Security Arguments for Partial Delegation with Warrant Proxy Signature Schemes *

Qin Wang, Zhenfu Cao†

Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200030, P. R. China

E-mail address: chhwangqin@sjtu.edu.cn, cao-zf@cs.sjtu.edu.cn

November 17, 2004

Abstract

Proxy signature is an important cryptographic primitive and has been suggested in numerous applications. In this paper, we present an attack on the aggregate-signature-based proxy signature schemes, then point out there are two flaws in BPW notion of security for proxy signature. Furthermore, we give arguments for partial delegation with warrant proxy signature schemes. We construct a new proxy signature scheme and prove that it is secure against existentially forgery on adaptively chosen-message attacks and adaptively chosen-warrant attacks under the random oracle model.

Keywords: digital signature, proxy signature, partial delegation with warrant, provable security.

1 Introduction

The idea of proxy signature was first discussed in [12]. The scheme allows an entity, called the original signer, to delegate his/her signing capability to another entity, called the proxy signer, in a way that the latter can sign messages on behalf of the former. Proxy signature schemes have been suggested to use in a number of applications, particularly in distributed computing where delegation of rights is quite common.

Mambo et al. [12] classified proxy signatures based on delegation type as full delegation, partial delegation and delegation by warrant. For most of real-world settings, full delegation is obviously impractical and insecure. Partial delegation is further classified as proxy-unprotected and proxy-protected according to protection of proxy signer, and proxy -protected signature schemes can provide more security level than proxy-unprotected signature schemes. In this paper we only discuss proxy-protected signature schemes. Compared with delegation by warrant, partial delegation brings faster processing speed, however, in

*This research is partially supported by the National Natural Science Foundation of China for Distinguished Young Scholars under Grant No. 60225007, the National Research Fund for the Doctoral Program of Higher Education of China under Grant No. 20020248024, and the Science and Technology Research Project of Shanghai under Grant Nos. 04JC14055 and 046407067.

†corresponding author, phone:+86-021-62835602

such schemes the range of messages a proxy signer can sign is not limited. In 1997, Kim et al. [8] introduced the notion of partial delegation with warrant which combines the benefits of the partial delegation and the delegation by warrant. After that, most work on proxy signature focuses on partial delegation with warrant schemes.

The security requirements for proxy signature are first specified in [12, 13], and later are enhanced by [10, 11]. That is, a secure proxy signature scheme should satisfy the following five requirements: verifiability, strong unforgeability, strong identifiability, strong undeniability, prevention of misuse. But these requirements are informal and cannot give a precise meaning of security for proxy signature schemes. After the first scheme was constructed by Mambo et al. [12], a number of new schemes have been proposed. However, most of them do not fully meet the desired security requirements. Since those schemes lack provably-security guarantee, almost every other paper breaks some previously proposed proxy signature scheme and proposes a new scheme [8, 9, 14, 17, 15, 19].

In 2003, Boldyreva et al. [4] formalized a notion of security for proxy signature schemes, and that was the first work on proxy signature in the provable-security direction. We refer to their security model BPW security model. They also proposed three schemes and proved their security: the delegation-by-certificate schemes, the aggregate-signature-based schemes and the triple schnorr scheme.

In 2004, Zuowen Tan et al. [18] presented an attack to the delegation-by-certificate schemes. They showed the schemes suffering attacks mounted by a malicious original signer. They also formalized a notion of security for delegation-by-certificate proxy signature schemes.

Compared with the delegation-by-certificate proxy signature schemes, the aggregate-signature-based proxy signature schemes have possibility to obtain an improvement in terms of both bandwidth and efficiency. In fact, we find the aggregate-signature-based schemes in [4] are not secure too. Furthermore, we point out there are two flaws in BPW security model. Because most of the research work on proxy signature focuses on partial delegation with warrant which is believed to combine the benefits of security and efficiency, we offer security arguments for it in this paper. Moreover, we construct a proxy scheme and prove its security in the random oracle model assuming the Computation Diffie-Hellman problem in gap Diffie-Hellman groups is hard to solve.

The rest of this paper is organized as follows: In section 2, we introduce related mathematical problems, recall the components of a digital signature scheme and its security definition. In section 3, we analyze the aggregate-signature-based proxy signature schemes. In section 4, we define a notion of security for partial delegation with warrant proxy signature schemes. Then in section 5, we construct a proxy signature scheme and prove its security in our new model. Finally, concluding remarks are made in Section 6.

2 Preliminaries

Let G_1 be a cyclic additive group generated by P, whose order is a large prime q, and G_2 be a cyclic multiplicative group of the same order q. A bilinear pairing is a map $e : G_1 \times G_1 \to G_2$ with the following properties:

- Bilinear: $e(aP, bQ) = e(P, Q)^{ab}$.
- Non-degenerate: there exists $P, Q \in G_1$, such that $e(P, Q) \neq 1$.

- Computable: there is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.

Now we describe some mathematical problems in G_1.

Discrete Logarithm Problem (DLP). Given two group elements P and Q, find an integer n, such that $Q = nP$ whenever such an integer exists.

Decision Diffie-Hellman Problem (DDHP). For a,b,$c \in Z_q^*$, given P, aP, bP, cP, decide whether $c = ab \bmod q$.

Computational Diffie-Hellman Problem (CDHP). For a,$b \in Z_q^*$, given P, aP, bP, compute abP.

When the DDHP is easy but the CDHP is hard on the group G_1, we call G_1 a Gap Diffil-Hellman (GDH) group. Such groups can be found on supersingular elliptic curves or hyperelliptic curves over finite field, and the bilinear parings can be derived from the Weil or Tate pairing. We can refer to [1, 3, 5, 7] for more details.

Now, we give a precise definition of digital signature schemes and secure digital signature schemes. These definitions are based on [6]:

Definition 2.1 [Digital Signature Scheme] A digital signature scheme $DS = (G, S, V)$ is defined by the following:

- on input 1^k, where k is the security parameter, the algorithm G produces a pair (pk, sk) of matching public and secret keys. Algorithm G is probabilistic.
- Giving a message m and a pair of matching public and secret keys (pk, sk), S can produce a signature σ. The signing algorithm might be probabilistic.
- Given a signature σ, a message m and a public key pk, V tests whether σ is a valid signature of m with respect to pk. In general, the verification algorithm need not be probabilistic.

Definition 2.2 [Secure Digital Signature Scheme] A digital signature scheme is secure if an existential forgery is computationally impossible, even under adaptively chosen-message attacks.

3 Analysis of Aggregate-signature-based Proxy Signature Schemes

In this section, we will analyze the aggregate-signature-based proxy signature schemes proposed in [4] and present an attack on them. After careful observation, we find there is no fault in the proof, so we'll try to discuss the flaws in their security model.

3.1 Aggregate-signature-based Proxy Signature Schemes

Let $AS = (G, S, V, A, AV)$be an aggregate signature scheme, where $DS = (G, S, V)$ is an ordinary digital signature scheme, A is the aggregation algorithm, and AV is the aggregate verification algorithm. The algorithms of the corresponding proxy signature scheme $PS[AS] = (G_1, S_1, V_1, (D, P), PS, PV, ID)$ are defined as follows:

- The key generation algorithms are the same as the aggregate scheme: $K_1 = K$.
- An ordinary signature for message M is obtained by prepending 11 to the message, and signing the result using S, i.e., $S_1(sk, M) = S(sk, 11||M)$.

- The verification of a signature σ for message M is done by computing $V_1(pk, M, \sigma) = V(sk, 11||M, \sigma)$.
- (D, P) is a pair of interactive randomized algorithms forming the proxy designation protocol. In this protocol, user i is a designator, and user j is a proxy signer. The result of the interaction is $(\omega, cert)$ where ω is the designator's warrant and $cert$ is a signature for $00||pk_j||\omega$ under sk_i.
- The proxy signing algorithm PS uses A to aggregate the certificate and a proxy signature as follows:

$$a\sigma = A(pk_i, pk_j, 00||pk_j||\omega, 01||pk_i||M, cert, S(sk_j, 01||pk_i||M)).$$

- The proxy verification algorithm PV is defined by

$$PV(pk, M, (\omega, pk', a\sigma)) = AV(pk, pk', 00||pk'||\omega, 01||pk||M, a\sigma).$$

- The identification algorithm is defined by $ID(\omega, pk', a\sigma) = pk'$.

In [4], the authors argued if AS is a secure aggregate signature scheme, then the scheme $PS[AS]$ is a secure proxy signature scheme.

3.2 Attack to Aggregate-signature-based Proxy Signature Schemes

Let user i be an original signer and user j a proxy signer. We use BGSL bilinear aggregate signature scheme [2] as AS. BGSL works in GDH groups, and its security has been proved in the random oracle model assuming hardness of the Computational Diffie-Hellman assumption. A GDH group G_1 and its generator P, a hash function $H : \{0,1\}^* \to G_1$ and the bilinear map $e : G_1 \times G_2 \to G_T$ are system parameters. We refer the reader to [2] for details. The public/private key pairs of user i and user j are $(pk_i, x_i)$and (pk_j, x_j), i.e., $Pk_i = x_iP$, $Pk_j = x_jP$. We show an attack as follows:

step 1: User i gives a pair of (ω, σ_i) to user j under x_i, that is

$$\sigma_i = x_iH(00||pk_j||\omega).$$

step 2: User j checks whether $e(\sigma_i, P) = e(H(00||pk_j||\omega), pk_i)$. If it is, user j computes

$$\sigma_j = x_jH(01||pk_i||M),$$

then computes

$$a\sigma = \sigma_i + \sigma_j.$$

$(\omega, M, a\sigma)$ is user j's proxy signature on user i's delegation warrant ω.

step 3: After getting the above proxy signature, user i can compute user j's signature on M,

$$\sigma_j = a\sigma - \sigma_i.$$

step 4: With user j's signature on M, user i can generate any other warrant ω' and its signature σ_i' with his private key,

$$\sigma_i' = x_iH(00||pk_j||\omega'),$$

then use the aggregate algorithm to compute

$$a\sigma' = \sigma'_i + \sigma_j.$$

$(\omega', M, a\sigma')$ is a valid forged proxy signature, the original signer is user i and the proxy signer is user j. A third party can verify the validity of the proxy signature as follows:

$$e(a\sigma', P) = e(H(00||pk_j||\omega'), pk_i)e(H(01||pk_i||M), pk_j).$$

Like the attack to delegation-by-certificate schemes in [18]. Here the attack is also mounted by the malicious original signer. In many cases, the forgery will impair the proxy signer because what written in the warrant may relate to the proxy signer's interest or responsibility. For example, perhaps the warrant is a contract between the original signer and the proxy signer, using the above attack, the original signer can change the contract (such as the price or time period, etc.) to impair the proxy signer.

With a slight modification, the aggregate-signature-based schemes can resist the above attack. The simplest way is to add the original signer's warrant together with the signed message in step 2:

$$\sigma_j = x_j H(01||pk_i||M||\omega).$$

3.3 Two Flaws in BPW Security Model

There are two flaws in the notion of security for proxy signature schemes [4].

1. In the BPW model, if an honest user 1 *never* delegates user i (we allow user i to be colluded by an adversary) as a proxy signer, but eventually an adversary can forge a proxy signature by user i on behalf of user 1, the proxy scheme is broken. We find besides the above case, if user 1 have delegated user i as a proxy signer with some warrants which are put into a list $Warr_1$, but an adversary can forge a proxy signature by user i on behalf of user 1 with a new warrant ω' ($\omega' \notin Warr_1$), the scheme is also broken. Because in this case, user 1 never delegate user i with warrant ω', and the forgery perhaps will impair the original signer user 1.

2. In the BPW model, user i (we allow user i to be colluded by an adversary) can delegate an honest user 1 with n different warrants which are put into a list $Warr_i$, then an adversary can query user 1's proxy signature on any message $m_j (j \in N)$ conforming to the restriction in warrant $\omega_h (\omega_h \in Warr_i)$ and the query (ω_h, m_j) are added in a query list Que_i. If the adversary can forge a proxy signature by user 1 on behalf of user i on a message m' which *never* appears in Que_i, the scheme is broken. We find besides the above case, if in the end an adversary can forge a proxy signature on (ω', m') where m' has appeared in Que_i, but (ω', m') is not in the Que_i list, the scheme is also broken. Because in this case, user 1 never sign m' under ω', and the forgery perhaps will impair the proxy signer user 1.

4 Partial Delegation with Warrant Proxy Signature Schemes

4.1 Syntax of Partial Delegation with Warrant Proxy Signature Schemes

In a partial delegation with warrant proxy signature scheme, the original signer uses its standard signature algorithm to sign a warrant which includes the type of the information delegated, both the parties' identities and the period of delegation, etc. The signature of

the warrant is called certificate. Here we only consider proxy-protected schemes. With the certificate and his private key, the proxy signer generates a proxy private key. After that, the proxy signer can sign any messages according to the warrant. And a third party would verify the validity of the proxy signature.

Definition 4.1 [Partial Delegation with Warrant Proxy Signature Scheme] A partial delegation with warrant proxy signature scheme is a tuple $PS = (G, S, V, P, PS, PV)$, and the components are defined as follows:

- G is used to produce private/public key pairs as usual. The public and private key pairs for the original and the proxy signers are (pk_o, sk_o), (pk_p, sk_p).
- S is a randomized standard signing algorithm. The original signer uses it to sign a warrant ω which includes the type of the information delegated, both of the parties' identities and the period of delegation, etc. Then the original signer produces $(\omega, cert)$ and gives it to the proxy signer without a secure channel.
- V is a deterministic standard verification algorithm. It takes input $(\omega, cert)$, then outputs 0 or 1. In the latter case, we say that $cert$ is a valid signature for ω relative to pk_o.
- P is a (possibly) randomized algorithm, taking $cert$ and his private key as input, the proxy signer generates a proxy signing key skp.
- PS is a (possibly) randomized proxy signing algorithm. It takes input a proxy signing key skp and a message $m \in \{0,1\}^*$, then outputs a proxy signature $p\sigma$.
- PV is a deterministic proxy verification algorithm. It takes input $(pk_o, pk_p, \omega, m, p\sigma)$ (here for clarity, we write out the public keys of original and proxy signers which are included in ω), then outputs 0 or 1. In the latter case, we say that $p\sigma$ is a valid proxy signature for m by the proxy signer on behalf of the original signer.

4.2 A Notion of Security for Partial Delegation with Warrant Proxy Signature Schemes

We first informally describe some of the features of our adversarial model. In real world applications, sometimes the users want to change their public keys, an adversary is also possible to collude some original signers or proxy signers, and in most cases, there is no secure channel between an original signer and a proxy signer. An adversary aims at personating the original signer or the proxy signer and forging a standard signature, or forging a proxy signature. In our model, such as in [4, 18], there is only one honest user, user 1, against an adversary A. A can register public keys but he must output both of the public keys and the matching secret keys or prove his knowledge of the secret keys, such that a nasty problem of collusion is considered in our model. User 1 can delegate others (they are controlled by A) or be delegated by others as the proxy signer, such that in our model both the original and the proxy signers' security is considered. We also allow self-delegation in our model. And we don't assume the existence of a secure channel between an original signer and a proxy signer. Then a secure proxy signature should satisfy three requirements: 1. A can't forge a stand signature relative to user 1's public key; 2. A can't forge a proxy

signature by user 1; 3. A can't forge a proxy signature on behalf of user 1. We formalize this intuition as adaptively chosen-message attack and adaptively chosen-warrant attack.

Definition 4.2 [Secure Partial Delegation with Warrant Proxy Signature Scheme] A partial delegation with warrant proxy signature scheme is secure if an existential forgery is computationally impossible, even under adaptively chosen-message attacks and adaptively chosen-warrant attacks.

Our notion of security for partial delegation with warrant is formally defined as follows and the adversary's advantage is his probability of success in the following game:

1. First, user 1 runs G on input 1^k to produce public and secret key pair (pk_1, sk_1).

2. An adversary A is given a single public key pk_1, and he can access to two signing oracles, that is $O_S(sk_1, \cdot)$, and $O_{PS}(skp, \cdot)$.

3. A can choose all public keys except the challenged one pk_1, i.e., A can register all other key pairs in the game, (pk_2, sk_2),...,(pk_n, sk_n).

4. A can choose r warrants $\omega_i (i \in \{1, 2, ..., r\})$ in which user 1 is delegated as the proxy signer and user j relative to pk_j is the original signer, then signs ω_i under sk_j $(j \in \{2, ..., n\})$ and (ω_i, σ_i) is a signature. User 1 accepts the signature after verifying the validity of it, then runs the algorithm P to get the corresponding skp. We allow A to sign several different warrants with the same private key, i.e., an original signer can delegate the same proxy signer several times with different warrants. In the end, all the warrants are added to a list $Warrp$.

A can query standard signature oracle O_S and proxy signature oracle O_{PS}.

5. Signature queries. A can make q_S queries to the signing oracle $O_S(sk_1, \cdot)$ on any messages of his choice:

(1). Some messages are warrants in which user 1 is stated as an original signer and user j $(j \in \{2, ..., n\})$ is a proxy signer. These messages are added to a list $Warro$;

(2). Some messages are warrants in which user 1 is stated as both an original signer and a proxy signer (this is user 1's self-delegation). Then user 1 will run P to get the corresponding skp. These messages are added to a list $Warrs$;

(3). Others are ordinary messages which are added to a list Squ.

6. Proxy signature queries. A can query the proxy signature oracle $O_{PS}(skp, \cdot)$ on any message of his choice q_{PS} times. The queries are like (ω_i, m_i) where $\omega_i \in \{Warrp \cup Warrs\}$, $i \in \{1, 2, ..., q_{PS}\}$ and m_i satisfies ω_i. Then the queries are added to a list $PSqu$. User 1 will produce a proxy signature under the corresponding skp to A.

The adversary A wins if any one of the following events occurs:

$E1$: A outputs a forgery (m', σ) where $V(pk_1, m', \sigma) = 1$ and m' was not queried to $O_S(sk_1, \cdot)$ $(m' \notin \{Warro \cup Warrs \cup Squ\})$, i.e., A forges user 1's stand signature.

$E2$: A outputs a forgery $(\omega', m', p\sigma)$ where $PV(pk_i, pk_1, \omega', m', p\sigma) = 1$ $(i \in \{1, 2, ..., n\})$ and $(\omega', m') \notin PSqu$, i.e., A forges user 1's proxy signature, including self-delegation.

$E3$: A outputs a forgery $(\omega', m', p\sigma)$ where $PV(pk_1, pk_i, \omega', m', p\sigma) = 1$ $(i \in \{2, ..., n\})$ and $\omega' \notin Warro$, i.e., A forges a proxy signature on behalf of user 1.

5 A Secure Partial Delegation with Warrant Proxy Signature Scheme

In this section, we propose a new proxy signature scheme which is provably secure in the random oracle model assuming the Computational Diffie-Hellman problem is hard in

gap Diffie-Hellman groups.

5.1 The Proposed Proxy Signature Scheme

In the proposed proxy signature, we use BLS short signature scheme [3] as a standard signature algorithm, Cha-Cheon ID-based signature scheme [5] as a proxy signature algorithm. They were both proved secure in the random oracle model. The proposed scheme can be regarded as the Cha-Cheon version proxy signature scheme discussed in [20]. We propose it here because it just can be proved secure in our new security model. The proposed proxy signature scheme $PS = (G, S, V, P, PS, PV)$ is as follows:

Use a GDH parameter generator [5] to provide a GDH group G with a prime order q. P is a generator of G. Define two cryptographic hash functions $H1 : \{0,1\}^* \times G \rightarrow Z_q^*$, $H2 : \{0,1\}^* \rightarrow G$.

G**:** Pick a random number $x \in Z_q^*$, compute $v = xP$, then the secret key is x, the public key is v.

S**:** Given a private key $x \in Z_q^*$, and a message $m \in \{0,1\}^*$, compute $f \leftarrow H_2(m) \in G$ and $\sigma \leftarrow xf$. The signature is $\sigma \in G$;

V**:** Given a public key v, a message/signature pair (m, σ), compute $f \leftarrow H_2(m) \in G$ and verify that (P, v, f, σ) is a valid Diffie-Hellman tuple. If so, output 1; if not, output 0.

Assuming the original signer has public/secret key pair (v_o, x_o), the proxy signer has public/secret key pair (v_p, x_p).

P**:** The original signer computes $\sigma_\omega = x_o H_2(\omega)$, and sends (ω, σ_ω) to the proxy signer. The proxy signer checks if $(P, v_o, H_2(\omega), \sigma_\omega)$ is a valid Diffie-Hellman tuple. If so, then computes $skp_\omega = \sigma_\omega + x_p H_2(\omega) = (x_o + x_p)H_2(\omega)$.

PS**:** On message $m \in \{0,1\}^*$, the proxy signer picks a random number $r \in Z_q^*$ and outputs a signature $p\sigma = (U, h, V)$ where $U = rH_2(\omega)$, $h = H_1(\omega||m, U)$ and $V = (r + h)skp_\omega$.

To verify a proxy signature $p\sigma = (U, h, V)$ of a message m for a warrant ω, check whether $(P, v_o + v_p, U + hH_2(\omega), V)$ is a Diffie-Hellman tuple. If so, output 1; if not, output 0.

5.2 Security Proof

Theorem 1: The above proxy signature is existentially unforgeable against adaptively chosen-message attacks and adaptively chosen-warrant attacks in the random oracle model if the Computational Diffie-Hellman Problem in GDH groups is hard to solve.

Proof. Suppose A is a forger algorithm that breaks the proxy signature scheme. We show how to construct an algorithm B against CDHP, such that if A has a non-negligible advantage in creating a forgery for the proxy signature scheme, B will have a non-negligible advantage to solve CDHP. This will contradict the fact that G is a GDH group.

Let P be a generator of G, the only honest user, user 1, runs G to produce a pair of public/secret keys (x_1P, x_1). A can query to four oracles: H_1 oracle at most q_{H_1} times, H_2 oracle at most q_{H_2} times, signature query $O_S(x_1, \cdot)$ at most q_S times , and proxy signature query $O_{PS}(skp, \cdot)$ at most q_{PS} times. Algorithm B simulates user 1 and interacts with A as follows:

1. Algorithm B starts by giving A the generator P, and the public key x_1P.

2. A produce $n-1$ pairs of public/secret keys: $(x_2P, x_2), ..., (x_nP, x_n)$. Now B is given $n+1$ tuples $(P, x_1P, yP), (P, x_1P + x_1P, yP), (P, x_1P + x_2P, yP), ..., (P, x_1P + x_nP, yP)$

$(x_1, ..., x_n, y \in Z_q^*)$, its goal is to output any one of the following: $x_1yP, (x_1 + x_1)yP, (x_1 + x_2)yP, ..., (x_1 + x_n)yP$. It's obvious that the above problems are all CDH problems.

3. H_1-queries. At any time algorithm A can query the random oracle H_1. To respond to these, algorithm B maintains a list of tuples $< m_j, u_j, h_j >$. We refer to this list H_1-list. The list is initially empty. When A queries the oracle at point $(m_i, u_i), (m_i \in \{0,1\}^*, u_i \in G)$, algorithm B responds as follows:

(1). If the query (m_i, u_i) already appears on the H_1-list in a tuple $< m_i, u_i, h_i >$, algorithm B responds with $H_1(m_i, u_i) = h_i$.

(2). Otherwise, B generates a random $h_i \in Z_q^*$ and sends $H_1(m_i, u_i) = h_i$ to A.

4. H_2-queries. At any time algorithm A can query the random oracle H_2. To respond to these, algorithm B maintains a list of tuples $< m_j, \beta_j, b_j, c_j >$ as explained below. We refer to this list H_2-list. The list is initially empty. When A queries the oracle at point $m_i \in \{0,1\}^*$, algorithm B responds as follows:

(1). If the query m_i already appears on the H_2-list in a tuple $< m_i, \beta_i, b_i, c_i >$, algorithm B responds with $H_2(m_i) = \beta_i \in G$.

(2). Otherwise, B generates a random coin $c_i \in \{0,1\}$ so that $Pr[c_i = 0] = 1/(q_S + 1)$.

(3). Algorithm B picks a random $b_i \in Z_q^*$ and computes $\beta_i = (1 - c_i)yP + b_iP \in G$.

(4). Algorithm B adds the tuple $< m_i, \beta_i, b_i, c_i >$ to the H_2-list and responds to A by setting $H(m_i) = \beta_i$.

Note that either way β_i is uniform in G and is independent of A's current view.

5. Signature queries. Let m_i be a signature query issued by A. Algorithm B responds to this query as follows. B maintains three lists $Warro$, $Warrs$ and Squ which initially are all empty.

(1). Algorithm B runs the above algorithm for responding to H_2-queries to obtain $\beta_i \in G$ such that $H(m_i) = \beta_i$. Let $< m_i, \beta_i, b_i, c_i >$ be the corresponding tuple on the H_2-list. If $c_i = 0$ then B reports failure and terminates.

(2). Otherwise, we know $c_i = 1$ and hence $\beta_i = b_iP \in G$. Define $\sigma_i = b_i \cdot x_1P$. And therefore σ_i is a valid signature on m_i under the public key x_1P. Algorithm B gives σ_i to algorithm A. If m_i is a warrant in which user 1 is stated as an original signer and user j $(j \in \{2, ..., n\})$ is a proxy signer, then m_i is added to a list $Warro$-list. If m_i is a self-delegation warrant, then m_i is added to $Warrs$-list. Otherwise, m_i is added to Squ-list.

6. A chooses r warrants $\omega_i (i \in \{1, 2, ..., r\})$ in which user 1 is delegated as the proxy signer and user j relative to pk_j is the original signer, then signs ω_i under sk_j $(j \in \{2, ..., n\})$ and the signature is σ_i. A sends the tuple (ω_i, σ_i) to B. B searches in H_2-list for a tuple containing ω_i to get $H_2(\omega_i)$(with only a negligible probability A doesn't query ω_i to H_2-query). B accepts after verifying the validity of the signature. ω_i is added to a list $Warrp$ which is also maintained by B.

7. Proxy signature queries. Let $(\omega_i, m_i)(\omega_i \in (Warrp \cup Warrs))$ be a proxy signature query issued by A. B responds to this query as follows:

(1). B searches in the H_2-list to get $< \omega_i, \beta_i, b_i, c_i >$.

(2). B chooses randomly $y_i \in Z_q^*$, and $h_i \in Z_q^*$, then compute $U_i = y_iP - h_i\beta_i$, $V_i = y_i(x_oP + x_1P)$ (o is the original signer and $o \in \{1, 2, ...n\}$). B adds $(\omega_i || m_i, U_i, h_i)$to the H_1-list. If A has queried $(\omega_i || m_i, U_i)$ to H_1, B reports failure and terminates. Since U_i is random, the probability is negligible. So $(\omega_i, m_i, U_i, h_i, V_i)$is a valid proxy signature. The queries are added to a list $PSqu$ which is also a list maintained by B.

As long as B does not abort, the simulation for A is perfect. Since A has a non-negligible advantage, at least one of the following events will occur with non-negligible probability.

E1: A outputs a valid signature (m', σ') under the given public key x_1P, such that no signature query was issued for $m'(m' \notin \{Warro \cup Warrs \cup Squ\})$. Then B will find the tuple $< m', \beta', b', c' >$ on the H_2-list. If $c' = 1$, B reports failure and terminates. Otherwise, $c' = 0$, therefore $\sigma' = x_1(yP + b'P)$, then $x_1yP = \sigma' - b'(x_1P)$. Now B computes x_1yP.

E2: A outputs a valid proxy signature $(\omega', m', U', h', V')$ where $\omega' \in \{Warrp \cup Warrs\}$ and $(\omega', m') \notin PSqu$, $V' = (x_o+x_1)(U'+h'H_2(\omega'))(o \in \{1, 2, ..., n\})$. Then algorithm B will find the tuple $< \omega', \beta', b', c' >$ on the H_2-list. If $c' = 1$, B reports failure and terminates. Otherwise $c' = 0$, B applies the oracle replay attack which was invented by Pointcheval and Stern in [16, 17], and obtains a valid proxy signature $(\omega', m', U', h'', V'')$, $V'' = (x_o + x_1)(U' + h''H_2(\omega'))$. B subtracts the two equations $V' - V'' = (h' - h'')(x_o + x_1)H_2(\omega')$, and $(x_o+x_1)(b'+y)P = (h'-h'')^{-1}(V'-V'')$, $(x_o+x_1)yP = (h'-h'')^{-1}(V'-V'')-b'(x_oP+x_1P)$. That is, B computes $(x_o + x_1)yP$ $(o \in \{1, 2, ..., n\})$.

E3: A outputs a valid proxy signature $(\omega', m', U', h', V')$ where $\omega' \notin Warro$, x_1P is the original signer's public key and x_pP $(p \in \{2, ..., n\})$is the delegator's public key. Then B will find the tuple $< \omega', \beta', b', c' >$ on the H_2-list. If $c' = 1$, B reports failure and terminates. Otherwise $c' = 0$, B applies the oracle replay attack, and obtains a valid proxy signature $(\omega', m', U', h'', V'')$, $V'' = (x_p + x_1)(U' + h''H_2(\omega'))$. B subtracts the two equations $V' - V'' = (h' - h'')(x_p + x_1)H_2(\omega')$, and $(x_p+x_1)(b'+y)P = (h'-h'')^{-1}(V'-V'')$, $(x_p + x_1)yP = (h' - h'')^{-1}(V' - V'') - b'(x_pP + x_1P)$. That is, B computes $(x_p + x_1)yP$ $(p \in \{2, ..., n\})$.

From the above analysis, if the proposed scheme isn't a secure partial delegation with warrant proxy signature scheme, the CDHP in G will be solved with a non-negligible probability.

By a word, in [4], Boldyerva et al. proposed a triple schnorr scheme. It's also a partial delegation with warrant proxy signature scheme. It was proven in BPW security model. We find it's still secure in our model, the proof can be done in the similar way except that the two flaws we discuss in section 3 should be considered.

6 Conclusion

In this paper, we analyze the aggregate-based-signature schemes and give a simple improvement. Furthermore, we point out two flaws in BPW security notion for proxy signature. We formalize the syntax of partial delegation with warrant and propose a security model for it. After that, based on BLS short signature and Cha-Cheon ID-based signature we propose a proxy signature scheme which is secure in the random oracle model assuming the CDHP is hard in GDH groups. We find triple schnorr scheme is still secure in our security model.

References

[1] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing, Advances in Cryptology-Crypto 2001, LNCS 2139, pp.213-229, Springer-Verlag, 2001.

[2] D. Boneh, C. Gentry, H. Shacham, and B. Lynn. Aggregate and verifiably encrypted signatures from bilinear maps. In E. Biham, editor, Proceedings of Advances is Cryptology C Eurocrypt03, LNCS. Springer-Verlag, 2003.

[3] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. In C. Boyd, editor, Asiacrypt 01, volume 2248 of LNCS. Springer Verlag, 2001.

[4] A. Boldyreva, A. Palacio, B. Warinschi. Secure Proxy Signature Schemes for Delegation of Signing Rights. At:http://eprint.iacr.org/2003/096.

[5] J.C. Cha and J.H. Cheon. An identity-based signature from gap Diffie-Hellman groups, Public Key Cryptography - PKC 2003, LNCS 2139, pp.18-30, Springer- Verlag, 2003.

[6] S. Goldwasser, S. Micali, and R. Rivest. A Digital Signature Scheme Secure Against Adaptative Chosen- Message Attacks. SIAM Journal of Computing, 17(2):281C308, April 1988.

[7] F. Hess, Efficient identity based signature schemes based on pairings, SAC 2002, LNCS 2595, pp.310-324, Springer-Verlag, 2002.

[8] S. Kim, S. Park, and D. Won. Proxy signatures, revisited. In Y. Han, T. Okamoto, and S. Quing, editors, Proceedings of International Conference on Information and Communications Security (ICICS)'97, volume 1334 of LNCS, pages 223-232. Springer-Verlag, 1997.

[9] J. Lee, J. Cheon, and S. Kim. An analysis of proxy signatures: Is a secure channel necessary? In M. Joye, editor, Topics in Cryptology-CT-RSA'03, volume 2612 of LNCS, pages 68-79. Spinger-Verlag, 2003.

[10] B. Lee, H. Kim, and K. Kim. Strong proxy signature and its applications. In: Proceedings of the 2001 Symposium on Cryptography and Information Security (SCIS'01), Vol. 2/2, pp. 603-608. Oiso, Japan, Jan. 23-26, 2001.

[11] B. Lee, H. Kim, and K. Kim. Secure mobile agent using strong non-designated proxy signature. In: Information Security and Privacy (ACISP'01), LNCS 2119, pp. 474-486. Springer-Verlag, 2001.

[12] M. Mambo, K. Usuda, and E. Okamoto. Proxy signatures for delegating signing operation. In Proceedings of the 3rd ACM Conference on Computer and Communications Security (CCS), pages 48-57. ACM, 1996.

[13] M. Mambo, K. Usuda, and E. Okamoto. Proxy Signature: Delegation of the power to sign messages. IEICE Trans. Fundamentals, Sep. 1996, Vol. E79-A, No. 9, pp.1338-1353.

[14] T. Okamoto, M. Tada, and E. Okamoto. Extended proxy signatures for smart cards. In LNCS, volume 1729 of LNCS. Springer-Verlag, 1999.

[15] H.M. Sun and B.T. Hsieh. Remarks on two nonrepudiable proxy signature schemes. In Ninth National Conference on Information Security, volume 241-246, 1999.

[16] D. Pointcheval and J. Stern. Security proofs for signature schemes, Proc. of Eurocrypt '96, Lecture Notes in Computer Sciences, Vol.1070, pp.387-398, Springer-Verlag, 1996.

[17] D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. Journal of Cryptology, 13(3):361C369, 2000.

[18] Zuowen Tan, Zhuojun Liu. Provably Secure Delegation-by-Certification Proxy Signature Schemes. http://eprint.iacr.org/2004/148/

[19] S.M. Yen, C.P. Hung, and Y.Y. Lee. Remarks on some proxy signature schemes. In Workshop on Cryptology and Information Security, 2000 ICS, pages 54C59, 2000.

[20] Fangguo Zhang and Reihaneh Safavi-Naini and Chih-Yin Lin. New Proxy Signature, Proxy Blind Signature and Proxy Ring Signature Schemes from Bilinear Pairing. http://eprint.iacr.org/2003/104