STRĀTUMN

CRYPTO NIGHTS
№1

HASH-BASED SIGNATURES

MOURAD BEJI

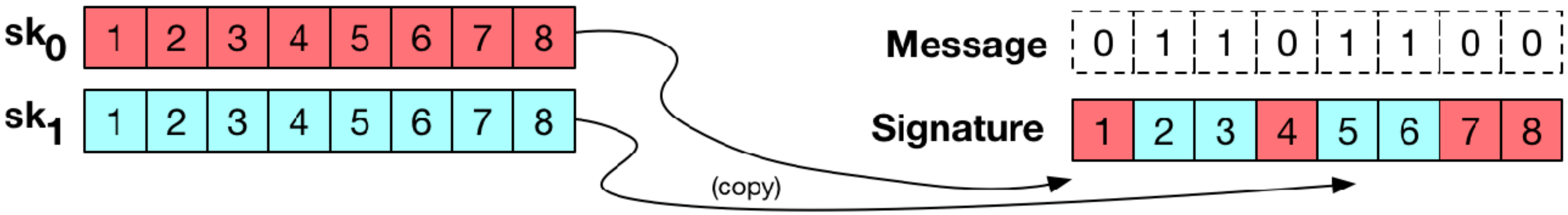
–

11.06.2018

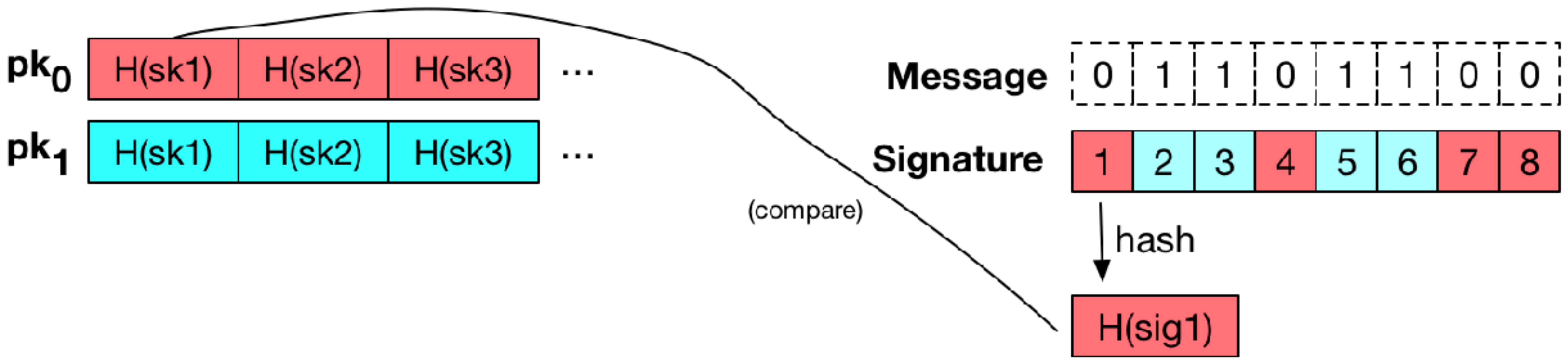
Lamport One-Time Signature (1979)

- We want to sign a 256-bit message
- We generate 512 random strings of 256 bits

$$\mathbf{sk_0} = sk_1^0, sk_2^0, \ldots, sk_{256}^0 \qquad \mathbf{pk_0} = H(sk_1^0), H(sk_2^0), \ldots, H(sk_{256}^0)$$
$$\mathbf{sk_1} = sk_1^1, sk_2^1, \ldots, sk_{256}^1 \qquad \mathbf{pk_1} = H(sk_1^1), H(sk_2^1), \ldots, H(sk_{256}^1)$$



Lamport One-Time Signature

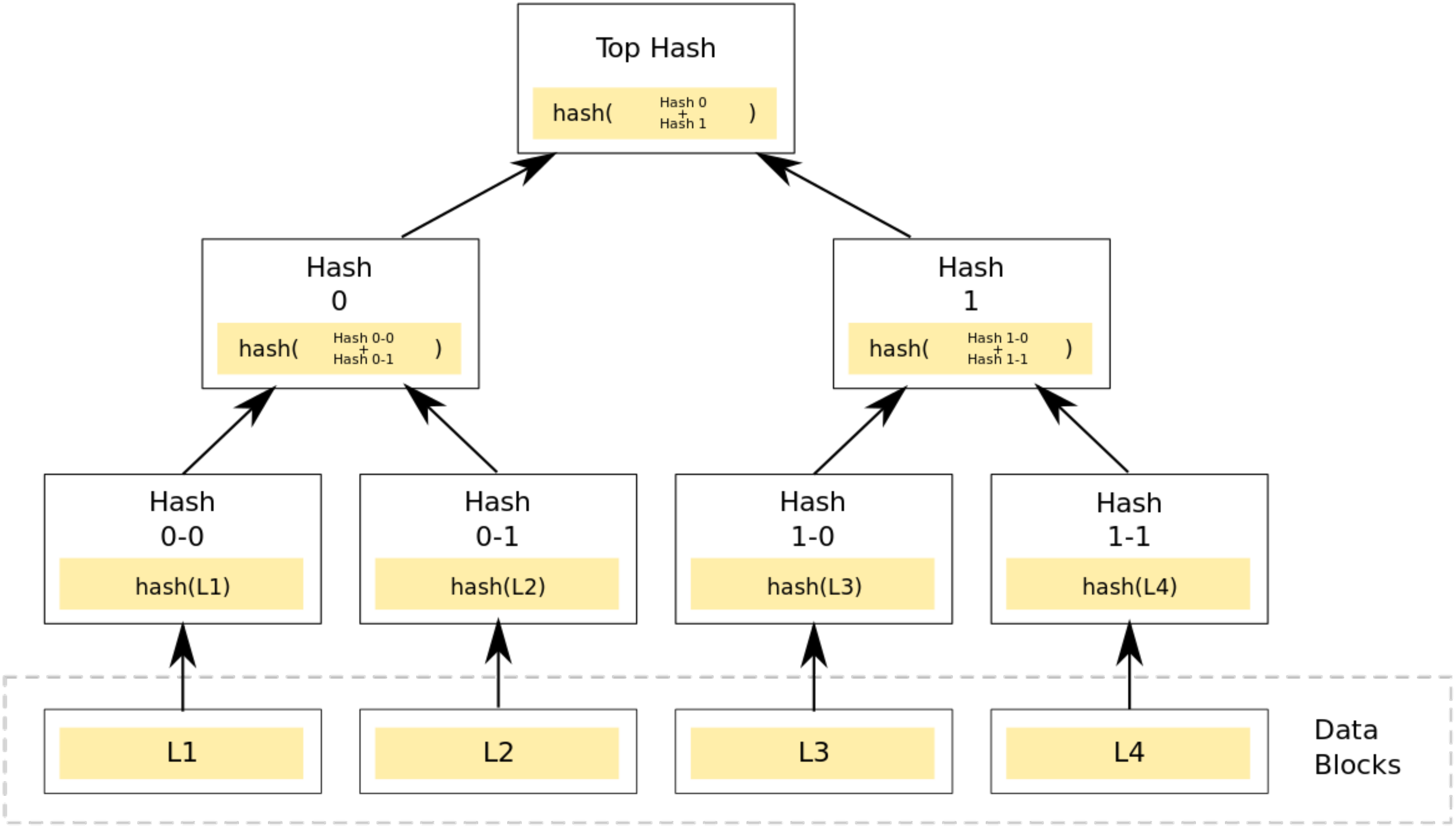


- Big Buts:
  - Huge public key
  - Huge signature
  - One-time signature

Merkle Tree-Based Signature

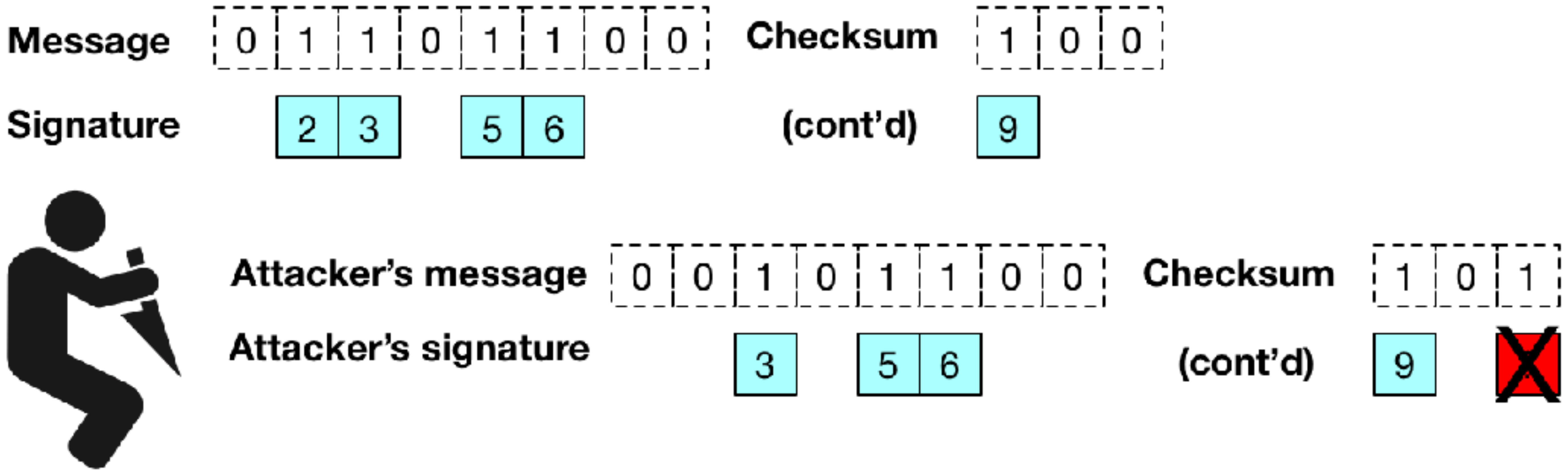
- To sign N messages, generate N separate Lamport keypairs
- Compute the root of the Merkle tree of all the public keys (Master Public Key) and distribute it
- When signing, include the Merkle Proof
- Buts:
  - Big signature
  - Big private key (but can be optimized by using a PRNG and keeping just a seed)

Merkle Trees



Merkle: Compression And Checksum

- Sign only the 1’s
- AND a checksum!



Message	0	1	1	0	1	1	0	0	Checksum	1	0	0
Signature		2	3		5	6			(cont’d)	9		
Attacker’s message	0	0	1	0	1	1	0	0	Checksum	1	0	1
Attacker’s signature			3		5	6			(cont’d)	9		☒

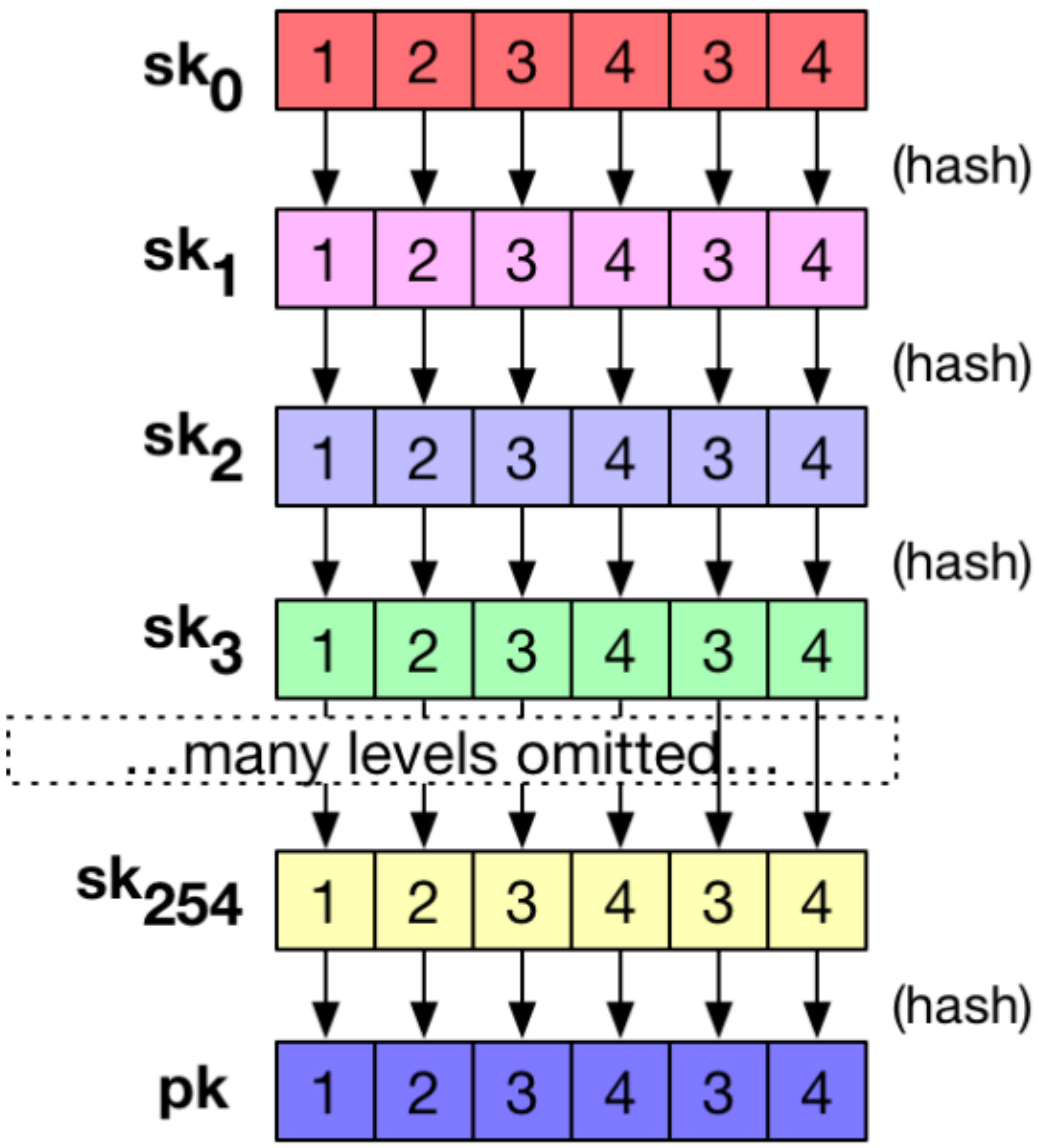
Winternitz OTS: Trading Space For Time

- Sign bytes rather than bits
- We now need 256 random lists: Hash all the things!

Message (bytes)	0	1	2	3	**Checksum**	03	246
Signature	1	2	3	4	**(cont'd)**	03	246

- And a clever checksum

$$\sum_{i=1}^{\ell} 255 - M_i$$



Why?

- Fast and simple: just hash evaluations
- Quantum resistant… Or at least not broken by Shor Algorithm and the likes

Questions?