*International Journal of Computer Mathematics*, 2014
http://dx.doi.org/10.1080/00207160.2014.948869





# An efficient certificateless undeniable signature scheme

Rouzbeh Behnia[a]* Swee-Huay Heng[a] and Che-Sheng Gan[b]

[a]*Faculty of Information Science and Technology, Multimedia University, Malaysia;*
[b]*Faculty of Engineering and Technology, Multimedia University, Malaysia*

(*Received 26 March 2014; revised version received 4 June 2014; accepted 18 July 2014*)

Certificateless cryptography addresses the private key escrow problem in identity-based systems, while overcoming the costly issues in traditional public key cryptography. Undeniable signature schemes were proposed with the aim of limiting the public verifiability of ordinary digital signatures. The first certificateless undeniable signature scheme was put forth by Duan. The proposed scheme can be considered as the certificateless version of the identity-based undeniable signature scheme which was introduced by Libert and Quisquater. In this paper, we propose a new scheme which is much more efficient comparing to Duan's scheme. Our scheme requires only one pairing evaluation for signature generation and provides more efficient confirmation and disavowal protocols for both the signer and the verifier. We also prove the security of our scheme in the strong security model based on the intractability of some well-known pairing-based assumptions in the random oracle model.

**Keywords:** certificateless cryptography; identity-based cryptography; undeniable signature; designated verifier; the random oracle model

*2010 AMS Subject Classifications*: 11T71; 94A60

## 1. Introduction

In conventional public key cryptography, the authenticity of users' public keys is delivered by means of signed certificates. Unfortunately, the costs of generating, verifying and managing these certificates would be excessive when such systems are employed in a large scale. With the aim of addressing these costly issues, Shamir [26] proposed the idea of identity-based cryptography. In identity-based systems, the public key of the user is calculated from her publicly known information (e.g. passport number, Media Access Control address, or email address) and her private key is computed by a fully trusted third party called the Private Key Generator. Consequently, such systems suffer from an inherited private key escrow problem.

In contemplation of bridging the gap between conventional public key cryptography and identity-based cryptography, Al-Riyami and Paterson [1] introduced the concept of certificateless cryptography. In certificateless paradigms, the semi-trusted third party called the Key Generation Center (KGC) only supplies one half of the user's private key (i.e. partial private key) which is derived from her publicly available information. The other half of the user's private key (i.e. secret value) is computed and kept secure by the user herself. Mainly, in a certificateless system, users are in charge of computing and publishing (e.g. on a public bulletin) their public keys.

*Corresponding author. Email: rouzbeh.behnia@mmu.edu.my

Since there is no certificate to deliver the authenticity of public keys, it should be expected that the adversary is able to replace the public keys of users with public keys of its choice. We will further discuss about the adversarial models of certificateless systems in Section 3.

Ordinary digital signatures have countless applications in today's digital world. They provide authentication, integrity and non-repudiation. Ordinary digital signatures are publicly verifiable, more precisely, any user in the system with knowledge of the signer's public key is able to verify the validity of the signatures generated by that signer. This feature, however, may not be suitable in some situations (i.e. when two parties sign a confidential document). Chaum and van Antwerpen [5] introduced the notion of undeniable signature scheme with the aim of limiting the public verifiability of ordinary digital signatures. In the new notion, verifying the validity or invalidity of a signature is only possible with the direct help of its signer and via the confirmation or disavowal protocol. In addition to the advantages of ordinary digital signatures, undeniable signatures provide privacy for signers by limiting the public verifiability of their signatures. Among the main applications of undeniable signature schemes, we can name software licensing, e-cash and e-voting [3,5,25].

The first certificateless undeniable signature scheme was proposed by Duan [12]. The proposed scheme requires two expensive pairing evaluations in its sign algorithm and its confirmation and disavowal protocols are quite costly as well. Duan's scheme can be viewed as the certificateless version of the identity-based undeniable signature scheme proposed by Libert and Quisquater [23]. Recently, an efficient certificateless undeniable signature scheme was put forth by Zhao and Ye [27]. The new scheme does not need any pairing evaluation in its sign algorithm. However, the unforgeability of the scheme is questionable since it is only secure in a weak security model where the adversary is not allowed to query for any signatures associated with the private key of the target signer. This is a strong assumption since in the real world, the adversary can easily get hold of signatures which were issued by the target signer. Moreover, the adversary is not allowed to query for the secret value nor replace the target user's public key which is again a strong assumption since a Type I adversary (will be defined in Section 3.1) should be able to perform both or at least one of the mentioned queries. Therefore, the scheme does not possess the notion of invisibility against such adversary.

### 1.1 *Contribution*

In this paper, we first formalize the strong security model for certificateless undeniable signature schemes. We then put forth an efficient certificateless undeniable signature scheme. Comparing to the only certificateless undeniable signature scheme that is secure in the strong security model [12], our scheme is much more efficient in its signature generation, proof generation and proof verification algorithms. We employ the pairing-based version of Jakobsson et al.'s technique [20] in order to provide non-interactive designated verifier proofs in the confirmation and disavowal protocols of our scheme to address the man-in-the-middle [10] and blackmailing attacks [11,19]. Furthermore, we show how the signer in our scheme is able to selectively convert her undeniable signatures to ordinary digital signatures by eliminating the trapdoor commitments in the proof of confirmation/disavowal protocol. We prove the security of our efficient scheme in the strong security model and based on the hardness of some well-known mathematical problems.

### 1.2 *Paper organization*

The rest of the paper is organized as follows. In Section 2, we provide some preliminaries and definitions which are going to be used throughout this paper. In Section 3, we define the notion of certificateless undeniable signature scheme and formalize the strong security model for such

schemes. In Section 4, we proffer our efficient certificateless undeniable signature scheme and compare its efficiency with the only secure scheme in the strong model [12]. We provide the security analysis of our scheme in Section 5 and conclude our paper in Section 6.

## 2. Preliminaries

### 2.1 *Bilinear pairing and the related problems*

We let $\mathbb{G}_1$ be an additive cyclic group of prime order $q$, and $\mathbb{G}_2$ be a multiplicative cyclic group of the same order. We denote $P$ as a generator of $\mathbb{G}_1$. We also pick an admissible bilinear mapping function $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ which satisfies the following properties:

(1) *Bilinearity*: For every $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q$ we have $e(aP, bQ) = e(P, Q)^{ab}$.
(2) *Non-degeneracy*: There exist $P, Q \in \mathbb{G}_1$ such that $e(P, Q) \neq 1$.
(3) *Computability*: $e$ is efficiently computable.

*Bilinear Diffie–Hellman* (*BDH*) *problem*: Given $(P, aP, bP, cP)$, the BDH problem is to compute $e(P, P)^{abc}$, for $P$ as a random generator of $\mathbb{G}_1$ and a random selection of $a, b, c \in \mathbb{Z}_q$.

*Decisional Bilinear Diffie–Hellman* (*DBDH*) *problem*: Given $(P, aP, bP, cP, h)$, the DBDH problem is to decide whether $h = e(P, P)^{abc}$, for for $P$ as a random generator of $\mathbb{G}_1$ and a random selection of $a, b, c \in \mathbb{Z}_q$.

The hardness of the BDH problem in $\mathbb{G}_1$ is assumed to be equivalent to the Discrete Logarithm problem [7]. As it was shown in [7], the DBDH problem in $\mathbb{G}_1$ is certainly not any harder than the conventional Decisional Diffie–Hellman problem, nevertheless, there is no known probabilistic polynomial time (PPT) algorithm capable of solving the DBDH problem so far.

## 3. Certificateless undeniable signature scheme

Generally, a certificateless undeniable signature scheme consists of the following algorithms and protocols.

*Setup*: Given the security parameter(s), proper instances of groups $\mathbb{G}_1$ and $\mathbb{G}_2$ will be generated, the KGC's key pair $(s, P_{Pub})$ would be computed, and the system public parameters *params* will be produced and published in the system.

*Set-user-key*: Using this algorithm, the user with identity ID picks her secret value $x_{ID}$ and computes the corresponding public key $PK_{ID}$.

*Partial-private-key-extract*: Provided the user's identity ID and public key $PK_{ID}$, the KGC uses the system wide secret key $s$ to compute the user's partial private key $d_{ID}$.

*Set-private-key*: The user with identity ID and public key $PK_{ID}$ uses her secret value $x_{ID}$ and partial private key $d_{ID}$ to compute her private key $S_{ID}$.

*Sign*: Provided a message $m$ to be signed, the signer with identity ID and public key $PK_{ID}$ uses her private key $S_{ID}$ to issue an undeniable signature $\sigma$.

*Confirmation*: Given a tuple $(m, \sigma, ID, PK_{ID})$, where $\sigma$ is a valid signature on message $m$ for a signer with identity ID and public key $PK_{ID}$, the signer uses her private key $S_{ID}$ to generate a confirmation proof transcript for a verifier (possibly designated) to confirm the validity of the tuple $(m, \sigma, ID, PK_{ID})$.

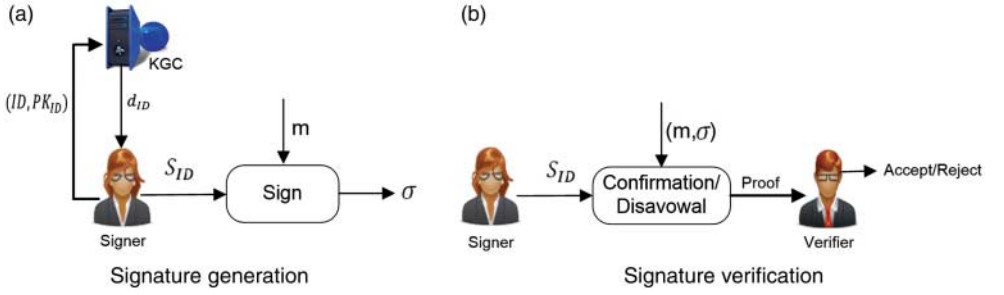


Figure 1. Certificateless undeniable signature generation and verification.

*Disavowal*: Similar to the confirmation protocol, except that $\sigma$ is an invalid signature and the claimed signer uses her private key $S_{ID}$ to generate a proof to disavow the validity of the signature $\sigma$.

As it is shown in Figure 1, a certificateless undeniable signature schemes can be viewed in two phases, namely a signature generation phase and a signature verification phase.

### 3.1 *Security models of certificateless undeniable signature schemes*

Since there is no certificate to deliver the authenticity of public keys in certificateless systems, we always consider two types of adversaries when formulating the security models of certificateless schemes.

- A Type I adversary $\mathcal{A}_I$ models a third party adversary who has no knowledge over the system wide secret key $s$, but is allowed to replace the users' public keys with public keys of its choice.
- A Type II adversary $\mathcal{A}_{II}$, on the other hand, models an eavesdropping KGC who has knowledge over the system wide secret key $s$. However, this type of adversary is not allowed to replace the users' public keys.

In some of the proposed security models [1,18,22], a Type II adversary (malicious KGC) is assumed to generate its key pair honestly and initiate attacks only after the setup step (i.e. the KGC is initially benign). In [2], Au *et al*. defined a security model against a *malicious-but-passive* KGC, whereas the malicious KGC is assumed to generate its key pair dishonestly (i.e. the KGC is malicious from the beginning) and somehow extracts the private key of the target user from her public key. Even though, this type of adversary was never captured in the security models of [1,17,22], the authors showed that these schemes and any other scheme which has the same key generation as [1], is vulnerable against this type of attack.

In this paper, we make use of the binding method [1] in order to ensure the security against malicious-but-passive KGC attacks (we will further discuss on how we address the attack in Section 5). Moreover, using the binding technique, we can lift the trust level of the KGC in our scheme to level 3 of Girault's trust level hierarchy[1] [14].

The following oracles can be queried by an adversary $\mathcal{A} \in \{\mathcal{A}_I, \mathcal{A}_{II}\}$ based on the games' specifications which are to be discussed a bit later.

*Hash-oracle* ($\mathcal{O}^{hash}$): $\mathcal{A}$ can query different hash functions $H$ in the system with any inputs of his choice.

*Public-key-request* ($\mathcal{O}^{pub\_key\_req}$): $\mathcal{A}$ can query for the public key of any user in the system.

*Partial-private-key-extract* ($\mathcal{O}^{part\_key\_extract}$): By providing the user's identity ID and the corresponding public key $PK_{ID}$, $\mathcal{A}$ can query for the user's partial private key $d_{ID}$.

*Secret-value-extract* ($\mathcal{O}^{sec\_val\_extract}$): By providing the user's identity ID and the corresponding public key $PK_{ID}$, $\mathcal{A}$ can query for the user's secret value $x_{ID}$.

*Private-key-extract* ($\mathcal{O}^{priv\_key\_extract}$): By providing the user's identity ID and the corresponding public key $PK_{ID}$, $\mathcal{A}$ can query for the user's private key $S_{ID}$.

*Public-key-replacement* ($\mathcal{O}^{pub\_key\_replace}$): $\mathcal{A}$ is allowed to replace the public key $PK_{ID}$ of any user ID with public key of his choice $PK'_{ID}$.

*Sign-oracle* ($\mathcal{O}^{sign}$): By providing a message $m$, and the identity ID of the alleged signer (with public key $PK_{ID}$) no matter whether the identity ID appears in the public key replace query or not, the oracle returns a valid undeniable signature $\sigma$.

*Confirmation/Disavowal-oracle* ($\mathcal{O}^{conf/disav}$): By providing a valid/invalid message-signature pair $(m, \sigma)$, the identity ID of the claimed signer with public key $PK_{ID}$, and possibly the identity and public key of the designated verifier, this oracle returns the confirmation/disavowal proof transcript in order to prove the validity/invalidity of the signature $\sigma$.

An undeniable signature scheme is said to be secure if it meets the following security notions:

(1) *Existential unforgeability:* The notion of existential unforgeability of a certificateless undeniable signature scheme ensures the inability of an adversary $\mathcal{A} \in \{\mathcal{A}_I, \mathcal{A}_{II}\}$ to generate signatures on behalf of any user (which the private key has not been exposed) in the system.
(2) *Invisibility:* The notion of invisibility [6], distinguishes undeniable signatures from ordinary digital signatures. Provided a message-signature pair $(m, \sigma)$ for the claimed signer with identity $ID$ and public key $PK_{ID}$, the notion of invisibility implies the inability of an adversary $\mathcal{A} \in \{\mathcal{A}_I, \mathcal{A}_{II}\}$ to determine the validity or invalidity of the signature $\sigma$ without the help of its signer.
(3) *Anonymity:* The notion of anonymity was introduced by Galbraith and Mao [13], and it is considered as the most relevant notion to undeniable signature scheme. Informally, given the identities and public keys of two possible signers and a message-signature pair $(m, \sigma)$, the notion of anonymity implies the inability of an adversary $\mathcal{A} \in \{\mathcal{A}_I, \mathcal{A}_{II}\}$ to determine which signer has generated the signature $\sigma$.

In the following, we propose our security models for certificateless undeniable signature schemes. In addition to the capabilities of adversaries in [12], we allow the adversary $\mathcal{A} \in \{\mathcal{A}_I, \mathcal{A}_{II}\}$ to replace the public keys of any user in the system (excluding the target user for a Type II adversary $\mathcal{A}_{II}$).

We first define our security models against a Type I adversary (through Definition 1–3) and then against a Type II adversary (through Definition 4–6).

DEFINITION 1 *We consider a certificateless undeniable signature scheme to be existentially unforgeable under adaptive chosen message, identity and public key attacks if no PPT Type I adversary $\mathcal{F}_I$ has a non-negligible advantage in the following game*:

*Setup phase*. The challenger $\mathcal{C}$ runs the setup algorithm and provides $\mathcal{F}_I$ with the system wide public parameters params.

*Query phase*. $\mathcal{F}_I$ can adaptively query $\mathcal{O}^{hash}$, $\mathcal{O}^{pub\_key\_req}$, $\mathcal{O}^{part\_key\_extract}$, $\mathcal{O}^{priv\_key\_extract}$, $\mathcal{O}^{pub\_key\_replace}$, $\mathcal{O}^{sign}$ and $\mathcal{O}^{conf/disav}$. $\mathcal{C}$ responds to all the queries accordingly (as stated in their definition above).

At the end of the game, $\mathcal{F}_I$ outputs a valid message-signature pair $(m^*, \sigma^*)$ for the signer with identity $ID^*$ and public key $PK_{ID^*}$. $\mathcal{F}_I$ wins the above game if $(ID^*, PK_{ID^*})$ was never queried

to $\mathcal{O}^{part\_key\_extract}$ or $\mathcal{O}^{priv\_key\_extract}$ and $\sigma^*$ was never outputted by $\mathcal{O}^{sign}$ on the input of $m^*$ and $(ID^*, PK_{ID^*})$.

DEFINITION 2 *A certificateless undeniable signature scheme is considered to fulfill the notion of invisibility under adaptive chosen message, identity and public key attacks if no PPT Type I adversary $\mathcal{D}_I$ has a non-negligible advantage in the following game:*

*Setup phase*. This phase takes place identical to the game of Definition 1.

*Query phase (before challenge)*. $\mathcal{D}_I$ can initiate polynomially bounded number of queries as defined in the game of Definition 1.

*Challenge phase*. After the first round of queries, $\mathcal{D}_I$ requests a challenge signature on a message $m^*$ for a signer with identity $ID^*$ and public key $PK_{ID^*}$. Where $(ID^*, PK_{ID^*})$ was never queried to $\mathcal{O}^{part\_key\_extract}$ or $\mathcal{O}^{priv\_key\_extract}$. Then, $\mathcal{C}$ generates the challenge signature $\sigma^*$ based on the outcome of a random coin toss $c \in \{0, 1\}$. If $c = 0$, $\mathcal{C}$ will select a random $\sigma^* \in \mathcal{S}$, where $\mathcal{S}$ is the signature space and sends $\sigma^*$ to $\mathcal{D}_I$. Otherwise, if $c = 1$, the challenger generates a valid signature $\sigma^*$ and sends it back to $\mathcal{D}_I$.

*Query phase (after challenge)*. $\mathcal{D}_I$ initiates the second round of queries, this time, $\mathcal{D}_I$ is not allowed to query $\mathcal{O}^{part\_key\_extract}$ or $\mathcal{O}^{priv\_key\_extract}$ on the identity $ID^*$ with public key $PK_{ID^*}$, nor the confirmation/disavowal oracle on $(m^*, \sigma^*, ID^*, PK_{ID^*})$.

At the end of the game, $\mathcal{D}_I$ outputs its decision bit $d \in \{0, 1\}$ and wins the game if $d = c$.

DEFINITION 3 *A certificateless undeniable signature scheme is considered to fulfill the notion of anonymity under adaptive chosen message, identity and public key attacks if no PPT Type I adversary $\mathcal{D}_I$ has a non-negligible advantage in the following game*:

*Setup phase*. This phase takes place identical to the game of Definition 1.

*Query phase (before challenge)*. $\mathcal{D}_I$ can initiate polynomially bounded number of queries as defined in the game of Definition 1.

*Challenge phase*. After the first round of queries, $\mathcal{D}_I$ produces a message $m^*$, and two tuples $(ID_0, PK_{ID_0})$ and $(ID_1, PK_{ID_1})$ containing the identities and the public keys of two possible signers with the limitation that they were never queried to $\mathcal{O}^{part\_key\_extract}$ or $\mathcal{O}^{priv\_key\_extract}$. The challenger $\mathcal{C}$ responds based on the outcome of a random coin toss $c \in \{0, 1\}$ and generates the challenge signature $\sigma^*$ on the message $m^*$ for a signer with identity $ID_c$ and public key $PK_{ID_c}$.

*Query phase (after challenge)*. $\mathcal{D}_I$ initiates the second round of queries, this time, $\mathcal{D}_I$ is not allowed to query $\mathcal{O}^{part\_key\_extract}$ or $\mathcal{O}^{priv\_key\_extract}$ on $(ID_0, PK_{ID_0})$ or $(ID_1, PK_{ID_1})$, nor the confirmation/disavowal oracle on tuples $(m^*, \sigma^*, ID_0, PK_{ID_0})$ or $(m^*, \sigma^*, ID_1, PK_{ID_1})$.

At the end of the game, $\mathcal{D}_I$ outputs its decision bit $d \in \{0, 1\}$ and wins the game if $d = c$.

DEFINITION 4 *We consider a certificateless undeniable signature scheme to be existentially unforgeable under adaptive chosen message, identity and public key attacks if no PPT Type II adversary $\mathcal{F}_{II}$ has a non-negligible advantage in the following game:*

*Setup phase*. The challenger $\mathcal{C}$ runs the setup algorithm and provides $\mathcal{F}_{II}$ with the master secret key $s$ and the system wide public parameters params.
*Query phase*. $\mathcal{F}_{II}$ can adaptively query $\mathcal{O}^{hash}$, $\mathcal{O}^{pub\_key\_req}$, $\mathcal{O}^{sec\_val\_extract}$, $\mathcal{O}^{priv\_key\_extract}$, $\mathcal{O}^{pub\_key\_replace}$, $\mathcal{O}^{sign}$ and $\mathcal{O}^{conf/disav}$. $\mathcal{C}$ responds to all the queries accordingly (as stated in their definition above).

At the end of the game, $\mathcal{F}_{II}$ outputs a valid message-signature pair $(m^*, \sigma^*)$ for a signer with identity $ID^*$ and public key $PK_{ID^*}$. $\mathcal{F}_{II}$ wins the above game if $(ID^*, PK_{ID^*})$ was never queried to $\mathcal{O}^{sec\_val\_extract}$, $\mathcal{O}^{priv\_key\_extract}$ or $\mathcal{O}^{pub\_key\_replace}$, and $\sigma^*$ was never outputted by $\mathcal{O}^{sign}$ on the input of $m^*$ and $(ID^*, PK_{ID^*})$.

DEFINITION 5 *A certificateless undeniable signature scheme is considered to fulfill the notion of invisibility under adaptive chosen message, identity and public key attacks if no PPT Type II adversary $\mathcal{D}_{II}$ has a non-negligible advantage in the following game:*

*Setup phase*. This phase takes place identical to the game of Definition 4.

*Query phase (before challenge).* $\mathcal{D}_{II}$ can initiate polynomially bounded number of queries as defined in the game of Definition 4.

*Challenge phase*. After the first round of queries, $\mathcal{D}_{II}$ requests a challenge signature on a message $m^*$ for a signer with identity $ID^*$ and public key $PK_{ID^*}$. Where $(ID^*, PK_{ID^*})$ was never queried to $\mathcal{O}^{sec\_val\_extract}$, $\mathcal{O}^{priv\_key\_extract}$ or $\mathcal{O}^{pub\_key\_replace}$. The challenger $\mathcal{C}$ then generates the challenge signature $\sigma^*$ based on the outcome of a random coin toss $c \in \{0, 1\}$. If $c = 0$, $\mathcal{C}$ selects a random $\sigma^* \in \mathcal{S}$, where $\mathcal{S}$ is the signature space and sends $\sigma^*$ to $\mathcal{D}_{II}$. Otherwise, if $c = 1$, the challenger generates a valid signature $\sigma^*$ and sends it back to $\mathcal{D}_{II}$.

*Query phase (after challenge).* $\mathcal{D}_{II}$ initiates the second round of queries, this time, $\mathcal{D}_{II}$ is not allowed to query $\mathcal{O}^{sec\_val\_extract}$, $\mathcal{O}^{priv\_key\_extract}$ or $\mathcal{O}^{pub\_key\_replace}$ on the identity $ID^*$ with public key $PK_{ID^*}$, nor the confirmation/disavowal oracle on $(m^*, \sigma^*, ID^*, PK_{ID^*})$.

At the end of the game, $\mathcal{D}_{II}$ outputs its decision bit $d \in \{0, 1\}$ and wins the game if $d = c$.

DEFINITION 6 *A certificateless undeniable signature scheme is considered to fulfill the notion of anonymity under adaptive chosen message, identity and public key attacks if no PPT Type II adversary $\mathcal{D}_{II}$ has a non-negligible advantage in the following game*:

*Setup phase*. This phase takes place identical to the game of Definition 4.

*Query phase (before challenge).* $\mathcal{D}_{II}$ can initiate polynomially bounded number of queries as defined in the game of Definition 4.

*Challenge phase*. After the first round of queries, $\mathcal{D}_{II}$ produces a message $m^*$, and two tuples $(ID_0, PK_{ID_0})$ and $(ID_1, PK_{ID_1})$ containing the identities and public keys of two possible signers with the limitation that they were never queried to $\mathcal{O}^{sec\_val\_extract}$, $\mathcal{O}^{priv\_key\_extract}$ or $\mathcal{O}^{pub\_key\_replace}$. The challenger $\mathcal{C}$ responds based on the outcome of a random coin toss $c \in \{0, 1\}$ and generates the challenge signature $\sigma^*$ on the message $m^*$ for a signer with identity $ID_c$ and public key $PK_{ID_c}$.

*Query phase (after challenge).* $\mathcal{D}_{II}$ initiates the second round of queries, this time, $\mathcal{D}_{II}$ is not allowed to query $\mathcal{O}^{sec\_val\_extract}$, $\mathcal{O}^{priv\_key\_extract}$ or $\mathcal{O}^{pub\_key\_replace}$ on $(ID_0, PK_{ID_0})$ or $(ID_1, PK_{ID_1})$, nor the confirmation/disavowal oracle on tuples $(m^*, \sigma^*, ID_0, PK_{ID_0})$ or $(m^*, \sigma^*, ID_1, PK_{ID_1})$.

At the end of the game, $\mathcal{D}_{II}$ outputs its decision bit $d \in \{0, 1\}$ and wins the game if $d = c$.

## 4. Efficient certificateless undeniable signature scheme

In this section, we propose our efficient scheme in detail and then compare its efficiency to the only scheme that is secure in the strong security model [12].

### 4.1 *The proposed scheme*

*Setup*: The setup algorithm is initiated by the KGC. It takes two security parameters $k$ and $l$, and generates groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order $q \geq 2^k$, a generator $P$ of $\mathbb{G}_1$, and an admissible bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. It also chooses 4 cryptographic hash functions: $H_1 : \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_1$, $H_2 : \{0,1\}^* \times \{0,1\}^l \times \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_1$, and $H_3, H_4 : \mathbb{G}_2 \times \ldots \times \mathbb{G}_2 \rightarrow \mathbb{Z}_q$. Next, it picks $s \in \mathbb{Z}_q$ randomly as the master secret key and calculates $P_{Pub} = sP$ as the corresponding public key. The KGC's public key $P_{Pub}$ and the system's public parameters $params = (q, \mathbb{G}_1, \mathbb{G}_2, P, P_{Pub}, H_1, H_2, H_3, H_4)$ will be made available to all the users in the system.

*Set-user-keys:* The user with identity $ID$ chooses $x_{ID} \in \mathbb{Z}_q$ randomly as her secret value and computes her public key as $PK_{ID} = (TV_{ID} = x_{ID}P, TS_{ID} = x_{ID}P_{Pub})$.

*Partial-private-key-extraction*: Provided the user's identity $ID$ and public key $PK_{ID} = (TV_{ID}, TS_{ID})$, the KGC computes her partial private key as $d_{ID} = sQ_{ID} = sH_1(ID, TV_{ID}, TS_{ID})$, and delivers it to the user in a secure manner.

*Set-private-key*: After the user with identity $ID$ and public key $PK_{ID}$ received her partial private key $d_{ID}$, she will form her private key as $S_{ID} = x_{ID}d_{ID}$.

*Sign*: To issue a signature on a message $m \in \{0,1\}^*$, the signer Alice with identity $ID_A$ and public key $PK_{ID_A} = (TV_{ID_A}, TS_{ID_A})$ chooses a random string $r \in \{0,1\}^l$ and computes $h_S = H_2(m, r, ID_A, TV_{ID_A}, TS_{ID_A}) \in \mathbb{G}_1$. She then uses her private key $S_{ID_A}$ to calculate $\lambda = e(h_S, S_{ID_A})$, and forms the signature $\sigma = (r, \lambda)$.

*Confirmation*: Given a valid message-signature pair $(m, \sigma = (r, \lambda))$, the alleged signer (Alice) with identity $ID_A$ and public key $PK_{ID_A} = (TV_{ID_A}, TS_{ID_A})$ generates a non-interactive confirmation proof for the designated verifier Bob (with identity $ID_B$ and public key $PK_{ID_B} = (TV_{ID_B}, TS_{ID_B})$) as follows.

- Choose $\nu \in \mathbb{Z}_q$ and $U, Y \in \mathbb{G}_1$ at random, and compute $W = e(P, U)e(TS_{ID_B}, Q_B)^{\nu}, N = e(P, Y)$, and $O = e(H_2(m, r, ID_A, TV_{ID_A}, TS_{ID_A}), Y)$.
- Set the values of $h_C = H_3(W, N, O, m, \sigma)$ and $B = Y - (h_C + \nu)S_{ID_A}$ in order to form the confirmation proof transcript as $(U, \nu, h_C, B)$.

Upon receiving the confirmation proof transcript $(U, \nu, h_C, B)$, Bob checks if $e(TV_{ID_A}, P_{Pub}) = e(TS_{ID_A}, P)$ holds, he computes $W' = e(P, U)e(TS_{ID_B}, Q_B)^{\nu}$, $N' = e(P, B)e(TS_{ID_A}, Q_A)^{(h_C+\nu)}$ and $O' = e(H_2(m, r, ID_A, TV_{ID_A}, TS_{ID_A}), B)\lambda^{(h_C+\nu)}$ and will confirm the validity of the message-signature pair $(m, \sigma = (r, \lambda))$ for the alleged signer (Alice) if and only if $h_C = H_3(W', N', O', m, \sigma)$ holds.

*Disavowal*: Given an invalid message-signature pair $(m, \sigma = (r, \lambda))$, the claimed signer (Alice) with identity $ID_A$ and public key $PK_{ID_A} = (TV_{ID_A}, TS_{ID_A})$ generates a non-interactive disavowal proof for the designated verifier Bob (with identity $ID_B$ and public key $PK_{ID_B} = (TV_{ID_B}, TS_{ID_B})$) as follows.

- Parse $\sigma$ into $(r, \lambda)$ and choose $\nu, \alpha \in \mathbb{Z}_q$ and $U \in \mathbb{G}_1$ at random in order to compute $W = e(P, U)e(TS_{ID_B}, Q_B)^{\nu}$ and $C = (e(H_2(m, r, ID_A, TV_{ID_A}, TS_{ID_A}), S_{ID_A})/\lambda)^{\alpha}$.
- Alice has to prove her knowledge of a pair $(J, \alpha) \in \mathbb{G}_1 \times \mathbb{Z}_q$ such that $C = (e(H_2(m, r, ID_A, TV_{ID_A}, TS_{ID_A}), J)/\lambda^{\alpha})$ and $e(P, J) = e(TS_{ID_A}, Q_A)^{\alpha}$ hold. In order to do so, she works as follows.
  - Choose $y \in \mathbb{Z}_q$ and $I \in \mathbb{G}_1$ randomly to compute $K = e(P, I)e(TS_{ID_A}, Q_A)^{-y}$, and $L = e(H_2(m, r, ID_A, TV_{ID_A}, TS_{ID_A}), I)\lambda^{-y}$.

Table 1. Efficiency comparison.

| | | Duan's scheme [12] | Proposed scheme |
|---|---|---|---|
| Sign | – | $2pe + 1ex$ | $1pe$ |
| Confirmation | Signer | $6pe + 3ex + 3sm$ | $4pe + 1ex + 1sm$ |
| | Verifier | $8pe + 6ex + 1sm$ | $7pe + 3ex$ |
| Disavowal | Signer | $10pe + 8ex + 4sm$ | $6pe + 4ex + 2sm$ |
| | Verifier | $8pe + 7ex + 1sm$ | $7pe + 4ex$ |

- Set the values of $h_D = H_4(C, W, K, L, m, \sigma)$, $R = I - (h_D + \nu)J$, and $\mu = y - (h_D + \nu)\alpha$ in order to form the disavowal proof transcript as $(C, U, \nu, h_D, R, \mu)$.

Given the disavowal proof transcript $(C, U, \nu, h_D, R, \mu)$, Bob first checks if $e(TV_{ID_A}, P_{Pub}) = e(TS_{ID_A}, P)$ holds and $C \neq 1$, he computes $W' = e(P, U)e(TS_{ID_B}, Q_B)^{\nu}$, $K' = e(P, R)e(TS_{ID_A}, Q_A)^{-\mu}$ and $L' = e(H_2(m, r, ID_A, TV_{ID_A}, TS_{ID_A}), R)\lambda^{-\mu}C^{(h_D+\nu)}$ and will only accept the proof if $h_D = H_4(C, W', K', L', m, \sigma)$ holds.

### 4.2 *Efficiency and extensions*

*Efficiency*. Efficiency is one of the major concerns when designing cryptographic schemes. The efficiency of pairing-based cryptographic schemes is usually evaluated based on the number of the pairing evaluations, exponentiations and scalar multiplications. However, pairing evaluations are far more expensive comparing to exponentiations and scalar multiplications which makes them the main benchmark when evaluating the efficiency of a pairing-based cryptographic scheme. Table 1 illustrates the efficiency comparison between our scheme and the only certificateless undeniable signature scheme which is secure in the strong security model [12]. It depicts the number of pairing evaluations (*pe*), exponentiation (*ex*) (in $\mathbb{G}_2$), and scalar multiplications (*sm*) (in $\mathbb{G}_1$) in both schemes.

As it is shown in Table 1, our scheme is much more efficient in signature generation, proof generation, and proof verification compared to the one proposed by Duan [12]. Besides, the length of our confirmation and disavowal proofs are each $2q$ bits shorter comparing to the ones in [12].

Furthermore, we can reduce our signature size by using the technique of Katz and Wang [21] by replacing the $l$-bit random value $r$ with a single bit while maintaining the same security level.

*Convertibility*. The signer in our scheme is able to selectively convert her undeniable signatures to universally verifiable ones by omitting the trapdoor functions in the confirmation\disavowal protocols. In order to generate a universally verifiable proof on a valid tuple $(m, \sigma, ID_A, TV_{ID_A}, TS_{ID_A})$, she computes $N = e(P, Y)$, and $O = e(H_2(m, r, ID_A, TV_{ID_A}, TS_{ID_A}), Y)$, $h_{SC} = H_3(N, O, m, \sigma)$, and $B = Y - h_{SC}S_{ID_A}$ and publishes the proof as $(h_{SC}, B)$. It can be easily shown that any user in the system is able to verify the validity of the signature $\sigma$ using the proof $(h_{SC}, B)$. This technique can be directly applied to generate universally verifiable disavowal proofs for an invalid signature.

## 5. Security analysis

In [2], the malicious-but-passive KGC initiates the attack in the Setup algorithm (i.e. before the system parameters are published). It starts by picking $\alpha \in \mathbb{Z}_q$ at random and setting the group random generator $P = \alpha(H(ID))$ where $ID$ is the identity of the target user.

Evidently, it can easily compute the target user's private key $S_{ID} = x_{ID}d_{ID}$ from her public key $PK_{ID} = (TV_{ID} = x_{ID}P, TS_{ID} = x_{ID}P_{Pub})$ by computing $(1/\alpha)(TS_{ID}) = (1/\alpha)(x_{ID}P_{Pub}) = (1/\alpha)(x_{ID}(sP)) = (1/\alpha)(x_{ID}(s\alpha H(ID)) = x_{ID}(sH(ID)) = x_{ID}d_{ID} = S_{ID})$. In our scheme, we prevented this attack by employing the binding method [1] and including the public key of the user in $H_1(ID, TV_{ID}, TS_{ID})$. It is easy to see that in the Setup algorithm, before the system public parameters are generated and published in the system, the users have not yet computed their public keys (the system public parameters (e.g. $P$) are essential for users to set up their keys). Therefore, the malicious-but-passive KGC does not have knowledge on the public key of the target user when setting up the system public parameters and hence, it cannot set the value of $P$ maliciously as in the above attack.

Beside addressing the attack against a malicious-but-passive KGC [2], employment of the binding method elevates the trust level of the KGC to level 3 in Girault's hierarchy [14].

We used the pairing-based version of the Jakobsson et al.'s method [20] in the body of our confirmation protocol and the method of Camenisch and Shoup [4] in the disavowal protocol of our scheme to prove the inequality of two discrete logarithms. Similar to [20,23], it is straightforward to show that both protocols are sound and complete while enjoying the property of non-transferability, therefore, we do not do it here.

We prove that our scheme is existentially unforgeable and has the property of invisibility against both Type I and Type II adversaries in the random oracle model. Based on the work of Galbraith and Mao [13], the notion of anonymity is equivalent to the notion of invisibility in the sense we stated in our security models. Consequently, we can use the same technique as in [13] to prove the anonymity of our scheme against Type I and Type II adversaries under the hardness of the DBDH problem. For the details of proving the equivalency of the invisibility and anonymity notions, we refer the reader to the proof of Theorem 3 in [13].

We employed Goh and Jarecki's approach [15] in our scheme so as to avoid using the forking lemma [24] and obtain a tighter security reduction in our security proofs.

We prove the security of our scheme against a Type I adversary in Theorems 1 and 2, and will do the same for a Type II adversary in Theorems 3 and 4.

THEOREM 1 *If there exists a Type I adversary $\mathcal{F}_I$ that can submit $q_E$ private key and partial private key extraction queries, $q_{CD}$ confirmation and disavowal queries, and $q_{H_i}$ queries to random oracle $H_i$ for $i \in \{1, 2, 3, 4\}$ and succeed in an existential forgery (in the game defined in Definition 1) with a non-negligible success probability $\epsilon_{\mathcal{F}_I}$, then there exists a PPT algorithm $\mathcal{C}$ which can use $\mathcal{F}_I$ to solve a random instance $(P, aP, bP, cP)$ of the BDH problem with probability*:

$$\epsilon_{\mathcal{C}} \geqslant \frac{\epsilon_{\mathcal{F}_I} - (2q_{H_3} + q_{CD} + 1)2^{-k}}{\mathbf{e}(q_E + 1)(q_{CD} + 1)}$$

*Proof* Please refer to Appendix. ■

THEOREM 2 *If there exists a Type I adversary $\mathcal{D}_I$ that can submit $q_E$ private key and partial private key extraction queries, $q_{CD}$ confirmation and disavowal queries, and $q_{H_i}$ queries to random oracle $H_i$ for $i \in \{1, 2, 3, 4\}$ and be able to breach the invisibility property (win the game defined in Definition 2) with non-negligible success probability $\epsilon_{D_I}$, then there exists a PPT algorithm $\mathcal{C}$ which can use $\mathcal{D}_I$ to solve a random instance $(P, aP, bP, cP, h)$ of the DBDH problem with probability*:

$$\epsilon_{\mathcal{C}} \geqslant \frac{\epsilon_{\mathcal{D}_I} - (q_{H_3} + q_{CD})2^{-k}}{\mathbf{e}(q_E + 1)}$$

*Proof* Please refer to Appendix. ■

THEOREM 3 *If there exists a Type II adversary $\mathcal{F}_{II}$ that can submit $q_E$ secret value extraction, private key and public key replacement queries, $q_{CD}$ confirmation and disavowal queries, and $q_{H_i}$ queries to random oracle $H_i$ for $i \in \{1, 2, 3, 4\}$ and succeed in an existential forgery (win the game defined in Definition 4) with non-negligible success probability $\epsilon_{\mathcal{F}_{II}}$, then there exists a PPT algorithm $\mathcal{C}$ which can use $\mathcal{F}_{II}$ to solve a random instance $(P, aP, bP, cP)$ of the BDH problem with probability*:

$$\epsilon_{\mathcal{C}} \geqslant \frac{\epsilon_{\mathcal{F}_{II}} - (2q_{H_3} + q_{CD} + 1)2^{-k}}{\mathbf{e}(q_E + 1)(q_{CD} + 1)}$$

*Proof* Please refer to Appendix. ■

THEOREM 4 *If there exists a Type II adversary $\mathcal{D}_{II}$ that can submit $q_E$ secret value extraction, public key replacement, and private key extraction queries, $q_{CD}$ confirmation and disavowal queries, and $q_{H_i}$ queries to random oracle $H_i$ for $i \in \{1, 2, 3, 4\}$ and be able to breach the invisibility property (win the game defined in Definition 5) with non-negligible success probability $\epsilon_{\mathcal{D}_{II}}$, then there exists a PPT algorithm $\mathcal{C}$ which can use $\mathcal{D}_{II}$ to solve a random instance $(P, aP, bP, cP, h)$ of the DBDH problem with probability*:

$$\epsilon_{\mathcal{C}} \geqslant \frac{\epsilon_{\mathcal{D}_{II}} - (q_{H_3} + q_{CD})2^{-k}}{\mathbf{e}(q_E + 1)}$$

*Proof* Please refer to Appendix. ■

## 6. Conclusion

In this paper, we proffered a new certificateless undeniable signature scheme which has noticeable efficiency advantages to the only existing scheme that is secure in a strong security models in the literature [12]. The trust level of KGC in our scheme is equivalent to the trust level of the Certificate Authority in conventional public key cryptography. Moreover, we proved that our scheme is secure against both Type I and Type II adversaries, where we also considered a special Type II adversary (malicious-but-passive KGC) which was introduced by Au *et al*. [2].

## Acknowledgements

This research was supported by the FRGS grant (FRGS/2/2013/ICT04/MMU/01/1) and Multimedia University Graduate Research Assistant (GRA) scheme.

## Note

1. This was the main incentive in [1].

## References

[1] S. Al-Riyami and K. Paterson, *Certificateless public key cryptography*, in *Advances in Cryptology – ASIACRYPT 2003, Vol. 2894 of Lecture Notes in Computer Science*, C.-S. Laih, ed., Springer, Berlin, 2003, pp. 452–473.
[2] M.H. Au, Y. Mu, J. Chen, D.S. Wong, J.K. Liu, and G. Yang, *Malicious KGC attacks in certificateless cryptography*, Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security – ASIACCS '07, ACM, New York, 2007, pp. 302–311.

[3] C. Boyd and E. Foo, *Offline fair payment protocols using convertible signatures*, in *Advances in Cryptology – ASIACRYPT'98*, Vol. 1514 of Lecture Notes in Computer Science, K. Ohta and D. Pei eds., Springer, Berlin, 1998, pp. 271–285.
[4] J. Camenisch and V. Shoup, *Practical verifiable encryption and decryption of discrete logarithms*, in *Advances in Cryptology – CRYPTO 2003*, Vol. 2729 of Lecture Notes in Computer Science, D. Boneh, ed., Springer, Berlin, 2003, pp. 126–144.
[5] D. Chaum and H. van Antwerpen, *Undeniable signatures*, in *Advances in Cryptology – CRYPTO 89 Proceedings*, Vol. 435 of Lecture Notes in Computer Science, G. Brassard, ed., Springer, Berlin, 1990, pp. 212–216.
[6] D. Chaum, E. van Heijst, and B. Pfitzmann, *Cryptographically strong undeniable signatures, unconditionally secure for the signer*, in *Advances in Cryptology – CRYPTO 91*, Vol. 576 of Lecture Notes in Computer Science, J. Feigenbaum, ed., Springer, Berlin, 1992, pp. 470–484.
[7] J.H. Cheon and D.H. Lee, *Diffie–Hellman problems and bilinear maps*, Cryptology ePrint Archive, Report 2002/117, 2002. Available at http://eprint.iacr.org/.
[8] J-S. Coron, *On the exact security of full domain hash*, in *Advances in Cryptology – CRYPTO 2000*, Vol. 1880 of Lecture Notes in Computer Science, M. Bellare, ed., Springer, Berlin, 2000, pp. 229–235.
[9] I. Damgård, *Towards practical public key systems secure against chosen ciphertext attacks*, in *Advances in Cryptology CRYPTO 91*, Vol. 576 of Lecture Notes in Computer Science, J. Feigenbaum, ed., Springer, Berlin, 1992, pp. 445–456.
[10] Y. Desmedt, C. Goutier, and S. Bengio, *Special uses and abuses of the Fiat-Shamir passport protocol*, in *Advances in Cryptology – CRYPTO '87*, Vol. 293 of Lecture Notes in Computer Science, C. Pomerance, ed., Springer, Berlin,1987, pp. 21–39.
[11] Y. Desmedt and M. Yung, *Weaknesses of undeniable signature schemes*, in *Advances in Cryptology – EUROCRYPT 91*, Vol. 547 of Lecture Notes in Computer Science, D. Davies, ed., Springer, Berlin,1991, pp. 205–220.
[12] S. Duan, *Certificateless undeniable signature scheme*, Inf. Sci. 178(3) (2008), pp. 742–755.
[13] S.D. Galbraith and W. Mao, *Invisibility and anonymity of undeniable and confirmer signatures*, in *Topics in Cryptology – CT-RSA 2003*, Vol. 2612 of Lecture Notes in Computer Science, M. Joye, ed., Springer, Berlin, 2003, pp. 80–97.
[14] M. Girault, *Self-certified public keys*, in *Advances in Cryptology – EUROCRYPT 91*, D. Davies, ed., Springer, Berlin, 1991, pp. 490–497.
[15] E-J. Goh and S. Jarecki, *A signature scheme as secure as the Diffie–Hellman problem*, in *Advances in Cryptology – EUROCRYPT 2003*, Vol. 2656 of Lecture Notes in Computer Science, E. Biham, ed., Springer, Berlin, 2003, pp. 401–415.
[16] S. Hada and T. Tanaka, *On the existence of 3-round zero-knowledge protocols*, in *Advances in Cryptology – CRYPTO 98*, Vol. 1462 of Lecture Notes in Computer Science, H. Krawczyk, ed., Springer, Berlin,1998, pp. 408–423.
[17] X. Huang, Y. Mu, W. Susilo, D. Wong, and W. Wu, *Certificateless signature revisited*, in *Information Security and Privacy*, Vol. 4586 of Lecture Notes in Computer Science, J. Pieprzyk, H. Ghodosi, and E. Dawson, eds., Springer, Berlin, 2007, pp. 308–322.
[18] X. Huang, W. Susilo, Y. Mu, and F. Zhang, *On the security of certificateless signature schemes from Asiacrypt 2003*, in *Cryptology and Network Security*, Vol. 3810 of Lecture Notes in Computer Science, Y. Desmedt, H. Wang, Y. Mu and Y. Li, eds., Springer, Berlin, 2005, pp. 13–25.
[19] M. Jakobsson, *Blackmailing using undeniable signatures*, in *Advances in Cryptology – EUROCRYPT '94*, Vol. 950 of Lecture Notes in Computer Science, A. De Santis, ed., Springer, Berlin,1995, pp. 425–427.
[20] M. Jakobsson, K. Sako, and R. Impagliazzo, *Designated verifier proofs and their applications*, in *Advances in Cryptology – EUROCRYPT 96*, Vol. 1070 of Lecture Notes in Computer Science, U. Maurer, ed., Springer, Berlin, 1996, pp. 143–154.
[21] J. Katz and N. Wang, *Efficiency improvements for signature schemes with tight security reductions*, Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS '03, ACM, New York, NY, 2003, pp. 155–164.
[22] X. Li, K. Chen, and L. Sun, *Certificateless signature and proxy signature schemes from bilinear pairings*, Lithuanian Math. J. 45 (2005), pp. 76–83.
[23] B. Libert and J.-J. Quisquater, *Identity-based undeniable signatures*, in *Topics in Cryptology – CT-RSA 2004*, Vol. 2964 of Lecture Notes in Computer Science, T. Okamoto, ed., Springer, Berlin, 2004, pp. 112–125.
[24] D. Pointcheval and J. Stern, *Security proofs for signature schemes*, in *Advances in Cryptology – EUROCRYPT 96*, U. Vol. 1070 of Lecture Notes in Computer Science, Maurer, ed., Springer, Berlin, 1996, pp. 387–398.
[25] K. Sakurai and S. Miyazaki, *An anonymous electronic bidding protocol based on a new convertible group signature scheme*, in *Information Security and Privacy*, Vol. 1841 of Lecture Notes in Computer Science, E. Dawson, A. Clark, and C. Boyd, eds, Springer, Berlin, 2000, pp. 385–399.
[26] A. Shamir, *Identity-based cryptosystems and signature schemes*, in *Advances in Cryptology – CRYPTO 84*, Vol. 196 of Lecture Notes in Computer Science, G. Blakley and D. Chaum, eds., Springer, Berlin, 1985, pp. 47–53.
[27] W. Zhao and D. Ye, *Certificateless undeniable signatures from bilinear maps*, Inf. Sci. 199 (2012), pp. 204–215.

## Appendix

### *Proof of Theorem 1*

We prove that if there exists a Type I adversary $\mathcal{F}_I$ which can win the game defined in Definition 1, then one can construct a PPT algorithm $\mathcal{C}$ that can run $\mathcal{F}_I$ as its subroutine to solve a random instance $(P, aP, bP, cP)$ of the BDH problem with probability at least $\epsilon_{\mathcal{C}}$. $\mathcal{C}$ works as $\mathcal{F}_I$'s challenger. It starts by initiating the set-up algorithm, and provides $\mathcal{F}_I$ with the system wide public parameters $params = (q, \mathbb{G}_1, \mathbb{G}_2, P, P_{Pub}, H_1, H_2, H_3, H_4)$. Where $P_{Pub} = aP$ and $a$ is unknown to $\mathcal{C}$.

$\mathcal{F}_I$ can make queries to random oracles $H_i$ for $i = \{1, 2, 3, 4\}$ and other oracles as defined in the game of Definition 1. $\mathcal{C}$ responds to these queries by keeping lists $\kappa_i$ for $i = \{1, 2, 3, 4\}$ and a list $\kappa_0$ in order to keep track of the values of identity, public key and the corresponding secret value. We assume $\mathcal{F}_I$ always makes a public key request before a $H_1$ query, and a $H_1$ query before it requests for the partial private key of the user.

*Query on* $H_1(ID, TV_{ID}, TS_{ID})$: In order to handle queries on $H_1$ for an identity $ID$ with public key $PK_{ID} = (TV_{ID}, TS_{ID})$, $\mathcal{C}$ first chooses a random $\alpha \in \mathbb{Z}_q$ and flips a random coin $X$ that is taking the value of 0 with probability $\varphi_1$ and the value of 1 with probability $1 - \varphi_1$ (the value of $\varphi_1$ will be calculated in our proof later). Lastly, $\mathcal{C}$ inserts $(ID, TV_{ID}, TS_{ID}, \alpha, X)$ into $\kappa_1$ and returns $H_1(ID, TV_{ID}, TS_{ID}) = \alpha(bP)$ if $X = 1$, and $H_1(ID, TV_{ID}, TS_{ID}) = \alpha P$ if $X = 0$.

*Query on* $H_2(m, r, ID, TV_{ID}, TS_{ID})$:To answer queries on $H_2$, $\mathcal{C}$ first chooses $\beta \in \mathbb{Z}_q$ randomly and flips a random coin $Y$ that takes the value of 0 with probability $\varphi_2$ and the value of 1 with probability $1 - \varphi_2$ (the value of $\varphi_2$ will be calculated in our proof later). Lastly, $\mathcal{C}$ records $(m, r, ID, TV_{ID}, TS_{ID}, \beta, Y)$ in $\kappa_2$ and returns $H_2(m, r, ID, TV_{ID}, TS_{ID}) = \beta(cP)$ if $Y = 1$, and $H_2(m, r, ID, TV_{ID}, TS_{ID}) = \beta P$ if $Y = 0$.

*Query on* $H_3$ and $H_4$: Queries on $H_3$ and $H_4$ will be handled by $\mathcal{C}$ in a random manner, and the outputs will be stored in $\kappa_3$ and $\kappa_4$ respectively.

*Public key request*: To handle a public key request on an identity $ID$, $\mathcal{C}$ checks if $(ID, x_{ID}, TV_{ID}, TS_{ID})$ already exists in $\kappa_0$, then $\mathcal{C}$ returns $PK_{ID} = (TV_{ID}, TS_{ID})$. Otherwise, it picks a random $x_{ID} \in \mathbb{Z}_q$, computes $TV_{ID} = x_{ID}P$ and $TS_{ID} = x_{ID}P_{Pub}$, returns $PK_{ID} = (TV_{ID}, TS_{ID})$ to $\mathcal{F}_I$, and lastly, records $(ID, x_{ID}, TV_{ID}, TS_{ID})$ in $\kappa_0$.

*Partial private key extraction*: Upon receiving an identity $ID$ with public key $PK_{ID} = (TV_{ID}, TS_{ID})$, $\mathcal{C}$ scans $\kappa_1$ for a tuple $(ID, TV_{ID}, TS_{ID}, \alpha, X)$, if $X = 1$, $\mathcal{C}$ reports *failure* and aborts the simulation. Otherwise, it outputs the partial private key as $d_{ID} = \alpha P_{Pub}$.

*Private key extraction*: To handle a private key extraction query on an identity $ID$ with public key $PK_{ID} = (TV_{ID}, TS_{ID})$, $\mathcal{C}$ scans $\kappa_1$ for $(ID, TV_{ID}, TS_{ID}, \alpha, X)$. If $X = 1$, $\mathcal{B}$ reports *failure* and aborts the simulation. Otherwise, it searches $\kappa_0$ to find $(ID, x_{ID}, TV_{ID}, TS_{ID})$ and returns the private key of the user $ID$ as $S_{ID} = \alpha TS_{ID}$.

*Public key replacement*: If $\mathcal{F}_I$ wishes to replace the public key $PK_{ID} = (TV_{ID}, TS_{ID})$ for identity $ID$ with public key of its choice $PK'_{ID} = (TV'_{ID}, TS'_{ID})$, $\mathcal{C}$ checks $\kappa_0$ to find $(ID, x_{ID}, TV_{ID}, TS_{ID})$, if such tuple exists, it will replace it with $(ID, -1, TV'_{ID}, TS'_{ID})$, where $-1$ means that the public key has been replaced. Otherwise, $\mathcal{C}$ simply adds a tuple $(ID, -1, TV'_{ID}, TS'_{ID})$ to $\kappa_0$.

*Sign query*: $\mathcal{F}_I$ is allowed to query the sign oracle in order to receive valid signatures on any tuple $(m, ID, TV_{ID}, TS_{ID})$, where $m$ is a message to be signed by the signer with identity $ID$ and public key $PK_{ID} = (TV_{ID}, TS_{ID})$. This oracle is able to produce valid signatures even for identities where the public key of the user has been replaced. $\mathcal{C}$ starts by picking a random $r \in \{0, 1\}^l$, and scans $\kappa_2$ for a tuple $(m, r, ID, TV_{ID}, TS_{ID}, \ldots)$. If such tuple already exists in $\kappa_2$, $\mathcal{C}$ picks another $r$ until no tuple $(m, r, ID, TV_{ID}, TS_{ID}, \ldots)$ is found in $\kappa_2$. When a proper $r$ is found, $\mathcal{C}$ picks a random $\beta \in \mathbb{Z}_q$ and inserts $(m, r, ID, TV_{ID}, TS_{ID}, \beta, 0)$ (this implies that a $H_2$ query on $(m, r, ID, TV_{ID}, TS_{ID})$ will be replied by $\beta P$). Lastly, $\mathcal{C}$ computes $\lambda = e(\beta TS_{ID}, Q_{ID})$ and forms the signature as $\sigma = (\lambda, r)$.

*Confirmation/disavowal query*: Upon $\mathcal{F}_I$'s request for a confirmation/disavowal proof transcript on any tuple $(m, \sigma' = (\lambda', r'), TV_{ID_S}, TS_{ID_S}, ID_S, ID_V)$, where $ID_S$ is the identity of a signer with public key $PK_{ID_S} = (TV_{ID_S}, TS_{ID_S})$ and $ID_V$ is the identity of a designated verifier. $\mathcal{C}$ responds in one of the following ways:

(1) If the tuple $(m, r', ID_S, TV_{ID_S}, TS_{ID_S}, \ldots)$ was never queried to $H_2$ oracle, $\mathcal{C}$ proceeds as in sign oracle to generate a valid sub-signature $\lambda$, and checks if $\lambda = \lambda'$, it will return the confirmation protocol transcript, and the disavowal protocol transcript if $\lambda \neq \lambda'$.
(2) If $(m, r', ID_S, TV_{ID_S}, TS_{ID_S}, \beta, Y)$ exists in $\kappa_2$, and $Y = 0$, $\mathcal{C}$ will compute the valid sub-signature as $\lambda = e(\beta TS_{ID_S}, Q_S)$. Similar to above, it will output the confirmation protocol transcript if $\lambda = \lambda'$, and the disavowal protocol transcript if $\lambda \neq \lambda'$.
(3) If $(m, r', ID_S, TV_{ID_S}, TS_{ID_S}, \beta, Y)$ exists in $\kappa_2$ and $Y = 1$, $\mathcal{C}$ scans $\kappa_1$ in order to find a tuple $(ID_S, TV_{ID_S}, TS_{ID_S}, \alpha, X)$. If $X = 1$, $\mathcal{C}$ outputs *failure* and aborts. Otherwise, if $X = 0$, it will form the valid signature as $\lambda = e(\beta(cP), TS_{ID_S})^{\alpha}$ and output the confirmation protocol transcript if $\lambda = \lambda'$, and the disavowal protocol transcript if $\lambda \neq \lambda'$.

$\mathcal{C}$ may fail in the simulation of the non-interactive designated verifier proofs of confirmation/disavowal protocol if a collision occurs in simulating $H_3$ or $H_4$ oracle. The probability for the occurrence of such collision is at most $(q_{H_3} + q_{CD})2^{-k}$, considering $q_{H_3} \approx q_{H_4}$.

At the end of the game, $\mathcal{F}_I$ outputs a tuple $(m^*, \sigma^*, ID^*, TV_{ID^*}, TS_{ID^*})$ where $\sigma^* = (r^*, \lambda^*)$ is a valid signature on message $m^*$ for identity $ID^*$ with public key $PK_{ID^*} = (TV_{ID^*}, TS_{ID^*})$. In order for $\mathcal{F}_I$ to win, $(ID^*, TV_{ID^*}, TS_{ID^*})$ should have never been queried to the partial private key or the private key extraction oracles. Upon $\mathcal{F}_I$'s success, $\mathcal{C}$ searches $\kappa_1$ and $\kappa_2$ to find $(ID^*, \alpha^*, TV_{ID^*}, TS_{ID^*}, X)$ (due to the assumption made above, existence of such tuple is certain in $\kappa_1$) and $(m^*, r^*, ID^*, TV_{ID^*}, TS_{ID^*}, \beta^*, Y)$; if $X = 0$ or $Y = 0$, $\mathcal{C}$ reports *failure* and aborts. Again, if $(m^*, r^*, ID^*, TV_{ID^*}, TS_{ID^*}, \beta^*, Y)$ does not exist in $\kappa_2$, $\mathcal{C}$ will report *failure* and aborts. In the case that the public key $PK_{ID^*} = (TV_{ID^*}, TS_{ID^*})$ was never replaced before, $\mathcal{C}$ can simply extract the user's secret value from $\kappa_0$

and compute $(\lambda^*)^{1/x_{ID^*}\alpha^*\beta^*}$ as the solution of the random instance $(P, aP, bP, cP)$ of the BDH assumption. On the other hand, if the public key $PK_{ID^*} = (TV_{ID^*}, TS_{ID^*})$ has been replaced before, then for $\sigma^*$ to be valid we know that $e(TV_{ID^*}, P_{Pub}) = e(TS_{ID^*}, P)$. Therefore, based on the knowledge of exponent assumption in [9,16], $\mathcal{F}_I$ can extract $x$ since $e(xP, aP) = e(x(aP), P)$, and consequently $\mathcal{C}$ outputs $(\lambda^*)^{1/x\alpha^*\beta^*}$ as the solution of the random instance $(P, aP, bP, cP)$ of the BDH assumption.

In order to compute the success probability of $\mathcal{C}$, we have to consider the cases that $\mathcal{C}$ may fail. $\mathcal{C}$ can fail in either the simulation process or in solving the BDH problem after $\mathcal{F}_I$ outputted the forgery signature. $\mathcal{C}$ will fail in the simulation process if $\mathcal{F}_I$ queries a partial private key extraction on an identity $ID$ and public key $PK_{ID} = (TV_{ID}, TS_{ID})$ where $H_1(ID, TV_{ID}, TS_{ID}) = \alpha(bP)$. $\mathcal{C}$ will also fail in simulating the confirmation/disavowal protocol when $\mathcal{F}_I$ queries a tuple $(m, \sigma, TV_{ID_S}, TS_{ID_S}, ID_S, ID_V)$, where $H_2(m, r, ID_S, TV_{ID_S}, TS_{ID_S}) = \beta(cP)$ and $H_1(ID_S, TS_{ID_S}, TV_{ID_S}) = \alpha(bP)$. Moreover, $\mathcal{C}$ can fail in solving the BDH problem if the forgery tuple $(m^*, \sigma^*, ID^*, TV_{ID^*}, TS_{ID^*})$ is such that $H_1(ID^*, TV_{ID^*}, TS_{ID^*})$ was defined to be $\alpha P$ or $H_2(m^*, r^*, ID^*, TV_{ID^*}, TS_{ID^*})$ was defined to be $\beta P$. Therefore, following Coron's method [8], the probability for $\mathcal{C}$ to avoid all the failure states is $\varphi_1^{q_E}(1-\varphi_1)\,\varphi_2^{q_{CD}}(1-\varphi_2)$ where $q_E$ is the number of partial private key and private key extraction queries and $q_{CD}$ is the number of confirmation/disavowal queries. By optimizing the probabilities $\varphi_1$ and $\varphi_2$, the probability for $\mathcal{C}$ to avoid all the failure states is equal to $1/\mathbf{e}(q_E+1)(q_{CD}+1)$. Where $\mathbf{e}$ is the base of natural logarithm. There is also the probability that $\mathcal{F}_I$ never queried $H_2(m^*, r^*, ID^*, TV_{ID^*}, TS_{ID^*})$, this may occur with probability $2^{-k}$. It is possible that $\mathcal{F}_I$ produced the forgery signature $\sigma^*$ and proved its validity where it did not use the valid private key $S_{ID^*}$, this case may only happen if $H_3(W, N, O, m^*, \sigma^*)$ is set as a particular value. This case may only happen with probability $q_{H_3}2^{-k}$. As mentioned above, $\mathcal{C}$ may also fail in simulating the confirmation and disavowal protocol if a collision occurs in the domain of $q_{H_3}$, this incident may happen with probability $(q_{H_3}+q_{CD})2^{-k}$. Following the proof, $\mathcal{C}$'s success probability is at least $(\epsilon_{\mathcal{F}_I} - (2q_{H_3}+q_{CD}+1)2^{-k})/\mathbf{e}(q_E+1)(q_{CD}+1)$.

## *Proof of Theorem 2*

We prove that if there exists a Type I adversary $\mathcal{D}_I$ which is able to win the game defined in Definition 2 with probability $\epsilon_{\mathcal{D}_I}$, then one can build another algorithm $\mathcal{C}$ which is able to solve a random instance $(P, aP, bP, cP, h)$ of the DBDH problem with probability $\epsilon_{\mathcal{C}}$. $\mathcal{C}$ acts as $\mathcal{D}_I'$s challenger, it starts by initiating the set-up algorithm as in Proof of Theorem 1 wherein, $P_{Pub} = aP$ and $a$ is unknown to $\mathcal{C}$. $\mathcal{D}_I$ starts by querying different oracles as explained in Definition 2. We assume $\mathcal{D}_I$ always makes a public key request before a $H_1$ query, and a $H_1$ query before it requests for the partial private key of the user. Similar to the Proof of Theorem 1, $\mathcal{C}$ answers to $\mathcal{D}_I$ queries by using lists $\kappa_i$ for $i = \{1, 2, 3, 4\}$ and a list $\kappa_0$ in order to keep track of the values of identity, public keys and the corresponding secret value.

*Query on* $H_1(ID, TV_{ID}, TS_{ID})$: Queries to $H_1$ are handled identical to Proof of Theorem 1.

*Query on* $H_2(m, r, ID, TV_{ID}, TS_{ID})$: To answer queries on $H_2$, $\mathcal{C}$ scans $\kappa_2$ to find $(m, r, ID, TV_{ID}, TS_{ID}, \beta, Y)$, If such tuples exists, $\mathcal{C}$ outputs $\beta P$ when $Y = 0$ and $\beta(cP)$ when $Y = 1$. Otherwise, if no such tuple exists in $\kappa_2$, $\mathcal{C}$ picks a random $\beta \in \mathbb{Z}_q$, returns $\beta P$ to $\mathcal{D}_I$, and inserts $(m, r, ID, TV_{ID}, TS_{ID}, \beta, 0)$ into $\kappa_2$.

*Queries on* $H_3$ and $H_4$: Queries to $H_3$ and $H_4$ are handled identical to Proof of Theorem 1.

Queries on public key, partial private key extract, private key, public key replacement, and sign oracles are handled identical to Proof of Theorem 1.

*Confirmation/disavowal query*: Due to the behaviour of $H_2$, $\mathcal{C}$ is able to calculate a valid signature $\sigma$ in order to compare with any signature $\sigma'$ queried to the confirmation/disavowal oracle and generate confirmation (disavowal) proofs consistent with validity (invalidity) of $\sigma'$.

Similar to Proof of Theorem 1, a collision may occur in the domain of $H_3$ or $H_4$ oracles when simulating confirmation/disavowal protocol.

After the first round of queries, $\mathcal{D}_I$ outputs a challenge tuple $(m^*, ID^*, TV_{ID^*}, TS_{ID^*})$, where $m^*$ is a message to be signed, $ID^*$ is the identity of a signer, and $PK_{ID^*} = (TV_{ID^*}, TS_{ID^*})$ is the corresponding public key. If the public key has been replaced before, then, similar to Proof of Theorem 1, and based on the knowledge of exponent assumption [9,16], $\mathcal{D}_I$ should have knowledge on the corresponding secret value and can provide it for $\mathcal{C}$. Otherwise, if $PK_{ID^*}$ is the original public key of the signer, then $\mathcal{C}$ can easily retrieve the value of $x_{ID^*}$ from $\kappa_0$. Note that $(ID^*, TV_{ID^*}, TS_{ID^*})$ was never queried to the partial private key or the private key extraction oracles. $\mathcal{C}$ scans $\kappa_1$ to find $(ID^*, TV_{ID^*}, TS_{ID^*}, \alpha, X)$ (due to the assumption made above, we know that such tuple exists in $\kappa_1$). If $X = 0$, $\mathcal{C}$ aborts and outputs failure. Otherwise, if $X = 1$, $\mathcal{C}$ proceeds by picking a random $r \in \{0,1\}^l$ and checking if $(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*}, \ldots)$ exists $\kappa_2$. If it does, $\mathcal{C}$ picks another $r$ until it finds an appropriate $r$ whereas, no such tuple $(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*}, \ldots)$ exists in $\kappa_2$. Thereupon, $\mathcal{C}$ defines $H_2(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*})$ as $\beta(cP)$ and records $(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*}, \beta, 1)$ in $\kappa_2$. Lastly, $\mathcal{C}$ computes $\lambda^* = h^{x_{ID^*}\alpha\beta}$ and sets the challenge signature as $\sigma^* = (r, \lambda^*)$.

$\mathcal{D}_I$ starts the second round of queries, however, this time, $\mathcal{D}_I$ is withheld from a partial private key or private key extraction query on $(ID^*, TV_{ID^*}, TS_{ID^*})$, sign query on $(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*})$, or confirmation/disavowal query on $(m^*, \sigma^*, ID^*, TV_{ID^*}, TS_{ID^*})$.

After the second round of queries, $\mathcal{D}_I$ outputs its decision bit $b \in \{0, 1\}$. If $b = 0$, it indicates that $\sigma^*$ is an invalid signature, consequently, $\mathcal{C}$ outputs 0 to declare that $(P, aP, bP, cP, h)$ is an invalid DBDH tuple. If $b = 1$, it indicates that $\sigma^*$ is a valid signature and consequently, $\mathcal{C}$ outputs 1 to declare that $(P, aP, bP, cP, h)$ is a valid DBDH tuple.

In order to compute $\mathcal{C}$'s success probability, we first consider the situations that $\mathcal{C}$ might fail. $\mathcal{C}$ may fail in partial private key or private key extraction queries where $H_1(ID, TV_{ID}, TS_{ID})$ was defined to be $\alpha(bP)$. $\mathcal{C}$ may also fail in the challenge phase where the challenge identity $ID^*$ is such that $H_1(ID^*, TV_{ID^*}, TS_{ID^*})$ was defined as $\alpha P$. Consequently,

the probability for $\mathcal{C}$ not to fail is $\varphi_1^{q_E}(1-\varphi_1)$ which is maximized at $1/\mathbf{e}(q_E+1)$, when the optimal value of $\varphi_1$ is used. Similar to Proof of Theorem 1, $\mathcal{C}$ may also fail in simulation of the confirmation and disavowal protocol (in a case of collision) with probability $(q_{H_3}+q_{CD})2^{-k}$. Following the proof, given $\epsilon_{\mathcal{D}_I}$ as the success probability of $\mathcal{D}_I$, $\mathcal{C}$'s success probability is at least $(\epsilon_{\mathcal{D}_I}-(q_{H_3}+q_{CD})2^{-k})/\mathbf{e}(q_E+1)$.

## *Proof of Theorem 3*

We prove that if there exists a Type II adversary $\mathcal{F}_{II}$ that is able to win the game defined in Definition 4 with probability $\epsilon_{\mathcal{F}_{II}}$, then a PPT algorithm $\mathcal{C}$ can be built that runs $\mathcal{F}_{II}$ as its subroutine and is able to solve a random instance $(P, aP, bP, cP)$ of the BDH problem with probability $\epsilon_{\mathcal{C}}$. $\mathcal{C}$ plays as $\mathcal{F}_{II}$'s challenger, and starts by initiating the set-up algorithm and providing $\mathcal{F}_{II}$ with the master secret key $s$ and the system's public parameters $params = (q, \mathbb{G}_1, \mathbb{G}_2, P, H_1, H_2, H_3, H_4)$. Evidently, $P_{Pub}$ is not included in the public parameters as it can be easily computed by $\mathcal{F}_{II}$.

$\mathcal{F}_{II}$ performs polynomially bounded number of queries as defined in Definition 4. As in Theorem 1, $\mathcal{C}$ handles these queries by keeping lists $\kappa_i$ for $i \in \{1,2,3,4\}$, and a list $\kappa_0$ in order to keep track of the values of identity, secret value, and the corresponding public keys of users in the system. We assume $\mathcal{F}_{II}$ makes a public key request before a $H_1$ query.

*Query on* $H_1(ID, TV_{ID}, TS_{ID})$: To answer such queries on $H_1$ for an identity $ID$ with public key $PK_{ID} = (TV_{ID}, TS_{ID})$, $\mathcal{C}$ picks a random $\alpha \in \mathbb{Z}_q$, inserts $(ID, TV_{ID}, TS_{ID}, \alpha)$ in $\kappa_1$, and returns $Q_{ID} = \alpha(bP)$ to $\mathcal{F}_{II}$.

*Query on* $H_2(m, r, ID, TV_{ID}, TS_{ID})$: In order to answer queries on $H_2$, $\mathcal{C}$ first picks a random $\beta \in \mathbb{Z}_q$ and flips a coin $Y$ that is truly random taking the value of 0 with probability $\varphi_2$ and the value of 1 with probability $1-\varphi_2$ (the value of $\varphi_2$ will be computed later in our proof). Next, $\mathcal{C}$ inserts $(m, r, ID, TV_{ID}, TS_{ID}, \beta, Y)$ into $\kappa_2$ and returns $H_2(m, r, ID, TV_{ID}, TS_{ID}) = \beta(cP)$ if $Y = 1$ and $H_2(m, r, ID, TV_{ID}, TS_{ID}) = \beta P$ if $Y = 0$.

*Query on* $H_3$ and $H_4$: Queries on $H_3$ and $H_4$ will be handled by $\mathcal{C}$ in a random manner and the outputs will be stored in $\kappa_3$ and $\kappa_4$, respectively.

*Public key request*: Upon submitting an identity $ID$, $\mathcal{C}$ picks a random $\delta \in \mathbb{Z}_q$ and flips a coin $X$ that is truly random taking the value of 0 with probability $\varphi_1$ and the value of 1 with probability $1-\varphi_1$ (the value of $\varphi_1$ will be computed later in our proof). If $X = 0$, $\mathcal{C}$ sets the public key as $PK_{ID} = (TV_{ID} = \delta P, TS_{ID} = \delta P_{Pub})$. Otherwise, if $X = 1$, $\mathcal{C}$ sets the public key as $PK_{ID} = (TV_{ID} = \delta(aP), TS_{ID} = s\delta(aP))$. In both cases, $\mathcal{C}$ inserts the tuple $(ID, \delta, TV_{ID}, TS_{ID}, X)$ in $\kappa_0$.

*Secret value extraction*: In order to respond to a secret key extraction query on an identity $ID$ with public key $PK_{ID} = (TV_{ID}, TS_{ID})$, $\mathcal{C}$ scans $\kappa_0$ for $(ID, \delta, TV_{ID}, TS_{ID}, X)$. If $X = 1$, $\mathcal{C}$ reports *failure* and aborts the simulation. Otherwise, it returns $\delta$ as the secret value of the user.

*Private key extraction*: In order to respond to a private key extraction query on identity $ID$ with public key $PK_{ID} = (TV_{ID}, TS_{ID})$, $\mathcal{C}$ scans $\kappa_0$ for $(ID, \delta, TV_{ID}, TS_{ID}, X)$. If $X = 1$, $\mathcal{C}$ reports *failure* and aborts the simulation. Otherwise, it searches $\kappa_1$ to find $(ID, TV_{ID}, TS_{ID}, \alpha)$ and returns the private key of the user as $S_{ID} = s\delta Q_{ID}$.

*Public key replacement*: If $\mathcal{F}_{II}$ wishes to replace the public key $PK_{ID} = (TV_{ID}, TS_{ID})$ for identity $ID$ with public key of its choice $PK'_{ID} = (TV'_{ID}, TS'_{ID})$, $\mathcal{C}$ checks $\kappa_0$ to find $(ID, x_{ID}, TV_{ID}, TS_{ID}, \ldots)$, if such tuple exists, it will replace it with $(ID, -1, TV'_{ID}, TS'_{ID}, \ldots)$, where $-1$ means that the public keys have been replaced. Otherwise, $\mathcal{C}$ adds a tuple $(ID, -1, TV'_{ID}, TS'_{ID}, \ldots)$ to $\kappa_0$.

*Sign query*: In order to respond to a signature query on any tuple $(m, ID, TV_{ID}, TS_{ID})$, $\mathcal{C}$ picks a random $r \in \{0,1\}^l$ and checks if $\kappa_2$ already contains $(m, r, ID, TV_{ID}, TS_{ID}, \ldots)$, if yes, it proceeds until it finds an appropriate $r$ where no such tuple $(m, r, ID, TV_{ID}, TS_{ID}, \ldots)$ exists in $\kappa_2$. When such an acceptable $r$ is found, $\mathcal{C}$ picks a random $\beta \in \mathbb{Z}_q$ and inserts $(m, r, ID, TV_{ID}, TS_{ID}, \beta, 0)$ in $\kappa_2$, implying that the value of $H_2(m, r, ID, TV_{ID}, TS_{ID})$ is set as $\beta P$. Lastly, $\mathcal{C}$ computes $\lambda = e(\beta TS_{ID}, Q_{ID})$ and forms the signature $\sigma = (r, \lambda)$.

*Confirmation/disavowal query*: Upon $\mathcal{F}_{II}$'s request for a confirmation/disavowal proof transcript on any tuple $(m, \sigma' = (r, \lambda'), TV_{ID_S}, TS_{ID_S}, ID_S, ID_V)$, where $ID_S$ is the identity of a signer with public key $PK_{ID_S} = (TV_{ID_S}, TS_{ID_S})$ and $ID_V$ is the identity of a designated verifier. $\mathcal{C}$ responds in one of the following ways:

(1) If the tuple $(m, r, ID_S, TV_{ID_S}, TS_{ID_S}, \ldots)$ was never queried to $H_2$ oracle, $\mathcal{C}$ proceeds as in sign oracle to generate a valid sub-signature $\lambda$, and checks if $\lambda = \lambda'$, it will return the confirmation protocol transcript, and the disavowal protocol transcript if $\lambda \neq \lambda'$.
(2) If $(m, r, ID_S, TV_{ID_S}, TS_{ID_S}, \beta, Y)$ exists in $\kappa_2$ and $Y = 0$, $\mathcal{C}$ will compute the valid sub-signature as $\lambda = e(\beta TS_{ID_S}, Q_S)$. Similar to above, it will output the confirmation protocol transcript if $\lambda = \lambda'$, and the disavowal protocol transcript if $\lambda \neq \lambda'$.
(3) If $(m, r, ID_S, TV_{ID_S}, TS_{ID_S}, \beta, Y)$ exists in $\kappa_2$ and $Y = 1$, $\mathcal{C}$ scans $\kappa_0$ in order to find a tuple $(ID_S, \delta, TV_{ID_S}, TS_{ID_S}, X)$. If $X = 1$, $\mathcal{C}$ outputs failure and aborts. Otherwise, if $X = 0$, it will form the valid sub-signature as $\lambda = e(\beta(cP), sQ_S)^{\delta}$ and output the confirmation protocol transcript if $\lambda = \lambda'$, and the disavowal protocol transcript if $\lambda \neq \lambda'$.

$\mathcal{C}$ may fail in the simulation of the non-interactive designated verifier proofs of confirmation/disavowal protocols if a collision occurs in simulating $H_3$ or $H_4$ oracle. The probability for the occurrence of such collision is at most $(q_{H_3} + q_{CD})2^{-k}$, considering $q_{H_3} \approx q_{H_4}$.

At the end of the game, $\mathcal{F}_{II}$ outputs a tuple $(m^*, \sigma^*, ID^*, TV_{ID^*}, TS_{ID^*})$, whereas $\sigma^* = (r^*, \lambda^*)$ is a valid signature on message $m^*$ for a signer with identity $ID^*$ and public key $PK_{ID^*} = (TV_{ID^*}, TS_{ID^*})$. In order for $\mathcal{F}_{II}$ to win, $(ID^*, TV_{ID^*}, TS_{ID^*})$ should have not been queried to secret value extraction, public key replacement or private key extraction oracles. Upon $\mathcal{F}_{II}$'s success, $\mathcal{C}$ scans $\kappa_0$ and $\kappa_2$ in order to find tuples $(ID^*, \delta^*, TV_{ID^*}, TS_{ID^*}, X)$ and $(m^*, r^*, ID^*, TV_{ID^*}, TS_{ID^*}, \beta^*, Y)$, respectively. Then if $X = 0$ or $Y = 0$, $\mathcal{C}$ outputs failure and aborts. Also if no such

tuple $(m^*, r^*, ID^*, TV_{ID^*}, TS_{ID^*}, \beta^*, Y)$ exists in $\kappa_2$, $\mathcal{C}$ will output failure and aborts the simulation. Otherwise, if $\sigma^*$ is a valid signature, $\mathcal{C}$ outputs $(\lambda^*)^{\frac{1}{s\alpha\beta\delta}}$ as the solution of the random instance $(P, aP, bP, cP)$ of the BDH problem.

In order to compute the success probability of $\mathcal{C}$, we have to consider the cases that $\mathcal{C}$ may fail. $\mathcal{C}$ can fail in either the simulation process or in solving the BDH problem after $\mathcal{F}_{II}$ outputted the forgery signature. $\mathcal{C}$ will fail in the simulation process if $\mathcal{F}_{II}$ initiates a secret value or private key extraction query on an identity $ID$ where $TV_{ID} = \delta(aP)$ and $TS_{ID} = s\delta(aP)$. $\mathcal{C}$ will also fail in simulating the confirmation/disavowal protocol when $\mathcal{F}_{II}$ queries a tuple $(m, \sigma, TV_{ID_S}, TS_{ID_S}, ID_S, ID_V)$, where $H_2(m, r, ID, TV_{ID_S}, TS_{ID_S}) = \beta(cP)$ and $TV_{ID_S} = \delta(aP)$ and $TS_{ID_S} = s\delta(aP)$. $\mathcal{C}$ will also fail if the forgery tuple $(m^*, \sigma^*, ID^*, TV_{ID^*}, TS_{ID^*})$ is such that $TV_{ID^*}$ and $TS_{ID^*}$ were defined to be $\delta P$ and $\delta P_{Pub}$, respectively. Besides, $\mathcal{C}$ will again fail, if the forgery tuple is such that $H_2(m^*, r^*, ID^*, TV_{ID^*}, TS_{ID^*})$ was defined as $\beta P$. The probability for $\mathcal{C}$ to avoid all the failure states is $\varphi_1^{q_E}(1-\varphi_1)\,\varphi_2^{q_{CD}}(1-\varphi_2)$, whereas $q_E$ is the number of secret value and private key extraction queries and $q_{CD}$ is the number of confirmation/disavowal queries. By maximizing the probabilities $\varphi_1$ and $\varphi_2$, the success probability of $\mathcal{C}$ would be $1/\mathbf{e}(q_E+1)(q_{CD}+1)$. Similar to the proof of Theorem 1, considering the probability that $(m^*, r^*, ID^*, TV_{ID^*}, TS_{ID^*})$ was never queried to $H_2$ oracle, the probability that $\mathcal{F}_{II}$ did not use the valid private key $S_{ID^*}$ when generating the forgery signature $\sigma^*$, and the probability of failure in simulating the confirmation/disavowal protocol; $\mathcal{C}$ success probability is at least $(\epsilon_{\mathcal{F}_{II}} - (2q_{H_3} + q_{CD} + 1)2^{-k})/\mathbf{e}(q_E+1)(q_{CD}+1)$.

## *Proof of Theorem 4*

We prove that if there exists a Type II adversary $\mathcal{D}_{II}$ which is able to win the game defined in Definition 5 with probability $\epsilon_{\mathcal{D}_{II}}$, then one can build another PPT algorithm $\mathcal{C}$ which is able to solve a random instance $(P, aP, bP, cP, h)$ of the DBDH problem with probability $\epsilon_{\mathcal{C}}$. $\mathcal{C}$ acts as $\mathcal{D}_{II}$'s challenger, it starts by initiating the setup algorithm as in Proof of Theorem 3. $\mathcal{D}_{II}$ starts by querying different oracles as explained in Definition 5. We assume $\mathcal{D}_{II}$ makes a public key request before a $H_1$ query. Similar to Proof of Theorem 3, $\mathcal{C}$ answers to $\mathcal{D}_{II}$ queries by using lists $\kappa_i$ for $i = \{1, 2, 3, 4\}$ and a list $\kappa_0$ in order to keep track of the values of identity, secret value, and the corresponding public keys of users in the system.

*Query on* $H_1(ID, TV_{ID}, TS_{ID})$: Queries to $H_1$ are handled identical to Proof of Theorem 3.

*Query on* $H_2(m, r, ID, TV_{ID}, TS_{ID})$: In order to answer queries on $H_2$, $\mathcal{C}$ scans $\kappa_2$ to find $(m, r, ID, TV_{ID}, TS_{ID}, \beta, Y)$, If such tuples exists, $\mathcal{C}$ outputs $\beta P$ when $Y = 0$, and $\beta(cP)$ when $Y = 1$. Otherwise, if no such tuple exists in $\kappa_2$, $\mathcal{C}$ first picks a random $\beta \in \mathbb{Z}_q$, returns $\beta P$ to $\mathcal{D}_{II}$, and inserts $(m, r, ID, TV_{ID}, TS_{ID}, \beta, 0)$ into $\kappa_2$.

*Queries on* $H_3$ and $H_4$: Queries to $H_3$ and $H_4$ are handled identical to Proof of Theorem 3.

Queries on public key, secret value extraction, private key extraction, public key replacement, and sign oracles are handled identical to Proof of Theorem 3.

*Confirmation/disavowal query*: Due to the behaviour of $H_2$ oracle, $\mathcal{C}$ is able to calculate valid signature $\sigma$ in order to compare with any signature $\sigma'$ queried to the confirmation/disavowal oracle and generate confirmation (disavowal) proofs consistent with validity (invalidity) of $\sigma'$.

Similar to Proof of Theorem 3, a collision may occur in the domain of $H_3$ or $H_4$ oracle when simulating confirmation/disavowal protocol.

After the first round of queries, $\mathcal{D}_{II}$ outputs a challenge tuple $(m^*, ID^*, TV_{ID^*}, TS_{ID^*})$, where $m^*$ is a message to be signed, $ID^*$ is the identity of a signer, and $PK_{ID^*} = (TV_{ID^*}, TS_{ID^*})$ is the corresponding public key. Note that $(ID^*, TV_{ID^*}, TS_{ID^*})$ should have never been queried to the secret value extraction, public key replacement or the private key extraction oracles. $\mathcal{C}$ scans $\kappa_0$ to find $(ID^*, \delta, TV_{ID^*}, TS_{ID^*}, X)$. If $X = 0$, $\mathcal{C}$ aborts and outputs failure. Otherwise, if $X = 1$, $\mathcal{C}$ proceeds by picking a random $r \in \{0,1\}^l$ and checking if $(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*}, \ldots)$ exists $\kappa_2$. If it does, $\mathcal{C}$ picks another $r$ until it finds an appropriate $r$ whereas, no such tuple $(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*}, \ldots)$ exists in $\kappa_2$. Thereupon, $\mathcal{C}$ defines $H_2(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*})$ as $\beta(cP)$ and records $(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*}, \beta, 1)$ in $\kappa_2$. Lastly, $\mathcal{C}$ computes $\lambda^* = h^{s\delta\alpha\beta}$ and sets the challenge signature as $\sigma^* = (r, \lambda^*)$.

$\mathcal{D}_{II}$ starts the second round of queries, however, this time $\mathcal{D}_{II}$ is withheld from a secret value extraction, public key replacement or private key extraction query on $(ID^*, TV_{ID^*}, TS_{ID^*})$, sign query on $(m^*, r, ID^*, TV_{ID^*}, TS_{ID^*})$, or confirmation/disavowal query on $(m^*, \sigma^*, ID^*, TV_{ID^*}, TS_{ID^*})$.

After the second round of queries, $\mathcal{D}_{II}$ outputs its decision bit $b \in \{0,1\}$. If $b = 0$, indicates that $\sigma^*$ is an invalid signature, consequently, $\mathcal{C}$ outputs 0 to declare that $(P, aP, bP, cP, h)$ is an invalid DBDH tuple. If $b = 1$, it indicates that $\sigma^*$ is a valid signature and consequently, $\mathcal{C}$ outputs 1 to declare that $(P, aP, bP, cP, h)$ is a valid DBDH tuple.

In order to assess $\mathcal{C}$'s success probability, we first consider the situations that $\mathcal{C}$ might fail. $\mathcal{C}$ may fail if $\mathcal{D}_{II}$ initiate a secret value or private key extraction query on an identity $ID$ where $TV_{ID} = \delta(aP)$ and $TS_{ID} = s\delta(aP)$. $\mathcal{C}$ might also fail if the challenge identity $ID^*$ is such that the public key is set as $PK_{ID^*} = (TV_{ID^*} = \beta P, TS_{ID^*} = \beta P_{Pub})$. Consequently, the probability for $\mathcal{C}$ not to fail is $\varphi_1^{q_E}(1-\varphi_1)$ which is maximized at $1/\mathbf{e}(q_E+1)$ when the optimal value of $\varphi_1$ is used. Similar to Proof of Theorem 3, $\mathcal{C}$ may also fail in simulation of the confirmation and disavowal protocols with probability $(q_{H_3} + q_{CD})2^{-k}$. Following the proof, given $\epsilon_{\mathcal{D}_{II}}$ as the success probability of $\mathcal{D}_{II}$, $\mathcal{C}$'s success probability is at least $(\epsilon_{\mathcal{D}_{II}} - (q_{H_3} + q_{CD})2^{-k})/\mathbf{e}(q_E+1)$.