How to Construct Identity-Based Signatures without the Key Escrow Problem[⋆]

Tsz Hon Yuen, Willy Susilo, and Yi Mu

University of Wollongong, Australia
{thy738, wsusilo, ymu}@uow.edu.au

Abstract. The inherent key escrow problem is one of the main reasons for the slow adoption of identity-based cryptography. The existing solution for mitigating the key escrow problem is by adopting multiple Private Key Generators (PKGs). Recently, there was a proposal that attempted to reduce the trust of the PKG by allowing a malicious PKG to be caught if he reveals the user's identity-based secret key illegally. Nonetheless, the proposal does not consider that the PKG can simply decrypt the ciphertext instead of revealing the secret key itself (in the case of identity-based encryption schemes).

The aim of this paper is to present an escrow-free identity-based signature (IBS) scheme, in which the malicious PKG will be caught if it releases a signature on behalf of the user but signed by itself. We present a formal model to capture such a scheme and provide a concrete construction.

1 Introduction

The notion of identity-based cryptography was put forth by Shamir [18]. This notion was proposed to simplify the authentication of a public key by merely using an identity string as the public key. From the verifier's or the encryptor's point of view, only the identity of the other party is required. Hence, there is no necessity to ensure the validity of the public key. Due to this nice property, a series of identity-based schemes have been proposed, including identity-based signatures [18], identity-based encryption [7], hierarchical identity-based cryptography [13] and so forth. For identity-based signatures (IBS), there exists a comprehensive discussion conducted by Bellare *et al.* [4]. Galindo *et al.* [11] further extended the discussion to IBS with various additional properties, which has more practical applications.

In these identity-based cryptosystems, there is a trusted party called the private key generator (PKG) who generates the secret key for each user identity. As the PKG generates and holds the secret key for all users, a complete trust must be placed on the PKG. Nonetheless, this may not be desirable in a real world scenario, where a malicious PKG can sell users' keys, sign messages or decrypt ciphertexts on behalf of users without being confronted in a court of law. This is known as the *key escrow problem*. This problem seems to be inherent in identity-based cryptosystems. Boneh and Franklin [7] proposed that employing multiple PKGs is a possible solution to the key escrow problem. The master secret key is jointly computed by a number of PKGs, such that no single PKG has the knowledge of it. However, this approach requires an extra infrastructure and communication cost between users and different PKGs. A user needs to run the key extraction protocol with different PKGs by proving his identity to them. Furthermore, maintaining multiple independent PKGs for a commercially used infrastructure is a daunting task.

[⋆] This is the full version of the EuroPKI 2009 paper: T. H. Yuen, W. Susilo and Y. Mu, "How to Construct Identity-Based Signatures without the Key Escrow Problem"

Some cryptosystems have been proposed to solve the the key escrow problem. They use a "combination" of identity-based cryptography and the traditional public key cryptography, such as the certificateless cryptosystems [1], the certificate-based cryptosystems [12] and the self-certificated cryptosystems [14], in a non-trivial way. In these systems, a user possesses a user public key and a user secret key, together with his identity-based secret key computed by the PKG. The user secret key protects the user from the key escrow problem. The PKG acts like a certificate authority (CA) who authenticates the user public key using his master secret key. Unfortunately, these cryptosystems are *no longer* identity-based - the encryptor or the verifier has to know the user public key in addition to the user identity. Therefore these schemes lost the original advantages of identity-based cryptography.

Girault [14] defined three level of trust to the PKG:

- Level 1: The PKG can compute users' secret keys and, therefore, can impersonate any user without being detected. Identity-based signature schemes are the examples.
- Level 2: The PKG cannot compute users' secret keys. However, the PKG can still impersonate any user without being detected. Certificateless signature schemes are the examples.
- Level 3: The PKG cannot compute users' secret keys, and the PKG cannot impersonate any user without being detected. Certificate-based signature schemes and self-certificated signature schemes are the examples.

The current schemes achieving level 2 or level 3 of trust are no longer identity-based. It is an open problem to construct an identity-based signatures with level 2 or level 3 of trust, without publishing the user public key.

Schemes	Public Information	Level of Trust
Identity-based Signatures [18]	ID	Level 1
Certificateless Sigatures [1]	ID, upk, W	Level 2
Certificate-based Signatures [17]	ID, upk	Level 3
Self-certificated Signatures [14]	ID, upk	Level 3
Our Scheme in §6	ID	Level 3

Table 1. Comparison of the public information known by the verifier and the level of trust to the PKG. ID is the identity, upk is the user public key, and W is the commitment of the user secret key using the public key of the PKG.

Recently, Goyal [15] proposed the concept of accountable authority identity-based encryption (A-IBE) to reduce the trust in the PKG and it was further strengthened by [2, 16]. In [15], the PKG helps the user to compute his identity-based secret key without knowing it. If the PKG computes another set of secret key by himself and reveals it to other parties, this key will be different from the user's original secret key with a high probability. Therefore the PKG can be caught when revealing the secret key and the user's original secret key is the evidence. However, the malicious PKG is still able to sell a signed message or decrypted ciphertext instead, without being detected. This is clearly an issue that is not yet addressed in Goyal's model [15]. Goyal *et al.* [16] further proposed the concept of black-box A-IBE. In black-box A-IBE, if a PKG sells a decoder box which can decrypt ciphertexts with non-negligible probability, he will be caught in a trace algorithm. It is an open problem to construct a similar blaming mechanism in the IBS setting.

Our Contributions. In this paper, we introduce the concept of *escrow-free identity-based signatures* to reduce the trust in the PKG. In this model, each signer has his own public key and secret key. The PKG generates the identity-based secret key for the signer with respect to the user public key (*à la* Goyal's approach [15]). Then, the signer uses both secret keys to sign a message. Therefore, the signer is protected against a malicious PKG. To verify the signature, it only requires the signer's identity and the message. This is the main difference between certificate-based signatures (CBS), certificateless signatures (CLS), self-certificated signatures (SCS) and our model. Their verification protocols require the signer's public key to verify. Hence, our model mimics closely the original IBS in this regard, and solves the key escrow problem at the same time.

Our scheme achieves level 3 of trust to the PKG, which is the best in the model proposed by Girault [14]. Theoretically, the escrow-free IBS is more efficient than CBS, CLS and SCS since the user public key is not involved and is not sent to the verifier. In this paper, we give the *first* construction of the escrow-free IBS. When comparing with the multiple PKGs solution by Boneh and Franklin [7], our scheme interacts with at most two authorities. While Boneh and Franklin's scheme interacts with a large number of authorities, the communication complexity of the their scheme is higher.

We then extend the escrow-free IBS to have an extra property called *user public key anonymity.* In CBS, CLS and SCS, user public keys are needed to verify a signature. Since the escrow-free IBS only use the identity to verify a signature, it is possible for the signature to be anonymous with respect to the user public key. We provide an additional security model to capture the user public key anonymity property and present a secure construction with anonymity.

2 Backgrounds

We briefly review the pairings and some candidate hard problems that will be used later. Let $\mathbb{G}, \mathbb{G}_T$ be cyclic groups of prime order p, writing the group action multiplicatively. Let g be a generator of $\mathbb{G}$. A map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is called a pairings if, for all $g \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$, and if g is a generator of $\mathbb{G}$, then $\hat{e}(g, g)$ generates $\mathbb{G}_T$.

DL Problem. The Discrete Logarithm problem is that, given g, $y \in \mathbb{G}$, to output $x = \log_g y$. We say that the (ϵ, t)-DL assumption holds in $\mathbb{G}$ if no t-time algorithm has the non-negligible probability ϵ in solving the DL problem.

DBDH Problem [7]. The decisional Bilinear Diffie-Hellman problem is that, given g, g^a, g^b, $g^c \in \mathbb{G}$ and $T \in \mathbb{G}_T$ for unknown $a, b, c \in \mathbb{Z}_p^*$, to decide if $T = \hat{e}(g, g)^{abc}$. We say that the (ϵ, t)-DBDH assumption holds in $\mathbb{G}$ if no t-time algorithm has the non-negligible probability ϵ over half in solving the DBDH problem.

***q*-SDH Problem [6].** The q-Strong Diffie-Hellman problem is that, given g, g^{α}, $\ldots$, $g^{\alpha^q} \in \mathbb{G}$ for unknown $\alpha \in \mathbb{Z}_p^*$, to output a pair $(g^{\frac{1}{\alpha+c}}, c)$ where $c \in \mathbb{Z}_p^*$. We say that the (ϵ, t, q)-SDH assumption holds in $\mathbb{G}$ if no t-time algorithm has the non-negligible probability ϵ in solving the q-SDH problem.

3 Security Model for Escrow-free Identity-based Signatures

3.1 Syntax

An escrow-free identity-based signature scheme has six polynomial-time algorithms, namely Setup, $\mathsf{UserKeyGen}$, $\mathsf{Extract}$, Sign, Verify, Blame.

1. Setup: On input a security parameter 1^k, it generates the system parameter param, the master secret key msk and the master public key mpk.
2. $\mathsf{UserKeyGen}$: On input the system parameter param, the user generates the user secret key usk and the user public key upk.
3. $\mathsf{Extract}$: This is an interactive algorithm between the PKG and the user. The common input are param, upk and an identity ID. The PKG's algorithm $\mathsf{Extract}_p$ private input is msk. The user's algorithm $\mathsf{Extract}_u$ private input is usk. The interaction includes the user giving the PKG a joining proof Pf which shows the user's participation with respect to upk[1]. Finally the user obtains the identity-based secret key sk_{ID}.
4. Sign: On input param, usk, sk_{ID} and a message m, the user with identity ID generates a signature σ.
5. Verify: On input param, mpk, ID, m and σ, it returns 1 or 0 for accept or reject, respectively.
6. Blame: This is an interactive algorithm between the PKG, the user and the judge. The common input are param, mpk, ID, upk, m and σ. The user's algorithm Blame_u with private input usk sends a blame request φ to a judge. The judge's algorithm Blame_j outputs "PKG" if:
   - φ shows that σ is related to upk, and
   - the PKG's algorithm Blame_p, with private input msk, fails to provide a public key upk', a joining proof Pf and a transcript ρ, such that:
     - upk' is related to σ,
     - Pf shows the user's participation with respect to upk', and
     - ρ is the transcript of the extract algorithm with upk'.

   Otherwise, the judge outputs "upk".

3.2 Joining Proof

The joining proof Pf can be either an online proof or a proof in the real world. For the online proof, it can consist of a certificate issued by some authority with respect to upk, and a proof of knowledge with respect to upk. For the real world proof, it can be the user's signature on an application form, or the photocopy of the user's documentation.

The joining proof Pf is needed to protect both the PKG and the user in the Blame protocol. If there is no such proof:

- a malicious PKG can generate sk_{ID} using any upk generated by himself and an honest user cannot show that upk is not his public key;
- a malicious user can claim that the upk used in sk_{ID} is not his public key and frame an honest PKG.

The joining proof can be viewed as an authentication of user public key, which is separated from the identity-based secret key issuing. Similar concepts can be found in "anonymous

[1] The joining proof will be defined in section 3.2

identity-based key issuing" [19], where the duties of authentication and key issuing are separated to local registration authorities (LRA) and the PKG. Recently, Chow [10] proposed a new system architecture to realize "anonymous key issuing", by employing non-colluding identity-certifying authority (ICA) and PKG. However, these two systems only authenticate the user identity. If we modify the LRA or ICA to authenticate user public key as well, it can be used as a joining proof.

3.3 Correctness

Let $sk_{\mathsf{ID}} \leftarrow \mathsf{Extract}(\mathsf{param}, upk, \mathsf{ID})$ and $(usk, upk) \leftarrow \mathsf{UserKeyGen}(\mathsf{param})$. Then We define the *verification correctness* as follows:

$$\mathsf{Verify}(\mathsf{param}, mpk, \mathsf{ID}, m, \mathsf{Sign}(\mathsf{param}, usk, sk_{\mathsf{ID}}, m)) = 1.$$

We also define the *blaming correctness* as follows:

$$\mathsf{Blame}(\mathsf{param}, mpk, \mathsf{ID}, upk, m, \mathsf{Sign}(\mathsf{param}, usk, sk_{\mathsf{ID}}, m)) = upk.$$

3.4 Unforgeability

The security model for unforgeability captures the attack from the outsider to forge a signature when the PKG is honest. The adversary can obtain signatures of an honest user and can get the identity-based secret key of any identity except the challenge identity. We have the following game for unforgeability:

1. The simulator $\mathcal{S}$ gives param, mpk and upk' to the adversary $\mathcal{A}$.
2. $\mathcal{A}$ is allowed to query the following oracles adaptively:
   - Key Extraction Oracle $\mathcal{KEO}(upk, \mathsf{ID})$: $\mathcal{A}$ runs the $\mathsf{Extract}_u$ protocol to query the oracle. Finally the oracle returns an identity-based secret key sk_{ID} with respect to ID and upk.
   - Signing Oracle $\mathcal{SO}(m, \mathsf{ID})$: it returns a valid signature σ for the message m and the identity ID with respect to upk'.
3. $\mathcal{A}$ returns a signature σ^* for a message m^* and an identity ID^*.

$\mathcal{A}$ wins the game if $\mathsf{Verify}(\mathsf{param}, mpk, \mathsf{ID}^*, m^*, \sigma^*) = 1$, such that there was no query that $\mathcal{SO}(m^*, \mathsf{ID}^*)$ and there was no query that $\mathcal{KEO}(\cdot, \mathsf{ID}^*)$.

Definition 1. *An escrow-free IBS scheme is (ϵ, t, q_e, q_s)-secure against unforgeability if there is no t time adversary winning the above game with probability at least ϵ with q_e and q_s queries to $\mathcal{KEO}$ and $\mathcal{SO}$ respectively.*

3.5 PKG Non-frameability

The security model for PKG non-frameability captures the attack from a malicious user having an identity-based secret key that wants to frame an honest PKG. If the attacker without any identity-based secret key wants to frame an honest PKG, he must firstly forge a valid signature. Since this scenario has been captured in the model of unforgeability, we only consider the case that a malicious user, who already obtains an identity-based secret key, wants to frame an honest PKG. We have the following game for PKG non-frameability:

1. The simulator $\mathcal{S}$ gives param and mpk to the adversary $\mathcal{A}$.
2. $\mathcal{A}$ is allowed to adaptively query the Key Extraction Oracle $\mathcal{KEO}(upk, \mathsf{ID})$: $\mathcal{A}$ runs the $\mathsf{Extract}_u$ protocol to query the oracle. Finally the oracle returns an identity-based secret key sk_{ID} with respect to ID and upk. $\mathcal{S}$ saves the transcript ρ in this query and also the user's joining proof Pf.
3. $\mathcal{A}$ returns a signature σ^* for a message m^* and an identity ID^*, such that he can blame the PKG by the Blame_u protocol with a public key upk^* and a blame request φ^*.

$\mathcal{A}$ wins the game if $\mathsf{Verify}(\mathsf{param}, mpk, \mathsf{ID}^*, m^*, \sigma^*) = 1$, $\mathsf{Blame}_j(\mathsf{param}, mpk, \mathsf{ID}^*, upk^*, m^*, \sigma^*, \varphi^*) =$ PKG, and there was a query in the form of $\mathcal{KEO}(\cdot, \mathsf{ID}^*)$.

Definition 2. *An escrow-free IBS scheme is (ϵ, t, q_e)-secure against PKG non-frameability if there is no t time adversary winning the above game with probability at least ϵ with q_e queries to $\mathcal{KEO}$.*

3.6 User Non-frameability

The security model for user non-frameability captures the attack from a malicious PKG that wants to frame an honest user. We have the following game for user non-frameability:

1. The simulator $\mathcal{S}$ gives param to the adversary $\mathcal{A}$. $\mathcal{A}$ gives a master public key mpk to $\mathcal{S}$. $\mathcal{S}$ gives a user public key upk^* and a joining proof Pf^* to $\mathcal{A}$.
2. $\mathcal{A}$ is allowed to query the following oracles adaptively:
   – User Join Oracle $\mathcal{JO}(\mathsf{ID})$: it acts as the $\mathsf{Extract}_u$ protocol with input (upk^*, Pf^*) and interacts with $\mathcal{A}$ (running $\mathsf{Extract}_p$) for the identity ID. Finally the oracle obtains a identity-based secret key sk_{ID} and $\mathcal{A}$ obtains a transcript ρ.
   – Signing Oracle $\mathcal{SO}(m, \mathsf{ID})$: it returns a valid signature σ for the message m with respect to the identity ID and the user public key upk^*.
3. $\mathcal{A}$ returns a signature σ^* for a message m^* and an identity ID^*.

$\mathcal{A}$ wins the game if $\mathsf{Verify}(\mathsf{param}, mpk, \mathsf{ID}^*, m^*, \sigma^*) = 1$ and $\mathsf{Blame}(\mathsf{param}, mpk, \mathsf{ID}^*, upk, m^*, \sigma^*) = upk$ for all upk. The latter equation is always satisfied by $\mathcal{A}$ running Blame_p and giving (upk^*, Pf^*, ρ^*) to the judge (where ρ^* is the output of $\mathcal{JO}(\mathsf{ID}^*)$). We require that there was no query that $\mathcal{SO}(m^*, \mathsf{ID}^*)$.

Definition 3. *An escrow-free IBS scheme is (ϵ, t, q_j, q_s)-secure against user non-frameability if there is no t time adversary winning the above game with probability at least ϵ with q_j and q_s queries to $\mathcal{JO}$ and $\mathcal{SO}$ respectively.*

4 Generic Construction

We present a generic construction of escrow-free IBS from standard signatures. This is similar to the construction of certificate-based IBS in [4].

4.1 Our Scheme

Suppose there is a standard digital signature scheme $\mathcal{SS} = (\mathsf{SKg}, \mathsf{Sign}, \mathsf{Vf})$ which is unforgeable against chosen message attack (UF-CMA), we construct our escrow-free IBS scheme as follows:

Setup: On input the security parameter 1^k, it outputs $(mpk, msk) \leftarrow \mathsf{SKg}(1^k)$. The system parameter param is just the security parameter 1^k.

$\mathsf{UserKeyGen}$: On input param, the user obtains $(upk, usk) \leftarrow \mathsf{SKg}(1^k)$.

$\mathsf{Extract}$: The PKG algorithm $\mathsf{Extract}_p$ has input $(\mathsf{param}, upk, \mathsf{ID}, msk)$. The user algorithm $\mathsf{Extract}_u$ has input $(\mathsf{param}, upk, \mathsf{ID}, usk)$. The user computes $s \leftarrow \mathsf{Sign}_{usk}(\mathsf{ID})$ and sends $(s, \mathsf{ID}, upk, Pf)$ to the PKG. The PKG checks if $1 \leftarrow \mathsf{Vf}_{upk}(\mathsf{ID}, s)$ and Pf is a joining proof. If they are correct, then the PKG computes the identity-based secret key $sk_{\mathsf{ID}} \leftarrow \mathsf{Sign}_{msk}(\mathsf{ID}||upk)$. The PKG saves the join transcript $\rho = (s, \mathsf{ID}, upk, Pf)$ and then sends sk_{ID} to the user.

Sign: On input param, usk, sk_{ID} and a message m, the user computes $\sigma_1 \leftarrow \mathsf{Sign}_{usk}(m||\mathsf{ID})$. The user outputs the signature $\sigma = (\sigma_1, upk, sk_{\mathsf{ID}})$.

Verify: On input param, mpk, ID, m and $\sigma = (\sigma_1, upk, sk_{\mathsf{ID}})$, it returns 1 if $1 \leftarrow \mathsf{Vf}_{upk}(m||\mathsf{ID}, \sigma_1)$ and $1 \leftarrow \mathsf{Vf}_{mpk}(\mathsf{ID}||upk, sk_{\mathsf{ID}})$.

Blame: On common input param, mpk, ID, upk, m and $\sigma = (\sigma_1, upk, sk_{\mathsf{ID}})$, the user asks the judge to blame the PKG. The judge asks the PKG to provide a transcript $\rho = (s, \mathsf{ID}, upk, Pf)$. If $1 \leftarrow \mathsf{Vf}_{upk}(\mathsf{ID}, s)$ and Pf is a valid joining proof, the judge outputs upk. Otherwise, the judge outputs PKG.

Remarks. Although the user public key is part of the signature, the scheme is still considered as IBS. Similar approach is proposed by Shamir [18] and discussed in [4, 11].

4.2 Security Proofs

The correctness of the scheme is straightforward. We state the security of the above construction in the following theorems.

Theorem 1. *The scheme is unforgeable if $\mathcal{SS}$ is a UF-CMA secure signature scheme.*

Proof. Assume there is a (ϵ, t, q_e, q_s)-adversary $\mathcal{A}$. We will construct another PPT $\mathcal{B}$ that uses $\mathcal{A}$ to forge a signature of $\mathcal{SS}$ with probability at least ϵ and in time at most t.

Setup. $\mathcal{B}$ runs the $\mathcal{SS}$ simulator twice and obtains two public keys pk_1 and pk_2. $\mathcal{B}$ gives $\mathcal{A}$ the master public key $mpk = pk_1$ and the honest user public key $upk' = pk_2$.

Oracles Simulation. $\mathcal{B}$ simulates the oracles as follow:

(*Key Extraction oracle.*) On input $(upk, \mathsf{ID}, s, Pf)$ from the $\mathsf{Extract}_u$ protocol, $\mathcal{B}$ first check if Pf is a valid joining proof for upk and $1 \leftarrow \mathsf{Vf}_{upk}(\mathsf{ID},\ s)$. If they are correct, $\mathcal{B}$ queries the signing oracle of $\mathcal{SS}$ for pk_1 with input $(\mathsf{ID}||upk)$. $\mathcal{B}$ forwards the result to $\mathcal{A}$.

(*Signing oracle.*) On input (m, ID), $\mathcal{B}$ queries the signing oracle of $\mathcal{SS}$ for pk_1 with input $(\mathsf{ID}||pk_2)$ and obtains sk. $\mathcal{B}$ queries the signing oracle of $\mathcal{SS}$ for pk_2 with input $m||\mathsf{ID}$ and obtains σ_1. $\mathcal{B}$ returns (σ_1, pk_2, sk).

Output. Finally $\mathcal{A}$ outputs a signature $\sigma^* = (\sigma_1^*, upk^*, sk^*)$ for a message m^* and an identity ID^*.

- If $upk^* \neq pk_2$, then $\mathcal{B}$ returns sk^* to the $\mathcal{SS}$ simulator. It is the forgery for the message $\mathsf{ID}^*||upk^*$ with respect to the public key pk_1.
- If $upk^* = pk_2$, then $\mathcal{B}$ returns σ_1^* to the $\mathcal{SS}$ simulator. It is the forgery for the message $m^*||\mathsf{ID}^*$ with respect to the public key pk_2. □

Theorem 2. *The scheme is PKG non-frameable if $\mathcal{SS}$ is a UF-CMA secure signature scheme.*

Proof. Assume there is a (ϵ, t, q_s)-adversary $\mathcal{A}$. We will construct another PPT $\mathcal{B}$ that uses $\mathcal{A}$ to forge a signature of $\mathcal{SS}$ with probability at least ϵ and in time at most t.

Setup. $\mathcal{B}$ runs the $\mathcal{SS}$ simulator and obtains a public key pk. $\mathcal{B}$ gives $\mathcal{A}$ the master public key $mpk = pk$.

Oracles Simulation. The simulation of the *key extraction oracle* is the same as that of theorem 1.

Output. Finally $\mathcal{A}$ outputs a signature $\sigma^* = (\sigma_1^*, upk^*, sk^*)$ for a message m^* and an identity ID^*. $\mathcal{A}$ blames the PKG with a public key upk^*.

- If $(upk, \mathsf{ID}, \cdot, \cdot)$ was not successfully queried in the key extraction oracle, $\mathcal{B}$ returns sk^* as the forgery for the message $\mathsf{ID}^*||upk^*$ with respect to the public key mpk.
- Otherwise, $\mathcal{B}$ tries to reply to the judge with the transcript $\rho = (s', \mathsf{ID}^*, upk^*, Pf)$ with respect to the blame from $\mathcal{A}$. $\mathcal{A}$ wins the game if either Pf is not a valid joining proof or s' is not a valid signature. However it is not possible since the transcript is checked during the oracle query. □

Theorem 3. *The scheme is user non-frameable if $\mathcal{SS}$ is a UF-CMA secure signature scheme.*

Proof. Assume there is a (ϵ, t, q_j, q_s)-adversary $\mathcal{A}$. We will construct another PPT $\mathcal{B}$ that uses $\mathcal{A}$ to forge a signature of $\mathcal{SS}$ with probability at least ϵ and in time at most t.

Setup. $\mathcal{B}$ gives $\mathsf{param} = 1^k$ to $\mathcal{A}$. $\mathcal{A}$ gives the master public key mpk and the target identity ID^* to $\mathcal{B}$. $\mathcal{B}$ runs the $\mathcal{SS}$ simulator with 1^k and obtains a public key pk. $\mathcal{B}$ obtains a joining proof Pf^* for pk from an honest CA. $\mathcal{B}$ gives $\mathcal{A}$ the user public key $upk^* = pk$ and Pf^*.

Oracles Simulation. $\mathcal{B}$ simulates the oracles as follow:

(*Join oracle.*) On input ID, $\mathcal{B}$ queries the signing oracle of $\mathcal{SS}$ with input (ID) to obtain s. $\mathcal{B}$ sends $\rho = (s, \mathsf{ID}, upk^*, Pf^*)$ to $\mathcal{A}$. $\mathcal{A}$ stores the transcript ρ. $\mathcal{A}$ finally replies $\mathcal{B}$ with sk_{ID}.

(*Signing oracle.*) On input (m, ID), $\mathcal{B}$ first runs as the join oracle with input ID. Finally $\mathcal{B}$ obtains sk_{ID}. Then $\mathcal{B}$ queries the signing oracle of $\mathcal{SS}$ with input $m||\mathsf{ID}$ and obtains σ_1. $\mathcal{B}$ returns $(\sigma_1, upk^*, sk_{\mathsf{ID}})$.

Output. Finally $\mathcal{A}$ outputs a signature $\sigma^* = (\sigma_1^*, upk^*, sk^*)$ for a message m^* and an identity ID^*. $\mathcal{A}$ blames the user with a public key upk^* and a transcript ρ^*. $\mathcal{B}$ returns σ_1^* as the forgery of the $\mathcal{SS}$ signature for the message $(m^*||\mathsf{ID}^*)$. □

5 User Public Key Anonymity

In the previous section, we propose a generic construction of escrow-free IBS. However, the user public key is included in the ciphertext. Therefore it is similar to the certificate-based signatures to some extent. In some applications, it may not be desirable to let the verifier knowing the user public key (not the identity only). For example, assume a student has a long-term user public key. He may apply for an identity-based secret key for his student ID from the university. He may also apply for an identity-based secret key for his email address from the internet service provider. When a user uses the escrow-free IBS, he may not want the signatures for two different identities to be linked to the same user public key.

In order to construct an escrow-free IBS scheme which is *fully identity-based*, we require that the ciphertext contains no information about the user public key. We call this additional property as "user public key anonymity" [2]. In this section, we define the additional security model for the user public key anonymity.

5.1 Security Model for Anonymity

The security model for user public key anonymity captures the attack that wants to distinguish if a signature is signed by an honest user with a user public key upk. The attacker is given the master secret key, but cannot query any join oracle. In order words, the attacker can retrieve the master secret key from the real PKG, but not the join transcript from the real PKG. The users joining the real PKG will have anonymity even if the master secret key is stolen. We have the following game for anonymity:

1. The simulator $\mathcal{S}$ gives param, a master public key mpk, a master secret key msk, two user public keys upk_0, upk_1 and two corresponding certificates $cert_0$, $cert_1$ to the adversary $\mathcal{A}$.
2. $\mathcal{A}$ is allowed to query the oracle adaptively: Signing Oracle $\mathcal{SO}(m, \mathsf{ID}, b)$: it returns a valid signature σ for the message m and the identity ID with respect to upk_b.
3. $\mathcal{A}$ sends a message m^* and an identity ID^* to $\mathcal{B}$. $\mathcal{B}$ picks a random bit b' and computes $\sigma^* \leftarrow \mathsf{Sign}(\mathsf{param}, usk_{b'}, sk_{\mathsf{ID}^*}, m)$, where sk_{ID^*} is the identity-based secret key computed using $(msk, upk_{b'}, \mathsf{ID}^*)$ and $usk_{b'}$ is the user secret key for $upk_{b'}$. $\mathcal{B}$ sends σ^* to $\mathcal{A}$.
4. $\mathcal{A}$ is allowed to query the above oracles adaptively.
5. $\mathcal{A}$ returns a bit b^*.

$\mathcal{A}$ wins the game if $b' = b^*$. We require that there was no query that $\mathcal{SO}(m^*, \mathsf{ID}^*, \cdot)$. The advantage of $\mathcal{A}$ is the probability of $\mathcal{A}$ winning the above game over 1/2.

Definition 4. *An identity-based signature scheme is (ϵ, t, q_s)-secure against anonymity if there is no t time adversary winning the above game with probability at least ϵ with q_s queries to $\mathcal{SO}$.*

Remark. The security model for *key-privacy* or *anonymity* in traditional public key encryption was proposed by Bellare *et al.* [3]. In this section, we follow their notion of "indistinguishability of keys under chosen-ciphertext attacks" and adopt the indistinguishability game into our IBS setting.

The main difference between Bellare *et al.*'s model and our model is that the challenge user secret keys and the user public keys are not chosen by the adversary in our model. It is

[2] An escrow-free IBS scheme can either has the "user public key anonymity" property or not.

because our Blame algorithm requires that the PKG is able to show that "the upk is related to the signature σ" if σ is signed by the corresponding usk. If both the msk, usk_0 and usk_1 are known to the adversary, he can generate the join transcript by himself and checks if upk_0 or upk_1 is related to the challenge signature σ^*. It will break the anonymity. Therefore in our anonymity model, the adversary is not given usk_0 and usk_1. The adversary is given the signing oracle for usk_0 and usk_1 instead.

6 Construction with User Public Key Anonymity

In this section, we provide a concrete construction with user public key anonymity. Our construction for escrow-free IBS is based on the signature schemes from Boneh and Boyen [6] and Boneh *et al.* [8]. We also use the "signatures of knowledge" (SoK) notion from Chase and Lysyanskaya [9].

6.1 Intuition

We use the signature scheme from Boneh and Boyen [6] as the identity-based secret key. Suppose the master secret key is α and the master public key is g^α. For a user with secret key x and public key $y = g^x$, his identity-based secret key is A where

$$A^{\alpha + \mathsf{ID}} v^x = u,$$

and g, u, v are a generator of $\mathbb{G}$.

For the signing protocol, the part of the signature useful for the blame protocol is derived from Boneh *et al.* [8]. Denote this part as S and we have

$$S = \hat{e}(v, H_2(m))^x,$$

where m is the message. The the signing protocol becomes:

$$SoK\{(A, x) : A^{\alpha + \mathsf{ID}} v^x = u \quad \wedge \quad S = \hat{e}(v, H_2(m))^x\}(m).$$

6.2 Our Scheme

We give the detailed construction of the escrow-free IBS with anonymity.

Setup: The algorithm first chooses a random prime p of bit size $\Theta(k)$. Let $\mathbb{G}$, $\mathbb{G}_T$ be a bilinear group of order p and a pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. It also chooses generators $g, u, v \in \mathbb{G}$. It picks collision resistant hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$ for hashing the identity string, and $H_2 : \{0,1\}^* \to \mathbb{G}$ for hashing the message. It also chooses generators $g_0, g_1, g_2 \in \mathbb{G}$ used for the signature of knowledge. The system parameter param is ($\hat{e}$, $\mathbb{G}$, $\mathbb{G}_T$, p, g, u, v, g_0, g_1, g_2, H_1, H_2).

The PKG randomly selects his master secret key $\alpha \in \mathbb{Z}_p^*$. He computes the master public key $g_a = g^\alpha$.

UserKeyGen: The user randomly selects his user secret key $x \in \mathbb{Z}_p^*$. He computes $y = g^x$ as his user public key.

Extract: The user calculates $v' = v^x$. He also computes a non-interactive zero-knowledge (NIZK) proof [3] Σ of x with respect to v' and v (We omit the details of the NIZK proof for discrete logarithm for simplicity). He sends v', ID, y, a joining proof Pf and the NIZK proof Σ to the PKG. The PKG checks the validity of Pf, Σ. If so, the PKG computes:

$$A = (uv'^{-1})^{\frac{1}{\alpha+i}},$$

where $i = H_1(\mathsf{ID})$ and returns A to the user. The PKG stores the transcript $\rho = (v', \Sigma, \mathsf{ID}, y, Pf)$.

Sign: The user signs a message m with the user secret key x and the identity-based secret key A. He computes the signature of knowledge (SoK):

$$SoK\{(A,x) : A^{\alpha+i}v^x = u \quad \wedge \quad S = \hat{e}(v, H_2(m))^x\}(m)$$

The SoK is specified as follows. The user randomly chooses $s, r, r_2 \in \mathbb{Z}_p^*$, $R_1 \in \mathbb{G}$ and computes:

$$t_0 = g_0^s, \qquad t_1 = Ag_1^s, \qquad t_2 = v^x g_2^s, \qquad \tau_0 = g_0^r, \qquad \tau_1 = R_1 g_1^r,$$
$$\tau_2 = v^{r_2} g_2^r, \qquad \tau_3 = [\hat{e}(g_1, g_a g^i) \cdot \hat{e}(g_2, g)]^r, \qquad \tau_4 = \hat{e}(g_2, H_2(m))^r.$$

The user computes $c = H_3(t_0, t_1, t_2, \tau_0, \ldots, \tau_4, m, mpk, \mathsf{ID})$ and:

$$z_0 = r - cs, \qquad Z_1 = R_1 A^{-c}, \qquad z_2 = r_2 - cx.$$

The signature is $\sigma = (t_0, t_1, t_2, c, z_0, Z_1, z_2, S)$.

Verify: Upon input a signature σ for a message m and an identity ID, it computes:

$$i = H_1(\mathsf{ID}), \quad t_3 = \hat{e}(t_1, g_a g^i) \cdot \hat{e}(t_2, g) \cdot \hat{e}(u, g)^{-1}, \quad t_4 = \hat{e}(t_2, H_2(m)) \cdot S^{-1},$$
$$\tau_0 = g_0^{z_0} t_0^c, \quad \tau_1 = Z_1 g_1^{z_0} t_1^c, \quad \tau_2 = v^{z_2} g_2^{z_0} t_2^c,$$
$$\tau_3 = [\hat{e}(g_1, g_a g^i) \cdot \hat{e}(g_2, g)]^{z_0} \cdot t_3^c, \quad \tau_4 = \hat{e}(g_2, H_2(m))^{z_0} \cdot t_4^c.$$

It outputs 1 if $c = H_3(t_0, t_1, t_2, \tau_0, \ldots, \tau_4, m, mpk, \mathsf{ID})$. Otherwise, it outputs 0.

Blame: On common input the master public key mpk, an identity ID, a message m, a signature σ, a user public key y, the user with user secret key x first computes $\varphi = v^x$. The user sends φ to the judge as the blame request.

The judge checks if $\sigma = (t_0, t_1, t_2, c, z_0, Z_1, z_2, S)$ is a valid signature and:

$$\hat{e}(v, y) = \hat{e}(\varphi, g) \qquad \wedge \qquad \hat{e}(\varphi, H_2(m)) \neq S.$$

If they are not equal, the judge returns "*upk*".

Otherwise, the judge requests the PKG to provide a transcript $\rho = (v', \Sigma, \mathsf{ID}, y', Pf')$. If Pf' is a valid joining proof for y' and

$$\hat{e}(v, y') = \hat{e}(v', g) \qquad \wedge \qquad \hat{e}(v', H_2(m)) = S.$$

If they are equal, the judge returns "*upk*". Otherwise, the judge returns "PKG".

[3] Although v' can be used to prove the knowledge of x via pairing, we need the extractor of the NIZK proof to obtain x in the security proof.

6.3 Security Proofs

The correctness of the signature scheme is straightforward.

We first prove that the SoK protocol above is a secure signature of knowledge. We use the game-based definition (SimExt-secure) in [9]. Chase and Lysyanskaya [9] proved the equivalence of the game-based definition and the UC framework definition.

Lemma 1. *The SoK protocol above is a SimExt-secure signature of knowledge of a witness* (A, x).

Proof. The *correctness* of the signature of knowledge scheme is straightforward.

For *simulatability*, after the simulator randomly picks A and x to compute t_0, t_1, t_2, S, he picks a challenge $c \in \mathbb{Z}_p^*$ and also $z_0, z_2 \in \mathbb{Z}_p^*$, $Z_1 \in \mathbb{G}$. Then compute:

$$\begin{aligned}
&t_3 = \hat{e}(t_1, g_a g^i) \cdot \hat{e}(t_2, g) \cdot \hat{e}(u, g)^{-1}, \qquad t_4 = \hat{e}(t_2, H_2(m)) \cdot S^{-1}, \\
&\tau_0 = g_0^{z_0} t_0^c, \qquad \tau_1 = Z_1 g_1^{z_0} t_1^c, \qquad \tau_2 = v^{z_2} g_2^{z_0} t_2^c, \\
&\tau_3 = [\hat{e}(g_1, g_a g^i) \cdot \hat{e}(g_2, g)]^{z_0} \cdot t_3^c, \qquad \tau_4 = \hat{e}(g_2, H_2(m))^{z_0} \cdot t_4^c.
\end{aligned}$$

Therefore the transcript above is simulatable.

For *extraction*, we now show that there exists an extractor for A and x. Assume there are two transcripts with the same commitment $(t_0, t_1, t_2, \tau_0, \ldots, \tau_4)$ and different challenges c, c' and responses (z_0, Z_1, z_2), (z_0', Z_1', z_2'). Let:

$$\tilde{s} = \frac{z_0' - z_0}{c - c'}, \qquad \tilde{A} = (Z_1'/Z_1)^{\frac{1}{c-c'}}, \qquad \tilde{x} = \frac{z_2' - z_2}{c - c'}.$$

Then, we have

$$t_0 = g_0^{\tilde{s}}, \qquad t_1 = \tilde{A} g_1^{\tilde{s}}, \qquad t_2 = v^{\tilde{x}} g_2^{\tilde{s}}.$$

From τ_3 and τ_4, we have the following relations:

$$\begin{aligned}
\hat{e}(g_1^{z_0} t_1^c, g_a g^i) \cdot \hat{e}(g_2^{z_0} t_2^c, g) \cdot \hat{e}(u, g)^{-c} &= \hat{e}(g_1^{z_0'} t_1^{c'}, g_a g^i) \cdot \hat{e}(g_2^{z_0'} t_2^{c'}, g) \cdot \hat{e}(u, g)^{-c'} \\
\hat{e}(g_1^{-\tilde{s}} t_1, g_a g^i) \cdot \hat{e}(g_2^{-\tilde{s}} t_2, g) &= \hat{e}(u, g) \\
\hat{e}(\tilde{A}, g_a g^i) \cdot \hat{e}(v^{\tilde{x}}, g) &= \hat{e}(u, g),
\end{aligned}$$

$$\begin{aligned}
\hat{e}(g_2, H_2(m))^{z_0} \cdot [\hat{e}(t_2, H_2(m)) \cdot S^{-1}]^c &= \hat{e}(g_2, H_2(m))^{z_0'} \cdot [\hat{e}(t_2, H_2(m)) \cdot S^{-1}]^{c'} \\
\hat{e}(g_2, H_2(m))^{\tilde{s}} &= \hat{e}(t_2, H_2(m)) \cdot S^{-1} \\
S &= \hat{e}(v^{\tilde{x}}, H_2(m)).
\end{aligned}$$

Therefore we extract $(\tilde{A}, \tilde{x})$ that satisfy the required relations. □

Theorem 4. *The scheme is* (ϵ, t, q_e, q_s)*-unforgeable if the* (ϵ', t', q)*-SDH assumption holds in* $\mathbb{G}$ *in the random oracle model, with:*

$$t \le t' + \Theta((q_e + q_s)\delta + q_s\tau), \qquad q = q_e + 1, \qquad \epsilon' \ge (\frac{\epsilon}{C_{q_e}^{q_h}} - \frac{1}{p})^2$$

where q_h *is the number of query to the* H_1 *oracle,* δ *and* τ *are the time for computing exponentiation in* $\mathbb{G}$ *and pairing respectively.*

Proof. Assume there is a (ϵ, t, q_e, q_s)-adversary $\mathcal{A}$. We will construct another PPT $\mathcal{B}$ that makes use of $\mathcal{A}$ to solve the q-SDH problem in $\mathbb{G}$ with probability at least ϵ' and in time at most t'. $\mathcal{B}$ is given a q-SDH problem instance $(\bar{g}, \bar{g}^a, \ldots, \bar{g}^{a^q})$.

Setup. $\mathcal{B}$ randomly selects $id_1, \ldots, id_{q-1}$. Let $f(a) = \prod_{k=1}^{q-1}(a + id_k)$. $\mathcal{B}$ computes $g = \bar{g}^{f(a)}$ and $g_a = \bar{g}^{af(a)}$ using the q-SDH problem instance. $\mathcal{B}$ randomly picks $\mu, \nu \in \mathbb{Z}_p^*$ and computes $u = g^\mu$, $v = g^\nu$. $\mathcal{B}$ honestly generates the rest of the system parameter.

Oracles Simulation. $\mathcal{B}$ simulates the oracles as follow:
(*H oracle.*) To respond to the queries $\mathcal{B}$ maintains a list of tuples called the H_i^{list} for $i = 1, 2, 3$. Initially the lists are all empty. $\mathcal{B}$ first selects a random $j \in \{1, \ldots, q_{h_1} + q_e\}$. Let ctr be the current size of the H_1^{list}. If the H_i query already appears on the H_i^{list}, then $\mathcal{B}$ responses with the same answer. If the query is new, then:

- $H_1(\mathsf{ID}_i)$: If $ctr = j - 1$, $\mathcal{B}$ picks a random $d_i \in \mathbb{Z}_p$. If $ctr \neq j - 1$, $\mathcal{B}$ picks a new id_k not used before and sets $d_i = id_k$. Then $\mathcal{B}$ adds the tuple $\langle ctr + 1, \mathsf{ID}_i, d_i \rangle$ to the H_1^{list} and responds to $\mathcal{A}$ with $H_1(\mathsf{ID}_i) = d_i$.
- $H_2(m_i)$: $\mathcal{B}$ just picks a random string S_i in the corresponding domain and adds the tuple $\langle m_i, S_i \rangle$ to the H_2^{list}. It responds to $\mathcal{A}$ with $H_2(m_i) = S_i$.
- $H_3(T_i)$: $\mathcal{B}$ just picks a random $c_i \in \mathbb{Z}_p$ and adds the tuple $\langle T_i, c_i \rangle$ to the H_3^{list}, where $T_i = (t_0, t_1, t_2, \tau_0, \ldots, \tau_4, m, mpk, \mathsf{ID})$. It responds to $\mathcal{A}$ with $H_3(T_i) = c_i$.

(*Key Extraction oracle.*) $\mathcal{A}$ sends $(v', \mathsf{ID}_i, y_i, Pf, \Sigma)$ according to the Extract protocol. If Pf or Σ is not valid, $\mathcal{B}$ returns $\perp$. Otherwise, $\mathcal{B}$ use the extractor of the NIZK protocol to obtain $x_i = \log_g y_i$. Then $\mathcal{B}$ first looks through list H_1^{list}.

- If ID_i is not on the list, then $\mathcal{B}$ queries $H_1(\mathsf{ID}_i)$ and obtains d_i.
- Else $\mathcal{B}$ looks for $\langle \cdot, \mathsf{ID}_i, d_i \rangle \in H_1^{list}$ and returns d_i.

If $d_i \neq id_k$ for all $k \in \{1, \ldots, q-1\}$, $\mathcal{B}$ declares failure and exits. Otherwise, $\mathcal{B}$ computes

$$A = (uv')^{\frac{1}{a+id_k}} = \bar{g}^{\frac{f(a)(\mu - \nu x_i)}{a+id_k}}.$$

$\mathcal{B}$ returns the identity-based secret key A to the adversary.

(*Signing oracle.*) $\mathcal{B}$ runs the simulator of the signature of knowledge as in lemma 1 to obtain a valid signature σ. $\mathcal{B}$ patches the corresponding value c to the oracle H_2. $\mathcal{B}$ returns σ to $\mathcal{A}$.

Output. Finally $\mathcal{A}$ outputs a signature σ for ID^*. $\mathcal{B}$ looks for $\langle i, \mathsf{ID}^*, d^* \rangle \in H_1^{list}$. If $i \neq j$, $\mathcal{B}$ declares failure and exits. Otherwise, $\mathcal{B}$ rewinds and obtains another signature σ'. Similar to lemma 1, $\mathcal{B}$ can extract $\tilde{A}, \tilde{x}$. Since ID^* is not queried to the key extraction oracle,

$$\tilde{A} = g^{\frac{\mu - \nu\tilde{x}}{a+d^*}} = \bar{g}^{\frac{f(a)(\mu - \nu\tilde{x})}{a+d^*}} = \bar{g}^{\sum_{k=0}^{q-2} C_k a^k + \frac{C_{-1}}{a+d^*}}$$

where $C_{-1}, C_0, \ldots, C_{1-2}$ can be computed. Notice that $C_{-1} \neq 0$ if $d^* \neq id_k$ for all $k \in \{1, \ldots, q-1\}$. Then $\mathcal{B}$ computes:

$$A^* = (\tilde{A}\bar{g}^{\sum_{k=0}^{q-2} -C_k a^k})^{1/C_{-1}},$$

and returned (A^*, d^*) as the solution to the q-SDH problem.

Probability and Time Analysis. We consider the two events that $\mathcal{B}$ could fail:

- Event E_i: For the i-th query to the $\mathcal{KEO}$, $d_i \neq id_k$ for all $k \in \{1, \ldots, q-1\}$.
- Event E^*: $d^* = id_k$ for some $k \in \{1, \ldots, q-1\}$.

Notice that the event $\neg\mathsf{E}^*$ implies $\cup_{i=1}^{q_e} \neg\mathsf{E}_i$, since the model requires that the challenge identity has never been submitted to $\mathcal{KEO}$. Therefore the probability of $\mathcal{B}$ fails is at most $\Pr[\mathsf{E}^*] = 1/(q_{h_1} + q_e)$. By the reset lemma [5], the probability that $\mathcal{B}$ solves the q-SDH problem is $\epsilon' \geq (\epsilon/(q_{h_1} + q_e) - 1/p)^2$.

For each key extraction oracle query, $\mathcal{B}$ runs $O(1)$ exponentiation in $\mathbb{G}$. For each signing oracle query, $\mathcal{B}$ runs $O(1)$ exponentiation in $\mathbb{G}$ and $O(1)$ pairing computation. □

Theorem 5. *The scheme is (ϵ, t, q_e)-PKG non-frameable if the (ϵ', t', q)-SDH assumption holds in $\mathbb{G}$ in the random oracle model, with:*

$$t \leq t' + \Theta(q_e \delta), \qquad q = q_e + 1, \qquad \epsilon' \geq (\frac{\epsilon}{C_{q_e}^{q_h}} - \frac{1}{p})^2$$

where q_h is the number of query to the H_1 oracle, δ is the time for computing exponentiation in $\mathbb{G}$, respectively.

Proof. Assume there is a (ϵ, t, q_e)-adversary $\mathcal{A}$. We will construct another PPT $\mathcal{B}$ that makes use of $\mathcal{A}$ to solve the q-SDH problem in $\mathbb{G}$ with probability at least ϵ' and in time at most t'. $\mathcal{B}$ is given a q-SDH problem instance $(\bar{g}, \bar{g}^a, \ldots, \bar{g}^{a^q})$.

Setup. $\mathcal{B}$ randomly selects $i_1, \ldots, i_{q-1}$. Let $f(a) = \prod_{k=1}^{q-1}(a + i_k)$. $\mathcal{B}$ computes $g = \bar{g}^{f(a)}$ and $g_a = \bar{g}^{af(a)}$ using the q-SDH problem instance. $\mathcal{B}$ randomly picks $\mu \in \mathbb{Z}_p^*$ and computes $u = g^{\mu}$. $\mathcal{B}$ honestly generates the rest of the system parameter param.

Oracles Simulation. $\mathcal{B}$ simulates the H oracle and the key extraction oracle as in theorem 4.

Output. Finally $\mathcal{A}$ outputs a signature σ for a message m^* and an identity ID^*. If $\mathcal{A}$ wins without querying $\mathcal{KEO}(\cdot, \mathsf{ID}^*)$, he can also win as in the unforgeability game. Then $\mathcal{B}$ can calculate the solution to the q-SDH problem as in the proof of the theorem 4. We now focus on the case that $\mathcal{A}$ has queried $\mathcal{KEO}(\cdot, \mathsf{ID}^*)$.

Denote $i^* = H_1(\mathsf{ID}^*)$. $\mathcal{A}$ also returns $upk^* = g^{x^*}$ and $\varphi^* = v^{x^*}$ to blame the PKG. $\mathcal{B}$ rewinds and obtains another signature σ'. Similar to lemma 1, $\mathcal{B}$ can extract A_σ, x_σ. Denote $upk_\sigma = g^{x_\sigma}$. Let $\mathcal{A}$ has queried $\mathcal{KEO}(upk_e, \mathsf{ID}^*)$ before. We consider the following cases:

- $upk^* = upk_\sigma$. As the claim protocol outputs "PKG", it means that $\hat{e}(v^{x^*}, H_2(m^*)) \neq S$. However by lemma 1, we have $S = \hat{e}(v, H_2(m^*))^{x_\sigma}$ which is a contradiction to $x^* = x_\sigma$.
- $upk_e = upk_\sigma \neq upk^*$. $\mathcal{B}$ retrieves the transcript $(v'_e, \mathsf{ID}^*, upk_e, Pf_e, \Sigma_e)$ and sends Σ_e (the proof of knowledge of $\log_g upk_e$ and $\log_v v'_e$) and Pf_e to the judge. $\mathcal{A}$ cannot win the game unless he can break the proof of knowledge protocol.
- $upk_e \neq upk_\sigma$. We have $A_e^{a+i^*} v^{x_e} = A_\sigma^{a+i^*} v^{x_\sigma}$. Then it means $\mathcal{A}$ wins by forging a new identity-based secret key using a different user secret key. Assume there exist an adversary $\mathcal{A}_1$ wins a game that produces a new pair (A_σ, x_σ) for the identity i^* only. In this game the adversary $\mathcal{A}_1$ firstly sends the challenge identity i^* to $\mathcal{B}_1$. $\mathcal{B}_1$ sets $g = \bar{g}$, $v = g^{\nu}$, $A_e = g^{\omega}$ and $u = g^{(a+i^*)\omega + \nu x_e}$ for some random $\nu, \omega \in_R \mathbb{Z}_p^*$. $\mathcal{B}_1$ gives the master public key and the pair (A_e, x_e) to $\mathcal{A}_1$. Finally $\mathcal{A}_1$ returns a new pair (A_σ, x_σ). Then finally $\mathcal{B}$ can compute $g^{1/(a+i^*)} = (A_e/A_\sigma)^{1/\nu(x_\sigma - x_e)}$, which is the solution to the SDH problem.

The probability and the running time of the algorithms are similar to that of theorem 4. □

Theorem 6. *The scheme is (ϵ, t, q_j, q_s)-user non-frameable if the (ϵ', t')-DL assumption holds in $\mathbb{G}$ in the random oracle model, where:*

$$t \le t' + \Theta((q_j + q_s)\nu + q_s\tau), \qquad \epsilon' \ge (\epsilon - \frac{1}{p})^2$$

where ν and τ are the time for computing exponentiation in $\mathbb{G}$ and pairing respectively.

Proof. Assume there is a (ϵ, t, q_j, q_s)-adversary $\mathcal{A}$. We will construct another PPT $\mathcal{B}$ that makes use of $\mathcal{A}$ to solve the DL problem in $\mathbb{G}$ with probability at least ϵ' and in time at most t'. $\mathcal{B}$ is given a DL problem instance $(g, \bar{y})$.

Setup. $\mathcal{B}$ generates the public parameters for verifying Σ, randomly chooses generators $g_0, g_1, g_2 \in \mathbb{G}$ and computes $v = g^{\nu}$ for some random $\nu \in_R \mathbb{Z}_p$. $\mathcal{B}$ honestly generates the rest of param and sends it to the adversary $\mathcal{A}$.

$\mathcal{A}$ gives a master public key mpk to $\mathcal{B}$. $\mathcal{B}$ sets the user public key $upk^* = \bar{y}$ and generates a joining proof Pf^* for upk^*. $\mathcal{B}$ gives upk^* and Pf^* to $\mathcal{A}$.

Oracles Simulation. $\mathcal{B}$ simulates the oracles as follow:
(*H oracle.*) H_1, H_2 and H_3 are simulated as normal random oracles.

(*User Join oracle.*) Upon input ID, $\mathcal{B}$ runs the simulator of the proof of knowledge to obtains a valid Σ. $\mathcal{B}$ sends $(v' = \bar{y}^{\nu}$, ID, upk^*, Pf^*, $\Sigma)$ to $\mathcal{A}$. $\mathcal{A}$ returns sk_{ID}.

(*Signing oracle.*) $\mathcal{B}$ runs the simulator of the signature of knowledge as in lemma 1 to obtain a valid signature σ. $\mathcal{B}$ returns σ to $\mathcal{A}$. $\mathcal{B}$ fails and exits if the challenge c is already set in the H_2 query.

Output. Finally $\mathcal{A}$ outputs a signature σ. $\mathcal{B}$ rewinds and obtains another signature σ'. Similar to lemma 1, $\mathcal{B}$ can extract $\tilde{x}$ and returns it as the solution to the DL problem.

The probability and the time analysis are straightforward. □

Theorem 7. *The scheme is (ϵ, t, q_s)-anonymous if the (ϵ', t')-DBDH assumption holds in the random oracle model, with:*

$$t \le t' + \Theta(q_s(\delta + \tau)), \qquad \epsilon' \ge (\frac{\epsilon}{q_h} - \frac{1}{p})^2$$

where q_h is the number of query to the H_2 oracle, δ and τ are the time for computing exponentiation in $\mathbb{G}$ and pairing respectively.

Proof. Assume there is a (ϵ, t, q_s)-adversary $\mathcal{A}$. We will construct another PPT $\mathcal{B}$ that makes use of $\mathcal{A}$ to solve the DBDH problem with probability at least ϵ' and in time at most t'. $\mathcal{B}$ is given a DBDH problem instance (g, g^a, g^b, g^c, T).

Setup. $\mathcal{B}$ randomly selects $x_0, x_1 \in \mathbb{Z}_p$. He computes $upk_0 = g^{ax_0}$ and $upk_1 = g^{ax_1}$. $\mathcal{B}$ sets $v = g^b$. $\mathcal{B}$ honestly generates the rest of param.

Oracles Simulation. $\mathcal{B}$ simulates the oracles as follow:

(*H oracle.*) To respond to the queries $\mathcal{B}$ maintains a list of tuples called the H_i^{list} for $i = 1, 2, 3$. Initially the lists are all empty. $\mathcal{B}$ first selects a random $j \in \{1, \dots, q_{H_2} + q_s\}$. Let ctr be the current size of the H_2^{list}. If the H_i query already appears on the H_i^{list}, then $\mathcal{B}$ responses with the same answer. If the query is new, then:

- $H_1(\mathsf{ID}_i)$: $\mathcal{B}$ just picks a random string id_i in the corresponding domain and adds the tuple $\langle \mathsf{ID}_i, id_i \rangle$ to the H_1^{list}. It responds to $\mathcal{A}$ with $H_1(\mathsf{ID}_i) = id_i$.
- $H_2(m_i)$: If $ctr \neq j - 1$, $\mathcal{B}$ picks a random $S_i \in \mathbb{G}$. If $ctr = j - 1$, $\mathcal{B}$ sets $S_i = g^c$. Then $\mathcal{B}$ adds the tuple $\langle ctr + 1, m_i, S_i \rangle$ to the H_2^{list} and responds to $\mathcal{A}$ with $H_2(m_i) = S_i$.
- $H_3(T_i)$: $\mathcal{B}$ just picks a random $c_i \in \mathbb{Z}_p$ and adds the tuple $\langle T_i, c_i \rangle$ to the H_3^{list}, where $T_i = (t_0, t_1, t_2, \tau_0, \dots, \tau_4, m, mpk, \mathsf{ID})$. It responds to $\mathcal{A}$ with $H_3(T_i) = c_i$.

(*Signing oracle.*) $\mathcal{B}$ runs the simulator of the signature of knowledge as in lemma 1 to obtain a valid signature σ. In the process, $\mathcal{B}$ has to query $H_2(m_i)$ by himself. Finally $\mathcal{B}$ patches the corresponding value c to the oracle H_3. $\mathcal{B}$ fails and exits if c is already set. $\mathcal{B}$ returns σ to $\mathcal{A}$.

Challenge. $\mathcal{A}$ sends a message m^* and an identity ID^*. If $H_2(m^*) \neq g^c$, then $\mathcal{B}$ declares failure and exits. Otherwise, $\mathcal{B}$ picks a random bit b' and calculates $S^* = T^{x_b}$. $\mathcal{B}$ runs the simulator of the signature of knowledge as in lemma 1 to obtain a valid signature σ using S^*. $\mathcal{B}$ sends σ^* to $\mathcal{A}$.

Output. Finally $\mathcal{A}$ outputs a bit b^*. If $b^* = b'$, then $\mathcal{B}$ outputs $T = \hat{e}(g, g)^{abc}$ as the answer to the DBDH problem. Otherwise, $\mathcal{B}$ outputs $T \neq \hat{e}(g, g)^{abc}$.

Probability and Time Analysis. The probability of $\mathcal{B}$ exits in the challenge phase is $1 - 1/q_{h_2} + q_s$. By the reset lemma [5], the probability is $\epsilon' \geq (\epsilon/q_{h_2} + q_s - 1/p)^2$.

For each signing oracle query, $\mathcal{B}$ runs $O(1)$ exponentiation in $\mathbb{G}$ and $O(1)$ pairing computation. $\square$

7 Comparison

In this section, we provide a comparison of our scheme against the existing schemes. Denote (s, P) as a pair of secret key and public key computed by the user. Denote (d, I) as a pair of identity-based secret key and identity computed by the PKG. Let (α, β) be a pair of secret key and public key of the PKG. Let c be the secret key of a certificate authority. Let $Sig_a(b)$ be a signature of message b using the secret key a. Let $Com_a(b)$ be a commitment of the value a using the public parameter b. We compare the public information that a verifier needs to know (except β), the secret keys used by the signer and the witness to link the identity with the public key. We use W to represent a witness which is different from the above parameters.

Notice that the certificateless signatures, the certificate-based signatures and the self-certificated signatures aim to resolve the key escrow problem. Nonetheless, these schemes are no longer identity-based since the user public key P has been introduced into the public information. On the contrary, our scheme in section 6 is *the only scheme* that solves this problem while staying at the framework of identity-based cryptography in a strict sense. However the price we have to pay is to include a joining proof involved in the extraction protocol.

On the other hand, our generic construction in section 4 provides a more efficient solution than our scheme in section 6. The signature of the escrow-free IBS in section 4 only consists

Schemes	Public Information	Secret Key	Witness
IBS [18]	I	d	-
IBS + Cert	I, P, W	s, d	$W = Sig_c(I, P)$
Certificateless Sig [1]	I, P, W	s, d	$W = Com_s(\beta)$
Certificate-based Sig [17]	I, P	s, d	$d = Sig_\alpha(I, P)$
Self-Certificated Sig [14]	I, P	s	$P = d = Sig_\alpha(I)$
Our Scheme in §4	I, P, d	s	$d = Sig_\alpha(I, P)$
Our Scheme in §6	I	s, d	$d = Sig_\alpha(I, P)$

Table 2. Comparison of our scheme against the existing schemes.

of two standard signatures and a user public key. The computational cost of signing is the same as signing one standard signature; the computational cost of verifying is the same as verifying two standard signatures. It is as efficient as the generic IBS scheme in [4].

8 Conclusion

In this paper, we introduced the concept of escrow-free identity-based signatures to solve the key escrow problem in identity-based signature. We proposed an extra *user public key anonymity* property to escrow-free identity-based signatures and proposed a concrete construction. Our construction solves the open problem of key escrow in identity-based signatures, without requiring multiple PKGs. Our scheme is the *first* to achieve level 3 of trust of the PKG in Girault's model [14], in the identity-based setting.

Acknowledgements

We thank David Galindo and the anonymous reviewers of EuroPKI 2009 for helpful comments and suggestions.

References

1. S. S. Al-Riyami and K. G. Paterson. Certificateless public key cryptography. In *ASIACRYPT 2003*, volume 2894 of *LNCS*, pages 452–473. Springer, 2003.
2. M. H. Au, Q. Huang, J. K. Liu, W. Susilo, D. S. Wong, and G. Yang. Traceable and retrievable identity-based encryption. In *ACNS 2008*, volume 5037 of *LNCS*, pages 94–110, 2008.
3. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 566–582. Springer, 2001.
4. M. Bellare, C. Namprempre, and G. Neven. Security proofs for identity-based identification and signature schemes. In *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 268–286. Springer, 2004.
5. M. Bellare and A. Palacio. GQ and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In *CRYPTO 2002*, volume 2442 of *LNCS*, pages 162–177. Springer, 2002.
6. D. Boneh and X. Boyen. Short signatures without random oracles. In *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73. Springer, 2004.
7. D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.
8. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. In *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, 2001.
9. M. Chase and A. Lysyanskaya. On signatures of knowledge. In *CRYPTO 2006*, volume 4117 of *LNCS*, pages 78–96. Springer, 2006.
10. S. S. M. Chow. Removing escrow from identity-based encryption. In *PKC 2009*, volume 5443 of *LNCS*, pages 256–276. Springer, 2009.

11. D. Galindo, J. Herranz, and E. Kiltz. On the generic construction of identity-based signatures with additional properties. In *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 178–193. Springer, 2006.
12. C. Gentry. Certificate-based encryption and the certificate revocation problem. In *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 272–293. Springer, 2003.
13. C. Gentry and A. Silverberg. Hierarchical ID-based cryptography. In *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 548–566. Springer, 2002.
14. M. Girault. Self-certified public keys. In *EUROCRYPT '91*, volume 547 of *LNCS*, pages 490–497. Springer, 1991.
15. V. Goyal. Reducing trust in the PKG in Identity Based Cryptosystems. In *CRYPTO 2007*, volume 4622 of *LNCS*, pages 430–447. Springer, 2007.
16. V. Goyal, S. Lu, A. Sahai, and B. Waters. Black-box accountable authority identity-based encryption. In *CCS 2008*, pages 427–436. ACM, 2008.
17. B. G. Kang, J. H. Park, and S. G. Hahn. A certificate-based signature scheme. In *CT-RSA 2004*, volume 2964 of *LNCS*, pages 99–111. Springer, 2004.
18. A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO '84*, volume 196 of *LNCS*, pages 47–53. Springer, 1984.
19. A. F. Sui, S. S. M. Chow, L. C. K. Hui, S.-M. Yiu, K. P. Chow, W. W. Tsang, C. F. Chong, K. K. H. Pun, and H. W. Chan. Separable and anonymous identity-based key issuing. In *ICPADS 2005*, pages 275–279. IEEE Computer Society, 2005.