An Efficient Dynamic Group Signature with Non-frameability

Run Xie[1,2] , Chunxiang Xu[1], Chanlian He[2], Xiaojun Zhang[3],
1.School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China
[e-mail: xrryun@126.com]
2. Yibin university, Yibin, 644000, China
3. School of Computer Science, Southwest Petroleum University, 610500, Chengdu, China
*Corresponding author: Run Xie

Received November 17, 2015; revised February 28, 2016; accepted March 26, 2016; published May 31, 2016

Abstract

A group signature scheme allows any member to sign on behalf of a group. It is applied to practical distributed security communication environments, such as privacy-preserving, data mining. In particular, the excellent features of group signatures, including membership joining and revocation, anonymity, traceability, non-frameability and controllable linkability, make group signature scheme more attractive. Among these features, non-frameability can guarantee that a member's signature cannot be forged by any other (including issuer), and controllable linkability supports to confirm whether or not two group signatures are created by the same signer while preserving anonymity. Until now, only Hwang et al.'s group schemes (proposed in 2013 and 2015) can support all of these features. In this paper, we present a new dynamic group signature scheme which can achieve all of the above excellent features. Compared with their schemes, our scheme has the following advantages. Firstly, our scheme achieves more efficient membership revocation, signing and verifying. The cost of update key in our scheme is two-thirds of them. Secondly, the tracing algorithm is simpler, since the signer can be determined without the judging step. Furthermore, in our scheme, the size of group public key and member's private key are shorter. Lastly, we also prove security features of our scheme, such as anonymity, traceability, non-frameability, under a random oracle model.

Keywords: dynamic group signature; non-frameability; anonymity; traceability; security proof

1. Introduction

In traditional digital signature schemes, the identity of signer can be identified with the corresponding the public key. It is hard to protect identity privacy.

In 1991, Chaum et al. introduced the concept of a group signature scheme [1]. A group signature scheme allows any member of a group to sign messages on behalf of a group while his identity is kept secret from the verifier. So a group signature scheme has the *anonymity*. Meanwhile, to prevent abuses, the group is controlled by a group manager that has the ability to open a group signature, i.e. to reveal the identity of the signature's originator (*traceability*). Furthermore, in 2005, Bellare, Shi and Zhang [2-3] extended these notions to dynamic groups and added the notion of non-frameability. In dynamic group scheme, it allows a new user to join group. The non-frameability can guarantee that any one (including the group manager) should not be able to produce a valid group signature on behalf of another member. Obviously, a dynamic group scheme is more flexible and efficient in practical application. In addition, Hwang et al. introduce the notion of the controllable linkability. The controllable linkability allows an entity who holds a linking key to determine whether two group signatures were created by the same signer or not without revealing its identity. These security features of group signature schemes make it attractive for various applications, such as, privacy-preserving, data mining, anonymous online communications and so on.

With the different functionalities and levels of efficiency, a variety of group signature schemes have been presented in the last two decades [4-13]. However, very few schemes allows one to add members to the group and revoke membership with time, while they satisfy all security features, including anonymity, traceability, non-frameability. For example, Boneh et al. [14] presented a group signature system using bilinear maps. Their scheme not only provides the very short signature but also allows group manager to revoke members. However, their scheme requires the trusted party to generate the private key in the party performing setup. Since the trusted party knows the signing keys of all members, the non-frameability is not guaranteed. Furthermore, if the trusted party be removed after the setup, a new user can not be added to group. To resolve this issue, Delerable et al. proposed dynamic group signature scheme [15]. In this scheme, based on XDH assumptions, the non-frameability is achieved and the size of signature is shorter. However, using their open key, the opening algorithm is non-available when key update by revocation is considered. Obviously, this issue limits their scheme practical application. Others, such as, Benoit Libert et al. [18] presented scalable revocable group signature schemes. In these schemes, a new user can't be added to group and the size of signature is still quite big compared with short group signatures [14, 15].

Very recently, Hwang [16, 17] proposed a short dynamic group signature scheme. Their scheme allows one to add members to the group and revoke membership, while it satisfy the anonymity, traceability and non-frameability. Furthermore, their scheme support a very useful functionality, namely the controllable linkability. The controllable linkability allows an entity who holds a linking key to determine whether two group signatures were created by the same signer or not without revealing its identity. Moreover, in their scheme, the signature length is very short. Therefore, their scheme is brilliant. However, in [16], the opening algorithm can't determine the signer's identity since issuer may provide more than one members with a "y". Therefore, in their scheme, the judging is indispensable to determine the signer's identity. In addition, a limitation of their scheme is that the cost of update key is

expensive. Furthermore, the signature of upk should have been added to registry, otherwise the members of group may be framed by an issuer.

Therefore, how to design more efficient and flexible group signature scheme supporting anonymity, traceability, non-frameability and controllable linkability is significance research work which has obtained more attention.

1.1 Our Contributions

Motivated by the issues discussed above, we present an new dynamic group signature scheme. It allows a new user to join the group and supports membership revocation. Compared with previous related works, our scheme has several advantages as follows.

Firstly, our scheme can achieve many desirable features, such as user dynamic joining and revocation, anonymity, traceability, especially non-frameability and controllable linkability. To the best of our knowledge, only few schemes [16, 17] can achieve these features. Therefore, our scheme is more flexible solution in practical application.

Secondly, our scheme is more effective. Compared with [16, 17], our scheme have lower cost of signing, verifying and updating-key. Particularly, in our scheme, to update his own key, an unrevoked member only needs to compute one multi-exponentiation with two bases while one multi-exponentiation with four bases in their scheme when one user is revoked. In addition, in our scheme, both the size of group public key and member's private key are shorter than that [16].

Lastly, in our scheme, the opening algorithm can determine the identity of signer without judging while preserving the non-frameability. In [16], the judging is indispensable, otherwise a member may be framed by the issuer. Therefore, our scheme is simpler to track the identity of signer.

2. Preliminaries

2.1 Dynamic Group Signature Schemes

The model of dynamic group signature was introduced by Bellare et al. [3] (for short BSZ). We describe a slight variant of dynamic group signature in a form suitable for our paper, since our scheme can support membership revocation that is not a part of the BSZ model.

Here, suppose there exists a fully trusted authority (example PKI) separated from the group environment. Both a user U_i and the group manager (*GM*) have obtained their certificate of keys from a fully trusted authority before setting up the system. U_i's certificate is denoted by ($PK_{UID}[i]$, $SK_{UID}[i]$), and $PK_{UID}[i]$ stored in the registration table *Reg*. The *GM* certificate is denoted by (PK_{GID}, SK_{GID}). In addition, the group manager and users are involved in our dynamic group signature without the trusted party. Specifically, our dynamic group signature scheme (**DGS**) consists of a tuple polynomial-time algorithms **DGS**=(**Gkg**, **Join**, **Gsig**, **Gvf**, **Open**, **Grev**). There algorithms are defined as follows [3]:

Gkg: Given security parameter *n*, it generates *Gpk* and *Gmsk*, where *Gpk* and *Gmsk* are the public key and the private key of *GM*, respectively.

Join: By this algorithm, U_i can co-ordinating his private key Usk_i with *GM*. Finally, the group manager adds an $Reg[i]$ corresponding to U_i to the registration table *Reg*.

Gsig: Given a message *M*, it takes the *Gpk* and U_i's privacy key Usk_i and outputs a signature σ on a message *M*.

Gvf: This algorithm takes the *Gpk*, message *M* and the signature σ corresponding to *M* as input, output accept if the signature comes from a legitimate member of the group and reject otherwise.

Open: This takes the valid signature σ, *Gmsk* and the *Reg* as input, and outputs the identity of σ's signer.

Grev: It takes the *Gpk*, a *Reg*[*i*] and *Gmsk* as input, and generates revocation list *RL*. Unrevoked members update their private keys with *RL*. So, a group membership can be selectively disabled without affecting the signing ability of unrevoked members.

2.2 Notions of Correctness and Security

According to the model of Bellare et al. in [3], a dynamic group signature scheme must satisfy the correctness and three security requirements: anonymity, traceability and non-frameability. Now, we show these definitions with slightly differences for suitable our scheme. In the following experiments, the $\mathcal{F}$ is polynomial-time adversary and all of oracles are specified in [3] (including *AddU*(.), *CrptU*(.), *SndTol*(.), *SndToU*(.), *USK*(.), *RReg*(.), *WReg*(.), *GSig*(.), *Ch*(.), *Open*(.)). Let HU be the lists of honest, CU be the lists of corrupted users, GSet be the set of message-signature pairs.

1. Correctness To **DGS**, any $\mathcal{F}$, and security parameter *n*, the correctness experiment $Exp_{\mathcal{F}}^{cor}(n,N)$ is as follows.

(*Gpk*, *Gmsk*) $\leftarrow$ **Gkg**(*n*, *N*); HU $\leftarrow \phi$; CU $\leftarrow \phi$; (*i*, *M*) $\leftarrow \mathcal{F}$ (*Gpk*: *AddU*(.), *RReg*(.));
If $i \notin$ HU, *return* 0; *If* $Usk_i = \varepsilon$, *return* 0;
$\sigma_i \leftarrow$ **Gsig**(*M*, *Gpk*, *Usk*[*i*]); *If* **Gvf**(σ_i, *M*, *Gpk*)=0, *return* 1;
$j \leftarrow$ **Open**(*Gpk*, *Gmsk*, *Reg*, *M*, σ_i) *and* $j \neq i$, *return* 1;

Let $Adv_{\mathcal{F}}^{cor} = Pr[Exp_{\mathcal{F}}^{cor}(n,N)=1]$, then we claim that the **DGS** is correct if Adv_F^{cor}=0.

2. Anonymity To any $\mathcal{F}$, security parameter *n* and $b \leftarrow \{0,1\}$, the anonymity experiment $Exp_{\mathcal{F}}^{an-b}(n,N)$ is as follows.

(*Gpk*, *Gmsk*) $\leftarrow$ Gkg(*n*, *N*); HU $\leftarrow \phi$; CU $\leftarrow \phi$; GSet $\leftarrow \phi$;
d $\leftarrow \mathcal{F}$ (*Gpk*, *Gmsk*(*issue key*): $Ch_b(M^*, i_0, i_1)$, *SndToU*(.), *WReg*(.), *USK*(.), *CrptU*(.))
Return d;

Let $Adv_F^{an} = Pr[\,Exp_F^{an-1}(n,N)=1\,] - Pr[\,Exp_F^{an-0}(n,N)=1\,]$, then we claim that the DGS is *CPA*-anonymous if $Adv_F^{an}(\cdot)$ is negligible function.

3. Traceability To any $\mathcal{F}$, and security parameter *n*, the traceability experiment $Exp_{\mathcal{F}}^{tr}(n,N)$ is as follows.

(*Gpk*, *Gmsk*) $\leftarrow$ **Gkg**(*n*, *N*); HU $\leftarrow \phi$, CU $\leftarrow \phi$;
(M^*, σ_b) $\leftarrow \mathcal{F}$ (*Gpk*, *Gmsk*(*open key*): *SndTol*(.), *AddU*(.), *RReg*(.), *USK*(.), *CrptU*(.));
If **Gvf**(M^*, σ^*)=0, *return* 0;
If **Open**(*Gmsk*, M^*, σ^*) *failed then return* 1, *else return 0*;

Let $Adv_{\mathcal{F}}^{tr} = Pr[Exp_{\mathcal{F}}^{tr}(n,N)=1]$, then we claim that the **DGS** is traceable if $Adv_{\mathcal{F}}^{tr}(.)$ is negligible function.

4. Non-frameability To any $\mathcal{F}$, and security parameter *n*, the non-frameability experiment $Exp_{\mathcal{F}}^{nf}(n,N)$ is as follows.

(*Gpk*, *Gmsk*, *Usk*) $\leftarrow$ **Gkg**(*n*, *N*); HU $\leftarrow \phi$, CU $\leftarrow \phi$;
(M, σ_i) $\leftarrow \mathcal{F}$ (*Gpk*, *Gmsk*: *SndToU*(.), *WReg*(.),*GSig*(.) , *CrptU*(.),*USK*(.));
If **Gvf**(*Gpk*, *M*, σ_i)=0, *return* 0;

If **Open**(*Gpk*, *Gmsk*, *M*, σ_i)=*i* ($i \in \{1,\ldots, N\}$) *and* $\mathcal{F}$ *did not query Usk*[i] *or* **Gsig**(*i*, *M*) *then return* 1 *else return* 0.

Let $Adv_{\mathcal{F}}^{nf} = Pr[Exp_{\mathcal{F}}^{nf}(n, N) = 1]$, then we claim that the **DGS** is non-frameability if $Adv_{\mathcal{F}}^{nf}(.)$ is negligible function.

2.2. Bilinear Map on Group

Let G_1, G_2 , G_T be multiplicative cyclic groups of prime order *p*. The groups are defined in certain families of nonsupersingular elliptic curves which are defined by Miyaji et al. in [19]. In G_1, the size of the elements is 171-bit, and that discrete log on G_1 has identical difficulty with discrete log in Z_q where *q* is 1020 bits. Let $g_1 \in G_1$ be generator, while $g_2 \in G_2$ be generator. Let $\varphi(g_2) = g_1$, where $\varphi : G_2 \to G_1$ is a computable isomorphism.

Next, we review the notion of bilinear maps. Let $e : G_1 \times G_2 \to G_T$ be a map on G_1, G_2 , G_T with three properties.

(1) Bilinearity: $\forall\ \eta \in G_1$, $\forall\ \pi \in G_2$ and *a*, $b \in Z$, $e(\eta^a, \pi^b) = e(\eta, \pi)^{ab}$.

(2) Non-degeneracy: $e(g_1, g_2) \neq 1_{G_T}$.

(3) Computability: for all $\eta \in G_1$, $\pi \in G_2$, then a $e(\eta, \pi)$ can be efficient computed.

In the next content, we always suppose that (G_1, G_2) are a bilinear group pair as above.

2.3. The Main Assumptions

Let g_1, g_2, G_1 , G_2 be defined as above, where possibly G_1=G_2. Now, we review the *q*-SDH problem. Consider the following problem: The *q*-SDH problem [14] in (G_1, G_2) is that given (*q*+2)-tuple (g_1, g_2, g_2^{γ}, $g_2^{\gamma^2}, \ldots, g_2^{\gamma^q}$) , find a pair $\left(g_1^{1/(x+\gamma)}, x\right)$, where $x \in Z_p^*$.

The *q*-SDH Assumption is defined as a difficult problem to be solved *q*-SDH problem.

As a natural extension of *q*-SDH problem, we introduce the extended *q*-SDH problem and new complexity problem. Consider the following problems:

The Extended *q*-SDH Problem (*q*-eSDH). Given tuple $\left(g_1, g_2, g_2^{\gamma}, g_2^{(\gamma^2)}, \ldots, g_2^{(\gamma^q)}, g_2^{\beta}\right)$, find a tuple $\left(g_1^{\frac{y+\beta}{x+\gamma}}, x, y\right)$, where $x, y \in Z_p^*$ and $\beta = \gamma^k$ ($k \neq 0,1$).

Formally, an algorithm $\mathcal{F}$ has advantage ε in solving *q*-eSDH problem if

$$Pr\left[\mathcal{F}\left(k, g_1, g_2, g_2^{\gamma}, g_2^{(\gamma^2)}, \ldots, g_2^{(\gamma^q)}, g_2^{\beta}\right) = \left(g_1^{\frac{y+\beta}{x+\gamma}}, x, y\right)\right] \geq \varepsilon$$

where the probability is over the random choices of generator $g_2 \in G_2$ (with $\varphi(g_2) \to g_1$), of $\gamma, \beta \in Z_p^*$ ($\beta = \gamma^k, k \neq 0,1$) and of the random bits of $\mathcal{F}$.

Proof: Obviously, if an algorithm $\mathcal{F}$ can be used to solve *q*-eSDH problem, two *q*-eSDH tuples $\left(g_1^{\frac{y+\beta}{x+\gamma}}, x, y\right)$ and $\left(g_1^{\frac{y'+\beta}{x+\gamma}}, x, y'\right)$ can be obtained. According to the two tuples, we can compute: $\left[\left(g_1^{\frac{y+\beta}{x+\gamma}}\right) / \left(g_1^{\frac{y'+\beta}{x+\gamma}}\right)\right]^{(y-y')^{-1}} = g_1^{\frac{1}{x+\gamma}}$. So $\mathcal{F}$ can obtain a *q*-SDH tuple $\left(g_1^{1/(x+\gamma)}, x\right)$.

Definition 1. *The* (q, t, ε)*-eSDH assumption holds in* (G_1, G_2) *states if no* t*-time algorithm has at least* ε *advantage in solving the q-eSDH problem.*

New Complexity Problem (*q*-NC) Take g_1, g_2, G_1 , G_2 as above, the *q*-NC problem is

defined as given tuple $\left(g_1, g_2, g_2^{\gamma}, g_2^{(\gamma^2)}, \ldots, g_2^{(\gamma^q)}, x, y, g_1^{\frac{y+\beta}{x+\gamma}}\right)$, compute $\left(g_1^{\frac{y}{x+\gamma}}\right)$ or $\left(g_1^{\frac{\beta}{x+\gamma}}\right)$, where $x, y \in Z_p^*$.

An algorithm $\mathcal{F}$ has ε advantage in solving q-NC problem if

$$Pr\left[\mathcal{F}\left(g_1, g_2, g_2^{\gamma}, g_2^{\gamma^2}, \ldots, g_2^{\gamma^q}, x, y, g_1^{\frac{y+\beta}{x+\gamma}}\right) = g_1^{\frac{y}{x+\gamma}}\right] \geq \varepsilon$$

where the probability is over the random choices of generator $g_2 \in G_2$, of $\gamma, \beta \in Z_p^*$.

Remark: 1. we emphasize that γ and β are unknown to the algorithm $\mathcal{F}$.

2. According to q-SDH assumption, $g_1^{\frac{1}{x+\gamma}}$ cannot be obtained from x, g_1, g_2, g_2^{γ}, $g_2^{(\gamma^2)}$,..., $g_2^{(\gamma^q)}$, so $g_1^{\frac{y}{x+\gamma}}$ cannot be obtained by $\left(g_1^{\frac{1}{x+\gamma}}\right)^y$. So the q-NC assumption is reasonable model to consider.

Definition 2. *The (q,t,ε)-NC assumption holds in (G_1, G_2) states if no t-time algorithm has at least ε advantage in solving the q-NC problem.*

Lemma 1 *If the q-NC assumption holds, given tuple $\left(g_1, g_2, g_2^{\gamma}, g_2^{\gamma^2}, \ldots, g_2^{\gamma^q}, x, y, g_1^{\frac{y+\beta}{x+\gamma}}\right)$ for any $y'(\neq y) \in Z_p^*$, it is difficult to compute $g_1^{\frac{y'+\beta}{x+\gamma}}$.*

Proof: Given the known condition, if an algorithm $\mathcal{F}$ is able to compute $g_1^{\frac{y'+\beta}{x+\gamma}}$ for some $y'(\neq y) \in Z_p^*$, then we obtain $\left(g_1^{\frac{y+\beta}{x+\gamma}}\right) \cdot \left(g_1^{\frac{y'+\beta}{x+\gamma}}\right)^{-1} = \left(g_1^{\frac{y-y'}{x+\gamma}}\right)$. So, $\left(g_1^{\frac{y-y'}{x+\gamma}}\right)^{(y-y')^{-1}} = \left(g_1^{\frac{1}{x+\gamma}}\right)$. As a result, $\left(g_1^{\frac{y}{x+\gamma}}\right)$ is calculated.

The Decision Linear Diffie-Hellman Assumption (LA) [14] Let G_1 be defined as above. Let $\eta, \pi, \tau \in G_1$ be arbitrary generators of G_1 and a, b, $c \in Z_p$, the **LA** holds is that given η, π, τ, η^a, π^b, $\tau^c \in G_1$, it is difficult to decide whether $a+b$ is equal to c. More formally, based on the coin tosses, an algorithm $\mathcal{F}$ chooses uniformly random the parameters. Then, the advantage of probability of $\mathcal{F}$ in deciding a decision linear problem on G_1 is $Adv = |Adv_1 - Adv_2|$, where $Adv_1 = Pr[(\eta, \pi, \tau\ \eta^a,\ \pi^b\ \tau^{a+b}) = yes; \eta, \pi, \tau \leftarrow G_1,\ a, b \leftarrow Z_p]$ and $Adv_2 = Pr[(\eta, \pi, \tau\ \eta^a,\ \pi^b, \mu) = yes; \eta, \pi,\ \mu \leftarrow G_1,\ a, b \leftarrow Z_p]$.

Definition 3. *The (t,ε)-**LA** holds if no t-time algorithm has at least ε advantage in solving decision linear problem.*

Linear Encryption (LE)[14] Based on the **LA**, the **LE** scheme can be obtained. Specifically, take G_1's generators η, π, τ as the public key, then the corresponding private key is $\lambda_1, \lambda_2 \in Z_p^*$ such that $\eta^{\lambda_1} = \pi^{\lambda_2} = \tau$. If $\mathcal{F}$ wishes to send a message $M \in G_1$ to $\mathcal{B}$, then randomly selects $\xi_1, \xi_2 \in Z_p$, computes $\eta^{\xi_1}, \pi^{\xi_2}, M \cdot \tau^{\xi_1+\xi_2}$, and outputs the ciphertext (C_1, C_2, C_3)=($\eta^{\xi_1}, \pi^{\xi_2}, M \cdot \tau^{\xi_1+\xi_2}$) where η, π, τ are $\mathcal{B}$'s public key. To decrypt the ciphertext, $\mathcal{B}$ computes $M = C_3 / \left(C_1^{\lambda_1} \cdot C_2^{\lambda_2}\right)$ with his private key λ_1, λ_2. Similar to prove ElGamal's security, we can show that if the **LA** holds, **LE** is also against a chosen-plaintext attack.

3. A ZK Protocol For eSDH

A zero-knowledge (ZK) protocol is a proof protocol with the following qualities.

The first, a verifier, after having been convinced the validity of what is proved, cannot have learned the knowledge possessed by the prover in order to conduct the proof; The second, after the protocol terminates, any other third party cannot see any meaningful thing which has taken place between the prover and the verifier. The ZK protocol is described as follows.

Let (***P***, ***V***) be ZK protocol, where ***P*** is a prover and ***V*** is a verifier. In the general process of interactive zero-knowledge proof, the ***P*** commits some values to the ***V***. After the ***V*** received, a challenge value be sent to the ***P***. The last, the ***V*** verifies the values responded by ***P*** to decide whether ***P*** possesses some knowledge.

In group signature scheme, a verifier has been convinced the validity of signature without learning the identity of signer. Therefore, the process for verifying group signature is essentially a process of zero-knowledge proof.

3.1 ZKP-eSDH

In this section, based on *q*-eSDH problem, we construct a zero-knowledge protocol (ZKP-eSDH).

Let ***P*** be a prover, and ***V*** be a verifier. In this protocol, ***P*** can prove that he possesses a solution to the *q*-eSDH problem to ***V***.

Setup: Generate the public values $g_1, \eta, \pi, \tau \in G_1$ and $g_2, \omega_1, \omega_2 \in G_2$. Here g_2 is a random generator of G_2 such that $\varphi(g_2) = g_1$, $\omega_1 = g_2^{\gamma}$, $\omega_2 = g_2^{\beta}$ for some $\gamma, \beta \in Z_p$. η, π, τ are randomly selected in G_1. A prover ***P*** possesses a tuple (R, x, y), where $R = g^{\frac{y+\beta}{x+\gamma}}$ and $x, y \in Z_p$ such that $e(R, \omega_1 g_2^{x}) \cdot e(g_1, g_2^{-y}) = e(g_1, \omega_2)$.

P **Commit**: ***P*** chooses ξ_1, ξ_2 and r_{ξ}, r_x, r_y, r_{δ_1}, r_{δ_2} at random from Z_p. Take (R, x, y) as input, ***P*** computes the following values:

$$C_1 = \eta^{\xi_1},\ C_2 = \pi^{\xi_2},\ C_3 = R \cdot \tau^{\xi_1+\xi_2},\ D_1 = C_1^{r_x} \eta^{-r_{\delta_1}},\ D_2 = C_2^{r_x} \pi^{-r_{\delta_2}},$$

$$D_3 = e(C_3, g_2)^{r_x} \cdot e(\tau, \omega_1)^{r_{\xi}} \cdot e(\tau, g_2)^{-r_{\delta_1}-r_{\delta_2}} \cdot e(g_1, g_2)^{r_y}$$

P sends (C_1, C_2, C_3, D_1, D_2, D_3) to verifier ***V***.

V **Challenge**: ***V*** sends a challenge value c chosen uniformly at random from Z_p.

P **Reply**: ***P*** computes $v_{\xi} = r_{\xi} - c(\xi_1 + \xi_2)$, $v_x = r_x + cx$, $v_y = r_y - cy$, $v_{\delta_1} = r_{\delta_1} + cx\xi_1$, $v_{\delta_2} = r_{\delta_2} + cx\xi_2$ and replies $\mathcal{B}$ with these values.

V **Verify**: ***V*** computes the following five values: $D'_1 = C_1^{v_x} \eta^{-v_{\delta_1}}$, $D'_2 = C_2^{v_x} \pi^{-v_{\delta_2}}$, $D'_3 = e(C_3, g_2)^{v_x} \cdot e(\tau, \omega_1)^{v_{\xi}} \cdot e(\tau, g_2)^{-v_{\delta_1}-v_{\delta_2}} \cdot e(g_1, g_2)^{v_y} \cdot e(C_3, \omega_1)^{c} \cdot e(g_1, \omega_2)^{-c}$.

V accepts if $D_i = D'_i$ holds (i=1,2,3).

Next, we show that ZKP-eSDH has following features:

(1) If ***P*** is honest, then he can always be accepted by ***V*** (completeness);

(2) The prover's transcript can be simulated (zero-knowledge);

(3) There exists an extractor for protocol.

Feature 1. *The ZKP-eSDH is complete*.

Proof: If ***V*** is honest, he can correctly recover D_i=D'_i (i=1,2,3) with the challenge value c and $v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2}$. Specifically, ***V*** can computes the values:

$D'_1 = C_1^{v_x}\eta^{-v_{\delta_1}} = \eta^{\xi_1 r_x + cx\xi_1}\eta^{-r_{\delta_1} - cx\xi_1} = D_1$, $D'_2 = C_2^{v_x}\pi^{-v_{\delta_2}} = \pi^{\xi_2 r_x + cx\xi_2}\pi^{-r_{\delta_2} - cx\xi_2} = D_2$

In addition, he can also compute:

$$D'_3 = e(C_3, g_2)^{v_x} \cdot e(\tau, \omega_1)^{v_\xi} \cdot e(\tau, g_2)^{-v_{\delta_1} - v_{\delta_2}} \cdot e(g_1, g_2)^{v_y} \cdot e(C_3, \omega_1)^{c} \cdot e(g_1, \omega_2)^{-c}$$

Because $D_3 = e(C_3, g_2)^{r_x} \cdot e(\tau, \omega_1)^{r_\xi} \cdot e(\tau, g_2)^{-r_{\delta_1} - r_{\delta_2}} \cdot e(g_1, g_2)^{r_y}$, and

$$1_{G_T} = e(C_3, g_2)^{cx} \cdot e(\tau, \omega_1)^{-c(\xi_1+\xi_2)} \cdot e(\tau, g_2)^{-cx(\xi_1+\xi_2)} \cdot e(g_1, g_2)^{-cy} \cdot e(C_3, \omega_1)^{c} \cdot e(g_1, \omega_2)^{-c}$$

So $D'_3 = D_3$, the completeness of ZKP-eSDH is proved.

Feature 2. *If the **LA** holds and verifier V is honest , then the transcripts of ZKP-eSDH can be perfectly simulated.*

Proof: Suppose, ***X*** is the distribution of the transcript output by the simulator. If the simulator succeeds, then the ***X*** is indistinguishable from the distribution of the transcript output by any actual prover under the **LA** on G_1.

Firstly, the simulator randomly selects $R \in G_1$ and $\xi_1, \xi_2 \in Z_p^*$ and computes $C_1 = \eta^{\xi_1}$, $C_2 = \pi^{\xi_2}$, $C_3 = R \cdot \tau^{\xi_1+\xi_2}$. In this simulation, to compute D_i=D'_i (i=1,2,3), the simulator does not need to know R, x, y, ξ_1, ξ_2, so the pre-specified (C_1, C_2, C_3) can also be used. By the method of simulation of proof transcript, when (C_1, C_2, C_3) are the ciphertext **LE** of some R, the simulator can obtain simulation of the rest of proof transcript. After a challenge value c and $v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2}$ are randomly selected from Z_p, the simulator compute the values:

$D_1 = C_1^{v_x}\eta^{-v_{\delta_1}}$, $D_2 = C_2^{v_x}\pi^{-v_{\delta_2}}$,

$D_3 = e(C_3, g_2)^{v_x} \cdot e(\tau, \omega_1)^{v_\xi} \cdot e(\tau, g_2)^{-v_{\delta_1} - v_{\delta_2}} \cdot e(g_1, g_2)^{v_y} \cdot e(C_3, \omega_1)^{c} \cdot e(g_1, \omega_2)^{-c}$

So, the simulator can output (C_1, C_2, C_3, D_1, D_2, D_3, $v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2}$). Obviously, these values satisfy verification equations and are indistinguishable with the proof transcripts of ZKP-eSDH on distribution if the **LA** holds.

Feature 3. *Extraction is feasible for ZKP-eSDH*.

Proof: Now we show that the extractor can obtain a tuple q-eSDH if $\mathcal{F}$ can be rewind in ZKP-eSDH above to the point before he receives a challenge c. At the beginning of the protocol, A sends (C_1, C_2, C_3, D_1, D_2, D_3) to $\mathcal{B}$. Then, to challenge c, $\mathcal{F}$ replies with ($v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2}$). In this way, to challenge $c' \neq c$, $\mathcal{F}$ replies with ($v'_\xi, v'_x, v'_y, v'_{\delta_1}, v'_{\delta_2}$). If $\mathcal{F}$ is convincing, all verification equations D_i (i=1, 2, 3) hold for each set of values. For convenience, denote $\Delta c = c - c'$, $\Delta v_\xi = v_\xi - v'_\xi$, and similarly for $\Delta v_\xi, \Delta v_x, \Delta v_y, \Delta v_{\delta_1}, \Delta v_{\delta_2}$. Let $\xi' = \frac{\Delta v_\xi}{\Delta c}$, Since the values of D_1 and D_2 remain the same, the following equations are obtained: $C_1^{\Delta v_x} = \eta^{\Delta v_{\delta_1}}$, $C_2^{\Delta v_x} = \pi^{\Delta v_{\delta_2}}$. Let $x' = \frac{\Delta v_x}{\Delta c}$, $y' = \frac{\Delta v_y}{\Delta c}$, from D_3 expression, we can deduce:

$$e(g_1, \omega_2) = e(C_3, g_2)^{\frac{\Delta v_x}{\Delta c}} \cdot e(\tau, \omega_1)^{\frac{\Delta v_\xi}{\Delta c}} \cdot e(\tau, g_2)^{-\frac{\Delta v_{\delta_1} + \Delta v_{\delta_2}}{\Delta c}} \cdot e(g_1, g_2)^{\frac{\Delta v_y}{\Delta c}} \cdot e(C_3, \omega_1)$$

$$e(g_1, \omega_2) = e\left(C_3 \tau^{-\xi'}, \omega_1 g_2^{x'}\right) \cdot e\left(g_1, g_2^{y'}\right)$$

$(R, x', y') = \left(C_3 \tau^{-\xi'}, \frac{\Delta v_x}{\Delta c}, \frac{\Delta v_y}{\Delta c}\right)$ is a q-eSDH tuple.

Since (C_1, C_2, C_3, D_1, D_2, D_3) are equal to the challenge value $c' \neq c$. In fact, we already know: $\Delta v_\xi = \Delta r_\xi + \Delta c\xi$, $\Delta v_x = \Delta r_x + \Delta cx$, $\Delta v_y = \Delta r_y + \Delta cy$.

For the same $C_i(i=1,2,3)$ and $D_i(i=1,2,3)$ in two running, it is easy to see that $\Delta r_\xi = 0$, $\Delta r_x = 0$, $\Delta r_y = 0$, then $\xi' = \xi$, $x' = x$, $y' = y$.

As a result, the R' in this q-eSDH tuple is identical to R in the **LE** (C_1, C_2, C_3).

Putting the proof of three properties together, then the following theorem has been proved.

Theorem *The ZKP-eSDH is an honest-verifier zero-knowledge proof of knowledge of an eSDH tuple under the* ***LA****.*

4. Our Dynamic Group Signature Scheme from eSDH

4.1 Our Signature Scheme from eSDH

In this section, we propose a new dynamic group signature scheme. Roughly speaking, in our scheme, a signer who possess a valid q-eSDH tuple can generate a group signature which are the transcript of ZKP-eSDH protocol. Therefore, armed with theorem, we obtain from ZKP-eSDH protocol a regular signature scheme secure in the random oracle model by applying the Fiat-Shamir heuristic [20]. In our scheme, a hash function $H:\{0,1\}^* \to Z_p$ be employed. Suppose further the **LA** assumption holds on G_1, the q-eSDH assumption and the q-NC assumption hold on (G_1, G_2). Now we describe our group signature as follows. To simplify, our group signature scheme is denoted by eSDH-DGSS.

Setup: Takes φ and (G_1, G_2) as the section 2, let $H:\{0,1\}^* \to Z_p$ is an ideal hash function, treated as a random oracle in the proof of security.

Gkg: Randomly selected a generator $g_2 \in G_2$, $\gamma, \beta \leftarrow Z_p^*$ ($\beta = \gamma^k$, secret $k \neq 0,1$), let $g_1 = \varphi(g_2)$. GM selects private values $\lambda_1, \lambda_2 \leftarrow Z_p^*$ and computes $\omega_1 = g_2^\gamma$, $\omega_2 = g_2^\beta$. Let $\tau \leftarrow G_1 \setminus \{1_{G_1}\}$, η, $\pi \in G_1$, such that $\eta^{\lambda_1} = \pi^{\lambda_2} = \tau$. This algorithm generates $Gpk=(g_1, g_2, \eta, \pi, \tau, \omega_1, \omega_2)$ and $Gmsk=(\gamma, \lambda_1, \lambda_2, \zeta_0)$. Here ζ_0 is assumed initially to be 1.

Join: U_i can be added to the group after running this algorithm between GM and U_i. The GM add an item $Reg[i]$ to the Reg.

1. Selecting a secret value $y_i \in Z_p^*$, U_i computes $g_1^{y_i}$ and sends $g_1^{y_i}$ to GM.
2. GM selects $x_i \in Z_p$ and computes $\left(g_1^{y_i} \cdot \omega_2'\right)^{\frac{1}{x+\gamma}}$. He sends $g_1^{\frac{y+\beta}{x+\gamma}}$ to U_i, where $\omega_2' = \varphi(\omega_2)$.
3. U_i sends S_i to GM, where S_i is the signature of $g_1^{\frac{y+\beta}{x+\gamma}}$ with $SK_{UID}[i]$.
4. Using $PK_{UID}[i]$, GM verifies S_i. If S_i is a valid signature, then GM sends x_i to U_i.

As a result, U_i obtains his private key $Usk_i=(R_i, x_i, y_i)=(g_1^{\frac{y+\beta}{x+\gamma}}, x_i, y_i)$. GM adds an $Reg[i]=(R_i, g_1^{y_i}, x_i, S_i, PK_{UID}[i])$ which is corresponding to U_i to the registration table Reg.

Gsig: Given message M, U_i chooses random $\xi_1, \xi_2, r_\xi, r_x, r_y, r_{\delta_1}, r_{\delta_2}$ from Z_p and computes the following values with his private key $Usk_i=(R_i, x_i, y_i)$: $C_1=\eta^{\xi_1}$, $C_2=\pi^{\xi_2}$, $C_3=R_i \cdot \tau^{\xi_1+\xi_2}$, $D_1=C_1^{r_x}\eta^{-r_{\delta_1}}$, $D_2=C_2^{r_x}\pi^{-r_{\delta_2}}$, $D_3=e(C_3,g_2)^{r_x} \cdot e(\tau,\omega_1)^{r_\xi} \cdot e(\tau,g_2)^{-r_{\delta_1}-r_{\delta_2}} \cdot e(g_1,g_2)^{r_y}$, and the value of hash $c=H(M, C_1, C_2, C_3, D_1, D_2, D_3)$. He also computes:

$v_\xi = r_\xi - c(\xi_1 + \xi_2)$, $v_x = r_x + cx_i$, $v_y = r_y - cy_i$, $v_{\delta_1} = r_{\delta_1} + cx_i\xi_1$, $v_{\delta_2} = r_{\delta_2} + cx_i\xi_2$.

At the end, U_i generates a group signature: σ =(C_1, C_2, C_3, $c, v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2}$), and sends ($\sigma$, M) to the verifier.

Gvf: Given Gpk and (σ, M), a verifier verify that σ is a valid signature as follows:

$D'_1 = C_1^{v_x}\eta^{-v_{\delta_1}}$, $D'_2 = C_2^{v_x}\pi^{-v_{\delta_2}}$,

$D'_3 = e(C_3, g_2)^{v_x} \cdot e(\tau, \omega_1)^{v_\xi} \cdot e(\tau, g_2)^{-v_{\delta_1} - v_{\delta_2}} \cdot e(g_1, g_2)^{v_y} \cdot e(C_3, \omega_1)^{c} \cdot e(g_1, \omega_2)^{-c}$

$c' = H(M, C_1, C_2, C_3, D'_1, D'_2, D'_3)$

If c' =c accepts and reject otherwise.

Open: To trace a signature's signer, GM takes the signature of M σ =(C_1, C_2, C_3, $c, v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2}$), together with Gpk and $Gmsk$ as input, and do the following:

1. Verify the signature σ is correct on M.

2. Take elements (C_1, C_2, C_3) of σ, and obtain someone R_i from $R_i \leftarrow \left[C_3 / \left(C_1^{\lambda_1} \cdot C_2^{\lambda_2}\right)\right]^{\xi_0^{-1}}$. GM finds the index i with respect to R_i in the registration table Reg.

3. If the user index i is corresponding to $Reg[i]$=(R_i, $g_1^{y_i}$, x_i, S_i, $PK_{UID}[i]$) in Reg and the validity of S_i on R_i is validated by GM, then the algorithm claims that the group member with identity i produced σ. It provides a proof of this claim with GM which verifies the validity of signature S_i on R_i with $PK_{UID}[i]$.

4.2 Revocation

Now we give procedure that users can be revoked by the GM in the eSDH group signature above. Given Gpk=(g_1, g_2, η, π, τ, ω_1, ω_2), where $\omega_1 = g_2^\gamma$, $\omega_2 = g_2^\beta \in G_2$, $\gamma, \beta \in Z_p^*$, $\eta, \pi, \tau \in G_1$. The private key of U_i is a tuple Usk_i=(R_i, x_i, y_i).

Update Key If GM have revoked users $j_1, \ldots, j_r$ that are corresponding to their index. The revocation algorithm consists of three sub-algorithms, **Update-*Gmsk***, **Update-*Gpk*** and **Update-*Usk*** which update a group public key and a user signature key, respectively.

Update-*Gmsk*: Updates $Gmsk=(\gamma, \lambda_1, \lambda_2, \zeta_0) \rightarrow (\gamma, \lambda_1, \lambda_2, \zeta_r)$, where $\zeta_r = \prod_{m=1}^{r}\left(\frac{1}{x_{j_m} + \gamma}\right)$.

Update-*Gpk*: Let initial Gpk=(g_1, g_2, η, π, τ, ω_1, ω_2). Using new $Gmsk=(\gamma, \lambda_1, \lambda_2, \zeta_r)$, GM updates Gpk to Gpk_r=(g_{1r}, g_{2r}, η_r, π_r, τ_r, ω_{1r}, ω_{2r}), where $g_{1r} = g_1^{\zeta_r}$, $g_{2r} = g_2^{\zeta_r}$, $\eta_r = \eta$, $\pi_r = \pi$, $\tau_r = \tau$, $\omega_{1r} = \omega_1^{\zeta_r} = g_{2r}^{\gamma}$, $\omega_{2r} = \omega_2^{\zeta_r} = g_{2r}^{\beta}$.

GM generate the revocation list RL that containing all revoked users (let in order of revoked). Specifically, RL is defined as ((g_{11}, R'_{j1}, R''_{j1}, x_{j1}) , ... , (g_{1r}, R'_{jr}, R''_{jr}, x_{jr})), Where $g_{1k} = g_1^{\zeta_k}$, $R'_{jk} = \left[(\varphi\ (\omega_2))^{\zeta_k}\right]^{\frac{-1}{x_{jk}+\gamma}} = g_{1k}^{\frac{\beta}{x_{jk}+\gamma}}$, $R''_{jr} = g_1^{\frac{\zeta_k}{x_{jk}+\gamma}} = g_{1k}^{\frac{1}{x_{jk}+\gamma}}$.

For (k=1,…,r), GM publishes the RL and the new public key. The RL and the new public key are given to all signers and verifiers in the system

Update-*Usk*: Given tuple (Gpk, RL), U_i is able to obtain his new Usk corresponding to the new Gpk_r locally if he has not been revoked. However, revoked users cannot do so.

Let U_i's private key Usk_i=(R_i, x_i, y_i)=$\left(g_{1r}^{\frac{y_i+\beta}{x_i+\gamma}}, x_i, y_i\right)$, U_i compute:

$$R_{i1} = \left(\frac{R_{i0}}{R'_{j1} \cdot \left(R''_{j1}\right)^{y_i}}\right)^{(x_{j1}-x_i)^{-1}} = R_{l_0}^{(x_i - x_{j1})^{-1}} \left(R'_{j_1}\right)^{-(x_i - x_{j1})^{-1}} \left(R''_{j_1}\right)^{-y(x_i - x_{j1})^{-1}} = \left(g_{11}\right)^{\frac{y_i+\beta}{x_i+\gamma}}.$$

Let $\alpha_k = \prod_{m=k}^{r}\left(x_i - x_{j_m}\right)^{-1} \bmod p$, then $R_{ir}= R_{i_0}^{\alpha_1}\prod_{k=1}^{r}\left(R'_{j_k}\cdot\left(R''^{y_i}_{j_k}\right)\right)^{(-1)^k\alpha_k}$. U_i can compute his new private key $Usk_i=(R_{ir}, x_i, y_i)=\left((g_{1r})^{\frac{y_i+\beta}{x_i+\gamma}}, x_i, y_i\right)$.

In fact, if a revoked user is able to update his private key, then he can execute update procedure as above. This means that revoked user's x_j is not equal to x_{j_m} (m=1,…,r), contradicting $x_j \in \{x_{j_m}; m=1,\ldots,r\}$.

Update-Open When key update by revocation is considered, *GM* trace a signature's signer as follows. Given updated signature σ_r =(M, C_1, C_2, C_3, c, v_ξ, v_x, v_y, v_{δ_1}, v_{δ_2}), *GM* compute: $R_{ir}\leftarrow C_3/\left(C_1^{\lambda_1}\cdot C_2^{\lambda_2}\right)$, $\left(R_{ir}\right)^{\zeta_r^{-1}}=R_i$. *GM* finds the index i with respect to R_i in the registration table *Reg*. Obviously, only *GM* can generate the correct ζ_r and *RL* in the opening and the revocation mechanism above. So this revocation mechanism can overcome the problem [14] that someone can fool a verifier.

4.3 Obtain Controllable Linkability

The controllable linkability of group signatures enables an entity who has a linking key to find whether or not two group signatures were generated by the same signer, while preserving the anonymity. In our scheme, the controllable linkability can also be easy to achieve as follows. Let $(L_1, L_2)=\left((\omega_2)^{\lambda_1},(\omega_2)^{\lambda_2}\right)$ is linking key. Given two valid pairs of signatures and messages, (σ', M') and (σ'', M''), proceed as follows:

$T_1 = e(C'_3,\omega_2)e(C''_1,L_1)^{-1}e(C''_2,L_2)^{-1}$, $T_2 = e(C''_3,\omega_2)e(C'_1,L_1)^{-1}e(C'_2,L_2)^{-1}$

If T_1=T_2 then output 1, otherwise output 0. Obviously, anyone can verify whether or not two group signatures were generated by the same signer if he holds the linkability key.

5. Our Scheme Security

In this section, we proof the security of ours scheme. Here, we relax the full-anonymity requirement. While we follow a slightly weaker security model CPA-fully-Anonymity given in [14]. Given G_1 and G_2, we know that bilinear map evaluation, exponentiation and sampling are constant-time. To simplify the details of addition terms in size bound or time bounds, we use the big-O notation in the next content. To prove the security of our scheme (eSDH-DGSS) as above, we also need to the forking lemma (see [21]).

Theorem 5.1 *Our scheme (eSDH-DGSS) is correct.*

Proof: Let Gpk=(g_1, g_2, η, π, τ, ω_1, ω_2) be the public key, and Usk_i=(R_i, x_i, y_i) be the user's private key. The $R_i^{\frac{x_i+\gamma}{y_i+\beta}}=g_1$ is always true, so (R_i, x_i, y_i) is an q-eSDH tuple. According ZKP-eSDH protocol, a correct signature σ is a part transcript of the ZKP-eSDH. One can easily show that a correct σ will always pass validation of the verifier. Furthermore, (C_1, C_2, C_3)=(η^{ξ_1}, π^{ξ_2}, $R_i\cdot\tau^{\xi_1+\xi_2}$) in signature σ are a random **LE** of R_i with the public key (η, π, τ), thus these values can always be restored by the *GM* holding the private key (λ_1, λ_2). So each of the valid σ which is output by an honest signer will be correctly opened.

Theorem 5.2 *If the* ***LE*** *is* (t',ε')*-semantically secure on* G_1*, our scheme (eSDH-DGSS) is* (t,q_H,ε)*-CPA-fully-anonymous, where* $t=t'-q_HO(1)$ *and* $\varepsilon=\varepsilon'$*. Here* q_H *is the number of hash function inquiries made by the adversary*

Proof: If an algorithm $\mathcal{F}$ can (t,q_H,ε)-break the anonymity of eSDH-DGSS, we are able to construct an algorithm $\mathcal{B}$ in $t+q_HO(1)$-time that can break the semantic security of **LE** with at least ε. Next, we give the procedure to construct the $\mathcal{B}$ as follows:

Assuming $\mathcal{B}$ possesses an **LE** public key (η,π,τ). Calling the **Gkg**, $\mathcal{B}$ can obtain the $g_1, g_2, \omega_1, \omega_2$. Then $\mathcal{F}$ is equipped with the *Gpk*=($g_1, g_2, \eta, \pi, \tau, \omega_1, \omega_2$) generated by $\mathcal{B}$, while a user is equipped with a Usk_i=(R_i, x_i, y_i). When the hash oracle *H* is asked by $\mathcal{F}$, $\mathcal{B}$ reply with elements chosen uniformly at random from Z_p, ensuring to reply identically when the same inquiry is occurs.

Given a message *M*, i_0 and i_1, $\mathcal{F}$ accepts its fully-anonymity test. While $\mathcal{B}$ accepts its indistinguishability test by offering the **LE** of the two user's private keys R_{i_0} and R_{i_1}. Now, suppose $\mathcal{B}$ is given an **LE** of R_{i_b} (C_1, C_2, C_3), where the **LE** challenger chooses b equal to 0 or 1. Using (C_1, C_2, C_3) , $\mathcal{B}$ generates from this **LE** a complete transcript (C_1, C_2, C_3, D_1, D_2, D_3, c, $v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2}$) with the simulator of feature.2. Because the tuple (C_1, C_2, C_3) are the ciphertext (**LE**) of R_{i_b}, the rest of this transcript is identical distribution with in a real ZKP-eSDH, where a prover's private *R* is R_{i_b}. Obviously, given (C_1, C_2, C_3), even though $\mathcal{B}$ does not know ξ_1, ξ_2, *x* and *y*, the simulator can generate a trace.

Then $\mathcal{B}$ patches *H* at (*M*, C_1, C_2, C_3, D_1, D_2, D_3) to equal *c*. $\mathcal{B}$ proclaims failure and exit if a collision occurred. Otherwise, it returns a correct signature: $\sigma \leftarrow$ (C_1, C_2, C_3, $c, v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2}$) to $\mathcal{B}$. In the light of the definition of *H*, we know that the probability of emerging a collision is negligible.

Finally, a *b*' is output by $\mathcal{F}$. $\mathcal{B}$ replies its own challenge with the *b*'. Since the part (C_1, C_2, C_3) of the group signature is obtained from a random **LE** of R_{i_b} of user i_b, $\mathcal{B}$ replies its own challenge rightly whenever $\mathcal{F}$ does.

As mentioned above, we know that the *Gpk* and the replies to $\mathcal{F}$'s inquiries are all correct and accurately distributed. So $\mathcal{B}$ succeeds in distinguishing the **LE** (C_1, C_2, C_3) with identical advantage ε if $\mathcal{F}$ can succeed to break the anonymity of the σ with advantage ε.

Suppose each hash query can be answered in constant time, and there are at most q_H $\mathcal{F}$'s queries. Then $\mathcal{B}$'s running time exceeds $\mathcal{F}$'s by the amount it takes to answer $\mathcal{F}$'s queries. As result, $\mathcal{B}$ runs in time $t+q_HO(1)$.

CPA-Fully-traceability Follow the fully-traceability [3] security notion, there are two cases breaking fully-traceability. One case is the adversary provides a correct signature σ such that the honest *GM* cannot trace the signer. The other case is the *GM* believes the origin of signature has been identified but he cannot provide a correct proof of its claim.

Suppose, there is a list of tuple (R_i, x_i, y_i) for index *i*=1,..,*n*, where *n* is the number of the members of the group. For each *i*, either (R_i, x_i, y_i) is an eSDH tuple $e(R_i, \omega_1 g_2^{x}) \cdot e(g_1, g_2^{-y}) = e(g_1, \omega_2)$, or else $(x_{i^*}, y_{i^*}) = ?$, indicating that the (x_{i^*}, y_{i^*}) corresponding to R_{i^*} is not known. In accordance with the fully-traceability notion, then an algorithm $\mathcal{F}$ that breaks the full-traceability of the group signature scheme needs to complete the following work. It outputs a forged group signature σ on a message M.

Then $\mathcal{F}$ is successful if σ is trace to some $R^* \notin \{R_1,\ldots, R_n\}$ or $R^* = R_{i^*}$ for i^* with $(x_{i^*}, y_{i^*}) = ?$.

To prove the fully-traceability of eSDH-DGSS, the following theorem needs to be proved. With the same as above, the number of $\mathcal{F}$ inquiries signature oracle is denoted by q_S, while the number of $\mathcal{F}$ inquiries hash oracle is denoted by q_H.

Theorem 5.3. *If both q-NC and q-eSDH are (q,t',ε')-difficult on (G_1,G_2), our scheme(eSDH-DGSS) is $(t,q_H,q_S,n,\varepsilon)$-fully-traceable, where $\varepsilon = 4n\sqrt{2\varepsilon' q_H} + n/p$, $t = \Theta(1)t'$ and $n = q - k_1$ is the number of members of the group.*

Proof: Suppose $\mathcal{F}$ is the algorithm that can break the full-traceability of eSDH-DGSS. The algorithm $\mathcal{B}$ replies $\mathcal{F}$'s inquiries as oracle. $\mathcal{B}$ can obtain a eSDH solution if $\mathcal{F}$ can break the full-traceability.

Firstly, we describe the follow preparatory work needs to be completed by $\mathcal{B}$.

Setup: Given groups (G_1, G_2) as above, a list of tuples (R_i, x_i, y_i) for index i=1,…,n, and generators g_1, g_2 such that $g_1 = \varphi(g_2)$. Also given $\omega_1 = g_2^{\gamma}$, $\omega_2 = g_2^{\beta}$ ($\beta = \gamma^k$), $\mathcal{B}$ picks a generator $\tau \leftarrow G_1 \setminus \{1_{G_1}\}$ and values $\lambda_1, \lambda_2 \in Z_p^*$, then computes η, $\pi \in G_1$ such that $\eta^{\lambda_1} = \pi^{\lambda_2} = \tau$. It generates the polynomial $p(z) = \prod_{i=1}^{n-1}(z + x_i) = \sum_{i=0}^{n-1} \rho_i z^i$ with (R_i, x_i, y_i), where $\rho_0, \rho_1, \ldots, \rho_{n-1} \in Z_p$ are the coefficients of the polynomial $p(z)$.

Given a eSDH instance $\left(g_1, g_2, g_2^{\gamma}, \ldots, g_2^{\gamma^q}, g_2^{\beta}\right)$ on g_1, g_2, $\mathcal{B}$ can compute $\prod_{i=0}^{n-1}\left(g_2^{\gamma^i}\right)^{\rho_i}$.

Let $g_2' = \prod_{i=0}^{n-1}\left(g_2^{\gamma^i}\right)^{\rho_i} = g_2^{p(\lambda)}$, then $\prod_{i=0}^{n-1}\left(g_2^{\gamma^i}\right)^{\rho_{i-1}} = g_2^{\gamma p(\lambda)} = \left(g_2'\right)^{\gamma}$, $\prod_{i=0}^{n+k-1}\left(g_2^{\gamma^i}\right)^{\rho_{(i-k)}} = g_2^{\beta p(\lambda)} = \left(g_2'\right)^{\beta}$.

Moreover, if $(x_{i^*}, y_{i^*}) \neq ?$, then $R'_i = \left(g_1'\right)^{\frac{y_i+\beta}{x_i+\gamma}}$ can be obtained with (R_i, x_i, y_i). Let $t_i(z) = \frac{p(z)}{z + x_i} = \prod_{j=1, j\neq i}^{n-1}(z + x_i) = \sum_{j=0}^{n-2} t_j z^j$, where $t_0, t_1, \ldots, t_{n-2} \in Z_p$ are the coefficient of the polynomial $t_i(z)$. With $t_0, t_1, \ldots, t_{n-2} \in Z_p$, we compute:

$$\left(\prod_{j=0}^{n-2}\varphi\left(g_2^{\gamma_j}\right)^{t_j}\right)^{y_i} \cdot \left(\prod_{j=k}^{n+k-2}\varphi\left(g_2^{\gamma_j}\right)^{t_{j-k}}\right) = \left(g_1^{t_i(\gamma)y_i}\right)\cdot\left(g_1^{t_i(\gamma)\gamma^k}\right) = \left(g_1^{p(\gamma)}\right)^{\frac{y_i+\gamma^k}{x_i+\gamma}} = R'_i$$

For the tuple (R'_i, $(x_{i^*}, y_{i^*}) = ?$) at random index i^* filled with $R_i \leftarrow G_1$ and $(x_{i^*}, y_{i^*}) \leftarrow (*, *)$, where ($*$, $*$) are placeholder values. So, $\mathcal{B}$ obtains a list of distinct new (R'_i, x_i, y_i).

Secondly, we construct a framework for algorithm $\mathcal{B}$ interacting with $\mathcal{F}$ that wins the full-traceability game. Algorithm $\mathcal{B}$ sends $\left(g_1', g_2', (g')^{\gamma}, (g')^{\beta}, \eta, \pi, \tau\right)$ and the open key $\lambda_1, \lambda_2 \in Z_p^*$ to algorithm $\mathcal{F}$. In the process, algorithm $\mathcal{B}$ replies $\mathcal{F}$'s inquiries as random oracle. According to [3], $\mathcal{F}$'s attack capabilities are modeled by accessing the following oracles.

Hash Queries: When $\mathcal{F}$ asks the hash of (M, C_1, C_2, C_3, D_1, D_2, D_3), $\mathcal{B}$ replies with a random element of Z_p, ensuring to reply identically when the same query is occurred.

Signature Queries: Algorithm $\mathcal{F}$ inquires a signature on message M at index i^*.

If $(x_{i^*}, y_{i^*}) = ?$, $\mathcal{B}$ randomly chooses $\xi_1, \xi_2 \in Z_p^*$ and $R_i \in G_1$, let $C_1 = \eta^{\xi_1}$, $C_2 = \pi^{\xi_2}$,

$C_3=R_i \cdot \tau^{\xi_1+\xi_2}$, and calls simulator with the values C_1, C_2, C_3. The simulator returns a transcript ZKP-eSDH (C_1, C_2, C_3, D_1, D_2, D_3, c, v_ξ, v_x, v_y, v_{δ_1}, v_{δ_2}). Then $\mathcal{B}$ obtains a group signature σ =(C_1, C_2, C_3, c, v_ξ, v_x, v_y, v_{δ_1}, v_{δ_2}). In addition, it patches the H at (M, C_1, C_2, C_3, D_1, D_2, D_3) to equal c. If $\mathcal{B}$ previously sets the oracle at this point to some other c', it declares failure and exits. Otherwise, it returns σ to $\mathcal{F}$.

If $(x_{i^*}, y_{i^*}) \neq ?$, $\mathcal{B}$ generates a signature σ on M with key (R'_{i^*}, x_{i^*}, y_{i^*}) by running the signing procedure, and returns σ to $\mathcal{F}$.

Private Key Queries: Algorithm $\mathcal{F}$ asks for the private key of user at some i $(x_{i^*}, y_{i^*}) \neq ?$, $\mathcal{B}$ returns (R'_i ,x_i, y_i) to $\mathcal{F}$. Otherwise, $\mathcal{B}$ declares failure and exits.

Output: At the end, when $\mathcal{F}$ succeeds, it outputs a forged signature σ =(C_1, C_2, C_3, c, v_ξ, v_x, v_y, v_{δ_1}, v_{δ_2}) on a message M. $\mathcal{B}$ open σ with the key (λ_1, λ_2) and obtains someone $R^* \notin \{R'_1,\ldots, R'_n\}$ or $R^*= R'_{i^*}$ and (x_{i^*}, y_{i^*})=? at random index i^*.

1. If $R^* \notin \{R'_1,\ldots, R'_n\}$, then the probability of breaking fully-traceability is the probability of successfully forged the group signature.

2. If $R^*= R'_{i^*}$ and (x_{i^*}, y_{i^*})=?, at random index i^* and $\mathcal{F}$ never queries the private key oracle at i^*, but there exist a forged group signature that traces to R'_{i^*}, then the probability of breaking fully-traceability is equals to ε/n, where ε and $1/n$ are the probability of successfully forged signature and the probability of i^* selected , respectively.

From $\mathcal{F}$'s view, the i^* is independent. With $q_s << q_H << p$, we show the probability that $\mathcal{B}$ fails and $\mathcal{F}$ succeeds is negligible. In simulated signing queries, besides the message M, the hash oracle takes nine elements of bilinear group pair as input, so $(q_s+ q_H)\, q_s / p^9$ is the upper bound of the probability of collision. In fact, it is negligible.

Now we show that how to obtain two related group signatures with the forking lemma [21]. Denote signature as (M , σ_1 , c, σ_2), where σ_1 =(C_1, C_2, C_3, D_1, D_2, D_3) and σ_2 = (v_ξ, v_x, v_y, v_{δ_1}, v_{δ_2}). c is derived from the random oracle H by inputting σ_1 and M. In the above process, let $\mathcal{F}$ be a forger (of either type) with probability ε'.

Next, we show the procedure that $\mathcal{B}$ obtains another eSDH tuple with the results of $\mathcal{F}$'s success, and obtains contradicting the eSDH assumption. Apply the methodology of the forking lemma, we obtain the following procedure.

The interaction between $\mathcal{F}$ and $\mathcal{B}$ is fully depicted by the random string θ used by $\mathcal{F}$ and $\mathcal{B}$, and by the vector Γ of replies made by the oracle of H. Let S be the collection of pairs (θ;h) that enable $\mathcal{B}$ succeed with forgery (M ,σ_1, c,σ_2) by calling $\mathcal{F}$. In the circumstances, let ind(θ;Φ) be the index of Γ at which $\mathcal{F}$ inquiries (M,σ_1). We define $v = \Pr[S] = \varepsilon' - (1/p)$, where the $1/p$ is the probability that $\mathcal{F}$ guessed the hash of (M,σ_1) without any help. As above, denote S_j ($1 \le j \le q_H$) be the collection of pairs (θ;h) such that $ind(\theta;\Gamma)=j$, denote J be the collection consisting of the most likely Index j such that $J = \{ j \,|\, Pr[S_j|S] \} \ge 1/(2q_H)$, then $Pr[ind(\theta,\Gamma)|S] \ge 1/2$.

Let Γ_j denote the restriction of Γ index strictly less than i. Since $Pr[S_i] \ge v/(2q_H)$, there exist a subset Ω_j such that $\forall (\theta;\Gamma) \in \Omega_j$, $Pr_{\Gamma'}[(\theta,\Gamma') \in S_j, \Gamma' = \Gamma] \ge v/(4q_H)$, $Pr[\Gamma_j | S_j] > 1/2$. it shows that $Pr[\exists j \in J : \Omega_j \cap S_j] \ge 1/4$ with a simple reasoning processes.

If $\mathcal{B}$ replays $\mathcal{F}$ many times, with fixed θ but randomly chosen Γ' such that

$\Gamma'_{j_0}=\Gamma_{j_0}$. The $\mathcal{B}$ succeeds and obtains a forgery (M, σ_1, c, σ_2). It derives from Ω_j for some $j\in J$ an execution (θ; Φ) such that $Pr_{\Gamma'}\left[(\theta;\Gamma')\in S_j|\,\Gamma'\|^{j-1}=\Gamma\|^{j-1}\right]\geq v/(4q_H)$.
When $\mathcal{F}$ and $\mathcal{B}$ are rewound to the *j*th inquiry, and go on with an oracle Γ' ($\Gamma'\neq\Gamma$) from the *j*th item, then $\mathcal{B}$ succeeded at a second forgery (M, σ_1, c', σ'_2) with (M, σ_1) still inquiries at $\mathcal{F}$'s *j*-th hash inquiry with probability at least $v/(4q_H)$.

Thirdly, we know that there exists an extractor for two signatures. By using the extractor, $\mathcal{B}$ obtains a *n*-eSDH tuple ($C_3\tau^{-\xi'}$, x', y')=($R_i'^*$, x'_i, y'_i)=($(g'_1)^{\frac{y_i+\beta}{x_i+\gamma}}$, x'_i, y'_i) from (M, σ_1, c, σ_2). and (M, σ_1, c', σ'_2).
According to the **Feature3**, the $R_i'^*$ is identical with the R'_i in the **LE** (C_1, C_2, C_3) in σ_1. $\mathcal{B}$ proclaims success only when the extracted eSDH tuples ($R_i'^*$, x'_i, y'_i) is not amongst those that $\mathcal{B}$ created.
Following by the technique of lemma [21], compute:

$$p(z)=t(z)(z+x'_i)+r \quad , \quad \left(p(z)/(z+x'_i)\right)=t(z)+\left(r/(z+x'_i)\right),\ \ t(z)=\sum_{i=0}^{n-2}t_i z^i$$

Because (x'_i, y'_i)$\neq$(x_i, y_j) for all i, then $x'_i\neq x_i$. Otherwise, $\mathcal{B}$ obtains new tuple $\left(g_1^{\frac{y'_i+\beta}{x_i+\gamma}}, x_i, y'_i\right)$ for some *i*. According to the **lemma1**, this will lead to conflict with the *q*-NC.
So $r\neq 0$, $\mathcal{B}$ can compute: $\left((g'_1)^{\frac{y'_i+\beta}{x+\gamma}}\cdot\left(\prod_{i=0}^{n-2}\varphi\left(g_2^{\gamma^i}\right)^{-t_i}\right)^{y'_i}\cdot\left(\prod_{i=k}^{k+n-2}\varphi\left(g_2^{\gamma^i}\right)^{-t_{(i-k)}}\right)\right)^{1/r}=(g_1)^{\frac{y'_i+\beta}{x'_i+\gamma}}$.
Algorithm $\mathcal{B}$ obtains a new eSDH tuple $\left(g_1^{\frac{y'_i+\beta}{x'_i+\gamma}}, x'_i, y'_i\right)$, contradicting the eSDH assumption. Putting all together, the following claim has been proved.

There exists a $\mathcal{B}$ that solve an instance of n-eSDH with probability $(\varepsilon-1/p)^2/(16q_H)$ or $(\varepsilon/n-1/p)^2/(16q_H)$ in time $O(1)\cdot t$ if the forger $\mathcal{F}$ successfully break the fully-traceable with probability ε.

Next, we prove that our scheme can achieve the non-frameability.

Theorem 5.4. *If the discrete logarithm problem (DLP) is difficult on G_1, our scheme (eSDH-DGSS) is non-frameability*.

Proof: Suppose the error probability **Open** is negligible. We only consider that there exists an algorithm $\mathcal{F}$ that can break the non-frameability of eSDH-DGSS. Now we show how to construct an algorithm $\mathcal{B}$ that breaks the DLP on G_1.

Let (R_i, x_i, y_i) be the private key of user *i*. We show that $\mathcal{B}$ can solve the DLP on G_1 if $\mathcal{F}$ can frame user *i*. $\mathcal{F}$ frames user *i* means that $\mathcal{F}$ can generate a signature which traces to R_i.
Setup: Given private key (R_i, x_i, y_i) ($j\neq i$, j=1,…,n) of members group, the private key of *GM* (γ, λ_1, λ_2) and all user's *Reg*[*j*] (including *Reg*[*j*]), $\mathcal{B}$ responds $\mathcal{F}$'s queries as oracle. $\mathcal{F}$ is given *Gpk*=(g_1, g_2, η, π, τ, ω_1, ω_2). We describe the interactive process between $\mathcal{F}$ and $\mathcal{B}$.

Hash Queries: $\mathcal{F}$ inquires the hash of (M, C_1, C_2, C_3, D_1, D_2, D_3), $\mathcal{B}$ replies with an element random selected from Z_p, ensuring to reply identically when the same inquiry is occurs.
Signature Queries: $\mathcal{F}$ can query the signing oracle by at index j ($j \neq i$).
Private Key Queries: $\mathcal{F}$ queries for Usk_j of the user at index j ($j \neq i$).
***Reg* Queries**: $\mathcal{F}$ queries for the $Reg[j]$ of the user at any index j in the registration table. $\mathcal{F}$ forged $\sigma = (C_1, C_2, C_3, c, v_\xi, v_x, v_y, v_{\delta_1}, v_{\delta_2})$ that traces to R_i is output with probability ε when $\mathcal{F}$ succeeds. Replay this process, on the basis of the forking lemma, $\mathcal{B}$ can obtain $\sigma' = (C_1, C_2, C_3, c', v'_\xi, v'_x, v'_y, v'_{\delta_1}, v'_{\delta_2})$ with probability $\varepsilon^2 / (16 q_H)$, where q_H is the number of hash queries made by $\mathcal{F}$. Moreover, using the extractor, one can find the whole private key of user i (R_i, x_i, y_i). With y_j be obtained and $g_1^{y_i}$ contained $Reg[j]$, so DLP on G_1 be successfully solved by $\mathcal{B}$.

6. Performance Analysis

Next, we analyze the performance of our scheme in terms of features, keys size and computation overhead. This analysis includes a comparison between a few best-known group signature schemes [15-18].

6.1 Performance Analysis

With the families of curves described in [19], the size of elements of G_1 is l =171-bit, the size of c is 80-bit and the order bit-length of G_1 is taken as a l_G =170-bit prime, as a result the total length of the signature is 1443-bit. When given parameters as above, we obtain the security is identical with a level 1024-bit RSA signature. We also assume that the non-frameability, adding a member in group, controllable linkability, the size of group public key, the size of user's key and the signature size are denoted by **NF**, **ADD**, **CL**, **GPKS**, **USKS**, **SS**, respectively. The comparison results of features, keys size (for *Gpk*, *Usk*) and signature length are summarized in the following table.1.

From the **Table 1**, it shows that [16, 17] and our scheme have the all desirable features (including dynamic joining and revocation, anonymity, traceability, non-frameability and controllable linkability). However, compared with [16,17], the *Gpk* and *Usk* size are shorter in our scheme.

Table 1. A Comparison of Some Group Signature Schemes

Schemes	**NF**	**OPENR**	**CL**	**ADD**	**GPKS**	**USKS**	**SS**
LPY[18]	YES	YES	N/A	N/A	O(log$N$)	O(n)	O(log$N$)
BBS[14]	N/A	N/A	N/A	N/A	1020	341	1443
DP[15]	YES	N/A	N/A	YES	1020	511	1444
HL[17]	YES	YES	YES	YES	2052	852	2044
HL [16]	YES	YES	YES	YES	1368	681	1363
Ours	YES	YES	YES	YES	1191	511	1443

Furthermore, in our scheme, the computation costs for signing, verifying, opening, key-updating are as follows.
Signing: Since $e(C_3, g_2) = e(\tau, g_2)^{\xi_1 + \xi_2} \cdot e(R, g_2)$, by caching $e(\tau, g_2)$ and $e(R, g_2)$, signing does not need the pairing operation. Furthermore, $C_1 = \eta^{\xi_1}$ and $D_1 = \eta^{\xi_1 r_x - r_{\delta_1}}$ can be computed

by one exponentiation $\eta^{max\{\xi_1,\xi_1 r_x - r_{\delta_1}\}}$ (similarly, C_2 and D_2 can be computed), signing only need three exponentiations in G_1, one multi-exponentiation with four bases in G_T. For multi-exponentiation, all bases are fixed in signing, further speedup by precomputation is possible.

Verifying: A verifier can obtain $e(C_3,g_2)^{v_x} e(C_3,\omega_1)^c$ by computing $e\left(C_3, g_2^{v_x}\omega_1^c\right)$. Therefore, verifying requires two multi-exponentiations with two bases in G_1, one multi-exponentiations with two bases in G_2, one multi-exponentiation with four bases in G_T and one pairing operation.

Opening: Since $R_i=\left(R_{ir}\right)^{\zeta_r^{-1}} = \left(C_1^{-\lambda_1\zeta_r^{-1}} \cdot C_2^{-\lambda_2\zeta_r^{-1}}\right) \cdot C_3$, opening requires one multi-exponentiation with two bases in G_1.

Key-updating: All unrevoked users must update their private keys when a new revocation occurs. When one user revoked, this updating requires one multi-exponentiation with two bases in G_1.

The computation cost of signing, verifying, and user's key-updating are denoted by **SC**, **VC** and **UC**, respectively. In additional, let $TE_{w,h}$ denotes simultaneous multi-exponentiation using w elements of group G_h, where h=1, 2, T. Let P denotes simultaneous pairing operation. The comparison results of computation costs for signing, verifying, updating-key are summarized in the following **Table 2**.

Table 2. A Comparison of Some Group Signature Schemes

Schemes	**SC**	**VC**	**UC(one user revoked)**
BBS[14]	$3TE_{1,1} + TE_{4,T}$	$4TE_{2,1}+ TE_{2,1}+ TE_{4,T} +P$	$TE_{2,1}$
DP[15]	$3TE_{1,1} + TE_{4,T}$	$2TE_{2,1}+TE_{3,1}+ TE_{2,2}+TE_{4,T} +P$	$TE_{2,1}$
HL[17]	$2TE_{1,1}+3\ TE_{3,1}+ TE_{8,T}$	$4TE_{2,1}+ TE_{4,1}+ TE_{2,2}+ TE_{7,T}+P$	$TE_{3,1}$
HL [16]	$3TE_{1,1}+ TE_{2,1}+ TE_{4,T}$	$TE_{2,1}+ TE_{3,1}+ TE_{2,2}+ TE_{4,T}+P$	$TE_{3,1}$
Ours	$3TE_{1,1} + TE_{4,T}$	$2TE_{2,1}+ TE_{2,2}+ TE_{4,T}+P$	$TE_{2,1}$

In **Table 2**, it shows that our scheme has lower computation costs for signing, verifying, updating-key. Therefore, our scheme is more efficient. Particularly, comparing with [16] which use expensive pairing operations (in judging process) to determine a signer's identity, our scheme is more attractive because we don't need pairing operation.

6.2 Experimental Results

The test for these results was performed on an Pentium Dual-Core CPU E5400 clocked at 2.70GHz and RAM 1.96GB. This test makes use of three *D-type* curves from the PBC library [22] running on top of Gnu GMP on Windows XP.

Table 3. Experimental Results (time:ms)

Schemes	**D174**	**D201**	**D224**
SC	63.648	81.449	98.781
VC	147.386	188.807	229.112
UC	3.770	4.845	6.259
Open	6.670	8.572	11.074

Every test result is the average of 1,000 tests. In **Table 3**, we show the running time of signing, verifying, user's key-updating (with 1 revoked user), opening. According to [23], further speedup is possible, when this test was performed on *A-type* curves.

For the sake of space, we only show the parameters for D174 Curve in **Table 4**.

Table 4. The D174 Curve with the embedding degree $k = 6$

q	150287996139850344657555064507715652292828322178603901559964838400 17
n	150287996139850344657555064507715613525832547441255206392965411950 21
h	1
r	150287996139850344657555064507715613525832547441255206392965411950 21
a	187122416362466663186009248912893905994497834714229217732382564209 6
b	979550172334338054714415200677665314930646613801273064011412560570 1
k	6
nk	115224746950252173700626030137909803345380964294556891142220249121844323192283932046503836617818648060762472595563783505416699943448784301362027149457614883858906199255534576681585042027865805599709459366576368553467135988880675162146348593305546345057671984158571504793459447217103562740477075361562962155734127637351356009538654190003989202925352157572915393075256396752045979389195048074272387358115 20
hk	510149159366842656049004871952561608481935712442746488553324756616583045063163010061128871772773450108640129881278296554492564248710245003685979894623738130621892741509165526892628526032540112485023560412065442627554817791373980403762815429385139704739907870646157347 20
coeff0	119751892582596971662570378252275369314467079446824709511118594461 92
coeff1	134330422003479348277427380952495468040066875620882540574119013627 71
coeff2	832746452111779123807910517544812200675986362550804349577088741161 4
nqr	142721363302176037340346936780070353538541593770301992936740616924

7. Conclusion

In this paper we proposed dynamic group signature. In our scheme, the non-frameability is guaranteed without the trusted party. Compared with the contemporary counterparts, our scheme more efficient and flexible group signature scheme which support the non-frameability. Furthermore, our scheme can also easily achieve the controllable linking which introduced by Hwang et. al. In addition, our scheme has provided algorithm structure such that the opening algorithm is available when key update by revocation is considered without incurring other secure threats. Lastly, the group manager flexibility allows one member to add to the group and revoke membership with time in our scheme. However, in [14], the adding member can't be achieved.

8. Acknowledgments

This study was supported by the National Natural Science Foundation of China (No.61370203) and the Research Foundation of Education Bureau of Sichuan Province(12ZB348),China. The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] D. Chaum and E. VanHeyst, "Group signatures," in *Proc. of Int. Conf. Advances in cryptology lEUROCRYPT91*, Springer Berlin Heidelberg, pp. 257-265, 1991. Article(CrossRef Link)

[2] M. Bellare, D. Micciancio and B. Warinschi, "Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions," in *Proc. of Int. Conf. on the Theory and Applications of Cryptographic Techniques*, pp. 614-629, May 4–8, 2003. Article(CrossRef Link)

[3] M. Bellare, H. Shi and C. Zhang, "Foundations of group signatures: The case of dynamic groups," in *Proc. of Int. Conf. The Cryptographers' Track at the RSA Conference 2005*, pp. 136-153, February 14-18, 2005. Article(CrossRef Link)

[4] D. Boneh, and H. Shacham, "Group signatures with verifier-local revocation," in *Proc. of Int. Conf. Proceedings of the 11th ACM conference on Computer and communications security, ACM*, pp. 168-177, 2004. Article(CrossRef Link)

[5] E. Brickell, J. Camenisch and L. Chen, "Direct anonymous attestation," in *Proc. of Int. Conf. Proceedings of the 11th ACM conference on Computer and communications security, ACM*, pp.132-145, 2004. Article(CrossRef Link)

[6] T. Nakanishi and N. Funabiki, "Verifier-local revocation group signature schemes with backward unlinkability from bilinear maps," in *Proc. of 11th International Conference on the Theory and Application of Cryptology and Information Security*, pp. 533-548, December 4-8, 2005. Article(CrossRef Link)

[7] S. Zhou and D. Lin, "Shorter verifier-local revocation group signatures from bilinear maps," in *Proc. of 5th International Conference*, pp. 126-143, December 8-10, 2006. Article(CrossRef Link)

[8] B. Libert and D. Vergnaud, "Group signatures with verifier-local revocation and backward unlinkability in the standard model," in *Proc. of 8th International Conference, CANS 2009*, Springer Berlin Heidelberg, pp. 498-517, December 12-14, 2009. Article(CrossRef Link)

[9] T. Nakanishi, H . Fujii, H. Yuta, et al, "Revocable group signature schemes with constant costs for signing and verifying," in *Proc. of ,12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA*, pp. 463-480, March 18-20, 2009. Article(CrossRef Link)

[10] J. Camenisch and A. Lysyanskaya, "Dynamic accumulators and application to efficient revocation of anonymous credentials," in *Proc. of 22nd Annual International Cryptology Conference Santa Barbara*, pp. 61-76, August 18–22, 2002. Article(CrossRef Link)

[11] J. Camenisch, M. Kohlweiss and C. Soriente, "An accumulator based on bilinear maps and e_cient revocation for anonymous credentials," in *Proc. of 12th International Conference on Practice and Theory in Public Key Cryptography*, pp. 481-500, March 18-20, 2009. Article(CrossRef Link)

[12] C. I. Fan, R. H. Hsu and M. Manulis, "Group signature with constant revocation costs for signers and verifiers," in *Proc. of 10th International Conference, CANS 2011*, pp. 214-233, December 10-12, 2011. Article(CrossRef Link)

[13] M. H. Au, W. Susilo W, Y. Mu, et al, "Constant-size dynamic k-times anonymous authentication," *Systems Journal, IEEE* vol 7, no 2, pp. 249 -261, June, 2013. Article(CrossRef Link)

[14] D. Boneh, X. Boyen and H. hacham, "Short group signatures," in *Proc. of 24th Annual International Cryptology Conference*, pp. 41-55, August 15-19, 2004. Article(CrossRef Link)

[15] C. Delerable and D. Pointcheval, "Dynamic fully anonymous short group signatures," in *Proc. of First International Conference on Cryptology in Vietnam*, pp.193-210, September 25-28, 2006. Article(CrossRef Link)

[16] J. Y. Hwang, L. Chen, H. S. Cho, et al, "Short Dynamic Group Signature Scheme Supporting Controllable Linkability," *Information Forensics and Security, IEEE Transactions on*, vol 10, no 6, 1109-1124, 2015. Article(CrossRef Link)

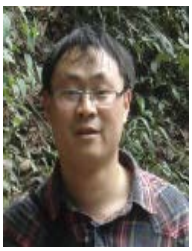
[17] J. Y. Hwang, S. Lee, B. H. Chung, et al, "Group signatures with controllable linkability for dynamic membership," *Information Sciences*, pp.761-778, 2013. Article(CrossRef Link)

[18] B. Libert, T. Peters and M. Yung, "Group signatures with almost-for-free revocation," *Advances in CryptologyCCRYPTO 2012*. Springer Berlin Heidelberg, pp. 571-589, August 19-23, 2012. Article(CrossRef Link)
[19] A. Miyaji, M. Nakabayashi, and S. Takano, "New explicit conditions of elliptic curve traces for FR-reduction," *IEICE Trans. Fundamentals*, E84-A(5), pp. 1234–43, May 2001. Article(CrossRef Link)
[20] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Proc. of A. M. Odlyzko, editor, Proceedings of Advances in Cryptology-CRYPTO' 86*, pp. 186–194, Aug, 1986. Article(CrossRef Link)
[21] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of cryptology*, vol. 13, no 3, pp. 361-396. 2000. Article(CrossRef Link)
[22] PBC Library. [Online]. Available: http://crypto.stanford.edu/pbc, *accessed Jun*. 2014
[23] F. Li, Z. Zheng and C. Jin, "Identity-based deniable authenticated encryption and its application to e-mail system," *Telecommunication Systems*, pp. 1-15, 2015. Article(CrossRef Link)

Run Xie received his M.Sc degree in mathematics and applied mathematics at Southwest Jiaotong University in 2006,P.R.China. He is a Ph.D. degree candidate in information security at University of Electronic Science Technology of China (UESTC). He is presently engaged in cryptography, network security and cloud computing security.

Chunxiang Xu received her B.Sc., M.Sc. and Ph.D. degrees at Xidian University, in 1985, 1988 and 2004 respectively, P.R.China. She is presently engaged in information security, cloud computing security and cryptography as a professor at University of Electronic Science Technology of China (UESTC).

Changlian He received her M.Sc degree in the soft engineering special at University of Electronic Science Technology of China (UESTC) in 200, P.R.China. She is presently engaged in cryptography, network security and cloud computing security.



Xiaojun Zhang received his B.Sc. degree in mathematics and applied mathematics at Hebei Normal University in 2009, P.R.China and received M.Sc degree in pure mathematics at Guangxi University in 2012. He received Ph.D. degree in information security at University of Electronic Science Technology of China (UESTC) in 2015. He is a lecturer at School of Computer Science in Southwest Petroleum University. He is presently engaged in cryptography, network security and cloud computing security.