An overview of (some) post quantum threshold schemes

Giacomo Borin - IBM Research Zurich & University of Zurich
CrypTOgraphy Days - 22.05.2026 – Castello del Valentino - Torino

NIST

Information Technology Laboratory

COMPUTER SECURITY RESOURCE CENTER

PUBLICATIONS

NIST IR 8214C

NIST First Call for Multi-Party Threshold Schemes

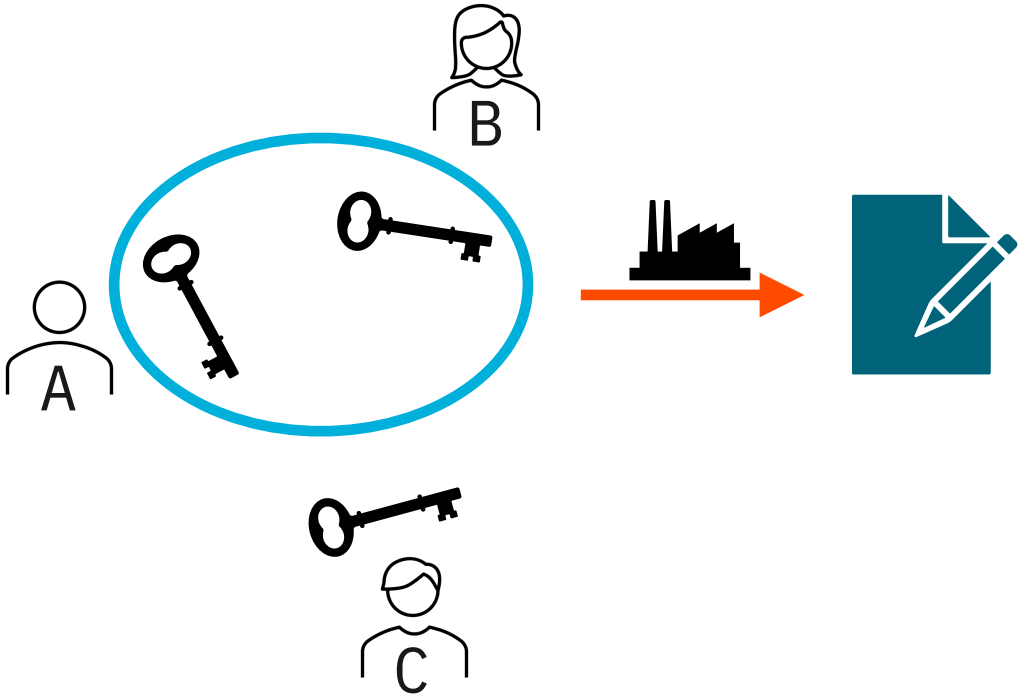
Date Published: January 2026

Author(s)

Luís T. A. N. Brandão (NIST, Strativia), Rene Peralta (NIST)

https://csrc.nist.gov/Projects/threshold-cryptography/tcall-1
https://csrc.nist.gov/pubs/ir/8214/c/final

What is threshold cryptography?



Some applications:

- Shared wallets
- Digital identity
- Encrypted mempools
- Integration in TLS
- More…

FROST2 (warm up)

Komlo C, Goldberg I. FROST: Flexible round-optimized Schnorr threshold signatures.
Crites E, Komlo C, Maller M. How to prove schnorr assuming schnorr: Security of multi- and threshold signatures

pp : E = <P> group of size p
[x]P : multiplication of P by x

Random combination of points

Preprocess(i)

1. $e_i, s_i \leftarrow \mathbb{Z}_p$
2. $E_i, S_i \leftarrow [e_i]P, [s_i]P$
3. out $pp_i = (E_i, S_i)$

Combine$(m, (z_j)_j)$

1. get R as in Sign
2. out $(c, \Sigma z_j)$

Sign$(m, (pp_j)_j)$

1. $r = H(vk, m, (pp_i)_i)$
2. $R = \Sigma E_i + [r]S_i$
3. $c = H(R, m)$
4. out $z_i = e_i + r_i s_i + c \lambda_i sk_i$

Verify(m,z,pk)

1. $c' = H([z]P - [c]pk, m)$
2. out $c == c'$

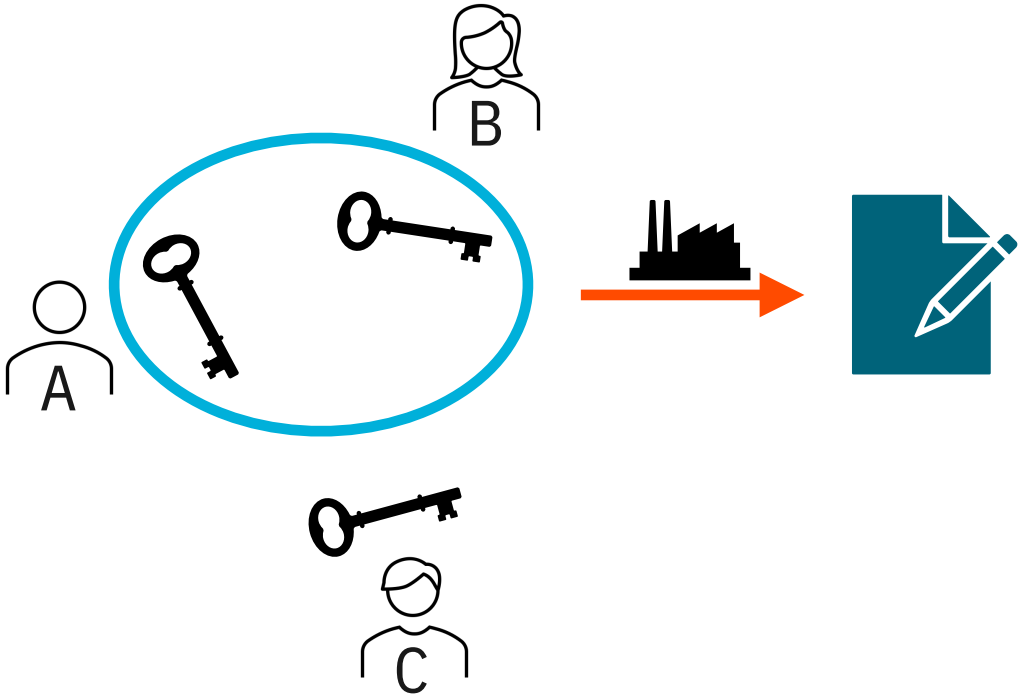
$c \lambda_i$ are Lagrange coefficients:
$\Sigma \lambda_i sk_i = sk$

What can FROST do

- 👍 Partially non interactive threshold signing
- 👍 Stronger unforgeability guarantees
- 👍 Verification of pre-signatures
- 👍 Distributed Key Generation
- 👍 Proactive Refresh
- 👍 Scalability to large thresholds
- ⚙️ Extensive research on adaptive security

Universität Zürich[UZH]

What is threshold cryptography?



Some applications:

- Shared wallets
- Digital identity
- Encrypted mempools
- Integration in TLS
- More…

NIST First Call for Multi-Party Threshold Schemes

The NIST Threshold Call is expected to motivate broad community engagement, to result in a diverse set of high-quality submission packages that propose the consideration of pertinent crypto-systems. […] The result of this process will ***inform the development of future NIST recommendations*** *and processes.*

The scope-inclusion of post-quantum (PQ) and quantum-vulnerable (QV) techniques will enable an ***observation of the feasibility "gap"*** *between PQ and QV solutions, which is especially relevant in the setting of PQC-migration.*

Universität Zürich[UZH]

Overview of the NIST call

Table 1. Multiple categories per class

	Sign	PKE	Symmetric	KeyGen	FHE	ZKPoK	Gadgets
Class N	N1	N2	N3	N4	—	—	—
Class S	S1	S2	S3	S4	S5	S6	S7

Legend: Class N = **N**IST-specified primitives; Class S = **S**pecial others, not specified by NIST. FHE = Fully-Homomorphic Encryption. KeyGen = Key Generation. PKE = Public-Key Encryption. ZKPoK = Zero-Knowledge Proof of Knowledge.

First batch of previewed schemes in Jan

NIST Signatures (N1)

- BDLR-Gargos
- BICYCCLIST
- Fireblocks
- Haystack
- FROST
- Mithril
- RedETA
- Quorus
- SplitForge

Symmetric (N3/S3)

- MPC MINIons
- Symphony

Non-NIST Signatures (S1)

- BBDL-tBLS
- Hermine
- LEAST
- PQarrots
- Tanuki
- Vinaigrette

Gadgets (S7)

- Fireblocks
- MPC MINIons
- PiVer
- Schmivitz
- Symphony
- Zama

PKE/KEM (S2)

- Amber
- Fireblocks
- PQarrots
- SplitForge

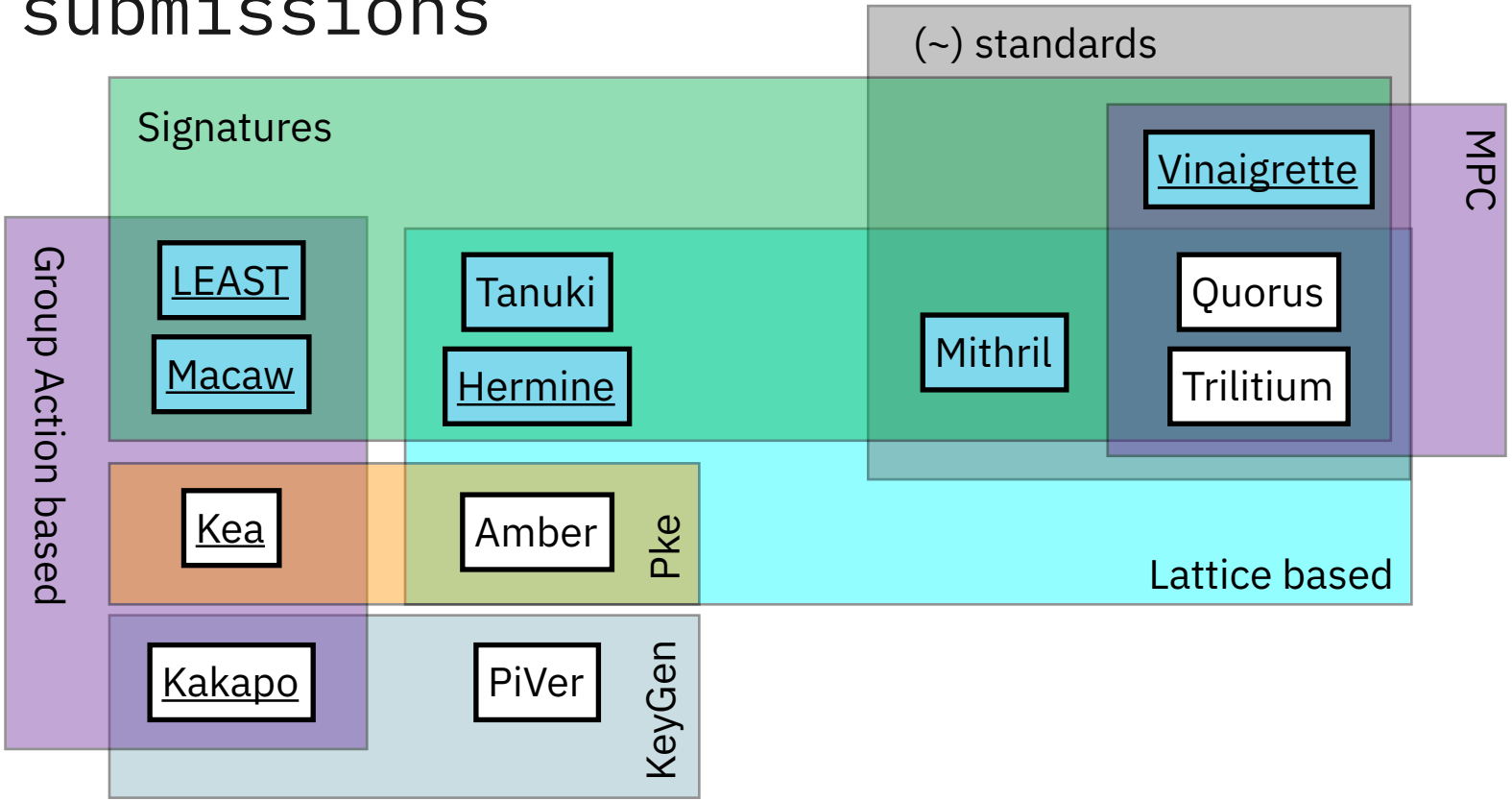
FHE (S5)

- PANTHERIA
- Zama

ZKP (S6)

- Schmivitz
- SmallWood
- Zama

PQ submissions



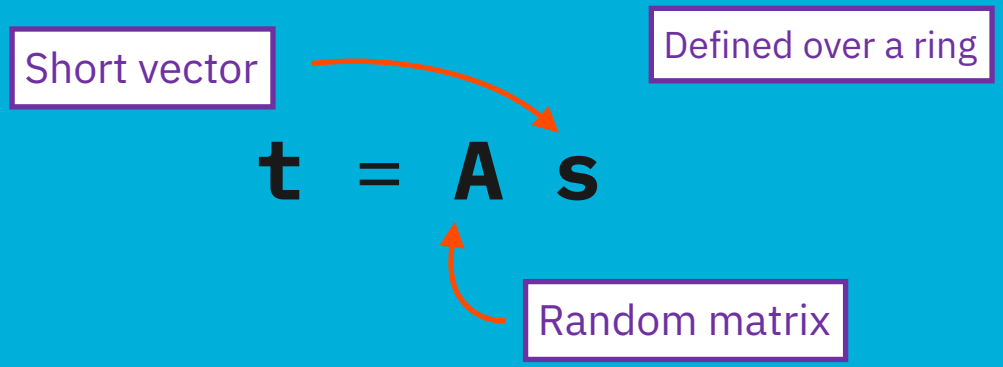
Disclaimer: the presenter only worked on underlined schemes, but we will speak about the one highlighted in blue to show a interesting variety of schemes

Generic constructions from lattices

Mithril, Hermine, Tanuki

Notation on lattices

- (M)LWE: **t** looks random
- (M)SIS: is difficult to find a short vector **s**



Lyubashevsky V. Basic Lattice Cryptography.

Mithril (threshold ML-DSA)

Preprocess(i)
1. sample s_i short
2. $w_i \leftarrow A\ s_i$
3. $cm_i \leftarrow Cmt(w_i)$
4. out (cm_i)

Sign1$((cm_j)_j, m)$
1. collect all cm_j
2. out w_i

Not the same rejection as in ML-DSA

Sign2$((w_j)_j, m)$
1. $w = \Sigma\ w_j$
2. $c = H(w, vk, msg)$
3. $zi = \mathbf{Rej}(c\lambda_i sk_i, s_i)$
4. **require** z_i not $\perp$
5. out $z_i = c\lambda_i sk_i + s_i$

Combine(…)
1. get R as in Sign
2. $z = \Sigma\ z_i$
3. **require** $|z| < B$
4. out (c, z)

Combine cannot be run in parallel

Sign2 can be run in parallel

We need $\lambda_i = 1$ and sk_i small

Celi S, del Pino R, Espitau T, Niot G, Prest T. Efficient Threshold ML-DSA.

Universität Zürich[UZH]

What Mithril can and cannot do

- 🤔 Average of 6 rounds
- ☀️ **Compatibility w/ ML-DSA verification and security**
- 👍 Distributed Key Generation
- 🤔 Handle up to 6 users

T. Espitau, S. Katsumata, and K. Takemure. Two-round threshold signature from algebraic one-more learning with errors.
Del Pino R, Katsumata S, Maller M, Mouhartem F, Prest T, Saarinen MJ. Threshold raccoon: Practical threshold signatures from standard lattice assumptions.
del Pino R, Espitau T, Katsumata S, Maller M, Mouhartem F, Prest T, Rossi M, Saarinen MJ. Raccoon.

Evolution of Th-Raccoon

Raccoon:
Masking friendly lattice signature w/ hintMLWE

→

Th-Raccoon:
Masking + masking

→

Ringtail & EKT24
2 round signing w/ AOMSIS

→

Hermine
Short sharing

Tanuki
Shamir sharing

Tanuki

Boschini C, Kaviani D, Lai RW, Malavolta G, Takahashi A, Tibouchi M. Ringtail: practical two-round threshold signatures from learning with errors.
Cecilia Boschini, Thomas Espitau, Aaron Kaiser, Shuichi Katsumata, Darya Kaviani, Russell W.F. Lai, Giulio Malavolta, Thomas Prest, Peter Schwabe, Akira Takahashi, Kaoru Takemure, Mehdi Tibouchi , “Tanuki”

Preprocess(i)
1. sample $(r_{iu})_u$ short vectors
2. for u : $w_{iu} = A r_{iu}$
3. out $pp_i = (w_{iu})_u$

Combine(…)
1. get R as in Sign
2. out $(c, \Sigma z_i)$

Sign$(m, sk_i, (pp_i)_i)$
1. $(\beta_u)_u = H(vk, mes, (pp_i)_i)$
2. $R = \Sigma \beta_u w_{iu}$
3. $c = H(R, vk, msg)$
4. out $z_i = \Sigma \beta_u r_{iu} + c\, \lambda_i sk_i + \mathbf{mask}$

Verify(m, z, pk)
1. $c' = H(Az - cpk, m)$
2. out $c == c'$

What makes z_i different from random?

$c\, \lambda_i\, sk_i$ is large
> we need to mask it

Luís T. A. N. Brandão René Peralta , “NIST First Call for Multi-Party Threshold Schemes”

What TANUKI can (and cannot) do

- 👍 Partially non interactive threshold signing
- 💰 Verification of pre-signatures w/ NIZK
- ⚙️ Adaptive security (in progress)
- ⬆️ TOP Can handle very large thresholds > 1024

Borin G, Celi S, del Pino R, Espitau T, Katsumata S, Niot G, Prest T, Takemure K.
Hermine: An Efficient Lattice-based FROST-like Threshold Signature.

Hermine

Preprocess(i)
1. sample $(r_{iu})_u$
2. for u $w_{iu} = A r_{iu}$
3. out $pp_i = (w_{iu})_u$

Combine(…)
1. get R as in Sign
2. out $(c, \Sigma z_i)$

Sign($m, sk_i, (pp_i)_i$)
1. $(\beta_u)_u = H(vk, mes, (pp_i)_i)$
2. $R = \Sigma \beta_u w_{iu}$
3. $c = H(R, vk, msg)$
4. out $z_i = \Sigma \beta_u r_{iu} + c\lambda_i sk_i$

Verify(m,z,pk)
1. $c' = H(Az - cpk, m)$
2. out $c == c'$

No mask: z_i has "meaning"

We use $c\,\lambda_i$ of short norm, no need for a mask

Recursive/Vandermonde Secret Sharing

Vandermonde identity

$$\binom{N}{T} = \sum_{k=0}^{T} \binom{b}{k} \cdot \binom{N-b}{T-k}$$

```
x = x3 + x2 + x6
  = x1 + x5 + x9
  = x4 + x7 + x10
  = x1 + x8 + x11
```

VandShare$(x, \mathcal{P}, T, \mathsf{idx}, \mathsf{Shr})$

1: **if** $T = 1$ **then**
2: **for** $i \in \mathcal{P}$ **do** $\mathsf{Shr}[i][\mathsf{idx}] \leftarrow x$
3: **else**
4: $b \leftarrow \lfloor |\mathcal{P}|/2 \rfloor, N \leftarrow |\mathcal{P}|$
5: $\mathcal{P}_L, \mathcal{P}_R \leftarrow \mathcal{P}[1 \dots b], \mathcal{P}[b+1 \dots N]$
6: $\underline{T}_L, \overline{T}_L \leftarrow \max(0, T-(N-b)), \min(T, b)$
7: **for** $k \in [\underline{T}_L, \overline{T}_L]$ **do**
8: $\mathsf{idx}_L \leftarrow \mathsf{idx} \| \texttt{":L:"} \| k$
9: $\mathsf{idx}_R \leftarrow \mathsf{idx} \| \texttt{":R:"} \| (T-k)$
10: **if** $k = 0$ **then**
11: $\mathsf{Shr} \leftarrow \text{VandShare}(x, \mathcal{P}_R, T, \mathsf{idx}_R, \mathsf{Shr})$
12: **elseif** $k = T$ **then**
13: $\mathsf{Shr} \leftarrow \text{VandShare}(x, \mathcal{P}_L, T, \mathsf{idx}_L, \mathsf{Shr})$
14: **else**
15: $x_L \leftarrow \chi, x_R \leftarrow (x - x_L) \bmod q$
16: $\mathsf{Shr} \leftarrow \text{VandShare}(x_L, \mathcal{P}_L, k, \mathsf{idx}_L, \mathsf{Shr})$
17: $\mathsf{Shr} \leftarrow \text{VandShare}(x_R, \mathcal{P}_R, T-k, \mathsf{idx}_R, \mathsf{Shr})$
18: **return** Shr

VandRecover$(\mathcal{P}, \mathsf{act}, \mathsf{idx}, \mathsf{Idx})$

1: $N \leftarrow |\mathcal{P}|;\ T \leftarrow |\mathsf{act}|$
2: **if** $T = 1$ **then**
3: **for** $i \in \mathsf{act}$ **do** $\mathsf{Idx}[i] \leftarrow \mathsf{idx}$
4: **else**
5: $b \leftarrow \lfloor N/2 \rfloor$
6: $\mathcal{P}_L, \mathcal{P}_R \leftarrow \mathcal{P}[1 \dots b], \mathcal{P}[b+1 \dots N]$
7: $\mathsf{act}_L, \mathsf{act}_R \leftarrow \mathsf{act} \cap \mathcal{P}_L, \mathsf{act} \cap \mathcal{P}_R$
8: $k \leftarrow |\mathsf{act}_L|$
9: $\mathsf{idx}_L \leftarrow \mathsf{idx} \| \texttt{":L:"} \| k$
10: $\mathsf{idx}_R \leftarrow \mathsf{idx} \| \texttt{":R:"} \| (T-k)$
11: **if** $k = 0$ **then**
12: $\mathsf{Idx} \leftarrow \text{VandRecover}(\mathcal{P}_R, \mathsf{act}_R, \mathsf{idx}_R, \mathsf{Idx})$
13: **elseif** $k = T$ **then**
14: $\mathsf{Idx} \leftarrow \text{VandRecover}(\mathcal{P}_L, \mathsf{act}_L, \mathsf{idx}_L, \mathsf{Idx})$
15: **else**
16: $\mathsf{Idx} \leftarrow \text{VandRecover}(\mathcal{P}_L, \mathsf{act}_L, \mathsf{idx}_L, \mathsf{Idx})$
17: $\mathsf{Idx} \leftarrow \text{VandRecover}(\mathcal{P}_R, \mathsf{act}_R, \mathsf{idx}_R, \mathsf{Idx})$
18: **return** Idx

$$\mathbf{G}_{3,5} = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & 1 & 1 & 1 \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & -1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & -1 & \cdot \\ 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -1 & \cdot & -1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & -1 \\ \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & -1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot & -1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & -1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix}$$

Yvo Desmedt, Giovanni Di Crescenzo, and Mike Burmester. “Multiplicative Non-abelian Sharing Schemes and their Application to Threshold Cryptography”
Ronald Cramer and Serge Fehr. “Optimal Black-Box Secret Sharing over Arbitrary Abelian Groups”.

Universität Zürich[UZH]

What can Hermine do

- 👍 Partially non interactive threshold signing
- 👍 Stronger unforgeability guarantees (TS-UF-2)
- ⬆️ TOP Non interactive verification of pre-signatures
- 👍 Distributed Key Generation
- 👍 Proactive Refresh
- 🤔 Handle up to 64 users

Universität Zürich[UZH]

Sizes

Targets			Scheme parameters								Security			Sizes			
κ	Q_{Sign}	Q_{rf}	$\lceil \log q \rceil$	k	ℓ	ω	$u_{\mathbf{s}}$	$u_{\mathbf{r}}$	$\nu_{\mathbf{t}}$	$\nu_{\mathbf{w}}$	$\nu_{\mathbf{w}'}$	κ_{pvk}	κ_{vk}	κ_{sig}	$\lvert\mathsf{vk}\rvert$	$\lvert\mathsf{sig}\rvert$	$\lvert\mathsf{pp}_i\rvert$	$\lvert\mathsf{sig}_i\rvert$
128	2^{60}	2^{10}	48	4	4	19	13	36	37	40	32	133	129	124	2,816	11,296	53,248	19,968
192	2^{64}	2^{14}	48	6	6	31	13	36	37	40	32	202	195	211	4,224	16,944	122,880	29,952
256	2^{64}	2^{14}	48	7	8	44	13	35	37	40	32	271	268	261	4,928	21,952	186,368	36,480
128	2^{28}	2^{5}	30	3	3	19	7	17	20	23	16	138	126	162	1,920	4,832	34,944	7,680
192	2^{27}	2^{4}	30	4	4	31	7	17	20	22	15	208	193	245	2,560	6,448	76,800	10,240
256	2^{27}	2^{3}	30	5	5	44	7	17	20	22	14	273	261	325	3,200	8,064	133,120	12,800

Universität Zürich[UZH]

Generic constructions from group actions

Macaw (PQarrots) and LEAST

Cryptographic Group Action

Group: set with an associative invertible operation

Set

$$G \times X \to X$$

$$(a, x) \mapsto a \star x$$

Properties it needs to satisfy

$$a \star (b \star x) = (ab) \star x$$

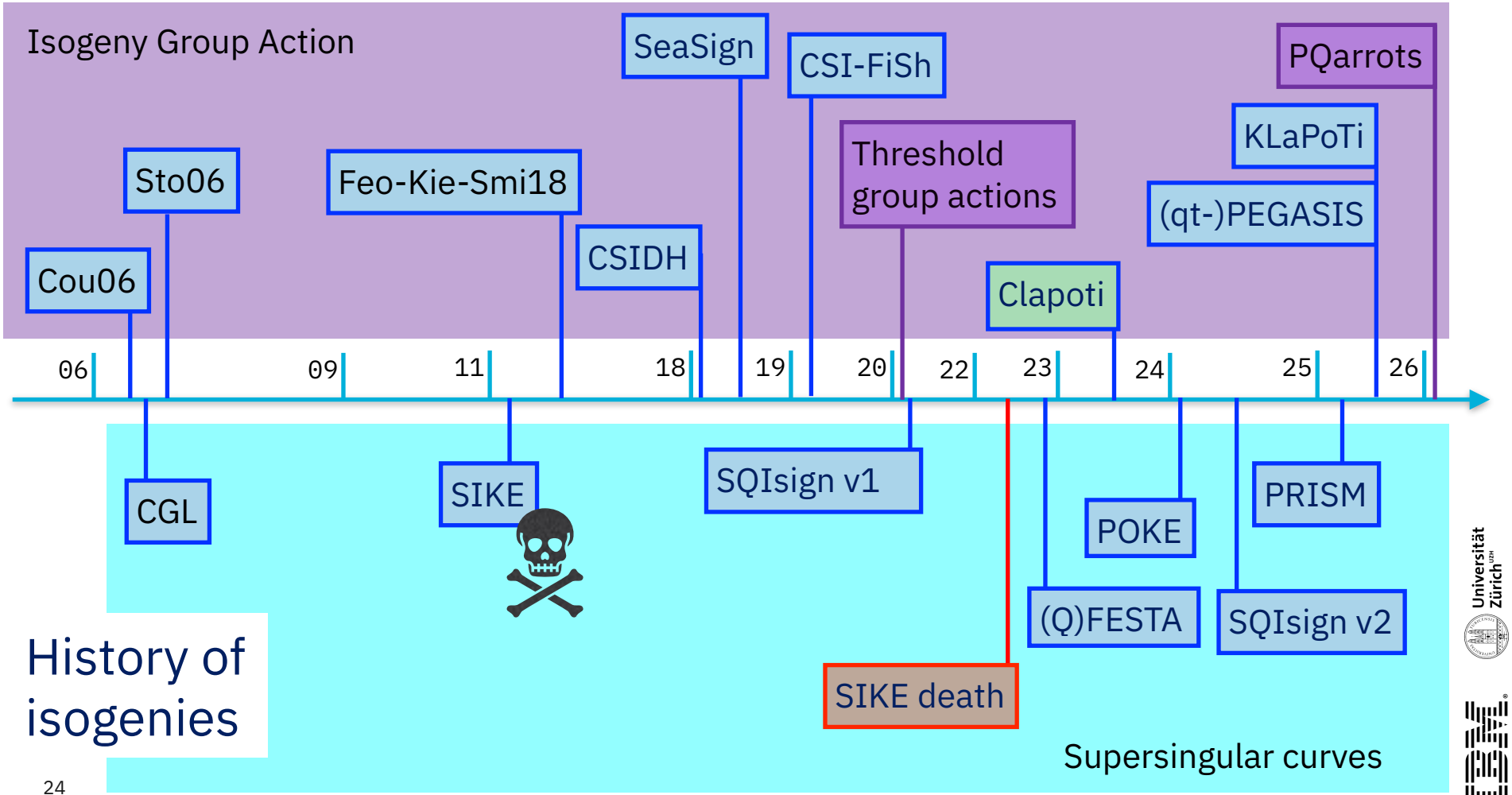
$$a \star (a^{-1} \star x) = x$$

gaDLOG

Given: $x,\ a \star x \in X$

Find: $a \in G$

Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. “Cryptographic Group Actions and Applications”



History of isogenies

Macaw

Sequential preprocessing

Low soundness protocol!

Combine$((z_i^K)^K_i)$
1. get c as in Sign
2. $z^K = \Pi\ z_i^K$
3. out $(c, salt, (z^K)^K)$

Preprocess(i, pm_{i-1})
1. **for** k in [1,…,r]:
    1. sample hi^K in G
    2. $x_i^K \leftarrow h_i^K \star x_{i-1}^K$
2. sample s_i
3. $c_i \leftarrow Cmt(s_i)$
4. out $pm_i = (x_i^K), c_i$

Need to be authenticated:
1. NIZK
2. Private coin proofs

Sign2$(m, sk_i, \ldots)$
1. $R = (x^K)^K$,
2. get salt = $\Sigma\ s_i$
3. require all xi^K valid
4. $(c^K)^K = H(R, vk, msg, salt)$
5. **for** k in [1,…,r]:
    1. $z_i^K = h_i^K \cdot sk_i\text{^}(-c^K\ \lambda_i)$
6. out $(z_i^K)^K$

Sign1$(m, (pm_i)_i)$
1. Collect all x^K and c_i
2. Out s_i

We need c λ_i integers!

We need different secret sharings

Lof of isogenists
"PQarrots: Macaw, Kea and Kakapo"
https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/PQarrots-PW01.pdf

De Feo L, Meyer M. Threshold schemes from isogeny assumptions

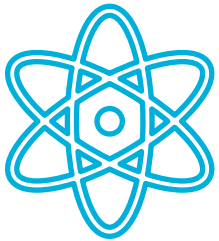
Universität Zürich

What Macaw can (or cannot) do

- 💰 2 online rounds + T preprocessing
- 💰 Distributed Key Generation (costly)
- 👍 Handle up to $t < 32$
- 💰 Verification of pre-signatures w/ NIZK
- 👍 Description only in terms of group actions
- ⚙️ Costly isogeny evaluations, less compact

Universität Zürich[UZH]

Group action evaluation estimates



- The Group Action is commutative, so **quantum sub-exponential algorithms applies**
- The cost needs to be concretely evaluated, we can't agree on that

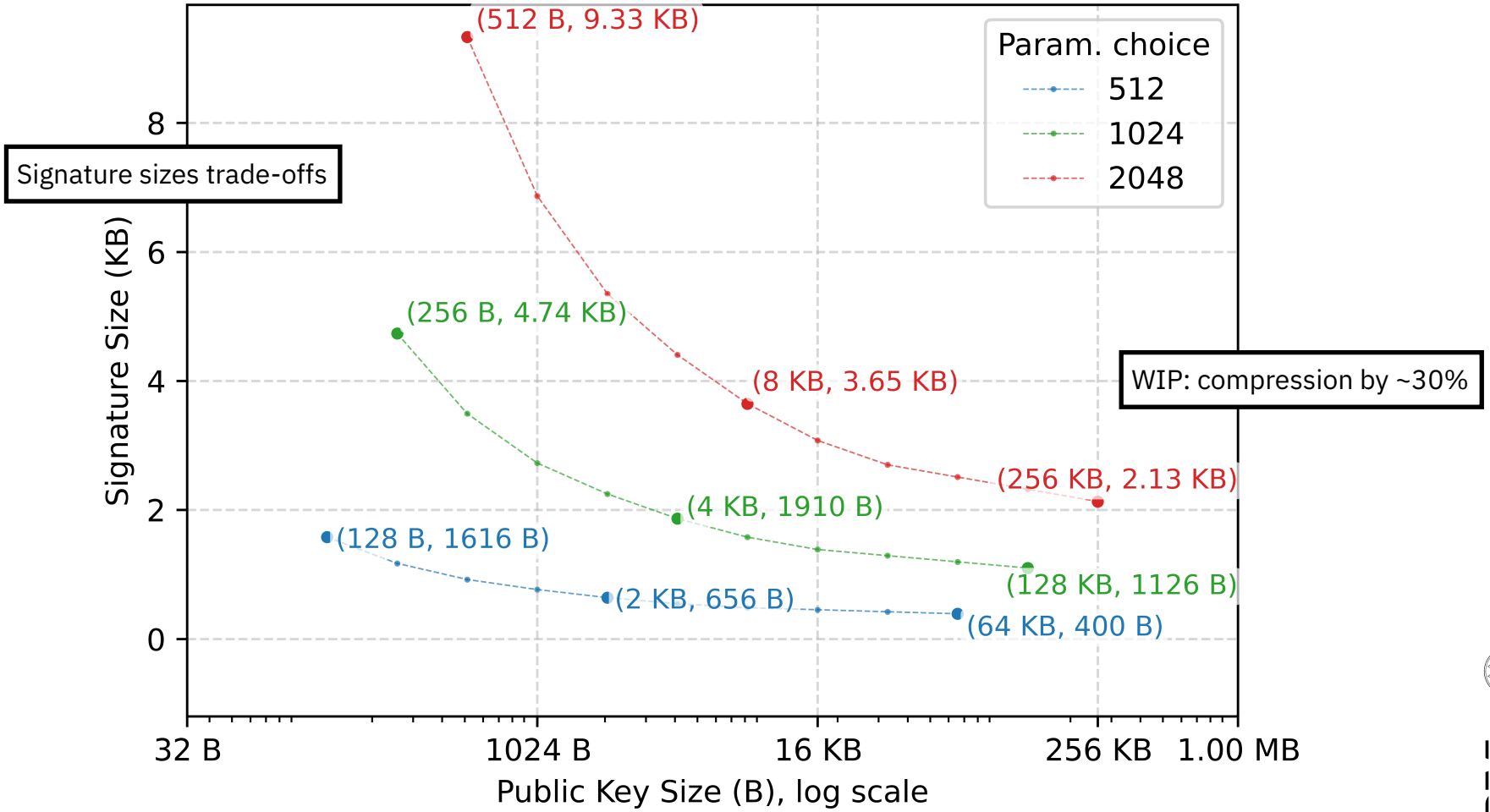
Parameter set Size of p (in bits)	*optimist* 512	*prudent* 1024	*pessimist* 2048
Estimated C timings (in MCycles)	103	510	3633
Estimated C timings (in ms)	41	204	1450

Same classical security as SQIsign NIST I, III, V

Group action operation median of 100 runs on a i7-11850H with turboboost disabled with 2.5 GHz clock speed, implementation of the scheme from: Pierrick Dartois, Jonathan Komada Eriksen, Riccardo Invernizzi, and Frederik Vercauteren. qt-Pegasis: Simpler and Faster Effective Class Group Actions

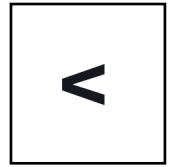
Universität Zürich[UZH]

Signature sizes trade-offs



WIP: compression by ~30%

LEAST (non commutative actions)



All sequential operations

Sign1$(m,(pm_1)_i)$
1. Collect all x^K and c_i
2. Out s_i

Low soundness protocol!

Preprocess$(pm_{1\,i-1})$
1. **for** k in [1,…,r]:
    1. sample $h_i{}^K$ in G
    2. $x_i{}^K \leftarrow h_i{}^K \star x_{i-1}{}^K$
2. sample s_i
3. $c_i \leftarrow Cmt(s_i)$
4. out $pm_{1\,i-1}=(x_i{}^K),c_i$

Signj$(pm_{2\,i-1},sk_i)$
1. $R = (x^K)^K$
2. get salt = $\Sigma\ s_i$
3. $c = H(R,vk,msg,salt)$
4. **require** all $z_{i-1}{}^K$ valid
5. **for** k in [1,…,r]:
    1. $z_i{}^K = h_i{}^K \cdot z_{i-1}{}^K \cdot sk_i\text{^}(-c^K\ \lambda_i)$
6. out $pm_{2\,i} = (z_i{}^K)^K$

We need $c\,\lambda_i = \pm 1,0$

Battagliola M, Borin G, Meneghetti A, Persichetti E. Cutting the grass: Threshold group action signature schemes
Battagliola M, Borin G, Di Crescenzo G, Meneghetti A, Persichetti E. Enhancing threshold group action signature schemes: Adaptive security and scalability improvements.

Universität Zürich[UZH]

What LEAST can (or cannot) do

- 💸 T+1 online rounds + T preprocessing
- 🤔 Handle up to t < 16
- 👍 Verification of pre-signatures
- 👍 Description only in terms of group actions
- 👍 Can use code based group actions

LEAST sig size

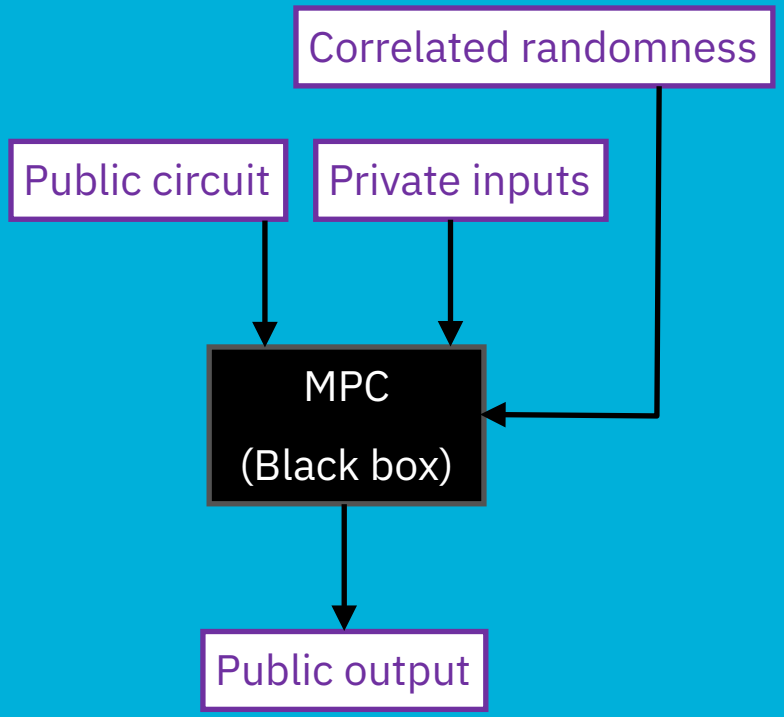
NIST Cat.	Parameter Set	Prot. Params n	k	q	Keys s	Rep. τ	pk size (B)	Signature size (B)
1	LEAST-252-2	252	126	127	2	128	13940	4160
	LEAST-252-4				4	64	41788	2112
	LEAST-252-8				8	43	97484	1440
3	LEAST-400-2	400	200	127	2	192	35074	9696
	LEAST-400-4				4	96	105174	4896
5	LEAST-548-2	548	274	127	2	256	65793	17792
	LEAST-548-4				4	128	197315	8960

Universität Zürich[UZH]

Use of generic MPC

Case study of Vinaigrette, a suite for UOV and MAYO

What is Multi Party Computation (MPC)?

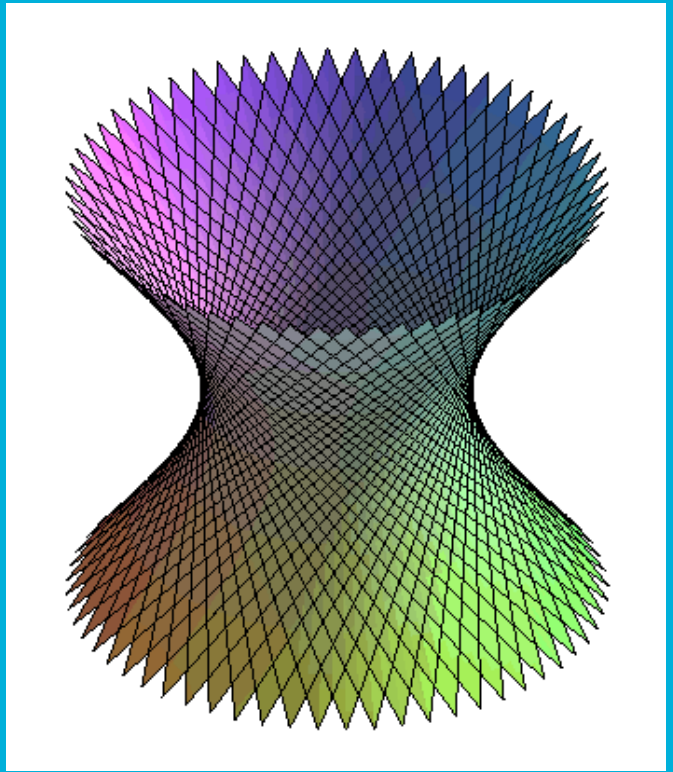


- Allowed operations (order of cost)
  - Addition and multiplication by scalars
  - Sampling of public/private randomness
  - Opening of secret inputs
  - Multiplication of secret inputs
    - Multiplications of secret matrices

OV-based (UOV and MAYO) framework

Given a quadratic map $P : \mathbb{F}^n \to \mathbb{F}^m$ signing means to find a vector $\vec{s} \in \mathbb{F}^n$ such that $P(\vec{s}) = H(m, \text{salt})$

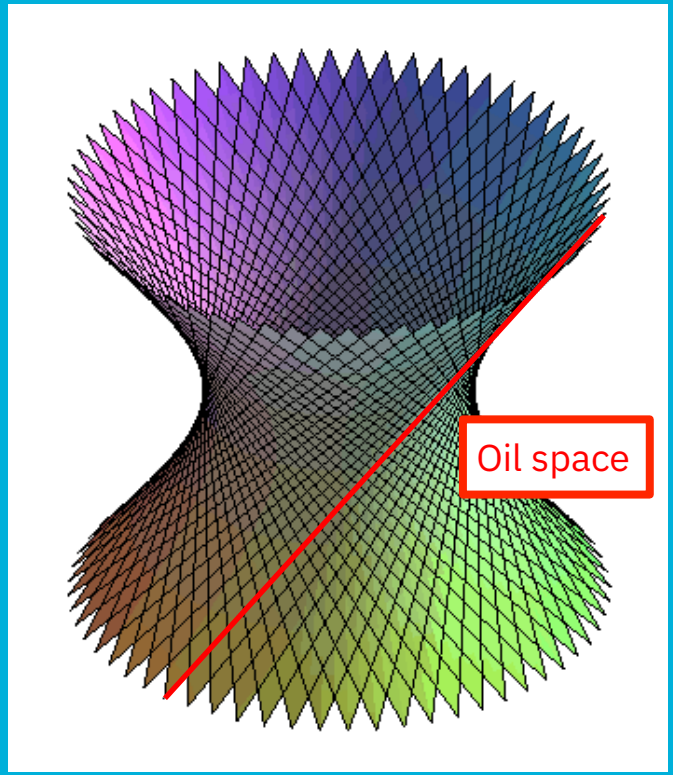
Searching points on a shifted variety



Universität Zürich[UZH]

OV-based trapdoor

There is a dimension o linear subspace $O \subset \mathbb{F}^n$ (oil space) on which P vanishes: $\forall \vec{o} \in O : P(\vec{o}) = 0$

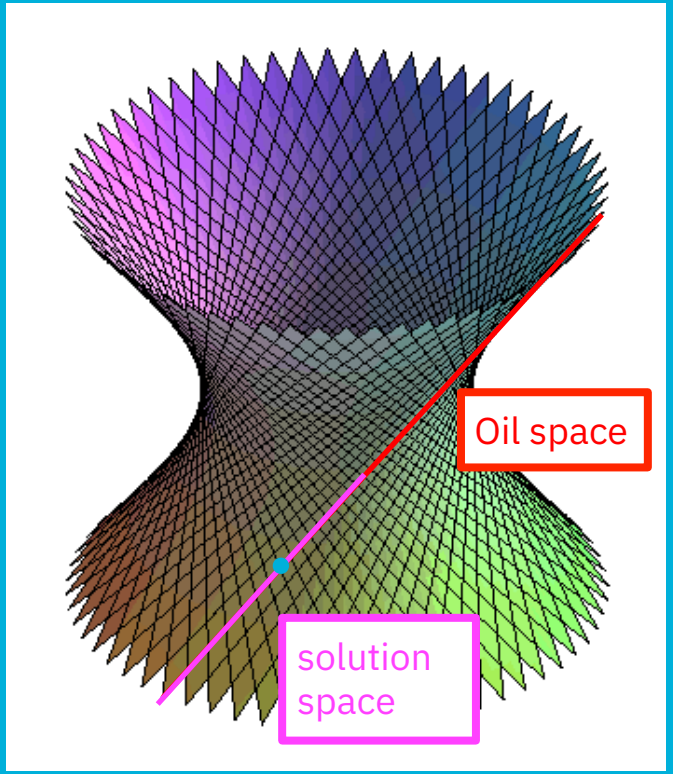


MAYO: build P^* from a smaller (but still ‘hard’) P

OV-based signing with the trapdoor

By writing the solution as $\vec{s} = \vec{v} + \begin{pmatrix} O \\ I_o \end{pmatrix} \vec{x}$ we can "collapse" the problem to a linear system:

$$A\vec{x} = H(m, \text{salt}) - P(\vec{v})$$



Signing

OV-based.Sign(sk, pk, msg)

1 : Parse $\mathsf{sk} = (\mathbf{O}, \{\mathbf{L}_i\}_{i\in[m]})$

2 : Parse $\mathsf{pk} = (\{\mathbf{P}_i^{(1)}\}_{i\in[m]}, \{\mathbf{P}_i^{(2)}, \cdot\}_{i\in[m]})$

3 : $\mathsf{salt} \xleftarrow{\$} \{0,1\}^{\lambda+64}$

4 : $\mathbf{t} \leftarrow \mathcal{H}(\mathsf{msg}\|\mathsf{salt})$

5 : **for** $\mathsf{ctr} = 1, ..., \infty$ **do**

6 : $\quad \mathbf{V} \xleftarrow{\$} \mathbb{F}_q^{k\times(n-o)}$

7 : $\quad$ **for** $i \leftarrow 1$ to m **do** // $\mathbf{M}_i \in \mathbb{F}_q^{k\times n}$

8 : $\quad\quad \mathbf{M}_i \leftarrow \mathbf{V} \cdot \mathbf{L}_i$

9 : $\quad\quad \mathbf{Y}_i \leftarrow \mathbf{V} \cdot \mathbf{P}_i^{(1)} \cdot \mathbf{V}^\top$ // $\mathbf{Y}_i \in \mathbb{F}_q^{k\times k}$

10 : $\quad \mathbf{A} \leftarrow$ OV-based.Compute_A$(\mathbf{O}, \{\mathbf{M}_i\}_{i\in[m]})$

11 : $\quad \mathbf{Y} \leftarrow$ OV-based.Compute_y$(\{\mathbf{Y}_i\}_{i\in[m]})$

12 : $\quad \mathbf{y} \leftarrow \mathbf{t} - \mathbf{Y}$

13 : $\quad$ **if** $\mathbf{A}$ is not full-rank **then continue**

14 : $\quad \mathbf{x} \leftarrow$ SampleSolution$(\mathbf{A}, \mathbf{y})$ // $\mathbf{x} = \mathbf{A}^{-1}(\mathbf{t} - \mathbf{Y}) \in \mathbb{F}_q^{kn}$

15 : $\mathbf{X} \leftarrow$ Matrixify$(\mathbf{x})$ // $\mathbf{X} \in \mathbb{F}_q^{k\times n}$

16 : $\mathbf{S} \leftarrow (\mathbf{V} + (\mathbf{O}\mathbf{X}^\top)^\top, \mathbf{X})$

17 : **return** $\mathsf{Sig} = (\mathbf{S}, \mathsf{salt})$

Hash salted message

Prepare the system / collapse the map

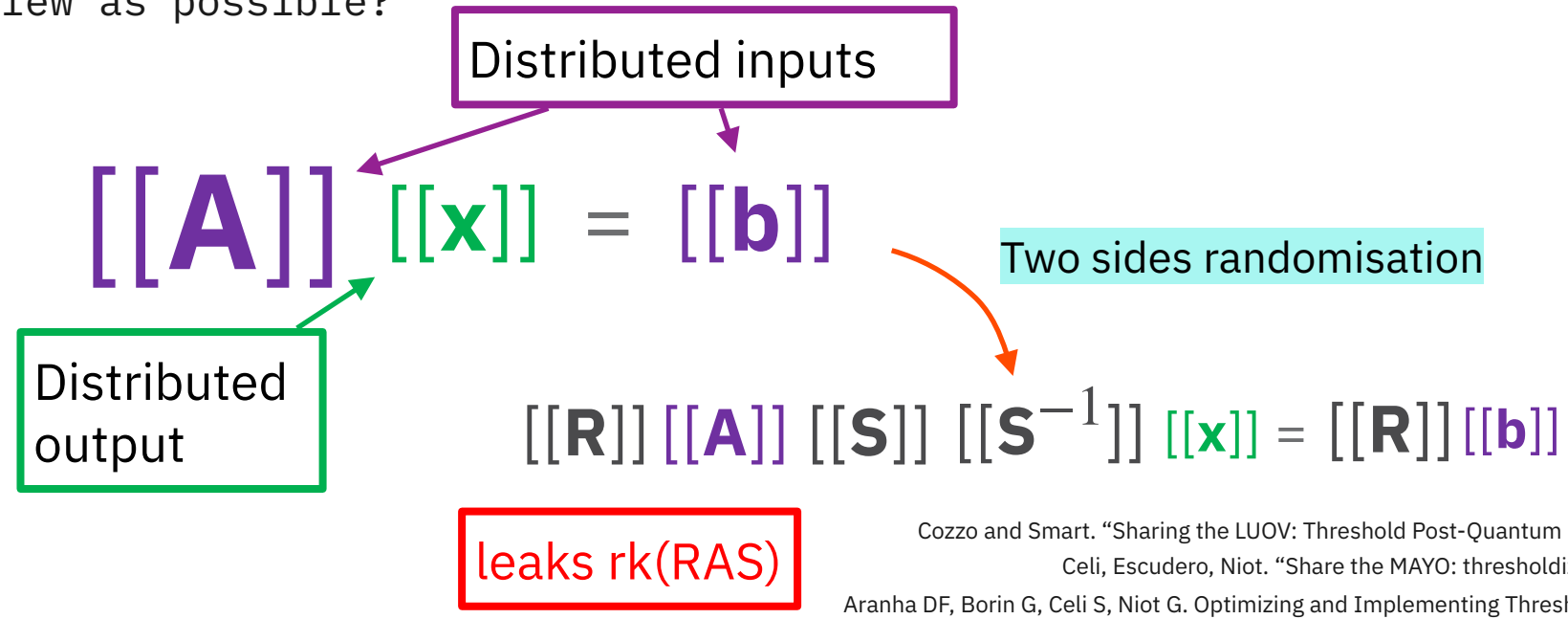
Message-dependent adjustment

Solve the linear system

Assemble the signature

Most important step: solving a linear system obliviously

How to solve a linear system obliviously using MPC and leaking as few as possible?



Cozzo and Smart. “Sharing the LUOV: Threshold Post-Quantum Signatures”

Celi, Escudero, Niot. “Share the MAYO: thresholdizing MAYO”

Aranha DF, Borin G, Celi S, Niot G. Optimizing and Implementing Threshold MAYO.

OV-based.Sign(sk, pk, msg)

Cost: ≈ 9 rounds

Goals:
1. non-interactive
2. more efficient

1 : Parse $\mathsf{sk} = (\mathbf{O}, \{\mathbf{L}_i\}_{i\in[m]})$
2 : Parse $\mathsf{pk} = (\{\mathbf{P}_i^{(1)}\}_{i\in[m]}, \{\mathbf{P}_i^{(2)}, \cdot\}_{i\in[m]})$
3 : $\mathsf{salt} \xleftarrow{\$} \{0,1\}^{\lambda+64}$
4 : $\mathbf{t} \leftarrow \mathcal{H}(\mathsf{msg}\|\mathsf{salt})$
5 : **for** $\mathsf{ctr} = 1, \ldots, \infty$ **do**
6 : $\quad \mathbf{V} \xleftarrow{\$} \mathbb{F}_q^{k\times(n-o)}$
7 : $\quad$ **for** $i \leftarrow 1$ to m **do** // $\mathbf{M}_i \in \mathbb{F}_q^{k\times n}$
8 : $\quad\quad \mathbf{M}_i \leftarrow \mathbf{V}\cdot\mathbf{L}_i$
9 : $\quad\quad \mathbf{Y}_i \leftarrow \mathbf{V}\cdot\mathbf{P}_i^{(1)}\cdot\mathbf{V}^\top$ // $\mathbf{Y}_i \in \mathbb{F}_q^{k\times k}$
10 : $\quad \mathbf{A} \leftarrow$ OV-based.Compute_A$(\mathbf{O}, \{\mathbf{M}_i\}_{i\in[m]})$
11 : $\quad \mathbf{Y} \leftarrow$ OV-based.Compute_y$(\{\mathbf{Y}_i\}_{i\in[m]})$
12 : $\quad \mathbf{y} \leftarrow \mathbf{t} - \mathbf{Y}$
13 : $\quad$ **if** $\mathbf{A}$ is not full-rank **then continue**
14 : $\quad \mathbf{x} \leftarrow$ SampleSolution$(\mathbf{A}, \mathbf{y})$ // $\mathbf{x} = \mathbf{A}^{-1}(\mathbf{t}-\mathbf{Y}) \in \mathbb{F}_q^{kn}$
15 : $\mathbf{X} \leftarrow$ Matrixify$(\mathbf{x})$ // $\mathbf{X} \in \mathbb{F}_q^{k\times n}$
16 : $\mathbf{S} \leftarrow (\mathbf{V} + (\mathbf{O}\mathbf{X}^\top)^\top, \mathbf{X})$
17 : **return** $\mathsf{Sig} = (\mathbf{S}, \mathsf{salt})$

salt can be precomputed

We can build A differently to reduced depth!

[[A]] and [[Y]] are independent of the message

$[[x_y]] = [[A^{-1}]]\,[[Y]]$ can be precomputed

Only t is message dependent, but it is a public value

$[[x]] = [[A^{-1}]]\,t - [[A^{-1}]][[Y]]$
$= [[A^{-1}]]\,t - [[x_y]]$ is a local operation

We can precompute $[[\mathsf{diag}(O)]]\,[[A^{-1}]]$ to skip this step

Celi, Escudero, Niot. "Share the MAYO: thresholdizing MAYO"

Aranha DF, Borin G, Celi S, Niot G. Optimizing and Implementing Threshold MAYO.

What Vinaigrette can do

- 👍 Partially non interactive scheme
  - (🤔 MPC preprocessing + 4/5 rounds)
- 👍 Compatibility w/ UOV and MAYO
- 👍 Distributed Key Generation
- 👍 Adaptive security and robustness from MPC
- 👍 Handle large thresholds (💰 up to MPC limits)

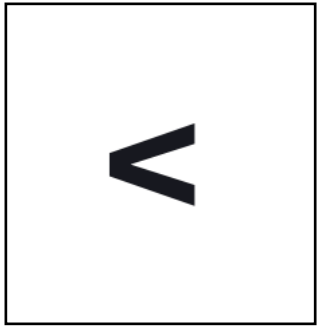
Signing performance for threshold MAYO1

(N,T)	Standard (3 online rounds) Non-Active	Standard (3 online rounds) Active	Explicit-Salt Non-Active	Explicit-Salt Active	Explicit-Salt & Depth-Reduced Non-Active	Explicit-Salt & Depth-Reduced Active
(2,2)	1.45 (**252.38**)	5.73 (**1033.36**)	**0.29** (173.52)	4.99 (2007.09)	0.30 (555.42)	**4.82** (5182.61)
(3,2)	1.13 (329.96)	9.98 (5190.42)	0.45 (**276.18**)	7.67 (**6591.91**)	**0.44** (790.84)	**7.34** (13798.31)
(8,4)	1.81 (**458.39**)	23.20 (**11658.59**)	1.20 (656.26)	19.69 (18649.15)	**1.19** (1869.27)	**19.58** (38854.61)
(15,8)	3.10 (1340.48)	48.12 (**27422.00**)	2.32 (**1332.61**)	37.25 (43788.12)	**2.30** (3415.58)	**37.02** (86646.07)
Comm./party	4.0 (272.6)	–	0.4 (213.4)	–	0.4 (671.9)	–

WIP on improving implementation and use of ad-hoc MPC procedure

WIP on better security

Benchmarks over 100 samples conducted on macOS 15.6.1 (Darwin 24.6.0) running on Apple Silicon (arm64), with an Apple M3 CPU and 24 GB of RAM with GOMAXPROCS=1.
Online signing time (total signing time in parenthesis) in ms, comm. is in KB.
Bold values indicate the best performance for each row and security model.



Thanks