SCIS 2006 The 2006 Symposium on Cryptography and Information Security
Hiroshima, Japan, Jan. 17-20, 2006
The Institute of Electronics, Information and Communication Engineers

Short Group Signatures with Efficient Flexible Join

Toshiaki Makita * Yoshifumi Manabe * † Tatsuaki Okamoto * †

Abstract— We present a short group signature scheme with an efficient (concurrent) join protocol. Signatures in our scheme are almost as short as Boneh, Boyen and Shacham's Short Group Signatures (BBS04) that has no join protocol, and the computational costs of our scheme are also almost as efficient as BBS04. The security of our group signature is based on the Decision Linear Diffie-Hellman assumption and the 2 Variable Strong Diffie-Hellman (2SDH) assumption, which is a slightly strong variant of the Strong Diffie-Hellman (SDH) assumption. We prove the security of our system, in the random oracle model, using a security definition for group signatures recently given by Bellare, Shi, and Zhang.

Keywords: Group Signature, Short Signature, Bilinear Mapping

1 Introduction

Group signatures were proposed by Chaum and van Heyst [6]. This mechanism provides signers anonymity, that is, any member of a group can sign on behalf of the group while hiding the identity of the actual signer within the group. However, Opener (one of group managers) can identify the actual signer if needed, for instance, in the case of unfair usage. We can apply group signatures to many various areas, such as voting and bidding.

Bilinear maps, which take advantage of pairings on elliptic curve, have been found useful, and many applications have been proposed recently, e.g., identity-based encryption, short signatures, and short group signatures. Boneh, Boyen and Shacham's short group signatures [4] (BBS04), which is based on Boneh and Boyen's short signatures [3] (BB04), is currently one of the most efficient and shortest signature schemes. However, BBS04 did not implement a join protocol and exculpability, and only showed how to implement it. Some works have proposed new schemes after BBS04, e.g., Kiayias and Yung [8] (KY05), Furukawa and Imai [7] (FI05), Boyen and Waters [5] (BW05), and Ateniese, Camenisch, de Medeiros and Hohenberger [1] (ACMH05). KY05 implemented a concurrent join protocol, but its signature length is too long. FI05 proposed an efficient and short signature, but it does not provide security proof and does not implement concurrent join. BW05 and ACMH05 proposed provably secure group signature schemes in the standard model. However, the signature length of BW05 depends on the number of members, and verifying or opening algorithm of ACMH05 is inefficient. Therefore, any provably secure group signatures (with or without random oracles) with (efficient concurrent) join protocols are much less efficient/longer than the BBS04 signatures.

We propose a provably secure group signature scheme that is efficient and whose signature length is almost as short as BBS04. Moreover, it is secure even if users concurrently join. Our scheme is based on Okamoto's new efficient signature scheme [9], while BBS04 is based on BB04.

2 Preliminaries

2.1 Bilinear Groups

This paper follows the notation regarding bilinear groups in [3]. Let $(\mathbb{G}_1, \mathbb{G}_2)$ be bilinear groups as follows:

1. $\mathbb{G}_1$ and $\mathbb{G}_2$ are two cyclic groups of prime order p,
2. g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$,
3. ψ is an isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$, with $\psi(g_2) = g_1$,
4. e is a non-degenerate bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = p$, and
5. e, ψ and the group action in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ can be computed efficiently.

2.2 Assumptions

2.2.1 The Decision Linear Diffie-Hellman Assumption [4]

Let $\mathbb{G}_1$ be a cyclic group of prime order p. Let u, v, h be generators of $\mathbb{G}_1$. Consider the following problem:

Decision Linear Problem in $\mathbb{G}_1$. Given $u, v, h, u^a, v^b, h^c \in \mathbb{G}_1$ as input, output yes if $a + b = c$ and no otherwise.

* Department of Social Informatics, Graduate School of Informatics, Kyoto University, Yoshidahonmachi, Sakyo-ku, Kyoto-shi, Japan, makita@lab7.kuis.kyoto-u.ac.jp
† NTT Laboratories, 1-1 Hikarino-oka, Yokosuka-shi, Japan, manabe.yoshifumi@lab.ntt.co.jp, okamoto.tatsuaki@lab.ntt.co.jp

The advantage of an adversary $\mathcal{A}$ in deciding the Decision Linear problem in $\mathbb{G}_1$ is defined as

$$\mathsf{Adv}_{\mathcal{A}}^{\text{Linear}} = \left| \begin{array}{l} \Pr\left[\begin{array}{c}\mathcal{A}(u,v,h,u^a,v^b,h^{a+b}) = \mathsf{yes}: \\ u,v,h \overset{R}{\leftarrow} \mathbb{G}_1,\ a,b \overset{R}{\leftarrow} \mathbb{Z}_p\end{array}\right] \\ -\Pr\left[\begin{array}{c}\mathcal{A}(u,v,h,u^a,v^b,\eta) = \mathsf{yes}: \\ u,v,h,\eta \overset{R}{\leftarrow} \mathbb{G}_1,\ a,b \overset{R}{\leftarrow} \mathbb{Z}_p\end{array}\right] \end{array} \right|.$$

Definition 2.1. The (t,ϵ)-Decision Linear Diffie-Hellman Assumption holds in $\mathbb{G}_1$ if no t-time algorithm has advantage at least ϵ in solving the Decision Linear problem in $\mathbb{G}_1$.

Linear Encryption [4]. We use an encryption scheme, called the Linear Encryption, that is an extension of ElGamal encryption. This scheme is derived from the Decision Linear Problem, and is secure even in groups where an algorithm which solves the Decisional Diffie-Hellman problem exists. In this scheme, a user's public key is a triple of generators $u,v,h \in \mathbb{G}_1$; its private key is the exponents $x,y \in \mathbb{Z}_p$ such that $u^x = v^y = h$. To encrypt a message $m \in \mathbb{G}_1$, choose random values $a,b \in \mathbb{Z}_p$, and output the triple (u^a, v^b, mh^{a+b}). To recover the message from an encryption (T_1,T_2,T_3), the user computes $T_3/(T_1^x \cdot T_2^y)$. This encryption scheme is semantically secure against chosen plaintext attacks, assuming the Decision Linear Diffie-Hellman Assumption holds.

2.2.2 The 2 Variable Strong Diffie-Hellman Assumption [9]

Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p. Let g_1 be a generator of $\mathbb{G}_1$ and g_2 a generator of $\mathbb{G}_2$. Consider the following problem:

q-2 Variable Strong Diffie-Hellman Problem. The q-2SDH problem in $(\mathbb{G}_1,\mathbb{G}_2)$ is defined as follows: given a $(2q+6)$-tuple $(g_1, g_2, g_2^x, \ldots, g_2^{x^q}, g_2^y, g_2^{yx}, \ldots, g_2^{yx^q}, g_2^{\frac{y+a}{x+b}}, a, b)$ as input, output a pair $(g_1^{\frac{1}{x+c}}, c)$ where $c \in \mathbb{Z}_p^*$. Algorithm $\mathcal{A}$ has advantage ϵ in solving q-2SDH in $(\mathbb{G}_1,\mathbb{G}_2)$ if

$$\Pr\left[\mathcal{A}\begin{pmatrix} g_1, g_2, g_2^x, \ldots, g_2^{x^q}, \\ g_2^y, g_2^{yx}, \ldots, g_2^{yx^q}, g_2^{\frac{y+a}{x+b}}, a, b \end{pmatrix} = (g_1^{\frac{1}{x+c}}, c)\right] \geq \epsilon,$$

where the probability is over the random choice of generator g_2 in $\mathbb{G}_2$ (with $g_1 \leftarrow \psi(g_2)$), of x,y,a,b in $\mathbb{Z}_p^*$, and of the random bits of $\mathcal{A}$.

Definition 2.2. The (q,t,ϵ)-2SDH assumption holds in $(\mathbb{G}_1,\mathbb{G}_2)$ if no t-time algorithm has advantage at least ϵ in solving the q-2SDH problem in $(\mathbb{G}_1,\mathbb{G}_2)$.

2.3 Security of Group Signature

2.3.1 A Model for Group Signature Scheme

Parties which appear in a group signature scheme are users, which consist of members (group members) and others, Issuer, which can issue member certifications to users, and Opener, which can trace a signature to the member that created the signature. A group signature scheme is composed of the following algorithms:

GKEYGEN: A probabilistic algorithm that outputs the group public key, the Opener secret key, and the Issuer secret key.

UKEYGEN: A probabilistic algorithm that outputs a personal public and secret key pair ($\boldsymbol{upk}[i]$, $\boldsymbol{usk}[i]$). A user that wants to be a group member should begin by running UKEYGEN. We assume that the table $\boldsymbol{upk}$ is public.

JOIN, ISSUE: JOIN and ISSUE are interactive algorithms that realize a protocol with which a user can join the group. They are, respectively, used by users, given the user's secret key, and Issuer, given the Issuer secret key. By this protocol, the user gets the group signing key and Issuer gets values unique to the member related to OPEN. Issuer writes the new member's secret information in its registration table $\boldsymbol{reg}$.

SIGN: A probabilistic algorithm that outputs a group signature, given a group member's secret key and a message.

VERIFY: A deterministic algorithm that determines whether a signature is valid or not, given the group public key, a message, and the signature.

OPEN: A deterministic algorithm that outputs the signer of a valid signature, given the Opener secret key, a message, and its signature. Here, Opener can read the content of Issuer's registration table $\boldsymbol{reg}$. OPEN also makes a proof-string that can be verified by JUDGE to prove that OPEN is correctly executed.

JUDGE: A deterministic algorithm that verifies a proof-string output by OPEN, given a user identity, a message, a valid signature, and its proof-string.

2.3.2 Security Definitions

We use a security definition given by Bellare, et al. [2]. Following the definition, a group signature scheme must fulfill correctness and the following security requirements: anonymity, traceability, and non-frameability.

Anonymity. The anonymity demands that no signature can be traced to some member identity or linked to any other signature by anyone other than Opener.

CPA-Anonymity. In our proofs, we use a relaxed anonymity requirement, called CPA-anonymity that is similar to CPA-full-anonymity defined in BBS04 [4]. In the CPA-anonymity experiment, the adversary cannot query the opening oracle. We assume that Opener is highly trusted and adversary cannot access it as long as it is honest, and we prove security in this CPA-anonymity model.

Traceability. Traceability demands that the adversary be unable to produce a signature such that either the honest opener declares itself unable to identify the origin of the signature, or, the honest opener believes it has identified the origin but is unable to produce a correct proof of its claim.

Non-frameability. Non-frameability asks that the adversary be unable to create a judge-accepted proof that an honest user produced a certain valid signature unless the user really did produce this signature.

2.4 Basic Signature Scheme [9]

We describe a signature scheme that is strongly existentially unforgeable against chosen plaintext attacks, and this scheme is a fundamental element of the proposed group signature scheme.

Public key: $g_2, u, v \in \mathbb{G}_2, g_1 \leftarrow \psi(g_2), w \leftarrow g_2^x$

Secret key: $x \in \mathbb{Z}_p^*$

Signature generation: Let m be a message to be signed. Signer randomly selects (r, s) from $\mathbb{Z}_p$, (if $x + r = 0$, selects r again,) and computes

$$\sigma \leftarrow (g_1^m \psi(uv^s))^{\frac{1}{x+r}}.$$

(σ, r, s) is the signature of m.

Verification: Check $\sigma \neq 1$ and $e(\sigma, wg_2^r) \stackrel{?}{=} e(g_1, g_2^m uv^s)$.

Security: This scheme is (q_S, t, ϵ)-strongly existentially unforgeable against adaptively chosen message attacks under the (q, t', ϵ')-2SDH assumption, where $q_S < q$, $\epsilon \geq 3q\epsilon'$, $t \leq t' - \Theta(q^2 T)$, and T is the maximum time for an exponentiation in $\mathbb{G}_1$ and $\mathbb{G}_2$.

3 Proposed Group Signature Scheme

We now describe the construction of the proposed group signature scheme. We use a bilinear group pair $(\mathbb{G}_1, \mathbb{G}_2)$ with a computable isomorphism ψ, as in Section 2.1. We assume the basic signature scheme is strongly existentially unforgeable against chosen message attacks and the Decision Linear Diffie-Hellman assumption holds on $\mathbb{G}_2$. Moreover, we use a signature scheme that is existentially unforgeable against chosen message attacks as the user's signature scheme. Let a hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$ be a random oracle.

In JOIN and ISSUE, although users do not prove the knowledge of the logarithm of B_i, this scheme is still secure, because a user that does not know q_i cannot generate a valid signature. In addition, as long as q_i and sk_i is secret, any other secret value of users can be public, so users and Issuer can communicate via public channels.

3.1 Parameters

Public key: $g_2, u, v \in \mathbb{G}_2, g_1 \leftarrow \psi(g_2), w \leftarrow g_2^x$, $U \leftarrow g_2^{\xi_1}$, $V \leftarrow g_2^{\xi_2}$, $\{pk_i | \ i \in member\}$

Secret key of Issuer: $x \in \mathbb{Z}_p^*$ (only for Issuer); $\{(B_i \leftarrow g_2^{q_i}, r_i, s_i) \mid i \in member\}$ (shared with member i)

Secret key of Opener: $\xi_1, \xi_2 \in \mathbb{Z}_p^*$

Secret key of member i: $q_i \in \mathbb{Z}_p^*$, sk_i (only for member i);
$A_i \leftarrow \psi(B_i uv^{s_i})^{\frac{1}{x+r_i}}, r_i, s_i$ (shared with Issuer)

3.2 Algorithms

GKEYGEN: Issuer uniformly selects $g_2, u, v \stackrel{R}{\leftarrow} \mathbb{G}_2$, $x \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$, sets $g_1 \leftarrow \psi(g_2)$, $w \leftarrow g_2^x$, and publishes (g_1, g_2, u, v, w). Opener then uniformly picks $\xi_1, \xi_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$, computes $U \leftarrow g_2^{\xi_1}$, $V \leftarrow g_2^{\xi_2}$, and publishes (U, V). The group public key is $(g_1, g_2, u, v, w, U, V)$, the Issuer key is x, and the Opener key is (ξ_1, ξ_2).

UKEYGEN: A user i runs a key generation algorithm of a certain signature scheme, called user's signature scheme, and obtain a key pair $(pk_i, sk_i) = (\boldsymbol{upk}[i], \boldsymbol{usk}[i])$ (where any signature scheme can be used as long as it is existentially unforgeable against chosen message attacks).

JOIN: User i uniformly selects $q_i \in \mathbb{Z}_p^*$ and computes $B_i \leftarrow g_2^{q_i}$. User i then creates signature sig_i on B_i using sk_i, and sends (B_i, sig_i) to Issuer.

ISSUE: Issuer checks whether sig_i is a valid signature on B_i for i's public key pk_i. If it is valid, Issuer randomly selects r_i and s_i from $\mathbb{Z}_p$ (if $x + r_i = 0$, selects r_i again), computes $A_i \leftarrow \psi(B_i uv^{s_i})^{\frac{1}{x+r_i}}$, and sends (A_i, r_i, s_i) to i (Here, $e(A_i, wg_2^{r_i}) = e(g_1, B_i uv^{s_i})$). Issuer writes $(A_i, B_i, r_i, s_i, sig_i)$ in the ith entry of its registration table $\boldsymbol{reg}$.

SIGN: Member i randomly selects α_1, α_2 and β from $\mathbb{Z}_p$, and computes

$$\begin{aligned} a &\leftarrow A_i g_1^{\alpha_1+\alpha_2}, & d_1 &\leftarrow \psi(U)^{\alpha_1}, \\ b &\leftarrow (wg_2^{r_i})^{\beta}, & d_2 &\leftarrow \psi(V)^{\alpha_2}, \\ c &\leftarrow (B_i uv^{s_i})^{\beta} b^{\alpha_1+\alpha_2}. \end{aligned}$$

Member i also generates a (Fiat-Shamir heuristic) signature to prove the knowledge of $(\alpha_1, \alpha_2, \beta, \beta r_i, \beta q_i, \beta s_i)$ for $b = g_2^{\beta r_i} w^{\beta}$, $c = g_2^{\beta q_i} u^{\beta} v^{\beta s_i} b^{\alpha_1+\alpha_2}$, $d_1 = \psi(U)^{\alpha_1}$, $d_2 = \psi(V)^{\alpha_2}$ as follows:

$$\begin{aligned} \tilde{t}_1 &\leftarrow g_2^{\tilde{R}_1} w^{\tilde{R}_2}, & \tilde{Z}_1 &\leftarrow \tilde{R}_1 + \tilde{h}\beta r_i \mod p, \\ \tilde{t}_2 &\leftarrow g_2^{\tilde{R}_3} u^{\tilde{R}_2} v^{\tilde{R}_4} b^{\tilde{R}_5+\tilde{R}_6}, & \tilde{Z}_2 &\leftarrow \tilde{R}_2 + \tilde{h}\beta \mod p, \\ \tilde{t}_3 &\leftarrow \psi(U)^{\tilde{R}_5}, & \tilde{Z}_3 &\leftarrow \tilde{R}_3 + \tilde{h}\beta q_i \mod p, \\ \tilde{t}_4 &\leftarrow \psi(V)^{\tilde{R}_6}, & \tilde{Z}_4 &\leftarrow \tilde{R}_4 + \tilde{h}\beta s_i \mod p, \\ \tilde{h} &\leftarrow H(\tilde{t}_1, \tilde{t}_2, \tilde{t}_3, \tilde{t}_4, m), & \tilde{Z}_5 &\leftarrow \tilde{R}_5 + \tilde{h}\alpha_1 \mod p, \\ & & \tilde{Z}_6 &\leftarrow \tilde{R}_6 + \tilde{h}\alpha_2 \mod p. \end{aligned}$$

Here $\tilde{R}_1, \ldots, \tilde{R}_6$ are randomly selected from $\mathbb{Z}_p$. i outputs $\sigma \leftarrow (a, b, c, d_1, d_2, \tilde{h}, \tilde{Z}_1, \tilde{Z}_2, \tilde{Z}_3, \tilde{Z}_4, \tilde{Z}_5, \tilde{Z}_6)$.

VERIFY: A verifier with public key $(g_1, g_2, w, u, v, U, V)$ and message m along with signature σ checks whether the following equations hold or not.

$$\begin{aligned} & e(a, b) \stackrel{?}{=} e(g_1, c), & \tilde{t}_3' &\leftarrow \psi(U)^{\tilde{Z}_5} / d_1^{\tilde{h}}, \\ \tilde{t}_1' &\leftarrow g_2^{\tilde{Z}_1} w^{\tilde{Z}_2} / b^{\tilde{h}}, & \tilde{t}_4' &\leftarrow \psi(V)^{\tilde{Z}_6} / d_2^{\tilde{h}}, \\ \tilde{t}_2' &\leftarrow g_2^{\tilde{Z}_3} u^{\tilde{Z}_2} v^{\tilde{Z}_4} b^{\tilde{Z}_5+\tilde{Z}_6} / c^{\tilde{h}}, & \tilde{h} &\stackrel{?}{=} H(\tilde{t}_1', \tilde{t}_2', \tilde{t}_3', \tilde{t}_4', m). \end{aligned}$$

OPEN: Opener computes $A_i = a/(d_1^{1/\xi_1} d_2^{1/\xi_2})$ and generates a signature to prove the knowledge of (ξ_1, ξ_2)

for $A_i = a/(d_1^{1/\xi_1} d_2^{1/\xi_2}), U = g_2^{\xi_1}, V = g_2^{\xi_2}$ as follows:

$$
\begin{aligned}
&X_1 \leftarrow d_1^{1/\xi_1}, && \hat{t}_4 \leftarrow g_2^{\hat{R}_2},\\
&X_2 \leftarrow d_2^{1/\xi_2}, && \hat{h} \leftarrow H(\hat{t}_1, \hat{t}_2, \hat{t}_3, \hat{t}_4),\\
&\hat{t}_1 \leftarrow X_1^{\hat{R}_1}, && \hat{Z}_1 \leftarrow \hat{R}_1 + \hat{h}\xi_1 \mod p,\\
&\hat{t}_2 \leftarrow X_2^{\hat{R}_2}, && \hat{Z}_2 \leftarrow \hat{R}_2 + \hat{h}\xi_2 \mod p.\\
&\hat{t}_3 \leftarrow g_2^{\hat{R}_1}, &&
\end{aligned}
$$

Here $\hat{R}_1$ and $\hat{R}_2$ are randomly selected from $\mathbb{Z}_p$. Opener reads Issuer's registration table $\boldsymbol{reg}$, and determines the identity of σ's signer i. Opener outputs i and proof-string $\tau \leftarrow (sig_i, B_i, A_i, r_i, s_i, X_1, X_2, \hat{h}, \hat{Z}_1, \hat{Z}_2)$.

JUDGE: A judge with a member identity i, a message m, a valid signature σ of m and a proof-string τ checks whether sig_i is a valid signature on B_i for pk_i and whether the following equations hold or not.

$$
\begin{aligned}
&e(A_i, wg_2^{r_i}) \stackrel{?}{=} e(g_1, B_i uv^{s_i}), && \hat{t}'_3 \leftarrow g_2^{\hat{Z}_1}/U^{\hat{h}},\\
&A_i \stackrel{?}{=} a/(X_1 X_2), && \hat{t}'_4 \leftarrow g_2^{\hat{Z}_2}/V^{\hat{h}},\\
&\hat{t}'_1 \leftarrow X_1^{\hat{Z}_1}/d_1^{\hat{h}}, && \hat{h} \stackrel{?}{=} H(\hat{t}'_1, \hat{t}'_2, \hat{t}'_3, \hat{t}'_4).\\
&\hat{t}'_2 \leftarrow X_2^{\hat{Z}_2}/d_2^{\hat{h}}, &&
\end{aligned}
$$

4 Security

Theorem 4.1. *The group signature scheme is correct.*

Proof. A group signature is verified by an equation $e(a,b) \stackrel{?}{=} e(g_1, c)$ and a Fiat-Shamir heuristic signature to prove the knowledge. As long as the Fiat-Shamir heuristic signature is correctly generated, it is always accepted and the equation $e(a,b) = e(A_i, (wg_2^{r_i})^{\beta}) \cdot e(g_1^{\alpha_1+\alpha_2}, (wg_2^{r_i})^{\beta}) = e(g_1, B_i uv^{s_i}) \cdot e(g_1, b^{\alpha_1+\alpha_2}) = e(g_1, c)$ always holds, so a correct signature is always accepted by VERIFY.

Moreover, Opener proves $A_i = a/(d_1^{1/\xi_1} d_2^{1/\xi_2})$, $e(A_i, wg_2^{r_i}) = e(g_1, B_i uv^{s_i})$ and that sig_i is valid. A correct signature is opened correctly as long as Opener is honest, Issuer's registration table is not rewritten by the adversary, and the user's signature scheme is correct. Besides, since Opener is honest, his proof-string is always accepted by JUDGE. □

Theorem 4.2. *If the (t', ϵ')-Decision Linear Diffie-Hellman assumption holds in $\mathbb{G}_2$ then the group signature scheme is (t, q_H, ϵ)-CPA-anonymous, where $\epsilon = 2\epsilon'$ and $t = t' - q_H O(1)$.*

Proof. Assume $\mathcal{A}$ is an algorithm that (t, q_H, ϵ)-breaks the anonymity of the group signature scheme. We construct an algorithm $\mathcal{B}$ that, by interacting with $\mathcal{A}$, solves the Decision Linear Problem in time t' with advantage ϵ'.

Algorithm $\mathcal{B}$ is given a random instance $(U, V, g_2, U^{\alpha_1}, V^{\alpha_2}, g_2^{\gamma})$ of the Decision Linear Problem. It generates the components of the group signature public key and the Issuer key, i.e., sets $g_1 \leftarrow \psi(g_2)$, picks random $u, v \stackrel{R}{\leftarrow} \mathbb{G}_2$, $x \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and computes $w \leftarrow g_2^x$. It then provides to $\mathcal{A}$ the group public key $(g_1, g_2, w, u, v, U, V)$ and issuer key x.

If a member with index i is added, $\mathcal{B}$ generates the user secret key, i.e., selects random $q_i \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$, $r_i, s_i \stackrel{R}{\leftarrow} \mathbb{Z}_p$ (if $x + r_i = 0$, selects r_i again) and a random user's signature key pair pk_i, sk_i. It then carries out JOIN and ISSUE and writes $(A_i, B_i, r_i, s_i, sig_i)$ in Issuer's registration table $\boldsymbol{reg}[i]$. When $\mathcal{A}$ corrupts a user, $\mathcal{B}$ hands $\mathcal{A}$ its secret key.

At any time, $\mathcal{A}$ can query the random oracle H. Algorithm $\mathcal{B}$ responds with elements selected uniformly at random from $\mathbb{Z}_p$, making sure to respond identically to repeated queries.

$\mathcal{A}$ requests its anonymity challenge by providing two indices, i_0 and i_1, and a message m. $\mathcal{B}$, in turn, choices a random bit $\ell \in \{0,1\}$ ($\Pr[\ell = 0] = \Pr[\ell = 1] = 1/2$), selects random $\beta \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and computes $a \leftarrow A_{i_\ell} \psi(g_2^{\gamma})$, $b \leftarrow (wg_2^{r_{i_\ell}})^{\beta}$, $c \leftarrow (B_{i_\ell} uv^{s_{i_\ell}})^{\beta} g_2^{\gamma(x+r_{i_\ell})\beta}$, $d_1 \leftarrow \psi(U^{\alpha_1})$, $d_2 \leftarrow \psi(V^{\alpha_2})$.

$\mathcal{B}$ randomly generates the remaining signature values, $\tilde{h}, \tilde{Z}_1, \ldots, \tilde{Z}_6 \stackrel{R}{\leftarrow} \mathbb{Z}_p$. $\mathcal{B}$ then computes $\tilde{t}'_1, \tilde{t}'_2, \tilde{t}'_3, \tilde{t}'_4$ from $\tilde{h}, \tilde{Z}_1, \ldots, \tilde{Z}_6$, and patches $H(\tilde{t}'_1, \tilde{t}'_2, \tilde{t}'_3, \tilde{t}'_4, m)$ to equal $\tilde{h}$. It returns a signature $\sigma \leftarrow (a, b, c, d_1, d_2, \tilde{h}, \tilde{Z}_1, \tilde{Z}_2, \tilde{Z}_3, \tilde{Z}_4, \tilde{Z}_5, \tilde{Z}_6)$ to $\mathcal{A}$.

Finally, $\mathcal{A}$ outputs a bit ℓ'. If $\ell' = \ell$, $\mathcal{B}$ outputs yes (guesses $\gamma = \alpha_1 + \alpha_2$). Else (if $\ell' \neq \ell$), $\mathcal{B}$ outputs no.

Here,

$$
\begin{aligned}
&\Pr[\ell' = \ell]\\
&\quad = \Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-}\ell}(k) = \ell']\\
&\quad = \Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-1}}(k) = 1] \cdot \Pr[\ell = 1]\\
&\qquad + \Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-0}}(k) = 0] \cdot \Pr[\ell = 0]\\
&\quad = \frac{1}{2} \cdot (\Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-1}}(k) = 1] + \Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-0}}(k) = 0]).
\end{aligned}
$$

If $\gamma = \alpha_1 + \alpha_2$,

$$\Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-1}}(k) = 1] + \Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-0}}(k) = 0] = \epsilon + 1.$$

So,

$$\Pr\left[\begin{matrix}\mathcal{B}(U, V, g_2, U^{\alpha_1}, V^{\alpha_2}, g_2^{\alpha_1+\alpha_2}) = \mathsf{yes} :\\ U, V, g_2 \stackrel{R}{\leftarrow} \mathbb{G}_2,\ \alpha_1, \alpha_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p\end{matrix}\right] = \frac{\epsilon + 1}{2}.$$

Else (if $\gamma \neq \alpha_1 + \alpha_2$),

$$\Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-1}}(k) = 1] + \Pr[\mathsf{Exp}_{\mathcal{A}}^{\text{anon-0}}(k) = 0] = 1.$$

Therefore,

$$\Pr\left[\begin{matrix}\mathcal{B}(U, V, g_2, U^{\alpha_1}, V^{\alpha_2}, \eta) = \mathsf{yes} :\\ U, V, g_2, \eta \stackrel{R}{\leftarrow} \mathbb{G}_2,\ \alpha_1, \alpha_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p\end{matrix}\right] = \frac{1}{2}.$$

Consequently, $\epsilon' = \mathsf{Adv}_{\mathcal{B}}^{\text{Linear}} = \epsilon/2$. □

Theorem 4.3. *If the basic signature scheme is (q_S, t', ϵ')-strongly existentially unforgeable against chosen message attacks, then the group signature scheme is (t, q_H, n, ϵ)-traceable, where $n = q_S$, $\epsilon = 4\sqrt{\epsilon' q_H} + 1/p$, and $t = \Theta(1) \cdot t'$.*

Proof. Assume $\mathcal{A}$ is an algorithm that (t, q_H, n, ϵ)-breaks the traceability of the group signature scheme. We construct an algorithm $\mathcal{B}$ that, by interacting with $\mathcal{A}$, breaks the basic signature scheme in time t' with advantage ϵ', where q_S is the maximum number of signing oracle queries made by $\mathcal{B}$.

Algorithm $\mathcal{B}$ is given a random public key (g_1, g_2, w, u, v) of the basic signature scheme. It generates the components of the group signature public key and the Opener key, i.e., picks random $\xi_1, \xi_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and computes $U \leftarrow g_2^{\xi_1}, V \leftarrow g_2^{\xi_2}$. It then provides to $\mathcal{A}$ the group public key $(g_1, g_2, w, u, v, U, V)$ and Opener key (ξ_1, ξ_2).

If a member with index i is added, $\mathcal{B}$ generates the user secret key, i.e., selects random $q_i \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$, and a random user's signature key pair pk_i, sk_i. It then carries out JOIN and ISSUE, and Issuer uses the signing oracle of the basic signature scheme to output (A_i, r_i, s_i) given q_i as input (message) and writes $(A_i, B_i, r_i, s_i, sig_i)$ in Issuer's registration table $\boldsymbol{reg}[i]$. When $\mathcal{A}$ corrupts a user, $\mathcal{B}$ hands $\mathcal{A}$ its secret key.

At any time, $\mathcal{A}$ can query the random oracle H. Algorithm $\mathcal{B}$ responds with elements selected uniformly at random from $\mathbb{Z}_p$, making sure to respond identically to repeated queries.

Eventually, $\mathcal{A}$ outputs a valid message-signature pair, the signer of which cannot be identified by Opener (since the proof-string τ is accepted by JUDGE as long as Opener is honest). This means the component A of the signature does not correspond to that of any member in $\boldsymbol{reg}$. Therefore, $(q, (A, r, s))$ used to make the signature is a forgery (message-signature pair) of the basic signature scheme.

We now describe how to extract (q, A, r, s). From now on, we abbreviate signatures as $(m, \tilde{\sigma}, \tilde{t}, \tilde{h}, \tilde{Z})$, where $\tilde{\sigma} = (a, b, c, d_1, d_2)$, $\tilde{t} = (\tilde{t}_1, \tilde{t}_2, \tilde{t}_3, \tilde{t}_4)$, and $\tilde{Z} = (\tilde{Z}_1, \tilde{Z}_2, \tilde{Z}_3, \tilde{Z}_4, \tilde{Z}_5, \tilde{Z}_6)$.

An execution of $\mathcal{A}$ completely depends on a random string ω used by $\mathcal{A}$ and a vector f of the responses of the hash oracle. Let S be the set of pairs (ω, f) such that $\mathcal{A}$ completes successfully with a forgery $(m, \tilde{\sigma}, \tilde{t}, \tilde{h}, \tilde{Z})$ and $\mathcal{A}$ queried the hash oracle on $(\tilde{t}, m)$. In this case, we let $\mathrm{Ind}(\omega, f)$ be the index of f at which $\mathcal{A}$ queried $(\tilde{t}, m)$ and define $\nu = \Pr[S] = \epsilon - 1/p$, where the $1/p$ corresponds to the possibility that $\mathcal{A}$ guessed the response of hash oracle on $(\tilde{t}, m)$ for itself. For each j, $1 \le j \le q_H$, let S_j be the set of pairs (ω, f) such that $\mathrm{Ind}(\omega, f) = j$, and let J be the set of indices j such that $\Pr[S_j \mid S] \ge 1/(2q_H)$. Then $\Pr[\mathrm{Ind}(\omega, f) \in J \mid S] \ge 1/2$.

Let $f|_a^b$ represents elements at indices $a, a+1, \ldots, b$ of f. For each $j \in J$, we consider the heavy-rows lemma [10, Lemma 1] where rows X and columns Y are all possible values of $(\omega, f|_1^{j-1})$ and $f|_j^{q_H}$ respectively. Clearly $\Pr_{(x,y)}[(x, y) \in S_j] \ge \nu/(2q_H)$. Let the heavy rows Ω_j be those rows such that, $\forall (x, y) \in \Omega_j$: $\Pr_{y'}[(x, y') \in S_j] \ge \nu/(4q_H)$. Then, by the heavy-rows lemma, $\Pr[\Omega_j \mid S_j] \ge 1/2$. It obviously follows that $\Pr[\exists j \in J : \Omega_j \cap S_j \mid S] \ge 1/4$.

Thus, with probability $\nu/4$, $\mathcal{A}$ succeeds and obtains a forgery $(m, \tilde{\sigma}, \tilde{t}, \tilde{h}, \tilde{Z})$ derived from a heavy row $(x, y) \in \Omega_j$ for some $j \in J$, i.e., an execution (ω, f) such $\Pr_{f'}[(\omega, f') \in S_j \mid f'|_1^{j-1} = f|_1^{j-1}] \ge \nu/(4q_H)$.

If we now rewind the execution of $\mathcal{A}$ to the jth query, and proceed with an oracle vector f' that differs from f from the jth entry on, we obtain, with probability at least $\nu/(4q_H)$, a successful execution completion and a second forgery $(m, \tilde{\sigma}, \tilde{t}, \tilde{h}', \tilde{Z}')$, with $(\tilde{t}, m)$ still queried at $\mathcal{A}$'s jth hash query.

By using the extractor, we obtain from $(\tilde{\sigma}, \tilde{t}, \tilde{h}, \tilde{Z})$ and $(\tilde{\sigma}, \tilde{t}, \tilde{h}', \tilde{Z}')$ a forgery $(q, (A, r, s))$, i.e., $\beta r \leftarrow (\tilde{Z}_1 - \tilde{Z}_1')/(\tilde{h} - \tilde{h}')$, $\beta \leftarrow (\tilde{Z}_2 - \tilde{Z}_2')/(\tilde{h} - \tilde{h}')$, $\beta q \leftarrow (\tilde{Z}_3 - \tilde{Z}_3')/(\tilde{h} - \tilde{h}')$, $\beta s \leftarrow (\tilde{Z}_4 - \tilde{Z}_4')/(\tilde{h} - \tilde{h}')$ and $q \leftarrow \beta q/\beta$, $A \leftarrow a/(d_1^{1/\xi_1} d_2^{1/\xi_2})$, $r \leftarrow \beta r/\beta$, $s \leftarrow \beta s/\beta$. They meet $e(A, w g_2^r) = e(g_1, g_2^q u v^s)$, A is not among those that have appeared so far, and the advantage of $\mathcal{B}$ is at least $(\epsilon - 1/p)^2/16q_H$. □

Theorem 4.4. *If the user's signature scheme is (q_S', t', ϵ')-existentially unforgeable against chosen message attacks and the discrete logarithm problem in $\mathbb{G}_2$ is (t'', ϵ'')-hard, then the group signature scheme is $(t, q_H, q_S, n, \epsilon)$-non-frameable, where $\epsilon = \max(n\epsilon', 4n\sqrt{\epsilon'' q_H} + n/p)$ and $t = \max(t' - q_H O(1), \Theta(1) \cdot t'')$.*

Proof. Assume $\mathcal{A}$ is an algorithm that $(t, q_H, q_S, n, \epsilon)$-breaks the non-frameability of the group signature scheme. We construct an algorithm $\mathcal{B}$ that, by interacting with $\mathcal{A}$, breaks the user's signature scheme in time t' with advantage ϵ', where q_S is the maximum number of signing oracle queries (of the user's signature scheme) made by $\mathcal{B}$, or solves the discrete logarithm problem in time t'' with advantage ϵ''.

Algorithm $\mathcal{B}$ is given a random public key pk of the user's signature scheme and parameters $g_2, B = g_2^q \in \mathbb{G}_2$ of the discrete logarithm problem. It generates the components of the group signature public key, the Issuer key, and the Opener key, i.e., picks random $x, y, z, \xi_1, \xi_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and computes $g_1 \leftarrow \psi(g_2), w \leftarrow g_2^x, u \leftarrow g_2^y, v \leftarrow g_2^z, U \leftarrow g_2^{\xi_1}, V \leftarrow g_2^{\xi_2}$. It then provides to $\mathcal{A}$ the group public key $(g_1, g_2, w, u, v, U, V)$ the Issuer key x, and the Opener key (ξ_1, ξ_2).

If a member i is added, $\mathcal{B}$ picks random $q_i \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$, $r_i, s_i \stackrel{R}{\leftarrow} \mathbb{Z}_p$ (if $x + r_i = 0$, selects r_i again) and a random user's signature key pair sk_i, pk_i, and computes $B_i \leftarrow g_2^{q_i}$. It then carries out JOIN and ISSUE and writes $(A_i, B_i, r_i, s_i, sig_i)$ in Issuer's registration table $\boldsymbol{reg}[i]$. For one arbitrary member i', $\mathcal{B}$ uses pk and $B = g_2^q$ as $pk_{i'}$ and $B_{i'}$ respectively instead of random values in JOIN. In this process, $\mathcal{B}$ uses the signing oracle of the user's signature scheme to sign B for pk. When $\mathcal{A}$ corrupts the member i' for which $\mathcal{B}$ used pk and B, $\mathcal{B}$ halts. If $\mathcal{A}$ corrupts another member, $\mathcal{B}$ hands $\mathcal{A}$ its secret key.

At any time, $\mathcal{A}$ can query the random oracle H. Algorithm $\mathcal{B}$ responds with elements selected uniformly at random from $\mathbb{Z}_p$, making sure to respond identically to repeated queries. Moreover, $\mathcal{A}$ anytime can query the signing oracle of the group signature scheme for any member i and message m'. $\mathcal{B}$ generates sig-

	Our Scheme	BBS04 [4] without JOIN	KY05 [8]	ACMH05 [1] ($O(1)$OPEN)
JOIN	$3Exp_{E_{170}} + 2Pair$	-	$3Exp_{E_{195}} + 3Pair$	$22Exp_{E_{170}} + 4Pair$
ISSUE	$2Exp_{E_{170}} + Pair$	-	$3Exp_{E_{195}}$	$12Exp_{E_{170}} + Pair$
User→Issuer (bits)	512	-	1000	2043
User←Issuer (bits)	511	-	783	513
SIGN	$9Exp_{E_{170}}$	$7Exp_{E_{170}} + Exp_{\mathbb{Z}_{1020}}$	$6Exp_{E_{195}} + 36Exp_{\mathbb{Z}_{1024}} + 3Pair$	$43Exp_{E_{170}}$
VERIFY	$4Exp_{E_{170}} + 2Pair$	$5Exp_{E_{170}} + Exp_{\mathbb{Z}_{1020}} + Pair$	$3Exp_{E_{195}} + 21Exp_{\mathbb{Z}_{1024}} + 3Pair$	$13Pair$
Sig.Length (bits)	2045	1533	24813	3249

Table 1: Comparison

nature values $(a, b, c, d_1, d_2, \tilde{h}, \tilde{Z}_1, \ldots, \tilde{Z}_6)$ correctly using A_i, r_i, s_i, q_i. If $\mathcal{A}$ requests i''s signature, $\mathcal{B}$ picks randomly $\tilde{h}, \tilde{Z}_1, \ldots, \tilde{Z}_6 \stackrel{R}{\leftarrow} \mathbb{Z}_p$, computes $\tilde{t}'_1, \ldots, \tilde{t}'_4$ from $\tilde{h}, \tilde{Z}_1, \ldots, \tilde{Z}_6$, and patches $H(\tilde{t}'_1, \tilde{t}'_2, \tilde{t}'_3, \tilde{t}'_4, m')$ to equal $\tilde{h}$. $\mathcal{B}$ returns the signature to $\mathcal{A}$.

Eventually, $\mathcal{A}$ outputs a valid message-signature pair $(m, \sigma = (a, b, c, d_1, d_2, \tilde{h}, \tilde{Z}_1, \ldots, \tilde{Z}_6))$, an uncorrupted member with identity j, and a proof-string $\tau = (sig_j, B_j, A_j, r_j, s_j, X_1, X_2, \hat{h}, \hat{Z}_1, \hat{Z}_2)$ which is acceptable with (m, σ, j) by JUDGE. This means that $A_j = a/(d_1^{\xi_1} d_2^{\xi_2})$, $e(A_j, wg_2^{r_j}) = e(g_1, B_j uv^{s_j})$, and that sig_j is a valid signature on B_j for pk_j.

Assume sig_j is a forgery. If j is not the member i' for which $\mathcal{B}$ used pk as pk_j, $\mathcal{B}$ halts. Else, the theorem is proven and advantage $\epsilon' = \epsilon/n$. If sig_j is not a forgery, B_j is surely the value sent by j to Issuer in JOIN (B_j in $\boldsymbol{reg}[j]$ has not been rewritten). If j is not the member i' for which $\mathcal{B}$ used $B = g_2^q$ as B_j, $\mathcal{B}$ halts. Else, it is possible that A_j, r_j, s_j that meets $e(A_j, wg_2^{r_j}) = e(g_1, Buv^{s_j})$ and $r'_j, s'_j, B' = g_2^{q'}$ that meets $e(A_j, wg_2^{r'_j}) = e(g_1, B'uv^{s'_j})$ were newly generated by $\mathcal{A}$, the former was written into $\boldsymbol{reg}[j]$, and the latter was used to comprise σ. However, (q', r'_j, s'_j) can be extracted from σ in the same way as theorem 4.3 and $(q + y + s_j z)/(x + r_j) = (q' + y + s'_j z)/(x + r'_j)$ holds, so $\mathcal{B}$ can compute q. Consequently, $\mathcal{B}$ can obtain the discrete logarithm q of B with advantage $\epsilon'' = (\epsilon/n - 1/p)^2/16q_H$ in time $t'' = \Theta(1) \cdot t$. □

5 Comparison

We show the comparison of our scheme with other schemes in table 1. Exp_{E_N} and $Exp_{\mathbb{Z}_{N'}}$ are the costs for computing a multi-exponentiation on an elliptic curve with N-bit order and on $\mathbb{Z}_{N'}$ respectively. $Pair$ represents the costs for computing a pairing. User→Issuer and User←Issuer represent the volume of communication in JOIN and ISSUE. Sig.Length is the length of a signature.

ACMH05 proposed group signature schemes with $O(n)$ and $O(1)$ OPEN algorithms, where n is the number of members of the group. Although SIGN and VERIFY of the $O(n)$ OPEN scheme is more efficient than that of the $O(1)$ OPEN scheme, we compare our scheme with the $O(1)$ OPEN scheme because OPEN of our scheme is $O(1)$.

6 Conclusion

We presented a group signature scheme based on the 2 Variable Strong Diffie-Hellman assumption and the Decision Linear Diffie-Hellman assumption. This scheme has an efficient concurrent join protocol and its signature length is short. Our scheme is the most efficient and practical group signature scheme among group signature schemes that have efficient concurrent join protocols and are provably secure with/without random oracles.

References

[1] Ateniese, G., Camenisch, J., de Medeiros, B. and Hohenberger, S., "Practical Group Signatures without Random Oracles," `http://eprint.iacr.org/2005/385`, 2005.

[2] Bellare, M., Shi, H. and Zhang, C., "Foundations of Group Signatures: The Case of Dynamic Groups," In *Proceedings of CT-RSA 2005*, LNCS Vol. 3376, pp. 136–153.

[3] Boneh, D. and Boyen, X., "Short Signatures without Random Oracles," In *Proceedings of EUROCRYPT 2004*, LNCS Vol. 3027, pp. 56–73.

[4] Boneh, D., Boyen, X. and Shacham, H., "Short Group Signatures," In *Proceedings of CRYPTO 2004*, LNCS Vol. 3152, pp. 41–55.

[5] Boyen, X. and Waters, B., "Compact Group Signatures Without Random Oracles," `http://eprint.iacr.org/2005/381`, 2005.

[6] Chaum, D. and van Heyst, E., "Group Signatures," In *Proceedings of EUROCRYPT 1991*, LNCS Vol. 547, pp. 257–265.

[7] Furukawa, J. and Imai, H., "An Efficient Group Signature Scheme from Bilinear Maps," In *Proceedings of ACISP 2005*, LNCS Vol. 3574, pp. 455–467.

[8] Kiayias, A. and Yung, M., "Group Signatures with Efficient Concurrent Join," `http://eprint.iacr.org/2005/345`, 2005.

[9] Okamoto, T., "Efficient Blind and Partially Blind Signatures Without Random Oracles," To appear in *Proceedings of TCC 2006*.

[10] Pointcheval, D. and Stern, J., "Security Arguments for Digital Signatures and Blind Signatures," *Journal of Cryptology*, Vol. 13, No. 3, pp. 361–396, 2000.