Quantencomputer – der Angriff aus der Zukunft auf unsere Daten von heute

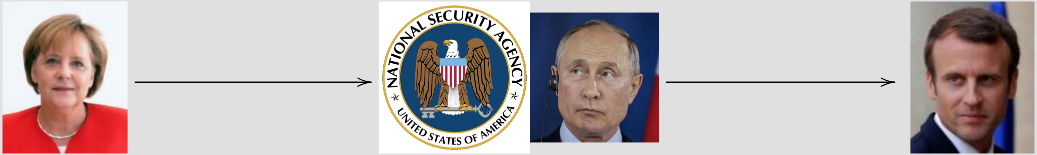
Tanja Lange

Eindhoven University of Technology

18 November 2024

Kryptographie

Kryptographie

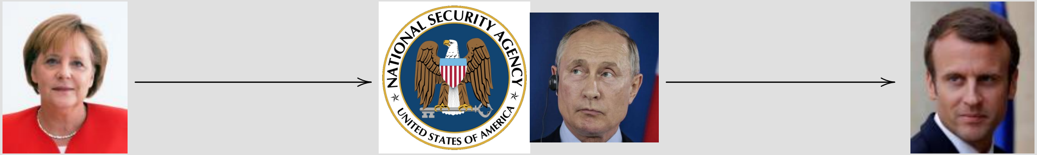


Absender
“Alice”

abgehörtes Netz
“Eve”

Empfänger
“Bob”

Kryptographie



Absender
“Alice”

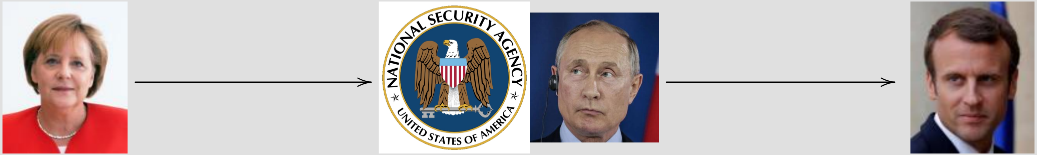
abgehörtes Netz
“Eve”

Empfänger
“Bob”

- Kreditkarten, EC-Karten, TANs, PINs
- ePässe, Perso mit Chip
- Online Einkäufe, Webseiten mit https

- Facebook, Gmail, WhatsApp, iMessage
- Festplattenverschlüsselung (iPhone, Bitlocker; siehe auch Apple vs. FBI)

Kryptographie



Absender
“Alice”

abgehörtes Netz
“Eve”

Empfänger
“Bob”

- Kreditkarten, EC-Karten, TANs, PINs
- ePässe, Perso mit Chip
- Online Einkäufe, Webseiten mit https
- Verschlüsselte Emails (PGP)
- Signal, Torbrowser
- Facebook, Gmail, WhatsApp, iMessage
- Festplattenverschlüsselung (iPhone, Bitlocker; siehe auch Apple vs. FBI)
- Tails, Qubes OS
- Extra Schritte um Privatsphäre und Sicherheit zu schützen

Kryptographische Software

Kryptographische Software

. . . und kann man der Hardware vertrauen?

Algorithms for Quantum Computation: Discrete Logarithms and Factoring

Peter W. Shor
AT&T Bell Labs
Room 2D-149
600 Mountain Ave.
Murray Hill, NJ 07974, USA

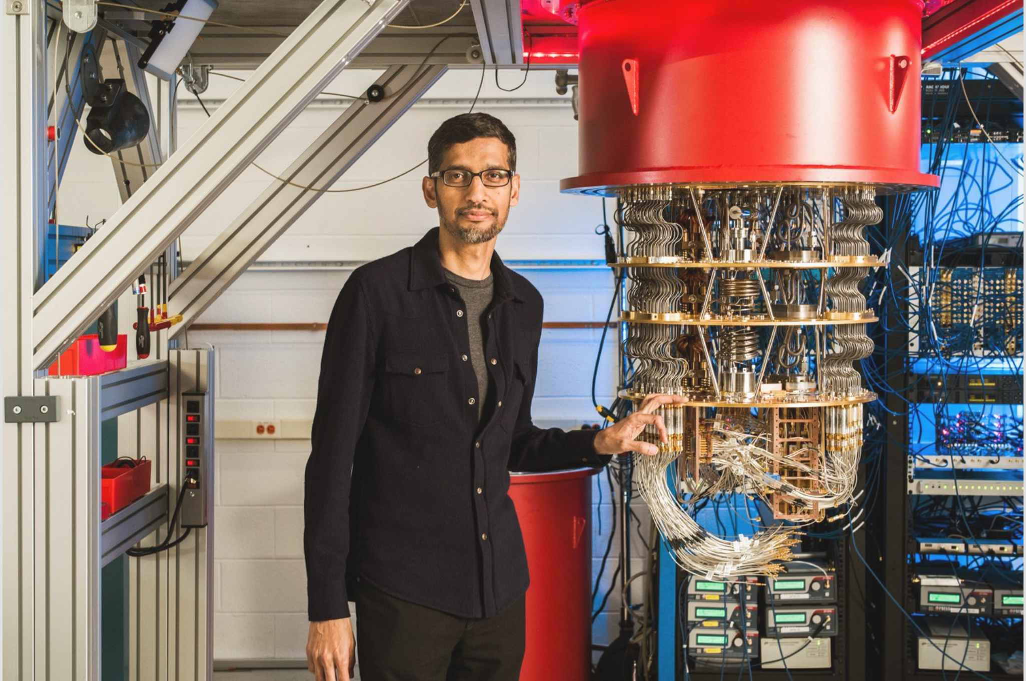
Abstract

A computer is generally considered to be a universal computational device; i.e., it is believed able to simulate any physical computational device with a cost in computation time of at most a polynomial factor. It is not clear whether this is still true when quantum mechanics is taken into consideration. Several researchers, starting with David Deutsch, have developed models for quantum

[1, 2]. Although he did not ask whether quantum mechanics conferred extra power to computation, he did show that a Turing machine could be simulated by the reversible unitary evolution of a quantum process, which is a necessary prerequisite for quantum computation. Deutsch [9, 10] was the first to give an explicit model of quantum computation. He defined both quantum Turing machines and quantum circuits and investigated some of their properties.

The next part of this paper discusses how quantum com-

IBM Q
NITROGEN
HELIUM



Weit verbreitete Systeme

Absender
“Alice”

abgehörtes Netz
“Eve”

Empfänger
“Bob”

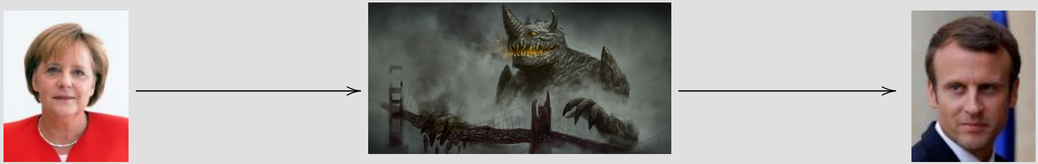
Kryptographie mit symmetrischen Schlüsseln

AES-128. AES-192. AES-256. AES-GCM. ChaCha20. HMAC-SHA-256. Poly1305. SHA-2. SHA-3. Salsa20.

Kryptographie mit öffentlichen Schlüsseln

BN-254. Curve25519. DH. DSA. ECDH. ECDSA. EdDSA. NIST P-256. NIST P-384. NIST P-521. RSA encrypt. RSA sign. secp256k1.

Weit verbreitete Systeme



Absender
“Alice”

abgehörtes Netz
“Eve” mit Quantencomputer

Empfänger
“Bob”

Kryptographie mit symmetrischen Schlüsseln

AES-128. AES-192. AES-256. AES-GCM. ChaCha20. HMAC-SHA-256. Poly1305. SHA-2. SHA-3. Salsa20.

Kryptographie mit öffentlichen Schlüsseln

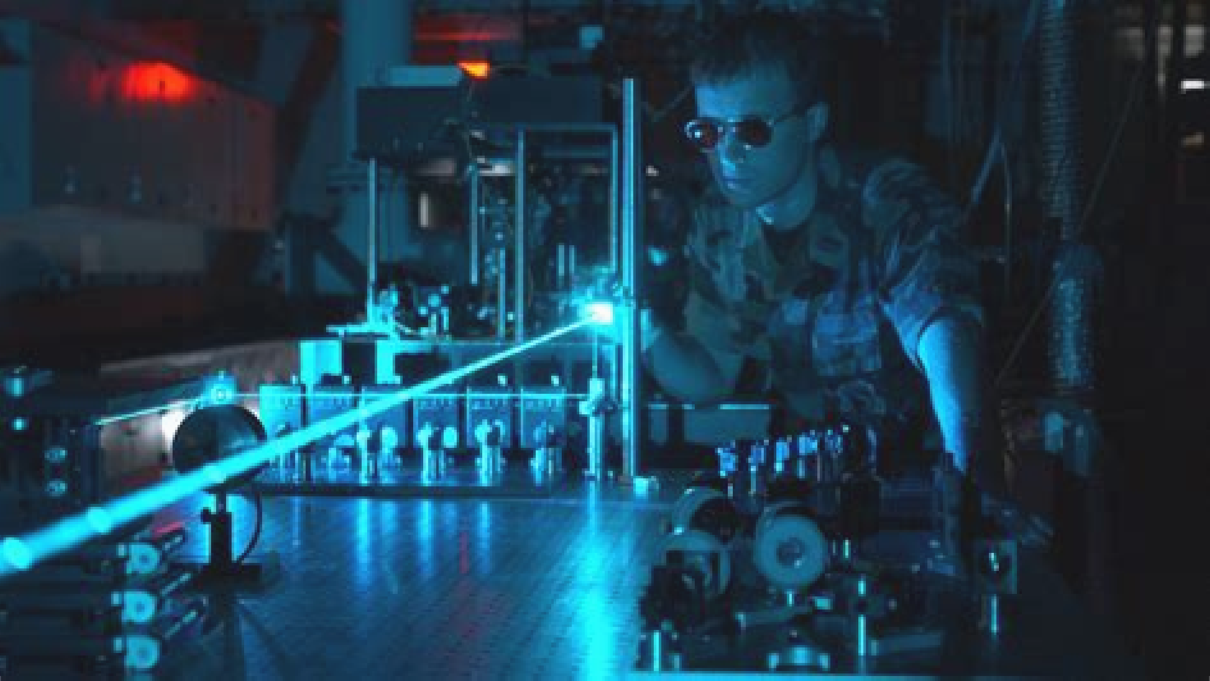
BN-254. Curve25519. DH. DSA. ECDH. ECDSA. EdDSA. NIST P-256. NIST P-384. NIST P-521. RSA encrypt. RSA sign. secp256k1.

Post-Quanten Kryptographie

Post-Quanten Kryptographie

Kryptographie mit Angriffs-Modell Quantencomputer

Zurück in die Steinzeit?



Post-Quanten Kryptographie

Algorithmische Kryptographie

mit Angriffs-Modell

Quantencomputer

Was bleibt?

- Systeme basierend auf Codierungstheorie
- Signaturen basierend auf Hash-Funktionen
- Systeme basierend auf Isogenien zwischen elliptischen Kurven
- Systeme basierend auf Gittern
- Systeme basierend auf multi-variaten Gleichungen
- Symmetrische Kryptographie

Dies sind grobe Kategorien, konkrete Systeme können trotzdem vollständig unsicher sein!

Was bleibt?

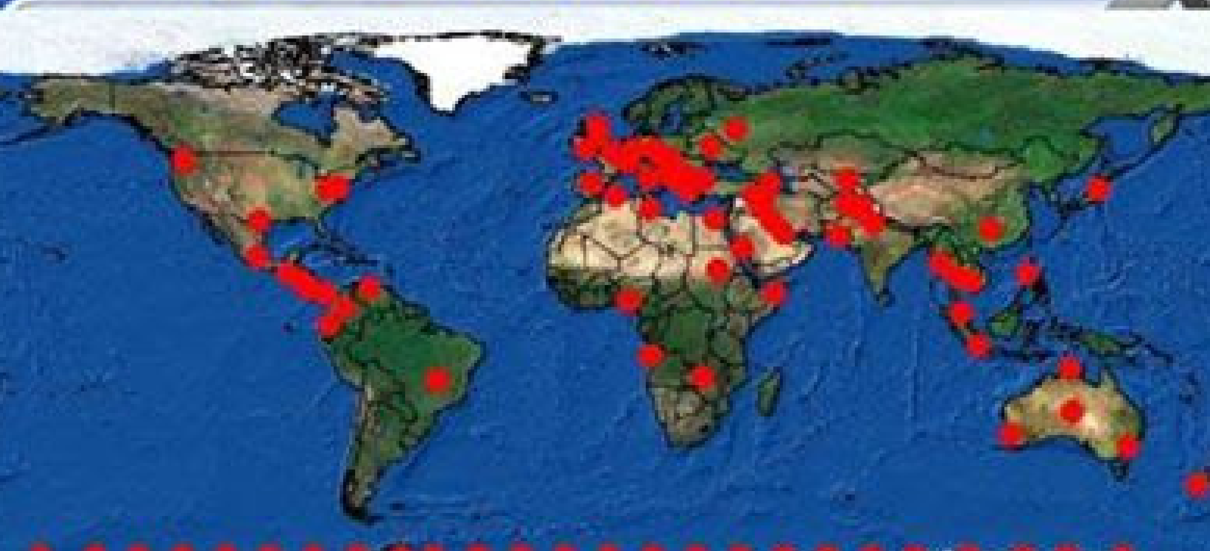
- Systeme basierend auf Codierungstheorie
- Signaturen basierend auf Hash-Funktionen
- Systeme basierend auf Isogenien zwischen elliptischen Kurven
- Systeme basierend auf Gittern
- Systeme basierend auf multi-variaten Gleichungen
- Symmetrische Kryptographie

Dies sind grobe Kategorien, konkrete Systeme können trotzdem vollständig unsicher sein!

NIST (National Institute of Standards and Technology) hält einen Wettbewerb zu Standards in Post-Quanten Kryptographie und die ersten Standards sind raus.

Warum jetzt?

Where is X-KEYSCORE?



Es eilt für Langzeitsicherheit!

- Heute fangen Angreifer alle Nachrichten ab und speichern sie. Viele Jahre später können sie diese mit einem Quantumcomputer entschlüsseln.
  Dies bringt Menschenrechtler, Journalisten, Patientendossiers (ärztliche Schweigepflicht), nationale Sicherheit, Rechtsakten, ... in Gefahr.

- Signatursysteme können später ersetzt werden, wenn es einen großen Quantencomputer gibt – aber das wird sicher geheim gehalten

Es eilt für Langzeitsicherheit!

- Heute fangen Angreifer alle Nachrichten ab und speichern sie. Viele Jahre später können sie diese mit einem Quantumcomputer entschlüsseln.
  Dies bringt Menschenrechtler, Journalisten, Patientendossiers (ärztliche Schweigepflicht), nationale Sicherheit, Rechtsakten, . . . in Gefahr.

- Signatursysteme können später ersetzt werden, wenn es einen großen Quantencomputer gibt – aber das wird sicher geheim gehalten . . . und eine der Hauptfunktionen von Signaturen sind Updates für Betriebssysteme.
- Wir müssen also *jetzt schon* Upgrades mit Post-Quanten Signaturen sichern.

Bericht der National Academy of Sciences (US, 2018)

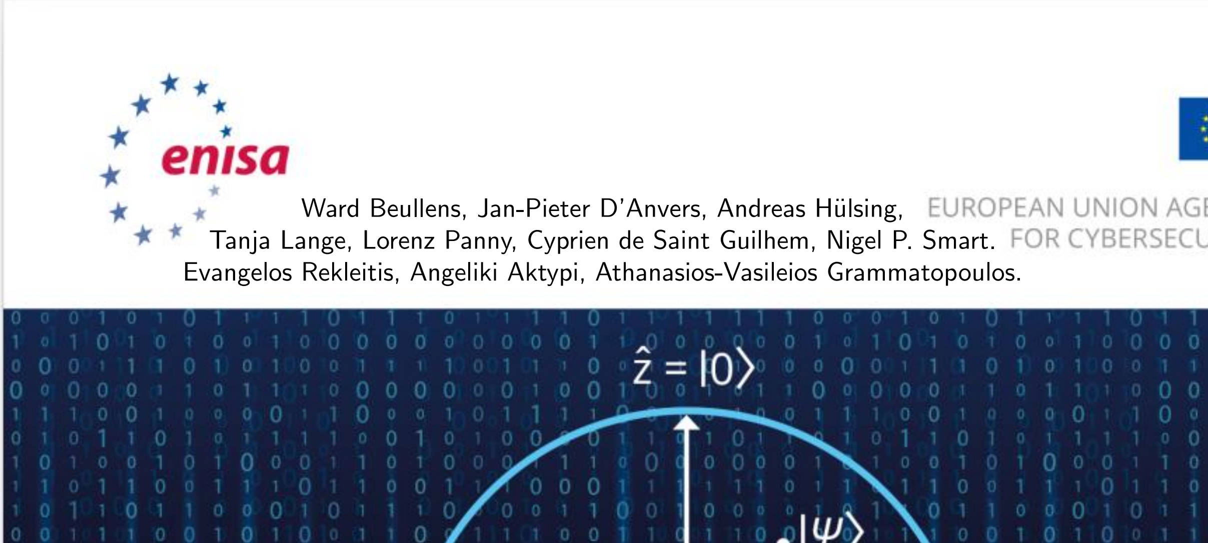
Don’t panic. “Key Finding 1: Given the current state of quantum computing and recent rates of progress, it is highly unexpected that a quantum computer that can compromise RSA 2048 or comparable discrete logarithm-based public key cryptosystems will be built within the next decade.”

Bericht der National Academy of Sciences (US, 2018)

Don't panic. "Key Finding 1: Given the current state of quantum computing and recent rates of progress, it is highly unexpected that a quantum computer that can compromise RSA 2048 or comparable discrete logarithm-based public key cryptosystems will be built within the next decade."

Panic. "Key Finding 10: Even if a quantum computer that can decrypt current cryptographic ciphers is more than a decade off, the hazard of such a machine is high enough—and the time frame for transitioning to a new security protocol is sufficiently long and uncertain—that prioritization of the development, standardization, and deployment of post-quantum cryptography is critical for minimizing the chance of a potential security and privacy disaster."

Post-Quantum Cryptography: Current state and quantum mitigation



Ward Beullens, Jan-Pieter D'Anvers, Andreas Hülsing,
Tanja Lange, Lorenz Panny, Cyprien de Saint Guilhem, Nigel P. Smart.
Evangelos Rekleitis, Angeliki Aktypi, Athanasios-Vasileios Grammatopoulos.

ENISA studies: Current state and quantum mitigation (2021)
Post-Quantum Cryptography - Integration study (2022)

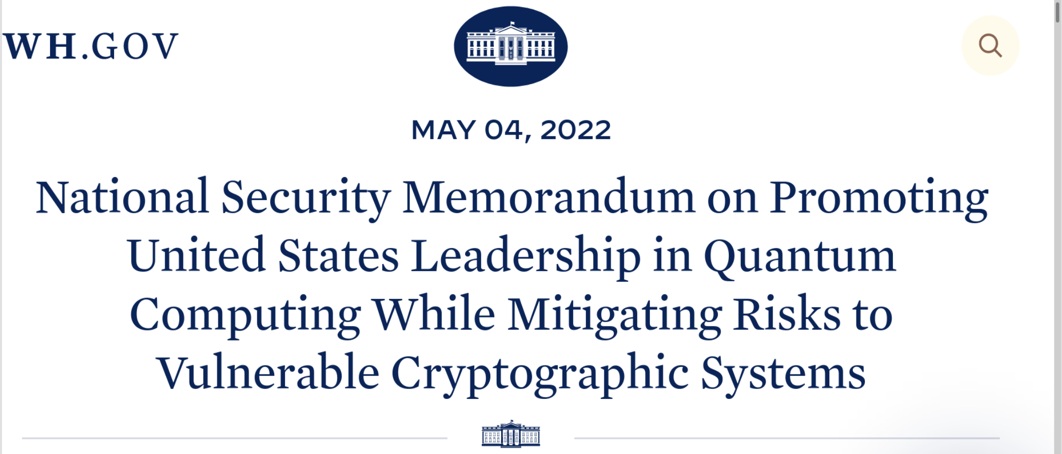
Table of contents:

1. Introduction
2. Families of Post-Quantum Algorithms
3. Security Notions and Generic Transforms
4. NIST Round 3 Finalists
5. Alternate Candidates
6. Quantum Mitigation
   - 6.1 Hybrid schemes
   - 6.2 Protective measures for pre-quantum cryptography

1. Introduction
2. Integrating post-quantum systems into existing protocols
3. New protocols designed around post-quantum systems
4. Double encryption and double signatures
5. Security proofs in the presence of quantum attackers
6. Standardization efforts for protocols

Beide Studien sind online auf ENISAs website: 2021 und 2022

In den USA tut sich was



WH.GOV

MAY 04, 2022

National Security Memorandum on Promoting United States Leadership in Quantum Computing While Mitigating Risks to Vulnerable Cryptographic Systems

White House briefing drängt zum Umstieg auf PQC aber setzt 2035 als Ziel.

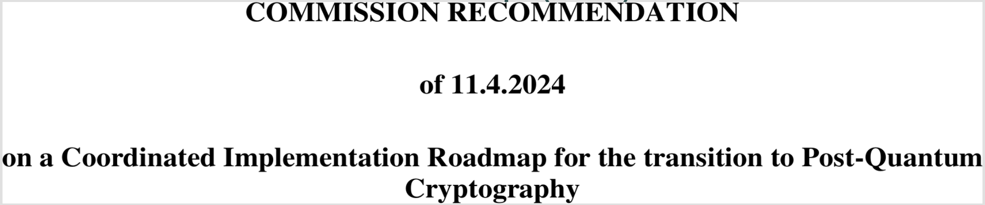
Nov 2024: NISTs erster draft einer “transition roadmap” bleibt mit 2035 zögerlich; hybrids nur für Übergang.

Französisches ANSSI

ANSSIs Stellungname von 2022 ist deutlich aktiver:
"Although this new post-quantum toolbox may seem handy for developers, the maturity level of the post-quantum algorithms presented to the NIST process should not be overestimated. Many aspects lack cryptanalytical hindsight or are still research topics, e.g. analysis of the difficulty of the underlying problem in the classical and quantum computation models, dimensioning, integration of schemes in protocols and more importantly the design of secure implementations. This situation will probably last some time after the publication of NIST standards. **Acknowledging the immaturity of PQC is important: ANSSI will not endorse any direct drop-in replacement of currently used algorithms in the short/medium term.** However, this immaturity should not serve as an argument for postponing the first deployments." (Hervorhebungen von mir)

ANSSI zertifiziert, sofern die pre-quantum Komponente zertifizierbar ist.

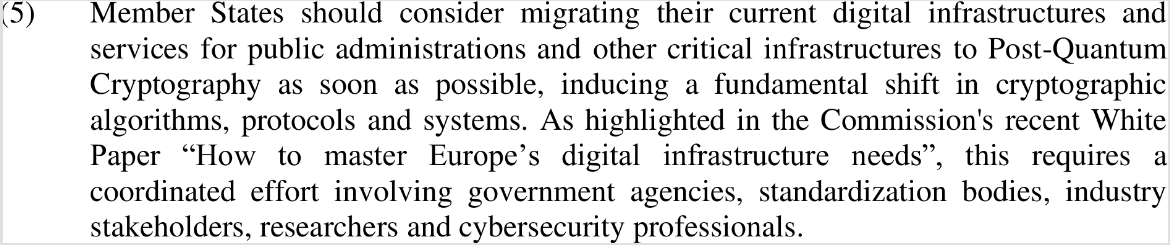
2024 EU PQC transition roadmap (link)



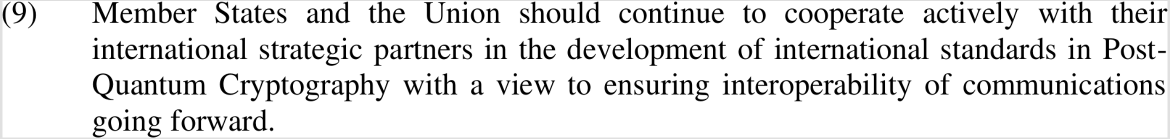
COMMISSION RECOMMENDATION

of 11.4.2024

on a Coordinated Implementation Roadmap for the transition to Post-Quantum Cryptography



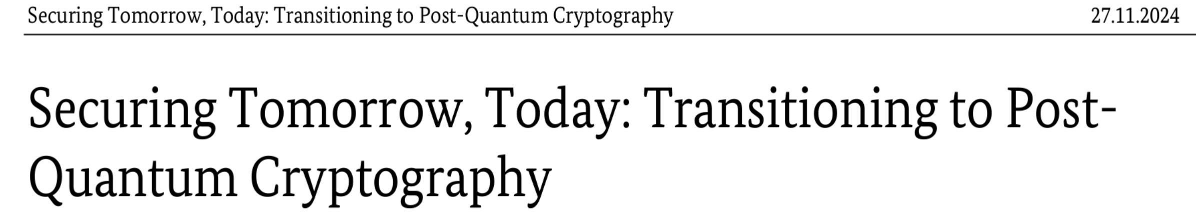
(5) Member States should consider migrating their current digital infrastructures and services for public administrations and other critical infrastructures to Post-Quantum Cryptography as soon as possible, inducing a fundamental shift in cryptographic algorithms, protocols and systems. As highlighted in the Commission's recent White Paper “How to master Europe’s digital infrastructure needs”, this requires a coordinated effort involving government agencies, standardization bodies, industry stakeholders, researchers and cybersecurity professionals.



(9) Member States and the Union should continue to cooperate actively with their international strategic partners in the development of international standards in Post-Quantum Cryptography with a view to ensuring interoperability of communications going forward.

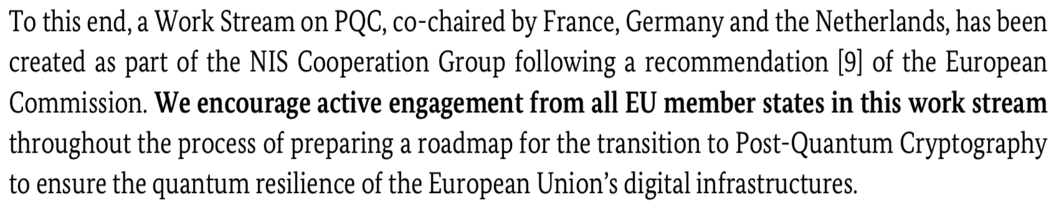
In der EU tut sich was

27.11.2024: Gemeinsames statement von 17 Ländern (link geht zur BSI Seite):



Securing Tomorrow, Today: Transitioning to Post-Quantum Cryptography 27.11.2024

Securing Tomorrow, Today: Transitioning to Post-Quantum Cryptography



To this end, a Work Stream on PQC, co-chaired by France, Germany and the Netherlands, has been created as part of the NIS Cooperation Group following a recommendation [9] of the European Commission. **We encourage active engagement from all EU member states in this work stream** throughout the process of preparing a roadmap for the transition to Post-Quantum Cryptography to ensure the quantum resilience of the European Union's digital infrastructures.

Generell hat das BSI etliche Studien zu quantun online.

Was selbst tun?

1. Wo wird crypto benutzt und wofür?
2. Was ist der impact wenn das später gebrochen wird (confidentiality, privacy, authenticity)?
3. Wer verwaltet die Software, die das liefert & gibt es updates? Plan zur Migration? Zeitplan?
4. Gibt es ggf. Alternativen, die jetzt schon gehen?
5. Genug Zeit und Geld einplanen (neue Software schreiben oder kaufen, verwalten, einpflegen, . . .)

Es gibt schon einige Handbücher für den Umstieg. Über den NIS2 PQC Workstream wird es bald (?) auch eins geben.

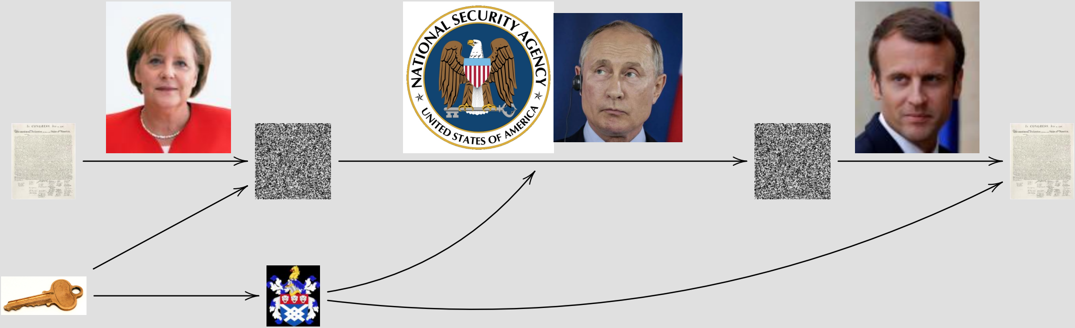
Wir haben ein Projekt bekommen, in dem wir versuchen werden, sowas zu koordinieren & Branchen einzubinden. Stay tuned!

Mehr Information

- NIST PQC Wettbewerb
- Quantum Threat Timeline 2019; upddates von 2021, 2022, 2023, und 2024.
- Status of quantum computer development (Studie vom BSI).
- ENISA Studien: Post-quantum cryptography: Integration study,
  Post-quantum cryptography: current state and quantum mitigation
- YouTube Kanal Tanja Lange: Post-quantum cryptography.
- `https://2017.pqcrypto.org/school`: PQCRYPTO Sommer-Schule
  mit 21 Vorlesungen auf Video, mit Folien und Übungsaufgaben.
- `https://2017.pqcrypto.org/exec` und `https://pqcschool.org/index.html`:
  Executive school (weniger Mathe, mehr Überblick.)
- `https://pqcrypto.org`: Übersichtsseite von Daniel J. Bernstein & mir.
- PQCrypto 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024
  Folien+Videos.
- `https://pqcrypto.eu.org`: PQCRYPTO Projekt (Empfehlungen, software, . . .)
- Migration handbook von NL Dienst

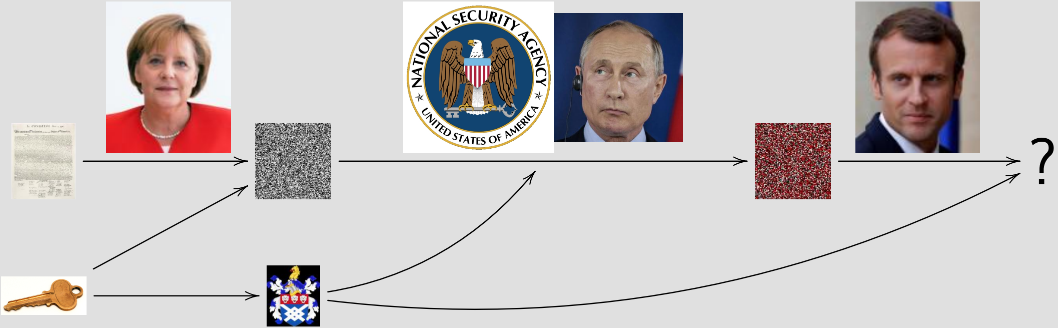
Bonus Folien

Public-key signatures



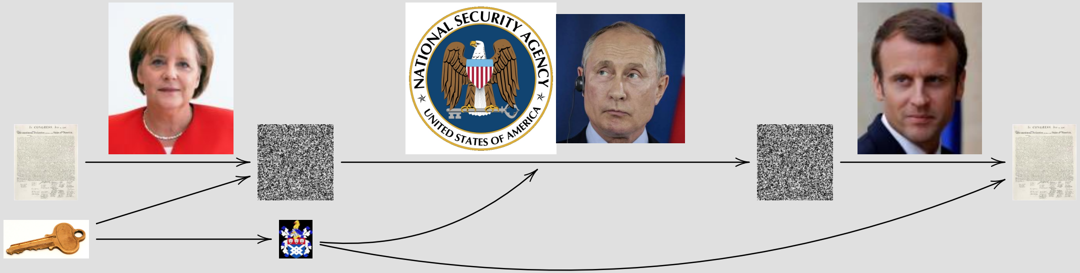
- Prerequisite: Alice has a private key 🔑 and public key 🛡.
- Prerequisite: Everyone knows 🛡 as belonging to Alice.
- Alice signs messages using 🔑. Other people verify using 🛡.

Public-key signatures



- Prerequisite: Alice has a private key 🔑 and public key 🛡.
- Prerequisite: Everyone knows 🛡 as belonging to Alice.
- Alice signs messages using 🔑. Other people verify using 🛡.
- Security goals: Integrity and authenticity.
- Nobody can produce signatures valid under 🛡 without 🔑.
- Modifications to signed message get caught.

Post-quantum public-key signatures: hash-based

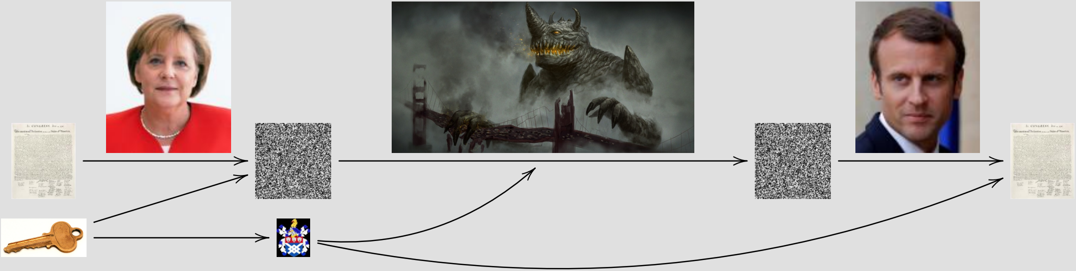


- Only one prerequisite: a good hash function, e.g. SHA3-512, . . .
  Hash functions map long strings to fixed-length strings.
  $H : \{0,1\}^* \to \{0,1\}^n$.

  Signature schemes use hash functions in handling .

- Old idea: 1979 Lamport one-time signatures;
  1979 Merkle extends to more signatures.

Post-quantum public-key signatures: hash-based



- Only one prerequisite: a good hash function, e.g. SHA3-512, ...
  Hash functions map long strings to fixed-length strings.
  $H : \{0,1\}^* \to \{0,1\}^n$.

  Signature schemes use hash functions in handling .
- Quantum computers affect the hardness only marginally (Grover, not Shor).
- Old idea: 1979 Lamport one-time signatures;
  1979 Merkle extends to more signatures.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.

Private key: bit string s, public key: $H(s)$.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.

Private key: bit string s, public key: $H(s)$.

Can only use once as s known after first use.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.
To sign 0 reveal s_0, to sign 1 reveal s_1.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.
To sign 0 reveal s_0, to sign 1 reveal s_1.

Lamport signs m via $H(m) = (h_0, h_1, \dots, h_{255})$.
Private key: 256×2 bit strings $\mathrm{s} = (s_{0,0}, s_{0,1}, s_{1,0}, s_{1,1}, \dots, s_{255,0}, s_{255,1})$,
public key: $\mathrm{p} = (H(s_{0,0}), H(s_{0,1}), H(s_{1,0}), H(s_{1,1}), \dots, H(s_{255,0}), H(s_{255,1}))$.
To sign m reveal $s_{0,h_0}, s_{1,h_1}, \dots, s_{255,h_{255}}$.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.
To sign 0 reveal s_0, to sign 1 reveal s_1.

Lamport signs m via $H(m) = (h_0, h_1, \dots, h_{255})$.
Private key: 256×2 bit strings $\mathrm{s} = (s_{0,0}, s_{0,1}, s_{1,0}, s_{1,1}, \dots, s_{255,0}, s_{255,1})$,
public key: $\mathrm{p} = (H(s_{0,0}), H(s_{0,1}), H(s_{1,0}), H(s_{1,1}), \dots, H(s_{255,0}), H(s_{255,1}))$.
To sign m reveal $s_{0,h_0}, s_{1,h_1}, \dots, s_{255,h_{255}}$.
Tradeoff: define public key as $H(\mathrm{p})$, also reveal rest of p to sign,
for short public key at expense of longer signature.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

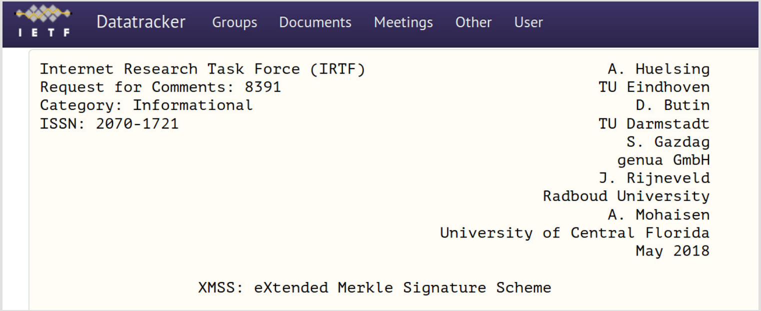
Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.
To sign 0 reveal s_0, to sign 1 reveal s_1.

Lamport signs m via $H(m) = (h_0, h_1, \ldots, h_{255})$.
Private key: 256×2 bit strings $\mathrm{s} = (s_{0,0}, s_{0,1}, s_{1,0}, s_{1,1}, \ldots, s_{255,0}, s_{255,1})$,
public key: $\mathrm{p} = (H(s_{0,0}), H(s_{0,1}), H(s_{1,0}), H(s_{1,1}), \ldots, H(s_{255,0}), H(s_{255,1}))$.
To sign m reveal $s_{0,h_0}, s_{1,h_1}, \ldots, s_{255,h_{255}}$.
Tradeoff: define public key as $H(\mathrm{p})$, also reveal rest of p to sign,
for short public key at expense of longer signature.

Winternitz achieves short public keys and signatures costing more calls to H.

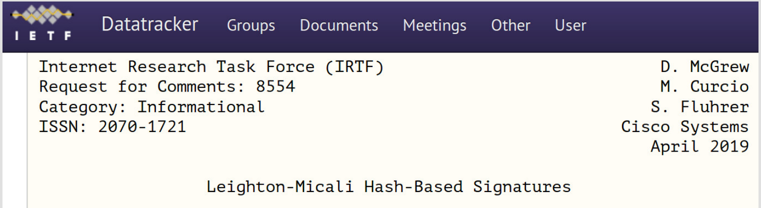
On the fast track: stateful hash-based signatures

- CFRG has published 2 RFCs: RFC 8391 and RFC 8554



```
Internet Research Task Force (IRTF)                          A. Huelsing
Request for Comments: 8391                                 TU Eindhoven
Category: Informational                                         D. Butin
ISSN: 2070-1721                                            TU Darmstadt
                                                              S. Gazdag
                                                             genua GmbH
                                                            J. Rijneveld
                                                      Radboud University
                                                             A. Mohaisen
                                             University of Central Florida
                                                                May 2018

                 XMSS: eXtended Merkle Signature Scheme
```



```
Internet Research Task Force (IRTF)                            D. McGrew
Request for Comments: 8554                                     M. Curcio
Category: Informational                                       S. Fluhrer
ISSN: 2070-1721                                            Cisco Systems
                                                              April 2019

                    Leighton-Micali Hash-Based Signatures
```

On the fast track: stateful hash-based signatures

- CFRG has published 2 RFCs: RFC 8391 and RFC 8554
- NIST has standardized the same two schemes.

NIST

Information Technology Laboratory

COMPUTER SECURITY RESOURCE CENTER

PROJECTS

Stateful Hash-Based Signatures

On the fast track: stateful hash-based signatures

- CFRG has published 2 RFCs: RFC 8391 and RFC 8554
- NIST has standardized the same two schemes.

Information Technology Laboratory

COMPUTER SECURITY RESOURCE CENTER

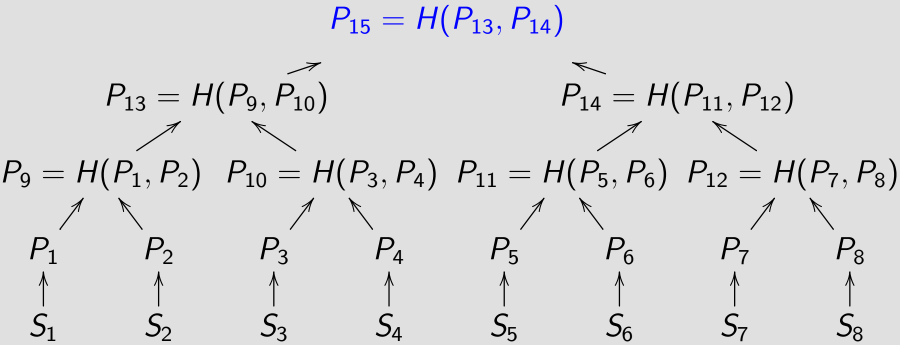
PROJECTS

Stateful Hash-Based Signatures

- ISO SC27 JTC1 WG2 has standard for stateful hash-based signatures.

Merkle's (e.g.) 8-time signature system

Hash 8 one-time public keys into a single Merkle public key P_{15}.



$P_{15} = H(P_{13}, P_{14})$

$P_{13} = H(P_9, P_{10})$ $P_{14} = H(P_{11}, P_{12})$

$P_9 = H(P_1, P_2)$ $P_{10} = H(P_3, P_4)$ $P_{11} = H(P_5, P_6)$ $P_{12} = H(P_7, P_8)$

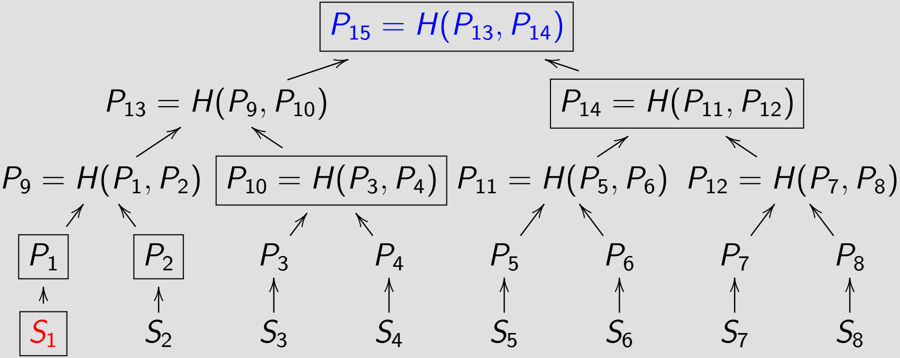
P_1 P_2 P_3 P_4 P_5 P_6 P_7 P_8

S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8

$S_i \rightarrow P_i$ can be Lamport or Winternitz one-time signature system.
Each such pair (S_i, P_i) may be used only once.

Signature in 8-time Merkle hash tree

Signature of first message: $(\texttt{sign}(m, S_1), P_1, P_2, P_{10}, P_{14})$.



$\boxed{P_{15} = H(P_{13}, P_{14})}$

$P_{13} = H(P_9, P_{10})$ $\boxed{P_{14} = H(P_{11}, P_{12})}$

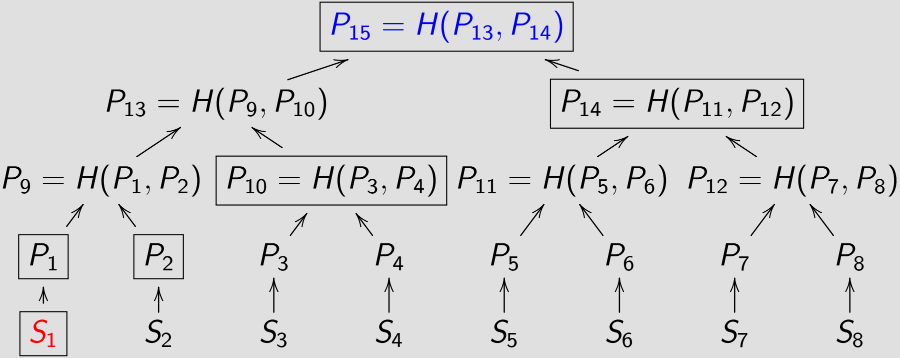
$P_9 = H(P_1, P_2)$ $\boxed{P_{10} = H(P_3, P_4)}$ $P_{11} = H(P_5, P_6)$ $P_{12} = H(P_7, P_8)$

$\boxed{P_1}$ $\boxed{P_2}$ P_3 P_4 P_5 P_6 P_7 P_8

$\boxed{S_1}$ S_2 S_3 S_4 S_5 S_6 S_7 S_8

Signature in 8-time Merkle hash tree

Signature of first message: $(\texttt{sign}(m, S_1), P_1, P_2, P_{10}, P_{14})$.



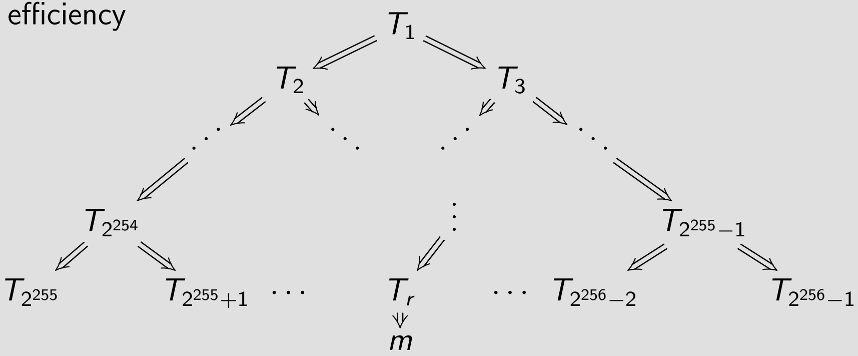
Verify signature $\texttt{sign}(m, S_1)$ with public key P_1 (provided in signature).
Link P_1 against public key P_{15} by computing $P'_9 = H(P_1, P_2)$, $P'_{13} = H(P'_9, P_{10})$, and comparing $H(P'_{13}, P_{14})$ with P_{15}. Reject if $H(P'_{13}, P_{14}) \neq P_{15}$.

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 m; uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) on path to T_1.

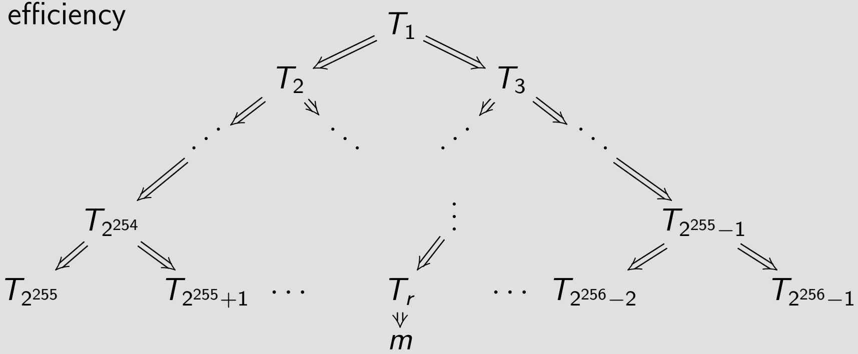
Generates ith secret key deterministically as $H_k(i)$ where k is master secret.

Important for efficiency



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 m; uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) on path to T_1. Generates ith secret key deterministically as $H_k(i)$ where k is master secret. Important for efficiency

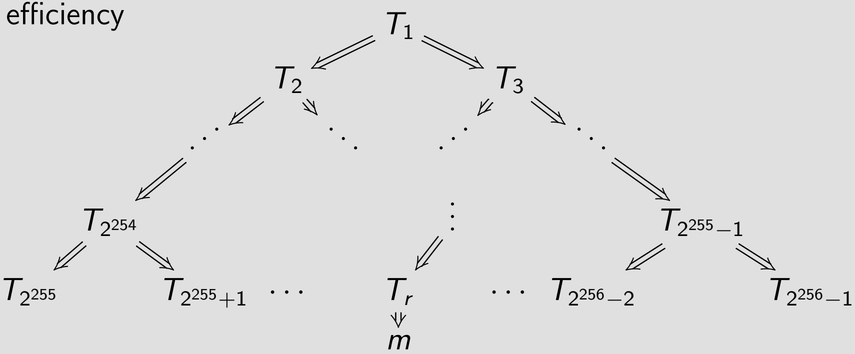


T_i for small i gets used repeatedly (each time an m falls in that sub-tree)

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 m; uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) on path to T_1. Generates ith secret key deterministically as $H_k(i)$ where k is master secret.

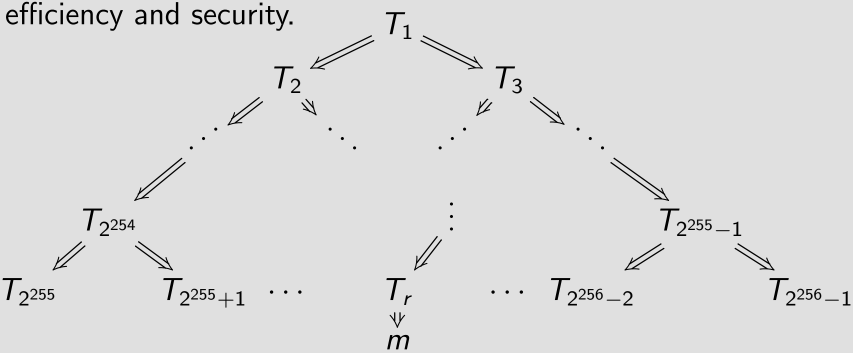
Important for efficiency



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.

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 m; uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) on path to T_1. Generates ith secret key deterministically as $H_k(i)$ where k is master secret. Important for efficiency and security.



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.

NIST submission SPHINCS+

- Post-quantum signature based on hash functions.
- Requires only a secure hash function, no further assumptions.
- Based on ideas of Lamport (1979) and Merkle (1979).
- Developed starting from SPHINCS with
  - improve multi-signature,
  - smaller keys,
  - Option for shorter signatures (30kB instead of 41kB) if “only” 2^{50} messages signed.
- Three versions (using different hash functions)
  - SPHINCS+-SHA3 (with SHAKE256),
  - SPHINCS+-SHA2 (with SHA-256),
  - SPHINCS+-Haraka (with Haraka, a hash function for short inputs).

More info at `https://sphincs.org/`.

See also my course page for more detailed videos and slides.