Short Shares, Small Coefficients

A New Secret Sharing Scheme and its Applications to Lattice-based Threshold Cryptography

Guilhem Niot, joint works with *Rafael del Pino, Thomas Espitau,Thomas Prest*

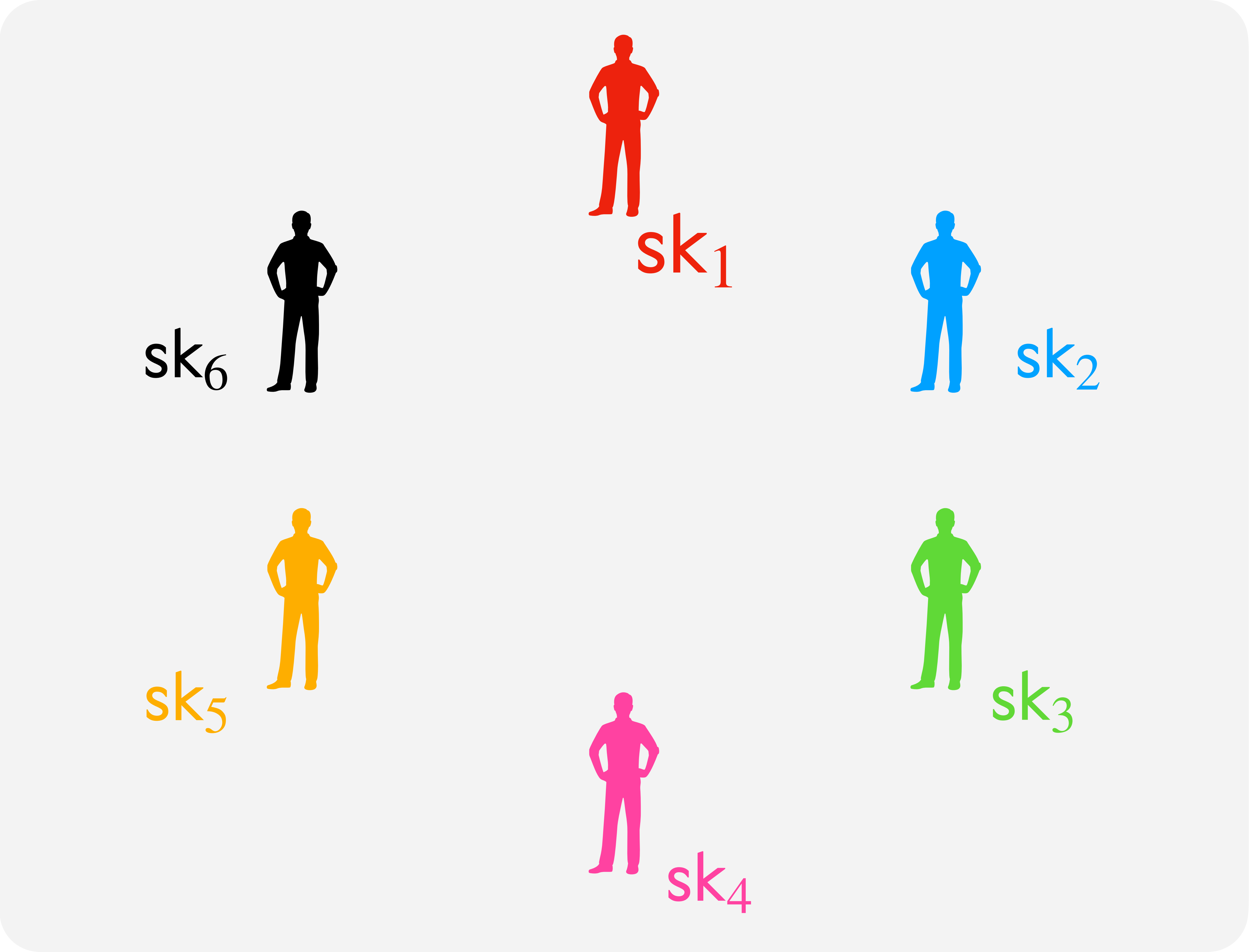
Séminaire ALMASTY - 21. Mar 2025

1. Background

(T-out-of-N) threshold signatures

What are they?

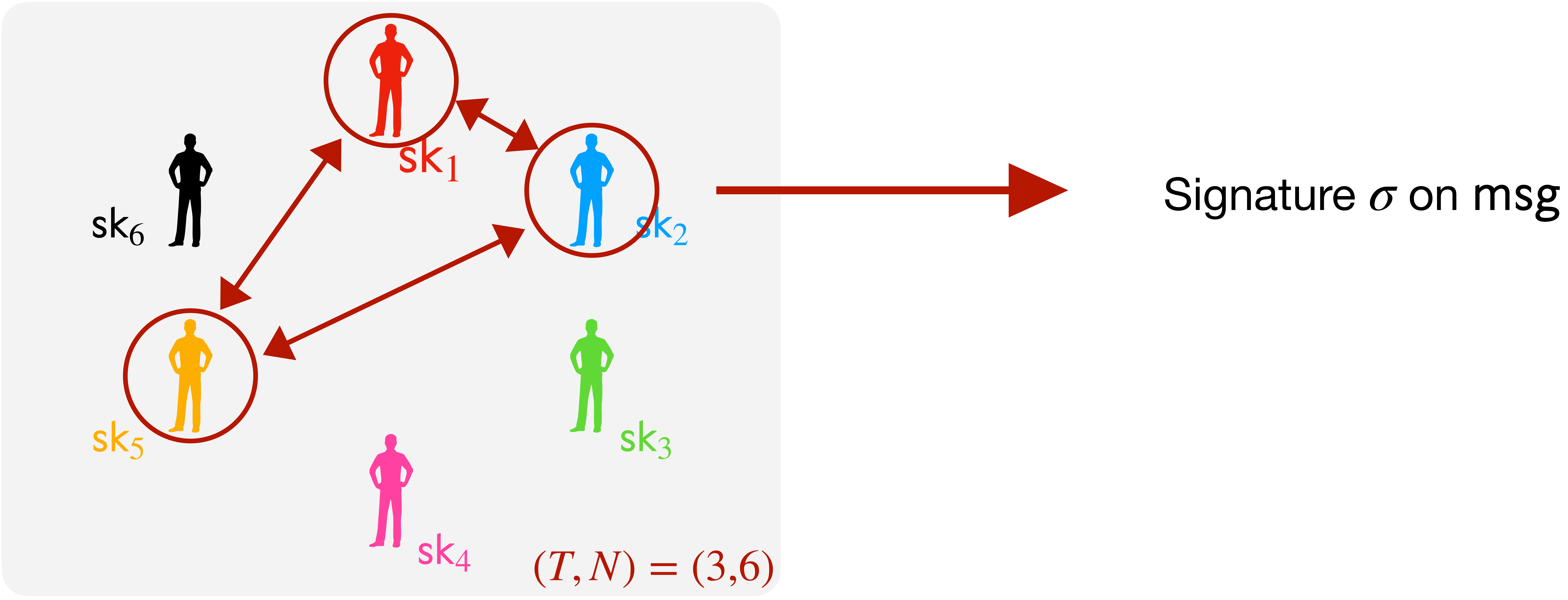
An interactive protocol to distribute signature generation.



- Global verification key vk
- 1 partial signing key sk_i per party
- T-out-of-N:
  - **Correctness:** Any T out of N parties can collaborate to sign a message under vk.
  - **Unforgeability:** $T-1$ corrupted parties cannot sign.

(T-out-of-N) threshold signatures
What are they?

An interactive protocol to distribute signature generation.



Lattice-based Threshold Signatures

An active field of research.

Threshold Raccoon: Practical Threshold Signatures from Standard Lattice Assumptions

Rafael del Pino[1], Shuichi Katsumata[1,2], Mary Maller[1,3], Fabrice Mouhartem[4], Thomas Prest[1], Markku-Juhani Saarinen[1,5]

Two-Round Threshold Signature from Algebraic One-More Learning with Errors

Thomas Espitau[1], Shuichi Katsumata[1,2], Kaoru Takemure* [1,2]

Ringtail: **Practical Two-Round Threshold Signatures from Learning with Errors**

Cecilia Boschini, *ETH Zürich, Switzerland*
Darya Kaviani, *UC Berkeley, USA*
Russell W. F. Lai, *Aalto University, Finland*
Giulio Malavolta, *Bocconi University, Italy*
Akira Takahashi, *JPMorgan AI Research & AlgoCRYPT CoE, USA*
Mehdi Tibouchi, *NTT Social Informatics Laboratories, Japan*

***Flood and Submerse*: Distributed Key Generation and Robust Threshold Signature from Lattices**

Thomas Espitau[1], Guilhem Niot[1,2], and Thomas Prest[1]

Two-round n-out-of-n and Multi-Signatures and Trapdoor Commitment from Lattices*

Ivan Damgård[1], Claudio Orlandi[1], Akira Takahashi[1], and Mehdi Tibouchi[2]

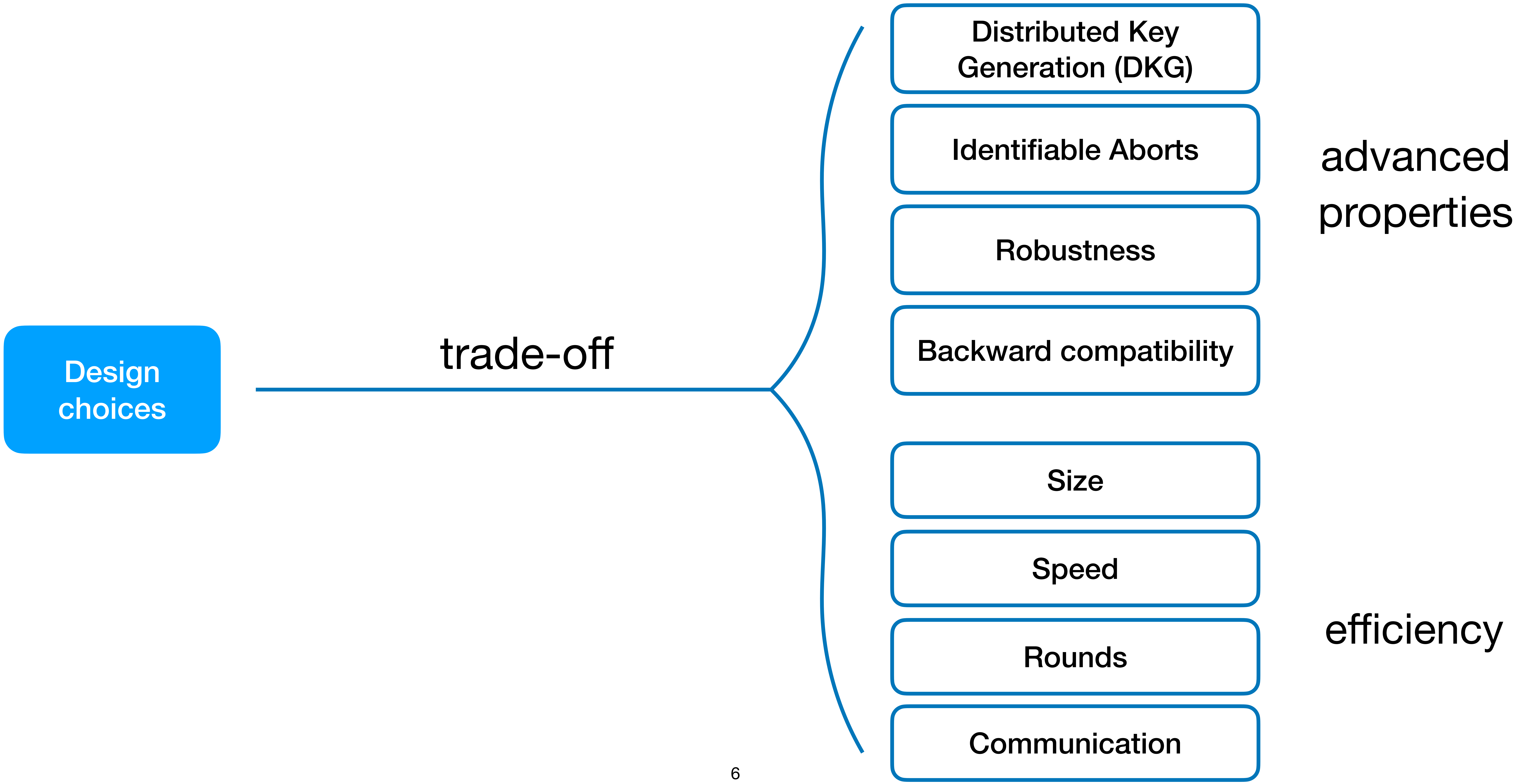
MuSig-L: Lattice-Based Multi-Signature With Single-Round Online Phase*

Cecilia Boschini[1], Akira Takahashi[2], and Mehdi Tibouchi[3]

Two-Round Threshold Lattice-Based Signatures from Threshold Homomorphic Encryption*

Kamil Doruk Gur[1], Jonathan Katz[2**], and Tjerand Silde[3***]

Designing a threshold scheme



Designing a threshold scheme

Lattice-based Threshold Signatures

Candidate schemes

	Hash & Sign	Fiat-Shamir
Gaussian Sampling	Eagle [YJW23]	G+G [DPS23]
Rejection Sampling	Phoenix [JRS24]	Dilithium [LDK+22]
Noise Flooding	Plover [EEN+24]	Raccoon [dEK+24]

Easier to thresholdize ↓

More compact ↑

Lattice-based Threshold Signatures

Candidate schemes

	Hash & Sign	Fiat-Shamir
Gaussian Sampling	Eagle [YJW23]	G+G [DPS23]
Rejection Sampling	Phoenix [JRS24]	Dilithium [LDK+22]
Noise Flooding	Plover [EEN+24]	Raccoon [dEK+24]

Easier to thresholdize ↓

More compact ↑

This talk: Raccoon and Dilithium threshold variants.

Lattice-based Threshold Signatures

An active field of research, with different designs.

Thresholdization technique	Size	Speed	Rounds	Comm/party
MPC	S	Slow	15	≥ 1MB
FHE	M	As fast as FHE	2	≥ 1MB
Tailored	S-M	Fast	2-4	20 kB $\rightarrow 56T$ kB

Lattice-based Threshold Signatures

An active field of research, with different designs.

Thresholdization technique	Size	Speed	Rounds	Comm/party
MPC	S	Slow	15	≥ 1MB
FHE	M	As fast as FHE	2	≥ 1MB
Tailored	S-M	Fast	2-4	20 kB $\rightarrow 56T$ kB

This talk: Tailored

Raccoon

Threshold Raccoon: Practical Threshold Signatures from Standard Lattice Assumptions

Rafael del Pino[1], Shuichi Katsumata[1,2], Mary Maller[1,3], Fabrice Mouhartem[4], Thomas Prest[1], Markku-Juhani Saarinen[1,5]

$\rightarrow$ advanced properties?

Dilithium-like

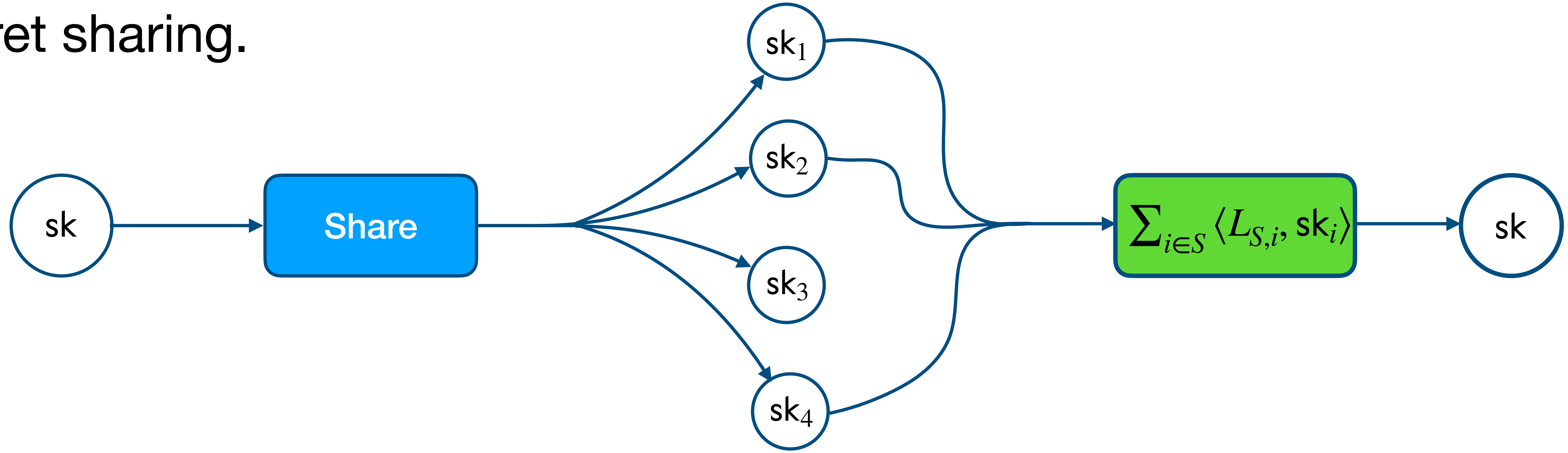
Two-round n-out-of-n and Multi-Si[...] Trapdoor Commitment from Lattices⋆

Ivan Damgård[1], Claudio Orlandi[1], Akira Takahashi[1], and Mehdi Tibouchi[2]

$\rightarrow$ more compact and T-out-of-N?

Main technique of this talk

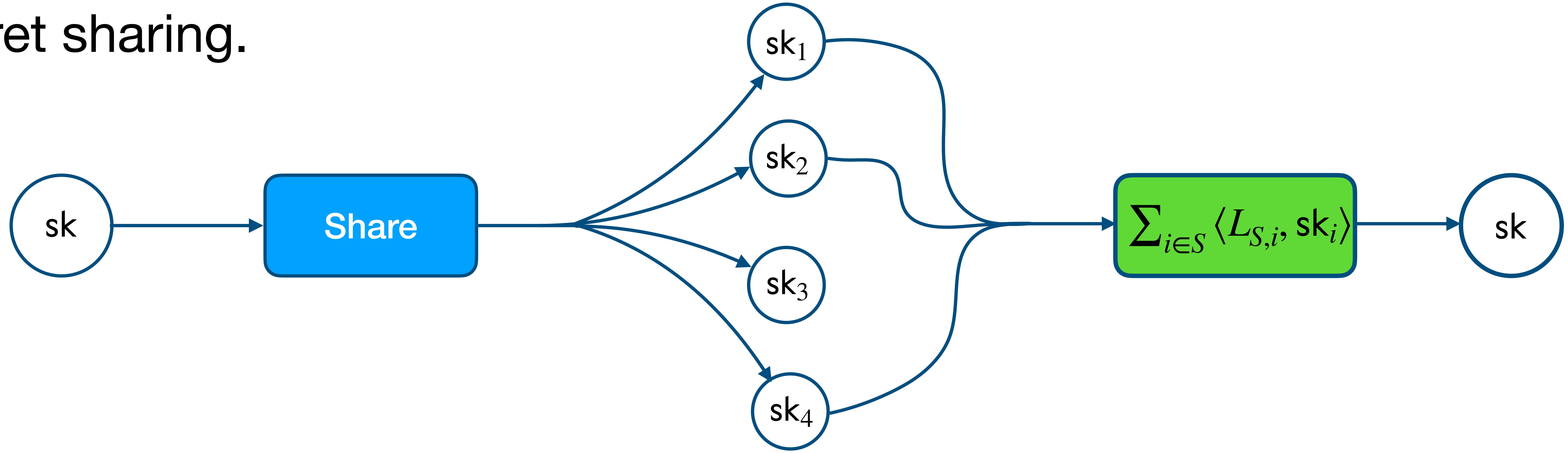
Short secret sharing.



- Individual pool of short shares $\mathsf{sk}_i = (\mathbf{s}_i^{(1)}, \mathbf{s}_i^{(2)}, \ldots)$
- T shares: can recover sk
  - Reconstruction vector $L_{S,i}$ with small coefficients
- $\leq T - 1$ shares: can't recover sk

Main technique of this talk

Short secret sharing.



- Individual pool of short shares $\mathsf{sk}_i = (\mathbf{s}_i^{(1)}, \mathbf{s}_i^{(2)}, \ldots)$
- T shares: can recover sk
  - Reconstruction vector $L_{S,i}$ with small coefficients
- $\leq T - 1$ shares: can't recover sk

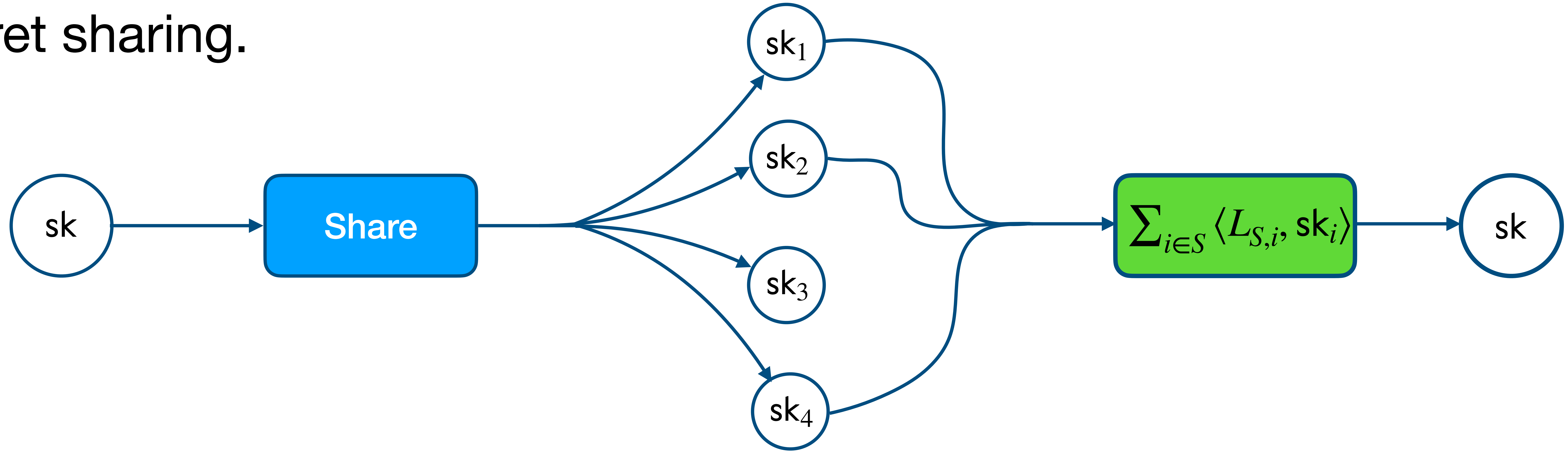
Example: N-out-of-N sharing (one share per party)

- $\mathsf{sk}_1, \ldots, \mathsf{sk}_N \leftarrow \mathscr{D}_\sigma^N$ and $\mathsf{sk} = \sum_i \mathsf{sk}_i$
- $L_{S,i} = 1$

Extends to T-out-of-N by having several shares per party.

Main technique of this talk

Short secret sharing.



- Individual pool of short shares $\mathsf{sk}_i = (\mathbf{s}_i^{(1)}, \mathbf{s}_i^{(2)}, \ldots)$
- T shares: can recover sk
  - Reconstruction vector $L_{S,i}$ with small coefficients
- $\leq T - 1$ shares: can't recover sk

Applications:

- Identifiable aborts in Threshold Raccoon
- A compact Dilithium-like Threshold Signature

2. Threshold Raccoon

Threshold Raccoon: Practical Threshold Signatures from Standard Lattice Assumptions

Rafael del Pino[1], Shuichi Katsumata[1,2], Mary Maller[1,3], Fabrice Mouhartem[4], Thomas Prest[1], Markku-Juhani Saarinen[1,5]

Raccoon signature scheme

Raccoon . Keygen() $\rightarrow$ sk, vk

- vk $= [\mathbf{A} \quad \mathbf{I}] \cdot$ sk, for sk short

Raccoon . Sign(sk, msg) $\rightarrow$ sig

- Sample a short $\mathbf{r}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \text{msg})$
- $\mathbf{z} = c \cdot \text{sk} + \mathbf{r}$
- Output sig $= (c, \mathbf{z})$

Raccoon . Verify(vk, msg, sig $= (c, \mathbf{z})$)

- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{z} - c \cdot \text{vk}$
- Assert $c = H(\mathbf{w}, \text{msg})$
- Assert $\mathbf{z}$ short

* omitting usual rounding techniques

Raccoon signature scheme

Raccoon . Keygen() $\to$ sk, vk

- vk $= [\mathbf{A} \quad \mathbf{I}] \cdot$ sk, for sk short

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

- Sample a short $\mathbf{r}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \text{msg})$
- $\mathbf{z} = c \cdot \text{sk} + \mathbf{r}$
- Output sig $= (c, \mathbf{z})$

Raccoon . Verify(vk, msg, sig $= (c, \mathbf{z})$)

- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{z} - c \cdot \text{vk}$
- Assert $c = H(\mathbf{w}, \text{msg})$
- Assert $\mathbf{z}$ short

Unforgeable assuming

- **Hint-MLWE**
- **SelfTargetMSIS**

Hint-MLWE assumption [KLSS23].

$(\mathbf{A}, \text{vk})$ is pseudorandom even if given Q “hints”:

$$(c_i, \mathbf{z}_i := c_i \cdot \text{sk} + \mathbf{r}_i) \text{ for } i \in [Q]$$

As hard as MLWE$_\sigma$ if

$$\sigma_{\mathbf{r}} \geq \sqrt{Q} \cdot \|c\| \cdot \sigma$$

Threshold Raccoon

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

- $\text{vk} = [\mathbf{A} \quad \mathbf{I}] \cdot \text{sk}$, for sk short

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

- Sample a short $\mathbf{r}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \text{msg})$
- $\mathbf{z} = c \cdot \text{sk} + \mathbf{r}$
- Output $\text{sig} = (c, \mathbf{z})$

$\text{Raccoon.Verify}(\text{vk}, \text{msg}, \text{sig} = (c, \mathbf{z}))$

- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{z} - c \cdot \text{vk}$
- Assert $c = H(\mathbf{w}, \text{msg})$
- Assert $\mathbf{z}$ short

Shamir sharing on secret $\text{sk} \in \mathcal{R}_q^\ell$

Sample polynomial $f \in \mathcal{R}_q^\ell[X]$ s.t.

- $f(0) = \text{sk}$ and $\deg f \leq T - 1$
- Partial signing keys $\text{sk}_i := [\![\text{sk}]\!]_i = f(i)$

Properties:

- with $< T$ shares, sk is perfectly hidden
- with a set S of $\geq T$ shares, reconstruct sk via Lagrange interpolation

$$\text{sk} = \sum_{i \in S} L_{S,i} \cdot [\![\text{sk}]\!]_i$$

Threshold Raccoon

Raccoon . Keygen() → sk, vk

- vk $= [\mathbf{A} \quad \mathbf{I}] \cdot$ sk, for sk short

Raccoon . Sign(sk, msg) → sig

- Sample a short $\mathbf{r}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \text{msg})$
- $\mathbf{z} = c \cdot \text{sk} + \mathbf{r}$
- Output sig $= (c, \mathbf{z})$

Raccoon . Verify(vk, msg, sig $= (c, \mathbf{z})$)

- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{z} - c \cdot \text{vk}$
- Assert $c = H(\mathbf{w}, \text{msg})$
- Assert $\mathbf{z}$ short

First (insecure) attempt

ThRaccoon . Sign(sk, msg) → sig

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\text{cmt}_i = H_{\text{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \text{msg})$
- Broadcast $\mathbf{z}_i = L_{S,i} \cdot c \cdot [\![sk]\!]_i + \mathbf{r}_i$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Threshold Raccoon

- Prevent ROS attack with commit-reveal of $\mathbf{w}_i$

First (insecure) attempt

$\mathsf{ThRaccoon.Sign}(\mathsf{sk}, \mathsf{msg}) \to \mathsf{sig}$

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = L_{S,i} \cdot c \cdot [\![sk]\!]_i + \mathbf{r}_i$

Combine: the final signature is

$$(c, \sum_{i \in S} \mathbf{z}_i)$$

Threshold Raccoon

- Prevent ROS attack with commit-reveal of $\mathbf{w}_i$
- But, $\mathbf{r}_i$ is small vs $L_{S,i} \cdot c \cdot [\![\mathsf{s}k]\!]_i$ is large

$\rightarrow$ Leaks $[\![\mathsf{s}k]\!]_i$

First (insecure) attempt

$\mathsf{ThRaccoon.Sign}(\mathsf{sk}, \mathsf{msg}) \rightarrow \mathsf{sig}$

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = L_{S,i} \cdot c \cdot [\![\mathsf{s}k]\!]_i + \mathbf{r}_i$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Threshold Raccoon

- Prevent ROS attack with commit-reveal of $\mathbf{w}_i$
- But, $\mathbf{r}_i$ is small vs $L_{S,i} \cdot c \cdot [\![sk]\!]_i$ is large

  $\rightarrow$ Leaks $[\![sk]\!]_i$

- Solution: add a zero-share Δ_i:
  - Derived with a PRF, using pre-shared pairwise keys
  - Any set of $< T$ values Δ_i is uniformly random
  - $\sum_{i \in S} \Delta_i = 0$

ThRaccoon . Sign(sk, msg) $\rightarrow$ sig

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = L_{S,i} \cdot c \cdot [\![sk]\!]_i + \mathbf{r}_i + \Delta_i$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Threshold Raccoon, a practical threshold signature

Speed	Rounds	| vk |	| sig |	Total communication
Fast	3	4 kB	13 kB	40 kB

… but does not provide a DKG, or robustness / identifiable aborts.

3. Another direction for ThRaccoon

Flood and Submerse: Distributed Key Generation and Robust Threshold Signature from Lattices

Thomas Espitau[1], Guilhem Niot[1,2], and Thomas Prest[1]

How to Shortly Share a Short Vector

DKG with Short Shares and Application to Lattice-Based Threshold Signatures with Identifiable Aborts

Rafael del Pino[1], Thomas Espitau[1], Guilhem Niot[1,2], and Thomas Prest[1]

Challenge of detecting malicious behaviour in ThRaccoon

$\mathsf{ThRaccoon.Sign}(\mathsf{sk}, \mathsf{msg}) \rightarrow \mathsf{sig}$

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Compute zero-share Δ_i
- Broadcast $\mathbf{z}_i = L_{S,i} \cdot c \cdot [\![\mathsf{sk}]\!]_i + \mathbf{r}_i + \Delta_i$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Why is it challenging to tackle malicious behaviour to ThRaccoon?

- Main issue: computation of Δ_i using PRF to hide the secret when using Shamir sharing.

Challenge of detecting malicious behaviour in ThRaccoon

$\mathsf{ThRaccoon.Sign}(\mathsf{sk}, \mathsf{msg}) \to \mathsf{sig}$

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Compute zero-share Δ_i
- Broadcast $\mathbf{z}_i = L_{S,i} \cdot c \cdot [\![\mathsf{sk}]\!]_i + \mathbf{r}_i + \Delta_i$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Let's take a step back!

The key challenge in ThRaccoon is to hide a secret $L_{S,i} \cdot [\![\mathsf{sk}]\!]_i$ with the randomness $\mathbf{r}_i$.

Direction 1 (Threshold Raccoon):

- The shares of sk are **uniform**
- The randomness shares $\mathbf{r}_i$ are **short**

A **uniform** zero-share Δ_i is added to partial signatures to hide $L_{S,i} \cdot [\![\mathsf{sk}]\!]_i$.

Direction 2: Can we make both $L_{S,i} \cdot [\![\mathsf{sk}]\!]_i$ and $\mathbf{r}_i$ uniform?

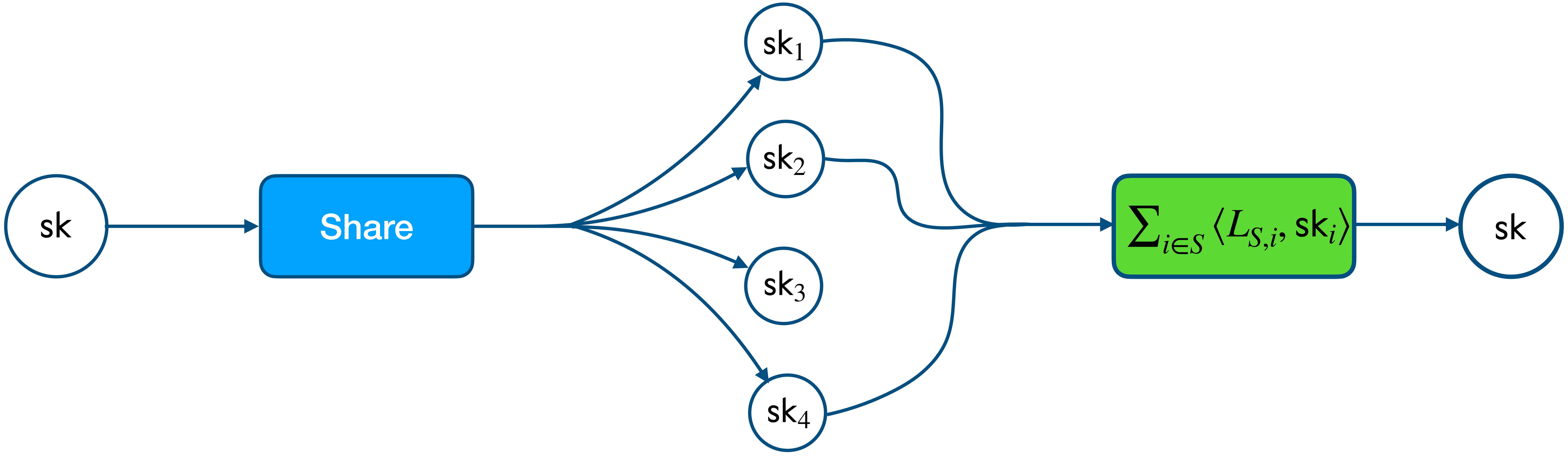
- Use Shamir-sharing for both sk and $\mathbf{r}$ $\to$ Flood and submerse [ENP24]

Direction 3: Can we make both $L_{S,i} \cdot [\![\mathsf{sk}]\!]_i$ and $\mathbf{r}_i$ short?

- Use a **short secret-sharing** for both sk and $\mathbf{r}$

With Short Secret Sharing

- Another approach relies on sampling a sharing of sk such that we have:
  - Individual pool of short shares $\mathsf{sk}_i = (\mathbf{s}_i^{(1)}, \mathbf{s}_i^{(2)}, \ldots)$
  - T shares: can recover sk + reconstruction vector $L_{S,i}$ with small coefficients
  - $\leq T - 1$ shares: can't recover sk



With Short Secret Sharing

ShortSS . Sign(sk, msg) → sig

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = c \cdot \langle L_{S,i}, \mathsf{sk}_i \rangle + \mathbf{r}_i$

Combine: the final signature is

$$(c, \sum_{i \in S} \mathbf{z}_i)$$

Security.

- $c \cdot \langle L_{S,i}, \mathsf{sk}_i \rangle$ is short → $\mathbf{r}_i$ hides it.
  - Prove security with Hint-MLWE

With Short Secret Sharing

ShortSS . Sign(sk, msg) $\rightarrow$ sig

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = c \cdot \langle L_{S,i}, \mathsf{sk}_i \rangle + \mathbf{r}_i$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Security.

- $c \cdot \langle L_{S,i}, \mathsf{sk}_i \rangle$ is short $\rightarrow$ $\mathbf{r}_i$ hides it.
  - Prove security with Hint-MLWE

Identifiable aborts.

- Each $\mathsf{vk}_i^{(j)} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{s}_i^{(j)}$ is a valid public key ($\mathbf{s}_i^{(j)}$ is short), for $\mathsf{sk}_i = (\mathbf{s}_i^{(1)}, \mathbf{s}_i^{(2)}, \ldots)$

  $\rightarrow$ Each $(c, \mathbf{z}_i)$ is a valid signature for $\langle L_{S,i}, (\mathsf{vk}_i^{(j)})_j \rangle$
- Identifiable abort is as easy as verifying partial signatures!
- *Akin to abort identification in Sparkle (Threshold Schnorr): perform partial verifications.*

With Short Secret Sharing

Instantiating this scheme.

- In the T-out-of-N setting, the number of shares grows with $\binom{N}{T-1}$, this scheme thus only supports a small number of parties.

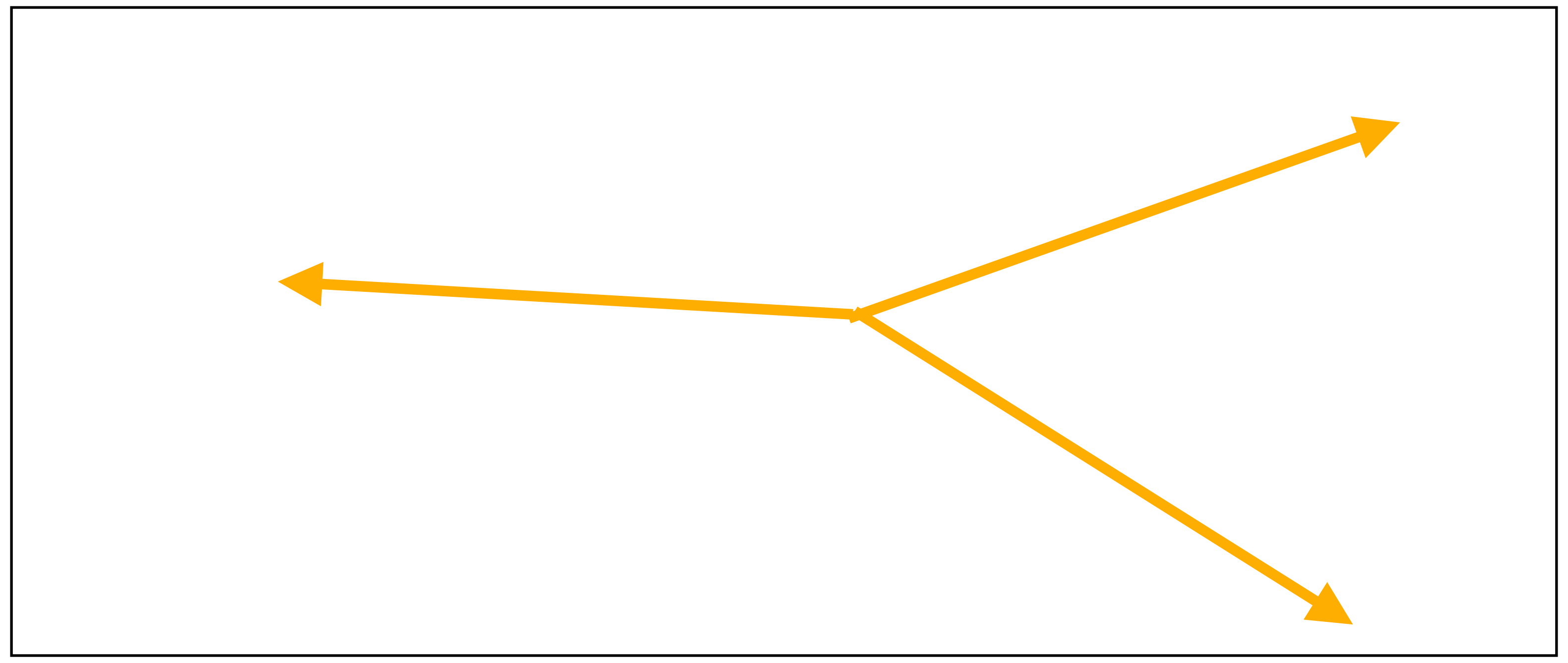
For $N \leq 16$,

<table>
<tr><th>Phase</th><th># rounds</th><th>| vk |</th><th>| sig |</th><th>Total communication</th></tr>
<tr><td>Signing</td><td>3</td><td rowspan="2">4 kB</td><td rowspan="2">11 kB</td><td>25 kB</td></tr>
<tr><td>Abort Identification</td><td>0</td><td></td></tr>
</table>

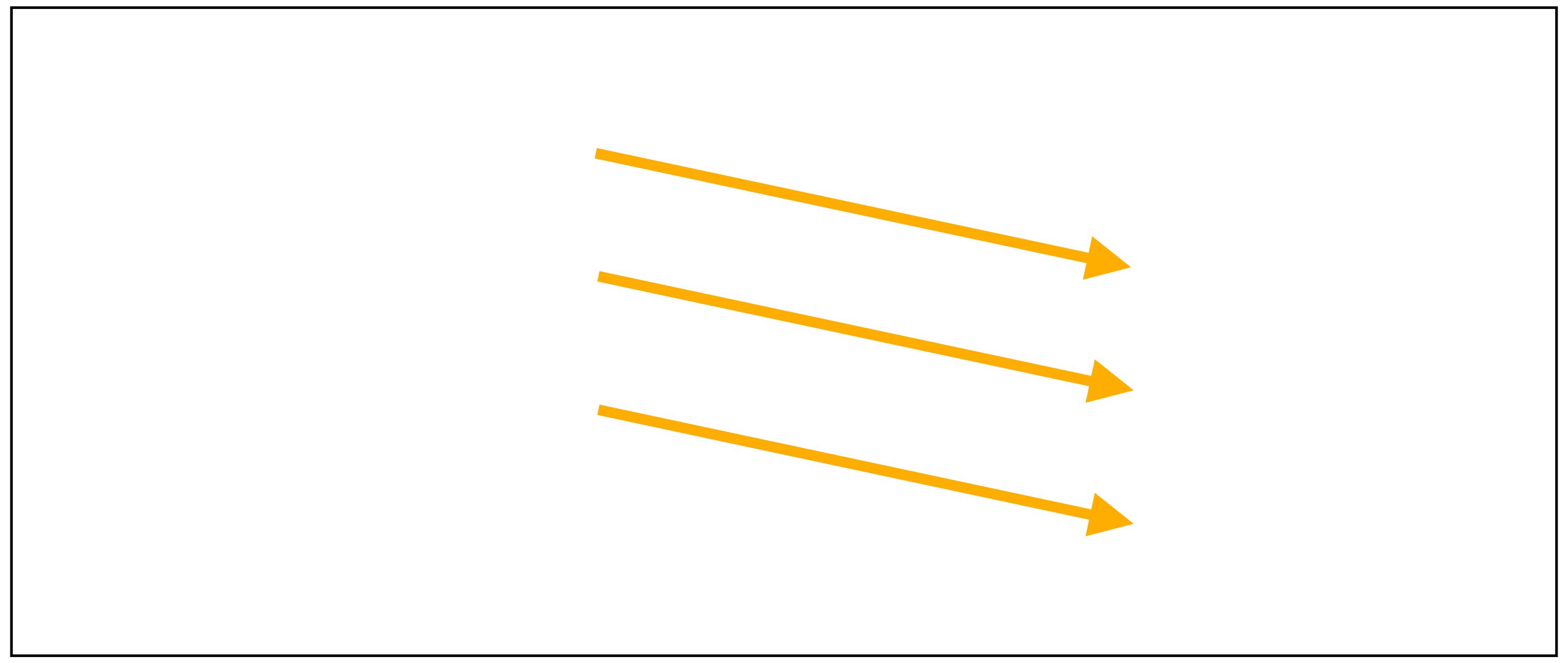
Bonus: tighter check bounds using Short SS

Looking in more detail, the correctness of the previous schemes relies on the shortness of $\mathbf{z} = \sum_i \mathbf{z}_i$.

What can we say about the norm of T Gaussians?



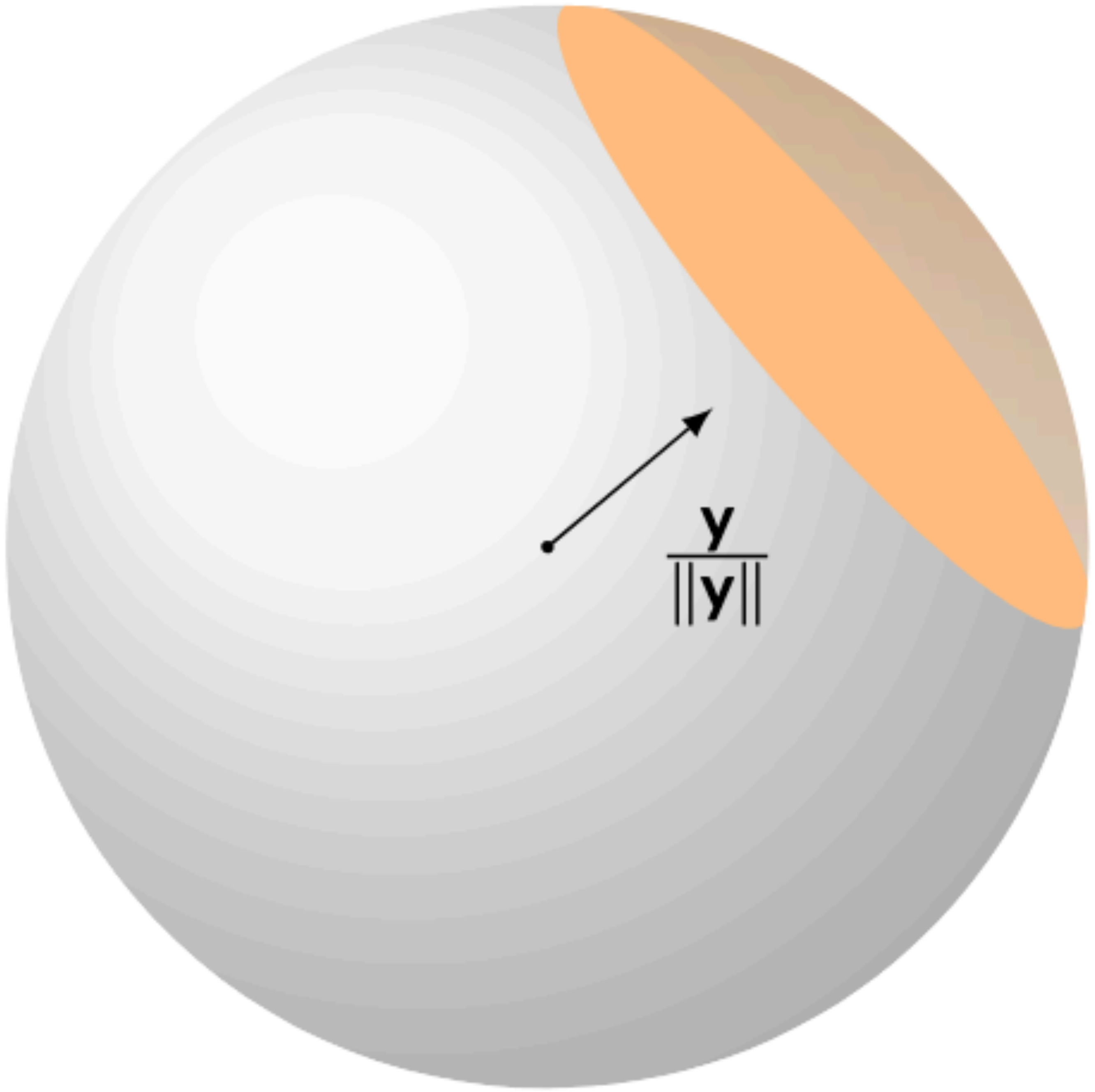
Average-case: $O(\sqrt{T})$



Worst-case: $O(T)$

- When users are honest: average-case.
- Colliding malicious users can force worst-case.

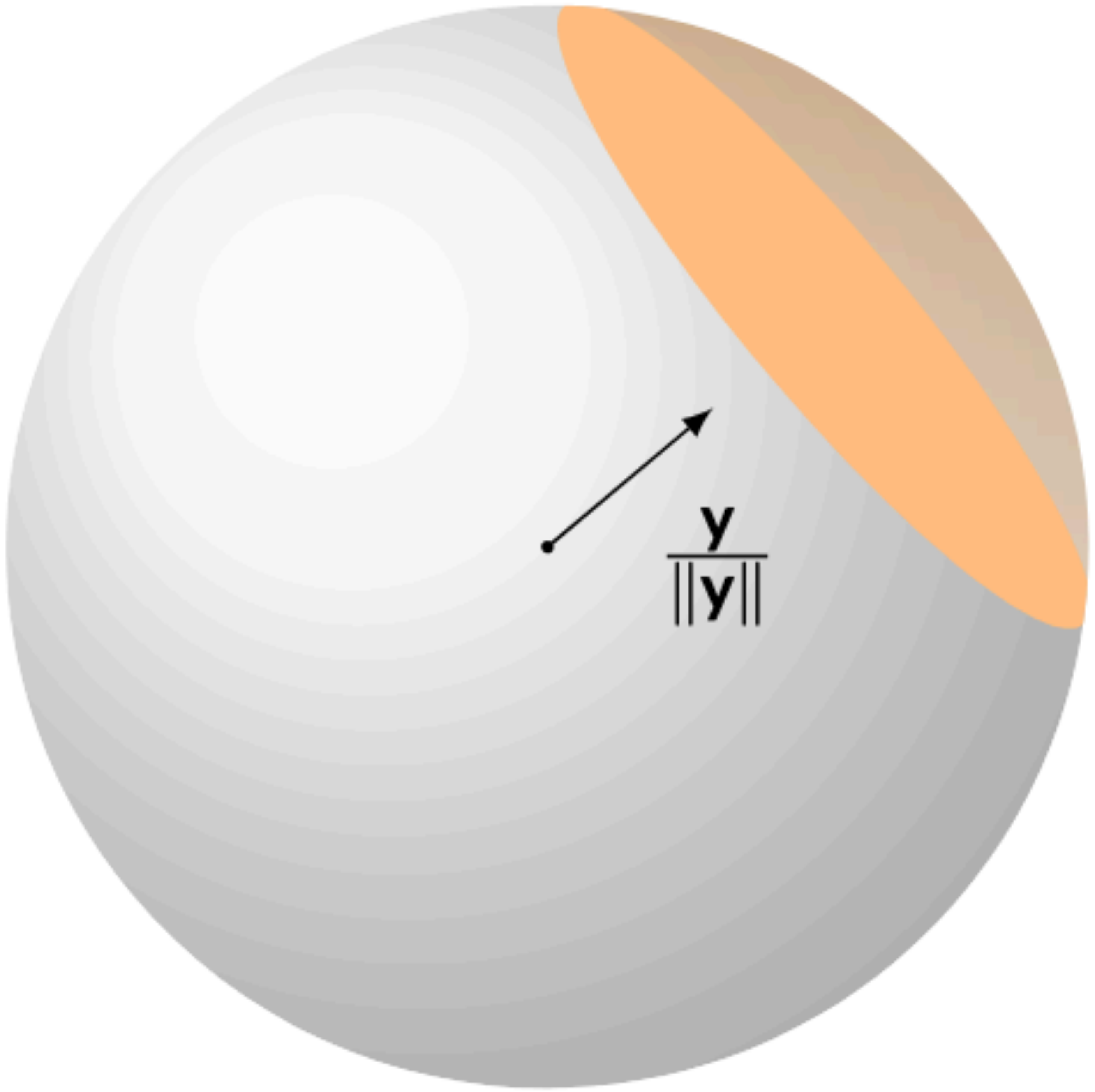
The Death Star Algorithm



If $\mathbf{x} \leftarrow \mathscr{D}_\sigma$,

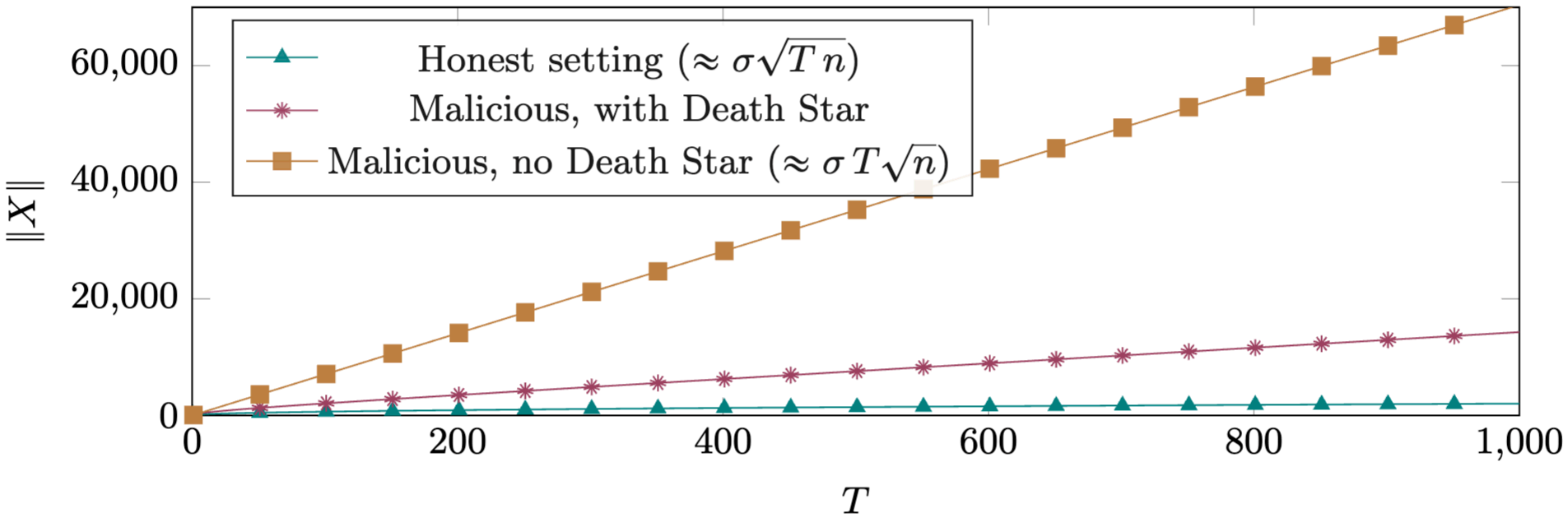
- For any vector $\mathbf{y}$, $\langle \mathbf{x}, \mathbf{y} \rangle \lessapprox \|x\| \|y\| / \sqrt{n/\lambda}$ except with probability $2^{-\lambda}$.

The Death Star Algorithm



If $\mathbf{x} \leftarrow \mathscr{D}_\sigma$,

- For any vector $\mathbf{y}$, $\langle \mathbf{x}, \mathbf{y} \rangle \lessapprox \|x\| \|y\| / \sqrt{n/\lambda}$ except with probability $2^{-\lambda}$.



Norm of $\mathbf{x} = \sum_i \mathbf{x}_i$ for $\sigma = 1$, $n = 4096$, 128 bits of security, and $T \leq 1000$

4. Compact Dilithium-like Threshold Signatures

Finally! A Compact Lattice-Based Threshold Signature

Rafael del Pino[1] and Guilhem Niot[1,2]

Fiat-Shamir with Aborts signature

Rej$(\mathbf{v}, \chi_r, \chi_z, M) \rightarrow \mathbf{z} \mid \perp$

- $\mathbf{r} \leftarrow \chi_{\mathbf{r}}$
- $\mathbf{z} = \mathbf{v} + \mathbf{r}$
- $b \leftarrow \mathscr{B}\left(\max\left(\frac{\chi_{\mathbf{z}}(\mathbf{z})}{M\chi_{\mathbf{r}}(\mathbf{r})}, 1\right)\right)$
- If $b = 0$ then $\mathbf{z} = \perp$
- Return $\mathbf{z}$

Ideal$(\chi_z, M) \rightarrow \mathbf{z} \mid \perp$

- $\mathbf{z} \leftarrow \chi_{\mathbf{z}}$
- $b \leftarrow \mathscr{B}\left(\frac{1}{M}\right)$
- If $b = 0$ then $\mathbf{z} = \perp$
- Return $\mathbf{z}$

For proper parameters, Rej$(\mathbf{v}, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M) \sim$ Ideal$(\chi_{\mathbf{z}}, M)$.

$\rightarrow$ distribution of $\mathbf{z}$ is independent of the secret value $\mathbf{v}$

Fiat-Shamir with Aborts signature

$\mathsf{Rej}(\mathbf{v}, \chi_r, \chi_z, M; \mathbf{r}) \to \mathbf{z} \mid \perp$

- $\mathbf{z} = \mathbf{v} + \mathbf{r}$
- $b \leftarrow \mathscr{B}\left(\max\left(\dfrac{\chi_{\mathbf{z}}(\mathbf{z})}{M\chi_{\mathbf{r}}(\mathbf{r})}, 1\right)\right)$
- If $b = 0$ then $\mathbf{z} = \perp$
- Return $\mathbf{z}$

$\mathsf{FSwA}.\mathsf{Sign}(\mathsf{sk}, \mathsf{msg}) \to \mathsf{sig}$

- $\mathbf{r} \leftarrow \chi_{\mathbf{r}}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \mathsf{msg})$
- $\mathbf{z} = \mathsf{Rej}(c \cdot \mathsf{sk}, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r})$
- If $\mathbf{z} = \perp$ then **restart**
- Return $(c, \mathbf{z})$

$\mathsf{FSwA}.\mathsf{Verify}(\mathsf{vk}, \mathsf{msg}, \mathsf{sig} = (c, \mathbf{z}))$

- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{z} - c \cdot \mathsf{vk}$
- Assert $c = H(\mathbf{w}, \mathsf{msg})$
- Assert $\mathbf{z}$ short

In the ROM, the distribution of signatures of the above scheme is independent of the secret sk.

$\to$ allows to prove unforgeability

Threshold FSwA signature?

FSwA . Sign(sk, msg) $\to$ sig

- $\mathbf{r} \leftarrow \chi_{\mathbf{r}}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \mathsf{msg})$
- $\mathbf{z} = \mathsf{Rej}(c \cdot \mathsf{sk}, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r})$
- If $\mathbf{z} = \perp$ then **restart**
- Return $(c, \mathbf{z})$

TH-FSwA . Sign(sk, msg) $\to$ sig

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = \mathsf{Rej}(c \cdot \mathsf{sk}_i, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r}_i)$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Intuition N-out-of-N setting: $\mathsf{sk} = \sum_i \mathsf{sk}_i$

Threshold FSwA signature?

$\mathsf{FSwA}.\mathsf{Sign}(\mathsf{sk}, \mathsf{msg}) \to \mathsf{sig}$

- $\mathbf{r} \leftarrow \chi_{\mathbf{r}}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \mathsf{msg})$
- $\mathbf{z} = \mathsf{Rej}(c \cdot \mathsf{sk}, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r})$
- If $\mathbf{z} = \perp$ then **restart**
- Return $(c, \mathbf{z})$

- $\mathbf{w}_i$ is revealed even in case of rejection
  - Need proof strategy to show independence from secret
  - [DOTT22] hides rejected $\mathbf{w}_i$ with a trapdoor commitment scheme
  - [BTT22] simulates rejected $\mathbf{w}_i$ but with regularity lemma (degraded parameters)

$\mathsf{TH\text{-}FSwA}.\mathsf{Sign}(\mathsf{sk}, \mathsf{msg}) \to \mathsf{sig}$

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = \mathsf{Rej}(c \cdot \mathsf{sk}_i, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r}_i)$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Intuition N-out-of-N setting: $\mathsf{sk} = \sum_i \mathsf{sk}_i$

Threshold FSwA signature?

$\mathsf{FSwA.Sign}(\mathsf{sk}, \mathsf{msg}) \rightarrow \mathsf{sig}$

- $\mathbf{r} \leftarrow \chi_{\mathbf{r}}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \mathsf{msg})$
- $\mathbf{z} = \mathsf{Rej}(c \cdot \mathsf{sk}, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r})$
- If $\mathbf{z} = \perp$ then **restart**
- Return $(c, \mathbf{z})$

- $\mathbf{w}_i$ is revealed even in case of rejection
  - Need proof strategy to show independence from secret
  - [DOTT22] hides rejected $\mathbf{w}_i$ with a trapdoor commitment scheme
  - [BTT22] simulates rejected $\mathbf{w}_i$ but with regularity lemma (degraded parameters)

→ Tighter simulation lemma

$\mathsf{TH\text{-}FSwA.Sign}(\mathsf{sk}, \mathsf{msg}) \rightarrow \mathsf{sig}$

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = \mathsf{Rej}(c \cdot \mathsf{sk}_i, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r}_i)$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Intuition N-out-of-N setting: $\mathsf{sk} = \sum_i \mathsf{sk}_i$

Threshold FSwA signature?

Lemma: Rejected $\mathbf{w}_i$ is indistinguishable from uniform if:

- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$, with $\mathbf{r} \leftarrow \chi_{\mathbf{r}}$ is indistinguishable from uniform
- $[\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{z}$, with $\mathbf{z} \leftarrow \chi_{\mathbf{z}}$ is indistinguishable from uniform

Threshold FSwA signature?

FSwA . Sign(sk, msg) $\to$ sig

- $\mathbf{r} \leftarrow \chi_{\mathbf{r}}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \text{msg})$
- $\mathbf{z} = \text{Rej}(c \cdot \text{sk}, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r})$
- If $\mathbf{z} = \perp$ then **restart**
- Return $(c, \mathbf{z})$

TH-FSwA . Sign(sk, msg) $\to$ sig

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\text{cmt}_i = H_{\text{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \text{msg})$
- Broadcast $\mathbf{z}_i = \text{Rej}(c \cdot \text{sk}_i, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r}_i)$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

- $\mathbf{w}_i$ is revealed even in case of rejection
  - Need proof strategy to show independence from secret
  - [DOTT22] hides rejected $\mathbf{w}_i$ with a trapdoor commitment scheme
  - [BTT22] simulates rejected $\mathbf{w}_i$ but with regularity lemma (degraded parameters)

$\to$ Tighter simulation lemma

- How to support T-out-of-N?

Threshold FSwA signature?

FSwA . Sign(sk, msg) → sig

- $\mathbf{r} \leftarrow \chi_{\mathbf{r}}$
- $\mathbf{w} = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}$
- $c = H(\mathbf{w}, \mathsf{msg})$
- $\mathbf{z} = \mathsf{Rej}(c \cdot \mathsf{sk}, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r})$
- If $\mathbf{z} = \perp$ then **restart**
- Return $(c, \mathbf{z})$

- $\mathbf{w}_i$ is revealed even in case of rejection
  - Need proof strategy to show independence from secret
  - [DOTT22] hides rejected $\mathbf{w}_i$ with a trapdoor commitment scheme
  - [BTT22] simulates rejected $\mathbf{w}_i$ but with regularity lemma (degraded parameters)

→ Tighter simulation lemma

- How to support T-out-of-N?

→ Use short secret sharing

TH-FSwA . Sign(sk, msg) → sig

Round 1:

- Sample a short $\mathbf{r}_i$
- $\mathbf{w}_i = [\mathbf{A} \quad \mathbf{I}] \cdot \mathbf{r}_i$
- Broadcast $\mathsf{cmt}_i = H_{\mathsf{cmt}}(\mathbf{w}_i)$

Round 2:

- Broadcast $\mathbf{w}_i$

Round 3:

- $\mathbf{w} = \sum_i \mathbf{w}_i$
- $c = H(\mathbf{w}, \mathsf{msg})$
- Broadcast $\mathbf{z}_i = \mathsf{Rej}(c \cdot \langle L_{S,i}, \mathsf{sk}_i \rangle, \chi_{\mathbf{r}}, \chi_{\mathbf{z}}, M; \mathbf{r}_i)$

Combine: the final signature is

$$(c, \textstyle\sum_{i \in S} \mathbf{z}_i)$$

Threshold FSwA signature

For $N \leq 8$,

Distributions	Speed	Rounds	| vk |	| sig |	Total communication
Gaussians	Fast	3	2.6 kB	2.7 kB	5.6 kB
Uniforms			3.1 kB	4.8 kB	13.5 kB

Comparable to Dilithium size: 2.4kB at NIST level II!

4. How to concretely sample short sharings

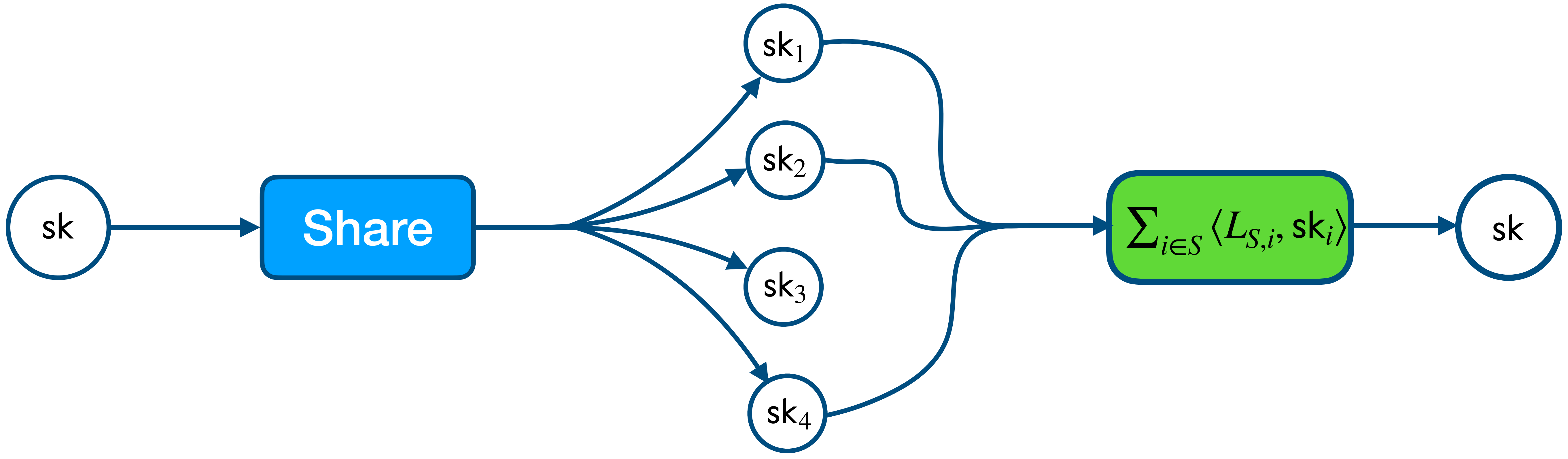
How to Shortly Share a Short Vector

DKG with Short Shares and Application to Lattice-Based Threshold Signatures with Identifiable Aborts

Rafael del Pino[1], Thomas Espitau[1], Guilhem Niot[1,2], and Thomas Prest[1]

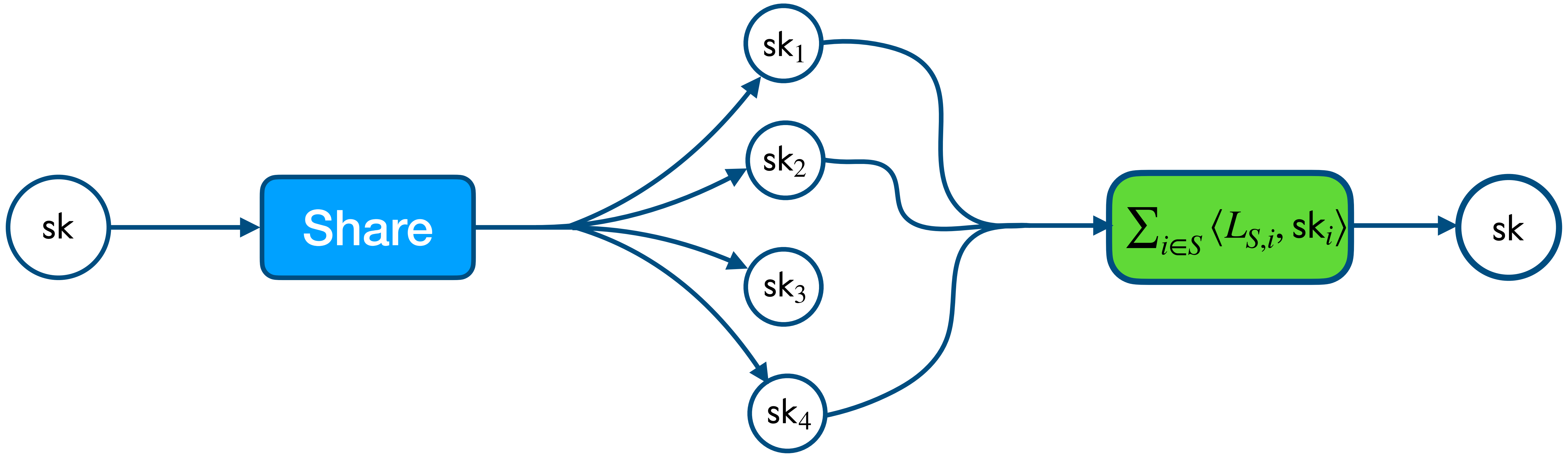
Short Secret Sharing

- Individual pool of short shares $\mathsf{sk}_i = (\mathbf{s}_i^{(1)}, \mathbf{s}_i^{(2)}, \ldots)$
- T shares: can recover sk + reconstruction vector $L_{S,i}$ with small coefficients
- $\leq T - 1$ shares: can't recover sk



Short Secret Sharing

- Individual pool of short shares $\mathsf{sk}_i = (\mathbf{s}_i^{(1)}, \mathbf{s}_i^{(2)}, \ldots)$
- T shares: can recover sk + reconstruction vector $L_{S,i}$ with small coefficients
- $\leq T - 1$ shares: can't recover sk



Observation: hard to not leak the secret with these constraints…

But, lattice-based schemes, often just need $[\mathbf{A} \quad \mathbf{I}] \cdot \mathsf{sk}$ to look uniform.

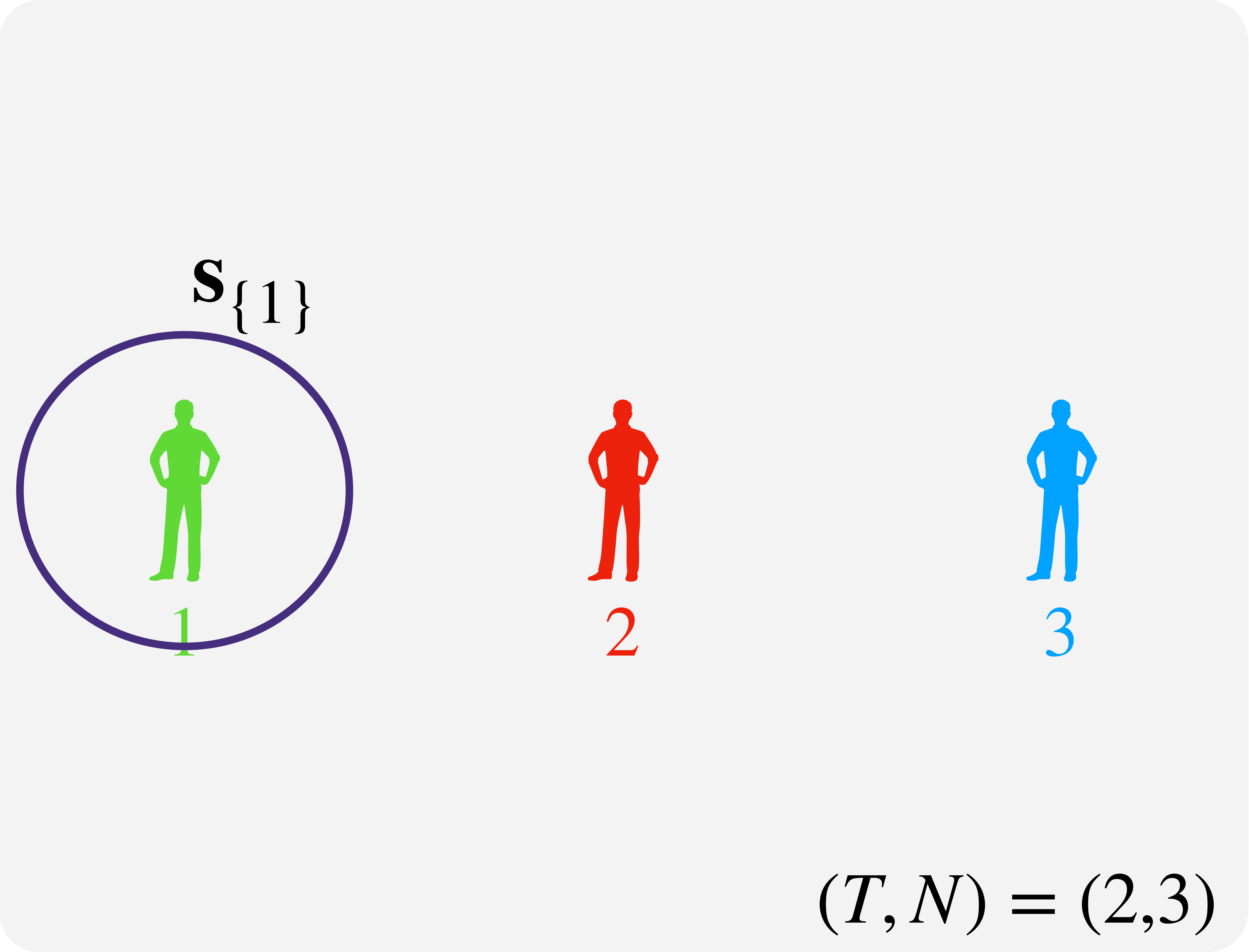
We can:

- Leak an offset of the secret: $\mathsf{sk} = \mathsf{sk}_{\mathsf{safe}} + \mathsf{sk}_{\mathsf{leak}}$
- Leak hints on the secrets $h = c \cdot \mathsf{sk} + y$, for large enough y

Solution 1: Replicated Secret Sharing

Idea: sample a share for any possible set of corrupted parties.

1. For any set $\mathcal{T}$ of $T-1$ parties, sample a uniform share $\mathbf{s}_{\mathcal{T}}$.

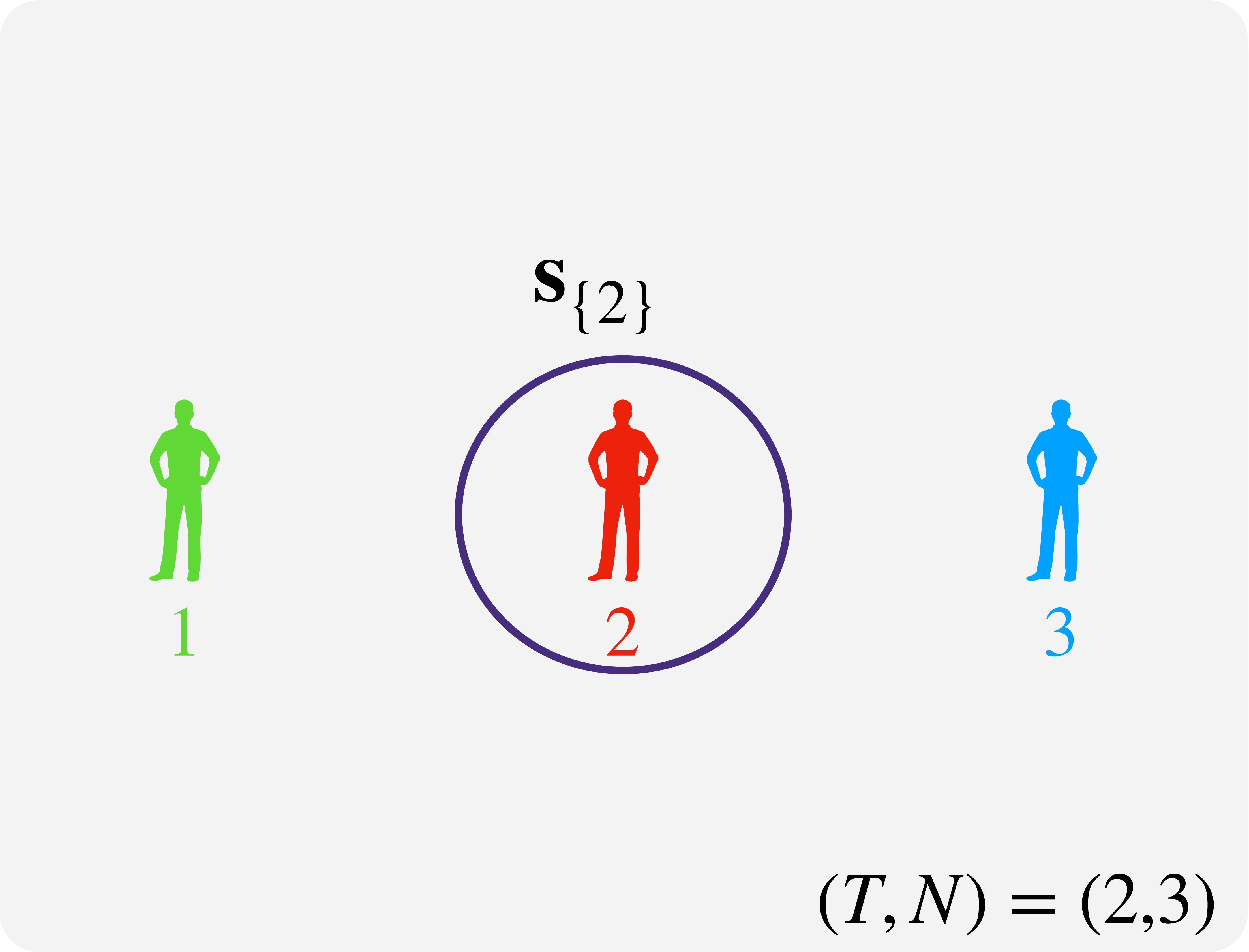


Solution 1: Replicated Secret Sharing

Idea: sample a share for any possible set of corrupted parties.

1. For any set $\mathscr{T}$ of $T-1$ parties, sample a uniform share $\mathbf{s}_{\mathscr{T}}$.

$\mathbf{s}_{\{1\}}$

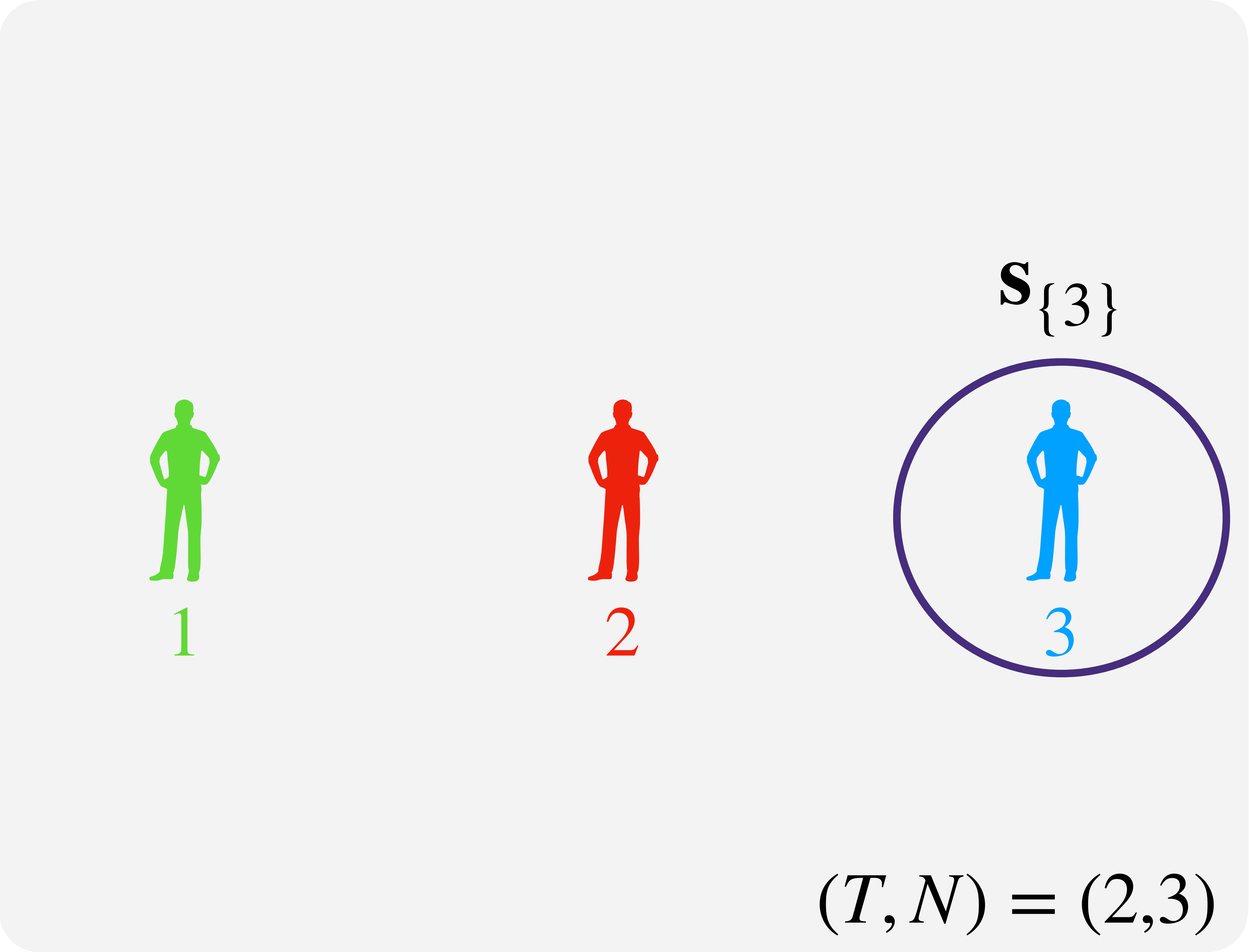


Solution 1: Replicated Secret Sharing

Idea: sample a share for any possible set of corrupted parties.

1. For any set $\mathscr{T}$ of $T-1$ parties, sample a uniform share $\mathbf{s}_{\mathscr{T}}$.

$\mathbf{s}_{\{1\}}$ $\mathbf{s}_{\{2\}}$

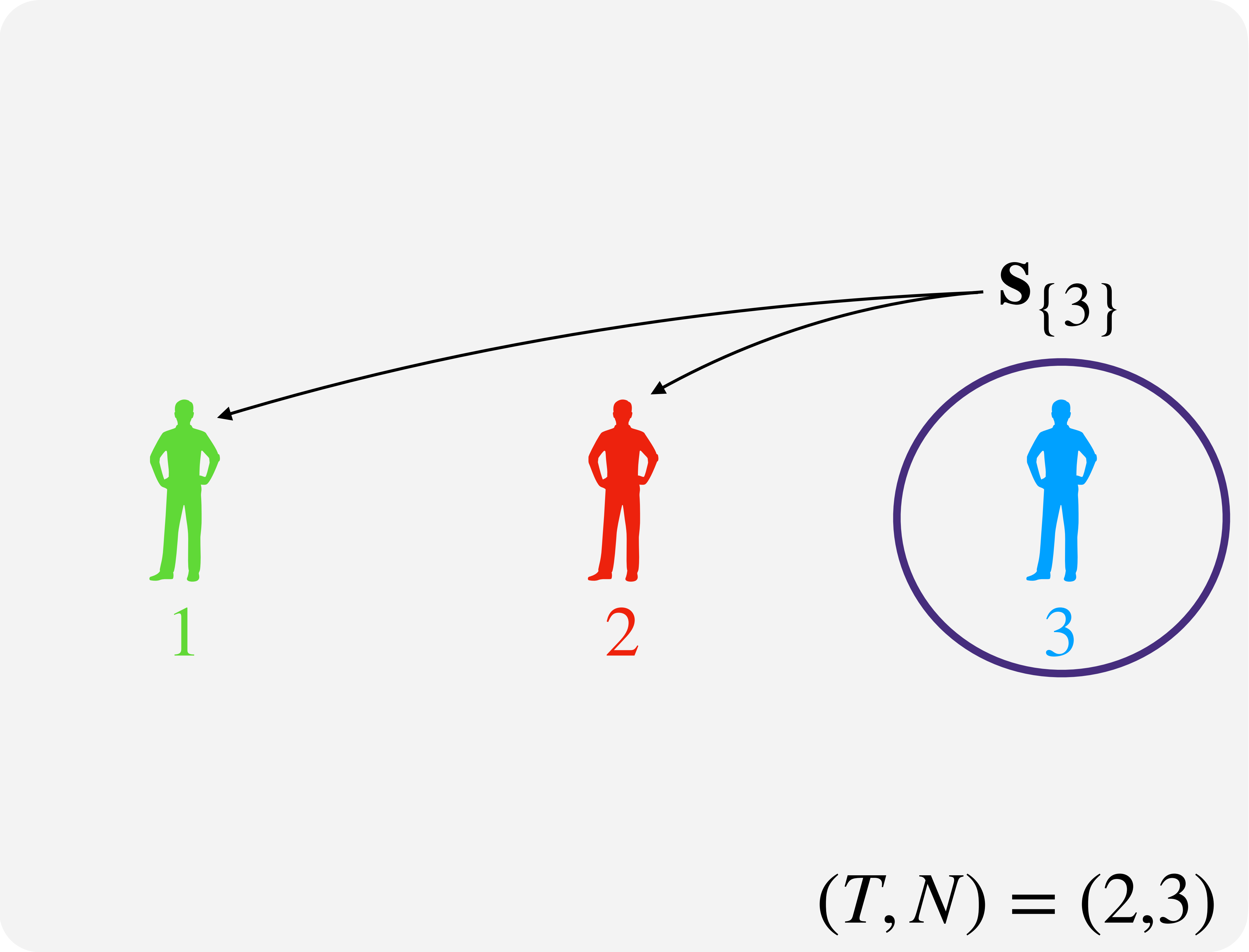


Solution 1: Replicated Secret Sharing

Idea: sample a share for any possible set of corrupted parties.

1. For any set $\mathcal{T}$ of $T-1$ parties, sample a uniform share $\mathbf{s}_{\mathcal{T}}$.

2. Distribute $\mathbf{s}_{\mathcal{T}}$ to the parties in $[N]\backslash\mathcal{T}$.



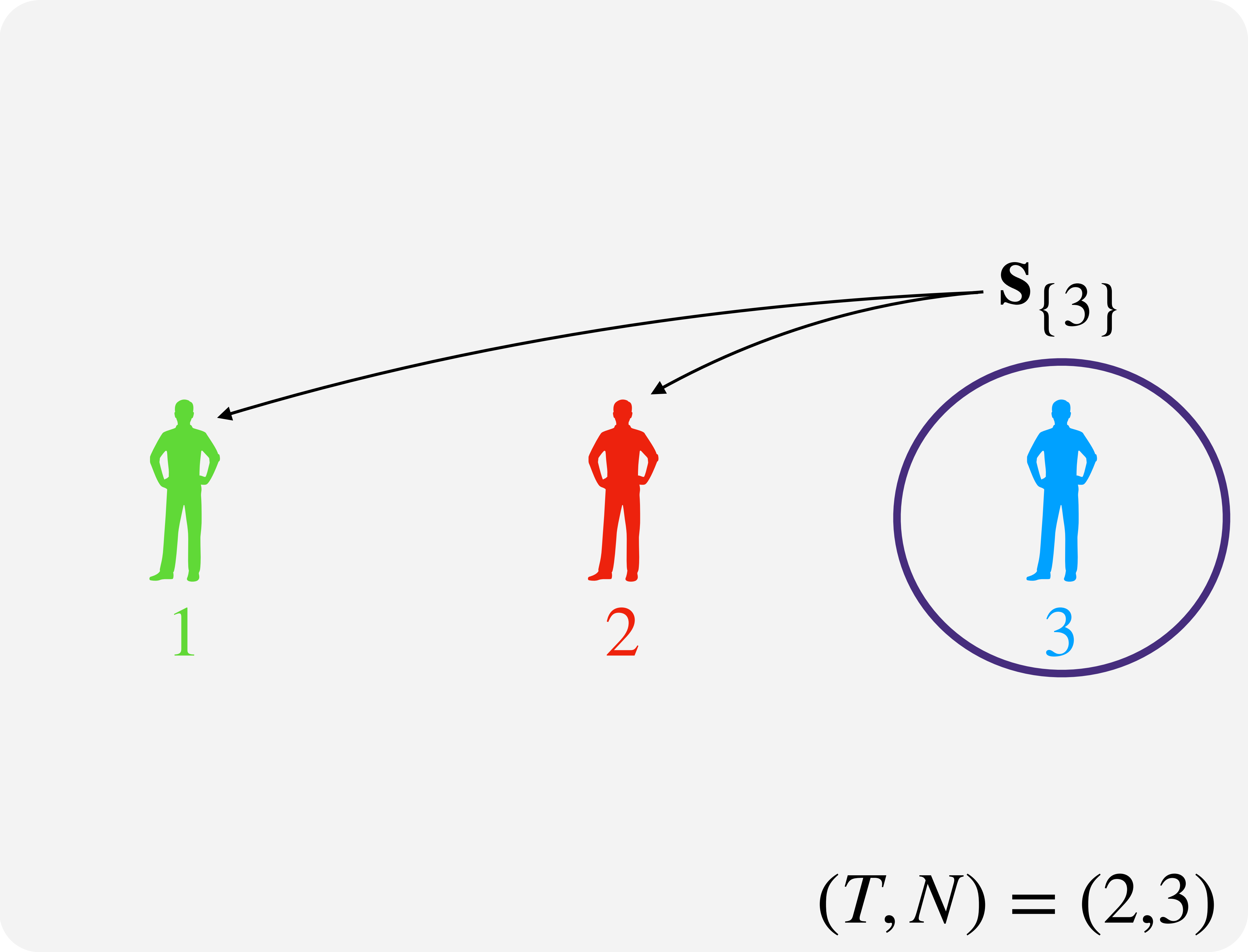
Solution 1: Replicated Secret Sharing

Idea: sample a share for any possible set of corrupted parties.

1. For any set $\mathcal{T}$ of $T-1$ parties, sample a uniform share $\mathbf{s}_{\mathcal{T}}$.

2. Distribute $\mathbf{s}_{\mathcal{T}}$ to the parties in $[N]\backslash\mathcal{T}$.

3. Define sk $= \sum_{\mathcal{T}} \mathbf{s}_{\mathcal{T}}$.



Solution 1: Replicated Secret Sharing

Idea: sample a share for any possible set of corrupted parties.

1. For any set $\mathcal{T}$ of $T-1$ parties, sample a uniform share $\mathbf{s}_{\mathcal{T}}$.
2. Distribute $\mathbf{s}_{\mathcal{T}}$ to the parties in $[N] \backslash \mathcal{T}$.
3. Define $\mathsf{sk} = \sum_{\mathcal{T}} \mathbf{s}_{\mathcal{T}}$.

Properties:

- Reconstruction coefficients 0 or 1
- When $< T$ corrupted parties, at least one $\mathbf{s}_{\mathcal{T}}$ remains hidden.

$\rightarrow$ guarantees that sk remains protected

Solution 1: Short Replicated Secret Sharing

Idea: sample a share for any possible set of corrupted parties.

1. For any set $\mathcal{T}$ of $T-1$ parties, sample a short share $\mathbf{s}_{\mathcal{T}}$.
2. Distribute $\mathbf{s}_{\mathcal{T}}$ to the parties in $[N]\backslash\mathcal{T}$.
3. Define sk $= \sum_{\mathcal{T}} \mathbf{s}_{\mathcal{T}}$.

Properties:

- Reconstruction coefficients 0 or 1
- When $< T$ corrupted parties, at least one $\mathbf{s}_{\mathcal{T}}$ remains hidden.

  $\rightarrow$ guarantees that $[\mathbf{A} \quad \mathbf{I}] \cdot$ sk looks uniform (MLWE assumption)

Solution 1: Short Replicated Secret Sharing

Idea: sample a share for any possible set of corrupted parties.

1. For any set $\mathscr{T}$ sample a short

Caveat: This scheme has a number of shares that is equal to $\binom{N}{T-1}$.

efficients 0 or 1

2. Distribute $\mathbf{s}_{\mathscr{T}}$ t $[N]\backslash\mathscr{T}$.

ted parties, at least one $\mathbf{s}_{\mathscr{T}}$ remains hidden.

3. Define sk $= \sum_{\mathscr{T}} \mathbf{s}_{\mathscr{T}}$.

$\rightarrow$ guarantees that $[\mathbf{A} \quad \mathbf{I}] \cdot$ sk looks uniform (MLWE assumption)

Solution 2: Coupon collector problem

Full collection

N cards

Solution 2: Coupon collector problem

Full collection

N cards

Draw with replacement

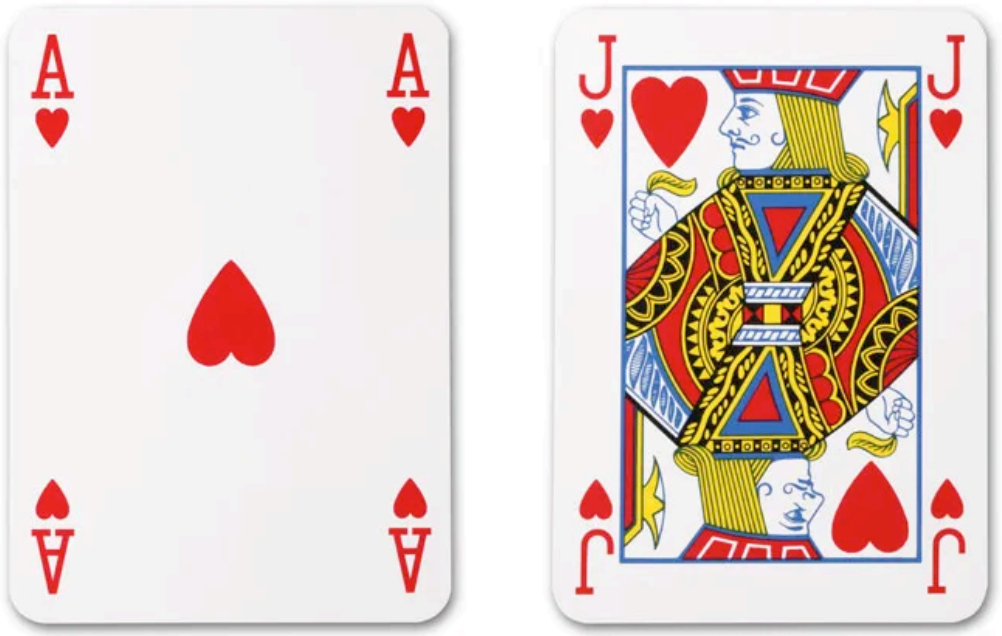
1

Solution 2: Coupon collector problem

Full collection

N cards

Draw with replacement



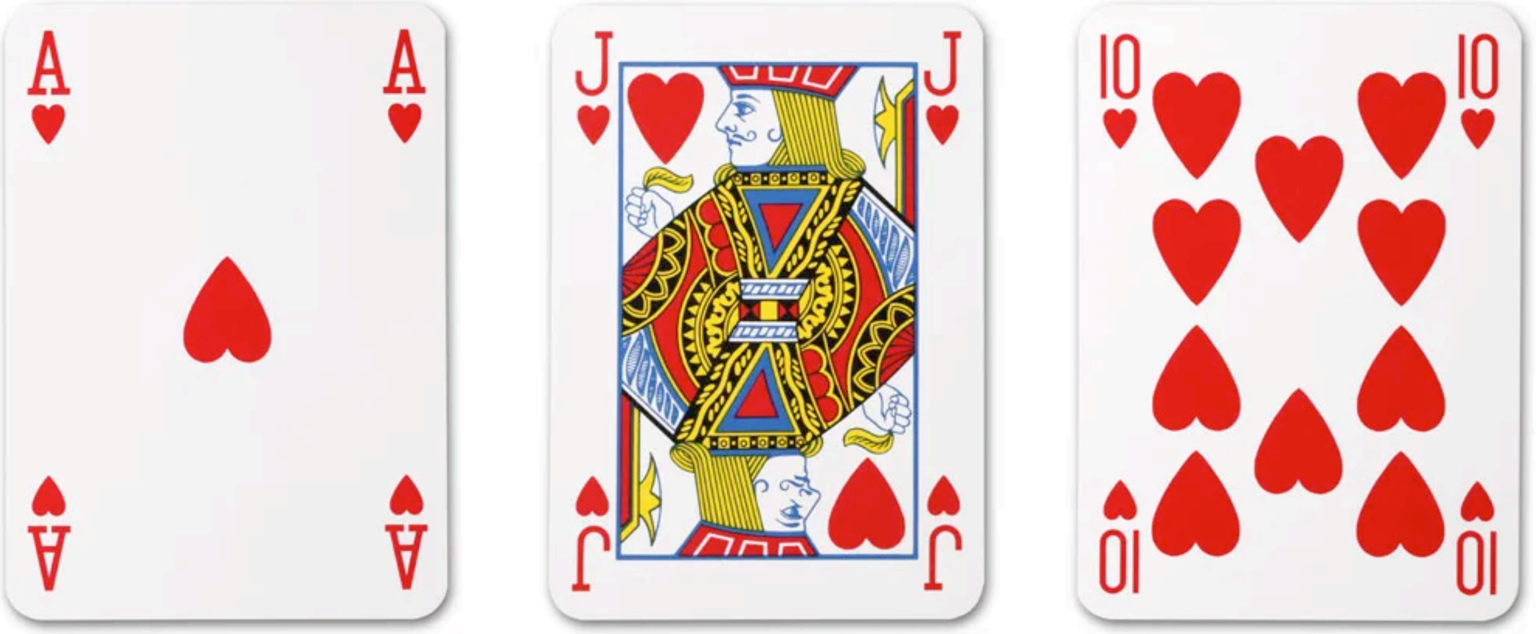
1 2

Solution 2: Coupon collector problem

Full collection

N cards

Draw with replacement



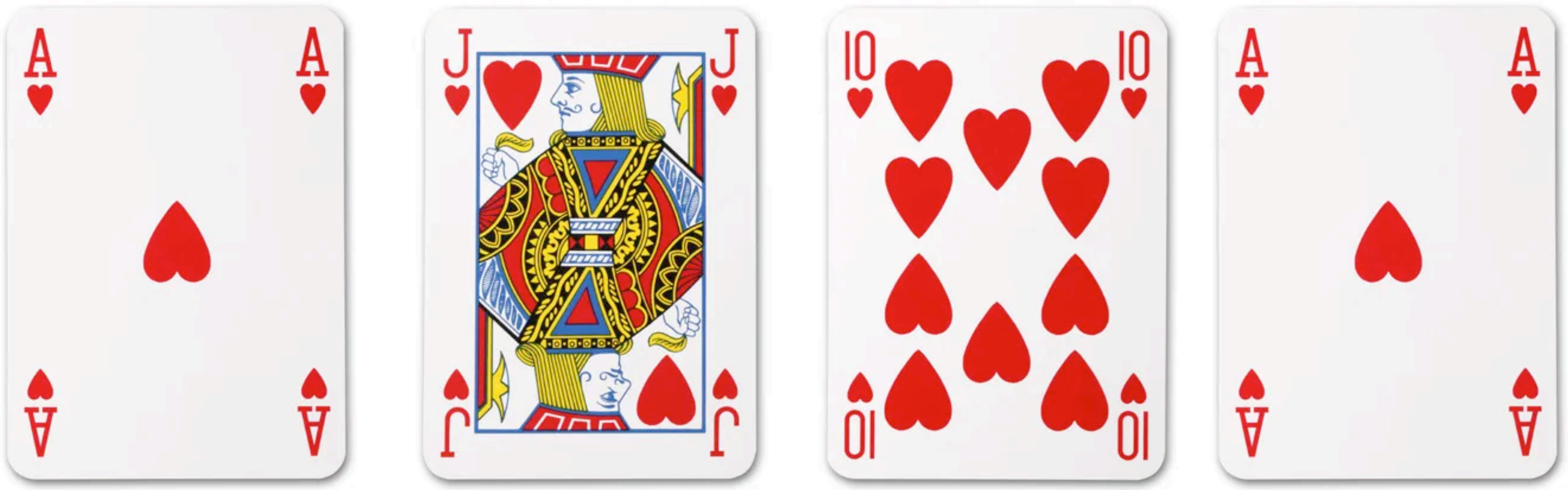
1 2 3

Solution 2: Coupon collector problem

Full collection

N cards

Draw with replacement



1 2 3 4

…

How many draws to get the full collection?

$\sim N \log N$

Solution 2: Coupon collector problem

Full collection $\mathsf{sk} = \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4$

N shares

Example:

- $\mathbf{s}_1, \ldots, \mathbf{s}_{N-1} \leftarrow \mathscr{D}_\sigma^{N-1}$ and $\mathbf{s}_N = \mathsf{sk} - \sum_{j<N} \mathbf{s}_i$

Solution 2: Coupon collector problem

Full collection $\mathsf{sk} = \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4$

N shares

Idea: Randomly distribute one share per party.

Example:

- $\mathbf{s}_1, \ldots, \mathbf{s}_{N-1} \leftarrow \mathscr{D}_\sigma^{N-1}$ and $\mathbf{s}_N = \mathsf{sk} - \sum_{j<N} \mathbf{s}_i$

Desired properties:

- **Reconstruction threshold:** Minimum number of parties T needed to gather all the shares? (with overwhelming probability)
- **Security threshold:** Maximum number of parties T' such that at least one share is not known (with overwhelming probability)

Solution 2: Coupon collector problem

Full collection $\mathsf{sk} = \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4$

N shares

Idea: Randomly distribute one share per party.

Example:

- $\mathbf{s}_1, \ldots, \mathbf{s}_{N-1} \leftarrow \mathscr{D}_\sigma^{N-1}$ and $\mathbf{s}_N = \mathsf{sk} - \sum_{j<N} \mathbf{s}_i$

Desired properties:

- **Reconstruction threshold:** Minimum number of parties T needed to gather all the shares? (with overwhelming probability)
- **Security threshold:** Maximum number of parties T' such that at least one share is not known (with overwhelming probability)

Bounds T, T' are exactly bounds of the coupon collector problem.

Both $T, T' \sim N \log N$, with gap $\underset{N\to\infty}{\approx} 1 + 128/\log N$

Solution 2: Coupon collector problem

Full collection $\mathsf{sk} = \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4$

N shares

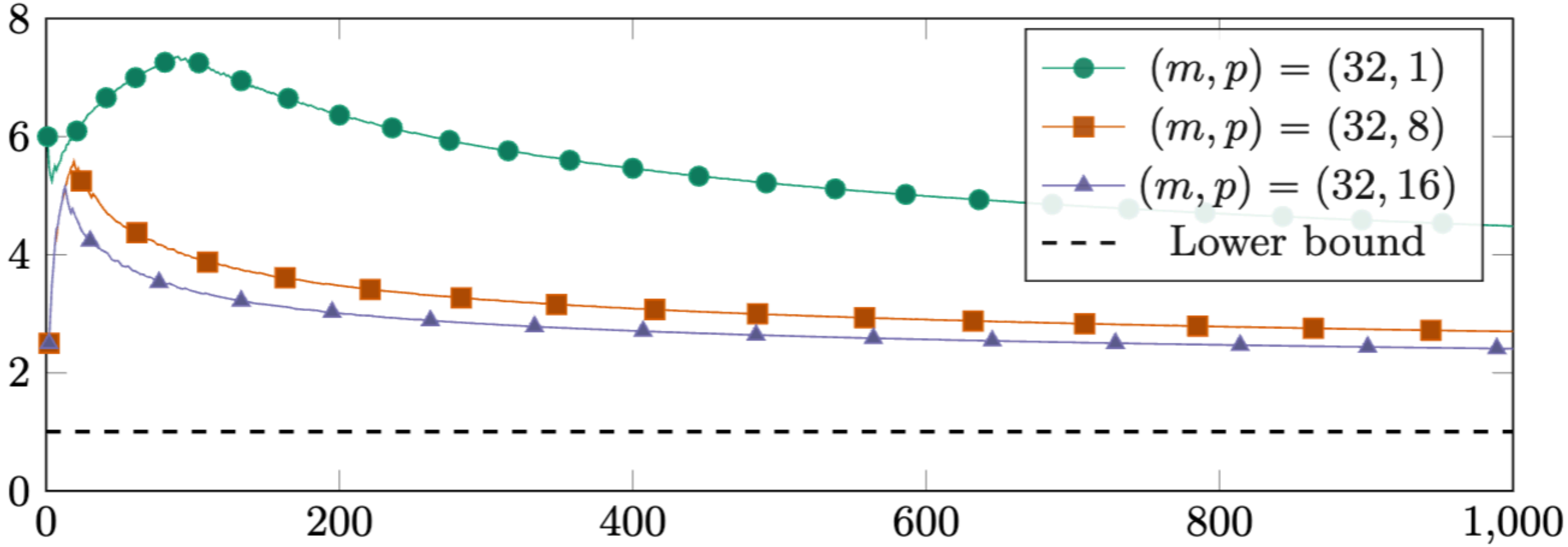
Better parameters by amplifying properties:

- **Reconstruction threshold:** Share same sk m times, just need at least one sharing fully known to recover sk.
- **Security threshold:** Share multiple secrets sk

$$\mathsf{sk} = \mathsf{sk}_1 + \mathsf{sk}_2 + \ldots + \mathsf{sk}_p$$

An adversary must know all the secrets to forge.

Solution 2: Coupon collector problem



Ratio T/T' achieved by our sharing as a function of T'. The dotted line corresponds to an ideal asymptotic $T/T' = 1$.

Recall: m, p correspond respectively to amplification for reconstruction and security thresholds.

Short secret sharing

This presentation assumes a trusted dealer to sample the short secret sharing.

But, in our paper, we show that it is quite easy to design DKGs.

Conclusion

Conclusion

- **Introduced two short secret sharing methods**
  - Based on replicated secret sharing (exponential number of shares $\rightarrow$ for small number of parties)
  - Based on coupon collector problem: scales to larger thresholds, but has a gap between T and T'
- **Two applications**
  - Threshold Raccoon with identifiable aborts (using partial verification keys)
  - A compact threshold FSwA signature scheme for $N \leq 8$

Questions?