Breaking and Repairing Trapdoor-free Group Signature Schemes from Asiacrypt 2004

Xinyi Huang[1], Willy Susilo[2] and Yi Mu[2]

[1]College of Mathematics and Computer Science
Nanjing Normal University, P.R. China
Email: xinyinjnu@126.com
[2]Centre for Information Security Research
School of Information Technology and Computer Science
University of Wollongong, Australia
Email: {wsusilo, ymu}@uow.edu.au

Abstract. Group signature schemes allow a member of a group to sign messages anonymously on behalf of the group. In the case of later dispute, a designated group manager can revoke the anonymity and identify the originator of a signature. In Asiacrypt 2004, Nguyen and Safavi-Naini proposed a group signature scheme that has a constant-size public key and signature length, and more importantly, their group signature scheme does *not* require trapdoor. Their scheme is very efficient and the sizes of signatures are shorter compared to the existing schemes that were proposed earlier. In this paper, we point out that Nguyen and Safavi-Naini's scheme is insecure. In particular, we provide a cryptanalysis of the scheme that allows a non-member of the group to sign on behalf of the group. The resulting group signature can convince any third party that a member of the group has indeed generated such a signature, although none of the members has done it. Therefore, in the case of dispute, the group manager cannot identify who has signed the message. We also provide a new scheme that does not suffer against this problem.

1 Introduction

Chaum and van Heyst proposed a new type of signature scheme for a group of entities, called *group signatures* in [10]. Such a scheme allows a group member to sign a message on behalf of the group such that everybody can verify the signature but *no one* can identify which group member provided it. However, there is a designated group manager who can reveal the identity of the originator of a signature in the case of later dispute. This act is referred to as "opening" a signature or also as revocation of a signer's anonymity. The role of a group manager is also to register new users by issuing membership certificates that contain registration details. In some schemes, the functions of the group manager can be split between two managers: an *issuer* and an *opener*.

In early group signature schemes [6, 11], the size of the public key and the signature grew linearly with the size of the group, and hence, the schemes were impractical for large groups. Schemes with fixed size group public key and signature length have been proposed

in [1,8,9]. In particular, an efficient group signature scheme with very short length and low computation cost has been proposed in [1]. An efficient group signature scheme without trapdoor, in the sense that none of parties in the system including the group manager need to know the trapdoor, has been proposed in [2]. A short group signature scheme has been proposed in [4]. More recently, Nguyen and Safavi-Naini proposed an efficient provably secure trapdoor-free group signature scheme in [13].

Our Contribution

In this paper, we point out that the scheme proposed in [13] is insecure. In particular, we shall show that anyone can *sign* on behalf of the group and convince any third party that a member of the group has indeed generated such a signature. More importantly, the group manager cannot revoke the identity of the signer since the signer is not a group member. We also show how to fix this scheme to make the scheme secure.

1.1 Cryptographic Tools

Basic Concepts on Bilinear Pairings

Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic additive groups generated by P_1, P_2, respectively, whose order are a prime q. Let $\mathbb{G}_M$ be a cyclic multiplicative group with the same order q. We assume there is an isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ such that $\psi(P_2) = P_1$. Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_M$ be a bilinear mapping with the following properties:

1. *Bilinearity*: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}_2, a, b, \in \mathbb{Z}_q$.
2. *Non-degeneracy*: There exists $P \in \mathbb{G}_1, Q \in \mathbb{G}_2$ such that $e(P, Q) \neq 1$.
3. *Computability*: There exists an efficient algorithm to compute $e(P, Q)$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}_2$.

For simplicity, hereafter, we set $\mathbb{G}_1 = \mathbb{G}_2$ and $P_1 = P_2$. We note that our scheme can be easily modified for a general case, when $\mathbb{G}_1 \neq \mathbb{G}_2$.

Bilinear pairing instance generator is defined as a probabilistic polynomial time algorithm $\mathcal{IG}$ that takes as input a security parameter ℓ and returns a uniformly random tuple $param = (p, \mathbb{G}_1, \mathbb{G}_M, e, P)$ of bilinear parameters, including a prime number p of size ℓ, a cyclic additive group $\mathbb{G}_1$ of order q, a multiplicative group $\mathbb{G}_M$ of order q, a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_M$ and a generator P of $\mathbb{G}_1$. For a group $\mathbb{G}$ of prime order, we denote the set $\mathbb{G}^* = \mathbb{G} \setminus \{\mathcal{O}\}$ where $\mathcal{O}$ is the identity element of the group.

Signature of Knowledge

The first signature based on proof of knowledge (SPK) was proposed in [6, 7]. We will use the following definition of SPK from [7].

Let q be a large prime and $p = 2q+1$ be also a prime. Let G be a finite cyclic group of prime order p. Let g be a generator of $\mathbb{Z}_p^*$ such that computing discrete logarithms of any group elements (apart from the identity element) with respect to one of the generators is infeasible. Let $H : \{0,1\}^* \rightarrow \{0,1\}^\ell$ denote a strong collision-resistant hash function.

Definition 1. *A pair* $(c, s) \in \{0,1\}^\ell \times \mathbb{Z}_q$ *satisfying* $c = H(g||y||g^s y^c||m)$ *is a signature based on proof of knowledge of discrete logarithm of a group element* y *to the base* g *of the message* $m \in \{0,1\}^*$ *and is denoted by* $SPK\{\alpha : y = g^\alpha\}(m)$.

An $SPK\{\alpha : y = g^\alpha\}(m)$ can only be computed *iff* the value (secret key) $\alpha = log_g(y)$ is known. This is also known as a non-interactive proof of the knowledge α.

This technique can be applied to elliptic curve domain. For completeness, we illustrate the technique as follows.

Definition 2. *A pair* $(c, s) \in \mathbb{Z}_q^2$ *satisfying* $c = H(P||Q||sP+cQ||m)$ *is a signature based on proof of knowledge of elliptic curve discrete logarithm of a group element* Q *to the base* P *of the message* $m \in \{0,1\}^*$ *and is denoted by* $ECSPK\{\alpha : Q = \alpha P\}(m)$.

We note that $ECSPK\{\alpha : Q = \alpha P\}(m)$ can only be computed *iff* the value of a, where $Q = aP$, is known. It can be computed as follows. Firstly, select a random $z \in \mathbb{Z}_q^*$ and compute $c = H(P||Q||zP||m)$, and then, compute $s = z - ca \pmod q$. Using the same technique, the following definition can be derived.

We extend this technique to provide a signature based on proof of knowledge of elliptic curve discrete logarithm of values α and β, where $\alpha, \beta \in \mathbb{Z}_q^*$, such that the statement $R = \alpha P + \beta Q$ is correct, for both P, Q be group elements. The signature on proof of knowledge is defined as follows.

Definition 3. *A tuple* $(c, \omega_1, \omega_2) \in \mathbb{Z}_q^3$ *satisfying* $c = H(P||Q||R||\omega_1 P + \omega_2 Q + cR||m)$ *is a signature based on proof of knowledge of elliptic curve discrete logarithm of values* α *and* β*, where* $\alpha, \beta \in \mathbb{Z}_q^*$*, such that the statement* $R = \alpha P + \beta Q$ *is correct, for both* P, Q *be group elements, of the message* $m \in \{0,1\}^*$ *and is denoted by* $ECSPK\{\alpha, \beta : R = \alpha P + \beta Q\}(m)$.

We note that $ECSPK\{\alpha, \beta : R = \alpha P + \beta Q\}(m)$ can only be computed *iff* the value of α and β satisfying $R = \alpha P + \beta Q$ is known. The computation is done as follows. Firstly, select two random numbers $\tau_1, \tau_2 \in \mathbb{Z}_q^*$ and compute $c = H(P||Q||R||\tau_1 P + \tau_2 Q)$, and then, compute $\omega_1 = \tau_1 - c\alpha \pmod q$ and $\omega_2 = \tau_2 - c\beta \pmod q$.

We note that this type of signature of knowledge has been widely used and extended, for instance in [2, 5], to provide a proof that the value of α, β lie in a specified interval.

We denote this type of proof as

$$ECSPK\{\alpha, \beta : R = \alpha P + \beta Q \ \wedge \ \alpha \in [\gamma_1, \gamma_2] \ \wedge \ \beta \in [\gamma_3, \gamma_4]\}(m)$$

to show a proof of knowledge on α, β, where α relies in the interval $[\gamma_1, \gamma_2]$ and β relies in the interval $[\gamma_3, \gamma_4]$, on a message $m \in \{0,1\}^*$.

Organization of the Paper

The rest of this paper is organized as follows. In section 2, we review the scheme proposed in [13], by firstly reviewing the model of group signature schemes that they used. In section 3, we present our motivation of the attack and we show a cryptanalysis of the scheme presented in [13]. In section 4, we show our modification to the scheme in [13] to make the scheme secure. Section 5 concludes the paper.

2 Review of Nguyen-Safavi-Naini's Group Signature Schemes

Nguyen-Safavi-Naini's group signature schemes use the formal model proposed in [3]. A group signature scheme consists of a trusted party for initial setup, two group managers, i.e. the issuer and the opener, and users with unique identities $i \in \mathbb{N}$, who can join the group and become a group member. The scheme consists of a tuple $\mathcal{GS} = (\mathsf{GKg}, \mathsf{UKg}, \mathsf{Join}, \mathsf{Iss}, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Open}, \mathsf{Judge})$, which are polynomial algorithms. GKg is the group key generation algorithm that outputs a triple of keys (gpk, ik, ok), where gpk is the group public key, ik is given to the issuer and ok is given to the opener. UKg is a user key generation algorithm that outputs a personal public and private key pair. $\mathsf{Join}, \mathsf{Iss}$ are interactive algorithms performed by a user and the issuer as two sides of a group joining protocol. GSig is the group signing algorithm, and GVf is the group verification algorithm. $\mathsf{Open}, \mathsf{Judge}$ are deterministic algorithms that are used in the case of dispute. For a formal definition of these algorithms, we refer the reader to [13].

El Gamal^{BP2} Encryption Scheme

An encryption scheme called *El Gamal*^{BP2} is introduced in [13] as part of their group signature scheme. This scheme is the bilinear pairing version of the scheme presented and proved by Fouque and Pointcheval in [12], that uses the twin-encryption paradigm and a simulation-sound proof of equality of plaintexts. We refer the reader to [13] for a more complex account.

2.1 Review of Nguyen-Safavi-Naini's Group Signature Scheme

Nguyen-Safavi-Naini's group signature scheme is defined by the following algorithms.

– GKg: Suppose l is a security parameter and the Bilinear Pairing Instance Generator $\mathcal{G}$ generates a tuple of bilinear pairing parameter $\mathbf{t} = (p, \mathbb{G}_1, \mathbb{G}_M, e, P) \leftarrow \mathcal{G}(1^l)$, that is also the publicly shared parameters. Choose a hash function $\mathcal{H}_2 : \{0,1\}^* \rightarrow \mathbb{Z}_p$, which is assumed to be a random oracle in the security proofs.
Choose $P_0, G, H \in_R \mathbb{G}_1$, $x, x'_a, x'_b \in_R \mathbb{Z}_p^*$ and compute $P_{pub} = xP$, $\Theta_a = e(G,G)^{x'_a}$ and $\Theta_b = e(G,G)^{x'_b}$. The group public key is $gpk = (P, P_0, P_{pub}, H, G, \Theta_a, \Theta_b)$, the issuing key is $ik = x$, and the opening key is $ok = (x'_a, x'_b)$.
– UKg: This algorithm generates keys that provide authenticity for messages sent by the user in the (Join, Iss) protocol. This algorithm is the key generation algorithm K_S of any digital signature scheme $(K_S, Sign, Ver)$ that is unforgeable against chosen message attacks (UNF-CMA). A user i runs the UKg algorithm that takes as input a security parameter 1^l and outputs a personal public and private signature key pair $(upk[i], usk[i])$.

– Join, Iss: In this protocol, a user i and the issuer first jointly generate a random value $x_i \in \mathbb{Z}_p^*$ whose value is only known by the user. The issuer then generates (a_i, S_i) for the user so that $e(a_iP + P_{pub}, S_i) = e(P, x_iP + P_0)$. The user uses $usk[i]$ to sign his messages in the protocol. Note that the formal model assumes the communication to be private and authenticated. It is assumed that the communication is protected from replay attacks. The protocol is as follows.

1. user $i \rightarrow$ issuer: $I = yP + rH$, where $y, r \in_R \mathbb{Z}_p^*$.
2. user $i \leftarrow$ issuer: $u, v \in_R \mathbb{Z}_p^*$.
3. The user computes $x_i = uy + v, P_i = x_iP$
4. user $i \rightarrow$ issuer: P_i and a proof of knowledge of (x_i, r') such that $P_i = x_iP$ and $vP + uI - P_i = r'H$.
5. The issuer verifies the proof, then chooses $a_i \in_R \mathbb{Z}_p^*$ different from all corresponding elements previously issued, and computes $S_i = \frac{1}{a_i + x}(P_i + P_0)$.
6. user $i \leftarrow$ issuer: a_i, S_i.
7. The user computes $\Delta_i = e(P, S_i)$, verifies if $e(a_iP + P_{pub}, S_i) = e(P, x_iP + P_0)$, and stores the *private signing key* $gsk[i] = (x_i, a_i, S_i, \Delta_i)$. Note that only the user knows x_i. The issuer also computes Δ_i and makes an entry in the table reg : $reg[i] = (i, \Delta_i, <$ Join, Iss $>$ $transcript)$.

– GSig: A group signature of a user i shows his knowledge of (a_i, S_i) and a secret x_i such that: $e(a_iP + P_{pub}, S_i) = e(P, x_iP + P_0)$. The signature does not reveal any information about his knowledge to anyone, except for the opener, who can compute Δ_i by decrypting an encryption of that value. The algorithm for a user i to sign a message $m \in \{0,1\}^*$ is as follows.

1. Encrypt Δ_i by El Gamal^{BP_2} with public key (G, Θ_a, Θ_b) as $(E_a = tG, \Lambda_a = \Delta_i \Theta_a^t, E_b, \Lambda_b, \varsigma)$
2. Perform the non-interactive version of a protocol, which the authors call the Signing protocol, as follows.
   (a) Generate $r_1, \cdots, r_3, k_0, \cdots, k_5 \in_R \mathbb{Z}_q^*$ and computes:

$$U = r_1(a_i P + P_{pub}); V = r_2 S_i,$$

$$W = r_1 r_2 (x_i P + P_0); X = r_2 U + r_3 H,$$

$$T_1 = k_1 P + k_2 P_{pub} + k_0 H; T_2 = k_3 P + k_2 P_0,$$

$$T_3 = k_4 U + k_0 H; T_4 = k_5 G - k_4 E_a; \Pi = \Theta_a^{k_5} \Lambda_a^{-k_4}.$$

   (b) Computes $c = \mathcal{H}_2(P||P_0||P_{pub}||H||G||\Theta_a||\Theta_b||E_a||\Lambda_a||E_b||\Lambda_b||\varsigma||U||\ V||W||X||T_1|| \cdots T_4||\Pi||m)$
   (c) Computes in $\mathbb{Z}_p$:

$$\begin{aligned} s_0 &= k_0 + cr_3, \\ s_1 &= k_1 + cr_1 r_2 a_i, \\ s_2 &= k_2 + cr_1 r_2, \\ s_3 &= k_3 + cr_1 r_2 x_i, \\ s_4 &= k_4 + cr_2, \\ s_5 &= k_5 + cr_2 t. \end{aligned}$$

3. Outputs the signature $(c, s_0, \cdots, s_5, U, V, W, X, E_a, \Lambda_a, E_b, \Lambda_b, \varsigma)$ for message m.

– GVf: The verification algorithm for m, $(c, s_0, \cdots, s_5, U, V, W, X, E_a, \Lambda_a, E_b, \Lambda_b, \varsigma)$ outputs accept if and only if verifying the proof ς outputs accept and the following two equations hold.

$$\begin{aligned} e(U, V) &\stackrel{?}{=} e(P, W) \\ c &\stackrel{?}{=} \mathcal{H}_2(P||P_0||P_{pub}||H||G||\Theta_a||\Lambda_a||E_b||\Lambda_b||\varsigma||U||V||W||X|| \\ &\quad s_1 P + s_2 P_{pub} + s_0 H - cX||s_3 P + s_2 P_0 - cW|| \\ &\quad s_4 U + s_0 H - cX||s_5 G - s_4 E_a||\Theta_a^{s_5} \Lambda_a^{-s_4} e(P, cV)||m) \end{aligned}$$

– Open: To open m and its valid signature $(c, s_0, \cdots, s_5, U, V, W, X, E_a, \Lambda_a, E_b, \Lambda_b, \varsigma)$ to find the signer, the opener performs the following steps.

1. Use the GVf algorithm to check the signature's validity. If the algorithm rejects, return $(0, \varepsilon)$, where ε denotes an empty string.
2. Computes $\Delta_i = \Lambda_a e(E_a, G)^{-x'_a}$ and find the corresponding entry i in the table reg. If no entry is found, return $(0, \varepsilon)$.
3. Return $Reg[i]$ and a non-interactive zero-knowledge proof ϱ of knowledge of x'_a so that $\Theta_a = e(G, G)^{x'_a}$ and $\Lambda_a / \Delta_i = e(E_a, G)^{x'_a}$.

– Judge: On an output by the Openalgorithm for a message m and its signature ω, the Judge algorithm is performed as follows:

1. If Open algorithm outputs $(0, \varepsilon)$, run GVf algorithm on m, ω. If GVf rejects, return accept; otherwise, return reject.
2. If Open algorithm outputs $(reg[i], \varrho)$, return reject if one of the following happens: (i) on m, ω, GVf algorithm rejects; (ii) verification of the proof ϱ rejects; (iii) the $<$ join, Iss $>$ transcript is invalid with regard to $upk[i]$; (iv) $\Delta_i \neq e(P_i, S_i)$ where S_i is extracted from the $<$ join, Iss $>$ transcript. Otherwise, return accept.

3 Cryptanalysis of Nguyen-Safavi-Naini's Group Signature Schemes

3.1 Background

Our attack is inspired by the proof of the following lemma from [13].

Lemma 1. *[13] The interactive Signing protocol underlying the* GSig *algorithm is a (honest-verifier) perfect zero-knowledge proof of knowledge of* (a_i, S_i), x_i *and* t *such that* $e(a_i P + P_{pub}, S_i) = e(P, x_i P + P_0)$, $E_a = tG$ *and* $\Lambda_a = e(P, S_i)\Theta_a^t$.

The proof of the above Lemma presented in [13] is summarized as follows. If the protocol accepts with non-negiligible probably, the interactive signing protocol underlying the GSig algorithm is a (honest-verifier) perfect zero-knowledge proof of knowledge of (a_i, S_i), x_i and t such that $e(a_i P + P_{pub}, S_i) = e(P, x_i P + P_0)$, $E_a = tG$ and $\Lambda_a = e(P, S_i)\Theta_a^t$ [13]. The Soundness is justified as follows.

Soundness [13]: Suppose the protocol accepts for the same commitment $(U, V, W, X, T_1, \cdots, T_4, \Pi)$, two different pairs of challenges and responses $(c, s_0, \cdots, s_5)$ and $(c', s'_0, \cdots, s'_5)$. Let $f_i = \frac{s_i - s'_i}{c - c'}$, $i = 0, \cdots, 5$, then $X = f_1 P + f_2 P_{pub} + f_0 H; W = f_3 P + f_2 P_0$ and

$X = f_4U + f_0H; E_a = f_5f_4^{-1}G; e(P,V) = \Theta^{-f_5}\Lambda_a^{f_4}$ so $U = f_1f_4^{-1}P + f_2f_4^{-1}P_{pub}$. Furthermore, let $a_i = f_1f_2^{-1}, S_i = f_4^{-1}, x_i = f_3f_2^{-1}, t = f_5f_4^{-1}$, then $E_a = tG$, $\Lambda_a = e(P,S_i)\Theta_a^t$, and $e(a_iP + P_{pub}, S_i) = e(P, x_iP + P_0)$.

Checking the above proof, the value of U is set to $U = f_1f_4^{-1}P + f_2f_4^{-1}P_{pub}$, but the authors fail to ensure that f_2 must be a non-zero value in this equation. Moreover, f_2 is set to $f_2 = \frac{(s_2 - s_2')}{(c-c')}$ but there is no assurance that s_2 and s_2' must be different that will lead f_2 to zero.

When the value of f_2 is set to zero, then U can be created by *anyone* who is not a group member to sign a message on behalf of the group. The detail of the attack is presented below.

3.2 Attack on Nguyen-Safavi-Naini's Group Signature Schemes

In this section, we show that a non-group member can sign on behalf of the group by performing the following GSig algorithm.

GSig Algorithm.

1. set $\Delta_i = e(P,P)$, also encrypt the Δ_i by El Gamal^{BP2} with the group public key as $(E_a = t_aG, \Lambda_a = \Delta_i\Theta_a^{t_a}, E_b, \Lambda_b, \varsigma)$, here $t_a \in \mathbb{Z}_p$ is randomly chosen by the adversary-Lan described the encryption scheme in section 3.2 .
2. Performs the non-interactive version of a protocol as follows:
   (a) Generate $r_1, \cdots, r_3, k_0, \cdots, k_5 \in_R \mathbb{Z}_p^*$ and compute:

$$
\begin{aligned}
U &= r_0r_1P, \\
V &= r_2P, \\
W &= r_0r_1r_2P, \\
X &= r_2U + r_3H, \\
T_1 &= k_1P + (k_2 + r_1r_2)P_{pub} + k_0H, \\
T_2 &= k_3P + (k_2 + r_1r_2)P_0, \\
T_3 &= k_4U + k_0H, \\
T_4 &= k_5G + k_4E_a, \\
\Pi &= \Theta_a^{k_5}\Lambda_a^{-k_4}.
\end{aligned}
$$

   (b) Compute $c = \mathcal{H}_2(P||P_0||P_{pub}||H||G||\Theta_a||\Theta_b||E_a||\Lambda_a||E_b||\Lambda_b||\varsigma||U||\ V||W||X||T_1||\cdots T_4||\Pi||m)$

(c) Compute in $\mathbb{Z}_p$:

$$
\begin{aligned}
s_0 &= k_0 + cr_3, \\
s_1 &= k_1 + cr_1r_2r_0, \\
s_2 &= k_2 + r_1r_2, \\
s_3 &= k_3 + cr_0r_1r_2, \\
s_4 &= k_4 + cr_2, \\
s_5 &= k_5 + cr_2t_a,
\end{aligned}
$$

3. Output the signature $(c, s_0, s_1, s_2, s_3, s_4, s_5, U, V, W, X, E_a, \Lambda_a, E_b, \Lambda_b, \varsigma)$ for message m.

Remarks: We note that the above GSig algorithm can be performed by *anyone* who does not have any knowledge of (a_i, S_i) and a secret x_i.

Theorem 1. *The above* GSig *algorithm can be used to sign a message on behalf of the group. Anyone can be convinced with the signature that a group member has indeed signed the message, by invoking the* GVf *algorithm. Hence, the group signature scheme is insecure.*

Proof. To show the correctness of the above theorem, we need to show that the output of GVf algorithm is accept, given a signature on a message that is produced by the above GSig algorithm. Note that the GVf algorithm can be invoked by anyone. The GVf algorithm that we will use is the original GVf algorithm as defined in [13]. We will show this argument as follows.

GVf Algorithm.

1. Verify whether the proof ς outputs *accept*
   We can easily see that if the encryption scheme is done correctly, it will definitely pass this verification. This is true because ς only provides a proof of the equality of the plaintexts between (E_a, Λ_a) and (E_b, Λ_b).
2. Verify whether $e(U, V) \stackrel{?}{=} e(P, W)$
   From the signature scheme, we can find that: $e(U, V) = e(r_0r_1P, r_2P) = e(P, r_0r_1r_2P) = e(P, W)$.
3. Verify whether

$$
\begin{aligned}
c = \mathcal{H}_2(&P||P_0||P_{pub}||H||G||\Theta_a||\Lambda_a||E_b||\Lambda_b||\varsigma||U||V||W||X|| \\
&s_1P + s_2P_{pub} + s_0H - cX||s_3P + s_2P_0 - cW|| \\
&s_4U + s_0H - cX||s_5G - s_4E_a||\Theta_a^{s_5}\Lambda_a^{-s_4}e(P, cV).||m)
\end{aligned}
$$

The signature can only pass this verification, if and only if the following equations hold with equality.

$$\begin{aligned}
T_1 &= s_1P + s_2P_{pub} + s_0H - cX, \\
T_2 &= s_3P + s_2P_0 - cW, \\
T_3 &= s_4U + s_0H - cX, \\
T_4 &= s_5G - s_4E_a, \\
\Pi &= \Theta_a^{s_5}\Lambda_a^{-s_4}e(P, cV).
\end{aligned}$$

We will show this verification as follows.

(a) $T_1 \stackrel{?}{=} s_1P + s_2P_{pub} + s_0H - cX$

$$\begin{aligned}
& s_1P + s_2P_{pub} + s_0H - cX \\
&= (k_1 + cr_1r_2r_0)P + (k_2 + r_1r_2)P_{pub} + (k_0 + cr_3)H - c(r_2U + r_3H) \\
&= k_1P + cr_1r_2r_0P + k_2P_{pub} + r_1r_2P_{pub} + k_0H + cr_3H - cr_2r_0r_1P - cr_3H \\
&= k_1P + (k_2 + r_1r_2)P_{pub} + k_0H = T_1.
\end{aligned}$$

(b) $T_2 \stackrel{?}{=} s_3P + s_2P_0 - cW$

$$\begin{aligned}
& s_3P + s_2P_0 - cW \\
&= (k_3 + cr_0r_1r_2)P + (k_2 + r_1r_2)P_0 - cr_0r_1r_2P \\
&= k_3P + cr_0r_1r_2P + (k_2 + r_1r_2)P_0 - cr_0r_1r_2P \\
&= k_3P + (k_2 + r_1r_2)P_0 = T_2.
\end{aligned}$$

(c) $T_3 \stackrel{?}{=} s_4U + s_0H - cX$

$$\begin{aligned}
& s_4U + s_0H - cX \\
&= (k_4 + cr_2)U + (k_0 + cr_3)H - c(r_2U + r_3H) \\
&= k_4U + cr_2U + k_0H + cr_3H - cr_2U - cr_3H \\
&= k_4U + k_0H = T_3.
\end{aligned}$$

(d) $T_4 \stackrel{?}{=} s_5G - s_4E_a$

$$\begin{aligned}
& s_5G - s_4E_a \\
&= (k_5 + cr_2t_a)G - (k_4 + cr_2)t_agG \\
&= k_5G + cr_2t_aG - k_4t_aG - cr_2t_aG \\
&= k_5G - k_4t_aG = k_5G - k_4E_a = T_4.
\end{aligned}$$

(e) $\Pi \stackrel{?}{=} \Theta_a^{s_5} \Lambda_a^{-s_4} e(P, cV)$

$$
\begin{aligned}
& \Theta_a^{s_5} \Lambda_a^{-s_4} e(P, cV) \\
&= \Theta_a^{(k_5 + cr_2 t_a)} \cdot (\Delta_i \Theta_a^{t_a})^{-(k_4 + cr_2)} \cdot e(P, cr_2 P) \\
&= \Theta_a^{k_5} \cdot \Theta_a^{cr_2 t_a} \cdot (\Delta_i)^{-(k_4 + cr_2)} \cdot (\Theta_a)^{-t_a(k_4 + cr_2)} \cdot e(P, P)^{cr_2} \\
&= \Theta_a^{k_5} \cdot \Theta_a^{cr_2 t_a} \cdot e(P, P)^{-(k_4 + cr_2)} \cdot (\Theta_a)^{-t_a k_4} \cdot (\Theta_a)^{-t_a cr_2} \cdot e(P, P)^{cr_2} \\
&= \Theta_a^{k_5} \cdot e(P, P)^{-k_4} \cdot (\Theta_a)^{-t_a k_4} \\
&= \Theta_a^{k_5} (\cdot e(P, P) \Theta_a^{t_a})^{-k_4} \\
&= \Theta_a^{k_5} \Lambda^{-k_4} = \Pi.
\end{aligned}
$$

From all of the above verification steps, we can find that the signature passes all the verification tests using the group's public key Θ_a. The receiver can believe that the signature was originated from the group. However the Open algorithm can not find the corresponding item in reg. This is due to $\Lambda_a e(E_a, G)^{-x'_a} = \Delta_i = e(P, P)$.

□

4 Our Improved Scheme

In this section, we present our modification to the scheme presented in [13] to make the scheme secure. As illustrated in section 3, the scheme presented in [13] is vulnerable against our attack due to the ability of an attacker to set the second part of U to zero. Therefore, to avoid such an attack, a simple solution is to employ a proof that the representation of U about P_{pub} and W about P_0 is not zero. Therefore, the GSig algorithm is modified as follows.

GSig:
The algorithm for a user i to sign a message $m \in \{0,1\}^*$ is as follows.

1. Encrypt Δ_i by El Gamal^{BP_2} with public key (G, Θ_a, Θ_b) as $(E_a = tG, \Lambda_a = \Delta_i \Theta_a^t, E_b, \Lambda_b, \varsigma)$

2. Perform the non-interactive version of a protocol, which the authors call the Signing protocol, as follows.
   (a) Generate $r_1, \cdots, r_3$, $k_0, \cdots, k_5 \in_R \mathbb{Z}_q^*$ and computes:

$$U = r_1(a_i P + P_{pub}); V = r_2 S_i,$$

$$W = r_1 r_2 (x_i P + P_0); X = r_2 U + r_3 H,$$

$$T_1 = k_1P + k_2P_{pub} + k_0H; T_2 = k_3P + k_2P_0,$$
$$T_3 = k_4U + k_0H; T_4 = k_5G - k_4E_a; \Pi = \Theta_a^{k_5}\Lambda_a^{-k_4}.$$

(b) Provide the following proof

$$\Gamma_1 = ECSPK\{\alpha, \beta : U = \alpha P + \beta P_{pub} \ \wedge \ \alpha \in [1, q-1] \ \wedge \ \beta \in [1, q-1]$$

$$\Gamma_2 = ECSPK\{\alpha, \beta : W = \alpha P + \beta P_0 \ \wedge \ \alpha \in [1, q-1] \ \wedge \ \beta \in [1, q-1]$$

(c) Computes $c = \mathcal{H}_2(P||P_0||P_{pub}||H||G||\Theta_a||\Theta_b||E_a||\Lambda_a||E_b||\Lambda_b||\varsigma||U||\ V||W||X||T_1||\cdots T_4||\Pi||m)$

(d) Computes in $\mathbb{Z}_p$:

$$s_0 = k_0 + cr_3,$$
$$s_1 = k_1 + cr_1r_2a_i,$$
$$s_2 = k_2 + cr_1r_2,$$
$$s_3 = k_3 + cr_1r_2x_i,$$
$$s_4 = k_4 + cr_2,$$
$$s_5 = k_5 + cr_2t.$$

3. Outputs the signature $(c, s_0, \cdots, s_5, U, V, W, X, E_a, \Lambda_a, E_b, \Lambda_b, \varsigma, \Gamma_1, \Gamma_2)$ for message m.

As part of the verification algorithm GVf , the correctness of Γ_1, Γ_2 must be verified.

5 Conclusion

In this paper, firstly we identified an attack to the recently proposed group signature schemes by Nguyen and Safavi-Naini in Asiacrypt 2004. We have shown how to fix the scheme to obtain a provably secure scheme as claimed in [13].

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. *Advances in Cryptology - CRYPTO 2000, Lecture Notes in Computer Science 1880*, pages 255–270, 2000.
2. G. Ateniese and B. de Medeiros. Efficient Group Signatures without Trapdoors. *Advances in Cryptology - Asiacrypt 2003, Lecture Notes in Computer Science 2894*, pages 246 – 268, 2003.

3. M. Bellare, H. Shi, and C. Zhang. Foundations of Group Signatures: The case of dynamic groups. *Cryptology ePrint Archive: Report 2004/077*.
4. D. Boneh, X. Boyen, and H. Shacham. Short Group Signatures. *Advances in Cryptology - Crypto '04, Lecture Notes in Computer Science 3152*, 2004.
5. F. Boudot. Efficient Proofs that A Committed Number lies in an Interval. *Advances in Cryptology - Eurocrypt 2000, Lecture Notes in Computer Science 1807*, pages 431 – 444, 2000.
6. J. Camenisch. Efficient and generalized group signatures. *Advances in Cryptology - Eurocrypt '97, Lecture Notes in Computer Science 1233*, pages 465–479, 1997.
7. J. Camenisch. Group signature schemes and payment systems based on the discrete logarithm problem. *PhD thesis, ETH Zürich*, 1998.
8. J. Camenisch and M. Michels. A group signature scheme with improved efficiency. *Advances in Cryptology - Asiacrypt '98, Lecture Notes in Computer Science 1514*, pages 160–174, 1998.
9. J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. *Advances in Cryptology - Crypto '97, Lecture Notes in Computer Science 1294*, pages 410–424, 1997.
10. D. Chaum and E. van Heyst. Group signatures. *Advances in Cryptology - Eurocrypt '91, Lecture Notes in Computer Science 547*, pages 257–265, 1991.
11. L. Chen and T. P. Pedersen. New group signature schemes. *Advances in Cryptology - Eurocrypt '94, Lecture Notes in Computer Science 950*, pages 171–181, 1995.
12. P. Fouque and D. Pointcheval. Threshold Cryptosystems Secure against Chosen-Ciphertext Attacks. *Asiacrypt 2001, Lecture Notes in Computer Science 2248*, 2001.
13. L. Nguyen and R. Safavi-Naini. Efficient and Provably Secure Trapdoor-free Groupo Siganture Schemes from Bilinear Pairings. *Advances in Cryptology - Asiacrypt 04, Lecture Notes in Computer Science 3329*, pages 372 – 386, 2004.