Designated Verifier Threshold Proxy Signature Scheme without Random Oracles

M.Beheshti-Atashgah[1], M.Bayat[2], M.Gardeshi[3], and M.R.Aref[4]

[1] Research Center of Intelligent Signal Processing, Tehran, Iran
[2] Department of Mathematics & Computer Sciences, Tarbiat Moallem University, Tehran, Iran
[3] Department of Communication & Information Technology, Imam Hossein University, Tehran, Iran
[4] Information Systems and Security Lab(ISSL),EE Department, Sharif University of Technology, Tehran, Iran

Abstract. In a (t, n) designated verifier threshold proxy signature scheme, an original signer can delegate his/her signing power to n proxy signers such that any t or more out of n proxy signers can sign messages on behalf of the original signer but $t-1$ or less of the proxy signers cannot generate a valid proxy signature. Of course, the signature is issued for a designated receiver and therefore only the designated receiver can validate the proxy signature. In this paper, we propose a new designated verifier threshold proxy signature scheme and also show that the proposed scheme has provable security in the standard model. The security of proposed scheme is based on the $GBDH$ assumption and the proposed scheme satisfies all the security requirements of threshold proxy signature schemes.

Keywords: Proxy signature scheme, Threshold proxy signature scheme, Provable security, Standard model, Bilinear pairing.

1 Introduction

A The concept of proxy signature scheme was first introduced by Mambo et al.'s in 1996 [1]. In a proxy signature scheme, an original signer can delegate his/her signing capability to a proxy signer and then the proxy signer can sign messages on behalf of the original signer. So far, many proxy signature schemes have been proposed; such as [2], [3] .

K.Zhang [4] and Kim et al. [5] independently proposed the first threshold proxy signature scheme by using the ideas of proxy signature scheme and secret sharing. In a (t, n) threshold proxy signature scheme, the original signer delegates his/her signing power to a proxy group of n member such that any t or more than t proxy signers can cooperatively sign messages on behalf of the original signer. Until now, many threshold proxy signature schemes have been proposed such as [6], [7]. A (t, n) threshold proxy signature scheme should satisfy the following security requirements [8], [9]:

verifiability, unforgeability, undeniability, identifiability and prevention of misuse.

Although many of proposed schemes were claiming that satisfy the above requirements, but their precise security meaning was unclear. First, Bellare & Rogaway [10] used the concept of provable security in the random oracle model for the key-agreement protocols. Then Boldyreva et al. [11] provide method to prove the security of the proxy signature scheme in the random oracle model and so far, many proxy signature schemes such as [12] and [13] have been proposed that have provable security in this model.

The concept of standard model and provable security in this model was first introduced by Boneh & Boyen [14] for an ID-based encryption scheme without random oracles. Of course, their scheme proposed in a weaker model of security known as Selective-ID model. Then Waters [15] in 2005 proposed an ID-based encryption scheme and proved that his proposed scheme has provable security in the standard model. In fact, the standard model described by Waters is more complete and more efficient from the last model.

In 2006, Huang et al. [16] proposed the first proxy signature scheme in the standard model and following them, other schemes such the Yu et al.'s [17] designated verifier proxy signature scheme were proposed. As far as we know, any threshold proxy signature scheme has been proposed in the standard model till now and so in this paper, we want to propose the first designated verifier threshold proxy signature scheme in the standard model based on the Yu et al.'s scheme. Additionally, we'll show that our scheme is based on the Gap Bilinear Diffie-Hellman (GBDH) intractability assumption without relying on the random oracle and satisfies all the security requirements for a secure threshold proxy signature scheme.

Roadmap. The reminder of this paper is organized as follows. In the next Section, some preliminary concepts are given. In Section 3, the formal model of designated verifier threshold proxy signature scheme is described. In Section 4, we will propose our new scheme. In Section 5, we analyze the proposed scheme and finally, the conclusions are given in Section 6.

2 Preliminaries

In this Section, we review fundamental backgrounds including bilinear pairings and complexity assumptions used in this paper.

2.1 Bilinear pairing

Let G, G_T be two cyclic multiplicative groups of prime order p and g be a generator of G. The map $e : G \times G \to G_T$ is said to be an admissible bilinear pairing if the following conditions hold true:

$$(1)\ \ bilinearity:\ \ For\ all\ a, b \in \mathbb{Z}_p,\ \ e\left(g^a,\ g^b\right) = e(g,g)^{ab}$$
$$(2)\ \ non-degeneracy:\ \ e\left(P,P\right) \neq l_{G_T}$$

(3) $computability : There\ is\ an\ efficient\ algorithm\ to\ compute\ e\left(g,h\right) for\ all$ $g,h \in G$.

2.2 Complexity assumption

Definition 1. *(CBDH problem). Given* $\left(g, g^a, g^b, g^c \in G\right)$ *for some unknown* $a, b, c \in \mathbb{Z}_p$*, compute* $e\left(g,g\right)^{abc}$.

Definition 2. *(DBDH problem). Given* $\left(g, g^a, g^b, g^c \in G\right)$ *for some unknown* $a, b, c \in \mathbb{Z}_p$ *and* $w \in G_T$*, decide whether* $w = e\left(g,g\right)^{abc}$.

Definition 3. *(GBDH problem). Given* $\left(g, g^a, g^b, g^c \in G\right)$ *for some unknown* $a, b, c \in \mathbb{Z}_p$*, compute* $w = e\left(g,g\right)^{abc}$ *with the help of DBDH oracle* $\mathcal{O}_{DBDH}$.

The probability that an adversary $\mathcal{A}$ can solve the $GBDH$ problem is defined as:

$$Succ_{\mathcal{A}}^{GBDH} = \Pr\left[e\left(g,g\right)^{abc} \leftarrow \mathcal{A}\left(G, G_T, g,\ g^a, g^b, g^c, \mathcal{O}_{DBDH}\right)\right] \quad (1)$$

3 Formal models of DVTPSS

3.1 Outline of DVTPSS

In a designated verifier threshold proxy signature scheme, there exist three participants namely Alice (original signer), proxy signers group $A = \{P_1, P_2, \cdots, P_n\}$ and Cindy (designated verifier). A DVTPSS[1] consists of the following algorithms.

1. **Setup**: Given a security parameter k, this algorithm outputs the system parameters.
2. **KeyGen**: This algorithm takes as input the security parameter k and outputs the secret/public key pair (sk_i, pk_i) for $i \in \{a, \{1, 2, \cdots, n\}, c\}$ denotes Alice, proxy signers group and Cindy.
3. **DelegationGen**: This algorithm takes as input the warrant ω and the original signer's private key, then outputs the delegation $\sigma_{\omega_{P_i}}$, $i = 1, 2, \cdots, n$ for proxy signers.
4. **DelegationVerify**: After receiving $(\omega, R_{a_i}, \sigma_{\omega_{P_i}})$, each proxy signer in the group A confirms its validity. Note that R_{a_i} is a public value that is computed and published by the original signer.
5. **ProxySignGen**: this algorithm takes as input the proxy signers' private key sk_i $(i = 1, 2, \cdots, t)$, the delegation shares $\sigma_{\omega_{P_i}}$ $(i = 1, 2, \cdots, t)$, the designated verifier's public key pk_c and a message M to produce a proxy signature σ_P.
6. **ProxySignVerify**: A deterministic algorithm that accepts a message M, the warrant ω , the proxy signature σ_P, the original signer and the proxy signers' public key $\left(pk_a, pk_{P_i}\right)$, $i = 1, 2, \cdots, t$, the designated verifier's private key sk_c and returns ω if the signature is valid, otherwise returns $\perp$ indicating the proxy signature is invalid.

[1] Designated Verifier Threshold Proxy Signature Scheme

7. **Transcript simulation**: This algorithm takes as input a message M, the warrant ω and the designated verifier's private key sk_c to generate an identically distributed transcript σ^* that is indistinguishable from the original DVTPS σ.

3.2 Security notions

There are four types adversary in the system as follows.
Type I: adversary $\mathcal{A}_{\text{I}}$ only has the public keys of the original signer and proxy signers.
Type II: adversary $\mathcal{A}_{\text{II}}$ has the public keys of the original signer and proxy signers, he/she additionally has the secret key of the original signer Alice.
Type III: adversary $\mathcal{A}_{\text{III}}$ has the public keys of the original signer and proxy signers, he/she additionally has the secret key of one of the proxy signers (like Bob).
Type IV: adversary $\mathcal{A}_{\text{IV}}$ has the public keys of the original signer and proxy signers, he/she additionally corrupts $t-1$ proxy signers.
Note that if DVTPSS is unforgeable against type II and III adversary, it is also unforgeable against type I adversary. On the other hand, type IV adversary is more powerful than type III adversary. Hence, if the scheme is secure against type IV adversary, then it is secure against type III.

Unforgeability against $\mathcal{A}_{\text{II}}$

The existential unforgeability of a DVTPS under $\mathcal{A}_{\text{II}}$ adversary requires that it is difficult for the original signer to generate a valid threshold proxy signature of a message M^* under the warrant ω^* that has not been signed by the proxy signers group. It is defined using the following game between the challenger $\mathcal{C}$ and $\mathcal{A}_{\text{II}}$ adversary:

1. **Setup**: The challenger $\mathcal{C}$ runs the Setup algorithm to obtain system parameters, and runs KeyGen algorithm to obtain the secret/public key pairs (sk_a, pk_a), (sk_i, pk_i) $i = 1, \cdots, n$, (sk_c, pk_c) of the original signer Alice, proxy signers and the designated verifier Cindy. Then $\mathcal{C}$ sends sk_a, pk_a, pk_i, pk_c (where i is one of the proxy signers) to the adversary $\mathcal{A}_{\text{II}}$.
2. **ProxySign queries**: The adversary $\mathcal{A}_{\text{II}}$ can request a proxy signature on the message M under the warrant ω. $\mathcal{C}$ runs the ProxySign algorithm to obtain the proxy signature σ_P and then sends it to $\mathcal{A}_{\text{II}}$.
3. **Verify queries**: The adversary $\mathcal{A}_{\text{II}}$ can request a proxy signature verification on a (M, ω, σ_P). If σ_P is a valid DVTPS, $\mathcal{C}$ outputs $\top$ and $\perp$ otherwise.
4. **Output**: Finally, $\mathcal{A}_{\text{II}}$ outputs a new DVTPS σ_P^* on the message M^* under the warrant ω^*, such that
   (a) (M^*, ω^*) has never been queried during the ProxySign queries.
   (b) σ_P^* is a valid DVTPS of message M^* under warrant ω^*.

The advantage of $\mathcal{A}_{\text{II}}$ in the above game is defined as $\text{Adv}_{\mathcal{A}_{\text{II}}}$= $\Pr[\mathcal{A}_{\text{II}}$ succeeds$]$.

Definition 4. *An adversary $\mathcal{A}_{\text{II}}$ is said to be an $(\epsilon, t, q_{PS}, q_v)$ −forger of a DVTPS if $\mathcal{A}_{\text{II}}$ in the above game: has advantage of at least ϵ, runs in time at most t, makes at most q_{PS} ProxySign queries and q_v Verify queries.*

Unforgeability against $\mathcal{A}_{\text{IV}}$

Similar to the last game, the following game is defined between the challenger $\mathcal{C}$ and $\mathcal{A}_{\text{IV}}$ adversary, but note that in this game, the adversary $\mathcal{A}_{\text{IV}}$ corrupts $t-1$ proxy signers:

1. **Setup**: The challenger $\mathcal{C}$ runs the Setup algorithm to obtain system parameters, and runs KeyGen algorithm to obtain the secret/public key pairs (sk_a, pk_a), (sk_i, pk_i) $i = 1, \cdots, n$, (sk_c, pk_c) of the original signer Alice, proxy signers and the designated verifier Cindy. Then $\mathcal{C}$ sends sk_i, pk_a, pk_i, pk_c to the adversary $\mathcal{A}_{\text{IV}}$ where $i = 1, 2, \cdots, t$. Note that the challenger $\mathcal{C}$ should guess right the proxy signers corrupted by adversary $\mathcal{A}_{\text{IV}}$, otherwise the game is failed and $\mathcal{C}$ will stop the simulation.
2. **Delegation queries**: $\mathcal{A}_{\text{IV}}$ can request $t-1$ proxy shares of the proxy signers (like Bob) under the warrant ω. $\mathcal{C}$ runs the DelegationGen algorithm to obtain the proxy shares $\sigma_{\omega_{P_i}}$, $i = 1, \cdots, t-1$ and then returns it to $\mathcal{A}_{\text{IV}}$.
3. **ProxySign queries**: The adversary $\mathcal{A}_{\text{IV}}$ can request individual proxy signatures (issued by corrupted proxy signers) and final proxy signature on the message M under the warrant ω. $\mathcal{C}$ runs the ProxySign algorithm to obtain the proxy signature σ_P and then sends it to $\mathcal{A}_{\text{IV}}$.
4. **Verify queries**: The adversary $\mathcal{A}_{\text{IV}}$ can request a proxy signature verification on a (M, ω, σ_P). If σ_P is a valid DVTPS, $\mathcal{C}$ outputs $\top$ and $\perp$ otherwise.
5. **Output**: Finally, $\mathcal{A}_{\text{IV}}$ outputs a new DVTPS σ_P^* on the message M^* under the warrant ω^*, such that
   (a) ω^* has never been queried during the Delegation queries.
   (b) (M^*, ω^*) has never been queried during the ProxySign queries.
   (c) σ_P^* is a valid DVTPS of message M^* under warrant ω^*.

The advantage of $\mathcal{A}_{\text{IV}}$ in the above game is defined as $\text{Adv}_{\mathcal{A}_{\text{IV}}} = \Pr[\mathcal{A}_{\text{IV}} \text{ succeeds}]$.

Definition 5. *An adversary $\mathcal{A}_{\text{IV}}$ is said to be an $(\epsilon, t, q_\omega, q_{PS}, q_v)$ −forger of a DVTPS if $\mathcal{A}_{\text{IV}}$ in the above game: has advantage of at least ϵ, runs in time at most t, makes at most q_ω Delegation queries, q_{PS} ProxySign queries and q_v Verify queries.*

3.3 Security requirements

Verifiability: The designated verifier should be convinced of the original signer's agreement on the signed message.
Identifiability: Anyone should determine the identities of the corresponding proxy signers from a proxy signature.
Undeniability: The proxy signers group should not be able to create the signature against anyone. This security requirement is also called "non-repudiation".
Prevention of misuse: A proxy signing key should not be used for purpose other than generating valid proxy signature. In case of misuse, the responsibility of the proxy signature should be determined explicitly.

4 Proposed DVTPSS in the standard model

In this Section, we describe the proposed DVPSS. As we assumed earlier, there are three participants in the system: original signer Alice, proxy signers group $A = \{P_1, P_2, \cdots, P_n\}$ and designated verifier Cindy. In continue, all the messages to be signed will be showed as bit string of length n.

It is possible to be quest that if the bit length of input messages is more than n, what we can do? Thus for more flexibility of the scheme, we can use a collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^n$ in the first and last of the proposed scheme.

Our scheme includes the following algorithms:

1. **Setup**: Let (G, G_T) is bilinear groups from prime order p. e denotes an admissible pairing and $g \in G$ is the generator of G. $u', m' \in G$ are two random integers and $u = (u_j)$, $m = (m_j)$ are vectors of length n that is choosen at random from group G. The system parameters are

$$\sigma = (G, G_T, p, e, g, u', m', u, m).$$

2. **KeyGen**: Alice sets her secret key $sk_a = (x_a, y_a) \in \mathbb{Z}_p^2$ and computes her corresponding public key $pk_a = (g^{x_a}, g^{y_a})$. Similarly, each proxy signer $P_i \in A$, $i = 1, 2, \cdots, n$ sets his/her secret-public keys $sk_i = (x_i, y_i) \in \mathbb{Z}_p^2$, $pk_i = (g^{x_i}, g^{y_i})$. The secret-public keys of the designated verifier (Cindy) are $sk_c = (x_c, y_c) \in \mathbb{Z}_p^2$, $pk_c = (g^{x_c}, g^{y_c})$.
3. **DelegationGen**: Let ω_j be the j-th bit of ω that ω is the warrant issued by the original signer and $\mathcal{W} \subseteq \{1, 2, \cdots, n\}$ be the set of all j for which $\omega_j = 1$. The original signer Alice randomly chooses $r_{a_i} \in_R \mathbb{Z}_p$, $i = 1, \cdots, n$ and then computes the proxy share of proxy signer P_i as follows. Alice also publishes the value R_{a_i} and $\sigma_{\omega_{P_i}{}_1}$.

$$\sigma_{\omega_{P_i}} = e\left(\sigma_{\omega_{P_i}{}_1}, \sigma_{\omega_{P_i}{}_2}\right) = e\left(g^{x_a y_a}\left(u' \prod_{j \in \mathcal{W}} u_j\right)^{r_{a_i}},\ pk_{ix}\right), \tag{2}$$
$$R_{a_i} = g^{r_{a_i}}$$

   Then, Alice sends $(\omega, R_{a_i}, \sigma_{\omega_{P_i}})$ to each proxy signer like P_i where $i = 1, 2, \cdots, n$.
4. **DelegationVerify**: To validate the correctness of $(\omega, R_{a_i}, \sigma_{\omega_{P_i}})$, each proxy signer P_i checks whether the following equation is satisfy?

$$\sigma_{\omega_{P_i}} = e\left(g^{x_a},\ g^{y_a}\right)^{x_i} \cdot e\left(u' \prod_{j \in \mathcal{W}} u_j,\ R_{a_i}\right)^{x_i} \tag{3}$$

Correctness

$$\begin{aligned}
\sigma_{\omega_{P_i}} &= e\left(g^{x_a y_a},\ g^{x_i}\right) \cdot e\left(\left(u' \prod_{j \in \mathcal{W}} u_j\right)^{r_{a_i}},\ g^{x_i}\right) \\
&= e\left(g^{x_a y_a},\ g^{x_i}\right) \cdot e\left(u' \prod_{j \in \mathcal{W}} u_j,\ \left(g^{x_i}\right)^{r_{a_i}}\right) \\
&= e\left(g^{x_a},\ g^{y_a}\right)^{x_i} \cdot e\left(u' \prod_{j \in \mathcal{W}} u_j,\ g^{r_{a_i}}\right)^{x_i} \\
&= e\left(pk_{ax},\ pk_{ay}\right)^{x_i} \cdot e\left(u' \prod_{j \in \mathcal{W}} u_j,\ R_{a_i}\right)^{x_i}
\end{aligned}$$

5. **ProxySignGen**: Let M be a n-bit message and M_j be the j-th bit of M.Assume that $\mathcal{M} \subseteq \{1, 2, \cdots, n\}$ be the set of all j for which $M_j = 1$.
   (a) Issuing individual signature: Each proxy signer P_i picks $r'_{a_i}, r_{b_i} \in_R \mathbb{Z}_p$ at random and computes and publishes $R'_{a_i} = g^{r'_{a_i}}$, $R_{b_i} = g^{r_{b_i}}$. Finally, each proxy signer P_i produces his/her individual signature as follows.

$$\sigma_{i_1} = \left(\sigma_{\omega_{P_i 1}} \left(u' \prod_{j \in \mathcal{W}} u_j\right)^{r'_{a_i}} g^{x_i y_i} \left(m' \prod_{j \in \mathcal{M}} m_j\right)^{r_{b_i}}\right), \tag{4}$$

$$\sigma_{i_2} = R_{a_i} g^{r'_{a_i}} = R_{a_i} R'_{a_i}, \sigma_{i_3} = g^{r_{b_i}} = R_{b_i}$$

Then, each P_i sends his/her partial proxy signature $\sigma_i = (\sigma_{i_1}, \sigma_{i_2}, \sigma_{i_3})$ to a clerk (The clerk is one of the proxy signers). The clerk validates individual signatures by checking whether the equality holds.

$$\begin{aligned}
e\left(\sigma_{i_1}, g\right) = e\left(pk_{ax},\ pk_{ay}\right)\ e\left(u' \prod_{j \in \mathcal{W}} u_j,\ R_{a_i} R'_{a_i}\right) \\
e\left(pk_{ix},\ pk_{iy}\right) e\left(m' \prod_{j \in \mathcal{M}} m_j,\ R_{b_i}\right)
\end{aligned} \tag{5}$$

Correctness

$$\begin{aligned}
e\left(\sigma_{i_1}, g\right) = \\
&= e\left(g^{x_a y_a} \left(u' \prod_{j \in \mathcal{W}} u_j\right)^{r_{a_i} + r'_{a_i}} g^{x_i y_i} \left(m' \prod_{j \in \mathcal{M}} m_j\right)^{r_{b_i}},\ g\right) \\
&= e\left(g^{x_a y_a}, g\right) \cdot e\left(u' \prod_{j \in \mathcal{W}} u_j, g^{r_{a_i} + r'_{a_i}}\right) \cdot e\left(g^{x_i y_i}, g\right) \cdot \\
&\quad e\left(m' \prod_{j \in \mathcal{M}} m_j, g^{b_i}\right) \\
&= e\left(pk_{ax}, pk_{ay}\right) \cdot e\left(u' \prod_{j \in \mathcal{W}} u_j, R_{a_i} R'_{a_i}\right) \cdot e\left(pk_{ix}, pk_{iy}\right) \cdot \\
&\quad e\left(m' \prod_{j \in \mathcal{M}} m_j, R_{b_i}\right)
\end{aligned}$$

(b) Issuing proxy signature: If all the individual signatures are valid, the clerk calculates final proxy signature as follows.

$$\begin{gathered}\sigma_{P_1} = \prod_{i=1}^{t} e\left(\sigma_{i_1}, pk_{cx}\right),\ \sigma_{P_2} = \sum_{i=1}^{t} R_{a_i} R'_{a_i} = R_a, \\ \sigma_{P_3} = \sum_{i=1}^{t} R_{b_i} = R_b\end{gathered} \tag{6}$$

Finally, $\sigma_P = (\sigma_{P_1}, \sigma_{P2}\ , \sigma_{P3})$ is the proxy signature and is send to the designated verifier Cindy.

6. **ProxySignVerify**: The designated verifier can check the validity of the proxy signature through the equation.

$$\begin{aligned}\sigma_{P_1} = {} & e\left(pk_{ax}, pk_{ay}\right)^{x_c} e\left(pk_{txy}, g\right)^{x_c} \cdot \\ & e\left(u' \prod_{j \in \mathcal{W}} u_j, \sigma_{P2}\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, \sigma_{P3}\right)^{x_c}\end{aligned} \tag{7}$$

Correctness

$$\begin{aligned}\sigma_{P_1} &= \prod_{i=1}^{t} e\left(\sigma_{i_1}, pk_{cx}\right) \\ &= \prod_{i=1}^{t} e\left(g^{x_a y_a}, g^{x_c}\right) e\left(\left(u' \prod_{j \in \mathcal{W}} u_j\right)^{r_{a_i} + r'_{a_j}}, g^{x_c}\right) \\ &\quad e\left(g^{x_i y_i}, g^{x_c}\right) e\left(\left(m' \prod_{j \in \mathcal{M}} m_j\right)^{r_{b_i}}, g^{x_c}\right) \\ &= e\left(pk_{ax}, pk_{ay}\right)^{x_c} \prod_{i=1}^{t} e\left(u' \prod_{j \in \mathcal{W}} u_j, R_{a_i} R'_{a_i}\right)^{x_c} \\ &\quad e\left(pk_{ixy}, g\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, R_{b_i}\right)^{x_c} \\ &= e\left(pk_{ax}, pk_{ay}\right)^{x_c} e\left(\sum_{i=1}^{t} pk_{ixy}, g\right)^{x_c} \\ &\quad e\left(u' \textstyle\prod_{j \in \mathcal{W}} u_j, \displaystyle\sum_{i=1}^{t} R_{a_i} R'_{a_i}\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, \sum_{i=1}^{t} R_{b_i}\right)^{x_c} \\ &= e\left(pk_{ax}, pk_{ay}\right)^{x_c} e\left(pk_{txy}, g\right)^{x_c} \\ &\quad e\left(u' \prod_{j \in \mathcal{W}} u_j, \sum_{i=1}^{t} R_a\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, \sum_{i=1}^{t} R_b\right)^{x_c} \sigma_{P_1} \\ &= e\left(pk_{ax}, pk_{ay}\right)^{x_c} e\left(pk_{txy}, g\right)^{x_c} \\ &\quad e\left(u' \prod_{j \in \mathcal{W}} u_j, \sigma_{P2}\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, \sigma_{P3}\right)^{x_c}\end{aligned}$$

Note that in the above equalities, we used from the following bilinear pairing property.

$$\prod_{i=1}^{t} e\left(P_i, Q\right) = e\left(\prod_{i=1}^{t} P_i, Q\right) \tag{8}$$

Additionally, in the above equalities, the group public key is $pk_{txy} = \prod_{i=1}^{t} g^{x_i y_i}$.

7. **Transcript simulation**: Cindy can use her private key to compute a proxy signature on a message M^* under the warrant ω^*. She denotes two random integers $r_1, r_2 \in \mathbb{Z}_p^*$ and computes $\sigma_P^* = \left(\sigma_{P_1}^*, \sigma_{P_2}^*, \sigma_{P_3}^*\right)$ where $\sigma_{P_2}^* = g^{r_1}$, $\sigma_{P_3}^* = g^{r_2}$.

$$\begin{aligned} \sigma_{P_1}^* = {} & e\left(pk_{ax}, pk_{ay}\right)^{x_c} e\left(pk_{txy}, g\right)^{x_c} \\ & e\left(u' \prod_{j \in \mathcal{W}} u_j, \sigma_{P_2}^*\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, \sigma_{P_3}^*\right)^{x_c} \end{aligned} \tag{9}$$

5 Analysis of the scheme

5.1 Unforgeability

Unforgeability against adversary $\mathcal{A}_{\text{II}}$

Theorem 1. *If there exists an adversary $\mathcal{A}_{\text{II}}$ who can $(\epsilon, t, q_{PS}, q_v)$ breaks our scheme, then there exists another algorithm $\mathcal{B}$ who can use $\mathcal{A}_{\text{II}}$ to solve an instance of the GBDH problem with probability*

$$Succ_{\mathcal{B}}^{GBDH} \geq \frac{\epsilon}{8(n+1)q_{PS}}$$

In time

$$t' \leq ((2n+6)\, q_{PS} + nq_v + 2)\, t_1 + (5q_v)\, t_2 \ + (3n + 12q_{PS} + q_v + 5)\, T_1 + \\ (3q_v)\, T_2 + (q_{PS} + 4q_v)\, t_e$$

where t_1, t_2 are the time for a multiplication in G and G_T respectively, T_1, T_2 are the time of an exponentiation in G and G_T respectively, and t_e is the time for a pairing computation in (G, G_T).

Proof. For the proof of this theorem, please refer to [17]. □

Unforgeability against adversary $\mathcal{A}_{\text{IV}}$

Theorem 2. *If there exists an adversary $\mathcal{A}_{\text{IV}}$ who can $(\epsilon, t, q_\omega, q_{PS}, q_v)$ breaks our scheme, then there exists another algorithm $\mathcal{B}$ who can use $\mathcal{A}_{\text{IV}}$ to solve an instance of the GBDH problem with probability*

$$Succ_{\mathcal{B}}^{GBDH} \geq \frac{(t-1)!\ (n-t+1)!\epsilon}{3(n+1)^3 n!(3(q_\omega+q_{ps}))^{q_v+2}}$$

In time

$$t' \leq ((2n+6)\, q_{PS} + (n+4)\, q_v + (n+4)\, q_\omega)\, t_1 + (3q_v)\, t_2 \ + \\ (8q_\omega + 12q_{PS} + 4q_v)\, T_1 + (4q_v)\, T_2 + (q_{PS} + 6q_v)\, t_e$$

where t_1, t_2 are the time for a multiplication in G and G_T respectively, T_1, T_2 are the time of an exponentiation in G and G_T respectively, and t_e is the time for a pairing computation in (G, G_T).

Proof. Assume that $\mathcal{B}$ receives a $GBDH$ problem instance $\left(g, g^a, g^b, g^c\right)$ of a bilinear group (G, G_T) whose orders are both a prime number p. His/Her goal is to output $e\,(g,g)^{abc}$ with the help of the $DBDH$ oracle $\mathcal{O}_{DBDH}$. $\mathcal{B}$ runs $\mathcal{A}_{\text{IV}}$ as a subroutine and act as $\mathcal{A}_{\text{IV}}$'s challenger. $\mathcal{B}$ will answer $\mathcal{A}_{\text{IV}}$'s queries as follows:
Setup: $\mathcal{B}$ chooses two random integers l_a, l_b and other two random integers k_a, k_b uniformly between 0 and n. Then, $\mathcal{B}$ picks two values x'_a, x'_b at random and two random n-vectors $\overrightarrow{x}_a = (x_{ai})$, $\overrightarrow{x}_b = (x_{bi})$ where $x'_a, x_{ai} \in \mathbb{Z}_{l_a}$, $x'_b, x_{bi} \in \mathbb{Z}_{l_b}$.

Additionally, $\mathcal{B}$ chooses two values y'_a, y'_b at random and two random n-vectors $y_a = (y_{ai})$, $\overrightarrow{y}_b = (y_{bi})$ where $y'_a, y'_b, y_{ai}, y_{bi} \in \mathbb{Z}_p$. All of these values are kept secret by $\mathcal{B}$.
For a message M and a warrant ω, we let $\mathcal{M} \subset \{1, 2, \cdots, n\}$ and $\mathcal{W} \subset \{1, 2, \cdots, n\}$ be the set of all i for which $M_i = 1$ and $\omega_i = 1$. For simplicity of analysis, we defines functions $F_a(\omega)$, $J_a(\omega)$, $K_a(\omega)$, $F_b(M)$, $J_b(M)$ and $K_b(M)$ as in [15].

$$\begin{aligned}
&(1)\ F_a(\omega) = (p - l_a k_a) + x'_a + \sum_{i \in \mathcal{W}} x_{ai},\ J_a(\omega) = y'_a + \sum_{i \in \mathcal{W}} y_{ai}, \\
&\quad K_a(\omega) = \begin{cases} 0, & if\ x'_a + \sum_{i \in \mathcal{W}} x_{ai} = 0\ \ (mod\ l_a) \\ 1, & Otherwise. \end{cases} \\
&(2)\ F_b(M) = (p - l_b k_b) + x'_b + \sum_{i \in \mathcal{M}} x_{bi},\ J_b(M) = y'_b + \sum_{i \in \mathcal{W}} y_{bi}, \\
&\quad K_b(M) = \begin{cases} 0, & if\ x'_b + \sum_{i \in \mathcal{M}} x_{bi} = 0\ \ (mod\ l_b) \\ 1, & Otherwise. \end{cases}
\end{aligned} \tag{10}$$

In the next step, $\mathcal{B}$ generates the follow common parameters:

(1) $\mathcal{B}$ assigns the public keys of the original signer and the designated verifier, respectively as $\left(pk_{ax}, pk_{ay}\right) = \left(g^a, g^b\right)$, $pk_{cx} = g^c$ where g^a, g^b, g^c are the input of the $GBDH$ problem.
(2) $\mathcal{B}$ chooses random integers $x_i, y_i \in \mathbb{Z}_p^*$, $i = 1, 2, \cdots, t-1$ and sets the $t-1$ proxy signers' public key as $\left(pk_{ix}, pk_{iy}\right) = \left(g^{x_i}, g^{y_i}\right)$.
We note that the simulator $\mathcal{B}$ should correctly guess $t-1$ signers corrupted by $\mathcal{A}_{\rm IV}$ from the signer group A. If the guess is right, the game continues and otherwise the game fails. The probability of right guess is

$$\frac{1}{\begin{pmatrix} n \\ t-1 \end{pmatrix}} = \frac{(t-1)!\ \ (n-t+1)!}{n!} \tag{11}$$

(3) $\mathcal{B}$ assigns $u' = pk_{ay}^{p - l_a k_a + x'_a} g^{y'_a}$, $u_i = pk_{ay}^{x_{ai}} g^{y_{ai}}$ and $\overrightarrow{u} = (u_1, u_2, \cdots, u)$.
(4) $\mathcal{B}$ assigns $m' = pk_{ay}^{p - l_b k_b + x'_b} g^{y'_b}$, $m_i = pk_{ay}^{x_{bi}} g^{y_{bi}}$ and $\overrightarrow{m} = (m_1, m_2, \cdots, m_n)$.
Note that, we have

$$m' \prod_{i \in \mathcal{M}} m_i = pk_{ay}^{F_b(M)} g^{J_b(M)}\ \ ,\ \ u' \prod_{i \in \mathcal{W}} u_i = pk_{ay}^{F_a(\omega)} g^{J_a(\omega)}$$

Finally, $\mathcal{B}$ returns $\left(G, G_T, e, p, g, u', \overrightarrow{u}, m', \overrightarrow{m}\right)$ as the system parameters and $\left(pk_{ax}, pk_{ay}, pk_{ix}, pk_{iy}, pk_{cx}, x_i, y_i\right)$ to adversary $\mathcal{A}_{\rm IV}$.

Here, we note that without loss of generality, we suppose that the all of $t-1$ proxy signers controlled by $\mathcal{A}_{\rm IV}$ identically participate in signature generating stage.

$$g^{x_i y_i} = pk_{ixy},\ pk_{txy} = \sum_{i=1}^{t} pk_{ixy}$$

$$pk_{txy} = \sum_{i=1}^{t-1} pk_{ixy} + pk_{(t-th)\ xy} \to pk_{(t-th)\ xy} = pk_{txy} - \sum_{i=1}^{t-1} pk_{ixy}$$

Delegation queries: Includes the following stages.

1. If $K_a(\omega) = 0$, $\mathcal{B}$ terminates the simulation and report failure.
2. If $K_a(\omega) \neq 0$, this implies $F_a(\omega) \neq 0\ (mod\ p)$ [15]. In this case, for generating the delegation of each proxy signer, $\mathcal{B}$ chooses $r_{a_i} \in \mathbb{Z}_p^*$ randomly and computes the proxy share of the i-th proxy signer as follows

$$\sigma_{\omega_{P_i}} = \left(\sigma_{\omega_{P_i}1}, \sigma_{\omega_{P_i}2}\right) = e\left(pk_{ax}^{-\frac{J_a(\omega)}{F_a(\omega)}}\left(u' \prod_{j\in\mathcal{W}} u_j\right)^{r_{a_i}},\ pk_{ix}\right), \qquad (12)$$
$$R_{a_i} = g^{r_{a_i}}$$

Correctness

$$\begin{aligned}
\sigma_{\omega_{P_i}} &= e\left(pk_{ax}^{-\frac{J_a(\omega)}{F_a(\omega)}}\left(u' \prod_{j\in\mathcal{W}} u_j\right)^{r_{a_i}},\ pk_{ix}\right)\\
&= e\left(g^{-\frac{aJ_a(\omega)}{F_a(\omega)}} \cdot g^{ab} g^{-ab}\left(g^{bF_a(\omega)} \cdot g^{J_a(\omega)}\right)^{r_{a_i}}, pk_{ix}\right)\\
&= e\left(g^{ab} \cdot \left(g^{bF_a(\omega)} \cdot g^{J_a(\omega)}\right)^{-\frac{a}{F_a(\omega)}} \cdot \left(g^{bF_a(\omega)} \cdot g^{J_a(\omega)}\right)^{r_{a_i}}, pk_{ix}\right)\\
&= e\left(g^{ab} \cdot \left(g^{bF_a(\omega)} \cdot g^{J_a(\omega)}\right)^{r_{a_i} - \frac{a}{F_a(\omega)}}, pk_{ix}\right)\\
&\Rightarrow e\left(pk_{ay}^{a}\left(u' \prod_{j\in\mathcal{W}} u_j\right)^{\hat{r}_{a_i}},\ pk_{ix}\right) R_{a_i} = g^{r_{a_i} - \frac{a}{F_a(\omega)}} = g^{\hat{r}_{a_i}}
\end{aligned}$$

In the above equality $\hat{r}_{a_i} = r_{a_i} - \frac{a}{F_a(\omega)}$. $\mathcal{B}$ also publishes the $\sigma_{\omega_{P_i}1}$.

ProxySign queries: During this stage, $\mathcal{A}_{\mathrm{IV}}$ can request the individual signatures of $t-1$ proxy signers corrupted by him/her. Additionally, the adversary $\mathcal{A}_{\mathrm{IV}}$ can request the final proxy signature.

1. If $K_a(\omega) = 0\ ,\ K_b(M) = 0$, $\mathcal{B}$ terminates the simulation and report failure.
2. If $K_a(\omega) = 0\ ,\ K_b(M) \neq 0$, $\mathcal{B}$ picks the random integers $r_{a_i}, r_{b_i} \in \mathbb{Z}_p^*$ and computes individual signature of the i-th proxy signer as follows

$$\sigma_{i_1} = pk_{ax}^{-\frac{J_b(M)}{F_b(M)}}\left(u' \prod_{j\in\mathcal{W}} u_j\right)^{r_{a_i}} g^{x_i y_i}\left(m' \prod_{j\in\mathcal{M}} m_j\right)^{r_{b_i}}, \qquad (13)$$
$$\sigma_{i2} = g^{r_{a_i}} = R_{a_i},\ \sigma_{i3} = g^{r_{b_i} - \frac{a}{F_b(M)}} = g^{\hat{r}_{b_i}}$$

where $\hat{r}_{b_i} = r_{b_i} - \frac{a}{F_b(M)}$.

Correctness

$$
\begin{aligned}
\sigma_{i_1} &= pk_{ax}^{-\frac{J_b(M)}{F_b(M)}} \left(u' \prod_{j\in\mathcal{W}} u_j\right)^{r_{a_i}} g^{x_i y_i} \left(m' \prod_{j\in\mathcal{M}} m_j\right)^{r_{b_i}} \\
&= g^{ab} \cdot g^{-ab} \cdot g^{-\frac{aJ_b(M)}{F_b(M)}} \left(pk_{ay}^{F_b(M)} g^{J_b(M)}\right)^{r_{b_i}} g^{x_i y_i} \left(u' \prod_{j\in\mathcal{W}} u_j\right)^{r_{a_i}} \\
&= g^{ab} \cdot \left(g^{bF_b(M)} g^{J_b(M)}\right)^{-\frac{a}{F(M)}} \left(pk_{ay}^{F_b(M)} g^{J_b(M)}\right)^{r_{b_i}} g^{x_i y_i} \left(u' \prod_{j\in\mathcal{W}} u_j\right)^{r_{a_i}} \\
&= g^{ab} \cdot \left(pk_{ay}^{F_b(M)} g^{J_b(M)}\right)^{r_{b_i} - \frac{a}{F(M)}} g^{x_i y_i} \left(u' \prod_{j\in\mathcal{W}} u_j\right)^{r_{a_i}} \\
&= g^{x_a y_a} \cdot \left(m' \prod_{j\in\mathcal{M}} m_j\right)^{\hat{r}_{b_i}} g^{x_i y_i} \left(u' \prod_{j\in\mathcal{W}} u_j\right)^{r_{a_i}}
\end{aligned}
$$

3. Note that during the simulation of signature, $K_a(\omega) = 0$ should satisfy. If $K_a(\omega) \neq 0$, $\mathcal{B}$ should return to the Delegating simulation and computes the proxy signature again, such that $K_a(\omega) = 0$.

ProxySign Verify queries: Assume that $\mathcal{A}_{\mathrm{IV}}$ issues a verify query for the message/signature pair $(M, \omega, \sigma_{P_1}, \sigma_{P2}, \sigma_{P3})$.

1. If $F_a(\omega) \neq 0$ and $F_b(M) \neq 0$, $\mathcal{B}$ terminates the simulation and report failure.
2. If $F_a(\omega) = 0$, $F_b(M) = 0$, $\mathcal{B}$ submits

$$
\left(g, g^a, g^b, g^c, \frac{\sigma_{P_1}}{e\left(\sum_{i=1}^{t} g^{x_i y_i}, g\right)^c e\left(g^c, \sigma_{P2}\right)^{J_a(\omega)} e\left(g^c, \sigma_{P3}\right)^{J_b(M)}}\right) \tag{14}
$$

to the $DBDH$ oracle $\mathcal{O}_{DBDH}$. If $\mathcal{O}_{DBDH}$ returns 1, $\mathcal{B}$ outputs “valid” and otherwise, $\mathcal{B}$ outputs “invalid”.

Correctness

$$
\begin{aligned}
\sigma_{P1} &= e\left(pk_{ax}, pk_{ay}\right)^{x_c} e\left(pk_{ix}, pk_{iy}\right)^{x_c} \\
&\quad .e\left(u' \prod_{j\in\mathcal{W}} u_j, \sigma_{P2}\right)^{x_c} e\left(m' \prod_{j\in\mathcal{M}} m_j, \sigma_{P3}\right)^{x_c} \\
&= e\left(g^a, g^b\right)^c e\left(\sum_{i=1}^{t} g^{x_i y_i}, g\right)^c .e\left(g^{J_a(\omega)}, \sigma_{P2}\right)^c e\left(g^{J_b(M)}, \sigma_{P3}\right)^c \\
&= e\left(g, g\right)^{abc} e\left(\sum_{i=1}^{t} g^{x_i y_i}, g\right)^c .e\left(g^c, \sigma_{P2}\right)^{J_a(\omega)} e\left(g^c, \sigma_{P3}\right)^{J_b(M)}
\end{aligned}
$$

which indicates that

$$
\left(g, g^a, g^b, g^c, \tfrac{\sigma_{P_1}}{e\left(\sum_{i=1}^{t} g^{x_i y_i}, g\right)^c e\left(g^c, \sigma_{P2}\right)^{J_a(\omega)} e\left(g^c, \sigma_{P3}\right)^{J_b(M)}}\right)
$$

is a valid BDH tuple.

3. If $F_a(\omega) = 0$, $F_b(M) \neq 0$, $\mathcal{B}$ can compute a valid signature on the message M under the same warrant ω just as the second case that he/she responses to proxy signature queries. Let $\left(M, \omega, \sigma'_{P_1}, \sigma'_{P_2}, \sigma'_{P_3}\right)$ be the signature computed by $\mathcal{B}$. The simulator $\mathcal{B}$ submits

$$\left(\left(g^b\right)^{F_b(M)} g^{J_b(M)}, \frac{\sigma_{P_3}}{\sigma'_{P_3}}, g^c, \left(\frac{\sigma_{P_1}}{\sigma'_{P_1}}\right) e\left(g^c, \frac{\sigma'_{P_2}}{\sigma_{P_2}}\right)^{J_b(M)}\right) \tag{15}$$

to the $DBDH$ oracle $\mathcal{O}_{DBDH}$. If $\mathcal{O}_{DBDH}$ returns 1, $\mathcal{B}$ outputs "valid" and otherwise, $\mathcal{B}$ outputs "invalid".

Correctness

If $(M, \omega, \sigma_{P_1}, \sigma_{P2}\ , \sigma_{P3}\ )$ is a valid DVTPS, then

$$\begin{aligned}\sigma_{P1} = {} & e\left(pk_{ax}, pk_{ay}\right)^{x_c} e\left(pk_{ix}, pk_{iy}\right)^{x_c} \\ & e\left(u' \prod_{j \in \mathcal{W}} u_j, \sigma_{P2}\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, \sigma_{P3}\right)^{x_c}\end{aligned}$$

Similarly, since $\left(M, \omega, \sigma'_{P_1}, \sigma'_{P_2}, \sigma'_{P_3}\right)$ is another valid DVTPS computed by $\mathcal{B}$, then

$$\begin{aligned}\sigma'_{P_1} = {} & e\left(pk_{ax}, pk_{ay}\right)^{x_c} e\left(pk_{ix}, pk_{iy}\right)^{x_c} \\ & e\left(u' \prod_{j \in \mathcal{W}} u_j, \sigma'_{P_2}\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, \sigma'_{P_3}\right)^{x_c}\end{aligned}$$

we can write

$$\begin{aligned}\tfrac{\sigma_{P_1}}{\sigma'_{P_1}} &= \left(\tfrac{e\left(u' \prod_{j \in \mathcal{W}} u_j, \sigma_{P2}\right)}{e\left(u' \prod_{j \in \mathcal{W}} u_j, \sigma'_{P_2}\right)}\right)^{x_c} \left(\tfrac{e\left(m' \prod_{j \in \mathcal{M}} m_j, \sigma_{P3}\right)}{e\left(m' \prod_{j \in \mathcal{M}} m_j, \sigma'_{P_3}\right)}\right)^{x_c} \\ &= e\left(u' \prod_{j \in \mathcal{W}} u_j, \tfrac{\sigma_{P_2}}{\sigma'_{P_2}}\right)^{x_c} e\left(m' \prod_{j \in \mathcal{M}} m_j, \tfrac{\sigma_{P_3}}{\sigma'_{P_3}}\right)^{x_c} \\ &= e\left(g^c, \tfrac{\sigma_{P_2}}{\sigma'_{P_2}}\right)^{J_a(\omega)} e\left(pk_{ay}^{F_b(M)} g^{J_b(M)}, \tfrac{\sigma_{P_3}}{\sigma'_{P_3}}\right)^c\end{aligned}$$

therefore

$$\left(\left(g^b\right)^{F_b(M)} g^{J_b(M)}, \tfrac{\sigma_{P_3}}{\sigma'_{P_3}}, g^c, \left(\tfrac{\sigma_{P_1}}{\sigma'_{P_1}}\right) e\left(g^c, \tfrac{\sigma'_{P_2}}{\sigma_{P_2}}\right)^{J_b(M)}\right)$$

is a valid BDH tuple.

4. If $F_a(\omega) \neq 0$, $F_b(M) = 0$, $\mathcal{B}$ can compute a valid signature on the message M under the same warrant ω just as the second case that he/she responses to proxy signature queries. Suppose that $\left(M, \omega, \sigma'_{P_1}, \sigma'_{P_2}, \sigma'_{P_3}\right)$ be the signature computed by $\mathcal{B}$. The simulator $\mathcal{B}$ submits

$$\left(\left(g^b\right)^{F_a(\omega)} g^{J_a(\omega)}, \frac{\sigma_{P_2}}{\sigma'_{P_2}}, g^c, \left(\frac{\sigma_{P_1}}{\sigma'_{P_1}}\right) e\left(g^c, \frac{\sigma'_{P_3}}{\sigma_{P_3}}\right)^{J_b(M)}\right) \tag{16}$$

to the $DBDH$ oracle $\mathcal{O}_{DBDH}$. If $\mathcal{O}_{DBDH}$ returns 1, $\mathcal{B}$ outputs "valid" and otherwise, $\mathcal{B}$ outputs "invalid".

Correctness

It is similar to the previous case.

If $\mathcal{B}$ does not abort during the simulation, the adversary $\mathcal{A}_{\mathrm{IV}}$ will output a valid DVTPS $\sigma_P^* = \left(\sigma_{P_1}^*, \sigma_{P_2}^*, \sigma_{P_3}^*\right)$ on the message M^* under the warrant ω^* with success probability ϵ.

1. If $F_a(\omega^*) \neq 0$, $F_b\left(M^*\right) \neq 0$; $\mathcal{B}$ will abort.
2. Otherwise, $F_a\left(\omega^*\right) = 0$, $F_b\left(M^*\right) = 0$ and $\mathcal{B}$ computes

$$\frac{\sigma_{P_1}^*}{e\left(\sum_{i=1}^{t} g^{x_i y_i}, g\right)^c e\left(g^c, \sigma_{P_2}^*\right)^{J_a(\omega^*)} e\left(g^c, \sigma_{P_3}^*\right)^{J_b(M^*)}} \tag{17}$$

and outputs it as the value of $e\left(g, g\right)^{abc}$.

This completes the description of the simulation. Now we have to compute $\mathcal{B}$'s probability of success. $\mathcal{B}$ will not abort if the following conditions hold.

G: The simulator $\mathcal{B}$ guesses $t-1$ proxy signers corrupted by $\mathcal{A}_{\mathrm{IV}}$, correctly.
A: $K_a\left(\omega\right) \neq 0 \;\; (mod\ l_a)$ during delegation queries.
B: $K_a\left(\omega\right) \neq 0 \;\; (mod\ l_a)$ or $K_b\left(M\right) \neq 0 \;\; (mod\ l_b)$ during ProxySign queries.
C: $F_a\left(\omega\right) = 0 \;\; (mod\ p)$ or $F_b\left(M\right) = 0 \;\; (mod\ p)$ during Verify queries.
D: $F_a\left(\omega^*\right) = 0 \quad (mod\ p)$ and $F_b\left(M^*\right) = 0 \quad (mod\ p)$ in output phase.

Finally, the probability of success is $Succ_{\mathcal{B}}^{GBDH} = \Pr\left[G \wedge A \wedge B \wedge C \wedge D\right] \epsilon$. Now, we compute this probability using Waters' technique [15]. Note that the guess probability $\Pr\left[G\right]$ is independent of other probabilities.

$$\begin{aligned}
&\Pr\left[G \wedge A \wedge B \wedge C \wedge D\right] \\
&= \Pr\left[G\right] \cdot \Pr\left[A \wedge B \wedge C \wedge D\right] \\
&= \tfrac{(t-1)!\ (n-t+1)!}{n!} \cdot \Pr\left[A \wedge B \wedge C \wedge D\right] \text{But} \Pr\left[A \wedge B \wedge C \wedge D\right] \\
&= \Pr\Bigg[\bigcap_{i=1}^{q_\omega} K_a\left(\omega_i\right) \neq 0 \bigcap_{i=1}^{q_{ps}} \left(K_a\left(\omega_i\right) \neq 0 \bigcup K_b\left(M_i\right) \neq 0\right) \\
&\quad \bigcap_{i=1}^{q_v} \left(F_a\left(\omega_i\right) = 0\,(mod\ p) \bigcup F_b\left(M\right) = 0\,(mod\ p)\right) \\
&\quad \bigcap \left(F_a\left(\omega^*\right) = 0\,(mod\ p) \bigcap F_b\left(M^*\right) = 0\,(mod\ p)\right)\Big] \\
&\geq \Pr\Bigg[\bigcap_{i=1}^{q_\omega+q_{ps}} K_a\left(\omega_i\right) \neq 0 \bigcap_{i=1}^{q_v} F_a\left(\omega_i\right) = 0\,(mod\ p) \\
&\quad \bigcap F_a\left(\omega^*\right) = 0\,(mod\ p) \bigcap F_b\left(M^*\right) = 0\,(mod\ p)\Big] \\
&= \Pr\Bigg[\bigcap_{i=1}^{q_\omega+q_{ps}} K_a\left(\omega_i\right) \neq 0\Bigg] \Pr\Bigg[\bigcap_{i=1}^{q_v} F_a\left(\omega_i\right) = 0\,(mod\ p) \\
&\quad \bigcap F_a\left(\omega^*\right) = 0\,(mod\ p) \bigcap F_b\left(M^*\right) = 0\,(mod\ p) \\
&\quad | \bigcap_{i=1}^{q_\omega+q_{ps}} K_a\left(\omega_i\right) \neq 0\Bigg] \\
&\geq \tfrac{1}{(n+1)^3}\ \left(1 - \tfrac{q_\omega+q_{ps}}{l_a}\right) \Pr\Bigg[\bigcap_{i=1}^{q_v} K_a\left(\omega_i\right) = 0 \bigcap K_a\left(\omega^*\right) = 0
\end{aligned}$$

$$\bigcap K_b\left(M^*\right)=0 \mid \bigcap_{i=1}^{q_\omega+q_{ps}} K_a\left(\omega_i\right) \neq 0\Bigg]$$

$$= \tfrac{1}{(n+1)^3}\ \left(1-\tfrac{q_\omega+q_{ps}}{l_a}\right) \frac{\Pr\left[\bigcap_{i=1}^{q_v} K_a(\omega_i)=0 \bigcap K_a\left(\omega^*\right)=0\ \ \bigcap K_b\left(M^*\right)=0\right]}{\Pr\left[\bigcap_{i=1}^{q_\omega+q_{ps}} K_a(\omega_i)\neq 0\right]}$$

$$\Pr\left[\bigcap_{i=1}^{q_\omega+q_{ps}} K_a\left(\omega_i\right) \neq 0 \mid \bigcap_{i=1}^{q_v} K_a\left(\omega_i\right)=0 \bigcap K_a\left(\omega^*\right)=0\ \ \bigcap K_b\left(M^*\right)=0\right]$$

$$\geq \tfrac{1}{(n+1)^3 l_a^{q_v+1} l_b}\ \left(1-\tfrac{q_\omega+q_{ps}}{l_a}\right)$$

$$\times\left(1-\Pr\left[\bigcup_{i=1}^{q_\omega+q_{ps}} K_a\left(\omega_i\right) \neq 0 \mid \bigcap_{i=1}^{q_v} K_a\left(\omega_i\right)=0 \bigcap K_a\left(\omega^*\right)=0\ \ \bigcap K_b\left(M^*\right)=0\right]\right)$$

$$\geq \tfrac{1}{(n+1)^3 l_a^{q_v+1} l_b}\ \left(1-\tfrac{q_\omega+q_{ps}}{l_a}\right)^2$$

$$\geq \tfrac{1}{(n+1)^3 l_a^{q_v+1} l_b}\ \left(1-\tfrac{2\left(q_\omega+q_{ps}\right)}{l_a}\right)$$

Therefore $Succ_{\mathbb{B}}^{GBDH} \leq \frac{(t-1)!\ (n-t+1)!}{(n+1)^3 n!\ l_a^{q_v+1} l_b}\left(1-\frac{2(q_\omega+q_{ps})}{l_a}\right)\epsilon.$
We can get a simplified result by setting $l_a = l_a = 3\left(q_\omega + q_{ps}\right)$. Then

$$Succ_{\mathbb{B}}^{GBDH} \leq \tfrac{(t-1)!\ (n-t+1)!\epsilon}{3(n+1)^3 n!(3(q_\omega+q_{ps}))^{q_v+2}}\ .\ \ \square$$

5.2 Security requirements

1. *Verifiability.* In the proposed scheme, since the original signer's public key is indeed to verify the proxy signature, the designated verifier can be convinced of the original signer's agreement on the signed message.

2. *Undeniability.* Anyone cannot find the proxy signers' private key due to the difficulty of discrete logarithm problem (DLP) and thus each proxy signer know his/her private key. Therefore, when the proxy signers create a valid proxy signature, they can repudiate it because the signature is created by using their private key (x_i, y_i).

3. *Identifiability.* In the proposed scheme, identities information of proxy signers is included explicitly in a valid proxy signature and ω as a form of public key. So, anyone can determine the identities of the proxy signers from the signature created by them and confirm the identities of the proxy signers from the ω.

4. *Prevention of misuse.* Only the actual proxy signers group can issue a valid signature because only they know their private key (x_i, y_i). So, if the proxy signers uses the proxy shares for other purposes, it is their responsibility because only they can generate it. Moreover, the original signer's misuse is also prevented because she cannot compute the valid individual proxy signatures.

6 Conclusions

In recent years, proxy signature schemes in the standard model or in other words, proxy signature schemes without random oracles have attracted the interest of many researchers. In this paper, we proposed the first designated threshold proxy signature scheme and showed that the proposed scheme has provable security based on $GBDH$ assumption in the standard model.

Additionally, the proposed scheme provides all the other security requirements for a threshold proxy signature scheme. The proposed scheme is proven to be existentially unforgeable against four types of adversaries.

References

1. Mambo, M., Usuda, K., Okamoto, E.: proxy signature: delegation of the power to sign messages. IEICE Transactions on Fundamentals. **76** (1996) 1338–1353
2. Lee, J.Y., Cheon, J.H., Kim, S.: An analysis of proxy signatures: Is a secure channel necessary?. In: CT-RSA 2003, in: LNCS, vol. 2612, Springer-Verlag, Berlin. (2003) 68–79
3. Lee, B., Kim, H., Kim, K.: Secure mobile agent using strong nondesignated proxy signature. In: ACISP01, in: LNCS, vol. 2119, Springer-Verlag, Berlin. (2001) 474–486
4. Kim, S.J., Park, S.J., Won, D.H.: Proxy Signatures, revisited. ICICS'97, LNCS 1334, Springer-Verlag, Berlin. (1997) 223–232
5. Zhang, K.: Threshold proxy signature schemes. Information Security Workshop, Japan. (1997) 191–197
6. Hu, J., Zhang, J.: Cryptanalysis & improvement of a threshold proxy signature scheme. Computer Standards & Interfaces. (2009) 169–173
7. Tan, Z.: Improvement on C.-L Hsu et al's threshold proxy signature scheme with known signers. International Conference on Convergence Information Technology. (2007) 1463–1467
8. Kong, F., Yu, J., Qin, B., Li, M., Li, D.: Security Analysis and Improvement of a (t, n) Threshold Proxy Signature Scheme. 8^{th} ACIS International Conference on Engineering, Artificial Intelligent, Networking and Parallel/Distributed Computing. (2007) 923–926
9. Seo, S.H., Shim, K.A., Lee, S.H.: A mediated proxy signature scheme with fast revocation for electronic transactions. Proceedings of the 2^{nd} International Conference on Trust, Privacy and Security in Digital Business, Aug 22-26, 2005, Copenhagen, Denmark. LNCS 3592, Berlin, German: Springer-Verlag. (2005) 22–26
10. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. Proceeding of the First ACM Conference on Computer and Communications Security. (1993) 62–73
11. Boldyreva, A., Palacio, A., Warinschi, B.: Secure Proxy Signature Schemes for Delegation of Signing Rights. http://eprint.iacr.org/2003/096.
12. Gu, C., Zhu, Y.: Provable Security of ID-Based Proxy Signature Schemes. ICC-NMC 2005, LNCS 3619, Springer-Verlag Heidelberg. (2005) 1277–1286
13. Ji, H., Han, W., Zhao, L., Wang, Y.: An Identity-Based Proxy Signature from Bilinear Pairings. WASE International Conference on Information Engineering. (2009) 14–17

14. Boneh, D., Boyen, X.: Efficient selective-id secure identity based encryption without random oracles. In Proceeding of the International Conference on Advances in Cryptology (EUROCRYPT'04), Lecture Notes in Computer Science. Springer-Verlag. (2004)
15. Waters, B.: Efficient identity based encryption without random oracles. Proceedings of Advances in Cryptology-Eurocrypt 2005, May 22 26, 2005, Aarhus, Denmark. LNCS 3494, Berlin, German: Springer-Verlag. (2005) 114–127
16. Huang, X., Susilo, W., Mu, Y., Wu, W.: Proxy Signature Without Random Oracles. In: MSN 2006, in: LNCS, vol. 4325, Springer-Verlag, Berlin (2006) 473–484
17. Yu, Y., Xu, C., Zhang, X., Liao, Y.: Designated verifier proxy signature scheme without random oracles. Computers and Mathematics with Applications. **57**(2009) 1352–1364