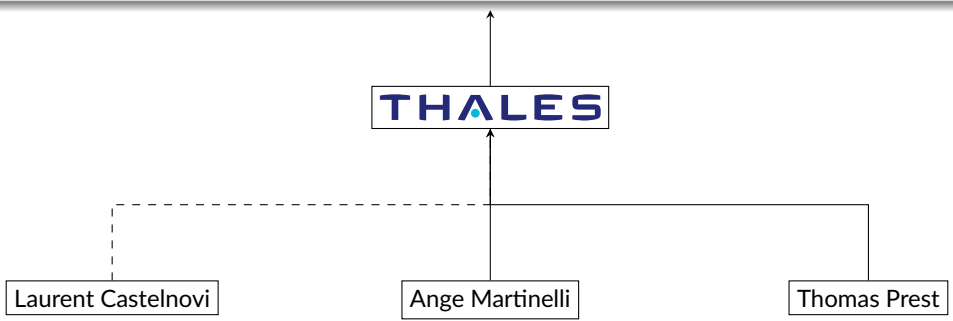
Grafting Trees: a Fault Attack against the SPHINCS framework



Introduction

Hash-based signatures:

- Signatures based on the collision or preimage resistance of hash functions
- Optimal from a security perspective [Rom90]
- Post quantum: two proposals to NIST's CFP [AE17, BDE+17]

Obvious question: do they resist to fault attacks?

- Short answer: No.
- This talk: a fault attack against schemes of the SPHINCS family:
  - The original SPHINCS [BHH+15]
  - Gravity-SPHINCS [AE17]
  - SPHINCS+ [BDE+17]

Let's fault stuff! 

Outline of this talk

One-time signatures (OTS) from hash functions

A toy example:

- $\mathsf{sk} = (s_1, s_2) \in \{0,1\}^{256\times 2}$
- $\mathsf{pk} = (p_1, p_2) = (\mathsf{H}^N(s_1), \mathsf{H}^N(s_2))$
- $\mathsf{Sign}(\mathsf{m} \in \{0, \dots, N\})$:

$$\mathsf{sig}(\mathsf{m}) = (\sigma_1, \sigma_2) = (\mathsf{H}^{\mathsf{m}}(s_1), \mathsf{H}^{N-\mathsf{m}}(s_2)) \tag{1}$$

- $\mathsf{Verify}(\mathsf{m}, \mathsf{sig})$: accept if and only if $\left(\mathsf{H}^{N-\mathsf{m}}(\sigma_1), \mathsf{H}^{\mathsf{m}}(\sigma_2)\right) = \mathsf{pk}$
- one signature $\Rightarrow$ existentially unforgeable
- two signatures $\Rightarrow$ existential forgery for a proportion $\approx \frac{|\mathsf{m}_1 - \mathsf{m}_2|}{N}$ of the messages

One-time signatures (OTS) from hash functions

A toy example:

- $\mathsf{sk} = (s_1, s_2) \in \{0,1\}^{256\times 2}$
- $\mathsf{pk} = (p_1, p_2) = (\mathsf{H}^N(s_1), \mathsf{H}^N(s_2))$
- $\mathsf{Sign}(\mathsf{m} \in \{0, \dots, N\})$:

$$\mathsf{sig}(\mathsf{m}) = (\sigma_1, \sigma_2) = (\mathsf{H}^{\mathsf{m}}(s_1), \mathsf{H}^{N-\mathsf{m}}(s_2)) \tag{1}$$

- $\mathsf{Verify}(\mathsf{m}, \mathsf{sig})$: accept if and only if $\left(\mathsf{H}^{N-\mathsf{m}}(\sigma_1), \mathsf{H}^{\mathsf{m}}(\sigma_2)\right) = \mathsf{pk}$
- one signature $\Rightarrow$ existentially unforgeable
- two signatures $\Rightarrow$ existential forgery for a proportion $\approx \frac{|\mathsf{m}_1 - \mathsf{m}_2|}{N}$ of the messages

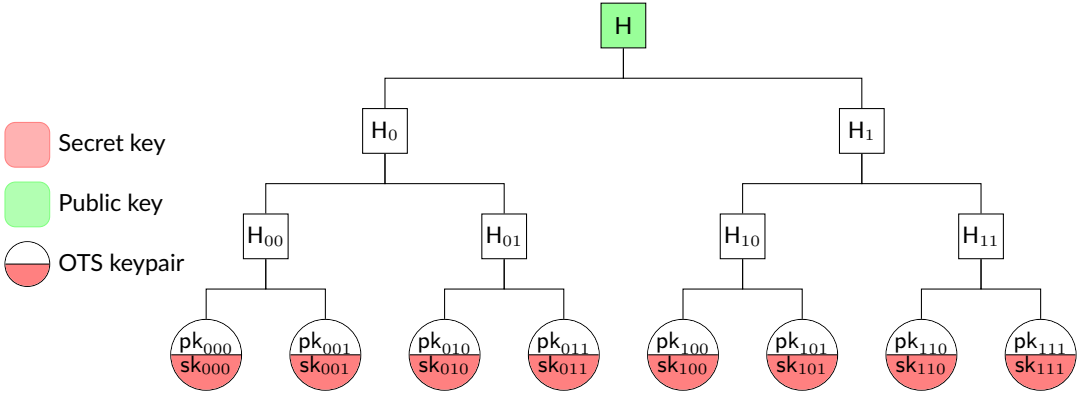
For WOTS(+), the OTS used in schemes of the SPHINCS family:

- one signature $\Rightarrow$ existentially unforgeable
- two signatures $\Rightarrow$ existential forgery for a proportion 2^{-34} of the messages

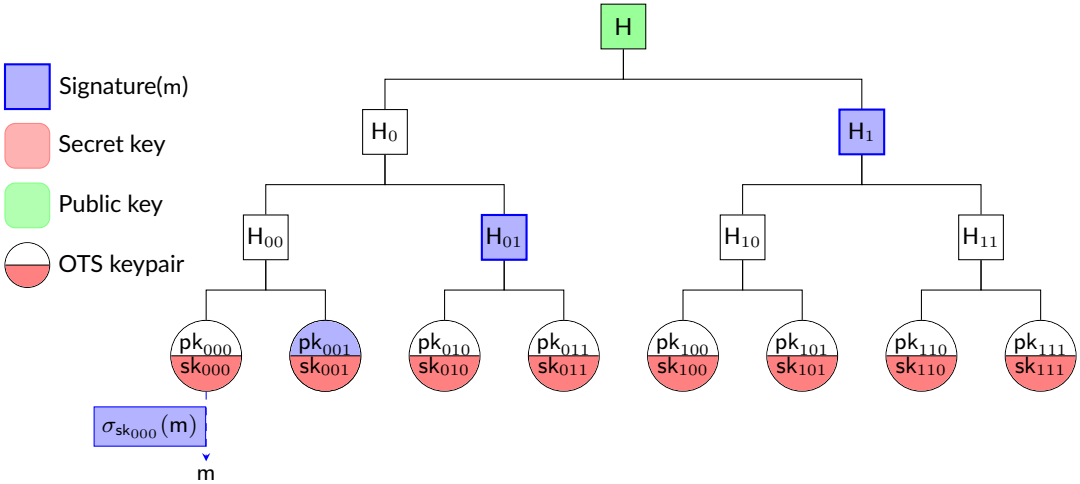
Feature common to all hash-based signatures:

From a valid signature, one can recover the public key.

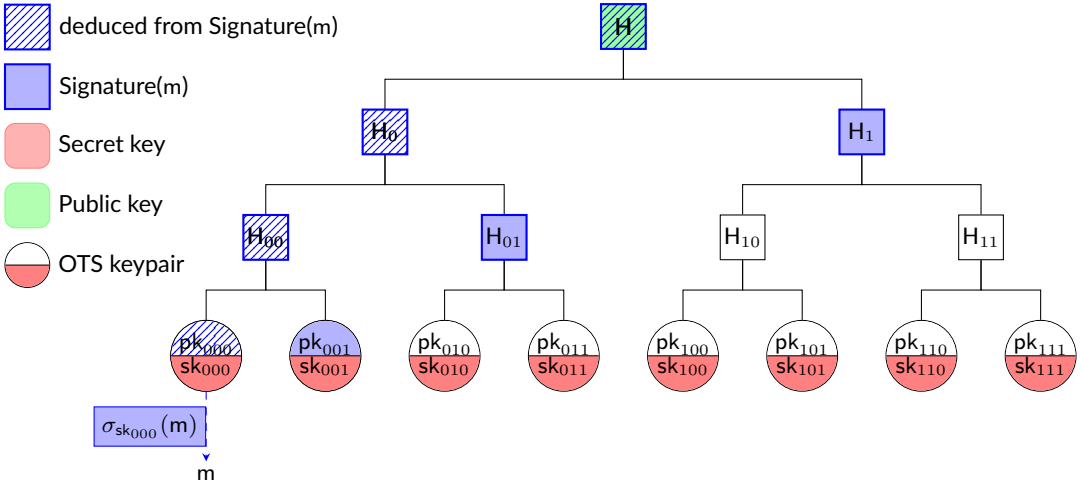
Merkle's construction [Mer90]



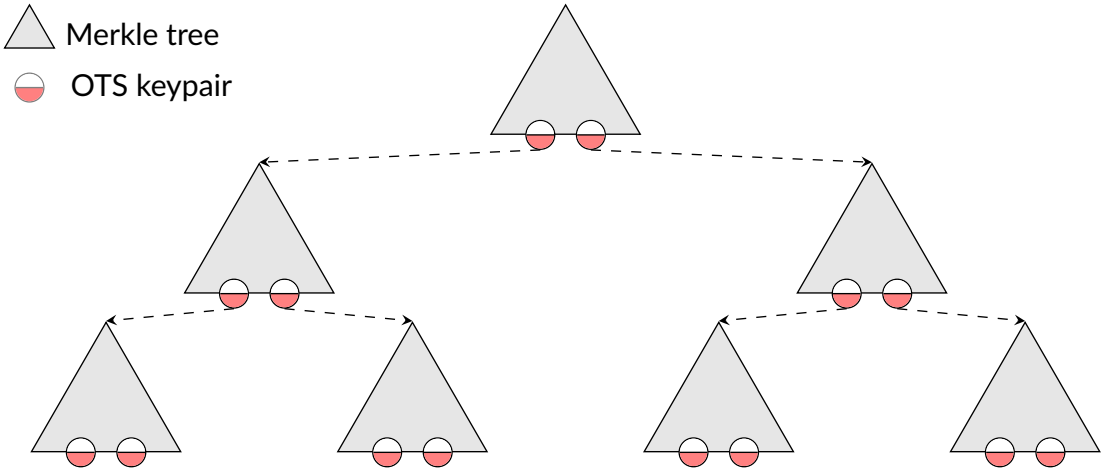
Merkle's construction [Mer90]



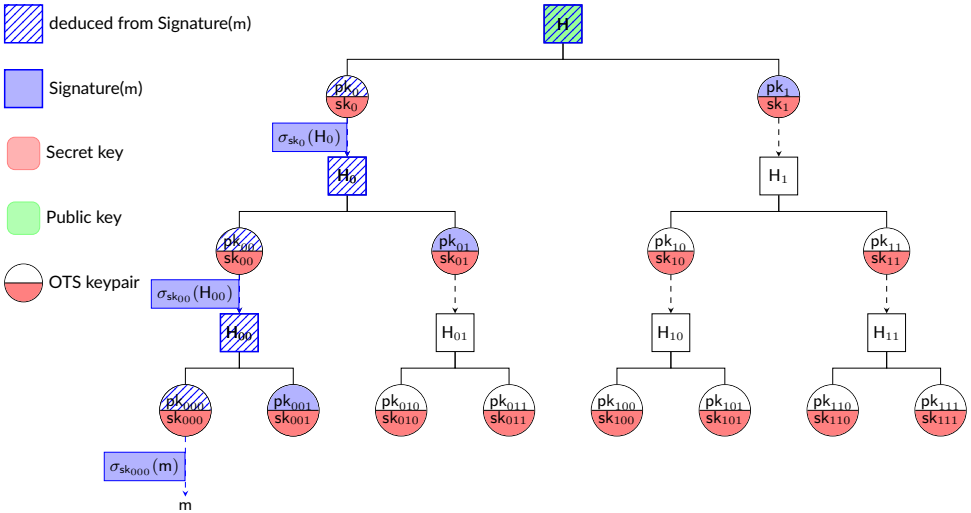
Merkle's construction [Mer90]



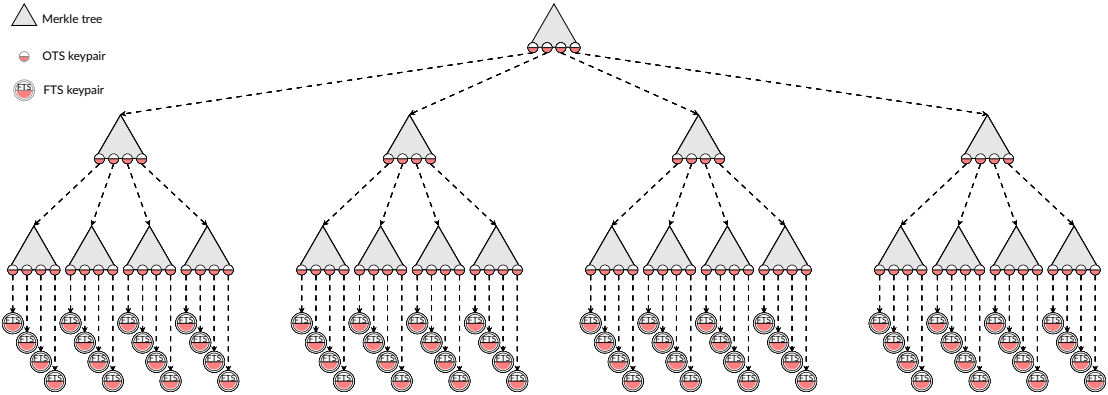
Goldreich's construction (abstract) [Gol86]



Goldreich's construction (detailed)



The SPHINCS framework



- Common to SPHINCS [BHH+15], Gravity-SPHINCS [AE17] and SPHINCS+ [BDE+17]
- Typical parameters: layers = 8, height of each Merkle tree = 8, total height = 64

Outline of the attack

Observations useful for our attack:

- In all hash-based signatures:

$$[\text{a valid signature } \sigma_{\mathsf{sk}}(\mathsf{m})] \Rightarrow [\text{one can recover } \mathsf{pk}]$$

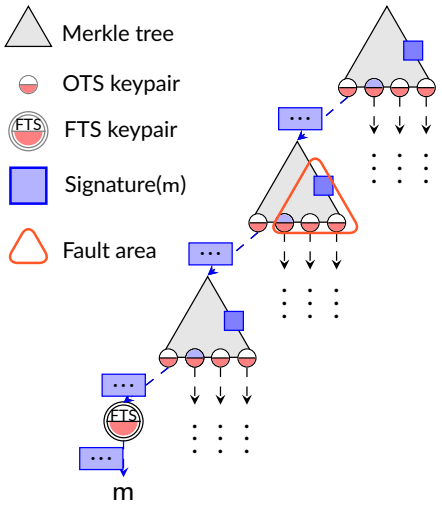
- For the OTS used in SPHINCS:

$$[2 \text{ signatures}] \Rightarrow [\text{one can forge for 1 message over } 2^{34}]$$

Outline of our attack:

1. *Faulting step.* We provoke a fault to make an OTS sign two different values
2. *Grafting step.* We use the compromised OTS to obtain an universal forgery

The faulting step



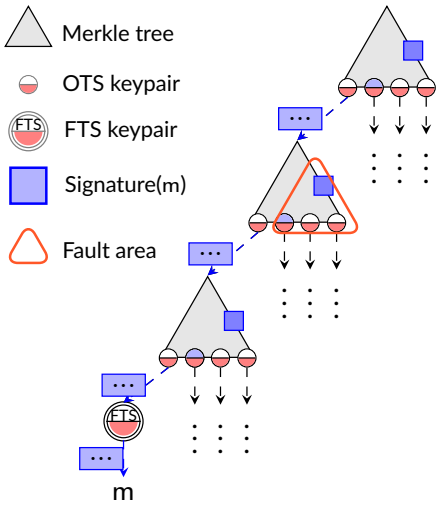
The faulting step:

- One normal sig(m), one faulted sig(m)
- Target the Merkle tree just below the top
- We may fault any computation ”below” the authentication path

Regular vs faulted signature:

- Two $\neq$ values are computed for the root of the faulted Merkle tree
- The top OTS signs two $\neq$ values

The faulting step



The faulting step:

- One normal sig(m), one faulted sig(m)
- Target the Merkle tree just below the top
- We may fault any computation "below" the authentication path

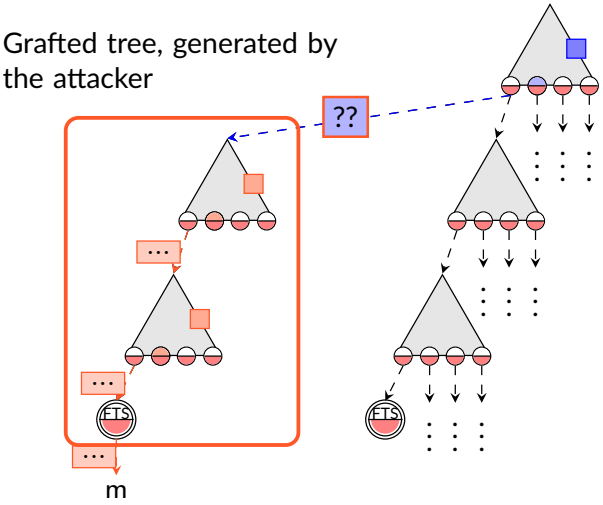
Regular vs faulted signature:

- Two $\neq$ values are computed for the root of the faulted Merkle tree
- The top OTS signs two $\neq$ values

Features of this fault:

- One fault
- Little precision required
- Stealthy

The grafting step



Goal of the attacker:

- Sign his own tree with the compromised OTS

Naïve approach:

- Generate trees until a suitable one is found
- Time: $2^{34}\times$ (generate a tree)

Adaptive approach:

- Only modify the top of the grafted tree
- Time: $2^{34}+$ (generate a tree)

Specifics of each scheme and countermeasures

Selection of the FTS index:

1. SPHINCS: $\mathsf{idx} \leftarrow \mathsf{H}(r, \mathsf{m})$, where r is private
   $\Rightarrow$ very easy
2. Gravity-SPHINCS: $\mathsf{idx} \leftarrow \mathsf{H}(r, \mathsf{m})$, where $r \leftarrow \mathsf{H}(\mathsf{sk}, \mathsf{m})$
   $\Rightarrow$ easy
3. SPHINCS$^+$: $\mathsf{idx} \leftarrow \mathsf{H}(r, \mathsf{pk}, \mathsf{m})$, where $r \leftarrow \mathsf{H}(\mathsf{sk}, \$, \mathsf{m})$
   $\Rightarrow$ no control on the FTS index anymore, but still easy

Height of the top Merkle tree:

1. SPHINCS and SPHINCS$^+$: no more than 8
2. Gravity-SPHINCS: 20

Countermeasures:

1. Generic: redundancy
2. Specific: ?

Conclusion

Key takeaways:

1. A fault attack on schemes of the SPHINCS family
2. Universal forgery with one fault
3. Fault model is very weak:
   1. little to no control on the time of the fault
   2. little to no control on the precision of the fault
   3. independent of underlying hash function(s)
4. Stealthy
5. Specific countermeasures are ineffective (to our knowledge)

Conclusion

Key takeaways:

1. A fault attack on schemes of the SPHINCS family
2. Universal forgery with one fault
3. Fault model is very weak:
   1. little to no control on the time of the fault
   2. little to no control on the precision of the fault
   3. independent of underlying hash function(s)
4. Stealthy
5. Specific countermeasures are ineffective (to our knowledge)

Related works:

- This work was based on Laurent Castelnovi's Master thesis [Cas17]
- Independently studied by Genêt [Gen17] and Kannwischer [Kan17]

https://eprint.iacr.org/2018/102

Thanks!


Jean-Philippe Aumasson and Guillaume Endignoux.
Improving stateless hash-based signatures.
Cryptology ePrint Archive, Report 2017/933, 2017.
https://eprint.iacr.org/2017/933.


Daniel J. Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Andreas Hülsing, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, and Peter Schwabe.
SPHINCS+, 2017.
https://sphincs.org/.


Daniel J. Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O'Hearn.
SPHINCS: practical stateless hash-based signatures.
In *EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 368–397. Springer, 2015.


Laurent Castelnovi.
Sécurité physique de schémas cryptographiques post-quantiques.
Master thesis, 2017.
Available at https://tprest.github.io/Publications/rapport-laurent-castelnovi.pdf.


Aymeric Genêt.
Hardware attacks against hash-based cryptographic algorithms.

Master thesis, 2017.
Available at https://infoscience.epfl.ch/record/253317.


Oded Goldreich.
Two remarks concerning the Goldwasser-Micali-Rivest signature scheme.
In *CRYPTO '86*, volume 263 of *LNCS*, pages 104–110. Springer, 1986.


Matthias Kannwischer.
Physical attack vulnerability of hash-based signature schemes.
Master thesis, 2017.
Available at https://www.cdc.informatik.tu-darmstadt.de/fileadmin/user_upload/Group_CDC/Documents/theses/Matthias_Kannwischer.master.pdf.


Ralph C. Merkle.
A certified digital signature.
In *CRYPTO' 89*, volume 435 of *LNCS*, pages 218–238. Springer, 1990.


John Rompel.
One-way functions are necessary and sufficient for secure signatures.
In *STOC*, pages 387–394. ACM, 1990.