Threshold Raccoon

Rafael del Pino
PQShield

Thomas Espitau
PQShield

Shuichi Katsumata
PQShield & AIST

Mary Maller
Ethereum Foundation & PQShield

Fabrice Mouhartem
XWIKI

Thomas Prest
PQShield

Markku-Juhani Saarinen
Tampere University & PQShield

Kaoru Takemure
PQShield & AIST

Fifth PQC Standardization Conference

Threshold Cryptography

When disaster strikes

Devices can be **compromised** by…

- 🏴‍☠️ Malwares
- 🏴‍☠️ Zero-day exploits
- 🏴‍☠️ Human error
- 🏴‍☠️ …

Devices can be made **out of order** by…

- ⚰️ Network or energy failure
- ⚰️ Attack on the infrastructure
- ⚰️ Destruction
- ⚰️ …

The solution is redundancy

PQ SHIELD

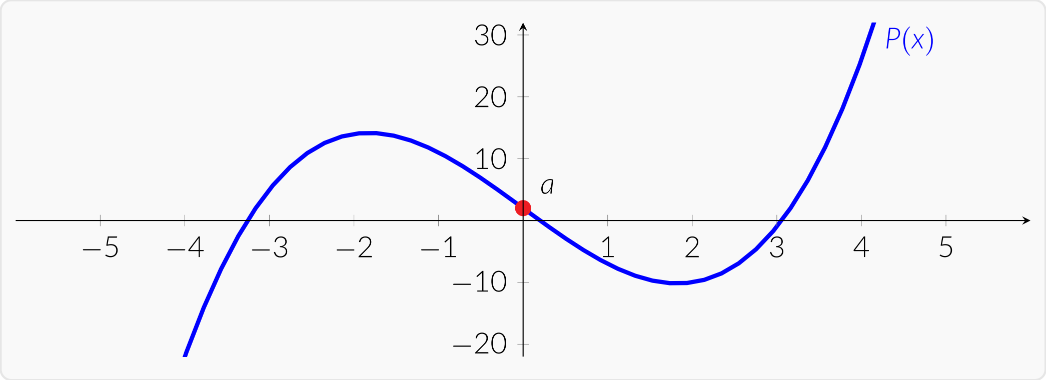
Key idea: distribute trust across several devices

		🏴‍☠️ Attacker: how many devices to compromise?	⚰️ Attacker: how many devices to destroy?
1 device	**1** key	1 / 1	1 / 1
N devices	**1** key	1 / N	N / N
N devices	**N** keys	N / N	1 / N
N devices	**T-out-of-N** keys	T / N	(N - T + 1) / N

→ The two last solutions fall under **threshold cryptography**

→ Main focus of the NIST MPTC programme

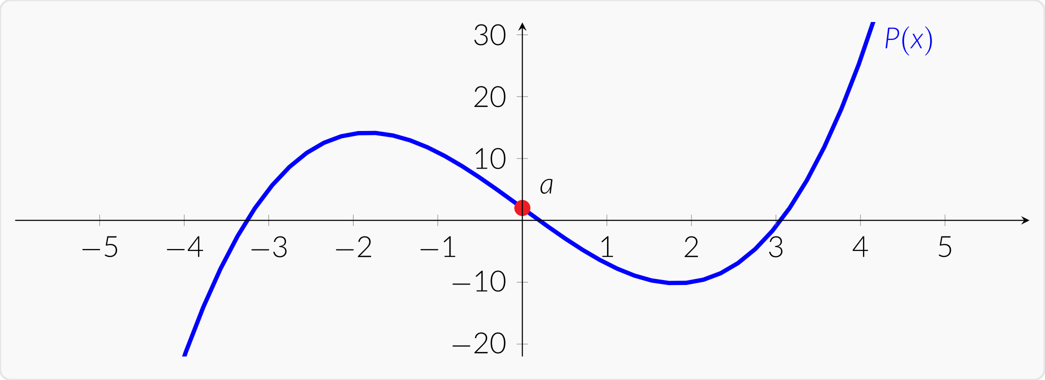
Shamir secret sharing



Secret-sharing a secret $a \in \mathbb{Z}_p$:

- Generate $P(x)$ of degree at most $T - 1$ such that $P(0) = a$
- Each party $i \in \mathbb{Z}_p$ receives a share $a_i P(i)$

Shamir secret sharing



Properties:

- With $< T$ shares, a is perfectly hidden
- With a set $\mathcal{S}$ of T shares, a can be recovered via Lagrange interpolation:

$$a = \sum_{i \in \mathcal{S}} \lambda_{i,\mathcal{S}} \cdot a_i, \quad \text{where} \quad \lambda_{i,\mathcal{S}} = \prod_{j \in \mathcal{S} \setminus \{i\}} \frac{j}{i - j} \tag{1}$$

Raccoon ≈ Schnorr over lattices

Schnorr.Keygen$() \to \mathtt{sk}, \mathtt{vk}$

1. Sample uniform $\mathtt{sk}$, set $\mathtt{vk} = g^{\mathtt{sk}}$

Schnorr.Sign$(\mathtt{sk}, \mathtt{msg}) \to \mathtt{sig}$

1. Sample r
2. $w = g^r$
3. $c = H(w, \mathtt{msg})$
4. $z = r + c \cdot \mathtt{sk}$
5. Output $\mathtt{sig} = (c, z)$

Schnorr.Verify$(\mathtt{vk}, \mathtt{msg}, \mathtt{sig})$

1. $w' = g^z \cdot \mathtt{vk}^{-c}$
2. Assert $H(\mathbf{w}', \mathtt{msg}) = c$

Raccoon.Keygen$() \to \mathtt{sk}, \mathtt{vk}$

1. Sample short $\mathtt{sk}$, set $\mathtt{vk} = [\mathbf{A} \quad 1] \cdot \mathtt{sk}$

Raccoon.Sign$(\mathtt{sk}, \mathtt{msg}) \to \mathtt{sig}$

1. Sample a short $\mathbf{r}$
2. $\mathbf{w} = [\mathbf{A} \quad 1] \cdot \mathbf{r}$
3. $c = H(\mathbf{w}, \mathtt{msg})$
4. $\mathbf{z} = \mathbf{r} + c \cdot \mathtt{sk}$
5. Output $\mathtt{sig} = (c, \mathbf{z})$

Raccoon.Verify$(\mathtt{vk}, \mathtt{msg}, \mathtt{sig})$

1. $\mathbf{w}' = [\mathbf{A} \quad 1] \cdot \mathbf{z} - c \cdot \mathtt{vk}$
2. Assert $H(\mathbf{w}', \mathtt{msg}) = c$

Threshold Schnorr signatures

Sparkle (CRYPTO 2023)

Each signer i knows a share sk_i of sk.

- **Round 1:**
  1. Sample r_i
  2. $w_i = g^{r_i}$
  3. $\mathtt{com}_i = H_{\mathtt{com}}(w_i, \mathtt{msg}, \mathcal{S})$
  4. Broadcast $\mathtt{com}_i$
- **Round 2:**
  1. Broadcast w_i
- **Round 3:**
  1. $w = \prod_i w_i$
  2. $c = H(\mathsf{vk}, \mathtt{msg}, w)$
  3. $z_i = r_i + c \cdot \lambda_{i,\mathcal{S}} \cdot \mathsf{sk}_i$
  4. Broadcast z_i
- **Combine:** the final signature is $(c, z = \sum_{i \in \mathcal{S}} z_i)$

✔ This produces valid Schnorr signatures:

$$\begin{aligned} g^z &= g^{\sum_i z_i} \\ &= \left(g^{\sum_i r_i}\right) \cdot \left(g^{c \sum_i \lambda_{i,\mathcal{S}} \cdot \mathsf{sk}_i}\right) \\ &= w \cdot \mathsf{vk}^c \end{aligned}$$

🔒 Security: in z_i, r_i is uniform and perfectly hides $c \sum_i \lambda_{i,\mathcal{S}} \cdot \mathsf{sk}_i$

⚠ We commit to w_i before revealing it to avoid ROS attacks [DEF+19, BLL+22]

? Can we transpose this to Raccoon?

Threshold Raccoon

First attempt

Insecure Threshold Raccoon

→ **Round 1:**
1. Sample short $\mathbf{r}_i$
2. $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
3. $\mathtt{com}_i = H_{\mathtt{com}}(\mathbf{w}_i, \mathtt{msg}, \mathcal{S})$
4. Broadcast $\mathtt{com}_i$

→ **Round 2:**
1. Broadcast $\mathbf{w}_i$

→ **Round 3:**
1. $\mathbf{w} = \sum_i \mathbf{w}_i$
2. $c = H(\mathsf{vk}, \mathtt{msg}, \mathbf{w})$
3. $\mathbf{z}_i = \mathbf{r}_i + c \cdot \lambda_i \cdot \mathsf{sk}_i$
4. Broadcast $\mathbf{z}_i$

→ **Combine:** the final signature is $(c, \mathbf{z} = \sum_{i \in \mathcal{S}} \mathbf{z}_i)$

✓ This gives valid Raccoon signatures (up to slight parameter changes)

⚠ Issue: when we consider

$$\mathbf{z}_i = \mathbf{r}_i + c \cdot \lambda_i \cdot \mathsf{sk}_i, \qquad (2)$$

$\mathbf{r}_i$ is small whereas $c \cdot \lambda_i \cdot \mathsf{sk}_i$ is large.

- Breaks the security proof
- For a fixed i, with enough $\mathbf{z}_i$ of the form in (2) one can recover sk_i

Our idea

PQ SHIELD

	1	2	3	4	5
1	$\mathbf{m}_{1,1}$	$\mathbf{m}_{1,2}$	$\mathbf{m}_{1,3}$	$\mathbf{m}_{1,4}$	$\mathbf{m}_{1,5}$
2	$\mathbf{m}_{2,1}$	$\mathbf{m}_{2,2}$	$\mathbf{m}_{2,3}$	$\mathbf{m}_{2,4}$	$\mathbf{m}_{2,5}$
3	$\mathbf{m}_{3,1}$	$\mathbf{m}_{3,2}$	$\mathbf{m}_{3,3}$	$\mathbf{m}_{3,4}$	$\mathbf{m}_{3,5}$
4	$\mathbf{m}_{4,1}$	$\mathbf{m}_{4,2}$	$\mathbf{m}_{4,3}$	$\mathbf{m}_{4,4}$	$\mathbf{m}_{4,5}$
5	$\mathbf{m}_{5,1}$	$\mathbf{m}_{5,2}$	$\mathbf{m}_{5,3}$	$\mathbf{m}_{5,4}$	$\mathbf{m}_{5,5}$

- Users (i, j) share a symmetric key, and can generate a fresh $\mathbf{m}_{i,j}$ each session
- Each user knows all $\mathbf{m}_{i,j}$'s on their corrresponding row and column

Our idea

PQ SHIELD

$$
\begin{array}{c|ccccccccccccc}
 & 1 & & 2 & & 3 & & 4 & & 5 & & \\
1 & \mathbf{m}_{1,1} & + & \mathbf{m}_{1,2} & + & \mathbf{m}_{1,3} & + & \mathbf{m}_{1,4} & + & \mathbf{m}_{1,5} & = & \mathbf{m}_1 \\
 & + & & + & & + & & + & & + & & + \\
2 & \mathbf{m}_{2,1} & + & \mathbf{m}_{2,2} & + & \mathbf{m}_{2,3} & + & \mathbf{m}_{2,4} & + & \mathbf{m}_{2,5} & = & \mathbf{m}_2 \\
 & + & & + & & + & & + & & + & & + \\
3 & \mathbf{m}_{3,1} & + & \mathbf{m}_{3,2} & + & \mathbf{m}_{3,3} & + & \mathbf{m}_{3,4} & + & \mathbf{m}_{3,5} & = & \mathbf{m}_3 \\
 & + & & + & & + & & + & & + & & + \\
4 & \mathbf{m}_{4,1} & + & \mathbf{m}_{4,2} & + & \mathbf{m}_{4,3} & + & \mathbf{m}_{4,4} & + & \mathbf{m}_{4,5} & = & \mathbf{m}_4 \\
 & + & & + & & + & & + & & + & & + \\
5 & \mathbf{m}_{5,1} & + & \mathbf{m}_{5,2} & + & \mathbf{m}_{5,3} & + & \mathbf{m}_{5,4} & + & \mathbf{m}_{5,5} & = & \mathbf{m}_5 \\
 & \| & & \| & & \| & & \| & & \| & & \| \\
 & \mathbf{m}_1^* & + & \mathbf{m}_2^* & + & \mathbf{m}_3^* & + & \mathbf{m}_4^* & + & \mathbf{m}_5^* & = & \mathbf{m}
\end{array}
$$

- Users (i, j) share a symmetric key, and can generate a fresh $\mathbf{m}_{i,j}$ each session
- Each user knows all $\mathbf{m}_{i,j}$'s on their corrresponding row and column

Our idea

	1	2	3	4	5	

$$
\begin{array}{c|ccccccccccc}
 & 1 & & 2 & & 3 & & 4 & & 5 & & \\
\hline
1 & \mathbf{m}_{1,1} & + & \mathbf{m}_{1,2} & + & \mathbf{m}_{1,3} & + & \mathbf{m}_{1,4} & + & \mathbf{m}_{1,5} & = & \mathbf{m}_1 \\
 & + & & + & & + & & + & & + & & + \\
2 & \mathbf{m}_{2,1} & + & \mathbf{m}_{2,2} & + & \mathbf{m}_{2,3} & + & \mathbf{m}_{2,4} & + & \mathbf{m}_{2,5} & = & \mathbf{m}_2 \\
 & + & & + & & + & & + & & + & & + \\
3 & \mathbf{m}_{3,1} & + & \mathbf{m}_{3,2} & + & \mathbf{m}_{3,3} & + & \mathbf{m}_{3,4} & + & \mathbf{m}_{3,5} & = & \mathbf{m}_3 \\
 & + & & + & & + & & + & & + & & + \\
4 & \mathbf{m}_{4,1} & + & \mathbf{m}_{4,2} & + & \mathbf{m}_{4,3} & + & \mathbf{m}_{4,4} & + & \mathbf{m}_{4,5} & = & \mathbf{m}_4 \\
 & + & & + & & + & & + & & + & & + \\
5 & \mathbf{m}_{5,1} & + & \mathbf{m}_{5,2} & + & \mathbf{m}_{5,3} & + & \mathbf{m}_{5,4} & + & \mathbf{m}_{5,5} & = & \mathbf{m}_5 \\
 & \parallel & & \parallel & & \parallel & & \parallel & & \parallel & & \parallel \\
 & \mathbf{m}^*_1 & + & \mathbf{m}^*_2 & + & \mathbf{m}^*_3 & + & \mathbf{m}^*_4 & + & \mathbf{m}^*_5 & = & \mathbf{m}
\end{array}
$$

- Users (i, j) share a symmetric key, and can generate a fresh $\mathbf{m}_{i,j}$ each session
- Each user knows all $\mathbf{m}_{i,j}$'s on their corrresponding row and column

Our idea

$$
\begin{array}{c|ccccccccccc}
 & 1 & & 2 & & 3 & & 4 & & 5 & & \\
1 & \mathbf{m}_{1,1} & + & \mathbf{m}_{1,2} & + & \mathbf{m}_{1,3} & + & \mathbf{m}_{1,4} & + & \mathbf{m}_{1,5} & = & \mathbf{m}_1 \\
 & + & & + & & + & & + & & + & & + \\
2 & \mathbf{m}_{2,1} & + & \mathbf{m}_{2,2} & + & \mathbf{m}_{2,3} & + & \mathbf{m}_{2,4} & + & \mathbf{m}_{2,5} & = & \mathbf{m}_2 \\
 & + & & + & & + & & + & & + & & + \\
3 & \mathbf{m}_{3,1} & + & \mathbf{m}_{3,2} & + & \mathbf{m}_{3,3} & + & \mathbf{m}_{3,4} & + & \mathbf{m}_{3,5} & = & \mathbf{m}_3 \\
 & + & & + & & + & & + & & + & & + \\
4 & \mathbf{m}_{4,1} & + & \mathbf{m}_{4,2} & + & \mathbf{m}_{4,3} & + & \mathbf{m}_{4,4} & + & \mathbf{m}_{4,5} & = & \mathbf{m}_4 \\
 & + & & + & & + & & + & & + & & + \\
5 & \mathbf{m}_{5,1} & + & \mathbf{m}_{5,2} & + & \mathbf{m}_{5,3} & + & \mathbf{m}_{5,4} & + & \mathbf{m}_{5,5} & = & \mathbf{m}_5 \\
 & \parallel & & \parallel & & \parallel & & \parallel & & \parallel & & \parallel \\
 & \mathbf{m}_1^* & + & \mathbf{m}_2^* & + & \mathbf{m}_3^* & + & \mathbf{m}_4^* & + & \mathbf{m}_5^* & = & \mathbf{m}
\end{array}
$$

- Users (i, j) share a symmetric key, and can generate a fresh $\mathbf{m}_{i,j}$ each session
- Each user knows all $\mathbf{m}_{i,j}$'s on their corrresponding row and column

Our idea

$$
\begin{array}{c|ccccccccccc}
 & 1 & & 2 & & 3 & & 4 & & 5 & & \\
1 & \mathbf{m}_{1,1} & + & \mathbf{m}_{1,2} & + & \mathbf{m}_{1,3} & + & \mathbf{m}_{1,4} & + & \mathbf{m}_{1,5} & = & \mathbf{m}_1 \\
 & + & & + & & + & & + & & + & & + \\
2 & \mathbf{m}_{2,1} & + & \mathbf{m}_{2,2} & + & \mathbf{m}_{2,3} & + & \mathbf{m}_{2,4} & + & \mathbf{m}_{2,5} & = & \mathbf{m}_2 \\
 & + & & + & & + & & + & & + & & + \\
3 & \mathbf{m}_{3,1} & + & \mathbf{m}_{3,2} & + & \mathbf{m}_{3,3} & + & \mathbf{m}_{3,4} & + & \mathbf{m}_{3,5} & = & \mathbf{m}_3 \\
 & + & & + & & + & & + & & + & & + \\
4 & \mathbf{m}_{4,1} & + & \mathbf{m}_{4,2} & + & \mathbf{m}_{4,3} & + & \mathbf{m}_{4,4} & + & \mathbf{m}_{4,5} & = & \mathbf{m}_4 \\
 & + & & + & & + & & + & & + & & + \\
5 & \mathbf{m}_{5,1} & + & \mathbf{m}_{5,2} & + & \mathbf{m}_{5,3} & + & \mathbf{m}_{5,4} & + & \mathbf{m}_{5,5} & = & \mathbf{m}_5 \\
 & \parallel & & \parallel & & \parallel & & \parallel & & \parallel & & \parallel \\
 & \mathbf{m}^*_1 & + & \mathbf{m}^*_2 & + & \mathbf{m}^*_3 & + & \mathbf{m}^*_4 & + & \mathbf{m}^*_5 & = & \mathbf{m}
\end{array}
$$

- Users (i, j) share a symmetric key, and can generate a fresh $\mathbf{m}_{i,j}$ each session
- Each user knows all $\mathbf{m}_{i,j}$'s on their corrresponding row and column

Our idea

$$
\begin{array}{c|ccccccccccc}
 & 1 & & 2 & & 3 & & 4 & & 5 & & \\
\hline
1 & \mathbf{m}_{1,1} & + & \mathbf{m}_{1,2} & + & \mathbf{m}_{1,3} & + & \mathbf{m}_{1,4} & + & \mathbf{m}_{1,5} & = & \mathbf{m}_1 \\
 & + & & + & & + & & + & & + & & + \\
2 & \mathbf{m}_{2,1} & + & \mathbf{m}_{2,2} & + & \mathbf{m}_{2,3} & + & \mathbf{m}_{2,4} & + & \mathbf{m}_{2,5} & = & \mathbf{m}_2 \\
 & + & & + & & + & & + & & + & & + \\
3 & \mathbf{m}_{3,1} & + & \mathbf{m}_{3,2} & + & \mathbf{m}_{3,3} & + & \mathbf{m}_{3,4} & + & \mathbf{m}_{3,5} & = & \mathbf{m}_3 \\
 & + & & + & & + & & + & & + & & + \\
4 & \mathbf{m}_{4,1} & + & \mathbf{m}_{4,2} & + & \mathbf{m}_{4,3} & + & \mathbf{m}_{4,4} & + & \mathbf{m}_{4,5} & = & \mathbf{m}_4 \\
 & + & & + & & + & & + & & + & & + \\
5 & \mathbf{m}_{5,1} & + & \mathbf{m}_{5,2} & + & \mathbf{m}_{5,3} & + & \mathbf{m}_{5,4} & + & \mathbf{m}_{5,5} & = & \mathbf{m}_5 \\
 & \| & & \| & & \| & & \| & & \| & & \| \\
 & \mathbf{m}^*_1 & + & \mathbf{m}^*_2 & + & \mathbf{m}^*_3 & + & \mathbf{m}^*_4 & + & \mathbf{m}^*_5 & = & \mathbf{m}
\end{array}
$$

- Users (i, j) share a symmetric key, and can generate a fresh $\mathbf{m}_{i,j}$ each session
- Each user knows all $\mathbf{m}_{i,j}$'s on their corrresponding row and column

Our idea

$$
\begin{array}{c|ccccccccccc}
 & 1 & & 2 & & 3 & & 4 & & 5 & & \\
\hline
1 & \mathbf{m}_{1,1} & + & \mathbf{m}_{1,2} & + & \mathbf{m}_{1,3} & + & \mathbf{m}_{1,4} & + & \mathbf{m}_{1,5} & = & \mathbf{m}_1 \\
 & + & & + & & + & & + & & + & & + \\
2 & \mathbf{m}_{2,1} & + & \mathbf{m}_{2,2} & + & \mathbf{m}_{2,3} & + & \mathbf{m}_{2,4} & + & \mathbf{m}_{2,5} & = & \mathbf{m}_2 \\
 & + & & + & & + & & + & & + & & + \\
3 & \mathbf{m}_{3,1} & + & \mathbf{m}_{3,2} & + & \mathbf{m}_{3,3} & + & \mathbf{m}_{3,4} & + & \mathbf{m}_{3,5} & = & \mathbf{m}_3 \\
 & + & & + & & + & & + & & + & & + \\
4 & \mathbf{m}_{4,1} & + & \mathbf{m}_{4,2} & + & \mathbf{m}_{4,3} & + & \mathbf{m}_{4,4} & + & \mathbf{m}_{4,5} & = & \mathbf{m}_4 \\
 & + & & + & & + & & + & & + & & + \\
5 & \mathbf{m}_{5,1} & + & \mathbf{m}_{5,2} & + & \mathbf{m}_{5,3} & + & \mathbf{m}_{5,4} & + & \mathbf{m}_{5,5} & = & \mathbf{m}_5 \\
 & \| & & \| & & \| & & \| & & \| & & \| \\
 & \mathbf{m}^*_1 & + & \mathbf{m}^*_2 & + & \mathbf{m}^*_3 & + & \mathbf{m}^*_4 & + & \mathbf{m}^*_5 & = & \mathbf{m}
\end{array}
$$

- Users (i, j) share a symmetric key, and can generate a fresh $\mathbf{m}_{i,j}$ each session
- Each user knows all $\mathbf{m}_{i,j}$'s on their corrresponding row and column

	1	2	3	4	5	
1	$\mathbf{m}_{1,1}$	$\mathbf{m}_{1,2}$	$\mathbf{m}_{1,3}$	$\mathbf{m}_{1,4}$	$\mathbf{m}_{1,5}$	$= \mathbf{m}_1$
2	$\mathbf{m}_{2,1}$	$\mathbf{m}_{2,2}$	$\mathbf{m}_{2,3}$	$\mathbf{m}_{2,4}$	$\mathbf{m}_{2,5}$	$= \mathbf{m}_2$
3	$\mathbf{m}_{3,1}$	$\mathbf{m}_{3,2}$	$\mathbf{m}_{3,3}$	$\mathbf{m}_{3,4}$	$\mathbf{m}_{3,5}$	$= \mathbf{m}_3$
4	$\mathbf{m}_{4,1}$	$\mathbf{m}_{4,2}$	$\mathbf{m}_{4,3}$	$\mathbf{m}_{4,4}$	$\mathbf{m}_{4,5}$	$= \mathbf{m}_4$
5	$\mathbf{m}_{5,1}$	$\mathbf{m}_{5,2}$	$\mathbf{m}_{5,3}$	$\mathbf{m}_{5,4}$	$\mathbf{m}_{5,5}$	$= \mathbf{m}_5$
	$= \mathbf{m}^*_1$	$= \mathbf{m}^*_2$	$= \mathbf{m}^*_3$	$= \mathbf{m}^*_4$	$= \mathbf{m}^*_5$	$= \mathbf{m}$

- ✔ $(\mathbf{m}_1, \ldots, \mathbf{m}_T, -\mathbf{m}_1, \ldots, -\mathbf{m}^*_T)$ is a secret-sharing of 0
- 🔒 Even if the $\mathbf{m}_i$ are made public and some parties are corrupted, the values $\mathbf{m}^*_i$ of honest parties remain secret.

Second attempt

PQ SHIELD

Threshold Raccoon

- **Round 1:**
  1. Generate uniform masks $\mathbf{m}_{i,j}$
  2. Sample short $\mathbf{r}_i$
  3. $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
  4. $\mathtt{com}_i = H_{\mathtt{com}}(\mathbf{w}_i, \mathtt{msg}, \mathcal{S})$
  5. Broadcast $\mathtt{com}_i$ & $\mathbf{m}_i = \sum_j \mathbf{m}_{i,j}$
- **Round 2:** Broadcast $\mathbf{w}_i$
- **Round 3:**
  1. $\mathbf{w} = \sum_i \mathbf{w}_i$
  2. $c = H(\mathsf{vk}, \mathtt{msg}, \mathbf{w})$
  3. $\mathbf{m}_i^* = \sum_j \mathbf{m}_{j,i}$
  4. $\mathbf{z}_i = \mathbf{r}_i + c \cdot \lambda_i \cdot \mathsf{sk}_i + \mathbf{m}_i^*$
  5. Broadcast $\mathbf{z}_i$
- **Combine:** the final signature is $(c, \mathbf{z} = \sum_{i \in \mathcal{S}} \mathbf{z}_i - \mathbf{m}_i)$

✓ This gives valid Raccoon signatures:

$$\begin{aligned}\mathbf{z} &= \sum_{i\in\mathcal{S}} (\mathbf{z}_i - \mathbf{m}_i) \\ &= \sum_{i\in\mathcal{S}} (\mathbf{r}_i + c \cdot \lambda_i \cdot \mathsf{sk}_i + \mathbf{m}_i^* - \mathbf{m}_i) \\ &= c \cdot \mathsf{sk} + \sum_{i\in\mathcal{S}} \mathbf{r}_i\end{aligned}$$

The previous attack no longer applies

Second attempt

Threshold Raccoon

- **Round 1:**
  1. Generate uniform masks $\mathbf{m}_{i,j}$
  2. Sample short $\mathbf{r}_i$
  3. $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
  4. $\mathtt{com}_i = H_{\mathtt{com}}(\mathbf{w}_i, \mathtt{msg}, \mathcal{S})$
  5. Broadcast $\mathtt{com}_i$ & $\mathbf{m}_i = \sum_j \mathbf{m}_{i,j}$
- **Round 2:** Broadcast $\mathbf{w}_i$ and signature of view of Round 1
- **Round 3:**
  1. $\mathbf{w} = \sum_i \mathbf{w}_i$
  2. $c = H(\mathsf{vk}, \mathtt{msg}, \mathbf{w})$
  3. $\mathbf{m}_i^* = \sum_j \mathbf{m}_{j,i}$
  4. $\mathbf{z}_i = \mathbf{r}_i + c \cdot \lambda_i \cdot \mathsf{sk}_i + \mathbf{m}_i^*$
  5. Broadcast $\mathbf{z}_i$
- **Combine:** the final signature is $(c, \mathbf{z} = \sum_{i \in \mathcal{S}} \mathbf{z}_i - \mathbf{m}_i)$

✓ This gives valid Raccoon signatures:

$$\mathbf{z} = \sum_{i \in \mathcal{S}} (\mathbf{z}_i - \mathbf{m}_i)$$
$$= \sum_{i \in \mathcal{S}} (\mathbf{r}_i + c \cdot \lambda_i \cdot \mathsf{sk}_i + \mathbf{m}_i^* - \mathbf{m}_i)$$
$$= c \cdot \mathsf{sk} + \sum_{i \in \mathcal{S}} \mathbf{r}_i$$

🔒 The previous attack no longer applies

🔒 One last thing: we sign the view of Round 1 to avoid a fork attack

Second attempt

Threshold Raccoon

- **Round 1:**
  1. Generate uniform masks $\mathbf{m}_{i,j}$
  2. Sample short $\mathbf{r}_i$
  3. $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
  4. $\mathtt{com}_i = H_{\mathtt{com}}(\mathbf{w}_i, \mathtt{msg}, \mathcal{S})$
  5. Broadcast $\mathtt{com}_i$ & $\mathbf{m}_i = \sum_j \mathbf{m}_{i,j}$
- **Round 2:** Broadcast $\mathbf{w}_i$ and signature of view of Round 1
- **Round 3:**
  1. $\mathbf{w} = \sum_i \mathbf{w}_i$
  2. $c = H(\mathsf{vk}, \mathtt{msg}, \mathbf{w})$
  3. $\mathbf{m}_i^* = \sum_j \mathbf{m}_{j,i}$
  4. $\mathbf{z}_i = \mathbf{r}_i + c \cdot \lambda_i \cdot \mathsf{sk}_i + \mathbf{m}_i^*$
  5. Broadcast $\mathbf{z}_i$
- **Combine:** the final signature is $(c, \mathbf{z} = \sum_{i \in \mathcal{S}} \mathbf{z}_i - \mathbf{m}_i)$

✓ This gives valid Raccoon signatures:

$$\begin{aligned}\mathbf{z} &= \sum_{i \in \mathcal{S}} (\mathbf{z}_i - \mathbf{m}_i) \\ &= \sum_{i \in \mathcal{S}} (\mathbf{r}_i + c \cdot \lambda_i \cdot \mathsf{sk}_i + \mathbf{m}_i^* - \mathbf{m}_i) \\ &= c \cdot \mathsf{sk} + \sum_{i \in \mathcal{S}} \mathbf{r}_i\end{aligned}$$

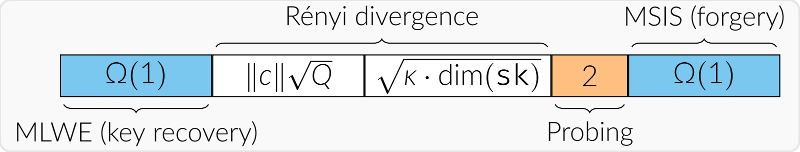
- The previous attack no longer applies
- One last thing: we sign the view of Round 1 to avoid a fork attack
- We can prove security under MSIS and Hint-MLWE

Hint-MLWE?

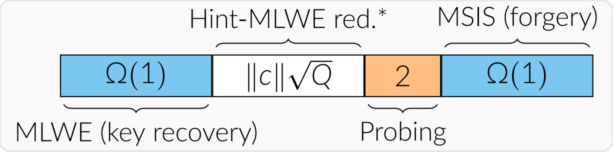
- *Toward Practical Lattice-based Proof of Knowledge from Hint-MLWE* [KLSS23]
- {MLWE + “hints” (essentially signatures)} $\geq$ {MLWE with smaller variance}
- Better parameters than Rényi divergence

Impact on the modulus

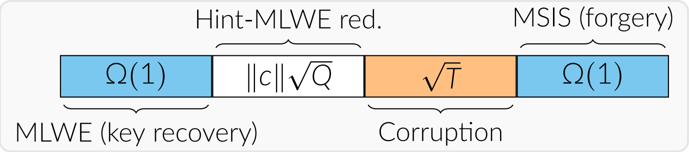
Raccoon
[Rényi]



Raccoon
[Hint-MLWE]



Threshold Raccoon
[Hint-MLWE]



Performances

Bit security	T	|vk|	|sig|	Comm. / Signer	Runtime / Signer
128	4 16 64 256 1024	**3.9 KB**	**12.7 KB**	**40.8 KB**	11 ms 13 ms 24 ms 72 ms 256 ms

Bottom line:

→ Signature size is $\tilde{O}(1)$

→ Communication cost / signer is $\tilde{O}(1)$

→ Runtime / signer is $\tilde{O}(T)$

Conclusion and next steps

Further reading:

- del Pino et al. *Threshold Raccoon: Practical Threshold Signatures from Standard Lattice Assumptions*, EUROCRYPT 2024.
- Espitau, Katsumata and Takemure. *Two-Round Threshold Signature from Algebraic One-More Learning with Errors*, ePrint 2024/496.

Raccoon is the **only** NIST PQC candidate (2017 and 2023 calls) that is easy to thresholdize. Natural next steps:

→ Improved properties (distributed key generation, etc.)

→ NIST MPTC call

Questions?

Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova.
On the (in)security of ROS.
Journal of Cryptology, 35(4):25, October 2022.

Elizabeth C. Crites, Chelsea Komlo, and Mary Maller.
Fully adaptive Schnorr threshold signatures.
In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part I*, volume 14081 of *LNCS*, pages 678–709. Springer, Heidelberg, August 2023.

Manu Drijvers, Kasra Edalatnejad, Bryan Ford, Eike Kiltz, Julian Loss, Gregory Neven, and Igors Stepanovs.
On the security of two-round multi-signatures.
In *2019 IEEE Symposium on Security and Privacy*, pages 1084–1101. IEEE Computer Society Press, May 2019.

Duhyeong Kim, Dongwon Lee, Jinyeong Seo, and Yongsoo Song.
Toward practical lattice-based proof of knowledge from hint-MLWE.
In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 549–580. Springer, Heidelberg, August 2023.