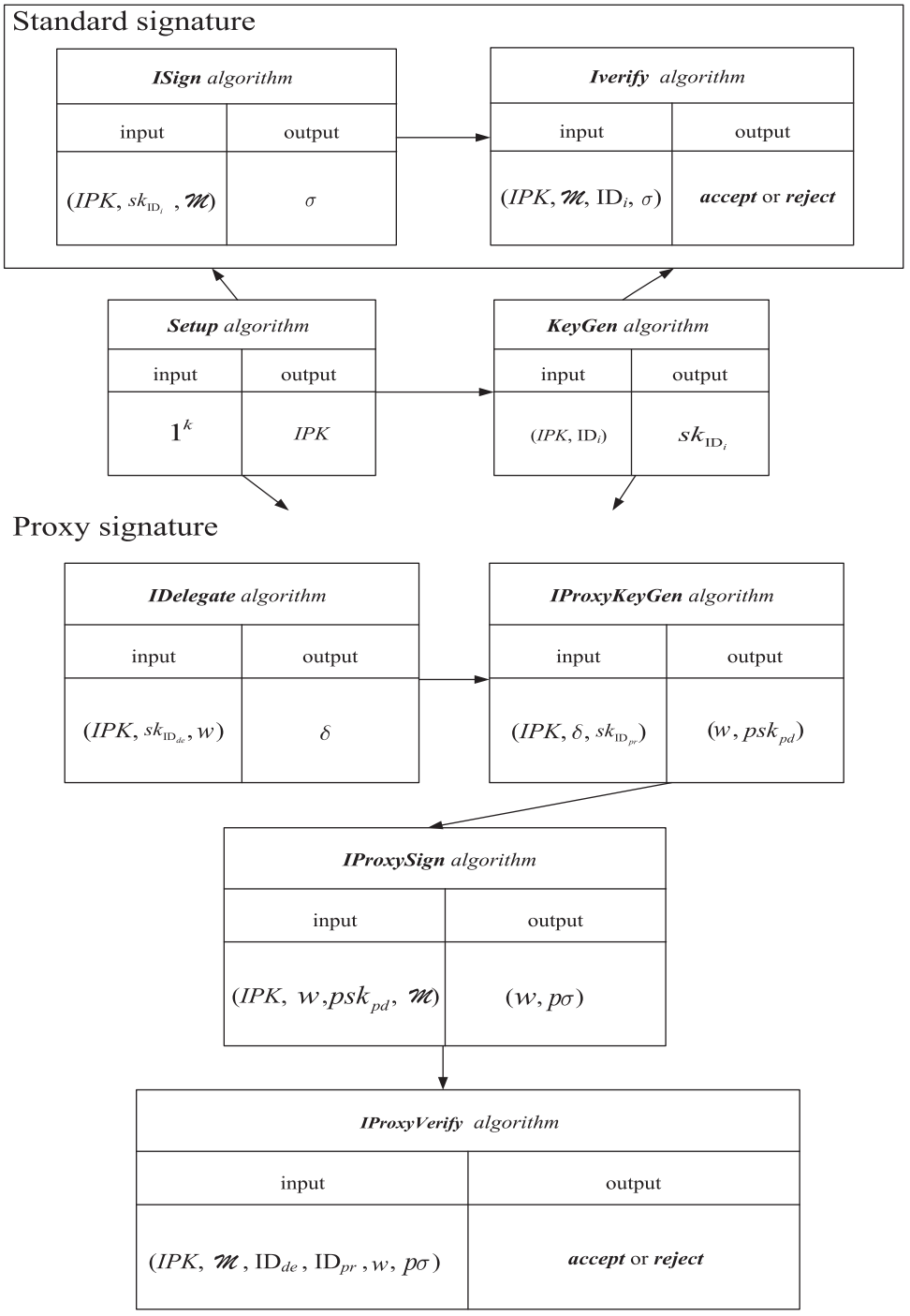

FIGURE 1. Framework of IBPS.

IProxySign: *The proxy signer needs to sign a message* $\mathcal{M} \in \{0,1\}^*$. *The randomized algorithm inputs* $(IPK, w, psk_{pd}, \mathcal{M})$, *and then outputs a* PS $(w, p\sigma)$ *on* $\mathcal{M}$, *where* $p\sigma \in \{0,1\}^* \cup \{\perp\}$.

IProxyVerify: *The signature receivers verify an IBPS* $(w, p\sigma)$ *on* $\mathcal{M} \in \{0,1\}^*$. *The deterministic algorithm inputs* $(IPK, \mathcal{M}, ID_{de}, ID_{pr}, w, p\sigma)$, *and then outputs the Boolean value*, *accept or reject*.

To make the framework of IBPS easier to understand, we also describe the framework of IBPS by Fig. 1.

The *correctness* of **IBPS** requires that for any $IPK \leftarrow \textbf{\textit{Setup}}(1^k)$, $sk_{ID_i} \leftarrow \textbf{\textit{KeyGen}}(IPK, ID_i)$ for

$i \in \{\text{delegator, proxy\_signer}\}$,

$w \subseteq \{0,1\}^*$ and $\mathcal{M} \in \{0,1\}^*$,

$$\boxed{\begin{array}{l}\delta \leftarrow \textbf{\textit{IDelegate}}\ (IPK, sk_{\mathrm{ID}_{de}}, w),\\ (w, psk_{pd}) \leftarrow \textbf{\textit{IProxyKeyGen}}(IPK, \delta, sk_{\mathrm{ID}_{pr}}),\end{array}}$$

$\Pr[\textbf{\textit{IProxyVerify}}(IPK, \mathcal{M}, \mathrm{ID}_{de}, \mathrm{ID}_{pr},$
$\textbf{\textit{IProxySign}}\ (IPK, w, psk_{pd}, \mathcal{M})) = 1] = 1.$

5. SECURITY MODEL

In the section, we show a detailed security model for IBPS. Compared with ordinary IBS schemes, IBPS schemes involve a delegating procedure. Thus, the security models for IBPS are more complicated than the security models for ordinary IBS. Boldyreva *et al.* [5] proposed an accurate provable security model for PS. Thus, based on the security models of [5, 19, 23, 24], we show a more complete security model for IBPS by considering exposure arbitrary proxy signing keys.

Remark 5.1. In [8, 21], the security models have considered exposure arbitrary proxy signing keys.

We assume that one user u^* is not corrupted by an adversary in an IBPS scheme. And we maximize the adversary's advantage [8], where the adversary can get all useful information except for the private key of u^*. Based on Boldyreva's work [5], there are the following four situations for analyzing security of an IBPS scheme:

(1) verifying a standard IBS by the identity of u^*;
(2) u^* computes an identity-based self-PS after u^* delegates his signing rights to oneself; in the situation, u^* is a delegator and is also a proxy signer;
(3) one user corrupted by an adversary delegates his signing rights to u^*, and then the adversary forges an IBPS of the corrupted user which pretends that u^* participates in signing;
(4) one user corrupted by an adversary does not get the signing rights of u^*, and then the adversary uses the corrupted user to forge an IBPS of u^*.

Based on the above four situations, we propose a complete security model for IBPS. Typically a security model is described in terms of an adversary $\mathcal{A}$ interacting with a challenger $\mathcal{C}$ in a security game. However, to make our security model easier to understand, we construct an algorithm $\mathcal{B}$ interacting with the adversary, which may make attack experiments to IBPS schemes in the above four situations. And we also introduce Schuldt's work [8] into our security model. We further strengthen our security model by considering exposure arbitrary proxy signing keys. In our security model, we maximize the adversary's advantage, and assume that all the attacking conditions needed by the adversary hold and the adversary may forge signatures after limitedly querying signature oracle in the above four situations. All symbols and parameters are defined as follows in $\mathcal{B}$:

(1) ***req_key***$(\mathrm{ID}_i \subseteq \mathbb{A})$ represents private key query; the procedure registers one user and outputs a private key on ID_i, where i is the indexed number of user and ID_i is the identity of the user i; the procedure of ***req_key***() is described as follows:

$$sk_{\mathrm{ID}_i} \leftarrow \textbf{\textit{req\_key}}(\mathrm{ID}_i) \Leftrightarrow \boxed{sk_{\mathrm{ID}_i} \leftarrow \textbf{\textit{KeyGen}}(parameters, \mathrm{ID}_i \subseteq \mathbb{A}).}$$

Remark 5.2. We assume that the public parameters have been generated by the algorithm ***Setup***.

(2) ***req_proxykey***$(param1, param2)$ represents proxy signing key query, which involves delegation query; the procedure of ***req_proxykey***(,) is described as follows:

$$psk \leftarrow \textbf{\textit{req\_proxykey}}(,) \Leftrightarrow \boxed{\begin{array}{l}\delta \leftarrow \textbf{\textit{IDelegate}}(param1, *, param2),\\ (w, psk) \leftarrow \textbf{\textit{IProxyKeyGen}}(param1, \delta, *).\end{array}}$$

Remark 5.3. It is responsible for finishing a delegating procedure between a delegator and a proxy signer, and generating a proxy signing key; the symbol $*$ represents the corresponding parameter.

$param1$ and $param2$ are the input parameters, where $param1$ is the public parameter and $param2$ is the warrant of delegator; additionally, ***req_proxykey***(,) outputs a proxy signing key.

(3) $\textbf{\textit{req\_sig}}_w(\cdot)$ represents IBPS query or standard IBS query; the procedure of $\textbf{\textit{req\_sig}}_w(\cdot)$ is described as follows:

$$\sigma \text{ or } p\sigma \leftarrow \textbf{\textit{req\_sig}}_w(\cdot) \Leftrightarrow \boxed{\begin{array}{l}\delta \leftarrow \textbf{\textit{IDelegate}}(*, *, w),\\ (w, psk) \leftarrow \textbf{\textit{IProxyKeyGen}}(*, \delta, *)\\ (w, p\sigma) \leftarrow \textbf{\textit{IProxySign}}(*, w, psk, \cdot),\\ \text{or } \sigma \leftarrow \textbf{\textit{ISign}}(*, *, \cdot).\end{array}}$$

where w is the warrant of delegator, the dot '$\cdot$' represents a queried message and $\textbf{\textit{req\_sig}}_w(\cdot)$ returns a simulated IBPS; *in the first situation*, w *is null*, ***req_sig***$(\cdot)$ *represents standard* IBS *query* and ***req_sig***$(\cdot)$ *returns a simulated standard* IBS;

(4) ***versign***$(param1, param2, param3, param4, param5, param6)$ represents verification of signature; the

procedure of ***versign***(, , , , ,) is described as follows:

$$1 \text{ or } 0 \leftarrow \textbf{\textit{versign}}(,,,,,) \Leftrightarrow \boxed{\begin{array}{l} 1 \text{ or } 0 \leftarrow \textbf{\textit{IProxyVerify}}(param1, param2, \\ \quad param3, param4, param5, param6) \\ \text{or, } 1 \text{ or } 0\ \textbf{\textit{IVerify}}(param1, \quad param2, \\ param3, param6). \end{array}}$$

all parameters are the input parameters, *param*1 is the public parameter, *param*2 is a verified message, *param*3 is the identity of delegator, *param*4 is the identity of proxy signer, *param*5 is the warrant of delegator and *param*6 is a verified signature; additionally, ***versign***(, , , , ,) returns 1 or 0; *in the first situation*, ***versign***(*param*1, *param*2, *param*3, *param*4, *param*5, *param*6) *represents verification of standard signature*, *where param*1 *is the public parameter*, *param*2 *is a verified message*, *param*3 *is the identity of signer*, *param*4 and *param*5 *are null* and *param*6 *is a verified standard signature*;

(5) k is a secure parameter, $\mathcal{A}$ represents an adversary and T represents the types of attack whose value is the enumerable set $\{type1, type2, type3, type4\}$, where *type*1 represents the attack in the first situation, *type*2 represents the attack in the second situation, *type*3 represents the attack in the third situation and *type*4 represents the attack in the fourth situation.

Now, algorithm $\mathcal{B}$ is described as follows:

1. **Setup**: Running ***Setup*** and ***KeyGen***, $parameters \leftarrow \textbf{\textit{Setup}}(1^k)$ and $sk_{\mathrm{ID}_1} \leftarrow \textbf{\textit{KeyGen}}(parameters, \mathrm{ID}_1)$, where we assume that the user 1 is a no corrupted user (for the simplicity of our description). Then all public parameters and ID_1 are passed to $\mathcal{A}$ (ID_1 is the identity of the user 1).

2. **Queries**: $\mathcal{A}$ makes queries to the following oracle for polynomially many times according to four situations:

Key Queries:

(I) $T = type1$

–***req_key***(ID_i): Given the identity of the user i, the oracle returns a private key sk_{ID_i} to $\mathcal{A}$, where i is the indexed number of user and $i > 1$.

(II) $T = type2$

–***req_proxykey***(,): Given the public parameters and a warrant w of the delegator (the user i), the oracle returns a proxy signing key *psk* for the proxy signer (the user i) on w to $\mathcal{A}$ (*psk* is valid with respect to the corresponding identities), where w may be an arbitrary forgery generated by $\mathcal{A}$ and i is the indexed number of user ($i > 1$).

(III) $T = type3$

–***req_proxykey***(,): Given the public parameters and a warrant w of the delegator (the user i), the oracle returns a proxy signing key *psk* for the proxy signer (the user 1) on w to $\mathcal{A}$ (*psk* is valid with respect to the corresponding identities), where w may be an arbitrary forgery generated by $\mathcal{A}$ and i is the indexed number of user ($i > 1$).

(IV) $T = type4$

–***req_proxykey***(,): Given the public parameters and a warrant w of the delegator (the user 1), the oracle returns a proxy signing key *psk* for the proxy signer (the user i) on w to $\mathcal{A}$ (*psk* is valid with respect to the corresponding identities), where w may be an arbitrary forgery generated by $\mathcal{A}$ and i is the indexed number of user ($i > 1$).

Signature Queries:

(I) $T = type1$

–***req_sig***($\cdot$): Given a message $\mathcal{M}$, the oracle returns a simulated standard IBS σ on $\mathcal{M}$ to $\mathcal{A}$, which is valid with respect to the corresponding identity.

(I) $T = type2$ or *type*3 or *type*4

–$\textbf{\textit{req\_sig}}_w(\cdot)$: Given a message $\mathcal{M}$ and a warrant w, the oracle returns a simulated IBPS $p\sigma$ on $\mathcal{M}$ and w to $\mathcal{A}$, which is valid with respect to the corresponding identities.

3. **Forgery**:

(I) $T = type1$, $\mathcal{A}$ outputs its forgery, $(\mathcal{M}^*, \sigma^*)$ for ID^*, where the identity ID^* is an arbitrary forgery generated by $\mathcal{A}$. It succeeds if

(a) $1 \leftarrow \textbf{\textit{versign}}(parameters, \mathcal{M}^*, \mathrm{ID}^*, \sigma*)$ and

(b) $\mathcal{A}$ did not query ***req_sig***($\cdot$) on input $\mathcal{M}^*$, and did not query ***req_key***() for ID^*.

(II) $T = type2$, $\mathcal{A}$ outputs its forgery, $(\mathcal{M}^*, p\sigma^*)$ for the identity ID^* of the delegator, the identity ID^* of the proxy signer and a warrant w^*, where the identity ID^* or the warrant w^*, or the identity ID^* and the warrant w^* are arbitrary forgeries generated by $\mathcal{A}$. It succeeds if

(a) $1 \leftarrow \textbf{\textit{versign}}(parameters, \mathcal{M}^*, \mathrm{ID}^*, \mathrm{ID}^*, w^*, p\sigma*)$ and

(b) $\mathcal{A}$ did not query $\textbf{\textit{req\_sig}}_w(\cdot)$ on input $\mathcal{M}^*$ and w^*, and did not query ***req_key***() for ID^* and did not query ***req_proxykey***(,) for w^* and ID^*.

(III) $T = type3$, $\mathcal{A}$ outputs its forgery, $(\mathcal{M}^*, p\sigma*)$ for the identity ID^* of the delegator (the user i with $i \in \{2, 3 \ldots\ldots \textbf{max}(\text{the number of queries})\}$), the identity ID_1 of the proxy signer (the user 1) and a warrant w^*, where the identity ID^* or the warrant w^*, or the identity ID^* and the warrant w^* are arbitrary forgeries generated by $\mathcal{A}$. It succeeds if

(a) $1 \leftarrow \textbf{\textit{versign}}(parameters, \mathcal{M}^*, \mathrm{ID}^*, \mathrm{ID}_1, w^*, p\sigma*)$ and

(b) $\mathcal{A}$ did not query $\textbf{\textit{req\_sig}}_w(\cdot)$ on input $\mathcal{M}^*$ and w^*, and did not query ***req_proxykey***(,) for w^*, ID^* and ID_1.

(IV) $T = type4$, $\mathbb{A}$ outputs its forgery, $(\mathcal{M}^*, p\sigma^*)$ for the identity ID_1 of the delegator (the user 1), the identity ID^* of the proxy signer (the user i with

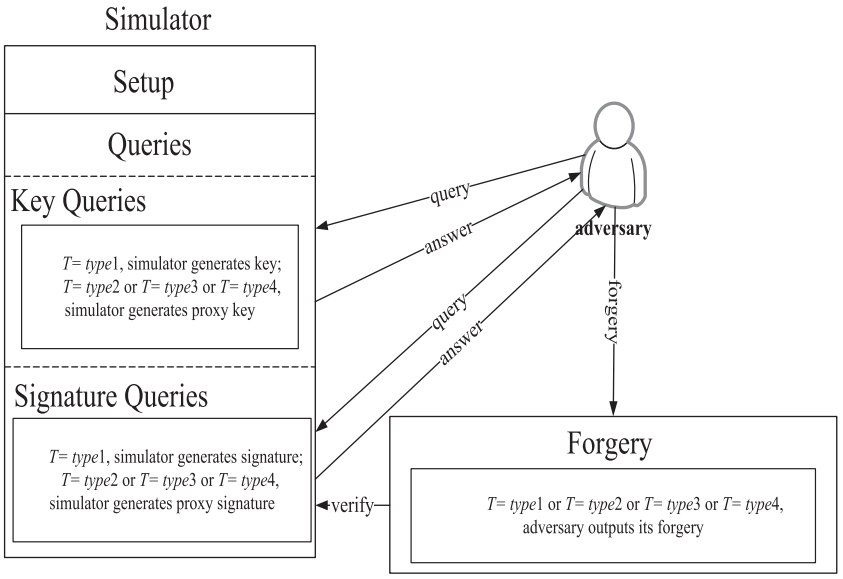

FIGURE 2. Procedure of security model.

$i \in \{2, 3 \ldots\ldots \textbf{max}(\text{the number of queries})\}$), and a warrant w^*, where the identity ID* or the warrant w^*, or the identity ID* and the warrant w^* are arbitrary forgeries generated by $\mathcal{A}$. It succeeds if

(a) $1 \leftarrow \textbf{\textit{versign}}(parameters, \mathcal{M}^*, \text{ID}_1, \text{ID}^*, w^*, p\sigma*)$ and

(b) $\mathcal{A}$ did not query $\textbf{\textit{req\_sig}}_w(\cdot)$ on input $\mathcal{M}^*$ and w^*, and did not query ***req_proxykey***(,) for w^*, ID_1 and ID*.

Algorithm $\mathcal{B}$ completely describes a security model for IBPS. Interacting with algorithm $\mathcal{B}$, the adversary $\mathcal{A}$ must undergo the system initialization, the stage of queries and the stage of forgery (Fig. 2 describes the procedure of the security model). Additionally, the views generated by the simulators are indistinguishable from the views generated by the actual execution.

DEFINITION 5.1 (SECURITY OF AN IBPS SCHEME). *Let* **IBPS** = (**IBS**, ***IDelegate***, ***IProxyKeyGen***, ***IProxySign***, ***IProxyVerify***) *be an* IBPS *scheme on* $\mathbb{A}$, *where* **IBS** = (***Setup***, ***KeyGen***, ***ISign***, ***IVerify***) *is an* IBS *scheme and* $\mathbb{A}$ *is the universe of possible identities. Additionally, we set that k is a secure parameter and* $\Pr(\mathcal{B}_{IBPS}(k, \mathcal{A}) = 1)$ *is the probability that algorithm* $\mathcal{B}$ *returns* 1. *Then the advantage that the adversary* $\mathcal{A}$ *breaks* **IBPS** *is defined as follows:*

$$\textbf{Adv}^{ibps\text{-}uf}_{IBPS,A}(k, q_e, q_s, t) = \Pr(\mathcal{B}_{IBPS}(k, \mathcal{A}) = 1),$$

where q_e *is the maximal number of key queries,* q_s *is the maximal number of signature queries and t is the running time of* $\mathcal{B}$. *If the advantage that the adversary breaks* **IBPS** *is negligible, then the scheme* **IBPS** *is secure.*

6. IBPS SCHEME

In the section, we show an IBPS scheme in the standard model under our framework for IBPS. Let **IBPS** = (**IBS**, ***IDelegate***, ***IProxyKeyGen***, ***IProxySign***, ***IProxyVerify***) be an IBPS scheme on $\mathbb{A}$, where **IBS** = (***Setup***, ***KeyGen***, ***ISign***, ***IVerify***) is an IBS scheme and $\mathbb{A}$ is the universe of possible identities. In **IBPS**, all algorithms are described as follows:

***Setup*:** The PKG system inputs a security parameter 1^k. Additionally, let $\mathbb{G}_1$ and $\mathbb{G}_2$ be groups of prime order q and g be a generator of $\mathbb{G}_1$, and let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ denote the bilinear map. The size of the group is determined by the security parameter. And one hash function, $H : \{0, 1\}^* \rightarrow \mathbb{Z}_{1^k \cdot q}$, can be defined and used to generate any integer value in $\mathbb{Z}_{1^k \cdot q}$ (where 1^k represents the corresponding decimal number).

Then the system parameters are generated as follows. The system chooses a random $a \in \mathbb{Z}_q$ and sets $g_1 = g^a$. Five group elements g_2, μ, υ, ϖ and $\tau \in \mathbb{G}_1$ are randomly chosen. Finally, the system outputs the public parameters $IPK = (\mathbb{G}_1, \mathbb{G}_2, e, g, g_1, g_2, \mu, \upsilon, \varpi, \tau)$.

***KeyGen*:** The PKG system generates a user's private key with respect to the identity of the user. To the identity $\text{ID} \subseteq \mathbb{A}$ of a user, the system randomly chooses $r \in \mathbb{Z}_q$, computes $x_0 = g_2^a \cdot \mu^{a \cdot r \cdot H(\text{ID})}$ and $x_1 = g_1^r$. Then the system outputs a private key $sk_{\text{ID}} = \{x_0, x_1\}$ for the user.

Remark 6.1. In **IBPS**, the ID/private key pair of the delegator is $(\text{ID}_{de}, sk_{\text{ID}_{de}})$, where $sk_{\text{ID}_{de}} = \{x_{de,0}, x_{de,1}\} = \{g_2^a \cdot \mu^{a \cdot r_{de} \cdot H(\text{ID}_{de})}, g_1^{r_{de}}\}$; the ID/private key pair of the proxy signer is $(\text{ID}_{pr}, sk_{\text{ID}_{pr}})$, where $sk_{\text{ID}_{pr}} = \{x_{pr,0}, x_{pr,1}\} = \{g_2^a \cdot \mu^{a \cdot r_{pr} \cdot H(\text{ID}_{pr})}, g_1^{r_{pr}}\}$.

***ISign*:** We assume that a signer (user) with an identity (ID) signs a message $\mathcal{M}$. The algorithm run by the signer inputs (IPK, sk_{ID}, $\mathcal{M}$). Then the algorithm chooses a random $d \in \mathbb{Z}_q$, and computes

$$X_0 = x_0 \cdot \varpi^d \cdot \tau^{d \cdot H(\mathcal{M})} = g_2^a \cdot \mu^{a \cdot r \cdot H(\mathrm{ID})} \cdot \varpi^d \cdot \tau^{d \cdot H(\mathcal{M})},$$
$$X_1 = x_1 = g_1^r, \ X_2 = g^d.$$

Lastly, the algorithm outputs an IBS $\sigma = \{X_0, X_1, X_2\}$.

***IVerify*:** The algorithm run by the verifier inputs (IPK, ID, $\mathcal{M}$, σ). Then the algorithm computes

$$e(X_0, g) = e(g_2, g_1) \cdot e(\mu, X_1^{H(\mathrm{ID})}) \cdot e(\varpi, X_2) \cdot e(\tau, X_2^{H(\mathcal{M})}).$$

If the above equation is equal, then the algorithm outputs the Boolean value *accept*; otherwise, the algorithm outputs the Boolean value *reject*.

***IDelegate*:** The algorithm run by the delegator inputs (IPK, $sk_{\mathrm{ID}_{de}}$, w), where w is the warrant of the delegator. Then the algorithm chooses a random $s \in \mathbb{Z}_q$, and computes

$$T_0 = x_{de,0} \cdot \upsilon^{s \cdot H(w)} = g_2^a \cdot \mu^{a \cdot r_{de} \cdot H(\mathrm{ID}_{de})} \cdot \upsilon^{s \cdot H(w)},$$
$$T_1 = x_{de,1} = g_1^{r_{de}}, \ T_2 = g^s;$$

and the algorithm outputs a delegation $\delta = \{T_0, T_1, T_2, w\}$ for a proxy signer.

***IProxyKeyGen*:** The algorithm run by the proxy signer generates a proxy signing key. In the algorithm, the following two steps need to be finished:

- The algorithm run by the proxy signer verifies the delegation, $\delta = \{T_0, T_1, T_2, w\}$ by the equation $e(T_0, g) = e(g_2, g_1) \cdot e(\mu, T_1^{H(\mathrm{ID}_{de})}) \cdot e(\upsilon, T_2^{H(w)})$; if the verifying result is not true, then the proxy signer may require the delegator to resend the delegation δ
- The algorithm run by the proxy signer computes $y_0 = x_{pr,0} \cdot T_0$, $y_1 = x_{pr,1} = g_1^{r_{pr}}$, $y_2 = T_1 = g_1^{r_{de}}$, $y_3 = T_2 = g^s$, and then outputs (w, psk_{pd}), where $psk_{pd} = \{y_0, y_1, y_2, y_3\}$ is a proxy signing key in w.

***IProxySign*:** The proxy signer needs to sign a message $\mathcal{M}$. The algorithm inputs (IPK, w, psk_{pd}, $\mathcal{M}$), and then chooses a random $d \in \mathbb{Z}_q$ and computes

$$Y_0 = y_0 \cdot \varpi^d \cdot \tau^{d \cdot H(\mathcal{M})}, \quad Y_1 = y_1,$$
$$Y_2 = y_2, \quad Y_3 = y_3, \quad Y_4 = g^d.$$

Lastly, the algorithm outputs an IBPS (w, $p\sigma = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$).

***IProxyVerify*:** The signature receivers verify an IBPS (w, $p\sigma$) on $\mathcal{M}$. The algorithm inputs (IPK, $\mathcal{M}$, ID_{de}, ID_{pr}, w, $p\sigma$), and then computes the equation,

$$e(Y_0, g) = e(g_2, g_1)^2 \cdot e(\mu, Y_1^{H(\mathrm{ID}_{pr})} \cdot Y_2^{H(\mathrm{ID}_{de})}) \cdot e(\upsilon, Y_3^{H(w)}) \cdot e(\varpi, Y_4) \cdot e(\tau, Y_4^{H(\mathcal{M})}).$$

If the equation is correct, then the algorithm outputs the Boolean value *accept*; otherwise, the algorithm outputs the Boolean value *reject*.

7. ANALYSIS OF THE PROPOSED SCHEME

7.1. Correctness

In the proposed scheme, IBPS is (w, $p\sigma = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$), where

$$Y_0 = y_0 \cdot \varpi^d \cdot \tau^{d \cdot H(\mathcal{M})} = x_{pr,0} \cdot T_0 \cdot \varpi^d \cdot \tau^{d \cdot H(\mathcal{M})}$$
$$= x_{pr,0} \cdot x_{de,0} \cdot \upsilon^{s \cdot H(w)} \cdot \varpi^d \cdot \tau^{d \cdot H(\mathcal{M})},$$
$$Y_1 = y_1 = x_{pr,1} = g_1^{r_{pr}}, \quad Y_2 = y_2 = T_1 = g_1^{r_{de}},$$
$$Y_3 = y_3 = T_2 = g^s, \quad Y_4 = g^d.$$

$p\sigma$ may be verified by the following equation:

$$\begin{aligned}
e(Y_0, g) &= e(x_{pr,0} \cdot x_{de,0} \cdot \upsilon^{s \cdot H(w)} \cdot \varpi^d \cdot \tau^{d \cdot H(\mathcal{M})}, g) \\
&= e(x_{pr,0}, g) \cdot e(x_{de,0}, g) \cdot e(\upsilon^{s \cdot H(w)}, g) \\
&\quad \cdot e(\varpi^d, g) \cdot e(\tau^{d \cdot H(\mathcal{M})}, g) \\
&= e(g_2^a \cdot \mu^{a \cdot r_{pr} \cdot H(\mathrm{ID}_{pr})}, g) \cdot e(g_2^a \cdot \mu^{a \cdot r_{de} \cdot H(\mathrm{ID}_{de})}, g) \\
&\quad \cdot e(\upsilon, g^{s \cdot H(w)}) \cdot e(\varpi, g^d) \cdot e(\tau, g^{d \cdot H(\mathcal{M})}) \\
&= e(g_2^a, g) \cdot e(\mu^{a \cdot r_{pr} \cdot H(\mathrm{ID}_{pr})}, g) \cdot e(g_2^a, g) \\
&\quad \cdot e(\mu^{a \cdot r_{de} \cdot H(\mathrm{ID}_{de})}, g) \cdot e(\upsilon, g^{s \cdot H(w)}) \\
&\quad \cdot e(\varpi, g^d) \cdot e(\tau, g^{d \cdot H(\mathcal{M})}) \\
&= e(g_2, g_1)^2 \cdot e(\mu, g^{a \cdot r_{pr} \cdot H(\mathrm{ID}_{pr})}) \cdot e(\mu, g^{a \cdot r_{de} \cdot H(\mathrm{ID}_{de})}) \\
&\quad \cdot e(\upsilon, g^{s \cdot H(w)}) \cdot e(\varpi, g^d) \cdot e(\tau, g^{d \cdot H(\mathcal{M})}) c \\
&= e(g_2, g_1)^2 \cdot e(\mu, g_1^{r_{pr} \cdot H(\mathrm{ID}_{pr})}) \cdot e(\mu, g_1^{r_{de} \cdot H(\mathrm{ID}_{de})}) \\
&\quad \cdot e(\upsilon, g^{s \cdot H(w)}) \cdot e(\varpi, g^d) \cdot e(\tau, g^{d \cdot H(\mathcal{M})}) \\
&= e(g_2, g_1)^2 \cdot e(\mu, Y_1^{H(\mathrm{ID}_{pr})} \cdot Y_2^{H(\mathrm{ID}_{de})}) \cdot e(\upsilon, Y_3^{H(w)}) \\
&\quad \cdot e(\varpi, Y_4) \cdot e(\tau, Y_4^{H(\mathcal{M})}).
\end{aligned}$$

7.2. Efficiency

In the proposed scheme, $p\sigma = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$, where $Y_0 = y_0 \cdot \varpi^d \cdot \tau^{d \cdot H(\mathcal{M})}$, $Y_1 = y_1$, $Y_2 = y_2$, $Y_3 = y_3$, $Y_4 = g^d$. So the length of the signature is $5 \cdot |\mathbb{G}_1| + |w|$, where $|\mathbb{G}_1|$ is the size of element in $\mathbb{G}_1$ and $|w|$ represents the length of the warrant. Additionally, because $y_0 \cdot \varpi^d$, $Y_4 = g^d$ and τ^d in $\tau^{d \cdot H(\mathcal{M})}$ may be pre-computed, and we assume that the time for integer multiplication and hash computation can be ignored, signing a message needs to compute at most 1 exponentiation in $\mathbb{G}_1$ and 1 multiplication in $\mathbb{G}_1$. Also, the signature receiver needs to verify an IBPS by the following equation:

$$e(Y_0, g) = e(g_2, g_1)^2 \cdot e(\mu, Y_1^{H(\mathrm{ID}_{pr})} \cdot Y_2^{H(\mathrm{ID}_{de})}) \cdot e(\upsilon, Y_3^{H(w)}) \cdot e(\varpi, Y_4) \cdot e(\tau, Y_4^{H(\mathcal{M})}).$$

Because the value $e(g_2, g_1)^2$ can be pre-computed and cached, verification requires five pairing computations, one multiplication in $\mathbb{G}_1$, four exponentiations in $\mathbb{G}_1$ and four multiplications in $\mathbb{G}_2$.

In this paper, we also compare the proposed scheme (the scheme of Section 6) with the public-key-based PS scheme [9] and the IBPS scheme [47] in the standard model, and another IBPS scheme [20] in the random oracle model. In Appendix 1, we show the comparisons of the four schemes. Additionally, although Kim *et al.* [49] recently proposed a provably secure IBPS scheme from lattices in the adaptive security model, their scheme is not efficient because of requiring many computations based on matrix. Under the current framework of computer, our scheme is more efficient compared with Kim's scheme.

7.3. Security

In the section, we show that the proposed scheme has a security reduction to CDH assumption and the full IBPS unforgeability under an adaptive chosen message attack. Our proof for Theorem 7.1 is based on the security model of Section 5 (we defer the proof for Theorem 7.1 to Appendix 2).

THEOREM 7.1. *The scheme of Section* 6 *is* $(t, \varepsilon, q_e, q_s)$*-secure, assuming that the* (t', ε')*-CDH assumption holds in* $\mathbb{G}_1$*, where*:

$$\varepsilon' = \left(1 - \frac{q_e}{q}\right)^2 \cdot \left(1 - \frac{q_s}{q}\right)^2 \cdot \frac{\varepsilon}{q^3},$$
$$t' = t + O(q_e \cdot (14 \cdot C_{\text{mul}} + 21 \cdot C_{\text{exp}}) + q_s \cdot (27 \cdot C_{\text{mul}} + 38 \cdot C_{\text{exp}})),$$

and q_e *is the maximal number of key queries,* q_s *is the maximal number of signature queries and* C_{mul} *and* C_{exp} *are respectively the time for a multiplication and an exponentiation in* $\mathbb{G}_1$.

8. CONCLUSIONS

In this paper, we present an efficient IBPS scheme in the standard model. Compared with other IBPS schemes proposed by Gu and Zhu [18, 19], Singh and Verma [20], Wen *et al.* [21], Wu *et al.* [22], Xu *et al.* [23], Zhang and Kim [24] in the random oracle model, the proposed IBPS scheme is constructed in the standard model and is efficient. Also, we present a detailed security model for IBPS. Compared with the security models for IBPS [19, 23, 24], we introduce Boldyreva's model [5] and Schuldt's work [8] to our security model. Under our security model, the proposed IBPS scheme is proved to be secure in the standard model, and has a security reduction to CDH assumption. Because it is very challenging to construct a secure and efficient IBPS scheme in the standard model, the work about IBPS still needs to be further progressed.

FUNDING

This work was supported by grants from the Research Grants Council of the Hong Kong SAR, China, No. (CityU 114609) and CityU Applied R&D Funding (ARD) Nos. 9681001, 6351006 and CityU Strategic Research Grant No. 7008110; CityU Applied Research Grant (ARG) No. 9667052; ShenZhen-HK Innovation Cycle Grant No. ZYB200907080078A and NSF (China) No. 61070222/F020802.

REFERENCES

[1] Boldyreva, A., Palacio, A. and Warinschi B. (2003) Secure proxy signature schemes for delegation of signing rights. IACR ePrint Archive. http://eprint.iacr.org/2003/096.

[2] Boneh, D. and Boyen, X. (2004) Short Signatures without Random Oracles. *Proc. Advances in Cryptology—EUROCRYPT 2004*, Interlaken, Switzerland, May 2–6, pp. 56–73. Springer, Berlin.

[3] Paterson, K.G. and Schuldt, J.C.N. (2006) Efficient Identity-Based Signatures Secure in the Standard Model. *Proc. ACISP 2006*, Melbourne, Australia, July 3–5, pp. 207–222. Springer, Berlin.

[4] Waters, B. (2005) Efficient Identity-Based Encryption without Random Oracles. *Proc. Advances in Cryptology—EUROCRYPT 2005*, Aarhus, Denmark, May 22–26, pp. 114–127. Springer, Berlin.

[5] Boldyreva, A., Palacio, A. and Warinschi, B. (2012) Secure proxy signature schemes for delegation of signing rights. *J. Cryptol.*, **25**, 57–115.

[6] Fuchsbauer, G. and Pointcheval D. (2008) Anonymous consecutive delegation of signing rights: unifying group and proxy signatures. IACR ePrint Archive. http://eprint.iacr.org/2008/037.pdf.

[7] Malkin, T., Obana, S. and Yung M. (2004) The Hierarchy of Key Evolving Signatures and a Characterization of Proxy Signatures. *Proc. Advances in Cryptology—EUROCRYPT 2004*, Santa Barbara, USA, August 15–19, pp. 306–322. Springer, Berlin.

[8] Schuldt, J.C.N., Matsuura, K. and Paterson, K.G. (2008) Proxy Signatures Secure Against Proxy Key Exposure. *Proc. PKC 2008*, Barcelona, Spain, March 9–12, pp. 141–161. Springer, Berlin.

[9] Sun, Y., Xu, C.X., Yu, Y. and Mu, Y. (2011) Strongly unforgeable proxy signature scheme secure in the standard model. *J. Syst. Softw.*, **84**, 1471–1479.

[10] Huang, X.Y., Susilo, W., Mu, Y. and Wu, W. (2006) Proxy Signature without Random Oracles. *Proc. Mobile Ad-hoc and Sensor Networks 2006*, Hong Kong, China, December 13–15, pp. 473–484. Springer, Berlin.

[11] Zhang, J.H. and Mao, J. (2011) Another efficient proxy signature scheme in the standard model. *J. Inf. Sci. Eng.*, **27**, 1249–1264.

[12] Herranz, J. and Saez, G. (2003) Revisiting fully distributed proxy signature schemes. IACR ePrint Archive. http://eprint.iacr.org/2003/197.

[13] Boneh, D. and Franklin, M. (2001) Identity-Based Encryption from the Weil Pairing. *Proc. Advances in Cryptology—CRYPTO 2001*, Santa Barbara, CA, USA, August 19–23, pp. 213–229. Springer, Berlin.

[14] Boneh, D. and Hanburg, M. (2008) Generalized Identity Based and Broadcast Encryption Schemes. *Proc. Advances in Cryptology—ASIACRYPT 2008*, Melbourne, Australia, December 7–11, pp. 455–470. Springer, Berlin.

[15] Barreto, P.S.L.M., Libert, B., McCullagh, N. and Quisquater, J. (2005) Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps. *Proc. Advances in Cryptology—ASIACRYPT 2005*, Chennai (Madras), India, December 4–8, pp. 515–532. Springer, Berlin.

[16] Cha, J.C. and Cheon, J.H. (2003) An Identity-Based Signature from Gap Diffie–Hellman Groups. *Proc. PKC 2003*, Miami, USA, January 6–8, pp. 18–30. Springer, Berlin.

[17] Hess, F. (2002) Efficient Identity Based Signature Schemes Based on Pairings. *Proc. Selected Areas in Cryptography 9th Annual International Workshop*, Newfoundland, Canada, August 15–16, pp. 310–324. Springer, Berlin.

[18] Gu, C.X. and Zhu, Y.F. (2006) An efficient ID-based proxy signature scheme from pairings. IACR ePrint Archive. http://eprint.iacr.org/2006/158.

[19] Gu, C.X. and Zhu, Y.F. (2005) Provable Security of ID-Based Proxy Signature Schemes. *Proc. ICCNMC 2005*, Zhangjiajie, China, August 2–4, pp. 1277–1286. Springer, Berlin.

[20] Singh, H. and Verma, G.K. (2012) ID-based proxy signature scheme with message recovery. *J. Syst. Softw.*, **85**, 209–214.

[21] Wen, F.T., Cui, S.J. and Cui, J.N. (2011) An ID-based proxy signature scheme secure against proxy key exposure. *Int. J. Adv. Comput. Technol.*, **3**, 108–116.

[22] Wu, W., Mu, Y., Susilo, W., Seberry, J. and Huang, X.Y. (2007) Identity-Based Proxy Signature from Pairings. *Proc. ATC 2007*, Hong Kong, China, July 11–13, pp. 22–31. Springer, Berlin.

[23] Xu, J., Zhang, Z. and Feng, D. (2005) ID-Based Proxy Signature using Bilinear Pairings. *Proc. ISPA 2005*, Nanjing, China, November 2–5, pp. 359–367. Springer, Berlin.

[24] Zhang, F. and Kim, K. (2003) Efficient ID-Based Blind Signature and Proxy Signature from Bilinear Pairings. *Proc. ACISP 2003*, Wollongong, Australia, July 9–11, pp. 312–323. Springer, Berlin.

[25] Au, M.H., Liu, J.K., Yuen, T.H. and Wong, D.S. (2006) ID-Based Ring Signature Scheme Secure in the Standard Mode. *Proc. IWSEC 2006*, Kyoto, Japan, October 23–24, pp. 1–16. Springer, Berlin.

[26] Zhang, F. and Kim, K. (2002) ID-Based Blind Signature and Ring Signature from Pairings. *Proc. Advances in Cryptology—ASIACRYPT 2002*, Queenstown, New Zealand, December 1–5, pp. 533–547. Springer, Berlin.

[27] Cui, S.J. and Wen, F.T. (2010) Improvement of a Forward-Secure Proxy Signature Scheme. *Proc. ICCET2010*, Chengdu, China, April 16–18, pp. 441–444. IEEE Computer Society, Washington.

[28] Eslami, Z. and Pakniat, N. (2012) A Certificateless Proxy Signature Scheme Secure in Standard Model. *Proc. ICLCT 2012*, Bangkok, Thailand, March 17–18, pp. 81–84. Planetary Scientific Research Center, Washington.

[29] Hwang, S.J. and Chen, C.C. (2005) New threshold-proxy threshold-signature schemes. *Comput. Electr. Eng.*, **31**, 69–80.

[30] Huang, H.F. and Chang, C.C. (2006) A novel efficient (t, n) threshold proxy signature scheme. *Inf. Sci.*, **176**, 1338–1349.

[31] Hu, J.H. and Zhang, J.Z. (2009) Cryptanalysis and improvement of a threshold proxy signature scheme. *Comput. Stand. Interfaces*, **31**, 169–173.

[32] Hsu, C.L. and Wu, T.S. (2005) Efficient nonrepudiable threshold proxy signature scheme with known signers against the collusion attack. *Appl. Math. Comput.*, **168**, 305–319.

[33] Huang, X., Mu, Y., Susilo, W., Zhang, F. and Chen, X. (2005) A Short Proxy Signature Scheme: Efficient Authentication in the Ubiquitous World. *Proc. EUC Workshops 2005*, Nagasaki, Japan, December 6–9, pp. 480–489. Springer, Berlin.

[34] Jin, Z.P. and Wen, Q.Y. (2011) Certificateless multi-proxy signature. *Comput. Commun.*, **34**, 344–352.

[35] Kim, S.J., Park, S.J. and Won, D.H. (1997) Proxy Signatures, Revisited. *Proc. ICICS 97*, Beijing, China, November 11–14, pp. 223–232. Springer, Berlin.

[36] Liu, Z.H., Hu, Y.P., Zhang, X.S. and Ma. H. (2011) Provably secure multi-proxy signature scheme with revocation in the standard model. *Comput. Commun.*, **34**, 494–501.

[37] Li, J.G., Xu, L.Z. and Zhang, Y.C. (2009) Provably secure certificate-based proxy signature schemes. *J. Comput.*, **4**, 444–452.

[38] Mehta, M. and Harn, L. (2005) Efficient one-time proxy signatures. *IEE Proc. Commun.*, **152**, 129–134.

[39] Okamoto, T., Inomata, A. and Okamoto, E. (2005) A Proposal of Short Proxy Signature using Pairing. *Proc. ITCC 2005*, Las Vegas, USA, April 4–6, pp. 631–635. IEEE Computer Society, Washington.

[40] Shao, Z.H. (2009) Provably secure proxy-protected signature schemes based on RSA. *Comput. Electr. Eng.*, **35**, 497–505.

[41] Wang, A.Q., Li, J.G. and Wang, Z.J. (2010) A provably secure proxy signature scheme from bilinear pairings. *Chin. J. Electron.*, **27**, 298–304.

[42] Xiong, H., Li, F.G. and Qin, Z.G. (2010) A provably secure proxy signature scheme in certificateless cryptography. *Informatica*, **21**, 277–294.

[43] Yu, Y., Mu, Y., Susilo, W., Sun, Y. and Ji, Y.F. (2012) Provably secure proxy signature scheme from factorization. *Math. Comput. Model.*, **55**, 1160–1168.

[44] Zhang, F.G., Naini, R.Sa. and Lin, C.Y. (2003) New proxy signature, proxy blind signature and proxy ring signature schemes from bilinear pairing. IACR ePrint Archive. http://eprint.iacr.org/2003/104.

[45] Zhang, J.H. and Liu, X. (2010) A Proposal of Short Proxy Signature using Pairing. *Proc. ISME 2010*, Xian, China, August 7–8, pp. 71–74. IEEE Computer Society, Washington.

[46] Cao, F. and Cao, Z.F. (2009) A secure identity-based multi-proxy signature scheme. *Comput. Electr. Eng.*, **35**, 86–95.

[47] Cao, F. and Cao, Z.F. (2010) An Identity Based Proxy Signature Scheme Secure in the Standard Model. *Proc. GRC 10*, San Jose, USA, August 14–16, pp. 67–72. IEEE Computer Society, Washington.

[48] Xiong, H., Hu, J.B., Chen, Z. and Li, F.G. (2011) On the security of an identity based multi-proxy signature scheme. *Comput. Electr. Eng.*, **37**, 129–135.

[49] Kim, K.S., Hong, D. and Jeong, I.R. (2013) Identity-based proxy signature from lattices. *J. Commun. Netw.*, **15**, 1–7.

APPENDIX 1. A COMPARISONS OF FOUR SCHEMES

Tables A1–A3 show the comparisons of the four schemes (the scheme of Section 6, Sun's scheme [9], Cao's scheme [47] and Singh's scheme [20]). Table A1 shows the key length

TABLE A1. Key length comparison of four schemes.

	Private key length	Public key length
Scheme [47]	$2 \cdot \|\mathbb{G}_1\|$	Zero
Scheme [9]	$2 \cdot \|\mathbb{Z}_q\|$	$2 \cdot \|\mathbb{G}_1\|$
Scheme [20]	$\|\mathbb{G}_1\|$	$\|\mathbb{G}_1\|$
Our scheme	$2 \cdot \|\mathbb{G}_1\|$	Zero

$|\mathbb{Z}_q|$ represents the length of element in $\mathbb{Z}_q$ and $|\mathbb{G}_1|$ represents the length of element in $\mathbb{G}_1$.

TABLE A2. Delegation and signature length comparison of four schemes.

	The length of delegation	The length of signature
Scheme [47]	$3 \cdot \|\mathbb{G}_1\| + \|w\|$	$5 \cdot \|\mathbb{G}_1\| + \|w\|$
Scheme [9]	$2 \cdot \|\mathbb{G}_1\| + \|w\|$	$3 \cdot \|\mathbb{G}_1\| + \|w\|$
Scheme [20]	$\|\mathbb{G}_1\| + \|\mathbb{G}_2\| + \|w\|$	$\|\mathbb{G}_1\| + \|\mathbb{G}_2\| + \|\mathbb{Z}_q\| + \|w\|$
Our scheme	$3 \cdot \|\mathbb{G}_1\| + \|w\|$	$5 \cdot \|\mathbb{G}_1\| + \|w\|$

$|\mathbb{G}_1|$ represents the length of element in $\mathbb{G}_1$, $|\mathbb{G}_2|$ represents the length of element in Γ_2, $|\mathbb{Z}_q|$ represents the length of element in $\mathbb{Z}_q$ and $|w|$ represents the length of warrant.

comparison of the four schemes. Sun's scheme has the shortest key length because Sun's scheme is based on public key cryptography. Similarly, although Singh's scheme is based on identity-based cryptography and has the shorter key length, Singh's scheme still requires public key. Therefore, based on identity-based cryptography, our scheme and Cao's scheme do not require public key. Then our scheme has more advantage in simplifying key management. Table A2 shows the delegation and signature length comparison of the four schemes. Table A3 shows the performance comparison of the four schemes (where we do not consider pre-computation and do assume that Singh's scheme is constructed on multiplicative cyclic group). In Table A3, compared with Sun's scheme and Cao's scheme, our scheme has more performance advantage in signing and verification, and further reduces the computation cost; and compared with Singh's scheme, our scheme has less performance advantage in signing, but requires more computations in verification (because Singh's scheme employs many hash functions in the random oracle model). Therefore, our scheme is more efficient on the whole by key length, delegation and signature length and performance comparisons of the four schemes.

APPENDIX 2. SECURITY PROOF

A.1. Proof of Theorem 7.1

Proof. Let **IBPS** be an identity-based PS scheme of Section 6. Additionally, let $\mathcal{A}$ be an $(t, \varepsilon, q_e, q_s)$-adversary attacking **IBPS**. From the adversary $\mathcal{A}$, we construct an algorithm $\mathcal{B}$, for $(g, g^a, g^b) \in \mathbb{G}_1$, algorithm $\mathcal{B}$ is able to use $\mathcal{A}$ to compute g^{ab}. Thus, we assume that algorithm $\mathcal{B}$ can solve CDH with probability at least ε' and in time at most t', contradicting the (t', ε')-CDH assumption. Such a simulation may be created in the following way:

Setup: The PKG system inputs a security parameter 1^k. Additionally, let $\mathbb{G}_1$ and $\mathbb{G}_2$ be groups of prime order q and g be a generator of $\mathbb{G}_1$, and let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ denote the bilinear map. The size of the group is determined by the security parameter, and we set $\mathbb{A}$ as the universe of identities. One hash function, $H : \{0, 1\}^* \to \mathbb{Z}_{1^k \cdot q}$, can be defined and used to generate any integer value in $\mathbb{Z}_{1^k \cdot q}$ (where 1^k represents the corresponding decimal number).

Then the system parameters are generated as follows. The system sets $g_1 = g^a$ and $g_2 = g^b$ with $a, b \in \mathbb{Z}_q$ ($\mathcal{B}$ does not know a and b), chooses $\ell, \partial, \lambda, \eta \in \mathbb{Z}_q$ and then sets $\mu = g_2^{\ell} \cdot g$, $\upsilon = g_2^{\partial} \cdot g$, $\tau = g_2^{\lambda} \cdot g$ and $\varpi = g^{\eta}$. Then the system outputs the public parameters $IPK = (\mathbb{G}_1, \mathbb{G}_2, e, g, g_1, g_2, \mu, \upsilon, \varpi, \tau)$. Additionally, because algorithm B does not know a and b, the algorithm constructs a private key sk_{ID_1} by the following computation (where the user 1 is a no corrupted user and $\mathrm{ID}_1 \subseteq \mathbb{A}$ is the identity of the user 1): algorithm $\mathcal{B}$ chooses a random $r_1 \in \mathbb{Z}_q$, and computes $x_{1,0} = g_1^{-1/\ell} \cdot \mu^{r_1}$, $x_{1,1} = (g_1^{-1/\ell} \cdot g^{r_1})^{1/H(\mathrm{ID}_1)}$; then the algorithm generates a private key $sk_{\mathrm{ID}_1} = \{x_{1,0}, x_{1,1}\}$. Finally, ID_1 is passed to $\mathcal{A}$.

TABLE A3. Performance comparison of four schemes.

	Signing	Verification
Scheme [47]	$n_m \cdot C_{\mathrm{mul1}} + 2 \cdot C_{\mathrm{exp}}$	$2 \cdot n_u \cdot C_{\mathrm{mul1}} + n_w \cdot C_{\mathrm{mul1}} + n_m \cdot C_{\mathrm{mul1}} + C_{\mathrm{exp}} + 6 \cdot C_{\mathrm{pairing}} + 4 \cdot C_{\mathrm{mul2}}$
Scheme [9]	$C_h + 3 \cdot C_{\mathrm{mul1}} + 4 \cdot C_{\mathrm{exp}}$	$C_h + (n_w + 1) \cdot C_{\mathrm{mul1}} + (n_w + 1) \cdot C_{\mathrm{exp}} + 5 \cdot C_{\mathrm{pairing}} + 3 \cdot C_{\mathrm{mul2}}$
Scheme [20]	$4 \cdot C_h + C_{\mathrm{mul1}} + C_{\mathrm{mul2}} + C_{\mathrm{mul3}} + 2 \cdot C_{\mathrm{exp}}$	$4 \cdot C_h + C_{\mathrm{mul1}} + C_{\mathrm{mul2}} + C_{\mathrm{mul3}} + C_{\mathrm{exp}} + 2 \cdot C_{\mathrm{pairing}} + C_{\mathrm{xor}}$
Our scheme	$C_h + C_{\mathrm{mul3}} + 3 \cdot C_{\mathrm{exp}} + 2 \cdot C_{\mathrm{mul1}}$	$4 \cdot C_h + 5 \cdot C_{\mathrm{exp}} + C_{\mathrm{mul1}} + 6 \cdot C_{\mathrm{pairing}} + 4 \cdot C_{\mathrm{mul2}}$

C_{mul1} represents one multiplication in $\mathbb{G}_1$, C_{mul2} represents one multiplication in $\mathbb{G}_2$, C_{mul3} represents one multiplication for integer, C_{exp} represents one exponentiation in $\mathbb{G}_1$, C_{pairing} represents one pairing computation, C_h represents one hash computation, C_{xor} represents one XOR computation, n_u represents the length of the bit string of identity, n_w represents the length of the bit string of warrant and n_m represents the length of the bit string of message.

Remark A.1. To the correctness of sk_{ID_1}, sk_{ID_1} may be changed as follows:

$$\begin{aligned}
x_{1,0} &= g_1^{-1/\ell} \cdot \mu^{r_1} \\
&= g_2^a \cdot g_2^{-a} \cdot g_1^{-1/\ell} \cdot \mu^{r_1} = g_2^a \cdot g_2^{-a} \cdot g^{-a/\ell} \cdot (\mu)^{a \cdot r_1/a} \\
&= g_2^a \cdot (g_2^\ell \cdot g)^{-a/\ell} \cdot (\mu)^{a \cdot r_1/a} = g_2^a \cdot (\mu)^{-a/\ell} \cdot (\mu)^{a \cdot r_1/a} \\
&= g_2^a \cdot (\mu)^{a \cdot r_1/a - a/\ell} = g_2^a \cdot (\mu)^{a \cdot (r_1/a - 1/\ell)}, \\
x_{1,1} &= (g_1^{-1/\ell} \cdot g^{r_1})^{1/H(\mathrm{ID}_1)} = (g_1^{-1/\ell} \cdot g^{a \cdot r_1/a})^{1/H(\mathrm{ID}_1)} \\
&= (g_1^{-1/\ell} \cdot g_1^{r_1/a})^{1/H(\mathrm{ID}_1)} = (g_1^{r_1/a - 1/\ell})^{1/H(\mathrm{ID}_1)} \\
&= g_1^{(r_1/a - 1/\ell) \cdot (1/H(\mathrm{ID}_1))}.
\end{aligned}$$

Setting $r_1' = (r_1/a - 1/\ell) \cdot (1/H(\mathrm{ID}_1))$, $sk_{\mathrm{ID}_1} = \{x_{1,0}, x_{1,1}\} = \{g_2^a \cdot \mu^{a \cdot r_1' \cdot H(\mathrm{ID}_1)}, g_1^{r_1'}\}$ is a valid private key, where we assure that $a \cdot \ell \cdot H(\mathrm{ID}_1) \neq 0 \mod q (g_1^{\ell \cdot H(\mathrm{ID}_1)} \neq 1)$.

Queries: When running the adversary $\mathcal{A}$, key queries and signature queries can occur. Algorithm $\mathcal{B}$ answers these in the following way:

Key Queries:

(I) $T = type1$

Given an identity $\mathrm{ID}_i \subseteq \mathbb{A}$ of the user i with $i > 1$, algorithm $\mathcal{B}$ similarly constructs a private key $sk_{\mathrm{ID}_i} = \{x_{i,0}, x_{i,1}\} = \{g_1^{-1/\ell} \cdot \mu^{r_i}, (g_1^{-1/\ell} \cdot g^{r_i})^{1/H(\mathrm{ID}_i)}\}$. Setting $r_i' = (r_i/a - 1/\ell) \cdot (1/H(\mathrm{ID}_i))$, $sk_{\mathrm{ID}_i} = \{x_{i,0}, x_{i,1}\} = \{g_2^a \cdot \mu^{a \cdot r_i' \cdot H(\mathrm{ID}_i)}, g_1^{r_i'}\}$ is a valid private key. Then the algorithm outputs a private key $sk_{\mathrm{ID}_i} = \{x_{i,0}, x_{i,1}\}$ for the adversary $\mathcal{A}$.

If $a \cdot \ell \cdot H(\mathrm{ID}_i) = 0 \mod q (g_1^{\ell \cdot H(\mathrm{ID}_i)} = 1)$, then the above computation cannot be performed and the simulator will abort; otherwise, a private key sk_{ID_i} is passed to the adversary $\mathcal{A}$.

(II) $T = type2$

Given the public parameters and a warrant w of the delegator (the user i), the oracle returns a proxy signing key for the proxy signer (the user i) on w to $\mathcal{A}$, where w may be an arbitrary forgery generated by $\mathcal{A}$ and ID_i is the identity of the user i with $i > 1$. In the second situation, the user i is a delegator and is also a proxy signer, so the simulator simulating as the user i needs to delegate his signing rights to oneself. The algorithm randomly chooses $r_i, s \in \mathbb{Z}_q$, and computes

$$\begin{aligned}
&y_0 = g_1^{-1/\ell} \cdot \mu^{r_i} \cdot g_1^{-1/\partial} \cdot \upsilon^s, \\
&y_1 = y_2 = (g_1^{-1/\ell} \cdot g^{r_i})^{1/2 \cdot H(\mathrm{ID}_i)} \quad \text{and} \\
&y_3 = (g_1^{-1/\partial} \cdot g^s)^{1/H(w)},
\end{aligned}$$

where $\mathcal{B}$ gets the warrant $w$and the identity ID_i by the query of $\mathcal{A}$. And then algorithm $\mathcal{B}$outputs a proxy signing key $psk_{pd} = \{y_0, y_1, y_2, y_3\}$. Similarly, setting $r_i' = [r_i/a - 1/\ell] \cdot (1/2 \cdot H(\mathrm{ID}_i))$ and $s' = [s - a/\partial] \cdot (1/H(w))$, $psk_{pd} = \{y_0, y_1, y_2, y_3\} = \{g_2^{2 \cdot a} \cdot \mu^{2 \cdot a \cdot r_i' \cdot H(\mathrm{ID}_i)} \cdot \upsilon^{s' \cdot H(w)}, g_1^{r_i'}, g_1^{r_i'}, g^{s'}\}$ is a valid proxy signing key.

Remark A.2. To the correctness of psk_{pd}, psk_{pd} may be changed as follows:

$$\begin{aligned}
y_0 &= g_1^{-1/\ell} \cdot \mu^{r_i} \cdot g_1^{-1/\partial} \cdot \upsilon^s = g_2^a \cdot g_2^{-a} \cdot g_2^a \cdot g_2^{-a} \cdot g_1^{-1/\ell} \\
&\quad \cdot (\mu)^{a \cdot r_i/a} \cdot g_1^{-1/\partial} \cdot \upsilon^s \\
&= g_2^a \cdot g_2^a \cdot g_2^{-a} \cdot g_1^{-1/\ell} \cdot (\mu)^{a \cdot r_i/a} \cdot g_2^{-a} \cdot g_1^{-1/\partial} \cdot \upsilon^s \\
&= g_2^a \cdot g_2^a \cdot g_2^{-a} \cdot g^{-a/\ell} \cdot (\mu)^{a \cdot r_i/a} \cdot g_2^{-a} \cdot g^{-a/\partial} \cdot \upsilon^s \\
&= g_2^{2 \cdot a} \cdot (g_2^\ell \cdot g)^{-a/\ell} \cdot (\mu)^{a \cdot r_i/a} \cdot (g_2^\partial \cdot g)^{-a/\partial} \cdot \upsilon^s \\
&= g_2^{2 \cdot a} \cdot (\mu)^{-a/\ell} \cdot (\mu)^{a \cdot r_i/a} \cdot (\upsilon)^{-a/\partial} \cdot \upsilon^s \\
&= g_2^{2 \cdot a} \cdot \mu^{a \cdot [r_i/a - 1/\ell]} \cdot \upsilon^{s - a/\partial} \\
&= g_2^{2 \cdot a} \cdot \mu^{2 \cdot a \cdot r_i' \cdot H(\mathrm{ID}_i)} \cdot \upsilon^{s' \cdot H(w)}, \\
y_1 &= y_2 = (g_1^{-1/\ell} \cdot g^{r_i})^{1/2 \cdot H(\mathrm{ID}_i)} = (g_1^{-1/\ell} \cdot g^{a \cdot (r_i/a)})^{1/2 \cdot H(\mathrm{ID}_i)} \\
&= (g_1^{-1/\ell} \cdot g_1^{r_i/a})^{1/2 \cdot H(\mathrm{ID}_i)} = (g_1^{-1/\ell} \cdot g_1^{r_i/a})^{1/2 \cdot H(\mathrm{ID}_i)} \\
&= g_1^{[r_i/a - 1/\ell] \cdot (1/2 \cdot H(\mathrm{ID}_i))} = g_1^{r_i'}, \\
y_3 &= (g_1^{-1/\partial} \cdot g^s)^{1/H(w)} = (g^{s - a/\partial})^{1/H(w)} \\
&= g^{[s - a/\partial] \cdot (1/H(w))} = g^{s'}.
\end{aligned}$$

If $a \cdot \ell \cdot H(\mathrm{ID}_i) = 0 \mod q (g_1^{\ell \cdot H(\mathrm{ID}_i)} = 1)$ or $\partial \cdot H(w) = 0 \mod q$, then the above computation cannot be performed and the simulator will abort; otherwise, a proxy signing key psk_{pd} is passed to the adversary $\mathcal{A}$.

(III) $T = type3$

Given the public parameters and a warrant w of the delegator (the user i), the oracle returns a proxy signing key for the proxy signer (the user 1) on w to $\mathcal{A}$, where w may be an arbitrary forgery generated by $\mathcal{A}$ and ID_i is the identity of the user i with $i > 1$. In the third situation, the user i is a delegator, so the user i needs to delegate his signing rights to the user 1. Algorithm $\mathcal{B}$ randomly chooses $r_i, s \in \mathbb{Z}_q$, and then computes

$$\begin{aligned}
&y_0 = x_{1,0} \cdot g_1^{-1/2 \cdot \ell} \cdot \mu^{r_i} \cdot g_1^{-1/2 \cdot \partial} \cdot \upsilon^s, \quad y_1 = x_{1,1}, \\
&y_2 = (g_1^{-1/2 \cdot \ell} \cdot g^{r_i})^{1/H(\mathrm{ID}_i)}
\end{aligned}$$

and

$$y_3 = (g_1^{-1/2 \cdot \partial} \cdot g^s)^{1/H(w)},$$

where $\mathcal{B}$ gets the warrant w and the identity ID_i by the query of $\mathcal{A}$. And then algorithm $\mathcal{B}$ outputs a proxy signing key $psk_{pd} = \{y_0, y_1, y_2, y_3\}$. Similarly, setting $r_i' = (r_i/a - 1/2 \cdot \ell) \cdot (1/H(\mathrm{ID}_i))$ and $s' = [s - a/2 \cdot \partial] \cdot (1/H(w))$, $psk_{pd} = \{y_0, y_1, y_2, y_3\} = \{x_{1,0} \cdot g_2^a \cdot \mu^{a \cdot r_i' \cdot H(\mathrm{ID}_i)} \cdot \upsilon^{s' \cdot H(w)}, x_{1,1}, g_1^{r_i'}, g^{s'}\}$ is a valid proxy signing key.

Remark A.3. To the correctness of psk_{pd}, psk_{pd} may be changed as follows:

$$\begin{aligned}
y_0 &= x_{1,0} \cdot g_1^{-1/2\cdot\ell} \cdot \mu^{r_i} \cdot g_1^{-1/2\cdot\partial} \cdot \upsilon^s \\
&= x_{1,0} \cdot g_2^{a/2} \cdot g_2^{-a/2} \cdot g_2^{a/2} \cdot g_2^{-a/2} \cdot g_1^{-1/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \\
&\quad \cdot g_1^{-1/2\cdot\partial} \cdot \upsilon^s \\
&= x_{1,0} \cdot g_2^{a/2} \cdot g_2^{-a/2} \cdot g_2^{a/2} \cdot g_2^{-a/2} \cdot g^{-a/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \\
&\quad \cdot g^{-a/2\cdot\partial} \cdot \upsilon^s \\
&= x_{1,0} \cdot g_2^{a/2} \cdot g_2^{a/2} \cdot g_2^{-a/2} \cdot g^{-a/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \cdot g_2^{-a/2} \\
&\quad \cdot g^{-a/2\cdot\partial} \cdot \upsilon^s \\
&= x_{1,0} \cdot g_2^{a} \cdot (g_2^{\ell} \cdot g)^{-a/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \cdot (g_2^{\partial} \cdot g)^{-a/2\cdot\partial} \cdot \upsilon^s \\
&= x_{1,0} \cdot g_2^{a} \cdot \mu^{-a/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \cdot \upsilon^{-a/2\cdot\partial} \cdot \upsilon^s \\
&= x_{1,0} \cdot g_2^{a} \cdot \mu^{a\cdot[r_i/a-1/2\cdot\ell]} \cdot \upsilon^{s-a/2\cdot\partial} \\
&= x_{1,0} \cdot g_2^{a} \cdot \mu^{a\cdot r_i'\cdot H(\mathrm{ID}_i)} \cdot \upsilon^{s'\cdot H(w)}, \\
y_2 &= (g_1^{-1/2\cdot\ell} \cdot g^{r_i})^{1/H(\mathrm{ID}_i)} = (g_1^{-1/2\cdot\ell} \cdot g^{a\cdot r_i/a})^{1/H(\mathrm{ID}_i)} \\
&= (g_1^{-1/2\cdot\ell} \cdot g_1^{r_i/a})^{1/H(\mathrm{ID}_i)} \\
&= (g_1^{r_i/a-1/2\cdot\ell})^{1/H(\mathrm{ID}_i)} = g_1^{[r_i/a-1/2\cdot\ell]\cdot(1/H(\mathrm{ID}_i))} = g_1^{r_i'}, \\
y_3 &= (g_1^{-1/2\cdot\partial} \cdot g^{s})^{1/H(w)} = g^{[s-a/2\cdot\partial]\cdot(1/H(w))} = g^{s'}.
\end{aligned}$$

If $a \cdot \ell \cdot H(\mathrm{ID}_i) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_i)} = 1)$ or $\partial \cdot H(w) = 0 \bmod q$, then the above computation cannot be performed and the simulator will abort; otherwise, a proxy signing key psk_{pd} is passed to the adversary $\mathcal{A}$.

(IV) $T = type4$

Given the public parameters and a warrant w of the delegator (the user 1), the oracle returns a proxy signing key psk for the proxy signer (the user i) on w to $\mathcal{A}$, where w may be an arbitrary forgery generated by $\mathcal{A}$ and ID_i is the identity of the user i with $i > 1$. In the fourth situation, the user 1 is a delegator, so the user 1 needs to delegate his signing rights to the user i. Algorithm $\mathcal{B}$ randomly chooses $r_i, s \in \mathbb{Z}_q$, and then computes $y_0 = x_{1,0} \cdot g_1^{-1/2\cdot\ell} \cdot \mu^{r_i} \cdot g_1^{-1/2\cdot\partial} \cdot \upsilon^s$, $y_1 = (g_1^{-1/2\cdot\ell} \cdot g^{r_i})^{1/H(\mathrm{ID}_i)}$, $y_2 = x_{1,1}$ and $y_3 = (g_1^{-1/2\cdot\partial} \cdot g^{s})^{1/H(w)}$, where $\mathcal{B}$gets the warrant w and the identity ID_i by the query of $\mathcal{A}$. And then algorithm $\mathcal{B}$ outputs a proxy signing key $psk_{pd} = \{y_0, y_1, y_2, y_3\}$. Similarly, setting $r_i' = (r_i/a - 1/2\cdot\ell)\cdot(1/H(\mathrm{ID}_i))$ and $s' = [s - a/2\cdot\partial]\cdot(1/H(w))$, $psk_{pd} = \{y_0, y_1, y_2, y_3\} = \{x_{1,0} \cdot g_2^{a} \cdot \mu^{a\cdot r_i'\cdot H(\mathrm{ID}_i)} \cdot \upsilon^{s'\cdot H(w)}, g_1^{r_i'}, x_{1,1}, g^{s'}\}$ is a valid proxy signing key.

If $a \cdot \ell \cdot H(\mathrm{ID}_i) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_i)} = 1)$ or $\partial \cdot H(w) = 0 \bmod q$, then the above computation cannot be performed and the simulator will abort; otherwise, a proxy signing key psk_{pd} is passed to the adversary $\mathcal{A}$.

Signature Queries:

(I) $T = type1$

Given a message $\mathcal{M}$, the oracle returns a simulated standard IBS identity-based signature σ on $\mathcal{M}$ to $\mathcal{A}$, which is valid with an identity ID_i of the user i with $i > 1$. Thus, algorithm $\mathcal{B}$ randomly chooses $r_i, d \in \mathbb{Z}_q$, and computes

$$\begin{aligned}
X_0 &= g_1^{-1/2\cdot\ell} \cdot \mu^{r_i} \cdot g_1^{-1/2\cdot\lambda} \cdot \tau^{d} \cdot (\varpi^{d} \cdot g_1^{-\eta/2\cdot\lambda})^{1/H(\mathcal{M})}, \\
X_1 &= (g_1^{-1/2\cdot\ell} \cdot g^{r_i})^{1/H(\mathrm{ID}_i)}, \quad X_2 = (g_1^{-1/2\cdot\lambda} \cdot g^{d})^{1/H(\mathcal{M})}.
\end{aligned}$$

Lastly, $\mathcal{B}$ outputs an IBS $\sigma = \{X_0, X_1, X_2\}$. Setting $r_i' = (r_i/a - 1/2\cdot\ell)\cdot(1/H(\mathrm{ID}_i))$ and $d' = [d - a/2\cdot\lambda]\cdot(1/H(\mathcal{M}))$,, $\sigma = \{X_0, X_1, X_2\} = \{g_2^{a} \cdot \mu^{a\cdot r_i'\cdot H(\mathrm{ID}_i)} \cdot \varpi^{d'} \cdot \tau^{d'\cdot H(\mathcal{M})}, g_1^{r_i'}, g^{d'}\}$ is a valid IBS.

Remark A.4. To the correctness of σ, σ may be changed as follows:

$$\begin{aligned}
X_0 &= g_1^{-1/2\cdot\ell} \cdot \mu^{r_i} \cdot g_1^{-1/2\cdot\lambda} \cdot \tau^{d} \cdot (\varpi^{d} \cdot g_1^{-\eta/2\cdot\lambda})^{1/H(\mathcal{M})} \\
&= g_2^{a/2} \cdot g_2^{-a/2} \cdot g_2^{a/2} \cdot g_2^{-a/2} \cdot g_1^{-1/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \cdot g_1^{-1/2\cdot\lambda} \\
&\quad \cdot \tau^{d} \cdot (\varpi^{d} \cdot g_1^{-\eta/2\cdot\lambda})^{1/H(\mathcal{M})} \\
&= g_2^{a/2} \cdot g_2^{-a/2} \cdot g_2^{a/2} \cdot g_2^{-a/2} \cdot g^{-a/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \cdot g^{-a/2\cdot\lambda} \\
&\quad \cdot \tau^{d} \cdot (\varpi^{d} \cdot g_1^{-\eta/2\cdot\lambda})^{1/H(\mathcal{M})} \\
&= g_2^{a/2} \cdot g_2^{a/2} \cdot g_2^{-a/2} \cdot g^{-a/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \cdot g_2^{-a/2} \cdot g^{-a/2\cdot\lambda} \\
&\quad \cdot \tau^{d} \cdot (\varpi^{d} \cdot g^{-a\cdot\eta/2\cdot\lambda})^{1/H(\mathcal{M})} \\
&= g_2^{a} \cdot (g_2^{\ell} \cdot g)^{-a/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \cdot (g_2^{\lambda} \cdot g)^{-a/2\cdot\lambda} \cdot \tau^{d} \\
&\quad \cdot (\varpi^{d} \cdot \varpi^{-a/2\cdot\lambda})^{1/H(\mathcal{M})} \\
&= g_2^{a} \cdot \mu^{-a/2\cdot\ell} \cdot (\mu)^{a\cdot r_i/a} \cdot \tau^{-a/2\cdot\lambda} \cdot \tau^{d} \\
&\quad \cdot (\varpi^{d-a/2\cdot\lambda})^{1/H(\mathcal{M})} \\
&= g_2^{a} \cdot \mu^{a\cdot[r_i/a-1/2\cdot\ell]} \cdot \tau^{d-a/2\cdot\lambda} \cdot \varpi^{(d-a/2\cdot\lambda)\cdot(1/H(\mathcal{M}))} \\
&= g_2^{a} \cdot \mu^{a\cdot r_i'\cdot H(\mathrm{ID}_i)} \cdot \tau^{d'\cdot H(\mathcal{M})} \cdot \varpi^{d'} \\
&= g_2^{a} \cdot \mu^{a\cdot r_i'\cdot H(\mathrm{ID}_i)} \cdot \varpi^{d'} \cdot \tau^{d'\cdot H(\mathcal{M})}, \\
X_1 &= (g_1^{-1/2\cdot\ell} \cdot g^{r_i})^{1/H(\mathrm{ID}_i)} = (g_1^{-1/2\cdot\ell} \cdot g^{a\cdot r_i/a})^{1/H(\mathrm{ID}_i)} \\
&= (g_1^{-1/2\cdot\ell} \cdot g_1^{r_i/a})^{1/H(\mathrm{ID}_i)} \\
&= (g_1^{r_i/a-1/2\cdot\ell})^{1/H(\mathrm{ID}_i)} = g_1^{[r_i/a-1/2\cdot\ell]\cdot(1/H(\mathrm{ID}_i))} = g_1^{r_i'}, \\
X_2 &= (g_1^{-1/2\cdot\lambda} \cdot g^{d})^{1/H(\mathcal{M})} = (g^{d-a/2\cdot\lambda})^{1/H(\mathcal{M})} \\
&= g^{[d-a/2\cdot\lambda]\cdot(1/H(\mathcal{M}))} = g^{d'}.
\end{aligned}$$

If $a \cdot \ell \cdot H(\mathrm{ID}_i) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_i)} = 1)$ or $\lambda \cdot H(\mathcal{M}) = 0 \bmod q$, then the above computation cannot be performed and the simulator will abort; otherwise, a valid IBS σ is passed to the adversary $\mathcal{A}$.

(II) $T = type2$

Given a message $\mathcal{M}$ and a warrant w, the oracle returns a simulated IBPS $p\sigma$ on $\mathcal{M}$ and w to $\mathcal{A}$, which is valid with respect to the corresponding identities. In the second situation, the user i is a delegator and is also a proxy signer (ID_i is the identity of the user i with $i > 1$). Thus, algorithm $\mathcal{B}$ randomly

chooses $r_i, s, d \in \mathbb{Z}_q$, and computes

$$Y_0 = g_1^{-1/\ell} \cdot \mu^{r_i} \cdot g_1^{-1/\lambda} \cdot \upsilon^{s \cdot H(w)} \cdot \tau^d \cdot (\varpi^d \cdot g_1^{-\eta/\lambda})^{1/H(\mathcal{M})},$$
$$Y_1 = Y_2 = (g_1^{-1/\ell} \cdot g^{r_i})^{1/2 \cdot H(\mathrm{ID}_i)}$$
$$Y_3 = g^s \quad \text{and} \quad Y_4 = (g_1^{-1/\lambda} \cdot g^d)^{1/H(\mathcal{M})}.$$

Lastly, $\mathcal{B}$ outputs an IBPS $p\sigma = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$.

Similarly, setting $r_i' = [r_i/a - 1/\ell] \cdot 1/2 \cdot H(\mathrm{ID}_i)$ and $d' = [d - a/\lambda] \cdot (1/H(\mathcal{M}))$, $p\sigma = [Y_0, Y_1, Y_2, Y_3, Y_4] = \{g_2^{2 \cdot a} \cdot \mu^{2 \cdot a \cdot r_i' \cdot H(\mathrm{ID}_i)} \cdot \upsilon^{s \cdot H(w)} \cdot \varpi^{d'} \cdot \tau^{d' \cdot H(M)}, g_1^{r_i'}, g_1^{r_i'}, g^s, g^{d'}\}$ is a valid IBPS.

If $a \cdot \ell \cdot H(\mathrm{ID}_i) = 0 \bmod q (g_1^{\ell \cdot H(\mathrm{ID}_i)} = 1)$ or $\lambda \cdot H(M) = 0 \bmod q$, then the above computation cannot be performed and the simulator will abort; otherwise, a valid IBPS $p\sigma$ is passed to the adversary $\mathcal{A}$.

(III) $T = type3$

Given a message $\mathcal{M}$ and a warrant w, the oracle returns a simulated IBPS $p\sigma$ on $\mathcal{M}$ and w to $\mathcal{A}$, which is valid with respect to the corresponding identities. In the third situation, the user i is a delegator, so the user i needs to delegate his signing rights to the user 1 (ID_i is the identity of the user i with $i > 1$). Thus, algorithm $\mathcal{B}$ randomly chooses $r_i, s, d \in \mathbb{Z}_q$, and computes

$$Y_0 = x_{1,0} \cdot g_1^{-1/2 \cdot \ell} \cdot \mu^{r_i} \cdot g_1^{-1/2 \cdot \lambda} \cdot \upsilon^{s \cdot H(w)} \cdot \tau^d \cdot (\varpi^d \cdot g_1^{-\eta/2 \cdot \lambda})^{1/H(\mathcal{M})}, \quad Y_1 = x_{1,1},$$
$$Y_2 = (g_1^{-1/2 \cdot \ell} \cdot g^{r_i})^{1/H(\mathrm{ID}_i)}, \quad Y_3 = g^s \quad \text{and}$$
$$Y_4 = (g_1^{-1/2 \cdot \lambda} \cdot g^d)^{1/H(\mathcal{M})}.$$

Lastly, $\mathcal{B}$ outputs an IBPS $p\sigma = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$.

Setting $r_i' = (r_i/a - 1/2 \cdot \ell) \cdot (1/H(\mathrm{ID}_i))$ and $d' = [d - a/2 \cdot \lambda] \cdot (1/H(\mathcal{M}))$, $p\sigma = \{Y_0, Y_1, Y_2, Y_3, Y_4\} = \{x_{1,0} \cdot g_2^a \cdot \mu^{a \cdot r_i' \cdot H(\mathrm{ID}_i)} \cdot \upsilon^{s \cdot H(w)} \cdot \varpi^{d'} \cdot \tau^{d' \cdot H(\mathcal{M})}, x_{1,1}, g_1^{r_i'}, g^s, g^{d'}\}$ is a valid IBPS.

If $a \cdot \ell \cdot H(\mathrm{ID}_i) = 0 \bmod q (g_1^{\ell \cdot H(\mathrm{ID}_i)} = 1)$ or $\lambda \cdot H(\mathcal{M}) = 0 \bmod q$, then the above computation cannot be performed and the simulator will abort; otherwise, a valid IBPS $p\sigma$ is passed to the adversary $\mathcal{A}$.

(IV) $T = type4$

Given a message $\mathcal{M}$ and a warrant w, the oracle returns a simulated IBPS $p\sigma$ on $\mathcal{M}$ and w to $\mathcal{A}$, which is valid with respect to the corresponding identities. In the fourth situation, the user 1 is a delegator, so the user 1 needs to delegate his signing rights to the user i (ID_i is the identity of the user i with $i > 1$). Thus, algorithm $\mathcal{B}$ randomly chooses $r_i, s, d \in \mathbb{Z}_q$, and computes

$$Y_0 = x_{1,0} \cdot g_1^{-1/2 \cdot \ell} \cdot \mu^{r_i} \cdot g_1^{-1/2 \cdot \lambda} \cdot \upsilon^{s \cdot H(w)} \cdot \tau^d \cdot (\varpi^d \cdot g_1^{-\eta/2 \cdot \lambda})^{1/H(\mathcal{M})},$$
$$Y_1 = (g_1^{-1/2 \cdot \ell} \cdot g^{r_i})^{1/H(\mathrm{ID}_i)}, \quad Y_2 = x_{1,1},$$
$$Y_3 = g^s \quad \text{and} \quad Y_4 = (g_1^{-1/2 \cdot \lambda} \cdot g^d)^{1/H(\mathcal{M})}.$$

Lastly, $\mathcal{B}$ outputs an IBPS $p\sigma = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$.

Setting $r_i' = (r_i/a - 1/2 \cdot \ell) \cdot (1/H(\mathrm{ID}_i))$ and $d' = [d - a/2 \cdot \lambda] \cdot (1/H(\mathcal{M}))$, $p\sigma = \{Y_0, Y_1, Y_2, Y_3, Y_4\} = \{x_{1,0} \cdot g_2^a \cdot \mu^{a \cdot r_i' \cdot H(\mathrm{ID}_i)} \cdot \upsilon^{s \cdot H(w)} \cdot \varpi^{d'} \cdot \tau^{d' \cdot H(M)}, g_1^{r_i'}, x_{1,1}, g^s, g^{d'}\}$ is a valid IBPS.

If $a \cdot \ell \cdot H(\mathrm{ID}_i) = 0 \bmod q (g_1^{\ell \cdot H(\mathrm{ID}_i)} = 1)$ or $\lambda \cdot H(\mathcal{M}) = 0 \bmod q$, then the above computation cannot be performed and the simulator will abort; otherwise, a valid IBPS $p\sigma$ is passed to the adversary $\mathcal{A}$.

Forgery: If algorithm $\mathcal{B}$ does not abort as a consequence of one of the queries above, the adversary $\mathcal{A}$ will, with probability at least ε, return a forgery according to the following situation:

(I) $T = type1$

The adversary $\mathcal{A}$ returns a message $\mathcal{M}^*$ and a valid standard signature forgery for ID^*, $\sigma^* = \{X_0^*, X_1^*, X_2^*\} = \{g_2^a \cdot \mu^{a \cdot r^* \cdot H(\mathrm{ID}^*)} \cdot \varpi^{d^*} \cdot \tau^{d^* \cdot H(\mathcal{M}^*)}, g_1^{r^*}, g^{d^*}\}$, where $\mathcal{A}$ did not query ***req_sig***$(\cdot)$ on input $\mathcal{M}^*$ and did not query ***req_key***() for ID^*.

If $a \cdot \ell \cdot H(\mathrm{ID}^*) \neq 0 \bmod q (g_1^{\ell \cdot H(\mathrm{ID}^*)} \neq 1)$ or $\lambda \cdot H(\mathcal{M}^*) \neq 0 \bmod q$, then algorithm $\mathcal{B}$ will abort.

If $a \cdot \ell \cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell \cdot H(\mathrm{ID}^*)} = 1)$ and $\lambda \cdot H(\mathcal{M}^*) = 0 \bmod q$, then algorithm $\mathcal{B}$ computes and outputs

$$\frac{X_0^*}{(g_1^{r^*})^{H(\mathrm{ID}^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}} = \frac{g_2^a \cdot \mu^{a \cdot r^* \cdot H(\mathrm{ID}^*)} \cdot \varpi^{d^*} \cdot \tau^{d^* \cdot H(\mathcal{M}^*)}}{(g_1^{r^*})^{H(\mathrm{ID}^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}}$$
$$= \frac{g_2^a \cdot (g_2^{\ell} \cdot g)^{a \cdot r^* \cdot H(\mathrm{ID}^*)} \cdot (g^{\eta})^{d^*} \cdot (g_2^{\lambda} \cdot g)^{d^* \cdot H(\mathcal{M}^*)}}{(g_1^{r^*})^{H(\mathrm{ID}^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}}$$
$$= g_2^a = g^{a \cdot b},$$

which is the solution to the given CDH problem.

(II) $T = type2$

The adversary $\mathcal{A}$ returns a message $\mathcal{M}^*$, and a valid identity-based self-PS forgery, $p\sigma^* = \{Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*\} = \{g_2^{2 \cdot a} \cdot \mu^{2 \cdot a \cdot r^* \cdot H(\mathrm{ID}^*)} \cdot \upsilon^{s^* \cdot H(w^*)} \cdot \varpi^{d^*} \cdot \tau^{d^* \cdot H(\mathcal{M}^*)}, g_1^{r^*}, g_1^{r^*}, g^{s^*}, g^{d^*}\}$ for ID^* and w^*, where $\mathcal{A}$ did not query ***req_sig***$_w(\cdot)$ on input $\mathcal{M}^*$ and w and did not query ***req_key***() for ID^* and did not query ***req_proxykey***(,) for w^* and ID^*.

If $a \cdot \ell \cdot H(\mathrm{ID}^*) \neq 0 \bmod q (g_1^{\ell \cdot H(\mathrm{ID}^*)} \neq 1)$ or $\partial \cdot H(w^*) \neq 0 \bmod q$ or $\lambda \cdot H(\mathcal{M}^*) \neq 0 \bmod q$, then algorithm $\mathcal{B}$ will abort.

If $a \cdot \ell \cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell \cdot H(\mathrm{ID}^*)} = 1)$ and $\partial \cdot H(w^*) = 0 \bmod q$ and $\lambda \cdot H(M^*) = 0 \bmod q$, then algorithm $\mathcal{B}$ computes and outputs

$$\left(\frac{Y_0^*}{(g_1^{r^*})^{2 \cdot H(\mathrm{ID}^*)} \cdot (g^{s^*})^{H(w^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}}\right)^{1/2} = \left(\frac{g_2^{2 \cdot a} \cdot \mu^{2 \cdot a \cdot r^* \cdot H(\mathrm{ID}^*)} \cdot \upsilon^{s^* \cdot H(w^*)} \cdot \varpi^{d^*} \cdot \tau^{d^* \cdot H(\mathcal{M}^*)}}{(g_1^{r^*})^{2 \cdot H(\mathrm{ID}^*)} \cdot (g^{s^*})^{H(w^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}}\right)^{1/2}$$

$$= \left(\frac{g_2^{2\cdot a} \cdot (g_2^{\ell} \cdot g)^{2\cdot a\cdot r^*\cdot H(\mathrm{ID}^*)} \cdot (g_2^{\partial} \cdot g)^{s^*\cdot H(w^*)} \cdot (g^{\eta})^{d^*} \cdot (g_2^{\lambda} \cdot g)^{d^*\cdot H(\mathcal{M}^*)}}{(g_1^{r^*})^{2\cdot H(\mathrm{ID}^*)} \cdot (g^{s^*})^{H(w^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}} \right)^{1/2}$$
$$= (g_2^{2\cdot a})^{1/2} = g_2^a = g^{a\cdot b},$$

which is the solution to the given CDH problem.

(III) $T = type3$

The adversary $\mathcal{A}$ returns a message $\mathcal{M}^*$, and a valid IBPS forgery, $p\sigma^* = \{Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*\} = \{x_{1,0} \cdot g_2^a \cdot \mu^{a\cdot r^*\cdot H(\mathrm{ID}^*)} \cdot \upsilon^{s^*\cdot H(w^*)} \cdot \varpi^{d^*} \cdot \tau^{d^*\cdot H(\mathcal{M}^*)}, x_{1,1}, g_1^{r^*}, g^{s^*}, g^{d^*}\}$ for w^*, ID^* and ID_1, where $\mathcal{A}$ did not query ***req_sig***$_w(\cdot)$ on input $\mathcal{M}^*$ and w^* and did not query ***req_proxykey***(,) for w^*, ID^* and ID_1.

If $a \cdot \ell \cdot H(\mathrm{ID}^*) \neq 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} \neq 1)$ or $\partial \cdot H(w^*) \neq 0 \bmod q$ or $\lambda \cdot H(\mathcal{M}^*) \neq 0 \bmod q$, then algorithm B will abort.

If $a \cdot \ell \cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} = 1)$ and $\partial \cdot H(w^*) = 0 \bmod q$ and $\lambda \cdot H(\mathcal{M}^*) = 0 \bmod q$, then algorithm $\mathcal{B}$ computes and outputs

$$\frac{Y_0^*}{x_{1,0} \cdot (g_1^{r^*})^{H(\mathrm{ID}^*)} \cdot (g^{s^*})^{H(w^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}}$$
$$= \frac{x_{1,0} \cdot g_2^a \cdot \mu^{a\cdot r^*\cdot H(\mathrm{ID}^*)} \cdot \upsilon^{s^*\cdot H(w^*)} \cdot \varpi^{d^*} \cdot \tau^{d^*\cdot H(\mathcal{M}^*)}}{x_{1,0} \cdot (g_1^{r^*})^{H(\mathrm{ID}^*)} \cdot (g^{s^*})^{H(w^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}}$$
$$= \frac{g_2^a \cdot (g_2^{\ell} \cdot g)^{a\cdot r^*\cdot H(\mathrm{ID}^*)} \cdot (g_2^{\partial} \cdot g)^{s^*\cdot H(w^*)} \cdot (g^{\eta})^{d^*} \cdot (g_2^{\lambda} \cdot g)^{d^*\cdot H(\mathcal{M}^*)}}{(g_1^{r^*})^{H(\mathrm{ID}^*)} \cdot (g^{s^*})^{H(w^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}}$$
$$= g_2^a = g^{a\cdot b},$$

which is the solution to the given CDH problem.

(IV) $T = type4$

The adversary $\mathcal{A}$ returns a message $\mathcal{M}^*$, and a valid IBPS forgery, $p\sigma^* = \{Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*\} = \{x_{1,0} \cdot g_2^a \cdot \mu^{a\cdot r^*\cdot H(\mathrm{ID}^*)} \cdot \upsilon^{s^*\cdot H(w^*)} \cdot \varpi^{d^*} \cdot \tau^{d^*\cdot H(\mathcal{M}^*)}, g_1^{r^*}, x_{1,1}, g^{s^*}, g^{d^*}\}$ for w^*, ID_1 and ID^*, where $\mathcal{A}$ did not query ***req_sig***$_w(\cdot)$ on input $\mathcal{M}^*$ and w^* and did not query ***req_proxykey***(,) for w^*, ID_1 and ID^*.

If $a \cdot \ell \cdot H(\mathrm{ID}^*) \neq 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} \neq 1)$ or, $\partial \cdot H(w^*) \neq 0 \bmod q$ or $\lambda \cdot H(\mathcal{M}^*) \neq 0 \bmod q$, then algorithm $\mathcal{B}$ will abort.

If $a \cdot \ell \cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} = 1)$ and $\partial \cdot H(w^*) = 0 \bmod q$ and $\lambda \cdot H(\mathcal{M}^*) = 0 \bmod q$, then algorithm $\mathcal{B}$ computes and outputs

$$\frac{Y_0^*}{x_{1,0} \cdot (g_1^{r^*})^{H(\mathrm{ID}^*)} \cdot (g^{s^*})^{H(w^*)} \cdot (g^{d^*})^{\eta} \cdot (g^{d^*})^{H(\mathcal{M}^*)}}$$
$$= g_2^a = g^{a\cdot b},$$

which is the solution to the given CDH problem.

Now, we analyze the probability of algorithm $\mathcal{B}$ not aborting. For the simulation to complete without aborting, we require that all key queries will have $a\cdot\ell\cdot H(\mathrm{ID}_i) \neq 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_i)} \neq 1)$ and $\partial \cdot H(w) \neq 0 \bmod q$ whenever $T = type2$ or $type3$ or $type4$ (obviously the condition includes $a\cdot\ell\cdot H(\mathrm{ID}_i) \neq 0 \bmod q$ when $T = type1$), and all signature queries will have $a \cdot \ell \cdot H(\mathrm{ID}_i) \neq 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_i)} \neq 1)$ and $\lambda \cdot H(\mathcal{M}) \neq 0 \bmod q$ whenever $T = type1$ or $type2$ or $type3$ or $type4$, and that $a \cdot \ell \cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} = 1)$ and $\lambda \cdot H(\mathcal{M}^*) = 0 \bmod q$ when $T = type1$, and $a \cdot \ell \cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} = 1)$ and $\partial \cdot H(w^*) = 0 \bmod q$ and $\lambda \cdot H(\mathcal{M}^*) = 0 \bmod q$ whenever $T = type2$ or $type3$ or $type4$ in forgery. If algorithm $\mathcal{B}$ does not abort, then the following three conditions must hold:

(a) $a\cdot\ell\cdot H(\mathrm{ID}_i) \neq 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_i)} \neq 1)$ and $\partial\cdot H(w_i) \neq 0 \bmod q$ in key queries, with $i = 1, 2 \ldots, q_e$;
(b) $a \cdot \ell \cdot H(\mathrm{ID}_j) \neq 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_j)} \neq 1)$ and $\lambda \cdot H(\mathcal{M}_j) \neq 0 \bmod q$ in signature queries, with $j = 1, 2 \ldots, q_s$;
(c) algorithm $\mathcal{B}$ does not abort in forgery, namely $a \cdot \ell \cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} = 1)$ and $\partial \cdot H(w^*) = 0 \bmod q$ and $\lambda \cdot H(\mathcal{M}^*) = 0 \bmod q$ in forgery (the condition includes $a\cdot\ell\cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} = 1)$ and $\lambda \cdot H(\mathcal{M}^*) = 0 \bmod q$ when $T = type1$).

Thus, we will provide an upper bound on the probability that $\mathcal{B}$ aborts. To make the analysis simpler, we will define the events E_i, F_i, T_j, S_j, R^*, F^*, S^* as

E_i: $a \cdot \ell \cdot H(\mathrm{ID}_i) \neq 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_i)} \neq 1)$, with $i = 1, 2 \ldots, q_e$, where q_e is the maximal number of key queries;
F_i: $\partial \cdot H(w_i) \neq 0 \bmod q$, with $i = 1, 2 \ldots, q_e$, where q_e is the maximal number of key queries;
T_j: $a \cdot \ell \cdot H(\mathrm{ID}_j) \neq 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}_j)} \neq 1)$, with $j = 1, 2 \ldots, q_s$, where q_s is the maximal number of signature queries;
S_j: $\lambda \cdot H(\mathcal{M}_j) \neq 0 \bmod q$, with $j = 1, 2 \ldots, q_s$, where q_s is the maximal number of signature queries;
R^*: $a \cdot \ell \cdot H(\mathrm{ID}^*) = 0 \bmod q (g_1^{\ell\cdot H(\mathrm{ID}^*)} = 1)$;
F^*: $\partial \cdot H(w^*) = 0 \bmod q$;
S^*: $\lambda \cdot H(\mathcal{M}^*) = 0 \bmod q$.

Then the probability of $\mathcal{B}$ not aborting is

$$\Pr(not\_abort) = \Pr\left(\bigcap_{i=1}^{q_e} (\mathrm{E}_i \wedge \mathrm{F}_i) \wedge \bigcap_{j=1}^{q_s} (\mathrm{T}_j \wedge \mathrm{S}_j) \wedge \mathrm{R}^* \wedge \mathrm{F}^* \wedge \mathrm{S}^*\right).$$

It is easy to see that the events $\bigcap_{i=1}^{q_e} \mathrm{E}_i$, $\bigcap_{i=1}^{q_e} \mathrm{F}_i$, $\bigcap_{j=1}^{q_s} \mathrm{T}_j$, $\bigcap_{j=1}^{q_s} \mathrm{S}_j$, R^*, F^* and S^* are independent. Then we may compute

$$\Pr\left(\bigcap_{i=1}^{q_e} \mathrm{E}_i\right) = 1 - \Pr\left(\bigcup_{i=1}^{q_e} \neg\mathrm{E}_i\right) = 1 - q_e \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_e}{q};$$
$$\Pr\left(\bigcap_{i=1}^{q_e} \mathrm{F}_i\right) = 1 - \Pr\left(\bigcup_{i=1}^{q_e} \neg\mathrm{F}_i\right) = 1 - q_e \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_e}{q};$$

$$\Pr\left(\bigcap_{j=1}^{q_s} T_j\right) = 1 - \Pr\left(\bigcup_{j=1}^{q_s} \neg T_j\right) = 1 - q_s \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_s}{q};$$

$$\Pr\left(\bigcap_{j=1}^{q_s} \mathrm{S}_j\right) = 1 - \Pr\left(\bigcup_{j=1}^{q_s} \neg \mathrm{S}_j\right) = 1 - q_s \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_s}{q};$$

$$\Pr(\mathrm{R}^*) = \frac{1^k}{1^k \cdot q} = \frac{1}{q}; \quad \Pr(\mathrm{F}^*) = \frac{1^k}{1^k \cdot q} = \frac{1}{q};$$

$$\Pr(\mathrm{S}^*) = \frac{1^k}{1^k \cdot q} = \frac{1}{q}.$$

Thus,

$$\begin{aligned}
&\Pr(not_abort) \\
&= \Pr\left(\bigcap_{i=1}^{q_e} (\mathrm{E}_i \wedge \mathrm{F}_i) \wedge \bigcap_{j=1}^{q_s} (\mathrm{T}_j \wedge \mathrm{S}_j) \wedge \mathrm{R}^* \wedge \mathrm{F}^* \wedge \mathrm{S}^*\right) \\
&= \Pr\left(\bigcap_{i=1}^{q_e} \mathrm{E}_i\right) \cdot \Pr\left(\bigcap_{i=1}^{q_e} \mathrm{F}_i\right) \cdot \Pr\left(\bigcap_{j=1}^{q_s} \mathrm{T}_j\right) \cdot \Pr\left(\bigcap_{j=1}^{q_s} \mathrm{S}_j\right) \\
&\quad \cdot \Pr(\mathrm{R}^*) \cdot \Pr(\mathrm{F}^*) \cdot \Pr(\mathrm{S}^*) \\
&= \left(1 - \frac{q_e}{q}\right)^2 \cdot \left(1 - \frac{q_s}{q}\right)^2 \cdot \frac{1}{q^3}.
\end{aligned}$$

So we can get that $\varepsilon' = (1 - q_e/q)^2 \cdot (1 - q_s/q)^2 \cdot \varepsilon/q^3$.

If the simulation does not abort, adversary $\mathcal{A}$ will create a valid signature forgery with probability at least ε. Algorithm $\mathcal{B}$ can then compute g^{ab} from the forgery as shown above. The time complexity of algorithm $\mathcal{B}$ is dominated by the time for the exponentiations and multiplications in the queries. We assume that the time for integer addition and integer multiplication and the time for hash computation can both be ignored, and then the time complexity of algorithm $\mathcal{B}$ is

$$t' = t + O(q_e \cdot (14 \cdot C_{\mathrm{mul}} + 21 \cdot C_{\mathrm{exp}}) + q_s \cdot (27 \cdot C_{\mathrm{mul}} + 38 \cdot C_{\mathrm{exp}})).$$

Thus, Theorem 7.1 follows.