Extensible Decentralized Secret Sharing and Schnorr Signatures

Michele Battagliola[1] Riccardo Longo[2] Alessio Meneghetti[3]

1. Marche Polytechnic University battagliola.michele@proton.me
2. Fondazione Bruno Kessler, Center for Cybersecurity rlongo@fbk.eu
3. University of Trento alessio.meneghetti@unitn.it

International Workshop on Coding and Cryptography (WCC)
June 17th, 2024

Outline of the Talk

Introduction & Motivation

Our Contribution: Extensible DKG

Application: Threshold Schnorr

Secure Secret Sharing (SSS)

- In cryptography, security comes from **secret** keys
- Key management is of paramount importance
- *Secure Secret Sharing* allows to enhance:
  - **security**: no single party has the whole secret
  - **resiliency**: some shares can be lost without compromising the whole secret

Secure Multi-Party Computation (SMPC)

- Semi-collaborative environment with multiple parties
  - some may be **malicious**
- **Goal**: correctly compute a function over the parties' inputs while keeping those inputs private
- SSS is a widely exploited tool in SMPC

Decentralized Computation

Goals:

- distribute trust
- avoid *single points of failure*
  - for both security and availability

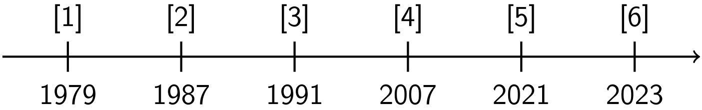
Decentralized Key Generation

- no single party controls the key
- key never reconstructed, only shares used within SMPC

Decentralized Key Generation: Timeline (1)

[1] Original Shamir's Secret Sharing

[2] Feldman's VSS: add verifiability of shares



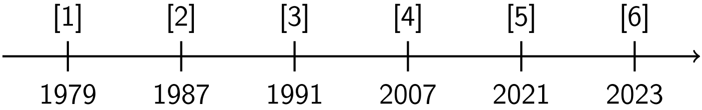
[1] Adi Shamir. "How to share a secret". in *Communications of the ACM*: 22.11 (1979), pages 612–613.

[2] Paul Feldman. "A practical scheme for non-interactive verifiable secret sharing". in *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*: IEEE. 1987, pages 427–438.

Decentralized Key Generation: Timeline (2)

[3] 2-round DKG, each participant acts as a dealer in a Feldman VSS protocol

[4] demonstrates a weakness in [3], proposes a secure variant by adding a commitment step (3-round DKG)



---

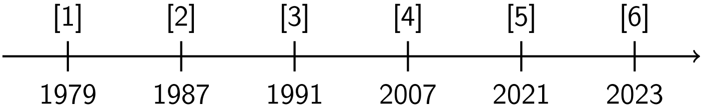
[3] Torben Pryds Pedersen. "A threshold cryptosystem without a trusted party". in*Advances in Cryptology—EUROCRYPT'91: Workshop on the Theory and Application of Cryptographic Techniques Brighton, UK, April 8–11, 1991 Proceedings 10*: Springer. 1991, pages 522–526.

[4] Rosario Gennaro andothers. "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems". in*J. Cryptology*: 20 (2007), pages 51–83. DOI: 10.1007/s00145-006-0347-3.

Decentralized Key Generation: Timeline (3)

[5] (PedPop) in FROST is used a variant of Pedersen DKG where each participant prove the knowledge of their key by using a ZKP

[6] (Simpl.PedPoP) variant of PedPoP where dishonest participants trigger the abort of the protocol



[5] Chelsea Komlo and Ian Goldberg. "FROST: Flexible Round-Optimized Schnorr Threshold Signatures". in*Selected Areas in Cryptography*: byeditorOrr Dunkelman, Michael J. Jacobson Jr. and Colin O'Flynn. Cham: Springer International Publishing, 2021, pages 34–65.

[6] Hien Chu andothers. "Practical schnorr threshold signatures without the algebraic group model". in*Annual International Cryptology Conference*: Springer. 2023, pages 743–773.

Outline of the Talk

Introduction & Motivation

Our Contribution: Extensible DKG

Application: Threshold Schnorr

Extensibility for more resilience

- New shares can be created after generation
- Any threshold of parties can collaborate to create new shares
- New shares are verifiably compatible with old ones
- Lost shares can be reconstructed

Extensible Decentralized Secret Sharing

- n shares created initially
- $t \leq n$ shares required to add further shares (or reconstruct the secret)
- public values for checking share correctness

Definition (Homomorphic Commitment)

A commitment HCom such that $\forall m_0, m_1, z_0, z_1, \gamma \in \mathbb{F}_q$:

$$\mathsf{HCom}(m_0; z_0) \cdot \mathsf{HCom}(m_1; z_1) = \mathsf{HCom}(m_0 + m_1; z_0 + z_1),$$
$$\mathsf{HCom}(m_0; z_0)^{\gamma} = \mathsf{HCom}(\gamma \cdot m_0; \gamma \cdot z_0).$$

E.g.: Pedersen commitment

SecGen(pp)

$p^{(i)} \xleftarrow{\$} \mathbb{F}_q[x], \deg(p^{(i)}) = t - 1$

$z^{(i)} \xleftarrow{\$} \mathbb{F}_q[x], \deg(z^{(i)}) = t - 1$

Publish $\mathtt{C}_{0,i,k} := \mathsf{HCom}(p_k^{(i)}; z_k^{(i)})$

Send to P_j $\beta_{i,j} := p^{(i)}(\alpha^j)$, $\gamma_{i,j} := z^{(i)}(\alpha^j)$

// To all $j \in [n]$

if $\mathsf{HCom}(\beta_{j,i}; \gamma_{j,i}) \neq \prod_{k=0}^{t-1} (\mathtt{C}_{0,j,k})^{(\alpha^i)^k}$ then

return $\perp$

$\beta_i := \sum_{j=1}^{\tau} \beta_{j,i}; \quad \gamma_i := \sum_{j=1}^{\tau} \gamma_{j,i}$

return β_i, γ_i

Addition of New Parties

$\mathsf{PAdd}(\mathbb{J}, \{\beta_i\}_{\mathbb{J}}, \{\gamma_i\}_{\mathbb{J}})$

$b_{n+1,\mathbb{J},i,j}, z_{n+1,\mathbb{J},i,j} \xleftarrow{\$} \mathbb{F}_q,\ \ k \in [t], j \neq i$

$b_{n+1,\mathbb{J},i,i} := f(\beta_i, n+1, \mathbb{J}, i) - \sum_{j=1, j\neq i}^{t} b_{n+1,\mathbb{J},i,j}$

$z_{n+1,\mathbb{J},i,i} := f(\gamma_i, n+1, \mathbb{J}, i) - \sum_{j=1, j\neq i}^{t} z_{n+1,\mathbb{J},i,j}$

// $f(x, h, \mathbb{J}, \ell) = x \cdot e_\ell G_{\mathbb{J}}^{-1} G_h$, G MDS generator matrix

Publish $\mathtt{C}_{n+1,\mathbb{J},i,j} = \mathsf{HCom}\left(b_{n+1,\mathbb{J},i,j}; z_{n+1,\mathbb{J},i,j}\right)$

if $\prod_{j=1}^{t} \mathtt{C}_{n+1,\mathbb{J},i,j} \neq \left(\prod_{j=0}^{t-1}\left(\prod_{k=1}^{\tau} \mathtt{C}_{0,k,j}\right)^{(\alpha^i)^j}\right)^{e_i G_{\mathbb{J}}^{-1} G_{n+1}}$ then

  return $\perp$

if $\prod_{j=1}^{t} \prod_{i=1}^{t} \mathtt{C}_{n+1,\mathbb{J},i,j} \neq \prod_{j=0}^{t-1}\left(\prod_{k=1}^{\tau} \mathtt{C}_{0,k,j}\right)^{(\alpha^{n+1})^j}$ then

  return $\perp$

Send to P_j $b_{n+1,\mathbb{J},i,j}$ and $z_{n+1,\mathbb{J},i,j}$

if $\mathsf{HCom}\left(b_{n+1,\mathbb{J},j,i}; z_{n+1,\mathbb{J},j,i}\right) \neq \mathtt{C}_{n+1,\mathbb{J},k,i}$ then

  return $\perp$

$b_{n+1,\mathbb{J},i} := \sum_{j=1}^{t} b_{n+1,\mathbb{J},j,i} \quad z_{n+1,\mathbb{J},i} := \sum_{j=1}^{t} z_{n+1,\mathbb{J},j,i}$

Send to P_{n+1} $b_{n+1,\mathbb{J},i}$ and $z_{n+1,\mathbb{J},i}$

Security

Theorem (Security of Secret Generation)

If HCom *is **binding** then the Secret Generation is **correct**, if* HCom *is **hiding** then the Secret Generation is **secure**.*

Definition (Security of Addition of New Parties)

Let $S \subseteq \{1, \dots, t, n+1\}$ such that $|S| = t-1$, $\texttt{view}_S$ the messages seen by parties in S when adding a new party. Addition of New Parties is secure if and only if:

$$\mathbb{P}(P_i \text{ has secret } \omega_i | \texttt{view}_S) = \mathbb{P}(P_i \text{ has secret } \omega_i), \quad i \notin S$$
$$\mathbb{P}(\text{The shared secret is } p_0 | \texttt{view}_S) = \mathbb{P}(\text{The shared secret is } p_0).$$

Theorem

If HCom *is **hiding**, then the Addition of New Parties is secure.*

Outline of the Talk

Introduction & Motivation

Our Contribution: Extensible DKG

Application: Threshold Schnorr

Threshold Signatures

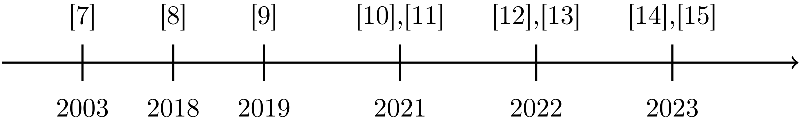
- Drop-in replacement of the original (centralized) signature
  - Same verification algorithm
  - Signature indistinguishable from a centralized one
- Application: custody of crypto-assets
  - In 2020 Bitcoin added support to **Schnorr Signatures**

Threshold Schnorr Signatures: Timeline (1)

[7] first decentralized version of Schnorr

[8] and [9] improvements

all of these only work in the (n, n) setting



---

[7] Antonio Nicolosi andothers. "Proactive Two-Party Signatures for User Authentication". in*Proceedings of the Network and Distributed System Security Symposium, NDSS 2003, San Diego, California, USA*: The Internet Society, 2003. URL: `https://www.ndss-symposium.org/ndss2003/proactive-two-party-signatures-user-authentication/`.

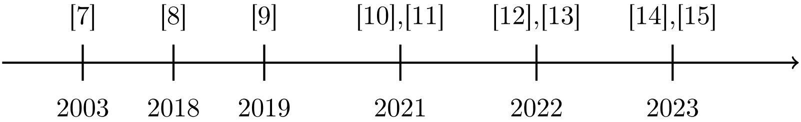
[8] Dan Boneh, Manu Drijvers and Gregory Neven. "Compact Multi-signatures for Smaller Blockchains". in*Advances in Cryptology – ASIACRYPT 2018*: **byeditor**Thomas Peyrin **and** Steven Galbraith. Cham: Springer International Publishing, 2018, **pages** 435–464. ISBN: 978-3-030-03329-3.

[9] Gregory Maxwell andothers. "Simple Schnorr multi-signatures with applications to Bitcoin". in*Designs, Codes and Cryptography*: 87 (**september** 2019). DOI: `10.1007/s10623-019-00608-x`.

Threshold Schnorr Signatures: Timeline (2)

[10] (FROST) first (t, n)-threshold Schnorr, non-classical security assumptions (AOMDL)

[11] (FROST2) variant of FROST that optimizes the number of exponentiations when signing, security still with AOMDL



[10] Chelsea Komlo and Ian Goldberg. "FROST: Flexible Round-Optimized Schnorr Threshold Signatures". in*Selected Areas in Cryptography*: byeditorOrr Dunkelman, Michael J. Jacobson Jr. and Colin O'Flynn. Cham: Springer International Publishing, 2021, pages 34–65.

[11] Elizabeth Crites, Chelsea Komlo and Mary Maller. "How to prove schnorr assuming schnorr: Security of multi-and threshold signatures". in*Cryptology ePrint Archive*: (2021).

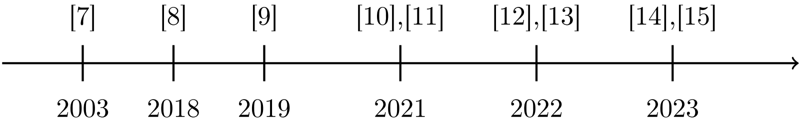
Threshold Schnorr Signatures: Timeline (3)

[12] (FROST3), further optimisation of FROST

[13] pre-print of our work

[14] adds improved distributed key-generation to FROST3

[15] (SPARKLE+) very similar to [13], focus on adaptive corruption of parties after key generation (with standard assumptions)



[12] Tim Ruffing andothers. "ROAST: Robust asynchronous Schnorr threshold signatures". in*Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*: 2022, pages 2551–2564.

[13] Michele Battagliola, Riccardo Longo and Alessio Meneghetti. *Extensible Decentralized Secret Sharing and Application to Schnorr Signatures*. Cryptology ePrint Archive, Paper 2022/1551. https://eprint.iacr.org/2022/1551. 2022. URL: https://eprint.iacr.org/2022/1551.

[14] Hien Chu andothers. "Practical schnorr threshold signatures without the algebraic group model". in*Annual International Cryptology Conference*: Springer. 2023, pages 743–773.

[15] Elizabeth Crites, Chelsea Komlo and Mary Maller. "Fully Adaptive Schnorr Threshold Signatures". in*Advances in Cryptology – CRYPTO 2023*: byeditorHelena Handschuh and Anna Lysyanskaya. Cham: Springer Nature Switzerland, 2023, pages 678–709. ISBN: 978-3-031-38557-5.

Threshold Schnorr Signature Generation (1)

Preliminary: nonce shares generation and commitment

$$\underline{\mathsf{TSign}(\{w_i\}_{i\in\mathbb{S}}; \mathsf{R}) \to \{\mathtt{cmt}_i\}_{i\in\mathbb{S}}}$$

1 : $r_i \xleftarrow{\$} \mathbb{Z}_p$

2 : $R_i \leftarrow g^{r_i}$

3 : $\mathtt{cmt}_i \leftarrow \mathsf{Com}(\mathbb{S}, R_i)$

4 : return $\mathtt{cmt}_i$

Threshold Schnorr Signature Generation (2)

Nonce shares decommitment and challenge computation

$\mathsf{TSign}'(\{w_i\}_{i\in\mathbb{S}}, \{\mathtt{cmt}_i\}_{i\in\mathbb{S}}) \to \mathtt{ch}$

Party P_i sends R_i

if $1 \neq \mathsf{Ver}(\mathtt{cmt}_j, \mathbb{S}, R_j), j \in \mathbb{S}$

  return $\perp$

$$R = \prod_{i\in\mathbb{S}} R_i$$

$\mathtt{ch} \leftarrow \mathsf{H}(\mathtt{m}||R)$

return $\mathtt{ch}$

Threshold Schnorr Signature Generation (3)

Output of signature

$$\mathsf{TSign}''(\{w_i\}_{i\in\mathbb{S}}, \mathtt{ch}) \rightarrow (\mathtt{rsp})$$

$z_i \leftarrow r_i + \lambda_i \cdot w_i \cdot \mathtt{ch}$

// λ_i `is the Lagrange coefficient`

Party P_i sends z_i

$$z \leftarrow \sum_{i\in\mathbb{S}} z_i$$

return $(R, z) \leftarrow \sigma$

Threshold Schnorr Signature Verification

Identical to centralized Schnorr

$\mathsf{Ver}(y, \sigma) \to 0/1$

Parse $(R, z) \leftarrow \sigma$
if $Ry^{\mathtt{ch}} = g^{z}$ then
    return accept
return reject

Security (1)

Definition (Unforgeability)

A (t, n)-threshold signature scheme is **unforgeable** if no malicious adversary who **corrupts at most** $\mathrm{t}-1$ **players** can produce the **signature on a new message** m **with non negligible probability**, given the **view** of the threshold sign on input messages $m_1, \ldots, m_Q$ (**adaptively chosen** by the adversary), as well as the **signatures** on those messages.

Theorem

*Assuming that the **Schnorr** signature scheme instantiated on the group $\mathbb{G}$ of prime order q with the hash function H is **unforgeable**,* $\mathrm{Com}, \mathrm{Ver}$ *is a **non-malleable commitment** scheme, and that the **Decisional Diffie-Hellman** Assumption holds, then our threshold protocol is **unforgeable**.*

Security (2)

- Adversary can corrupt parties **before key generation**
- In SPARKLE+ the corruption may begin only after key shares are already distributed
- Complementary proofs: security all-around

Thank you for your attention!

Any questions?

`rlongo@fbk.eu`