Flood and Submerse:

Distributed Key Generation and Robust Threshold Signature from Lattices

Thomas Espitau **<u>Guilhem Niot</u>** **Thomas Prest**

ENSL/CWI/KCL/IRISA Joint Seminar - 30 Sept. 2024

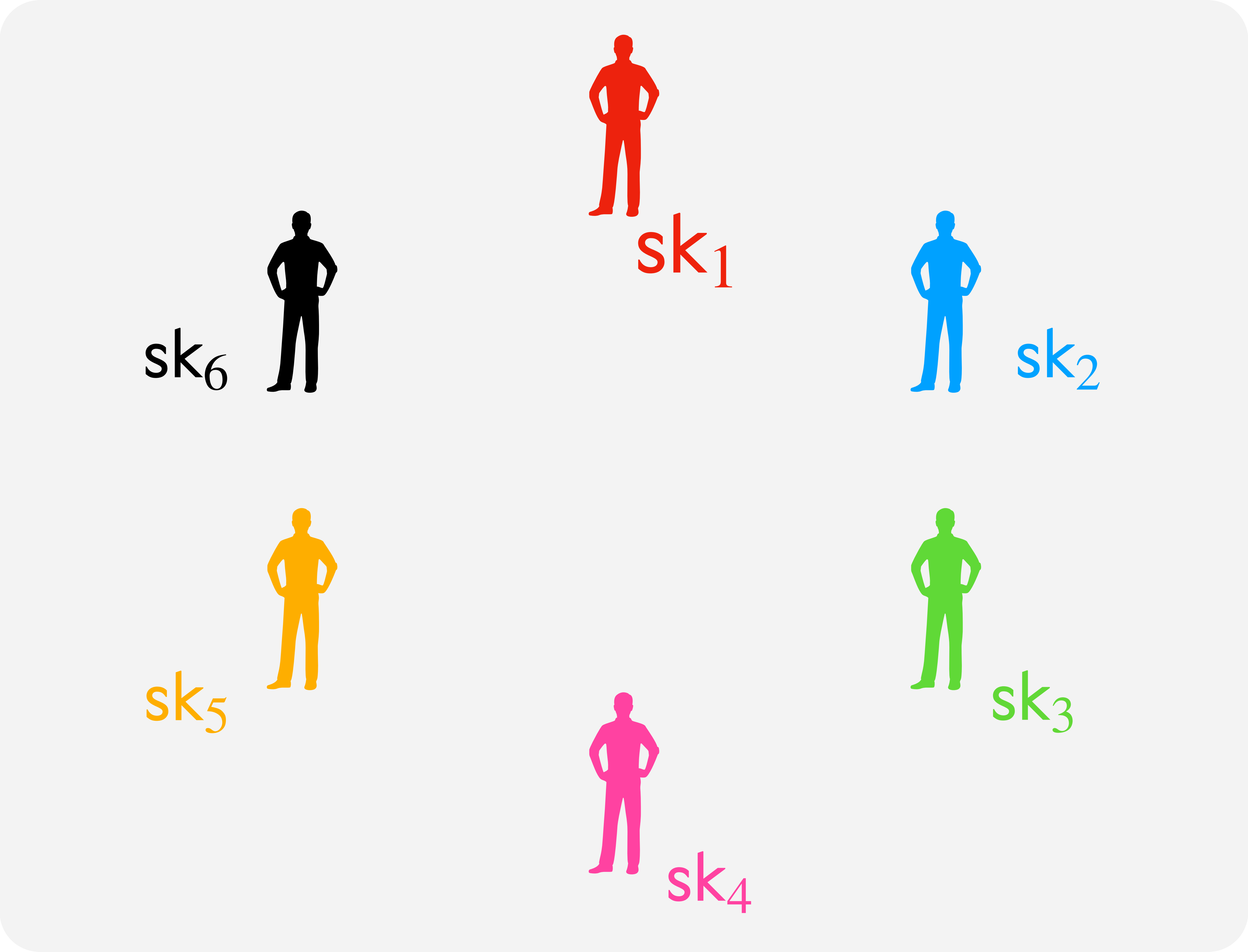
PQ SHIELD

1. Background

(T-out-of-N) threshold signatures

What are they?

An interactive protocol to distribute signature generation.

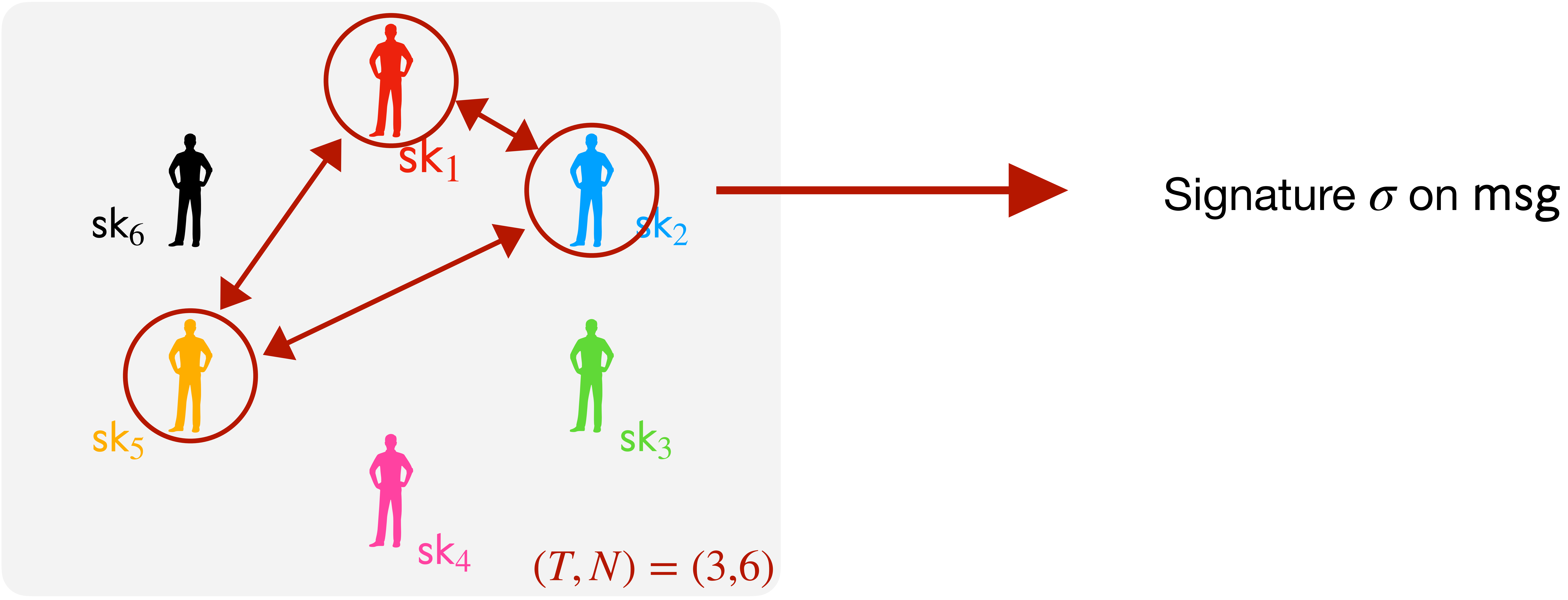


- 1 verification key vk
- 1 partial signing key sk_i per party
- Given at least T-out-of-N partial signing keys, we can sign.

(T-out-of-N) threshold signatures

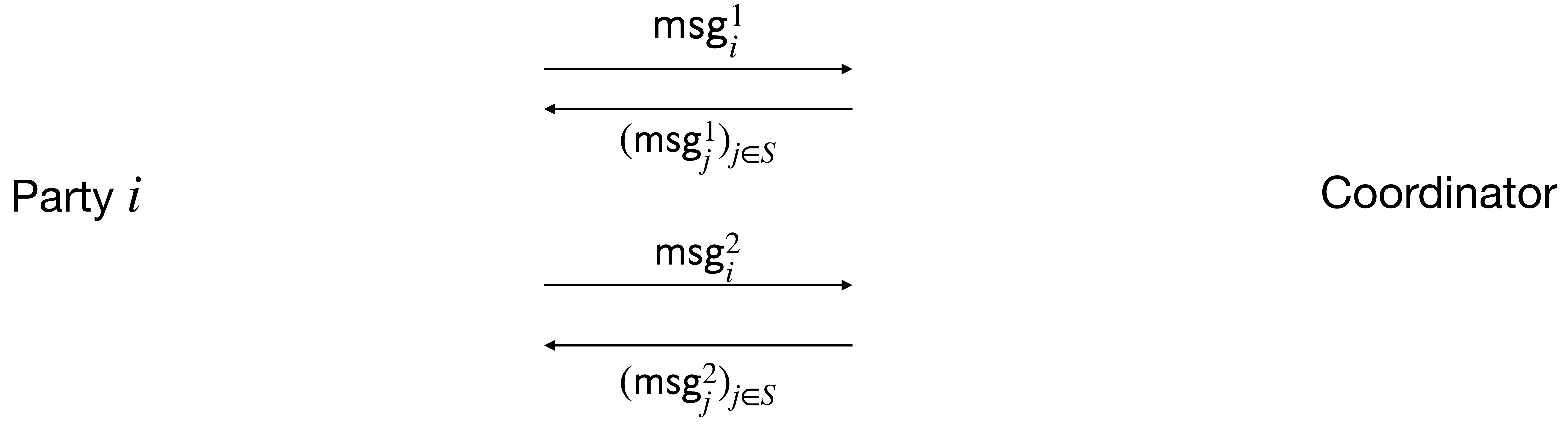
What are they?

An interactive protocol to distribute signature generation.



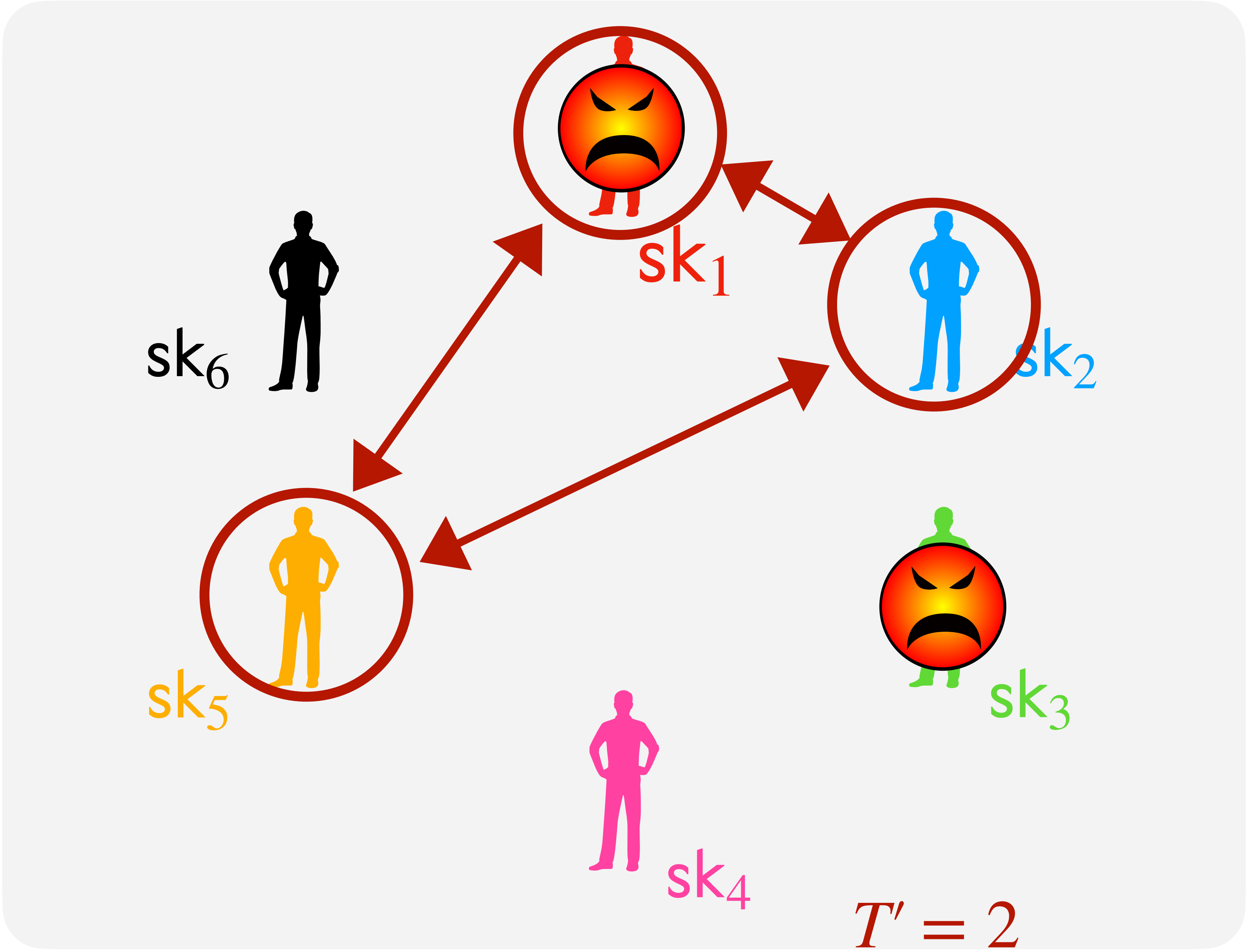
(T-out-of-N) threshold signatures
What are they?

Round-based communication model:



Core security properties

- **Correctness:** Given at least T-out-of-N partial signing keys, we can sign.
- **Unforgeability:** The signature scheme remains unforgeable even if up to $T' < T$ parties are corrupted. Often $T' = T - 1$.



It's not possible to forge a new signature, even by taking part in the signing protocol.

More desirable properties

- **Adaptive security:** (vs static security) Corrupted users can be chosen adaptively over the lifetime of the signature scheme. More realistic than static security, i.e. corrupted users chosen before setup.
- **Distributed Key Generation:** Protocol allowing to distributively sample key material.
- **Robustness (resp. identifiable abort):** In the presence of malicious users, signature protocol is guaranteed to produce a valid signature (resp. to identify misbehaving users)
- **Small round complexity:** Ideally can be as low as one round.
- **Backward compatibility:** Threshold schemes should ideally be compatible with existing primitives.

Pre-quantum solutions

- Mature solutions:
  - EdDSA: FROST [KG20]
  - ECDSA: [ANOS+21]
  - BLS: [Bol03]
  - RSA: [Sho00]

- Provide all desirable properties.

An active field of research for post-quantum security

- Aggregating hash-based signatures: [KCLM22]
- Sequential TS scheme based on isogenies: [DM20]
- Lattice-based threshold signatures:
  - 2-round TS via FHE: [BGG+18], [ASY22], [GKS23]
  - TS with noise flooding (based on Raccoon): 3-round [dPKM+23], 2-round [EKT24], [BKLM+24], 5-round adaptively secure [KRT24]

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]

Adaptively Secure 5 Round Threshold Signatures from MLWE/MSIS and DL with Rewinding

Shuichi Katsumata[1,2], Michael Reichle[3], Kaoru Takemure*[1,2]

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; MPI-SP, Germany*
Akira Takahashi, *JPMorgan AI Research & AlgoCRYPT CoE, USA*
Mehdi Tibouchi, *NTT, Japan*

An active field of research for post-quantum security

- Aggregating hash-based signatures: [KCLM22]

- Sequential TS scheme based on isogenies: [DM20]

- Lattice-based threshold signatures:
  - 2-round TS via FHE: [BGG+18], [ASY22], [GKS23]
  - TS with noise flooding (based on Raccoon): 3-round [dPKM+23], 2-round [EKT24], [BKLM+24], 5-round adaptively secure [KRT24]

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]

Adaptively Secure 5 Round Threshold Signatures from MLWE/MSIS and DL with Rewinding

Shuichi Katsumata[1,2], Michael Reichle[3], Kaoru Takemure*[1,2]

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; MPI-SP, Germany*
Akira Takahashi, *JPMorgan AI Research & AlgoCRYPT CoE, USA*
Mehdi Tibouchi, *NTT, Japan*

Threshold Raccoon, a practical 3-round threshold signature

κ	Number Signers	| vk |	| sig |	Total communication
128	≤ 1024	4 kB	13 kB	40 kB

… but only considers core security properties: correctness and unforgeability.

Advanced properties of lattice-based schemes

Active research since 2024.

- **Adaptive security:** 5-round [KRT24]
- **Small round complexity:** 2-round [EKT24], [BKLM+24]
- **Backward compatibility:** These schemes can be made compatible with the NIST proposal Raccoon.

No efficient solution for:

- **Distributed Key Generation (DKG)**
- **Robustness / identifiable abort**

Focus of this presentation

- **Distributed Key Generation** (DKG)
- **Robustness**: Guarantee valid signature in the presence of malicious signers

Our techniques for DKG + robust signing are quite generic:

- in our paper, applied to Plover [EENP+24]: hash-and-sign scheme
- can be applied to all **3-round [dPKM+23]**, 2-round [EKT24], [BKLM+24]

Raccoon signature scheme

Lyubashevsky's signature scheme (without aborts)

$$\mathsf{vk} = \mathbf{t} = \underbrace{\begin{bmatrix}\mathbf{A}' & \mathbf{I}\end{bmatrix}}_{\mathbf{A}} \cdot \mathbf{s} \in \mathscr{R}_q^k \qquad \mathsf{sk} = \mathbf{s} \in \mathscr{R}_q^\ell \text{ short}$$

$\mathbf{r} \leftarrow \chi$

$\mathbf{w} = \mathbf{A} \cdot \mathbf{r}$ $\xrightarrow{\mathbf{w}}$

$\longleftarrow$ $c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w}) \in \mathscr{R}_q$ "small"

$\mathbf{z} = c \cdot \mathbf{s} + \mathbf{r}$ $\longrightarrow$ Accept if

- $\mathbf{z}$ is short
- $\mathbf{A} \cdot \mathbf{z} = c \cdot \mathbf{t} + \mathbf{w}$

Prove security via Hint-MLWE assumption

Hint-MLWE assumption [KLSS23]

Consider $\mathbf{t} = \underbrace{\begin{bmatrix} \mathbf{A}' & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \cdot \mathbf{s} \in \mathcal{R}_q^k$ and reveal hints $(\mathbf{z}_i = c_i \cdot \mathbf{s} + \mathbf{r}_i)_{i \in [Q]}$

t is indistinguishable from uniform (as hard as MLWE) for some parameter regimes.

Rule of thumb: secure if $\sigma_{\mathbf{r}} \approx \sqrt{Q} \cdot s_1(c) \cdot \sigma_{\mathbf{s}}$

Threshold Raccoon [dPKM+23]

Threshold signature: use (T, N)-Shamir sharing on secret

$\mathsf{sk} = \mathbf{s} \in \mathcal{R}_q^\ell$ short

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

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

For any set S of T shares, reconstruct $\mathbf{s}$:

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

Lagrange coefficient

Threshold Raccoon [dPKM+23]

Threshold signature: use (T, N)-Shamir sharing on secret

sk = $\mathbf{s} \in \mathscr{R}_q^{\ell}$ short

For any set S of T shares, reconstruct $\mathbf{s}$:

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

$\mathbf{r} \leftarrow \chi$

$\mathbf{w} = \mathbf{A} \cdot \mathbf{r}$

$c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w})$

$\mathbf{z} = c \cdot \mathbf{s} + \mathbf{r}$

Threshold Raccoon [dPKM+23]

Threshold signature: use (T, N)-Shamir sharing on secret

sk = $\mathbf{s} \in \mathcal{R}_q^{\ell}$ short

For any set S of T shares, reconstruct $\mathbf{s}$:

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

$\mathbf{r} \leftarrow \chi$

$\mathbf{w} = \mathbf{A} \cdot \mathbf{r}$

$c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w})$

$\mathbf{z} = c \cdot \mathbf{s} + \mathbf{r}$

$\mathbf{r_i} \leftarrow \chi$

$\mathbf{w}_i = \mathbf{A} \cdot \mathbf{r_i}$

$\mathsf{cmt}_i = H(\mathbf{w_i})$ →

← $(\mathsf{cmt}_j)_{j \in S}$

$\mathbf{w}_i$ →

← $(\mathbf{w}_j)_{j \in S}$

Threshold Raccoon [dPKM+23]

Threshold signature: use (T, N)-Shamir sharing on secret

sk = $\mathbf{s} \in \mathcal{R}_q^{\ell}$ short

For any set S of T shares, reconstruct $\mathbf{s}$:

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

$\mathbf{r} \leftarrow \chi$

$\mathbf{w} = \mathbf{A} \cdot \mathbf{r}$

$c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w})$

$\mathbf{z} = c \cdot \mathbf{s} + \mathbf{r}$

$\mathbf{r_i} \leftarrow \chi$

$\mathbf{w}_i = \mathbf{A} \cdot \mathbf{r_i}$

$\mathsf{cmt}_i = H(\mathbf{w_i})$ →

← $(\mathsf{cmt}_j)_{j \in S}$

$\mathbf{w}_i$ →

← $(\mathbf{w}_j)_{j \in S}$

$\mathbf{w} = \sum_{j \in S} \mathbf{w_j}$

$c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w})$

$[\![\mathbf{z}]\!]_i = c \cdot L_{S,i} \cdot [\![\mathbf{s}]\!]_i + \mathbf{r}_i + \Delta_i$ $\quad \mathbf{z}_i$ →

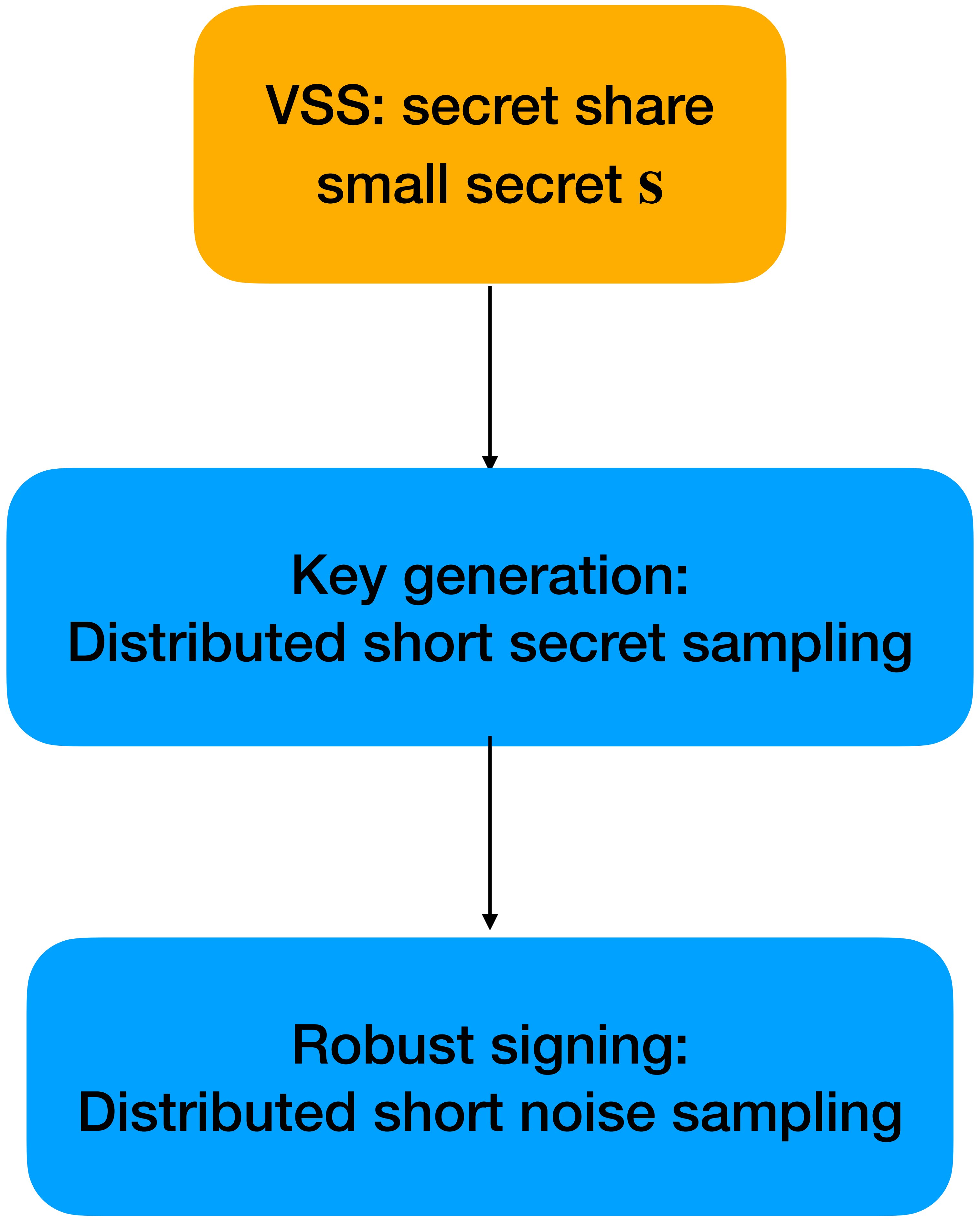
Additive sharing of 0

Accept if

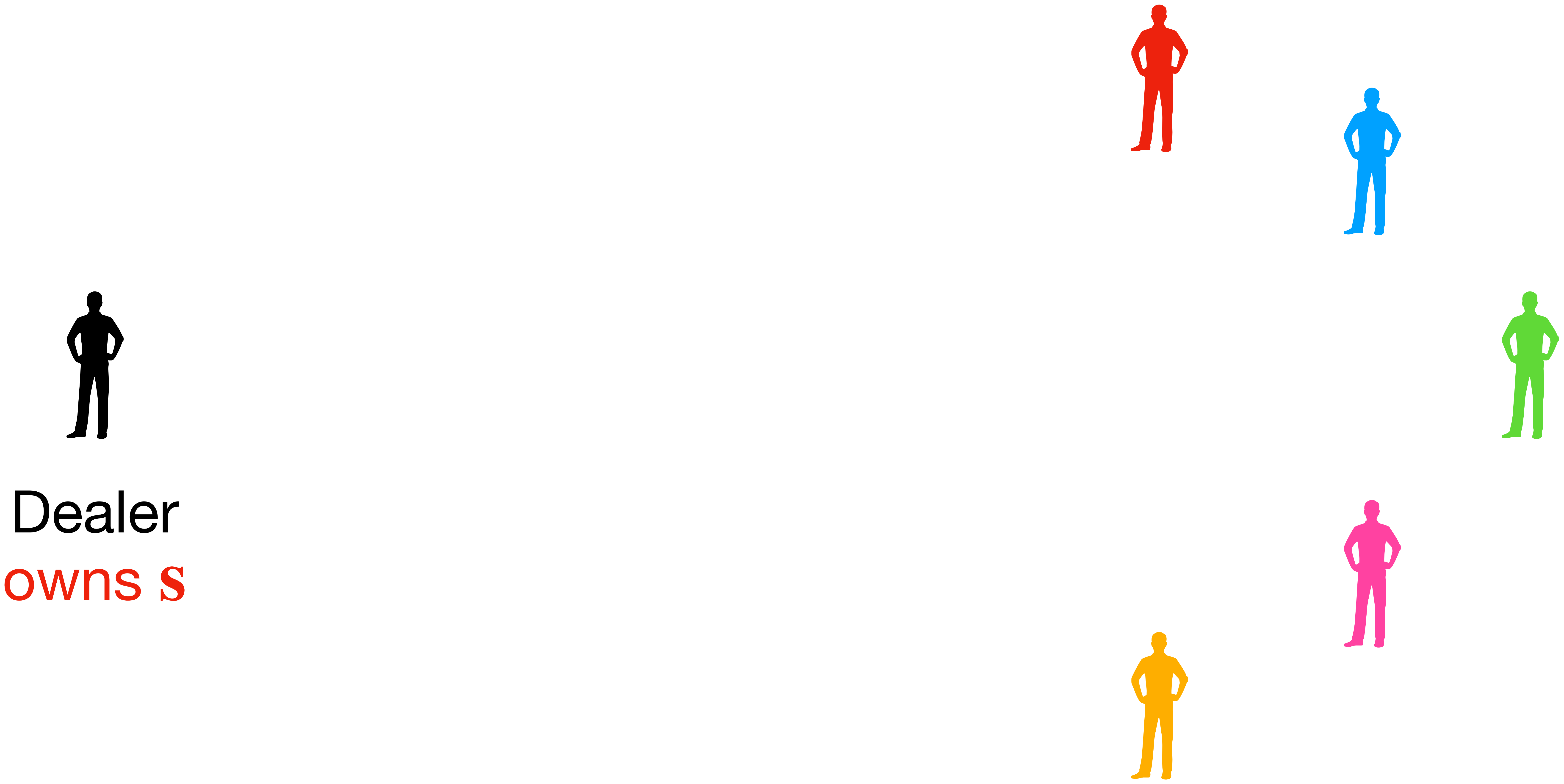
- $\mathbf{z} = \sum_{j \in S} [\![\mathbf{z}]\!]_j = c \cdot \mathbf{s} + \sum_{j \in S} \mathbf{r}_j$ is short
- $\mathbf{A} \cdot \mathbf{z} = c \cdot \mathbf{t} + \mathbf{w}$

2. Achieving additional threshold properties with Verifiable Secret Sharing

Achieving additional threshold properties with Verifiable Secret Sharing

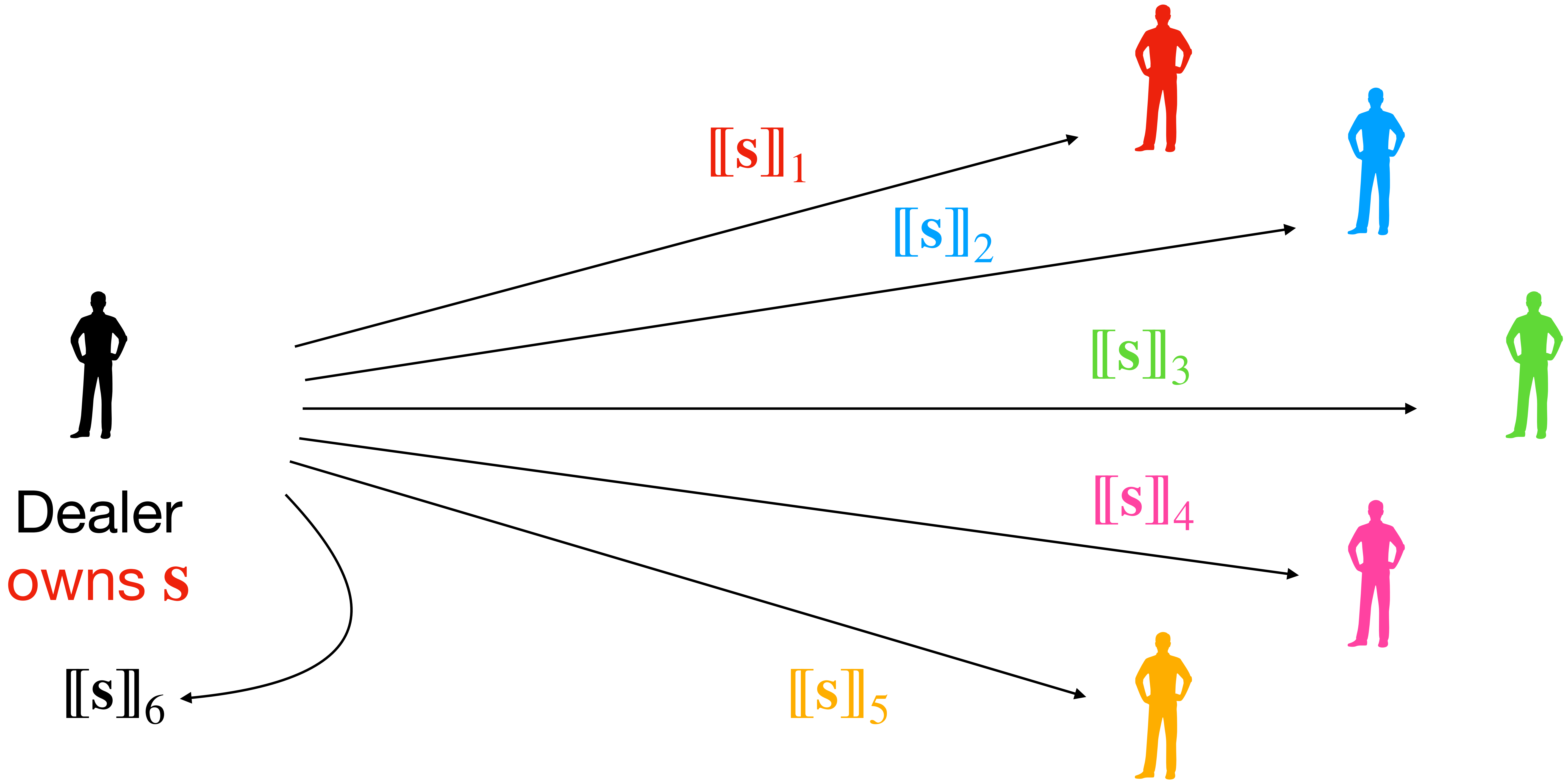


Verifiable Secret Sharing (VSS)



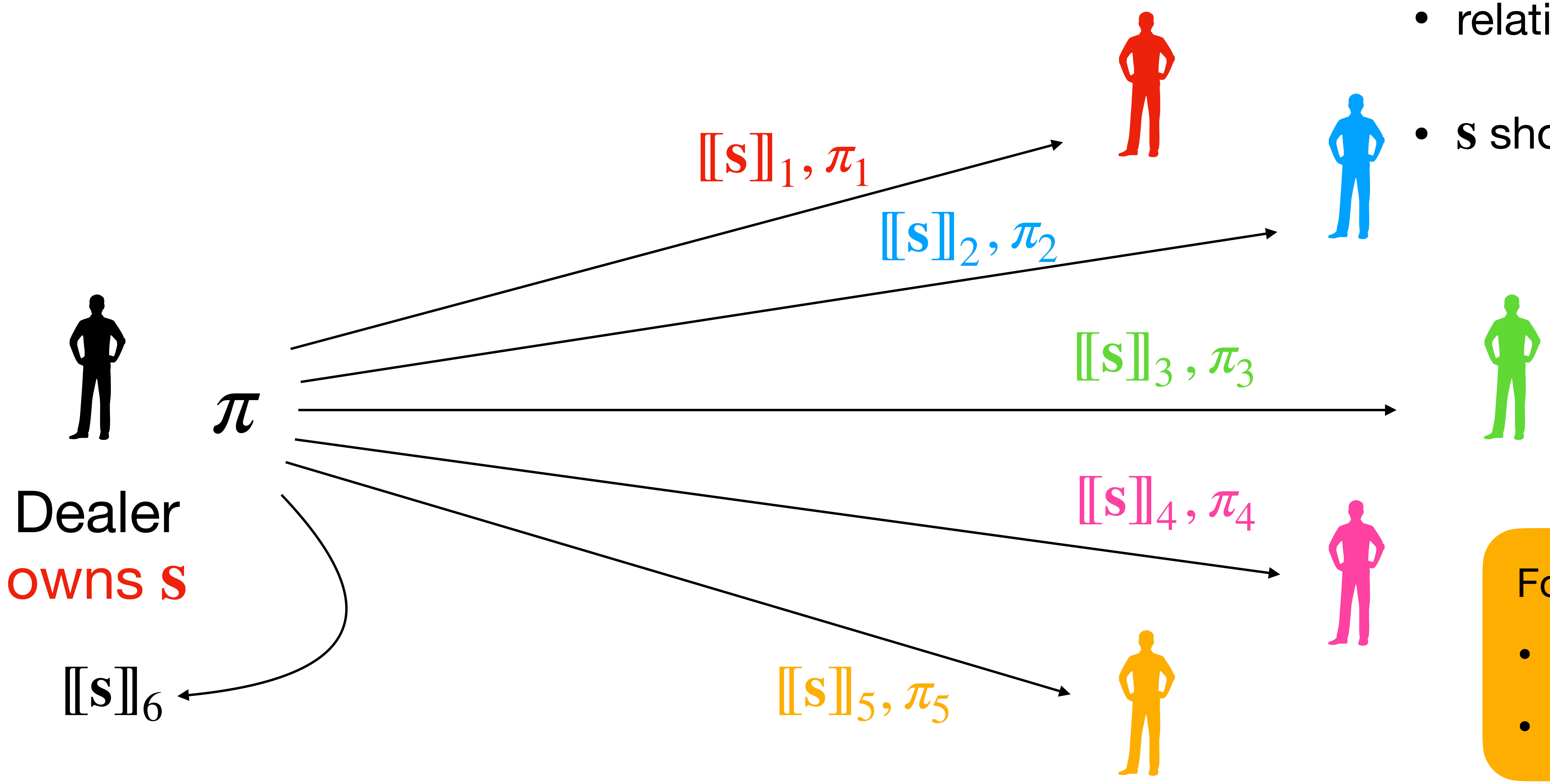
Verifiable Secret Sharing (VSS)

1) Send individual shares



Verifiable Secret Sharing (VSS)

1) Send individual shares



2) Prove correct sharing, i.e.

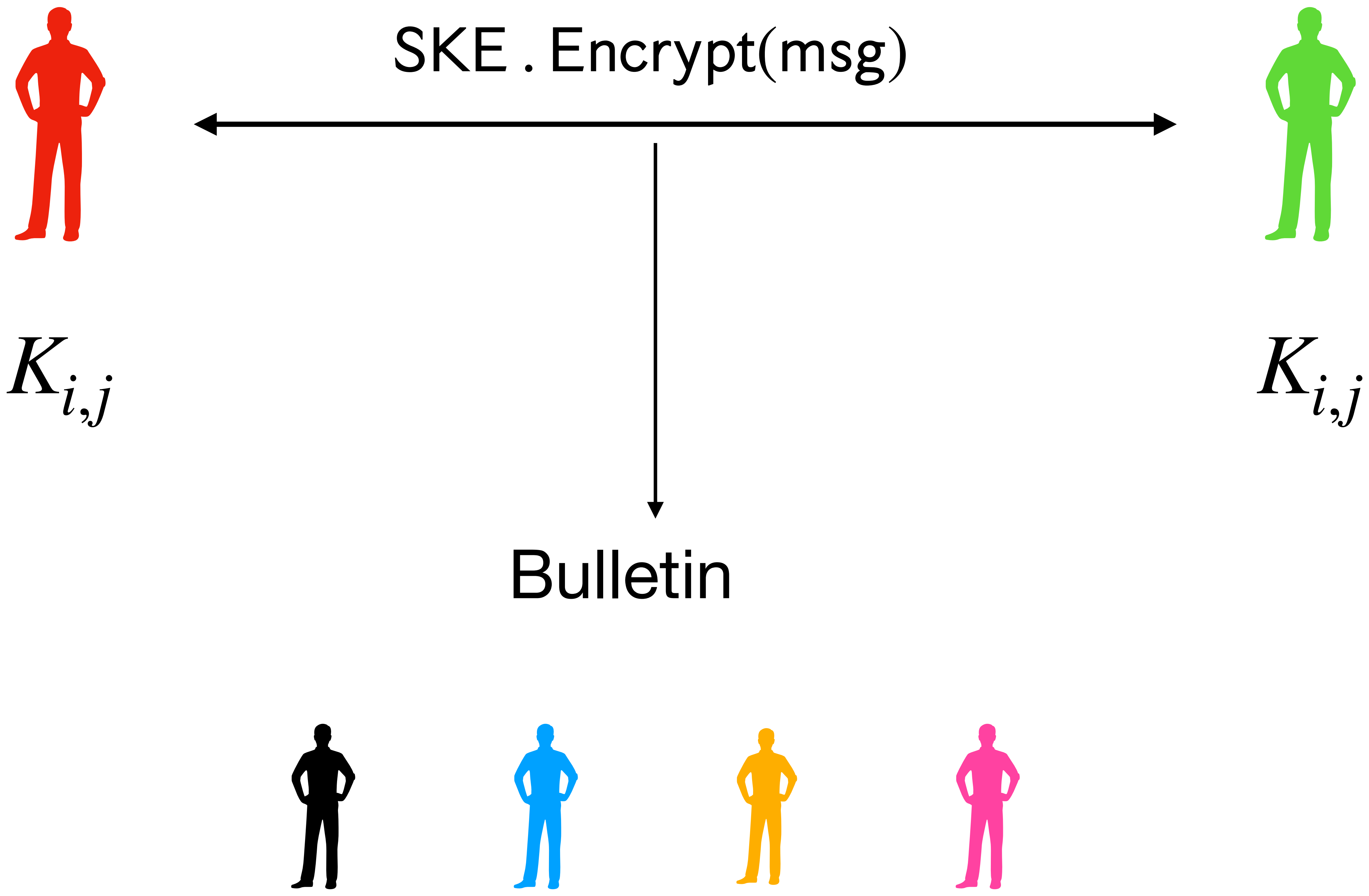
- relation $\mathbf{s} = \sum_{i \in S} L_{S,i} \cdot [\![\mathbf{s}]\!]_i$ for $|S| = T$
- $\mathbf{s}$ short

Formally,

- $\mathsf{VSS.Share}(\mathbf{s}) \rightarrow (\pi, ([\![\mathbf{s}]\!]_i, \pi_i)_{1 \leq i \leq N})$
- $\mathsf{VSS.Verify}(i, [\![\mathbf{s}]\!]_i, \pi, \pi_i) \rightarrow \mathbf{ok} \,|\, \mathbf{fail}$

Distributed Key Generation (DKG) from VSS

- Assume the existence of a broadcast or bulletin board.
- Assume the existence of non-repudiable pairwise channels.



Allows to prove that a message was sent.

Distributed Key Generation (DKG) from VSS

$$\text{vk} = \mathbf{t} = \underbrace{[\mathbf{A}' \mid \mathbf{I}]}_{\mathbf{A}} \cdot \mathbf{s} \in \mathscr{R}_q^k$$

$$\text{sk} = \mathbf{s} \in \mathscr{R}_q^\ell \quad \text{short}$$

Distributed Key Generation (DKG) from VSS

$$\mathsf{vk} = \mathbf{t} = \underbrace{\begin{bmatrix} \mathbf{A}' & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \cdot \mathbf{s} \in \mathcal{R}_q^k$$

$$\mathsf{sk} = \mathbf{s} \in \mathcal{R}_q^\ell \quad \text{short}$$

1. Construct and share secret key $\mathbf{s}$

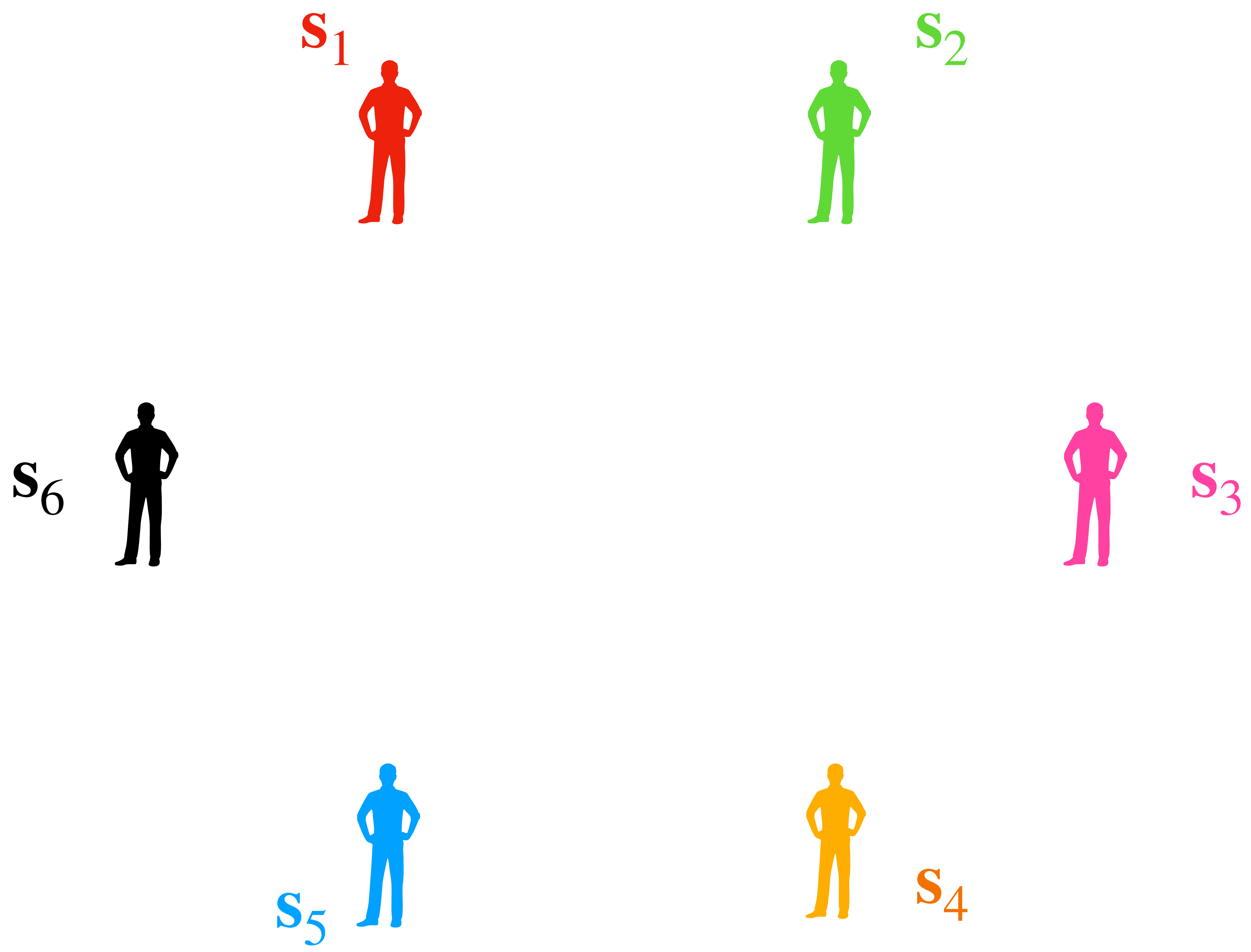
Distributed Key Generation (DKG) from VSS

$$\mathsf{vk} = \mathbf{t} = \underbrace{\begin{bmatrix} \mathbf{A}' & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \cdot \mathbf{s} \in \mathcal{R}_q^k$$

$$\mathsf{sk} = \mathbf{s} \in \mathcal{R}_q^\ell \quad \text{short}$$

1. Construct and share secret key $\mathbf{s} = \sum_i \mathbf{s}_i$

1.a) Sample small secrets $\mathbf{s}_i$



Distributed Key Generation (DKG) from VSS

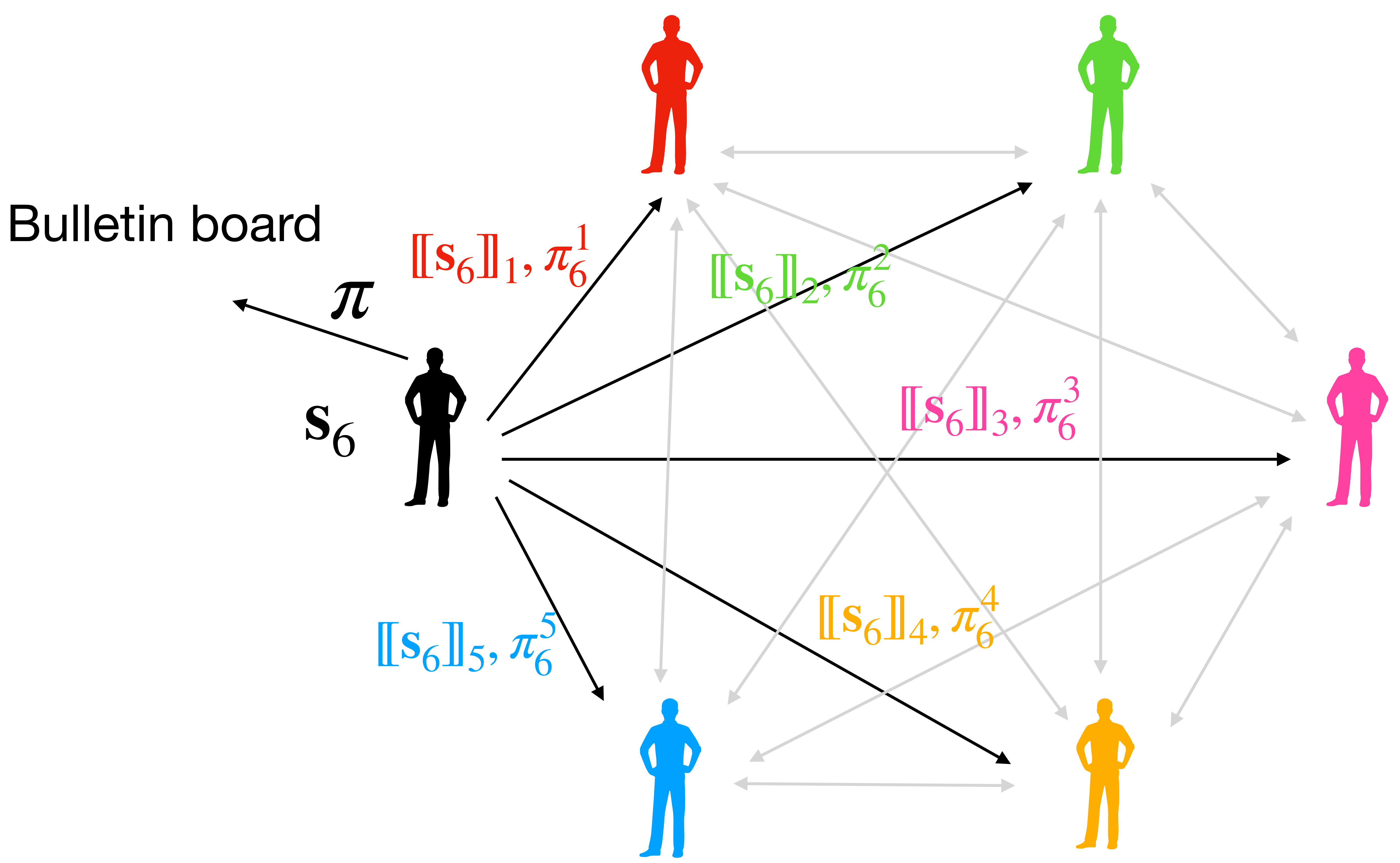
$\text{vk} = \mathbf{t} = \underbrace{\begin{bmatrix} \mathbf{A}' & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \cdot \mathbf{s} \in \mathcal{R}_q^k$

$\text{sk} = \mathbf{s} \in \mathcal{R}_q^\ell$ short

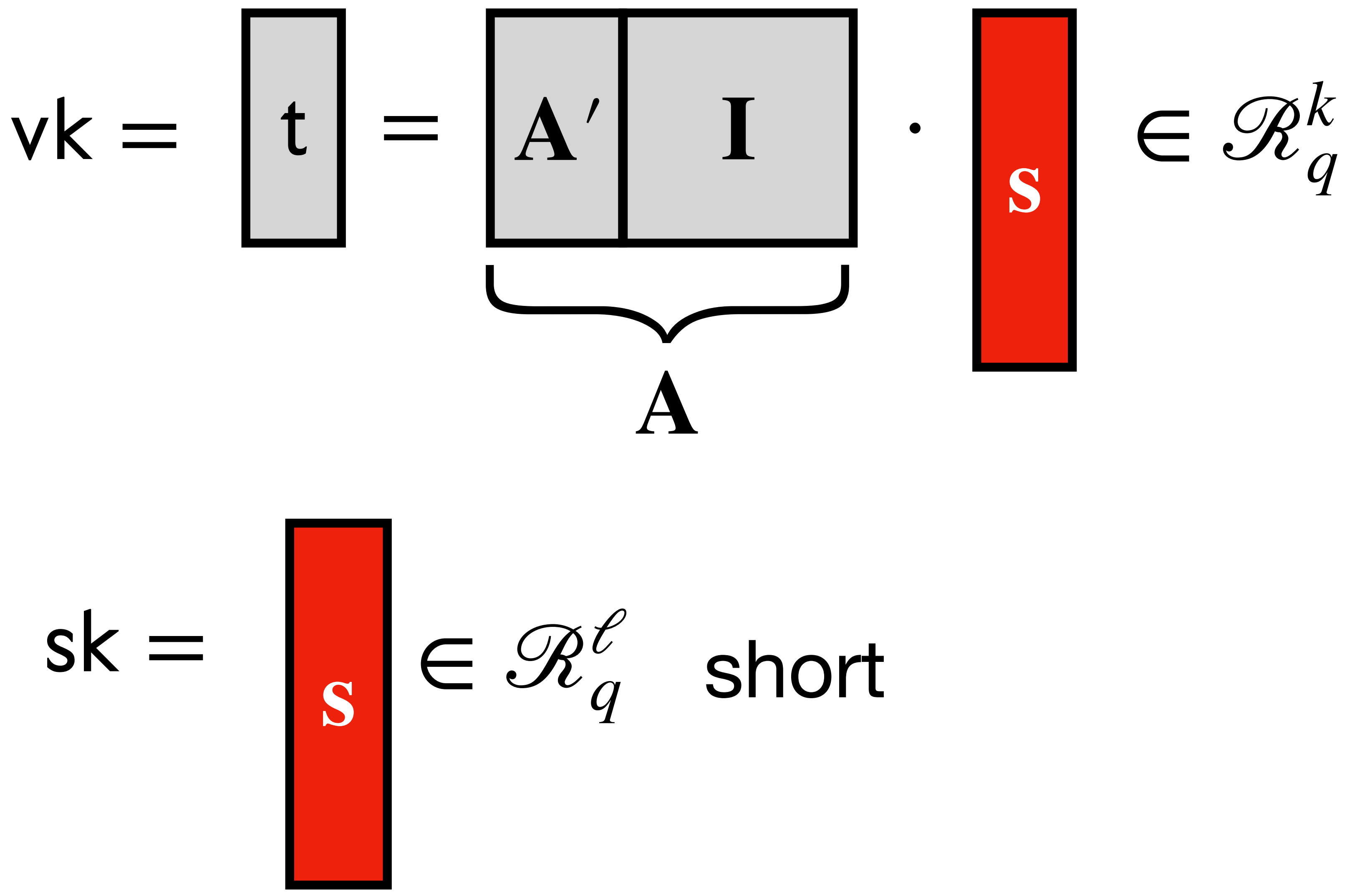
1. Construct and share secret key $\mathbf{s} = \sum_i \mathbf{s}_i$

1.a) Sample small secrets $\mathbf{s}_i$

1.b) Send shares $([\![\mathbf{s}_i]\!]_j, \pi_i^j)_j$



Distributed Key Generation (DKG) from VSS

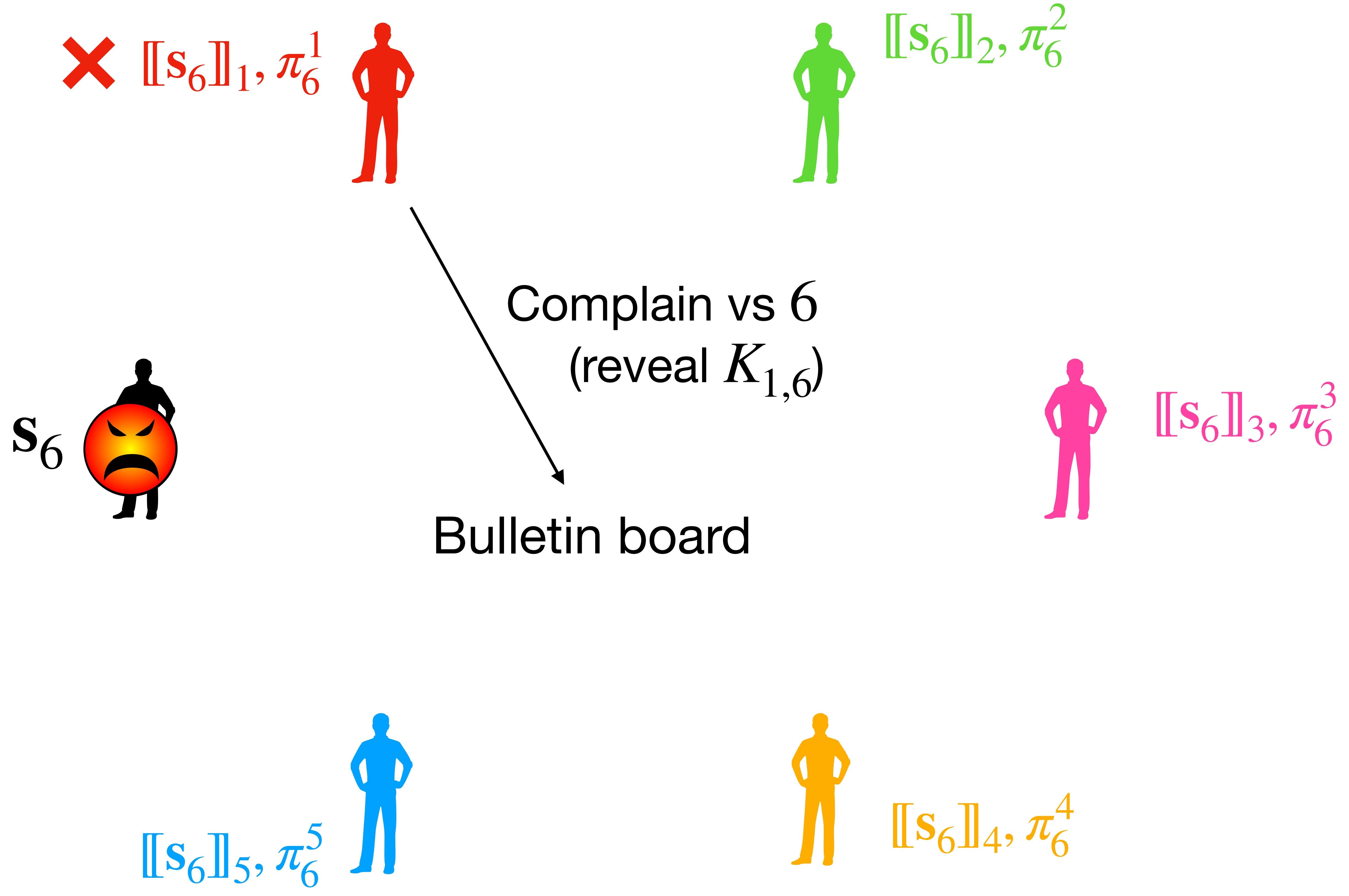


1. Construct and share secret key $\mathbf{s} = \sum_i \mathbf{s}_i$

   1.a) Sample small secrets $\mathbf{s}_i$

   1.b) Send shares $([\![\mathbf{s}_i]\!]_j, \pi_i^j)_j$

   1.c) Verify shares $([\![\mathbf{s}_i]\!]_j, \pi_i^j)_j$ and complain



Distributed Key Generation (DKG) from VSS

$$\mathrm{vk} = \mathbf{t} = \begin{bmatrix} \mathbf{A}' & \mathbf{I} \end{bmatrix} \cdot \mathbf{s} \in \mathcal{R}_q^k$$

where $\mathbf{A} = \begin{bmatrix} \mathbf{A}' & \mathbf{I} \end{bmatrix}$

$$\mathrm{sk} = \mathbf{s} \in \mathcal{R}_q^\ell \quad \text{short}$$

1. Construct and share secret key $\mathbf{s} = \sum_i \mathbf{s}_i$

   1.a) Sample small secrets $\mathbf{s}_i$

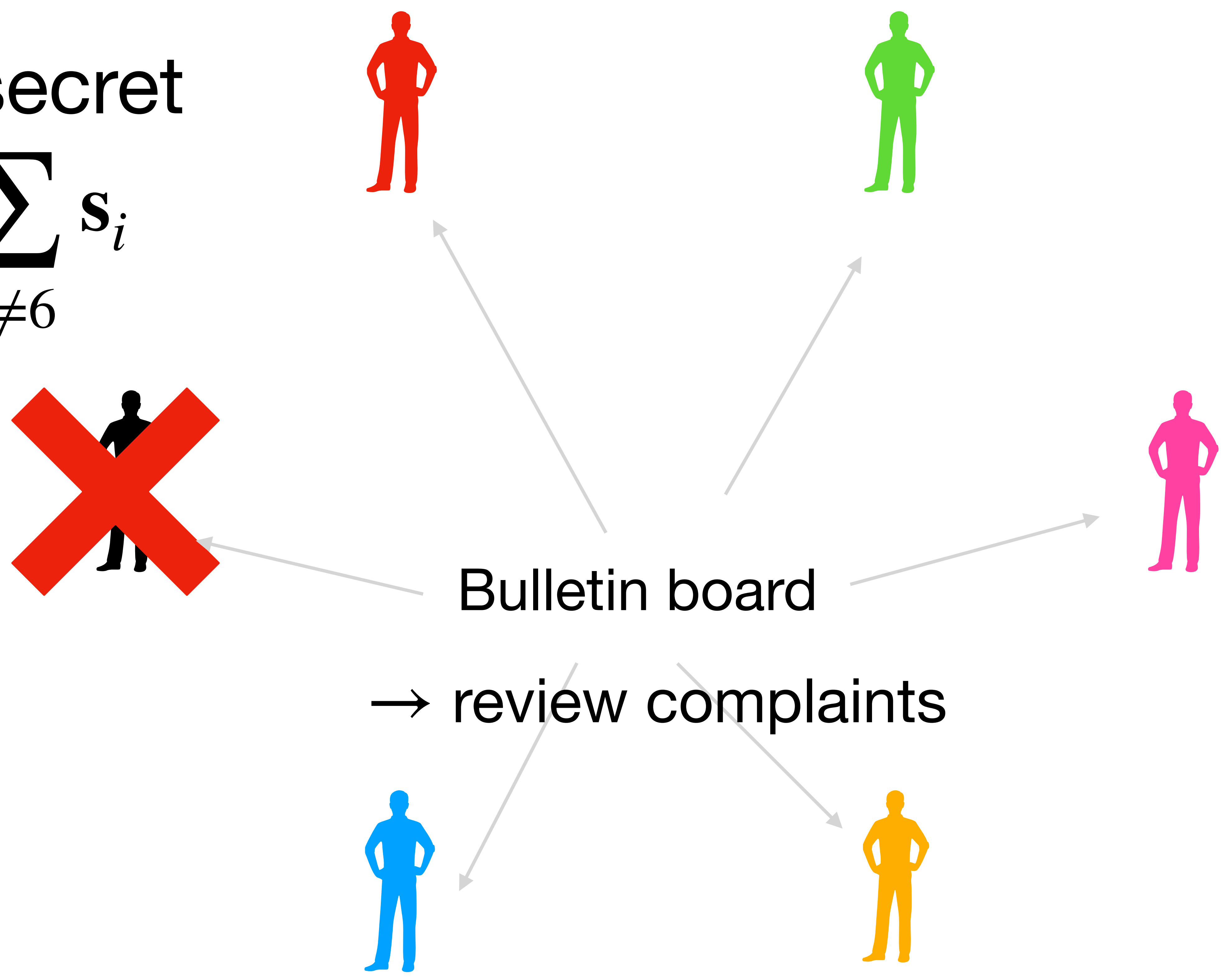
   1.b) Send shares $([\![\mathbf{s}_i]\!]_j, \pi_i^j)_j$

   1.c) Verify shares $([\![\mathbf{s}_i]\!]_j, \pi_i^j)_j$ and complain

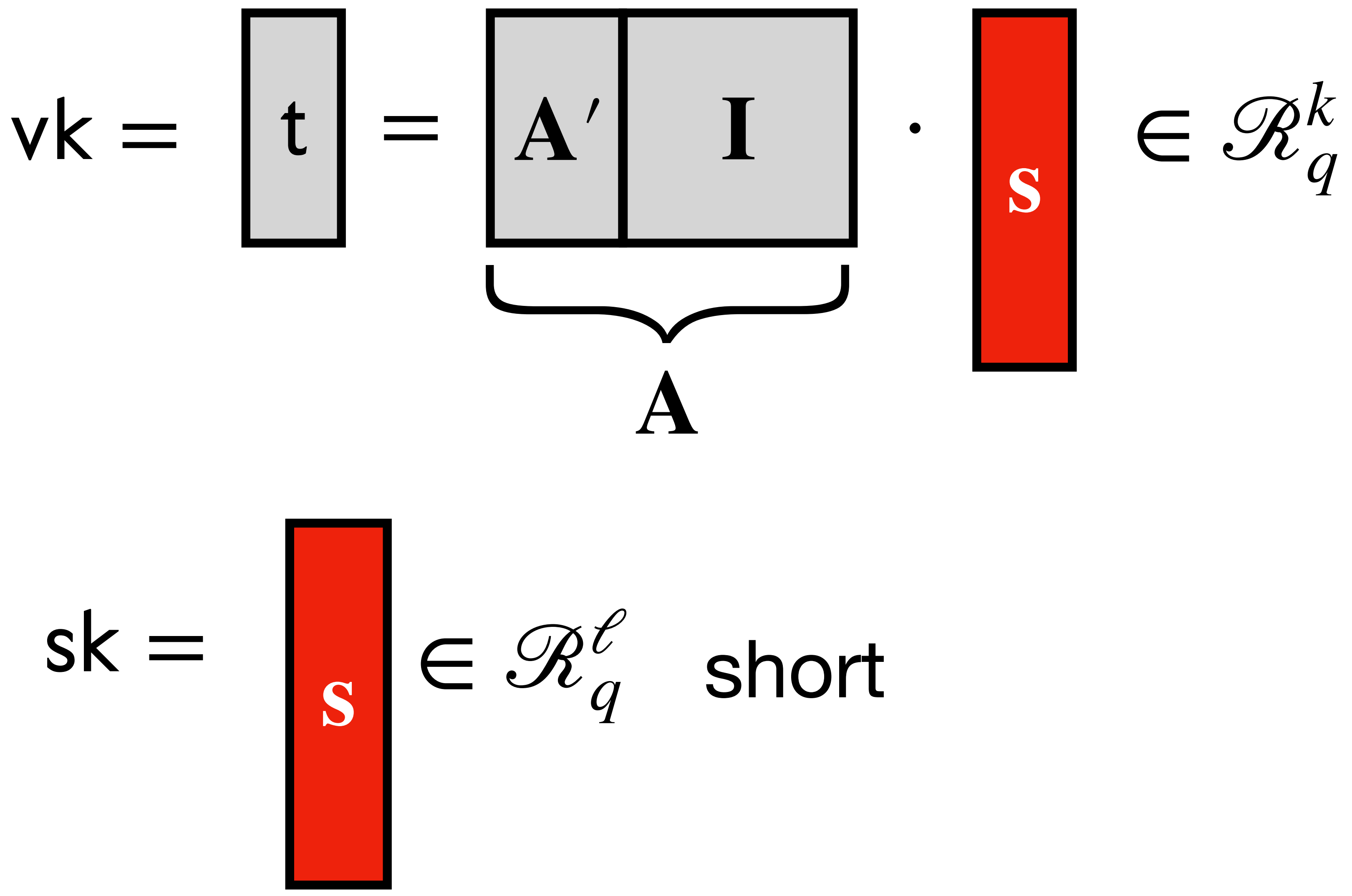
   1.d) Aggregate

Final secret

$$\mathbf{s} = \sum_{i \neq 6} \mathbf{s}_i$$



Distributed Key Generation (DKG) from VSS



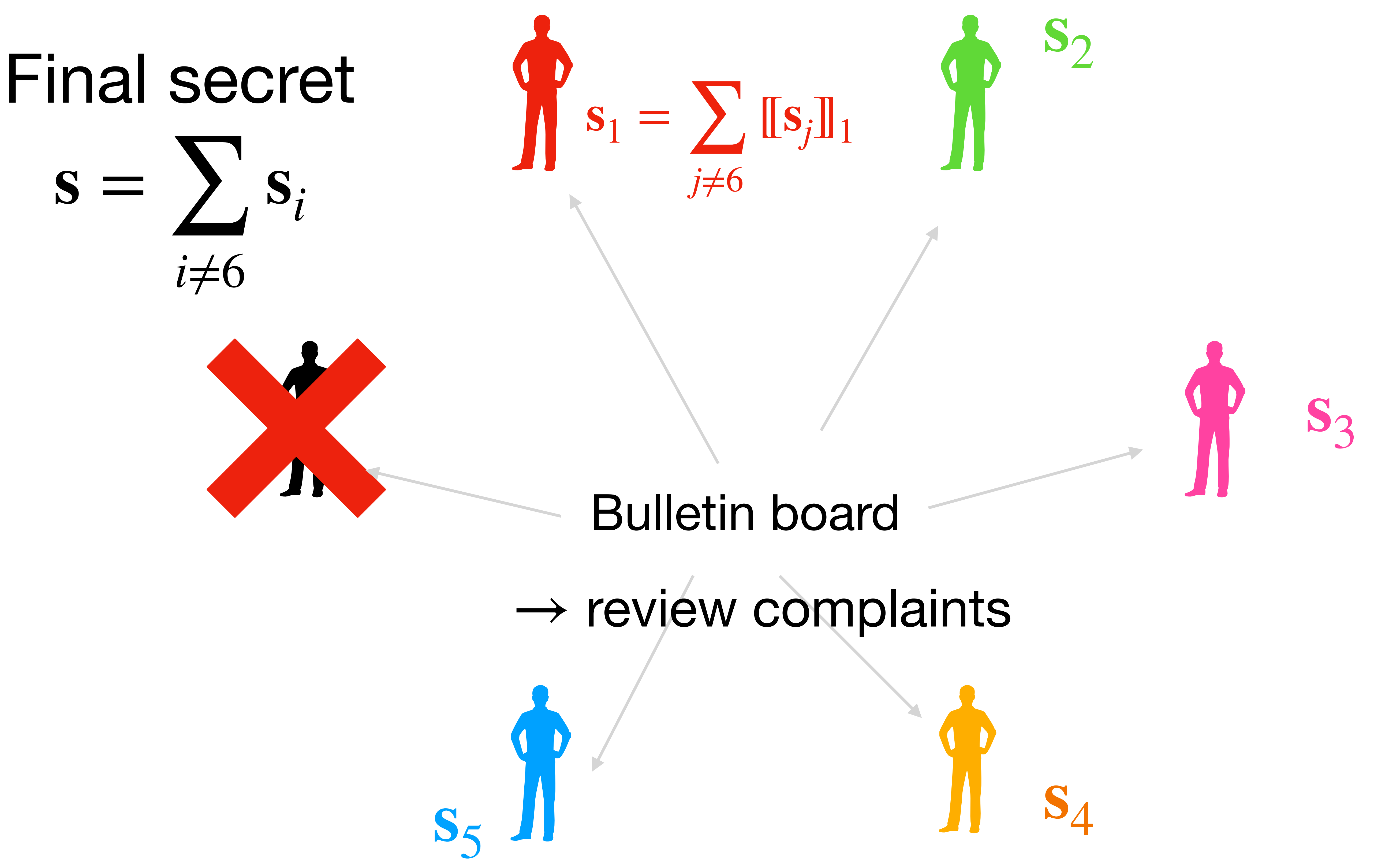
1. Construct and share secret key $\mathbf{s} = \sum_i \mathbf{s}_i$

1.a) Sample small secrets $\mathbf{s}_i$

1.b) Send shares $([\![\mathbf{s}_i]\!]_j, \pi_i^j)_j$

1.c) Verify shares $([\![\mathbf{s}_i]\!]_j, \pi_i^j)_j$ and complain

1.d) Aggregate



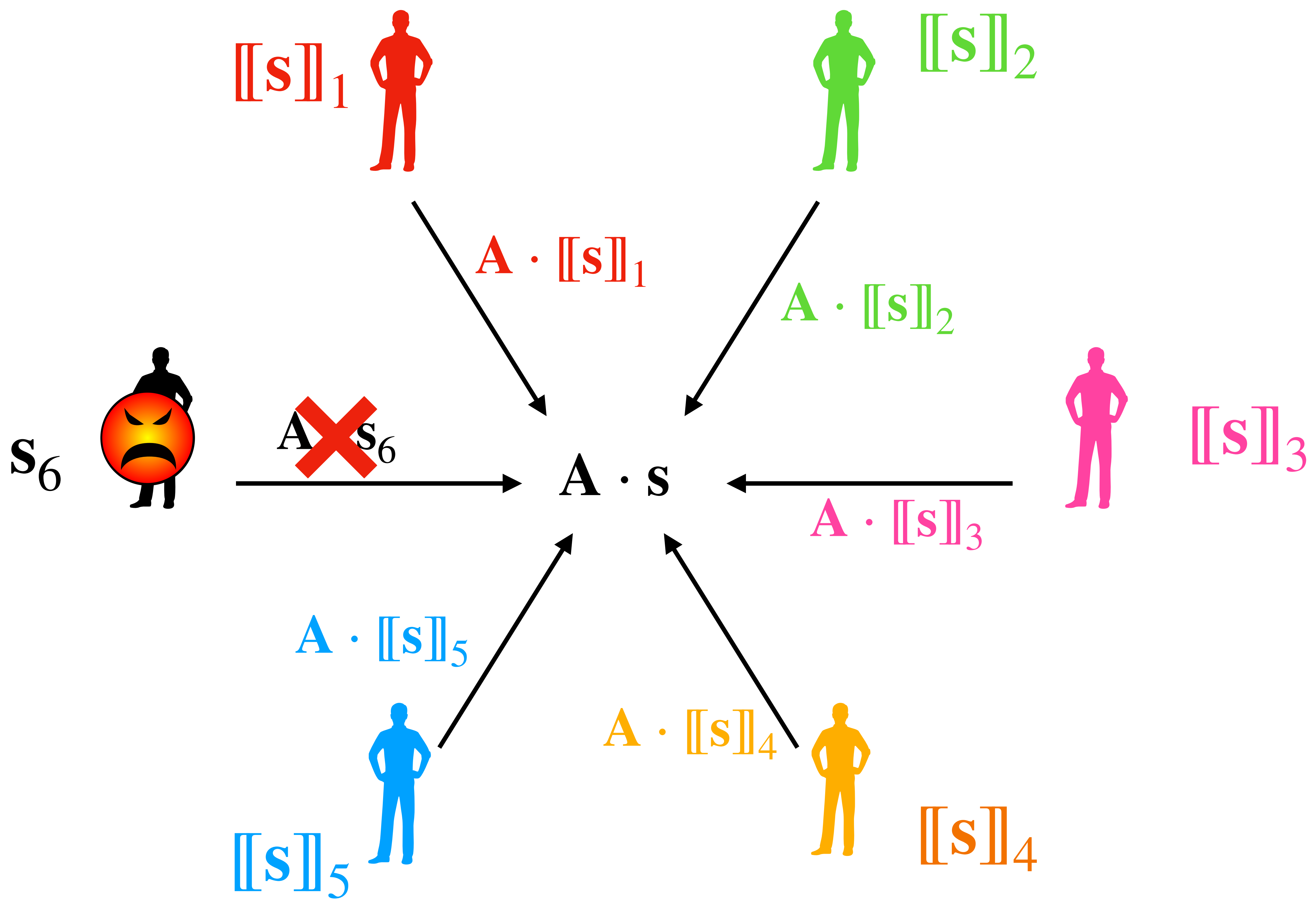
Distributed Key Generation (DKG) from VSS

$$\mathsf{vk} = \mathbf{t} = \underbrace{\begin{bmatrix} \mathbf{A}' & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \cdot \mathbf{s} \in \mathscr{R}_q^k$$

$$\mathsf{sk} = \mathbf{s} \in \mathscr{R}_q^\ell \quad \text{short}$$

1. Construct and share secret key $\mathbf{s} = \sum_i \mathbf{s}_i$

2. Compute $\mathsf{vk} = \mathbf{A} \cdot \mathbf{s}$



Distributed Key Generation (DKG) from VSS

1. Construct and share secret key $\mathbf{s} = \sum_i \mathbf{s}_i$

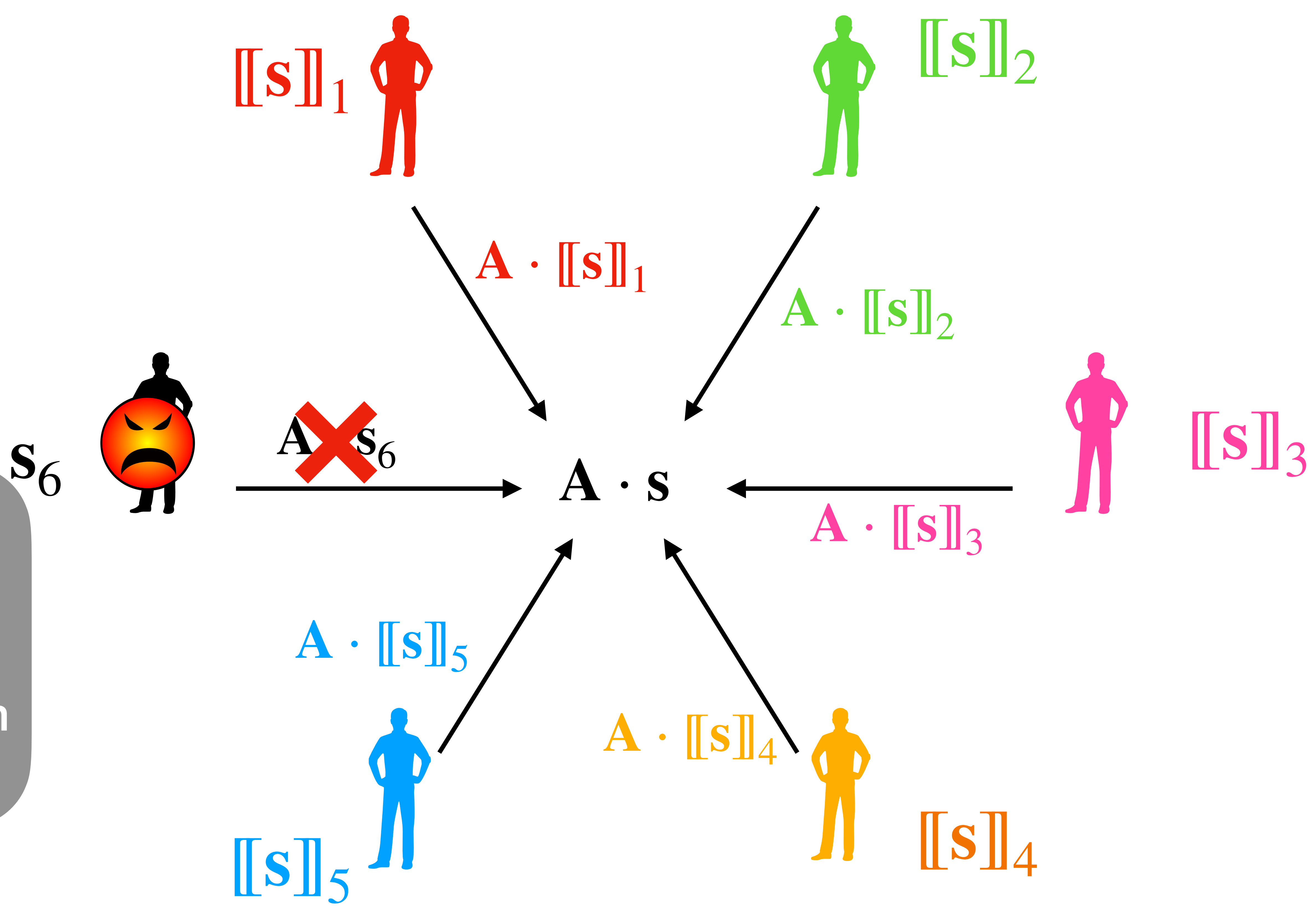
2. Compute vk $= \mathbf{A} \cdot \mathbf{s}$

vk $= \boxed{\mathrm{t}} = \underbrace{\boxed{\mathbf{A}' \mid \mathbf{I}}}_{\mathbf{A}} \cdot \boxed{\mathbf{s}} \in \mathcal{R}_q^k$

sk $= \boxed{\mathbf{s}} \in \mathcal{R}_q^\ell$ short

Use Reed-Solomon error correction to recover vk $= \mathbf{A} \cdot \mathbf{s}$

→ can only support T’=T/3 corruption



Robust Signing with VSS

Threshold Raccoon

$\mathbf{r_i} \leftarrow \chi$

$\mathbf{w}_i = \mathbf{A} \cdot \mathbf{r_i}$

$\mathsf{cmt_i} = H(\mathbf{w_i})$ →

← $(\mathsf{cmt}_j)_{j \in S}$

$\mathbf{w}_i$ →

← $(\mathbf{w}_j)_{j \in S}$

$$\mathbf{w} = \sum_{j \in S} \mathbf{w_j}$$

$$c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w})$$

$$[\![\mathbf{z}]\!]_i = c \cdot L_{S,i} \cdot [\![\mathbf{s}]\!]_i + \mathbf{r}_i + \Delta_i$$

$\mathbf{z}_i$ →

Robust Signing with VSS

Threshold Raccoon

$\mathbf{r_i} \leftarrow \chi$

$\mathbf{w}_i = \mathbf{A} \cdot \mathbf{r_i}$

$\text{cmt}_\text{i} = H(\mathbf{w_i})$ →

← $(\text{cmt}_j)_{j \in S}$

$\mathbf{w}_i$ →

← $(\mathbf{w}_j)_{j \in S}$

$\mathbf{w} = \sum_{j \in S} \mathbf{w_j}$

$c = H(\text{vk}, \text{msg}, \mathbf{w})$

$[\![\mathbf{z}]\!]_i = c \cdot L_{S,i} \cdot [\![\mathbf{s}]\!]_i + \mathbf{r}_i + \Delta_i$ — $\mathbf{z}_i$ →

Robust ThRaccoon

1) Use DKG to sample secret $\mathbf{r} = \sum_i \mathbf{r}_i$ and compute $\mathbf{w} = \mathbf{A} \cdot \mathbf{r}$: 3 rounds

Robust Signing with VSS

Threshold Raccoon

$\mathbf{r_i} \leftarrow \chi$

$\mathbf{w}_i = \mathbf{A} \cdot \mathbf{r_i}$

$\mathsf{cmt_i} = H(\mathbf{w_i})$ →

← $(\mathsf{cmt}_j)_{j \in S}$

$\mathbf{w}_i$ →

← $(\mathbf{w}_j)_{j \in S}$

$\mathbf{w} = \sum_{j \in S} \mathbf{w_j}$

$c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w})$

$[\![\mathbf{z}]\!]_i = c \cdot L_{S,i} \cdot [\![\mathbf{s}]\!]_i + \mathbf{r}_i + \Delta_i$ $\xrightarrow{\mathbf{z}_i}$

Robust ThRaccoon

1) Use DKG to sample secret $\mathbf{r} = \sum_i \mathbf{r}_i$ and compute $\mathbf{w} = \mathbf{A} \cdot \mathbf{r}$: 3 rounds

2) Compute signature shares: 1 round

$$c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w})$$

$$[\![\mathbf{z}]\!]_i = c \cdot [\![\mathbf{s}]\!]_i + [\![\mathbf{r}]\!]_i$$

If corruption threshold $T' \leq T/3$, *Reed-Solomon error correction* guarantees signature output.

3. A practical VSS with approximate shortness proof

Prior work on VSS

- Classical setting (uniform secret)
  - BGW VSS [BGW88]: IT security
  - Pedersen VSS [Ped92]: relies on DL
  - [ABCP23] based on hash functions
- VSS with shortness proof [GHL21]: quite large and DL aggregation

Our VSS

How to prove shortness of a vector $\mathbf{s}$ without revealing it?

Use a random projection to a smaller space!

Modular Johnson-Lindenstrauss lemma with offset [Ngu22]: Take a vector $\mathbf{y}$.

If a matrix $\mathbf{R}$ is sampled from a discrete distribution with coefficients ± 1 with proba $\frac{1}{4}$, and 0 with proba $\frac{1}{2}$.

Then, $\|\mathbf{R} \cdot \mathbf{s} + \mathbf{y} \bmod q\|_2$ is at least as large as $C \cdot \|\mathbf{s}\|_2$ for some $C = \omega(1)$.

Our VSS

How to prove shortness of a vector $\mathbf{s}$ without revealing it?

Use a random projection to a smaller space!

Modular Johnson-Lindenstrauss lemma with offset [Ngu22]: Take a vector $\mathbf{y}$.

If a matrix $\mathbf{R}$ is sampled from a discrete distribution with coefficients ± 1 with proba $\frac{1}{4}$, and 0 with proba $\frac{1}{2}$.

Then, $||\mathbf{R} \cdot \mathbf{s} + \mathbf{y} \bmod q||_2$ is at least as large as $C \cdot ||\mathbf{s}||_2$ for some $C = \omega(1)$.

Use small Gaussian noise keeping enough entropy in $\mathbf{s}$ instead of information theoretic.

Our VSS

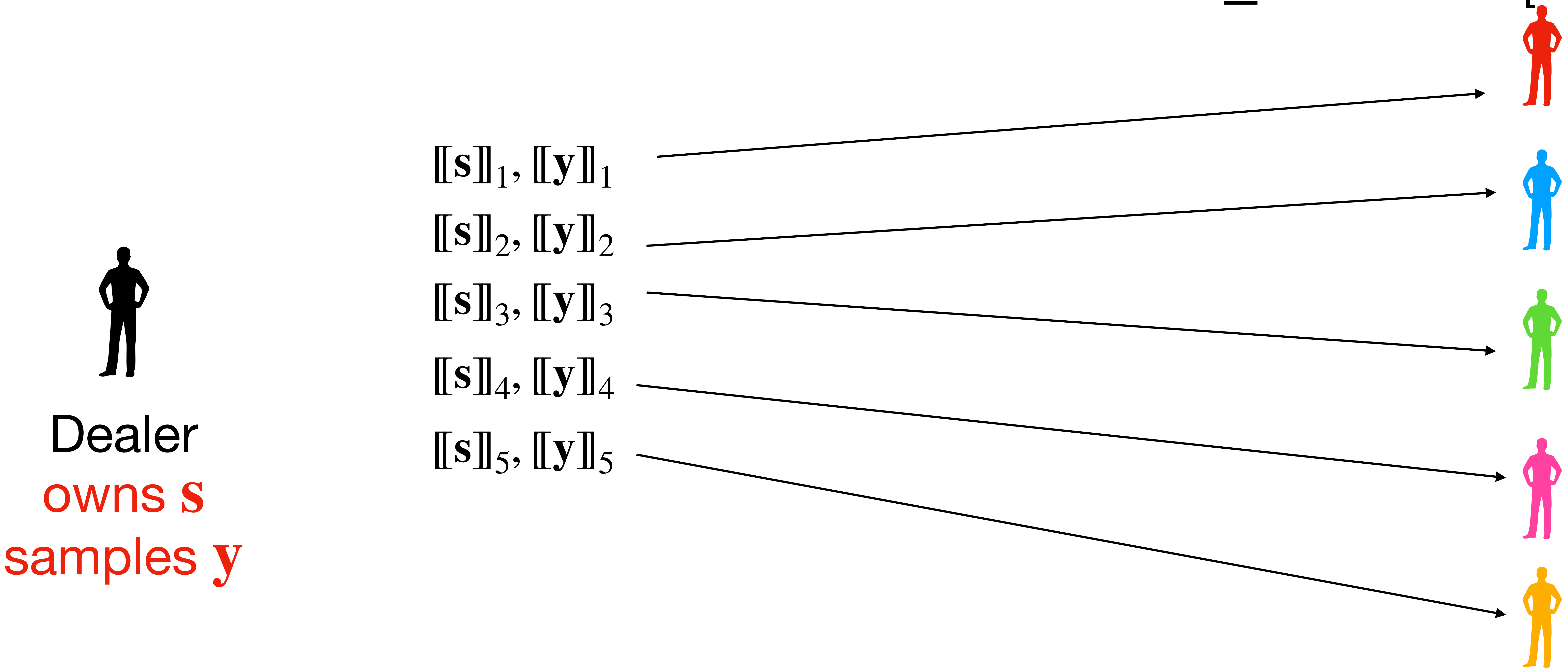
⚠ Johnson-Lindenstrauss only applies if $\mathbf{R}$ is sampled after $\mathbf{s}$ and $\mathbf{y}$.

→ Solution: hash-based verifiable randomness for $N \geq 2T'$ akin to [ABCP23].

Our VSS

Johnson-Lindenstrauss only applies if $\mathbf{R}$ is sampled after $\mathbf{s}$ and $\mathbf{y}$.

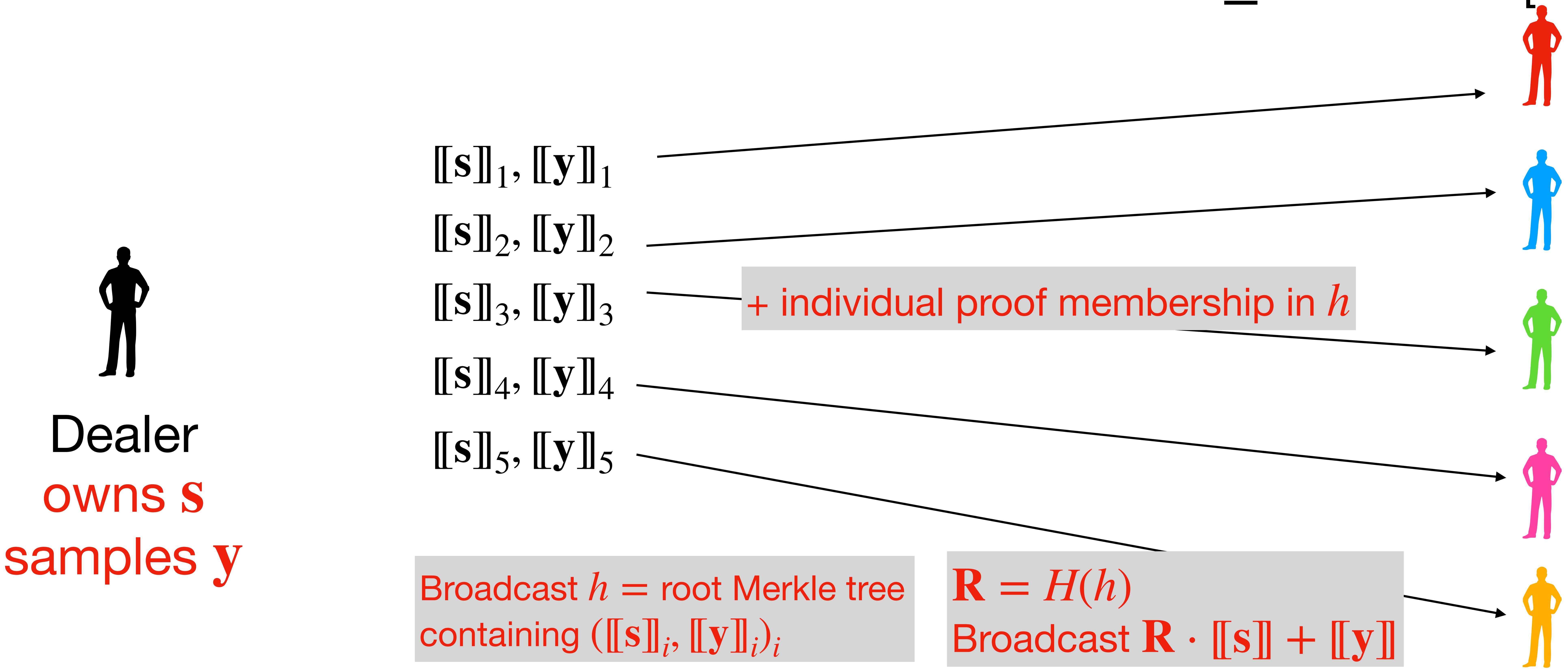
→ Solution: hash-based verifiable randomness for $N \geq 2T'$ akin to [ABCP23].



Our VSS

Johnson-Lindenstrauss only applies if $\mathbf{R}$ is sampled after $\mathbf{s}$ and $\mathbf{y}$.

→ Solution: hash-based verifiable randomness for $N \geq 2T'$ akin to [ABCP23].



Our VSS

- Our VSS reveals $\mathbf{R} \cdot \mathbf{s} + \mathbf{y}$ where $\mathbf{y}$ is Gaussian: smaller shortness gap compared to rejection sampling.
  - Not purely ZK

Zero-knowledge:

$$\left.\begin{array}{l} \pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,N} = \mathsf{VSS.Share}(\mathbf{x}) \\ \pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,T-1} = \mathsf{SimShare}() \end{array}\right\} \quad \pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,T-1} \quad \text{is indistinguishable}$$

Our VSS

- Our VSS reveals $\mathbf{R} \cdot \mathbf{s} + \mathbf{y}$ where $\mathbf{y}$ is Gaussian: smaller shortness gap compared to rejection sampling.
  - Not purely ZK : reduce security to Hint-MLWE with matrix hints

Zero-knowledge:

$\pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,N} = \mathsf{VSS.Share}(\mathbf{x})$

$\pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,T-1} = \mathsf{SimShare}()$

} $\pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,T-1}$ is indistinguishable

Fragmentary knowledge:

$\pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,N} = \mathsf{VSS.Share}(\mathbf{x})$

$\pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,T-1} = \mathsf{SimShare}(\mathbf{R} \cdot \mathbf{x} + \mathbf{y})$

} $\pi, ([\![x]\!]_i, \pi_i)_{i=1,\ldots,T-1}$ is indistinguishable

Our VSS

- Our VSS reveals $\mathbf{R} \cdot \mathbf{s} + \mathbf{y}$ where $\mathbf{y}$ is Gaussian: smaller shortness gap compared to rejection sampling.
  - Not purely ZK : reduce security to Hint-MLWE with matrix hints

- **Approximation gap ~70**, vs $\gg 2500$ in [GHL21] using JL lemma

4. Bonus: application to hash-and-sign

Fiat-Shamir vs Hash-and-Sign signatures

Fiat-Shamir

… Dilithium, Raccoon

$\mathbf{r} \leftarrow \chi$

$\mathbf{w} = \mathbf{A} \cdot \mathbf{r} \xrightarrow{\mathbf{w}}$

$\longleftarrow \quad c = H(\mathsf{vk}, \mathsf{msg}, \mathbf{w})$

$\mathbf{z} = c \cdot \mathbf{s} + \mathbf{r} \longrightarrow$

Accept if

- $\mathbf{z}$ is short
- $\mathbf{A} \cdot \mathbf{z} = c \cdot \mathbf{t} + \mathbf{w}$

Hash-and-Sign

… Falcon, Plover

$\mathbf{u} = H(\mathsf{vk}, \mathsf{msg})$

$\mathbf{z} = \mathsf{Inv}(\mathsf{sk}, \mathbf{u})$

Accept if

- $\mathbf{z}$ is short
- $\mathbf{A} \cdot \mathbf{z} = \mathbf{u} \quad (= H(\mathsf{vk}, \mathsf{msg}))$

Plover signature scheme

Based on Eagle [YJW23]

$$\mathsf{vk} = \mathbf{t} = \underbrace{\begin{bmatrix} \mathbf{A}' & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \cdot \mathbf{s} - 2^{\nu} \in \mathscr{R}_q^k \qquad \mathsf{sk} = \mathbf{s} \in \mathscr{R}_q^{\ell} \text{ short}$$

$$\mathbf{u} = H(\mathsf{vk}, \mathsf{msg})$$

$$\mathbf{r} \leftarrow \chi$$

$$\mathbf{w} = \mathbf{A} \cdot \mathbf{r}$$

$$\mathbf{u}' = \mathbf{u} - \mathbf{w} = 2^{\nu} \cdot c_1 + c_2$$

$$\mathbf{z} = \begin{bmatrix} c_1 \cdot \mathbf{s} + \mathbf{r} \\ c_1 \end{bmatrix}$$

Accept if

- $\mathbf{z}$ is short
- $[\mathbf{A} \quad \mathbf{t}] \cdot \mathbf{z} = \mathbf{u} \quad (= H(\mathsf{vk}, \mathsf{msg}))$

Conclusion

Conclusion

- Framework relying on VSS to achieve robust DKG and robust signature scheme with corruption threshold $T' = T/3$.
- **Pelican:** first lattice hash-and-sign threshold signature + DKG + robustness

Pelican = application to Plover, in this presentation applied to Raccoon

- Practical VSS scheme with approximate shortness proof: slack ~70

κ	**max T'**	**| vk |**	**| sig |**	**Communication**
128	16	12.8kB	12.3kB	26.8 + 19N kB
196	1024	25.6kB	26.4kB	53.8 + 38N kB

Proposed parameter sets for Pelican