MASTER'S THESIS

Master of Science in Advanced Mathematics and Mathematical Engineering

Title: Anonymous threshold signatures

Author: Petar Hlad Colic

Advisor: Javier Herranz Sotoca

Department: Department of Mathematics

Academic year: 2017-2018

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Facultat de Matemàtiques i Estadística
UNIVERSITAT POLITÈCNICA DE CATALUNYA

ANONYMOUS THRESHOLD SIGNATURES

A master's thesis
Submitted to the

Facultat de Matemàtiques i Estadística

Universitat Politècnica de Catalunya

By

Petar Hlad Colic

In partial fulfillment
of the requirements for the master's degree in

Advanced Mathematics and Mathematical Engineering

Advisor: Javier Herranz Sotoca

Barcelona, June 2018

Abstract

The goal of this work was to find an anoymous threshold signature scheme with compact signature. Although such scheme is not found in this work, few solutions published by other authors are reviewed and an improvement is proposed to obtain a scheme similar to the desired one computationaly expensive and interactive

Resum

Aquest treball tenia l'objectiu de trobar un esquema de llindar de signatura anònima compacte. Tot i no haver-ne trobat cap, s'analitzen diverses solucions que s'acosten a l'objectiu publicades per altres autors i es proposa una millora per obtenir un esquema com el desitjat costós i interactiu.

Resumen

Este trabajo tenía el objetivo de encontrar un esquema umbral de firma anónima compacta. Aún no haberla encontrado, se analizan diversas soluciones que se acercan al objetivo publicadas por otros autores y se propone una mejora para obtener un esquema como el deseado costoso e interactivo.

ACKNOWLEDGEMENTS

Special thanks to my advisor Javier, for his help and patience.
And to Ari, for her support.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

There are many practical applications for threshold signature schemes.

Case 1: There is a toll system that gives a discount on the price if there are at least three passengers in the vehicle. The passengers prove that they are at least three by running a threshold signature with their personal devices (e.g. their smart-phones).

Case 2: There is an e-voting system where each candidate needs a certain amount of signatures to get to the next round. This can be done with a threshold signature from the voters on the candidate's identifier.

Case 3: An advertising company pays to website holders for showing their ads. But they pay depending on the amount of distinct users that have seen (or clicked) the ad instead of the total amount of watches (or clicks). This can be done with a threshold signature from the users on the ad identifier.

We would want the schemes, in all three cases, to be anonymous. Also we would want them to be unlinkable for different messages. Otherwise, if the anonymity of a signer is compromised at some point, an attacker link all messages that the signer has signed. In case 1 this would result in the ability of tracking the geographic location of a signer, in case 2 the information of who the signer has voted for, and in case 3 the information on which sites visits the signer.

For case 1, we want the signatures to be compact and not excessively complex to compute, but the signing protocol could be interactive. For case 3, since the value of the threshold is large, we want the signature to be compact and the signing protocol should not be interactive.

In this work, we attempt to find a threshold signature scheme such that signatures are anonymous, unlinkable, compact and that does not require interaction between the signers.

In chapter 2 we introduce the cryptographic notions necessary to understand and review the solutions we explain.

In chapter 3 we discuss the concept of anonymity in threshold signature schemes to clear it up, since many authors use the concept with different meanings.

In chapter 4 we detail some solutions that are valid for only once because they do not have the non-traceability property, so the signature scheme needs to be setup newly every time a signature is computed.

In chapter 5 we detail three solutions that are anonymous and non-traceable:

– The first gives a compact signature, but depending on the setup not every set of participants of same size as the threshold will be able to compute a signature.

– The second gives a signature whose size grows linearly with the threshold.

– The third is an improvement we propose on the scheme detailed in 2.7.1 that is compact and any set of participants of same size as the threshold is able to compute a signature. The counterpart is that the protocol is interactive and requires a number of interactions between the signers that grows with the square of the threshold.

In chapter 6 we sum up the advantages and disadvantages that has every detailed solution and discuss the utility of each, and explain the conclusions of the work.

CHAPTER 2

PRELIMINARIES

Here we introduce some basic notions of cryptography and other fields that we will need to understand the schemes described in the subsequent chapters.

2.1 Homomorphic Public Key Encryption

A public key encryption scheme $PKE = (KG, \mathcal{E}, \mathcal{D})$ consists of three probabilistic and polynomial time algorithms:

- The key generation algorithm KG generates a pair (sk, pk) of secret and public keys on the input of a security parameter
- The encryption algorithm $\mathcal{E}$ takes as input a plaintext m and a public key pk, and outputs a ciphertext $c = \mathcal{E}_{pk}(m)$.
- The decryption algorithm $\mathcal{D}$ takes as input a ciphertext c and a secret key and outputs a plaintext $m = \mathcal{D}_{sk}(c)$

For any pair (sk, pk) of secret and public keys, and any plaintext m, it must hold $m = \mathcal{D}_{sk}(\mathcal{E}_{pk}(m))$.

A PKE scheme $(KG, \mathcal{E}, \mathcal{G})$ has an homomorphic property if there exist two operations, defined on the set of plaintexts and ciphertexts, respectively, such that the result of operating two ciphertexts is an encryption of the result of operating the two corresponding plaintexts.

Formally written: let $\mathcal{M}$ be the set of plaintexts and let $\bullet$ be an operation on $\mathcal{M}$ s.t. $m_1 \bullet m_2 \in \mathcal{M} \quad \forall m_1, m_2 \in \mathcal{M}$. Let $\mathcal{C}$ the set of ciphertexts and let $\circ$ be an operation on $\mathcal{C}$ s.t. $c_1 \circ c_2 \in \mathcal{C} \quad \forall c_1, c_2 \in \mathcal{C}$. $(KG, \mathcal{E}, \mathcal{D})$ is an homomorphic PKE if

$$\mathcal{D}_{sk}\big(\mathcal{E}_{pk}(m_1) \circ \mathcal{E}_{pk}(m_2)\big) \;=\; m_1 \bullet m_2 \quad \forall m_1, m_2 \in \mathcal{M}.$$

Write $\mathcal{M}$ additively and $\mathcal{C}$ multiplicatively. For $a \in \mathbb{Z}^+$, we have

$$\mathcal{D}_{sk}\left(\mathcal{E}_{pk}(m)^a\right) = a \cdot m$$

An example of an homomorphic PKE was developed by ElGamal in [ElG85].

2.2 Oblivious Polynomial Evaluation

Oblivious polynomial evaluation is a protocol involving two parties, a sender whose input is a polynomial $P \in \mathbb{F}[x]$, and a receiver whose input is a value $\alpha \in \mathbb{F}$. At the end of the protocol, the receiver learns $P(\alpha)$ and the sender learns nothing.

Before explaining the first oblivious polynomial evaluation protocol we have to introduce the concept of oblivious transfer protocols.

Oblivious Transfer (OT) protocols allow the sender to transmit part of its inputs to the receiver in a manner that: the sender does not receive more information than it is entitled, while the receiver is assured that the sender does not learn which part of the inputs it received. In a 1-out-of-N OT protocol, the receiver can choose only one input of the sender between N possible choices. Naor and Pinkas desccribe in [NP01] an n-out-of-N OT protocol where the receiver chooses n inputs of the sender from M possible choices.

2.2.1 Noisy Polynomial problem

Naor and Pinkas proposed in [NP99] a protocol for Oblivious Polynomial Evaluation that relies on the hardness of the Noisy Polynomial Reconstruction problem.

- Input:
  - Sender: a polynomial $P(y) = \sum_{i=0}^{d_P} b_i y^i$ of degree d_P in the field $\mathbb{F}$.
  - Receiver: a value $\alpha \in \mathbb{F}$
- Output:
  - Sender: nothing.
  - Receiver: $P(\alpha)$
- Protocol security parameters: m, k.

Generic protocol for oblivious polynomial evaluation:

1. **The sender hides P in a bivariate polynomial:** The sender generates a random masking polynomial $P_x(x)$ of degree d, s.t. $P_x(0) = 0$, where $d = k \cdot d_P$.

$$P_x(x) = \sum_{i=1}^{d} a_i x^i$$

The sender defines a bivariate polynomial

$$Q(x, y) = P_x(x) + P(y) = \sum_{i=1}^{d} a_i x^i + \sum_{i=0}^{d_P} b_i y^i$$

2. **The receiver hides α in a univariate polynomial:** The receiver chooses a random polynomial S of degree k, such that $S(0) = \alpha$.

Define $R(x) = Q(x, S(x))$. $R(x)$ is a polynomial of degree $d_R = d = k \cdot d_P$. The goal of the receiver is to use $R(x)$ to learn $P(\alpha)$. Note that $R(0) = Q(0, S(0)) = P(S(0)) = P(\alpha)$.

3. **The receiver learns points of R:** The receiver learns $d_R + 1$ values of the form $\langle x_i, R(x_i)\rangle$.

4. **The receiver computes $P(\alpha)$:** The receiver uses the values of R that it learned to interpolate $R(0) = P(\alpha)$.

The key is in the third step on how the receiver learns $d_R + 1$ evaluations of $R(x)$ in a secure way.

- The receiver sets $n = d_R+1$ and chooses $N = nm$ distinct random values $x_1, \cdots, x_N \in \mathbb{F}$, all different from 0.
- The receiver chooses a random set T of n indices $1 \leq i_1 < i_2 < \cdots < i_n \leq N$. Then defines N values y_i:
$$y_i = \begin{cases} S(x_i) & \text{if } i \in T \\ y_i \in_R \mathbb{F} & \text{if } i \notin T \end{cases}$$
- The receiver sends the N points $\{(x_i, y_i)\}_{i=1}^N$ to the sender.
- The receiver and sender execute an n-out-of-N oblivious transfer protocol, for the N values $Q(x_i, y_i)$. The receiver chooses to learn $\{Q(x_i, y_i)\}_{i\in T}$

2.2.2 Homomorphic PKE

Let $(KG, \mathcal{E}, \mathcal{D})$ be a Homomorphic PKE Scheme.

– The sender knows a polynomial $P(x) = \sum_{i=0}^{d} a_i x^i$ with $a_i \in \mathbb{Z}$ of degree d which can be evaluated on $\mathcal{M}$ (i.e. we can define an additive operation and a multiplicative operation on $\mathcal{M}$).

– Let (sk, pk) be the secret and public key pair of the receiver. The receiver sends to the sender the powers of α encrypted: $\mathcal{E}_{pk}(\alpha^i)$ for $i \in \{0, ..., d\}$.

– The sender uses the homomorphic property of the encryption scheme to compute the encryption of $P(\alpha)$:

$$\mathcal{E}_{pk}(P(\alpha)) = \mathcal{E}_{pk}(a_0 + a_1\alpha + ... + a_n\alpha^n) =\leftarrow \mathcal{E}_{pk}(\alpha^0)^{a_0} \cdot \mathcal{E}_{pk}(\alpha^1)^{a_1} \cdot ... \cdot \mathcal{E}_{pk}(\alpha^n)^{a_n}$$

and sends it to the receiver.

– The receiver decrypts the received value to get $P(\alpha)$:

$$\mathcal{D}_{sk}(\mathcal{E}_{pk}(P(\alpha))) = P(\alpha)$$

2.3 Bilinear pairings

Let G_1 and G_2 be two (multiplicative) cyclic groups of prime order q. Let g_1 be a fixed generator of G_1 and g_2 be a fixed generator of G_2.

Definition 2.3.1. Computation Diffie-Hellman (CDH) Problem: Given a randomly chosen $g \in G_1$, g^a, and g^b (for unknown randomly chosen $a, b \in \mathbb{Z}_q$), compute g^{ab}.

Definition 2.3.2. Decision Diffie-Hellman (DDH) Problem: Given randomly chosen $g \in G_1$, g^a, g^b, and g^c (for unknown randomly chosen $a, b, c \in \mathbb{Z}_q$), decide whether $c = ab$. (If so, (g, g^a, g^b, g^c) is called a valid Diffie-Hellman tuple.)

Definition 2.3.3. Computational co-Diffie-Hellman (co-CDH) Problem on (G_1, G_2): Given $g_2, g_2^a \in G_2$ and $h \in G_1$ as input, compute $h^a \in G_1$.

Definition 2.3.4. Decision co-Diffie-Hellman (co-DDH) on (G_1, G_2): Given $g_2, g_2^a \in G_2$ and $h, h^b \in G_1$ as input, decide whether $a = b$. If so, we say that (g_2, g_2^a, h, h^a) is a co-Diffie-Hellman tuple.

Definition 2.3.5. Bilinear map: Let G_T be an additional group such that $|G_1| = |G_2| = |G_T|$. A bilinear map is a map $e : G_1 \times G_2 \to G_T$ with the following properties:

1. Bilinear: for all $u \in G_1, v \in G_2$ and $a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degenerate: $e(g_1, g_2) \neq 1$.

Definition 2.3.6. A Gap co-Diffie-Hellman (co-GDH) group pair is a pair of groups (G_1, G_2) on which co-DDH is easy but co-CDH is hard. When (G_1, G_1) is a co-GDH group pair, we say G_1 is a Gap group (GDH).

Remark 2.3.7. If there s a bilinear map on G_1, G_2, then they are a co-GDH group pair. If there is a bilinear map over $G_1 \times G_1$, then G_1 is a gap group, since one can use the bilinear map to solve the DDH problem.

2.4 Secret Sharing

2.4.1 Definitions

Definition 2.4.1. An access structure Γ is the set of all subsets of $\mathcal{P}$ that can recover the secret.

Definition 2.4.2. Let $\mathcal{P} := \{P_1, \ldots, P_n\}$ be a set of participants. A *monotone access structure* Γ on $\mathcal{P}$ is a subset $\Gamma \subseteq 2^{\mathcal{P}}$, which is monotone increasing

$$A \in \Gamma, \quad A \subseteq A' \subseteq \mathcal{P} \Rightarrow A' \in \Gamma$$

Definition 2.4.3. Let $\mathcal{P} := \{P_1, \ldots, P_n\}$ be a set of participants and let $A \subseteq 2^{\mathcal{P}}$. The *closure* of A, denoted $\mathrm{cl}(A)$, is the set

$$\mathrm{cl}(A) = \{C : \exists B \in A \text{ s.t. } B \subseteq C \subseteq \mathcal{P}\}$$

For a monotone access structure Γ we have $\Gamma = \mathrm{cl}(\Gamma)$.

Definition 2.4.4. Let Γ be an access structure on a set of participants $\mathcal{P}$. $B \in \Gamma$ is a *minimal* qualified set if $A \notin \Gamma$ whenever $A \subsetneq B$.

Definition 2.4.5. Let Γ be an access structure on a set of participants $\mathcal{P}$. The family of minimal qualified sets Γ_0 of Γ is called the *basis* of Γ.

For a basis Γ_0 of an access structure Γ we have $\Gamma = \mathrm{cl}(\Gamma_0)$

Definition 2.4.6. An access structure Γ is *trivial* if either $\Gamma = 2^{\mathcal{P}}$ or $\Gamma = \{\mathcal{P}\}$.

Let $\mathcal{K}$ be a set of q elements called *secret keys*, and let $\mathcal{S}$ be a finite set whose elements are called *shares*. Let D be a *dealer* who wants to share a secret key $\mathbf{k} \in \mathcal{K}$ among the participants in $\mathcal{P}$.

Definition 2.4.7. A *distribution rule* is a function $f : \mathcal{P} \cup \{D\} \to \mathcal{K} \cup \mathcal{S}$ which satisfies the conditions $f(D) \in \mathcal{K}$ and $f(P_i) \in \mathcal{S}$ for $i = 1, 2, \ldots, n$.

Secret sharing schemes will be represented by a collection of distribution rules, which represent a possible distribution of shares to the participants where $f(D)$ is the secret key being shared and $f(P_i)$ is the share given to P_i.

Definition 2.4.8. Let $\mathscr{F}$ be a family of distribution rules, and let $\mathbf{k} \in \mathcal{K}$. Then $\mathscr{F}_{\mathbf{k}} := \{f \in \mathscr{F} : f(F) = \mathbf{k}\}$ is the family of all distribution rules having $\mathbf{k}$ as secret.

If $\mathbf{k} \in \mathcal{K}$ is the secret that D wants to share, then D will chose a distribution rule $f \in \mathscr{F}_{\mathcal{K}}$ uniformly at random.

Let $\{p_{\mathcal{K}}(\mathbf{k})\}_{\mathbf{k} \in \mathcal{K}}$ be a probability distribution on $\mathcal{K}$, and let a collection of distribution rules for secrets in $\mathcal{K}$ be fixed.

Definition 2.4.9. A *perfect secret sharing scheme*, with respect to a monotone access structure $\Gamma \subseteq 2^{\mathcal{P}}$, is a collection of distribution rules that satisfy the following two properties:

1. If a subset $A \in \Gamma$ of participants pool their shares, then they can determine the value of the secret $\mathbf{k}$.

2. If a subset $A \notin \Gamma$ of participants pool their shares, then they can determine nothing about the value of the secret $\mathbf{k}$. Formally, if $A \notin \Gamma$ then for all $a = \{(P_i, s_i) : P_i \in A \text{ and } s_i \in \mathcal{S}\}$ with $p(a) > 0$, and for all $\mathbf{k} \in \mathcal{K}$, it holds $p(\mathbf{k}|a) = p_{\mathcal{K}}(\mathbf{k})$. In other words, the *a priori* probability of the value of $\mathbf{k}$ does not change after knowing the shares held by A.

Definition 2.4.10. An *ideal secret sharing scheme* is a secret sharing scheme for which $|\mathcal{K}| = |\mathcal{S}|$. An access structure admitting an ideal secret sharing scheme will be referred as *ideal access structure*.

2.4.2 Shamir Secret Sharing

A classic example of a secret sharing scheme is the one described by Shamir in [Sha79] and it is based on polynomial interpolation. The goal of the scheme is to share a secret $s \in \mathbb{Z}_p$, for p a large prime number, among n parties s.t. any t parties can recover the secret, and no $t-1$ can learn anything about the secret.

- Let $s \in_R \mathbb{Z}_p$ be the secret we want to share among the set of participants $\mathcal{P} = \{P_1, ..., P_n\}$.

- Let $P(x) = a_0 + a_1 x + \cdots + a_{t-1} x^{t-1}$ of degree $t-1$ with $a_1, ..., a_{t-1} \in_R \mathbb{Z}_p$, $a_{t-1} \neq 0$ and $a_0 = s$. Let $\alpha_1, ..., \alpha_n \in_R \mathbb{Z}_p$ all distinct.

- Each participant $P_i \in \mathcal{P}$ is given the share (α_i, y_i) where $y_i = P(\alpha_i)$.

– If a set $\{P_{i_1}, ..., P_{i_t}\}$ of t participants want to recover the secret, they share their shares and compute

$$s = q(0) \leftarrow \sum_{j=1}^{t} y_{i_j} \prod_{k \in [t] \setminus \{j\}} \frac{-\alpha_{i_k}}{x_{i_j} - x_{i_k}}$$

2.4.3 Anonymous secret sharing

In an anonymous secret sharing scheme the secret can be reconstructed without the knowledge of which participants hold which shares. Examples of such anonymous sharing schemes can be found in [BS97].

2.5 Digital Signatures

To ensure integrity of data in communications and authentication, the concept of digital signatures appeared.

A digital signature scheme consists of 3 algorithms:

- **Key generation**: on input of a security parameter k (usually the desired length for the keys), outputs a pair (sk, pk) of secret and public keys.
- **Signature**: given an input message m and the secret key sk, outputs a signature σ.
- **Verification**: given an input message m, a signature σ on the message and a public key pk, outputs whether the signature is valid or not.

A signature scheme must satisfy the following properties:

- **Correctness**: A signature generated with the signing algorithm must always be accepted by the verifier.
- **Unforgeability**: An adversary who only knows the public key pk cannot obtain valid signatures for this public key. That is, only the owner of the matching secret key sk can properly sign with respect to pk.

2.5.1 Examples

ElGamal

[ElG85] Let H be a collision-resistant hash function. Let p be a large prime such that the *discrete logarithm problem* is difficult over $\mathbb{Z}_p$. Let g be a randomly chosen generator of $\mathbb{Z}_p^*$

Key generation. Randomly choose a secret key $x \in \mathbb{Z}_p^*$, and compute the public key $y = g^x$.

Signature. To sign a message m, the signer chooses a random $k \in \mathbb{Z}_p^*$. Compute $r = g^k$. To compute s, the following equation must be satisfied: $g^{H(m)} = g^{xr} g^{ks}$. So $s = (H(m) - xr)\, k^{-1} \pmod{p-1}$

If $s = 0$, it starts over again with a different k.

The pair (r, s) is the digital signature for m.

Verification. Check $g^{H(m)} = y^r r^s$

The use of $H(\cdot)$ prevents an existential forgery attack.

Boneh-Lynn-Shacham (BLS)

[BLS01] Let G_1, G_2 be a bilinear group pair of prime order p. Let g be a generator of G_1. Let $e : G_1 \times G_2 \to G_T$ be a non-degenerate bilinear pairing. Let $H : \{0,1\}^* \to G_1$ be a full-domain hash function.

Key generation. Randomly choose a secret key $x \in \mathbb{Z}_p$. The public key is $y = g_2^x$.

Signature. Given a private key $x \in \mathbb{Z}_p$, and a message $m \in \{0,1\}^*$, compute $h = H(m) \in G_1$ and $\sigma = h^x$. The signature is $\sigma \in G_1$.

Verification. Given a public key y, a message $m \in \{0,1\}^*$ and a signature $\sigma \in G_1$, compute $h = H(m) \in G_1$ and verify that $e(\sigma, g_2) = e\,(h, y)$.

Schnorr

[Sch90] Let G be a cyclic group of prime order p. Let g be a generator of G. Let $H : \{0,1\}^* \to \mathbb{Z}_p$ be a hash function.

Key generation. Randomly choose a secret key $x \in \mathbb{Z}_p$. The public key will be $y = g^x$.

Signature. Randomly choose $z \in \mathbb{Z}_p$, and compute $L := g^z$.
Compute $c := H(L \parallel m)$.
Compute $s := z + c \cdot x$
The signature on m is $\sigma = (c, s)$.

Verification. Given a signature σ and a public key y, computes $L^\dagger := g^s y^{-c}$ and then check that $c = H(L^\dagger \parallel m)$

2.6 Group Signatures

(With linkability) Use of aggregation: group signatures. They are used to sign on behalf of the group, prove group membership.

The easiest way is to give everyone the secret key, so they can sign. But this would let any colluded user to share the secret key to other parties, which is not admissible.

Some group signatures need what is called a Dealer, which will deal with the keys.

Examples of Group signatures:

2.7 Threshold Digital Signatures

A (t, n)-threshold signature scheme is a signature scheme in which any t participants of the group $\mathcal{P} = \{P_1, ..., P_n\}$ of participants is able to compute a signature on behalf of the group.

2.7.1 Boldyreva

This signature scheme is based on the BLS scheme (see section 2.5.1) and was proposed by Boldyreva in [Bol03].

Setup Algorithm

- Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be the set of participants. Let G be a gap group of large prime order $p > n$. Let $g \in G$ be a generator of the group.
- Let $sk \in_R \mathbb{Z}_p$ be the secret key of the threshold signature scheme. Set $pk = g^{sk}$ the public key. Let $P(x)$ be a random polynomial over $\mathbb{Z}_p$ of degree $t-1$ with $P(0) = sk$.
- Let $\alpha_1, ..., \alpha_n \in_R \mathbb{Z}_p$ all distinct. Each participant $P_i \in \mathcal{P}$ is assigned a public key $pk_i = \alpha_i$ and a secret key $sk_i = s_i := P(\alpha_i)$.

  Note that, for any set $\{P_{i_1}, ..., P_{i_t}\} = P \subseteq \mathcal{P}$ of t participants:

$$sk = P(0) = \sum_{j=1}^{t} s_{i_j} \lambda_{i_j}^P$$

  where $\lambda_{i_j}^P := \prod_{k \in [t] \setminus \{j\}} \frac{-\alpha_k}{\alpha_j - \alpha_k}$

Signing Algorithm

- Let $P = \{P_{i_1}, ..., P_{i_t}\}$ be a set of t participants. Each participant P_{i_j} computes his partial signature $\sigma_{i_j}(m) = H(m)^{s_{i_j}}$ and broadcasts the pair $(\alpha_{i_j},\ \sigma_{i_j}(m))$.
- The signature σ on m is computed:

$$\sigma(m) = \prod_{P_i \in P} \sigma_i(m)^{\lambda_i^P} = H(m)^{\sum_{P_i \in P} \lambda_i^P s_i} = H(m)^{sk}$$

Verifying Algorithm

- Let $e : G \times G \to G_t$ be a bilinear pairing. Let σ be a signature on a message m.
- The signature σ on m is valid if and only if $e(\sigma, g) = e(H(m), pk)$

CHAPTER 3

ANONYMITY IN THRESHOLD SIGNATURES

In order to compare different schemes we need to clear up the definition of anonymity.

The word anonymity is derived from the Greek word *anonymia*, meaning "without a name". In technical terms, the "name" of a participant would be something that uniquely identifies him, e.g. his public key. So, a scheme would be anonymous if the public key of the participant is not disclosed or cannot be obtained in any way at any moment.

Many authors use the term *anonymous threshold scheme* referring to a threshold scheme that does not require knowing which participant holds which share. The scheme proposed by Boldyreva in [Bol03] does not require knowing which participant holds each share, but we cannot call it an anonymous scheme because there is only one participant P_i holding the value α_i even though the proper identity of P_i was not disclosed. Any party can check if a certain participant P_i that holds the value α_i has participated in a signature σ just by checking if α_i was used in the computation of σ. But still without knowing which participant holds it. We will call this property *linkability*.

What we are looking for in this work, by saying *anonymity*, is two properties:

- Unlinkability: cannot decide whether two different signatures were signed by the same user.
- Untraceability: cannot get the public key of the signer from a valid signature.

A threshold signature scheme that has the linkability property is suitable for a "one-time anonymity". That is, once a signature has been computed, the whole scheme has to be set up again. We discuss a ¿few? solutions of this kind in chapter 4.

A threshold signature scheme that is unlinkable is suitable for anonymity still after multiple uses. We discuss three solutions in chapter 5.

CHAPTER 4

SINGLE USE: ANONYMITY

In this chapter we consider threshold signature schemes that with non-traceability but with linkability.

As we commented previously, we can avoid the linkability of a threshold signature scheme by setting it up newly every time a signature is computed. We describe two examples.

4.1 Anonymous Secret Sharing Scheme

Consider a signature scheme with secret sk. Using the (t, n)-anonymous threshold secret sharing scheme described in [BS97] we can share the secret sk among the participants through a secure channel. When a set of t participants want to sign a message, they can just recover the secret sk and compute a signature with it.

After that, the signature scheme is newly setup by choosing a different secret sk and sharing it again using an anonymous threshold secret sharing scheme.

In this solution, the dealer (the one who gives the shares to the participants) can know which participant holds which share because the participants have to authenticate themselves to avoid a single participant getting multiple shares. We can approach this by running a 1-out-of-N OT protocol randomly choosing a share for sufficiently large $N \gg n$ s.t. the probability of two participants choosing the same share is sufficiently low. In this way, the dealer does not know which participants holds which share, but the execution of an OT protocol for large N is computationaly way too expensive.

4.2 Threshold BLS signature

Using the BLS threshold signature scheme described in 2.7.1 we can avoid linkability setting up a new secret after a signature is computed.

After the signature scheme is set up with a new secret sk and a new random polynomial $P(x)$, a participant P_i chooses $\alpha_i \in_R \mathbb{Z}_p$ and learns $P(\alpha_i)$ running an oblivious polynomial evaluation protocol. In this way, the dealer does not learn the value of α_i.

Since the participant actually does not need explicitly the value of $P(\alpha_i)$ but $H(m)^{P(\alpha_i)}$ to compute the partial signature on m, the participant can send the exponentiation of the powers of α_i to the dealer ($H(m)^{\alpha_i^j}$ for $j \in \{0, ..., t-1\}$) and the dealer returns them exponentiated to the coefficients of the polynomial ($\left(H(m)^{\alpha_i^j}\right)^{a_j}$ for $j \in \{0, ..., t-1\}$).

In this way, the participant can compute:

$$H(m)^{P(\alpha_i)} = \prod_{0 \leq j \leq t-1} \left(H(m)^{\alpha_i^j} \right)^{a_j} = H(m)^{\sum_{0 \leq j \leq t-1} a_j \alpha_i^j}$$

The issue with this last method is that the participant has to wait the message to be chosen and then start communicating with the dealer. This could happen to many participants at the same time which may result in an overload for the dealer.

CHAPTER 5

MULTIPLE USE: ANONYMITY WITH NON-LINKABILITY

Here we describe three solutions for anonymous threshold signature schemes with non-linkability property.

5.1 Constant size anonymous threshold signature

Daza et al. proposed in [DDSV09] an anonymous threshold signature scheme that sets a (t, r)-threshold signature scheme based on Shamir's secret sharing over a partition of the set $\mathcal{P}$ of participants.

5.1.1 Description

Setup Algorithm

- Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be the set of participants. Consider d distinct partitions of $\mathcal{P}$ into r parts. $\mathcal{P}^i = \{\mathcal{P}^i_1, ..., \mathcal{P}^i_r\}$ for $i \in \{1, ..., d\}$. This algorithm will set d different threshold signature schemes, one for each partition of $\mathcal{P}$.

- Let $p > n$ be a sufficiently large prime. Let $sk_i \in_R \mathbb{Z}_p$ be the secret key of the i-th threshold signature scheme. Let $P_i(x)$ be a random polynomial over $\mathbb{Z}_p$ of degree $t-1$ with $P_i(0) = \mathrm{sk}_i$ for $i \in \{1, ..., d\}$.

- For $i \in \{1, ..., d\}$ let $\alpha^{(i)}_1, ..., \alpha^{(i)}_r \in_R \mathbb{Z}_p$ all distinct (for fixed i). Each participant $P_k \in \mathcal{P}^i_j$ is given public key $pk^{(i)}_k = \alpha^{(i)}_j$ and secret key $sk^{(i)}_k = P_i(\alpha^{(i)}_j)$ for the i-th threshold signature scheme.

Signing Algorithm

- Let $\{P_{i_1}, ..., P_{i_t}\} \subseteq \mathcal{P}$ a set of t participants which will try to sign a message m.

- A signature on m over the i-th threshold signature scheme is attempted using the protocol described in section 2.7.1 providing the message, the signature and the signature scheme over which is signed: (m, σ, i).

- If the signature fails (because $\alpha^{(i)}_{i_1}, ..., \alpha^{(i)}_{i_t}$ are not all distinct), a new signature on m is attempted over a threshold signature scheme different from the previous tried.

- Eventually, the signature will succeed over a certain signature scheme.

Verifying Algorithm

– Let (m, σ, i) be a signature on a message m. The verification is done using the verifying algorithm described in 2.7.1.

– Let e be a bilinear pairing. The signature is valid if and only if $e(\sigma, g) = e(H(m), pk_i)$.

5.1.2 Analysis

In this signature scheme it is not sure that any set of t participants will be able to compute a signature on a given message. The probability of succeeding on signing a message depends on the parameters t, n, r, d and how the partitions are made.

To see the relation between these parameters and the probability of success, we will describe a few examples.

Given $\{P_{i_1}, ...P_{i_t}\}$ a set of t participants, they will succeed only if $\alpha_{i_1}^{(i)}, ..., \alpha_{i_t}^{(i)}$ are all distinct for a certain $i \in \{1, ..., d\}$. Hence, $t \leq r$.

Random partitions

Suppose that for each participant $P_k \in \mathcal{P}$ and for all $i \in \{1, ..., d\}$ the probability $Pr(P_k \in \mathcal{P}_j^i) = \frac{1}{r}$ holds.

The probability that, given t participants, they succeed on signing a message in the i-th threshold signature scheme (i.e. no two participants share the same public key) is given by $p_{succ,i} = \prod_{k=0}^{t-1}(1 - \frac{i}{r}) = 1 - \frac{t(t-1)}{2r} + o\left(\frac{t^2}{r}\right)$

Thus, the probability of failing to sign a message in the i-th threshold signature scheme is $p_{fail,i} = 1 - p_{succ,i} = \frac{t^2}{2r} + o\left(\frac{t^2}{r}\right)$.

The global probability of failing to sign a message is $p_{fail} = \prod_{i=1}^{d} p_{fail,i} = \left(\frac{t^2}{2r}\right)^d + o\left(\left(\frac{t^2}{r}\right)^d\right)$

$\frac{t^2}{2r}$	t	$\frac{n}{r}$	n	r	p_{fail}
10^{-3}	5	10^3	$12.5 \cdot 10^6$	$12.5 \cdot 10^3$	$0.80 \cdot 10^{-3}$
	10		$50 \cdot 10^6$	$50 \cdot 10^3$	$0.90 \cdot 10^{-3}$
	50		$1.25 \cdot 10^9$	$1.25 \cdot 10^6$	$0.98 \cdot 10^{-3}$
	100		$5 \cdot 10^9$	$5 \cdot 10^6$	$0.99 \cdot 10^{-3}$

Table 5.1: Sample values for $p_{fail,i}$

Deterministic partitions

To simplify the analysis, for each participant $P_k \in \mathcal{P}$ we will consider the corresponding codeword in a (not necessary linear) code of length d over an alphabet of size r given by:

$$c_k := (j_1, ..., j_d) \text{ iff } P_k \in \mathcal{P}_{j_i}^i \quad \forall i \in \{1, ..., d\}$$

Theorem 5.1.1 (Singleton [Sin64]). *Let C be a code of length n, minimum distance d over an alphabet of size q and cardinality M. The cardinality is upper bounded by* $\log_q M \leq n - d + 1$.

Changing the notation to adapt it to our case, if we want all codewords associated to the participants to be different, we have $log_r n \leq d - \delta + 1$ where δ would be the minimum distance of the code (i.e. the least number of schemes that any two participants have distinct public keys in them). For $\delta \geq 1$ we have $n \leq r^d$.

It is reasonable to wonder whether it is possible or not to define d partitions of $\mathcal{P}$, such that any set of t participants lie in diferent parts of a certain partition $\mathcal{P}^i$. To answer that question, we need the following definition.

Definition 5.1.2. Let $m \geq w \geq 2$. An (n, m, w)-perfect hash family is a set of functions $\mathscr{F}$ where $|Y| = n$, $|X| = m$ and $f : Y \to X$ for each $f \in \mathscr{F}$, such that, for any $C \subseteq Y$ with $|C| = w$, there exists at least one function $f \in \mathscr{F}$ such that $f|_C$ is one-to-one. When $|\mathscr{F}| = N$, an (n, m, w)-perfect hash family will be denoted by $\mathbf{PHF}(N; n, m, w)$.

Getting back to our question: let $Y = \mathcal{P}$, $X = \{1, ..., r\}$ and let $\mathscr{F} = \{f_1, ..., f_d\}$ where $f_i(P_k) = (c_k)_i$. It is possible to define d partitions of P in a way that any set of t participants lie in diferent parts of a certain partition P^i if and only if there exists a $\mathbf{PHF}(d; n, r, t)$. PHF exist only for suitable sets of values for the parameters. The following theorem gives a (not tight) lower bound on $|\mathscr{F}|$ s.t. there exists a PHF.

Theorem 5.1.3 (2.1 [DSW04]). *There exists a* $\boldsymbol{PHF}(d; n, r, t)$ *if*

$$d > \frac{\log \binom{n}{t}}{\log r^t - \log(r^t - t!\binom{r}{t})}$$

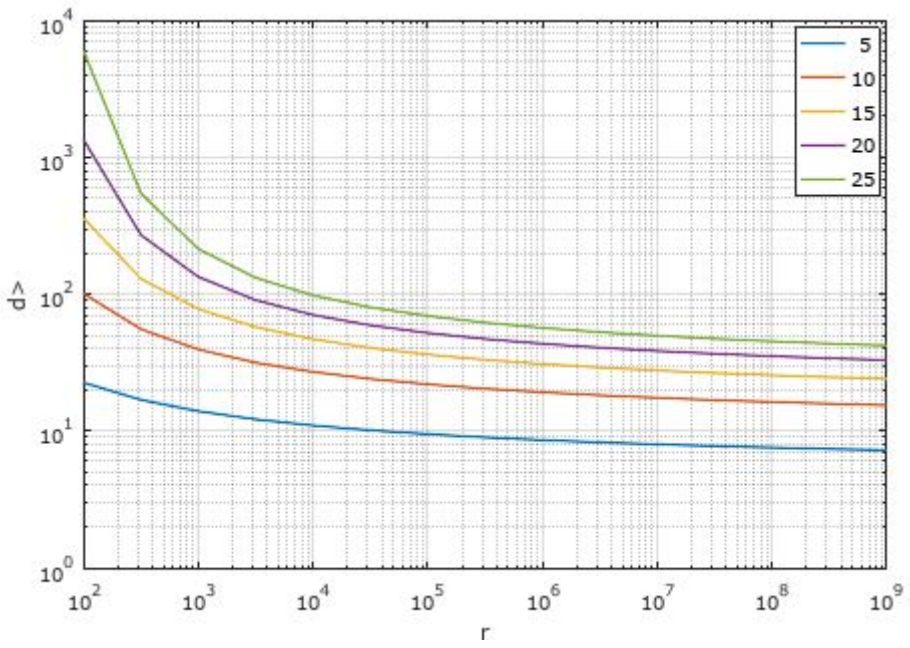

Figure 5.1: Values for the lower bound of d where $\frac{n}{r} = 10^3$

The plot in Figure 5.1 gives a hint that the bound might have limit when $r \to \infty$ fixing $\frac{n}{r} = c$ constant. Note that $r^t - t!\binom{r}{t} = \frac{t(t-1)}{2} r^{t-1} + o(r^{t-1})$ for fixed t. Using the

inequality $\left(\frac{n}{t}\right)^t \leq \binom{n}{t} \leq n^t$ and the previous approximation for $r \gg t$, we have:

$$\frac{t \log \frac{n}{t}}{\log r + \log \frac{t(t-1)}{2}} \lesssim \frac{\log \binom{n}{t}}{\log r^t - \log(r^t - t!\binom{r}{t})} \lesssim \frac{t \log n}{\log r + \log \frac{t(t-1)}{2}}$$

Clearly, the limit when $r \to \infty$ and $n = c \cdot r$ (c a constant), is t on both sides of the inequality.

5.2 Linkable Group Signature Scheme

Chen, Ng, and Wang proposed a (t, n)-threshold signature scheme in [CNW11] using a group signature scheme where you can link two signatures on the same message but cannot link signatures on different messages. As one can link signatures on the same message, the threshold signature is simply a set of t valid signatures on the same message such that no two are linked.

5.2.1 Description

Setup Algorithm

- Let G_1, G_2, G_T be cyclic groups of sufficiently large prime order q. Two random generators $g_1 \in G_1$, $g_2 \in G_2$, and a bilinear pairing $\hat{t} : G_1 \times G_2 \to G_T$.

  DDH problem in G_1, Gap-DL problem in G_1 and G_2 and the blind bilinear LRSW problem are hard.

- Let $H_0 : \{0,1\}^* \to \mathbb{Z}_q$ and $H_1 : \{0,1\}^* \to G_1$ be two hash functions.

- For each issuer $i \in \mathcal{I}$ the following is performed.

  Two integers are selected $x, y \in_R \mathbb{Z}_q$ and the issuer secret key **isk** is assigned to be (x, y). Then the values $X = g_2^x \in G_2$ and $Y = g_2^y \in G_2$ are computed. The issuer public key **ipk** is assigned to be (X, Y).

- The system public parameters *par* are set to be $par = (G_1, G_2, G_T, \hat{t}, g_1, g_2, H_0, H_1, \text{ipk}_k)$ and are published.

Join protocol

- In this a protocol, an issuer $i \in \mathcal{I}$ computes a credential for a signer $\mathfrak{s} \in S$ that allows him to compute valid signatures on messages.

- The signer $\mathfrak{s}$ randomly chooses the secret key $sk_{\mathfrak{s}} = f \in_R \mathbb{Z}_p$ and computes the public key $pk_{\mathfrak{s}} = F = g_1^f$.

- The issuer $\mathfrak{i}$ randomly chooses $r \in_R \mathbb{Z}_q$ and computes $A = g_1^r$, $B = A^y$ and $C = A^x F^{rxy}$. The credential for $\mathfrak{s}$ is set to be $cre_{\mathfrak{s}} \leftarrow (A, B, C)$.

Signing Algorithm

- Let $m \in \{0,1\}^*$ be the message that a signer $\mathfrak{s}$ wants to sign where $sk_{\mathfrak{s}} = f$, $pk_{\mathfrak{s}} = F$ and $cre_{\mathfrak{s}} = (A, B, C)$.
- $\mathfrak{s}$ chooses $z \in_R \mathbb{Z}_q$ and computes $J = H_1(m)$, $K = J^f$ and $L = J^z$.
- $\mathfrak{s}$ chooses $a \in_R \mathbb{Z}_q$ and randomizes its credential into $(R, S, T) \leftarrow (A^a, B^a, C^a)$, and computes $\tau = \hat{t}(S, X)^z$ where X is from the issuer's public key.
- $\mathfrak{s}$ computes $c \leftarrow H_0(R \| S \| T \| \tau \| J \| K \| L \| m)$ and sets $s = z + c \cdot f$.
- The signature on m is set to be $\sigma = (R, S, T, J, K, L, c, s)$

The computation of L,c and s gives a non-interactive Zero-Knowledge Proof of Knowledge for f.

The computation of τ gives a proof of knowledge of the credentials of the signer. This is based on the *Schnorr Signature Scheme* detailed in section 2.5.1.

Verifying Algorithm

- Let $\sigma = (R, S, T, K, L, c, s)$ be a signature on a message m computed by a signer $\mathfrak{s}$.
- The verifier $\mathfrak{v}$ checks if $J = H_1(m)$ and $\hat{t}(R, Y) = \hat{t}(S, g_2)$ hold. If not, the signature is not valid.
- $\mathfrak{v}$ computes $\tau' = \hat{t}(R, X)^c \hat{t}(S, X)^s \hat{t}(T, g_2)^{-c}$ and $L' = H_1(m)^s \cdot K^{-c}$.
- The signature is valid only if $c = H_0(R \| S \| T \| \tau' \| K \| L' \| m)$

Threshold Checking Algorithm

- When a verifier $\mathfrak{v}$ receives a signature (σ, m), checks that the signature is valid, and then checks the signature against the list of ℓ valid signatures (σ_i, m) on the message m already received to ensure it is not a duplicate message signed by some verifier. To do it, just check $K \neq K_i$ for $i \in \{1, ..., \ell\}$.
- If it is not a duplicate, then $\mathfrak{v}$ adds (σ, m) to the list of valid signatures.
- When $\ell = t$, the threshold signature is set to be the collection $\{(\sigma_i)\}_{i \in \{1,...,t\}}$ of t valid signatures on m computed by t distinct signers.

5.3 Anonymous interactive protocol

We propose an interactive protocol based on the signature scheme described in section 2.7.1. Recall that this scheme does not have the *unlinkability* property because it uses "pseudonyms" and they are shared to compute the signature. Thus, the idea of this new protocol is to hide these "pseudonyms".

As in the BLS signature scheme, a participant P_i from a subset $P \subset \mathcal{P}$ of t participants will compute a partial signature $\sigma_i(m)$ on a message m. The partial signature will be $\sigma_i(m) = H(m)^{s_i \prod_{P_j \in (P \setminus P_i)} \frac{-\alpha_j}{\alpha_i - \alpha_j}}$.

The goal of this modification is to compute $a^{\frac{-\alpha_j}{\alpha_i-\alpha_j}}$ without sharing the values of α_i and α_j, for $1 \neq a \in G$ and a given $P_j \in P$.

To compute $a^{\frac{-\alpha_j}{\alpha_i-\alpha_j}}$ we need participants P_i, P_j and a third party P_s that could be any other participant or a reliable party (like secure hardware).

5.3.1 Description

Interaction

Let $a^{\frac{-\alpha_j}{\alpha_i-\alpha_j}} \leftarrow \mathcal{B}(a, P_i, P_j)$ the protocol that outputs $a^{\frac{-\alpha_j}{\alpha_i-\alpha_j}}$ given $1 \neq a \in G$, a first participant P_i and a second participant P_j.

This protocol is split in five steps. In the figures, the arrows between participants represent communication through a secure channel. The BC block represents a broadcast channel, so anything sent to BC is broadcasted to the rest.

First step: P_i chooses $x_i, x_j, x_s \in \mathbb{Z}_p^*$ three random values and shares them with P_j and P_s. These will be the new "pseudonyms".

P_i and P_j randomly choose polynomials f_i, g_i, z_i and f_j, g_j, z_j, respectively, where: f_i, f_j and g_i, g_j are linear, $g_i(0) = \alpha_i$ and $g_j(0) = \alpha_j$, and z_i, z_j are quadratic polynomials with $z_i(0) = z_j(0) = 0$.

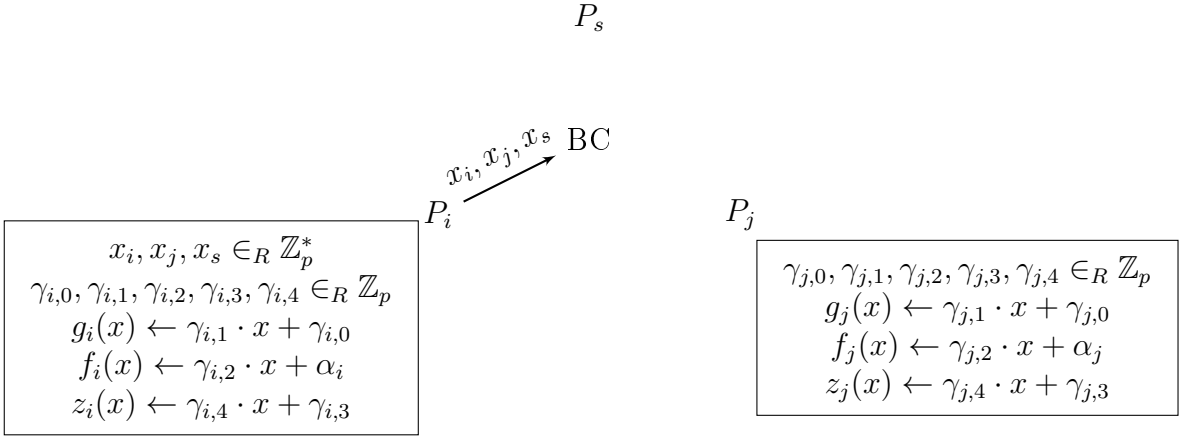

Figure 5.2: Step 1

Second step: Let $h(x) = (g_i(x) + g_j(x))(f_i(x) - f_j(x)) + z_i(x) + z_j(x)$. Note that $h(0) = v \cdot (\alpha_i - \alpha_j)$ for $v := g_i(0) + g_j(0)$.

P_i and P_j share with the rest the evaluations of the random polynomials s.t. P_i, P_j, P_s can compute the evaluations $h(x_i), h(x_j), h(x_s)$ respectively.

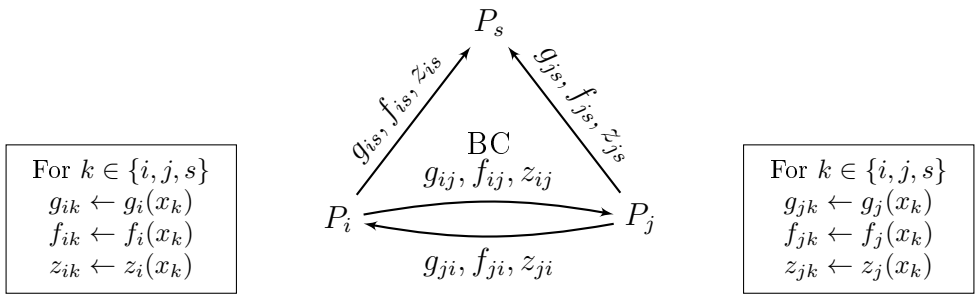

Figure 5.3: Step 2

Third step: P_i,P_j,P_s compute the evaluation of h and share it with the rest. P_i and P_j interpolate the value $h(0)$.

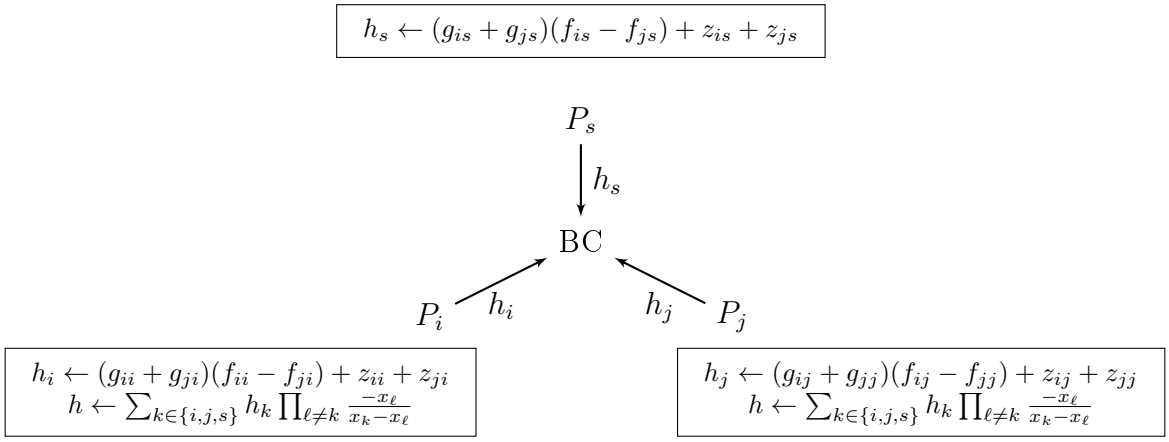

Figure 5.4: Step 3

Fourth step: P_i,P_j compute $A_i = a^{\frac{1}{h(0)}(g_i(x_i)+g_j(x_i))}$, $A_j = a^{\frac{1}{h(0)}(g_i(x_j)+g_j(x_j))}$ respectively. P_j shares A_j with P_i.

P_j can interpolate the exponents of A_i and A_j and compute $a^{\frac{g_i(0)+g_j(0)}{h(0)}} = a^{\frac{v}{v(\alpha_i-\alpha_j)}} = a^{\frac{1}{\alpha_i-\alpha_j}}$.

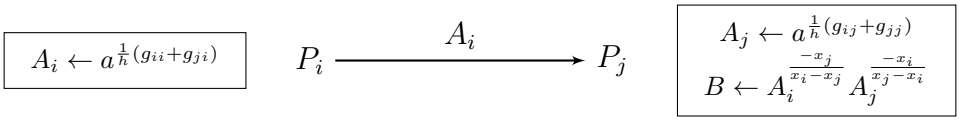

Figure 5.5: Step 4

Fifth step: P_j computes $B^{-\alpha_j} = a^{\frac{-\alpha_j}{\alpha_i-\alpha_j}}$ and shares it with P_i.

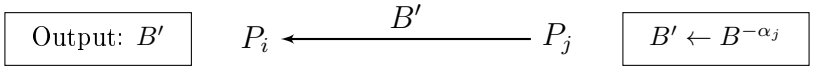

Figure 5.6: Step 5

Partial signature

Let $P = \{P_i, P_{j_1}, \ldots, P_{j_{t-1}}\}$.

For P_i to compute the partial signature over a message m, computes $\sigma_i(m) = a_{t-1}$ where $a_k \leftarrow \mathcal{B}(a_{k-1}, P_i, P_{j_k})$ for $k \in \{1, ..., t-1\}$ and $a_0 = H(m)^{s_i}$

$$\sigma_i(m) = a_{t-1}^{\frac{-\alpha_{j_{t-1}}}{\alpha_i - \alpha_{j_{t-1}}}} = a_k^{\frac{-\alpha_{j_k}}{\alpha_i - \alpha_{j_k}} \cdots \frac{-\alpha_{j_{t-1}}}{\alpha_i - \alpha_{j_{t-1}}}} = a_1^{\frac{-\alpha_{j_1}}{\alpha_i - \alpha_{j_1}} \cdots \frac{-\alpha_{j_{t-1}}}{\alpha_i - \alpha_{j_{t-1}}}} = H(m)^{s_i \frac{-\alpha_{j_1}}{\alpha_i - \alpha_{j_1}} \cdots \frac{-\alpha_{j_{t-1}}}{\alpha_i - \alpha_{j_{t-1}}}}$$

Signature

The signature $\sigma(m)$ on a message m from a group of t participants $P = \{P_1, ..., P_t\}$ is

$$\sigma(m) = \prod_{i=1}^{m} \sigma_i(m)$$

5.3.2 Analysis

Unlinkability

The scheme keeps the unlinkability property while all participants remain honest but curious. Consider an interaction step with participants P_i, P_j, P_s. If an adversary $\mathcal{A}$ corrupts P_j and P_s, $\mathcal{A}$ knows $f_i(x_j)$ and $f_i(x_s)$, being able to interpolate f_i to obtain $\alpha_i = f_i(0)$.

If we want the scheme to be still unlinkable after corrupting ℓ participants, we can extend the protocol in an analogous way where f_k and g_k are polynomials of degree ℓ and each interaction step needs $2\ell + 1$ participants (P_i, P_j and $2\ell - 1$ other parties P_{s_k}.

Computational Cost

For a participant P_i to compute σ_i, $t-1$ interactions with different participants are needed. Then, to compute a valid signature, $t(t-1)$ interactions. We can consider this computationally expensive for large t.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

Conclusions and future work

BIBLIOGRAPHY

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. "Short Signatures from the Weil Pairing". In: *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*. ASIACRYPT '01. London, UK, UK: Springer-Verlag, 2001, pp. 514–532. ISBN: 3-540-42987-5. URL: http://dl.acm.org/citation.cfm?id=647097.717005.

[Bol03] Alexandra Boldyreva. "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 Theory and Practice in Public Key Cryptography, Miami, FL, USA, January 6-8, 2003, Proceedings*. 2003, pp. 31–46. DOI: 10.1007/3-540-36288-6_3. URL: https://doi.org/10.1007/3-540-36288-6_3.

[BS97] C. Blundo and D.R. Stinson. "Anonymous secret sharing schemes". In: *Discrete Applied Mathematics* 77.1 (1997), pp. 13 –28. ISSN: 0166-218X. DOI: https://doi.org/10.1016/S0166-218X(97)89208-6. URL: http://www.sciencedirect.com/science/article/pii/S0166218X97892086.

[CNW11] Liqun Chen, Siaw-Lynn Ng, and Guilin Wang. "Threshold Anonymous Announcement in VANETs". In: *IEEE Journal on Selected Areas in Communications* 29.13 (2011), pp. 605–615. DOI: 10.1109/JSAC.2011.110310.

[DDSV09] Vanesa Daza, Josep Domingo-Ferrer, Francesc Sebé, and Alexandre Viejo. "Trustworthy Privacy-Preserving Car-Generated Announcements in Vehicular Ad Hoc Networks". In: *IEEE Transactions on Vehicular Technology* 58.4 (2009), pp. 1876–1886. DOI: 10.1109/TVT.2008.2002581.

[DH76] W. Diffie and M. E. Hellman. "New Directions in Cryptography". In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654.

[DSW04] D. Deng, D. R. Stinson, and R. Wei. "The Lovász Local Lemma and Its Applications to some Combinatorial Arrays". In: *Designs, Codes and Cryptography* 32.1 (2004), pp. 121–134. ISSN: 1573-7586. DOI: 10.1023/B:DESI.0000029217.97956.26. URL: https://doi.org/10.1023/B:DESI.0000029217.97956.26.

[ElG85] Taher ElGamal. "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms". In: *Advances in Cryptology*. Ed. by George Robert Blakley and David Chaum. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985, pp. 10–18. ISBN: 978-3-540-39568-3.

[NP01] Moni Naor and Benny Pinkas. "Efficient Oblivious Transfer Protocols". In: *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '01. Washington, D.C., USA: Society for Industrial and Applied Mathematics, 2001, pp. 448–457. ISBN: 0-89871-490-7. URL: http://dl.acm.org/citation.cfm?id=365411.365502.

[NP99] Moni Naor and Benny Pinkas. "Oblivious Transfer and Polynomial Evaluation". In: *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*. STOC '99. Atlanta, Georgia, USA: ACM, 1999, pp. 245–254. ISBN: 1-58113-067-8. DOI: 10.1145/301250.301312. URL: http://doi.acm.org/10.1145/301250.301312.

[PP92] Steven J. Phillips and Nicholas C. Phillips. "Strongly ideal secret sharing schemes". In: *Journal of Cryptology* 5.3 (1992), pp. 185–191. ISSN: 1432-1378. DOI: 10.1007/BF02451114. URL: https://doi.org/10.1007/BF02451114.

[Sch90] C. P. Schnorr. "Efficient Identification and Signatures for Smart Cards". In: *Advances in Cryptology — CRYPTO' 89 Proceedings*. Ed. by Gilles Brassard. New York, NY: Springer New York, 1990, pp. 239–252. ISBN: 978-0-387-34805-6.

[Sha79] A. Shamir. "How to Share a Secret". In: *Communications of the ACM* 22.11 (1979), pp. 612–613.

[Sin64] R. Singleton. "Maximum distance q-nary codes". In: *IEEE Transactions on Information Theory* 10.2 (1964), pp. 116–118. ISSN: 0018-9448. DOI: 10.1109/TIT.1964.1053661.