Constant Size Traceable Ring Signature Scheme without Random Oracles

Ke Gu and Na Wu
[†] School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, China
Email: gk4572@163.com

Abstract

Ring signature is a useful cryptographic primitive for many anonymous applications, which makes a signer sign any message anonymously. Namely ring signature allows that ring member signs any message without revealing his/her specific identity. Also, compared with group signature, ring signature provides more flexibility: no group manager, no group setup, and the dynamics of group (ring) update. However, ring signature may make the irresponsible signers abuse their signing rights if there are no measures of keeping them from abusing signing rights. Thus, practical ring signature must be able to trace or reveal the identity of the signer by the signature when the result of the signature needs to be arbitrated. Traceable ring signature is a ring signature that restricts abusing anonymity. Traceable ring signature has a tag that consists of a list of ring members and an event that refers to. In this model, if a ring member generates two linkable ring signatures in the same event, the identity of the ring member is immediately revealed.

Currently several traceable (or linkable) identity-based ring signature schemes have been proposed. However, most of them are constructed in the random oracle model. In this paper, we present a fully traceable ring signature (TRS) scheme without random oracles, which has the constant size signature and a security reduction to the computational Diffie-Hellman (CDH) assumption. Also, we give a formal security model for traceable ring signature and prove that the proposed scheme has the properties of traceability and anonymity.

Index Terms

ring signature, traceability, constant size signature, security model

I. INTRODUCTION

A. Background

Ring signature [1,2,3,4,5,6,7] allows ring member to hide his identifying information to a ring when ring member signs any message, thus ring signature only reveals the fact that a message was signed by possible one of ring members (a list of possible signers). Ring signature is also called as a special group signature [8]. However, compared with group signature, ring signature has more advantages: the group (ring) must not be constructed by a group manager, who can revoke the anonymity of any signer or identify the real group signer; additionally, because a list of possible signers must be constructed to form a group, some intricate problems need to be solved in a group signature scheme, such as joining the new members and the revocation of group members. Although ring signature can provide more flexibility and full anonymity, it is vulnerable to keep the signers from abusing their signing rights. Namely, it is infeasible for the verifier to determine whether the signatures are generated by the same signer on the same event. Thus, in a practical ring signature scheme, the third trusted authority or the verifier must be able to know who signs the messages on the same event many times and the verifier can not accept the signatures generated by the same signer on the same event [9,10,11,12,13,14].

Traceable ring signature[1] [18] is a ring signature that restricts abusing anonymity. Unlike group signature has too strong a traceability characteristic and ring signature has too strong an anonymity characteristic, traceable ring signature has the balance characteristic of anonymity and traceability. Namely, traceable ring signature provides restricted anonymity and traceability. In a traceable ring signature scheme, traceable ring signature can provide full anonymity for the responsible or honest signer when the singer signs any message, and provide traceability for the verifier (or the third trusted authority) to determine whether the signatures are generated by the same signer on the same event when the irresponsible signer abuses anonymity in some applications. In order to achieve this requirement of traceable ring signature, we need to consider the two notions ”one-more unforgeability” and ”double-spending traceability” [18,19,20] in the context of ring signature, which originate from blind signature. First, any user can not generate a ”one-more” new signature after he obtained a signature from the original signer. Second, if an irresponsible user signs any message twice on the same event, the signatures generated by the user can be traced to reveal the identity of the signer [21,22]. In the second notion, a responsible user can be anonymously protected. Obviously, traceable ring signature can provide more practicality because of its restricted anonymity in many no full anonymous applications.

[1]This notion is closely related to linkable ring signature in [12,15,16,17].

Currently ring signatures are used in many different applications, such as whistle blowing [1], anonymous authentication for ad-hoc network [12], e-voting [23] and e-cash [24], non-interactive deniable authentication [25] and multi designated verifiers signature [26], etc. Because ring signature is not linkable, no one can determine whether two ring signatures are generated by the same signer. Thus, it exists high risk that ring signatures are used in e-voting and e-cash. For example, if a user signs a message twice for double votes in anonymous e-voting, no one can find the two signatures are linkable so as to detect the irregularity. Obviously, traceable ring signature is suitable for the kind of applications, because it can find the two signatures are linkable. There also are other applications for traceable ring signature. In the "off-line" anonymous e-cash systems, a user is permitted to anonymously signs a message once during one cash transaction, thus traceable ring signature is a natural choice for this application [18]. Damgard et al. [27] proposed an unclonable group identification without the group manager, traceable ring signature is also suitable for this application because of not employing the group manager and its balance of anonymity and traceability.

B. Our Contributions

In this paper, we present a traceable ring signature scheme without random oracles. Also, we give the formal security model for traceable ring signature. Under our security model, the proposed scheme is proved to have the properties of anonymity and traceability with enough security. In this paper, our contributions are as follows:

- We present a fully traceable ring signature scheme without random oracles, which has the constant size signature. No poly-time adversary can produce a valid TRS signature on any messages when the adversary may adaptively be permitted to choose messages after executing signature oracle.
- We present a framework for TRS and show a detailed security model for TRS. Compared with the security models of TRS [15,18], we integrate the Fujisaki et al.'s frame and the Au et al.'s frame to our security model. In our security model, we consider four situations for the security of TRS and further strengthen our security model on public key cryptography, where we still consider the trusted authority is fully trusted[2]. Under our security model, the proposed TRS scheme is proved to be secure without random oracles, and has a security reduction to the simple standard assumption (computational Diffie-Hellman assumption).

[2]In the Au et al.'s frame [15], they consider the PKG system is partially trusted.

- Compared with other traceable (or linkable) ring signature schemes proposed by [15,16,18, 28,29], the proposed TRS scheme has some advantages (the comparisons of the schemes are given in Appendix A).

C. *Outline*

The rest of this paper is organized as follows. In Section 2, we discuss the related work about TRS. In Section 3, we review the bilinear pairings and complexity assumptions on which we build. In Section 4, we show a framework for TRS. In Section 5, we set up the security model for TRS. In Section 6, we propose a traceable ring signature scheme without random oracles under our framework for TRS. In Section 7, we analyze the correctness, efficiency and security of the proposed scheme. Finally, we draw our conclusions in Section 8.

II. Related Work

Liu et al. [12] first proposed the notion of linkable ring signature. In their scheme, if an irresponsible user anonymously signs any message twice on the same event, the two signatures generated by the user can be linked. Base on this notion, some similar schemes were proposed in [12,16,17,24,30,31]. In [12,16], the proposed schemes cannot resist the attack that an irresponsible signer forges the signature of a honest signer so as to make the honest signer accused of ”double-signing”. In [30,31], the proposed schemes overcome this weakness, but the security conditions are more complicated. In [24], Tsang et al. proposed a short linkable ring signature scheme, which is based on the group identification scheme from [32]. Their scheme provides weak traceability, namely it can only detect the linkable ring signatures. In [30], Tsang et al. proposed a separable linkable threshold ring signature scheme, where the threshold setting is to restrict abusing signing. However, their scheme is complicated. In [33], Liu et al. proposed a revocable ring signature scheme, which supports that any ring member may revoke the anonymity of the real signer when the ring signature is proved to be argumentative. Their scheme provides that all the ring members can reveal the identity of the real signer of any ring signature generated on behalf of their ring.

In 2007 and 2011, Fujisaki et al. [18,29] proposed two traceable ring signature schemes based on public key cryptography, and a security model of traceable ring signature was formally proposed. In their scheme, if two signatures are linked, the identity of this signer will be revealed. In other words, the anonymity of the signer will be revoked if and only if the signer generates two ring signatures on the same event. Compared with revocable ring signature [33], traceable ring signature needs the condition of revoking anonymity that the same signer generates two ring signatures on the same event. Although the scheme proposed by [29] is constructed without random oracles, the signature size of the scheme is sub-linear with $O(\sqrt{n})$, where n is the number of users in the ring. In 2013, Yuen et al. [50] proposed an efficient linkable and/or threshold ring signature scheme without random oracles. However, the signature size of their scheme is still sub-linear with $O(d\cdot\sqrt{n})$, where d is the threshold value and n is the number of users in the ring. In 2014, Liu et al. [51] proposed a linkable ring signature scheme with unconditional anonymity. However, their scheme is still constructed in the random oracle model.

With the rapid development of identity-based cryptography [34,35,36,37], many researchers proposed many identity-based signature (IBS) schemes in the random oracle model or standard model [36,38,39,40]. Also, with these identity-based signature schemes, a lot of variants, such as the identity-based proxy signature schemes [41,42,43], the identity-based ring signature schemes [15,43,44,46], the identity-based group signature schemes [47,48], etc, have also been proposed. In 2006, Au et al. [46] proposed a constant size identity-based linkable and revocable-iff-linked ring signature. However, their scheme was later proved to be insecure [49]. In 2012, Au et al. [15] proposed a new identity-based event-oriented linkable ring signature scheme with an option as revocable-iff-linked. With this option, if a user generates two linkable ring signatures in the same event, everyone can compute his identity from these two signatures. However, their scheme is constructed in the random oracle model.

III. Preliminaries

A. Bilinear Maps

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

B. Computational Diffie-Hellman Assumption

Definition 3.1 Computational Diffie-Hellman (CDH) Problem: *Let $\mathbb{G}_1$ be a group of prime order q and g be a generator of $\mathbb{G}_1$; for all $(g, g^a, g^b) \in \mathbb{G}_1$, with $a, b \in \mathbb{Z}_q$, the CDH problem is to compute $g^{a\cdot b}$.*
Definition 3.2 *The $(\hbar, \varepsilon)$-CDH assumption holds if no $\hbar$-time algorithm can solve the CDH problem with probability at least ε.*

IV. A Framework for TRS

Definition 4.1 Traceable Ring Signature Scheme: *Let* **TRS**=***(System-Setup*, *Generate-Key*, *Sign*, *Verify*, *Trace-User)*** *be a traceable ring signature scheme. In* **TRS**, *all algorithms are described as follows:*

1) ***System-Setup**: The randomized algorithm run by the trusted authority inputs a security parameter 1^k, and then outputs all system parameters TRK and a ring key spk on the security parameter 1^k.*
2) ***Generate-Key**: The randomized algorithm run by a ring member inputs TRK, and then the following steps are finished:*
   - *The algorithm run by the trusted authority outputs a user's partial private key psk_i to the ring member according to the ring key spk, where $i \in \{1, 2......n\}$ (n is the number of the ring members in the ring).*
   - *The algorithm run by the ring member outputs the corresponding signing (private) key sk_i according to psk_i.*
   - *The algorithm run by the ring member outputs and publishes the corresponding public key pk_i.*
3) ***Sign**: The randomized algorithm is a standard traceable ring signature algorithm. A ring member needs to sign a message $\mathfrak{M} \in \{0,1\}^*$ on an event identifier $\mathfrak{E} \in \{0,1\}^*$. The algorithm run by the ring member inputs (TRK, sk_i, RL_PK, $\mathfrak{M}$, $\mathfrak{E}$), and then outputs a signature σ, where RL_PK is a public key list including all public keys of the ring members belong to this ring, $\sigma \in \{0,1\}^* \cup \{\perp\}$, sk_i is the private key of the ring member with $i \in \{1, 2......n\}$.*
4) ***Verify**: The signature verifiers verify a standard traceable ring signature σ. The deterministic algorithm run by a signature verifier inputs (TRK, RL_PK, $\mathfrak{M}$, $\mathfrak{E}$, σ), and then outputs the boolean value, $accept$ or $reject$.*
5) ***Trace-User**: The trusted authority traces a real ring member (signer) by two traceable ring signatures σ_1 on $\mathfrak{M}_1$ and σ_2 on $\mathfrak{M}_2$. The deterministic algorithm run by the trusted authority inputs (TRK, RL_PK, $\{\mathfrak{M}_1, \sigma_1\}$, $\{\mathfrak{M}_2, \sigma_2\}$, $\mathfrak{E}$), and then outputs one of the following results: "the public key pk of the real signer", or "$Independent$" or "$Linked$", where $pk \in RL\_PK$.*

The *correctness* of **TRS** requires that for any $(TRK, spk) \leftarrow$***System-Setup***(1^k), $(sk_i, pk_i) \leftarrow$ ***Generate-key***(TRK) for all i with $i \in \{1, 2......n\}$, $\mathfrak{M} \in \{0,1\}^*$ and $\mathfrak{E} \in \{0,1\}^*$, then:

$$\textbf{Pr}[\textbf{\textit{Verify}}(TRK,\ RL\_PK,\ \mathfrak{M},\ \mathfrak{E},\ \textbf{\textit{Sign}}(TRK,\ sk_i,\ RL\_PK,\ \mathfrak{M},\ \mathfrak{E}))=1]=1.$$

The *traceability* of **TRS** requires that for any $(TRK, spk) \leftarrow$***System-Setup***(1^k), $(sk_i, pk_i) \leftarrow$ ***Generate-key***(TRK) for all i with $i \in \{1, 2......n\}$, $\mathfrak{M}_1, \mathfrak{M}_2 \in \{0,1\}^*$ and $\mathfrak{E} \in \{0,1\}^*$, if $\sigma_1 \leftarrow$***Sign***$(TRK$, sk_i, RL_PK, $\mathfrak{M}_1$, $\mathfrak{E})$ with $i \in \{1, 2......n\}$ and $\sigma_2 \leftarrow$***Sign***$(TRK$, sk_j, RL_PK, $\mathfrak{M}_2$, $\mathfrak{E})$ with $j \in \{1, 2......n\}$, then:

Trace-User(TRK, RL_PK, $\{\mathfrak{M}_1, \sigma_1\}$, $\{\mathfrak{M}_2, \sigma_2\}$, $\mathfrak{E}$)=	"$Independent$"; if $i \neq j$ "$Linked$"; else if $\mathfrak{M}_1 = \mathfrak{M}_2$ "the public key pk of the real signer"; otherwise.

V. Security Model

In a secure TRS scheme, we need to consider the two notions "one-more unforgeability" and "double-spending traceability". First, any user cannot forge a new signature. Second, the anonymity of the signer will be revoked if and only if the signer generates two ring signatures on the same event. Thus, we consider that a fully secure TRS scheme must meet the following security requirements according to [15,18]:

1) **Unforgeability**: A valid TRS signature must be signed by a valid ring member (signer). Therefore, no poly-time adversary can produce a valid TRS signature on any messages when the adversary may adaptively be permitted to choose messages after executing signature oracle. Then we split the requirement to the following two small security notions[3]:
   a) the first one is called security against *linkability attacks*[4], which requires that every two signatures generated by the same signer with respect to the same tag of event are linked − namely the total number of signatures with respect to the same tag cannot exceed the total number of ring members in the tag if every any two signatures are not linked.
   b) the second one is called security against *exculpability attacks*, which requires that an honest ring member cannot be accused of signing twice with respect to the same tag − namely an adversary cannot produce a traceable ring signature such that, along with one signature generated by the target member (the attacked member), it can designate the target member in the presence of the public traceable method. This should be infeasible even after the adversary has corrupted all ring members but the target member.

[3]The two security notions should be more detailedly expanded from the correctness of unforgeability.

[4]In this paper, we also call the notion **Traceability**: anyone who creates two signatures for different messages with respect to the same tag of event can be traced, where the trace can be done only with pairs of message/signature pairs and tag.

2) **Anonymity**: As long as a signer does not sign two different messages with respect to the same tag, the identity of the signer is indistinguishable from any of the possible ring members. In addition, any two signatures generated with respect to two distinct tags are always unlinkable, namely it is infeasible for anyone to determine whether they are generated by the same signer.

Then, based on the above situations, we propose a complete security model for traceable ring signature. To make our security model easier to understand, we construct several algorithms interacting with adversary, which may make attack experiments to the traceable ring signature schemes in the above situations. In our security model, we maximize adversary's advantage, and assume that all attacking conditions needed by adversary hold and adversary may forge signatures after limitedly querying oracles.

In our security model, we assume there are $n+1$ users in a traceable ring signature scheme ($n \in \mathbb{N}$ is a maximal number of ring members in a ring), and at least one user u^* of $n+1$ users is not corrupted by adversary.

Definition 5.1 Unforgeability of A Traceable Ring Signature Scheme: *Let* **TRS**=(***System-Setup***, ***Generate-Key***, ***Sign***, ***Verify***, ***Trace-User***) *be a traceable ring signature scheme. Additionally, we set that k is a secure parameter, and* ***Pr***($\mathcal{B}_{U\_TRS}(k,\mathcal{A})=1$) *is the probability that the algorithm $\mathcal{B}_{U\_TRS}$ returns 1. Then the advantage that the adversary $\mathcal{A}$ breaks* **TRS** *is defined as follows:*

$$\mathbf{Adv}_{TRS}^{u\_trs-uf}(k, q_g, q_s, \hbar)=\mathbf{Pr}(\mathcal{B}_{U\_TRS}(k,\mathcal{A})=1),$$

where q_g is the maximal number of "Generate-Public Key" oracle queries, q_s is the maximal number of "Sign" oracle queries and $\hbar$ is the running time of $\mathcal{B}$. If the advantage that the adversary breaks **TRS** *is negligible, then the scheme* **TRS** *is secure.*

According to the Definition 5.1, the algorithm $\mathcal{B}_{U\_TRS}$ is described as follows:

1.**Setup**: Running ***System-Setup***, (TRK, spk) ←***System-Setup***(1^k), and then TRK is passed to $\mathcal{A}$.

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

- ***Generate-Public Key***(): Given the public parameters TRK, the oracle outputs the public key pk with respect to the corresponding private key.
- ***Sign***(): Given the public parameters TRK, the public key list RL_PK, the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the oracle returns a signature σ to $\mathcal{A}$, where $\sigma \in \{0,1\}^* \cup \{\perp\}$.

3.**Forgery**: $\mathcal{A}$ outputs its forgery, ($\mathfrak{M}^*$, $\mathfrak{E}^*$, σ^*) for RL_PK^*, where RL_PK^* is a public key list including all public keys of the ring members belong to this ring. It succeeds if

(a) 1 ←***Verify***(TRK, RL_PK^*, $\mathfrak{M}^*$, $\mathfrak{E}^*$, σ^*);

(b) $\mathcal{A}$ did not query ***Sign*** on inputs RL_PK^*, $\mathfrak{M}^*$ and $\mathfrak{E}^*$.

The following two definitions are expanded from unforgeability (see [15,18] for more details).

An adversary cannot generate two signatures in the same event without being linked. We generalize the notion that an adversary with t user (ring member) private keys cannot create $t+1$ signatures in the same event without being linked.

Definition 5.2 Linkability (or Traceability) of A Traceable Ring Signature Scheme: *Let* **TRS**= ***(System-Setup**, **Generate-Key**, **Sign**, **Verify**, **Trace-User**) be a traceable ring signature scheme. Additionally, we set that k is a secure parameter, and **Pr**($\mathcal{B}_{L\_TRS}$(k,$\mathcal{A}$)=1) is the probability that the algorithm $\mathcal{B}_{L\_TRS}$ returns 1. Then the advantage that the adversary $\mathcal{A}$ breaks* **TRS** *is defined as follows:*

$$\mathbf{Adv}_{TRS}^{l\_trs-uf}(k, q_g, q_s, t, \hbar)=\mathbf{Pr}(\mathcal{B}_{L\_TRS}(k,\mathcal{A})=1),$$

where q_g is the maximal number of "Generate-Public Key" oracle queries, q_s is the maximal number of "Sign" oracle queries, t is the number of user (ring member) private keys possessed by $\mathcal{A}$ and $\hbar$ is the running time of $\mathcal{B}$. If the advantage that the adversary breaks **TRS** *is negligible, then the scheme* **TRS** *is secure.*

According to the Definition 5.2, the algorithm $\mathcal{B}_{L\_TRS}$ is described as follows:

1.**Setup**: Running ***System-Setup***, $(TRK, spk) \leftarrow$***System-Setup***(1^k), and then TRK is passed to $\mathcal{A}$.

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

- ***Generate-Public Key***(): Given the public parameters TRK, the oracle outputs the public key pk with respect to the corresponding private key.
- ***Sign***(): Given the public parameters TRK, the public key list RL_PK, the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the oracle returns a signature σ to $\mathcal{A}$, where $\sigma \in \{0,1\}^* \cup \{\perp\}$.

3.**Forgery**: $\mathcal{A}$ outputs its forgery, a set of tuples ($\mathfrak{M}_i^*$, $\mathfrak{E}^*$, σ_i^*, $RL\_PK_i^*$) with $i \in \{1......t+1\}$. It succeeds if

(a) $1 \leftarrow$***Verify***(TRK, $RL\_PK_i^*$, $\mathfrak{M}_i^*$, $\mathfrak{E}^*$, σ_i^*) for all $i \in \{1......t+1\}$;

(b) $\mathcal{A}$ did not query ***Sign*** on inputs $RL\_PK_i^*$, $\mathfrak{M}_i^*$ and $\mathfrak{E}^*$ for all $i \in \{1......t+1\}$;

(c) "$Independent$" $\leftarrow$***Trace-User***(TRK, $RL\_PK_i^* \cup RL\_PK_j^*$, $\{\mathfrak{M}_i^*, \sigma_i^*\}$, $\{\mathfrak{M}_j^*, \sigma_j^*\}$, $\mathfrak{E}^*$) for all $i, j \in \{1......t+1\}$ with $i \neq j$;

(d) $\mathcal{A}$ has no more than t user private keys, where the public keys of the t users are included in $RL\_PK_1^* \cup RL\_PK_2^* .... \cup RL\_PK_{t+1}^*$.

Definition 5.3 Exculpability of A Traceable Ring Signature Scheme: *Let* **TRS**=*(**System-Setup**, **Generate-Key**, **Sign**, **Verify**, **Trace-User**) be a traceable ring signature scheme. Additionally, we set that k is a secure parameter, and **Pr**($\mathcal{B}_{E\_TRS}$(k,$\mathcal{A}$)=1) is the probability that the algorithm $\mathcal{B}_{E\_TRS}$ returns 1. Then the advantage that the adversary $\mathcal{A}$ breaks* **TRS** *is defined as follows:*

$$\mathbf{Adv}_{TRS}^{e\_trs-uf}(k, q_g, q_s, \hbar)\text{=}\mathbf{Pr}(\mathcal{B}_{E\_TRS}(k,\mathcal{A})\text{=}1),$$

where q_g is the maximal number of "Generate-Public Key" oracle queries, q_s is the maximal number of "Sign" oracle queries and $\hbar$ is the running time of $\mathcal{B}$. If the advantage that the adversary breaks **TRS** *is negligible, then the scheme* **TRS** *is secure.*

According to the Definition 5.3, the algorithm $\mathcal{B}_{E\_TRS}$ is described as follows:

1.**Setup**: Running ***System-Setup***, (TRK, spk) ←***System-Setup***(1^k), and then TRK is passed to $\mathcal{A}$.

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

- ***Generate-Public Key***(): Given the public parameters TRK, the oracle outputs the public key pk with respect to the corresponding private key.
- ***Sign***(): Given the public parameters TRK, the public key list RL_PK, the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the oracle returns a signature σ to $\mathcal{A}$, where $\sigma \in \{0,1\}^* \cup \{\perp\}$.

3.**Forgery**: $\mathcal{A}$ outputs its forgery, ($\mathfrak{M}^*$, $\mathfrak{E}^*$, σ^*) for RL_PK^*. It succeeds if

(a) 1 ←***Verify***(TRK, RL_PK^*, $\mathfrak{M}^*$, $\mathfrak{E}^*$, σ^*);

(b) $\mathcal{A}$ did not query ***Sign*** on inputs RL_PK^*, $\mathfrak{M}^*$ and $\mathfrak{E}^*$;

(c) "*Linked*" ←***Trace-User***(TRK, RL_PK^*, {$\mathfrak{M}^*$, σ^*}, {$\mathfrak{M}'$, σ'}, $\mathfrak{E}^*$), where σ' is any signature outputted from ***Sign*** on inputs RL_PK^*, $\mathfrak{M}'$ and $\mathfrak{E}^*$.

Definition 5.4 Anonymity of A Traceable Ring Signature Scheme: *Let* **TRS**=*(**System-Setup**, **Generate-Key**, **Sign**, **Verify**, **Trace-User**) be a traceable ring signature scheme. Additionally, we set that k is a secure parameter, and **Pr**($\mathcal{B}_{A\_TRS}$(k,$\mathcal{A}$)=1) is the probability that the algorithm $\mathcal{B}_{A\_TRS}$ returns 1. Then the advantage that the adversary $\mathcal{A}$ breaks* **TRS** *is defined as follows:*

$$\mathbf{Adv}_{TRS}^{a\_trs-uf}(k, q_g, q_s, \hbar)\text{=}\mathbf{Pr}(\mathcal{B}_{A\_TRS}(k,\mathcal{A})\text{=}1),$$

where q_g is the maximal number of "Generate-Public Key" oracle queries, q_s is the maximal number of "Sign" oracle queries and $\hbar$ is the running time of $\mathcal{B}$. If the advantage that the adversary breaks **TRS** *is negligible, then the scheme* **TRS** *is secure.*

According to the Definition 5.4, the algorithm $\mathcal{B}_{A\_TRS}$ is described as follows:

1.**Setup**: Running ***System-Setup***, $(TRK, spk) \leftarrow$***System-Setup***(1^k), and then TRK is passed to $\mathcal{A}$.

2.**Queries Phase 1**: $\mathcal{A}$ makes queries to the following oracles for polynomially many times:

- ***Generate-Public Key***(): Given the public parameters TRK, the oracle outputs the public key pk with respect to the corresponding private key.
- ***Sign***(): Given the public parameters TRK, the public key list RL_PK, the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the oracle returns a signature σ to $\mathcal{A}$, where $\sigma \in \{0,1\}^* \cup \{\perp\}$.

3.**Challenge**: $\mathcal{A}$ sends to the challenger its forgeries, the public keys pk_0^* and pk_1^* of the two ring members and the tuple ($\mathfrak{M}^*$, $\mathfrak{E}^*$, $RL\_PK^* \cup \{pk_0^*\} \cup \{pk_1^*\}$). The forgeries satisfy the condition that $\mathcal{A}$ did not query ***Sign*** on input pk_0^* (and pk_1^*).

The challenger picks a random bit $x \in \{0,1\}$, and then runs and outputs $\sigma^* \leftarrow$***Sign***$(TRK$, $*$, $RL\_PK^* \cup \{pk_0^*\} \cup \{pk_1^*\}$, $\mathfrak{M}^*$, $\mathfrak{E}^*$) to $\mathcal{A}$.

4.**Queries Phase 2**: $\mathcal{A}$ makes queries to the following oracles for polynomially many times:

- ***Generate-Public Key***(): Given the public parameters TRK, the oracle outputs the public key pk with respect to the corresponding private key.
- ***Sign***(): Given the public parameters TRK, the public key list RL_PK, the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the oracle returns a signature σ to $\mathcal{A}$, where $\sigma \in \{0,1\}^* \cup \{\perp\}$, and $\mathcal{A}$ did not query ***Sign*** on inputs pk_0^* and $\mathfrak{E}^*$ (and pk_1^* and $\mathfrak{E}^*$).

5.**Guess**: $\mathcal{A}$ outputs a bit $x' \in \{0,1\}$ and succeeds if $x' = x$.

VI. Traceable Ring Signature Scheme

In the section, we show a traceable ring signature scheme without random oracles under our framework for **TRS**. Let **TRS**=(***System-Setup***, ***Generate-Key***, ***Sign***, ***Verify***, ***Trace-User***) be a traceable ring signature scheme. In **TRS**, all algorithms are described as follows:

1) **TRS.*System-Setup***: The algorithm run by the trusted authority inputs a security parameter 1^k. Additionally, let $\mathbb{G}_1$ and $\mathbb{G}_2$ be groups of prime order q and g be a generator of $\mathbb{G}_1$, and let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ denote the bilinear map. The size of the group is determined by the security parameter, and we set $\mathbb{A} \subseteq \mathbb{Z}_q$ as the universe of identities. And one hash function,

$H : \{0,1\}^* \rightarrow \mathbb{Z}_{1^k \cdot q}$ can be defined and used to generate any integer value in $\mathbb{Z}_{1^k \cdot q}$ (where 1^k represents the corresponding decimal number).

Then the system parameters are generated as follows for a ring system setup. The algorithm chooses a random $a \in \mathbb{Z}_q$, and then sets $g_1 = g^a$. Eight group elements g_2, ϑ, ψ, ϖ, μ, τ, χ and $\kappa \in \mathbb{G}_1$ are randomly chosen. Finally, the algorithm outputs the public parameters TRK=($\mathbb{G}_1$, $\mathbb{G}_2$, e, g, g_1, g_2, ϑ, ψ, ϖ, μ, τ, χ, κ), where $spk = g_2^a$ is seen as a ring key[5].

2) **TRS.*Generate-Key***: The algorithm run by a ring member generates the private key of the ring member with respect to the public key when the ring member joins a ring. The algorithm inputs TRK, and then the following steps are finished:

- The algorithm run by the trusted authority randomly chooses $r_1 \in \mathbb{Z}_q$, computes $x_0 = g_2^a \cdot \varpi^{r_1}$, $s_L = g^{r_1}$, and then outputs a partial private key $psk = \{x_0, s_L\}$ to the ring member, where s_L is used to trace the real signer.

  Remark: Every ring member may verify his partial private key by the following equation:
$$e(x_0, g) = e(g_1, g_2) \cdot e(\varpi, s_L).$$
- The algorithm run by the corresponding ring member chooses $r_2 \in \mathbb{Z}_q$, computes the public key $pk = g^{r_2}$, and $x_1 = x_0 \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2} = g_2^a \cdot \varpi^{r_1} \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2}$, and then outputs the private key $sk = \{x_1, s_L\}$.
- The algorithm run by the corresponding ring member outputs and publishes the public key $pk = g^{r_2}$, which is added to the public key ring RL_PK, where RL_PK is a public key list including all public keys of the ring members belong to this ring and $pk \in RL\_PK$.

3) **TRS.*Sign***: A ring member with the private key sk needs to sign a message $\mathfrak{M} \in \{0,1\}^*$ on an event identifier $\mathfrak{E} \in \{0,1\}^*$. The algorithm run by the ring member inputs (TRK, sk, RL_PK, $\mathfrak{M}$, $\mathfrak{E}$), and then randomly chooses $r_3, r_4, r_5 \in \mathbb{Z}_q$, computes

$$\sigma_0 = x_1 \cdot \vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5}$$
$$= g_2^a \cdot \vartheta^{(r_2+r_3) \cdot H(pk)} \cdot \psi^{r_2+r_3} \cdot \varpi^{r_1+r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5},$$
$$\sigma_1 = e(\vartheta^{H(pk)} \cdot \psi, pk) \cdot e(\vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3}, g)$$
$$= e(\vartheta^{H(pk)} \cdot \psi, g^{r_2}) \cdot e(\vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3}, g)$$
$$= e(\vartheta^{(r_2+r_3) \cdot H(pk)} \cdot \psi^{r_2+r_3}, g),$$
$$\sigma_2 = s_L \cdot g^{r_3} = g^{r_1+r_3},$$

[5]For the different ring, the ring key spk is different. Namely when a new ring system setup is initialized, the new public parameters are published with respect to the new ring.

$\sigma_3 = g^{r_4}$,

$\sigma_4 = g^{r_5}$.

Finally, the algorithm outputs a signature $\Phi = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4\}$.

4) **TRS.*Verify***: The signature verifiers verify a standard traceable ring signature Φ. The algorithm run by a signature verifier inputs (TRK, RL_PK, $\mathfrak{M}$, $\mathfrak{E}$, Φ), and then the following computation is finished:

$$e(\sigma_0, g) = e(g_1, g_2) \cdot \sigma_1 \cdot e(\varpi, \sigma_2) \cdot e(\mu^{H(RL\_PK)} \cdot \tau, \sigma_3) \cdot e(\chi^{H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa, \sigma_4).$$

If the above equation is correct, then the algorithm outputs the boolean value $accept$, otherwise the algorithm outputs the boolean value $reject$.

5) **TRS.*Trace-User***: The trusted authority traces a ring member (signer) by two traceable ring signatures Φ_1 on $\mathfrak{M}_1$ and Φ_2 on $\mathfrak{M}_2$ when the signatures need to be arbitrated. The algorithm run by the trusted authority inputs (TRK, RL_PK, $\{\mathfrak{M}_1, \Phi_1\}$, $\{\mathfrak{M}_2, \Phi_2\}$, $\mathfrak{E}$), and then the following steps are finished:

a) For any potential public key $pk_1 \in RL\_PK$ and the tuple $\{\mathfrak{M}_1, \Phi_1\}$, the algorithm computes the equation:

$$e(\vartheta^{H(pk_1)} \cdot \psi, pk_1 \cdot \tfrac{\sigma_2}{s_L}) = \frac{e(\sigma_0, g)}{e(g_1, g_2) \cdot e(\varpi, \sigma_2) \cdot e(\mu^{H(RL\_PK)} \cdot \tau, \sigma_3) \cdot e(\chi^{H(\mathfrak{M}_1 \| \mathfrak{E})} \cdot \kappa, \sigma_4)}.$$

If the above equation is correct, then the algorithm securely records the public key pk_1 of the real signer, otherwise if the algorithm does not find the corresponding public key, the algorithm aborts; similarly, the same computation is finished for any potential identity $pk_2 \in RL\_PK$ and the tuple $\{\mathfrak{M}_2, \Phi_2\}$, and then the algorithm securely records the public key pk_2 of the real signer, otherwise the algorithm aborts.

b) The algorithm outputs the following results according to the comparisons:

- *Result*="$Independent$", if $pk_1 \neq pk_2$;
- *Result*="$Linked$", else if $\mathfrak{M}_1 = \mathfrak{M}_2$;
- *Result*="pk_1", otherwise.

VII. Analysis of the Proposed Scheme

A. Correctness

In the proposed scheme, the traceable ring signature is $\Phi = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4\}$, where

$$\sigma_0 = x_1 \cdot \vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5}$$

$$= g_2^a \cdot \vartheta^{(r_2+r_3) \cdot H(pk)} \cdot \psi^{r_2+r_3} \cdot \varpi^{r_1+r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5},$$

$$\begin{aligned}
\sigma_1 &= e(\vartheta^{H(pk)} \cdot \psi, pk) \cdot e(\vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3}, g) \\
&= e(\vartheta^{H(pk)} \cdot \psi, g^{r_2}) \cdot e(\vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3}, g) \\
&= e(\vartheta^{(r_2+r_3) \cdot H(pk)} \cdot \psi^{r_2+r_3}, g), \\
\sigma_2 &= s_L \cdot g^{r_3} = g^{r_1+r_3}, \\
\sigma_3 &= g^{r_4}, \\
\sigma_4 &= g^{r_5}.
\end{aligned}$$

So, we have that

$$\begin{aligned}
&e(\sigma_0, g) = e(g_2^a \cdot \vartheta^{(r_2+r_3)\cdot H(pk)} \cdot \psi^{r_2+r_3} \cdot \varpi^{r_1+r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M}\|\mathfrak{E})} \cdot \kappa^{r_5}, g) \\
&= e(g_2^a, g) \cdot e(\vartheta^{(r_2+r_3)\cdot H(pk)} \cdot \psi^{r_2+r_3}, g) \cdot e(\varpi^{r_1+r_3}, g) \cdot e(\mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4}, g) \cdot e(\chi^{r_5 \cdot H(\mathfrak{M}\|\mathfrak{E})} \cdot \kappa^{r_5}, g) \\
&= e(g_1, g_2) \cdot \sigma_1 \cdot e(\varpi, \sigma_2) \cdot e(\mu^{H(RL\_PK)} \cdot \tau, \sigma_3) \cdot e(\chi^{H(\mathfrak{M}\|\mathfrak{E})} \cdot \kappa, \sigma_4).
\end{aligned}$$

B. Efficiency

In the proposed scheme, $\Phi = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4\}$, where

$$\begin{aligned}
\sigma_0 &= x_1 \cdot \vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M}\|\mathfrak{E})} \cdot \kappa^{r_5}, \\
\sigma_1 &= e(\vartheta^{H(pk)} \cdot \psi, pk) \cdot e(\vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3}, g), \\
\sigma_2 &= s_L \cdot g^{r_3}, \;\; \sigma_3 = g^{r_4}, \;\; \sigma_4 = g^{r_5}.
\end{aligned}$$

Thus, the length of signature is $4 \cdot |\mathbb{G}_1| + |\mathbb{G}_2|$, where $|\mathbb{G}_1|$ is the size of element in $\mathbb{G}_1$ and $|\mathbb{G}_2|$ is the size of element in $\mathbb{G}_2$. Additionally, because $x_1 \cdot \vartheta^{r_3 \cdot H(pk)} \cdot \psi^{r_3} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \kappa^{r_5}$, χ^{r_5} in $\chi^{r_5 \cdot H(\mathfrak{M}\|\mathfrak{E})}$, σ_1, σ_2, σ_3 and σ_4 may be precomputed, and we assume that the time for integer multiplication and hash computation can be ignored, signing a message for a traceable ring signature only needs to compute at most 1 exponentiation in $\mathbb{G}_1$ and 1 multiplication in $\mathbb{G}_1$. Also, in the following equation

$$e(\sigma_0, g) = e(g_1, g_2) \cdot \sigma_1 \cdot e(\varpi, \sigma_2) \cdot e(\mu^{H(RL\_PK)} \cdot \tau, \sigma_3) \cdot e(\chi^{H(\mathfrak{M}\|\mathfrak{E})} \cdot \kappa, \sigma_4),$$

because the value $e(g_1, g_2)$ can be precomputed and cached, verification requires 4 pairing computations, 2 exponentiations in $\mathbb{G}_1$, 2 multiplications in $\mathbb{G}_1$ and 4 multiplications in $\mathbb{G}_2$.

In this paper, we compare the proposed scheme (the scheme of Section 6) with other traceable (or linkable) ring signature schemes proposed by [15,16,18,28,29]. In Appendix A, we show the comparisons of the six schemes.

C. Security

In the section, we show the proposed scheme (the scheme of Section 6) has a security reduction to the CDH assumption and the TRS unforgeability (against *linkability attacks* and *exculpability attacks*) under the adaptive chosen message attacks, and has the TRS anonymity. Our proofs for the following theorems are based on the security model of Section 5 (We defer the proofs to Appendix B). In addition, although the unforgeability may be proved by jointing the Theorem 7.2 and the Theorem 7.3 (see [15,18] for more details), we still show the proof of Theorem 7.1.

Theorem 7.1 *The scheme of Section 6 is ($\hbar$, ε, q_g, q_s)-unforgeable (according to the Definition 5.1), assuming that the ($\hbar'$, ε')-CDH assumption holds in $\mathbb{G}_1$, where*

$$\varepsilon' = (1 - \tfrac{q_g}{q}) \cdot (1 - \tfrac{q_s}{q})^2 \cdot \varepsilon,$$
$$\hbar' = \hbar + O(q_g \cdot (5 \cdot C_{exp} + 3 \cdot C_{mul}) + q_s \cdot (12 \cdot C_{exp} + 7 \cdot C_{mul})),$$

and q_g is the maximal number of "Generate-Public Key" oracle queries, q_s is the maximal number of "Sign" oracle queries, C_{mul} and C_{exp} are respectively the time for a multiplication and an exponentiation in $\mathbb{G}_1$.

Theorem 7.2 *The scheme of Section 6 is a linkable (traceable) TRS scheme when it satisfies the following condition (according to the Definition 5.2)—the scheme of Section 6 is ($\hbar$, ε, q_g, q_s)-secure, assuming that the ($\hbar'$, ε')-CDH assumption holds in $\mathbb{G}_1$, where*

$$\varepsilon' = (1 - \tfrac{q_g}{q}) \cdot (1 - \tfrac{q_s}{q})^2 \cdot \varepsilon,$$
$$\hbar' = \hbar + O(q_g \cdot (5 \cdot C_{exp} + 3 \cdot C_{mul}) + q_s \cdot (12 \cdot C_{exp} + 7 \cdot C_{mul})),$$

and q_g is the maximal number of "Generate-Public Key" oracle queries, q_s is the maximal number of "Sign" oracle queries, C_{mul} and C_{exp} are respectively the time for a multiplication and an exponentiation in $\mathbb{G}_1$, $t = 1$ is the number of user (ring member) private keys possessed by adversary.

Theorem 7.3 *The scheme of Section 6 is exculpable when it satisfies the following condition (according to the Definition 5.3)—the scheme of Section 6 is ($\hbar$, ε, q_g, q_s)-secure, assuming that the ($\hbar'$, ε')-CDH assumption holds in $\mathbb{G}_1$, where*

$$\varepsilon' = (1 - \tfrac{q_g}{q}) \cdot (1 - \tfrac{q_s}{q})^2 \cdot \varepsilon,$$
$$\hbar' = \hbar + O(q_g \cdot (5 \cdot C_{exp} + 3 \cdot C_{mul}) + q_s \cdot (12 \cdot C_{exp} + 7 \cdot C_{mul})),$$

and q_g is the maximal number of "Generate-Public Key" oracle queries, q_s is the maximal number of "Sign" oracle queries, C_{mul} and C_{exp} are respectively the time for a multiplication and an exponentiation in $\mathbb{G}_1$.

Theorem 7.4 *The scheme of Section 6 is ($\hbar$, ε, q_g, q_s)-anonymous (according to the Definition 5.4), assuming that the ($\hbar'$, ε')-CDH assumption holds in $\mathbb{G}_1$, where*

$$\varepsilon' = (1 - \tfrac{q_{g_1}}{q}) \cdot (1 - \tfrac{q_{s_1}}{q})^2 \cdot (1 - \tfrac{q_{g_2}}{q}) \cdot (1 - \tfrac{q_{s_2}}{q})^2 \cdot \varepsilon,$$
$$\hbar' = \hbar + O\left((q_{g_1} + q_{g_2}) \cdot (5 \cdot C_{exp} + 3 \cdot C_{mul}) + (q_{s_1} + q_{s_2}) \cdot (12 \cdot C_{exp} + 7 \cdot C_{mul})\right),$$

q_{g_1} and q_{g_2} are respectively the maximal numbers of "Generate-Public Key" oracle queries in the Queries Phase 1 and 2, q_{s_1} and q_{s_2} are respectively the maximal numbers of "Sign" oracle queries in the Queries Phase 1 and 2, C_{mul} and C_{exp} are respectively the time for a multiplication and an exponentiation in $\mathbb{G}_1$.

VIII. Conclusions

In this paper, we present a fully traceable ring signature scheme, which has a security reduction to the computational Diffie-Hellman assumption. Also, we give a formal security model for traceable ring signature. Under our security model, the proposed scheme is proved to have the properties of anonymity and traceability with enough security. Compared with other traceable ring signature schemes, the proposed scheme is efficient. However, because the proposed scheme is not enough efficient in computing linking of signatures, the work about TRS still needs to be further progressed.

References

[1] R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In C. Boyd, editor, Asiacrypt 2001, LNCS 2248, pages 552–565. Springer-Verlag, 2001.

[2] M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n Signatures from a Variety of Keys. In ASIACRYPT 2002, volume 2501 of Lecture Notes in Computer Science, pages 415–432. Springer, 2002.

[3] F. Zhang and K. Kim. ID-Based Blind Signature and Ring Signature from Pairings. In ASIACRYPT 2002, volume 2501 of Lecture Notes in Computer Science, pages 533–547. Springer, 2002.

[4] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and Verifieably Encrypted Signatures from Bilinear Maps. In EURO–CRYPT 2003, volume 2656 of Lecture Notes in Computer Science, pages 416–432. Springer, 2003.

[5] D. S. Wong, K. Fung, J. K. Liu, and Victor K. Wei. On the rs-code construction of ring signature schemes and a threshold setting of rst. In ICICS 2003, volume 2836 of Lecture Notes in Computer Science, pages 34–46. Springer, 2003.

[6] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous Identification in Ad Hoc Groups. In EUROCRYPT 2004, volume 3027 of Lecture Notes in Computer Science, pages 609–626, Springer, 2004.

[7] S. S. M. Chow, S. M. Yiu, and L. C. K. Hui. Efficient Identity Based Ring Signature. In ACNS 2005, volume 3531 of Lecture Notes in Computer Science, pages 499–512, 2005.

[8] D. Chaum and E. Van Heyst. Group signatures. In D. W. Davies, editor, EUROCRYPT91, LNCS 547, pages 257–265. Springer-Verlag, 1991.

[9] M. Naor. Deniable ring authentication. In CRYPTO 2002, pages 481–498, 2002.

[10] E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In Moti Yung, editor, CRYPTO 2002, LNCS 2442, pages 465–480. Springer-Verlag, 2002.

[11] M. Abe, M. Ohkubo, and K. Suzuki. Efficient threshold signer-ambiguous signatures from variety of keys. IEICE Trans. Fund., vol.E87-A, no.2:471–479, 2004.

[12] J. K. Liu, V. K. Wei, and D. S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups (extended abstract). In ACISP 2004, LNCS 3108, pages 325–335, 2004.

[13] Y. Komano, K. Ohta, A. Shimbo, and S. Kawamura. Toward the fair anonymous signatures: Deniable ring signatures. In D. Pointcheval, editor, CT-RSA 06, LNCS 3860, pages 174–191. Springer-Verlag, 2006.

[14] A. Bender, J. Katz, and R. Morselli. Ring signatures:stronger definitions, and constructions without random oracles. In S. Halevi and T. Rabin, editors, TCC 2006, LNCS 3876, pages 60–79. Springer-Verlag, 2006.

[15] M. H. Au, J. K. Liu, W. Susilo, T. H. Yuen. Secure ID-Based Linkable and Revocable-iff-Linked Ring Signature with Constant-Size Construction. Preprint submitted to Theoretical Computer Science, April 23, 2012.

[16] J. K. Liu and D. S. Wong. Linkable ring signatures: Security models and new schemes. In ICCSA, volume 3481 of Lecture Notes in Computer Science, pages 614–623. Springer, 2005.

[17] J. K. Liu and D. S. Wong. Enhanced security models and a generic construction approach for linkable ring signature. Int. J. Found. Comput. Sci., 17(6):1403–1422, 2006.

[18] E. Fujisaki, K. Suzuki. Traceable ring signature. In Public Key Cryptography, volume 4450 of Lecture Notes in Computer Science, pages 181–200. Springer, 2007.

[19] D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. Rivest, and A. Sherman, editors, CRYPTO 82, pages 199–204.

[20] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In S. Goldwasser, editor, CRYPTO 88, LNCS 403, pages 319–327. Springer-Verlag, 1990.

[21] T. Okamoto and K. Ohta. Universal electronic cash. In CRYPTO 91, LNCS 576, pages 324–337. Springer-Verlag, 1992.

[22] S. Brands. Untraceable off-line cash in wallet with observers. In D. Stinson, editor, CRYPTO 93, LNCS 773, pages 302–318. Springer-Verlag, 1993.

[23] S. S. M. Chow, J. K. Liu, and D. S. Wong. Robust receipt-free election system with ballot secrecy and verifieability. In NDSS. The Internet Society, 2008.

[24] P. P. Tsang and V. K. Wei. Short linkable ring signatures for e-voting, e-cash and attestation. In ISPEC 2005, volume 3439 of Lecture Notes in Computer Science, pages 48–60. Springer, 2005.

[25] W. Susilo and Y. Mu. Non-Interactive Deniable Ring Authentication. In ICISC 2003, volume 2971 of Lecture Notes in Computer Science, pages 386–401. Springer, 2004.

[26] F. Laguillaumie and D. Vergnaud. Multi-designated Verifiers Signatures. In ICICS 2004, volume 3269 of Lecture Notes in Computer Science, pages 495–507, Malaga, Spain, October 2004. Springer.

[27] I. Damgard, K. Dupont, and M. Pedersen. Unclonable group identification. In S. Vaudenay, editor, EUROCRYPT 2006, LNCS 4004, pages 555–572. Springer-Verlag, 2006.

[28] D. Zheng, X. Li, K. Chen, and J. Li. Linkable ring signatures from linear feedback shift register. In EUC Workshops, volume 4809 of Lecture Notes in Computer Science, pages 716–727. Springer, 2007.

[29] E. Fujisaki. Sub-linear size traceable ring signatures without random oracles. In CT-RSA, volume 6558 of Lecture Notes in Computer Science, pages 393–415. Springer, 2011.

[30] P. P. Tsang, V. K. Wei, T. K. Chan, M. H. Au, J. K. Liu, and D. S. Wong. Separable linkable threshold ring signatures. In INDCRYPT 2004, LNCS 3348, pages 389–398, 2004.

[31] M. H. Au, S. S. M. Chow, W. Susilo, and P. P. Tsang. Short linkable ring signatures revisited. In EUROPKI 2006, LNCS 4043, pages 101–115, 2006.

[32] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous identification in ad hoc groups. In C. Cachin and J. Camenisch, editors, EUROCRYPT 2004, LNCS 3027, pages 609–626. Springer-Verlag, 2004.

[33] Dennis Y. W. Liu, Joseph K. Liu, Yi Mu, Willy Susilo, and Duncan S. Wong. Revocable ring signature. J. Comput. Sci. Technol., 22(6):785–794, 2007.

[34] D. Boneh, M. Franklin. Identity-based encryption from the Weil pairing. In: J.Kilian, ed. Advances in Cryptology-CRYPTO 2001. LNCS 2139, Berlin:Springer-Verlag,2001. 213–229.

[35] D. Boneh, M. Hanburg. Generalized identity based and broadcast encryption schemes. In: J. Pieprzyk, ed. Advances in Cryptology-ASIACRYPT 2008. LNCS 5350, Berlin:Springer-Verlag, 2008. 455–470.

[36] K. G. Paterson, J. C. N. Schuldt. Efficient identity-based signatures secure in the standard model, ACISP2006, LNCS 4058, Springer-Verlag, 2006, pp.207–222.

[37] B. Waters, Efficient identity-based encryption without random oracles, Advances in Cryptology-EUROCRYPT 2005, LNCS 3494, Springer-Verlag, 2005, pp.114–127.

[38] P. S. L. M. Barreto, B. Libert, N. McCullagh, J. Quisquater. Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps. In B. Roy, editor(s), Asiacrypt 2005, LNCS 3788, Berlin:Springer-Verlag,2005. 515–532.

[39] J.C. Cha, J.H. Cheon. An identity-based signature from gap Diffie-Hellman groups. In Y. Desmedt, editor, Public Key Cryptography - PKC 2003, LNCS 2567, Berlin:Springer-Verlag,2002. 18–30.

[40] F. Hess. Efficient identity based signature schemes based on pairings. In K. Nyberg, H. Heys, editors, Selected Areas in Cryptography 9th Annual International Workshop, SAC 2002, LNCS 2595, Berlin:Springer-Verlag,2003. 310–324.

[41] F.T. Wen, S.J. Cui, J.N. Cui. An ID-based Proxy Signature Scheme Secure Against Proxy Key Exposure, International Journal of Advancements in Computing Technology, 2011, 3(5):108–116.

[42] W. Wu, Y. Mu, W. Susilo, J. Seberry, X.Y. Huang. Identity-Based Proxy Signature from Pairings, In B.Xiao et al. editor(s), ATC 2007, LNCS 4610, Berlin:Springer-Verlag, 2007. 22–31.

[43] H. Singh, G.K. Verma. ID-based proxy signature scheme with message recovery, Journal of Systems and Software, 2012, 85(1):209–214.

[44] M. H. Au, J. K. Liu, T. H. Yuen, D. S. Wong. ID-based ring signature scheme secure in the standard model, In Proceeding of IWSEC 2006, pp.1–16.

[45] F. Zhang, K. Kim, ID-based blind signature and ring signature from pairings. Asiacrypt2002, LNCS 2501, Berlin:Springer-Verlag, 2002. 533–547.

[46] M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen. Constantsize id-based linkable and revocable-iff-linked ring signature. In IN-DOCRYPT, volume 4329 of Lecture Notes in Computer Science, pages 364–378. Springer, 2006.

[47] L. Ibraimi, S. Nikova, P. Hartel, W. Jonker. An Identity-Based Group Signature with Membership Revocation in the Standard Model. Available at: http:/doc.utwente.nl/72270/1/Paper.pdf.

[48] K. Emura, A. Miyaji, K. Omote. An r-Hiding Revocable Group Signature Scheme: Group Signatures with the Property of Hiding the Number of Revoked Users. Journal of Applied Mathematics, Volume 2014, Article ID 983040, 14 pages.

[49] I. R. Jeong, J. O. Kwon, D.g H. Lee. Analysis of revocable-iff-linked ring signature scheme. IEICE Transactions on Fundamentals of Electronics Communications & Computer Sciences, 2009, 92-A(1):322–325.

[50] T. H. Yuen, J. K. Liu, M. H. Au, W. Susilo, J. Zhou. Efficient linkable and/or threshold ring signature without random oracles. The Computer Journal, 2013, 56(4):407–421.

[51] J. K. Liu, M. H. Au, W. Susilo, J. Zhou. Linkable Ring Signature with Unconditional Anonymity. IEEE Transactions on Knowledge and Data Engineering, 2014, 26(1):157–165.

Appendix A
Comparisons of Traceable or Linkable Ring Signature Schemes

Table 1 shows the comparisons of the traceable or linkable ring signature schemes. Compared with other schemes, our scheme is constructed without random oracles, and has the constant signature size in the comparison of the performance.

TABLE I
Comparisons of the Six Schemes

	Signature Size	Cryptography	Traceability	Linking Cost	Model
Scheme [16]	$O(n)$	Public Key	No	$O(1)$	random oracle
Scheme [28]	$O(n)$	Public Key	No	$O(1)$	random oracle
Scheme [29]	$O(\sqrt{n})$	Public Key	Yes	$O(n \cdot \log n)$	without random oracle
Scheme [18]	$O(n)$	Public Key	Yes	$O(n)$	random oracle
Scheme [15]	$O(1)$	Identity-Based	Yes	$O(1)$	random oracle
Our Scheme	$O(1)$	Public Key	Yes	$O(n)$	without random oracle

Appendix B
Security Proof

(**Proof of Theorem 7.1**).

Proof: Let **TRS** be a traceable ring signature scheme of Section 6. Additionally, let $\mathcal{A}$ be an ($\hbar$, ε, q_g, q_s)-adversary attacking **TRS**. From the adversary $\mathcal{A}$, we construct an algorithm $\mathcal{B}$, for (g, g^a, g^b)$\in \mathbb{G}_1$, the algorithm $\mathcal{B}$ is able to use $\mathcal{A}$ to compute $g^{a \cdot b}$. Thus, we assume the algorithm $\mathcal{B}$ can solve the CDH with probability at least ε' and in time at most $\hbar'$, contradicting the ($\hbar'$, ε')-CDH assumption. Such a simulation may be created in the following way:

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

Then the system parameters are generated as follows. The algorithm chooses $y \in \mathbb{Z}_q$, and then sets $g_1 = g^y$ and $g_2 = g^b$ with $b \in \mathbb{Z}_q$ ($\mathcal{B}$ doesn't know b). Also, the algorithm chooses ℓ, ∂, ν, λ, η, α and $\pi \in \mathbb{Z}_q$, and then sets $\vartheta = g_2^{\ell} \cdot g$, $\psi = g^{\partial}$, $\mu = g^{\nu}$, $\tau = g^{\lambda}$, $\chi = g_2^{\alpha} \cdot g$, $\kappa = g^{\pi}$ and $\varpi = g^{\eta}$. Finally, the system outputs the public parameters TRK=($\mathbb{G}_1$, $\mathbb{G}_2$, e, g, g_1, g_2, ϑ, ψ, μ, τ, χ, κ, ϖ) to $\mathcal{A}$.

Additionally, the user u^* is a challenger whose public key is pk^*, we set that the public key of u^*, $pk^* = g^a$ ($\mathcal{B}$ doesn't know a).

Queries: When running the adversary $\mathcal{A}$, the relevant queries can occur according to the Definition 5.1. The algorithm $\mathcal{B}$ answers these in the following way:

- **Generate-Public Key Queries**: Given the public parameters TRK, the algorithm $\mathcal{B}$ randomly chooses $r_1, r_2 \in \mathbb{Z}_q$, computes the public key $pk = g^{r_2}$ of the ring member u and the corresponding private key $sk = \left\{g_2^y \cdot \varpi^{r_1} \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2}, g^{r_1}\right\}$, and then outputs the public key pk to $\mathcal{A}$, which is added to the public key ring RL_PK, where RL_PK is a public key list including all public keys of the ring members belong to this ring. To finish the above computation, we assure that $H(pk) \neq 0$ **mod** q.
- **Sign Queries**: Given the public parameters TRK, the public key list RL_PK ($pk \in RL\_PK$ where pk is the public key of the ring member that belongs to this ring), the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the algorithm $\mathcal{B}$ chooses random $r_2, r_3, r_4, r_5 \in \mathbb{Z}_q$ and computes

$$\sigma_0 = g_2^y \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5},$$
$$\sigma_1' = g^{r_2},$$
$$\sigma_2 = g^{r_3},$$
$$\sigma_3 = g^{r_4},$$
$$\sigma_4 = g^{r_5}.$$

Finally, the algorithm outputs a forgery $\Phi = \{\sigma_0, \sigma_1', \sigma_2, \sigma_3, \sigma_4\}$ to the adversary $\mathcal{A}$, where we maximize the adversary's advantage, σ_1' is passed to $\mathcal{A}$. Thus, $\Phi' = \{\sigma_0, \sigma_1 = e(\vartheta^{H(pk)} \cdot \psi, \sigma_1'), \sigma_2, \sigma_3, \sigma_4\}$ is a valid signature, where we assure that $H(pk) \neq 0$ **mod** q and $H(\mathfrak{M} \parallel \mathfrak{E}) \neq 0$ **mod** q.

Forgery: If the algorithm $\mathcal{B}$ does not abort as a consequence of one of the queries above, the adversary $\mathcal{A}$ will, with probability at least ε, return its forgeries, ($\mathfrak{M}^*$, $\mathfrak{E}^*$, Φ^*, RL_PK^*) for the challenger u^*, with $pk^* \in RL\_PK^*$, $\Phi^* = \{\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, \sigma_4^*, \sigma_5^*, \sigma_6^*\}$, where

$$\begin{aligned}
&\sigma_0^* = g_2^y \cdot \vartheta^{(r_2^*+a)\cdot H(pk^*)} \cdot \psi^{r_2^*+a} \cdot \varpi^{r_3^*} \cdot \mu^{r_4^* \cdot H(RL\_PK^*)} \cdot \tau^{r_4^*} \cdot \chi^{r_5^* \cdot H(\mathfrak{M}_1^* \| \mathfrak{E}^*)} \cdot \kappa^{r_5^*},\\
&\sigma_1^* = g^{r_2^*},\\
&\sigma_2^* = g^{r_3^*},\\
&\sigma_3^* = g^{r_4^*},\\
&\sigma_4^* = g^{r_5^*},\\
&\sigma_5^* = g_2^{r_2^*},\\
&\sigma_6^* = g_2^{r_5^*}.
\end{aligned}$$

Remark: In fact, σ_1^* should be equal to $g^{r_2^*} \cdot pk^*$. Additionally, because the adversary $\mathcal{A}$ can compute $\sigma_1^* = g^{r_2^*}$ and $\sigma_4^* = g^{r_5^*}$, $\mathcal{A}$ can easily convert these computations to $\sigma_5^* = g_2^{r_2^*}$ and $\sigma_6^* = g_2^{r_5^*}$, where σ_5^* and σ_6^* return to the algorithm $\mathcal{B}$ so as to make $\mathcal{B}$ solve the CDH problem.

And the forgeries satisfy the following conditions:

(a) $1 \leftarrow$***Verify***$(TRK,\ RL\_PK^*,\ \mathfrak{M}^*,\ \mathfrak{E}^*,\ \Phi^*)$[6];

(b) $\mathcal{A}$ did not query ***Sign*** on inputs RL_PK^*, $\mathfrak{M}^*$ and $\mathfrak{E}^*$.

So, the algorithm $\mathcal{B}$ computes and outputs

$$\begin{aligned}
&\frac{\sigma_0^*}{g_2^y \cdot g_2^{r_2^* \cdot \ell \cdot H(pk^*)} \cdot g^{r_2^* \cdot H(pk^*)} \cdot pk^{*H(pk^*)} \cdot g^{r_2^* \cdot \partial} \cdot pk^{*\partial} \cdot g^{r_3^* \cdot \eta} \cdot g^{r_4^* \cdot \nu \cdot H(RL\_PK^*)} \cdot g^{r_4^* \cdot \lambda} \cdot g_2^{r_5^* \cdot \alpha \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot g^{r_5^* \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot g^{r_5^* \cdot \pi}}\\
&= \frac{g_2^y \cdot \vartheta^{(r_2^*+a)\cdot H(pk^*)} \cdot \psi^{r_2^*+a} \cdot \varpi^{r_3^*} \cdot \mu^{r_4^* \cdot H(RL\_PK^*)} \cdot \tau^{r_4^*} \cdot \chi^{r_5^* \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot \kappa^{r_5^*}}{g_2^y \cdot g_2^{r_2^* \cdot \ell \cdot H(pk^*)} \cdot g^{r_2^* \cdot H(pk^*)} \cdot pk^{*H(pk^*)} \cdot g^{r_2^* \cdot \partial} \cdot pk^{*\partial} \cdot g^{r_3^* \cdot \eta} \cdot g^{r_4^* \cdot \nu \cdot H(RL\_PK^*)} \cdot g^{r_4^* \cdot \lambda} \cdot g_2^{r_5^* \cdot \alpha \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot g^{r_5^* \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot g^{r_5^* \cdot \pi}}\\
&= g_2^{\ell \cdot a \cdot H(pk^*)}.
\end{aligned}$$

Further, we can compute $\sqrt[\ell \cdot H(pk^*)]{g_2^{\ell \cdot a \cdot H(pk^*)}} = g_2^a = g^{a \cdot b}$, which is the solution to the given CDH problem.

Now, we analyze the probability of the algorithm $\mathcal{B}$ not aborting. For the simulation to complete without aborting, we require that all ***Generate-Public Key*** queries will have $H(pk) \neq 0$ **mod** q, all ***Sign*** queries will have $H(pk) \neq 0$ **mod** q and $H(\mathfrak{M} \parallel \mathfrak{E}) \neq 0$ **mod** q. If the algorithm $\mathcal{B}$ does not abort, then the following conditions must hold:

(a) $H(pk_i) \neq 0$ **mod** q in ***Generate-Public Key*** queries, with i=$1, 2......q_g$;

(b) $H(pk_i) \neq 0$ **mod** q and $H(\mathfrak{M}_i \parallel \mathfrak{E}_i) \neq 0$ **mod** q in ***Sign*** queries, with i=$1, 2......q_s$.

To make the analysis simpler, we will define the events E_i, F_i, T_i as

E_i :$H(pk_i) \neq 0$ **mod** q, with i=$1, 2......q_g$;

F_i :$H(pk_i) \neq 0$ **mod** q, with i=$1, 2......q_s$;

T_i :$H(\mathfrak{M}_i \parallel \mathfrak{E}_i) \neq 0$ **mod** q, with i=$1, 2......q_s$.

[6] σ_5^* and σ_6^* do not anticipate in this computation.

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

$$\Pr(not\_abort) = \Pr\left(\bigcap_{i=1}^{q_g} E_i \wedge \bigcap_{i=1}^{q_s} (F_i \wedge T_i)\right).$$

It is easy to see that the events $\bigcap_{i=1}^{q_g} E_i$, $\bigcap_{i=1}^{q_s} F_i$, $\bigcap_{i=1}^{q_s} T_i$ are independent. Then we may compute

$$\Pr(\bigcap_{i=1}^{q_g} E_i) = 1 - \Pr(\bigcup_{i=1}^{q_g} \neg E_i) = 1 - q_g \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_g}{q};$$

$$\Pr(\bigcap_{i=1}^{q_s} F_i) = 1 - \Pr(\bigcup_{i=1}^{q_s} \neg F_i) = 1 - q_s \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_s}{q};$$

$$\Pr(\bigcap_{i=1}^{q_s} T_i) = 1 - \Pr(\bigcup_{i=1}^{q_s} \neg T_i) = 1 - q_s \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_s}{q};$$

Thus,

$$\begin{aligned} \Pr(not\_abort) &= \Pr\left(\bigcap_{i=1}^{q_g} E_i \wedge \bigcap_{i=1}^{q_s} (F_i \wedge T_i)\right) \\ &= \Pr(\bigcap_{i=1}^{q_g} E_i) \cdot \Pr(\bigcap_{i=1}^{q_s} F_i) \cdot \Pr(\bigcap_{i=1}^{q_s} T_i) \\ &= \left(1 - \frac{q_g}{q}\right) \cdot \left(1 - \frac{q_s}{q}\right)^2. \end{aligned}$$

We can get that $\varepsilon' = (1 - \frac{q_g}{q}) \cdot (1 - \frac{q_s}{q})^2 \cdot \varepsilon$.

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

$$\hbar' = \hbar + O(q_g \cdot (5 \cdot C_{exp} + 3 \cdot C_{mul}) + q_s \cdot (12 \cdot C_{exp} + 7 \cdot C_{mul})).$$

Thus, Theorem 7.1 follows.

(**Proof of Theorem 7.2**).

Proof: Let **TRS** be a traceable ring signature scheme of Section 6. Additionally, let $\mathcal{A}$ be an ($\hbar$, ε, q_g, q_s)-adversary attacking **TRS**. From the adversary $\mathcal{A}$, we construct an algorithm $\mathcal{B}$, for (g, g^a, g^b)$\in \mathbb{G}_1$, the algorithm $\mathcal{B}$ is able to use $\mathcal{A}$ to compute $g^{a \cdot b}$. Thus, we assume the algorithm $\mathcal{B}$ can solve the CDH with probability at least ε' and in time at most $\hbar'$, contradicting the ($\hbar'$, ε')-CDH assumption. Such a simulation may be created in the following way:

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

Then the system parameters are generated as follows. The algorithm chooses $y \in \mathbb{Z}_q$, and then sets $g_1 = g^y$ and $g_2 = g^b$ with $b \in \mathbb{Z}_q$ ($\mathcal{B}$ doesn't know b). Also, the algorithm chooses ℓ, ∂, ν, λ, η, α and $\pi \in \mathbb{Z}_q$, and then sets $\vartheta = g_2^{\ell} \cdot g$, $\psi = g^{\partial}$, $\mu = g^{\nu}$, $\tau = g^{\lambda}$, $\chi = g_2^{\alpha} \cdot g$, $\kappa = g^{\pi}$ and $\varpi = g^{\eta}$. Finally, the system outputs the public parameters TRK=($\mathbb{G}_1$, $\mathbb{G}_2$, e, g, g_1, g_2, ϑ, ψ, μ, τ, χ, κ, ϖ) to $\mathcal{A}$.

Additionally, the user u^* is a challenger whose public key is pk^*, we set that the public key of u^*, $pk^* = g^a$ ($\mathcal{B}$ doesn't know a). To make our description easier to understand, we only set $t = 1$, namely the adversary $\mathcal{A}$ gets the private key of one user.

Queries: When running the adversary $\mathcal{A}$, the relevant queries can occur according to the Definition 5.2. The algorithm $\mathcal{B}$ answers these in the following way:

- **Generate-Public Key Queries**: Given the public parameters TRK, the algorithm $\mathcal{B}$ randomly chooses $r_1, r_2 \in \mathbb{Z}_q$, computes the public key $pk = g^{r_2}$ of the ring member u and the corresponding private key $sk = \left\{g_2^y \cdot \varpi^{r_1} \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2}, g^{r_1}\right\}$, and then outputs the public key pk to $\mathcal{A}$, which is added to the public key ring RL_PK, where RL_PK is a public key list including all public keys of the ring members belong to this ring. To finish the above computation, we assure that $H(pk) \neq 0 \textbf{ mod } q$.
- **Sign Queries**: Given the public parameters TRK, the public key list RL_PK ($pk \in RL\_PK$ where pk is the public key of the ring member that belongs to this ring), the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the algorithm $\mathcal{B}$ chooses random $r_2, r_3, r_4, r_5 \in \mathbb{Z}_q$ and computes

$$\sigma_0 = g_2^y \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5},$$
$$\sigma_1' = g^{r_2},$$
$$\sigma_2 = g^{r_3},$$
$$\sigma_3 = g^{r_4},$$
$$\sigma_4 = g^{r_5}.$$

Finally, the algorithm outputs a forgery $\Phi = \{\sigma_0, \sigma_1', \sigma_2, \sigma_3, \sigma_4\}$ to the adversary $\mathcal{A}$, where we maximize the adversary's advantage, σ_1' is passed to $\mathcal{A}$. Thus, $\Phi' = \{\sigma_0, \sigma_1 = e(\vartheta^{H(pk)} \cdot \psi, \sigma_1'), \sigma_2, \sigma_3, \sigma_4\}$ is a valid signature, where we assure that $H(pk) \neq 0$ **mod** q and $H(\mathfrak{M} \parallel \mathfrak{E}) \neq 0$ **mod** q.

Forgery: If the algorithm $\mathcal{B}$ does not abort as a consequence of one of the queries above, the adversary $\mathcal{A}$ will, with probability at least ε, return its forgeries, ($\mathfrak{M}_1^*$, $\mathfrak{E}^*$, Φ_1^*, $RL\_PK_1^*$) for the challenger u^* and ($\mathfrak{M}_2^*$, $\mathfrak{E}^*$, Φ_2^*, $RL\_PK_2^*$), with $pk^* \in RL\_PK_1^*$, $\Phi_1^* = \{\sigma_{10}^*, \sigma_{11}^*, \sigma_{12}^*, \sigma_{13}^*, \sigma_{14}^*, \sigma_{15}^*, \sigma_{16}^*\}$ and $\Phi_2^* = \{\sigma_{20}^*, \sigma_{21}^*, \sigma_{22}^*, \sigma_{23}^*, \sigma_{24}^*\}$, where

$$\begin{aligned}
\sigma_{10}^* &= g_2^y \cdot \vartheta^{(r_{12}^*+a)\cdot H(pk^*)} \cdot \psi^{r_{12}^*+a} \cdot \varpi^{r_{13}^*} \cdot \mu^{r_{14}^* \cdot H(RL\_PK_1^*)} \cdot \tau^{r_{14}^*} \cdot \chi^{r_{15}^* \cdot H(\mathfrak{M}_1^* \parallel \mathfrak{E}^*)} \cdot \kappa^{r_{15}^*},\\
\sigma_{11}^* &= g^{r_{12}^*},\\
\sigma_{12}^* &= g^{r_{13}^*},\\
\sigma_{13}^* &= g^{r_{14}^*},\\
\sigma_{14}^* &= g^{r_{15}^*},\\
\sigma_{15}^* &= g_2^{r_{12}^*},\\
\sigma_{16}^* &= g_2^{r_{15}^*}.\\
\sigma_{20}^* &= g_2^y \cdot \vartheta^{r_{22}^* \cdot H(pk^{*'})} \cdot \psi^{r_{22}^*} \cdot \varpi^{r_{23}^*} \cdot \mu^{r_{24}^* \cdot H(RL\_PK_2^*)} \cdot \tau^{r_{24}^*} \cdot \chi^{r_{25}^* \cdot H(\mathfrak{M}_2^* \parallel \mathfrak{E}^*)} \cdot \kappa^{r_{25}^*},\\
\sigma_{21}^* &= g^{r_{22}^*},\\
\sigma_{22}^* &= g^{r_{23}^*},\\
\sigma_{23}^* &= g^{r_{24}^*},\\
\sigma_{24}^* &= g^{r_{25}^*}.
\end{aligned}$$

Remark: σ_{11}^* should be equal to $g^{r_{12}^*} \cdot pk^*$. And $pk^{*'}$ is the public key of the user corrupted by $\mathcal{A}$, where $pk^{*'} \in RL\_PK_2^*$. Similarly, because the adversary $\mathcal{A}$ can compute $\sigma_{11}^* = g^{r_{12}^*}$ and $\sigma_{14}^* = g^{r_{15}^*}$, $\mathcal{A}$ can easily convert these computations to $\sigma_{15}^* = g_2^{r_{12}^*}$ and $\sigma_{16}^* = g_2^{r_{15}^*}$, where σ_{15}^* and σ_{16}^* return to the algorithm $\mathcal{B}$ so as to make $\mathcal{B}$ solve the CDH problem.

And the forgeries satisfy the following conditions[7]:

(a) $1 \leftarrow$***Verify***(TRK, $RL\_PK_i^*$, $\mathfrak{M}_i^*$, $\mathfrak{E}^*$, Φ_i^*) for all $i \in \{1, 2\}$;

(b) $\mathcal{A}$ did not query ***Sign*** on inputs $RL\_PK_i^*$, $\mathfrak{M}_i^*$ and $\mathfrak{E}^*$ for all $i \in \{1, 2\}$;

(c) $''Independent''$ $\leftarrow$***Trace-User***(TRK, $RL\_PK_1^* \cup RL\_PK_2^*$, $\{\mathfrak{M}_1^*, \Phi_1^*\}$, $\{\mathfrak{M}_2^*, \Phi_2^*\}$, $\mathfrak{E}^*$);

(d) $\mathcal{A}$ has no more than 1 user (ring member) private key, where the public key of the user is included in $RL\_PK_1^* \cup RL\_PK_2^*$.

Thus, we have the followings:

[7] σ_{15}^* and σ_{16}^* do not anticipate in these following computations.

$$\frac{\sigma_{10}^*}{g_2^y \cdot g_2^{r_{12}^* \cdot \ell \cdot H(pk^*)} \cdot g^{r_{12}^* \cdot H(pk^*)} \cdot pk^{*H(pk^*)} \cdot g^{r_{12}^* \cdot \partial} \cdot pk^{*\partial} \cdot g^{r_{13}^* \cdot \eta} \cdot g^{r_{14}^* \cdot \nu \cdot H(RL\_PK_1^*)} \cdot g^{r_{14}^* \cdot \lambda} \cdot g_2^{r_{15}^* \cdot \alpha \cdot H(\mathfrak{M}_1^* \| \mathfrak{E}^*)} \cdot g^{r_{15}^* \cdot H(\mathfrak{M}_1^* \| \mathfrak{E}^*)} \cdot g^{r_{15}^* \cdot \pi}}$$
$$= \frac{g_2^y \cdot \vartheta^{(r_{12}^*+a) \cdot H(pk^*)} \cdot \psi^{r_{12}^*+a} \cdot \varpi^{r_{13}^*} \cdot \mu^{r_{14}^* \cdot H(RL\_PK_1^*)} \cdot \tau^{r_{14}^*} \cdot \chi^{r_{15}^* \cdot H(\mathfrak{M}_1^* \| \mathfrak{E}^*)} \cdot \kappa^{r_{15}^*}}{g_2^y \cdot g_2^{r_{12}^* \cdot \ell \cdot H(pk^*)} \cdot g^{r_{12}^* \cdot H(pk^*)} \cdot pk^{*H(pk^*)} \cdot g^{r_{12}^* \cdot \partial} \cdot pk^{*\partial} \cdot g^{r_{13}^* \cdot \eta} \cdot g^{r_{14}^* \cdot \nu \cdot H(RL\_PK_1^*)} \cdot g^{r_{14}^* \cdot \lambda} \cdot g_2^{r_{15}^* \cdot \alpha \cdot H(\mathfrak{M}_1^* \| \mathfrak{E}^*)} \cdot g^{r_{15}^* \cdot H(\mathfrak{M}_1^* \| \mathfrak{E}^*)} \cdot g^{r_{15}^* \cdot \pi}}$$
$$= g_2^{\ell \cdot a \cdot H(pk^*)}.$$

Further, we can compute $\sqrt[\ell \cdot H(pk^*)]{g_2^{\ell \cdot a \cdot H(pk^*)}} = g_2^a = g^{a \cdot b}$, which is the solution to the given CDH problem.

Similarly, we can get the probability of $\mathcal{B}$ not aborting,

$$\Pr(not\_abort) = \left(1 - \frac{q_g}{q}\right) \cdot \left(1 - \frac{q_s}{q}\right)^2.$$

So, we can get that $\varepsilon' = (1 - \frac{q_g}{q}) \cdot (1 - \frac{q_s}{q})^2 \cdot \varepsilon$ and the time complexity of the algorithm $\mathcal{B}$,

$$\hbar' = \hbar + O(q_g \cdot (5 \cdot C_{exp} + 3 \cdot C_{mul}) + q_s \cdot (12 \cdot C_{exp} + 7 \cdot C_{mul})).$$

Thus, Theorem 7.2 follows.

(**Proof of Theorem 7.3**).

Proof: Let **TRS** be a traceable ring signature scheme of Section 6. Additionally, let $\mathcal{A}$ be an ($\hbar$, ε, q_g, q_s)-adversary attacking **TRS**. From the adversary $\mathcal{A}$, we construct an algorithm $\mathcal{B}$, for (g, g^a, g^b)$\in \mathbb{G}_1$, the algorithm $\mathcal{B}$ is able to use $\mathcal{A}$ to compute $g^{a \cdot b}$. Thus, we assume the algorithm $\mathcal{B}$ can solve the CDH with probability at least ε' and in time at most $\hbar'$, contradicting the ($\hbar'$, ε')-CDH assumption. Such a simulation may be created in the following way:

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

Then the system parameters are generated as follows. The algorithm chooses $y \in \mathbb{Z}_q$, and then sets $g_1 = g^y$ and $g_2 = g^b$ with $b \in \mathbb{Z}_q$ ($\mathcal{B}$ doesn't know b). Also, the algorithm chooses ℓ, ∂, ν, λ, η, α and $\pi \in \mathbb{Z}_q$, and then sets $\vartheta = g_2^{\ell} \cdot g$, $\psi = g^{\partial}$, $\mu = g^{\nu}$, $\tau = g^{\lambda}$, $\chi = g_2^{\alpha} \cdot g$, $\kappa = g^{\pi}$ and $\varpi = g^{\eta}$. Finally, the system outputs the public parameters TRK=($\mathbb{G}_1$, $\mathbb{G}_2$, e, g, g_1, g_2, ϑ, ψ, μ, τ, χ, κ, ϖ) to $\mathcal{A}$.

Additionally, the user u^* is a challenger whose public key is pk^*, we set that the public key of u^*, $pk^* = g^a$ ($\mathcal{B}$ doesn't know a).

Queries: When running the adversary $\mathcal{A}$, the relevant queries can occur according to the Definition 5.3. The algorithm $\mathcal{B}$ answers these in the following way:

- **Generate-Public Key Queries**: Given the public parameters TRK, the algorithm $\mathcal{B}$ randomly chooses $r_1, r_2 \in \mathbb{Z}_q$, computes the public key $pk = g^{r_2}$ of the ring member u and the corresponding private key $sk = \{g_2^y \cdot \varpi^{r_1} \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2}, g^{r_1}\}$, and then outputs the public key pk to $\mathcal{A}$, which is added to the public key ring RL_PK, where RL_PK is a public key list including all public keys of the ring members belong to this ring. To finish the above computation, we assure that $H(pk) \neq 0$ **mod** q.
- **Sign Queries**: Given the public parameters TRK, the public key list RL_PK ($pk \in RL\_PK$ where pk is the public key of the ring member that belongs to this ring), the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the algorithm $\mathcal{B}$ chooses random $r_2, r_3, r_4, r_5 \in \mathbb{Z}_q$ and computes

$$\sigma_0 = g_2^y \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5},$$
$$\sigma_1' = g^{r_2},$$
$$\sigma_2 = g^{r_3},$$
$$\sigma_3 = g^{r_4},$$
$$\sigma_4 = g^{r_5}.$$

Finally, the algorithm outputs a forgery $\Phi = \{\sigma_0, \sigma_1', \sigma_2, \sigma_3, \sigma_4\}$ to the adversary $\mathcal{A}$, where we maximize the adversary's advantage, σ_1' is passed to $\mathcal{A}$. Thus, $\Phi' = \{\sigma_0, \sigma_1 = e(\vartheta^{H(pk)} \cdot \psi, \sigma_1'), \sigma_2, \sigma_3, \sigma_4\}$ is a valid signature, where we assure that $H(pk) \neq 0$ **mod** q and $H(\mathfrak{M} \parallel \mathfrak{E}) \neq 0$ **mod** q.

Forgery: If the algorithm $\mathcal{B}$ does not abort as a consequence of one of the queries above, the adversary $\mathcal{A}$ will, with probability at least ε, return its forgeries, ($\mathfrak{M}^*$, $\mathfrak{E}^*$, Φ^*, RL_PK^*) for the challenger u^*, with $pk^* \in RL\_PK^*$, $\Phi^* = \{\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, \sigma_4^*, \sigma_5^*, \sigma_6^*\}$, where

$$\sigma_0^* = g_2^y \cdot \vartheta^{(r_2^*+a) \cdot H(pk^*)} \cdot \psi^{r_2^*+a} \cdot \varpi^{r_3^*} \cdot \mu^{r_4^* \cdot H(RL\_PK^*)} \cdot \tau^{r_4^*} \cdot \chi^{r_5^* \cdot H(\mathfrak{M}_1^* \| \mathfrak{E}^*)} \cdot \kappa^{r_5^*},$$
$$\sigma_1^* = g^{r_2^*},$$
$$\sigma_2^* = g^{r_3^*},$$
$$\sigma_3^* = g^{r_4^*},$$
$$\sigma_4^* = g^{r_5^*},$$
$$\sigma_5^* = g_2^{r_2^*},$$
$$\sigma_6^* = g_2^{r_5^*}.$$

Remark: In fact, σ_1^* should be equal to $g^{r_2^*} \cdot pk^*$. Similarly, because the adversary $\mathcal{A}$ can compute $\sigma_1^* = g^{r_2^*}$ and $\sigma_4^* = g^{r_5^*}$, $\mathcal{A}$ can easily convert these computations to $\sigma_5^* = g_2^{r_2^*}$ and $\sigma_6^* = g_2^{r_5^*}$, where σ_5^* and σ_6^* return to the algorithm $\mathcal{B}$ so as to make $\mathcal{B}$ solve the CDH problem.

And the forgeries satisfy the following conditions[8]:

(a) $1 \leftarrow$***Verify***$(TRK,\ RL\_PK^*,\ \mathfrak{M}^*,\ \mathfrak{E}^*,\ \Phi^*)$;

(b) $\mathcal{A}$ did not query ***Sign*** on inputs RL_PK^*, $\mathfrak{M}^*$ and $\mathfrak{E}^*$;

(c) $"Linked" \leftarrow$***Trace-User***$(TRK,\ RL\_PK^*,\ \{\mathfrak{M}^*,\ \Phi^*\},\ \{\mathfrak{M}',\ \Phi'\},\ \mathfrak{E}^*)$, where Φ' is any signature outputted from ***Sign*** on inputs RL_PK^*, $\mathfrak{M}'$ and $\mathfrak{E}^*$.

So, the algorithm $\mathcal{B}$ computes and outputs

$$\frac{\sigma_0^*}{g_2^y \cdot g_2^{r_2^* \cdot \ell \cdot H(pk^*)} \cdot g^{r_2^* \cdot H(pk^*)} \cdot pk^{*H(pk^*)} \cdot g^{r_2^* \cdot \partial} \cdot pk^{*\partial} \cdot g^{r_3^* \cdot \eta} \cdot g^{r_4^* \cdot \nu \cdot H(RL\_PK^*)} \cdot g^{r_4^* \cdot \lambda} \cdot g_2^{r_5^* \cdot \alpha \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot g^{r_5^* \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot g^{r_5^* \cdot \pi}}$$
$$= \frac{g_2^y \cdot \vartheta^{(r_2^*+a) \cdot H(pk^*)} \cdot \psi^{r_2^*+a} \cdot \varpi^{r_3^*} \cdot \mu^{r_4^* \cdot H(RL\_PK^*)} \cdot \tau^{r_4^*} \cdot \chi^{r_5^* \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot \kappa^{r_5^*}}{g_2^y \cdot g_2^{r_2^* \cdot \ell \cdot H(pk^*)} \cdot g^{r_2^* \cdot H(pk^*)} \cdot pk^{*H(pk^*)} \cdot g^{r_2^* \cdot \partial} \cdot pk^{*\partial} \cdot g^{r_3^* \cdot \eta} \cdot g^{r_4^* \cdot \nu \cdot H(RL\_PK^*)} \cdot g^{r_4^* \cdot \lambda} \cdot g_2^{r_5^* \cdot \alpha \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot g^{r_5^* \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot g^{r_5^* \cdot \pi}}$$
$$= g_2^{\ell \cdot a \cdot H(pk^*)}.$$

Further, we can compute $\sqrt[\ell \cdot H(pk^*)]{g_2^{\ell \cdot a \cdot H(pk^*)}} = g_2^a = g^{a \cdot b}$, which is the solution to the given CDH problem.

Similarly, we can get the probability of $\mathcal{B}$ not aborting,

$$\Pr(not\_abort) = \left(1 - \frac{q_g}{q}\right) \cdot \left(1 - \frac{q_s}{q}\right)^2.$$

So, we can get that $\varepsilon' = (1 - \frac{q_g}{q}) \cdot (1 - \frac{q_s}{q})^2 \cdot \varepsilon$ and the time complexity of the algorithm $\mathcal{B}$,

$$\hbar' = \hbar + O(q_g \cdot (5 \cdot C_{exp} + 3 \cdot C_{mul}) + q_s \cdot (12 \cdot C_{exp} + 7 \cdot C_{mul})).$$

Thus, Theorem 7.3 follows.

(**Proof of Theorem 7.4**).

Proof: Let **TRS** be a traceable ring signature scheme of Section 6. Additionally, let $\mathcal{A}$ be an ($\hbar$, ε, q_g, q_s)-adversary attacking **TRS**. From the adversary $\mathcal{A}$, we construct an algorithm $\mathcal{B}$, for $(g,\ g^{a_1},\ g^b) \in \mathbb{G}_1$ or $(g,\ g^{a_2},\ g^b) \in \mathbb{G}_1$, the algorithm $\mathcal{B}$ is able to use $\mathcal{A}$ to compute $g^{a_1 \cdot b}$ or $g^{a_2 \cdot b}$. Thus, we assume the algorithm $\mathcal{B}$ can solve the CDH with probability at least ε' and in time at most $\hbar'$, contradicting the ($\hbar'$, ε')-CDH assumption. Such a simulation may be created in the following way:

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

[8] σ_5^* and σ_6^* do not anticipate in these following computations.

Then the system parameters are generated as follows. The algorithm chooses $y \in \mathbb{Z}_q$, and then sets $g_1 = g^y$ and $g_2 = g^b$ with $b \in \mathbb{Z}_q$ ($\mathcal{B}$ doesn't know b). Also, the algorithm chooses ℓ, ∂, ν, λ, η, α and $\pi \in \mathbb{Z}_q$, and then sets $\vartheta = g_2^{\ell} \cdot g$, $\psi = g^{\partial}$, $\mu = g^{\nu}$, $\tau = g^{\lambda}$, $\chi = g_2^{\alpha} \cdot g$, $\kappa = g^{\pi}$ and $\varpi = g^{\eta}$. Finally, the system outputs the public parameters TRK=($\mathbb{G}_1$, $\mathbb{G}_2$, e, g, g_1, g_2, ϑ, ψ, μ, τ, χ, κ, ϖ) to $\mathcal{A}$.

Additionally, we assume the users u_1^* and u_2^* are two challengers, whose public keys respectively are pk_1^*, and pk_2^*. We set that the public key of u_1^*, $pk_1^* = g^{a_1}$ ($\mathcal{B}$ doesn't know a_1), and that the public key of u_2^*, $pk_2^* = g^{a_2}$ ($\mathcal{B}$ doesn't know a_2).

Queries Phase 1: When running the adversary $\mathcal{A}$, the relevant queries can occur according to the Definition 5.4. The algorithm $\mathcal{B}$ answers these in the following way:

- **Generate-Public Key Queries**: Given the public parameters TRK, the algorithm $\mathcal{B}$ randomly chooses $r_1, r_2 \in \mathbb{Z}_q$, computes the public key $pk = g^{r_2}$ of the ring member u and the corresponding private key $sk = \{g_2^y \cdot \varpi^{r_1} \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2}, g^{r_1}\}$, and then outputs the public key pk to $\mathcal{A}$, which is added to the public key ring RL_PK, where RL_PK is a public key list including all public keys of the ring members belong to this ring. To finish the above computation, we assure that $H(pk) \neq 0$ **mod** q.
- **Sign Queries**: Given the public parameters TRK, the public key list RL_PK ($pk \in RL\_PK$ where pk is the public key of the ring member that belongs to this ring), the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the algorithm $\mathcal{B}$ chooses random $r_2, r_3, r_4, r_5 \in \mathbb{Z}_q$ and computes

$$\sigma_0 = g_2^y \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5},$$
$$\sigma_1' = g^{r_2},$$
$$\sigma_2 = g^{r_3},$$
$$\sigma_3 = g^{r_4},$$
$$\sigma_4 = g^{r_5}.$$

  Finally, the algorithm outputs a forgery $\Phi = \{\sigma_0, \sigma_1', \sigma_2, \sigma_3, \sigma_4\}$ to the adversary $\mathcal{A}$, where we maximize the adversary's advantage, σ_1' is passed to $\mathcal{A}$. Thus, $\Phi' = \{\sigma_0, \sigma_1 = e(\vartheta^{H(pk)} \cdot \psi, \sigma_1'), \sigma_2, \sigma_3, \sigma_4\}$ is a valid signature, where we assure that $H(pk) \neq 0$ **mod** q and $H(\mathfrak{M} \parallel \mathfrak{E}) \neq 0$ **mod** q.

Challenge: $\mathcal{A}$ sends to the challengers its forgeries, the public keys pk_1^* and pk_2^* and the tuple ($\mathfrak{M}^*$, $\mathfrak{E}^*$, $RL\_PK^* \cup \{pk_1^*\} \cup \{pk_2^*\}$). The forgeries satisfy the condition that $\mathcal{A}$ did not query ***Sign*** on input pk_1^* (and pk_2^*).

The challengers pick a random $x \in \{1, 2\}$, and then run $\Phi^{*'} = \{\sigma_0^*, \sigma_1^{*'}, \sigma_2^*, \sigma_3^*, \sigma_4^*\} \leftarrow$ ***Sign Queries***(TRK, $RL\_PK^{*'}$, $\mathfrak{M}^*$, $\mathfrak{E}^*$), where $RL\_PK^{*'} = RL\_PK^* \cup \{pk_1^*\} \cup \{pk_2^*\}$ and

$$\sigma_0^* = g_2^y \cdot \vartheta^{(r_2^*+a_x)\cdot H(pk_x^*)} \cdot \psi^{r_2^*+a_x} \cdot \varpi^{r_3^*} \cdot \mu^{r_4^* \cdot H(RL\_PK^{*'})} \cdot \tau^{r_4^*} \cdot \chi^{r_5^* \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot \kappa^{r_5^*},$$
$$\sigma_1^{*'} = g^{r_2^*},$$
$$\sigma_2^* = g^{r_3^*},$$
$$\sigma_3^* = g^{r_4^*},$$
$$\sigma_4^* = g^{r_5^*}.$$

And then the challengers output $\Phi^* = \{\sigma_0^*, \sigma_1^* = e(\vartheta^{H(pk_x^*)} \cdot \psi, \sigma_1^{*'} \cdot pk_x^*), \sigma_2^*, \sigma_3^*, \sigma_4^*\}$ to $\mathcal{A}$.

Queries Phase 2: When running the adversary $\mathcal{A}$, the relevant queries can occur according to the Definition 5.4. The algorithm $\mathcal{B}$ answers these in the following way:

- **Generate-Public Key Queries**: Given the public parameters TRK, the algorithm $\mathcal{B}$ randomly chooses $r_1, r_2 \in \mathbb{Z}_q$, computes the public key $pk = g^{r_2}$ of the ring member u and the corresponding private key $sk = \{g_2^y \cdot \varpi^{r_1} \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2}, g^{r_1}\}$, and then outputs the public key pk to $\mathcal{A}$, which is added to the public key ring RL_PK, where RL_PK is a public key list including all public keys of the ring members belong to this ring. To finish the above computation, we assure that $H(pk) \neq 0$ **mod** q.
- **Sign Queries**: Given the public parameters TRK, the public key list RL_PK ($pk \in RL\_PK$ where pk is the public key of the ring member that belongs to this ring), the message $\mathfrak{M}$ and the event identifier $\mathfrak{E}$, the algorithm $\mathcal{B}$ chooses random $r_2, r_3, r_4, r_5 \in \mathbb{Z}_q$ and computes
$$\sigma_0 = g_2^y \cdot \vartheta^{r_2 \cdot H(pk)} \cdot \psi^{r_2} \cdot \varpi^{r_3} \cdot \mu^{r_4 \cdot H(RL\_PK)} \cdot \tau^{r_4} \cdot \chi^{r_5 \cdot H(\mathfrak{M} \| \mathfrak{E})} \cdot \kappa^{r_5},$$
$$\sigma_1' = g^{r_2},$$
$$\sigma_2 = g^{r_3},$$
$$\sigma_3 = g^{r_4},$$
$$\sigma_4 = g^{r_5}.$$
Finally, the algorithm outputs a forgery $\Phi = \{\sigma_0, \sigma_1', \sigma_2, \sigma_3, \sigma_4\}$ to the adversary $\mathcal{A}$, where we maximize the adversary's advantage, σ_1' is passed to $\mathcal{A}$ and $\mathcal{A}$ did not query ***Sign*** on inputs pk_1^* and $\mathfrak{E}^*$ (and pk_2^* and $\mathfrak{E}^*$). Thus, $\Phi' = \{\sigma_0, \sigma_1 = e(\vartheta^{H(pk)} \cdot \psi, \sigma_1'), \sigma_2, \sigma_3, \sigma_4\}$ is a valid signature, where we assure that $H(pk) \neq 0$ **mod** q and $H(\mathfrak{M} \parallel \mathfrak{E}) \neq 0$ **mod** q.

Guess: If the algorithm $\mathcal{B}$ does not abort as a consequence of one of the queries above, the adversary $\mathcal{A}$ will, with probability at least ε ($\varepsilon \geq \frac{1}{2}$), output $x' \in \{1, 2\}$ and succeed ($x' = x$). We assume that $\Phi^{*'} = \{\sigma_0^{*'}, \sigma_1^{*''}, \sigma_2^{*'}, \sigma_3^{*'}, \sigma_4^{*'}, \sigma_5^{*'}, \sigma_6^{*'}\}$, where $RL\_PK^{*'} = RL\_PK^* \cup \{pk_1^*\} \cup \{pk_2^*\}$ and
$$\sigma_0^{*'} = g_2^y \cdot \vartheta^{(r_2^*+a_{x'})\cdot H(pk_{x'}^*)} \cdot \psi^{r_2^*+a_{x'}} \cdot \varpi^{r_3^*} \cdot \mu^{r_4^* \cdot H(RL\_PK^{*'})} \cdot \tau^{r_4^*} \cdot \chi^{r_5^* \cdot H(\mathfrak{M}^* \| \mathfrak{E}^*)} \cdot \kappa^{r_5^*},$$
$$\sigma_1^{*''} = g^{r_2^*},$$

$$\begin{aligned}
\sigma_2^{*'} &= g^{r_3^*},\\
\sigma_3^{*'} &= g^{r_4^*},\\
\sigma_4^{*'} &= g^{r_5^*},\\
\sigma_5^{*'} &= g_2^{r_2^*},\\
\sigma_6^{*'} &= g_2^{r_5^*}.
\end{aligned}$$

Similarly, if $x' = x$, we can get the followings:

$$\frac{\sigma_0^{*'}}{g_2^y \cdot g_2^{r_2^* \cdot \ell \cdot H(pk^*_{x'})} \cdot g^{r_2^* \cdot H(pk^*_{x'})} \cdot pk^{*\,H(pk^*_{x'})}_{x'} \cdot g^{r_2^* \cdot \partial} \cdot pk^{*\,\partial}_{x'} \cdot g^{r_3^* \cdot \eta} \cdot g^{r_4^* \cdot \nu \cdot H(RL\_PK^{*'})} \cdot g^{r_4^* \cdot \lambda} \cdot g_2^{r_5^* \cdot \alpha \cdot H(\mathfrak{M}^* \parallel \mathfrak{E}^*)} \cdot g^{r_5^* \cdot H(\mathfrak{M}^* \parallel \mathfrak{E}^*)} \cdot g^{r_5^* \cdot \pi}}$$
$$= \frac{g_2^y \cdot \vartheta^{(r_2^* + a_{x'}) \cdot H(pk^*_{x'})} \cdot \psi^{r_2^* + a_{x'}} \cdot \varpi^{r_3^*} \cdot \mu^{r_4^* \cdot H(RL\_PK^{*'})} \cdot \tau^{r_4^*} \cdot \chi^{r_5^* \cdot H(\mathfrak{M}^* \parallel \mathfrak{E}^*)} \cdot \kappa^{r_5^*}}{g_2^y \cdot g_2^{r_2^* \cdot \ell \cdot H(pk^*_{x'})} \cdot g^{r_2^* \cdot H(pk^*_{x'})} \cdot pk^{*\,H(pk^*_{x'})}_{x'} \cdot g^{r_2^* \cdot \partial} \cdot pk^{*\,\partial}_{x'} \cdot g^{r_3^* \cdot \eta} \cdot g^{r_4^* \cdot \nu \cdot H(RL\_PK^{*'})} \cdot g^{r_4^* \cdot \lambda} \cdot g_2^{r_5^* \cdot \alpha \cdot H(\mathfrak{M}^* \parallel \mathfrak{E}^*)} \cdot g^{r_5^* \cdot H(\mathfrak{M}^* \parallel \mathfrak{E}^*)} \cdot g^{r_5^* \cdot \pi}}$$
$$= g_2^{\ell \cdot a_{x'} \cdot H(pk^*_{x'})}.$$

Further, we can compute $\sqrt[\ell \cdot H(pk^*_{x'})]{g_2^{\ell \cdot a_{x'} \cdot H(pk^*_{x'})}} = g_2^{a_{x'}} = g^{a_{x'} \cdot b}$, which is the solution to the given CDH problem.

Now, we analyze the probability of the algorithm $\mathcal{B}$ not aborting. For the simulation to complete without aborting, we require that all ***Generate-Public Key*** queries will have $H(pk) \neq 0$ **mod** q, all ***Sign*** queries will have $H(pk) \neq 0$ **mod** q and $H(\mathfrak{M} \parallel \mathfrak{E}) \neq 0$ **mod** q in the Queries Phase 1 and 2. If the algorithm $\mathcal{B}$ does not abort, then the following conditions must hold:

(a) $H(pk_i) \neq 0$ **mod** q in ***Generate-Public Key*** queries, with i=1, 2......q_{g_1};

(b) $H(pk_i) \neq 0$ **mod** q and $H(\mathfrak{M}_i \parallel \mathfrak{E}_i) \neq 0$ **mod** q in ***Sign*** queries, with i=1, 2......q_{s_1};

(c) $H(pk_i) \neq 0$ **mod** q in ***Generate-Public Key*** queries, with i=1, 2......q_{g_2};

(d) $H(pk_i) \neq 0$ **mod** q and $H(\mathfrak{M}_i \parallel \mathfrak{E}_i) \neq 0$ **mod** q in ***Sign*** queries, with i=1, 2......q_{s_2}.

To make the analysis simpler, we will define the events E_{1_i}, F_{1_i}, T_{1_i} E_{2_i}, F_{2_i}, T_{2_i} as

E_{1_i} :$\ell \cdot H(pk_i) \neq 0$ **mod** q, with i=1, 2......q_{g_1};

F_{1_i} :$\ell \cdot H(pk_i) \neq 0$ **mod** q, with i=1, 2......q_{s_1};

T_{1_i} :$\alpha \cdot H(\mathfrak{M}_i \parallel \mathfrak{E}_i) \neq 0$ **mod** q, with i=1, 2......q_{s_1};

E_{2_i} :$\ell \cdot H(pk_i) \neq 0$ **mod** q, with i=1, 2......q_{g_2};

F_{2_i} :$\ell \cdot H(pk_i) \neq 0$ **mod** q, with i=1, 2......q_{s_2};

T_{2_i} :$\alpha \cdot H(\mathfrak{M}_i \parallel \mathfrak{E}_i) \neq 0$ **mod** q, with i=1, 2......q_{s_2}.

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

$$\Pr(not\_abort) = \Pr\left(\bigcap_{i=1}^{q_{g_1}} E_{1_i} \wedge \bigcap_{i=1}^{q_{s_1}} (F_{1_i} \wedge T_{1_i}) \wedge \bigcap_{i=1}^{q_{g_2}} E_{2_i} \wedge \bigcap_{i=1}^{q_{s_2}} (F_{2_i} \wedge T_{2_i})\right).$$

It is easy to see that the events $\bigcap_{i=1}^{q_{g_1}} E_{1_i}$, $\bigcap_{i=1}^{q_{s_1}} F_{1_i}$, $\bigcap_{i=1}^{q_{s_1}} T_{1_i}$, $\bigcap_{i=1}^{q_{g_2}} E_{2_i}$, $\bigcap_{i=1}^{q_{s_2}} F_{2_i}$, $\bigcap_{i=1}^{q_{s_2}} T_{2_i}$ are independent. Then we may compute

$$\Pr(\bigcap_{i=1}^{q_{g_1}} E_{1_i}) = 1 - \Pr(\bigcup_{i=1}^{q_{g_1}} \neg E_{1_i}) = 1 - q_{g_1} \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_{g_1}}{q};$$

$$\Pr(\bigcap_{i=1}^{q_{s_1}} F_{1_i}) = 1 - \Pr(\bigcup_{i=1}^{q_{s_1}} \neg F_{1_i}) = 1 - q_{s_1} \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_{s_1}}{q};$$

$$\Pr(\bigcap_{i=1}^{q_{s_1}} T_{1_i}) = 1 - \Pr(\bigcup_{i=1}^{q_{s_1}} \neg T_{1_i}) = 1 - q_{s_1} \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_{s_1}}{q};$$

$$\Pr(\bigcap_{i=1}^{q_{g_2}} E_{2_i}) = 1 - \Pr(\bigcup_{i=1}^{q_{g_2}} \neg E_{2_i}) = 1 - q_{g_2} \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_{g_2}}{q};$$

$$\Pr(\bigcap_{i=1}^{q_{s_2}} F_{2_i}) = 1 - \Pr(\bigcup_{i=1}^{q_{s_2}} \neg F_{2_i}) = 1 - q_{s_2} \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_{s_2}}{q};$$

$$\Pr(\bigcap_{i=1}^{q_{s_2}} T_{2_i}) = 1 - \Pr(\bigcup_{i=1}^{q_{s_2}} \neg T_{2_i}) = 1 - q_{s_2} \cdot \frac{1^k}{1^k \cdot q} = 1 - \frac{q_{s_2}}{q}.$$

So,

$$\begin{aligned}\Pr(\mathit{not\_abort}) &= \Pr\left(\bigcap_{i=1}^{q_{g_1}} E_{1_i} \wedge \bigcap_{i=1}^{q_{s_1}} (F_{1_i} \wedge T_{1_i}) \wedge \bigcap_{i=1}^{q_{g_2}} E_{2_i} \wedge \bigcap_{i=1}^{q_{s_2}} (F_{2_i} \wedge T_{2_i})\right) \\ &= \Pr(\bigcap_{i=1}^{q_{g_1}} E_{1_i}) \cdot \Pr(\bigcap_{i=1}^{q_{s_1}} F_{1_i}) \cdot \Pr(\bigcap_{i=1}^{q_{s_1}} T_{1_i}) \cdot \Pr(\bigcap_{i=1}^{q_{g_2}} E_{2_i}) \cdot \Pr(\bigcap_{i=1}^{q_{s_2}} F_{2_i}) \cdot \Pr(\bigcap_{i=1}^{q_{s_2}} T_{2_i}) \\ &= \left(1 - \frac{q_{g_1}}{q}\right) \cdot \left(1 - \frac{q_{s_1}}{q}\right)^2 \cdot \left(1 - \frac{q_{g_2}}{q}\right) \cdot \left(1 - \frac{q_{s_2}}{q}\right)^2.\end{aligned}$$

We can get that $\varepsilon' = (1 - \frac{q_{g_1}}{q}) \cdot (1 - \frac{q_{s_1}}{q})^2 \cdot (1 - \frac{q_{g_2}}{q}) \cdot (1 - \frac{q_{s_2}}{q})^2 \cdot \varepsilon$ and the time complexity of the algorithm $\mathcal{B}$,

$$\hbar' = \hbar + O\left((q_{g_1} + q_{g_2}) \cdot (5 \cdot C_{exp} + 3 \cdot C_{mul}) + (q_{s_1} + q_{s_2}) \cdot (12 \cdot C_{exp} + 7 \cdot C_{mul})\right).$$

Thus, Theorem 7.4 follows.