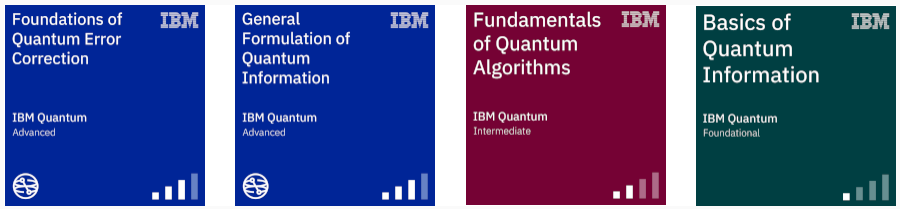
Quantum Computation and Information
The Crypto Threat

Dr. Thorsten Hehn



Outline

Problem statement

Overview: Symmetric & Asymmetric Cryptography

Impact of Quantum Computing Algorithms

Attack Vectors Relevant Today

Post-Quantum Cryptography

Real-World Examples: Status of TLS 1.3 and SSH

Summary

Problem statement

Problem statement

- The advent of a *cryptographically relevant quantum computer (CRQC)* has an impact on the security of most of the widely used cryptographic algorithms
- We provide an overview over the algorithms, the impact of the CRQC, and the attack which are already relevant today

Overview: Symmetric & Asymmetric Cryptography

Overview: Symmetric & Asymmetric Cryptography

	Asymmetric Cryptography	**Symmetric Cryptography**
System	Use pair of related keys	Use one shared key
Keys	Private key, public key	Singular key
Key-Mgmt	Private key kept secret, public key openly shared	Singular key kept secret, shared among participants
Main Methods & Features	Encryption, Key Exchange Digital signatures	Encryption, Message authentication
Unique strength	Key Exchange, PKI systems	Processing speed
Examples	RSA, ECC, DH, ElGamal	AES, ChaCha20

Impact of Quantum Computing Algorithms

Impact of Quantum Computing Algorithms

- All widely deployed classical asymmetric cryptography schemes based on factoring or discrete logarithms are broken by Shor's Algorithm
- Grover's Algorithm weakens all widely deployed symmetric cryptography schemes and the preimage security of hashing functions

Impact of Quantum Computing Algorithms

	Scheme	Core Math	Impact	Algorithm
Asymmetric	RSA DSA DH, ElGamal ECDSA ECDH, ECElGamal	Integer Factorization Discrete Logarithm Discrete Logarithm EC-Discrete Log. EC-Discrete Log.	Breaks Completely	Shor's
Symmetric + Hash	AES ChaCha20 SHA	SUB / MIX / KEY ADD / ROT / XOR XOR / ROT / MOD	Weakened	Grover's

Asymmetric cryptography

Current asymmetric cryptography systems are built on two core mathematical one-way problems:

- Integer Factorization
- Discrete Logarithm

Shor's algorithm:

- A cryptanalysis tool which solves integer factorization by a probabilistic algorithm using order finding $\Rightarrow$ RSA breaks
- A similar order-finding approach can be taken to solve the Discrete Logarithm problem $\Rightarrow$ DSA, ElGamal, ECC breaks
- The Diffie-Hellman protocol builds on top of the Discrete Logarithm problem $\Rightarrow$ DH, ECDH breaks

Symmetric cryptography & Hash functions

- Not affected by Shor's algorithm
- Prone to brute-force attacks (unstructured search problem)
- Grover's algorithm on a CRQC: quadratic speedup
- Effective security level is reduced by a factor of 2

Symmetric Cryptography	
AES Scheme	Reduced Security Level CRQC + Grover's
AES-128	$\rightarrow 64$
AES-256	$\rightarrow 128$
AES-512	$\rightarrow 256$
...	...
...	...

Hash functions	
SHA Algorithm	Reduced Preimage Security CRQC + Grover's
SHA-128	$\rightarrow 64$
SHA-256	$\rightarrow 128$
SHA-512	$\rightarrow 256$
...	...
...	...

Comparison

Asymmetric Cryptography

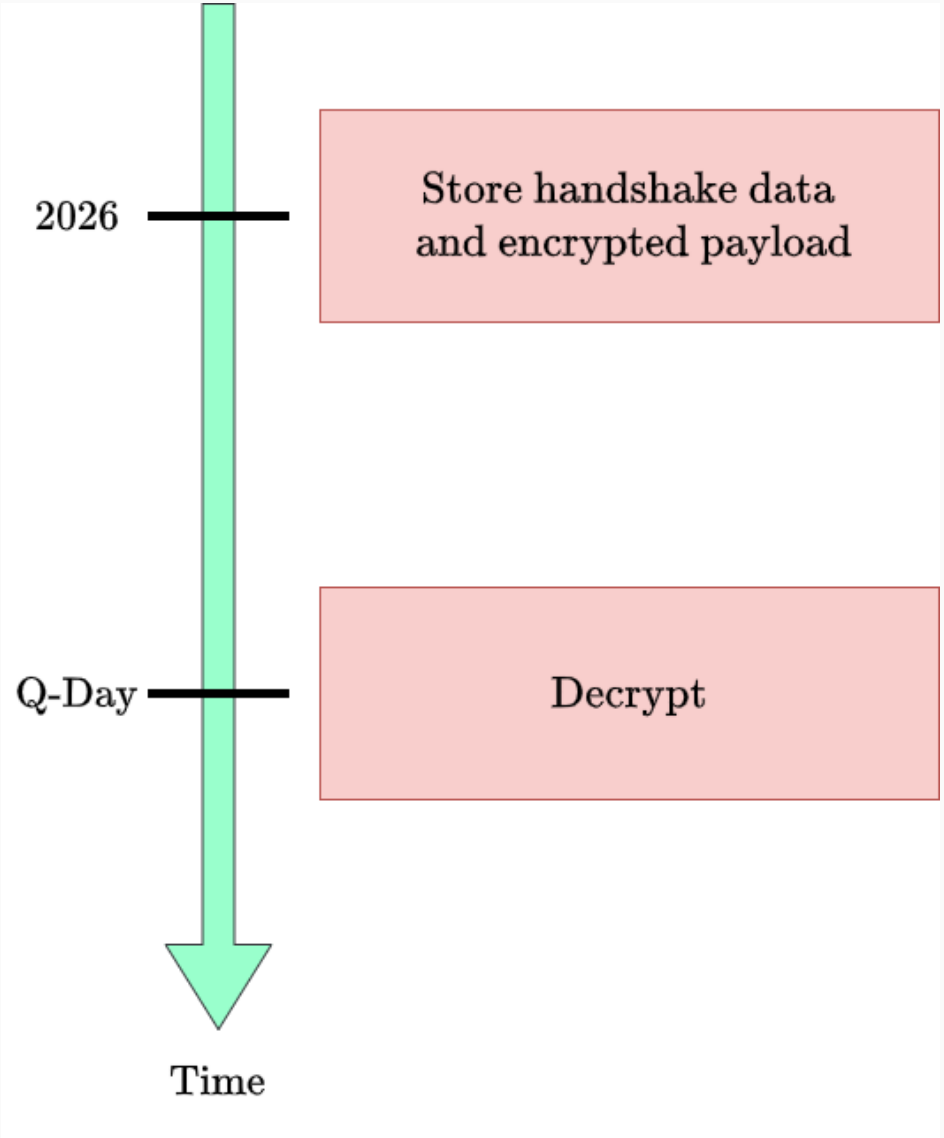
- Shor's cryptanalysis breaks existing algorithms
- New algorithms need to be developed, implemented, and deployed
- Lattice-based algorithms are believed to be secure against quantum cryptanalysis

Symmetric Cryptography and Hash Functions

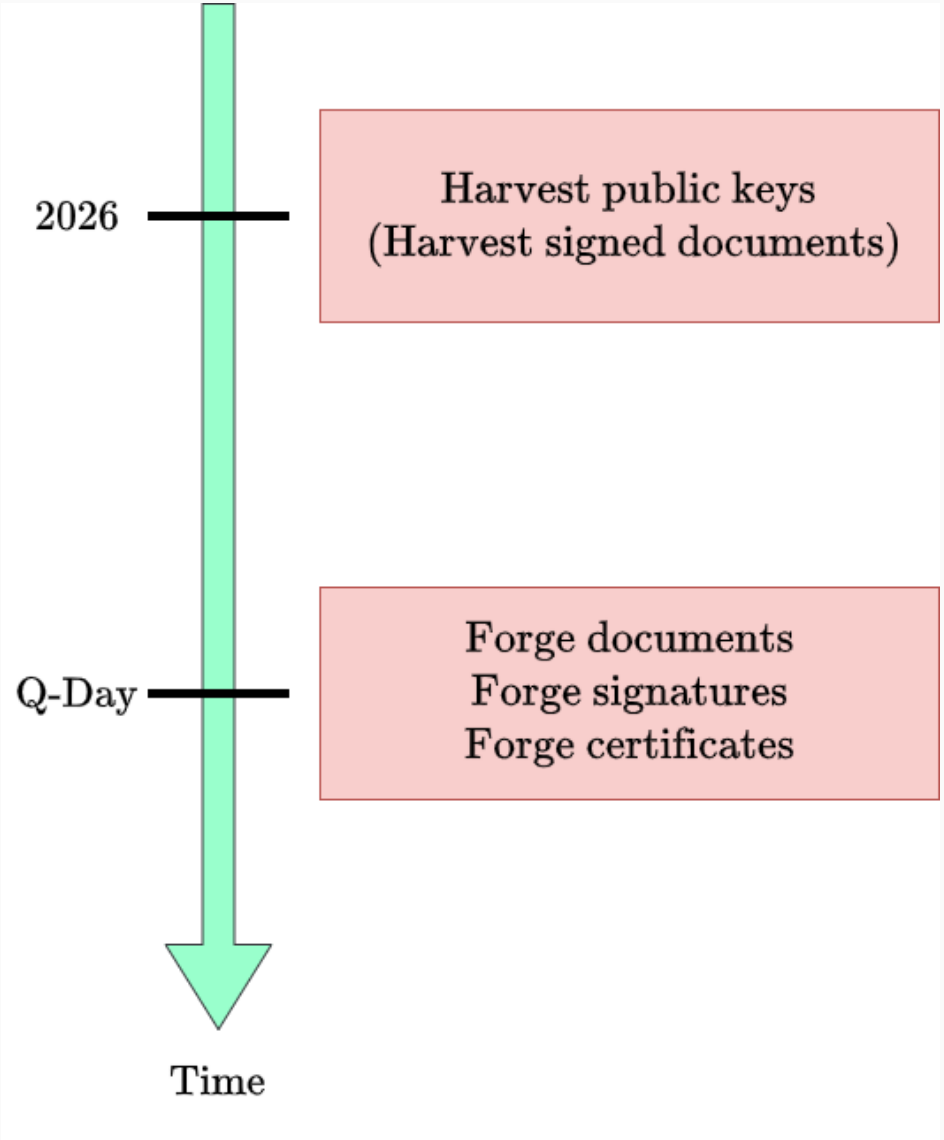
- Grover's algorithm weakens their security level
- Use higher-level schemes to provide the targeted security level
- Example: use AES-256 to provide a post-quantum security level of 128 bits

Attack Vectors Relevant Today

Attack Vector 1: Store Now - Decrypt Later



Attack Vector 2: Trust Now - Forge Later



Remarks on attack vectors

Store Now - Decrypt Later

- Confidentiality is lost due to retrospective attack

Trust Now - Forge Later

- Integrity authenticity is lost due to retrospective attack
- Of relevance for credentials with long lifetimes
  - TLS certificates
  - Firmware signing credentials
  - Long-lived authentication keys

The Firmware Update Trap

- Devices in the field need to update their firmware to quantum-safe cryptography (incl. the update routine)
- Trap 1: The update needs to be installed before quantum computing becomes ubiquitous
- Otherwise the update based on the old trust chain is untrustworthy (Trust Now - Forge Later)
- Trap 2: New cryptography schemes usually require more resources - development necessary
- Of particular interest for:
  - IoT devices
  - Aircraft and traffic
  - Industrial equipment
  - Telecom infrastructure

Note: This slide assumes that the root public key for validating the firmware update is not burnt into silicon and thus updating is possible.

Post-Quantum Cryptography

Post-Quantum Cryptography

Several cryptographic schemes are conjectured to protect against quantum attacks: post-quantum cryptography (PQC). They rely on other mathematical problems than integer factorization than and discrete logarithm.

- The National Institute of Standards and Technology (NIST) called for PQC proposals in 2016
- NIST ran multiple rounds, selected and derived
- NIST released three finalized standards in 2024
- Two further algorithms are considered for standardization
- Federal Information Processing Standards (FIPS)

PQC FIPS Standards

Standard	Status	Algorithm	Application
FIPS 203	Final	ML-KEM	Key Exchange
FIPS 204	Final	ML-DSA	Dig. Signature
FIPS 205	Final	SLH-DSA	Dig. Signature Backup
FIPS 206	In development	FN-DSA	Dig. Signature Backup
TBD in 2027	In development	HQC	Key Exchange Backup

- ML-KEM: Module-Lattice-Based Key-Encapsulation Mech.
- ML-DSA: Module-Lattice-Based Digital Signature
- SLH-DSA: Stateless Hash-Based Digital Signature
- FN-DSA: FFT over NTRU-Lattice-Based Digital Signature
- HQC: Hamming Quasi Cyclic

NIST Information on Migration to PQC

Key Message 1: Key algorithms are standardized and ready for migration. No need to wait for backup algorithms.

FIPS 203

Module-Lattice-Based
Key-Encapsulation Mechanism Standard
(ML-KEM)

FIPS 204

Module-Lattice-Based
Digital Signature Standard
(ML-DSA)

FIPS 205

Stateless Hash-Based
Digital Signature Standard
(SLH-DSA)

Figure: Standardized Algorithms. Source: nist.gov

NIST Information on Migration to PQC

Key Message 2: Legacy algorithms will be deprecated and later disallowed. Timeline considerations in NIST Internal Report 8547.

485 **4.1.1. Digital Signatures**

486 Table 2 lists currently approved quantum-vulnerable digital signature algorithm standards.

487

488 **Table 2: Quantum-vulnerable digital signature algorithms**

Digital Signature Algorithm Family	Parameters	Transition
ECDSA [FIPS186]	112 bits of security strength	***Deprecated*** after 2030 ***Disallowed*** after 2035
	≥ 128 bits of security strength	***Disallowed*** after 2035
EdDSA [FIPS186]	≥ 128 bits of security strength	***Disallowed*** after 2035
RSA [FIPS186]	112 bits of security strength	***Deprecated*** after 2030 ***Disallowed*** after 2035
	≥ 128 bits of security strength	***Disallowed*** after 2035

Figure: Screenshot NIST IR 8547. Source: nist.gov

Real-World Examples: Status of TLS 1.3 and SSH

Status of TLS 1.3

Transport Layer Security (TLS) is a protocol to provide communications security over networks, designed by IETF

- Goals: confidentiality, integrity, authenticity
- Key exchange: dynamic between client and server

Current status:

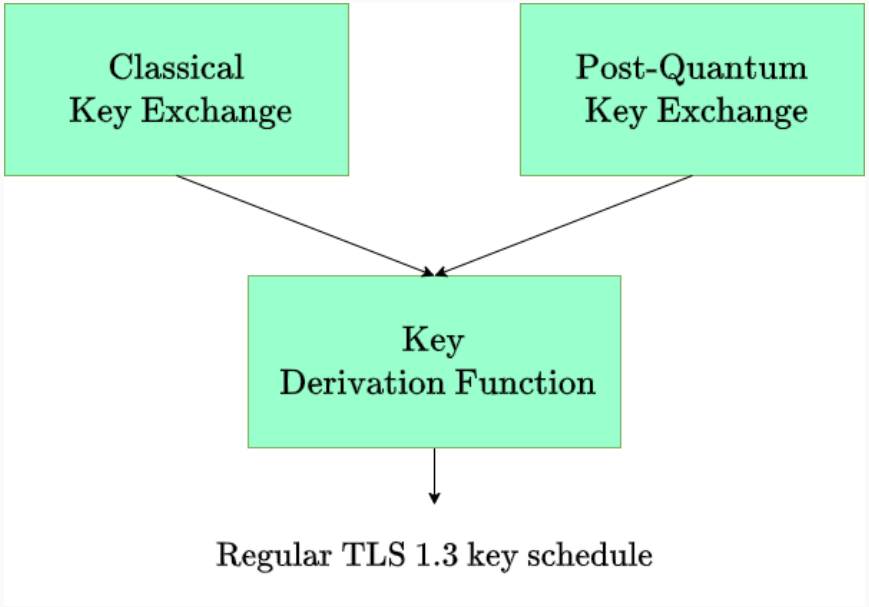
- Current version: TLS 1.3 released by IETF in 2018
- TLS 1.3 (RFC 8446 - Aug 2018, RFC 9846 - July 2026) specifies only classical algorithms

Dilemma:

- Early adopters wish to be safe against quantum threats
- As post-quantum cryptography algorithms are relatively new, conservative users may not trust them yet

Status of TLS 1.3

IETF RFC 9954: Hybrid Key Exchange in TLS 1.3 (July 2026)



- TLS 1.3 is not replaced, but a new key exchange method is added
- Potential security against Store Now - Decrypt Later
- Security level: strongest surviving component
- Example Group (RFC 10024, August 2026):
  X25519 ECDH (Classic) + ML-KEM-768 (PQC)

Status of TLS 1.3

Signatures and certificate chains:

- Same dilemma: early adopters and conservative users
- No easy-to-implement hybrid approach
- Work in progress

Verify whether a server supports Hybrid Key Exchange TLS 1.3

Step 1: Verify that you use openssl version 3.5 or later:

```
th@tp2:~$ openssl --version
OpenSSL 3.6.3 9 Jun 2026 (Library: OpenSSL 3.6.3 9 Jun 2026)
```

Step 2: Connect to an example server

```
th@tp2:~$ openssl s_client -connect <your_example>:443 -tls1_3
```

Step 3: Check that the negotiated group features MLKEM

```
Negotiated TLS1.3 group: X25519MLKEM768
```

Status of SSH

- Since openssh 10.0, a hybrid key exchange using ECDH with X25519 and ML-KEM-768 is used by default for key agreement
- As of August 2026, the implementation is ahead of IETF's standardization
- Authentication using ECDSA / RSA (public/private key) is still based on classical algorithms
- Note: signing request and signature are signalled over ssh's secure channel (with hybrid key exchange). The remaining threat is an attacker who has access to the public keys on the ssh server.

Summary

Summary

- A cryptographically relevant quantum computer does not exist today
- When it will exist, it breaks many of today's widely used asymmetric cryptographic algorithms, and weakens many symmetric cryptographic algorithms
- The attacks *Store Now - Decrypt Later* and *Trust Now - Forge Later* are relevant today
- As an example for real-world cryptography and protocols, we looked at TLS 1.3 and observed how the hybrid key exchange mechanism provides an interim solution with potential security against *Store Now - Decrypt Later*