Overview of Post-Quantum Hash-Based Signature Schemes

Nathan Manohar

IBM Research

Post-Quantum Signature Schemes

- Quantum computer break many existing signature schemes
  - RSA, ECDSA, BLS, El Gamal, Rabin, etc.
  - Factoring, discrete log, bilinear maps, etc.

- NIST ran a “not a competition” standardization process
  - Recently (July 2022) announced 3 signatures schemes to standardize
  - CRYSTALS-Dilithium, FALCON, SPHINCS+

PQ Hash-Based Signature Schemes

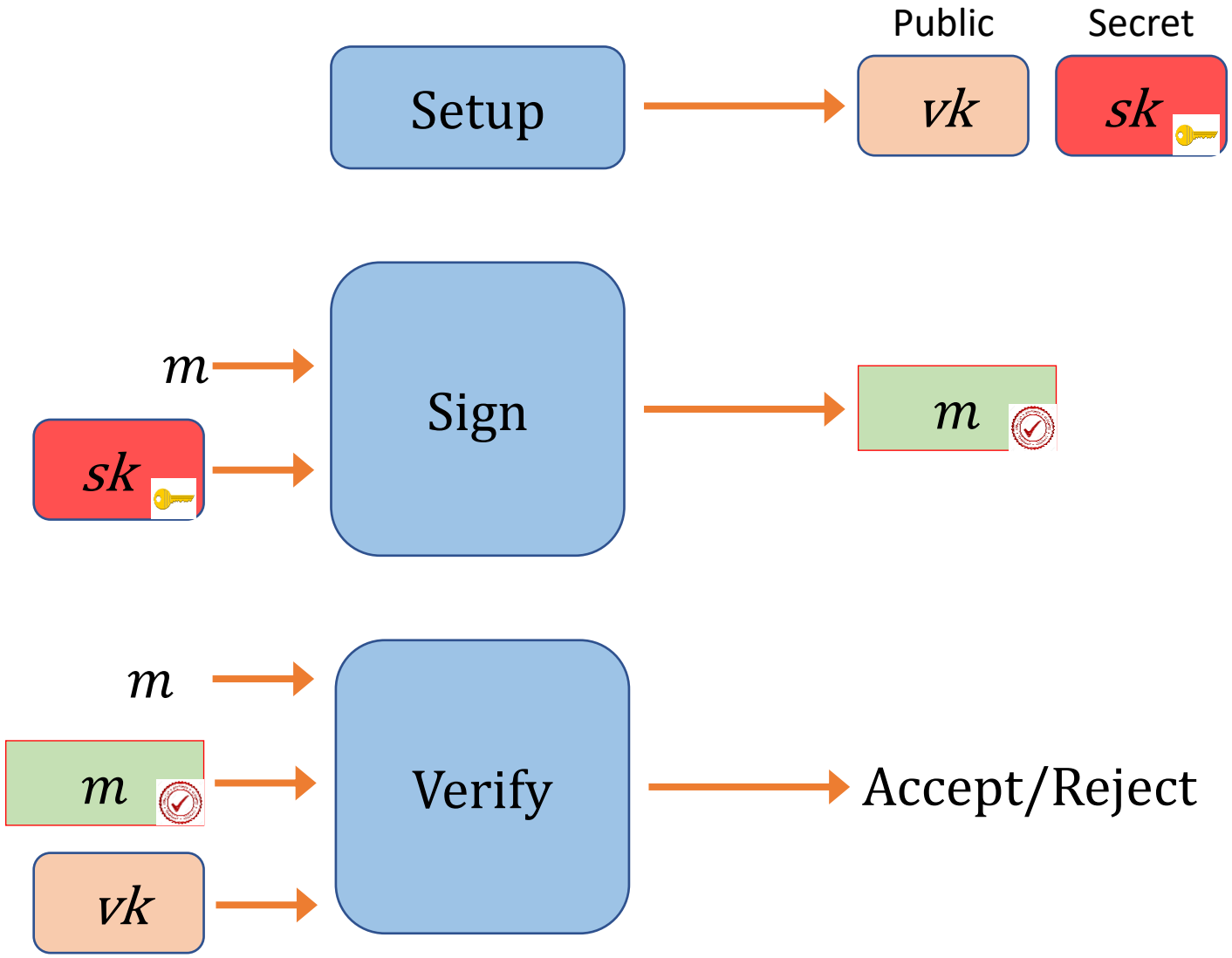
LMS, XMSS

SPHINCS+

Stateful

Use for software/firmware signing

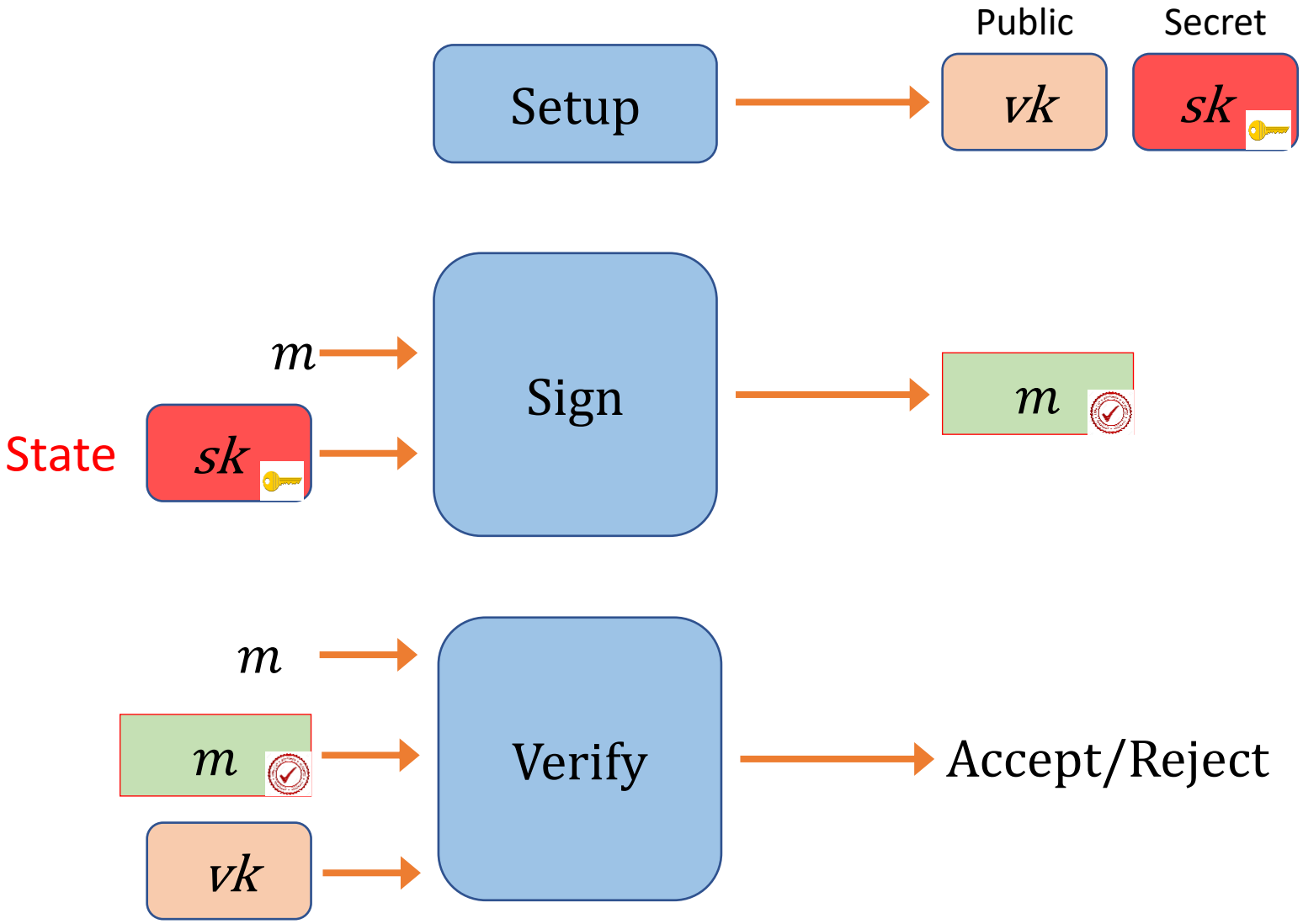
Digital Signatures



Two Properties:

1. Correctness
2. Unforgeability

Digital Signatures (Stateful)



Two Properties:

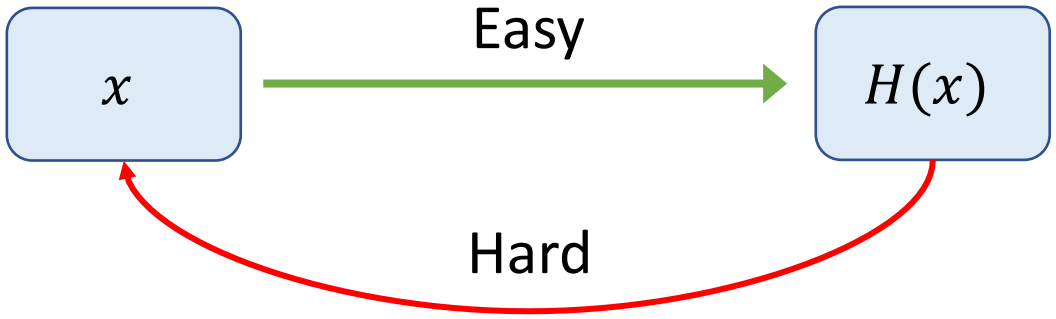
1. Correctness
2. Unforgeability

LMS and XMSS Signature Schemes

- Both very similar (focus on LMS)
- Take a one-time signature and transform to many-time signature
  - One-time signature: Only ever sign one message with any key pair
- Use Winternitz one-time signature as building block
- H is a cryptographic hash function
  - Every application uses a different prefix!

Properties of H

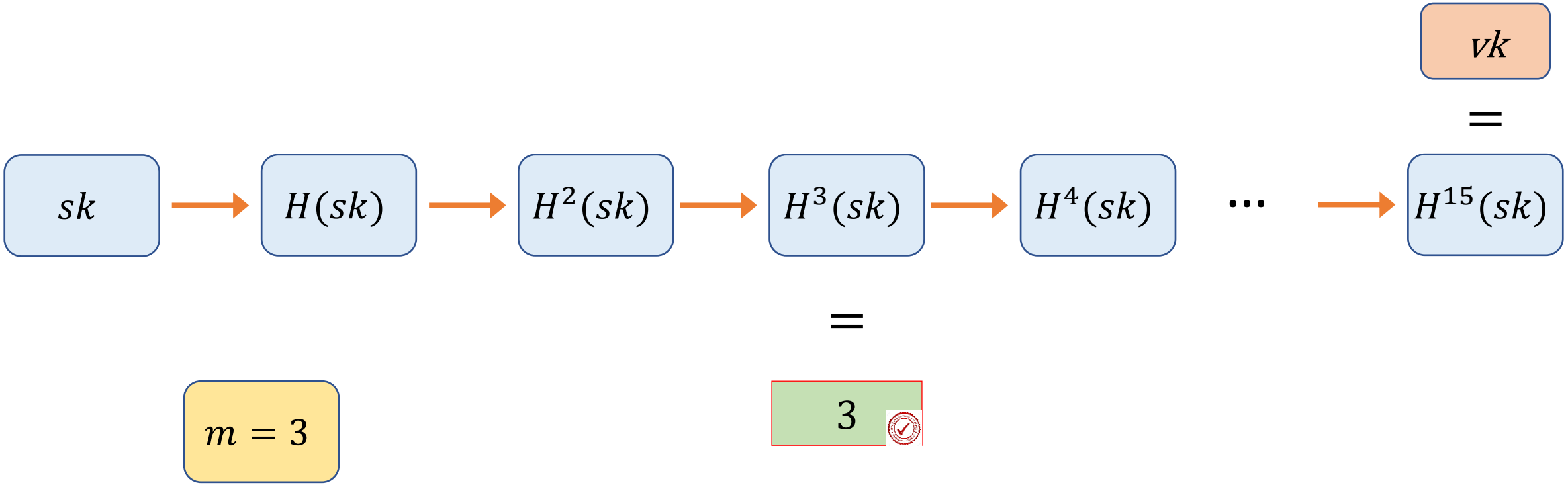
- H is "hard to invert" and output "looks random"
- H is collision-resistant
  - Cannot find $x \neq x'$ with $H(x) = H(x')$
- Can instantiate H with SHA256



Winternitz One-Time Signature Scheme

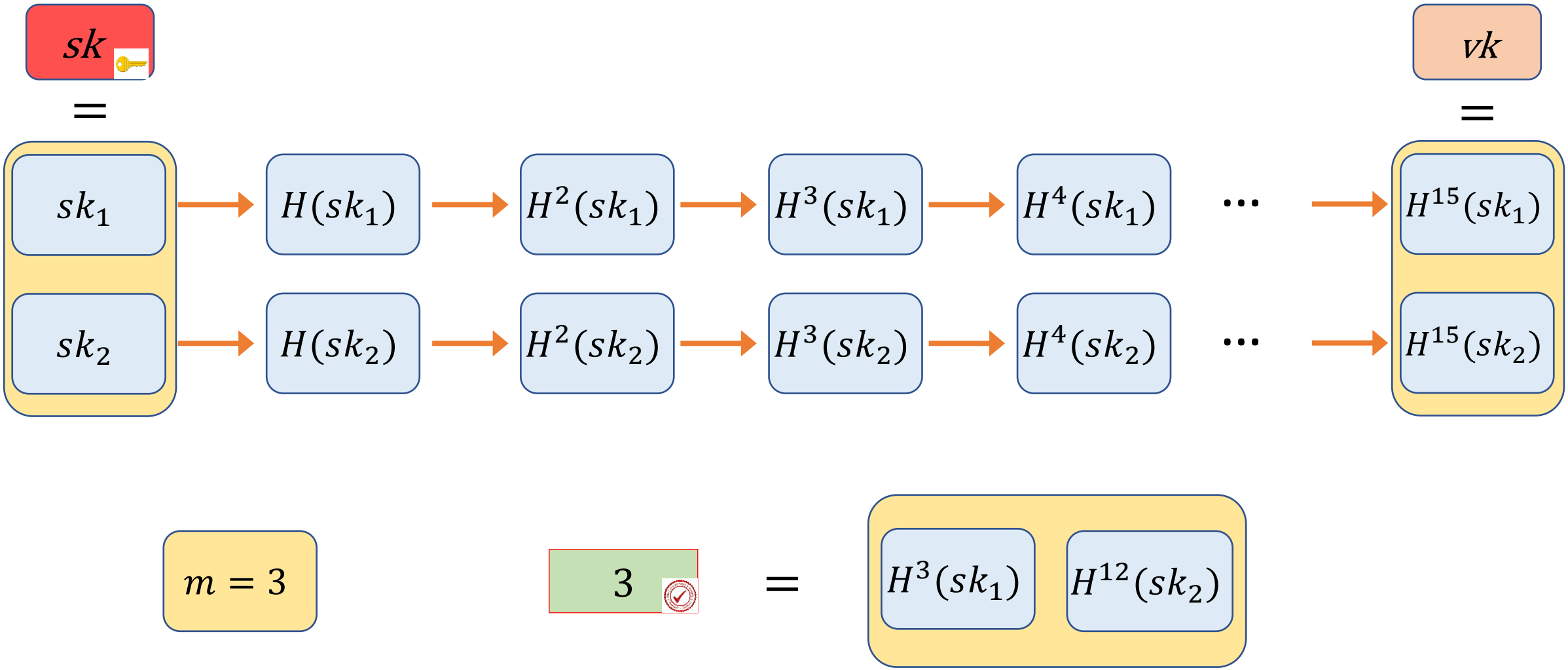
- Idea: Use hash chain to sign message
- Small message space: m, an integer in $0 - 15$
- Sample uniformly random $sk \leftarrow \{0,1\}^{256}$
- Verification key $vk = H^{15}(sk)$
- Signature $\sigma = H^{m}(sk)$

Winternitz One-Time Signature Scheme



This is insecure!

Winternitz One-Time Signature Scheme



Winternitz One-Time Signature Scheme

- Extend to arbitrary length messages

$H(m) =$

3	9	11	6

Sign each block separately!

Signature length:
$2 * NumBlocks$ hashes

Winternitz One-Time Signature Scheme

- Extend to arbitrary length messages

$H(m) =$

3	9	11	6

- Create $CheckSum = \sum_{i=1}^{4}(15 - block_i)$

$12 + 6 + 4 + 9 = 31 =$

1	15

Winternitz One-Time Signature Scheme

- Extend to arbitrary length messages

$H(m) =$ | 3 | 9 | 11 | 6 | $CheckSum =$ | 1 | 15 |

$\sigma =$ $H^3(sk_1)$ $H^9(sk_2)$ $H^{11}(sk_3)$ $H^6(sk_4)$ $H(sk_5)$ $H^{15}(sk_6)$

$sk = sk_1, sk_2, sk_3, sk_4, sk_5, sk_6$

Let $vk_i = H^{15}(sk_i)$.

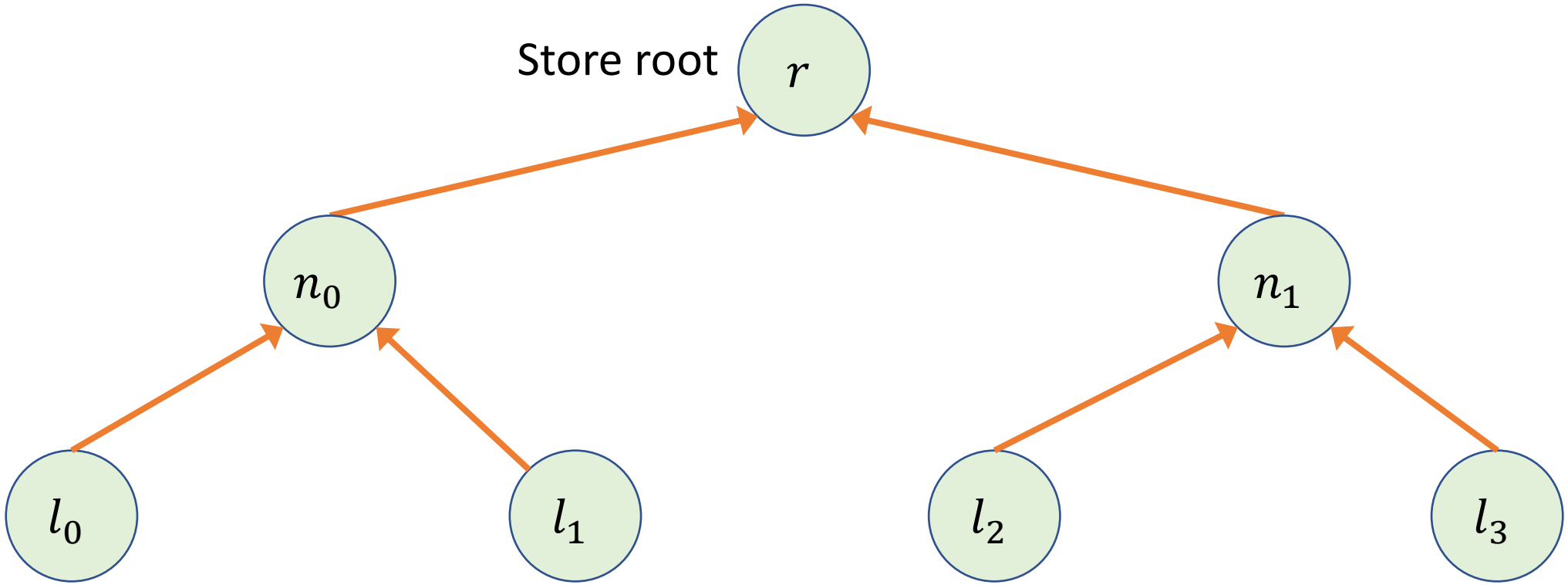
$vk = H(vk_1, vk_2, vk_3, vk_4, vk_5, vk_6)$

LMS Signature Scheme

- Winternitz can only sign one message!
- Extend to many messages
- To support N signatures, instantiate one-time signature N times
  - Generate N key pairs (vk_i, sk_i)
  - Use sk_i to sign ith message
  - Signer maintains counter state i

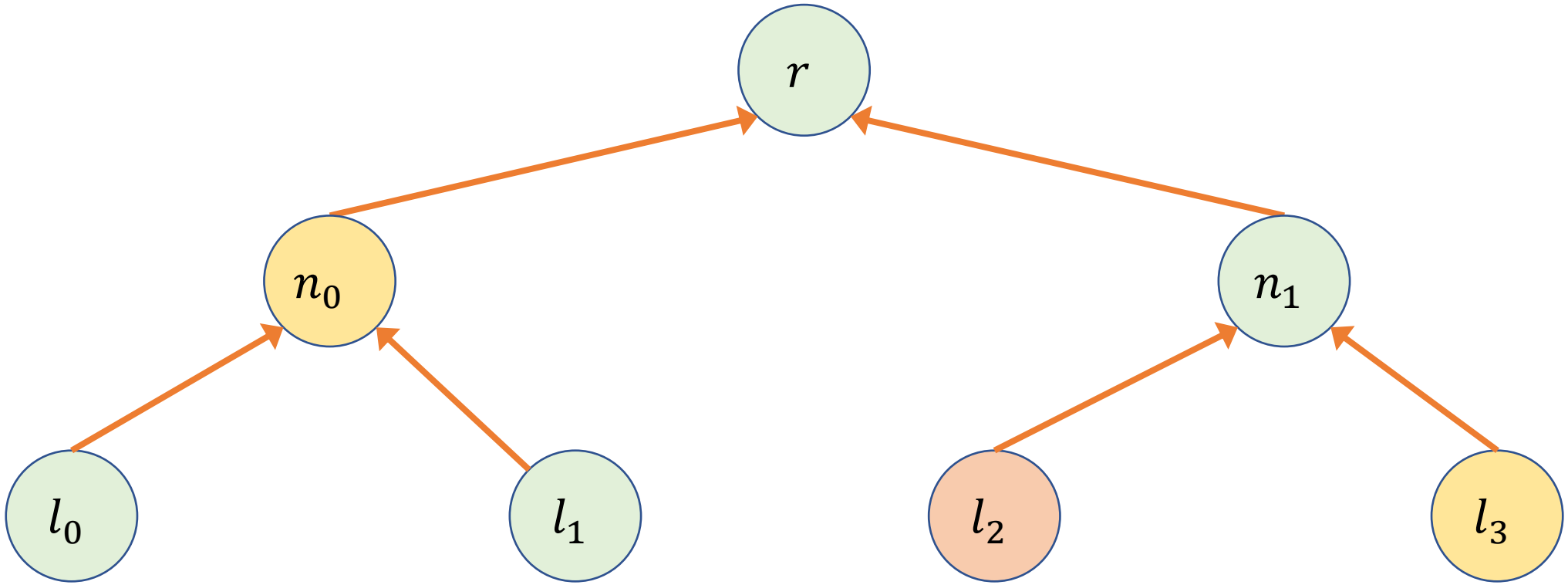
Inefficient!

Merkle Tree



$$n_0 = H(l_0, l_1)$$

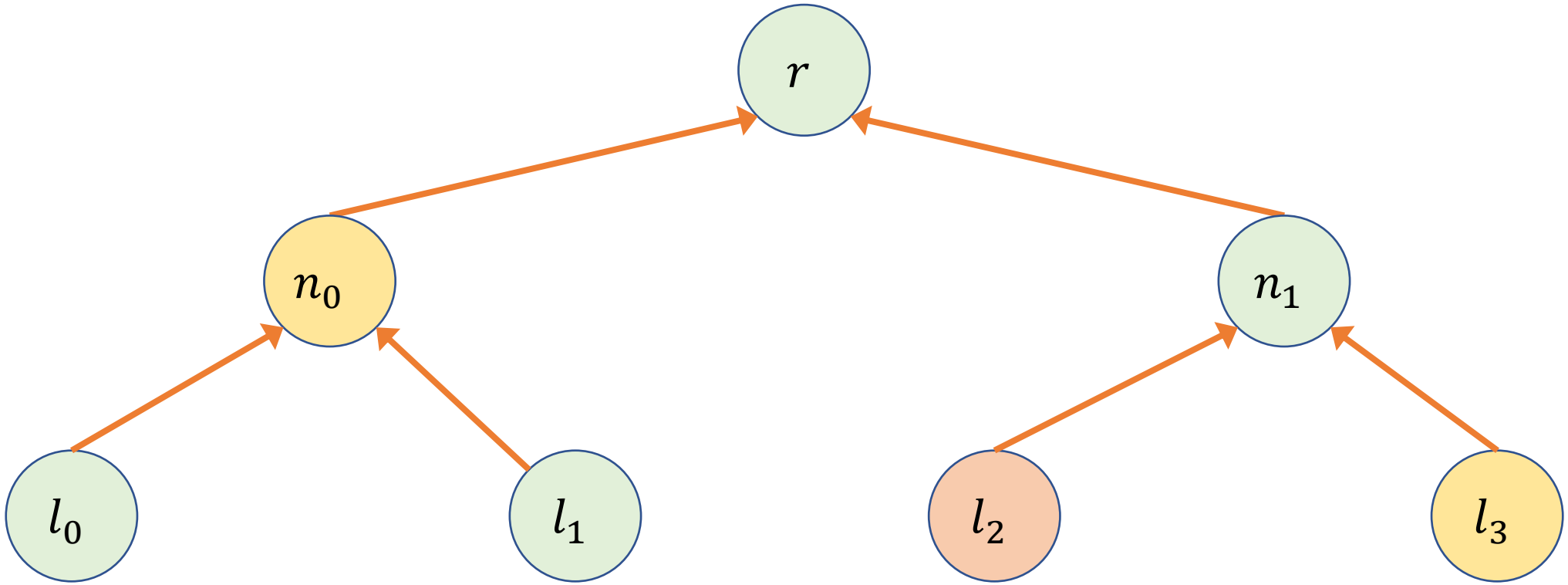
Merkle Tree



Store r

Check $H\left(n_0, H(l_2, l_3)\right) == r$

Merkle Tree



Form Merkle Tree of individual vk_i's

$$l_i = H(vk_i)$$

$$vk = r$$

LMS Signature Scheme

- Support $N = 2^h$ signatures
- N instantiations of Winternitz one-time signature (sk_i, vk_i)
- vk is Merkle tree root of Merkle tree with leaves $H(vk_i)$
- sk is $(sk_1, sk_2, \ldots, sk_N)$
- Signer maintains counter state i

LMS Signature Scheme

- Signature for m: Consists of two parts
  1. $\sigma_i = Sign(sk_i, m)$
  2. Merkle path for vk_i: $(p_0, p_1, \ldots, p_h)$

- Verification:
  1. Run Winternitz verification to get candidate vk_i'
  2. Use Merkle path with vk_i' to get candidate vk'
  3. Accept if $vk' == vk$

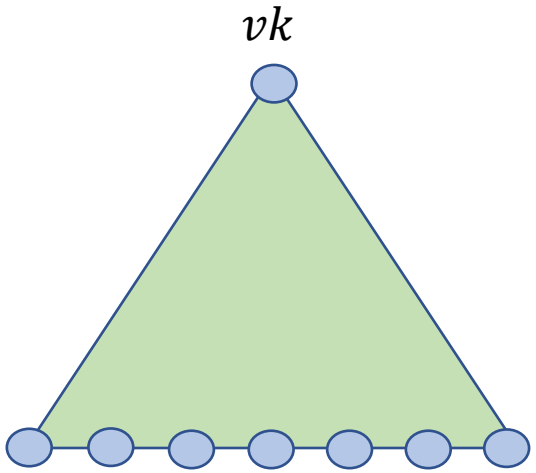
LMS vs. XMSS

- Very similar
- Main difference is how H is invoked
- LMS: $H(ctr, m)$ where ctr is a deterministic prefix
- XMSS: $H(ctr_1, m \oplus ctr_2)$ where ctr_1 and ctr_2 are pseudorandom
  - $ctr_i = H(seed, addr)$

Extensions

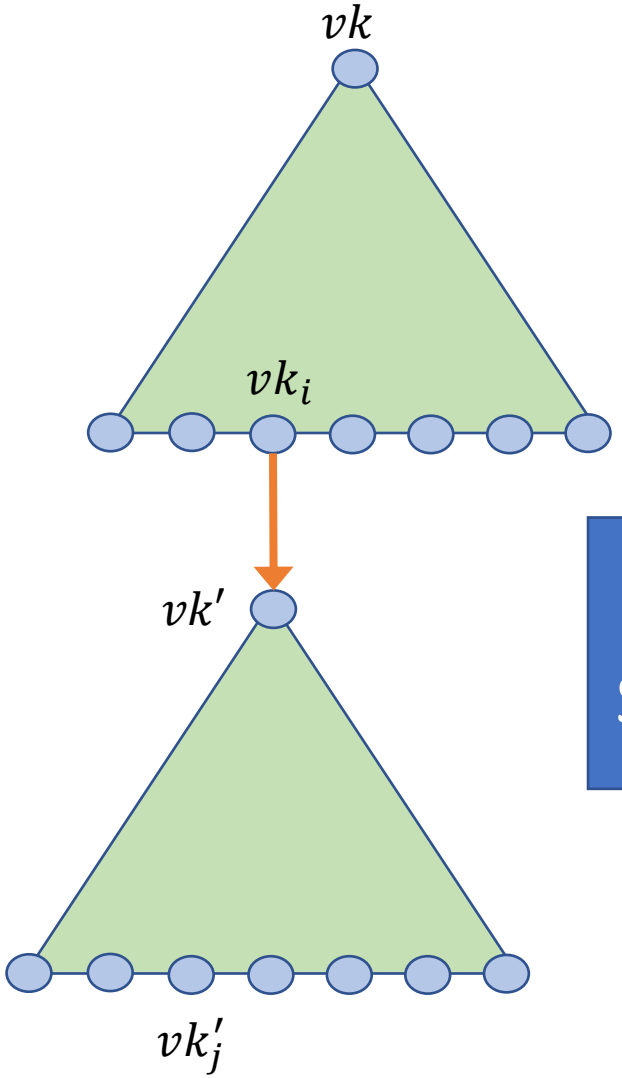
- Generating vk is inefficient for large N
  - Need to compute $N - 1$ hashes to construct Merkle tree

- Hierarchical extensions of LMS and XMSS
  - HSS and XMSS^{MT}, respectively



Extensions

Sign vk' using sk_i
Sign m using $sk_j{}'$



Can have L levels of trees
Sig. size increases by factor of L

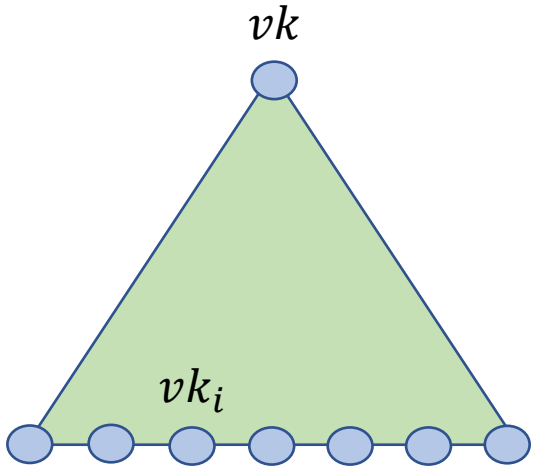
SPHINCS+

- Similar overall approach to LMS and XMSS

- Stateless
  - Used for general applications

How can we remove the state?

SPHINCS+



Sample index i to pick which sk_i to use to sign message randomly!

Need to never sample same index twice with overwhelming probability

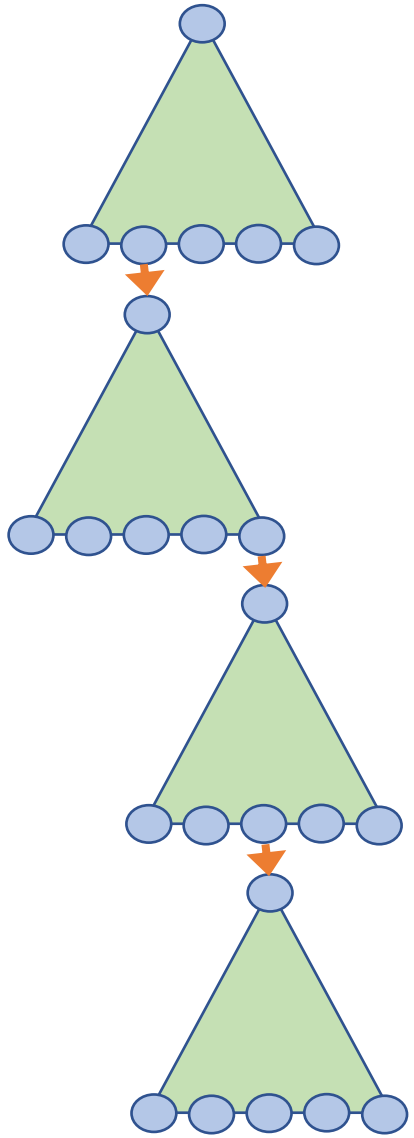
Need too many key pairs (vk_i, sk_i)

Merkle tree is too big!

SPHINCS+

Use many levels of trees

Need many levels of trees
Signature size is too long!



SPHINCS+

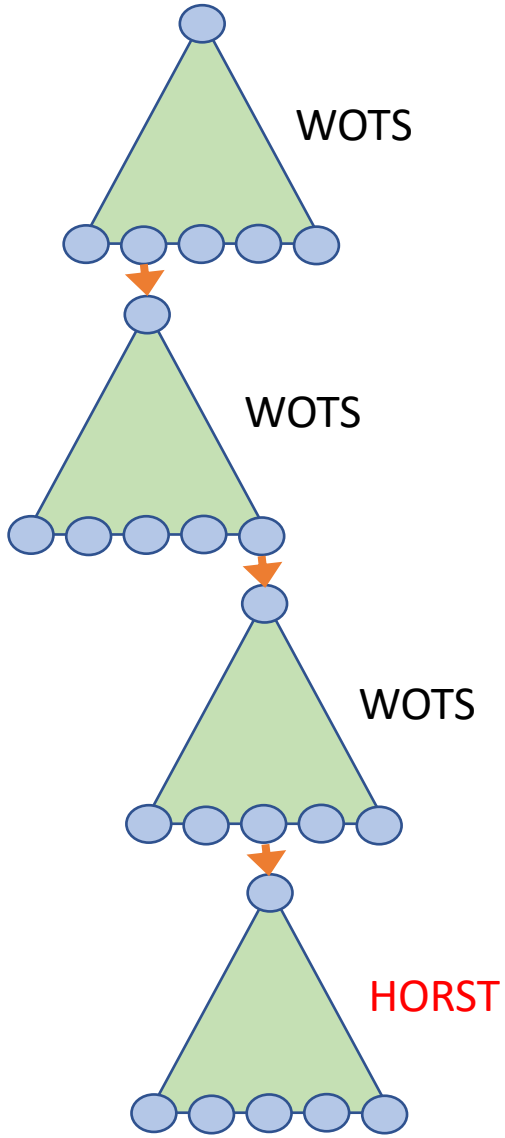
Use many levels of trees

Use few-time signature scheme for bottom tree

Can pick the same leaf index to use a few times

Greatly reduces number of levels of trees!

128 bits post-quantum security with 12 trees each of height 5



References

- Recommendation Doc: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-208.pdf
- LMS: https://www.rfc-editor.org/rfc/rfc8554.html
- XMSS: https://datatracker.ietf.org/doc/html/rfc8391
- SPHINCS+: https://sphincs.org/

Thank You!