Compartment-based and Hierarchical Threshold Delegated Verifiable Accountable Subgroup Multi-signatures

Ahmet Ramazan Ağırtaş[1,2] and Oğuz Yayla[1]

[1]*Institute of Applied Mathematics, Middle East Technical University, Ankara, Turkey*
[2]*Nethermind, UK*
{*agirtas.ramazan, oguz*}*@metu.edu.tr*

Abstract

In this paper, we study the compartment-based and hierarchical delegation of signing power of the verifiable accountable subgroup multi-signature (vASM). ASM is a multi-signature in which the participants are accountable for the resulting signature, and the number of participants is not fixed. After Micali et al.'s and Boneh et al.'s ASM schemes, the verifiable-ASM (vASM) scheme with a verifiable group setup and more efficient verification phase was proposed recently. The verifiable group setup in vASM verifies the participants at the group setup phase. In this work, we show that the vASM scheme can also be considered as a proxy signature in which an authorized user (original signer, designator) delegates her signing rights to a single (or a group of) unauthorized user(s) (proxy signer). Namely, we propose four new constructions with the properties and functionalities of an ideal proxy signature and a compartment-based/hierarchical structure. In the first construction, we apply the vASM scheme recursively; in the second one, we use Shamir's secret sharing (SSS) scheme; in the third construction, we use SSS again but in a nested fashion. In the last one, we use the hierarchical threshold secret sharing (HTSS) scheme for delegation. Then, we show the affiliation of our constructions to proxy signatures and compare our constructions with each other in terms of efficiency and security. Finally we compare the vASM scheme with the existing pairing-based proxy signature schemes.

Keywords: accountable subgroup multi-signatures, proxy signatures, threshold secret sharing, delegation

1 Introduction

Signatory authorities carry out their transactions by assigning a deputy to use this power of signing authority for periods when they cannot be present at their organization. While they can appoint a single deputy among their subordinates, there may also be cases in which they authorize different proxies on different issues. In the literature, several notions define solutions for delegating signing capability to unauthorized users by authorized ones in an organization. *Proxy signature*, which allows an unauthorized user (proxy signer) to sign on behalf of an authorized user (original signer), was first defined by Mambo et al. [28] in 1996. After its first proposal, proxy signatures have been studied in [1, 4, 19, 20, 21,

27, 30, 35, 40, 42]. Moreover the proxy multi-signatures [41], the multi-proxy signatures [17], and the multi-proxy multi-signatures [16] were proposed according to the number of proxies and original signers in the constructions. Among these variants, *proxy multi-signature* is a notion related to the delegation of signing authority, and it was first proposed by Yi et al. [41] in 2000. It is a multi-signature scheme in which a designated proxy signer generates a signature on a common message on behalf of several original signers. Hwang and Shi [17] proposed the concept of the multi-proxy signature scheme in which an original signer can authorize a group of proxy signers as a proxy agent, and a valid multi-proxy signature can be generated only with the participation of all proxy signers. Another proxy signature concept is *multi-proxy multi-signatures* proposed by Hwang and Chen [16] in 2004. In a multi-proxy multi-signature scheme, only the cooperation of a group of original signers designates a group of proxy signers. Then, only the cooperation of all members of the proxy group can generate a valid signature on behalf of the group of original signers. Several proxy signatures also exist, such as threshold, blind, and ring variants [15, 23, 25, 43]. Most of the proposals in the literature either show that the previous ones were insecure or propose a modified one because of the lack of a standard security model. The security properties of an ideal proxy signature are defined in [28] and extended in [21]. These properties are as follows:

- *Verifiability:* A proxy signature should convince any verifier that the original signer agrees with it.
- *Strong unforgeability:* Except for the designated proxy signer, no one can create a valid proxy signature.
- *Strong identifiability:* From a proxy signature, any verifier should identify the identity of the proxy signer.
- *Strong undeniability:* A proxy signature should have non-repudiation property.
- *Prevention of misuse:* Ensuring that a proxy signing key can be used only proxy signing process, no other purposes.

Boldyreva et al. [4] stated that the above security properties were defined informally and needed to be formalized. For this reason, they first defined a formal security model for the proxy signature schemes and defined the functionalities a proxy signature should have. In this work, we show that the verifiable accountable subgroup multi-signature (vASM) scheme [2] can also be used as a proxy signature scheme. The vASM scheme also supports one or more proxies and original signers. In addition, we also show that the signing power of vASM authority can be delegated via appropriate threshold secret sharing schemes [12, 18, 33, 36, 37].

Consider an organization with a complex topology consisting of many compartments; each has many sub-compartments and multi-level hierarchical structures. In this case, the issue of who will deputize for whom would emerge as a challenging problem. In this paper, we propose four constructions using the vASM scheme and several threshold secret sharing schemes to present alternative solutions to the problem of delegation of signing authority problem. We propose that one can have the functionalities of a proxy signature scheme via accountable subgroup multi-signature schemes [2, 5, 29], in particular via the vASM scheme [2], or a combination of the vASM scheme with some proper threshold secret sharing schemes [12, 18, 33, 36, 37]. Assume a scenario that there exists an organization with lots of compartments, and each has an authorized signer and many unauthorized users, just like managers and their subordinates in a hierarchical organizational structure. The authorized signers have signing

rights on behalf of their compartments (also on behalf of the entire organization). They want to assign proxies among unauthorized users in their compartments. To solve this assignment problem, firstly, we apply the vASM scheme recursively. The authorized signer (original signer) and the unauthorized users (proxy signer candidates) jointly participate in a vASM group setup. At the end of this setup phase, each unauthorized user has a *compartment membership key* which can be used to sign on behalf of the authorized user and the entire organization. Then we combine the methods from threshold signatures [3, 9, 13, 14] and vASM scheme to construct solutions for the delegation that supports one or more original/proxy signers and provides accountability at the same time. As a second method, we consider that the authorized users share their membership key via Shamir's secret sharing (SSS) scheme [33] with unauthorized users in their compartment. Then unauthorized users can sign with the sum of their shares and secret keys. Thirdly, we assume a trusted user exists in each compartment and apply Shamir's SSS in a nested fashion. The authorized users first share their membership key via a 2-out-of-2 Shamir's SSS, give one of the shares to the trusted user in their compartment, and then recursively apply another SSS to the second share. At the end of this nested SSS protocol, each unauthorized user has a compartment membership key to sign a proxy signature on behalf of the authorized signer. Finally, we consider a hierarchical structure and think the trusted user(s) are in the first level of the compartment. In this case, the authorized signers share their membership keys via a hierarchical threshold secret sharing (HTSS) scheme [18, 36, 37]. Each unauthorized user can sign on behalf of the authorized signers in an accountable way. After describing the constructions, we see that they all have the main functionalities of a proxy signature according to [4].

The outline of the paper is as follows. In Section 2 we give preliminary information, including definitions of proxy signature schemes, bilinear pairings, computational co-DHP/ψ-co-DHP, Shamir's secret sharing scheme (SSS) [33], Feldman's verifiable secret sharing (VSS) protocol [12], hierarchical threshold secret sharing scheme (HTSS) [18, 36, 37], security discussion of threshold secret sharing schemes and definition of the verifiable-ASM (vASM) scheme [2]. Then we give our proposed constructions in Section 3. In the same section, we also provide remarks about the security of the constructions. In Section 4, we compare our constructions in terms of the number of operations required in the phases of our proposed constructions. Then we compare the vASM scheme with the existing proxy signatures, proxy multi-signatures, multi-proxy signatures and multi-proxy multi-signatures in terms of their efficiency.

2 Preliminary

In [4] the authors give a detailed definition of a proxy signature and define the first formal security model for this notion. Below we give the formal definitions of the digital signature scheme and proxy signature scheme according to [4].

Definition 2.1 (Digital signature scheme [4])**.** A digital signature scheme $\mathcal{DS} = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ is specified by four algorithms which are defined below:

- $\mathcal{G}(1^\lambda)$ takes 1^λ, where λ is the security parameter as input, and outputs the public system parameters par including security parameter, hash functions, cyclic groups, generators, etc.
- $\mathcal{K}(par)$ takes system parameters par as input, and outputs a secret-public key pair (sk, pk).

- $\mathcal{S}(par, M, sk)$ takes system parameters par, message M, and secret key sk as inputs, and returns a signature σ.
- $\mathcal{V}(par, M, pk, \sigma)$ takes system parameters par, message $M \in \{0,1\}^*$, public key pk, a signature σ as inputs, and outputs accept or reject.

Definition 2.2 (Proxy signature scheme [4])**.** A proxy signature scheme is a tuple $\mathcal{PS} = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V}, (\mathcal{D}, \mathcal{P}), \mathcal{PS}, \mathcal{PV}, \mathcal{ID})$, where $\mathcal{DS} = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ are as defined in Definition 2.1, and the other components are defined below:

- $(\mathcal{D}, \mathcal{P})$ is an interactive proxy-designation protocol composed of a pair of algorithms, i.e., $\mathcal{D}$ and $\mathcal{P}$. Let i, j be the designator and the proxy, respectively.
  - $\mathcal{D}(pk_i, sk_i, j, pk_j, \omega)$ takes both sides' public keys, the designator's secret key, the proxy's identity, and the message space ω as inputs, and gives no local outputs.
  - $\mathcal{P}(pk_j, sk_j, pk_i)$ takes public keys of both sides and the secret key of the proxy and outputs proxy signing key skp after the interaction with $\mathcal{D}$.
- $\mathcal{PS}(skp, M)$ takes proxy signing key skp and the message M as inputs, and outputs a proxy signature $p\sigma$.
- $\mathcal{PV}(pk, M, p\sigma)$ takes public key pk, a message M, and a proxy signature $p\sigma$ as inputs, and outputs accept or reject.
- $\mathcal{ID}(p\sigma)$ takes a proxy signature $p\sigma$ and outputs an identity of a signer i or $\perp$.

We will compare the properties of our constructions with the functionalities given in Definition 2.2 in Section 4.

Definition 2.3. Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic additive groups of prime order q. Let $\mathbb{G}_T$ be another cyclic group that is multiplicative and of the same order. A pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ which satisfies the bilinearity and non-degeneracy properties:

- Bilinearity: $e(A^\alpha, B^\beta) = e(A, B)^{\alpha\beta}$ for all $\alpha, \beta \in \mathbb{Z}$, $A \in \mathbb{G}_1$ and $B \in \mathbb{G}_2$.
- Non-degeneracy: $e \neq 1$.

The definitions of underlying hard problems of the vASM scheme, i.e. computational co-DHP, and computational ψ-co-DHP are given below.

Definition 2.4 (Computational co-Diffie-Hellman Problem [7])**.** For groups $\mathbb{G}_1 = \langle g_1 \rangle$ and $\mathbb{G}_2 = \langle g_2 \rangle$ of prime order q, define $Adv^{co\text{-}CDH}_{\mathbb{G}_1, \mathbb{G}_2}$ of an adversary $\mathcal{A}$ as

$$Pr\left[y = g_1^{\alpha\beta} : (\alpha, \beta) \xleftarrow{\$} \mathbb{Z}_q^2, y \leftarrow \mathcal{A}(g_1^\alpha, g_1^\beta, g_2^\beta)\right],$$

where the probability is taken over the random choices of $\mathcal{A}$ and the random selection of (α, β). $\mathcal{A}$ (τ, ϵ)-breaks the co-CDH problem if it runs in time at most τ and has $Adv^{co\text{-}CDH}_{\mathbb{G}_1, \mathbb{G}_2} \geq \epsilon$. co-CDH is (τ, ϵ)-hard if no such adversary exists.

Definition 2.5 (Computational ψ-co-Diffie-Hellman Problem [5])**.** For groups $\mathbb{G}_1 = \langle g_1 \rangle$ and $\mathbb{G}_2 = \langle g_2 \rangle$ of prime order q, let $\mathcal{O}^{\psi}(.)$ be an oracle that on input $g_2^x \in \mathbb{G}_2$ returns $g_1^x \in \mathbb{G}_1$. Define $Adv^{\psi\text{-}co\text{-}CDH}_{\mathbb{G}_1,\mathbb{G}_2}$ of an adversary $\mathcal{A}$ as

$$Pr\left[y = g_1^{\alpha\beta} : (\alpha, \beta) \xleftarrow{\$} \mathbb{Z}_q^2, y \leftarrow \mathcal{A}^{\mathcal{O}^{\psi}(.)}(g_1^{\alpha}, g_1^{\beta}, g_2^{\beta})\right],$$

where the probability is taken over the random choices of $\mathcal{A}$ and the random selection of (α, β). $\mathcal{A}$ (τ, ϵ)-breaks the ψ-co-CDH problem if it runs in time at most τ and has $Adv^{\psi\text{-}co\text{-}CDH}_{\mathbb{G}_1,\mathbb{G}_2} \geq \epsilon$. ψ-co-CDH is (τ, ϵ)-hard if no such adversary exists.

2.1 Shamir's Secret Sharing (SSS) Scheme

Shamir's secret sharing (SSS) scheme [33] is a protocol that is used for sharing a secret among some predetermined players. Assume that we have n players. Let $\mathbb{F}_q$ be a finite field with prime order q. The dealer delivers the shares as follows:

- Chooses a polynomial over $\mathbb{F}_q$ of degree $t - 1 < q$,

  $$f(x) = \alpha_{t-1}x^{t-1} + \ldots + \alpha_1 x + \alpha_0$$

  with $\alpha_k \in \mathbb{F}_q$ for $k = 0, \ldots, t-1$, where α_0 is the secret to be shared.

- Sends $f(i)$ to the i-th player for $i = 1, 2, \ldots, n$.

If at least t or more players perform Lagrange interpolation with their shares, they can uniquely determine the secret polynomial $f(x)$ and $f(0)$ will yield the secret.

2.2 Feldman's Verifiable Secret Sharing (VSS) Scheme

Feldman's verifiable secret sharing (VSS) scheme [12] is a protocol that is used for sharing a secret in a verifiable fashion, where Shamir's secret sharing scheme [33] is directly used to share and reconstruct the secret. In addition to Shamir's scheme, the shares can be checked for consistency in Feldman's scheme. To this end, the dealer computes commitments with the coefficients of the secret polynomial so that users can verify that they receive consistent shares from the dealer.

Assume that we have n players. Let $\mathbb{F}_q$ be a finite field with prime order q and g be a primitive element in $\mathbb{F}_q$. The dealer shares a secret as follows:

- Chooses a polynomial over $\mathbb{F}_q$ of degree $t - 1 < q$,

  $$f(x) = \alpha_{t-1}x^{t-1} + \ldots + \alpha_1 x + \alpha_0$$

  with $\alpha_k \in \mathbb{F}_q$ for $k = 0, \ldots, t-1$, where α_0 is the secret to be shared.

- Computes a set of commitments $\text{COM} = \{C_k : C_k = g^{\alpha_k}, k = 0, 1, \ldots, t-1\}$.

- Sends $f(i)$ and COM to the i-th player for $i = 1, 2, \ldots, n$.

After receiving a share and the set of commitments, the i-th player checks

$$g^{f(i)} \stackrel{?}{=} \prod_{k=0}^{t-1} C_k^{i^k}. \tag{2.1}$$

The received share is consistent only if (2.1) is satisfied. The reconstruction of the secret is identical to Shamir's secret sharing scheme. The users can uniquely determine the secret polynomial by applying the Lagrange interpolation if and only if threshold t is satisfied.

Definition 2.6 (Lagrange Interpolation)**.** Given t points (x_i, y_i) for distinct x_i's and $i = 1, \ldots, t$, the unique polynomial of degree $t-1$ which satisfies all the points is the linear combination of Lagrange basis polynomials, and given by the equation

$$P(x) = \sum_{i=1}^{t-1} y_i \ell_i(x), \tag{2.2}$$

where the Lagrange basis polynomial $\ell_i(x)$ is given by the equation

$$\ell_i(x) = \prod_{\substack{0 \le k \le t-1 \\ i \ne k}} \frac{x - x_k}{x_i - x_k} \tag{2.3}$$

.

Now we define another notion called the Lagrange coefficient, which we will use in the signature aggregation and verification phases of our constructions in Section 3.2 and 3.3.

Definition 2.7 (Lagrange Coefficient)**.** Given a set of t points (x_i, y_i) for distinct x_i's and $i = 1, \ldots, t$, the Lagrange coefficient λ_i is the evaluation of the the Lagrange basis polynomial $\ell_i(x)$ at 0, i.e.

$$\lambda_i = \ell_i(0) = \prod_{\substack{0 \le k \le t-1 \\ i \ne k}} \frac{-x_k}{x_i - x_k}. \tag{2.4}$$

2.3 Hierarchical Threshold Secret Sharing Scheme

Hierarchical threshold secret sharing schemes in[18, 36, 37] were proposed for sharing secrets in a partitioned structure of users. Assume that we have a set $\mathcal{G}$ of n players, which is composed of m disjoint subsets, $\mathcal{G} = \bigcup_{i=1}^{m} \mathcal{G}_i$ where $\mathcal{G}_i \cap \mathcal{G}_j = \emptyset$ for $i \ne j$. Let $0 < k_1 < \ldots < k_m$ be a sequence of integers. The access structure Γ of the hierarchical threshold secret sharing scheme in [18] is

$$\Gamma = \{\mathcal{V} \subset \mathcal{G} : |\mathcal{V} \cap (\bigcup_{j=1}^{i} \mathcal{G}_j)| \ge k_i | \forall i \in \{1, 2, \ldots, m\}\}$$

Like in Shamir's SSS, the dealer delivers the shares as follows:

- Chooses a random polynomial of degree k_m

  $$f(x) = \sum_{i=0}^{k_m} \alpha_i x^i$$

  such that $\alpha_0 = s$.

- Sends a share $(x_u, f^{(k_{i-1}+1)}(x_u))$ to the user $u \in \mathcal{G}_i$, where $f^{(k_{i-1}+1)}$ is the $(k_{i-1}+1)$-th derivative of f, $k_0 = -1$ and $x_u \in \mathbb{F}_q$ is a part of the share corresponding to the user u.

The secret reconstruction is performed by Birkhoff interpolation defined below only if the level-specific thresholds are satisfied.

Definition 2.8 (Birkhoff Interpolation)**.** Let the triplet $\langle X, E, C \rangle$ be as follows:

- $X = \{x_1, \ldots, x_k\}$ be a given set of points in $\mathbb{R}$, where $x_1 < x_2 < \ldots < x_k$,
- $E = (e_{i,j})$ for $i = 0, \ldots, k$ and $j = 0, \ldots, \ell$ be a matrix with binary entries, $I(E) = \{(i,j) : e_{i,j} = 1\}$, $d = |I(E)|$, and
- $C = \{c_{i,j} : (i,j) \in I(E)\}$ be a set of d real values (we assume hereafter that the right-most column in E is nonzero).

Then, the Birkhoff interpolation problem that corresponds to the triplet $\langle X, E, C\rangle$ is the problem of finding a polynomial $P(x) \in \mathbb{R}_{d-1}[x]$ that satisfies the d equalities

$$P^{(j)}(x_i) = c_{i,j}, (i,j) \in I(E). \tag{2.5}$$

The matrix E is called the interpolation matrix [36].

We summarize the Birkhoff interpolation method as described in [11]. Let $\phi = \{g_0, g_1, \ldots, g_{d-1}\}$ be a system of linearly independent, $d-1$ times continuously differentiable real-valued, functions and $I'(E) = \{\alpha_i : i = 1, \ldots, d\}$ be a vector that is obtained by lexicographically ordering of entries of $I(E)$. Furthermore, let $\alpha_i(1)$ and $\alpha_i(2)$ denote the first and second elements of the pair $\alpha_i \in I'(E)$. Finally, let $C' = \{c'_i : i = 1, \ldots, d\}$ be another vector obtained by lexicographically ordering entries of C (according to the indexes of elements in C).

According to the above definition and clarifications, the Birkhoff interpolation problem is solved by the below equation:

$$P(x) = \sum_{j=0}^{d-1} \frac{det(A(E, X, \phi_j))}{det(A(E, X, \phi))} g_j(x), \tag{2.6}$$

where

$$A(E, X, \phi_j) = (\theta_{ij})_{d \times d}, \tag{2.7}$$

$\theta_{ij} = g_{j-1}^{(\alpha_i(2))}(x_{\alpha_i(1)})$ for $i, j = 1, \ldots, d$, and $A(E, X, \phi)$ can be computed by replacing $(j+1)$-th column of matrix (2.7) with C'. An explicit example of the application of Birkhoff interpolation using (2.6) can be found in [11].

Although the Birkhoff interpolation problems can be solved by (2.6), we cannot directly use this method. In our constructions, we use Birkhoff interpolation for signature aggregation and verification of the aggregated signature. In order to compute (2.6), any combiner and verifier need to know the sufficient number of shares. However, our constructions require only the shareholders to know their shares, and no one should learn the shares of others. To this end, we use the modified version of (2.6), which is also given in [11]:

$$P(x) = \sum_{i=0}^{d-1} c'_{i+1} \left(\sum_{j=0}^{d-1} (-1)^{(i+j)} \frac{det(A_i(E, X, \phi_j))}{det(A(E, X, \phi))} g_j(x) \right) \tag{2.8}$$

Now we can define the Birkhoff coefficient, which we will use in the signature aggregation and verification phases of our last construction in Section 3.4.

Definition 2.9. Let the triplet $\langle X, E, C\rangle$ be in Definition 2.8 then the Birkhoff coefficient β_i is the evaluation of the polynomial

$$P_i(x) = \sum_{j=0}^{d-1} (-1)^{(i+j)} \frac{det(A_i(E, X, \phi_j))}{det(A(E, X, \phi))} g_j(x),$$

at 0, i.e. $\beta_i = P_i(0)$.

2.4 Security of the secret sharing schemes

Shamir's secret sharing (SSS) scheme [33] is known to be information-theoretically secure, i.e. secure against even computationally unbounded adversaries. The hierarchical threshold secret sharing scheme (HTSS) [18, 36, 37] is a generalization of Shamir's SSS and is also information-theoretically secure. However, suppose an adversary can behave like a shareholder and interact actively with real shareholders. In that case, he can obtain the secret as described in [38] by Tompa and Woll. The attack works as follows. Consider an adversary behaving actively. He chooses a random *false* share and performs an interaction for reconstruction with $t-1$ honest shareholders. As a result, he will not obtain the secret because there are $t-1$ true shares where there must be at least t. However, he obtains a value, let us say s'. Then he interacts with another group of $t-1$ honest shareholders and gets another value $\hat{s}$. He can compute the secret without having a genuine share using the values s' and $\hat{s}$ and corresponding Lagrange coefficients. To avoid this attack, one should force the shareholders to behave passively. For example, using Feldman's verifiable secret sharing (VSS) scheme [12] would be a solution. Before the reconstruction phase, one checks whether the shares to be interpolated are consistent with the shared secret. This will avoid the active adversary attack defined in [38]. On the other hand, Feldman's VSS has its own security risks.

As we stated before, Feldman's VSS scheme uses SSS, and so it has the same security arguments about sharing and reconstruction phases. However, in the committing phase, it is not information-theoretically secure anymore. The commitment set contains $C_0 = g^s$, where g is the generator for the cyclic group, and s is the secret to be shared. This commitment may leak information about the secret s. The security of the commitments depends on the Discrete Logarithm Problem (DLP), defined over cyclic groups. In some cyclic groups, even with a large order, DLP may not be as hard as it is supposed to be. Therefore the space that we are working in should be chosen carefully. In this paper, all the schemes that we propose are pairing-based constructions. In the literature, there are many secure and efficient pairing-friendly curves that we can choose.

2.5 vASM Scheme

Accountable subgroup multi-signature (ASM) schemes were studied by [2, 5, 29]. For instance, the Schnorr-based ASM scheme was proposed in 2001 by Micali et al. [29], and the BLS-based ASM scheme was given in 2018 by Boneh et al. [5]. Then, the verifiable-ASM (vASM) scheme was proposed recently in [2] with a different group setup method and more efficient verification. It is also a BLS-based scheme, and its security depends on the hardness of computational co-DHP and computational ψ-co-DHP problems which are given in Definitions 2.4 and 2.5. In the vASM scheme, each user generates her secret and public key pair independently. Then all users jointly perform a group setup in which they participate in a verifiable secret sharing (VSS) protocol [12]. At the end of this procedure, each user obtains a membership key and a membership public key, which satisfy a common public commitment generated in the group setup phase. Then each user signs a common message M and sends her individual signature to the designated combiner. This combiner could be either one of the signers or a specifically assigned party.

Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic additive groups of prime order q. Let $\mathbb{G}_T$ be another cyclic group that is multiplicative and of order q. Let e be an efficient bilinear pairing, defined over the groups $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$

as in Definition 2.3. Let H be a hash function such that $\mathrm{H} : \{0,1\}^* \to \mathbb{G}_1$. Finally, assume that we have a group $\mathcal{G}$ of n potential signers, and the subgroup $\mathcal{S} \subseteq \mathcal{G}$ is the set of τ signers among those n potential ones. Below we give the steps of the vASM scheme given in [2].

1. Key Generation: Each user $i \in \mathcal{G}$ picks a secret key $sk_i \xleftarrow{\$} \mathbb{Z}_q$, and computes the public key $pk_i \leftarrow g_2^{sk_i}$, where g_2 is a generator of $\mathbb{G}_2$.

2. Group Setup: Each user $i \in \mathcal{G}$ proceeds as follows:

   - Chooses a polynomial $f_i(x) = \alpha_{n-1}^{(i)} x^{n-1} + \ldots + \alpha_1^{(i)} x + \alpha_0^{(i)} \in \mathbb{Z}_q[x]$, where $\alpha_0^{(i)} = sk_i$ and $\alpha_k^{(i)}$'s are all nonzero and distinct, for $k = 1, \ldots, n-1$.
   - Computes the set of commitments $\mathrm{COM}_i := \{C_k^{(i)} = g_2^{\alpha_k^{(i)}} | k = 0, \ldots, n-1\}$.
   - Sends $(f_i(j), \mathrm{COM}_i)$ to j-th user in $\mathcal{G}$, for $j = 1, \ldots, n$.
   - After receiving $(f_j(i), \mathrm{COM}_j)$,
     - computes the membership key $mk_i = \sum_{j \in \mathcal{G}} f_j(i)$.
     - computes $\mathrm{COM} := \{C_k = \prod_{j \in \mathcal{G}} C_k^{(j)} | k = 0, \ldots, n-1\}$.
   - Checks:

     (a) $C_0 \stackrel{?}{=} \prod_{i \in \mathcal{G}} pk_i$

     (b) $g_2^{mk_i} \stackrel{?}{=} \prod_{k=0}^{n-1} C_k^{i^k}$
   - If either (a) or (b) fails, then she aborts. Else, she defines $\mathrm{MPK} = \{mpk_i = g_2^{mk_i}\}_{i \in \mathcal{G}}$, and makes MPK and COM public.

3. Signature Generation: A signer $i \in \mathcal{G}$ computes his/her individual signature $s_i = H_0(m)^{mk_i}$ on the message m and sends s_i to the combiner.

4. Signature Aggregation: After receiving the individual signatures of the signers, the combiner first forms the set of signers $\mathcal{S} \subseteq \mathcal{G}$. Then, she computes the aggregated subgroup multi-signature $\sigma = \prod_{i \in \mathcal{S}} s_i$.

5. Verification: Anyone, who is given $\{par, \mathrm{MPK}, \mathrm{COM}, \mathcal{S}, m, \sigma\}$, can verify the signature σ by checking

$$e\big(H_0(m), \prod_{i \in \mathcal{S}} mpk_i\big) \stackrel{?}{=} e(\sigma, g_2). \tag{2.9}$$

Correctness of the vASM scheme follows from the following equation array.

$$\begin{aligned} e\big(H_0(m), \prod_{i\in\mathcal{S}} mpk_i\big) &= e(H_0(m), \prod_{i\in\mathcal{S}} g_2^{mk_i}) \\ &= e(H_0(m), g_2^{\sum_{i\in\mathcal{S}} mk_i}) \\ &= e(H_0(m)^{\sum_{i\in\mathcal{S}} mk_i}, g_2) \\ &= e(\prod_{i\in\mathcal{S}} H_0(m)^{mk_i}, g_2) \\ &= e(\prod_{i\in\mathcal{S}} s_i, g_2) \\ &= e(\sigma, g_2) \end{aligned}$$

Remark 2.10. Since all users share their secret keys, the first commitments in their individual commitment set have to be equal to their public key. Therefore, the first check in the group setup phase shows that users share their secret keys, not rogue ones [6].

Remark 2.11. The second check is the standard consistency check of Feldman's VSS scheme as described in Section 2.2, whose purpose is to check whether the shares received from other users are consistent with the shared secrets.

Detailed information about the vASM scheme, including security proof, efficiency comparison, and remarks, can be found in [2].

3 Compartment-based and Hierarchical Threshold Delegation of vASM Authority

In general, the delegation of signing capability can be achieved by giving a power of attorney. For example, consider an organization with a sophisticated and complicated nature whose structure expands in vertical and horizontal directions. In such a case, signing authorities may need to assign multiple proxies simultaneously. In this section, we propose four constructions. In the first one, we apply the vASM scheme recursively. In the second construction, we use Shamir's secret sharing scheme (SSS) [33] directly among the users of each compartment. In the third one, we assume the existence of at least one trusted user in each compartment, and we share the vASM authority by a secret sharing scheme in a nested fashion. In the last one, we use the hierarchical threshold secret sharing scheme [18, 36] to delegate the vASM signing authority of an authorized users to the unauthorized users in the same compartment, which is partitioned hierarchically.

Consider a group of users $\mathcal{G} = \bigcup_{i=1}^{m} \mathcal{U}_i$ which is a union of distinct compartments $\mathcal{U}_i$ for $i = 1, \ldots, m$. Without loss of generality, we assume that only one authorized user (or original signer) exists in each compartment $\mathcal{U}_i$, that is AU_i. Assume that each authorized user $\mathrm{AU}_i \in \mathcal{U}_i$ participates in a vASM group setup, obtains her membership key mk_i, and wants to delegate his vASM signing authority to some unauthorized users (or proxy signers) in her compartment, i.e. $u_{ij} \in \mathcal{U}_i$ for $j = 1, \ldots, k_i$, where $k_i \in \mathbb{Z}$ is the number of proxy candidates (unauthorized users) in the i-th compartment. In the remaining part, we use the notation above. Let the functions e and H be as in Section 2.5.

3.1 Recursive vASM as a proxy signature

The output of a vASM group setup is a membership key and a membership public key for each participant, which are used for signing and verification, respectively. Consider an authorized user $\mathrm{AU}_i \in \mathcal{U}_i$ has her membership key mk_i that she calculated with the other authorized users $\mathrm{AU}_j \in \mathcal{U}_j$, for $j = 1, \ldots, m$, in a group setup as described in Section 2.5. Moreover, they also publish a set of membership public keys MPK, and a global commitment set COM at the end of that group setup. Assume that AU_i wants to delegate the signing power of her membership key to a certain number of unauthorized users in her compartment. Let I_i be the union of the indices of AU_i and the set of unauthorized users in i-th compartments that AU_i wants to designate as proxies. To that end, the authorized user AU_i and the unauthorized users $u_{ij} \in I_i$ jointly participate in another vASM group setup, which we call as *compartment setup*. Each unauthorized participant $u_{ij} \in I_i$ joins the compartment setup protocol with his secret key sk_{ij} for $j = 1, \ldots, k_i$. However, the authorized user AU_i participates in the compartment setup with her membership key mk_i. At the end of this compartment setup, each unauthorized user $u_{ij} \in I_i$ obtains a compartment membership key cmk_{ij} for $j = 1, \ldots, k_i$, along with compartment public key set $\mathrm{CPK} := \{cpk_j : j = 1, \ldots, k_i\}$, a proxy commitment set $\mathrm{PCS} := \{PC_j : j = 1, \ldots, k_i\}$ that contains the commitments of the VSS protocol they participate in, see Figure 1. Here the compartment membership keys can be seen as proxy signing keys, and the compartment public keys in CPK can be seen as the proxy verification keys. Below we give the steps of this construction.

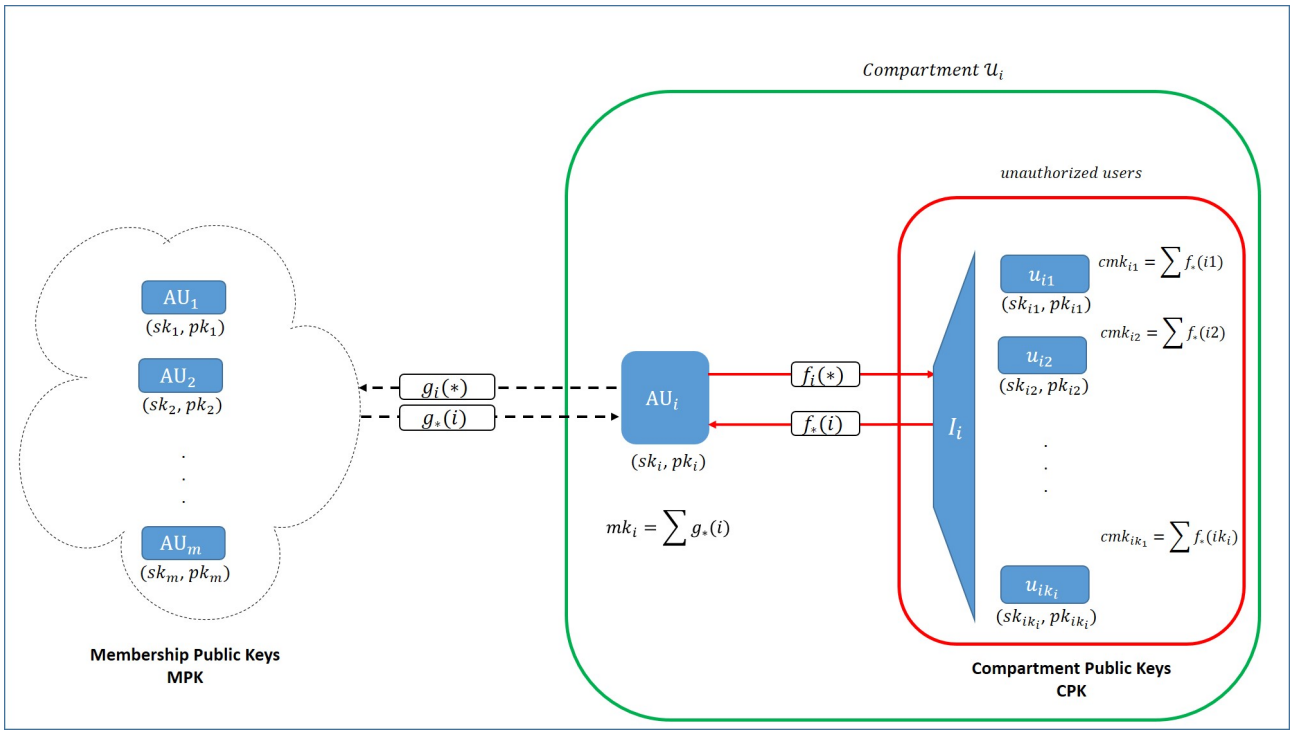

Figure 1: Membership key generation of recursive vASM

1. **Key Generation:**

   - The authorized user AU_i has her membership key mk_i, membership public key mpk_i, and corresponding commitment set COM by performing a group setup with other authorized

users.

- Each unauthorized user $u_{ij} \in I_i$ has their secret and public key pairs sk_{ij}, pk_{ij} as described in Section 2.5.

2. **Compartment Setup:** Both AU_i and $u_{ij} \in I_i$ proceed as follows, see also Figure 1:

   - Choose a polynomial $f_j(x) = \alpha_{k_i-1}^{(j)} x^{k_i-1} + \ldots + \alpha_1^{(j)} x + \alpha_0^{(j)} \in \mathbb{Z}_q[x]$, where $\alpha_0^{(j)} = sk_{ij}$ (or mk_i for AU_i) and $\alpha_w^{(j)}$'s are all nonzero and distinct, for $w = 1, \ldots, k_i - 1$.
   - Compute the set of commitments $\mathrm{PCS}_j := \{PC_w^{(j)} = g_2^{\alpha_w^{(j)}} | w = 0, \ldots, k_i - 1\}$.
   - Send $(f_j(z), \mathrm{PCS}_j)$ to z-th user in I_i, for $z = 1, \ldots, k_i$.
   - After receiving $(f_z(j), \mathrm{PCS}_z)$,
     - computes the membership key $cmk_{ij} = \sum_{z \in I_i} f_z(j)$, and
     - computes $\mathrm{PCS}_i := \{PC_w = \prod_{z \in I_i} PC_w^{(z)} | w = 0, \ldots, k_i\}$.
   - Checks:
     (a) $PC_0 \stackrel{?}{=} mpk_i \cdot \prod_{j \in I_i} pk_{ij}$ ($mpk_i \in \mathrm{MPK}$ is the membership public key of the authorized user AU_i)
     (b) $g_2^{cmk_{ij}} \stackrel{?}{=} \prod_{w=0}^{k_i-1} (PC_w)^{j^w}$
   - If either (a) or (b) fails, then they abort. Else, define $\mathrm{CPK}_i = \{cpk_{ij} = g_2^{cmk_{ij}}\}_{j \in \mathcal{U}_i}$, and make CPK_i and PCS_i public.

3. **Signature Generation:** A designated proxy signer (among the unauthorized users) $u_{ij} \in I_i$ computes his individual signature $s_{ij} = \mathrm{H}(M)^{cmk_{ij}}$ on the message M and sends s_{ij} to the designated combiner.

4. **Signature Aggregation:** After receiving the individual signatures of the proxy signers, the designated combiner first forms the subgroup of proxy signers $\mathcal{S}_i \subseteq I_i$. Then, she computes the aggregated subgroup multi-signature $\sigma_i = \prod_{j \in \mathcal{S}_i} s_{ij}$.

5. **Verification:** Anyone, who is given $\{par, \mathrm{CPK}_i, \mathcal{S}_i, M, \sigma_i\}$, can verify the signature σ_i by checking

$$e\big(H(M), \prod_{j \in \mathcal{S}_i} cpk_{ij}\big) \stackrel{?}{=} e(\sigma_i, g_2). \tag{3.1}$$

Verification satisfies correctness as given below:

$$\begin{aligned} e(\sigma_i, g_2) &= e(\prod_{j \in \mathcal{S}_i} s_{ij}, g_2) \\ &= e(\mathrm{H}(M)^{\sum_{j \in \mathcal{S}_i} cmk_{ij}}, g_2) \\ &= e(\mathrm{H}(M), g_2^{\sum_{j \in \mathcal{S}_i} cmk_{ij}}) \\ &= e(\mathrm{H}(M), \prod_{j \in \mathcal{S}_i} cpk_{ij}). \end{aligned}$$

Security. Note that this is indeed a vASM signature scheme. The only difference from the known vASM is the secret that is shared in the compartment setup phase. The authorized user AU_i shares

his membership key mk_i from the previous group setup with the other authorized users of the other compartments (see Figure 1) while the unauthorized users share their secret keys. Hence, the security of the construction follows from the security of the vASM scheme.

For the compartment setup phase, we should clarify the purpose of consistency checks. There are two consistency checks in the compartment setup. The first check is performed to ensure that the participants know their shared secret. Since the first proxy commitment $PC_0^{(j)} = g_2^{\alpha_0^{(j)}}$, and $\alpha_0^{(j)} = sk_j$(or mk_j), the aggregation of the first commitments $PC_0 = mpk_i \cdot \prod_{j \in I_i} pk_{ij}$ is nothing but the aggregation of the public keys pk_{ij} of the corresponding shared secret keys sk_{ij}(and mpk_i for mk_i). The second check is a standard consistency check for the VSS used in group setup, i.e., to guarantee that the received shares are consistent with the shared secrets. On the other hand, threshold solutions require a trusted combiner and honest majority assumption because any sufficient number (threshold) of malicious users can forge a valid signature. However, using the vASM scheme, each signer is responsible only for his signature. Even if all the unauthorized users (proxy) come together, they cannot forge the membership key of the authorized user.

Remark 3.1. Any authorized user can assign a proxy via a vASM signature scheme. If she wants to delegate her own or organizational signing power, she participates in the compartment setup with her secret or membership key, respectively.

Remark 3.2. One can also consider sharing the membership key mk_i of the authorized user AU_i via a verifiable secret sharing scheme instead of an interactive compartment setup. However, in this case, one should be aware of the need for honest majority assumption.

In the following, we propose three more constructions in which we consider the authorized user AU_i to share her membership key mk_i with different threshold SSS solutions with a trusted user and/or honest majority assumption.

3.2 Shamir's Secret Sharing Scheme Based Delegation

In this construction of the delegation, each authorized user $\mathrm{AU}_i \in \mathcal{U}_i$ delegates her signing authority by sharing her membership key mk_i to a subset of unauthorized users $u_{ij} \in \mathcal{U}_i$ via (t_i, k_i)-Shamir's SSS for $j = 1, \ldots, k_i$, where t_i is the threshold and k_i is the number of users in compartment $\mathcal{U}_i$ as shown in Figure 2.

Then, the unauthorized users $u_{ij} \in \mathcal{G}$ for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, k_i$ sign as follows.

1. **Key Generation:** Each user $u_{ij} \in \mathcal{G}$ picks uniformly at random a secret key $sk_{ij} \xleftarrow{\$} \mathbb{Z}_q$, and computes the public key $pk_{ij} \leftarrow g_2^{sk_{ij}}$, where g_2 is a generator of $\mathbb{G}_2$.

2. **Compartment Setup:** Each AU_i shares her membership key mk_i via a (t_i, k_i)-Shamir's SSS [33] to a subset of unauthorized users $u_{ij} \in \mathcal{U}_i$ as shown in Figure 2. At the end of this secret sharing procedure, each user $u_{ij} \in \mathcal{U}_i$ obtains a compartment membership key $cmk_{ij} = f_i(u_{ij})$.

3. **Signature Generation:** Each user $u_{ij} \in \mathcal{U}_i$ computes his individual signature on the message M as $s_{ij} = \mathrm{H}(M)^{sk_{ij} + cmk_{ij}}$, and sends it to the combiner.

4. **Signature Aggregation:** After receiving the individual signatures from the users, the combiner

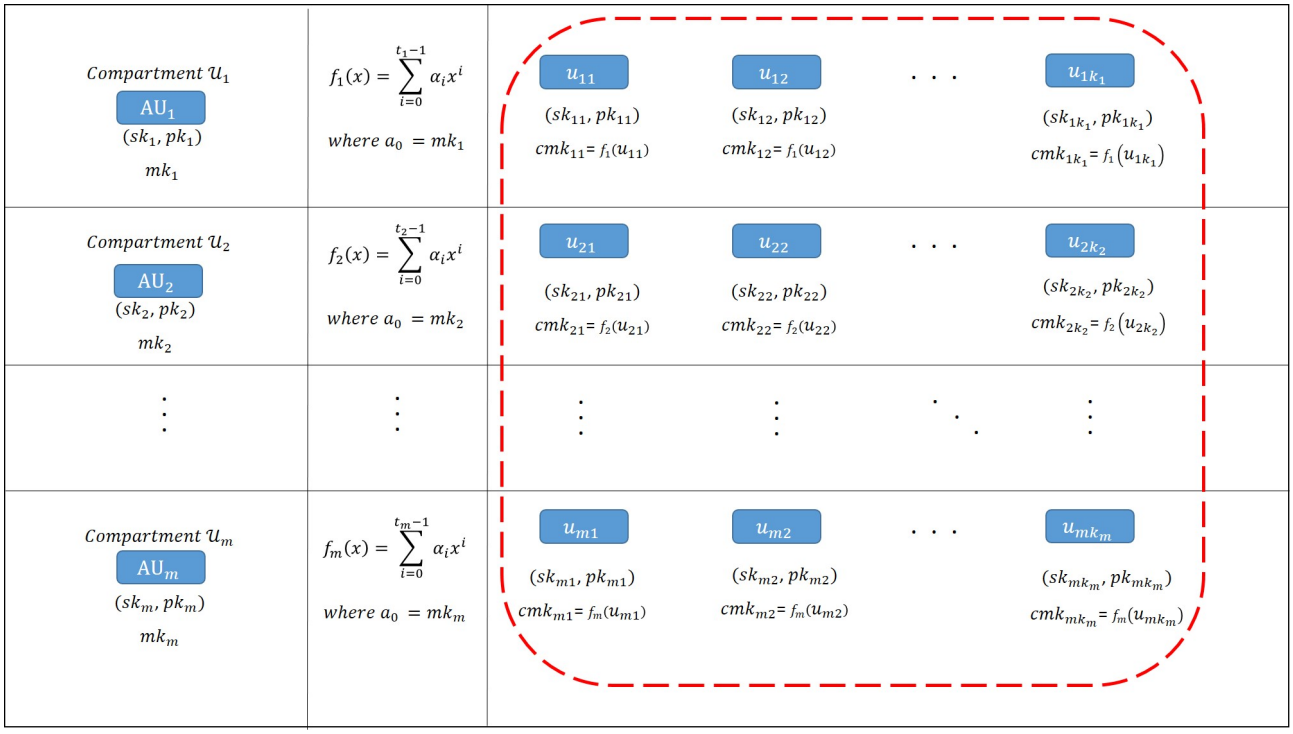

Figure 2: Shamir's Secret Sharing Scheme Based Delegation

(a) forms the subgroup of signers $\mathcal{S} := \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$, where $\mathcal{S}_i$ is the subgroup of signers from compartment $\mathcal{U}_i$, for $i = 1, \ldots, m$.

(b) computes $\sigma = \prod_{i=1}^{m} \prod_{j \in \mathcal{S}_i} s_{ij}^{\lambda_{ij}}$, where λ_{ij} is the appropriate Lagrange coefficient as described in Definition 2.7.

5. **Verification:** If $|\mathcal{S}_i| \geq t_i$ for $i = 1, \ldots, m$, any verifier given $(par, \text{MPK}, \mathcal{PK}, \mathcal{S}, M, \sigma)$ can verify the signature by checking the below equation

$$e\left(\text{H}(M), \prod_{i=1}^{m} mpk_i \prod_{j \in \mathcal{S}_i} pk_{ij}^{\lambda_{ij}}\right) \stackrel{?}{=} e(\sigma, g_2).$$

The verification equation satisfies correctness as given below:

$$\begin{aligned}
e(\sigma, g_2) &= e(\prod_{i=1}^{m} \prod_{j \in \mathcal{S}_i} s_{ij}^{\lambda_{ij}}, g_2) \\
&= e(\text{H}(M)^{\sum_{i=1}^{m} \sum_{j \in \mathcal{S}_i} \lambda_{ij} sk_{ij} + \lambda_{ij} cmk_{ij}}, g_2) \\
&= e(\text{H}(M), g_2^{\sum_{i=1}^{m} \sum_{j \in \mathcal{S}_i} \lambda_{ij} sk_{ij} + \lambda_{ij} cmk_{ij}}) \\
&= e(\text{H}(M), g_2^{\sum_{i=1}^{m} \sum_{j \in \mathcal{S}_i} \lambda_{ij} cmk_{ij}} g_2^{\sum_{i=1}^{m} \sum_{j \in \mathcal{S}_i} \lambda_{ij} sk_{ij}}) \\
&= e(\text{H}(M), \prod_{i=1}^{m} mpk_i \prod_{j \in \mathcal{S}_i} pk_{ij}^{\lambda_{ij}}).
\end{aligned}$$

Security. In this scenario, each user u_{ij} signs a common message M with the sum of his secret and compartment membership keys, i.e. $sk_{ij} + cmk_{ij}$. The resulting individual signature of user u_{ij} is

$s_{ij} = H(M)^{sk_{ij}+cmk_{ij}}$ which can be seen as a BLS signature [7] on message M with the key $sk_{ij} + cmk_{ij}$. Since sk_{ij} is sampled randomly and the cmk_{ij} is computed via an information-theoretically secure SSS scheme, the signing key $sk_{ij} + cmk_{ij}$ will also be random-looking. Assume that the membership key cmk_{ij} is somehow obtained by an adversary. Even in this case, if the secret key sk_{ij} is secure, the individual signature s_{ij} of the user u_{ij} cannot be forged. On the other hand, if a sufficient number of unauthorized users are corrupted, then they can reconstruct the membership key mk_i of the authorized user AU_i. Therefore, we assume for this scenario that the number of corrupted users is less than threshold t. Assigning a trusted user is a solution to this problem. In the following constructions, unauthorized users cannot obtain the membership key of the authorized user AU_i without compromising with the trusted user(s).

3.3 Trusted User Based Delegation

In this construction, each authorized user $\mathrm{AU}_i \in \mathcal{U}_i$ delegates her signing authority by sharing her membership key mk_i to a certain number of unauthorized users $u_{ij} \in \mathcal{U}_i$, including at least one trusted user. For simplicity, we assume that exactly one trusted user exists in each compartment. Note that one can also consider a group of trusted users. The authorized user $\mathrm{AU}_i \in \mathcal{U}_i$ uses Shamir's SSS in a nested way as described in Figure 3. Without loss of generality, we assume that $u_{i1} \in \mathcal{U}_i$ is the trusted user in the i-th compartment.

The authorized user $\mathrm{AU}_i \in \mathcal{U}_i$ chooses two polynomials and distributes the shares as follows:

1. Chooses $f_{i1}(x) = a_i x + mk_i$ for a random secret $a_i \in \mathbb{Z}_q$.
2. Sends $f_{i1}(1)$ to the trusted user u_{i1}.
3. Chooses $f_{i2}(x) = a^{(i)}_{t_i-2} x^{t_i-2} + \ldots + a^{(i)}_1 x_1 + f_{i1}(2)$, where $a^{(i)}_k \in \mathbb{Z}_q$ for $k = 1, \ldots, t_i - 2$.
4. Sends $f_{i2}(u_{ij})$ to the user u_{ij} for $j = 2, \ldots, k_i$.

With the first polynomial evaluation, the authorized user AU_i shares her membership key mk_i via $(2, 2)$-Shamir's SSS and sends one of the shares to the trusted user. For the second share, she applies one more $(t_i - 1, k_i - 1)$-Shamir's SSS, where t_i is the threshold, and k_i is the number of users in compartment $\mathcal{U}_i$.

The unauthorized users $u_{ij} \in \mathcal{G}$ for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, k_i$ sign as follows.

1. **Key Generation:** Each user $u_{ij} \in \mathcal{G}$ picks uniformly at random a secret key $sk_{ij} \xleftarrow{\$} \mathbb{Z}_q$, and computes her public key $pk_{ij} \leftarrow g_2^{sk_{ij}}$, where g_2 is a generator of $\mathbb{G}_2$.
2. **Compartment Setup:** Each authorized user AU_i shares her membership key mk_i via Shamir's SSS [33] (first $(2, 2)$-Shamir's SSS, then $(t_i - 1, k_i - 1)$-Shamir's SSS) as shown in Figure 3. At the end of this nested secret sharing procedure, each user in $\mathcal{U}_i$ obtains a compartment membership key cmk_{ij}.
3. **Signature Generation:** Each user $u_{ij} \in \mathcal{U}_i$ computes his individual signature on the message M as $s_{ij} = \mathrm{H}(M)^{sk_{ij}+cmk_{ij}}$, and sends it to the combiner.
4. **Signature Aggregation:** After receiving the individual signatures from the users, the combiner

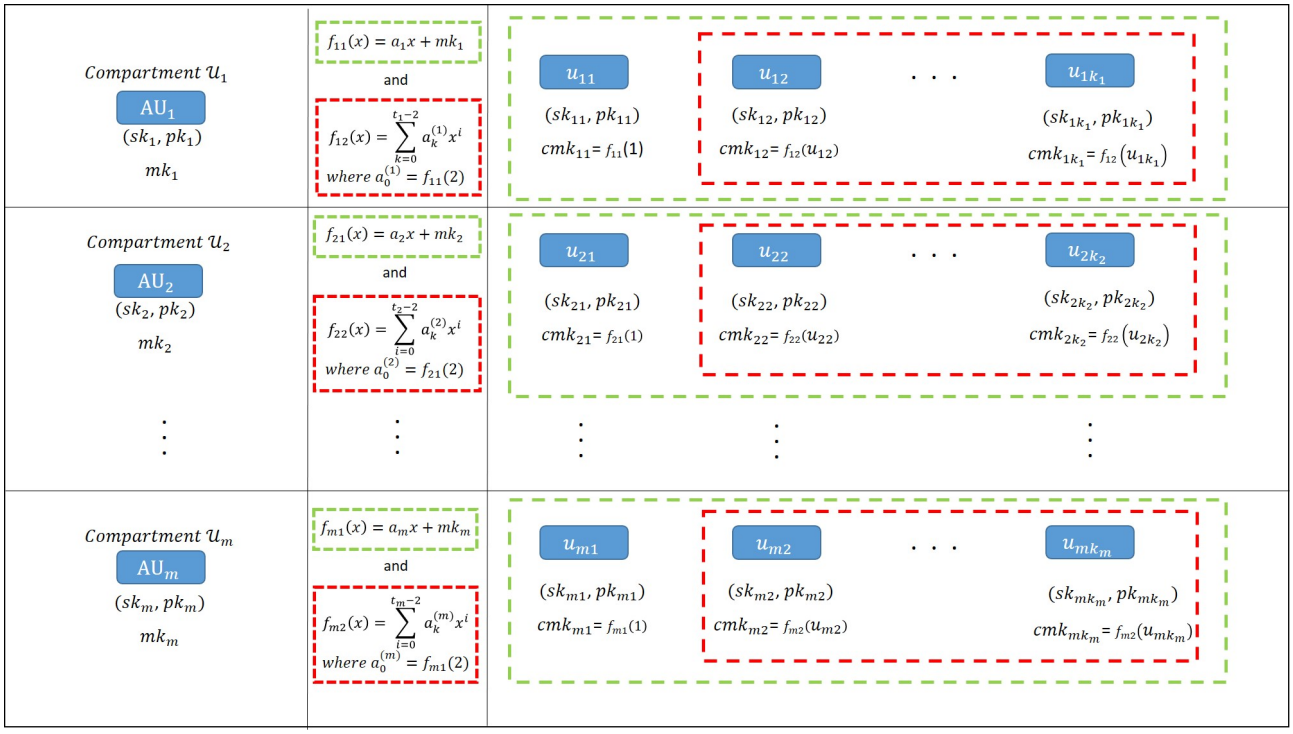

Figure 3: Trusted User Based Delegation

(a) forms the subgroup of signers $\mathcal{S} := \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$, where $\mathcal{S}_i$ is the subgroup of signers from the compartment $\mathcal{U}_i$, for $i = 1, \ldots, m$.

(b) computes $\sigma = \prod_{i=1}^{m} s_{i1}^{\gamma_{i1}} \big(\prod_{\substack{j \in \mathcal{S}_i \\ j \neq 1}} s_{ij}^{\lambda_{ij}} \big)^{\gamma_{i2}}$, where γ_{i1}, γ_{i2} and λ_{ij} are the appropriate Lagrange coefficients as described in Definition 2.7, for the first and the second secret sharing scheme, respectively.

5. **Verification:** If the thresholds are satisfied and the trusted users participate in the signing, then any verifier given $(par, \mathrm{MPK}, \mathcal{PK}, \mathcal{S}, M, \sigma)$ can verify the signature by checking the below equation

$$e\left(\mathrm{H}(M), \prod_{i=1}^{m} mpk_i \cdot pk_{i1}^{\gamma_{i1}} \big(\prod_{\substack{j \in \mathcal{S}_i \\ j \neq 1}} pk_{ij}^{\lambda_{ij}} \big)^{\gamma_{i2}}\right) \stackrel{?}{=} e(\sigma, g_2)$$

The verification equation satisfies correctness as given below:

$$
\begin{aligned}
e(\sigma, g_2) &= e\Bigg(\prod_{i=1}^{m} s_{i1}^{\gamma_{i1}} \big(\prod_{\substack{j\in\mathcal{S}_i \\ j\neq 1}} s_{ij}^{\lambda_{ij}}\big)^{\gamma_{i2}}, g_2\Bigg) \\
&= e\Bigg(\mathrm{H}(M)^{\sum_{i=1}^{m}\gamma_{i1}sk_{i1}+\gamma_{i1}cmk_{i1}+\sum_{i=1}^{m}\gamma_{i2}(\sum_{\substack{j\in\mathcal{S}_i \\ j\neq 1}}\lambda_{ij}sk_{ij}+\lambda_{ij}cmk_{ij})}, g_2\Bigg) \\
&= e\Bigg(\mathrm{H}(M), g_2^{\sum_{i=1}^{m}\gamma_{i1}sk_{i1}+\gamma_{i1}cmk_{i1}+\sum_{i=1}^{m}\gamma_{i2}(\sum_{\substack{j\in\mathcal{S}_i \\ j\neq 1}}\lambda_{ij}sk_{ij}+\lambda_{ij}cmk_{ij})}\Bigg) \\
&= e\Bigg(\mathrm{H}(M), g_2^{\sum_{i=1}^{m}\gamma_{i1}sk_{i1}+\sum_{i=1}^{m}\gamma_{i2}(\sum_{\substack{j\in\mathcal{S}_i \\ j\neq 1}}\lambda_{ij}sk_{ij})+\sum_{i=1}^{m}\gamma_{i1}cmk_{i1}+\sum_{i=1}^{m}\gamma_{i2}(\sum_{\substack{j\in\mathcal{S}_i \\ j\neq 1}}\lambda_{ij}cmk_{ij})}\Bigg) \\
&= e\Bigg(\mathrm{H}(M), g_2^{\sum_{i=1}^{m}\gamma_{i1}sk_{i1}+\sum_{i=1}^{m}\gamma_{i2}(\sum_{\substack{j\in\mathcal{S}_i \\ j\neq 1}}\lambda_{ij}sk_{ij})+\sum_{i=1}^{m}mk_i}\Bigg) \\
&= e\Bigg(\mathrm{H}(M), \prod_{i=1}^{m} mpk_i \cdot pk_{i1}^{\gamma_{i1}} \big(\prod_{\substack{j\in\mathcal{S}_i \\ j\neq 1}} pk_{ij}^{\lambda_{ij}}\big)^{\gamma_{i2}}\Bigg)
\end{aligned}
$$

Security. Security of this construction similarly follows from the security discussion of the previous construction. In this scenario, like in the previous one, each user u_{ij} signs a message M with the sum of his secret and membership keys, i.e. $sk_{ij}+cmk_{ij}$. The resulting individual signature is $s_{ij} = H(M)^{sk_{ij}+cmk_{ij}}$ which can also be seen as a BLS signature [7] on the message M under the key $sk_{ij}+cmk_{ij}$. Because sk_{ij} is sampled uniformly at random, and cmk_{ij} is computed via the SSS scheme, the signing key $sk_{ij}+cmk_{ij}$ will be uniformly random. Moreover, computing membership keys are done in a nested way. Namely, the membership key mk_i of the authorized user AU_i is partitioned into two parts. The first one is given to the trusted unauthorized user, and the other one is shared again among the other untrusted unauthorized users. Assume for a moment that the untrusted unauthorized users are corrupted, and they gather their membership keys cmk_{ij}. Even in this case, since the trusted user has the other half, corrupted users cannot forge the membership key mk_i of the authorized user AU_i.

We can assign a group of trusted users instead of a single one, and instead of using SSS recursively, we can use a hierarchical threshold secret sharing scheme for the same purpose.

3.4 Hierarchical Threshold Secret Sharing Scheme Based Delegation

Consider an organization with m compartments $\mathcal{G} = \bigcup_{i=1}^{m} \mathcal{U}_i$ and each compartment $\mathcal{U}_i$ has a hierarchical structure with r levels as shown in Figure 4. Assume that each authorized user $\mathrm{AU}_i \in \mathcal{U}_i$ delegates her signing capability to the unauthorized users via the hierarchical threshold secret sharing (HTSS) scheme proposed in [18] as defined in Section 2.3.

The unauthorized users $u_{ij} \in \mathcal{G}$ for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, k_i$ sign as follows.

1. **Key Generation:** Each user $u_{ij} \in \mathcal{G}$ picks uniformly at random a secret key $sk_{ij} \xleftarrow{\$} \mathbb{Z}_q$, and computes her public key $pk_{ij} \leftarrow g_2^{sk_{ij}}$, where g_2 is a generator of $\mathbb{G}_2$.

2. **Compartment Setup:** Each AU_i shares her membership key mk_i via a HTSS scheme [18] as described in Figure 4. She sends compartment membership keys $cmk_{ij} = f_i^{(k_{l-1}+1)}(u_{ij})$ to each user $u_{ij} \in \mathcal{U}_i$, for $j = 1, \ldots, k_i$, and where $l = 1, \ldots, r$ is the level of the user u_{ij} belongs to.

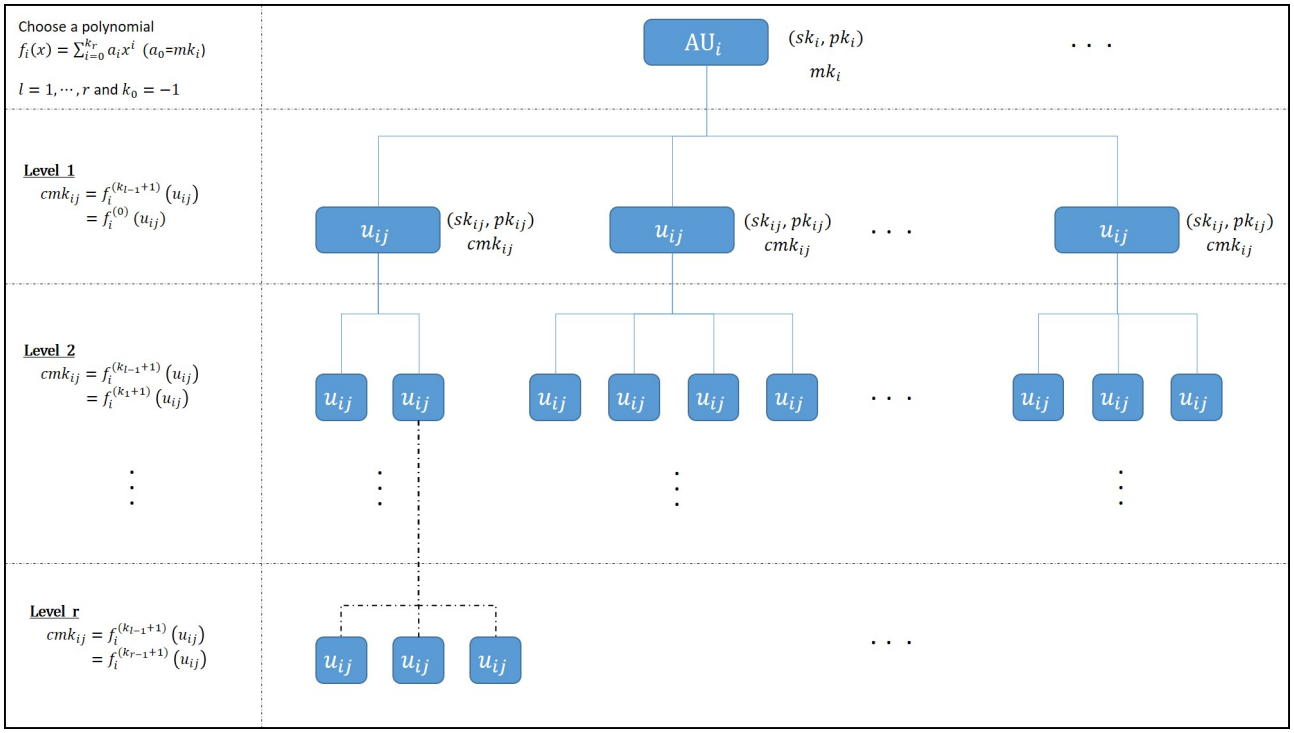

Figure 4: Hierarchical Threshold Secret Sharing Scheme Based Delegation

3. **Signature Generation:** Each user u_{ij} computes his individual signature on the message M as $s_{ij} = \mathrm{H}(M)^{sk_{ij}+cmk_{ij}}$, and sends it to the combiner.

4. **Signature aggregation:** After receiving the individual signatures from the users, the combiner

   (a) forms the subgroup of signers $\mathcal{S} := \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$, where $\mathcal{S}_i$ is the subgroup of signers from the compartment $\mathcal{U}_i$, for $i = 1, \ldots, m$.

   (b) computes $\sigma = \prod_{i=1}^{m} \prod_{j \in \mathcal{S}_i} s_{ij}^{\beta_{ij}}$, where β_{ij} is the appropriate Birkhoff coefficients as described in Definition 2.9.

5. **Verification:** If the level-specific thresholds are satisfied, then any verifier given $(par, \mathrm{MPK}, \mathcal{PK}, \mathcal{S}, M, \sigma)$ can verify the signature by checking the below equation

$$e\left(\mathrm{H}(M), \prod_{i=1}^{m} mpk_i \prod_{j \in \mathcal{S}_i} pk_{ij}^{\beta_{ij}}\right) \stackrel{?}{=} e(\sigma, g_2)$$

The verification equation satisfies correctness as given below:

$$\begin{aligned} e(\sigma, g_2) &= e(\prod_{i=1}^{m} \prod_{j \in \mathcal{S}_i} s_{ij}^{\beta_{ij}}, g_2) \\ &= e(\mathrm{H}(M)^{\sum_{i=1}^{m} \sum_{j \in \mathcal{S}_i} \beta_{ij} sk_{ij} + \beta_{ij} cmk_{ij}}, g_2) \\ &= e(\mathrm{H}(M), g_2^{\sum_{i=1}^{m} \sum_{j \in \mathcal{S}_i} \beta_{ij} sk_{ij} + \beta_{ij} cmk_{ij}}) \\ &= e(\mathrm{H}(M), g_2^{\sum_{i=1}^{m} \sum_{j \in \mathcal{S}_i} \beta_{ij} cmk_{ij}} g_2^{\sum_{i=1}^{m} \sum_{j \in \mathcal{S}_i} \beta_{ij} sk_{ij}}) \\ &= e(\mathrm{H}(M), \prod_{i=1}^{m} mpk_i \prod_{j \in \mathcal{S}_i} pk_{ij}^{\beta_{ij}}). \end{aligned}$$

Security. Like in the previous ones, each user u_{ij} signs a common message M with a combination of his secret and compartment membership keys, i.e. $sk_{ij}+cmk_{ij}$. Then the signature $s_{ij} = H(M)^{sk_{ij}+cmk_{ij}}$ can be also seen as a BLS signature [7] on the message M with the key $sk_{ij}+cmk_{ij}$. Because sk_{ij} is picked uniformly at random and the cmk_{ij} is computed in an HTSS scheme, the signing key $sk_{ij} + cmk_{ij}$ will also be random. Moreover, the authorized user AU_i chooses a secret polynomial such that the constant term is her membership key mk_i and shares it among unauthorized users (subordinates) according to their levels. The unauthorized users in the first level get their shares like in Shamir's SSS. Other users in the following levels get their shares from the polynomials' evaluations, which are some certain order derivatives of the first level polynomial. Since an order $d > 0$ derivative kills the constant term, the users in the lower levels cannot obtain the mk_i without cooperating with the first level (i.e. trusted) users. Therefore, while distributing the shares, the authorized user AU_i assumes that her trusted subordinates are in the first level.

Remark 3.3. In the original vASM scheme, the messages are signed by using only the membership keys. However, in our constructions, we assume that messages are signed using the sum of secret keys and compartment membership keys. In this way, we discard the natural anonymity of threshold secret sharing schemes; and ensure accountability.

Remark 3.4. Notice that the methods we give above, except Section 3.1, provide conditional security. If enough shareholders are corrupted, they can easily forge the membership key of the authorized user in their compartment. Since it is an accountable subgroup multi-signature, they can sign on behalf of the entire organization. Therefore one should be aware of the need for a trusted user/combiner in each construction except the one in Section 3.1.

4 Comparison

4.1 Comparison of the proposed constructions

We compare our proposed constructions with each other and give the number of operations in each phase of the proposed schemes in Table 1. The number of main operations, such that group operations and bilinear pairings are the same for Shamir's SSS-based, Trusted user-based, and HTSS-based constructions. The main difference between these three constructions emerges in computing Lagrange and Birkhoff coefficients. For Shamir's SSS-based and trusted user-based delegation, Lagrange coefficients should be

computed in the signature aggregation and in the verification phase by the designated combiner and the verifier, respectively. For HTSS-based delegation, Birkhoff coefficients should also be computed for the same phases. One needs to perform simple integer addition and multiplication to compute the Lagrange coefficients, whereas additional matrix operations should be conducted to compute the Birkhoff coefficients. Finally, since they all use threshold schemes, they all are secure when an adversary can corrupt up to t signers, where t is the threshold of the construction.

The number of operations required by the recursive vASM construction is less than the others. Only the compartment setup phase requires more operation than the others constructions. Since it is a one-time setup, it can be omitted. Moreover, the recursive vASM construction has a more robust assumption regarding the number of users an adversary can corrupt. Since the group setup is performed via an interactive (n, n)-VSS scheme, only one honest participant (possibly the authorized user or original signer) suffices to ensure the system's security.

Table 1: Comparison of the methods given in Section 3

Phases	Recursive vASM (Section 3.1)	Shamir's SSS based (Section 3.2)	Trusted user based (Section 3.3)	HTSS based (Section 3.4)
Key Generation	1 $\text{Exp}_{\mathbb{G}_2}$	1 $\text{Exp}_{\mathbb{G}_2}$	1 $\text{Exp}_{\mathbb{G}_2}$	1 $\text{Exp}_{\mathbb{G}_2}$
Compartment Setup	$m + 2k_i$ $\text{Exp}_{\mathbb{G}_2}$ $m + k_i(k_i + 1) - 2$ $\text{Mul}_{\mathbb{G}_2}$	(t_i, k_i)-SSS	$(2, 2)$-SSS $(t_i - 1, k_i - 1)$-SSS	HTSS with m levels
Signature Generation	1 Hash 1 $\text{Exp}_{\mathbb{G}_1}$	1 Hash 1 $\text{Exp}_{\mathbb{G}_1}$	1 Hash 1 $\text{Exp}_{\mathbb{G}_1}$	1 Hash 1 $\text{Exp}_{\mathbb{G}_1}$
Signature Aggregation	$k_i - 1$ $\text{Mul}_{\mathbb{G}_1}$	l $\text{Exp}_{\mathbb{G}_1}$ $l - 1$ $\text{Mul}_{\mathbb{G}_1}$ l Lagrange Coef.	l $\text{Exp}_{\mathbb{G}_1}$ $l - 1$ $\text{Mul}_{\mathbb{G}_1}$ $l + 1$ Lagrange Coef.	l $\text{Exp}_{\mathbb{G}_1}$ $l - 1$ $\text{Mul}_{\mathbb{G}_1}$ l Birkhoff Coef.
Verification	1 Hash $k_i - 1$ $\text{Mul}_{\mathbb{G}_2}$ 2 pairings	1 Hash $m + l$ $\text{Exp}_{\mathbb{G}_2}$ $m + l - 1$ $\text{Mul}_{\mathbb{G}_2}$ 2 pairings l Lagrange Coef.	1 Hash $m + l$ $\text{Exp}_{\mathbb{G}_2}$ $m + l - 1$ $\text{Mul}_{\mathbb{G}_2}$ 2 pairings $l + 1$ Lagrange Coef.	1 Hash $m + l$ $\text{Exp}_{\mathbb{G}_2}$ $m + l - 1$ $\text{Mul}_{\mathbb{G}_2}$ 2 pairings l Birkhoff Coef.

$\text{Exp}_{\mathbb{G}_i}$: Exponentiation in group $\mathbb{G}_i, i \in \{1, 2\}$
$\text{Mul}_{\mathbb{G}_i}$: Multiplication in group $\mathbb{G}_i, i \in \{1, 2\}$
k_i : Number of signers in $\mathcal{U}_i$, i.e. i-th compartment
m : Number of compartments in the organization $\mathcal{G}$
l : Number of all signers in the organization $\mathcal{G}$, i.e. $l = m \cdot k_i$

4.2 Comparison with the existing proxy signature schemes

In literature, many proxy signature schemes and variants of this notion have similar but not the same definitions. There are mainly three parties in all kinds of proxy signatures, i.e., the original signer, the proxy signer, and the verifier. Here is the high level description of existing proxy signatures. An original signer (or a set of original signers) generates and signs a warrant that contains detailed information about the delegation, such as proxy IDs, validity time, etc. Then the original signer sends this warrant signature to the proxy signer (or a set of proxy signers). The proxy signer signs on behalf of the original signer using this warrant signature and the corresponding public keys of the two sides. However, the verifier's job is the same as in standard signature schemes, with the difference that the warrant signature needs to be verified along with the signature on a message.

Boldyreva et al. formally defined the proxy signatures in [4] for the first time, and defined what functionalities a proxy signature scheme should provide to the users. We compare the functionalities of the formal definition of proxy signature that is given in Section 2.2 with our constructions.

1. **Proxy-designation protocol $(\mathcal{D}, \mathcal{P})$ functionality**: In the recursive vASM construction, instead of the protocols $\mathcal{D}$ and $\mathcal{P}$, we use a compartment setup in which all the original signers and the

proxy signers jointly participate in an interactive Feldman's VSS. In the end, proxy signers get their compartment membership keys cmk_{ij} as an analog of proxy signing keys skp. In the other constructions, we use secret sharing schemes in the compartment setup phases, and the outputs of these phases are the compartment membership keys used for proxy signing.

2. **Signing and verification functionalities:** Analogy between signing and verification functionalities are straightforward.

3. **Identification functionality:** All of our constructions have accountability properties because of the underlying signature scheme, i.e. vASM. The verifiers should know the identities of the proxy signers to verify the proxy signature properly.

Although the vASM scheme is an accountable subgroup multi-signature scheme, it provides the functionalities of a proxy signature with a flexible number of original and proxy signers. It can serve as several types of proxy signatures according to the number of original and proxy signers (n and l, respectively) as follows:

- **vASM signature scheme can serve as a proxy signature ($n = l = 1$).** To that end the original signer and the proxy signer generate their secret and public keys, participate in a group setup and obtain their membership keys. Then the proxy signer-using his membership key- can sign any message on behalf of the original signer in an accountable way. We compare vASM scheme with existing proxy signatures in Table 2. We choose the best -up to our knowledge- pairing-based proxy signature schemes for comparison. It can be seen from the table that the proxy key generation (group setup) of vASM scheme requires less operations than the existing ones which require at least one or more pairings and a number of elliptic curve additions, scalar multiplications and hash computations. In signature generation and the verification phases, vASM requires less operations than most of the schemes but not all. In terms of the signature size, vASM is better than all other schemes. The size of a vASM signature is just one group element whereas all others have more than one. Note that the actual efficiency depends on the system parameters (e.g., elliptic curve groups, hash functions, etc.) that are used to instantiate the schemes. Moreover, we also add the Schnorr-based scheme proposed by Boldyreva et al. [4] to the table. Although it is not a pairing-based one we add it in our table to make the readers able to compare the number of operations.

Table 2: Comparison with the existing proxy signature schemes

.

Schemes	Proxy key generation		Signature generation	Verification	Signature size
	Original signer	Proxy signer			
Boldyreva et al.$^{(*)}$ [4]	1 $Exp \pmod p$ 1 $H_{\mathbb{Z}_q}$	2 $Exp \pmod p$ 3 $H_{\mathbb{Z}_q}$ 1 $Mul \pmod q$	1 $Exp \pmod p$ 3 $H_{\mathbb{Z}_q}$	3 $Exp \pmod p$ 1 $H_{\mathbb{Z}_q}$ 3 $Mul \pmod q$	$3\lvert\mathbb{Z}_p\rvert + \lvert\mathbb{Z}_q\rvert + \lvert M_w\rvert$
Lee et al. [22]	1 $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$	1 $\mathcal{P}$ 1 $H_{\mathbb{G}}$	2 $\mathcal{P}$ 2 $Exp_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$ 1 $Exp_{\mathbb{G}_T}$	1 $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	$\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$
Shim [34]	1 $Exp_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$	2 $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{G}}$	1 $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$	2 $\mathcal{P}$ 2 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 3 $H_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}_T}$	$\lvert\mathbb{G}_T\rvert + \lvert M_w\rvert$
Zhang et al. [44]	3 $Exp_{\mathbb{G}}$ 2 $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{G}}$	4 $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ 2 $Mul_{\mathbb{G}}$ 4 $H_{\mathbb{G}}$ 2 $Mul_{\mathbb{G}_T}$	2 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{G}}$	5 $\mathcal{P}$ 2 $Mul_{\mathbb{G}}$ 5 $H_{\mathbb{G}}$ 3 $Mul_{\mathbb{G}_T}$	$3\lvert\mathbb{G}\rvert + \lvert M_w\rvert$
Seo et al. [32]	2 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	3 $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ 2 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}_T}$	2 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	4 $\mathcal{P}$ 2 $Exp_{\mathbb{G}}$ 3 $Mul_{\mathbb{G}}$ 4 $H_{\mathbb{G}}$ 2 $H_{\mathbb{Z}_q}$ 2 $Mul_{\mathbb{G}_T}$	$3\lvert\mathbb{G}\rvert + \lvert M_w\rvert$
Verma & Singh [39]	1 $Exp_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$	2 $\mathcal{P}$ 1 $H_{\mathbb{G}}$	1 $Exp_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	2 $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	$\lvert\mathbb{G}\rvert + \lvert M_w\rvert$
vASM [2]	4 $Exp_{\mathbb{G}_2}$ 4 $Mul_{\mathbb{G}_2}$		1 $H_{\mathbb{G}_1}$ 1 $Exp_{\mathbb{G}_1}$	2 $\mathcal{P}$ 1 $Mul_{\mathbb{G}_2}$ 1 $H_{\mathbb{G}_1}$	$\lvert\mathbb{G}_1\rvert$

Note that since the group operations determine the efficiency numbers we ignore the integer operations.

(*) Notice that all the schemes except [4] are pairing-based schemes.

$\mathcal{P}$**:** Bilinear pairing operation.

Exp_A**:** Exponentiation in group A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$.

Mul_A**:** Multiplication in group A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$.

H_A**:** Hash onto the set A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{Z}_q\}$.

$|M_w|$: Size of the warrant in bits.

$|A|$: Size of the elements of the set A, where $A \in \{G, G_1\}$

- **vASM signature scheme can serve as a proxy multi-signature ($n > 1$ and $l = 1$).** n original signers and the proxy signer jointly participate in the group setup of vASM scheme. This way, each original signer authorizes the proxy signer to sign on behalf of himself. We give an efficiency comparison of vASM and the existing pairing-based proxy multi-signature schemes in Table 3. It can be easily seen that vASM scheme is much more efficient than the existing pairing-based proxy multi-signature schemes in terms of all comparison parameters. vASM does

not require pairing operations for proxy key generation phases, whereas the others require many pairings. vASM has also better computational efficiency than the others in terms of signature generation and verification phases. The signature size of vASM is only one group element, while others result in more.

Table 3: Comparison with the existing proxy multi-signature schemes

Schemes	**Proxy key generation**		**Signature generation**	**Verification**	**Signature size**
	Original signer	**Proxy signer**			
Li & Chen [24]	3 $Exp_{\mathbb{G}}$ n $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	$3n$ $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ n $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$ n $Exp_{\mathbb{G}_T}$ n $Mul_{\mathbb{G}_T}$	1 $\mathcal{P}$ 2 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$ 1 $Exp_{\mathbb{G}_T}$	3 $\mathcal{P}$ n $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{Z}_q}$ 2 $Exp_{\mathbb{G}_T}$ 2 $Mul_{\mathbb{G}_T}$	$2\|\mathbb{G}\| + \|\mathbb{Z}_q\| + \|M_w\|$
Du & Wen [10]	3 $Exp_{\mathbb{G}}$ $(n+1)$ $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{G}}$	$4n$ $\mathcal{P}$ $(2n-1)$ $Mul_{\mathbb{G}}$ $2n$ $H_{\mathbb{G}}$ $2n$ $Mul_{\mathbb{G}_T}$	3 $Exp_{\mathbb{G}}$ 3 $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{G}}$	6 $\mathcal{P}$ $(2n-1)$ $Mul_{\mathbb{G}}$ 4 $H_{\mathbb{G}}$ 4 $Mul_{\mathbb{G}_T}$	$3\|\mathbb{G}\| + \|M_w\|$
vASM [2]	$2n$ $Exp_{\mathbb{G}_2}$ (n^2+n-2) $Mul_{\mathbb{G}_2}$		1 $H_{\mathbb{G}_1}$ 1 $Exp_{\mathbb{G}_1}$	2 $\mathcal{P}$ $(n-1)$ $Mul_{\mathbb{G}_2}$ 1 $H_{\mathbb{G}_1}$	$\|\mathbb{G}_1\|$

.

Note that since the group operations determine the efficiency numbers we ignore the integer operations.

$\mathcal{P}$**:** Bilinear pairing operation.

Exp_A**:** Exponentiation in group A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$.

Mul_A**:** Multiplication in group A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$.

H_A**:** Hash onto the set A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{Z}_q\}$.

n : Number of original signers.

$|M_w|$: Size of the warrant in bits.

$|A|$: Size of the elements of the set A, where $A \in \{G, G_1\}$

- **vASM signature scheme can serve as a multi-proxy signature ($n = 1$ and $l > 1$).** If an original signer participates in a group setup procedure with l proxy signers, he authorizes the proxy signers by sharing his own secret key in the group setup (for details see Section 2.5). In Table 4, we compare the vASM scheme with existing pairing-based multi-proxy signature schemes. In general, vASM scheme requires less number of operations that are hard to compute (e.g. bilinear pairings). But the other schemes may have better computational efficiency as the number of proxies increase. For example, for a very large l, the cost of proxy key generation (group setup) of vASM requires more time than a few pairings which are required by the other schemes in the same phase. But for other comparison parameters, i.e. signature generation, aggregation, verification and signature size, vASM has better efficiency numbers than the other existing pairing-based multi-proxy signature schemes.

- **vASM signature scheme can serve as a multi-proxy multi-signature ($n > 1$ and $l > 1$).** To this end, n original signers and l proxy signers jointly perform a group setup. As a result of this group setup, each proxy signer has a membership key, which can be used as a proxy key. Any number of proxy signers can sign on behalf of the original signers using these membership keys. We

Table 4: Comparison with the existing multi-proxy signature schemes

Scheme	Proxy key generation		Signature generation	Aggregation	Verification	Signature size
	Original signer	Proxy signer				
Li & Chen [24]	3 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	3 $\mathcal{P}$ 2 $Exp_{\mathbb{G}}$ 3 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$ 1 $Exp_{\mathbb{G}_T}$ 1 $Mul_{\mathbb{G}_T}$	3 $Exp_{\mathbb{G}}$ l $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	$3l$ $\mathcal{P}$ $(l-1)$ $Mul_{\mathbb{G}}$ l $H_{\mathbb{Z}_q}$ l $Exp_{\mathbb{G}_T}$ l $Mul_{\mathbb{G}_T}$	3 $\mathcal{P}$ l $Exp_{\mathbb{G}}$ $(3l-1)$ $Mul_{\mathbb{G}}$ $(l+1)$ $H_{\mathbb{G}}$ 2 $H_{\mathbb{Z}_q}$ 1 $Exp_{\mathbb{G}_T}$	$3\lvert\mathbb{G}\rvert + \lvert M_w\rvert$
Cao & Cao [8]	2 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$	3 $\mathcal{P}$ 1 $Exp_{\mathbb{G}}$ 1 $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$	2 $Exp_{\mathbb{G}}$ l $Mul_{\mathbb{G}}$ 1 $H_{\mathbb{G}}$	5 $\mathcal{P}$ $(l-1)$ $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$ 1 $Exp_{\mathbb{G}_T}$ 3 $Mul_{\mathbb{G}_T}$	5 $\mathcal{P}$ 3 $Exp_{\mathbb{G}}$ l $Mul_{\mathbb{G}}$ 2 $H_{\mathbb{G}}$ 1 $H_{\mathbb{Z}_q}$ 3 $Mul_{\mathbb{G}_T}$	$3\lvert\mathbb{G}\rvert + \lvert M_w\rvert$
Liu et al. [26]	7 $Exp_{\mathbb{G}}$ $2\lvert M_w\rvert$ $Mul_{\mathbb{G}}$	2 $\mathcal{P}$ $\lvert M_w\rvert$ $Mul_{\mathbb{G}}$ 2 $Mul_{\mathbb{G}_T}$	3 $\mathcal{P}$ 5 $Exp_{\mathbb{G}}$ $(2\lvert M_w\rvert + 7)$ $Mul_{\mathbb{G}}$ 3 $Mul_{\mathbb{G}_T}$	$3l$ $\mathcal{P}$ $(4l-2)$ $Mul_{\mathbb{G}}$ 3 $Mul_{\mathbb{G}_T}$	3 $\mathcal{P}$ $2\lvert M_w\rvert$ $Mul_{\mathbb{G}}$ $(l+2)$ $Mul_{\mathbb{G}_T}$	$3\lvert\mathbb{G}\rvert + \lvert M_w\rvert$
vASM [2]	$2l$ $Exp_{\mathbb{G}_2}$ (l^2+l-2) $Mul_{\mathbb{G}_2}$		1 $H_{\mathbb{G}_1}$ 1 $Exp_{\mathbb{G}_1}$	$(l-1)$ $Mul_{\mathbb{G}_1}$	2 $\mathcal{P}$ $(l-1)$ $Mul_{\mathbb{G}_2}$ 1 $H_{\mathbb{G}_1}$	$\lvert\mathbb{G}_1\rvert$

Note that since the group operations determine the efficiency numbers we ignore the integer operations.
$\mathcal{P}$**:** Bilinear pairing operation.
Exp_A**:** Exponentiation in group A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$.
Mul_A**:** Multiplication in group A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$.
H_A**:** Hash onto the set A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{Z}_q\}$.
l : Number of proxy signers
$|M_w|$: Size of the warrant in bits.
$|A|$: Size of the elements of the set A, where $A \in \{G, G_1\}$

.

compare the vASM scheme with the existing pairing-based multi-proxy multi-signature schemes in Table 5. vASM scheme has slightly better efficiency numbers than the other schemes.

Remark 4.1. Note that vASM scheme provides only a general delegation of signing rights. Since there is no warrant-like information the delegation is valid until it is canceled (by invalidating the commitment set COM.).

Remark 4.2. One important property of vASM is that it has aggregation property. For N distinct vASM signatures, one can aggregate all signatures into one which is a single $\mathbb{G}_1$ element. In this case, verification of the aggregated vASM signature requires $N + 1$ pairing operations instead of $3N + 1$. Further information about aggregation of vASM signatures can be found in [2].

5 Conclusion

In this work, we propose four constructions of compartment-based and hierarchical delegation of vASM signing authority for different organizational scenarios. We show that applying the vASM scheme recursively and combining the functionalities of threshold secret sharing schemes with the vASM scheme can solve an organizational problem of delegating the signing power of authorized users to single/multiple proxies in an accountable fashion. We present the comparison of our constructions with the existing proxy signatures in the literature in terms of their properties and functionalities. Our constructions provide us

Table 5: Comparison with the existing multi-proxy multi-signature schemes

Scheme	Proxy key generation			Signature generation	Aggregation	Verification	Signature size
	Original signer	Proxy signer	Chairman				
Li & Chen [24]	$3\ Exp_{\mathbb{G}}$ $(n+l)\ Mul_G$ $1\ H_{\mathbb{Z}_q}$	$3\ Exp_{\mathbb{G}}$ $(n+l)\ Mul_G$ $1\ H_{\mathbb{Z}_q}$	$3(n+l)\ \mathcal{P}$ $(n+l-1)\ Mul_G$ $1\ H_{\mathbb{Z}_q}$ $(n+l)\ Exp_{\mathbb{G}_T}$ $(n+l)\ Mul_{\mathbb{G}_T}$	$3\ Exp_{\mathbb{G}}$ $l\ Mul_G$ $1\ H_{\mathbb{Z}_q}$	$3l\ \mathcal{P}$ $(l-1)\ Mul_G$ $l\ H_{\mathbb{Z}_q}$ $l\ Exp_{\mathbb{G}_T}$ $l\ Mul_{\mathbb{G}_T}$	$6\ \mathcal{P}$ $(n+2l-2)\ Mul_G$ $2\ H_{\mathbb{Z}_q}$ $2\ Exp_{\mathbb{G}_T}$ $2\ Mul_{\mathbb{G}_T}$	$4\|\mathbb{G}\|+\|M_w\|$
Sahu & Padhye [31]	$2\ Exp_{\mathbb{G}}$ $(n-1)\ Mul_{\mathbb{G}}$ $1\ H_{\mathbb{Z}_q}$	$2n\ \mathcal{P}$ $(n+l)\ Exp_{\mathbb{G}}$ $(n+l-1)\ Mul_{\mathbb{G}}$ $(n+1)\ H_{\mathbb{Z}_q}$	NA	$1\ \mathcal{P}$ $2\ Exp_{\mathbb{G}}$ $1\ Mul_{\mathbb{G}}$ $1\ H_{\mathbb{Z}_q}$ $1\ Exp_{\mathbb{G}_T}$ $(l-1)\ Mul_{\mathbb{G}_T}$	$2\ \mathcal{P}$ $l\ Exp_{\mathbb{G}}$ $(nl+l-1)\ Mul_{\mathbb{G}}$ $l\ H_{\mathbb{Z}_q}$ $l\ Exp_{\mathbb{G}_T}$ $l\ Mul_{\mathbb{G}_T}$	$2\ \mathcal{P}$ $2\ Exp_{\mathbb{G}}$ $(nl-1)\ Mul_{\mathbb{G}}$ $2\ H_{\mathbb{Z}_q}$ $1\ Exp_{\mathbb{G}_T}$ $1\ Mul_{\mathbb{G}_T}$	$3\|\mathbb{G}\|+\|M_w\|$
vASM [2]	$2(n+l)\ Exp_{\mathbb{G}_2}$ $((n+l)^2+(n+l)-2)\ Mul_{\mathbb{G}_2}$			$1\ H_{\mathbb{G}_1}$ $1\ Exp_{\mathbb{G}_1}$	$(l-1)\ Mul_{\mathbb{G}_1}$	$2\ \mathcal{P}$ $(n+l-1)\ Mul_{\mathbb{G}_2}$ $1\ H_{\mathbb{G}_1}$	$\|\mathbb{G}_1\|$

Note that since the group operations determine the efficiency numbers we ignore the integer operations.
$\mathcal{P}$: Bilinear pairing operation.
Exp_A: Exponentiation in group A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$.
Mul_A: Multiplication in group A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}$.
H_A: Hash onto the set A, where $A \in \{\mathbb{G}, \mathbb{G}_1, \mathbb{Z}_q\}$.
n : Number of original signers.
l : Number of proxy signers
$|M_w|$: Size of the warrant in bits. $|A|$: Size of the elements of the set A, where $A \in \{G, G_1\}$

with all the functionalities of the proxy signatures that are defined in [4]. We also compared our constructions with each other according to the number of computations required and the security assumptions. Shamir's SSS-based and trusted user-based delegations require nearly the same number of computations, whereas HTSS-based delegation requires more operations than the others because of the determinant operations performed in signature aggregation and verification phases. On the other hand, all three constructions are insecure in case of an adversary that can corrupt $t+1$ signers, where t is the threshold of the secret sharing schemes used in the constructions. However, our proposed recursive vASM construction provides a much more efficient solution for delegating the vASM signing authority. Moreover, it is more secure than the other constructions based on the threshold secret sharing schemes. Existence of at least one honest participant during the one-time compartment setup is enough to ensure the security of recursive vASM construction. Since the authorized signer (original signer/designator/delegator) also participates in the compartment setup, even though all the unauthorized users are corrupted, they cannot reconstruct the membership key of the authorized user. In contrast, our constructions using threshold secret sharing schemes require at least one trusted unauthorized user. Finally we compare the vASM scheme with the existing pairing-based proxy signature variants, i.e. proxy signatures, proxy multi-signatures, multi-proxy signatures, multi-proxy multi-signatures, in terms of efficiency. vASM scheme can be used as a flexible and practical proxy signature scheme because of its simple structure. On the other hand it provides a general delegation, i.e. it doesn't have a warrant that gives detailed information about the delegation, such as validity time, delegation context, etc.

References

[1] Alomair, B., Sampigethaya, K., Poovendran, R.: Efficient generic forward-secure signatures and proxy signatures. In: Public Key Infrastructure: 5th European PKI Workshop: Theory and Practice, EuroPKI 2008 Trondheim, Norway, June 16-17, 2008 Proceedings 5. pp. 166–181. Springer (2008)

[2] Ağırtaş, A.R., Yayla, O.: Pairing-based accountable subgroup multi-signatures with verifiable group setup. Cryptology ePrint Archive, Report 2022/018 (2022), https://ia.cr/2022/018

[3] Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In: Public Key Cryptography—PKC 2003: 6th International Workshop on Practice and Theory in Public Key Cryptography Miami, FL, USA, January 6–8, 2003 Proceedings 6. pp. 31–46. Springer (2002)

[4] Boldyreva, A., Palacio, A., Warinschi, B.: Secure proxy signature schemes for delegation of signing rights. Journal of Cryptology **25**, 57–115 (2012)

[5] Boneh, D., Drijvers, M., Neven, G.: Compact multi-signatures for smaller blockchains. In: Peyrin, T., Galbraith, S. (eds.) Advances in Cryptology – ASIACRYPT 2018. pp. 435–464. Springer International Publishing, Cham (2018)

[6] Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and verifiably encrypted signatures from bilinear maps. In: Biham, E. (ed.) Advances in Cryptology — EUROCRYPT 2003. pp. 416–432. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)

[7] Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. J. Cryptol. **17**(4), 297–319 (Sep 2004)

[8] Cao, F., Cao, Z.: A secure identity-based multi-proxy signature scheme. Computers & Electrical Engineering **35**(1), 86–95 (2009)

[9] Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) Advances in Cryptology – CRYPTO'89 Proceedings. pp. 307–315. Springer New York, New York, NY (1990)

[10] Du, H., Wen, Q.: Certificateless proxy multi-signature. Information Sciences **276**, 21–30 (2014)

[11] Eslami, Z., Pakniat, N., Nojoumian, M.: Ideal social secret sharing using birkhoff interpolation method. Security and Communication Networks **9** (10 2016)

[12] Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: 28th Annual Symposium on Foundations of Computer Science (sfcs 1987). pp. 427–438 (Oct 1987)

[13] Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Robust threshold dss signatures. In: Maurer, U. (ed.) Advances in Cryptology — EUROCRYPT '96. pp. 354–371. Springer Berlin Heidelberg, Berlin, Heidelberg (1996)

[14] Harn, L.: Group-oriented (t, n) threshold digital signature scheme and digital multisignature. IEE Proceedings - Computers and Digital Techniques **141**, 307–313(6) (September 1994)

[15] Herranz, J., Saez, G.: Revisiting fully distributed proxy signature schemes. Cryptology ePrint Archive, Report 2003/197 (2003), https://ia.cr/2003/197

[16] Hwang, S.J., Chiu-Chin, C.: New multi-proxy multi-signature schemes. Applied Mathematics and Computation **147** (01 2004)

[17] Hwang, S.J., Shi, C.: A simple multi-proxy signature scheme (01 2000)

[18] Käsper, E., Nikov, V., Nikova, S.: Strongly multiplicative hierarchical threshold secret sharing. In: Desmedt, Y. (ed.) Information Theoretic Security. pp. 148–168. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)

[19] Kim, S., Park, S., Won, D.: Proxy signatures, revisited. In: Information and Communications Security: First International Conference, ICIS'97 Beijing, China, November 11–14, 1997 Proceedings 1. pp. 223–232. Springer (1997)

[20] Lee, B., Kim, H., Kim, K.: Secure mobile agent using strong non-designated proxy signature. In: Information Security and Privacy: 6th Australasian Conference, ACISP 2001 Sydney, Australia, July 11–13, 2001 Proceedings 6. pp. 474–486. Springer (2001)

[21] Lee, B., Kim, H., Kim, K.: Strong proxy signature and its applications. In: Proceedings of SCIS. vol. 2001, pp. 603–608 (2001)

[22] Lee, J.S., Chang, J.H., Lee, D.H.: Forgery attacks on kang et al.'s identity-based strong designated verifier signature scheme and its improvement with security proof. Comput. Electr. Eng. **36**(5), 948–954 (sep 2010)

[23] Li, J., Yuen, T.H., Chen, X., Wang, Y.: Proxy ring signature: Formal definitions, efficient construction and new variant. In: 2006 International Conference on Computational Intelligence and Security. vol. 2, pp. 1259–1264 (2006)

[24] Li, X., Chen, K.: Id-based multi-proxy signature, proxy multi-signature and multi-proxy multi-signature schemes from bilinear pairings. Appl. Math. Comput. **169**(1), 437–450 (oct 2005)

[25] Lin, W., Jan, J.K.: Security personal learning tools using a proxy blind signature scheme. Journal of Chinese Language and Computing (01 2000)

[26] Liu, Z., Hu, Y., Zhang, X., Ma, H.: Provably secure multi-proxy signature scheme with revocation in the standard model. Computer Communications **34**(3), 494–501 (2011), special Issue of Computer Communications on Information and Future Communication Security

[27] Malkin, T., Obana, S., Yung, M.: The hierarchy of key evolving signatures and a characterization of proxy signatures. vol. 3027, pp. 306–322 (05 2004)

[28] Mambo, M., Usuda, K., Okamoto, E.: Proxy signatures for delegating signing operation. In: Proceedings of the 3rd ACM Conference on Computer and Communications Security. p. 48–57. CCS '96, Association for Computing Machinery, New York, NY, USA (1996)

[29] Micali, S., Ohta, K., Reyzin, L.: Accountable-subgroup multisignatures: Extended abstract. In: Proceedings of the 8th ACM Conference on Computer and Communications Security. p. 245–254. CCS '01, Association for Computing Machinery, New York, NY, USA (2001)

[30] Okamoto, T., Tada, M., Okamoto, E.: Extended proxy signatures for smart cards. In: Information Security. pp. 247–258. Springer Berlin Heidelberg, Berlin, Heidelberg (1999)

[31] Sahu, R.A., Padhye, S.: Identity-based multi-proxy multi-signature scheme provably secure in random oracle model. Transactions on Emerging Telecommunications Technologies **26**(4), 547–558 (2015)

[32] Seo, S., Choi, K., Hwang, J., Kim, S.: Efficient certificateless proxy signature scheme with provable security. Information Sciences **188**, 322–337 (2012)

[33] Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (Nov 1979)

[34] Shim, K.A.: Short designated verifier proxy signatures. Computers & Electrical Engineering **37**(2), 180–186 (2011), modern Trends in Applied Security: Architectures, Implementations and Applications

[35] Sun, H.M.: Design of time-stamped proxy signatures with traceable receivers. IEE Proceedings-Computers and Digital Techniques **147**(6), 462–466 (2000)

[36] Tassa, T.: Hierarchical threshold secret sharing. In: Naor, M. (ed.) Theory of Cryptography. pp. 473–490. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)

[37] Tassa, T., Dyn, N.: Multipartite secret sharing by bivariate interpolation. Journal of Cryptology **22**(2), 227–258 (Apr 2009)

[38] Tompa, M., Woll, H.: How to share a secret with cheaters. Journal of Cryptology **1**(3), 133–138 (Oct 1989)

[39] Verma, G.K., Singh, B.B.: Short certificate-based proxy signature scheme from pairings. Transactions on Emerging Telecommunications Technologies **28**(12), e3214 (2017), e3214 ett.3214

[40] Wu, C.K., Varadharajan, V.: Modified Chinese Remainder Theorem and its application to proxy signatures. In: Proceedings of the 1999 ICPP Workshops on Collaboration and Mobile Computing (CMC'99). Group Communications (IWGC). Internet '99 (IWI'99). Industrial Applications on Network Computing (INDAP). Multime. pp. 146–151 (1999)

[41] Yi, L., Bai, G., Xiao, G.: Proxy multi-signature scheme: A new type of proxy signature scheme. Electronics Letters **36**, 527 – 528 (04 2000)

[42] Zhang, F., Safavi-Naini, R., Lin, C.Y.: New proxy signature, proxy blind signature and proxy ring signature schemes from bilinear pairing. Cryptology ePrint Archive, Report 2003/104 (2003), `https://ia.cr/2003/104`

[43] Zhang, K.: Threshold proxy signature schemes. In: International Workshop on Information Security. pp. 282–290. Springer (1997)

[44] Zhang, L., Zhang, F., Wu, Q.: Delegation of signing rights using certificateless proxy signatures. Inf. Sci. **184**(1), 298–309 (2012)