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, October 2017

Abstract

Resum

Resum

Resum

Resumen

Resum

ACKNOWLEDGEMENTS

Acknowledgements

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Intro

CHAPTER 2

PRELIMINARIES

Some cryptographic preliminaries and other definitions.

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 2.1.1. 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 2.1.2. Decision Diffie-Hellman (CDH) 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 2.1.3. 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 2.1.4. 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 2.1.5. 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 2.1.6. A gap group is a group on which the DDH problem is easy but the CDH is hard.

Homomorphic PKE

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 length), 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, ouptuts 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**:

Examples

ElGamal

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 $\mathbb{Z}_p^*$

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

2.3.1.1.2 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.

2.3.1.1.3 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, G_T be (¿gap?) groups of prime order p. Let g be a generator of G. Let $e : G \times G \to G_T$ be a non-degenerate bilinear pairing.

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

2.3.1.2.2 Signature The signature on m is $\sigma = H(m)^x$.

2.3.1.2.3 Verification Given a signature σ and a public key g^x, it verifies that $e(\sigma, g) = e\left(H(m), g^x\right)$.

Signature Aggregation

Explain how different signature schemes allow aggregation of n signatures on n messages from n signers.

Group Signatures

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

The easyest 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:

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, $s = q(0)$.

Threshold signature scheme using Shamir

This signature scheme will be based on the BLS scheme.

Let G be a gap group of some prime order p. Let $g \in G$ be a generator of the group.

Let $\mathcal{P} = \{1, \ldots, n\}$ be the set of participants in this scheme. Supose every participant $P_i \in \mathcal{P}$ is given a distinct random number $x_i \in \mathbb{Z}_p$ as in the Shamir Secret Sharing Scheme.

Let $SK \in \mathbb{Z}_p$ be the secret key of the scheme. Let $q(x) = \sum_{i=0}^{k-1} a_i x^i$ be a random polynomial of degree $k-1$ but fixing $a_0 = SK$.

Each participant P_i is given the share $s_i = q(x_i)$.

So

$$SK = \sum_{i \in P} s_i \lambda_i^P$$

To sign a message m, each participant P_i computes his partial signature $\sigma_i(m) = H(m)^{s_i}$ and broadcasts de pair $(x_i,\ \sigma_i(m))$.

Then, after a set P of at least t participants has broadcast their partial signatures for the message, a standard signature σ can be computed:

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

The signature is valid if $e(\sigma, g) = e(H(m), PK)$

Anonymity

The concept of anonimity is used in so many ways

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.

CHAPTER 3

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 $Z = z_1, \ldots, z_l$. 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 4

MULTIPLE USE: ANONYMITY AND NON-TRACEABILITY

To achieve full anonymity we need some Multiple use [CNW11] [DDSV09]

Setup Algorithm

G_1, G_2, G_T 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 perfomed.

Two integers are selected $x, y \in_R \mathbb{Z}_q$ and the isuer 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).

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

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

Figure 4.1: The Join Protocol

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

Figure 4.2: The Sign Algorithm

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

Figure 4.3: The Verify Algorithm

In the Sign Algorithm (Fig. 4.2), 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.

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.

Anonymous interactive protocol

We propose an interactive protocol based on the signature scheme described in section 2.7. Recall that this scheme does not have the *unlinkability* property because it uses "pseudonyms" and they are shared to compute the signature. So, the idea of this new protocol is to hide the "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.

In the following description, the partial signature will be $\sigma_i(m) = H(m)^{s_i \prod_{P_j \in (P \setminus P_i)} \frac{-\alpha_j}{\alpha_i - \alpha_j}}$. So the goal of this improvement is to compute $a^{\frac{-\alpha_j}{\alpha_i - \alpha_j}}$, for $1 \neq a \in G$, and a given participant $P_j \in P$ without sharing the values of α_i and α_j. The protocol needs t^2 interactions as the one described in figure 4.4 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 any other third participant P_k.

P_i chooses $x_i, x_j, x_k \in \mathbb{Z}_p^*$ three random values and shares them with P_j and P_k.

P_i and P_j randomly choose linear polynomials f_i, g_i and f_j, g_j, respectively, where the values of α_i and α_j are hidden in f_i and f_j, respectively. As they share with each other the values of the polynomials on respective values x_i, x_j, x_k, each can compute the value of the quadratic polynomial $(g_i + g_j) \cdot (f_i - f_j)$ on a certain value and share it with P_i. Then, P_i can compute the constant term of the quadratic polynomial $v \cdot (\alpha_i - \alpha_j)$ for some $v \in \mathbb{Z}_p$.

Now, P_i computes $A := a^{\frac{1}{v(\alpha_i - \alpha_j)}}$ and shares it with P_j. They compute $A^{(g_i+g_j)(x_i)}$ and $A^{(g_i+g_j)(x_j)}$, resp., and they share it. With the values x_i, x_j; P_j can do exponent interpolation and get $a^{\frac{(g_i+g_j)(0)}{v(\alpha_i - \alpha_j)}} = a^{\frac{v}{v(\alpha_i - \alpha_j)}} = a^{\frac{1}{\alpha_i - \alpha_j}}$, and finally compute $a^{\frac{-\alpha_j}{\alpha_i - \alpha_j}}$

Let $\{P_{j_1}, \ldots, P_{j_{t-1}}\} = P \setminus P_i$, and let $a_k = a_{k-1}^{\frac{-\alpha_{j_{k-1}}}{\alpha_i - \alpha_{j_{k-1}}}}$, where $a_1 = H(m)^{s_i}$. To compute the partial signature on m, P_i interacts with each P_{j_k} to compute $a_k^{\frac{-\alpha_k}{\alpha_i - \alpha_k}}$. Finally, we have:

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

P_i		P_j		P_k
Input: $a \in G$				
$x_i, x_j, x_k \in_R \mathbb{Z}_p \setminus \{0\}$	$\xrightarrow{x_i, x_j, x_k}$			
			$\xrightarrow{x_i, x_j, x_k}$	
$\gamma_{i,0}, \gamma_{i,1}, \gamma_{i,2} \in_R \mathbb{Z}_p$ $\gamma_{i,1}, \gamma_{i,2} \neq 0$		$\gamma_{j,0}, \gamma_{j,1}, \gamma_{j,2} \in_R \mathbb{Z}_p$ $\gamma_{j,1}, \gamma_{j,2} \neq 0$		
$g_i(x) \leftarrow \gamma_{i,1} \cdot x + \gamma_{i,0}$ $f_i(x) \leftarrow \gamma_{i,2} \cdot x + \alpha_i$		$g_j(x) \leftarrow \gamma_{j,1} \cdot x + \gamma_{j,0}$ $f_j(x) \leftarrow \gamma_{j,2} \cdot x + \alpha_j$		
$g_i(x_j), f_i(x_j)$	$\xrightarrow{g_i(x_j), f_i(x_j)}$			
$g_i(x_k), f_i(x_k)$			$\xrightarrow{g_i(x_k), f_i(x_k)}$	
	$\xleftarrow{g_j(x_i), f_j(x_i)}$	$g_j(x_i), f_j(x_i)$		
		$g_j(x_k), f_j(x_k)$	$\xrightarrow{g_j(x_k), f_j(x_k)}$	
$h_i \leftarrow ((g_i + g_j)(f_i - f_j))(x_i)$				
	$\xleftarrow{h_j}$	$h_j \leftarrow ((g_i + g_j)(f_i - f_j))(x_j)$		
			$\xleftarrow{h_k}$	$h_k \leftarrow ((g_i + g_j)(f_i - f_j))(x_k)$
$v(\alpha_i - \alpha_j) :=$ $(\gamma_{i,0} + \gamma_{j,0})(\alpha_i - \alpha_j) \leftarrow h_i, h_j, h_k$				
$A \leftarrow a^{\frac{1}{v(\alpha_i - \alpha_j)}}$				
$A^{G_i} \leftarrow A^{g_i(x_i) + g_j(x_i)}$	$\xrightarrow{A, A^{G_i}}$	$A^{G_j} \leftarrow A^{g_i(x_j) + g_j(x_j)}$		
		$A^{G_i \frac{-x_j}{x_i - x_j}} A^{G_j \frac{-x_i}{x_j - x_i}} = A^{g_i(0) + g_j(0)} =$		
$a^{\frac{-\alpha_j}{\alpha_i - \alpha_j}}$	$\xleftarrow{a^{\frac{-\alpha_j}{\alpha_i - \alpha_j}}}$	$a^{\frac{v}{v(\alpha_i - \alpha_j)}} = a^{\frac{1}{\alpha_i - \alpha_j}}$		
Output: $a^{\frac{-\alpha_j}{\alpha_i - \alpha_j}}$				

Figure 4.4: The Proposed Protocol

CHAPTER 5

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.

[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.

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