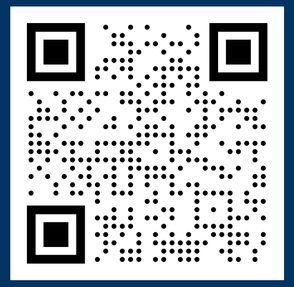
Signature Aggregation in Ethereum

Present and Future



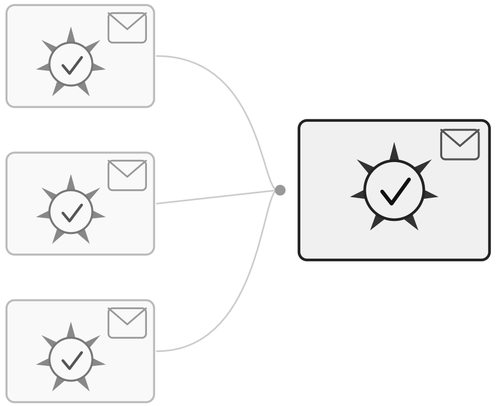
Benedikt Wagner

Ethereum Foundation

Three Concepts of This Talk

Ethereum and Proof of Stake

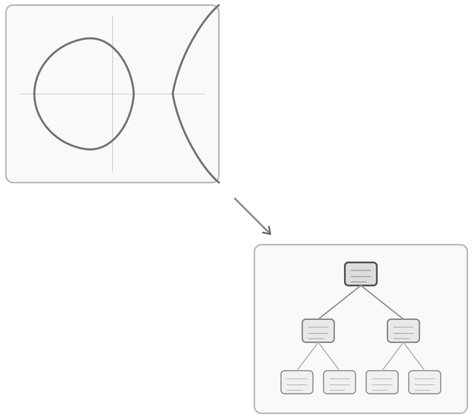
Non-Interactive Multi-Signatures



BLS

⟶

Post-Quantum

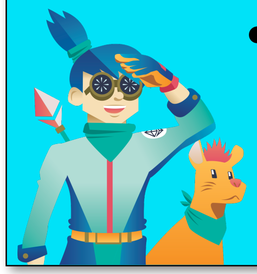


Signature Aggregation Today

Proof of Stake 101

Proposer

- Assemble block
- Gossip block



Block

Validators

pk_1, sk_1 pk_2, sk_2 pk_3, sk_3 pk_4, sk_4

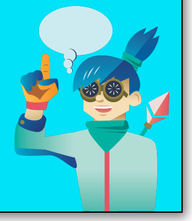
Sig_1 Sig_2 Sig_3 Sig_4

- Verify transactions
- Sign to vote
- Gossip signatures

$Sig_1, \ldots, Sig_4$

Aggregator

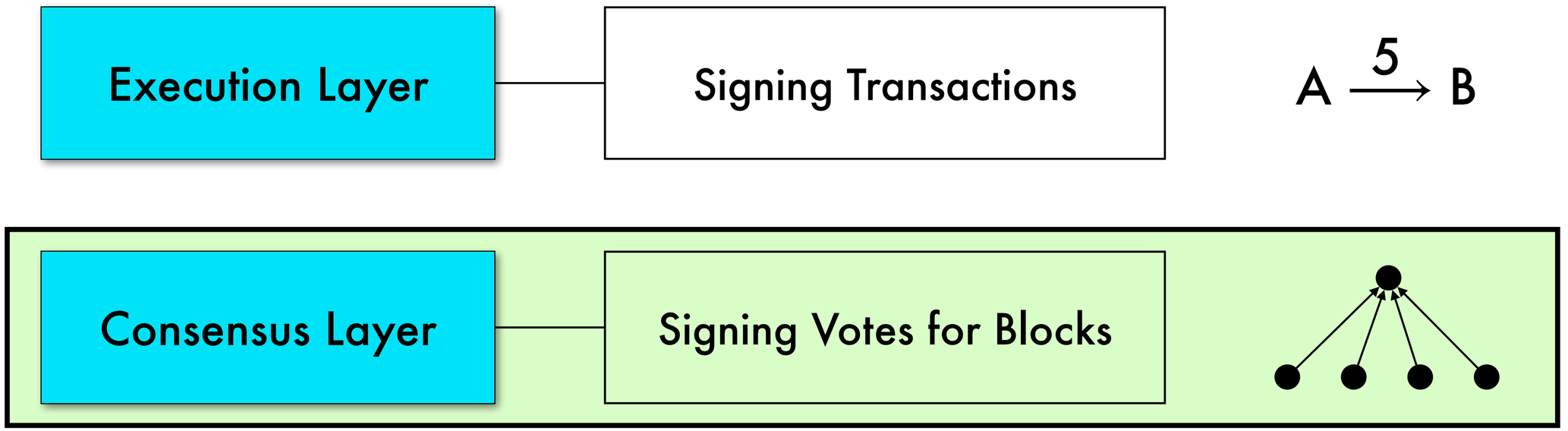
$\overline{Sig}$ ← Aggregate ← $Sig_1, \ldots, Sig_4$



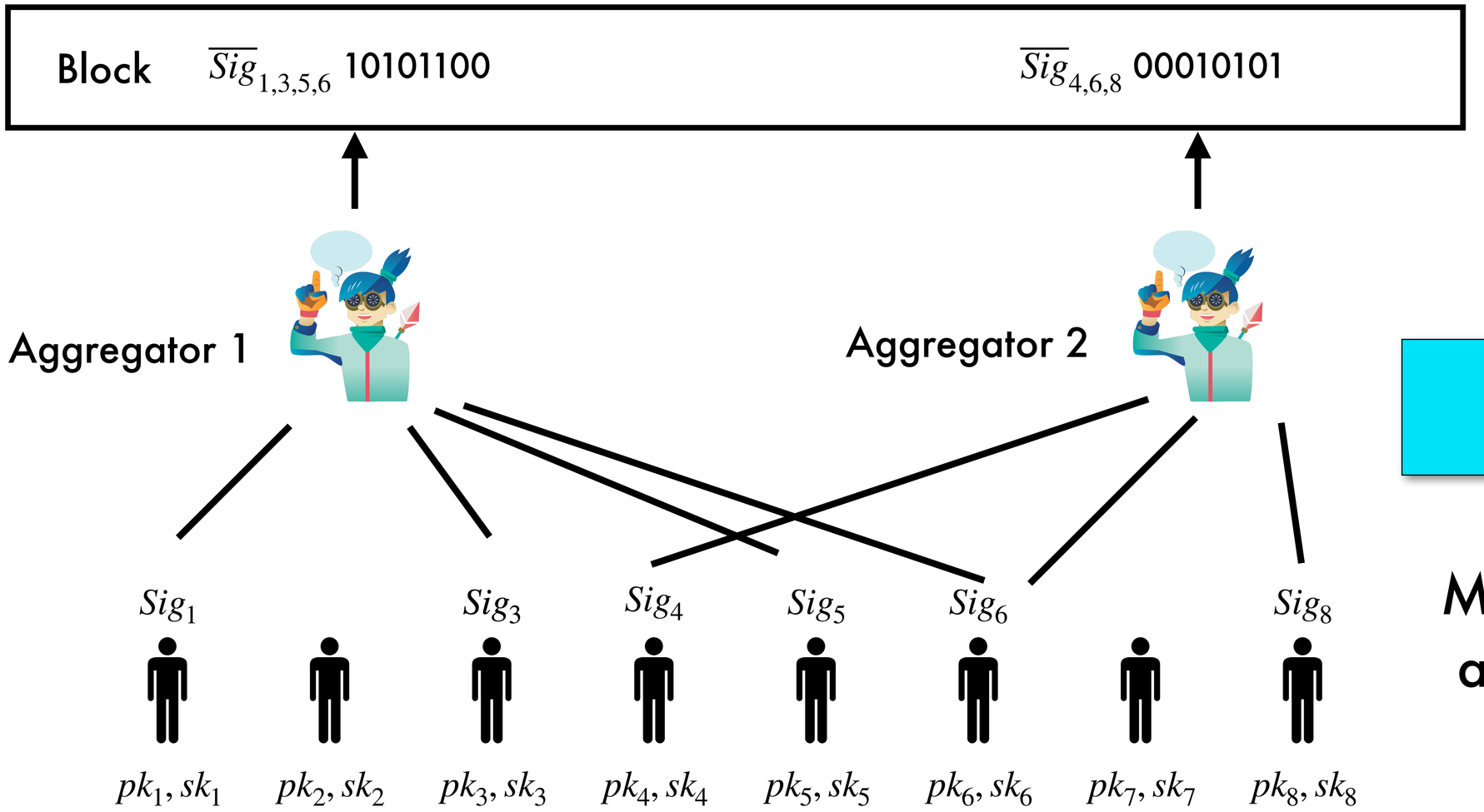
Block | Block | Block | Block $\overline{Sig}$

Block valid ⇔ enough votes in $\overline{Sig}$

Two Use Cases of Signatures



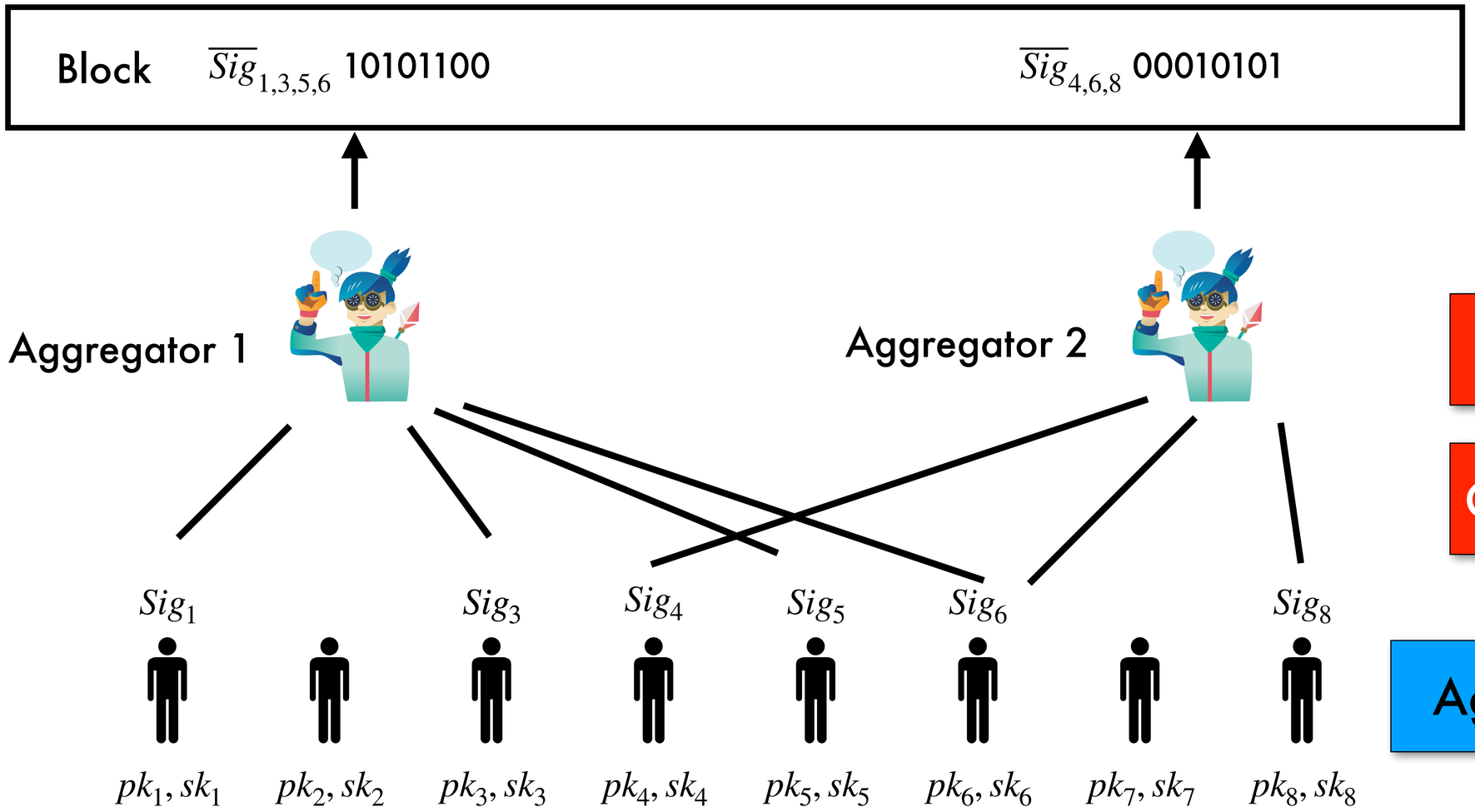
Signature Aggregation Today



BLS Multi-Signatures

Multi-signatures vs. aggregate signatures

Signature Aggregation Today



Bitvectors Large

One per aggregator

Aggregate aggregates?

BLS Signatures and Multi-Signatures

Prime Order Pairing Groups $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ $|\mathbb{G}| = p$

Secret Key $sk \leftarrow \mathbb{Z}_p$

Public Key $pk = g^{sk}$

Signing $Sig = \mathsf{H}(M)^{sk}$

Verification $e(Sig, g) = e(\mathsf{H}(M), pk)$

Aggregation $Sig_1, \ldots, Sig_n \longmapsto Sig_1 \cdot \cdots \cdot Sig_n = \mathsf{H}(M)^{sk_1 + \cdots + sk_n}$

Aggregating Aggregates with BLS

$\overline{Sig}_{1,3,5,6}$ 10101100

$\overline{Sig}_{4,6,8}$ 00010101

$\overline{Sig}_{1,3,5,6} = \mathrm{H}(M)^{sk_1+sk_3+sk_5+sk_6}$

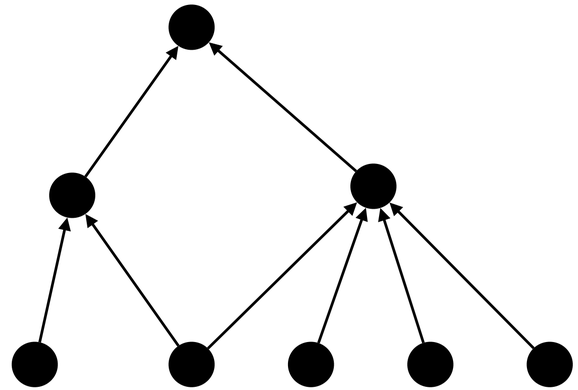
$\overline{Sig}_{4,6,8} = \mathrm{H}(M)^{sk_4+sk_6+sk_8}$

$\overline{Sig}_{1,3,5,6} \cdot \overline{Sig}_{4,6,8} = H(M)^{sk_1+sk_3+sk_4+sk_5+2sk_6+sk_8}$ 10111201

Counters instead of bitvectors!

Hint-Free Multi-Signatures [HRW26]

- KeyGen$(1^\lambda) \to (pk, sk)$
- Sign$(sk, M) \to Sig$
- Aggregate$(Sig_1, \ldots, Sig_\ell) \to Sig$
- Verify$(\{pk_1, \ldots, pk_n\}, Sig) \to 0/1$



Sig_i may already be aggregated

Keys given as a set
(no multiset, no counters, no topology)

Hint-Free Multi-Signatures [HRW26]

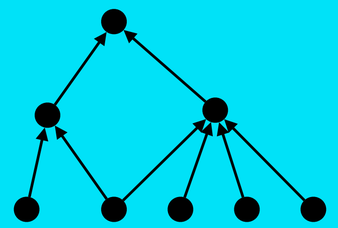
From Pairings / BLS

Need Multiplicities

10111201

From BARGs [DGKV22]

Need Topology



From Recursive SNARKs

Hint-Free

Signature Agnostic

Security unclear or depth bounded

Hint-Free Multi-Signatures [HRW26]

Feasibility without Recursion

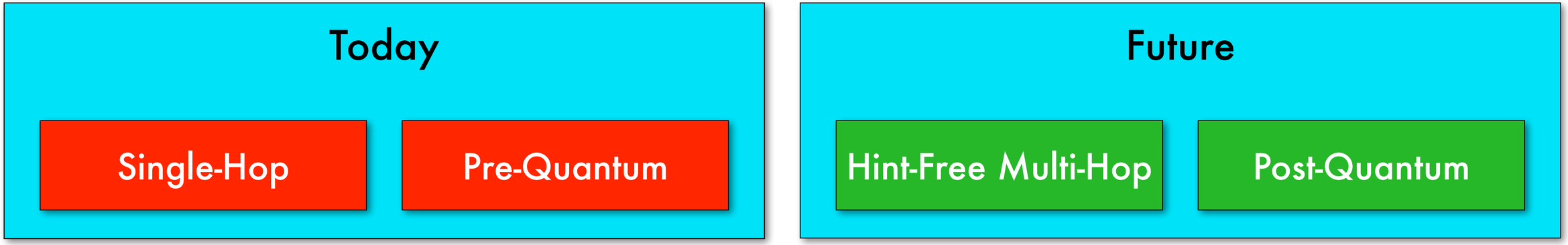
Not Practical (iO)

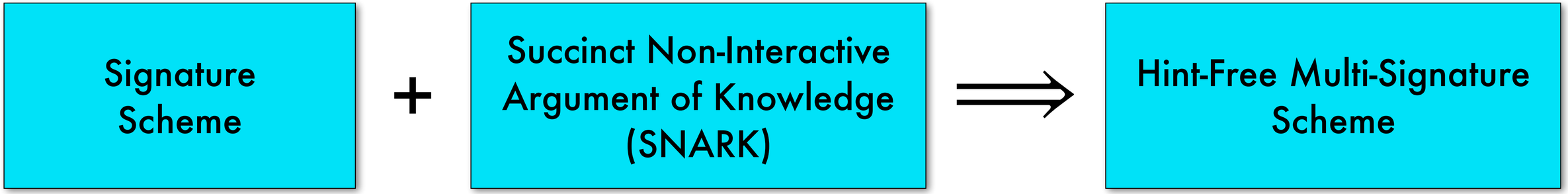
[DKGV22]: Rate-1 Non-Interactive Arguments for Batch-NP and Applications

[HRW26]: Hint-Free Multi-Signatures

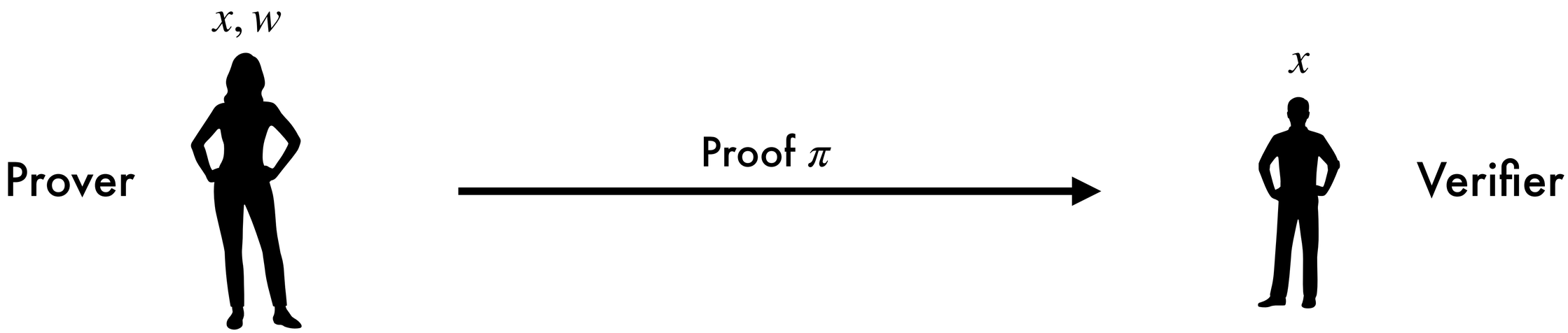
Signature Aggregation in the Future

Signature Aggregation in the Future





Non-Interactive Arguments for NP relation $\mathscr{R}$

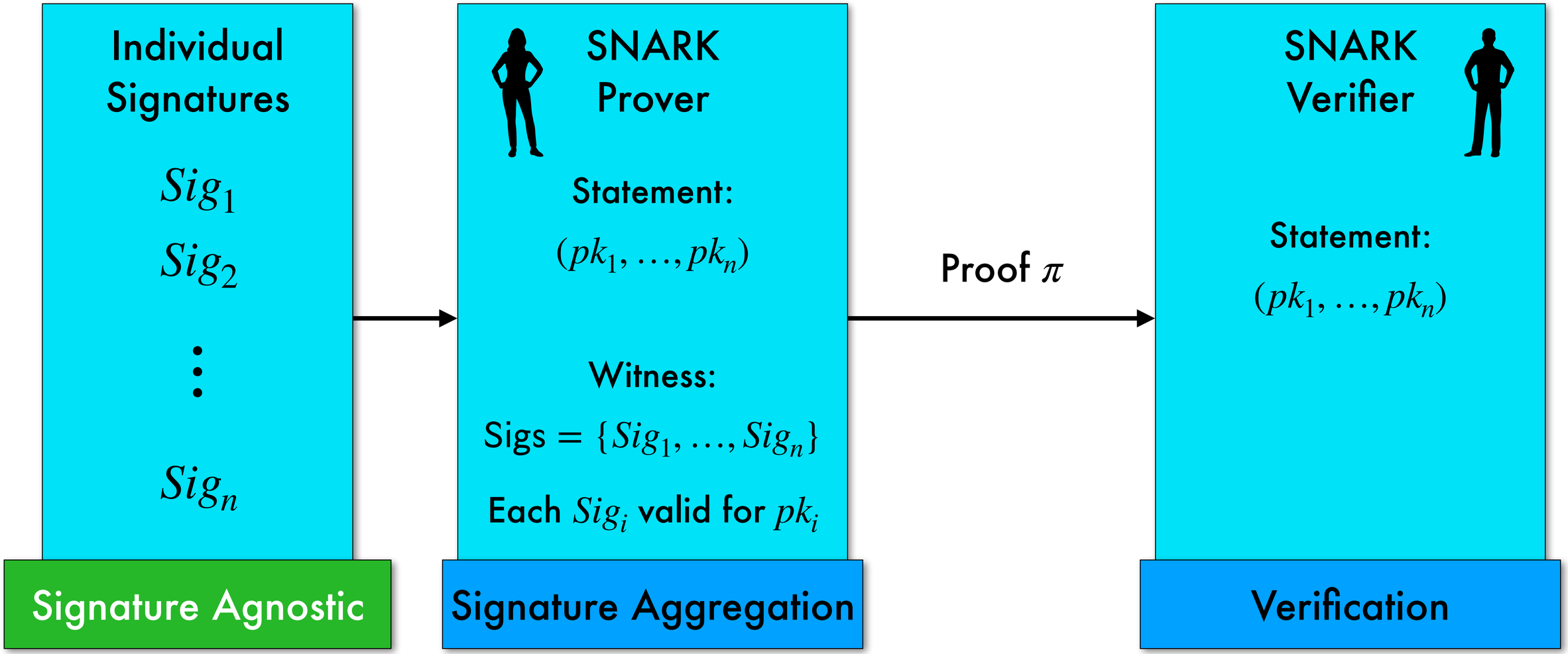


Completeness: $(x, w) \in \mathscr{R} \Longrightarrow$ V accepts

Succinctness: $|\pi| \ll |w|$

Knowledge Soundness: V accepts $(x, \pi) \Longrightarrow$ P knows $(x, w) \in \mathscr{R}$

Signature Aggregation via SNARKs



PQ Schemes!

Hint-Free Multi-Signatures via SNARKs

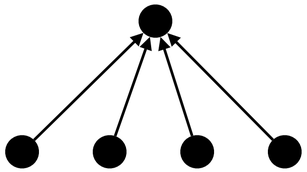
Relation $\mathscr{R}$

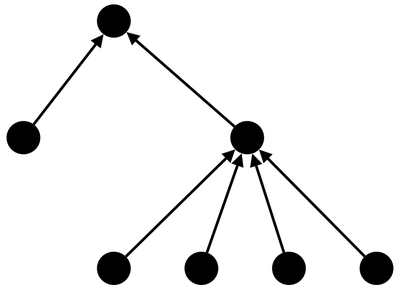
Statement: $(pk_1, \ldots, pk_n, S)$ where $S \subseteq [n]$ Witness: (Proofs, Sigs)

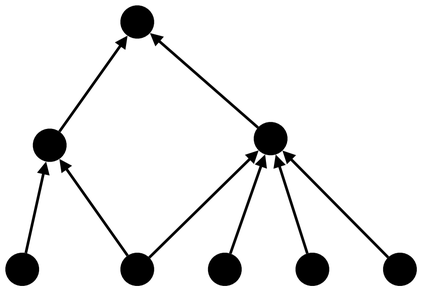
Constraint: $\forall i \in S$:

$\exists Sig \in$ Sigs : Sig valid signature for pk_i, M

OR $\exists(\pi, S') \in$ Proofs : $i \in S' \wedge$ π valid proof for $(pk_1, \ldots, pk_n, S')$ Recursion!



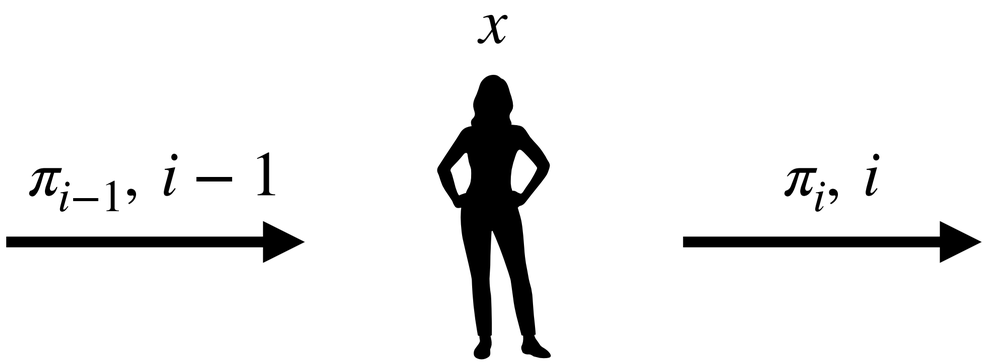




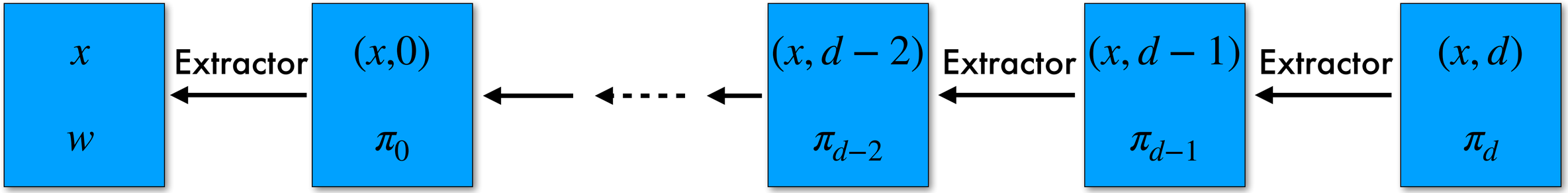
* depth needed in $\mathscr{R}$

Recursion and its Problems

1: Bounded Depth



$$\mathscr{R}' = \{((x,0), w) \mid (x, w) \in \mathscr{R}\}$$
$$\cup \{(x, i), \pi) \mid \mathsf{Ver}((x, i-1), \pi)\}$$



Must be poly!

Time $2^d T$

Time $2 \cdot 2T$

Time $2T$

Time T

Depth bounded!

What about straight-line extraction?

Recursion and its Problems

2: Relativized Arguments

Many SNARKs in Random Oracle Model are Straight-line Extractable!

$$\mathscr{R}' = \{((x,0), w) \mid (x, w) \in \mathscr{R}\} \cup \{(x, i), \pi) \mid \mathsf{Ver}^{\mathsf{H}}((x, i-1), \pi)\} \in \mathsf{NP}^{\mathsf{H}}$$

Relativized SNARK:

SNARK for NP^{H}

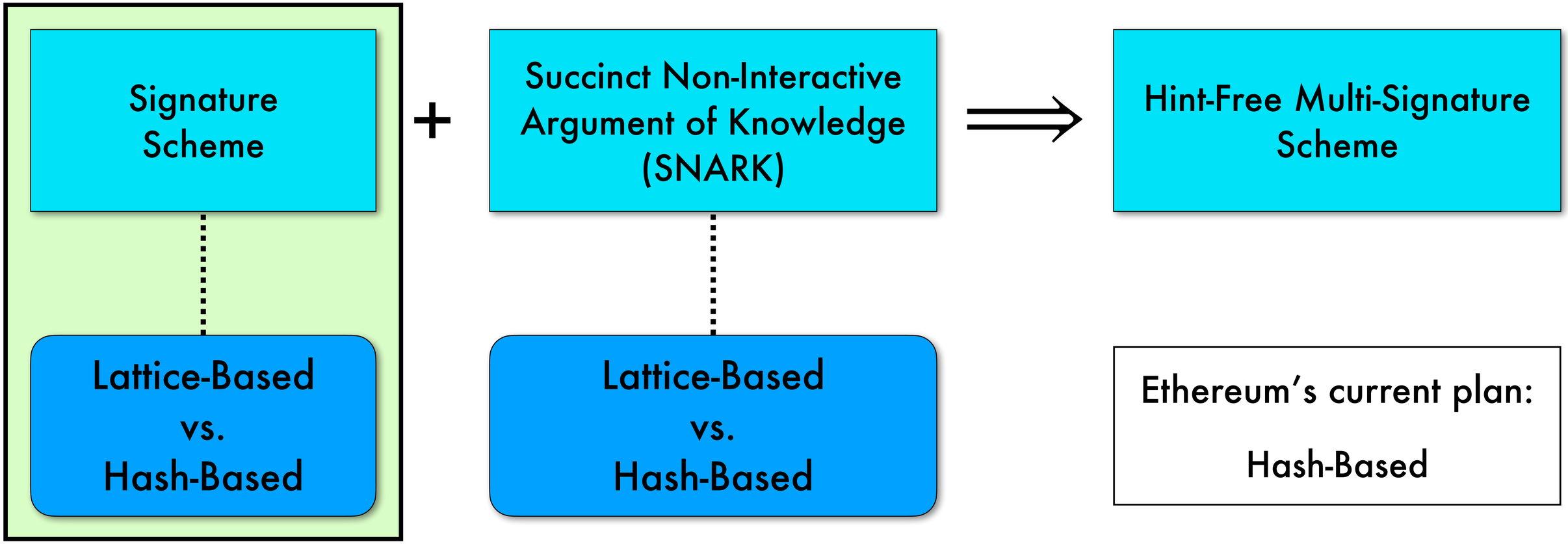
Impossible in ROM [BCG24]

$\Longrightarrow$

Better Understanding of Recursion needed!

[BCG24]: Relativized Succinct Arguments in the ROM Do Not Exist

Signature Aggregation in the Future

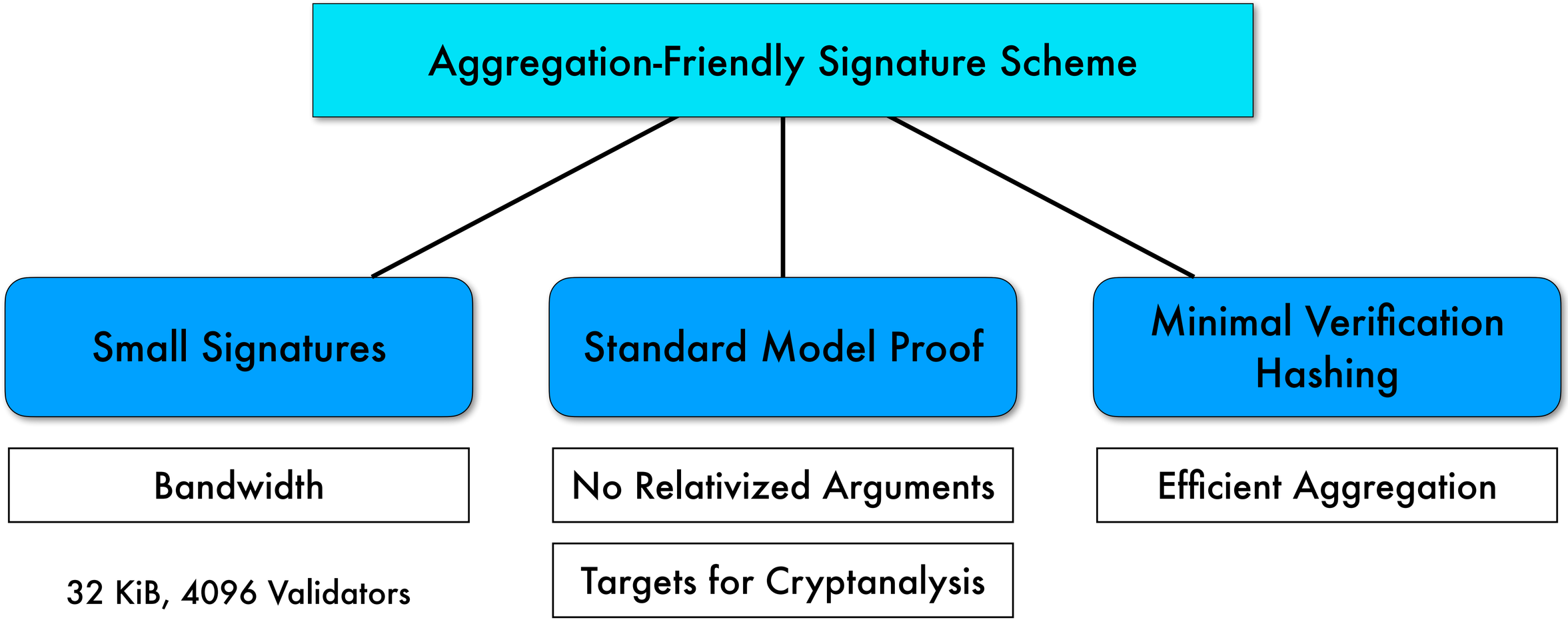


Why Hash-Based Signatures?

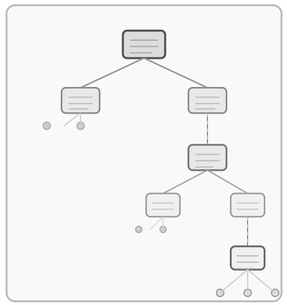
Assumption Minimality	Other schemes also need Hash Functions
Conceptual Simplicity	Ease of implementation and understanding
Information-Theoretic Security in ROM	Setting Parameters & Formal Verification
Standard Model Analysis	Concrete Targets for Cryptanalysis

Which Hash-Based Signature Scheme?

Signature Requirements

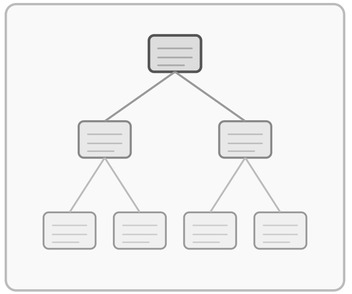


Hash-Based Signature Candidates



SPHINCS+

Somewhat Complex



XMSS

Simple

Synchronized / Stateful

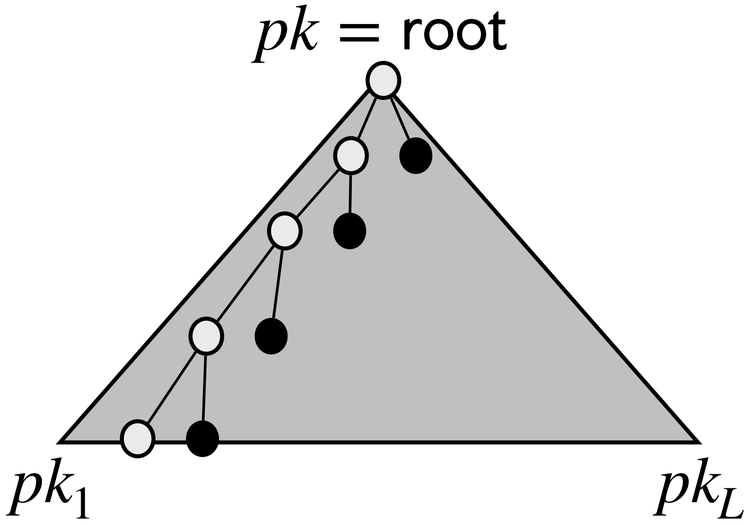
No problem for Proof of Stake!

XMSS Overview From One-time to Many-Time

One-Time Signature

Sign only once

Two signatures leak sk



Synchronized Many-Time Signature

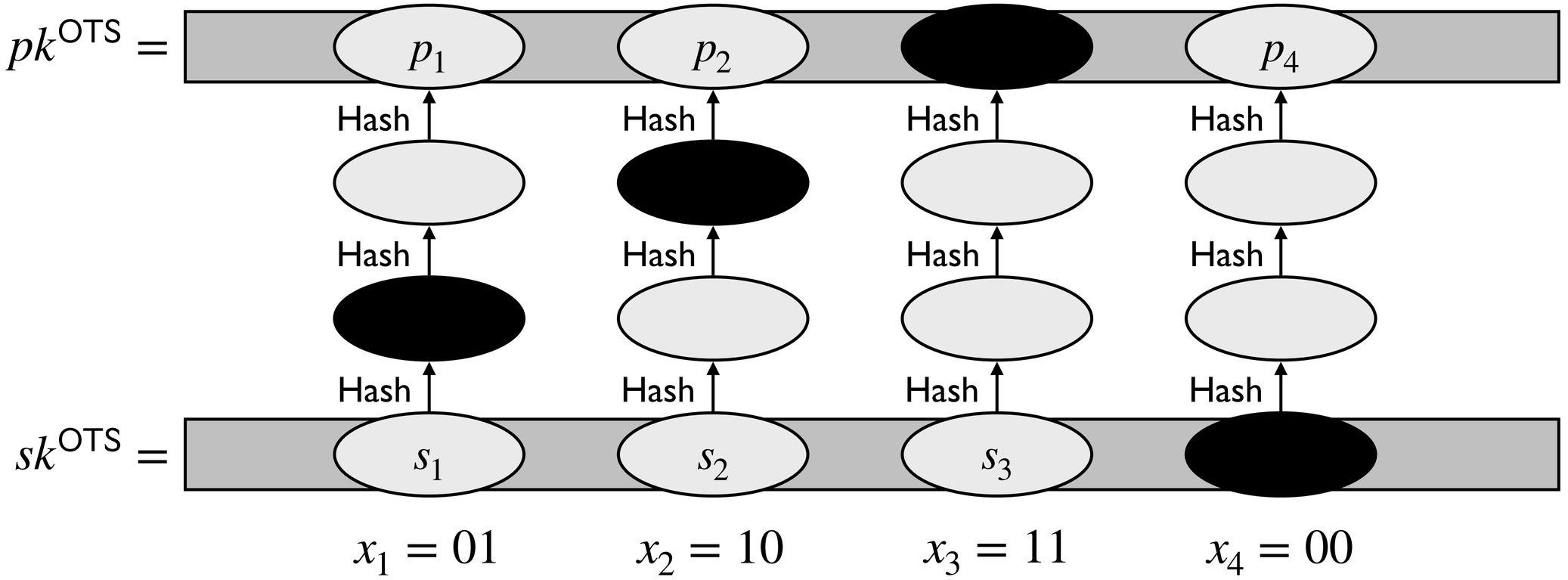
Sign wrt slots

Sign only once per slot

Good fit for PoS:
Only one vote per slot

XMSS Overview One-Time Signatures from Hash Chains

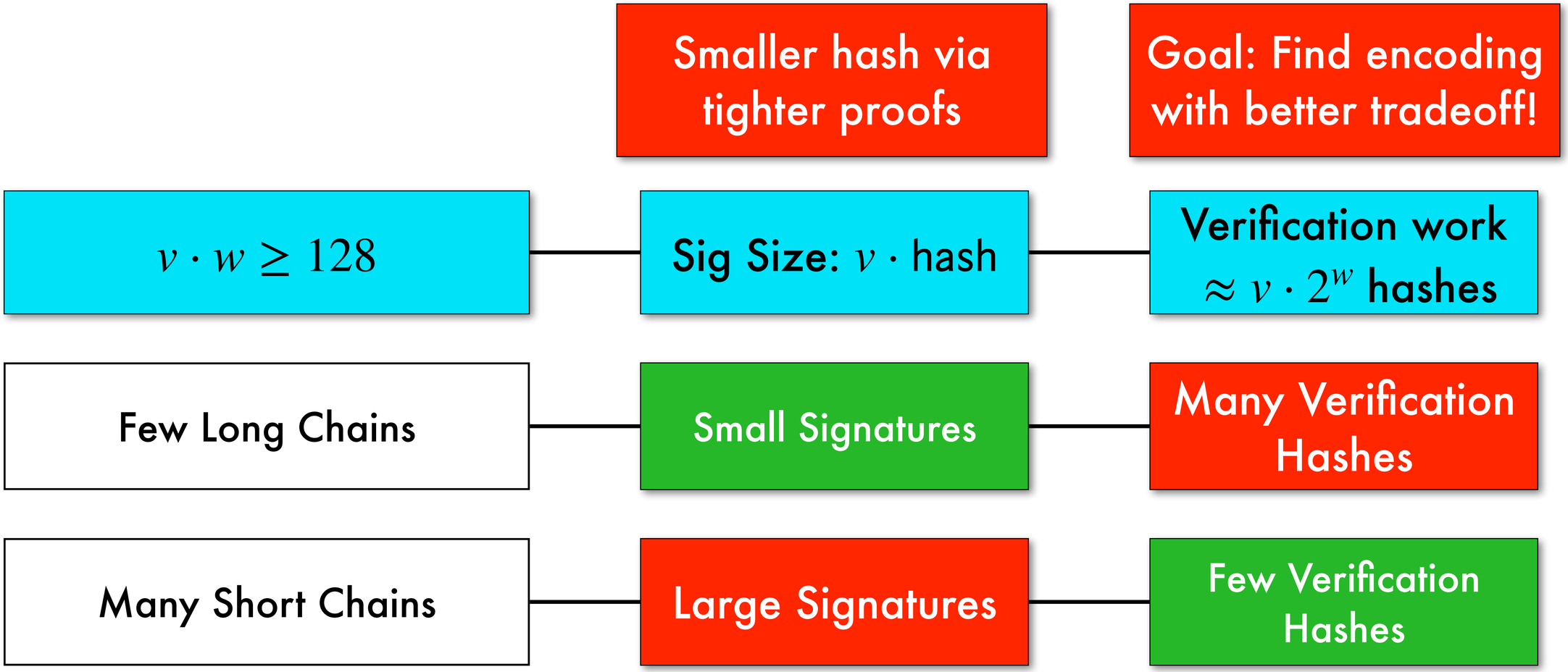
Example: $v = 4$ chains of length $2^w = 4$



Message m →
Randomness ρ →
Encoding → $(x_1, \ldots, x_v)$ Positions in Chains

+ incomparability
target collision-resistance

Tradeoffs for XMSS



Our Work on XMSS

Hash Sigs for Ethereum [DKKW25]

- Standard Model Proofs
- Tighter Proofs
- From Weaker Assumptions
- Concrete Parameters

Top of the Hypercube [KKW25]

- Hypercube Framework For encodings
- Tradeoff Lower Bound
- Improved Tradeoff

Aborting Random Oracles [HKKTW26]

- Idealized Model for Aborting hash functions
- SNARK-friendly Encodings
- Grinding in Fiat-Shamir

Summary and Research Directions

Summary and Research Directions

Signature Aggregation in Ethereum	Practical Hint-Free Multi-Signatures?
Recursive SNARKs $\Longrightarrow$ Hint-Free Multi-Signatures	Security of Recursion?
Post-Quantum Multi-Signatures Via SNARKs	PQ Signature with “native” aggregation?
Work on XMSS signatures	Security of Poseidon Hash Function?

Signature Aggregation in Ethereum

Thank you!

Benedikt Wagner

Ethereum Foundation