A Bilinear Spontaneous Anonymous Threshold Signature for Ad Hoc Groups

Victor K. Wei

Department of Information Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong
kwwei@ie.cuhk.edu.hk

Abstract. We present an adaptive chosen-plaintext cryptanalysis of Boneh, et al.'s bilinear spontaneous anonymous ad hoc group signature. Then we present a patch, and an extension to a threshold version complete with a security proof in the random oracle model (ROM).

1 Introduction

In an (n, t) threshold signature, t or more members can jointly generate a valid signature, whereas $t-1$ or fewer members cannot. There is almost always a *group secret*. The signature verification depends on some proof that the group of insiders (actual signers) jointly possess knowledge of that secret. Before the signature generation and signature verification protocols, there is almost always a setup stage where the group secret is generated (and then secret-shared) :

- centrally by a group mamanger or other kinds of TTP (trusted third party);
- distributively by joint actions of all n group members;
- hybrid.

The PKI support is usually assumed, and individual member's PKI key pair is often used to secure communication lines during the setup stage.

Recently, a new class of group cryptographic protocols, *spontaneous anonymous group (SAG) cryptography*, gained wide interests [5, 7, 1, 4]. One of the most prominent characteristics of these group protocols is the absence of any group secret. The verification depends on proof of certain partial knowledge of the PKI secret keys of all n group members. There is also no setup stage. Any individual entity with a published public key can spontaneously conscript anoth $n-1$ published public key to form a group, and generate a publicly verifiable 1-out-of-n signature. Furthermore, the signature is anonymous (signer-indistinguishable). The privacy protection is also just about the maxmimum one can imagine: the anonymity is unconditional (information-theoretic), irrevocable, and exculpable. Even if the secret keys of all n members and all communications transacripts are subpoemaed, anonymity remains.

Remark: Theoretically speaking, a group secret in the form of a binary relation still exsist protocols. But it exhibits many distinguishing characteristics different from traditional relations.

SAG cryptographic protocols are perfectly suited to applications in ad hoc groups. In an ad hoc groups, there is a frequent flow of new members joining and old members leaving a group. Complicated group secret setup protocols cannot be afforded. There are also many other reasons to use SAG cryptography in ad hoc groups [4, 3].

Boneh and Franklin [2] introduced bilinear maps to cryptography. It generates many intriguing theoretical research topics concerning gap Diffie-Hellman questions and co-Diffie-Hellman questions, bilinear maps in elliptic curve, Weil pairing and Tate paring, and so on. It also has many practical applications in IBE (identity-based encryption), short signatures, aggregate signatures, SAG signatures, and so on.

Our contributions

1. A simple adaptive chosen-plantext cryptanalysis of Boneh, et al.'s bilinear SAG (spontaneous anonymous group) signature.
2. A new bilinear threshold SAG signature which is existentially unforgeable by adaptive chosen-plaintext adversary in the random oracle model, provided the co-GDH Problem is hard.

Paper organization. The remainder of the paper is organized as follows: In Section 2, we review related results from SAG (spontaneous anonymous group) cryptography and from bilinear maps. In Section 3, we present the cryptanalysis. In Section 4, we present the threshold signature. Concluding remarks in Section 5

2 Related Results

We review related results from SAG (spontaneous anonymous group) cryptography for ad hoc groups and from bilinear maps.

2.1 SAG (Spontaneous Anonymous Group) cryptography

In the *spontaneity paradigm* for group cryptography, there is no (explicit) group secret. That is, there is no group secret that can be simply specified as a number, or a straightforward bit string. Consequently, there is no setup stage to generate the group secret, and there is no setup stage to generate and distribute shares of group secret by inter-member communications. In essentially all spontaneous group cryptographic results, the insiders are indistinghuishable by information-theoretic security or by a hard problem.

However, there is still a binary relation which can be viewed as the "group secret" of spontaneous anonymous group (SAG) cryptography.

Consider the binary relation with respect to DL public key cryptography: $\mathcal{R} \subset Z_q \times Z_p$ where $(x, v) \in \mathcal{R}$ if and only if (x, v) is a key pair. To prove the security of a group cryptography protocol with a group secret, one often reduces it to the problem of extracting/producing a witness x given a group public key v. There is typically only one possible witness for a given public value.

An (n, t) threhold SAG cryptosystem based on n public keys v_1, $\cdots$, v_n correspond to the following binary relation $\mathcal{R} \subset Z_q^t \times Z_p^n$, where

$$(\mathbf{x}, \mathbf{v}) \in \mathcal{R} \text{ iff } \{g^{x_1}, \cdots, g^{x_t}\} = \{v_i : i \in I\} \text{ for some } I \subset \{1, \cdots, n\}, |I| = t.$$

where $\mathbf{x} = (x_1, \cdots, x_t)$ and $\mathbf{v} = (v_1, \cdots, v_n)$.

To prove the security of an SAG protocol, one should reduce it to the extraction of a witness $\mathbf{x}$ satisfying $(\mathbf{x}, \mathbf{v}) \in \mathcal{R}$. There are $\binom{n}{t}$ possible witnesses for a given vector public key $\mathbf{v}$.

2.2 Bilinear maps and Diffie-Hellman issues

We adopt notations from [3]. G_1 (resp. G_2) is a multiplicative cyclic groups of prime order p with generator g_1 (resp. g_2). $\psi : G_2 \to G_1$ with $\psi(g_2) = g_1$ is a computable isomorphism, and $e : G_1 \times G_2 \to G_T$ is computable bilinear map described in [3]. In a bilinear map, we have $e(u^a, v^b) = e(u, v)^{ab}$, and $e(g_1, g_2) \neq 1$.

Computational co-Diffie-Hellman (co-CDH) Problem. Given $g_2, g_2^2 \in G_2$ and $h \in G_1$ compute $h^a \in G_1$.

See [2, 3] for further discussions.

3 Cryptanalysis

We review Boneh, et al.'s bilinear SAG signature for ad hoc groups, present a simple cryptanalysis, and discuss patches.

3.1 Review The Protocol

We first review Boneh, et al.'s scheme.

Key Generation For each user i, $1 \leq i \leq n$, pick random $x_i \in_R Z_p$ and compute $v_i = g_2^{x_i}$. All n public keys v_i's are assumed to be distinct.

Ring Signing Given public keys v_1, $\cdots$, $v_n \in G_2$, a message $M \in \{0, 1\}^*$, and a private key x_s, $1 \leq s \leq n$, pick random $a_i \in_R Z_p$ for all $i \neq s$. Compute $h = H(M) \in G_1$ and set

$$\sigma_s = (h/\psi(\prod_{i \neq s} v_1^{a_i}))^{1/x_s}.$$

For all $i \neq s$ let $\sigma_i = g_1^{a_i}$. Output the ring signature $\sigma = (\sigma_1, \cdots, \sigma_n) \in G_1^n$.

Ring Verification. Given public keys v_1, $\cdots$, $v_n \in G_2$, a message $M \in \{0, 1\}^*$, and a ring signature σ, compute $h = H(M)$ and verify that $e(h, g_2) = \prod_{i=1}^n e(\sigma_i, v_i)$.

3.2 The Cryptanalyses

The adaptive chosen-plaintext attack

The forger $\mathcal{F}$ can generate an anonymous ad hoc group signature on any given message $M \in \{0,1\}^*$ for any set of public keys $v_1, \cdots, v_n$ as follows

1. Twice, query the signing oracle with M and $v_1, \cdots, v_n$. Obtain two signatures $\sigma = (\sigma_1, \cdots, \sigma_n)$ and $\sigma' = (\sigma'_1, \cdots, \sigma'_n)$.
2. Generate a forged signature on M and public keys $v_1, \cdots, v_n$: $\sigma'' = (\sigma''_1, \cdots, \sigma''_n)$ where each $\sigma''_i = (\sigma_i \sigma'_i)^{1/2} \in G_1$.

It is easy to show the forged signature is valid. The square root in G_1 is usually computable in polynomial time, for most bilinear realizations in the literature. The signing oracle presented in Boneh, et al., upon two queries, generate two distinct signatures with overwhelming probability. Most signing oracles in the literature generate two distinct signatures upon queries with repeated message M.

By computing $\sigma''_i = (\sigma_i)^{a/(a+b)}(\sigma'_i)^{b/(a+b)}$ for integers a and b, additional forged signatures can be generated.

3.3 A patch.

The protocol can be patched secure by replacing $H(M)$ with $H(M, L, \rho)$, where L is the list of public keys, and ρ is is a *tag* randomly generated by the signer. In a sense, this "commits" the signature to L and ρ in addition to M. Then the patched the scheme can be proven secure in ROM.

Subsequently, we will extend the patch to a threshold version, and present a security proof. That proof of the threshold version implies a proof of the current 1-out-of-n version.

4 Bilinear SAG Threshold Signature

We patch Boneh, et al.'s bilinear anonymous ad hoc group signature, and then extend it to a threshold version. Furthermore, we prove the security of the threshold bilinear anonymous ad hoc group signature in the random oracle model.

4.1 The threshold extension

Let H_j, $0 \leq j < t$, be suitably chosen hash functions. We follow the notations in [3], where G_1, G_2, g_1, g_2, $e(\cdot, \cdot)$ be bilinear parameters. Let R be a suitable range of tag values.

We describe our security model, protocols, and proofs in that order.

4.2 Security model

Entities: Dealer $\mathcal{D}$, simulator $\mathcal{S}$, forger $\mathcal{F}$.

The Game:

1. $\mathcal{D}$ generates an instantiation of a hard problem, gives to $\mathcal{S}$ to solve. The hard problem is: given g_1^{ab} and g_2^a, compute g_1^b.
2. $\mathcal{S}$ constructs a list of public keys L and a threshold value t, requests $\mathcal{F}$ to return a message-signature-tag triple (M, σ, ρ).
3. $\mathcal{F}$ delivers, after being allowed to query signing oracle $\mathcal{SO}$ and the random oracle H which are both simulated by $\mathcal{S}$.
4. $\mathcal{S}$ uses $\mathcal{F}$'s answer to solve the hard problem.

Further details:

The signing oracle model. We allow $\mathcal{SO}$ to be queried with any message and any list of public keys in any list size, and for any (legitimate) threshold value. In particular, we allow list-size-changing attackers described in Boneh, et al.'s Section 5.4. ncluding list ize increase and list size decrease.

The static adversary model Our proof uses the static adversary model: When $\mathcal{S}$ requests $\mathcal{F}$ to existentially generate a threshold-of-t message-signature pair on the list of public keys $L = (v_1, \cdots, v_n)$, it also gives $\mathcal{F}$ the secret keys corresponding to the first $t-1$ public keys v_1, $\cdots$, v_{t-1}. Let R be a suitable domain of tag values.

Definition 1. *The $(n+1)$-tuple $\sigma = (\sigma_1, \cdots, \sigma_n, \rho) \in G_1^n \times R$ is an (n,t)-signature on the message M and the list of public keys $L = (v_1, \cdots, v_n) \in G_2^n$ if the following t equalities all hold:*

$$e(H_j(M, L, t, \rho), g_2) = \prod_{i=1}^{n} e(\sigma_i, v_i^{i^j}),$$

for each j, $0 \leq j < t$.

Theorem 1. *There exists a PPT bilinear (n,t)-signature generation algorithm, which is existentially unforgeable by adaptive chosen-plaintext adversaries in the random oracle model, provided the co-CDH Problem is hard.*

4.3 Proof

We prove our Theorem by rewinding once, and back patch t places in the second fork. To accomplish a proof with in ROM, we proceed as follows:

1. Exhibit a pair of signature generation algorithm (SIGN) and a signature verification algorithm (VERIFY). Prove its soundness and other ancillary properties.
2. Show $\mathcal{S}$ can generate a valid signature, given a list L of n public keys, a set of t secret keys corresponding to t public keys in L, and a message M.

3. (*Signing Oracle ($\mathcal{SO}$)*) Show $\mathcal{S}$ can generate a valid signature, once given a message M, a list L consisting of n' public keys, the threshold t'; and $\mathcal{S}$ can do so without any secret key, but by simulating (controlling) H.
4. *Witness Extraction ($\mathcal{W}.\mathcal{E}.$)* Show $\mathcal{S}$ can embed the hard problem instantiation into a signature request to $\mathcal{F}$, and by simulating H and $\mathcal{SO}$ and processing the answers returned by $\mathcal{F}$, solve the hard problem instantiation.
5. Specify the entire algorithm $\mathcal{S}$. Show its timing, success probability, and complexity are all satisfactory.

VERIFY: Given a signature $\sigma = (\sigma_1, \cdots, \sigma_n, \rho) \in G_1^n \times R$ on message M and the list of public keys $L = (v_1, \cdots, v_n)$, verify the following t equalities all hold:

$$e(H_j(M, L, t, \rho), g_2) = \prod_{i=1}^{n} e(\sigma_i, v_i^{i^j}),$$

for each j, $0 \leq j < t$.

SIGN: Let $I \subset \{1, \cdots, n\}$, $|I| = t$. Given public keys $v_i = g_2^{x_i}$, $1 \leq i \leq n$, message M, hashing functions H_j, $0 \leq j < t$, and private keys x_i, $i \in I$, generate signature $\sigma = (\sigma_1, \cdots, \sigma_n, \rho)$ as follows:

1. For each $i \in \{1, \cdots, n\} \setminus I$, randomly pick u_i, compute $\sigma_i = g_1^{u_i}$. Randomly pick $\rho \in R$.
2. Solve for $\{\sigma_s^{x_s} : s \in I\}$ in the following system

$$H_j(M, L, t, \rho) = (\prod_{s \in I} \sigma_s^{x_s s^j})(\prod_{i \notin I} \psi(v_i)^{u_i i^j}), \text{ for } 0 \leq j < t.$$

Solution is alwasy feasible because, by properties of the Vandermonde matrix, there exists matrices $\mathbf{A}^{(I)}$ and $\mathbf{B}^{(I)}$ such that

$$\sigma_s^{x_s} = (\prod_{0 \leq j < t} H_j(M, L, t, \rho)^{A_{s,j}^{(I)}})(\prod_{i \notin I} \psi(v_i^{u_i})^{B_{s,i}^{(I)}})$$

for each $s \in I$.

Soundness: Straightforward.

Other properties: Our threshold signature is *synchronized* in the sense that all individual insiders (signers) use the same value ρ and the same values u_i, $i \notin I$. It is *otherwise non-interactive* in the sense that once ρ and u_i, $i \notin I$, are synchronized, each individual insider can generate its share of the threshold signature by itself without any interaction with other users and send its share to a *combiner* who can then compute the threshold signature. The insider share is $(s, \sigma_s^{x_s})$, for insider $s \in I$.. If robustness is an issue, insider s can be required to send a proof-of-correctness by attaching a proof-of-knowledge of x_s along with its signature share which can then be verified against its public key v_s by the combiner. In this extended abstract, we omit discussions on proof-of-correctness or robustness.

Simulating S.O. Given threshold value t', message M' and a list of public keys $L' = (v'_1, \cdots, v'_{n'})$, $\mathcal{SO}$ generates an (n', t')-signature for M' and L' as follows: Randomly picks u_i, $1 \leq i \leq n$. Randomly picks $\rho' \in R$. Back patch to

$$H_j(M', L', t', \rho') = \psi(\prod_{i=1}^{n}(v'_i)^{u_i i^j}),$$

for $0 \leq j < t'$. Return the signature $(g_1^{u_1}, \cdots, g_1^{u_n}, \rho')$.

$\mathcal{D}$ asks $\mathcal{S}$ to solve a hard problem. $\mathcal{D}$ randomly generates a and b, computes $Z = g_1^{ab}$ and $Y = g_2^a$. Gives Z and Y to $\mathcal{S}$ and asks for $Z^{1/a}$ back.

$\mathcal{S}$ constructs a question to ask $\mathcal{F}$: For each i, $1 \leq i \leq n$, randomly picks x_i. randomly picks $s \in_R \{t, \cdots, n\}$. Ask $\mathcal{F}$ to generate a message-(n, t)-signature pair on the public keys $L = (g_2^{x_1}, \cdots, g_2^{x_{t-1}}, Y^{x_t}, \cdots, Y^{x_n}\})$.

W.E. The witness extraction is via back patching t critical queries, one for each $H_j(\cdots)$, $0 \leq j < t$, in the second fork run, and then solve for a Vandermonde system of t linear equations.

Among all H_j-queries combined, $\mathcal{S}$ randomly picks the ℓ-th query $H_{j_\ell}(M_\ell, L_\ell, t_\ell, \rho_\ell)$ and rewind. Let X_{j_ℓ} denote the hash query outcome during the first simulation run, $X_{j_\ell} = H_{j_\ell}(M_\ell, L_\ell, t_\ell, \rho_\ell)$. In the rewind, $\mathcal{S}$ back patches to

$$H_{j_\ell}(M_\ell, L_\ell, t_\ell, \rho_\ell) \leftarrow XZ.$$

and $\mathcal{S}$ back patches to, if possible,

$$H_j(M_\ell, L_\ell, t_\ell, \rho_\ell) \leftarrow X_j g_1^{ar_j},$$

for $j \neq j_\ell$, $0 \leq j < t$, where r_j is randomly picked by $\mathcal{S}$ during the second run, $X_j = H_j(M_\ell, L_\ell, t_\ell, \rho_\ell)$ during the first run. These $t-1$ back patches are possible if their queries have not been made before the ℓ-th query.

Let $\sigma = (\sigma_1, \cdots, \sigma_n)$ and $\sigma' = (\sigma'_1, \cdots, \sigma'_n)$ denote the two signatures returned by $\mathcal{F}$. on messages M and M' respectively, if $\mathcal{F}$ were successful in both runs. If $(M, L, t) = (M', L, t) = (M_\ell, L_\ell, t_\ell)$, then we have, for $j = j_\ell$,

$$\begin{aligned}
e(X, g_2) &= \prod_{i=1}^{n} e(\sigma_i, v_i)^{i^j} \\
&= \prod_{s=1}^{t-1} e(\sigma_s, g_2^{x_s})^{s^j} \prod_{i=t}^{n} e(\sigma_i, Y^{x_i})^{i^j} \\
&= \prod_{s=1}^{t-1} e(\sigma_s, g_2^{x_s})^{s^j} \prod_{i=t}^{n} e(\sigma_i^{x_i}, Y)^{i^j} \\
&= \prod_{s=1}^{t-1} e((\sigma_s)^{x_s/a}, Y)^{s^j} \prod_{i=t}^{n} e(\sigma_i^{x_i}, Y)^{i^j}
\end{aligned}$$

and

$$e(XZ, g_2) = \prod_{s=1}^{t-1} e((\sigma'_s)^{x_s/a}, Y)^{s^j} \prod_{i=t}^{n} e((\sigma'_i)^{x_i}, Y)^{i^j}$$

and therefore

$$e(Z, g_2) = e(Z^{1/a}, Y) = \prod_{s=1}^{t-1} e((\frac{\sigma'_s}{\sigma_s})^{x_s/a}, Y)^{s^j} \prod_{i=t}^{n} e((\frac{\sigma'_i}{\sigma_i})^{x_i}, Y)^{i^j}$$

$$Z^{1/a} = \prod_{s=1}^{t-1} (\frac{\sigma'_s}{\sigma_s})^{x_s s^j/a} \prod_{i=t}^{n} (\frac{\sigma'_i}{\sigma_i})^{x_i i^j} \quad (1)$$

And we have, for $j \neq j_\ell$, $0 \leq j < t$,

$$g_1^{r_j} = T_j \prod_{s=1}^{t-1} (\frac{\sigma'_s}{\sigma_s})^{x_s s^j/a}$$

where

$$T_j = \prod_{i=t}^{n} (\frac{\sigma'_i}{\sigma_i})^{x_i i^j}$$

for $0 \leq j \leq t$.

We have t unknowns: $Z^{1/a}$ and $(\sigma'_i/\sigma_i)^{1/a}$, $1 \leq i < t$, in a Vandermonde system of t linear equations. By properties of the Vandermonde matrix, there exists a matrix $\mathbf{C}$ such that

$$(\sigma'_s/\sigma_s)^{1/a} = \prod_{j \neq j_\ell, 0 \leq j < t} (g_1^{r_j}/T_j)^{C_{s,j}}$$

for $1 \leq s \leq t-1$. The right-hand side of each equation above can be computed. Therefore the right-hand side of Equation (1) can be computed.

The above witness extraction works provided the $t-1$ "critical" queries $H_j(M_\ell)$, $j \neq j_\ell$, $0 \leq j < t$, have not been made by $\mathcal{F}$ prior to the forking point in a direct H_j query or indirectly via an $\mathcal{SO}$ query.

The sequencing, complexity and probability of $\mathcal{S}$. We use the classification proof technique. We assume $\mathcal{F}$ is an (T, ϵ) forger, i.e. its average running time is T and its probability of success is ϵ. Assume $\mathcal{F}$ makes a combined total of q_H queries to all of H_j, $0 \leq j < t$, and it makes q_S queries to $\mathcal{SO}$ with public key list L' not longer than $n_{\max}$. The total number of H queries is no more than $q_T := q_H + n_{\max} q_S$.

$\mathcal{S}$ constructs the signature request to $\mathcal{F}$ as described above. For each ℓ, $1 \leq q_T$, $\mathcal{S}$ performs the Witness Extraction described above, rewinding to the ℓ-th H-query, and back patching t queries, one for each H_j, if possible.

Let p_ℓ denote the probability that $\mathcal{F}$ succeeds and the ℓ-th query is the first among all queries whose input (M, L, t, ρ) corresponds to $\mathcal{F}$'s answer. Note that by the lunchtime attack arguments, the probability of $\mathcal{F}$ not having made these t queries before delivery the signature to $\mathcal{S}$ is negligible. There exists ℓ, $1 \leq \ell \leq q_T$, such that $p_\ell \geq \epsilon/q_T$.

By the RoS lemma [6], $\mathcal{F}$'s probability of successfully producing a signature during the second fork is p_ℓ, in the ℓ-th round of $\mathcal{S}$. Therefore, the probability of $\mathcal{S}$'s successful witness extraction is essentially $\sum_\ell P_\ell^2 \geq (\epsilon/q_T)^2$.

Summarizing, $\mathcal{S}$ successfully solves the hard problem on behalf of $\mathcal{D}$ in time $2(q_H + n_{\max} q_S)T$ with probability $\epsilon/(q_H + n_{\max} q_S)^2$. □

Remark: Our proof uses one rewind and t back patches on the second fork. We also obtained a proof using t back patches but no rewinding by directly extending the proof in Boneh, et al.'s [3]. That proof has better simulation efficiency, but is wordier to describe.

5 Discussions

We cryptanalyzed the bilinear anonymous ad hoc group signature of Boneh, et al. Then we modified it and extended it to a threshold scheme, and provide security proof under the random oracle model. Finally we break our proven-secure scheme when a certain kind of hashing function is used, namely the (partially) first-component homomorphic hashing function.

Acknowledgements. Helpful discussions with Duncan S. Wong and Joseph K. Liu are acknowledged.

References

1. M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n signatures from a variety of keys. In *Proc. ASIACRYPT 2002*, pages 415–432. Springer-Verlag, 2002. Lecture Notes in Computer Science No. 2501.
2. D. Boneh and M. Franklin. Identity-based encryption from the Weil paring. *SIAM J. Computing*, 32(3):586–615, 2003.
3. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Proc. EUROCRYPT 2003*, pages 416–432. Springer-Verlag, 2003. Lecture Notes in Computer Science No. 2656.
4. E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In *Proc. CRYPTO 2002*, pages 465–480. Springer-Verlag, 2002. Lecture Notes in Computer Science No. 2442.
5. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Proc. CRYPTO 94*, pages 174–187. Springer-Verlag, 1994. Lecture Notes in Computer Science No. 839.
6. J. K. Liu, V. K. Wei, and D. S. Wong. Linkable and culpable ring signatures. *ePrint*, 2004(027).
7. R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *Proc. ASIACRYPT 2001*, pages 552–565. Springer-Verlag, 2001. Lecture Notes in Computer Science No. 2248.