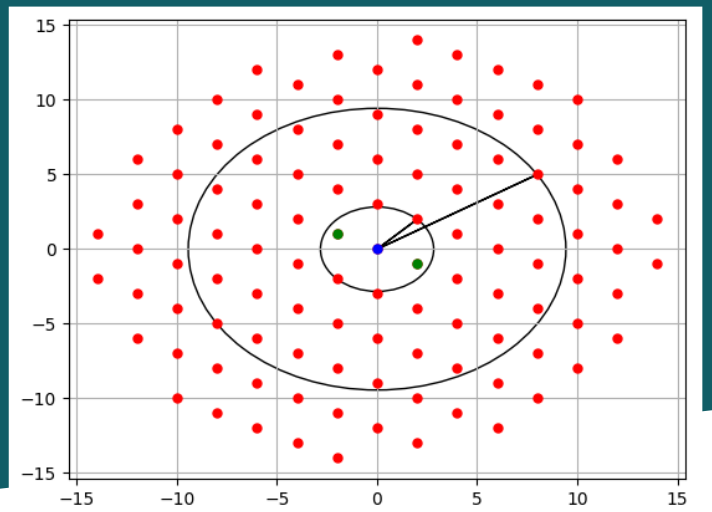
Aktuelle Entwicklungen in der Kryptologie im Zeitalter von Quantencomputern

Fachbereichskolloquium Informatik

2. Juni 2026

Konstantin Knorr

Hochschule Trier



Informatik
Hauptcampus

HOCHSCHULE TRIER

Agenda

- Quantencomputer
- Post-Quanten Kryptologie
  - Bedrohungslage für „klassische Kryptologie“
  - FIPS-Standards 203, 204, 205
  - Deepdive Dilithium
  - Crypto-Agility
  - Umsetzung an der Hochschule Trier
- Homomorphe Verschlüsselung
  - Grundidee
  - Bestehende Verfahren
  - Case Study: Homomorphic Grader
- Literatur

Was ist ein Quantencomputer (QC)?

- Basiert auf den Erkenntnissen aus der Quantenmechanik
- Qubits, statt Bits, können mehrere Zustände gleichzeitig annehmen
- Qubits beeinflussen sich gegenseitig
- Für bestimmte Probleme erlauben Qubits eine Form von Quantenparallelität.
- Große technische Hürden bei der praktischen Umsetzung

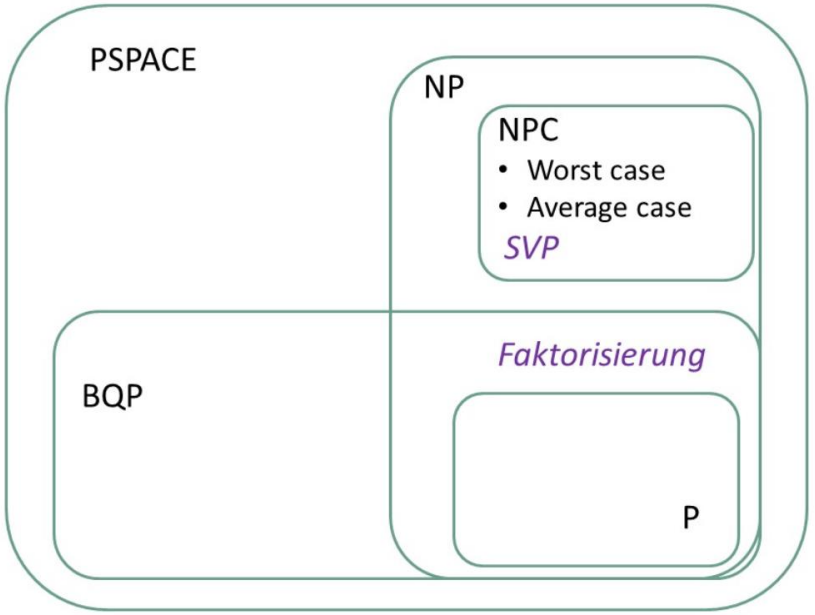
Wichtige Quanten-Algorithmen

- Shor, 1996: ermöglicht Faktorisierung
- Grover, 1996: „schnelle Suche“
- BB84 (Bennett & Brassard, 1984): Quantenschlüsselverteilung, Turing Award 2026
- Weitere Algorithmen: https://quantumalgorithmzoo.org/

Aktueller IBM Quantencomputer

Bildquelle: IBM

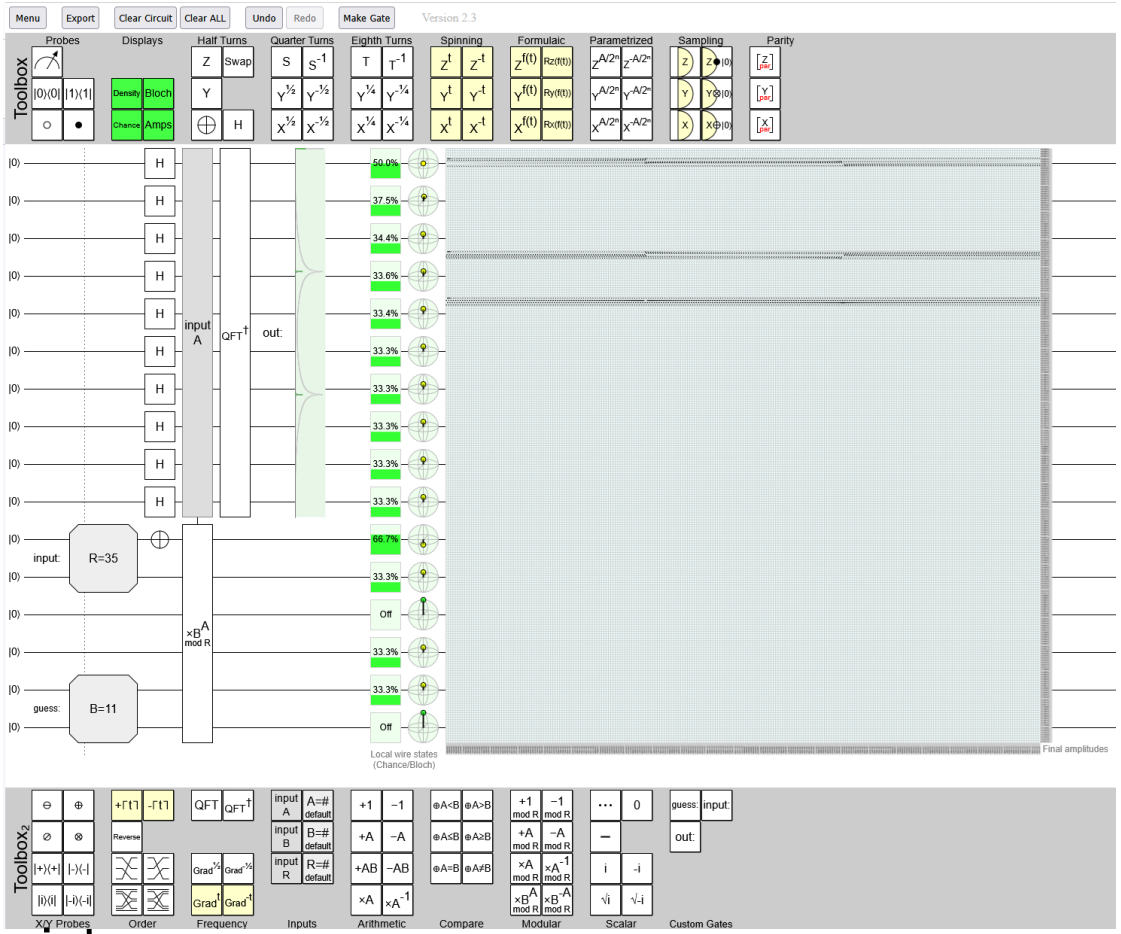
- Interdisziplinarität Informatik, Physik
- Programmierung
- Neue Komplexitätsklassen
- Kryptologie



Komplexitätsklassen inkl BQP. Quelle: Homeister

BQP = Bounded Error Quantum Polynomial Time

Implementierung von Shors Algorithmus im Simulator, https://algassert.com/quirk?ref=mlq.ai



Auswirkungen des Quantencomputer (QC) auf die Kryptologie

- Shors Algorithmus löst das Faktorisierungs- (RSA) und diskrete Logarithmusproblem (ECDH, ECDSA, DH) in polynomieller Zeit.
- Angriff: Harvest now, decrypt later
- Post-Quanten Kryptologie (PQK bzw. PQC) = Kryptologie auf klassischen Computern, die resistent gegen Angriffe mit QCs sind.
- Quanten-Kryptologie = Kryptologie mit QCs
- BSI rechnet mit relevanten QCs Anfang der 30er Jahre

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Keywords: algorithmic number theory, prime factorization, discrete logarithms, Church's thesis, quantum computers, foundations of quantum mechanics, spin systems, Fourier transforms

Timeline for Cryptographically Relevant Quantum Computers

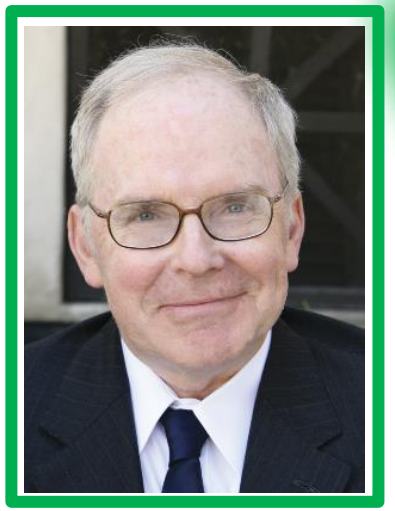
Quelle: https://qramm.org/learn/quantum-threat-timeline.html

Auswirkungen eines QCs auf die „klassische Kryptologie“

Kryptologische Verfahren	Bewertung der Quantenresistenz	Konsequenz / Probleme
Symmetrische Verschlüsselungsverfahren wie AES	Gut	Vergrößern der Parameter wie Schlüssellänge, Blockgröße und Rundenzahl z.B. um Faktor 2, vgl. Grover-Algorithmus
Hashfunktionen wie SHA-3	Gut	Vergrößern der Parameter wie Hashlänge und Rundenzahl z.B. um Faktor 2, vgl. Grover-Algorithmus
Asymmetrische Verfahren basierend auf Zahlentheorie für digitale **Signaturen** und **Schlüsselaustausch** wie z.B. RSA, DSA, DH, Elgamal, ECDH, ECDSA	**Schlecht**	**Nicht mehr sicher.** Der Shor-Algorithmus kann diese Verfahren effizient brechen. Vergrößerung der Schlüssellänge hilft nicht. ⇨ NIST Ausschreibung seit 2016
Verfahren basierend auf Informationstheorie wie z.B. OTP	Sehr gut	Kann weiter genutzt werden. Probleme: Schlüssel sehr groß + Schlüsselmanagement schwierig

KEM

HQC



NTRU
Number
Theorists 'R' Us

Signature

GeMSS: A Great Multivariate Short Signature

NIST PQ Round 3 Finalists, 19.5.2022, https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions

Aktueller Stand der PQK-Standardisierung
(Juni 2025)

- FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard (ML-KEM), ehemals CRYSTALS-Kyber
- FIPS 204: Module-Lattice-Based Digital Signature Algorithm (ML-DSA), ehemals CRYSTALS-Dilithium
- FIPS 205: Stateless Hash-Based Digital Signature Algorithm (SLH-DSA), ehem. SPHINCS+

Algebraische Struktur	**KEM**	**Signatures**
Gitter	**CRYSTALS-Kyber** NTRU, SABER, Frodo	**CRYSTALS-Dilithium** FALCON FIPS 206
Hash-Funktion		**SPHINCS+**
Fehlercodes	McEliece, BIKE, HQC	FIPS 207
Isogenie	SIKE	
Multivariate Polynome		Rainbow, GeMSS
Zero-Knowledge		PICNIC

Kritische Würdigung der neuen Standards

FIPS 203: ML-KEM

- Schnell
- Schlüssel etwas größer als bei RSA, viel größer als bei ECDH
- Relativ junge Sicherheitsanalyse,
- Mögliche Schwäche bei algebraischer Struktur

FIPS 204: ML-DSA

- Wie FIPS 203
- Geeignet für Zertifikate
- Signatur und Schlüssel groß (64 vs. 2.420 Byte)

FIPS 205: SLH-DSA

- Konservativste Sicherheitsannahmen
- Kleine Keys, gut für Wurzel-Zertifikate
- Extrem langsame Signierung (*3000 bzw. *200)
- Riesige Signaturen (64 Byte vs. 8 bzw. 17 KB)

Größen- und Zeitvergleich bei PQK-Signaturverfahren

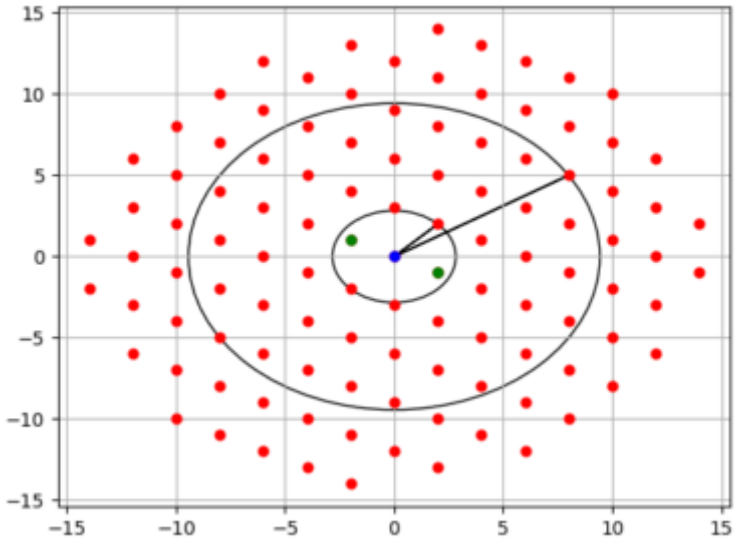
		Size (bytes)		Relative time	
		Public key	**Signature**	**Verification**	**Signing**
Non PQ	**NIST P-256**	64	64	1 (baseline)	1 (baseline)
	RSA-2048	256	256	0.2	25
NIST finalists	**Dilithium2**	1,320	2,420	0.3	2.5
	Falcon512	897	666	0.3	5 *
	Rainbow I	157,800	66	0.1	2.4
	Rainbow I CZ	58,800	66	12	2.4
NIST alternates*	**SPHINCS⁺-128ss har.**	32	7,856	1.7	3,000
	SPHINCS⁺-128fs har.	32	17,088	4	200
	Picnic-L1-full	34	32,061	21	60
	GeMMS128	352,190	33	0.4	5,000
Others	**SQISign**	64	204	500	60,000
	XMSS-SHAKE_20_128 *	32	900	2	10 *

Quelle: https://blog.cloudflare.com/sizing-up-post-quantum-signatures/

CRYSTALS-Kyber / FIPS 203 ML-KEM

- Autoren: Gruppe von ~10 Kryptologen rund um Joppe Bos
- Kyber ist Teil der „Cryptographic Suite for Algebraic Lattices" (CRYSTALS). CRYSTALS-Dilithium ist ein eng verwandtes Signaturverfahren (FIPS 204).
- Kyber ist ein **Key Encapsulation Mechanism (=KEM).**
- Kyber gilt als sicher gegen die aktuell bekannten Angriffe mit klassischen und mit Quantencomputern. Genauer: **IND-CCA-secure KEM** (Indistinguishability under adaptive chosen ciphertext attack).
- Die Sicherheit von Kyber basiert auf dem **Module Learning With Errors (MLWE)**-Problem, das verwandt ist mit dem **Shortest Vector Problem (SVP)**.
- Fun Fact: Kyber-Kristalle sind die in den **Laserschwertern der Jedi-Ritter** enthaltenen Kristalle.

	NIST-Level	Priv. Key	Pub Key	Cipher-text
Kyber512	1	1632	800	768
Kyber768	3	2400	1184	1088
Kyber1024	5	3168	1568	1568
RSA3072	1	384	384	384
RSA7860	3	960	960	960
RSA15360	5	1920	1920	1920



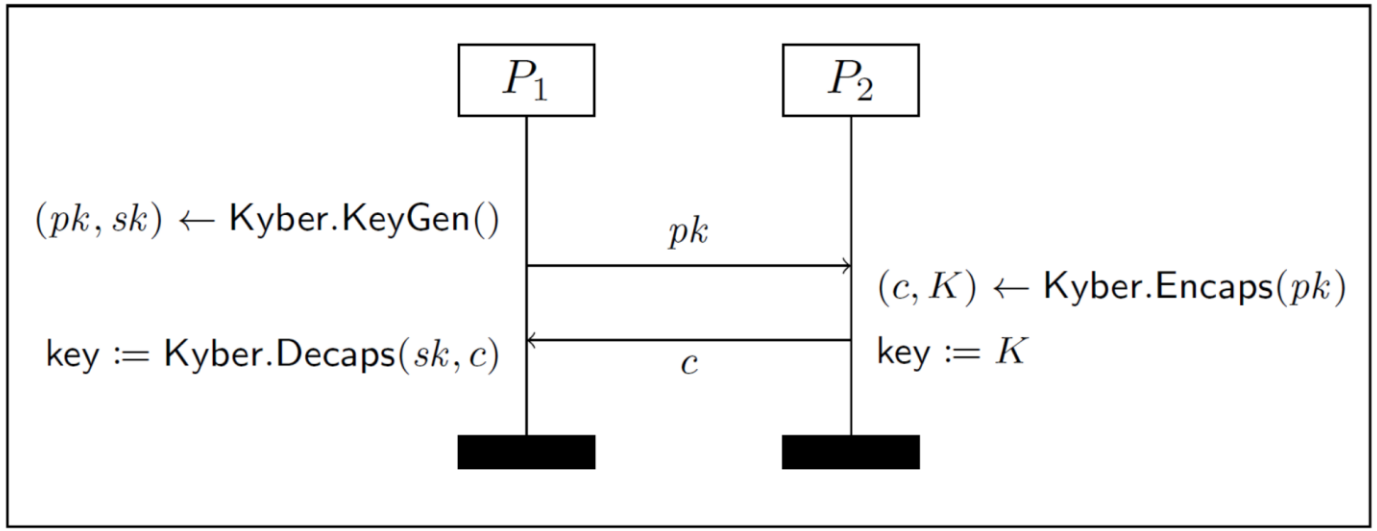
SVP-Beispiel
Basisvektoren (8, 5) und (2, 2)
SV = (-2,1) und (2,-1)

Kyber KEM: Vereinfachte Funktionsweise

Truncated Polynomial Ring

$$R = \mathbb{Z}_q[x]/(x^n+1)$$

Parameter:
n =256, q = 3389, k=2,3,4



Privater Schlüssel **sk**
sk $\in R^{k\times 1}$ mit kleinen Koeffs

Öffentlicher Key **pk=(A, t)**
Wähle **A** $\in R^{k\times k}$ zufällig mit beliebigen
Wähle **e** $\in R^{k\times 1}$ zufällig mit kleinen Koeffs
Berechne **t** = **A** $\cdot$ **sk** + **e** $\in R^{k\times 1}$

Verschlüsselung
Ciphertext c = (**u**,v), **u** $\in R^{k\times 1}$, v $\in$ R
Nachricht m ➔ binär $m_b \in R$ ➔ „gestreckt" $m_{bs} \in R$
Wähle $\mathbf{e_1}$, **r** $\in R^{k\times 1}$ zufällig mit kleinen Koeffs
Wähle $e_2 \in R$ zufällig mit kleinen Koeffs
Berechne **u** = $\mathbf{A}^T \cdot \mathbf{r} + \mathbf{e_1}$
Berechne v = $\mathbf{t}^T \cdot \mathbf{r} + e_2 + m_{bs}$

Entschlüsselung
Berechne $m_{noisy} = v - \mathbf{sk}^T \cdot \mathbf{u}$

Beseitige Fehler: m_{noisy} ➔ m_{bs}
„Kürzen": m_{bs} ➔ m_b
„Entbinärisieren": m_b ➔ m

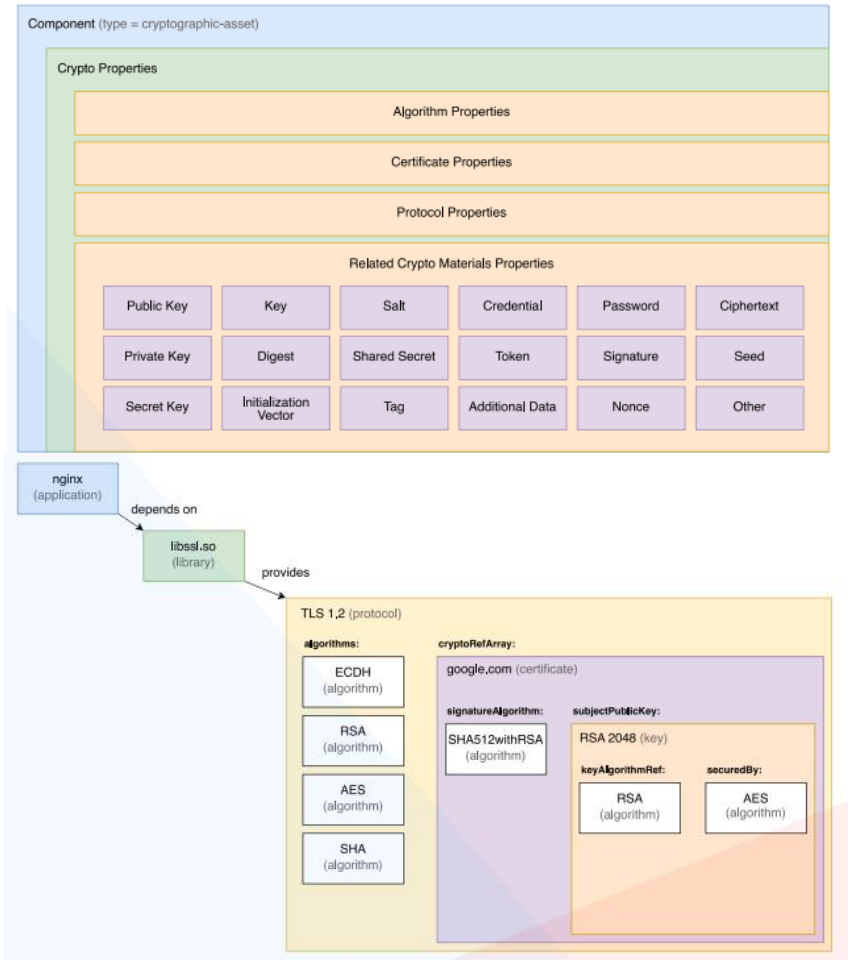
The Mosca Theorem
If your data needs protection for **X** years, your migration takes **Y** years, and quantum computers arrive in **Z** years, you need to start migrating when **X + Y > Z**. For most organizations, this means *now*.

NIST SP 1800-38B:
1. Establish a Quantum-Readiness Roadmap
2. Prepare a Cryptographic Inventory (CBOM)
3. Discuss Quantum Safe Roadmaps with Technology Vendors
4. Determine Supply Chain Quantum-Readiness

Gängige Krypto-Software beinhaltet PQK bereits wie:
- Signal
- TLS
- SSH

Herausforderungen:
- Einbettung von PQK-Schlüsseln in Zertifikate
- Alte Krypto fest „verlötet" in SW bzw. HW

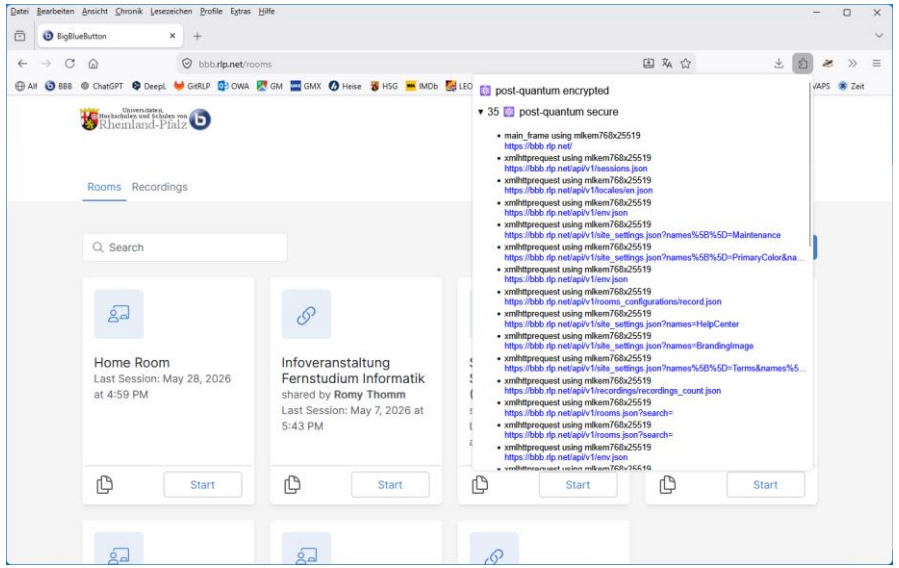
CBOM = Cryptographic Bill of Materials



Quelle:
https://cyclonedx.org/guides/OWASP_CycloneDX-Authoritative-Guide-to-CBOM-en.pdf

„PQC Readiness“ ausgewählter Web-Seiten

Web-Seite	PQ KEM	PQ Zertifikate
https://pqc.pqcrypta.com/pqc/	Ja	Ja
https://isitquantumsafe.info/	Ja	Nein
https://www.hochschule-trier.de/	Nein	Nein
https://studip.hochschule-trier.de/	Nein	Nein
https://vaps.hochschule-trier.de/	Nein	Nein
https://idp.fh-trier.de/	Nein	Nein
https://gitlab.rlp.net/	Ja	Nein
https://seafile.rlp.net/	Nein	Nein
https://bbb.rlp.net/	Ja	Nein
https://olat.vcrp.de/	Nein	Nein



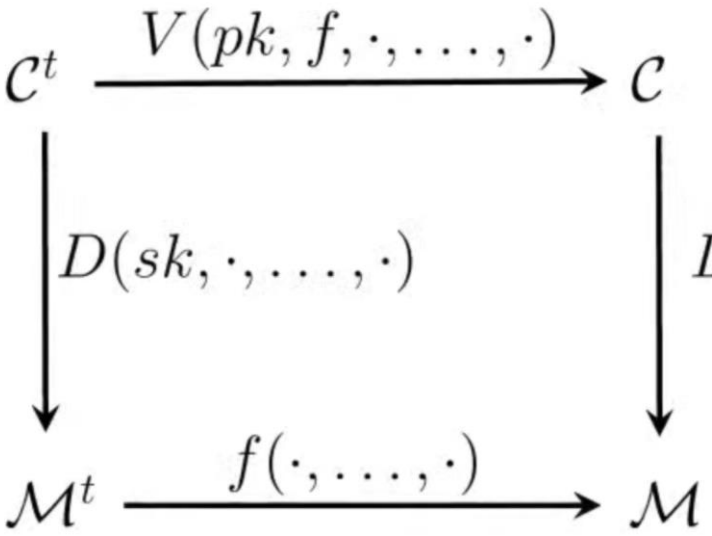
Getestet mit

- Firefox-Erweiterung PQSpy, https://github.com/bwesterb/pqspy
- Wireshark Filter
  `tls.handshake.extensions_key_share_group == 0x11EC` ➔ X25519MLKEM768

Homomorphe Verschlüsselung / Homomorphic Encryption (HE)

HE: Einführung

- HE erlaubt das Rechnen auf verschlüsselten Daten ➔ Vorteilhaft für vertrauliche, personenbezogene Daten
- HE ist eine Verallgemeinerung der Public Key Cryptography
- Das erste (sehr langsame) voll-homomorphe Verfahren stammt von Gentry aus dem Jahre 2009. Seitdem rasante Entwicklung mit erheblichen Beschleunigungen.
- Die meisten Verfahren basieren auf Gitterproblemen und gelten damit als resistent gegen Angriffe mit Quantencomputern.

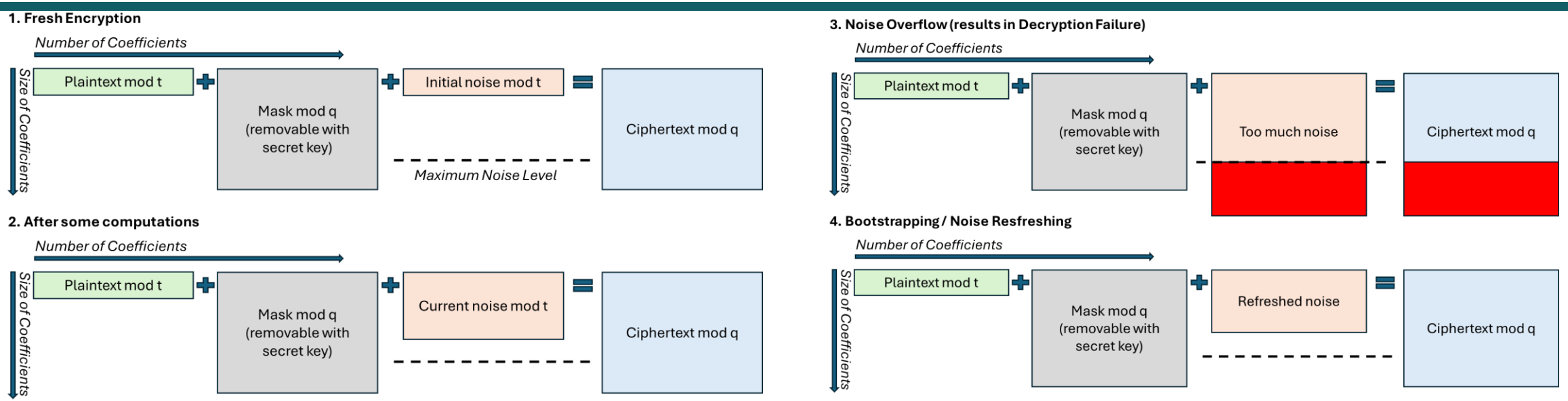


Legende:
C = Ciphertext
D = Decrypt
f = zu berechnende Funktion
M = Plaintext
pk = public key
sk = secret key
t = Dimension
V = Evaluierungsfunktion

HE-Typ	Beschreibung
Partial	Nur eine Operation wie + oder * wird unterstützt. Beispiele: RSA oder Paillier
Gestuft / Leveled	Mindestens zwei Operationen werden unterstützt aber nur bis zu einer bestimmten „Tiefe“ („multiplicative Depth“)
Voll / Fully	Beliebige Berechnungen auf verschlüsselten Daten sind möglich. Verwendet Bootstrapping zur Verringerung des Rauschens

Abb.-Quelle: Gentry, https://www.youtube.com/watch?v=487AjvFW1lk

Noise & Bootstrapping



Bootstrapping = Ver- und Entschlüsselung homomorph durchführen

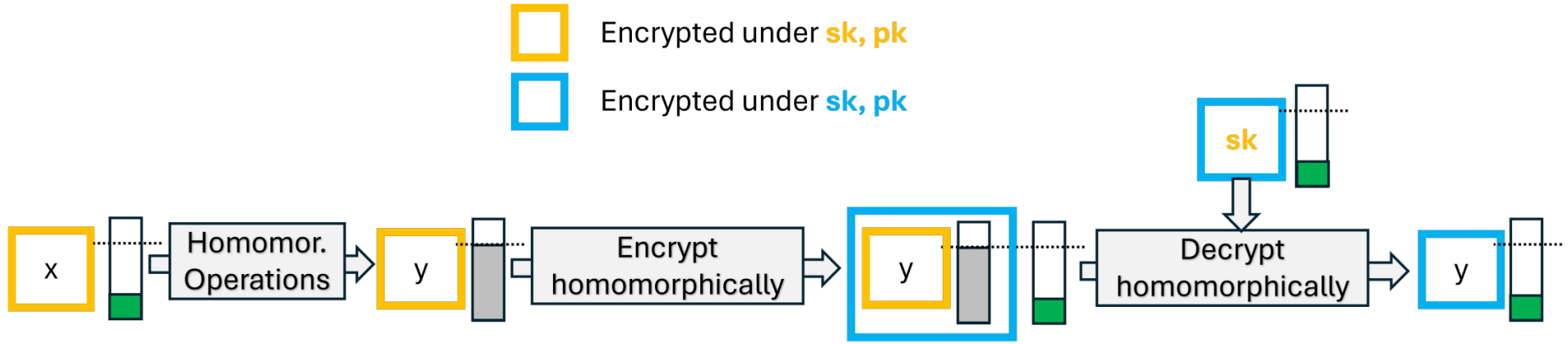
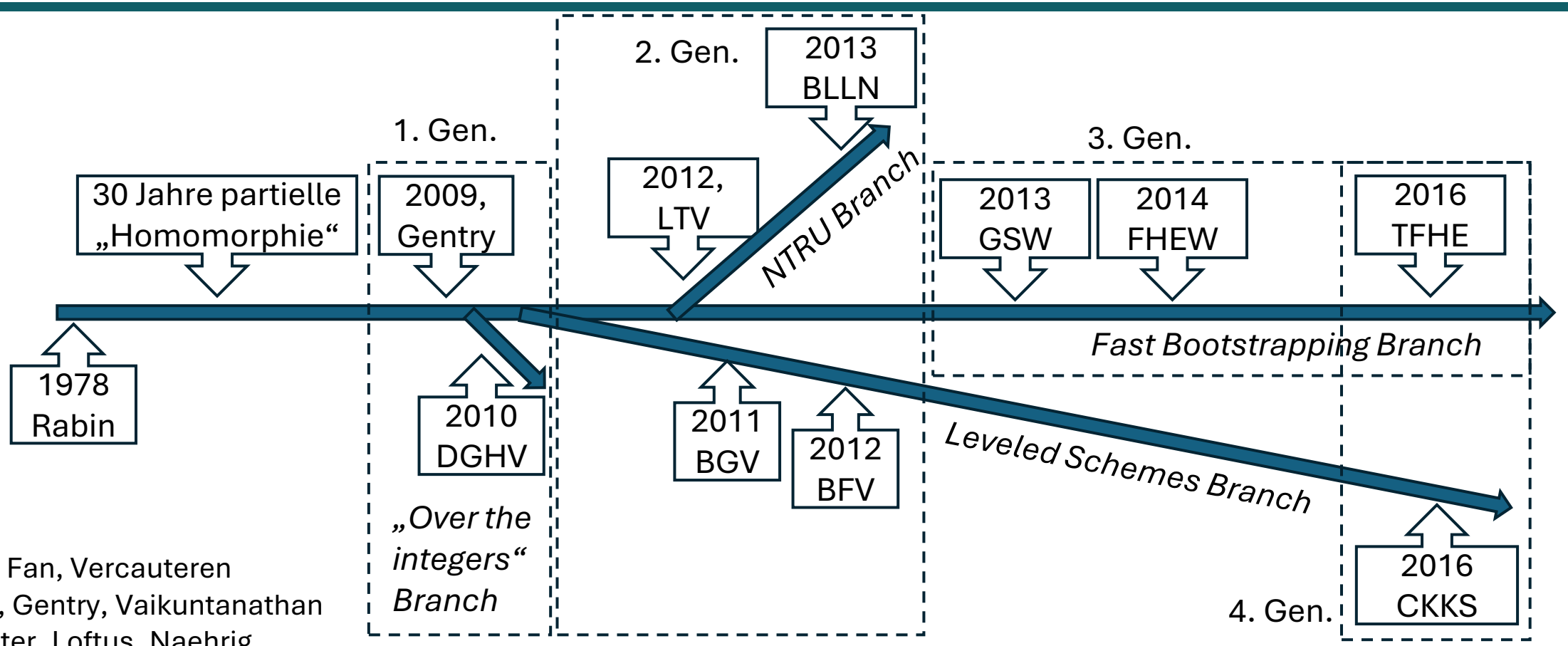


Abb.-Quelle: Paillier, https://www.youtube.com/watch?v=aruz58RarVA

Chronology of the FHE Zoo
based on Pascal Paillier, zama



Akronyme:
BFV = Brakerski, Fan, Vercauteren
BGV = Brakerski, Gentry, Vaikuntanathan
BLLN = Bos, Lauter, Loftus, Naehrig
CKSS = Cheon, Kim, Kim, Song
DGHV = van Dijk, Gentry, Halevi, Vaikuntanathan
FHEW = FHE in the West
GSW = Gentry, Sahai, Waters
TFHE = Torus based FHE
TLV = Tromer, Lopez-Alt, Vaikuntanathan

Schema	Arithmetik
BFV, BGV	(Genaue) modulare Arithmetik
CKSS	Approximative Arithmetik („Kommazahlen“)
TFHE	Boolesche Schaltkreise

SW-Bibliothek	HE-Verfahren	Programmier-sprache
SEAL	BFV, BGV, CKKS	C++
Zama	TFHE	Rust
openFHE	BFV, BGV, CKKS, TFHE	C++

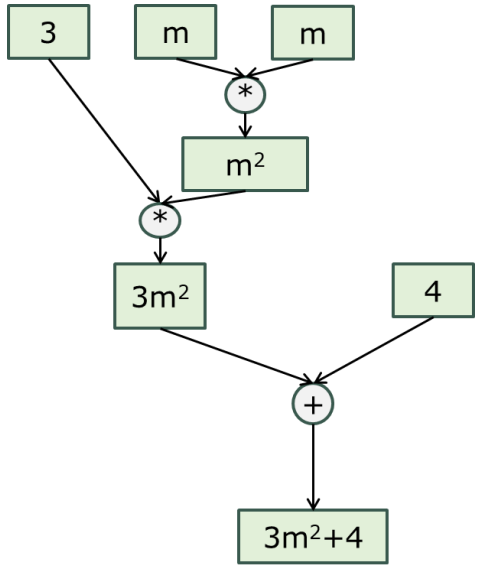
BFV-Schema

- BFV = Brakerski Fan Vercauteren, 2012
- Voll homomorph (in vielen Implementierungen aber nur “leveled”)
- Die Sicherheit von BFV basiert auf einem **Gitterproblem**, genauer dem **Ring Learning With Errors** Problem (verwandet mit MLWE).
- Berechnungen über Truncated Polynomial Ring, nur Berechnungen über Integern möglich

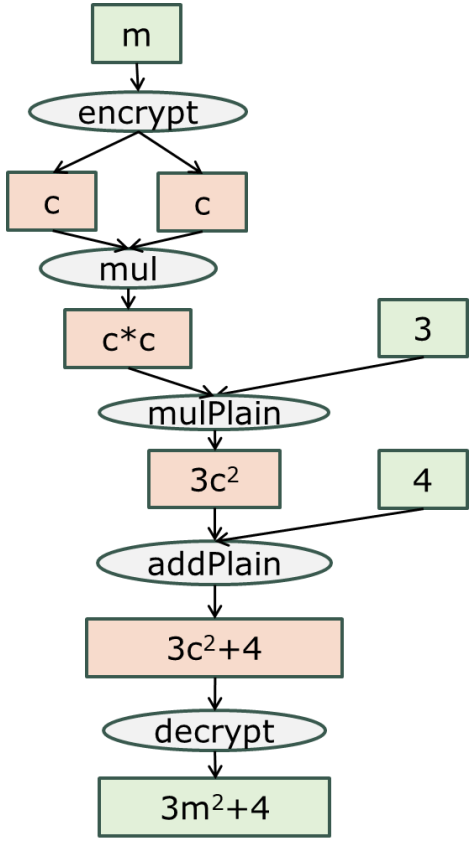
- BFV-Funktionen (Leveled Version):
  - `KeyGen()` (Public, Private, Relinearization)
  - `encrypt`
  - `decrypt`
  - `add`: Addition zweier Ciphertexte
  - `mult`: Multiplikation zweiter Ciphertexte
  - `addPlain`: Addition Plain- und Ciphertext
  - `mulPlain`: Multiplikation Plain- und Ciphertext

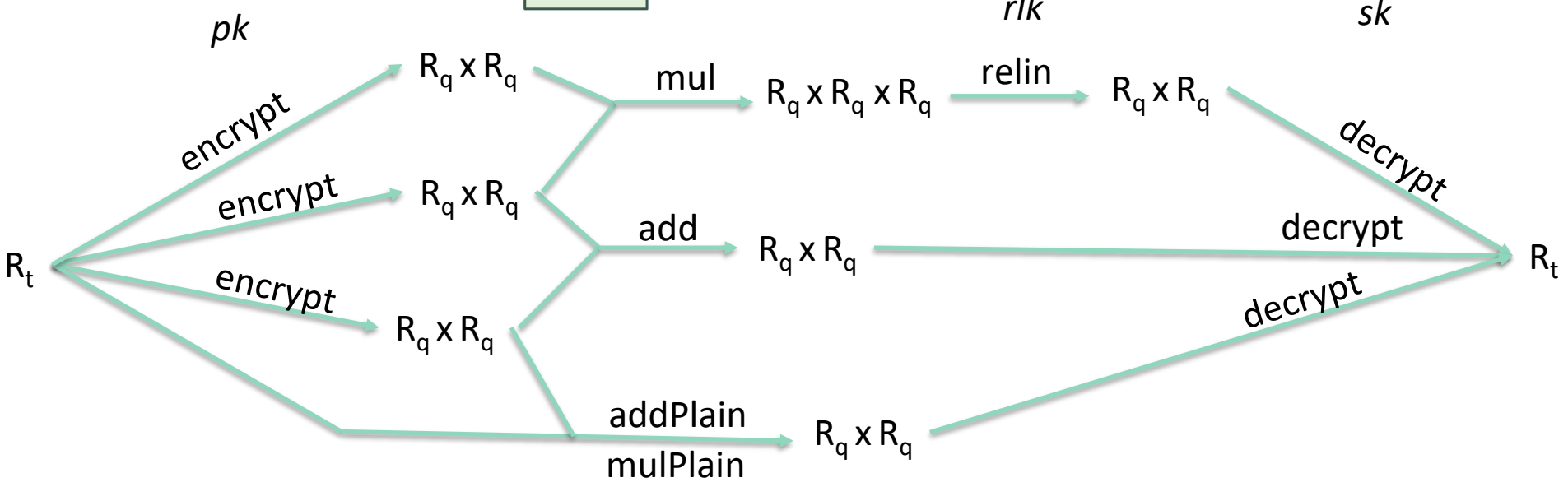
BFV-Berechnungsbeispiel für das Polynom $f(m) = 3m^2 + 4$

Berechnung im Klartext



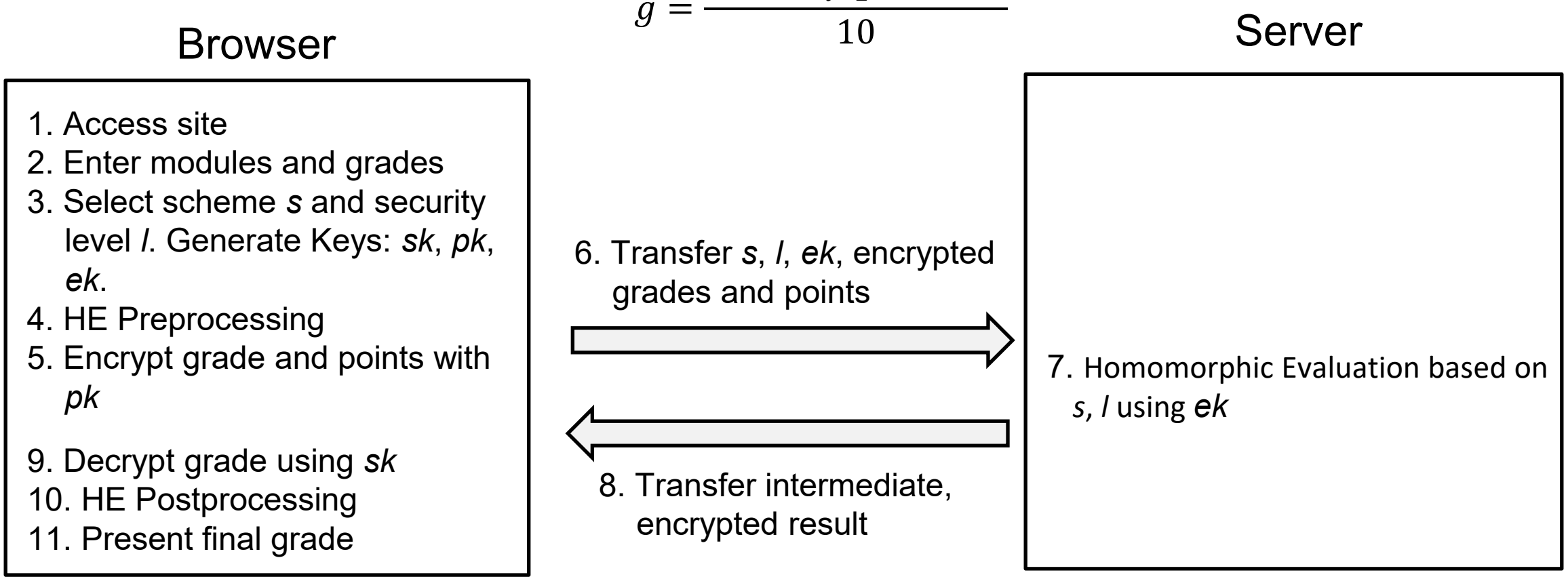
Homomorphe Berechnung





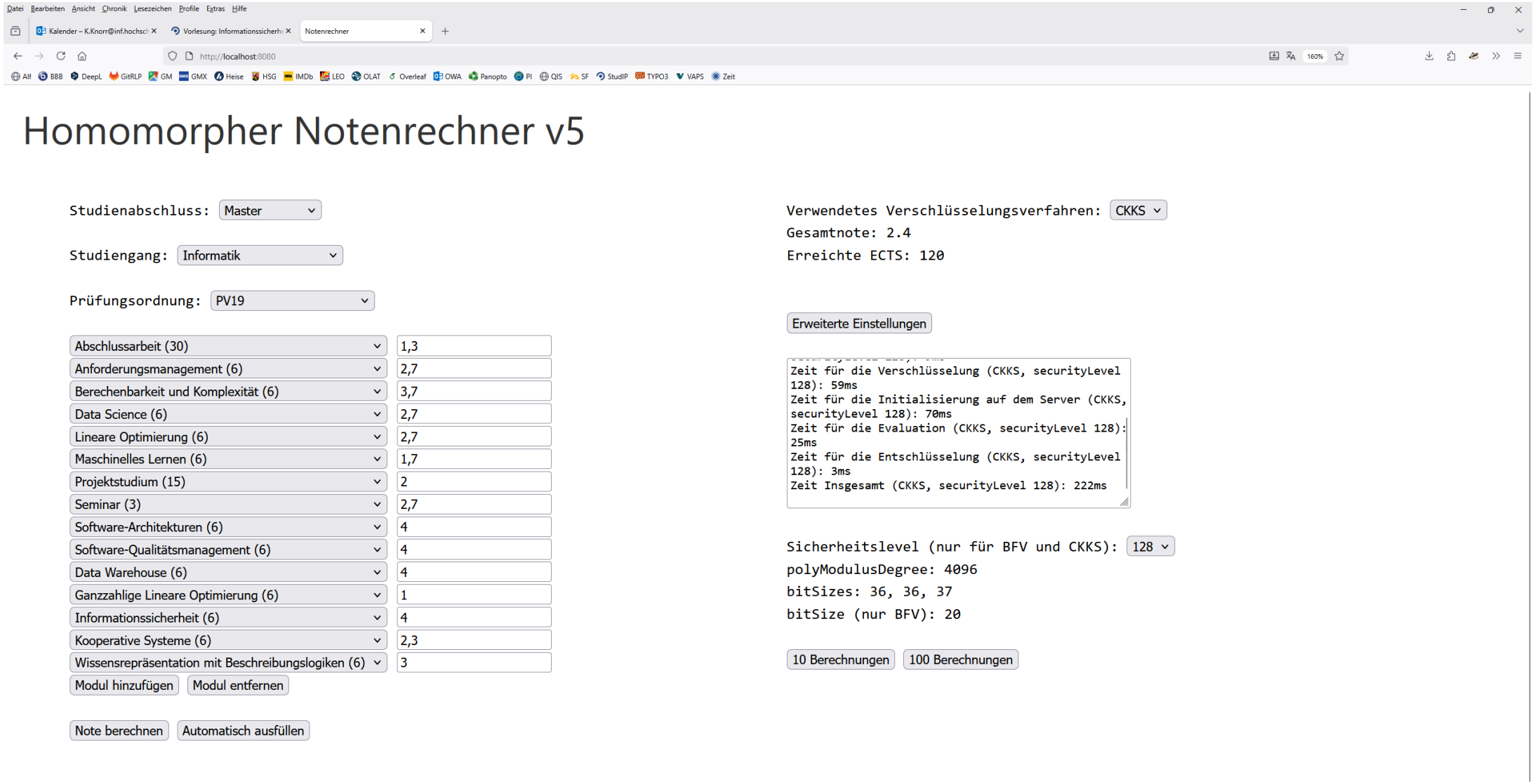
Die Endnote *g* auf dem Bachelor- bzw. Masterzeugnis ist das abgerundete, gewichtete ($w_i \in \{3, 5, 7, 12\}$ CPs) arithmetische Mittel der *n* Modulnoten ($m_i \in \{1.0, 1.3, 1.7, \ldots, 3.7, 4.0\}$):

$$g = \frac{\lfloor 10 \cdot (\sum_{i=1}^{n} w_i \cdot m_i) \rfloor}{10}$$



Quelle: Knorr & Müller

Demo of the Cryptographic Grader



- Erlaubt Berechnungen auf verschlüsselten Daten
  - Ideal für Anwendungen mit vertraulichen Daten z.B. im Gesundheitswesen
  - Einhaltung des Datenschutzes
  - Reduzierung des Vertrauensbedarfs (z.B. in Cloud-Szenarien)
- Die vorgestellten Verfahren gelten als resistent gegen Angriffe mit einem QC.
- Rechenaufwand und Speicherbedarf (Schlüssel, Ciphertext) gegenüber Klartextberechnungen immens (~50.000 im Beispiel des Homomorphic Graders). Neuere Verfahren deutlich schneller.
- Komplexe Verfahrens- und Parameterauswahl
- Kein Integritätsschutz in den vorgestellten Verfahren

Post-Quanten-Kryptologie

- FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard, https://csrc.nist.gov/pubs/fips/203/final
- FIPS 204: Module-Lattice-Based Digital Signature Standard, https://csrc.nist.gov/pubs/fips/204/final
- FIPS 205: Stateless Hash-Based Digital Signature Standard, https://csrc.nist.gov/pubs/fips/205/final
- BSI – Technische Richtlinie, Bezeichnung: Kryptographische Verfahren: Empfehlungen und Schlüssellängen, Kürzel: BSI TR-02102-1, Version: 2025-01
- Signal: https://signal.org/blog/spqr/
- TLS: Hybrid Handshake, https://www.ietf.org/archive/id/draft-reddy-uta-pqc-app-07.html, https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/
- Openssh, https://www.openssh.org/pq.html
- CBOM: https://cyclonedx.org/guides/OWASP_CycloneDX-Authoritative-Guide-to-CBOM-en.pdf
- NIST SPECIAL PUBLICATION 1800-38B, Migration to Post-Quantum Cryptography, Quantum Readiness: Cryptographic Discovery
- P. Shor: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, https://arxiv.org/abs/quant-ph/9508027
- Quantum Computing Threat Timeline: https://qramm.org/learn/quantum-threat-timeline.html
- Matthias Homeister: Quantum Computing verstehen, Springer, 2022 (als eBook in der Bibliothek vorhanden)

Homomorphe Verschlüsselung

- J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "Bootstrapping for approximate homomorphic encryption" in Advances in Cryptology – ASIACRYPT 2018, LNCS, vol. 11272, pp. 360–384.
- J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," Cryptology ePrint Archive, Paper 2016/421, 2016. Available: https://eprint.iacr.org/2016/421
- I. Chillotti, N. Gama, and M. Georgieva, "TFHE: Fast fully homomorphic encryption over the torus," Journal of Cryptology, vol. 33, no. 1, pp. 34–91, 2020, doi:10.1007/s00145-019-09319-x.
- J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," Cryptology ePrint Archive, Paper 2012/144, 2012. Available: https://eprint.iacr.org/2012/144.
- K. Knorr, D. Müller: Homomorphic Grade Calculations for Academic Study Programs, 20th International Technology, Education and Development Conference, Valencia, Spain. 2-4 March, 2026. ISSN: 2340-1079, doi: 10.21125/inted.2026
- L. Krier: "Konzeption und Erstellung eines Demonstrators für die Torus-basierte homomorphe Suche auf Gesundheitsdaten", MA, Hochschule Trier, 2024
- D. Müller, "Homomorphe Anwendungen im Hochschulkontext", MA, Hochschule Trier, 2025
- Homomorphic Encryption Standard, https://homomorphicencryption.org/standard/

Vielen Dank für die Aufmerksamkeit.

Fragen? Anmerkungen?

knorr@hochschule-trier.de