MASTER'S THESIS

Master of Science in Advanced Mathematics and Mathematical Engineering

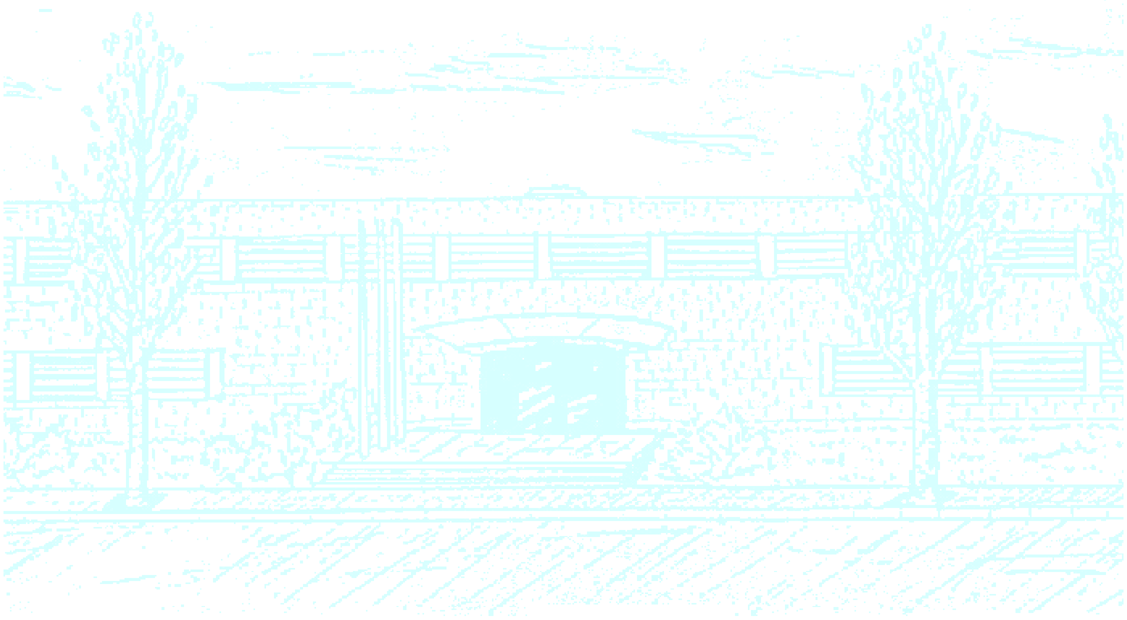
Title: Anonymous threshold signatures

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

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

Resum

Resum

Resum

Resumen

Resum

ACKNOWLEDGEMENTS

Acknowledgements

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Informal introduction on what is the problem we are approaching, what solutions we found, and what 'cons' have these solutions.

CHAPTER 2

PRELIMINARIES

Some cryptographic preliminaries and other definitions.

2.1 Homomorphic Public Key Encryption

Let ¿? be a public key cryptosystem with encryption algorithm (function?) $\mathcal{E}$ and decryption algorithm (function?) $\mathcal{D}$ where: $m' \leftarrow \mathcal{E}_{pk}(m)$ and $m \leftarrow \mathcal{D}_{sk}(m')$

Let $(G, \bullet)$ and $(H, \circ)$ be two groups such that $\mathcal{E}_{pk} : G \rightarrow H$.

Then, ¿? is a Homomorphic Public Key Encryption Scheme if

$$\mathcal{E}_{sk}(m_1 \bullet m_2) \leftarrow \mathcal{E}_{sk}(m_1) \circ \mathcal{E}_{sk}(m_2)$$

Example: **ElGamal**

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.

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.

Generic Protocol:

- 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 $R(x) = Q(x, S(x))$. 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 ? be a Homomorphic Public Key Encryption Scheme with (¿possibly probabilistic?) encoding and decoding functions $\mathcal{E} : G \to H$ and $\mathcal{D} : H \to G$. Where $(G, \bullet)$ and $(H, \circ)$ are groups.

We will write G additively, and H multiplicatively.

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 the group G.

The receiver encrypts the powers of α: $\beta_i \leftarrow \mathcal{E}(\alpha^i)$ for $i \in \{0, ..., d\}$ and sends them to the sender.

The sender uses the homomorphic property of the encryption scheme to compute $\beta = \mathcal{E}(P(\alpha)) \leftarrow \beta_0^{a_0} \cdot \beta_1^{a_1} \cdot ... \cdot \beta_n^{a_n}$ and sends it to the receiver.

The receiver decrypts the received value to get $P(\alpha)$: $\mathcal{D}(\beta) = \mathcal{D}(\mathcal{E}(P(\alpha))) = P(\alpha)$

2.3 Secret Sharing

2.3.1 Definitions

Definition of access structures and secret sharing
[PP92] [BS97]
An access structure Γ is the set of all subsets of $\mathcal{P}$ that can recover the secret.

Definition 2.3.1. Let $\mathcal{P} := \{P_1, \dots, P_n\}$ 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 2.3.2. Let $\mathcal{P} := \{P_1, \dots, P_n\}$ 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 2.3.3. 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 2.3.4. 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 2.3.5. 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 2.3.6. 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 $i = 1, 2, \dots, n$.

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 2.3.7. 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 2.3.8. 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 2.3.9. 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*.

2.3.2 Shamir Secret Sharing

Shamir secret sharing described in [Sha79].

The scheme is based on polynomial interpolation.

Let p be a large prime number. All operations are done in $\mathbb{Z}_p$

We want to share a secret s into n shares so that the secret can be recovered with any k distinct shares.

Let $q(x) = a_0 + a_1 x + \cdots + a_{k-1} x^{k-1}$ be a random polynomial of degree $k - 1$ in which $a_0 = s$.

Each participant i in $\mathcal{P} = \{1, ..., n\}$ is given a different random number $x_i \in \mathbb{Z}_p$ which identifies the participant. Then, each participant i is given the share $y_i = q(x_i)$.

To recover the secret, we only need k different shares. Let $P \subset \mathcal{P}$ be any subset of k participants. Then

Let $\lambda_i^P = \prod_{P_j \in (P \setminus P_i)} \frac{-x_j}{x_i - x_j}$

$$q(x) = \sum_{i \in P} y_i \prod_{\substack{j \in P \\ i \neq j}} \frac{x - x_j}{x_i - x_j}$$

Then, $s = q(0)$.

2.3.3 Anonymous secret sharing

2.4 Digital Signatures

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

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**: Only a user can sign messages on behalf of himself.
- **Non-repudiation**:

2.4.1 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 $c = H(L^\dagger \parallel m)$

2.5 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:

2.6 Threshold Digital Signatres

Definition and example (shamir)

2.6.1 Shamir

This signature scheme is based on the BLS scheme (see section 2.4.1).

Setup Algorithm

- Let $\mathcal{P} = \{P_1, \dots, P_n\}$ 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 $e(\sigma, g) = e(H(m), pk)$

2.7 Bilinear pairings

[DH76]

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 2.7.1. 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 2.7.2. 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 2.7.3. 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 2.7.4. 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 2.7.5. 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 2.7.6. 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 2.7.7. 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.

CHAPTER 3

ANONYMITY IN THRESHOLD SIGNATURES

Blabla Shamir is not anonymous

Single Use

Multiple use

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.

It is not very intuitive how we can punt the concept of anonymity in a signature scheme. An anonymous signature does not make much sense. There is no use of an information signed by an anonymous person. What is useful is a signature from a known group of people, but cannot determine from which one nor distinguish two signatures of different participants of the same group on the same message.

- Absolute anonymity. No use. Same as no signature - Anonymity inside a group.

You could use ring signatures which proofs the knowledge of a 1-out-of-N secret key. This could be useful for a small amount of signatures. But it is not useful to provide general anonymity.

A signature scheme provides anonymity if:

- 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.

[BS97]

An ideal secret sharing scheme is a scheme in which the size of the shares given to each participant is equal to the size of the secret.

In an anonymous secret sharing schemes the secret can be reconstructed without the knowledge of which participants hold which shares.

CHAPTER 4

SINGLE USE: ANONYMITY

To identify someone is to find his public key.

We can easily achieve anonimity using "pseudonyms", but usually there is a dealer which is the trusted party. So, the security relies on a single party.

The idea:

Let there be a votation with l choices $Z = z_1, \dots, z_l$. The candidate z_i needs at least t votes to be validated.

The system chooses a random polynomial $q_i(x) = a_{0,i} + a_{1,i}x + \cdots + a_{k-1,i}x^{k-1}$ of degree $k-1$ for each candidate, where $a_{0,i} = SK_i$ is the secret that validates each candidate.

CHAPTER 5

MULTIPLE USE: ANONYMITY AND NON-TRACEABILITY

To achieve full anonymity we need some Multiple use

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 2.6.1 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 2.6.1.

– 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*

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

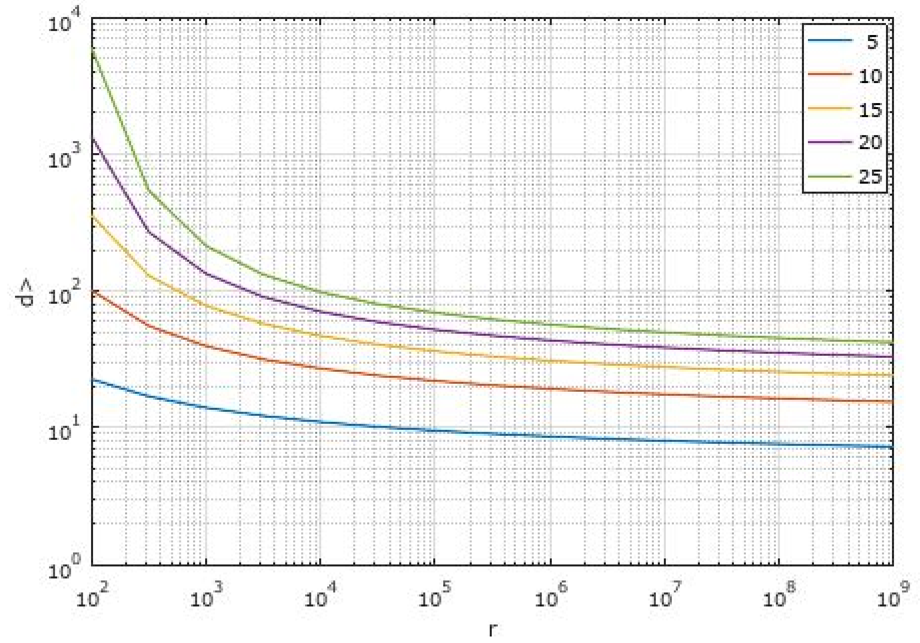


Figure 5.1: Values for the lower bound of d where $\frac{n}{r} = 10^3$

Table with sample values.

To be anonymous, we need $n \geq 2r$.

These schemes would be suitable for small values of t.

5.2 Linkable Group Signature Scheme

[CNW11]

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

This is a protocol between a given signer $s \in S$ and an issuer $i \in \mathcal{I}$.

(Maybe could be a random $f \in G_1$) The signer generates a secret value f using its internal seed **TAAseed**, along with the value $\mathbf{K}_{\mathrm{I}}$ provided by i and a count number **cnt**.

Signer($\mathfrak{s}$)		Issuer($\mathfrak{i}$)
$f \in_R \mathbb{Z}_q,\ F = g_1^f$		
$\text{str} \leftarrow X \parallel Y \parallel n_I$	$\xleftarrow{\text{comm}_{\text{req}}}$	$\text{comm}_{\text{req}} \leftarrow n_I$
	$\longrightarrow$	If $F = g_1^{f_i}$ for any f_i on the roge list then **abort**
		$r \in_R \mathbb{Z}_q;\ A = g_1^r;\ B = A^y$
If $\hat{t}(A, Y) \neq \hat{t}(B, g_2)$	$\xleftarrow{\text{cre}}$	$C = (A^x \cdot F^{rxy});\ cre \leftarrow (A, B, C)$
or $\hat{t}(A \cdot B^f, X) \neq \hat{t}(C, g_2)$		
then **abort**		

Figure 5.2: The Join Protocol

```
Signer($\mathfrak{s}$)
Input: $f \in \mathbb{Z}_q$; $n_T \leftarrow \{0,1\}^*$; $msg$
$a \in_R \mathbb{Z}_q$; $z \in_R \mathbb{Z}_q$
$J \leftarrow H_1(\text{msgt})$; $K = J^f$; $L = J^z$
$R = A^a$; $S = B^a$; $T = C^a$; $\tau = \hat{t}(S, X)^z$
$c \leftarrow H_0(R \parallel S \parallel T \parallel \tau \parallel J \parallel K \parallel L \parallel n_T \parallel \text{msgb})$
$s \leftarrow z + c \cdot f \mod q$
$\sigma \leftarrow (R, S, T, J, K, c, s, n_T)$
Output: $\sigma$
```

Figure 5.3: The Sign Algorithm

```
Verifier($\mathfrak{v}$)
Input: $\mathbf{ipk}_k = (X, Y)$; $\text{msg} = (\text{msgt}, \text{msgb})$
$\sigma = (R, S, T, J, K, c, s, n_T)$
If $K = J^{f_i}$, for any $f_i$in the set of rogue secret keys, or
    $\hat{t}(R, Y) \neq \hat{t}(S, g_2)$, or
    $J \neq H_1(\text{msgt})$ return **reject**
$\rho_a^\dagger = \hat{t}(R, X); \rho_b^\dagger = \hat{t}(S, X); \rho_c^\dagger = \hat{t}(T, g_2)$
$\tau^\dagger = (\rho_b^\dagger)^s \cdot (\rho_c^\dagger / \rho_a^\dagger)^{-c}$
$L^\dagger = J^s \cdot K^{-c}$
If $c \neq H_0(R \| S \| T \| \tau^\dagger \| J \| K \| L^\dagger \| n_T \| \text{msgb})$ return **reject**
Otherwise return **accept**
```

Figure 5.4: The Verify Algorithm

In the Sign Algorithm (Fig. 5.3), the computation of L, c and s is for the non-interactive Zero-knowledge proof of knowledge of f, and τ is to proof the knowledge of the issuer's credentials. This is based on the *Schnorr protocol* described in [Sch90]

To see if two signatures with same message title were signed by the same signer, only have to check whether $J_0 = J_1$ and $K_0 = K_1$.

In the case of using this scheme in a polling system, **msgt** could be used to identify the poll and **msgb** for the poll choice.

5.3 Anonymous interactive protocol

We propose an interactive protocol based on the signature scheme described in section 2.6.1. 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 **¿previous?** 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 $P_j \in P$. ~~The protocol needs t^2 interactions as the one described in figure ?? to be able to compute the whole signature.~~

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. The description follows:

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 $z_i(0) = z_j(0) = 0$.

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.

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 $h(0)$.

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 $a^{\frac{g_i(0)+g_j(0)}{h(0)}} = a^{\frac{v}{v(\alpha_i-\alpha_j)}} = a^{\frac{1}{\alpha_i-\alpha_j}}$.

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

Partial signature

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

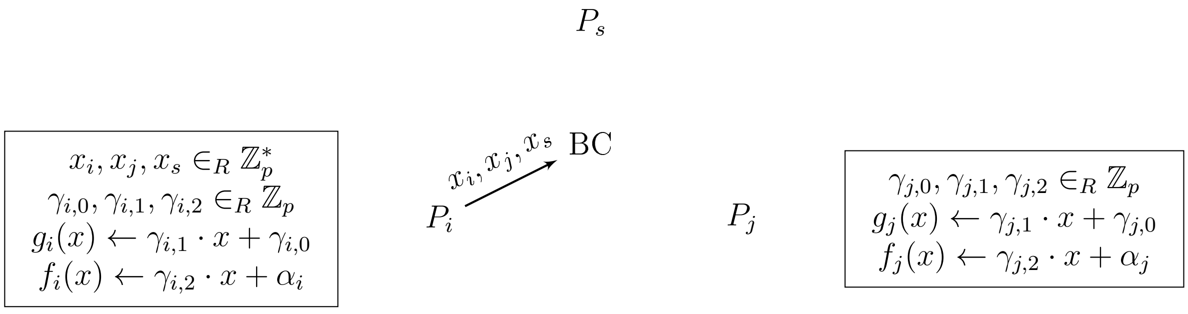

P_s
$x_i, x_j, x_s \in_R \mathbb{Z}_p^*$
$\gamma_{i,0}, \gamma_{i,1}, \gamma_{i,2} \in_R \mathbb{Z}_p$
$g_i(x) \leftarrow \gamma_{i,1} \cdot x + \gamma_{i,0}$
$f_i(x) \leftarrow \gamma_{i,2} \cdot x + \alpha_i$
x_i, x_j, x_s
BC
P_i
P_j
$\gamma_{j,0}, \gamma_{j,1}, \gamma_{j,2} \in_R \mathbb{Z}_p$
$g_j(x) \leftarrow \gamma_{j,1} \cdot x + \gamma_{j,0}$
$f_j(x) \leftarrow \gamma_{j,2} \cdot x + \alpha_j$

Figure 5.5: Step 1

$$\boxed{\begin{array}{c}\text{For } k \in \{i, j, s\} \\ g_{ik} \leftarrow g_i(x_k) \\ f_{ik} \leftarrow f_i(x_k) \\ z_{ik} \leftarrow z_i(x_k)\end{array}}$$

$$\begin{tikzcd}
 & P_s & \\
 & \text{BC} & \\
P_i \arrow[uur, "{g_{is}, f_{is}, z_{is}}"] \arrow[rr, bend left, "{g_{ij}, f_{ij}, z_{ij}}"] & & P_j \arrow[uul, "{g_{js}, f_{js}, z_{js}}"'] \arrow[ll, bend left, "{g_{ji}, f_{ji}, z_{ji}}"]
\end{tikzcd}$$

$$\boxed{\begin{array}{c}\text{For } k \in \{i, j, s\} \\ g_{jk} \leftarrow g_j(x_k) \\ f_{jk} \leftarrow f_j(x_k) \\ z_{jk} \leftarrow z_j(x_k)\end{array}}$$

Figure 5.6: Step 2

$$\boxed{h_s \leftarrow (g_{is} + g_{js})(f_{is} - f_{js}) + z_{is} + z_{js}}$$

$$\begin{tikzcd}
 & P_s \arrow[d, "h_s"] & \\
 & \text{BC} & \\
P_i \arrow[ur, "h_i"'] & & P_j \arrow[ul, "h_j"]
\end{tikzcd}$$

$$\boxed{\begin{array}{c} h_i \leftarrow (g_{ii} + g_{ji})(f_{ii} - f_{ji}) + z_{ii} + z_{ji} \\ h \leftarrow \sum_{k \in \{i,j,s\}} h_k \prod_{\ell \neq k} \frac{-x_\ell}{x_k - x_\ell} \end{array}}$$

$$\boxed{\begin{array}{c} h_j \leftarrow (g_{ij} + g_{jj})(f_{ij} - f_{jj}) + z_{ij} + z_{jj} \\ h \leftarrow \sum_{k \in \{i,j,s\}} h_k \prod_{\ell \neq k} \frac{-x_\ell}{x_k - x_\ell} \end{array}}$$

Figure 5.7: Step 3

$$\boxed{A_i \leftarrow a^{\frac{1}{h}(g_{ii} + g_{ji})}} \qquad P_i \xrightarrow{A_i} P_j \qquad \boxed{\begin{array}{c} A_j \leftarrow a^{\frac{1}{h}(g_{ij} + g_{jj})} \\ B \leftarrow A_i^{\frac{-x_j}{x_i - x_j}} A_j^{\frac{-x_i}{x_j - x_i}} \end{array}}$$

Figure 5.8: Step 4

$$\boxed{\text{Output: } B'} \qquad P_i \xleftarrow{B'} P_j \qquad \boxed{B' \leftarrow B^{-\alpha_j}}$$

Figure 5.9: Step 5

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 $a_0 = H(m)^{s_i}$

Signature

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

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

.

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

5.3.2 Analysis

Correctness

Interaction:

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.

[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.

[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.