Hash-based signatures IV

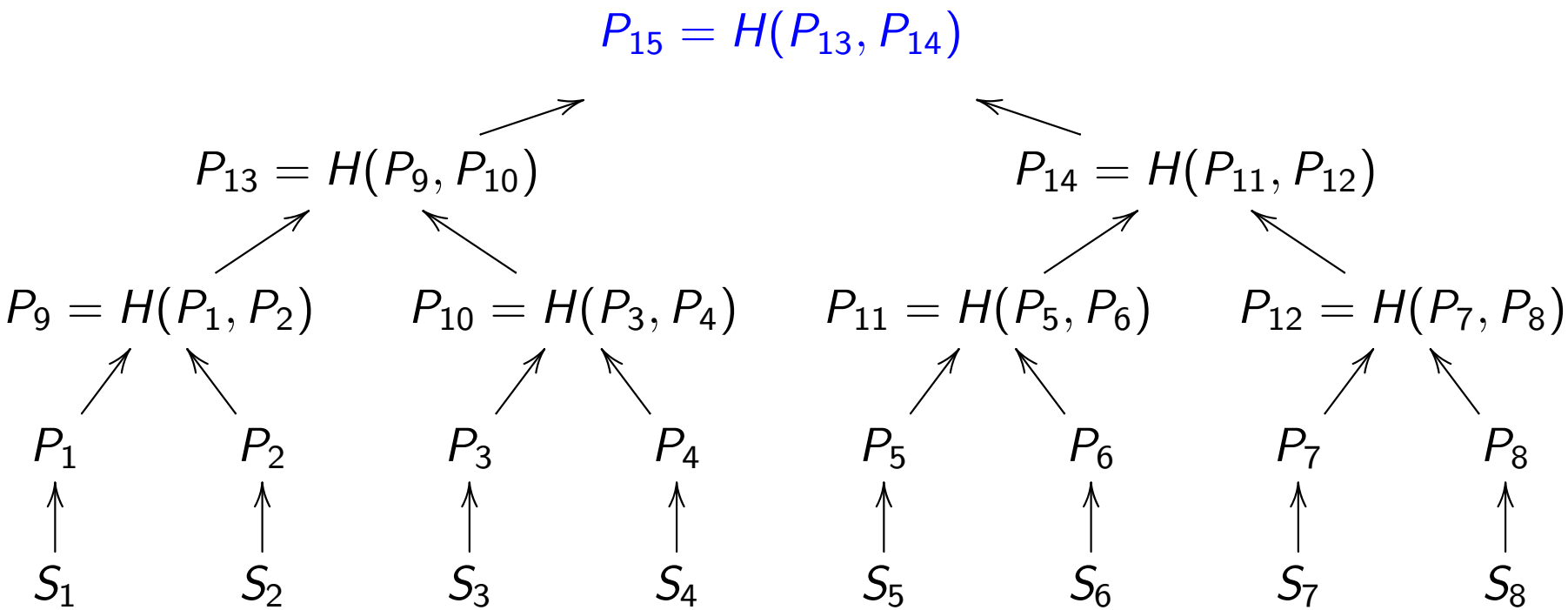
Stateless signatures

Tanja Lange
(with some slides by Daniel J. Bernstein and by Andreas Hülsing)

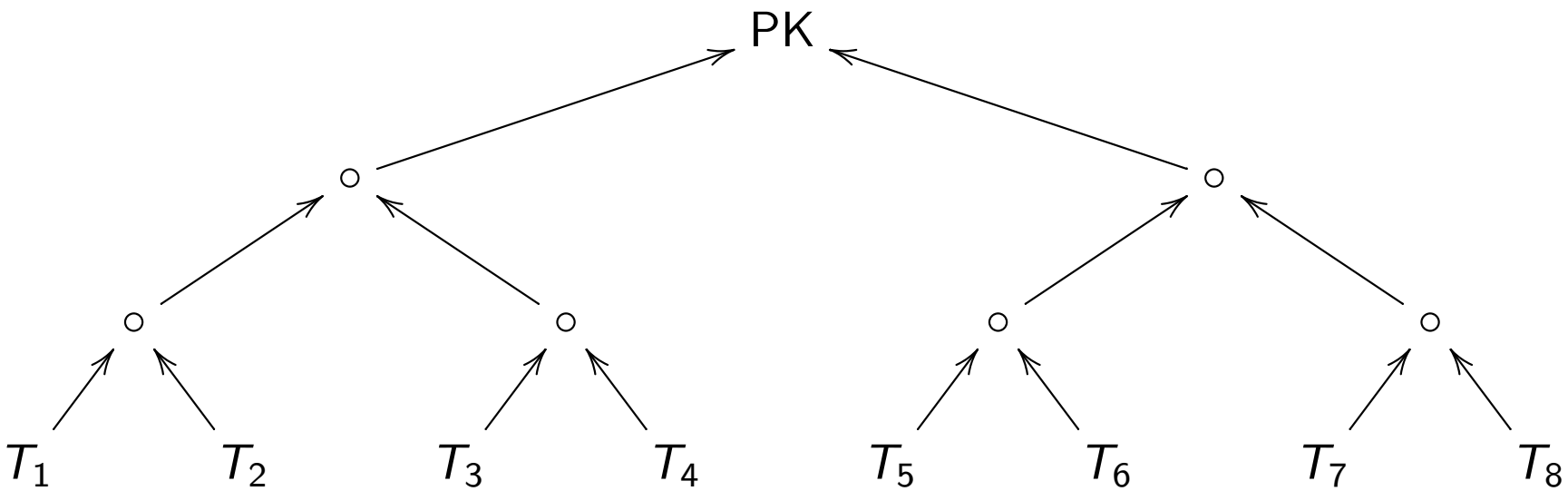
Eindhoven University of Technology

SAC – Post-quantum cryptography

Trees of Merkle trees

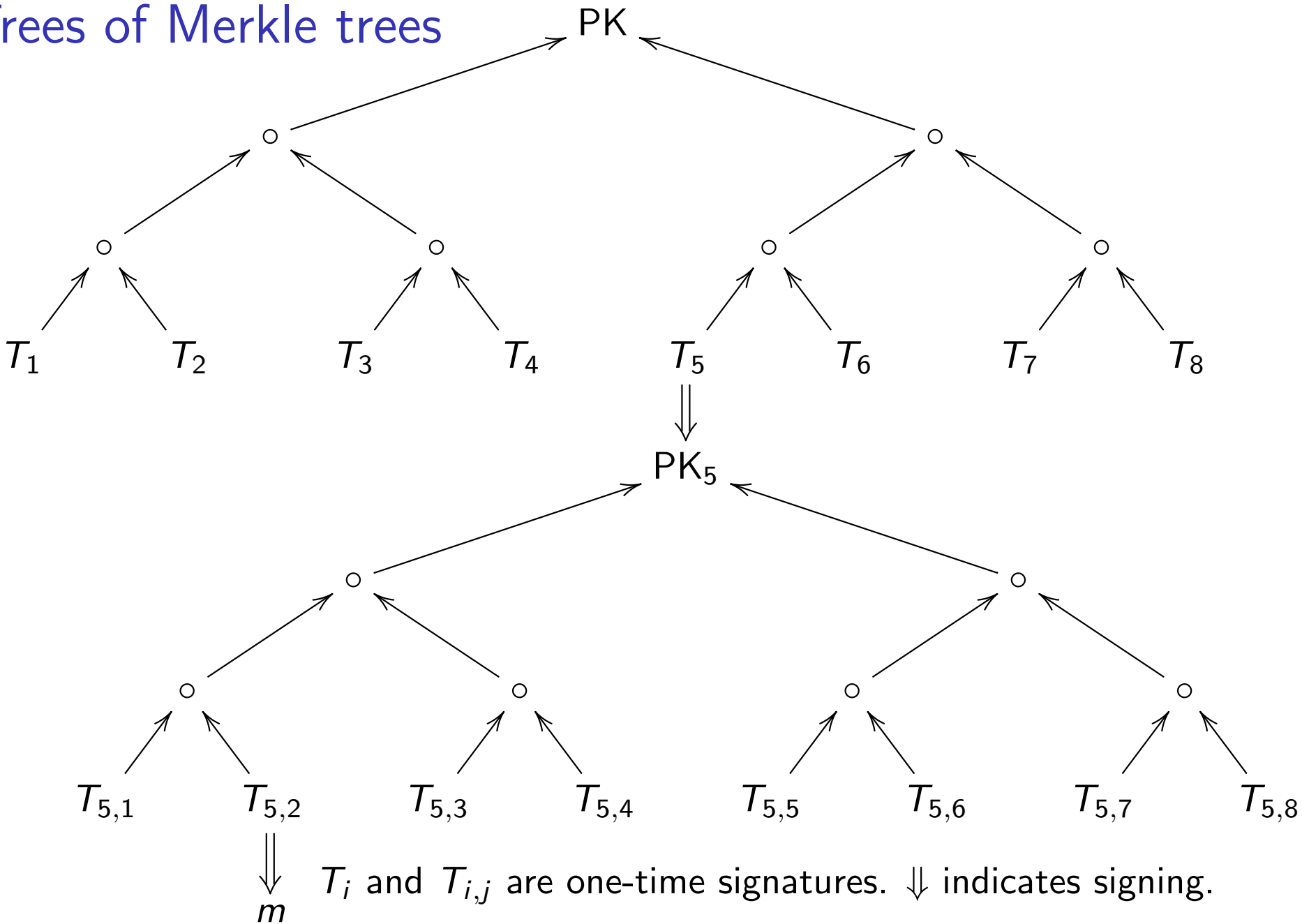


Trees of Merkle trees



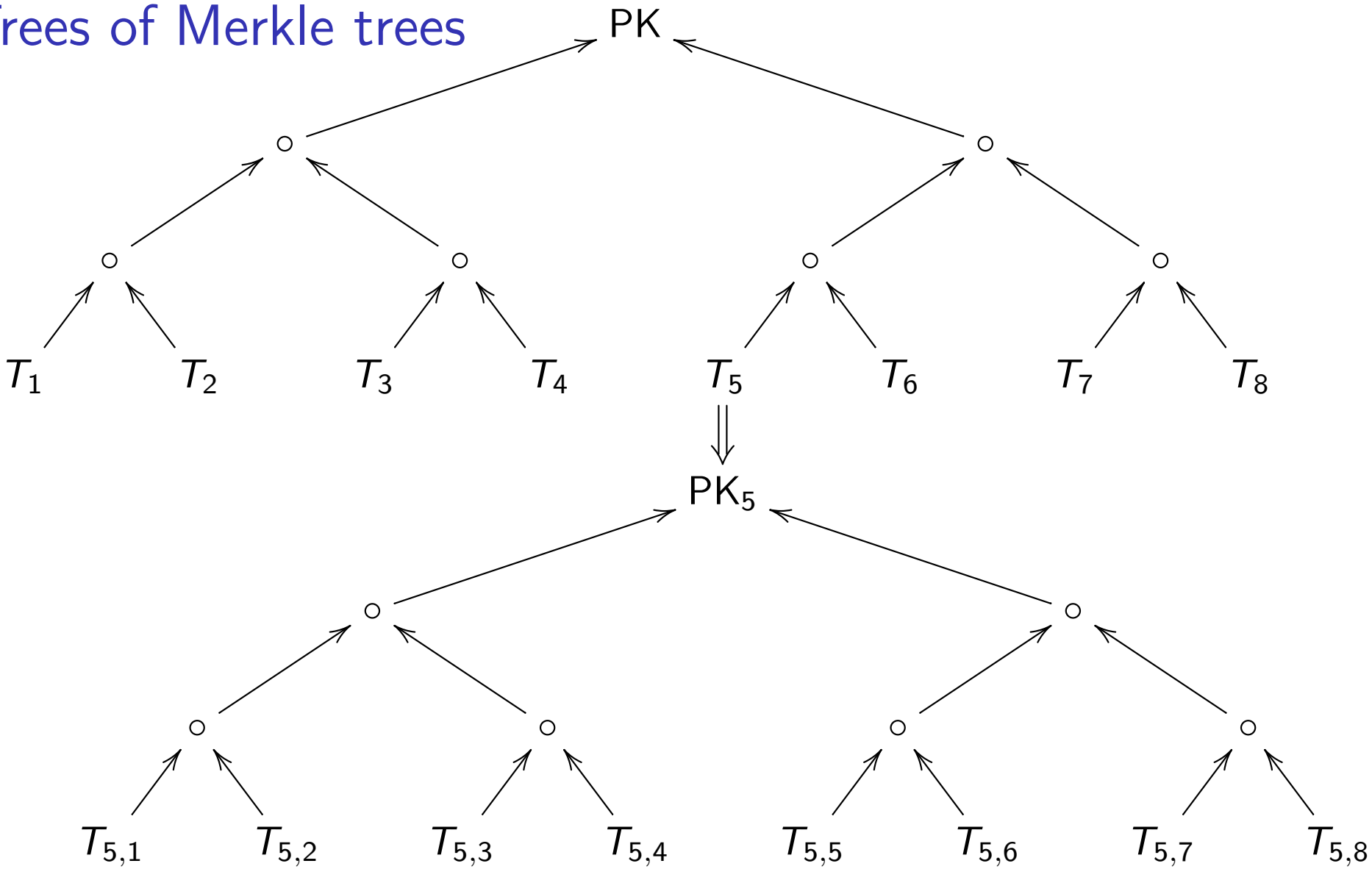
T_i are one-time signatures.
$\uparrow$ indicates input to hash function.

Trees of Merkle trees



T_i and $T_{i,j}$ are one-time signatures. $\Downarrow$ indicates signing.

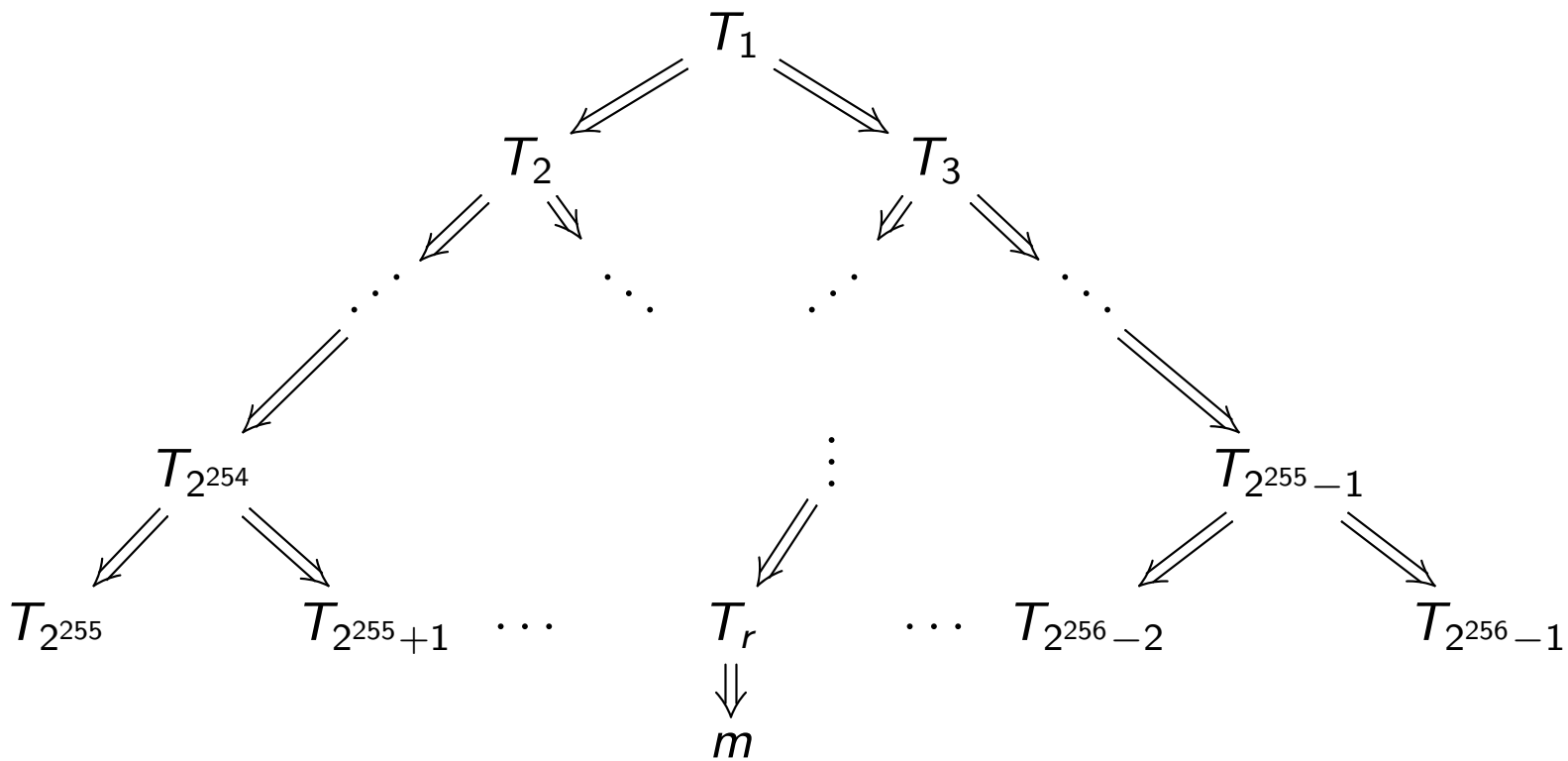
Trees of Merkle trees



T_i and $T_{i,j}$ are one-time signatures. $\Downarrow$ indicates signing.
No need to know PK_5 when generating the top tree.

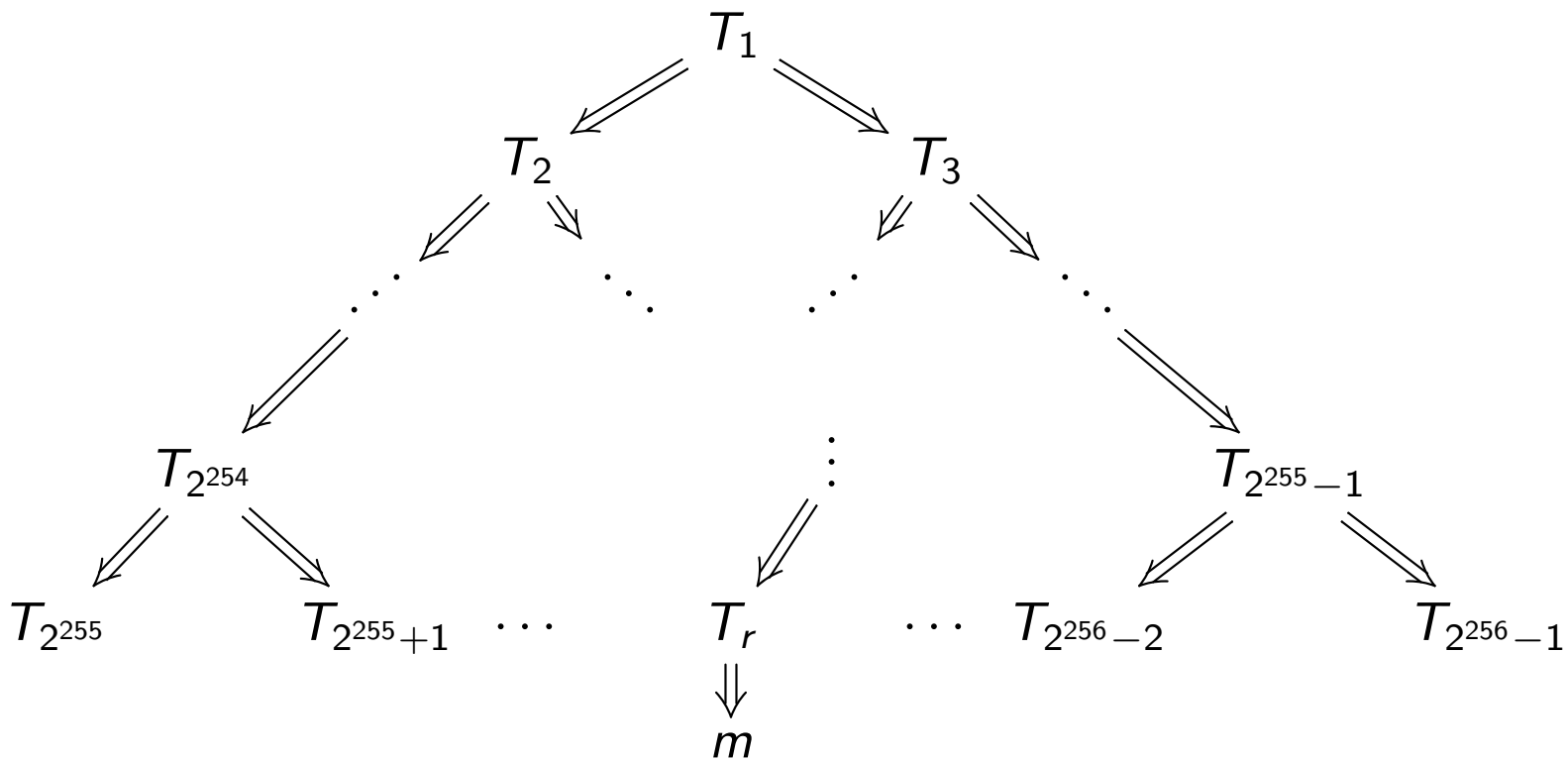
Huge trees (1987 Goldreich), keys on demand (Levin)

Signer chooses random $r \in \{2^{255}, 2^{255}+1, \ldots, 2^{256}-1\}$,
uses one-time public key T_r to sign message;
uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) for $i < 2^{255}$.
Generates ith secret key as $H_k(i)$ where k is master secret.



Huge trees (1987 Goldreich), keys on demand (Levin)

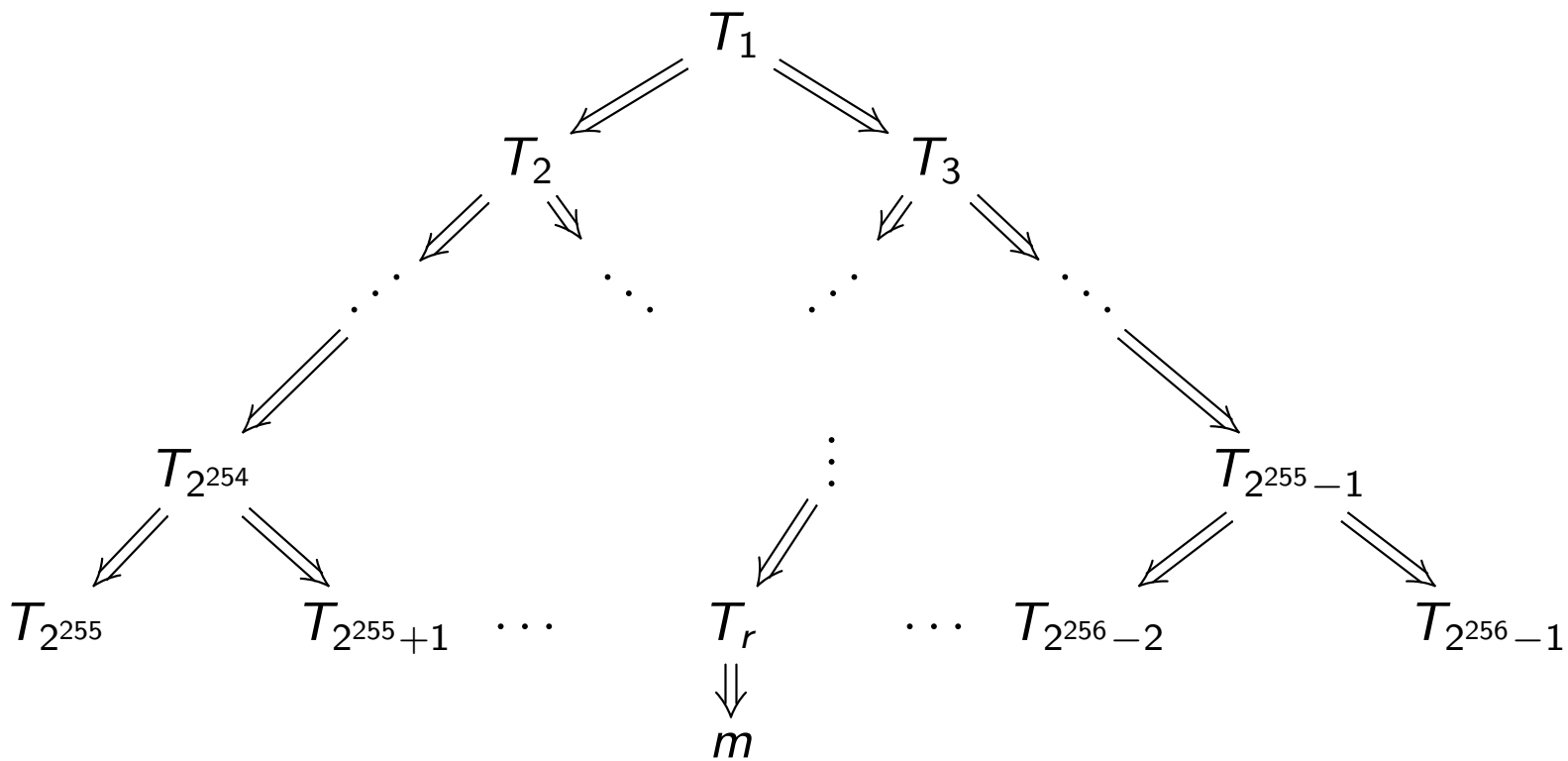
Signer chooses random $r \in \{2^{255}, 2^{255}+1, \ldots, 2^{256}-1\}$,
uses one-time public key T_r to sign message;
uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) for $i < 2^{255}$.
Generates ith secret key as $H_k(i)$ where k is master secret.



T_i for small i gets used repeatedly (each time an m falls in that sub-tree)

Huge trees (1987 Goldreich), keys on demand (Levin)

Signer chooses random $r \in \{2^{255}, 2^{255}+1, \ldots, 2^{256}-1\}$,
uses one-time public key T_r to sign message;
uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) for $i < 2^{255}$.
Generates ith secret key as $H_k(i)$ where k is master secret.



T_i for small i gets used repeatedly (each time an m falls in that sub-tree) but $H_k(i)$ being deterministic means it signs the same value, so no break.

Use Goldreich to create stateless hash-based signatures

0.6 MB for hash-based Goldreich signature using
short-public-key Winternitz-16 one-time signatures.

Would dominate traffic in typical applications,
and add user-visible latency on typical network connections.

Use Goldreich to create stateless hash-based signatures

0.6 MB for hash-based Goldreich signature using
short-public-key Winternitz-16 one-time signatures.

Would dominate traffic in typical applications,
and add user-visible latency on typical network connections.

Example:
Debian operating system is designed for frequent upgrades.
At least one new signature for each upgrade.
Typical upgrade: one package or just a few packages.
1.2 MB average package size.
0.08 MB median package size.

Use Goldreich to create stateless hash-based signatures

0.6 MB for hash-based Goldreich signature using
short-public-key Winternitz-16 one-time signatures.

Would dominate traffic in typical applications,
and add user-visible latency on typical network connections.

Example:
Debian operating system is designed for frequent upgrades.
At least one new signature for each upgrade.
Typical upgrade: one package or just a few packages.
1.2 MB average package size.
0.08 MB median package size.

Example:
HTTPS typically sends multiple signatures per page.
1.8 MB average web page in Alexa Top 1000000.

Ingredients of SPHINCS (and SPHINCS-256)

Drastically reduce tree height (to 60).

Replace one-time leaves with few-time leaves.

Optimize few-time signature size *plus* key size.
New few-time HORST, improving upon HORS
(see exercise sheet 4).

Use hyper-trees (12 layers), as in GMSS.

Use masks, as in XMSS and XMSS^{MT},
for standard-model security proofs.

Optimize short-input (256-bit) hashing speed.
Use sponge hash (with ChaCha12 permutation).

Use fast stream cipher (again ChaCha12).

Vectorize hash software and cipher software.

See paper for details: sphincs.cr.yp.to

Updated version is NIST submission SPHINCS+
https://sphincs.org/.