

Hochschule Wismar
University of Technology, Business and Design
Fakultät für Ingenieurwissenschaften
Bereich Elektrotechnik und Informatik



Masterthesis

POST - QUANTUM - CRYPTOGRAPHY

KRYPTOGRAPHISCHE LÖSUNGEN IM UMFELD VON QUANTENCOMPUTERN

Autor: Feridun Temizkan

Danksagung

Vor dem Einstieg in meine Masterthesis möchte ich meinem Arbeitgeber, der Bundesanstalt für den Digitalfunk der Behörden und Organisationen mit Sicherheitsaufgaben, sowie meinem Chef, Dr. Michael Pilgermann, danken. Sie ermöglichten mir die Teilnahme an einem Seminar zum Thema Post Quantenkryptographie mit Persönlichkeiten wie Tanja Lange und Daniel J. Bernstein wodurch ich Einblick in einen faszinierenden Bereich der Kryptographie erhielt.

Auch will ich in diesem Zuge meinen Betreuer Herrn Prof. Dr.-Ing. habil. Andreas Ahrens nicht vergessen. Seine Vorlesungen zum Thema Kryptographie und Kryptoanalyse fesselten mich vom ersten Moment und gaben mir eine fundierte Basis zum Schreiben dieser Arbeit. Auch für die Unterstützung und Betreuung über den gesamten Zeitraum bin ich sehr dankbar.

Nicht unerwähnt bleiben soll Jessica Wohlleben. Ich danke ihr für konstruktive Anregungen, Hinweise und die kritische Bewertung meiner Arbeit. Sie hat mir sehr geholfen und mich durch inhaltlichen Austausch und Diskussionen vorangebracht.

Abschließend möchte ich die größte Dankbarkeit gegenüber meiner Kommilitonin Caroline Schlieper zum Ausdruck bringen. Noch nie hat mir jemand so viel und so geduldig geholfen wie sie. Ihre unermüdliche und zeitintensive Unterstützung werde ich ihr nie vergessen und danke ihr im besonderen Maße.

Feridun Temizkan

Berlin, der 28. August 2019

Aufgabenstellung

Aktuelle Computer-Systeme arbeiten auf Basis klassischer Physik. Verschlüsselungen basieren auf symmetrischen und asymmetrischen Verfahren oder einer Kombination, den hybriden Verschlüsselungsverfahren. Die Sicherheit der asymmetrischen kryptographischen Systeme beruht hierbei auf mathematischen Problemen, der Schwierigkeit der Primfaktorzerlegung und der Berechnung diskreter Logarithmen. Asymmetrische kryptographische Verfahren können theoretisch durch den Shor-Algorithmus, der Mittel der Quanteninformatik verwendet, in polynomieller Laufzeit gelöst werden [1]. Das bedroht insbesondere Schlüsselaustauschverfahren wie Diffie-Helman, Verschlüsselung mittels RSA, El-Gamal sowie Verfahren basierend auf elliptischen Kurven. Auch wären analog funktionierende Signaturverfahren nicht mehr vertrauenswürdig [2] [3].

Diese Masterthesis wird die Möglichkeiten der Quanteninformatik bezogen auf bestehende kryptographische Verfahren untersuchen und die daraus resultierenden Bedrohungen belegen. Hierfür werden zunächst die gegenwärtige Kryptographie und ihre Sicherheit untersucht. Anschließend sollen Möglichkeiten und Chancen der Quantenmechanik betrachtet werden.

Schwerpunkt liegt auf der Untersuchung alternativer kryptographischer Primitive und Problemstellungen, die auch beim Einsatz eines Quantencomputers nur schwer lösbar sind. Ein besonderer Fokus liegt auf Schlüsselaustausch- und asymmetrischen Verschlüsselungsverfahren.

Abschließend sollen konkrete Möglichkeiten aufgezeigt werden, um auch künftig die Vertraulichkeit von Informationen gewährleisten zu können. Diese Handlungsempfehlungen müssen dabei sowohl heutige als auch künftige Sicherheit, zumindest nach aktuellem Stand der Forschung, gewährleisten können.

Kurzbeschreibung

Die Arbeit an der Masterthesis hat gezeigt, dass die Kryptographie künftig komplexer wird. Um die Bedrohungen aktuell eingesetzter kryptographischer Verfahren zu verstehen werden zunächst gegenwärtige kryptographische Systeme betrachtet. Anschließend werden die Bedrohung asymmetrischer Verfahren durch den Quantencomputer, insbesondere durch den Algorithmus von Peter Shor, vorgestellt. Dieser bedroht, bei ausreichend starken Quantencomputern, die asymmetrische Verschlüsselung, da er in der Lage ist das Faktorisierungsproblem in Polynomialzeit zu lösen, sowie in einer abgewandelten Version auch den diskreten Logarithmus. Als Antwort wird die Post Quantenkryptographie präsentiert. Darüber hinaus bietet die Quantenmechanik auch kryptographische Möglichkeiten, die sogenannte Quantenkryptographie. Basierend auf den sich ergebenden Kenntnissen werden Möglichkeiten und Handlungsempfehlungen ausgesprochen, anhand welcher nach heutigem Stand des Wissens Sicherheit bezüglich Quantencomputer geschaffen werden kann.

Abstract

The work on the Masterthesis has shown that cryptography will become more complex. First of all current cryptographic systems are considered to understand the thread of currently used cryptographic methods. Subsequently, the threat of asymmetric methods by the quantum computer, in particular by the algorithm of Peter Shor, will be presented. This algorithm threatens the asymmetric encryption with sufficiently strong quantum computers. It can solve the factorization problem and the discrete logarithm in polynomial time. As a solution of this problem, the post-quantum-cryptography is presented. In addition, quantum mechanics also offers cryptographic possibilities, the so-called quantum cryptography. Based on the resulting knowledge possibilities and recommendations for action are given, with which it is possible to create security regarding quantum computers according to the current state of knowledge.

Inhaltsverzeichnis

KURZBESCHREIBUNG.....	3
1 EINLEITUNG.....	9
1.1 MOTIVATION UND VORGEHENSWEISE	9
1.2 ABGRENZUNG	10
1.3 ZIELSETZUNG.....	11
2 VERSCHLÜSSELUNG IN KLASSISCHEN SYSTEMEN	12
2.1 ZIELSTELLUNGEN DER KRYPTOGRAPHIE	12
2.2 KLASSISCHE RECHNERSYSTEME UND ALGORITHMEN	13
2.3 P, NP UND NP-VOLLSTÄNDIGKEIT	15
2.4 VERSCHLÜSSELUNGSVERFAHREN	17
2.4.1 SYMMETRISCHE VERSCHLÜSSELUNGSVERFAHREN.....	18
2.4.2 ASYMMETRISCHE VERSCHLÜSSELUNGSVERFAHREN.....	21
2.5 ZUSAMMENFASSUNG	25
3 DER QUANTENCOMPUTER	27
3.1 CHARAKTERISTIKA VON QUANTENCOMPUTERN	27
3.2 BERECHNUNGEN AUF EINEM QUANTENCOMPUTER	29
3.2.1 SHOR-ALGORITHMUS	31
3.2.2 GROVER-ALGORITHMUS.....	33
3.3 ZUSAMMENFASSUNG	34
4 POST QUANTENKRYPTOGRAPHIE	36
4.1 WAS IST POST QUANTENKRYPTOGRAPHIE?	36
4.2 QUANTENKRYPTOGRAPHIE	36
4.2.1 DAS BB84-PROTOKOLL.....	38
4.2.2 DAS E91-PROTOKOLL	42
4.2.3 BEWERTUNG VON QUANTUM-KEY-DISTRIBUTION-VERFAHREN.....	44
4.3 QUANTENCOMPUTERRESISTENTE VERFAHREN.....	46

4.3.1	ALLGEMEIN	46
4.3.2	HASHBASIERTE KRYPTOGRAPHIE.....	47
4.3.3	GITTERBASIERTE KRYPTOGRAPHIE	52
4.3.4	CODEBASIERTE KRYPTOGRAPHIE	57
4.3.5	MULTIVARIANTE KRYPTOGRAPHIE.....	60
4.3.6	SUPERSINGULARE ISOGENIEBASIERTE KRYPTOGRAPHIE	62
5	STANDARDS IM BEREICH DER KRYPTOGRAPHIE	66
5.1	ENTWICKLUNG VON STANDARDS	66
5.2	FLEXIBILITÄT BEI KRYPTOGRAPHISCHEN LÖSUNGEN.....	68
5.3	HERAUSFORDERUNGEN FÜR STANDARDS	71
5.4	STANDARDISIERUNGS-WETTBEWERB DER NIST	73
5.5	PQCRYPTO	76
5.6	IPSEC UND INTERNET KEY EXCHANGE IKE	77
5.6.1	AUFGABEN DES INTERNET KEY EXCHANGE IKE	77
5.6.2	IKE QUANTENCOMPUTER-RESISTENT GESTALTEN	78
5.7	TRANSPORT LAYER SECURITY TLS	82
5.7.1	KRYPTOGRAPHISCHE ANSÄTZE AUF DER TRANSPORT-SCHICHT.....	82
5.7.2	EXPERIMENTELLER EINSATZ VON NEWHOPE IN GOOGLE CHROME	82
5.7.3	QUANTUM-SAFE HYBRID (QHS) KEY EXCHANGE.....	83
5.7.4	OPEN QUANTUM SAFE-PROJECT OQS	86
5.7.5	DIGITALE SIGNATUREN UND HYBRIDE KEM	87
6	REFLEXION.....	91
6.1	ZUSAMMENFASSUNG	91
6.2	BEWERTUNG	92
6.3	FAZIT	94
	LITERATURVERZEICHNIS.....	95
	BILDERVERZEICHNIS	101

TABELLENVERZEICHNIS	102
ANLAGEN	103
ABKÜRZUNGSVERZEICHNIS	104
THESEN ZUR MASTERARBEIT.....	108
SELBSTÄNDIGKEITSERKLÄRUNG.....	109

1 Einleitung

1.1 Motivation und Vorgehensweise

Die Masterthesis wird das vorliegende Thema theoretisch behandeln und die Bedrohung durch den Quantencomputer, basierend auf dem aktuellen Stand der Forschung. Dabei bedient sich die Arbeit aller relevanten Bereiche und soll so als Grundlage für einen fundierten Einstieg in die Thematik, aber auch für weitere Forschungen, zur Verfügung stehen.

Die Thesis gliedert sich in fünf Abschnitte. Der erste Abschnitt dient als Einleitung und Darstellung der aktuellen Ausgangssituation. Er wird die Kryptographie im Bereich der Kommunikation sowie symmetrische und asymmetrische Verfahren vorstellen. Auch soll die Authentizität durch Signaturen, basierend auf Public-Key-Verfahren, erläutert werden und wieso diese auf klassischen Systemen als sicher betrachtet werden können.

Um zu verstehen wieso diese Sicherheit künftig gefährdet ist muss zunächst ein Verständnis für die Funktionsweise und so die Möglichkeiten von Quantencomputern geschaffen werden. Der zweite Abschnitt widmet sich dem Quantencomputer und seiner Bedeutung für die Kryptoanalyse. Hier soll insbesondere auf die ausgewählten Algorithmen von Peter Shor und Lov Grover eingegangen werden.

Abschnitt drei geht auf die Verteilung von Schlüsseln in der Post-Quantum-Ära ein. Schwerpunkt liegt hier auf vielversprechenden quantencomputer-resistenten Verfahren, um die Schlüsselverteilung und die Authentizität weiterhin zu gewährleisten. Es sind alternative Probleme der Mathematik, eingeteilt in verschiedene Familien, die sowohl klassische als auch Quantencomputer vor Probleme stellen und so eine künftige Sicherheit in Aussicht stellen.

Anschließend erfolgt die Ausarbeitung von Empfehlungen für noch nicht standardisierte Verfahren. Es soll die Komplexität der Kryptographie über die reinen Primitiven hinaus aufgezeigt werden. Lange Laufzeiten in der Entwicklung und Standardisierung, eine große Anzahl an verschiedenen Bedarfen sowie

viele verschiedene Parteien müssen berücksichtigt werden. Es muss klar sein, dass das Thema hoch aktuell ist und nicht mehr in der Zukunft liegt.

Der letzte Abschnitt beinhaltet eine abschließende Betrachtung, Zusammenfassung und Beurteilung der Post Quantenkryptographie. Es soll festgestellt werden, ob alle Fragestellungen beantwortet werden konnten.

1.2 Abgrenzung

Nicht Bestandteil dieser Arbeit ist die Erläuterung der Arbeitsweise von Quantencomputer im Detail oder eine Erklärung der Gesetze der Quantenmechanik. Durch den Algorithmus von Shor, implementiert auf einem Quantencomputer, sind die mathematischen Probleme der Faktorisierung und der Berechnung des diskreten Logarithmus lösbar. Diese Masterthesis wird sich primär auf die Bedrohung dieser Verfahren fokussieren.

Symmetrische Verfahren werden durch den Quantenalgorithmus von Grover in geringerer Weise bedroht als asymmetrische Verfahren durch den Shor-Algorithmus. Die Geschwindigkeit eines Angriffs reduziert sich um die Quadratwurzel, was in der Praxis bedeuten würde, dass die Länge eines Schlüssels von z.B. 128 Bit auf 64 Bit in der Sicherheit reduziert würde. Die Vergrößerung des Schlüssels wäre ein adäquates Mittel, dieser Bedrohung entgegenzuwirken, so dass dieses Thema nur sekundär behandelt wird.

Des Weiteren soll nicht auf Side Channel Attacks, SCA, der vorgestellten kryptographischen Verfahren eingegangen werden, welche für die Post Quantenkryptographie eine Rolle spielen. Diese Methode Algorithmen anzugreifen richten sich weniger gegen Algorithmen an sich als gegen die physischen Implementierungen. Algorithmen werden lediglich von ihrer mathematischen Seite aus betrachtet.

Beispiele beziehen sich primär auf Kommunikation, die Verschlüsselung von z.B. Datenträgern wird nicht explizit betrachtet oder erwähnt.

1.3 Zielsetzung

Ziel dieser Masterthesis ist die Darstellung der Auswirkungen des Quantencomputers auf symmetrische und asymmetrische kryptographische Verschlüsselungsverfahren. Das betrifft insbesondere Verfahren zum Austausch eines Schlüssels bei einer Kommunikation sowie Signaturverfahren. Dabei soll ein Bewusstsein dafür geschaffen werden, dass bereits heute die Vertraulichkeit bedroht ist, selbst wenn es noch Jahre dauern wird, bis ein leistungsstarker Quantencomputer zur Verfügung steht. Die Masterthesis soll Antworten auf folgende Fragen finden:

- Wie sind herkömmliche Verfahren klassischer Systeme im Umfeld von Quantencomputern zu bewerten?
- Welche alternativen Verfahren gibt es für einen Schlüsselaustausch?
- Mit welchen Verfahren kann die Authentizität bei Signaturverfahren gewährleistet bleiben?
- Auf welchem Stand ist die Forschung auf dem Gebiet der Post Quantenkryptographie?

Ziel ist das Finden einer angemessenen Empfehlung, um das Schutzziel der Vertraulichkeit bei einer Kommunikation aufrecht zu erhalten und die Authentizität von Nachrichten weiterhin gewährleisten zu können, sollte der Quantencomputer für diese Einsatzbereiche zur Verfügung stehen. Dabei muss auch die gegenwärtige Situation berücksichtigt werden.

2 Verschlüsselung in klassischen Systemen

2.1 Zielstellungen der Kryptographie

Die Kryptographie stellt die Antwort auf die Bedrohung von Vertraulichkeit, d.h. der Offenlegung von Informationen in einer Kommunikation, dar und ermöglicht darüber hinaus die Sicherstellung abgeleiteter Schutzziele. Dabei geht sie stets vom Worst-Case-Szenario aus. In diesem ist der Angreifer Mallory sehr intelligent und hat beliebig Zugriff auf Übertragungsmedien. Der naive Ansatz der Security-by-Obscurity, also Sicherheit durch das Verschleiern des Vorgehens, soll so von vorne herein ausgeschlossen werden [4]. Die intuitiven Ziele der Kryptographie sind es Maßnahmen zu ergreifen, so dass

- Mallory mit abgefangenen Daten nichts anfangen kann,
- Mallory übertragene Daten nicht unbemerkt verändert,
- Mallory sich nicht unbemerkt als Sender, Alice, gegenüber dem Empfänger, Bob, ausgeben kann oder anders herum,
- aber auch, dass Alice nicht behaupten kann eine Nachricht wäre vom Angreifer, Mallory, an Bob geschickt worden und nicht von ihr.

Diese Szenarien werden als Schutzziele der Vertraulichkeit, Integrität, Authentizität und Nichtabstreitbarkeit bezeichnet [5]. Hierbei stehen sich Effizienz und das Problem der Schlüsselverteilung gegenüber und führen letztendlich zu den beiden Lösungsansätzen der symmetrischen und asymmetrischen Verschlüsselung wobei asymmetrische Verfahren das deutlich jüngere Gebiet darstellen [5]. Während letztere einen großen Teil der Schutzziele abdeckt, sind symmetrische Verfahren für gewöhnlich effizienter. Kryptographische Verfahren verhindern nicht, dass Daten abgefangen und geändert werden können. Auch wird eine Störung von Kommunikation nicht unterbunden. Jedoch werden sie entdeckt und eine entsprechende Reaktion ist möglich [6] [4].

2.2 Klassische Rechnersysteme und Algorithmen

Klassische Rechnersysteme sind Computer, die auf dem Referenzmodell der Von-Neumann-Architektur basieren. Sie basieren auf der klassischen Physik, die explizit die Quantenmechanik ausnimmt [7]. Von Neumann entwickelte ein Konzept für Rechenmaschinen, bei denen Programme nicht fest als Hardware verschaltet gewesen sind. Mit Hilfe dieser Architektur ist es möglich Turingmaschinen¹ praktisch zu realisieren. Das Leitbild der Von-Neumann-Architektur formuliert die Grundstruktur des modernen Computers [8, p. 129]. Jedes vom Computer ausgeführte Programm muss auf einem Medium gespeichert sein, das der Computer verstehen kann:

“[...] These instructions must be given in some form which the device can sense: Punched into a system of punchcards or on teletype tape, magnetically impressed on steel tape or wire, photographically impressed on motion picture film, wired into one or more fixed or exchangeable plugboards - this list being by no means necessarily complete. All these procedures require the use of some code to express the logical and the algebraical definition of the problem under consideration, as well as the necessary numerical material [...]” [9, p. 1].

Ein Computer, basierend auf der Von-Neumann-Architektur, ist in der Lage zusammengefasste und formalisierte Operationen automatisch durchzuführen. Diese werden eingelesen, verarbeitet und ausgegeben und haben so Anspruch an Ressourcen in Form von Hardware und Zeit. Die Verarbeitung und die Ausgabe hängen von den eingelesenen Informationen ab und sind als Algorithmus genau bestimmt.

Die Turingmaschine ist ein Modell der theoretischen Informatik. Alan Turing führte sie 1936 ein, um die Frage der fundamentalen Grenze mathematischer Beweisbarkeit aufzuwerfen. Es ist eine formale Definition der Berechenbarkeit von Funktionen und deren Zeit- und Ressourcenbedarf. Sie ist ein sehr einfaches, abstraktes Modell eines Computers, was jedoch mächtig genug ist, alles

¹ Siehe Anlage „A2 Turingmaschinen und Algorithmen“

zu berechnen, was berechenbar ist [10]. Turingmaschinen formalisieren Algorithmen und sind mathematische Objekte bestehend aus Zuständen, Eingabe- und Bandalphabet sowie Übergangsfunktionen. Hierbei entstehen Analogien zur Arbeitsweise eines Menschen bei Berechnungen von Problemen [11]. In genau definierten Einzelschritten wird die Eingabe schrittweise eingelesen, verändert und ausgegeben. Der verwendete, dynamische Speicher ist ein in Zellen unterteiltes, nach zwei Seiten hin unendliches Band und besitzt einen Lese-/Schreibkopf. Der Speicher hält die Eingabe und das Ergebnis der Berechnung als Ausgabe. Turingmaschinen eignen sich für den Einsatz in der Komplexitätstheorie und welche Probleme mit einer Maschine berechenbar sind. Es kann untersucht werden welche Probleme entscheidbar sind, d.h. welche Probleme durch einen Algorithmus gelöst werden können und welche nicht (Beispiel des Halteproblems²). Eine Turingmaschine entspricht hierbei einem Algorithmus bzw. Programm [10]. Algorithmen ermöglichen das Lösen von Berechnungen, Such- und Optimierungsproblemen.

Probleme, die auf einer deterministischen Turingmaschine gelöst werden können, werden der Komplexitätsklasse P zugeordnet, d.h. sie sind effizient in Polynomialzeit lösbar [10, p. 172]. Für eine große Anzahl praktischer Probleme, die nicht dieser Klasse zugewiesen werden konnten, konnte gezeigt werden, dass diese auf einer nichtdeterministischen Turingmaschine in Polynomialzeit gelöst werden können. Es gibt jedoch keine Realisierung einer solchen nichtdeterministischen Turingmaschine [10]. Die nichtdeterministische Turingmaschine bietet die Möglichkeit mehrerer Rechenschritte, die getätigt werden können und darf in jedem Schritt zwischen mehreren Alternativen beliebig wählen. Zu einer Eingabe gibt es damit nicht nur eine, sondern mehrere Rechnungen, von denen durchaus manche akzeptierend und andere verwerfend sein können [10, p. 173] [6]. Mit ihnen können Probleme effektiv, jedoch nicht effizient gelöst werden. Es handelt sich um die Komplexitätsklasse NP . Eine Problemstellung ist u. A. das Faktorisierungsproblem [10, p. 169]. Man kann sich das auch so vorstellen, dass eine nichtdeterministische Turingmaschine

² Siehe Anlage „A3 Das Halteproblem“

zunächst ein Element des zur Eingabe gehörenden Suchraums rät und dann deterministisch die Korrektheit überprüft [10, p. 174].

Eine universelle Turingmaschine U ist eine Maschine, welche eine bestimmte Turingmaschine T simulieren bzw. dasselbe Programm ausführen kann. Somit kann die universelle Turingmaschine U alle ihre übergebenen Turingmaschinen T mit einem Eingabewort w berechnen.



Bild 2.1 Schematische Darstellung einer universellen Turingmaschine U mit Ein- und Ausgabeparametern.

Eine solche universelle Turingmaschine entspricht in ihrer Funktion heutigen Computern. Diese können genauso neue Programme entgegennehmen und so Eingaben verarbeiten.

2.3 P, NP und NP-Vollständigkeit

Ziel von Algorithmen ist es so effizient wie möglich zu sein. Algorithmen mit langer Laufzeit sind nicht praxistauglich, da sie einen hohen Bedarf an Rechenzeit wie Speicherplatz haben, der, im Gegensatz zur Annahme bei der Turingmaschine, endlich ist. Hierfür wird das Laufzeitverhalten von Algorithmen bzw. das Laufzeitverhalten von Problemen untersucht. Die Komplexitätstheorie ist ein Teilgebiet der theoretischen Informatik, die sich mit der Frage beschäftigt wie komplex ein Problem und somit ein Algorithmus ist, mit dem das Problem gelöst werden soll. Sie klassifiziert alle lösbaren Probleme in die Menge der effizient lösbaren Probleme sowie der inhärent, also aus sich selbst heraus, schwierigen Probleme. Man untersucht das Verhalten eines Algorithmus in Abhängigkeit von der Problemgröße. Die Komplexitätstheorie interessiert sich für die Frage, wie viel Mehrarbeit für wachsende Problemgrößen notwendig sind. Steigt der Aufwand linear, polynomiell, exponentiell oder noch

stärker an (z.B. Problem des Handlungsreisenden³). Bei der Untersuchung von Laufzeitverhalten ist jedoch die Untersuchung konkreter Problemlösung nicht im Zentrum der Betrachtung. Je nach technischer Entwicklung können sich hier die Bedingungen sehr schnell ändern (z.B. die Sicherheit von DES und die vollständige Schlüsselsuche). Man versucht das Grenzverhalten von Problemen mit ihren Funktionen zu beschreiben und zu klassifizieren. Hierbei werden die untere und obere Schranke betrachtet, um den Ressourcenverbrauch eines Algorithmus bestimmen zu können. So wird z.B. bei Verschlüsselungsverfahren der Nachweis versucht, dass der erwartete Ressourcenverbrauch jedes Maß übersteigt, um einen Schlüssel zu knacken (Transcomputationales Problem). Für die Lösung von Problemen sind die Komplexitätsklassen P und NP interessant sowie die Probleme, die vollständig für die Klasse NP sind.

Die Komplexitätsklasse P ist die Klasse der Probleme, für die es einen Polynomialzeit-Algorithmus gibt. Beispiele für Probleme in P sind Suchprobleme, Sortierprobleme, das Finden des größten gemeinsamen Teilers, mit Laufzeit $\mathcal{O}(x^n)$ mit $n \in \mathbb{N}$. Man bezeichnet ein Problem als in Polynomialzeit lösbar, wenn die Rechenzeit einer deterministischen Rechenmaschine mit der Problemgröße nicht stärker als mit seiner Polynomfunktion oder ganzrationale Funktion wächst. Die Polynomfunktion ist die Summe von Potenzfunktionen mit natürlichem Exponenten. Potenzfunktionen sind elementare mathematische Funktionen der Form:

$$f: x \rightarrow ax^n \text{ mit } n \in \mathbb{N} \quad (2.1)$$

$$\begin{aligned} f(x) &= a_n x^n + a_{n-1} x^{n-1} + \dots + a_2 x^2 + a_1 x + a_0 \quad (2.2) \\ &= \sum_{i=0}^n a_i x^i, a_n \neq 0, a_i \in \mathbb{R} \end{aligned}$$

Sie stellt die Grenze zwischen praktisch lösbaren und nicht lösbaren Problemen dar. Der Aufwand, der nicht lösbaren Probleme wächst im Allgemeinen

³ Siehe Anlage „A4 Das Problem des Handlungsreisenden“

schneller und überschreitet damit die Möglichkeiten klassischer Rechnersysteme bezüglich Speicherverbrauch und Geschwindigkeit⁴. Ein Umstand, der in der asymmetrischen Kryptographie ausgenutzt wird.

Gemessen an der Größe der Eingabe n eines Problems gibt es, für z.B. eine deterministische Turingmaschine, eine Rechenzeit m , die der zugrundeliegende Algorithmus benötigt, um das Problem zu lösen. Es gibt also eine Konstante $k \geq 1$ mit $m(n) = \mathcal{O}(n^k)$. Mit Hilfe dieser Landauer-Notation wird das Grenzwertverhalten der Funktion beschrieben. $f \in \mathcal{O}(n^m)$ wächst polynomiell, was bedeutet, dass f auf ungefähr das 2^m -fache vergrößert, wenn sich n verdoppelt. Die NP -Klasse enthält Entscheidungsprobleme, deren Verifikation effizient in Polynomialzeit möglich ist. Dazu zählt das Testen, ob Primzahlen die Faktoren einer anderen Zahl sind. Man kann für diese Probleme einen Algorithmus finden, der das Problem in polynomieller Zeit löst, für dessen Ausführung jedoch ein nichtdeterministischer Computer nötig ist [10] [12].

Ein Problem L heißt NP -Vollständig, wenn es in NP liegt und jedes Problem innerhalb von NP in Polynomialzeit auf L reduziert werden kann bzw. ein Stellvertreter-Problem L' . Gilt nur der zweite Aspekt, so ist das Problem L NP -schwierig. NP -vollständige Probleme gelten trotz erfolgreicher Anwendung des dynamischen Programmierens als praktisch unlösbare Aufgaben [11, p. 145]⁵.

2.4 Verschlüsselungsverfahren

Verschlüsselung versucht primär die Vertraulichkeit einer Nachricht zu gewährleisten. Es geht darum eine Nachricht in einen Geheimtext zu überführen und ohne Informationsverlust wiederherzustellen. Hierbei finden Kombinationen aus Permutation und Substitution statt. All das hängt idealerweise von nur einem Geheimnis ab, das die Kommunikationspartner kennen. Verschlüsselung sollte seine Sicherheit nicht aus der Geheimhaltung des Verfahrens gewinnen (Security by Obscurity) [4, p. 42]. Idealerweise wird ein Schlüssel der

⁴ Siehe Anlage „A6 Beispiel des Ressourcenverbrauchs einer vollständigen Schlüsselsuche“

⁵ Siehe Anlage „A5 Beispiel von Problemen der Klasse P, NP und NP-Vollständig“

gleichen Länge der zu verschlüsselnden Botschaft verwendet. Hierbei werden Klartext der Nachricht mit dem Schlüssel Exklusiv-Oder-Verknüpft, XOR. Das, zumindest in der Theorie, beste und sicherste Verfahren ist das One-Time-Pad OTP, welches nicht gebrochen werden kann. Aufgrund der Schlüsselgröße, der die gleiche Größe wie die zu schützende Nachricht hat, ist es jedoch praktisch nicht umsetzbar. Sowohl die Schlüsselerzeugung also auch die Verteilung ist schwierig. Ein Problem ist die Erzeugung von Zufallsfolgen, die mathematisch deterministisch berechnet werden und somit nur pseudozufällig sind [13, p. 25]. Stromchiffren verschlüsseln Nachrichten beliebige Länge mit einem Schlüsselstrom in Abhängigkeit eines geheimen Schlüssels k . Beispiele sind die Verfahren RC4 und A5 (s. GSM Standard). Die hier verwendeten Schlüssel dürfen nur einmal angewendet werden [5, p. 72]. Daneben gibt es die Blockchiffren, die mittels Funktionen Nachrichten blockweise in Abhängigkeit eines geheimen Schlüssels k verschlüsseln und zuvor genannte Permutation und Substitution einsetzen [14]. Wesentliche Grundprinzipien der Verschlüsselungen sind Konfusion, Diffusion und Nichtlinearität⁶ [4, p. 94]. Die ersten beiden Anforderungen erschweren statistische Verfahren der Kryptanalyse, die Nichtlinearität ist eine Antwort auf differentielle und lineare Kryptanalyse [13, p. 46].

2.4.1 Symmetrische Verschlüsselungsverfahren

Symmetrische Kryptosysteme verwenden den gleichen Schlüssel k für die Verschlüsselung (Encryption e) und Entschlüsselung (Decryption d) von Klartextnachrichten m in ein Chiffre bzw. Ciphertext c und stellen die ältesten kryptographischen Verfahren dar. Sie sind besonders schnell, meistens einfacher und somit effizienter als asymmetrische Verfahren [5, p. 43],

$$c = e(k, m), \quad (2.3)$$

$$m = d(k, e(k, m)). \quad (2.4)$$

⁶ Siehe Anlage „A7 Konfusion, Diffusion und Nichtlinearität“

Eine Problematik symmetrischer Verschlüsselung ist der Austausch des gemeinsamen Schlüssels zwischen Sender und Empfänger vor einer Kommunikation. Erfolgt der Austausch unverschlüsselt, so ist auch die Folgekommunikation nicht gesichert. Dieser Schlüssel muss vorab ausgetauscht werden. Dieser Aufwand erhöht sich mit der Menge an Kommunikationspartnern auch insbesondere dann, wenn nach jeder Kommunikation ein neuer Schlüssel verwendet werden soll oder muss (Problem der Schlüsselverteilung). Ein Beispiel für ein symmetrisches Verschlüsselungssystem ist der Data Encryption Standard DES⁷. DES wurde bis heute nicht gebrochen, wenn man das Maß kryptographischer Verfahren, dass kein Angriff effizienter sein darf, als die vollständige Schlüsselsuche, auch Brute-Force, als Kriterium für Sicherheit ansetzt [15, p. 192].

Die gängigsten symmetrischen Verschlüsselungsverfahren sind oftmals der rundenweise Aufbau sowie die blockweise Verschlüsselung gemein. Sie sind effizient und sowohl hardware- als auch softwaretechnisch gut zu implementieren. Jede Verschlüsselungsrunde läuft im Wesentlichen identisch ab und besteht für gewöhnlich aus den einfachen Funktionen Exklusiv-Oder-Verknüpfung, Permutation sowie Substitution. Charakteristisch ist ihnen die einfache Umkehrbarkeit [4, p. 75],

$$\text{Chiffre-Block } c_i = \text{DES}(k, m_i) \text{ mit } i = 0, 1, 2, \dots, n \quad (2.5)$$

$$\text{Klartext-Block } m_i = \text{DES}^{-1}(k, c_i), \quad (2.6)$$

mit Schlüssel $k = 2^{64} \rightarrow \text{Quantität}, |k| = 2^{56} \rightarrow \text{Qualität}$

was die Symmetrie hervorhebt. DES ist ein Musterbeispiel, dass die Offenlegung der Funktionsweise eines Verschlüsselungsverfahrens die Sicherheit und das Vertrauen in es erhöht. In einem Zeitraum von 20 Jahren ist es nicht gelungen, abgesehen von der geringen Schlüssellänge, eine Schwachstelle im DES zu finden. Geht man von einer vollständigen Schlüsselsuche bei DES

⁷ Siehe Anlage „A8 Data Encryption Standard, DES“ für DES, doppelter DES, Triple DES

binnen 24 Stunden aus (Weltrekord Stand 2007), so ergeben sich für verschiedenen große Schlüssel folgende Werte:

Tabelle 2.1 Skalierte Dauer der vollständigen Schlüsselsuche, abgebildet auf den Referenzwert des Rekordes aus dem Jahr 2007 [4, p. 91].

Schlüssel- länge in Bit	Anzahl der Schlüssel	Dauer der vollständigen Schlüssel- suche
40	$1,1 * 10^{12}$	1,3 Sekunden
56	$7,1 * 10^{16}$	24 Stunden (DES, Referenzwert)
64	$1,8 * 10^{19}$	256 Tage
80	$1,2 * 10^{24}$	45.965 Jahre
128	$3,4 * 10^{38}$	$1,3 * 10^{19}$ Jahre
192	$6,3 * 10^{57}$	$2,4 * 10^{38}$ Jahre
256	$1,2 * 10^{77}$	$4,4 * 10^{57}$ Jahre

Trotz der Sicherheit des DES Verfahrens musste die Schwäche des kleinen Schlüssels kompensiert werden. Zunächst wurde die doppelte Anwendung in Betracht gezogen. Diese war jedoch durch Meet-in-the-Middle-Attacken einfach zu brechen. Gelöst wurde das Problem durch Triple-DES, welches jedoch trotz dreifacher Ausführung nur ca. die doppelte Sicherheit von DES erreichte und somit im Vergleich zum ursprünglichen Verfahren ineffizient ist [4, p. 91]⁸.

Die Antwort auf die Schwächen von DES war ein Wettbewerb im Jahre 1998, der den künftigen symmetrischen Verschlüsselungsalgorithmus Advanced Encryption Standard AES⁹ ermitteln sollte. AES unterstützt Schlüssellängen von 128 Bit, 192 Bit sowie 256 Bit. Als Maßstab für die Sicherheit galt, dass die vollständige Schlüsselsuche das beste Angriffsverfahren darstellen muss. Als Zielplattformen galten u. A. Smartcards, welche nur begrenzte Ressourcen zur Verfügung haben [16, p. 105]. Erst zehn Jahre später wurde AES erstmals

⁸ Siehe Anlage „A8 Data Encryption Standard, DES“ für Doppel- und Triple-DES

⁹ Siehe Anlage „A9 Advanced Encryption Standard, AES“

erfolgreich mit dem Bliclique-Angriff attackiert. Dieser Angriff ist im Schnitt ca. um den Faktor 4 schneller als die vollständige Schlüsselsuche, was jedoch für die praktische Sicherheit nicht relevant ist. Die angenommene Sicherheit lässt sich auch daraus ableiten, dass die National Security Agency NSA im Jahr 2003 das Verschlüsseln vertraulicher Dokumente bis zur Stufe SECRET mit AES mit allen Schlüssellängen und bis zur Stufe TOP SECRET mit den Schlüssellängen 192 oder 256 Bit zugelassen hat [16, p. 135].

2.4.2 Asymmetrische Verschlüsselungsverfahren

Symmetrische Verschlüsselungsverfahren haben gezeigt, dass die von ihnen gelieferte Sicherheit schon bei 128 Bit-Schlüsseln für heutige Verhältnisse hoch ist. Dennoch vermag symmetrische Verschlüsselung das Problem der Schlüsselverteilung nicht zu lösen. Je nach Anzahl der Adressaten n müssten $k_n = n/2 * (n - 1)$ Schlüssel generiert und vorverteilt werden, damit jeder mit jedem in diesem Netzwerk gesichert kommunizieren könnte.

Die asymmetrische Verschlüsselung, auch Public-Key-Kryptographie, löst dieses Problem unter Verwendung disjunkter, jedoch voneinander abhängiger Schlüssel $k_{public}, k_{private}$ mit

$$\text{Geheimtext } c = e(k_{public}, m), \quad (2.7)$$

$$\text{Klartext } m = d(k_{private}, c), \quad (2.8)$$

$$m = d(k_{private}, e(k_{public}, m)). \quad (2.9)$$

Im Prinzip kann aus dem öffentlichen Schlüssel der private Schlüssel abgeleitet werden, was praktisch jedoch nicht durchführbar ist. Asymmetrische Verschlüsselungsverfahren machen sich hierfür die Mathematik in endlichen Zahlkörper und die Division mit Rest zu Nutze [17]. Diese Rechenarten sind einfach durchführbar, ihre Umkehrung ist dagegen aufwendig. Grundlagen hierfür sind Einwegfunktionen, die über eine Falltür verfügen. Dazu zählen z.B. die diskrete Exponentialfunktion sowie die Umkehrung durch den diskreten

Logarithmus (diskreter Logarithmus Problem DLP)¹⁰ [5]. Dieser ist nicht effizient berechenbar. Man kann zeigen, dass Einwegfunktionen genau dann existieren, wenn $P \neq NP$, die Vermutung aus der Komplexitätstheorie, gilt. Eine Tatsache, die beim u. a. Diffie-Hellman-Schlüsselaustausch ausgenutzt wird¹¹ [18]. Zusätzlich ist die Multiplikation zweier Primzahlen p und q zu einer Zahl n trivial und kann auch für sehr große Zahlen sehr einfach und schnell durch Computer durchgeführt werden. Die Umkehrung der Operation, d.h. das Finden zweier Primzahlen p, q für eine Zahl n mit mehreren hundert Stellen, erfordert enormen Rechenaufwand. Es handelt sich um das Faktorisierungsproblem [4], welches für die Sicherheit¹² bei z.B. RSA¹³ verantwortlich ist. Es gibt zwar mathematische Ansätze, die schneller sind als eine vollständige Schlüsselsuche, dennoch existiert kein effizienter Algorithmus auf klassischen Systemen. Es ist jedoch nicht auszuschließen, dass es einen solchen Algorithmus geben könnte.

Tabelle 2.2 Rekorde bei der Berechnung des diskreten Logarithmus aus dem Jahr 2008 [19, p. 95].

Bit	Jahr	Gruppe	Algorithmus
193	1991	Lamacchia-Odlyzko	Gaußsche Methode
216	1995	Weber	Zahlkörpersieb
282	1996	Weber	Gaußsche Methode
299	1998	Joux-Lercier	Gaußsche Methode
332	1999	Joux-Lercier	Zahlkörpersieb
365	2000	Joux-Lercier	Zahlkörpersieb
398	2001	Joux-Lercier	Zahlkörpersieb
432	2005	Joux-Lercier	Zahlkörpersieb

¹⁰ Siehe Anlage „A10 DLP Computational-Diffie-Hellman-Problem“

¹¹ Siehe Anlage „A11 Der Diffie-Hellman-Schlüsselaustausch“

¹² Siehe Anlage „A13 Sicherheit von RSA“

¹³ Siehe Anlage „A12 Rivest, Shamir und Adleman, RSA“

Das quadratische Sieb¹⁴ ist eines der schnellsten Verfahren für die Faktorisierung mit einer Laufzeit von

$$O(n) = e^{\sqrt{\log_2 n \log_2 \log_2 n}}. \quad (2.10)$$

Dies ist somit subexponentiell jedoch immer noch superpolynomiell, d.h. das quadratische Sieb ist jenseits der effizient lösbaren polynomieller Probleme, welche zuvor als Grenze vorgestellt wurden [20].

Tabelle 2.3 Beispiele für die verschiedenen Größenordnungen von Laufzeiten [21].

Exponentiell	>	Polynomiell	>	Logarithmisch
e^x		$x\sqrt{(1+x^2)}$		$\ln(\ln(x))$
$\cosh x$		$x^2 + 1$		$(\ln(x))^3$
$3^{\sqrt{x}}$		\sqrt{x}		$\ln(x)$
2^x				

¹⁴ Siehe Anlage „A14 Vergleich von Angriffen mit dem Ziel der Faktorisierung“

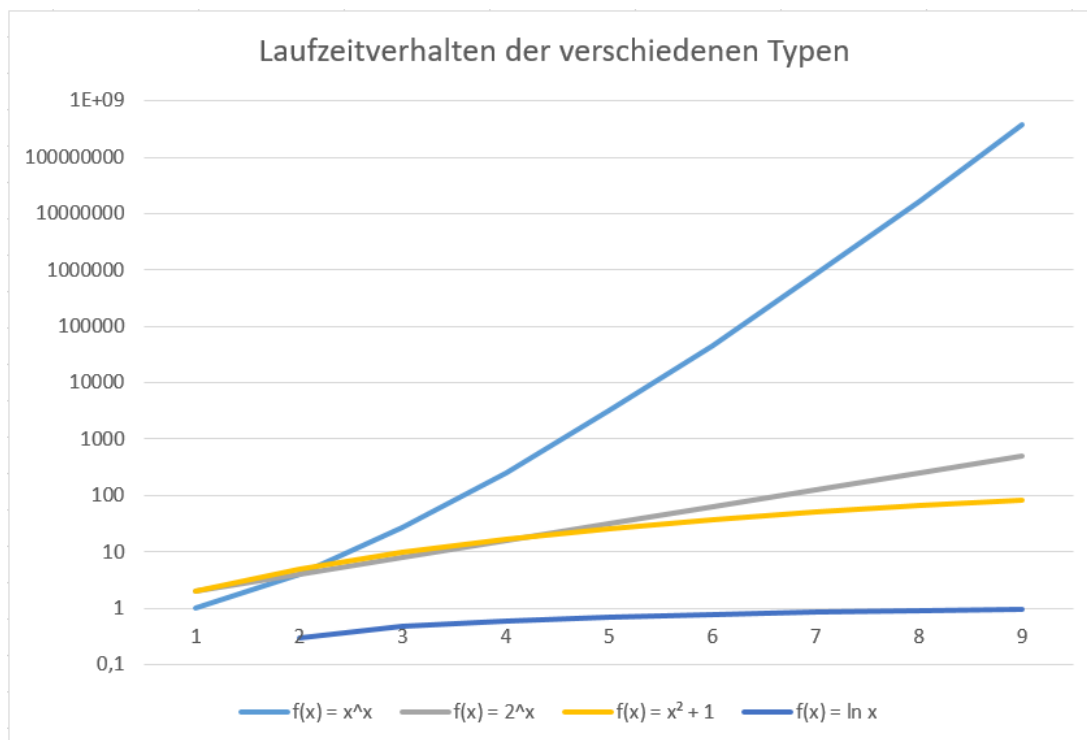


Bild 2.2 Beispielhafte Darstellung von Aufwänden.

Das quadratische Sieb kann durch das Zahlkörpersieb in der Laufzeit weiter optimiert werden. Jedoch sind auch diesem ähnliche Grenzen gesetzt, so dass nach heutigem Wissensstand die Faktorisierung großer Zahlen, wie sie derzeit in RSA eingesetzt werden, die Sicherheit asymmetrischer Verfahren nicht akut bedroht.

Tabelle 2.4 Stand der RSA Faktorisierungsrekorde 2008 [31, p. 89].

Zahl	Bits	faktoriert	Gruppe	Algorithmus
RSA-100	332	04. 1991	Lenstra	Quadr. Sieb
RSA-110	365	04. 1992	Lenstra	Quadr. Sieb
RSA-120	399	06. 1993	Denny	Quadr. Sieb
RSA-129	429	04. 1994	Lenstra	Quadr. Sieb
RSA-130	432	04. 1996	Lenstra	Zahlkörpersieb
RSA-140	465	02. 1999	te Riele	Zahlkörpersieb
RSA-155	512	08. 1999	te Riele	Zahlkörpersieb
RSA-160	532	04. 2003	Franke	Zahlkörpersieb
RSA-576	576	12. 2003	Franke	Zahlkörpersieb

RSA-640	640	11. 2005	Franke	Zahlenkörpersieb
RSA-703	703	offen		
...				
RSA-1024	1024	offen		

Asymmetrische Verschlüsselungsverfahren sowie ihre Grundlagen finden ebenfalls Anwendungen bei hybrider Verschlüsselung¹⁵, welche eine Mischform aus symmetrischen und asymmetrischen Verfahren darstellen. Auch basiert die Sicherheit heutige Signaturverfahren¹⁶ auf diesen eingesetzten Einwegfunktionen [17] [14] [22].

2.5 Zusammenfassung

Kryptographische Verfahren verfolgen die Schutzziele Vertraulichkeit, Integrität, Authentizität und Nichtabstreitbarkeit und bedienen sich dafür symmetrischer und asymmetrischer Verfahren. Dabei verfolgt die Kryptographie einfache Paradigmen. Die Komplexität der Verfahren sollte gering sein, die Sicherheit hängt nach dem Kerckhoffs'schen Prinzip nur vom Schlüssel ab und nicht von der Geheimhaltung des Verfahrens selbst. Auch sollte es keinen effizienteren Weg des Brechens als die vollständige Schlüsselsuche (Brute Force) geben. Symmetrische Verfahren sind relativ einfach und schnell. Die vollständige Schlüsselsuche ist, bei korrektem Aufbau, die effizienteste Angriffsmethode auf das Verfahren und erlaubt so aktuell Schlüsselgrößen im Bereich von 192 oder 256 Bit, welche z.B. für die höchste Geheimhaltungsstufe in den Vereinigten Staaten von Amerika zugelassen ist [23].

Asymmetrische Verfahren sind komplexer und stützen ihre Sicherheit auf die beiden mathematischen Probleme der Faktorisierung von Zahlen sowie dem diskreten Logarithmus in endlichen Zahlkörpern. Hierfür gibt es Algorithmen die schneller sind als die vollständige Schlüsselsuche, d.h. bei gleicher Schlüsselgröße zu symmetrischen Verfahren sind asymmetrische Verfahren deutlich unsicherer. Dies wird durch deutlich größere Schlüssel im Bereich von 2048

¹⁵ Siehe Anlage „A15 Hybride Verschlüsselungsverfahren“

¹⁶ Siehe Anlage „A16 Signaturverfahren“

Bit und mehr kompensiert. Die effizientesten Algorithmen sind zwar subexponentiell in ihrer Laufzeit, jedoch über polynomial. Genau das stellt die Grenze zwischen praktisch lösbaren und nicht lösbaren Problemen dar. Der Aufwand, der nicht lösbaren Probleme wächst im Allgemeinen schneller und überschreitet damit die Möglichkeiten klassischer Rechnersysteme bezüglich Speicherverbrauch und Geschwindigkeit. Es ist die Klasse der *NP*-Probleme, die auf einer nichtdeterministischen Turingmaschine in Polynomialzeit lösbar sind. Es handelt sich jedoch aus heutiger Sicht um theoretische Maschinenmodelle, die nicht ohne weiteres konstruiert werden können. Asymmetrische Verfahren stellen einen wichtigen Teil der heutigen Kryptographie dar und sind für die Schlüsselverteilung, aber auch für den Nachweis von Authentizität und der Nichtabstreitbarkeit, unerlässlich.

3 Der Quantencomputer

3.1 Charakteristika von Quantencomputern

Ein Quantencomputer ist ein Rechner, der anders funktioniert als heute gängige Systeme. Quantencomputer arbeiten auf sogenannten Qubits mit $|0\rangle$ und $|1\rangle$ als Basiswerte, vergleichbar als Nord- und Südpol auf einer Kugel. Sie können in Superposition $|\psi\rangle$, d.h. in sich überlagernden Zuständen zur Darstellung eines beliebigen Punktes auf dieser Kugel, überführt werden. Diese Form der Darstellung wird als Bloch-Kugel bezeichnet [24]. Ein Quantencomputer nutzt die Quantenphysik, um einen erheblichen Geschwindigkeitsvorteil bei der Lösung von speziellen Aufgaben gegenüber einem klassischen Rechner zu erzielen. So kann der Quantencomputer bestimmte und besonders schwere Aufgaben in Polynomialzeit oder besser lösen [25].

Ein Qubit kann, im Gegensatz zum klassischen Bit, die Werte 0 und 1 „zugleich“ einnehmen. Man spricht auch von sich überlagernden Zuständen (Parallelexistenz aller Möglichkeiten [25]) [26]. Hier ist es üblich Zustände, analog zur Abbildung auf der Bloch-Kugel, in der Notation einzuklammern. Die klassischen Werte werden somit als $|0\rangle$ und $|1\rangle$ dargestellt (ket-Notation, Dirac-Notation). Ein Zustand für ein Qubit wird wie folgt notiert,

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle. \quad (3.1)$$

Die Koeffizienten $\alpha, \beta \in \mathbb{C}$ (Komplexe Zahlen) werden als Amplitude bezeichnet für die

$$|\alpha|^2 + |\beta|^2 = 1 \quad (3.2)$$

gilt. Ein Qubit kann sich in zwei klassischen Zuständen gleichzeitig befinden wobei die komplexen Zahlen den jeweiligen Anteil ausdrücken. Für die Berechnung sind hierbei die Beträge interessant. Jedoch wird auch die Phase, z.B. in der Quanten-Fouriertransformation QFT, verwendet. Ein solch unbestimmter Zustand wird als Superposition bezeichnet. Die Superposition existiert in Form einer Wahrscheinlichkeitswelle. Die Zustände werden zwar durch

(3.2) eingeschränkt, dennoch gibt es unendlich viele Möglichkeiten. Es ist nicht möglich den Zustand eines Qubits ohne eine konkrete Messung festzustellen. Im Unterschied zum klassischen Bit kollabiert dann die Wahrscheinlichkeitswelle und die Superposition wird zerstört¹⁷. Als Ergebnis erhält man dann, in Abhängigkeit der vorhergehenden Wahrscheinlichkeiten der Amplituden α und β , den Wert 0 oder 1 im klassischen Sinne. Betragen $\alpha, \beta = \frac{1}{\sqrt{2}}$, so ist der Messausgang unbestimmt. Die Zustände $|0\rangle$ und $|1\rangle$ sind gleich wahrscheinlich [7]

$$\left(\frac{1}{\sqrt{2}}\right)^2 + \left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2} + \frac{1}{2} = 1. \quad (3.3)$$

Die Werte $|0\rangle$ oder $|1\rangle$ sind bei der Messung vom Zufall abhängig. Während in der klassischen Physik bei ausreichenden Informationen und Rechenkapazität z.B. ein Münzwurf berechenbar ist, so ist das in der Quantenwelt nicht möglich. Es handelt sich um keinen Pseudozufall mehr. Qubits und ihre Superposition können als Vektor (orthogonale Einheitsvektoren in einem zweidimensionalen Hilbertraum [27]) dargestellt werden,

$$|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \alpha |0\rangle + \beta |1\rangle. \quad (3.4)$$

Für die Realisierung eines Qubits benötigt man ein Teilchen, welches diese binären Eigenschaften besitzt, z.B. den Spin eines Elektrons (magnetischer Eigendrehimpuls). Dieser kann parallel oder antiparallel zum Magnetfeld verlaufen (Up-Spin = $|1\rangle$, Down-Spin = $|0\rangle$) [28].

Durch das Rechnen mit Qubits vor der Messung, d.h. vor der Zerstörung der Superposition, kann mit beliebig vielen Zuständen gearbeitet werden statt nur mit zwei, wie es klassische Rechensysteme können [28]. Während ein Bit nur die Werte 1 oder 0 annehmen kann, kann ein Qubit somit zwei Werte tragen (Quanteninformation). Analog können zwei Qubits vier Werte speichern, drei Qubits acht etc. Mit jedem weiteren Qubit verdoppelt sich die Anzahl. 300

¹⁷ Siehe Anlage „Das Doppelspaltexperiment“

Qubits können bereits mehr Werte speichern (2^{300}), als das Universum Teilchen enthält [25, p. 47].

Neben der Superposition ist die Quantenverschränkung eine weitere wichtige Eigenschaft, die bei Qubits für Berechnungen verwendet wird. Unter Verschränkung versteht man Teilchen die sich voneinander abhängig als System in einem gemeinsamen Quantenzustand (Quantenobjekt) befinden. Hierbei werden zwei oder mehr Teilchen erzeugt, die durch eine einzige Wellenfunktion beschrieben werden. Diese Teile können sich nicht unabhängig voneinander verhalten. Bildet man aus einem Teilchen, welches neutral im Spin ist, zwei Teilchen, bei dem sich das eine im Zustand Up-Spin = $|1\rangle$ befindet, dann ist das zweite Teilchen im entgegengesetzten Zustand Down-Spin = $|0\rangle$. Ein solches Quantenobjekt existiert in Superposition [29]. Auf derartige Systeme kann, ohne eine Messung zu tätigen, eingewirkt werden. Hat man 100 verschränkte Teilchen, dann kann man mit den zwei Zuständen der Teilchen 2^{100} Zustände zugleich beschreiben (1,27 Quintillionen Zustände bzw. 1,27 Billionen Billiarden Zustände). Ändert man den Zustand eines Qubits, so beeinflusst man alle anderen Qubits durch die Verschränkung parallel. Diese Art der immensen Parallelität ist mit klassischen Rechensystemen nicht möglich. Die Superposition von Qubits erlauben das Speichern einer großen Menge an Informationen während die Verschränkung die immense parallele Manipulation gestattet, was die wesentlichen Bausteine eines Quantencomputers darstellt und somit die Überlegenheit gegenüber einem klassischen Computer für bestimmte Problemstellungen [28].

3.2 Berechnungen auf einem Quantencomputer

Analog zu den Logikgattern heutiger Computer benötigt man für die Manipulation von Qubits auch bei Quantencomputer sogenannte Quantengatter. Während es bei den Rechenschritten klassischer Rechner keine weiteren Einschränkungen gibt, kann ein Qubit hingegen unendlich viele Zustände annehmen [7]. Quantengatter sind umkehrbar im Gegensatz zu z.B. einem klassi-

schen OR-Gatter, bei dem aus zwei Werten a, b nur noch ein Wert c als Ergebnis verbleibt ohne das noch eine Aussage über a, b getroffen werden kann. Drei Quantengatter genügen für den Bau eines universellen Quantencomputers. Zwei Gatter-Arten werden benötigt, um ein Qubit in jeden beliebigen Zustand zu versetzen (Ein-Qubit-Gatter) während das dritte Gatter zwei Qubits bedarf (Zwei-Qubit-Gatter).

Das NOT-Gatter ist ein Ein-Qubit-Gatter und verhält sich analog zum klassischen NOT-Gatter, indem es den Zustand eines Qubits invertiert,

$$N|0\rangle = |1\rangle, \quad (3.5)$$

$$N|1\rangle = |0\rangle. \quad (3.6)$$

Das Hadamard-Gatter ist ein Ein-Qubit-Gatter und wird verwendet, um Qubits in Superposition zu versetzen. Ein Qubit wird genau „zwischen“ die beiden Basiszustände $|0\rangle$ und $|1\rangle$ gebracht [30], d.h. würde man das Qubit im Anschluss direkt messen, so hätte man eine genau 50-prozentige Chance für jedes der beiden Ergebnisse $|0\rangle, |1\rangle$. Es „scheint“, als würde das Hadamard-Gatter so jede Information aus dem Qubit ursprünglichen Basiszustand $|0\rangle$ oder $|1\rangle$ auslöschen. Wiederholt man die Anwendung des Hadamard-Gatters, so hat das Qubit jedoch wieder den Ausgangswert. Die Information wäre also nicht gelöscht worden¹⁹. Das zeigt, dass die Zustände, in denen Qubits rein zufällige Messergebnisse ergeben, sehr genau definierte Zustände sind. Das heißt aber auch, dass sich zwischen der ersten und der zweiten Anwendung des Hadamard-Gatters nichts am Zustand des Qubits verändern darf. Wenn sich dennoch etwas verändert hat, kann das mit dem Hadamard-Gatter aufgedeckt werden, selbst wenn die Änderung gut versteckt war [31] [32],

$$H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \quad (3.7)$$

¹⁸ Siehe Anlage „A18 Quanten NOT-Gatter“

¹⁹ Siehe Anlage „A19 Hadamard-Gatter“

$$H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}. \quad (3.8)$$

Das NOT- und das Hadamard-Gatter sind die am häufigsten eingesetzten Ein-Qubit-Gatter [28]. Darüber hinaus gibt es weitere Gatter, die hier jedoch nicht erwähnt sein sollen.

Das Controlled NOT-Gatter, CNOT-Gatter, ist ein NOT-Gatter, welches jedoch von dem Zustand eines zweiten Qubits abhängt. Es ermöglicht so die Interaktion von zwei Qubits in einem Quantenregister. Steht das kontrollierende Qubit x auf $|1\rangle$ so wird das Ziel-Qubit y invertiert. Steht es auf $|0\rangle$ bleibt y unverändert [32]. Die Ergebnisse nach einem CNOT-Gatter entsprechen einem XOR-Gatter aus klassischen Rechnersystemen, bei denen das Eingangs-Bit aufbewahrt wird (unitäre Variante des Exklusiv-Oder [7]). Das CNOT-Gatter wird verwendet, um beide Qubits miteinander zu verschränken. Wenn das erste kontrollierende Qubit selbst nur eine 30-Prozent-Chance hat, im Zustand $|1\rangle$ zu sein, dann wird das zweite Qubit auch nur mit einer 30-Prozent-Chance in sein Gegenteil verkehrt. Welchen Zustand das zweite Qubit am Ende hat, ist jetzt also vollkommen abhängig davon, was der tatsächliche Zustand des ersten Qubits war. Die Zustände der beiden Qubits werden dadurch verschränkt. Somit können die beiden Qubits, auch wenn sie nicht mehr in unmittelbarer Nähe sind, miteinander interagieren [31]²⁰.

3.2.1 Shor-Algorithmus

Der Algorithmus von Shor besteht aus einem klassischen- und einem Quantenteil, bei dem die Quanten-Fouriertransformation QFT, ein Quantenalgorithmus, ein wesentlicher Bestandteil ist. Die Quanten-Fouriertransformation hilft Perioden zu messen, um so Rückschlüsse auf die Menge bestimmter Lösungen zu erhalten. Der Shor-Algorithmus verwendet zuvor beschriebene Superpositionen, um mittels Verschränkungen eine Superpositionsstruktur in eine andere zu überführen. Dabei kommen Phasen-Rotationen zur Anwendung. So

²⁰ Siehe Anlage „A20 Controlled NOT-Gatter“

wird ermöglicht, dass durch Überlagerungen Wahrscheinlichkeiten eliminiert werden und das in einer Geschwindigkeit, wie es auf klassischen Rechnersystemen nicht möglich wäre [33] [34]. Dieser Geschwindigkeitsvorteil wird umso größer je mehr Qubits verwendet werden. Eine Eigenschaft der QFT ist, dass die Rotation in den höheren Wertigkeiten der Qubits größere Sprünge macht als für den Wert 0. Hierdurch werden nicht nur periodische Funktionen erkannt²¹, sie beginnen auch bei 0, was in der Folge für den Algorithmus von Shor wichtig ist,

$$QFT_N: |j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i [0, \dots, x_k]} |k\rangle. \quad (3.9)$$

Der Algorithmus von Shor faktorisiert eine Zahl n in p und q mit Hilfe der Funktion $f_{a,n}(x)$ als

$$f_{a,n}(x) = a^x \bmod n, \quad (3.10)$$

wobei $a < N$ zufällig gewählt wird und N in der Größenordnung n^2 liegt. Shor verwendet für die Berechnung zwei Quantenregister $|a\rangle, |b\rangle$. Register $|a\rangle$ wird mit den Exponenten x beladen, anschließend erfolgt die Berechnung der Formel (3.10) wobei die Ergebnisse in $|b\rangle$ gespeichert werden. Beide Register sind somit verschränkt. Durch eine Messung in $|b\rangle$ erfolgt der Kollaps der Wellenfunktion und ein Restwert der Formel (3.10) verbleibt in Register $|b\rangle$. Die Verschränkung zu $|a\rangle$ bewirkt nun, dass lediglich die zugehörigen Exponenten x im Register bestehen bleiben. Mithilfe der Quanten-Fouriertransformation QFT als letzten Schritt auf dem Quantencomputer wird das Register $|a\rangle$ ausgelesen, so dass man die Periode r erhält. Die Faktoren p und q werden anschließend mit Hilfe der gefundenen Periode r auf einem klassischen System berechnet und getestet²² [7] [28] [35][22],

²¹ Siehe Anlage „A21 Quanten-Fouriertransformation“

²² Für eine umfangreiche Darstellung der größten Bedrohung bestehender asymmetrischer kryptographischer Verfahren inkl. Beispielen und Laufzeitbetrachtung siehe Anlage „A22 Shor-Algorithmus“

$$ggT(N, a^{\frac{r}{2}} \pm 1). \quad (3.11)$$

Für den Algorithmus werden zwei Register mit je $\log_2 n$ Qubits für die Faktorisierung einer Zahl n benötigt. Für den Aufbau und die Berechnungen innerhalb der Quantenregister sowie die Messungen ergeben sich ein Aufwand von $\mathcal{O}((\log_2 n)^3)$ Schritten. Der klassische Teil von Shors Algorithmus benötigt $\mathcal{O}(\log_2 n)$ Multiplikationen und hat somit in Summe eine erwartete Laufzeit von $\mathcal{O}((\log_2 n)^4)$ [7]. Die Laufzeit ist somit polynomiell und bedroht heutige asymmetrische Verschlüsselung sobald ausreichend viele Qubits innerhalb eines universellen Quantencomputer zur Verfügung stehen.

3.2.2 Grover-Algorithmus

Der zweite berühmte Quantenalgorithmus wurde 1996 von Lov Grover entwickelt und ist ein Suchalgorithmus für ungeordnete Daten. Durch Manipulation der Wellenfunktion in einem sich in Superposition befindlichen Quantenregister ist der Algorithmus in der Lage ein Suchergebnis in $\mathcal{O}(\sqrt{N})$ -Schritten zu finden. Für die Größenordnung von $N = 10.000$ fände er eine Lösung in 100 Schritten. Der Vorteil wächst mit steigender Größenordnung von N [28] [7]. Ein klassisches System wäre gezwungen eine vollständige Suche durchzuführen (vgl. Brute-Force oder vollständige (Schlüssel-)Suche) so dass im Mittel eine Lösung erst nach $N/2$ Schritten gefunden werden würde (vgl. die Suche nach einer Person in einem ungeordneten Telefonbuch oder das bloße Ausprobieren von PINs einer EC-Karte).

Grover verwendet dafür die Invertierung der Phase des gesuchten Wertes in der Datenbank, was keinen Einfluss auf die Wahrscheinlichkeit bei einer Messung hat. Dennoch kann er durch Spiegelungen am Mittelwert aller Amplituden so die Messwahrscheinlichkeit des gesuchten Wertes verstärken. Durch dieses Vorgehen steigt die Wahrscheinlichkeit einen bestimmten Wert nach $\mathcal{O}(\sqrt{N})$ Schritten zu finden. Eine Suche im Vergleich zu klassischen Systemen wird somit quadratisch beschleunigt [7] [36]. Die Problematik am Algorithmus von Grover ist, dass man zu Beginn den gesuchten Zustand invertieren muss,

damit er überhaupt starten kann. Lov Grover ließ die Frage offen und ging in seiner Arbeit davon aus, dass eine solche Funktion, auch „Orakel“ genannt, existiert. Darüber hinaus würde die Schwierigkeit eines Angriffs auf ein symmetrisches Verschlüsselungsverfahren auf die Quadratwurzel reduziert werden. Ein Schlüssel der Größe 256-Bit hätte damit nur noch die Wirkung eines Schlüssels der Größe 128-Bit. Dieser Verlust an Sicherheit kann leicht durch eine Verdoppelung des Schlüsselraums kompensiert werden [37] [38].²³

3.3 Zusammenfassung

Während Grovers Algorithmus aktuell eine einfach zu lösende Bedrohung für symmetrische Verschlüsselungsverfahren darstellt ist der Algorithmus von Shor in der Lage in polynomieller Laufzeit asymmetrische Verfahren zu brechen. Das gilt nicht nur für die Sicherheit, die sich auf die Faktorisierung großer Zahlen stützt. Shor hat einen weiteren Algorithmus entwickelt, der mit Hilfe von drei Quantenregistern und seinem zuvor vorgestellten Algorithmus das Problem des Diskreten Logarithmus effizient lösen kann [39]. Das Bundesamt für Sicherheit in der Informationstechnik nannte 2018 Computer mit 50-72 Qubits als aktuellen Stand der Technik [24] und skaliert, dass „*zum Brechen von 2048-Bit RSA in 100 Tagen etwa eine Million physikalische Qubits, in 1 Stunde etwa eine Milliarde physikalische Qubits benötigt werden*“ [38, p. 8]. Aktuell arbeiten mehrere große Wirtschaftsunternehmen und öffentliche Institutionen mit großem Einsatz an Quantencomputern. So hat die Europäische Union ein Programm namens „Quantum Flagship“ mit einer Laufzeit von über zehn Jahren (2017-2027) und einem Budget von 1 Milliarde Euro ins Leben gerufen mit dem Ziel „*to unlock the full potential of quantum technologies, accelerate their development and bring commercial products to public and private users.(ECI 19/4/2016)*“ [40, p. 15]. Aktuell scheint es schwierig abzuschätzen, ob und wann ausreichend starke Systeme zur Verfügung stehen werden. Auf der NIS Summer School 2018 schätzte die Firma Infineon einen Zeitraum von 15-20 Jahren [41, p. 5] bis zur Entwicklung eines ausreichend starken universellen

²³ Siehe Anlage „A23 Grovers Algorithmus“

Quantencomputer [42]. Angesichts dieses technischen Stands kann man davon ausgehen, dass Quantencomputer zumindest nicht kurzfristig asymmetrische Kryptoverfahren bedrohen. In Hinblick auf die Zukunft ist es jedoch notwendig kryptographische Verfahren zu entwickeln, die künftigen Quantencomputern standhalten insbesondere dann, wenn die Vertraulichkeit über das erwartete Erscheinungsjahr hinausgehen soll, da sonst heute aufgezeichnete Kommunikation nicht mehr sicher wäre (Store-now, Decrypt-later).

4 Post Quantenkryptographie

4.1 Was ist Post Quantenkryptographie?

Die Post Quantenkryptographie (engl. Post-Quantum-Cryptography, PQC) ist ein Bereich der Kryptographie, welcher sich mit Verfahren und kryptographischen Primitiven auseinandersetzt, die auch unter Einsatz eines Quantencomputers nicht gebrochen werden können. Der Begriff PQC wurde geprägt durch Daniel J. Bernstein, einem amerikanischen Mathematiker und Gründer der PQCrypto²⁴. Der Begriff ist nicht standardisiert. Das Forschungsfeld wird auch als quanten-resistent oder quanten-sicher bezeichnet. Aufgrund der Bedrohung, insbesondere von asymmetrischen Verfahren, die für den Schlüsselaustausch und Signaturen verwendet werden, konzentriert sich PQC auf diesen Bereich und dem Finden von Alternativen [43]. Dabei lässt sich die Post Quantenkryptographie in fünf Teilgebiete bzw. Familien unterteilen. Codebasierte-, Hashbasierte-, Multivariate-, Gitterbasierte- und Isogeniebasierte-Kryptographie [44]. Nachstehendes Kapitel wird sich mit diesen Teilbereichen auseinandersetzen. Darüber hinaus bietet die Quantenmechanik selbst eine mögliche Lösung für die Zukunft an.

4.2 Quantenkryptographie

Schlüsselaustauschverfahren mit Hilfe der Quantenmechanik werden als Quantenschlüsselaustausch (engl. Quantum-Key-Distribution, QKD), auch Quantenkryptographie, bezeichnet. Hierbei werden Aspekte der Quantenteleportation²⁵ und das No-Cloning-Theorem²⁶ ausgenutzt. Ziel ist die Erzeugung eines gemeinsamen Schlüssels zwischen Alice und Bob unter der Annahme, dass ein Angreifer Mallory sowohl passiv als auch aktiv Zugriff auf den Kommunikationskanal hat.

²⁴ <https://pqcrypto.org/>





²⁵ Siehe Anlage „A24 Quantenteleportation“

²⁶ Siehe Anlage „A25 No-Cloning-Theorem“

Nachstehend sollen die beiden Verfahren BB84 und E91 vorgestellt werden. Beiden Verfahren ist gleich, dass durch die Messung an Photonen ein Schlüssel generiert wird. Es soll gezeigt werden, dass es Mallory weder möglich ist den Schlüssel abzuhören noch eine solche Aktivität geheim zu halten. Beide Verfahren dienen lediglich dem Generieren von Schlüsseln. Die Realisierung der Verfahren ermöglicht die Erstellung von zufälligen Zahlen sowie die Anwendung quantencomputer-resistenter Verschlüsselungsverfahren wie des One-Time-Pad oder AES.

Der Quantenkryptographie liegt die Idee des Quantengeldes des Physikers Stephen Wiesner zugrunde. Dabei handelt es sich um fälschungssicheres Geld auf Grundlage der Quantenmechanik. Das Konzept beruht auf Schwingungen von Photonen. Die Schwingung kann in vier Richtungen erfolgen und wird als Polarisierung bezeichnet. Die Notation der vier Schwingrichtungen wird mit den beiden Basen ($\updownarrow, \leftrightarrow$), (\nearrow, \nwarrow) dargestellt. Die Schwingungen innerhalb der Basen sind stets senkrecht zueinander. Es besteht die Möglichkeit einen Polarisationsfilter in die Flugbahn eines Photons zu stellen, so dass nur Photonen mit einer bestimmten Schwingung diesen Filter sicher passieren können oder mit Sicherheit geblockt werden.

Tabelle 4.1 Einfaches Beispiel von schwingenden Photonen und Polarisationsfiltern. Spalte 3 zeigt Photonen, welche den Filter passieren können.

Photonen	Filter	Gemessene Photonen
$\updownarrow, \leftrightarrow, \nearrow, \nwarrow$		\updownarrow und 50% (\nearrow, \nwarrow) ändern sich auf \updownarrow
$\updownarrow, \leftrightarrow, \nearrow, \nwarrow$		\leftrightarrow und 50% (\nearrow, \nwarrow) ändern sich auf \leftrightarrow
$\updownarrow, \leftrightarrow, \nearrow, \nwarrow$		\nearrow und 50% ($\updownarrow, \leftrightarrow$) ändern sich auf \nearrow
$\updownarrow, \leftrightarrow, \nearrow, \nwarrow$		\nwarrow und 50% ($\updownarrow, \leftrightarrow$) ändern sich auf \nwarrow

Die Tabelle zeigt, dass die Polarisationsfilter parallel zu ihnen ausgerichteten Photonen durchlassen. Alle orthogonal schwingenden Photonen werden blockiert. 50% der halb versetzten Photonen passieren ebenfalls den Filter und

werden dann in ihrer Polarisation durch den Filter abgeändert und angeglichen. Es ist nicht möglich das Photon in beiden Basen gleichzeitig zu messen (vgl. Quanten-Dilemma, Eindeutigkeit).

Die Idee des Quantengeldes war es auf Banknoten 20 Photonenfallen zu platzieren, die jeweils ein Photon mit ihrer Schwingrichtung auf Dauer gefangen halten. Parallel dazu erhält der Geldschein eine Seriennummer. Die ausgebende Bank besitzt eine Liste von Polarisationen zur Seriennummer für jede Banknote.

Um eine solche Banknote zu imitieren müsste ein Fälscher die Photonenfallen auslesen und auf eine Banknote mit derselben Seriennummer übertragen. Das Auslesen erfolgt mit Polarisationsfiltern, vergleichbar mit Tabelle 4.1. Bei passendem Filter, parallel oder orthogonal, ist das eindeutig und sicher. Schwierig wird das Messen der diagonal zum Filter schwingenden Photonen. Diese kommen zu 50% durch den Filter und nehmen danach dessen Ausrichtung an. Es ist einem Fälscher nicht möglich zu sagen um was für eine Art Photon es sich wirklich gehandelt hat. Das liegt darin begründet, dass es nicht möglich ist jede Eigenschaft eines bestimmten Objektes mit vollkommender Genauigkeit zu einem Zeitpunkt zu messen (vgl. heisenbergische Unschärferelation). Die Bank hingegen kennt die genaue Reihenfolge der Polarisationsfilter und wird so jede Photonenfalle zu 100% korrekt auslesen können. Eine falsche Polarisation würde hier zwangsläufig entdeckt werden und somit die gesamte Fälschung. Quantengeld ist technisch nicht realisierbar bzw. zu teuer. Das Konzept wurde jedoch in der Quantenkryptographie aufgegriffen [45].

4.2.1 Das BB84-Protokoll

Diese Grundidee wurde 1984 von Charles Bennett und Gilles Brassard im nach ihnen benannten Protokoll BB84 veröffentlicht. Das BB84-Protokoll handelt zwischen zwei Kommunikationspartnern, Alice und Bob, unter Verwendung der Quantenmechanik einen gemeinsamen Schlüssel aus. Alice erzeugt

einen zufälligen Schlüssel. Sie bringt Qubits mit dem Hadamard-Gatter in Superposition und misst anschließend zufällig den Wert und erhält so einen absolut zufälligen Schlüssel. Ein weiteres zufälliges Bit entscheidet ob Alice das Qubit an Bob in der Standard-Basis B (rektilineare - bzw. $\uparrow, \leftrightarrow$ oder +- Schema $\uparrow=1, \leftrightarrow=0$) oder Hadamard-Basis B' (diagonalen- bzw. \nearrow, \nwarrow oder x-Schema $\nwarrow=0, \nearrow=1$) überträgt. Verwendet sie die Hadamard-Basis wird zunächst die entsprechende Transformation auf das Qubit angewendet. Somit liegt das Qubit in einem der vier Zustände

$$|0\rangle, |1\rangle, |+\rangle, |-\rangle,$$

wobei

$$|+\rangle = H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad (4.1)$$

$$|-\rangle = H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \quad (4.2)$$

mit den Filtereinstellungen

$$|+\rangle = |45^\circ\rangle, |-\rangle = |-45^\circ\rangle$$

und

$$|1\rangle = |0^\circ\rangle, |0\rangle = |90^\circ\rangle$$

vor. Bob wählt vor der Messung auf dem Quantenkanal ebenfalls zufällig eine Basis. Er befindet sich jedoch in derselben Situation wie ein potenzieller Geldfälscher beim zuvor vorgestellten Quantengeld. Es misst lediglich mit 50% Wahrscheinlichkeit in der korrekten Basis. Misst er in der falschen Basis kommt es in 50% der Messungen zu einem Fehler, der das übertragene Bit neu ausrichtet und somit nichts mehr mit dem Ursprünglichen zu tun hat.

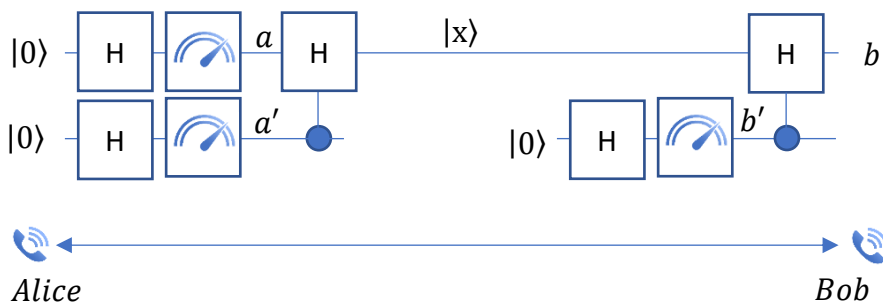


Bild 4.1 Schaltkreis für die Übertragung des Qubit $|x\rangle$ von Alice zu Bob [7, p. 176]. Von links nach rechts: Die ersten Hadamard-Gatter sorgen für die Zufälligkeit im Bit und der Basis, gefolgt vom entsprechenden Wechsel. Analog erfolgt die Messung und das Ergebnis von b auf der Seite von Bob. Ob b korrekt ist zeigt erst eine weitere Absprache der beiden über die gewählten Basen auf alternativem Wege.

Alice und Bob müssen sich nach der Übertragung auf einem ungesicherten Kanal über die verwendeten Basen austauschen, um festzustellen welche Messungen gleich waren und welche nicht. Bits bei ungleicher Basis werden von beiden verworfen. Der zufällige Schlüssel von Alice wird somit zufällig gekürzt.

Tabelle 4.2 Beispielhaft dargestellte Wertetabelle der übertragenen und gemessenen Werte aufbauend auf Bild 4.1.

BB84-Protokoll									
Alice Text	1	0	1	1	1	0	1	0	1
Alice Schema	+	x	x	x	+	x	+	+	x
Alice Filter		\	/	/		\		-	/
Schwingung	↑	↘	↗	↗	↑	↘	↑	↔	↗
Bob Schema	+	+	x	x	x	+	+	x	+
Bob Text	1	1	1	1	1	0	1	1	0
Bewertung	✓	x	✓	✓	x	x	✓	x	x
Ergebnis	1		1	1			1		

Mallory steht nun vor exakt demselben Problem und wählt mit einer Wahrscheinlichkeit $P = \frac{1}{2}$ das falsche Schema. Da die Wahl des falschen Schemas jedoch mit der Wahrscheinlichkeit $P = \frac{1}{2}$ die Polarisation eines Photons, von z.B. dem rektilineare- in das diagonale Schema, abändern greift er nicht

mehr nur passiv in die Übertragung ein. Je länger die Mitteilung ist, desto öfter unterlaufen Mallory Fehler beim Messen und desto öfter verfälscht er die Übertragung. Beim abschließenden Austausch der Polarisations-Einstellungen für eine kleine Menge von Kontroll-Bits werden Alice und Bob öfter feststellen, dass sie trotz gemeinsamer gleicher Basis unterschiedliche Resultate erhalten und, gemessen an einem vereinbarten Schwellwert für Fehler, entschließen den gesamten Schlüssel zu verwerfen. Statistisch wird Mallory bei jedem vierten Bit dieser Fehler unterlaufen [28]. Die Fehlerwahrscheinlichkeit P für die Wahl des Schemas bei der Messung durch Mallory für n -übertragenen Photonen steigt mit Länge der Übertragung,

$$P = 2^{-n}. \quad (4.3)$$

Auch hier wird Mallory mit einer Wahrscheinlichkeit von $P = \frac{1}{2}$ die Polarisation im Anschluss geändert haben. Belauscht Mallory jedes Bit, bleibt er nur mit einer Wahrscheinlichkeit von

$$P = \left(\frac{3}{4}\right)^k \quad (4.4)$$

unentdeckt, so dass einen Lauschangriff zu entdecken exponentiell mit der Zahl der Kontroll-Bits steigt. [7, p. 181]. Durch den Einsatz einer Pretty Good Quantum Cloning Machine PGQCM kann Mallory seine Fehlerrate verbessern. Es ist ihm möglich eine Genauigkeit von über 80% bei einer Kopie zu erreichen [46, p. 7]. Er drückt seine Fehlerrate. Durch stichprobenartige Messungen wäre er in der Lage einen Teilschlüssel zu belauschen während der Fehler als z.B. Leitungsruschen fehlinterpretiert werden könnten. Derartige Möglichkeiten müssen in der tolerierten Fehlerschwelle berücksichtigt werden. Die Sicherheit des Verfahrens kann durch die Privacy Amplification bzw. Leftover hash lemma (Lemma, dt. Hilfssatz, Zwischenschritt) erhöht werden.

Auch ein Angriff von Mallory durch Verschränkung scheint nur zunächst ohne Auswirkung. Jedoch verändert auch die Verschränkung des übertragenen

Qubits mit einem Qubit von Mallory die Wahrscheinlichkeit von Messergebnissen bei Bob zu seinen Ungunsten, was zum Entdecken des Eingriffs führt.

Das BB84-Protokoll wird heute bereits verwendet. Es bietet die Möglichkeit der Anwendung des One-Time-Pad-Verfahren, welches sicher ist und nachweislich nicht gebrochen werden kann. Ein Vorteil gegenüber klassischen Systemen ist, dass hier keine pseudozufälligen Zahlen generiert werden [27].

4.2.2 Das E91-Protokoll

Im Jahr 1991 entwickelte der polnisch-britische Physiker Artur Ekert ein Schlüsselverteilungsprotokoll auf Basis von Quantenverschränkung. Die Abschätzung der Sicherheit in diesem Protokoll erfolgt auf Basis der Bell-Ungleichung bzw. auf der von ihr abgeleiteten, experimentell nachweisbaren CHSH-Ungleichung. Alice und Bob sind mit ihr in der Lage festzustellen ob eine Verschränkung vorgelegen hat oder nicht.

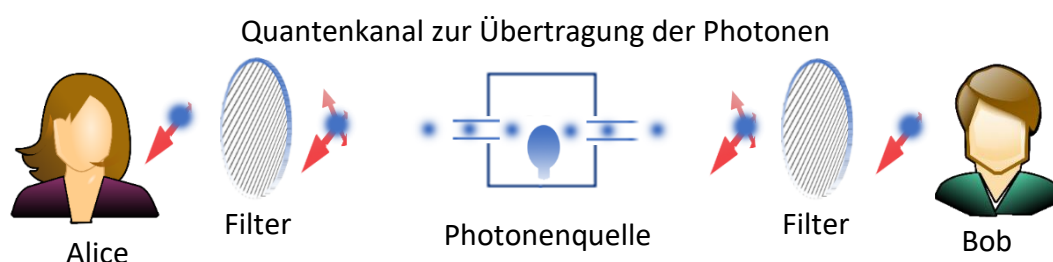


Bild 4.2 Über eine Photonengenerationsquelle erhalten Alice und Bob verschränkte Photonen, die sie durch drehbare Polarisationsfilter zufällig filtern.

Zwei Teilchen werden in einer EPR-Quelle generiert und an Alice und Bob verschickt. Die Teilchen sind in einem der Bell-Zustände verschränkt,

$$|\psi^+\rangle = \frac{1}{\sqrt{2}}(|V\rangle|H\rangle + |H\rangle|V\rangle), \quad (4.5)$$

$$|\psi^-\rangle = \frac{1}{\sqrt{2}}(|V\rangle|H\rangle - |H\rangle|V\rangle), \quad (4.6)$$

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|H\rangle|H\rangle + |V\rangle|V\rangle), \quad (4.7)$$

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|H\rangle|H\rangle - |V\rangle|V\rangle). \quad (4.8)$$

Die Zustände, benannt nach dem Physiker John Bell, bilden zusammen eine Orthonormalbasis des Zustandsraumes der zwei Photonen $|a\rangle, |b\rangle$ (Qubits), wobei $|\psi^+\rangle, |\psi^-\rangle$ und $|\Phi^+\rangle, |\Phi^-\rangle$ grundsätzlich entgegengesetzte Messergebnisse für die beiden Photonen liefern (antikorrelierend) [7, p. 55]. Trennt man nun beide Qubits $|a\rangle, |b\rangle$ räumlich, wobei der Raum auch Lichtjahre betragen kann, und misst Alice das Qubit $|a\rangle$ (sei hier das Erste der beiden Qubits $|a\rangle$) des Gesamtzustands

$$\frac{1}{\sqrt{2}}(|0\rangle_a|0\rangle_b + |1\rangle_a|1\rangle_b) \quad (4.9)$$

und misst sie $|a\rangle = |0\rangle$, so verfällt der zweite Term ($\dots + |1\rangle_a|1\rangle_b$) und Bob erhält für seine Messung zwangsläufig $|b\rangle = |0\rangle$. Dabei spielt die Reihenfolge, wer zuerst die Messung durchführt, keine Rolle. Die Messung des ersten Qubits gibt die zweite Messung vor, sie wird nicht lokal beeinflusst. Diese Kopplung wurde 1935 von Albert Einstein als „*spukhafte Fernwirkung*“ (engl. *spooky action at a distance*) bezeichnet. Es widerspricht dem Lokalitätsprinzip. Dieses fordert, dass wenn zwei Objekte aufeinander wirken, diese in klassischer Auffassung in Verbindung stehen müssen (z.B. Direktkontakt, Medium, Feld, etc.). Sind zwei Objekte vollständig voneinander getrennt, dann können sie sich nicht gegenseitig beeinflussen. Einstein war überzeugt, dass es nicht möglich sei, dass diese Information instant übertragen wird. Er ging von „verborgenen Variablen“ (LHV, local hidden variables) aus, die die Messung vorherbestimmen würde noch bevor beide Qubits getrennt werden. Diese „verborgenen Variablen“ werden „bei der Geburt“ mitgegeben [47].

Alice und Bob erhalten von einer EPR-Quelle ein EPR-Paar mit antikorrelierend verschränkten Qubits, z.B.

$$|\psi^-\rangle_{a,b} = \frac{1}{\sqrt{2}}(|V\rangle_a|H\rangle_b - |H\rangle_a|V\rangle_b), \quad (4.10)$$

und messen diese mit den ihnen zu Verfügung stehenden Filtereinstellungen von

$$\begin{aligned} a_1 &= 0^\circ, a_2 = 45^\circ, a_3 = 22,5^\circ, \\ b_1 &= 0^\circ, b_2 = -22,5^\circ, b_3 = 22,5^\circ. \end{aligned}$$

Je nach gewählter Filtereinstellung werden nun Alice a_i und Bob b_j mit $i, j \in \{1,2,3\}$ Photonen detektieren oder nicht. Im antikorrelierenden Fall wird Bob bei der Filtereinstellung $b_1 = 0^\circ$ ein Photon messen während Alice mit $a_1 = 0^\circ$ kein Photon detektieren wird. Alice und Bob führen Protokoll über ihre Messergebnisse und die verwendeten Filtereinstellungen. Anschließend tauschen sich beide über einen alternativen Kanal über das gewählte Verfahren aus. Wählten sie dasselbe Verfahren (a_1, b_1) , (a_3, b_3) , so behalten sie das Messergebnis bei. Diese sind eindeutig wobei einer von Beiden seine Messergebnisse im antikorrelierenden Fall noch einmal invertieren muss. Stehen die Winkel beider Filtereinstellungen 45° zueinander ergibt sich dieselbe Problematik wie zuvor im BB84-Protokoll. Die Wahrscheinlichkeit der Detektion eines Photons beträgt 50% und ist somit abhängig vom Zufall. Diese Messwerte können von Alice und Bob nicht verwendet werden und sind somit zu streichen. Sollten Alice und Bob die Filter so gewählt haben, dass der Differenzwinkel $\delta = a_i - b_i = \pm 22,5^\circ$ oder $67,5^\circ$ beträgt, können sie die Bedingung S der CHSH-Ungleichung testen und herausfinden ob die Qubits maximal verschränkt waren oder ob Mallory durch einen Abhörversuch diese Eigenschaft gemindert hat oder versuchte statische Qubits zu verschicken und so einen Schlüssel zu diktieren²⁷.

4.2.3 Bewertung von Quantum-Key-Distribution-Verfahren

Die Quantenkryptographie beschäftigt sich mit einem sicheren Schlüsselaustausch auf Basis von Messung und dient nicht selbst der Verschlüsselung von

²⁷ Siehe Anlage „A26 Zwei-Teilchen-Systeme, Verschränkung, Bell und EPR“

Informationen. Die ausgehandelten Schlüssel können in quantencomputer-resistenten Verfahren, wie z.B. dem One-Time-Pad oder AES eingesetzt werden [27]. Die Sicherheit des Schlüsselaustauschs liegt in der Quantenphysik begründet. Es ist einem Angreifer nicht möglich Messungen vorzunehmen, ohne die Übertragung nachhaltig zu beeinflussen und so den Schlüsselaustausch zu korrumpieren. Zwar ist es durch Quantenteleportation möglich Informationen von einem Qubit auf ein anders zu übertragen (vgl. Quantenrepeater) jedoch verhindert das No-Cloning-Theorem [28], dass Mallory eine perfekte Kopie während der Übertragung anfertigt. Ein Eingriff kann stets durch Alice und Bob bemerkt werden. Ein wesentlicher Unterschied des E91-Protokolls gegenüber BB84 ist, dass der Schlüssel niemals übertragen wird. Das E91-Verfahren selbst generiert den Schlüssel während der Durchführung. Beide Verfahren können durchgeführt werden, wobei die Messung, und somit die Schlüsselerzeugung, zu einem späteren Zeitpunkt erfolgt. Das E91-Protokoll offenbart den Schlüssel erst zur Messung. Im BB84-Protokoll initialisiert Alice den Austausch durch eine Reihe zufälliger Schlüssel-Bit. Dieser Ausgangsschlüssel muss bis zur Messung aufbewahrt werden [7], was einen Sicherheitsnachteil darstellen kann. Ein weiterer Vorteil der Quantenkryptographie ist die Erzeugung echter Zufallszahlen.

Trotz aktueller Erfolge und der nachweisbaren Sicherheit beim Einsatz der Quantenkryptographie ergeben sich technische Hürden beim Einsatz der beschriebenen Verfahren insbesondere in der Reichweite und den Übertragungsgeschwindigkeiten. Auch gehen die vorgestellten Verfahren von Idealbedingungen aus, die in der Praxis heute noch nicht garantiert sind. Technische Geräte werden nahezu unter Laborbedingungen betrieben und sind im Allgemeinen noch nicht office- oder alltagstauglich²⁸ [27].

²⁸ Siehe Anlage „A27 Gedanken zur QKD“

4.3 Quantencomputerresistente Verfahren

4.3.1 Allgemein

Quantencomputer-resistente Verfahren, Post Quantenkryptographie, sind Verfahren, die ohne den Einsatz von Quantenmechanik auskommen. Sie stellen eine Reihe von alternativen mathematischen Problemen zum Faktorisierungs- und diskretem Logarithmus-Problem dar von denen man annimmt, dass sie auch bei einem Einsatz von Quantencomputern die Sicherheit gewährleisten. Zusätzlich ist auch Sicherheit gegen den Einsatz von klassischen Rechnersystemen gegeben. Betrachtet man erneut die P -, NP - und NP -vollständige Probleme, so sucht man nun nach Verfahren, die außerhalb des Bounded-Error Quantum Polynomial-Time, BQP, wie z.B. das Faktorisierungsproblem, liegen.

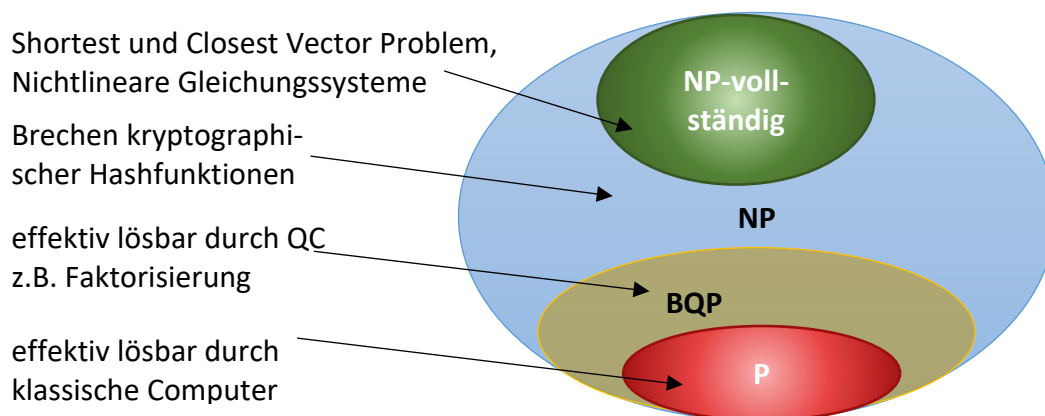


Bild 4.3 Darstellung von Problem-Klassen und der Möglichkeiten diese Effizient zu lösen nach [48].

Bereits vor der Veröffentlichung von RSA gab es Vorschläge für kryptographische Verfahren, die nicht auf dem Faktorisierungsproblem oder dem Problem des Lösens des diskreten Logarithmus beruhen. Aktuell werden primär Probleme der in der Tabelle (4.3) geführten Familien untersucht, auf die in den Folgekapiteln genauer eingegangen werden soll.

Tabelle 4.3 Aktuelle Familien in der Post Quantenkryptographie mit Einsatzmöglichkeiten nach [49, p. 10]. (*) Es gibt Vorschläge für den Einsatz in diesem Bereich, die jedoch noch nicht ausreichend untersucht oder bereits gebrochen sind.

Familie	Signatur	Encryption/ Key Exchange	Bemerkung
Hashbasiert	x	-	Limitierte Signaturen, basierend auf symmetrischen Hash-Funktionen und status-behafteten Schemata.
Multivariat	x	-(*)	Basiert auf multivariaten, quadratischen, nichtlinearen Gleichungen mit großen, öffentlichen Schlüsseln.
Codebasiert	-(*)	x	Älteste Familie, welche am besten untersucht ist, d.h. als sicher zu betrachten, jedoch sehr große Schlüssel. Je effizienter, desto unsicherer.
Gitterbasiert	x	x	Ebenfalls alte und gut untersuchte Verfahren mit guter Performance bei angemessenen Schlüsselgrößen.
Isogeniebasiert	-(*)	x	Neues Forschungsgebiet mit kleinen Ciphertexten und Schlüsseln.

4.3.2 Hashbasierte Kryptographie

Die hashbasierte Kryptographie bietet Lösungen für Signatursysteme, welche nicht durch einen Quantencomputer bedroht werden. Die Sicherheit der hashbasierten Kryptographie beruht auf der Kollisionsresistenz und der Resistenz gegenüber von Urbild-Angriffen (pre-image) ihrer Hashfunktionen, d.h. sie be-

sitzen Eigenschaften, bei denen es idealerweise unmöglich ist für zwei verschiedene Eingaben x, x' denselben Hashwert y zu generieren. Da Hashfunktionen gründlich untersuchte kryptographische Primitive sind, zählt die hashbasierte Kryptographie zu den bereits heute vertrauenswürdigsten Teilgebieten der Post Quantenkryptographie [44]. Am 14. Dezember 1979 veröffentlichte Ralph C. Merkle ein Papier zum Thema „A certified digital signature“²⁹, welches ein digitales Signatursystem beschreibt, das mit überschaubaren Ressourcen digitale Signaturen realisiert. Er nannte diese neue Methode „tree signatures“ [50].

Mit Hashfunktionen können die Schutzziele Integrität, Authentizität und Nicht-Abstreitbarkeit realisiert werden. Realisiert man alle drei Schutzziele in einem Verfahren spricht man von einer digitalen Signatur. Während die Integrität von Daten allein mittels der übertragenden Daten und ohne ein zusätzliches Geheimnis realisiert werden kann, benötigt die Authentizität den Einsatz von privaten und öffentlichen Schlüssel k_{privat}, k_{public} . So kann sichergestellt werden, dass eine Nachricht nicht von einem Außenstehenden stammt oder verändert wurde. Die Nicht-Abstreitbarkeit benötigt ebenfalls ein korrespondierendes Schlüsselpaar (digitale Unterschrift), welche eindeutig bei einem Absender verortet werden kann (passender öffentlicher zu privatem Schlüssel).

Hashfunktionen liegen Einweg-Funktionen C zugrunde, die einfach zu berechnen jedoch schwer zu invertieren sind. Bei gegebenen x ist y für $C(x) = y$ berechenbar jedoch ist die Berechnung von x mit $C^{-1}(y) = x$ idealerweise unmöglich. Die Anforderungen an eine Einweg-Funktion lauten

- die Einwegfunktion C kann auf alle Eingaben angewendet werden,
- sie generiert bei einer beliebigen Eingabe eine Ausgabe fixer Größe,
- die Einwegfunktion C mit $C(x) = y$ ist einfach zu berechnen,
- die Einwegfunktion C ist nicht umkehrbar und resistent gegen Urbild- bzw. pre-image Angriffe (pre-image und second pre-image),

²⁹ <http://www.merkle.com/papers/Certified1979.pdf>

- es ist idealerweise unmöglich eine zweite von x verschiedene Eingabe x' zu finden für die gilt $C(x) = C(x') = y$.

Da gerade letzteres nicht ausgeschlossen werden kann gilt, dass für eine Funktion $C(k, p) = c$ mit $length(p) = length(c) \leq length(k)$ die Eigenschaften,

- dass die durchschnittliche Berechnungszeit zum Finden der vier Werte k, k', p, c für $C(k, p) = c = C(k', p)$ mit $k \neq k'$ größer sein muss als $2^{\frac{length(p)}{2}}$,
- sowie einige Bedingungen an die zuvor genannten Parameter, nachlesbar unter [50, p. 6],

gelten müssen.

Heutige quantencomputer-resistente Signaturverfahren basieren auf sogenannten One-Time-Signature-Verfahren (OTS-Verfahren), bei denen Signaturen lediglich einmal für eine Nachricht verwendet werden können. Jede weitere Anwendung ermöglicht es einem Angreifer Informationen über den geheimen Schlüssel des Absenders zu gewinnen. Eine erste Strategie wurde im Lamport-Diffie One-Time-Signature, LD-OTS, als Schema für digitale Signaturen von Leslie Lamport und Whitfield Diffie im Jahr 1979 entwickelt³⁰. Es ist jedoch nicht praktikabel und wurde von Ralph Merkle unter dem Namen W-OTS, benannt nach Robert Winternitz, praxistauglicher gestaltet³¹.

Das eXtended Merkle Signature Scheme Verfahren XMSS wurde von der TU Darmstadt entwickelt und gemeinsam mit dem IT-Sicherheitsunternehmen genua GmbH zur Praxistauglichkeit gebracht. XMSS wurde über einen Zeitraum von 15 Jahren entwickelt und nun, gemeinsam mit der TU Eindhoven, als Standard eingereicht (IETF-Spezifikation). XMSS wurde im Jahr 2018 unter dem defacto Standard RFC 8391 veröffentlicht (Request for Comments)³² und

³⁰ Siehe Anlage „A28 Lamport-Diffie One Time Signature, LD-OTS“

³¹ Siehe Anlage „A29 Winternitz One Time Signature, W-OTS“

³² Siehe Anlage „A30 eXtended Merkle Signature Scheme, XMSS“

stellt dabei einen Container da, in dem eine Hashfunktion eingesetzt wird. Ist diese nicht sicher, so kann sie einfach ausgetauscht werden. Damit kommt XMSS ohne zusätzliche mathematische Hürde aus und die Sicherheit hängt nur von der Hash-Funktion ab. Auf diese Weise ist das Verfahren unabhängig und läuft nicht Gefahr, wie beispielsweise das diskrete Logarithmus Problem, später als Ganzes gelöst zu werden [51] [52]. Das Verfahren basiert auf einem binären Hash-Baum (Merkle-Tree) und generiert die Schlüssel mittels W-OTS⁺. Bezogen auf die Höhe eines solchen Baumes, wobei seine Wurzel, auch root, den öffentlichen Schlüssel k_{public} darstellt, werden seine Blätter zum Signieren verwendet. Die Menge der signierbaren Nachrichten m steht somit im direkten Zusammenhang der Höhe h des Hash-Baumes in diesem OTS-Verfahren und berechnet sich mit 2^h für $h \geq 2$.

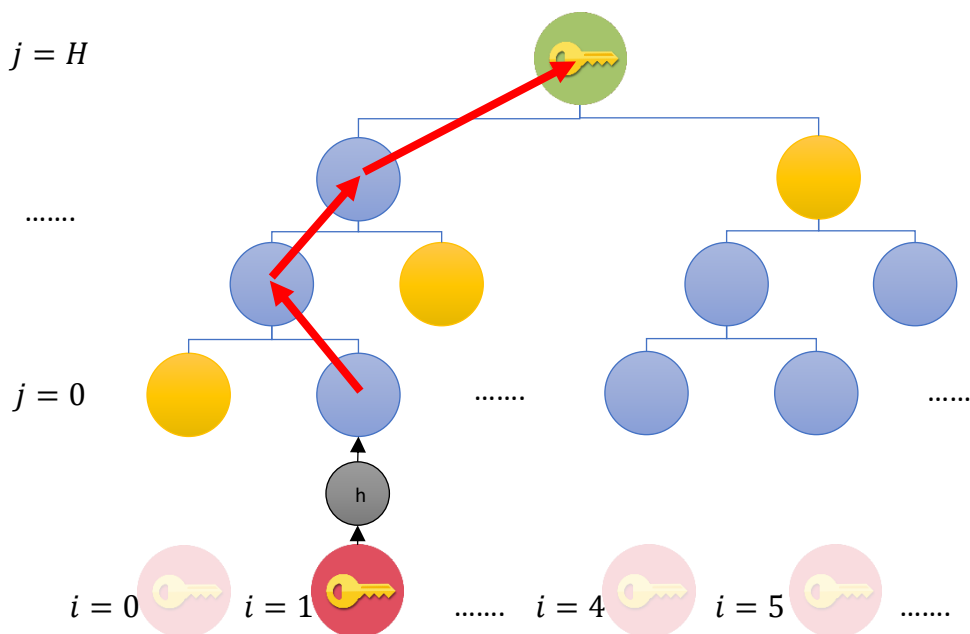


Bild 4.4 Verifikation in einem Merkle-Baum. Die gelb eingefärbten Knoten sind die notwendigen Pfadangaben und Bestandteil einer Signatur, um den öffentlichen Schlüssel, den Root-Knoten, zu bestätigen.

Die Verifikation einer Nachricht bedingt die Nachricht selber, die Signatur sowie die Angabe eines Pfades $auth = (auth_0, \dots, auth_{h-1})$ im Hash-Baum um den öffentlichen Schlüssel k_{public} zu berechnen. Seit Aufkommen der Post Quantenkryptographie werden hashbasierte Verfahren seit den 1990er Jahren verbessert und optimiert (Mehrschichtige Bäume bzw. Multi-Trees, Hyper-

Trees oder Tree-Chaining z.B. XMSS^{MT}). Für einige Schemen reicht die zweite pre-image Resistenz, was theoretisch sogar die Anwendung von MD5 ermöglichen würde, da es hier bis heute keine bekannte Attacke dafür gibt. Eine Brute-Force Attacke mittels Grover Algorithmus würde die Sicherheit von z.B. SHA-512 auf SHA-256 senken, was entsprechend in künftige Sicherheitsbetrachtungen aufgenommen werden muss. Für OTS-Verfahren ist ein Managementsystem bezüglich verwendeter Signaturen wichtig, um zu vermeiden Signaturen doppelt zu verwenden (Zustandsbehaftet).

Ferner gibt es zustandslose Verfahren wie SPHINCS³³. SPHINCS wurde 2014 erstmalig von Daniel J. Bernstein et al. vorgestellt und ermöglicht die Wiederverwendung zufällig generierter Schlüssel für einige Male. Neben dem bereits als RFC veröffentlichten Verfahren XMSS und XMSSTM sollen ebenfalls LMS (Leighton-Micali Hierarchical Signature System) und HSS (Hierarchical Signature System) als RFC veröffentlicht werden. SPHINCS⁺ wurde als Kandidat für eine NIST-Standardisierung eingereicht [53]. Neben dem Vorteil der Sicherheit gegenüber Quantencomputern haben aktuelle hashbasierte Signaturverfahren noch einige Nachteile. Zum einen haben sie zum Teil höhere Laufzeiten, zum anderen sind die Schlüssel- und Signaturgrößen im Vergleich zu den klassischen Verfahren RSA und ECDSA größer und erfordern ein angesprochenes Signatur-Managementsystem.

Tabelle 4.4 Vergleich von implementierten Signaturverfahren nach [54, p. 35]. Insbesondere in den Signaturgrößen gibt es Unterschiede.

	Signatur- Größe	Public Key	Secret Key	Security (Klas- sisch/QC)
XMSS(SHA2- 256_W16_H10)	2.500	64	132	256/128
XMSS^{MT}(SHA2- 256_W16_H20_D2)	4.964	64	132	256/128
SPHINCS-256	41.000	1.056	1.088	<256/<128
RSA-3072	384	384	1.728	128/Gebrochen

³³ <http://sphincs.cr.yp.to/index.html>

ECDSA (P-256) 64 64 96 128/Gebrochen

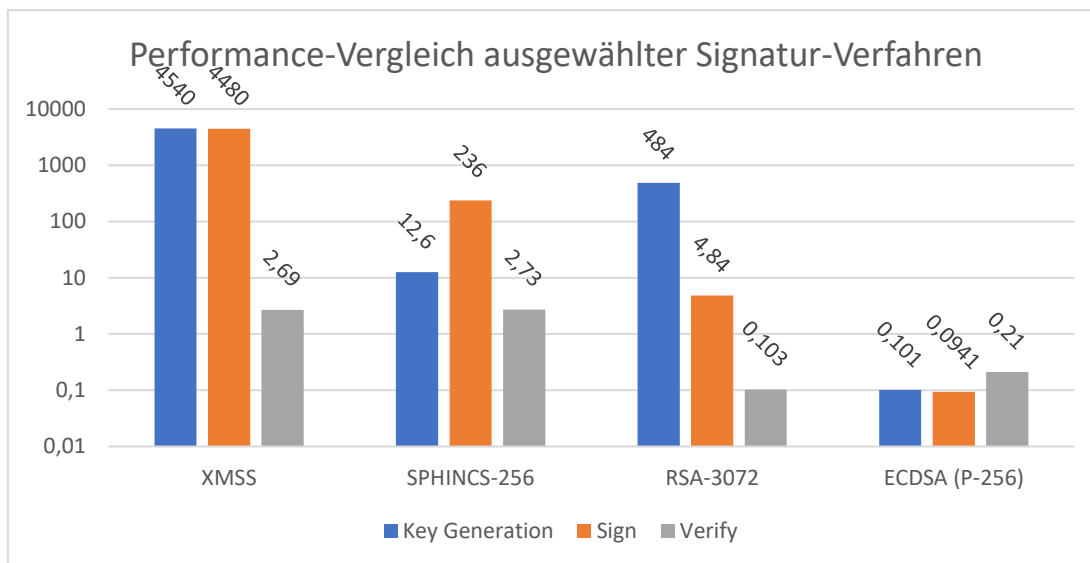


Bild 4.5 Vergleich durchschnittlicher Laufzeiten in Millisekunden der post Quantenverfahren XMSS und SPHINCS in Vergleich zu schnellen Implementierungen von RSA und ECDSA nach [54, p. 41] auf einem logarithmischen Maßstab.

4.3.3 Gitterbasierte Kryptographie

Gitterbasierte kryptographische Verfahren (engl. Lattice-based cryptography, LBC) sind Verfahren die sowohl als Signatur- als auch als Schlüsselaustausch- und Verschlüsselungsverfahren eingesetzt werden können [49]. Es gibt bereits Verfahren in der Praxis, die effizient eingesetzt werden. Das NTRUEncrypt existiert seit 1996 und hat sich bis heute bewährt, so dass es bisher keine signifikante Attacke auf das Verfahren gibt. Dabei können LBC-Schemen RSA/ECDSA-Schemen adäquat ersetzen oder gar übertreffen. LBC- Schemen stellen die größte Familie innerhalb der PQC da [55].

Tabelle 4.5 Übersicht über die PQC-Familien und ihrer Anwendungsbereiche bezüglich NIST [55].

Familie	Signatur	Verschlüsselung/KEM	Total
Lattice-Based	5	23	28
Code-Based	3	17	20
Multivariate	8	2	10

Hash-Based	3	0	3
Isogeny-Based	0	1	1
Andere	2	5	7
Total	21	48	67

Gitter sind ein endlicher Satz an linear unabhängigen Vektoren $G = \{g_1, g_2, \dots, g_n\}, n \in \mathbb{N}_0$, in einem n -Dimensionalen reellen Raum \mathbb{R}^n . Ein Gitter wird bezeichnet als $\Lambda \subset \mathbb{R}^n$ mit der Basis $G \subset \Lambda$ [56], d.h.

$$\Lambda = \Lambda(G) = G\mathbb{Z}^n := \{\sum_{i=1}^n z_i g_i \mid z_i \in \mathbb{Z}\} = \mathbb{Z}g_1 \oplus \dots \oplus \mathbb{Z}g_n. \quad (4.11)$$

Ist ein Gitter vom Rang 2, dann besitzt das Gitter unendlich viele Basen, wenn eine unimodulare Matrix $U \in \mathbb{Z}^{n \times n}$ existiert, $G' = GU$. Man kann zwischen „guten Basen“ und „schlechten Basen“ unterscheiden. Bei einer guten Basis stehen zwei Vektoren annähernd orthogonal zueinander, z.B. (g_1, g_2) , bei einer schlechten verlaufen sie annähernd parallel, z.B. (h_1, h_2) [52].

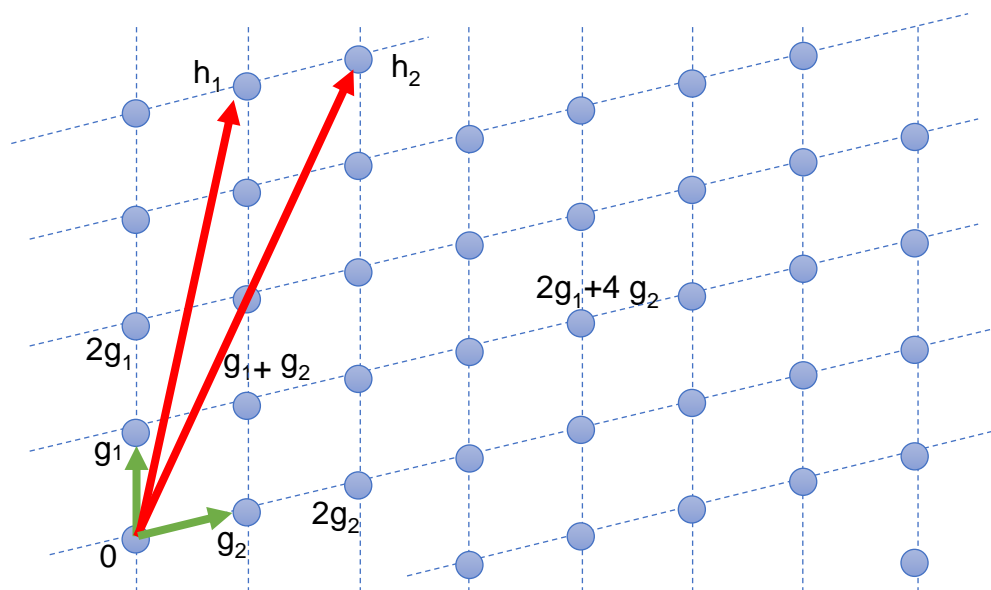


Bild 4.6 2-Dimensionales Beispielgitter mit der Basis (g_1, g_2) und (h_1, h_2) als vereinfachte Darstellung für das Verstehen annähernd senkrechter Vektoren und parallel verlaufenden Vektoren.

Auf dieser Art von Gittern gibt es schwer zu lösende Probleme, sogar für Quantencomputer. Sie sind geometrischer Natur und werden in die Kategorien

- Shortest Vector Problem, SVP,
- Closest Vector Problem, CVP, aber auch das
- Ring-LWE (Learning With Error) Problem

eingeteilt. Für diese Probleme sind keine effizienten Algorithmen bekannt. Das CVP ist NP -schwer. Ob SVP ebenfalls NP -schwer ist, ist eine offene Frage [38, p. 35].

Beim Closest Vector Problem ist ein Vektor a gegeben, der nicht Bestandteil eines Gitters Λ ist. Ziel ist es den am nächsten gelegenen Gitterpunkt, im Beispiel u , zu finden, so dass $|u - a|$ minimal ist³⁴.

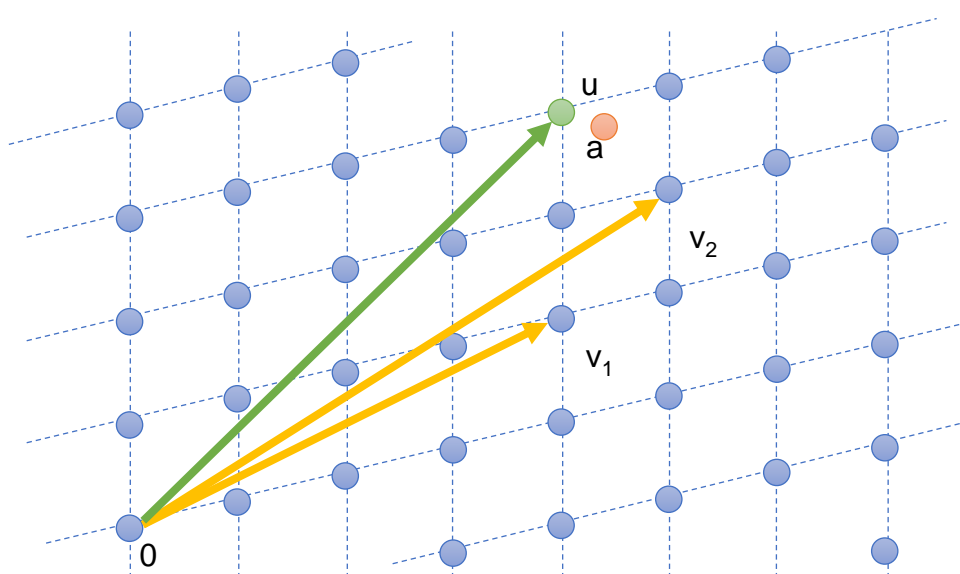


Bild 4.7 Beispielhafte Darstellung des Closest Vector Problem. Der Closest Vector u ist grün dargestellt und Bestandteil des Gitters mit der Basis (v_1, v_2) . Der Vector a muss das nicht zwingend sein.

Der LLL-Algorithmus, benannt nach Lenstra, Lenstra und Lovász, berechnet eine $2^{(d-1)/2}$ -Approximation für kürzeste Vektoren in Gittern der Dimension d . Es hat sich jedoch empirisch herausgestellt, dass der LLL-Algorithmus für Gitter geringer Dimensionen, etwa $1 \leq d \leq 100$, deutlich bessere Approximationen liefert [57]. Babai's Closest Vector-Algorithmus³⁵ versucht genau diese

³⁴ Siehe Anlage „A31 Aufbau und Eigenschaften von Gittern“

³⁵ Siehe Anlage „A32 Babai's Closest Vector-Algorithmus“

Probleme zu lösen. SVP kann dabei als ein Spezialfall des CVP betrachtet werden, bei dem der gesuchte Punkt bzw. Vektor $w = 0$ gesetzt wird um anschließend den nächsten Gitterpunkt zu suchen. Darauf aufbauend entwickelten Oded Goldreich, Shafri Goldwasser und Shai Halevi im Jahr 1997 das nach ihnen benannte GGH-Kryptosystem³⁶, welches sich jedoch als unsicher erwiesen hat. Somit kommt es als PQC-Verfahren nicht in Frage. Dennoch vermittelt es einen Eindruck von Problemen, die auf Gittern beruhen [52].

Heutige gitterbasierte Verfahren wie NTRU (NTRUEncrypt³⁷, NTRU-Sign), NewHope und Frodo sind durchaus in der Lage RSA und Elliptic Curve Cryptography zu ersetzen.

Die Schlüsselgrößen dieser Verfahren entsprechen bei gleicher Sicherheitsstufe ungefähr den Schlüsseln von RSA. Lediglich ECC verwendet kleinere Schlüssel. Ein Vorteil von NTRU sind die schnellen Ver- und Entschlüsselungswerte, die 10 bis 500-fach performanter sind als RSA. Ein Nachteil gegenüber RSA sind die höhere Komplexität des Verfahrens, insbesondere im NTRU-Sign, die kompliziertere Wahl der Parameter und nicht zuletzt, dass es sich um ein patentiertes Verfahren handelt. Auch besteht die Möglichkeit, dass NTRU bei korrektem Einsatz falsche Ergebnisse liefern kann [4].

Tabelle 4.6 Vergleich von Schlüsselgrößen der Verfahren NTRU, RSA und ECC [58, p. 22].

Sicherheitsniveau in Bit	n	NTRU Key in Bit	RSA Key in Bit	ECC Key in Bit
80	251	2008	1024	163
112	347	3033	2048	224
128	397	3501	3072	256
160	491	4383	4096	320
192	587	5193	7680	384
256	787	7690	15360	512

³⁶ Siehe Anlage „A33 Das GGH-Kryptosystem“

³⁷ Siehe Anlage „A34 NTRUEncrypt“

Neben den hier vorgestellten Verschlüsselungsverfahren, basierend auf Gittern, können Gitterverfahren auch Signaturen erstellen. Damit nehmen gitterbasierte Verfahren wie z.B. NTRU denselben Lebensraum ein wie z.B. RSA.

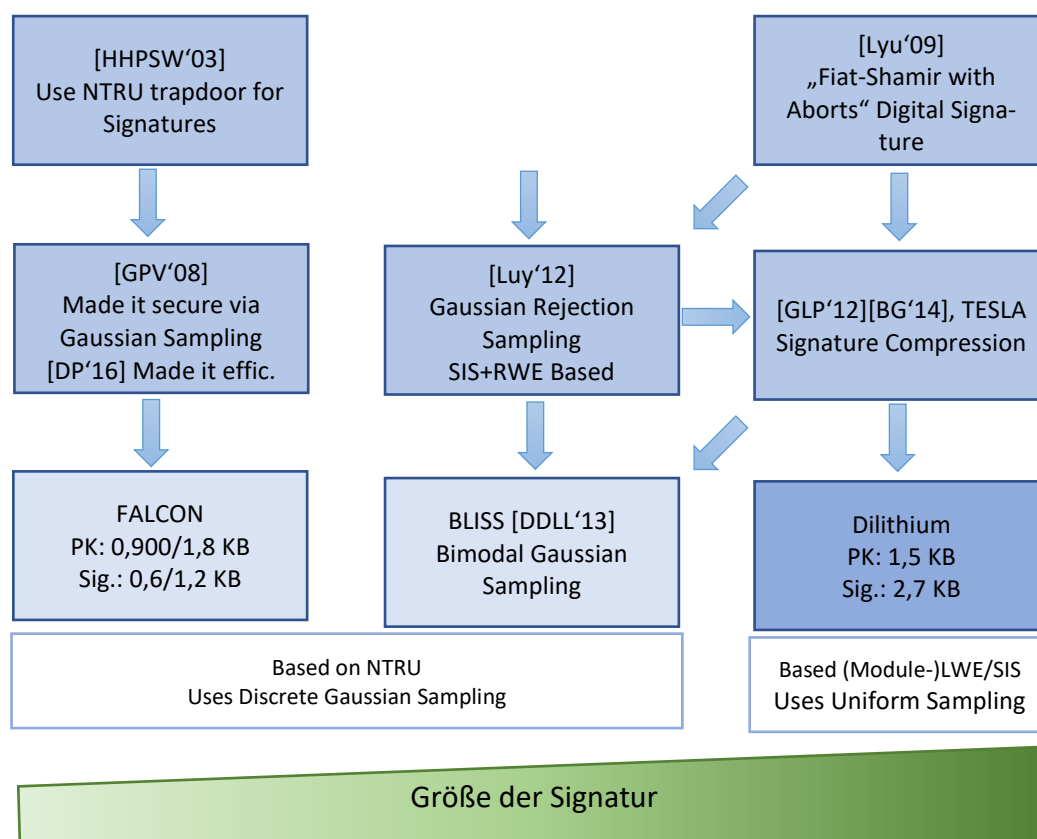


Bild 4.8 Übersicht gitterbasierter digitaler Signatur-Systeme nach [59, p. 43].

Gitterbasierte Verfahren sind die am meisten analysierten Post Quanten-Verfahren sowohl gegen klassische- als auch gegen Quantenalgorithm. Viele Spezialisten u. A. Lovasz, Lenstra H., Lenstra A., Babai, Schnorr, Kupferschmied, Shamir, Regev und nicht zuletzt Shor arbeiteten an Gitteralgorithmen oder Angriffen gegen ein Gitter-Schema, wobei es keine bahnbrechende neue Technik seit LLL gab. Es wird allgemein angenommen, dass sich die Kryptoanalyse unter Verwendung bekannter Techniken einer unteren Grenze annähert [59]. Mit BLISS, NewHope, Frodo und weiteren Verfahren sind bereits heute weitere, praktische Verfahren im Einsatz. NTRUEncrypt existiert seit 1996 und bisher ist kein signifikanter Angriff bekannt.

Das Verfahren NewHope wurde bereits in einem Pilotprojekt von Google in Kombination ihres Browsers Chrome und eigenen Google-Domains für den Schlüsselaustausch verwendet. NewHope kommt dabei in Kombination mit ECDH-Verfahren zum Einsatz. NewHope basiert auf dem Ring-Learning-With-Errors-Problem (R-LWE). Ziel ist das Erbnen besserer Untersuchungen derartiger Verfahren. [60].

Gitter können noch einmal differenziert betrachtet werden, zum einen in ideale und Standardgitter. Während Standardgitter eine höhere Sicherheit bieten benötigen sie jedoch eine erhöhte Rechenzeit bei großen Matrizen. Matrix-Vektor Multiplikationen haben dabei eine quadratische Komplexität. Ideale Gitter sind effizienter, bieten jedoch dafür weniger Sicherheit [55].

4.3.4 Codebasierte Kryptographie

Codebasierte kryptographische Verfahren sind ebenfalls im Bereich der Verschlüsselung und des Schlüsselaustausches angesiedelt. Sie sind der älteste Zweig der PQC-Verfahren. Beispielsweise wurde das McEliece Kryptosystem bereits im Jahr 1978 vorgestellt. Die zugrundeliegende Idee ist, dass es für einige spezielle lineare Codes³⁸ effiziente Fehlerkorrekturalgorithmen gibt, die als kryptographische Primitive (mit zugrundeliegende Einweg-Funktion) genutzt werden. Das Dekodieren von fehlerbehafteten Codes ohne das zugehörige Geheimnis ist im allgemeinen Fall nicht lösbar. Für einen Angreifer ist es daher nicht möglich die Nachricht ohne den privaten Schlüssel zu decodieren und dabei einen Fehler e zu entfernen. Eine Nachricht m ist somit im Allgemeinen ein Codewort c mit bestimmtem Fehler e [61]. Das Codewort c wird mittels Generator G erzeugt. Historisch kommen fehlerkorrigierende Codes aus der Untersuchung ob Übertragungsfehler bezüglich Informationen auf einem störanfälligen Kanal vorliegen und ob diese korrigiert werden können, $c + e = x \rightarrow \text{Korrektur}(x) = c$. Bekannte Vertreter der codebasierten Kryptographie sind das McEliece- und das Niederreiter-Kryptosystem unter Verwendung des Goppa-Codes, ein Typ von linearem Code, die u. A. die Eigenschaft der

³⁸ Siehe Anlage „A35 Codierung mit linearen Codes“

Fehlererkennung und -korrektur ausweisen. Durch besser strukturierte Codes konnten die Schlüsselgrößen reduziert werden z.B. im QC-MDPC, Quasi-Cyclic Moderate Density Parity-Check. Das geschah jedoch auf Kosten der Sicherheit [62]. Das McEliece-Verfahren besteht aus dem privaten Schlüssel

- Generatormatrix $G \in \mathbb{F}_2^{k \times n}$ basierend auf dem Tripel (n, k, d) , das maximal t Fehler korrigieren kann und den Code C erzeugt. G muss invertierbar sein mit $GG^{-1} = I$, was der Einheitsmatrix entspricht (Goppa-Code Matrix),
- einer Permutationsmatrix $P \in \mathbb{F}_2^{n \times n}$,
- sowie einer invertierbaren Matrix $S \in \mathbb{F}_2^{k \times k}$, auch „Scrambler“ Matrix.

Der öffentliche Schlüssel G' leitet sich aus den drei Matrizen ab als

$$G' = SGP \in \mathbb{F}_2^{k \times n}. \quad (4.12)$$

Die zu verschickende Nachricht $m \in \mathbb{F}_2^k$ mit dem Fehler $e \in \mathbb{F}_2^n$ über eine maximale Fehlerschwelle t wird als $x = mG' + e$ berechnet. Der Fehler e wird mittels Goppa-Codes korrigiert. Die Nachricht m kann durch die inversen Matrizen S^{-1}, G^{-1} wiederhergestellt werden³⁹.

Das Verfahren ist sicher gegen Angriffe sowohl von klassischen als auch von Quantencomputern. Wenn man jedoch das allgemeine Decodierungsproblem lösen kann, dann kann auch das McEliece-Problem gelöst werden. Grover's Algorithmus kann hierfür verwendet werden. Jedoch würde er das Finden einer Lösung nur ungefähr quadratisch beschleunigen. Die Laufzeit wäre immer noch exponentiell [63, p. 16]. Auch sind Ver- und Entschlüsselungsvorgänge vergleichsweise schnell. Dennoch findet das Verfahren aufgrund von Schlüsselgrößen im Megabytebereich wenig Anwendung [64].

In einem NIST-Vorschlag, eingereicht von Bernstein et al im November 2017, sind Parameter der folgenden Tabelle geführt.

³⁹ Siehe Anlage „A36 McEliece-Kryptosystem“

Tabelle 4.7 Notwendige Parameter sowie erwartete Größen der Texte in Bit, des öffentlichen k_{pb} und privaten Schlüssel k_{pv} um der NIST-Anforderung an ein IND-CCA2 KEM, Cat. 5 zu genügen, was ein Sicherheitslevel von 256 Bit fordert [63, p. 17].

kem/	k	n	t	Plaintext	Cipher	k_{pb}	k_{pv}
McEliece							
6906119	5.413	6.960	119	677B	870B	4,5MB	13,75MB
8192128	6.528	8.192	128	870B	1024B	6,4MB	19,45MB

Es ist jedoch möglich die Schlüsselgrößen weiter zu optimieren durch Anpassungen oder z.B. der Verwendung des QC-MDPC.

Tabelle 4.8 Reduzierung der Schlüsselgrößen durch Optimierungen nach [63].

kem/mceliece	k	n	t	k_{pb}	k_{pv}	k_{pb}	k_{pv}
6906119	5.413	6.960	119	4,5MB	13,75MB	1,3MB	4,8MB
8192128	6.528	8.192	128	6,4MB	19,45MB	1,6MB	6,7MB

Durch die Anwendung geeigneter Parameter kann so ein ausreichendes Sicherheitsniveau bei optimierter Schlüsselgröße erreicht werden. Durch die mehrfache Wiederholung mit weiteren Zufallselementen (k -Repetition) mit weiteren leichten Anpassungen ist das Verfahren IND-CCA1 sicher. Aktuell gibt es, vom Standpunkt der Sicherheit her betrachtet, einige gute Vorschläge, von denen auch diverse bei der NIST eingereicht wurden. Während QC-MDPC nicht mehr vertreten ist konnte sich das Classic McEliece-Verfahren für die zweite Runde qualifizieren⁴⁰. Dennoch sind weitere Anstrengungen im Bereich der Kryptoanalyse erforderlich sowie das Ziel Schlüsselgrößen auf eine praktikablere Größe zu reduzieren.

⁴⁰ <https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions>

4.3.5 Multivariate Kryptographie

Multivariate Kryptographie nutzt nichtlineare Gleichungssysteme, die im Allgemeinen schwer, im Spezialfall unter Ausnutzen einer Trapdoor (Hintertür) einfach zu lösen sind. Die Lösung von multivariaten quadratischen Gleichungen ist im Allgemeinen ein schwierig zu lösendes Problem und NP -schwer. Es ist vermutlich weder mit klassischen noch mit Quantencomputern effizient lösbar. Multivariate Kryptosysteme versuchen die algebraische Geometrie des 20. Jahrhunderts zu verwenden [65]. Für kryptographische Verfahren wird ein solches Gleichungssystem gezielt konstruiert und mit einer Hintertür versehen. Mit dieser geheimen Information lässt sich dann das Gleichungssystem einfach lösen [61].

Das MQ-Problem besteht in der Abbildung zwischen endlichen Körper $K = \mathbb{F}_q$ mit einem eindeutig bestimmbar Erweiterungskörper $L = \mathbb{F}_q^n$ über K mit $n \in \mathbb{N}$, und einem System von m quadratischen Gleichungen $P = p_1, \dots, p_m$ mit n unbekannt x_1, \dots, x_n , welche als Einwegfunktion bezeichnet wird, $P: K^n \rightarrow K^n$.

$$P = \begin{pmatrix} p_1(x_1, \dots, x_n) = y_1 \\ \dots \\ p_n(x_1, \dots, x_n) = y_n \end{pmatrix} \quad (4.13)$$

Es ist jedoch nicht sicher ob das der Fall ist. Dennoch basiert die Sicherheit des Verfahrens auf dieser Annahme. Zwischen dem Grundkörper K und dem Erweiterungskörper L besteht der Isomorphismus $\phi: L \rightarrow K^n$ und es ist möglich Abbildungen vorzunehmen (strukturerehaltende, umkehrbar eindeutige Abbildung). Der Zusammenhang zwischen beiden Körpern bleibt geheim. Eine univariate Funktion $\varphi: L \rightarrow L$, d.h. abhängig von nur einer Variablen, stellt die Hintertür da und ist innerhalb von P , dem öffentlichen Schlüssel, versteckt sowie leicht umkehrbar. Um das Verfahren nicht vom Isomorphismus abhängig zu machen werden weitere affin bijektive Abbildungen S, T mit dem Isomor-

phismus und φ verknüpft, diese stellen den privaten Schlüssel bei multivariaten kryptographischen Systemen da, während sich der öffentliche Schlüssel P mit

$$P = T \circ \varphi_{K^n} \circ S := T \circ \phi \circ \varphi \circ \phi^{-1} \circ S \quad (4.14)$$

berechnet.

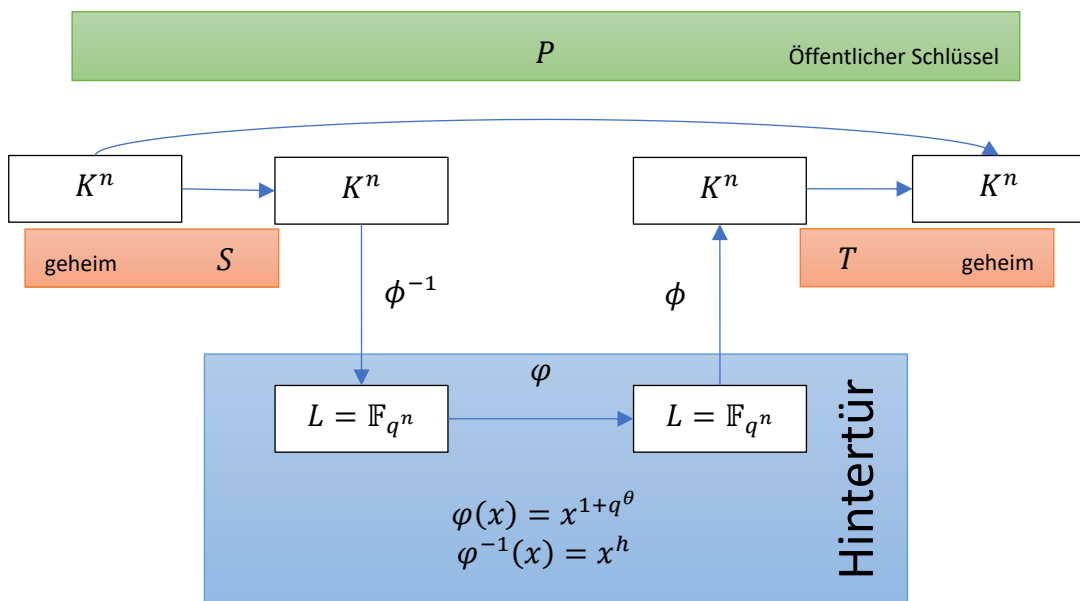


Bild 4.9 Darstellung des Matsumoto-Imai-Schema C* mit geheimen und öffentlichen Schlüssel sowie der versteckten Hintertür.

Der Angriff von Patatin auf das C*-Verfahren zeigte spektakulär wie derartige Probleme gelöst werden können auch ohne die Verwendung der in den Verfahren vorgesehenen Schlüssel. Er entdeckte, dass sich die im von Matsumoto-Imai⁴¹ vorgeschlagene Funktion nicht für ein derartiges System eignet und schlug seinerseits das Hidden Field Equations HFE-Verfahren vor, was bis heute nicht gebrochen wurde, so dass man sagen kann, dass das Verfahren von Matsumoto und Imai im Jahr 1988 der eigentliche Durchbruch der multivariaten Kryptosysteme darstellt. Ein Vorteil von multivariaten Verfahren

⁴¹ Siehe Anlage „A37 Das Matsumoto-Imai Kryptoschema C*“

ist die Möglichkeit der ressourcenschonenden Implementierung, was sich insbesondere bei Smartcards auszahlt. Die Signaturen sind vergleichsweise kurz und schnell in der Berechnung. HFE⁴²-Signaturen von ungefähr 128 Bit können durchaus als sicher angesehen werden. Im Vergleich dazu ist RSA 512 nicht mehr ausreichend [66] [49]. Nachteilig sind die im Vergleich zu anderen Verfahren großen öffentlichen Schlüssel, z.B. 100 kBit für ein Sicherheitsniveau von 100 Bit, verglichen mit RSA 1.776 Bit [67, p. 155], [68, p. 34]. Das stellt insbesondere ein Problem mit Smartcards und Geräten mit vergleichsweise wenig Ressourcen da, nicht jedoch mit Standardrechnern. Dasselbe Problem ergibt sich bei reduzierten Übertragungsraten einhergehend mit großen Mengen von Nachrichten.

Multivariate quadratische kryptographische Verfahren können sowohl als Signatur- als auch Verschlüsselungsverfahren eingesetzt werden. Für beide Einsatzgebiete gibt es Vorschläge. Auch gibt es mit Double Matrix Exponentiation DME einen Vorschlag, der beide Bereiche abdecken könnte. Im aktuellen NIST Standardisierungsprozess haben es jedoch nur Signaturverfahren in Runde zwei⁴³ geschafft. Es sind die Signaturverfahren GeMSS, LOUV, MQDSS sowie Rainbow [49, p. 10].

4.3.6 Supersingulare isogeniebasierte Kryptographie

Supersingulare isogeniebasierte kryptographische Verfahren sind mit die jüngste Familie kryptographischer Verfahren, die gegen den Einsatz von Quantencomputer resistent sind. Die Schlüsselgrößen, im Vergleich zu anderen Post Quantenverfahren, sind klein. Eine Größe von 330 Byte entsprechen einem Sicherheitsniveau von 128 Bit [69]. Nachteilig sind die vergleichsweise hohen Laufzeiten für einen vollständigen Schlüsselaustausch. Das Verfahren beruht auf Isomorphismus zwischen elliptischen Kurven bei denen mittels Isogenie φ eine elliptische Kurve E_1 auf E_2 abgebildet wird, d.h. $\varphi: E_1 \rightarrow E_2$ erfüllt den Homomorphismus für die beiden abelschen Varietäten E_1 und E_2 .

⁴² Siehe Anlage „A38 Hidden Field Equations HFE“

⁴³ <https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-2-Submissions>

Während ECC-Verfahren⁴⁴ eine versteckte Beziehung zwischen zwei Punkten auf derselben elliptischen Kurve pflegen ist bei z.B. Supersingular Isogeny Diffie-Hellman SIDH⁴⁵ die Beziehung zwischen zwei elliptischen Kurven innerhalb eines Graphen im Fokus.

Tabelle 4.9 Tabellarischer Vergleich der Diffie-Hellman-Schlüsselaustausch-Implementierungen nach [69].

	Classic DH	Elliptic Curve DH	Supersingular Isogeny DH
Elemente	Zahlen g mod Prime	Punkte P auf Kurve	Kurven E einer Isogenie-Klasse
Geheimnis	Exponent x	Skalar k	Isogenie φ
Berechnung	$g, x \rightarrow g^x$	$k, P \rightarrow [k]P$	$\varphi, E \rightarrow \varphi(E)$
Bekannt	g, g^x	$P, [k]P$	$E, \varphi(E)$
Schwierigkeit	Ermitteln von x	Ermitteln von k	Ermitteln von φ

Im isogeniebasierten Diffie-Hellman-Schlüsselaustausch ermitteln Alice und Bob, ausgehend von einer elliptischen Kurve E_0 , analog zum klassischen Diffie-Hellman, eine gemeinsame elliptische Kurve E_{AB} . Dabei stellt ihre zufällig gewählte Isogenie φ im Graphen (auch Random-Walk durch den Graphen) das Geheimnis da. Der Graph besteht aus einer Reihe über Kanten verbundener elliptischer Kurven. Alice und Bob bewegen sich, ausgehend von E_0 , mittels Random-Walk jeweils unterschiedlich zu den Kurven E_A, E_B und tauschen diese untereinander aus. Anschließend gehen sie denselben Pfad erneut, jedoch diesmal beginnend von den ausgetauschten Startpunkten, und erhalten eine gemeinsame Kurve E_{AB}, E_{BA} , die wenn nicht gleich, jedoch isomorph zueinander sind.

⁴⁴ Siehe Anlage „A39 Elliptic Curve Cryptography, ECC“

⁴⁵ Siehe Anlage „A40 Supersingular Isogeny Diffie-Hellman, SIDH“

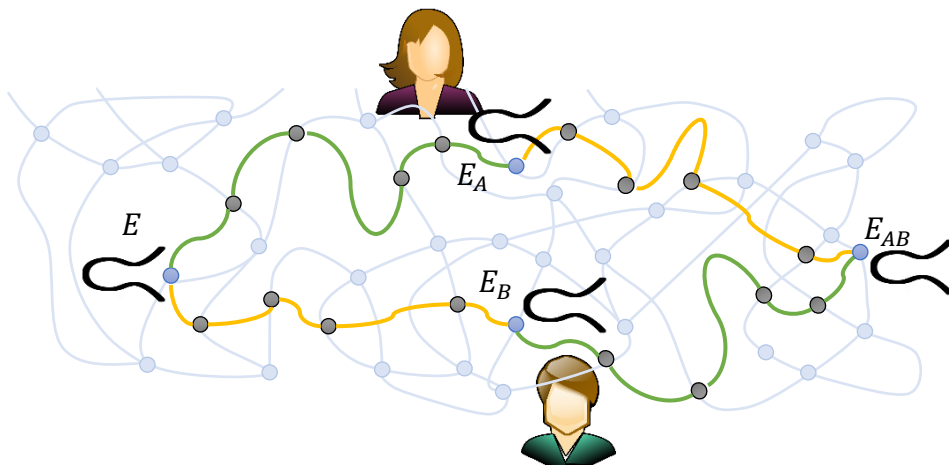


Bild 4.10 Berechnung der gemeinsamen Kurve E_{AB} analog zum Diffie-Hellman-Schlüsselaustausch ohne die Verwendung des diskreten Logarithmusproblems.

Die Schwierigkeit und somit die Sicherheit des Verfahrens beruht auf den Isogenien φ , basierend auf den Generatorpunkten $\{P, Q\} \in E$ sowie $\{\varphi(P), \varphi(Q)\} \in \varphi(E)$.

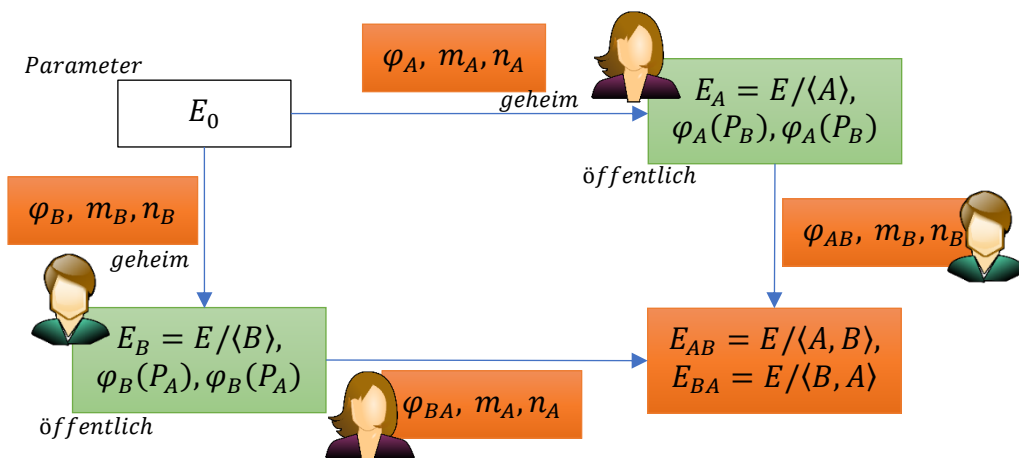


Bild 4.11 Alice und Bob berechnen ihren gemeinsamen und geheimen Schlüssel in Anlehnung an das Diffie-Hellman-Verfahren, mittels elliptischer Kurven in einem Graphen.

Die besten bekannten Angriffe auf dieses Verfahren haben eine Komplexität von $\mathcal{O}(p^{1/4})$ für klassische Systeme („Meet in the Middle Attack“) bzw. $\mathcal{O}(p^{1/6})$ für Quantencomputer („Claw-Finding-Algorithm“). Hierbei wird

versucht mit Hilfe des Claw-Finding-Algorithmus eine Kollision in Form einer Kurve E_x zu finden, die sowohl von E_0 als auch von E_A aus zu erreichen ist.

Verglichen mit anderen Post Quantenkryptographie-Verfahren verfügt SIDH über die geringsten Schlüsselgrößen, die durch Kompression auf Kosten der Performance weiter reduziert werden könnten. Ferner unterstützt SIDH Perfect Forward Secrecy PFS.

Tabelle 4.10 Vergleich von SIDH und gitterbasierten DH-Verfahren auf einer gemeinsamen Hardwareplattform nach [70].

Verfahren	Basis	Kompletter selaustausch (ms)	Schlüs- Größe	Public Key Größe (Byte)
Frodo	LWE	2,600		11.300
NewHope	R-LWE	0,310		1.792
NTRU	NTRU	2,429		1.024
SIDH	Supersingulare Isogenie	900		564
Komprimierter SIDH	Supersingulare Isogenie	2,390		330

Bei einem SIDH+ECDH Hybriden wächst die verwendete Schlüsselgröße im Vergleich zum einfachen SIDH nicht mehr als ca. 15% (Grober Richtwert für den privaten und den öffentlichen Schlüssel). Neben den ressourcenschonenden Vorteilen bieten isogeniebasierte Kryptosysteme durch ihre Flexibilität Vertraulichkeit und Authentizität, wobei Signaturverfahren aktuell wenig erforscht sind und auch teilweise geringere Geschwindigkeiten aufgrund des Rechenaufwandes besitzen. Neben dem hier vorgestellten SIDH Verfahren gibt es das Supersingular Isogeny Key Encapsulation-Verfahren, SIKE, welches sich aktuell in Runde 2 der NIST PQC Challenge befindet.

5 Standards im Bereich der Kryptographie

5.1 Entwicklung von Standards

In den bisherigen Kapiteln wurden die klassischen kryptographischen Primitive und ihre Sicherheit betrachtet. Es wurden Bedrohungen erörtert sowie künftige Möglichkeiten die Sicherheit zu gewährleisten und Verfahren, die als Ersatz für diese Primitive in Frage kommen. Diese Aspekte sind im Bereich der Forschung verortet. Es ist jedoch nicht ausreichend lediglich neue Primitive zu entwickeln und so die künftige Sicherheit in der IT als gegeben zu betrachten.

Ziel ist die Anwendung dieser Verfahren im Bereich des Endanwenders (Unternehmen, Behörden, Industrie, private Anwender, u.v.m.). Die Zielgruppen sind sehr heterogen aufgebaut und im Allgemeinen nicht in der Lage oder interessiert derartige Diskussionen und technische Details zu verstehen oder gar zu lösen. Es handelt sich hierbei lediglich um die Konsumenten von kryptographischen Verfahren. Unternehmen, Unternehmerinnen und Unternehmer, wie beispielsweise Rechtsanwälte, Arztpraxen und vergleichbare, sind zwar IT-gestützt, jedoch entwickeln sie eingesetzte Lösungen nicht selbstständig und haben für gewöhnlich keine Möglichkeit der direkten Einflussnahme oder das technische Verständnis, um die Sicherheit ggf. zu hinterfragen und Mängel zu entdecken. Sie beziehen ihre Systeme von Herstellern und Vertreiber von Software-Lösungen und Komplettsysteme. Idealerweise findet diese Zielgruppe die Möglichkeit der Abbildung ihrer Bedürfnisse (Use-Cases oder Anwendungsfälle) auf bestimmten Produkten, Produktkonstellationen und Parametrisierung vor, ohne dass es notwendig ist über Details Bescheid zu wissen. Auch derartige Systeme werden für gewöhnlich nicht aus einer Hand hergestellt und es werden OEM-Lösungen, Original Equipment Manufactur, zum Einsatz kommen, d.h. erworbene Systeme, die eigenen Produkten hinzugefügt werden.

Letztendlich werden diese Systeme, worunter auch die Entwicklung kryptographischer Lösungen im Bereich der Informationsverarbeitung fallen, von speziellen Unternehmen und Organisationen erstellt und vertrieben bzw. zur Verfügung gestellt. In diesem Bereich können allgemein Open-Source-Lösungen und kommerziell vertriebene Systeme unterschieden werden. Diese Unterscheidung hat Auswirkungen auf die Auslieferung neuer Verfahren. Während man davon ausgehen kann in einem Wartungsvertrag regelmäßig aktuelle Systeme zu haben, ohne aktiv eingreifen zu müssen, werden Anwender von Open-Source-Lösungen diesen Aufwand selber betreiben. Auch können OEM-Versionen branchenspezifisch von Herstellern vertrieben werden während Open-Source-Lösungen zwar durch eine breite Community entwickelt werden, jedoch liegen die Lösungen nicht unbedingt in optimierter Form vor und es gibt Bedarf an Korrektur- und Wartungsarbeiten.

Neben den drei genannten Parteien Endanwender, Vertrieb und Entwicklung spielen auch der Gesetzgeber sowie andere Entscheidungsträger eine Rolle. Diese sind in der Lage Druck auszuüben bzw. einen Rahmen zu definieren, in dem sich Lösungen bewegen müssen. Ein positives Beispiel stellt im Bereich des Datenschutzes die neue EU-DSGVO da, welche verbindlich für datenverarbeitende Unternehmen gilt. Ein weiteres Beispiel ist die Implementierung von Informationssicherheits-Managementsystemen ISMS für Betreiber kritischer Infrastruktur KRITIS. Auf diese Weise können einheitliche Mindeststandards an die Sicherheit in der Informationstechnologie gewährleistet werden. Derartige Standards können lokalen Variationen unterliegen. Beispielsweise ist der Einsatz von ECC in den Vereinigten Staaten von Amerika durch das NIST standardisiert, wobei die Standards unter Einfluss der NSA vorgegeben wurden. Das Bundesamt für Sicherheit in der Informationstechnik BSI empfiehlt seinerseits „brainpool“-Kurven zu verwenden. Beide Vorgaben stehen in der Kritik, dass Effizienz und Performance nicht berücksichtigt wurden. Es ist erstrebenswert, dass auf derartige Gegebenheiten reagiert werden kann.

Kryptographische Verfahren haben zum Teil lange Reifezeiten während technische Fortschritte und Innovationen in immer kürzeren Zeiträumen publiziert

werden. Diese Fortschritte berücksichtigen noch nicht unvorhersehbare Durchbrüche in der Kryptoanalyse oder mögliche Durchbrüche bei dem Bau von Quantencomputer. Nach der Veröffentlichung von RSA wurden Empfehlungen bezogen auf die Parameter seit den 70er Jahren von 512-Bit auf mittlerweile 3072-Bit Schlüssel korrigiert während beispielsweise DSA und MD5 als nicht mehr sicher betrachtet werden. Dennoch sind derartige Konfigurationen und Schemen aktuell noch immer im Einsatz. Es ist notwendig Produkte, welche kryptographische Verfahren einsetzen, stets aktuell zu halten solange sie Anwendung finden.

5.2 Flexibilität bei kryptographischen Lösungen

Aus heutiger Sicht ist es nicht einfach zu sagen welche Familie kryptographischer Verfahren künftig präferiert eingesetzt wird. Auch ist unklar ob und wann ausreichend starke Quantencomputer, die über eine große Anzahl von Qubits verfügen, Peter Shors Algorithmus zuverlässig ausführen können. Unabhängig dieser Fragestellungen besteht ständig Anpassungsbedarf an die Kryptographie, ihre Primitiven und die verwendeten Parameter der einzelnen Verfahren. Bekanntwerden von Side-Channel-Attacks können eine Anpassung oder gar einen Austausch von Verfahren erzwingen. Es scheint nicht sinnvoll und zielführend die Entwicklung von Quantencomputern abzuwarten. Während Bestellsysteme der privaten Marktwirtschaft hier beispielsweise eine einfachere Ausgangssituation haben sind Organisationen, welche mit Langzeitgeheimnissen arbeiten ggf. schon heute gezwungen sich für eine mögliche Zukunft aufzustellen.

Um derartige Anpassungen in Systemen durchzuführen, ohne in operative Prozesse einzugreifen, diese aufrecht zu erhalten oder gar neue Software-Lösungen auszuliefern, bietet sich das Konzept der losen Kopplung an. Unter loser Kopplung versteht man eine maximale Unabhängigkeit zwischen Komponenten eines Systems, z.B. Software und Hardware, Applikationen und Datenbanken etc. Die lose Kopplung von Komponenten erlaubt es Aktualisierungen in einem Bereich auszuspielen, ohne einen anderen zu stören oder in der Funktion einzuschränken. Eine solche Flexibilität bzw. Agilität erfordert den

Einsatz von Frameworks oder APIs zwischen der Applikation und einer möglichen kryptographischen Schicht. Eine solche Schicht könnte einen empfohlenen Pool an kryptographischen Primitiven, Protokollen und Parameter einer Applikation zur Verfügung stellen. Auch könnten diese ohne großen Anpassungsaufwand aktualisiert werden. Dem gegenüber steht die feste Implementierung kryptographischer Verfahren in Applikationen („harte Verdrahtung“). Eine Anpassung erfordert ggf. eine Neukompilierung und Auslieferung von Software-Updates gepaart mit Offline-Zeiten für Anwendungen innerhalb von ggf. kritischen Geschäftsprozessen (vgl. Industrie-Anlagen und ICS in Kombination mit Industrie 4.0). Derartige Szenarien und ein solcher Ansatz sollte bereits in der Konzeption von Software Anwendung finden (Security by Design).

Die Wichtigkeit von Kryptographie in der IT-gestützten Unternehmens- und Behördenlandschaft sowie Industrie und bei ggf. Privatanwendern bedürfen einer Sensibilisierung und einer besseren öffentlichen Wahrnehmung, um schneller auf Änderungen reagieren zu können. Zertifizierungs- und Trainingsprogramme können Unternehmen dazu bewegen standardisierte Verfahren zu verwenden. Längerfristig sollten in einer immer stärker von der IT abhängigen Welt Bildungswege IT-Sicherheitsthemen berücksichtigen oder stärker gewichten. Ein derartiger Grundstein führt in allen Bereichen, insbesondere bei den zuvor genannten Parteien, zu einem besseren Verständnis der Situation, was wiederum die Qualität der Entscheidung bezüglich des Schutzes von Daten und IT-Systeme deutlich verbessern kann. Auch können Zertifizierungsprogramme für Software-Produkte im Hinblick auf ihre kryptographischen Eigenschaften einen erhöhten nationalen oder internationalen (z.B. innerhalb der europäischen Union) Standard bewirken. Derartige Labels geben dem Endanwender Vertrauen in die Sicherheit der Produkte und haben ebenfalls Einfluss auf Drittstaaten. Derartige Mindeststandards könnten die Flexibilität an die Kryptographie voraussetzen wohin gegen ein solches Zertifikat oder Label für ein Software-Produkt als verpflichtend deklariert wird. Ein Beispiel für Zertifikate ist der vom BSI verabschiedete IT-Grundschutz, wenn auch im

deutlich größeren Ansatz als der hier gemeinte. Dieser Standard ist insbesondere für Vertreter von KRITIS verbindlich und soll so das Sicherheitsniveau der kritischen Infrastruktur auf ein einheitliches Mindestniveau heben. Eine mögliche Richtlinie könnte, analog zum Verzeichnisse in der EU-DSGVO, das Führen eines Verzeichnisses angewandeter Kryptographie sein. Die in der Masterthesis vorgestellten Verfahren sind idealisiert. Darüber hinaus ist jedoch eine Überführung in praktisch anwendbare Software-Produkte und IT-Systeme erforderlich.

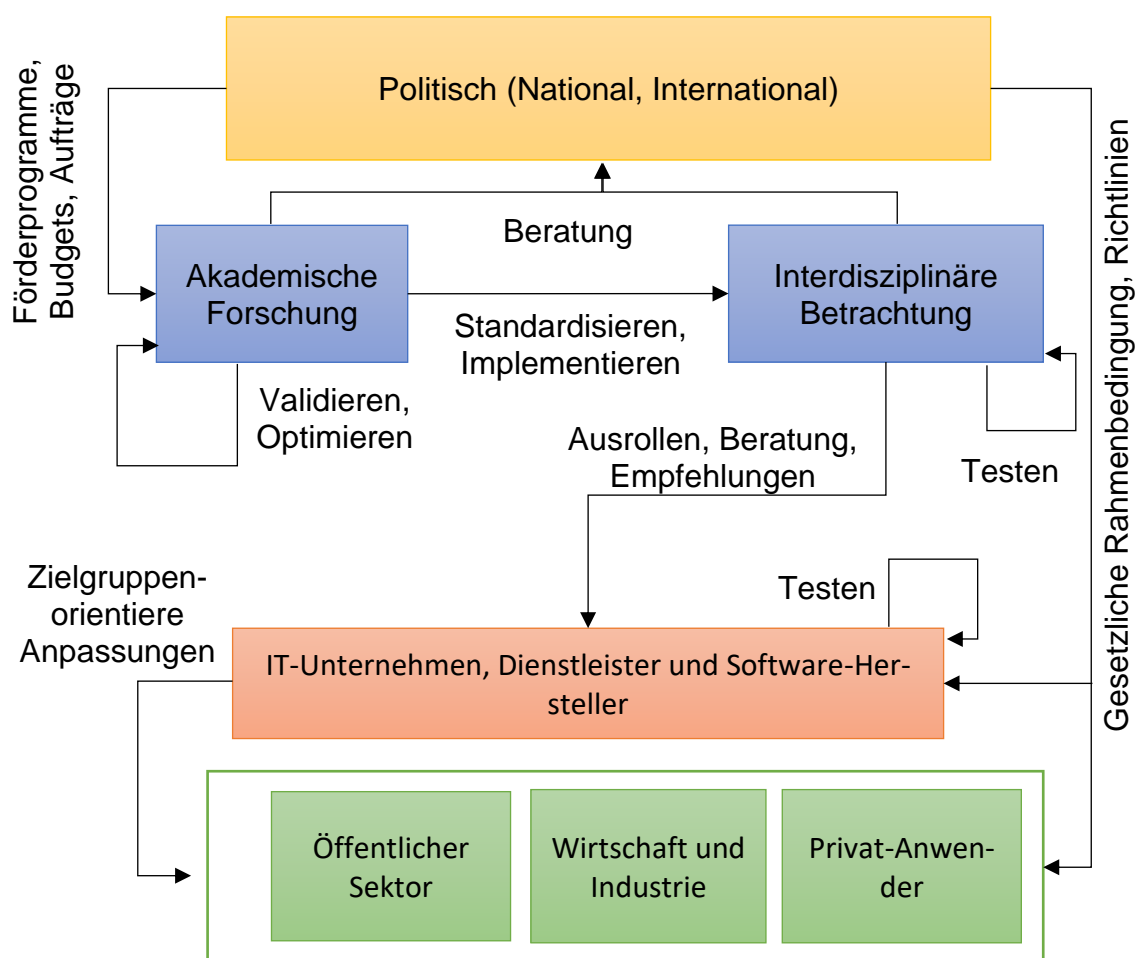


Bild 5.1 Grobkonzept des Verhältnisses zwischen einzelnen Parteien bei der Einführung von kryptographischen Verfahren ohne Darstellung des Rückflusses.

Interdisziplinäre Forschungsinstitute, wie z.B. das Nationale Forschungszentrum für angewandte Cybersicherheit (CRISP, Darmstadt) können dabei unterstützen akademische Lösungen zur Marktreife zu bringen und darüber hinaus

als Anlaufstelle für neue Software-Produkte und Empfehlungen dienen. Dabei geht es nicht ausschließlich um Kryptographie und Informatik. Auch werden Rechts- und Wirtschaftswissenschaften, Ethik und Psychologie in die Forschungsarbeit mit einbezogen [71]. Ein solches oder übergeordnetes Institut kann ebenfalls Beratungen für politische Entscheidungsträger anbieten, was wiederum Einfluss auf gesetzliche Standards, Richtlinien und Bildungswege, Forschung und somit auch auf Budget-Planungen hat.

5.3 Herausforderungen für Standards

Ein zentraler Aspekt für die Akzeptanz kryptographischer Verfahren ist Vertrauen. Vertrauen entwickelt sich auf etablierten Verfahren und Vorgehen, die lange Zeit ohne Auffälligkeiten funktioniert haben. Angesichts der Tatsache, dass die Forschung an Quantencomputern Fortschritte macht und seit kurzem auch kommerziell erste Systeme betrieben werden, die grundsätzlich geeignet sind Shors Algorithmus auszuführen, ist es jedoch fraglich wie viel Zeit für einen derartigen Reifeprozess verbleibt. Mehrere Organisationen und Regierungen treiben die Arbeiten an der Quantentechnologie mit stetig wachsendem Einsatz und zum Teil hohen Budgets voran [40]. Aktuell befinden sich Kandidaten der Post Quantenkryptographie in einem von der NIST ausgetragenen Wettbewerb, vergleichbar mit der Suche von Januar 1997 bis Oktober 2000 nach einem Nachfolger für das DES Verschlüsselungsverfahren. Jedoch benötigen derartige Wettbewerbe mit anschließendem Standardisierungsverfahren, Implementierung und Ausrollen in bestehende Software-Lösungen unter Umständen mehr Zeit als verfügbar ist berücksichtigt man die Bedrohung des „Store now, Decrypt later“.

Sicherheitsannahmen bezüglich Angriffen durch Quantencomputer sind nach heutigem Stand jedoch rein theoretisch. Noch ist unklar wie stark bzw. effektiv Quantencomputer eingesetzt werden können, um kryptographische Verfahren zu bedrohen. Kryptographische Verfahren können zu schwach oder aber auch zu stark und damit ineffizient gestaltet sein. Diese Problematik hat Auswirkungen auf die Benennung wirkungsvoller Parameter für aktuelle Post Quantenverfahren. Es stellte sich heraus, dass diese Art von Verfahren ihre Nachteile

bezüglich des Anspruchs an Ressourcen haben. Viele Versuche die Verfahren effizienter zu gestalten führten zu einem Verlust an Sicherheit.

Diese bisher ungelösten Probleme im Ressourcenbedarf stellen somit eine Herausforderung in der Implementierung von PQC-Verfahren dar insbesondere beim Einsatz in eingebetteten Systemen oder Smartcards. Einige Verfahren könnten für den Einsatz in diesem Bereich ungeeignet sein solange es keine Möglichkeit der Optimierung gibt. Mangelnde Erfahrung im Bereich der Implementierung zusammen mit einer steigenden Komplexität führen vermutlich zwangsläufig zu neuen Angriffsvektoren für die Kryptoanalyse in Form von Side-Channel-Attacks. Etablierte Verfahren wie RSA genießen hier, aufgrund von Erfahrungen und Untersuchungen, einen sehr hohen Vertrauensstatus, den Post Quantenverfahren noch nicht erreicht haben können.

Dennoch müssen bestehende kryptographische Systeme ausgetauscht bzw. ergänzt und korrigiert werden. Ein möglicher Ansatz sind hybride Verfahren. Das Unternehmen Google hat bereits mit seinem Browser Google Chrome auf eigenen Services erfolgreich das Post Quantenverfahren NewHope über einen begrenzten Zeitraum getestet. Nichts desto trotz gibt es noch keinen konsequenten Sicherheits-Layer und zentrale Bibliotheken, bei denen ein Update eine ausreichend hohe Reichweite hat. Auch würde ein solches Vorgehen nicht sich im Einsatz befindliche Systeme wie Smartcards und vergleichbares erreichen. Diese Systeme müssen ausgetauscht werden. Auch bezüglich der Wahl der Parameter kann künftig noch einiges passieren. Hier muss der Markt flexibel mit neuen Empfehlungen und Updates reagieren können, ohne ggf. ein Migrationsprojekt erneut durchführen zu müssen.

Bezogen auf den Ressourcen-Bedarf stellen sich ebenfalls Anforderungen an die Kompatibilität von Verfahren zu bestehenden Systemen. Erhöhter Speicherbedarf und Rechenleistung müssen ebenso berücksichtigt werden wie z.B. verfahrensspezifische Eigenschaften. Das XMSS Signaturverfahren generiert Hash-Bäume, deren Status für das weitere Verfahren relevant sind. Darüber hinaus dürfen Signaturen, im Gegensatz zum z.B. SPHINCS, nicht wiederverwendet werden. Das bedeutet, dass derart große Strukturen nicht nur für einen

längeren Zeitraum vorgehalten werden müssen, es müssen ggf. auch Historien betrachtet werden, um keine Duplikate zu generieren. So stellen sich Anforderungen an die Sicherheit während der Datenhaltung.

Kryptographische Verfahren sind heute nicht gesetzlich im Detail beschrieben. Jedoch sind sie z.B. im IT-Grundschutz vorgesehen. Dennoch haben fehlende oder schlechte kryptographische Verfahren ggf. Auswirkungen auf andere Gesetze und Richtlinien. Hier sei wieder die EU-DSGVO genannt. Diese stellen Ansprüche an die Sicherheit von Daten und somit auch an die verwendeten kryptographischen Verfahren.

Ein Vorteil ist jedoch die Tatsache, dass Quantencomputer aktuell noch eine hoch spezielle Art von Computern darstellen. Sie lösen spezielle Probleme mit speziellen Algorithmen. Nicht unbedingt jedes Problem wird sich durch einen Quantencomputer schnell lösen lassen. Durch das Fehlen ausreichend starker Quantencomputer sind Versuche heute noch sehr begrenzt. Es ist nicht anzunehmen, dass sofort mit der Bereitstellung starker Quantencomputer weitere Angriffsmöglichkeiten auf heutige Post Quantenverfahren zur Verfügung stehen werden. Dennoch sollte bereits heute der Grundstein gelegt werden, um auf neuartige Bedrohung in einem größer werdenden Feld schnell und effektiv zu reagieren.

5.4 Standardisierungs-Wettbewerb der NIST

Um optimale Verfahren aus der Forschung zu identifizieren und diese in eine marktreife Implementierung zu überführen werden Standardisierungsgremien benötigt. Das National Institute of Standards and Technology NIST hat die Verfahren DES, AES und die SHA-Familie standardisiert und ist im US-Handelsministerium verortet. Gerade in Bereich der Kryptographie und der Kryptoanalyse hat sich der Wettbewerbscharakter bewiesen. Ein Beispiel für ein Kryptoanalyse-Wettbewerb ist die RSA-Challenge, bei dem versucht wird immer größere Zahlen zu faktorisieren. Dieser Wettbewerb gibt einen Richtwert für die Sicherheitsparameter von RSA vor. Aber auch die Suche nach neuen

kryptographischen Verfahren wurde in der Vergangenheit erfolgreich in Wettbewerbsform durchgeführt. Neben der Sicherheit selber werden Kriterien wie die Patentfreiheit, Speicherbedarf und Performanz angesetzt, an denen sich ein Verfahren messen lassen muss. Mit Hilfe von Wettbewerben wurden die Erfolgreichen Beispiele DES und AES entwickelt und publiziert.

Ein Beispiel für ein durch ein Gremium entwickeltes Sicherheitsverfahren ist IPsec. Dieses sehr komplexe Verfahren verfolgt keine einheitliche Designlinie und hat vermeidbare Schwächen. Im Gremium waren Industrievertreter, die neben der Entwicklung des Verfahrens auch die eigenen Interessen wahren wollten. Angesichts dieser Nachteile stellt ein Wettbewerb eine sehr gute und effektive Alternative zum Finden von Verfahren dar [4].

Auf Basis dieser Erfahrung ist es naheliegend, dass das NIST auch für Post Quantenverfahren diese Wettbewerbsform erneut gewählt hat. Wissenschaftler und ihre Teams lassen Algorithmen gegeneinander antreten und versuchen gleichzeitig die Verfahren der Konkurrenten aus dem Wettbewerb zu drängen, in dem sie Schwachstellen finden, welche Angriffe erlauben oder Schwächen bezüglich der anderen Kategorien aufdecken. Im August 2016 rief das NIST zum Einreichen möglicher Kriterien für PQC Verfahren auf. Anschließend erfolgte ein Zeitfenster für das Melden möglicher Kandidaten bis November 2017. Ziel ist es diesmal nicht ein Verfahren zu finden, vielmehr geht es um die Bereitstellung einer ganzen Palette an kryptographischen Verfahren für verschiedene Zwecke (Schlüsselaustausch, Verschlüsselung, Signatur), wobei Überlappungen nicht ausgeschlossen sind.

Es wurden 69 Verfahren zur ersten Runde gemeldet, welche alle auf der Webseite des NIST veröffentlicht wurden⁴⁶. Einige der Verfahren sind im Zusammenhang dieser Masterthesis vorgestellt worden. Nach Abschluss der Einreichungen und Prüfung auf formale Korrektheit erfolgte die erste Untersuchung. Alle Kommentierungen werden dabei transparent auf der entsprechenden

⁴⁶ <https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions>

Seite zu den Verfahren geführt⁴⁷. Insgesamt sieht das NIST eine drei bis fünfjährige Analysephase in verschiedenen Runden vor. Aktuell befinden sich noch 26 Einreichungen im Wettbewerb [72]. Insbesondere die gitterbasierten Verfahren (z.B. NTRU, NewHope) aber auch die codebasierten Verfahren (z.B. classic McEliece) sind am stärksten vertreten. Auch das erwähnte supersingular isogene Verfahren SIKE hat es in Runde zwei des Wettbewerbes geschafft. Runde drei für das Auswahlverfahren ist für 2020/2021 vorgesehen. Man rechnet damit, dass erste Standards im Entwurf zwischen 2022 bis 2024 zur Verfügung stehen werden.

Tabelle 5.1 Zeitstrahl seit der Veröffentlichung des Shor-Algorithmus bis zum prognostizierter Wettbewerbsabschluss des NIST.

Jahr	Ereignis
1994	Peter Shor demonstriert mit seinem Algorithmus, dass Quantencomputer theoretisch in der Lage sind das Faktorisierungsproblem und das Problem des diskreten Logarithmus zu lösen.
2001	Ein Team um Isaac Chuang implementiert den Algorithmus auf einem 7 Qubit Quantencomputer.
2016	Das NIST startet einen weiteren Algorithmus Wettbewerb im Bereich Post Quantum Cryptography.
2018	Mit dem Jahreswechsel auf 2018 beginnt Runde Eins des Wettbewerbes mit 69 eingereichten Verfahren.
2019	Es werden die Verfahren für Runde zwei bekannt gegeben.
2020/21	Erwarteter Zeitrahmen für Runde drei des Wettbewerbes.
2022/24	Der Zeitraum, in dem erste Draft Standards zur Verfügung stehen sollen, d.h. Standards, die sich noch im Entwurf befinden.

Sollte das NIST mit seinem Zeitplan richtigliegen, so werden ca. 30 Jahre vergangen sein seit der theoretischen Bedrohung aktuell bestehender asymmet-

⁴⁷ <https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/official-comments/NTRUEncrypt-official-comment.pdf>

rischer Verfahren. Der Zeitplan des NIST bedeutet jedoch nicht, dass mit Abschluss des Wettbewerbes diese Verfahren direkt marktreif vorliegen werden, wie auch den vorhergehenden Abschnitten zu entnehmen ist.

5.5 PQCrypto

Die PQCrypto ist eine Konferenz zum Thema Post Quantenkryptographie und dient als Forum für Wissenschaft und Forschung für den Ideenaustausch und zum Präsentieren von Ergebnissen. Das Projekt ist EU-finanziert und hat zum Thema die Entwicklung neuer kryptographischer Systeme. Partner der PQCrypto sind u. A. die Bundesdruckerei, die Technische Universität Eindhoven, die Technische Universität Darmstadt und weitere⁴⁸. Das Projekt PQCrypto ICT-645622 arbeitet mit Work Packages WP von denen die Pakete WP1 – WP3 technischer Natur (Small Devices, Internet, Cloud), WP4 und WP5 nicht-technischer Natur sind (Management, Standardization) sind. Das Projekt veröffentlicht regelmäßig Arbeiten zum Thema PQC und gibt Empfehlungen zu bestehenden Verfahren, Referenz-Implementierungen, Untersuchungen der Verfahren auf verschiedenen Plattformen und Real-Szenarien. Auch wurden diverse Verfahren zum zuvor erwähnten NIST-Wettbewerb angemeldet, z.B. NewHope, SPHINCS+, MQDSS, FrodoKEM, Classic McEliece, NTRU Prime, NTRU-HRSS-KEM und mehr. Darüber hinaus wurden durch Mitglieder des Projektes erfolgreich Angriffe gegen 11 eingereichte Algorithmen verbucht was zur Folge hatte, dass einige Verfahren aus dem Wettbewerb zurückgezogen wurden.

Das Projekt entwickelt verschiedene Bibliotheken (libpqcrypto, pqm4, pqhw). Die Bibliothek libpqcrypto führt 77 kryptographische Systeme (50 Signatur-Systeme und 27 Verschlüsselungssysteme). 22 Verfahren davon wurden beim NIST eingereicht. Darüber hinaus beinhaltet die Bibliothek ein einheitliches

⁴⁸ <https://pqcrypto.eu.org/partners.html>

Framework, ein automatisches Test-Framework, C- und Python Interfaces sowie Command-Line Signatur- und Verifikations-, Ver- und Entschlüsselungs-Tools. Auch können Benchmarks durchgeführt werden.

5.6 IPsec und Internet Key Exchange IKE

5.6.1 Aufgaben des Internet Key Exchange IKE

Die Kommunikation zwischen Systemen erfolgt in der Regel via Internet Protokoll IP, verortet auf Schicht 3 des OSI-Referenzmodell. IP ist ohne kryptographische Funktionen entwickelt worden. Durch die Erweiterung IPsec konnte dieser Nachteil bezüglich Integrität, Authentizität (mit dem Authentication Header AH) und der Vertraulichkeit (Encapsulated Security Payload ESP) kompensiert werden. IPsec setzt dabei voraus, dass Alice und Bob über ein gemeinsames Geheimnis, einen Schlüssel, verfügen. Ein Schlüsselaustauschverfahren wurde in IPsec nicht vorgesehen. Hierfür wird das von der IETF vorgesehene Internet Key Exchange-Verfahren IKE verwendet, definiert in [RFC4306], welches IPsec ergänzt.

Das IKEv2 kombiniert die Protokolle IKE, ISAKMP [RFC2408] und Weitere. Es erfüllt dabei folgende Aufgaben:

- Aushandeln der verwendeten Algorithmen sowie Parameter für eine kryptographisch gesicherte Verbindung.
- Aufbau der Verbindung auf Basis zuvor ausgehandelter Bedingungen.
- Authentisierung der Kommunikationspartner.
- Gesichertes Aushandeln der Algorithmen, Parameter und der IPsec-Protokolle AH und ESP.
- Erstellung der für IPsec benötigten Schlüssel, so dass eine gesicherte Verbindung erfolgen kann.

In der ersten Phase des Ablaufes wird eine ISAKMP-SA (Security Association) aufgebaut (Auch IKE-SA). Das beinhaltet die IKE_SA_INIT (Schritt 1, 2) sowie die IKE_AUTH (Schritt 3, 4). Wesentlich für das Ableiten von Schlüsseln im

IKE ist die Diffie-Hellman-Gruppe (eine Primzahl P , Generator g , mit einer zyklischen Gruppe \mathbb{Z}_p^x oder die Parameter für eine elliptische Kurve E mit Basispunkt P) inkl. der Berechnung der Schlüssel. Dafür müssen sich die Kommunikationspartner auf eine Pseudo-Random-Function PRF einigen. Dabei lassen sich eine Reihe an Parametern für die Verwendung bestimmter Verfahren konfigurieren [73], [4].

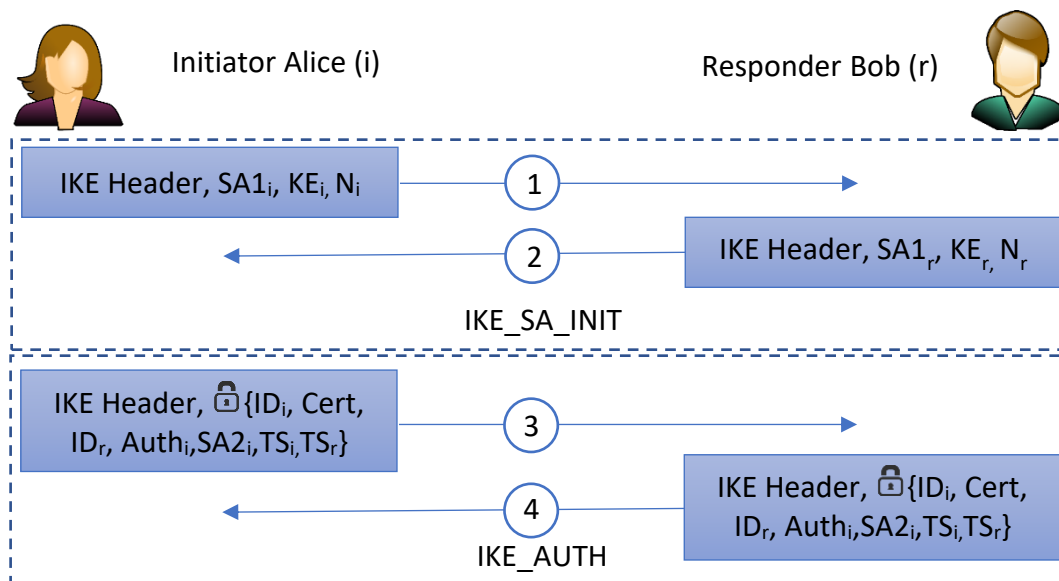


Bild 5.2 Nach Vorschlag und Auswahl von Algorithmen und Parametern für die IKE SA (1, 2) erfolgt die weitere Kommunikation bereits verschlüsselt (3, 4) und endet mit der gegenseitigen Authentifizierung der Identitäten, [74].

Sowohl die Schlüsselaustausch-Verfahren als auch die Parametrisierungen für die folgende symmetrische Verschlüsselung müssen bezüglich der Bedrohung durch Quantencomputer modifiziert werden.

5.6.2 IKE quantencomputer-resistent gestalten

Beim Einsatz von IPsec kommen im ESP und AH ein hashbasiertes sowie ein symmetrisches kryptographisches Verfahren zum Einsatz (vgl. RFC 8221 z.B. HMAC, SHA2, AES, CHACHA20, DES und weitere)⁴⁹ und benötigen eine Verdoppelung der Schlüssellängen und eine Vergrößerung von Hashwerten,

⁴⁹ <https://tools.ietf.org/html/rfc8221>

z.B. AES-256, SHA-384. Somit sind lediglich die Rahmenbedingungen der Parametrisierung neu zu definieren⁵⁰. Dasselbe gilt für die Verwendung der Verfahren im IKE-Protokoll. IKE verfügt über eine Cipher Suite für Verschlüsselung, Integritätssicherung, Pseudo-Zufallsfunktionen sowie einer Diffie-Hellman Group für das Aushandeln von Schlüsseln.

Im ersten Schritt (vgl. Bild 5.2, IKE_SA_INIT) bietet Alice eine Liste unterstützter kryptographischer Algorithmen mit Parametern Bob an, der seinerseits entsprechende Verfahren besitzen muss um seine Auswahl Alice bekannt zu geben. Diese müssen mit veränderten post quantenresistenten Parametern gewählt werden. Der Key Exchange KE_i und KE_r muss ebenfalls modifiziert werden, welche die Anteile des Diffie-Hellman-Verfahren hält. Durch den Austausch des Verfahrens durch PQC könnten potenziell andere Parameter in weiteren Nachrichten benötigt werden. Auch IKE_AUTH wäre durch die Änderung der Parameter betroffen. Wird das durch PQC generierte gemeinsame Geheimnis für eine Authentifizierung verwendet, so sind keine weiteren Modifikationen nötig. Ist es notwendig einen Personenbezug herzustellen, so werden post quantenresistente digitale Zertifikate benötigt.

Neben der sehr jungen Familie der isogeniebasierten Verfahren bieten sich die multivariat quadratischen-, gitterbasierten- sowie die codebasierten Verfahren an. Multivariate Verfahren sind ressourcenschonend, haben jedoch vergleichsweise große öffentliche Schlüssel. Einige Verfahren wurden in der Vergangenheit bereits gebrochen. Allgemein gelten MQ-Verfahren als noch nicht so gut erforscht, verglichen mit anderen Verfahren. Darüber hinaus befindet sich keines der bei NIST eingereichten Kandidaten mehr im Wettbewerb der Kategorie Schlüsselaustausch und Verschlüsselung. Gitterbasierte Verfahren eignen sich ebenfalls für Signatur- und Verschlüsselung bzw. einen Schlüsselaustausch. Im Gegensatz zu den multivariaten Verfahren sind entsprechend geeignete Kandidaten weiterhin in der NIST Challenge vertreten (NTRU, NewHope). Derartige Verfahren existieren bereits länger und gelten

⁵⁰ vgl. IKEv2-Parameter: <https://www.iana.org/assignments/ikev2-parameters/ikev2-parameters.xhtml>

als besser untersucht, so dass sie eine gewisse Vertrauensstellung genießen. In speziellen Instanzen gilt das Gitterproblem einschließlich Beweis als *NP*-schwer. NTRU verfügt nicht über einen derartigen Beweis und unterliegt patentrechtlichen Beschränkungen. Es ist jedoch effizient und hat geringe Anforderungen an Ressourcen.

Codebasierte Verfahren sind ebenfalls vergleichsweise schnell. Die Fehlerkorrektur in einer kodierten Nachricht ohne Kenntnis des zugrundeliegenden Codes konnte als *NP*-vollständig bewiesen werden⁵¹. Das McEliece-Verfahren hat Nachteile aufgrund seiner verwendeten Schlüsselgrößen im Megabyte-Bereich. Dennoch ist es als Classic McEliece nach wie vor in der laufenden NIST-Challenge vertreten. Das darauf basierende Niederreiter-Verschlüsselungssystem hat Performance- und Ressourcen-Vorteile gegenüber dem McEliece-Verfahren. Auch gegenüber RSA gibt es gerade im Bereich der Performance Verbesserungen, was das Verfahren zu einem künftigen und aussichtsreichen Kandidaten macht.

Tabelle 5.2 Vergleich der codebasierten Verfahren McEliece und Niederreiter (Parameter: 2048, 1718, t=30) mit RSA-2048 nach [75].

	McEliece	Niederreiter	RSA-2048
Public-Key (Kb)	429,5	69,2	0,5
Übertragungsrate	83,9%	67,3%	100%
Verschlüsselung (Komplexität)	1.025	46,63	40.555
Entschlüsselung (Komplexität)	2.311	8.450	6.557.176,5

Bei der Implementierung des Verfahrens kann ein Initiator codebasierend ein Schlüsselpaar erzeugen und den öffentlichen Teil an einen Empfänger senden. Dieser erzeugt in der IKE_SA_INIT dann seinerseits ein beidseitig zu verwendendes zufälliges Geheimnis für den symmetrisch verschlüsselten Kommunikationsanteil, welches er mit dem zuvor empfangenen öffentlichen

⁵¹ <https://authors.library.caltech.edu/5607/1/BERieetit78.pdf>

Schlüssel verschlüsselt und so dem Initiator in der IKE_SA_INIT-Antwort zukommen lässt. Dennoch sind die Schlüsselgrößen beim Einsatz codebasierter Verfahren weiterhin ein Problem. Das aktuelle IKEv2 nach [RFC 7296] besagt, dass „*All IKEv2 implementations [...] SHOULD be able to send, receive, and process messages that are up to 3000 octets long[...]*“ [74], was eine deutlich eingeschränkte Sicherheit bezüglich der möglichen Schlüsselgrößen nach sich zieht. Betrachtet man im Abschnitt der Zusammenfassung codebasierter Verfahren die dort dargestellten Größen wäre keine Sicherheit durch IKE-kompatibler Parameter möglich. Darüber hinaus verwendet IKE UDP-Pakete mit einer maximalen Größe von 65.535 Bytes. Sowohl die Spezifikationen für IKE als auch UDP werden durch die Verwendung des Niederreiter-Verfahrens bei entsprechender Sicherheit überschritten und führen so zwangsläufig zu einer Fragmentierung von übertragenden Paketen. Das „Hash and URL“ im IKEv2 [RFC 7296] bietet hier einen Lösungsansatz. Eine Implementierung ist demnach möglich. Neben dem Vorteil der Post Quantenresistenz sind auch die kryptographischen Operationen im Niederreiter-Verfahren sehr effizient und schnell. Nachteilig ist der anzunehmende Initialisierungsaufwand bei den im Vergleich hohen Datenmengen für Schlüssel. Bei einmaligem Aushandeln ggf. noch tolerierbar wäre das Aufrechterhalten einer Perfect Forward Secrecy ein Problem. Somit ist der Einsatz erneut vom Anwendungsszenario und den Sicherheitsanforderungen. Eine Open Source-Implementierung ist strongSwan und kann so auf verschiedene Einsatzszenarien zugeschnitten werden [76]. Bereits heute bietet strongSwan innerhalb der Diffie-Hellman Groups post-quantenresistente Schlüsselaustauschverfahren an wie NTRUEncrypt, das Verfahren Bimodal Lattice Signature Scheme BLISS sowie NewHope mit 128 Bit⁵².

⁵² <https://wiki.strongswan.org/projects/strongswan/wiki/IKEv2CipherSuites#Diffie-Hellman-Groups>

5.7 Transport Layer Security TLS

5.7.1 Kryptographische Ansätze auf der Transport-Schicht

Die Protokolle TCP und UDP der Transport-Schicht wurden ohne kryptographische Funktionen entworfen. Dieser Umstand wurde später mit der Transport Layer Security TLS, bzw. dem Vorläufer Secure Socket Layer SSL, korrigiert. Es handelt sich um kryptographische Netzwerkprotokolle, die Verschlüsselung, Integrität und Authentifizierung ermöglichen. Eine populäre Implementierung stellt OpenSSL dar. Genaugenommen wird TLS zwischen den Schichten 5-7 (Anwendungsschichten des TCP/IP-Modells) und Schicht 4 verortet. Hier finden die kryptographischen Operationen statt ohne dass das drunter liegende TCP davon etwas mitbekommt. Dennoch wird TLS allgemein zu Schicht 4 gezählt. Jede Anwendung oberhalb von TLS kann dieses unabhängig verwenden. Ein populäres Beispiel stellt HTTP dar, was mit TLS zusammen unter dem Namen HTTPS bekannt ist. Jedoch gibt es noch weitere bekannte Anwendungen wie FTPS, IMAPS LDAPS und weitere Anwendungsszenarien. TLS schreibt kein kryptographisches Verfahren vor. Kommunikationspartner handeln die Verfahren sowie die Parameter während der Verbindungsaufnahme aus [4]. Auch in diesem Fall müssen post quantenresistente Verfahren und Parameter angeboten werden.

5.7.2 Experimenteller Einsatz von NewHope in Google Chrome

Im Juni 2016 startete das Unternehmen Google eine über ein halbes Jahr andauernde experimentelle Phase, in der der von ihnen vertriebene Browser Chrome zu ausgewählten Servern ein quantencomputer-resistentes Schlüsselaustauschverfahren verwendet hat. Google setzte dabei auf das CECMQ1-Protokoll, das mit Hilfe von TLS-Socket-Kommunikation elliptische Kurven als ECDH mit Curve25519 [RFC7748] und das Ring-LWE NewHope⁵³ kombinierte (Curve25519 bezeichnet hierbei die verwendeten Kurven während sich X25519 auf die Diffie-Hellman-Funktion bezieht). Dieser kombinierte Ansatz

⁵³ <https://eprint.iacr.org/2015/1092.pdf>

sicherte sowohl gegen künftige Quantencomputer als auch aktuelle Angriffe ab. Der Schlüsselaustausch wurde mit Hilfe von ECDSA signiert und war somit nicht sicher gegenüber Quantencomputer. Google verwendete die eigene OpenSSL-Variante „BoringSSL“ (auch als Fork bezeichnet) und ist ausschließlich auf die Bedürfnisse von Google abgestimmt. Es stellt eine Bibliothek von kryptographischer Software bzw. Implementierungen von Netzwerkprotokollen und Verschlüsselungsverfahren dar. Google hat so die Möglichkeit eigene Untersuchungen vorzunehmen und ist, im Gegensatz zu z.B. OpenSSL, nicht so sehr auf die Wahrung von Kompatibilitäten zu alten Software-Produkten verpflichtet. Es ist leichtgewichtiger und konnte so ideal für die eigene Software Google Chrome verwendet werden. So konnten HTTP Verbindungen post-quantenresistent mittels HTTPS aufgebaut werden. Im Dezember 2016 wurde verkündet, dass es während der Testphase zu keinen unerwarteten Problemen gekommen sei. Dabei erhöhte sich die mittlere Verbindungslatenz lediglich um eine Millisekunde wobei 1% der langsamsten Verbindungen um 150ms angestiegen sind, was auf den erhöhten Datendurchsatz zurückzuführen sei [77]. Insbesondere langsame Verbindungen wären somit von der Problematik betroffen.

5.7.3 Quantum-Safe Hybrid (QHS) Key Exchange

Es ist erstrebenswert eine dauerhafte Lösung mit TLS anzubieten. Die Verfahren, die TLS zugrunde liegen, wurden durch Regierungen und Organisationen wie z.B. NIST SP800-131a, NSA Suite B, ISO Standards und andere geprüft und freigegeben. Es ist schwierig diese ohne weiteres auszutauschen, zu ändern oder zu erweitern [78], was als Beleg zu betrachten ist auch politisch Beteiligte in den Standardisierungsprozess hinzuzuziehen. Dennoch müssen entsprechende Cipher Suites angepasst werden. Idealerweise gibt es ein größeres Angebot von bestehenden Verfahren, so dass diese bei einem Sicherheitsbruch oder bei einem Update beim Aushandeln einfach durch andere ersetzt werden können. Um sich nicht ausschließlich auf die neuen, z.T. noch nicht ausreichend untersuchten, Verfahren zu verlassen ist auch bei TLS eine hybride Lösung erstrebenswert. Genau dieses Vorgehen wird als Hybrid Key

Exchange verstanden. Ein Beispiel ist die Verwendung von NTRU und Curve25519 als „Curve25519+NTRU“. Eine solche Kombination wird bei der Initialisierung durch einen Client einem Server vorgeschlagen. Dieser muss beide Verfahren kennen und akzeptieren. Ansonsten würde der Schlüsselaustausch scheitern. Um den Spielraum zu steigern sollten möglichst viele Kombinationen von ECDHE-Gruppen und quantencomputer-resistenten Verfahren angeboten werden. Beide Schlüssel werden anschließend kombiniert (Key Derivation Function KDF), so dass beide Kommunikationsteilnehmer über ein gemeinsames Geheimnis verfügen. Das Quantum-Safe Hybrid Key Exchange Verfahren QHS der IETF sieht in einem Draft-Standard [79] eben jene zusätzlichen PQC-Verfahren NTRU, LWE, HFE und McEliece vor.

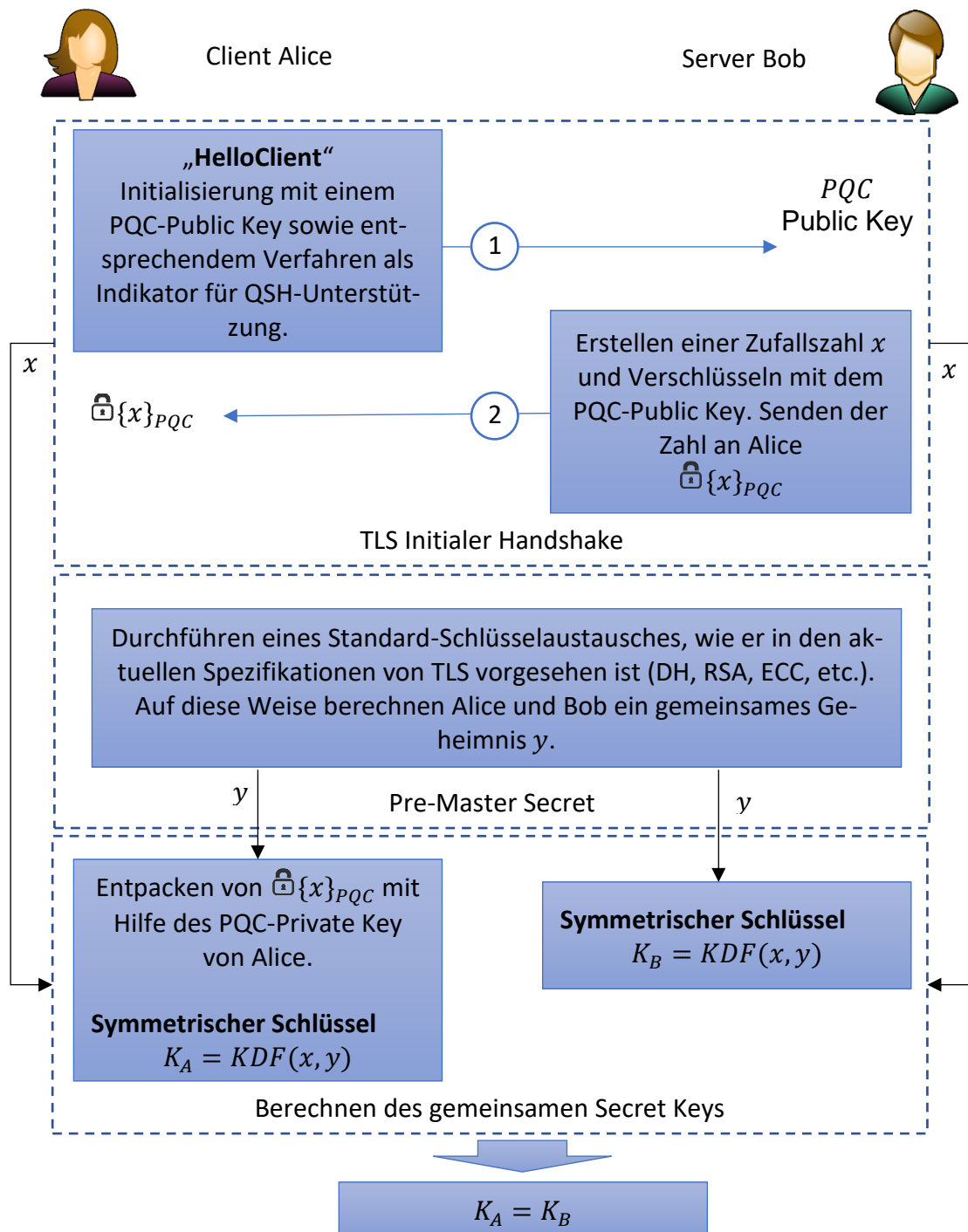


Bild 5.3 Vereinfachtes Ablaufschema von QSH aufbauend auf das TLS Protokoll.

Die Erweiterung von TLS durch einen solchen hybriden Ansatz bedarf nur geringer Änderungen bei den Datenübertragungen im ClientHello und ServerHello. Hier werden die Möglichkeit und Bereitschaft des hybriden Schlüsselaustausches kommuniziert, angenommen bzw. verworfen.

Für diesen Ansatz hat die PQCRYPTO, hinsichtlich des Aufrechterhalten von Langzeitsicherheit, für die asymmetrischen Verfahren vorgeschlagen auf seit 1978 studierte codebasierte kryptographische Verfahren zurück zu greifen mit der Argumentation, dass sich bis heute keine signifikanten Angriffe auf derartige Verfahren gezeigt haben. Für das Erreichen einer 2^{128} Bit Sicherheit gegenüber Quantencomputer gab man ebenfalls eine Parameter-Empfehlung ab, die unter [80] entnommen werden kann. Ein weiteres quantencomputer-resistentes Verfahren ist Kyber. Kyber ist nicht auf quantencomputer-resistente digitale Signaturen angewiesen. Sie kann das Schlüsselmaterial authentifizieren und wären ein Ersatz für ECDSA. Kyber verfügt über IND-CCA3 Sicherheitseigenschaften [81].

5.7.4 Open Quantum Safe-Project OQS

Wie bereits durch das Unternehmen Google in BoringSSL implementierte quantencomputer-resistente Vorgehen wird vom Open Quantum Safe Projekt ein Toolkit liboqs als Open-Source Library mit implementieren PQC-Algorithmen angeboten. So ist es möglich mit einem entsprechenden OpenSSL Fork (OQS-OpenSSH⁵⁴) die Mechanismen BIKE, SIKE, SIDH, NTRU, FrodoKEM, SABER, NewHope und Kyber zu verwenden. Mit Hilfe des entsprechenden Fork (Stebila's OpenSSL Fork with RLWE) können damit bereits heute gängige Web Server Cipher Suites wie z.B. RLWE-ECDSA-AES128-GCM-SHA256 anbieten. Bei Tests zeigte sich, dass die Performance im Schnitt von 3,9-fach (RSA) bis 7,9-fach zulegt [81].

Tabelle 3 Kryptographische Algorithmen innerhalb von liboqs nach Familien nach [82, p. 3].

Primitive	Protokoll	
Lattice-based	LWE	Frodo
	Ring-LWE	BCNS
		NewHope
		MSrIn

⁵⁴ <https://openquantumsafe.org/>

	Module-LWE	Kyber
	NTRU	
Supersingulare elliptische Kurven	SIDH	IQC Ref. MSR SIDH
Code-based	Fehlerkorrigierende Codes	McBits

Die Key Exchange Mechanism KEM im OpenSSL Fork sind namentlich RLWE-BCNS15, RLWE-NEWHOPE, RLWE-MSRLN16, LWE-FRODO-RECOMMENDED, SIDH-CLN16, SIDH-IQC-REF, CODE-MCBITS, NTRU, MLWE-KYBER und können für bestehende Verfahren den Schlüsselaustausch übernehmen. Beispiele sind X-ECDHE-RSA-AES256-GCM-SHA386 oder X-ECDHE-ECDSA-AES256-GCM-SHA386 wobei das X für die zuvor genannten, quantenresistenten Verfahren steht ([81], Stand Dez. 2017).

5.7.5 Digitale Signaturen und hybride KEM

Auch wenn für den Austausch von Schlüsseln einige Hürden zu nehmen sind, sind diese vergleichsweise geringer als ein Authentifizierungssystem vollständig auszutauschen, welche ebenfalls auf Algorithmen beruhen, die durch Quantencomputer bedroht werden. Daher müssen bestehende Public Key Infrastrukturen, PKI, ebenfalls sowohl rückblickend als auch für die Zukunft sicher aufgestellt werden und das mit möglichst geringen Eingriffen. Asymmetrische Verschlüsselung fehlt die Eigenschaft die Authentizität von z.B. Schlüsselmaterial sicherzustellen (Man-in-the-middle-Attacke). Ferner berücksichtigt asymmetrische Verschlüsselung keine Kompromittierung von Schlüsseln. Mit bekannt gewordenem privatem Schlüssel kann Kommunikation gelesen und Signaturen gefälscht werden. Ein Schlüssel selbst macht nicht kenntlich ob er gültig ist oder nicht. Eine Sperrung kann nicht bekannt gegeben werden. Prinzipiell kann so eine Nachricht abgestritten werden. Der Absender einer Nachricht kann behaupten, dass eine mit einem privaten Schlüssel signierte Nachricht nicht zu ihm gehört, da dieser nicht mehr gültig sei. Darüber hinaus besteht Bedarf an Richtlinien bezüglich Schlüssel und Schlüsselpaare z.B. Gültigkeitsdauer, Verbindung zu Unternehmen, Personen, Mindestlängen etc.

Um all diesen Anforderungen gerecht zu werden, werden Datenstrukturen benötigt, die u.a. den Schlüssel beinhalten. Derartige Strukturen werden als digitales Zertifikat bzw. auch nur Zertifikat bezeichnet. Diese Zertifikate werden ihrerseits von einer autorisierten Zertifizierungsstelle zertifiziert (Certification Authority CA). Ein hierbei verwendetes Standard-Format ist X.509, es wird bei TLS als digitaler Signaturen eingesetzt und dient der Authentifizierung der Identität zwischen zwei Systemen. Empfänger eines solchen Zertifikates können so die Identität bei einer vertrauenswürdigen CA prüfen und bezüglich des Absenders sichergehen. Ihre Größe und Inhalte variieren je nach Attributen, verwendeten Algorithmen (ECC, RSA) und Schlüssel bzw. Werten. Eben jene Algorithmen sind ebenfalls durch Quantencomputer bedroht und müssen angepasst werden. Um auch kompatibel zu bestehenden Systemen zu sein ist auch hier eine hybride Lösung nötig (PQ hybride X.509-Zertifikate mit verschiedenen Schlüsseln und Algorithmen). Für alternative Angaben in einem Zertifikat können entsprechende Erweiterungs-Felder verwendet werden,

Subjekt-Alt-Public-Key-Info,

korrespondiert mit Subject-Public-Key-Info-Type [RFC2986]. Dieses Attribut enthält Informationen zum alternativen öffentlichen Schlüssel, der zertifiziert wird. Die Informationen identifizieren auch den alternativen Public-Key-Algorithmus der Entität (und alle zugehörigen Parameter).

Alt-Signature-Algorithm,

korrespondiert mit signature-Algorithm [RFC2986]. Dieses Attribut enthält den Bezeichner für den alternativen kryptografischen Algorithmus, der von der anfordernden Entität zum Signieren verwendet wird.

Alt-Signature-Value,

korrespondiert mit Signatur-Feld, ebenfalls [RFC2986]. Dieses Attribut enthält

eine digitale Signatur, was den Besitz des zugehörigen, alternativen, privaten Schlüssel belegt.

Erweiterungen sind ab X.509v3 definierbar und können entsprechende Angaben zu Algorithmen und Signaturen tragen. Traditionelle und nicht modernisierte Systeme ignorieren diese Erweiterungen, solange sie nicht als kritisch klassifiziert sind. In diesem Fall operieren sie traditionell und sind somit abwärtskompatibel. Teilnehmer, welche die Erweiterungen erkennen, nutzen diese Angaben. Damit ergeben sich drei Szenarien:

1. Ein klassisches Serversystem ignoriert die Erweiterungen und setzt auf klassische Implementierungen. So erfolgt das Signatur- und Authentifizierungsverfahren mit einem Client nach heutigem Stand auf übliche Art.
2. Ein Server ist zwar in der Lage PQC-Verfahren anzuwenden, ein Client jedoch nicht. Der Client ist nicht in der Lage die Erweiterungen zuzuordnen und verfährt auf klassische Weise mit den Angaben innerhalb des digitalen Zertifikats.
3. Client und Server sind angepasst, so dass beide quantencomputer-resistente Algorithmen verwenden können. Der Client kann in dem Fall die DH-Public Key Daten, signiert mit dem privaten LMS-Schlüssel, verarbeiten und verzichtet auf den Einsatz klassischer Algorithmen.

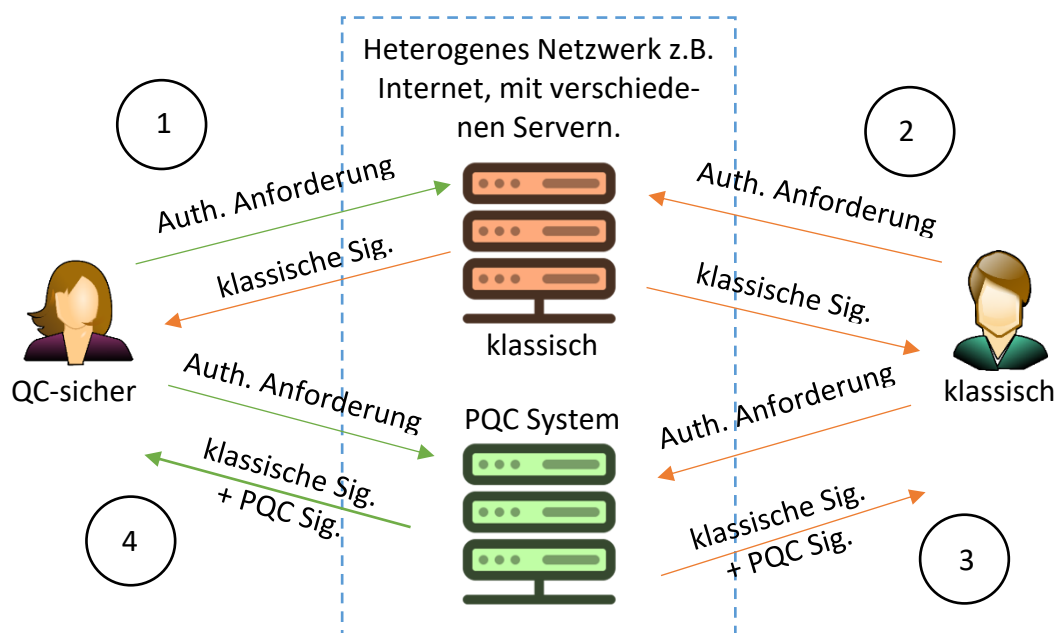


Bild 5.4 Während Bob im Fall 3 mit den alternativen Angaben des X.509 nicht anfangen kann und diese verwirft muss sich Alice im Fall 1 entscheiden ob sie die Authentifizierung akzeptiert oder z.B. gemäß einer Policy verwirft.

Bei der Implementierung eines solchen hybriden Verfahrens sind wachsende Signaturen bzw. große Schlüsselräume zu berücksichtigen. Ferner muss der Status von Hash-Bäumen, die bereits als vielversprechende Signaturverfahren vorgestellt wurden, mit einem eigenen Managementsystem versehen werden, um Fehler und Doppelverwendungen auszuschließen. Alternativ kann SPHINCS+ herangezogen werden. Statusbehaftete Verfahren müssen nicht nachteilig sein. Beispielsweise ist nach [83] ein Baum der Größe 50 in der Lage 21 Jahre verwendet zu werden, selbst wenn 2^{20} Nachrichten pro Sekunde signiert werden. Der X.509 Standard bedarf somit keiner Abweichung von seiner heutigen Gültigkeit. Lediglich die Sicherheitsprotokolle (z.B. OpenSSL für TLS 1.2 sowie StrongSwan für IKEv2) betreffender Systeme müssen angepasst werden, so dass diese in der Lage sind den hybriden Ansatz mit PQ-Signaturen und -Algorithmen zu verwenden. Vergleichbare Anpassungen müssten in weiteren Standards wie S/MIME (Bereich E-Mail-Kommunikation) vorgenommen werden.

6 Reflexion

6.1 Zusammenfassung

In dieser Masterthesis wurden aktuelle kryptographische Verfahren sowie einige Möglichkeiten der Kryptoanalyse vorgestellt und diskutiert. Heutige asymmetrische Verfahren nutzen mathematische Probleme aus, die einfach zu implementieren und vergleichsweise ressourcenschonend in der Berechnung von Schlüsseln, Ciphertexten und Signaturen, jedoch durch universelle Quantencomputer mit implementiertem Shor-Algorithmus bedroht werden.

Immer mehr Regierungen und Organisationen investieren große Budgets in die Entwicklung von Quantencomputern. Wirtschaftsunternehmen wie Google und IBM, aber auch Microsoft bauen, teilweise bereits im kommerziellen Bereich, Quantencomputer mit einer geringen Anzahl an Qubits. Noch sind, neben der Erhöhung der Anzahl physischer Qubits, weitere technische Hürden zu nehmen, insbesondere die Fehlerkorrekturen entsprechender Systeme. Nichts desto trotz werden weitere Fortschritte erwartet und man rechnet mit einer realen Bedrohung der heutigen Kryptographie in den nächsten 10-20 Jahren.

Zwar bietet die Quantenmechanik selbst eine Lösung des Problems mittels der Quantum Key Distribution QKD und Protokollen wie dem BB84 oder E91, jedoch gilt es auch in dieser Technik weitere Forschungsfragen zu klären.

Seit den 2000er Jahren rückt das Problem immer mehr in den Fokus von Organisationen und Instituten wie der Europäischen Union, NIST sowie Konsortien und Projekt-Gruppen wie der Vorgestellten PQCrypto. So werden aktuell kryptographische Primitive, basierend auf verschiedenen Problemstellungen, für eine Standardisierung untersucht. Diese neuen Verfahren sind zum Teil ähnlich lange verfügbar wie die bestehenden klassischen Verfahren RSA und ECC. Sie haben jedoch einen höheren Anspruch an Ressourcen und sind komplexer, wodurch sie längere Zeit im Hintergrund blieben.

Die Post Quantenkryptographie hat bereits Berücksichtigung in einigen Erweiterungen bestehender Standards wie OpenSSL-Forks gefunden. Es war möglich erste Feldtests unter Realbedingungen erfolgreich durchzuführen. Es wurde gezeigt, dass der Schlüsselaustausch auf verschiedenen Ebenen durchaus quantencomputer-resistent gestaltbar ist und das auch Signaturen in bestehende Systeme hybrid integriert werden können.

6.2 Bewertung

Im Verlaufe dieser Masterthesis hat sich gezeigt, dass die Bedrohung durch einen universellen Quantencomputer realistisch ist. Heutige Systeme liegen bei knapp unter 100 physischen Qubits und stellen keine Bedrohung dar. Die Zielgröße eines für die bestehende Kryptographie bedrohlichen Systems liegt bei 1.000.000 Qubits. Allerdings reicht die einfache Erhöhung der Qubits nicht aus, um einen Zeitpunkt zu extrapolieren wann die aktuelle Kryptographie genau bedroht ist. Dekohärenzzeiten, Fehlerkorrekturen, die Entwicklung eines Hardware-Standards und weitere Fragen sind auf verschiedenen Forschungsgebieten noch zu beantworten. Auch wenn nicht die gesamte wissenschaftliche Welt davon überzeugt ist, dass es ein solches System jemals geben wird, rechnen viele Organisationen mit maximal 20 Jahren bis, zumindest in Form eines Service on Demand, ein solches System zur Verfügung steht. Ein Quantencomputer mit implementiertem Algorithmus von Shor ist ein ernstzunehmendes Szenario. Quantencomputer von D-Wave habe heute schon eine deutlich höhere Anzahl an Qubits, stellen jedoch keinen universellen Quantencomputer dar und können nur sehr dedizierte Optimierungsprobleme lösen, nicht jedoch den Algorithmus von Shor.

Quantum Key Distribution QKD bedient sich quantenmechanischer Effekte und ist aus der Physik selbst heraus sicher, steht seinerseits jedoch ebenfalls erst am Anfang und bedarf dem Aufbau neuer Infrastrukturen (Netzwerke, Trusted Nodes, Satelliten, etc.). Somit stellt QKD keine Antwort auf heute benötigte Langzeitsicherheit dar.

Es gibt derzeit eine Reihe von kryptographischen Verfahren, die sowohl einen Ersatz für den Schlüsselaustausch, öffentliche Verschlüsselungsverfahren als auch Signaturen darstellen. Ihnen ist gemein, dass sie komplex sind und zum Teil noch nicht über ausreichend gründliche Sicherheitsbetrachtungen verfügen. Man nimmt jedoch an, dass sie eine Resistenz gegen Quantencomputer besitzen und auch klassischen Angriffen standhalten können. Diese Sicherheit erkauft man sich aktuell zumindest anteilig mit einem höheren Bedarf an Ressourcen, insbesondere im Speicherbedarf von Schlüsseln, Signatur- und Ciphertexten.

Viele Protokolle und Implementierungen sehen noch keine Integration der neuen kryptographischen Primitiven vor. Sie verwenden lediglich standardisierte Verfahren. Standardisierungen können unter Umständen jedoch längere Zeit in Anspruch nehmen. Nichts desto trotz konnten einige dieser neuen Verfahren erfolgreich als hybride Lösungen in aktuellen Sicherheitsprotokollen bestehender Systeme als Erweiterungen implementiert werden. Es konnte gezeigt werden, dass mit heutigen technischen Möglichkeiten auf die Bedrohung durch den Algorithmus von Shor angemessen reagiert werden kann, auch wenn die Standardisierungsprozesse noch nicht abgeschlossen sind. Die Mathematik, Informatik, Kryptographie und IT-Sicherheit bieten planbare Handlungsmaßnahmen um Software, Prozesse und Methoden für eine derartige Flexibilität bzw. Krypto-Agilität zukunftssicher aufzustellen.

Neben dem vorrangigen Vorantreiben des Standardisierungsprozesses für eine Übernahme in bestehende Protokolle wie OpenSSL und IPsec sind jedoch anwendungsfall-spezifische Überlegungen der Bedarfe bezüglich Vertraulichkeit erforderlich. Das „Store-Now, Decrypt-Later“ wird höchstwahrscheinlich schon heute betrieben⁵⁵. Das mag für Ad-Hoc Kommunikation, wie z.B. Einsatzlagen von Polizeien und Sicherheitskräften, Signaturen bei Online-Transaktionen von Online-Shops wie Amazon, und andere Anwendungsfälle aufgrund der kurzen Aktualität irrelevant sein. Die Kommunikation von Staats-

⁵⁵ <https://www.spiegel.tv/videos/161065-nsa-ausser-kontrolle>

geheimnissen, Patienten- und Versicherungsdaten, Urheberrechte bzw. Patente können jedoch durchaus darüber hinaus betroffen sein. Das Bundesministerium des Inneren schreibt in § 16 Abs. 1 und Ziffer 3 Anlage V zur VSA (Verschlusssachenanweisung) vor, dass die Einstufung einer VS des Geheimhaltungsgrades VS-Nur für den Dienstgebrauch (VS-NfD) auf 30 Jahre befristet ist, wobei es noch deutlich kritischere Einstufungen gibt (VS-V, Geheim, Streng Geheim). Je nach Kritikalität wäre hier eine sofortige Reaktion zu empfehlen. Dafür wurden Beispiele skizziert. Mit Hilfe der Krypto-Agilität ist es möglich Anwendungen und Geräte flexibel bezüglich des Einsatzes von Verschlüsselungsverfahren zu gestalten. Dazu zählt auch der Einsatz optimierter bzw. neuer quantencomputer-Resistenter Verfahren.

6.3 Fazit

Es ist schwierig zu prognostizieren in welche Richtung sich die zukünftige Kryptographie entwickeln wird. Fakt ist jedoch, dass heutige Schlüsselaustauschverfahren, auch als Bestandteil heute aufgezeichneten Kommunikation, in Zukunft durch einen Quantencomputer gebrochen werden, so dass Unbefugte auf die entsprechenden Daten zugreifen, sollte nicht bereits heute auf die Bedrohung reagiert werden. Heute technisch realisierbare Verfahren können einen Schutz dagegen bieten. Nichts desto trotz ist die Entwicklung von Strategien bezüglich eigener unternehmens- und organisationseigener Bedarfe, der Verwendung von PQC, möglicherweise QKD sowie dem Einsatz von Krypto-Agilität notwendig und darf nicht erst mit dem Erscheinen offizieller Standards erfolgen. Auch wenn zum heutigen Zeitpunkt noch kein Quantencomputer in ausreichend starker Form existiert wird er rückblickend eine ernstzunehmende Bedrohung darstellen, die es bereits heute zu berücksichtigen gilt, um nicht am Ende auf der Verliererseite im ständigen Duell von Kryptographie und Kryptoanalyse zu stehen.

Literaturverzeichnis

- [1] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," AT&T Research, 1996.
- [2] D. J. Bernstein, J. Buchmann and E. Dahmen, Post-Quantum Cryptography, Springer, 2009.
- [3] A. Cordel und S. Dr. Kousidis, „Quantencomputer,“ *BSI-Magazin, Referat Kryptographische Vorgaben und Entwicklungen*, Bd. 02, Nr. 2018/02, p. 24, 2018.
- [4] K. Schmeh, Kryptographie. Verfahren, Protokolle, Infrastruktur, dpunkt Verlag, 2007.
- [5] S. Spitz, M. Pramateftakis und J. Swoboda, Kryptographie und IT-Sicherheit. Grundlagen und Anwendungen, Wiesbaden: Vieweg+Teubner Verlag, Springer Fachmedien Wiesbaden GmbH, 2008.
- [6] H. U. Simon, „Das Rechenmodell namens "Turing-Maschine",“ Ruhr-Universität Bochum, Germany, 2011/12.
- [7] M. Homeister, Quantum Computing verstehen. Grundlagen, Anwendungen, Perspektiven., Wiesbaden: Springer Vieweg, 2018.
- [8] T. Haigh, „Von-Neumann-Architektur, Speicherprogrammierung und modernes Code-Paradigma. Drei Leitbilder früher Rechenanlagen,“ *zfm Zeitschrift für Medienwissenschaft*, Bd. Jg. 12 (2015) Nr. 1, pp. 127-139, 2015.
- [9] J. von Neumann, "First Draft of a Report on EDVAC," IEEE, University of Pennsylvania, 1945.
- [10] W. Reisig und J.-C. Freytag, Informatik. Aktuelle Themen im historischen Kontext., Berlin Heidelberg New York: Springer-Verlag, 2006.
- [11] C. Wagenknecht und M. Hielscher, Formale Sprachen, abstrakte Automaten und Compiler. Lehr- und Arbeitsbuch für Grundstudium und Fortbildung., Wiesbaden: Spring Vieweg, 2014.
- [12] M. Prof. Dr. Esponda, „Analyse von Algorithmen, die O-Notation - Freie Universität Berlin,“ 26 04 2012. [Online]. Available: http://www.inf.fu-berlin.de/lehre/SS12/ALP2/slides/V6_Rekursion_vs_Iteration_ALP2.pdf. [Zugriff am 22 07 2019].
- [13] C. Karpfinger und K. Hubert, Kryptologie. Algebraische Methoden und Algorithmen., Wiesbaden: Vieweg+Teubner, 2010.
- [14] J. Buchmann, Einführung in die Kryptographie, Berlin, Heidelberg: Springer-Verlag, 2016.
- [15] U. Baumann, E. Franz und A. Pfitzmann, Kryptographische Systeme, Berlin, Heidelberg: Springer-Verlag, 2014.
- [16] C. Paar und J. Pelzl, Kryptografie verständlich. Ein Lehrbuch für Studierende und Anwender, Berlin, Heidelberg: Springer-Verla, 2016.
- [17] A. Prof. Dr.-Ing. Ahrens, „Kryptografische Methoden und Anwendungen, Studienbrief, IT-Sicherheit und Forensik.,“ Wings - Wismar International Graduation Service GmbH, Wismar, 2019.
- [18] A. Beutelspracher, J. Schwenk und K.-D. Wolfenstetter, Moderne Verfahren der Kryptographie. Von RSA zu Zero-Knowledge., Wiesbaden: Springer Fachmedien, 2015.
- [19] A. May, „Public Key Kryptoanalyse.,“ Ruhr Universität Bochum. Alexander May., Bochum, 2008.
- [20] C. Pomerance, „A Tale of Two Sieves,“ University of Georgia, Athens, 1996.
- [21] R. Ghrist, „Orders of Growth.,“ 20 09 2016. [Online]. Available: <http://calculus.seas.upenn.edu/?n=Main.OrdersOfGrowth>. [Zugriff am 12 04 2019].
- [22] R. Küsters und T. Wilke, Moderne Kryptographie. Eine Einführung., Wiesbaden: Vieweg+Teubner, 2011.

- [23] CNSS Secretariat (I42) . National Security Agency., „FACT SHEET - CNSS Policy No. 15, Fact Sheet No. 1. National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Security Information,“ 06 2003. [Online]. Available: <https://csrc.nist.gov/csrc/media/projects/cryptographic-module-validation-program/documents/cnss15fs.pdf>. [Zugriff am 14 04 2019].
- [24] A. Cordel und S. Dr. Kousidis, „Quantencomputer. Eine BSI-Studie zum Entwicklungsstand.“, *BSI-Magazin 2018/02. Mit Sicherheit. Verschlüsselung als Grundlage für eine sichere Digitalisierung.*, pp. 22-25, 02 2018.
- [25] C. J. Meier, Eine kurze Geschichte des Quantencomputers. Wie bizarre Quantenphysik eine neue Technologie erschafft., Hannover: Heise Zeitschriften Verlag GmbH & Co KG, 2015.
- [26] C. Dr. Seidel, F. Dr. Neukart und G. Dr. Compostella, „Quantum Computing. Durch die Barrieren. Verkehrsflussoptimierung mit einem D-Wave Quantum Annealer.“, *iX. MAGAZINH für professionelle Informationstechnik*, pp. 94-99, 02 02 2018.
- [27] U. Seyfarth, „Quantenkryptographie. Akademisches Hirngespinnst oder bald schon verbreitete Realität?“, *<kes>. Die Zeitschrift für Informations-Sicherheit*, pp. 16-24, 08 2017.
- [28] M. Kurt, Vom Bit zum Qubit. Eine kleine Einführung in die Quantencomputer., München: Red Horse, 2018.
- [29] A. Tillemans, „wissenschaft.de,“ Konradin Medien GmbH, 15 02 2001. [Online]. Available: <https://www.wissenschaft.de/technik-digitales/einsteins-spukhafte-fernwirkung-ist-realitaet/>. [Zugriff am 09 05 2019].
- [30] J. Rödling, „Unkonventionelle Computer. Quantengatter und Quantenschaltkreise.“, Jan Rödling. Technische Universität Braunschweig, Braunschweig, 2001.
- [31] F. Wunderlich-Pfeiffer, „CNOT-Gatter verschränken Qubits,“ Golem Media GmbH, 25 07 2017. [Online]. Available: <https://www.golem.de/news/quantengatter-die-bauteile-des-quantencomputers-1707-128276-2.html>. [Zugriff am 09 05 2019].
- [32] G. Benenti, G. Casati und G. Strini, Principles of Quantum Computation and Information., Singapore: World Scientific Publishing Co. Pte. Ltd., 2005.
- [33] B. Trenwith, „Understanding Quantum Computers - 3.3 - Quantum Fourier Transform,“ Keio University, Tokio, 2018.
- [34] L. Gasser, „Quanten Fourier Transformation: Simulation eines Quantenalgorithmus in Matlab. Bachelorarbeit,“ Insitut für Theorerische Physik. Computational Physics. TU Graz, Graz, 2018.
- [35] T. Zoppke und C. Paul, „Shor's Algorithm. Faktorisierung großer Zahlen mit einem Quantencomputer.“, FU-Berlin. Zoppke, Till; Paul, Christian., Berlin, 2002.
- [36] H. Dr. Hagemeyer, S. Dr. Kousidis, M. Dr. Lochter und R. K. V. u. Entwicklung, „Public-Key-Kryptographie zukunftsicher machen,“ *BSI-Magazin "Mit Sicherheit"*, Bd. 02, Nr. 2018/02, pp. 22-23, 2018.
- [37] H. Böck, „Verschlüsselung. Was noch sicher ist.“, Golem Media GmbH, 09 09 2013. [Online]. Available: <https://www.golem.de/news/verschluesselung-was-noch-sicher-ist-1309-101457-9.html>. [Zugriff am 15 05 2019].
- [38] B. f. S. i. d. Informationstechnik, „Entwicklungsstand Quantencomputer. Deutsche Zusammenfassung,“ Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2018.
- [39] E. Begemann, „Morituri te salutant!(2).“, *<kes> Die Zeitschrift für Information-Sicherheit*, pp. 48-60, 02 2019.
- [40] G. Dr. Kalbe, „EU investment in Quantum Technologies. ENISA Summer School, 26. September2018.“, DG CNECT, European Commission, Heraklion, 2018.
- [41] T. Pöppelmann, „Bridging 1st PQC-functions and principles with the smart card world.“, Infineon., Heraklion, 2018.

-
- [42] U. Ostler, „Vorglühen im Quantencomputing. D-Wave knackt mit 2.048-Qubit Prozessor bisher unlösbares Magnetismusproblem.“ DataCenter Insider. Vogel IT-Medien GmbH., 25 07 2018. [Online]. Available: <https://www.datacenter-insider.de/d-wave-knackt-mit-2048-qubit-prozessor-bisher-unloesbares-magnetismusproblem-a-736032/>. [Zugriff am 16 05 2019].
- [43] W. Dr. Fumy, „Datenschutz und Datensicherheit: Quantencomputer und die Zukunft der Kryptographie.“ Springer Fachmedien Wiesbaden GmbH, Wiesbaden, 2017.
- [44] L. Bruckert und J.-M. Schmidt, „Post-Quanten-Kryptographie. Sicherheit in Zeiten des Quantencomputer.“ <kes>. *Die Zeitschrift für Informations-Sicherheit.*, pp. 54-59, 01 02 2018.
- [45] S. Singh, *Geheime Botschaften. Die Kunst der Verschlüsselung von der Antike bis in die Zeit des Internets.*, München: Deutscher Taschenbuch Verlag GmbH & Co. KG, 2001.
- [46] N. Gisin und B. Huttner, „Quantum Cloning, Eavesdropping and Bell's inequality.“ University of Geneva, Geneva, Switzerland., 1996.
- [47] P. Erker und M. Maiwöger, „Ekert-Protokoll.“ Universität Wien., Wien., 2011.
- [48] S. Aaronson, „Three Questions about Quantum Computing.“ University of Texas., Austin, USA., 2018.
- [49] T. Pöppelmann, „Bridging 1st PQC-functions and principles with the smart card world.“ infineon., Herakleon, Griechenland., 2018.
- [50] R. C. Merkle, „A Certified Digital Signature.“ BNR Inc., Palo Alto, California., 1979.
- [51] B. Lauer, „XMSS: Post-Quanten-Verfahren steht bereit.“ Ebner Media Group GmbH & Co. KG, 11 06 2018. [Online]. Available: <https://www.dotnetpro.de/diverses/xmss-post-quanten-verfahren-steht-bereit-1544960.html>. [Zugriff am 06 06 2019].
- [52] K. Schmeih, „Die Schnecke im Salatfeld. Post-Quantum-Kryptographie anschaulich erklärt.“ *iX Special 2019 - IT heute*, pp. 109-115, 2019.
- [53] S.-L. Gazdag, „Hash-Based Signatures.“ Genua. A Bundesdruckerei Company., Herakleon, Greek., 2018.
- [54] M. Sjöberg, „Post-quantum algorithms for digital signing in Public Key Infrastructures.“ TH Royal Institute of Technology., Stockhol, Schweden, 2017.
- [55] M. O'Neill, „Practical Implementation of Lattice-based cryptography.“ SAFEcrypto, Herakleon, Griechenland., 2018.
- [56] J. Buchmann, J. Krämer, N. A. Alkadri, N. Bindel, R. El Bansarkhari, F. Göpfert und T. Wunderer, „Bewertung gitterbasierter kryptografischer Verfahren.“ Bundesamt für Sicherheit in der Informationstechnik 2017, Bonn., 2017.
- [57] F. Vallentin, „Zur Komplexität des "Shortest Vector Problem" und seine Anwendungen in der Kryptographie.“ Universität Dortmund, Dortmund, 1999.
- [58] H. B. Nguyen, „An overview of the NTRU cryptographic system.“ San Diego State University, San Diego, 2014.
- [59] V. Lyubashevsky, „Lattice Cryptography in the NIST Standardization Process.“ IBM Research – Zurich, Herakleon., 2018.
- [60] H. Bröck, „Golem.de.“ Golem Media GmbH, 08 07 2016. [Online]. Available: <https://www.golem.de/news/new-hope-google-testet-post-quanten-algorithmus-1607-121989.html>. [Zugriff am 17 06 2019].
- [61] L. Bruckert und J.-M. Schmidt, „<kes>. Post-Quanten-Kryptografie. Sicherheit in Zeiten des Quantencomputers.“ <kes> online, 01 02 2018. [Online]. Available: <https://www.kes.info/archiv/leseproben/2018/post-quanten-kryptografie/>. [Zugriff am 18 06 2019].
- [62] T. Dr. Pöppelman, J. Dr. Haid und P. Schmitz, „Security Insider. Post-Quantum-Kryptographie. Die Zukunft der Kryptographie im Zeitalter der Quanten.“ Vogel IT-Medien GmbH, 26 09 2017. [Online]. Available: <https://www.security-insider.de/die->
-

- zukunft-der- kryptographie-im-zeitalter-der-quanten-a-645548/. [Zugriff am 18 06 2019].
- [63] D. Loebenberg, „Code-based Cryptography,“ genua. A Bundesdruckerei Company, Herakleon, 2018.
- [64] P. Luttmann, „Kryptologische Methoden und Alternativen.,“ Leibniz Universität Hannover., Hannover, 2015.
- [65] J. Ding und D. Schmidt, „Multivariable public-key cryptosystems,“ University of Cincinnati, Cincinnati, USA, 2004.
- [66] M. Daum, „Das Kryptosystem HFE und quadratische Gleichungssysteme über endliche Körper.,“ Universität Dortmund, Dortmund, 2001.
- [67] S. Heyse, „Post Quantum Cryptography: Implementing alternative public key schemes on embedded devices. Preparing for the rise of Quantum Computers.,“ Ruhr-University Bochum, Bochum, Germany., 2013.
- [68] J. Buchmann, „Post-Quantum Cryptography - an overview.,“ Technische Universität Darmstadt., Hitotsubashi Hall, Tokyo, 2015.
- [69] C. Costello, D. Jao, P. Longa, M. Naehrig, J. Renes und D. Urbanik, „Efficient compression of SIDH public keys,“ Microsoft/University of Waterloo/Radboud University, Redmond, USA; Waterloo, Canada; Nijmegen, Netherlands., 2017.
- [70] C. Costello, „An introduction to supersingular isogeny-based cryptography (Croatia),“ Microsoft Research, Sibenik, Croatia., 2017.
- [71] F.-I. f. S. Informationstechnologie, „CRISP - Fraunhofer SIT - Forschung für die Kernfragen der Cybersicherheit.,“ Fraunhofer SIT, 21 05 2019. [Online]. Available: <https://www.sit.fraunhofer.de/de/angebote/gremien-netzwerke/crisp/>. [Zugriff am 10 07 2019].
- [72] L. Dr. Chen, „Post-Quantum Cryptography. Round 2 Submissions.,“ NIST. National Institute of Standards and Technology., 08 07 2019. [Online]. Available: <https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-2-Submissions>. [Zugriff am 11 07 2019].
- [73] Bundesamt für Sicherheit in der Informationstechnik., „Bundesamt für Sicherheit in der Informationstechnik, Technische Richtlinie TR-02102-3 Kryptographische Verfahren: Empfehlungen und Schlüssellängen,“ 11 02 2019. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR02102/BSI-TR-02102-3.pdf?__blob=publicationFile&v=6. [Zugriff am 28 04 2019].
- [74] Internet Engineering Task Force (IETF) , „RFC7296: Internet Key Exchange Protocol Version 2 (IKEv2),“ [Online]. Available: <https://tools.ietf.org/html/rfc7296>. [Zugriff am 15 07 2019].
- [75] P. Véron, „Code based cryptography and steganography,“ Springer Verlag, Porquerolles, France, 2013.
- [76] A. Steffen und M. Willi, „VPNs mit Strongswan-IKEv2,“ Linux Magazin Online., 02 2009. [Online]. Available: <https://www.linux-magazin.de/ausgaben/2009/02/tunnel-design-2/>. [Zugriff am 19 07 2019].
- [77] V. Lynch, „Google’s Post-Quantum Cryptography Experiment Successful,“ hashedout. The SSL Store., 30 11 2016. [Online]. Available: <https://www.thessslstore.com/blog/googles-post-quantum-cryptography-experiment-successful/>. [Zugriff am 19 07 2019].
- [78] B. Huttner und J. Melia, „Applied quantum-safe security: Quantum-resistant algorithms and quantum key distribution. Cloud Security Alliance.,“ 2017. [Online]. Available: <https://downloads.cloudsecurityalliance.org/assets/research/quantum-safe-security/applied-quantum-safe-security.pdf>. [Zugriff am 22 07 2019].
- [79] W. Whyte, Z. Zhang, S. Fluhrer und O. Garcia-Morchon, „Quantum-Safe Hybrid (QSH) Key Exchange for Transport Layer Security (TLS) version 1.3.,“ 31 03 2017.

-
- [Online]. Available: <https://tools.ietf.org/html/draft-whyte-qsh-tls13-04>. [Zugriff am 20 07 2019].
- [80] PQCRYPTO, „Initial recommendations of long-term secure postquantum systems.“, 07 09 2015. [Online]. Available: <http://pqcrypto.eu.org/docs/initial-recommendations.pdf>. [Zugriff am 22 07 2019].
- [81] J. A. S. Velázquez, „Practical Implementations of Quantum-Resistant Cryptography“, 18 12 2017. [Online]. Available: https://courses.cs.ut.ee/MTAT.07.022/2017_fall/uploads/Main/antonio-report-f17.pdf. [Zugriff am 22 07 2019].
- [82] An, Hyeongcheol; Choi, Rakyong; Lee, Jeeun; Kim, Kwangjo, „Performance Evaluation of liboqs in Open Quantum Safe Project (Part I).“, 23 01 2018. [Online]. Available: <https://pdfs.semanticscholar.org/7ef2/2ef120fd8dcc64a7e1865b99b02e37ad3a48.pdf>. [Zugriff am 22 07 2019].
- [83] P. Kampanakis, P. Panburana, E. Daw und D. Van Geest, „The Viability of Post-Quantum X.509 Certificates.“, 2018. [Online]. Available: <https://eprint.iacr.org/2018/063.pdf>. [Zugriff am 22 07 2019].
- [84] K. Freiermuth, J. Hromkovic, L. Keller und B. Steffen, Einführung in die Kryptologie, Zürich, Schweiz: Springer Vieweg, 2014.
- [85] D. W. Hoffmann, Theoretische Informatik, München: Carl Hanser Fachbuchverlag, 2011.
- [86] I. Wegener, Komplexitätstheorie. Grenzen der Effizienz von Algorithmen, Berlin Heidelberg: Springer-Verlag, 2003.
- [87] A. Badach und E. Hoffmann, Technik der IP-Netze, München: Hanser, 2015.
- [88] M. Bartosch, „heise Security“, Heise Medien GmbH & Co. KG, 27 05 2008. [Online]. Available: <https://www.heise.de/security/artikel/Diffie-Hellman-Verfahren-270980.html>. [Zugriff am 30 04 30].
- [89] R.-H. Schulz und H. Witten, „Faktorisieren mit dem Quadratischen Sieb. Ein Beitrag zur Didaktik der Algebra und Kryptologie.“, Inst. f. Mathematik der FU Berlin, Berlin, 2011.
- [90] J. Döcker, „Pollards Rho-Methode zur Faktorisierung. Bachelorarbeit.“, Carl von Ossietzky Universität. Janosch Döcker., Oldenburg, 2011.
- [91] A. Prof. Dr.-Ing. Ahrens, „Kryptoanalyse, Studienbrief, IT-Sicherheit und Forensik.“, Wings - Wismar International Graduation Service GmbH, Wismar, 2018.
- [92] M. Goemans, „Factoring. 18.310 lecture notes.“, University of Michigan, Michigan, 2015.
- [93] M. Schneider, „Beschreibung des MPQS-Algorithmus und Implementierung eines Softwarepaketes als Beispiel für verteiltes Rechnens.“, OTH-Amberg, Weiden, 2017.
- [94] E. Landquist, „The Quadratic Sieve Factoring Algorithm-“, University of Virginia, Charlottesville, 2001.
- [95] Prashant, „The Church-Turing-Deutsch Principle and its Fundamental Ramifications“, Universite de Montreal, Quebec, Canada., Montreal, 2006.
- [96] D. Deutsch, „Quantum theory, the Church-Turing principle and the universal quantum computer.“, Centre for Quantum Computation, Oxford, 1984.
- [97] J. Draeger, „Klassische und Quanten Turing Maschinen, Teil I.“, 25 06 2017. [Online]. Available: https://www.researchgate.net/publication/317869909_Klassische_und_Quanten_Turing_Maschinen_Teil_I. [Zugriff am 21 03 2019].
- [98] Universität Ulm, Formale Grundlagen der Informatik, „Universität Ulm“, [Online]. Available: https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.040/Formale_Methoden_der_Informatik/%C3%9Cbungen/blatt08-zusatz_2.pdf. [Zugriff am 29 03 2019].
-

- [99] Divers, „Wikipedia - Elektrizität/Tabellen und Grafiken,“ Wikipedia - Die freie Enzyklopädie, 27 04 2019. [Online]. Available: https://de.wikipedia.org/wiki/Elektrizit%C3%A4t/Tabellen_und_Grafiken. [Zugriff am 30 04 2019].
- [100] H. W. Lang, „Hochschule Flensburg. University of Applied Sciences.,“ Hochschule Flensburg. University of Applied Sciences., 04 06 2018. [Online]. Available: <http://www.inf.fh-flensburg.de/lang/algorithmen/grundlagen/gruppe.htm>. [Zugriff am 20 04 2018].
- [101] B. f. S. i. d. Informationstechnik, „Bundesamt für Sicherheit in der Informationstechnik, Technische Richtlinie TR-02102-3 Kryptographische Verfahren: Empfehlungen und Schlüssellängen,“ 11 02 2019. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR02102/BSI-TR-02102-3.pdf?__blob=publicationFile&v=6. [Zugriff am 28 04 2019].
- [102] H. W. Lang, „Diffie-Hellman-Schlüsselvereinbarung,“ Hochschule Flensburg, 14 03 2019. [Online]. Available: <http://www.inf.fh-flensburg.de/lang/krypto/protokolle/diffie.htm>. [Zugriff am 03 04 2019].
- [103] R. Ghrist, „Orders of Growth.,“ 20 09 2016. [Online]. Available: <http://calculus.seas.upenn.edu/?n=Main.OrdersOfGrowth>. [Zugriff am 12 04 2019].
- [104] C. Diem und K. Schmeh, „Gegen die Apokalypse. Algorithmenwettbewerb zur Post-Quanten-Kryptographie.,“ *iX. Magazin für professionelle Informationstechnik.*, Nr. 06/2018, pp. 116-120, 2018.
- [105] E. Dipl.-Inf. Zimmer, „Universität Hamburg.,“ 20 11 2014. [Online]. Available: <https://svs.informatik.uni-hamburg.de/publications/2014/2014-11-20-Zimmer-CAST-PQC-fuer-IPsec.pdf>. [Zugriff am 19 07 2019].
- [106] R. Misoczki, J.-P. Tillich und N. B. P. S. L. M. Sendrier, „MDPC-McEliece: New McEliece Variants from Moderate Density Parity-Check Codes.,“ Project SECRET/Escola Polit´ecnica, Universidade de Sao Paulo, France/Brazil, 2012.

Bilderverzeichnis

Bild 2.1 Schematische Darstellung einer universellen Turingmaschine U mit Ein- und Ausgabeparametern. 15

Bild 2.2 Beispielhafte Darstellung von Aufwänden. 24

Bild 4.1 Schaltkreis für die Übertragung des Qubit $|x\rangle$ von Alice zu Bob [7, p. 176]. Von links nach rechts: Die ersten Hadamard-Gatter sorgen für die Zufälligkeit im Bit und der Basis, gefolgt vom entsprechenden Wechsel. Analog erfolgt die Messung und das Ergebnis von b auf der Seite von Bob. Ob b korrekt ist zeigt erst eine weitere Absprache der beiden über die gewählten Basen auf alternativem Wege. 40

Bild 4.2 Über eine Photonenquelle erhalten Alice und Bob verschränkte Photonen, die sie durch drehbare Polarisationsfilter zufällig filtern. 42

Bild 4.3 Darstellung von Problem-Klassen und der Möglichkeiten diese Effizient zu lösen nach [48]. 46

Bild 4.4 Verifikation in einem Merkle-Baum. Die gelb eingefärbten Knoten sind die notwendigen Pfadangaben und Bestandteil einer Signatur, um den öffentlichen Schlüssel, den Root-Knoten, zu bestätigen. 50

Bild 4.5 Vergleich durchschnittlicher Laufzeiten in Millisekunden der post Quantenverfahren XMSS und SPHINCS in Vergleich zu schnellen Implementierungen von RSA und ECDSA nach [54, p. 41] auf einem logarithmischen Maßstab. 52

Bild 4.6 2-Dimensionales Beispielgitter mit der Basis (g_1, g_2) und (h_1, h_2) als vereinfachte Darstellung für das Verstehen annähernd senkrechter Vektoren und parallel verlaufenden Vektoren. 53

Bild 4.7 Beispielhafte Darstellung des Closest Vector Problem. Der Closest Vector u ist grün dargestellt und Bestandteil des Gitters mit der Basis (v_1, v_2) . Der Vector a muss das nicht zwingend sein. 54

Bild 4.8 Übersicht gitterbasierter digitaler Signatur-Systeme nach [59, p. 43]. 56

Bild 4.9 Darstellung des Matsumoto-Imai-Schema C^* mit geheimen und öffentlichen Schlüssel sowie der versteckten Hintertür. 61

Bild 4.10 Berechnung der gemeinsamen Kurve E_{AB} analog zum Diffie-Hellman-Schlüsselaustausch ohne die Verwendung des diskreten Logarithmusproblems. 64

Bild 4.11 Alice und Bob berechnen ihren gemeinsamen und geheimen Schlüssel in Anlehnung an das Diffie-Hellman-Verfahren, mittels elliptischer Kurven in einem Graphen. 64

Bild 5.1 Grobkonzept des Verhältnisses zwischen einzelnen Parteien bei der Einführung von kryptographischen Verfahren ohne Darstellung des Rückflusses. 70

Bild 5.2 Nach Vorschlag und Auswahl von Algorithmen und Parametern für die IKE SA (1, 2) erfolgt die weitere Kommunikation bereits verschlüsselt (3, 4) und endet mit der gegenseitigen Authentifizierung der Identitäten, [74]. 78

Bild 5.3 Vereinfachtes Ablaufschema von QSH aufbauend auf das TLS Protokoll. 85

Bild 5.4 Während Bob im Fall 3 mit den alternativen Angaben des X.509 nichts anfangen kann und diese verwirft muss sich Alice im Fall 1 entscheiden ob sie die Authentifizierung akzeptiert oder z.B. gemäß einer Policy verwirft. 90

Tabellenverzeichnis

Tabelle 2.1 Skalierte Dauer der vollständigen Schlüsselsuche, abgebildet auf den Referenzwert des Rekordes aus dem Jahr 2007 [4, p. 91].	20
Tabelle 2.2 Rekorde bei der Berechnung des diskreten Logarithmus aus dem Jahr 2008 [19, p. 95].	22
Tabelle 2.3 Beispiele für die verschiedenen Größenordnungen von Laufzeiten [21].	23
Tabelle 2.4 Stand der RSA Faktorisierungsrekorde 2008 [31, p. 89].	24
Tabelle 4.1 Einfaches Beispiel von schwingenden Photonen und Polarisationsfiltern. Spalte 3 zeigt Photonen, welche den Filter passieren können.	37
Tabelle 4.2 Beispielhaft dargestellte Wertetabelle der übertragenen und gemessenen Werte aufbauend auf Bild 4.1.	40
Tabelle 4.3 Aktuelle Familien in der Post Quantenkryptographie mit Einsatzmöglichkeiten nach [49, p. 10]. (*) Es gibt Vorschläge für den Einsatz in diesem Bereich, die jedoch noch nicht ausreichend untersucht oder bereits gebrochen sind.	47
Tabelle 4.4 Vergleich von implementierten Signaturverfahren nach [54, p. 35]. Insbesondere in den Signaturgrößen gibt es Unterschiede.	51
Tabelle 4.5 Übersicht über die PQC-Familien und ihrer Anwendungsbereiche bezüglich NIST [55].	52
Tabelle 4.6 Vergleich von Schlüsselgrößen der Verfahren NTRU, RSA und ECC [58, p. 22].	55
Tabelle 4.7 Notwendige Parameter sowie erwartete Größen der Texte in Bit, des öffentlichen k_{pb} und privaten Schlüssel k_{pv} um der NIST-Anforderung an ein IND-CCA2 KEM, Cat. 5 zu genügen, was ein Sicherheitslevel von 256 Bit fordert [63, p. 17].	59
Tabelle 4.8 Reduzierung der Schlüsselgrößen durch Optimierungen nach [63].	59
Tabelle 4.9 Tabellarischer Vergleich der Diffie-Hellman-Schlüsselaustausch-Implementierungen nach [69].	63
Tabelle 4.10 Vergleich von SIDH und gitterbasierten DH-Verfahren auf einer gemeinsamen Hardwareplattform nach [70].	65
Tabelle 5.1 Zeitstrahl seit der Veröffentlichung des Shor-Algorithmus bis zum prognostizierter Wettbewerbsabschluss des NIST.	75
Tabelle 5.2 Vergleich der codebasierten Verfahren McEliece und Niederreiter (Parameter: 2048, 1718, $t=30$) mit RSA-2048 nach [75].	80
Tabelle 3 Kryptographische Algorithmen innerhalb von liboqs nach Familien nach [82, p. 3].	86

Anlagen

„Anlage zur Masterthesis: Post – Quantum – Cryptography, kryptographische Lösungen im Umfeld von Quantencomputern“ mit A1 – A40.

Abkürzungsverzeichnis

AES	Advanced Encryption Standard
AH	Authentication Header
API	Application Programming Interface
BB84	Bennett-Brassard 1984
BLISS	Bimodal Lattice Signature Scheme
BQP	Bounded-Error Quantum Polynomial-Time
BSI	Bundesamt für Sicherheit in der Informationstechnik
CA	Certification Authority
CECPQ1	Combined Elliptic Curve and Post Quantum Cryptography Key Exchange
CHSH	Clauser, Horne, Shimony, Holt
CNOT	Controlled-NOT
CRISP	Center for Research in Security and Privacy
CVP	Closest Vector Problem
DES	Data Encryption Standard
DH	Diffie-Hellman
DLP	Diskretes Logarithmus Problem
DME	Double Matrix Exponentiation
E91	Ekert-1991
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
ECI	European Citizens' Initiative
EPR	Einstein, Podolsky, Rosen
ESP	Encapsulated Security Payload
EU-DSGVO	Europäische Union – Datenschutz - Grundverordnung
GGH	Goldreich, Goldwasser, Halevi
ggT	Größter gemeinsamer Teiler
GSM	Global System for Mobile Communications

Halevi	Elliptic Curve Cryptography
HFE	Hidden Field Equations
HSS	Hierarchical Signature System
HTTP(S)	Hypertext Transfer Protocol (Secure)
ICS	Industrial Control System
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
IND-CCA	Ciphertext indistinguishability chosen-ciphertext attack
IP	Internet Protocol
IPsec	Internet Protocol Security
ISAKMP	Internet Security Association and Key Management Protocol
ISMS	Informationssicherheits-Managementsystemen
ISO	International Organization for Standardization
KDF	Key Derivation Function
KEM	Key Exchange Mechanism
KRITIS	Kritische Infrastruktur
LBC	Lattice-based cryptography
LD-OTS	Lamport-Diffie One-Time-Signature
LHV	local hidden variables
LLL	Lenstra, Lenstra und Lovász
LMS	Leighton-Micali Hierarchical Signature System
MD5	Message-Digest Algorithm 5
MQ	Multivariat quadratisch
NIS	Network and Information
NIST	National Institute of Standards and Technology
NP	Nichtdeterministisch Polynomiell
NSA	National Security Agency
NSA	National Security Agency

Abkürzungsverzeichnis

OEM	Original Equipment Manufactur
OQS	Open Quantum Safe
OTP	One-Time-Pad
OTS	One-Time-Signature
P	(deterministisch) Polynomiell
PFS	Perfect Forware Secrecy
PGQCM	Pretty Good Quantum Cloning Machine
PKI	Public Key Infrastrukturen
PQC	Post-Quantum-Cryptography
PRF	Pseudo-Random-Function
QC	Quantencomputer
QC-MDPC	Quasi-Cyclic Moderate Density Parity-Check
QFT	Quanten-Fouriertransformation
QHS	Quantum-Safe Hybrid Key Exchange
QKD	Quantum-Key-Distribution
RC4	Ron's Code 4
RFC	Request for Comments
R-LWE	Ring-Learning With Error
RSA	Rivest–Shamir–Adleman
SA	Security Association
SCA	Side Channel Attack
SHA	Secure Hash Algorithm
SIDH	Supersingular Isogeny Diffie-Hellman
SIKE	Supersingular Isogeny Key Encapsulation
SSL	Secure Socket Layer
SVP	Shortest Vector Problem
TCP	Transmission Control Protocol
TLS	Transport Layer Security

UDP	User Datagram Protocol
VSA	Verschlusssachenanweisung
VS-NfD	Verschlusssache-Nur für den Dienstgebrauch
W-OTS	Winternitz One-Time-Signature
XMSS	eXtended Merkle Signature Scheme
XOR	Exclusive-OR

Thesen zur Masterarbeit

1. Kryptographiebedrohende Quantencomputer werden wahrscheinlich in spätestens 20 Jahren zur Verfügung stehen.
2. Asymmetrische kryptographische Verfahren, basierend auf der Faktorisierung großer Zahlen sowie dem Problem des Lösens des diskreten Logarithmus bieten keinen langfristigen Schutz.
3. Heute zum Zwecke einer Übertragung verschlüsselte Daten können keine sichergestellte Vertraulichkeit mehr garantieren.
4. Symmetrische Verschlüsselungsverfahren sind bei Korrektur der Parametrisierung quantencomputer-resistent, ohne dass die Verfahren selbst ausgetauscht werden müssen.
5. Die Quantenmechanik bietet in Form der Quantum Key Distribution QKD ein sicheres Schlüsselaustauschverfahren, welches nicht unentdeckt angegriffen werden kann.
6. Die Post Quantenkryptographie PQC bietet die Möglichkeit bereits heute quantencomputer-resistente Verfahren einzusetzen.
7. Hybride Schlüsselaustauschverfahren bieten Sicherheit gegen Angriffe ausgehend von klassischen Systemen und Quantencomputern.
8. Das Erforschen, Implementieren, Standardisieren und Ausrollen quantencomputer-resistenter Verfahren wird mehrere Jahre in Anspruch nehmen.
9. Es wird künftig verschiedene kryptographische Primitive auf unterschiedlichen mathematischen Problemen geben, so dass die Komplexität der Kryptographie zunehmen wird.
10. Der Ressourcenbedarf kryptographischer Verfahren wird zugunsten der Quantencomputer-Resistenz zunehmen.

Selbständigkeitserklärung

Hiermit erkläre ich, dass ich die hier vorliegende Arbeit selbstständig, ohne unerlaubte fremde Hilfe und nur unter Verwendung der in der Arbeit aufgeführten Hilfsmittel angefertigt habe.

Berlin, 30.09.2019 Feridun Temizkan

Ort, Datum (Unterschrift)

Anlage
zur Masterthesis
Feridun Temizkan

POST - QUANTUM - CRYPTOGRAPHY
KRYPTOGRAPHISCHE LÖSUNGEN IM UMFELD
VON QUANTENCOMPUTERN

Vorwort

Diese Anlage entstand während meiner Ausarbeitung der Masterthesis

„Post-Quantum-Cryptography.

Kryptographische Lösungen im Umfeld von Quantencomputern“.

Aufgrund des Umfangs und der Komplexität einzelner Themen konnten nicht alle Ausarbeitungen und Beispiele in die Masterthesis übernommen werden. Für ein tiefergehendes Verständnis wurden aus der Masterthesis heraus auf Erweiterungen und Beispiele in diesem Dokument verwiesen.

Diese tiefergehenden Informationen und Beispiele sollen den Zugang zur Materie erleichtern und praxisnäher gestalten.

Inhaltsverzeichnis

A1 MATHEMATISCHE GRUNDBEGRIFFE	3
A2 TURINGMASCHINEN UND ALGORITHMEN	8
A3 DAS HALTEPROBLEM	15
A4 DAS PROBLEM DES HANDLUNGSREISENDEN	17
A5 BEISPIEL VON PROBLEMEN DER KLASSE P, NP UND NP-VOLLSTÄNDIG	18
A6 BEISPIEL DES RESSOURCENVERBRAUCHS EINER VOLLSTÄNDIGEN SCHLÜSSELSUCHE	20
A7 KONFUSION, DIFFUSION UND NICHTLINEARITÄT	24
A8 DATA ENCRYPTION STANDARD, DES	25
A9 ADVANCED ENCRYPTION STANDARD, AES	32
A10 DLP UND COMPUTATIONAL-DIFFIE-HELLMAN-PROBLEM	37
A11 DER DIFFIE-HELLMAN-SCHLÜSSELAUSTAUSCH.....	40
A12 RIVEST, SHAMIR UND ADLEMAN, RSA.....	46
A13 SICHERHEIT VON RSA.....	50
A14 VERGLEICH VON ANGRIFFEN MIT DEM ZIEL DER FAKTORISIERUNG	51
A15 HYBRIDE VERSCHLÜSSELUNGSVERFAHREN	68
A16 SIGNATURVERFAHREN.....	71
A17 DAS DOPPELSPALTEXPERIMENT.....	79
A18 QUANTEN NOT-GATTER	82
A19 HADAMARD-GATTER	83
A20 CONTROLLED NOT-GATTER.....	86
A21 QUANTEN-FOURIERTRANSFORMATION	90
A22 SHOR-ALGORITHMUS	95
A23 GROVERS ALGORITHMUS.....	107
A24 QUANTENTELEPORTATION	112

Inhaltsverzeichnis

A25 No-CLONING-THEOREM.....	117
A26 ZWEI-TEILCHEN-SYSTEME, VERSCHRÄNKUNG, BELL UND EPR	119
A27 GEDANKEN ZUR QKD.....	127
A28 LAMPORT-DIFFIE ONE TIME SIGNATURE, LD-OTS	130
A29 WINTERNITZ ONE TIME SIGNATURE, W-OTS.....	133
A30 eXTENDED MERKLE SIGNATURE SCHEME, XMSS	135
A31 AUFBAU UND EIGENSCHAFTEN VON GITTERN	143
A32 BABAI'S CLOSEST VECTOR-ALGORITHMUS.....	152
A33 DAS GGH-KRYPTOSYSTEM	157
A34 NTRUENCRYPT.....	162
A35 CODIERUNG MIT LINEAREN CODES.....	167
A36 McELIECE-KRYPTOSYSTEM	176
A37 DAS MATSUMOTO-IMAI KRYPTOSYSTEM C*.....	181
A38 HIDDEN FIELD EQUATIONS HFE	183
A39 ELLIPTIC CURVE CRYPTOGRAPHY, ECC.....	186
A40 SUPERSINGULAR ISOGENY DIFFIE-HELLMAN, SIDH.....	192
LITERATURVERZEICHNIS.....	200

A1 Mathematische Grundbegriffe

Nachstehend sollen einige, immer wieder auftretende, Begriffe erläutert werden sowie ggf. aufbauende Begrifflichkeiten, die das Verständnis unterstützen sollen. Dabei handelt es sich nicht um mathematisch präzise Definitionen. Diese können ggf. aus alternativen Quellen entnommen werden.

Surjektivität

Alle Abbilder in der Zielmenge Y müssen mindestens ein Urbild in der Definitionsmenge X aufweisen.

Injektivität

Eine Abbildung ist injektiv, wenn einem Abbild der Zielmenge Y höchstens ein Urbild in der Definitionsmenge X besitzt (linkseindeutig).

Bijektivität

Eine Abbildung ist bijektiv, wenn sie injektiv und surjektiv ist. Zusätzlich muss in der Zielmenge Y der Urbilder der Menge X jedes Urbild auch ein Abbild Y besitzen, d.h. es muss auch die inverse surjektive Abbildung gegeben sein. Beide Mengen haben dieselbe Mächtigkeit.

Gruppen

Eine Gruppe G mit (M, \circ) besteht aus einer Menge M sowie einer definierten Verknüpfung \circ und es gilt, dass die Verknüpfung zweier Elemente der Menge z.B. a, b ein weiteres Element der Gruppe G z.B. c ergibt $a \circ b = c$ mit $a, b, c \in M$, d.h. die Menge M ist abgeschlossen. Ferner müssen die Gruppenaxiome der Assoziativität $(a \circ b) \circ c = a \circ (b \circ c)$ (Bedingungen für eine *Halbgruppe*), Existenz eines neutralen Elements $\varepsilon \in M$ mit $a \circ \varepsilon = a$ (Bedingungen für ein *Monoid*) sowie eines inversen Elements z.B. a^{-1} für a mit der Eigenschaft $a^{-1} \circ a = \varepsilon$ erfüllt sein. Gilt zusätzlich zu diesen Eigenschaften einer Gruppe noch die Kommutativität mit $a \circ b = b \circ a$ handelt es sich um eine kommutative bzw. eine *abelsche Gruppe*.

Ring

Ein Ring R mit (M, \oplus, \otimes) besteht aus einer Menge M sowie zwei definierten

Verknüpfungen \oplus, \otimes wobei (M, \oplus) eine abelsche Gruppe und (M, \otimes) eine Halbgruppe darstellt. Ferner gilt das Distributivgesetz für linksseitig $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$ sowie rechtsseitig $(b \oplus c) \otimes a = (b \otimes a) \oplus (c \otimes a)$. Ist die zweite Verknüpfung ebenfalls kommutativ, so handelt es sich um einen *kommutativen Ring*. Besitzt die zweite Verknüpfung ein neutrales Element, so handelt es sich um einen *unitären Ring*.

Körper

Ein Körper K mit (M, \oplus, \otimes) besteht aus einer Menge M sowie zwei definierten Verknüpfungen \oplus, \otimes wobei für beide Verknüpfungen \oplus, \otimes und der Menge M , gilt, dass sie abelsche Gruppen darstellen. Des Weiteren gilt für beide Verknüpfungen das Distributivgesetz.

Charakteristik

Bei der Charakteristik eines Ringes R oder Körper K handelt es sich um die kleinste Anzahl der benötigten multiplikativen neutralen Elemente $\varepsilon_x = 1$ um das additiv neutrale Element $\varepsilon_+ = 0$ zu erreichen,

$$1_1 + 1_2 + 1_3 + \dots + 1_n = 0, (n \geq 1) \in \mathbb{N}.$$

Ist dieses nicht möglich, so ist die Charakteristik 0, $char(K) = 0$.

Vektorraum

Ein K -Vektorraum (V, \oplus, \odot) ist ein Vektorraum, auch linearer Raum, über dem Körper K mit Vektoren als Elemente in V . Vektoren im Vektorraum können addiert werden (Vektoraddition, kartesisches Produkt $V \times V = \{(v_1, v_2) | \forall v_1, v_2 \in V\} \rightarrow v_1 \oplus v_2 = v_3$). Bei der Multiplikation wird das *kartesische Produkt* aus $K \times V = \{(k, v_1) | \forall k \in K, v_1 \in V\} \rightarrow k \odot v_1 = v_2$ mit ein Körperelement k (auch *Skalar*) und einem Vektor v_1 gebildet (*skalare Multiplikation*). Das Ergebnis ist wieder ein Vektor des Vektorraums V . Die Menge (V, \oplus) muss eine abelsche Gruppe bilden. Für die Verknüpfungen \oplus, \odot muss das Distributivgesetz gelten. Für die Verknüpfung \odot muss die Assoziativität gelten und es muss bezüglich des Körpers K ein neutrales Element $\varepsilon \in K$ existieren. Mit Hilfe der *Basis eines Vektorraums* kann jeder andere Vektor berechnet

werden. Er besteht aus der minimalen Menge der dafür notwendigen Vektoren. Diese Menge wird als *Dimension* bezeichnet.

Untervektorraum

Ein Untervektorraum U ist eine Teilmenge eines Vektorraums V mit $U \subseteq V$ sowie den vorhandenen Verknüpfungen \oplus, \odot mit den Bedingungen, dass der Vektorraum nicht leer sein darf $U \neq \emptyset$, der Vektorraum U abgeschlossen bezüglich der Vektoraddition $U \oplus U = \{(u_1, u_2) | \forall u_1, u_2 \in U\} \rightarrow u_1 \oplus u_2 = u_3, u_3 \in U$ ist sowie der Abgeschlossenheit der Skalarmultiplikation $k \odot V = \{(k, v_1) | \forall k \in K, v_1 \in U\} \rightarrow k \odot v_1 = v_2$ mit $v_2 \in U$ gegeben ist. Ein Untervektorraum stellt selbst einen Vektorraum dar.

Abelsche Varietät

Eine abelsche Varietät ist eine spezielle abelsche Gruppe und entspringt elliptischen Kurven. Es sind geometrische Objekte, welche durch eine Polynomgleichung beschrieben werden und die Struktur einer Gruppe besitzen.

Kern

Als Kern werden die Elemente einer Definitionsmenge X bezeichnet, für die der Funktionswert $f(x)$ der Nullvektor der Zielmenge Y ist $\{x \in X | f(x) = 0 \in Y\}$.

Lineare Abbildung zwischen Vektorräumen

Eine lineare Abbildung zwischen Vektorräumen bildet Elemente über zwei K -Vektorräume X, Y mit zwei Verknüpfungen \oplus, \odot desselben Körper K ab. Die Abbildungen erfüllen die *Homogenität* ($f(\lambda x_1) = \lambda f(x_1)$) sowie die *Additivität* ($f(x_1 + x_2) = f(x_1) + f(x_2)$) für $\forall x_i \in X, \lambda \in K$.

Homomorphismus

Unter Morphismus versteht man eine strukturverträgliche Abbildung. Ein Homomorphismus bezeichnet eine lineare Abbildung zwischen zwei Vektorräumen $f: A \rightarrow B$ für die Homogenität und Additivität gilt.

Endomorphismus

Im Endomorphismus ist die Zielmenge in der Definitionsmenge enthalten wobei die beiden Mengen nicht gleich mächtig sein müssen, $f: A \rightarrow A$.

Monomorphismus

Ein Monomorphismus bezeichnet eine injektive, lineare Abbildung zwischen zwei Vektorräumen $f: A \rightarrow B$.

Epimorphismus

Ein Epimorphismus bezeichnet eine surjektive lineare Abbildung zwischen zwei Vektorräumen $f: A \rightarrow B$.

Automorphismus

Automorphismus liegt dann vor, wenn die lineare Abbildung sowohl injektiv als auch surjektiv, somit bijektiv ist. Im Unterschied zum Isomorphismus ist im Automorphismus die Definitions- und Zielmenge, wie im Endomorphismus, die gleiche, $f: A \rightarrow A$.

Isomorphie

Isomorphismus für $f: A \rightarrow B$ liegt dann vor, wenn die lineare Abbildung sowohl injektiv als auch surjektiv, somit bijektiv ist. Isomorphismus ist somit ebenfalls ein Epi- und Monomorphismus. Isomorphie heißt so viel wie „Strukturgleichheit“. Es werden zwei Strukturen S_1, S_2 z.B. Gruppen, Vektorräume etc., dahingehend untersucht ob relevante Aspekte in den Strukturen gleich sind obwohl sie unterschiedlich aussehen. Ein Beispiel ist die exklusiv-Oder Verknüpfung über eine Wahrheitstabelle der Elemente $\{true, false\}$ S_1 mit der Addition der beiden Elemente aus \mathbb{Z}_2 als S_2 . Es handelt sich um eine bijektive Funktion f von einer Struktur S_1 auf die Andere S_2 mit $f(a \odot b) = f(a) \oplus f(b)$ mit $a, b \in S_1$ und $f(a), f(b) \in S_2$. Bezogen auf eine elliptische Kurve bedeutet die Isomorphie den Wechsel der Koordinaten einer Kurve E_2 bezüglich einer Kurve E_1 über einem Körper K mit

$$x \rightarrow u^2x + r; y \rightarrow u^3y + su^3x + t$$

$$u, r, s, t \in K, u \neq 0.$$

Isogenie

Eine Isogenie nennt man einen Homomorphismus $f: A \rightarrow B$ für die Homogenität und Additivität gilt, f surjektiv ist, also die Zielmenge einer Abbildung φ müssen mindestens ein Urbild aufweisen, und einen endlichen Kern besitzt.

Supersingularität

Eine elliptische Kurve E über einen eine endliche Erweiterung \mathbb{F}_q eines endlichen Körpers \mathbb{F}_p , wobei $p > 3$ und eine Primzahl und $q = p^n, n \in \mathbb{N}$, heißt supersingulär, wenn die Menge der Punkte auf $\#E(\mathbb{F}_q) \equiv 1 \pmod{p}$ ist. Diese Art Kurven können stets über \mathbb{F}_{p^2} definiert werden was bedeutet, dass jede supersinguläre elliptische Kurve über \mathbb{F}_{p^n} isomorph zu einer supersingulären Kurve über \mathbb{F}_{p^2} ist, was das untersuchen einfacher gestaltet.

Irreduzibles Polynom

Ein Polynom P des Körpers K ist genau dann irreduzibel, wenn es sich nicht als ein Produkt zweier nicht invertierbarer Polynome Q, R (vgl. inverse Elemente) beschreiben lässt, d.h. es nicht mit $P = QR$ mit $P, Q, R \in K$ zu vereinfachen. Jedes Polynom P über einen Körper K vom Grad 2 oder 3 (höchster Exponent der im Polynom auftretenden Monomen) ist irreduzibel, wenn es keine Nullstellen in K hat.

A2 Turingmaschinen und Algorithmen

Eine Turingmaschine ist ein Modell der theoretischen Informatik. Alan Turing führte sie 1936 ein, um die Frage der fundamentalen Grenze mathematischer Beweisbarkeit aufzuwerfen. Es ist eine formale Definition der Berechenbarkeit von Funktionen und deren Zeit- und Platzbedarf. Sie ist ein sehr einfaches, abstraktes Modell eines Computers, was jedoch mächtig genug ist, alles zu berechnen, was berechenbar ist. Turingmaschinen formalisieren die Begriffe des Algorithmus und der Berechenbarkeit und sind mathematische Objekte.

Es wird unterschieden in deterministische Turingmaschinen DTM und nichtdeterministische Turingmaschinen NTM. Der dynamische Speicher ist ein in Zellen unterteiltes, zweiseitig unendliches Band und besitzt einen Lese-/Schreibkopf. Es erhält die Eingabe und dient als Arbeitsspeicher (vgl. Von-Neumann-Architektur).

Eine deterministische Turingmaschine $T = (Q, \Sigma, \Gamma, \sqcup, q_0, F, \delta)$ definiert sich wie folgt:

- Q endliche Zustandsmenge,
- Σ endliches Eingabealphabet, typischerweise $\Sigma = \{0, 1\}$,
- Γ endliches Bandalphabet mit $\Sigma \subseteq \Gamma$,
- $\sqcup \in \Gamma$ Blank-Symbol mit $\sqcup \notin \Sigma$,
- $q_0 \in Q$ Startzustand,
- $F \subseteq Q$ Menge der Endzustände,
 - $\delta(q_e, A)$ ist undefiniert für alle $q_e \in F$ und alle $A \in \Gamma$,
- $\delta: Q \setminus F \times \Gamma \rightarrow (\Gamma \times \{L, R, N\} \times Q)$ partielle Übergangsfunktion.

Erläuterung der Übergangsfunktion:

- $F \times \Gamma$ aktueller Zustand und Bandsymbol,
- Q Folgezustand,
- $\Gamma \times \{L, R, N\}$ Bandinschrift/Kopfposition.

Definition einer Sprache der Turingmaschine $T = (Q, \Sigma, \Gamma, \sqcup, q_0, F, \delta)$ mit:

- $w \in \Sigma^*$ Mögliche Eingaben,
- $L(T) := \{w \in \Sigma^* \mid T \text{ mit Eingabe } w \text{ stoppt im Zustand } q \in F \text{ Menge der von } T \text{ akzeptierten Worte,}$
- $L \subseteq \Sigma^*$ aufzählbar, wenn $L = L(T)$ für eine Turingmaschine,
- $L \subseteq \Sigma^*$ entscheidbar, wenn L und $\Sigma^* \setminus L$ aufzählbar.

Die Konfiguration einer Turingmaschine besteht aus

- dem aktuellen Zustand $q \in Q$,
- der Position des Kopfes auf dem Band,
- dem Bandinhalt $\in \Gamma^*$.

Beispiel:

- $\Sigma = \{0, 1\}$ Eingabealphabet,
- $\Gamma = \{0, 1, \sqcup\}$ Bandalphabet, erfüllt $\Sigma \subseteq \Gamma$,
- $T = (Q, \Sigma, \Gamma, \sqcup, q_0, F, \delta)$ Turingmaschine,
- $\sqcup \in \Gamma$ Blank-Symbol mit $\sqcup \notin \Sigma$,
- $Q = \{q_0, q_1, q_2, q_f\}$ endliche Zustandsmenge
- $F = \{q_f\} \subseteq Q$ Menge von Endzuständen,
- δ Übergangsfunktion mit

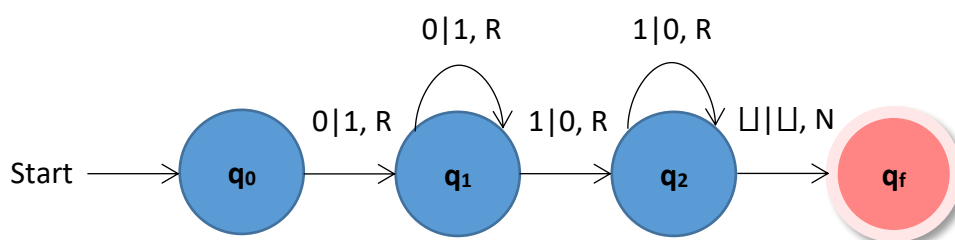


Bild 1 Grafische Darstellung der Turingmaschine $T = (Q, \Sigma, \Gamma, \sqcup, q_0, F, \delta)$, die Eingaben beginnend mit 0 und endend auf 1 akzeptiert, z.B. Eingabe: $\sqcup 0111 \sqcup$, Ausgabe: $\sqcup 1000 \sqcup, [1]$.

Der Endzustand q_f ist als Kreis mit doppeltem Rahmen dargestellt. Die Zustandsübergänge sind mit Tripeln dargestellt, bestehend aus dem gelesenen und dem geschriebenen Zeichen sowie der Kopfbewegung des Lese-/Schreibkopfes des Bandlesers.

Anfangs befindet sich T im Startzustand q_0 , das Band erhält das Eingabewort $w \in \Sigma^*$ (umrahmt von den Blank-Symbolen \sqcup) und der Kopf steht auf dem ersten Zeichen von w (bzw. auf \sqcup , falls $w = \sqcup$). Falls sich T im Zustand $q \in Q$ befindet, der Kopf das Zeichen $A \in \Gamma$ liest und $\delta(q, A) = (q', A', d) \in Q \times \Gamma \times \{L, R, N\}$, dann geht T in den Zustand q' über und ersetzt A durch A' . Für $d = L$ bzw. $d = R$ erfolgt zusätzlich eine Kopfbewegung auf die linke bzw. rechte Nachbarzelle des Bandes.

T stoppt genau dann, wenn T in einem Zustand q ist, ein Symbol A liest und $\delta(q, A)$ undefiniert ist. Eine Eingabe wird akzeptiert genau dann, wenn T im Laufe der Rechnung in einen Endzustand q_f gerät (und dann automatisch stoppt).

Eine nichtdeterministische Turingmaschine NTM ist analog definiert mit dem Unterschied, dass die Überföhrungsfunktion δ die Form

$$\delta: Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\}) \quad (1)$$

hat, wobei in Analogie zur deterministischen Turingmaschine für alle

$$\forall q_f \in F, A \in \Gamma: \delta(q_f, A) = \emptyset \text{ (leere Menge)}. \quad (2)$$

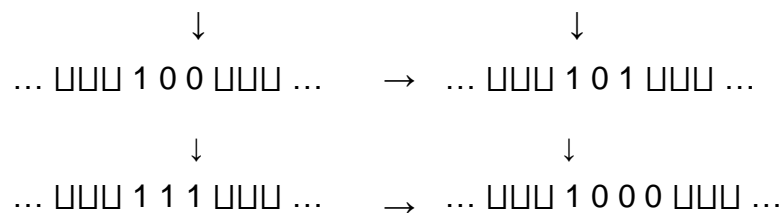
Die nichtdeterministische Turingmaschine bietet die Möglichkeit mehrerer Rechenschritte, die getätigt werden können [2].

Eine Turingmaschine T kann jede Rechenoperation durchführen, die ein Mensch auf dem Papier durchführen kann. Diese durch den Menschen durchgeführten Operationen folgen Algorithmen. Algorithmen lassen sich in Turingmaschinen kodieren.

Beispiel

Eine Turingmaschine T ist demnach ein Programm, welches z.B. eine Addition von 1 zu einer Dualzahl berechnen kann. Hierfür steht auf dem Eingabe-Band der Turingmaschine eine Dualzahl (Binärzahl) wobei links die höchstwertige und rechts die niederwertigste Ziffer steht. Jenseits der Zahl stehen links sowie rechts nur (unendlich viele) Blank-Zeichen \sqcup . Die gegebene Dualzahl soll um 1 erhöht werden. Am Ende soll der Lese-/Schreibkopf der Turingmaschine auf der ersten (linken) Stelle stehen.

Beispiele der Anfangs- und Endpositionen sowie Werte auf dem Band:



Mögliche Skizzierung eines Programms:

1. Zu Beginn soll der Lese-/Schreibkopf auf der niederwertigsten Stelle stehen. Steht er auf einer Ziffer, so muss man so weit nach rechts gehen bis er auf einem Blank \sqcup steht, anschließend wieder ein Schritt nach links.
2. Befindet sich an dieser Stelle eine 0 auf dem Band wird sie durch eine 1 ersetzt.

3. Befindet sich an dieser Stelle eine 1 auf dem Band wird sie durch eine 0 ersetzt. Es folgt ein Schritt nach links. Wiederholung der Schritte ab 2.
4. Die Vorgänge wiederholen sich so lange bis man am linken Ende auf dem Band angekommen ist und auf ein Blank \sqcup trifft.
5. Um den Lese-/Schreibkopf auf der höchstwertigen Stelle der Dualzahl zu positionieren muss der Kopf wieder einen Schritt nach rechts gesetzt werden.

Mit einer Turingmaschine T ist es möglich, alle Sprachen, denen wir bisher begegnet sind, zu akzeptieren und alle Modelle, die wir bisher hatten, zu simulieren. Jede Turingmaschine definiert für sich so einen fest vorgeschriebenen Algorithmus bzw. Programm.

Das Programm der Turingmaschine:

In der Ausgangsposition befindet sich der Lese-/Schreibkopf der Turingmaschine T im Startzustand q_0 ganz rechts es Eingabewertes auf der niederwertigsten Stelle.

$q_0 0 \rightarrow 1 L q_1$ Wenn sich die Maschine im Zustand q_0 befindet und eine 0 an dieser Stelle vorfindet, so wird diese Ziffer durch eine 1 ersetzt. Der Lese-/Schreibkopf wird um eine Stelle nach links versetzt in den Zustand q_1 . Die Addition ist abgeschlossen.

$q_0 1 \rightarrow 0 L q_0$ Wenn sich die Maschine im Zustand q_0 befindet und eine 1 auf dem Band steht, wird diese durch eine 0 ersetzt, der Lese-/Schreibkopf fährt um einen Schritt nach links und die Maschine verbleibt im Zustand q_0 . Die Addition ist noch nicht abgeschlossen.

- $q_1 0 \rightarrow 0 L q_1$ Wenn sich die Maschine im Zustand q_1 befindet, unabhängig davon, ob 0 oder 1 auf dem Band steht, fährt der Lese-/Schreibkopf um einen Schritt nach links. Die Maschine verbleibt im Zustand q_1 .
 $q_1 1 \rightarrow 1 L q_1$
- $q_1 \sqcup \rightarrow \sqcup R q_f$ Wenn sich die Maschine im Zustand q_1 befindet und ein Blank-Zeichen \sqcup auf dem Band befindet, so fährt der Lese-/Schreibkopf um einen Schritt nach rechts und die Maschine geht in den Endzustand q_f über.
- $q_f 0 \rightarrow 0 H q_f$ Wenn sich die Maschine im Endzustand q_f befindet, unabhängig $q_f 1 \rightarrow 1 H q_f$ davon, was gerade auf dem Band steht, so hält die Maschine und $q_f \sqcup \rightarrow \sqcup H q_f$ verbleibt im Endzustand q_f .
 $q_f 1 \rightarrow 1 H q_f$
 $q_f \sqcup \rightarrow \sqcup H q_f$
- $q_0 \sqcup \rightarrow 1 H q_f$ Wenn sich die Maschine im Zustand q_0 befindet und ein Blank-Zeichen \sqcup auf dem Band steht, d.h. der Übertrag bei der Addition setzte sich stellenweise bis zum linken Rand fort, wird dieses durch 1 ersetzt, die Maschine geht in den Endzustand q_f über und hält [3].

Tabelle 4 Tabellarische Darstellung des Programmes für die Turingmaschine T.

Eingaben	
Zustände	0 1 Blank \sqcup
q_0	$1, L, q_1$ $0, L, q_0$ $1, H, q_f$
q_1	$0, L, q_1$ $1, L, q_1$ \sqcup, R, q_f
q_f	$0, H, q_f$ $1, H, q_f$ \sqcup, H, q_f

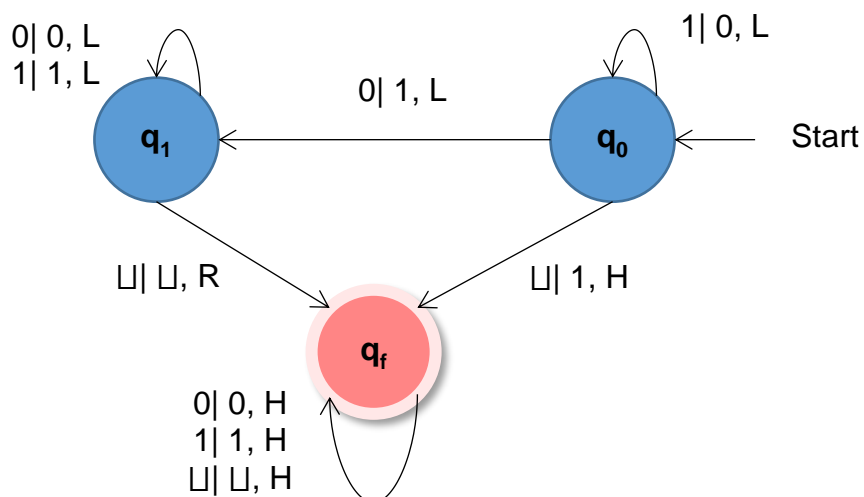


Bild 2 Grafische Darstellung der Turingmaschine T für die Addition von 1 auf eine Dualzahl.

Eine derartige Turingmaschine T befolgt genau ein Programm mit welchem sie eine Eingabe w berechnen kann. Eine universelle Turingmaschine U ist eine Maschine, welche eine bestimmte Turingmaschine T simulieren bzw. dasselbe Programm ausführen kann. Somit kann die universelle Turingmaschine U alle ihre übergebenen Turingmaschinen T mit einem Eingabewort berechnen.



Bild 3 Schematische Darstellung einer universellen Turingmaschine U mit Ein- und Ausgabeparametern.

A3 Das Halteproblem

Ob ein Algorithmus nach Ausführung zu einem Ende gelangt oder nicht wird durch das Halteproblem, eine Fragestellung der theoretischen Informatik, betrachtet (z.B. Endlosschleifen). Damit hängt das Halteproblem sowohl von der Eingabe als auch vom Programm bzw. Übergangsfunktion ab. Es ist ein Berechnungsproblem mit einem Eingabedatum w und einer Übergangsfunktion als Turingmaschine P , welche für Turingmaschinen geeignet sind.

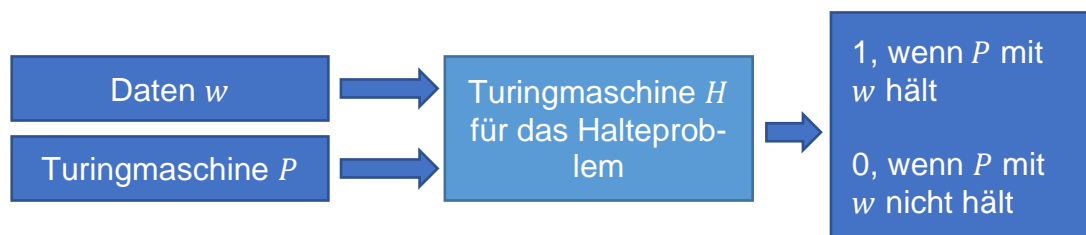


Bild 4 Die Turingmaschine H entscheidet durch ihre Ausgabe, ob P mit w halten wird oder nicht.

Eine solche Turingmaschine H kann es nicht geben, denn

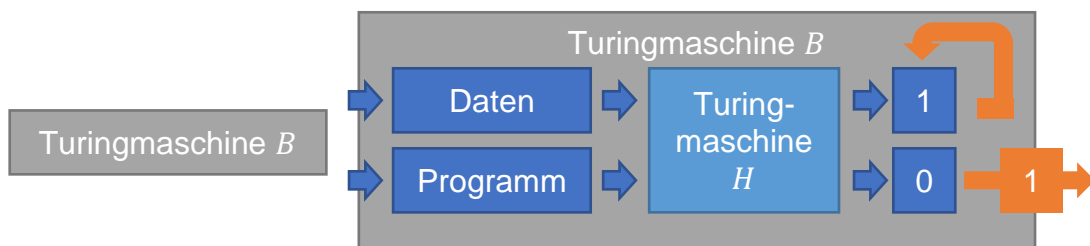


Bild 5 Prinzip des Halteproblems, bei dem eine Maschine B die Ausgabe von H invertiert. Das führt zu einem Widerspruch.

wobei die eingegebenen Daten und das übergebene Programm dieselbe Eingabe darstellt.

Fall 1: Maschine B hält mit Eingabe B , d.h. Turingmaschine H erkennt die Nichtendlichkeit der Eingabe und gibt eine 1 aus, welche wiederum im Programm von Maschine B eine Endlosschleife verursacht. Sie hält nicht und führt somit zu einem Widerspruch.

Fall 2: Maschine B hält nicht mit Eingabe von B , d.h. Turingmaschine H erkennt das B nicht hält und gibt eine 0 aus, welche von der Maschine B in eine 1 geändert wird. In der Folge hält B was zu einem Widerspruch führt.

Keine der beiden Fälle ist möglich. Das Halteproblem ist in dem Fall algorithmisch nicht entscheidbar.

In jedem formalen System, das Turingmaschinen enthält, lassen sich Aussagen formulieren, die weder bewiesen noch widerlegt werden können. Das gilt für Systeme, die einen bestimmten Grad an Komplexität erreichen (z.B. Collatz-Problem „ $(3n + 1)$ -Vermutung“).

A4 Das Problem des Handlungsreisenden

Bei dem Problem des Handlungsreisenden handelt es sich um ein Optimierungsproblem. Die Herausforderung besteht darin eine Handelsroute durch verschiedene Städte festzulegen, bei der lediglich die Ausgangs-Stadt doppelt besucht wird (genau am Anfang und am Ende der Reise) und die Reise als Ganzes möglichst kurz oder anderweitig zu optimieren ist, z.B. günstig, ökonomisch oder bezüglich von Zeitfenstern bei Verbindungen. Das Problem ist für die Problemgröße 2 trivial zu lösen, da es zwischen zwei Städte nur eine Route geben kann. Mit zunehmender Problemgröße steigt jedoch auch die Komplexität des Problems.

A5 Beispiel von Problemen der Klasse P, NP und NP-Vollständig

Beispiel eines Problems der Klasse P wäre ein Problem mit der Laufzeit von $O(n^{1000})$. Verdoppelt man das Problem $n^{1000} \rightarrow (2n)^{1000} = 2^{1000} * n^{1000}$ wobei $2^{1000} \sim 10^{333}$ d.h. bei der Verdoppelung der Eingabemenge $n \rightarrow 2n$ erhöht sich die Laufzeit auf 10^{333} . Nichts desto trotz wäre dieses Problem immer noch Bestandteil der Klasse P, da sie auf einem deterministischen Rechner gelöst werden können.

Nichtdeterministisch polynomiell lösbare Probleme der Klasse NP sind Probleme, deren Lösung in polynomieller Zeit getestet werden können. Im Gegensatz zu P-Problemen, bei denen stets nur ein Weg zur Lösung gegangen werden kann (deterministisch), werden bei der Lösung der NP-Probleme alle Wege zeitgleich begangen (Anwendung in Entscheidungsproblemen. Umgangssprachlich und für die Anschauung wird bei nichtdeterministischen Algorithmen oft von "Raten" gesprochen mit anschließender Verifizierung). NP enthält also Entscheidungsprobleme, deren Verifikation effizient in Polynomialzeit möglich ist. Beispielsweise ist die Zerlegung einer Zahl in ihre Primfaktoren schwer. Das Testen, ob Primzahlen die Faktoren einer anderen Zahl sind, ist einfach. Man kann für diese Probleme einen Algorithmus finden, der das Problem in polynomieller Zeit löst, für dessen Ausführung jedoch ein nichtdeterministischer Computer nötig ist. Nichtdeterministische Maschinen sind theoretische Modelle und im Allgemeinen nicht praktisch realisierbar.

Die schwierigsten Probleme der Klasse der NP-Probleme bezeichnet man als NP-Vollständig. Es sind somit Probleme, die in der Klasse NP liegen müssen und NP-Schwer sind. Man vermutet, dass sich diese Probleme nicht effizient in Bezug auf Geschwindigkeit und somit Zeit sowie Speicherverbrauch bzw. benötigte Ressourcen lösen lassen. Es scheint als gäbe es hierfür keine deterministischen polynomiellen Algorithmen.

Ein Problem p heißt NP-schwer, wenn sich jedes Problem r , das in NP liegt, in deterministisch polynomieller Zeit auf p reduzieren lässt. Ein Problem p heißt NP-vollständig, wenn es NP-schwer ist und selbst in NP liegt.

Ein NP-schweres Problem ist demnach mindestens so schwer wie das schwerste Problem in NP. Gelänge es für ein NP-vollständiges Problem p einen deterministischen polynomiellen Algorithmus zu finden, so lassen sich alle Probleme in NP in deterministisch polynomieller Zeit lösen. Probleme in NP lassen sich dann in deterministisch polynomieller Zeit auf das Problem p reduzieren und sind damit selbst in deterministisch polynomieller Zeit lösbar. Dann wäre die Menge NP gleich der Menge P. Dieses Problem wird als P-NP-Problem bezeichnet und ist Gegenstand der theoretischen Informatik. Es deutet jedoch alles darauf hin, dass die beiden Mengen nicht gleich sind.

Betrachtet man die Probleme L und L' mit $L, L' \subseteq \Sigma^*$ und gibt es eine in polynomieller Zeit berechenbare Funktion $f: \Sigma^* \rightarrow \Sigma^*$ für die alle Wörter $w \in \Sigma^*$ mit $w \in L$ äquivalent zu $f(w) \in L'$, dann ist das Problem deterministisch in polynomieller Zeit reduzierbar.

A6 Beispiel des Ressourcenverbrauchs einer vollständigen Schlüsselsuche

DES ist ein Musterbeispiel, dass die Offenlegung der Funktionsweise eines Verschlüsselungsverfahrens die Sicherheit erhöht. In einem Zeitraum von 20 Jahren ist es nicht gelungen, abgesehen von der Schlüssellänge, eine Schwachstelle im DES zu finden. DES ist ein Verfahren mit einem effektiven 56 Bit Schlüssel, welches jedoch als beste Angriffsmethode nur die vollständige Schlüsselsuche zulässt. Im Jahr 2007 lag der Rekord in etwa bei 22 Stunden, wozu jedoch ein enormer Hardware-Aufwand nötig war. Erhöht man nun den Schlüssel um ein Bit, so verdoppelt sich der Schlüsselraum. Schlüsselgrößen liegen heute im Bereich von 128 bis 256 Bit. Zu einer Schlüsselgröße von 128 Bit unter der Annahme, dass die vollständige Schlüsselsuche, wie beim DES, die effektivste Möglichkeit des Angriffs ist, soll nun folgende Überlegung an die Sicherheit durchgeführt werden:

Schlüssellänge	128 Bit
Schlüsselmenge	$2^{128} = 3,4 * 10^{38}$ Schlüssel

Ferner wird angenommen, dass ein hochspezialisierter Rechenchip für eine Brute-Force-Attacke 10 Millionen Schlüssel pro Sekunde durchprobieren kann. Hierbei soll für jeden Schlüssel auch der entsprechende Algorithmus ausgeführt werden.

Laufzeit:	$3,4 * 10^{31}$ Sekunden
-----------	--------------------------

Es wird ferner angenommen, dass von diesem Rechenchip 1 Millionen parallel zur Verfügung stehen.

Neue Laufzeit:	$3,4 * 10^{25}$ Sekunden
----------------	--------------------------

Bei einer Brute-Force-Attacke werden alle Schlüssel probiert. Im ungünstigsten Fall müssen alle $3,4 * 10^{38}$ Schlüssel getestet werden. Statistisch findet sich der Schlüssel nach 50% aller Versuche. Es wird nun angenommen, dass

Mallory Glück hat und den Schlüssel schon nach 1% aller möglichen Schlüssel findet.

Neue Laufzeit: $3,4 * 10^{23}$ Sekunden

Die Laufzeit würde somit bei 10^{16} Jahren für einen Brute-Force-Angriff auf einen einzigen Schlüssel bedeuten. Die Laufzeit eines Brute-Force-Angriffs liegt also höher als das Alter der Erde mit $4,5 * 10^9$ Jahren = 4,5 Milliarden Jahren bzw. dem Alter des Universums $13,8 * 10^9$ Jahre = 13,8 Milliarden Jahren. Selbst unter den geschilderten optimalen Voraussetzungen benötigt Mallory 1.000.000x öfter die Lebensdauer des Universums, um den Schlüssel auf diese Weise zu ermitteln. Das zeigt die Aussichtslosigkeit dieses Versuches unter dem Aspekt heutiger Möglichkeiten.

Diese Betrachtungsweise ist lediglich mathematischer Natur und könnte auch unter dem Aspekt von Randbedingungen wie dem Energieverbrauch des Systems betrachtet werden, welches einen solchen Schlüssel brechen soll.

Schlüssellänge 128 Bit

Schlüsselmenge $2^{128} = 3,4 * 10^{38}$ Schlüssel

Es wird angenommen, dass dem Angreifer Mallory ein Superrechner zur Verfügung steht, dieser Superrechner außerordentlich energieeffizient ist und für die Berechnung eines Schlüssels ein Billionstel Joule (vgl. Koomeys Gesetz) benötigt. Das entspricht einem Wert von 10^{-12} Joule / Schlüssel.

Gesamtverbrauch: $3,4 * 10^{38}$ Schlüssel * 10^{-12} Joule
= $3,4 * 10^{26}$ Joule

Umrechnung: $3,4 * 10^{26}$ Joule
= $3,4 * 10^{26}$ Watt-Sekunden
= $3,4 * 10^{23}$ Kilowatt-Sekunden
= $9,4 * 10^{19}$ Kilowatt-Stunden

A6 Beispiel des Ressourcenverbrauchs einer vollständigen Schlüsselsuche

Mallory soll unter ähnlich guten Bedingungen wie im Fall zuvor nach bereits 1% aller Versuche den Schlüssel finden.

Gesamtverbrauch: $9,4 * 10^{17}$ Kilowatt-Stunden

Weltweite Stromerzeugung 2016:

24.814,4 Terrawatt-Stunden [4]

= $24,8 * 10^{12}$ Kilowatt-Stunden

Setzt man die Werte ins Verhältnis so stellt man fest, dass es sich um den 38.000-fachen Wert des Energieausstoßes aus dem Jahr 2016 handelt. Bei einem Preis von 1 Cent / Kilowatt-Stunde entstünden Kosten in Höhe von $9,4 * 10^{15}$ Euro.

Geht man von einer vollständigen Schlüsselsuche bei DES binnen 24 Stunden aus (Weltrekord Stand 2007), so ergeben sich für verschieden große Schlüssel folgende Werte:

Tabelle 5 Skalierte Dauer der vollständigen Schlüsselsuche, abgebildet auf den Referenzwert des DES Rekord aus dem Jahr 2007 [5, p. 91].

Schlüssel- länge in Bit	Anzahl der Schlüssel	Dauer der vollständigen Schlüssel- suche
40	$1,1 * 10^{12}$	1,3 Sekunden
56	$7,1 * 10^{16}$	24 Stunden (DES, Referenzwert)
64	$1,8 * 10^{19}$	256 Tage
80	$1,2 * 10^{24}$	45.965 Jahre
128	$3,4 * 10^{38}$	$1,3 * 10^{19}$ Jahre
192	$6,3 * 10^{57}$	$2,4 * 10^{38}$ Jahre
256	$1,2 * 10^{77}$	$4,4 * 10^{57}$ Jahre

Neben dem Aspekt die maximale Sicherheit zu erreichen spielen auch noch andere Faktoren eine gewichtige Rolle. So ist für den Anspruch an die Performance eines Algorithmus entscheidend, wo und wie man diesen einsetzen

kann (Vgl. Verschlüsselung einer Festplatte gegenüber der Verschlüsselung eines Telefonats oder Video-Livestreams). Umfang des Codes, die Hardware-Implementierungstauglichkeit, der Ressourcenbedarf. All diese Faktoren sind ebenfalls Bestandteil von Verschlüsselungsverfahren und ihrer Praxistauglichkeit. Diese Aspekte sollen jedoch nicht direkt Bestandteil dieser Arbeit sein, werden jedoch bei den Vor- und Nachteilen der Post Quantenkryptographie teilweise betrachtet.

A7 Konfusion, Diffusion und Nichtlinearität

Wesentliche Grundprinzipien der Verschlüsselungen sind Konfusion, Diffusion und Nichtlinearität. Konfusion verschleiert den Zusammenhang zwischen Schlüssel und Geheimtext. Die zu verschlüsselnden Daten werden unkenntlich gemacht. Im DES wird diese Aufgabe z.B. von den S-Boxen übernommen. Diffusion verteilt die Information des Klartextes über den Geheimtext. Im Idealfall bewirkt das Ändern eines beliebigen einzelnen Bits des Klartextes, dass sich jedes Bit des Geheimtextes mit einer Wahrscheinlichkeit von $\frac{1}{2}$ ändert. Permutationen finden im DES Verfahren in den Rundenfunktionen statt. Nichtlinearität bedeutet, dass die Verschlüsselungs-Abbildung möglichst weit von einer linearen Abbildung entfernt sein soll. Lineare Funktionen sind einfach umkehrbar und verhalten sich neutral gegenüber Exklusive-Oder-Verknüpfungen. Bei einer Verschlüsselung ist die Regelmäßigkeit und einfache Umkehrbarkeit nicht erwünscht [5, p. 94].

Die ersten beiden Anforderungen, Konfusion und Diffusion, erschweren statistische Verfahren der Kryptoanalyse, die Nichtlinearität ist eine Antwort auf differenzielle und lineare Kryptoanalyse [6, p. 46].

A8 Data Encryption Standard, DES

Symmetrische kryptographische Systeme verwenden den gleichen Schlüssel k für die Ver- und Entschlüsselung von Klartextnachrichten m . Symmetrische Verfahren stellen die ältesten kryptographischen Verfahren dar. Sie sind besonders schnell, einfacher, somit effizienter als asymmetrische Verfahren und gelten bei hinreichender Schlüssellänge allgemein als sicher [7, p. 43].

Ein Beispiel für ein symmetrisches Verschlüsselungssystem ist der Data Encryption Standard DES. DES wurde bis heute nicht gebrochen, wenn man das Maß kryptographischer Verfahren, dass kein Angriff effizienter sein darf, als die vollständige Schlüsselsuche, auch Brute-Force [8, p. 192], ansetzt. Es war jedoch früh abzusehen, dass DES mit seiner 64 Bit Schlüssellänge, wovon 8 Paritätsbit sind, nicht lange ausreichen würde (die Untersuchung konkreter Problemlösung steht bei Laufzeitbetrachtungen nicht im Zentrum).

DES ist ein Feistel-Chiffre mit 16 Runden und das blockweise. Die Verwendung von Runden ermöglicht es den Speicherbedarf derartiger Verfahren zu minimieren. Jede der Runden läuft im Wesentlichen identisch ab und bestehen für gewöhnlich aus den einfachen Funktionen Exklusiv-Oder-Verknüpfung, Permutation sowie Substitution. Der einfache Aufbau bietet die Möglichkeit derartige Verschlüsselungssysteme als Hardware-Komponente zu bauen und so weitere Geschwindigkeitsvorteile zu erlangen. Feistel-Chiffren, auch Feistel-Netzwerke, arbeiten nach diesem Prinzip, sind einfach umkehrbar und arbeiten mit zwei Hälften des zu behandelnden Blocks. Eine Blockhälfte wird der Rundenfunktion zugeführt, die andere Hälfte wird im Anschluss mit dem Ergebnis der Rundenfunktion Exklusiv-Oder-Verknüpft [5, p. 75].

Jeder 64-Bit-Block Klartext wird von DES in einen 64-Bit-Block Geheimtext überführt. Auch der Schlüssel hat eine Länge von 64-Bit. Im DES Verfahren wird der zu verschlüsselnde Block zunächst einer Anfangspermutation P zugeführt. Anschließend wird der Block in die Hälften L_0 und R_0 eingeteilt. Am Ende des Algorithmus wird zur Anfangspermutation P die zugehörige inverse

Rückpermutation P^{-1} durchgeführt. Beide sind im Standard definiert und einsehbar, tragen jedoch nicht zur Sicherheit des Verfahrens bei [8, p. 192].

In jeder der folgenden 16 Runden $i = \{1, \dots, 16\}$ wird die rechte Hälfte R_{i-1} der Rundenfunktion F zugeführt.

$$L_i = R_{i-1} \quad (3)$$

$$R_i = L_{i-1} \oplus F_i(R_{i-1}, k_i) \quad (4)$$

Die Rundenfunktion F beinhaltet

- eine Expansion E zur Erweiterung eines Halb-Blocks von 32 Bit auf 48 Bit,
- eine Exklusiv-Oder-Verknüpfung des Rundenschlüssels k_i ,
- Substitutionen mit Hilfe der standardisierten Substitutionsboxen (S-Boxen) S_1 bis S_8 , sowie
- eine Permutation P .

Parallel hierzu erfolgt eine Schlüsselaufbereitung des Schlüssels k . Zunächst werden die Paritätsbit entfernt. Sollte kein Fehler vorliegen wird der Schlüssel geteilt. Zur jeder der 16 Funktionsrunden F erfolgt die Gewinnung eines neuen Teilschlüssels k_i , der in die Funktionsrunde F_i einfließt.

Nachfolgendes Schema zeigt den Ablauf der DES Verschlüsselung.

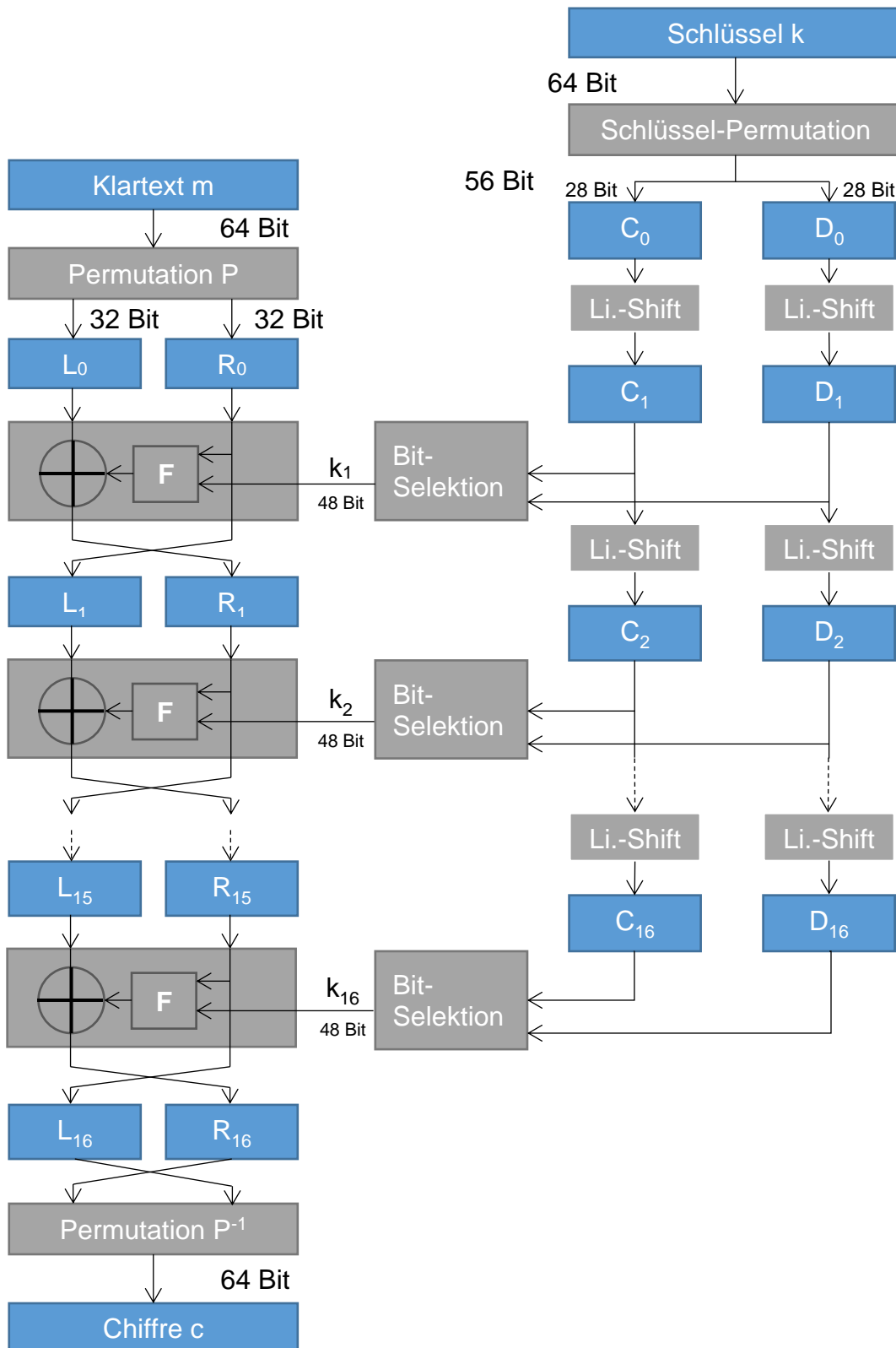


Bild 6 Das DES Verschlüsselungsschema über 16 Runden. Links wird der Klartext in den Chiffre überführt. Rechts werden 16 Teilschlüssel k_i aus dem Hauptschlüssel k gewonnen [7, p. 54].

Die Sicherheit des Verfahrens wird aus den S-Boxen und den Rundenschlüsseln gewonnen, die in jeder Runde nach einem standardisierten Schema erfolgt [8, p. 194]. Die Entschlüsselung erfolgt analog invers wobei zunächst die Rückpermutation P^{-1} durchgeführt wird. Anschließend erfolgt der Durchlauf aller Runden mit absteigendem Index, beginnend bei $i = 16$

$$R_{16} \oplus F_{16}(R_{15}, k_{16}) = L_{15} \oplus F_{16}(R_{15}, k_{16}) \oplus F_{16}(R_{15}, k_{16}) = L_{15}.$$

Auf diese Weise wurden aus R_{16}, L_{16} die Werte R_{15}, L_{15} gewonnen. Das Verfahren wird solange durchgeführt bis der Index $i = 0$ ist. Nach Durchlauf der Eingangspemutation P ist der ursprüngliche Klartext-Block m wiederhergestellt.

Formal lässt sich dieser Block-Chiffre mit Blöcken m_i wie folgt beschreiben:

$$\text{Chiffre-Block } c_i = \text{DES}(k, m_i) \text{ mit } i = 0, 1, 2, \dots, n$$

$$\text{Klartext-Block } m_i = \text{DES}^{-1}(k, c_i),$$

$$\text{mit Schlüssel } k = 2^{61} \rightarrow \text{Quantität } |k| = 2^{56},$$

was die Symmetrie noch einmal hervorhebt.

Doppelte Anwendung des DES-Verfahrens

DES ist nicht nur aufgrund seines Aufbaus und Sicherheit in der Vergangenheit ein ideales Beispiel für symmetrische Verschlüsselung. Auch die Versuche den Nachteil der Schlüssellänge zu korrigieren lassen sich gut diskutieren. So wurde zunächst versucht DES doppelt anzuwenden und dadurch die Schlüssellänge auf 112-Bit zu erhöhen, was theoretisch die vollständige Schlüsselsuche auf mehrere Tausend Jahre verlängern würde.

Alice würde in dem Fall ihre Nachricht m an Bob zweimal, mit Schlüssel k_1 und Schlüssel k_2 , verschlüsseln

$$c = e(k_2, e(k_1, m)) \text{ mit}$$

$$\text{es gibt kein } e(k_3, m) = e(k_2, e(k_1, m)),$$

denn dann würde die einfache Schlüsselsuche nach k_3 die doppelte Verschlüsselung ab absurdum führen. Verfahren, die ihren Schlüssel teilen und während der Chiffrierung einzeln einsetzen, sind jedoch anfällig auf Meet-in-the-middle-Angriffe. Das gilt insbesondere dann, wenn dasselbe Verfahren mehrfach mit verschiedenen Schlüsseln eingesetzt wird, was beim Doppel-DES der Fall ist. Beim Meet-in-the-Middle-Angriff handelt es sich um einen Known-Plaintext-Angriff, d.h. Mallory liegen ein Klartext m und Geheimtext c Paar (m, c) vor.

Mallory verschlüsselt den bekannten Klartext m mit allen 2^{56} möglichen DES Schlüsseln k_{1i} und erhält so die Zwischen-Geheimtexte $x_i^{\text{verschlüsselt}} = e(k_{1i}, m)$ mit $e = DES$. Mallory speichert die Kombination aus Schlüssel k_{1i} und zugehörigem Zwischen-Geheimtext $x_i^{\text{verschlüsselt}}$ in einer Tabelle. Hierbei entstehen derselbe zeitliche Aufwand wie bei der vollständigen Schlüsselsuche bei dem einfachen DES Verfahrens sowie eine hohe Anforderung an Speicherbedarf.

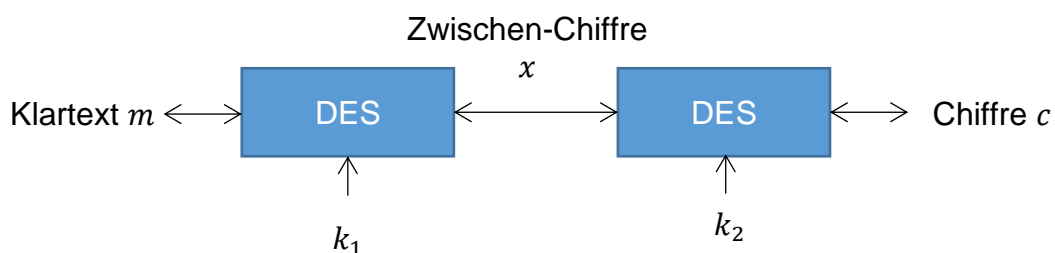


Bild 7 Schematische Darstellung des Meet-in-the-Middle-Angriffs, bei dem von der Klartext-Seite m und von der Chiffre-Seite c aus ein gemeinsames x gesucht wird.

Darüber hinaus greift Mallory das Verfahren auch von der anderen Seite an. Mallory nimmt den bekannten Chiffre c und entschlüsselt diesen mit allen 2^{56} möglichen DES Schlüsseln k_{2i} und erhält so die Zwischen-Geheimtexte $x_i^{\text{entschlüsselt}} = d(k_{2i}, c)$ mit $d = DES^{-1}$. Mallory muss nun untersuchen ob der

gefundene $x_i^{\text{entschlüsselt}}$ Geheimtext in der zuvor erstellten Tabelle als $x_i^{\text{verschlüsselt}}$ vorkommt. Ist dies der Fall hat Mallory das Schlüsselpaar k_1 und k_2 gefunden,

$$e(k_{1i}, m) = x_i^{\text{verschlüsselt}} = x_i^{\text{entschlüsselt}} = d(k_{2i}, c).$$

Das geschieht im ungünstigsten Fall nach zwei vollständigen Schlüsselsuchen, also nach ca. 2x 24 Stunden, gemessen an zuvor dargestellter Tabelle.

Triple-DES

Der nächste Versuch die Sicherheit von DES zu erhöhen war die dreifache Ausführung, Triple-DES. Dabei wird der Schlüsselraum mit den drei DES-Schlüsseln k_1, k_2, k_3 auf $2^{56} \cdot 2^{56} \cdot 2^{56} = 2^{168}$ erhöht. Der Geheimtext c setzt sich aus der Berechnung

$$c = e_{k_1}(d_{k_2}(e_{k_3}(m)))$$

zusammen. Es erfolgt also als zweiter Schritt eine Entschlüsselung $d = DES^{-1}$ mit Schlüssel k_2 als Unterschied zur einfachen und wiederholten hintereinander ausgeführten Verschlüsselung. Triple-DES hat, wie Doppel-DES, eine Anfälligkeit für Meet-in-the-Middle-Attacken. Auch hier liegt Mallory ein Klartext m sowie ein zuehöriger Geheimtext c vor. Mallory erstellt erneut eine vollständige Tabelle für

$$x_i^{\text{entschlüsselt}} = DES_{k_{3i}}^{-1}(c)$$

für alle 2^{56} Schlüssel für k_3 . Anschließend muss Mallory den Zwischenwert für

$$z_i^{\text{verschlüsselt}} = DES_{k_{2i}}^{-1} \circ DES_{k_{1i}}(m)$$

berechnen. Dadurch vermeidet Mallory zwar den Gesamtaufwand der vollständigen Schlüsselsuche von 2^{168} und reduziert diesen auf maximal

$2^{56} + 2^{56} \cdot 2^{56} \approx 2^{112}$, dennoch ist dieser Aufwand an Zeit und Speicherbedarf unrealistisch hoch, um ihn durchzuführen zu können. Durch die dreifache Ausführung des DES-Verfahrens ist Triple-DES jedoch von schlechter Performance im Vergleich zu anderen Verfahren [5, p. 91].

A9 Advanced Encryption Standard, AES

Die Antwort auf Schwächen von DES war ein Wettbewerb im Jahre 1998, der den künftigen symmetrischen Verschlüsselungsalgorithmus Advanced Encryption Standard AES ermitteln sollte, d.h. es gab den Namen des Algorithmus bereits vor seiner Entwicklung.

Eine der Bedingungen war, dass es sich um einen vollständig dokumentierten Blockchiffre mit Referenzimplementierung handelt, geeignet für Hard- und Software und frei von Patentrechten. Ferner sollten Schlüssellängen von 128 Bit, 192 Bit sowie 256 Bit unterstützt werden. Als Maßstab für die Sicherheit galt, dass die vollständige Schlüsselsuche das beste Angriffsverfahren darstellen muss. Als Zielplattformen galten u. A. Smartcards, welche nur begrenzte Ressourcen zur Verfügung haben.



Bild 8 AES ist ein Block-Chiffre Verfahren, welches 128 Bit Blöcke verschlüsselt und Schlüsselgrößen von 128, 192 sowie 256 unterstützt [9, p. 105].

In einem über mehrere Runden laufenden Wettbewerb etablierten sich fünf AES-Kandidaten, von denen letztendlich das Rijndael-Verfahren den Wettbewerb für sich entscheiden konnte. Seither ist er unter dem Namen Advanced Encryption Standard AES bekannt und gilt als Nachfolger von DES. Zehn Jahre danach wurde AES erstmals erfolgreich mit dem Bliclique-Angriff attackiert. Dieser Angriff ist im Schnitt ca. um den Faktor 4 schneller als die vollständige Schlüsselsuche, was jedoch für die praktische Sicherheit nicht relevant ist. Dieser Angriff kann AES mit einem 128-Bit Schlüssel in $2^{126,1}$ Schritten brechen. Ein Indikator für die angenommene Sicherheit lässt sich auch daraus ableiten, dass die National Security Agency NSA im Jahr 2003 das Verschlüs-

seln vertrauliche Dokument bis zur Stufe SECRET mit AES allen Schlüssel-
längen und bis zur Stufe TOP SECRET mit den Schlüssellängen 192 oder 256
Bit gebilligt hat [9, p. 135].

In der symmetrischen Verschlüsselung mit AES spielen endliche Körper
 $GF(2^8)$, ähnlich der asymmetrischen Verschlüsselung, eine zentrale Rolle.

Im Gegensatz zum DES ist AES nicht als Feistel-Netzwerk aufgebaut und ar-
beitet als Substitutions-Permutations-Chiffre. Je Schlüssellänge variiert die
Anzahl an Runden wobei die Blocklänge von 128 Bit mit 10 Runden standar-
disiert ist, Rijndael jedoch auch größere Schlüsse mit mehr Runden unter-
stützt. Der 128-Bit Block wird in eine 4x4 Matrix überführt, bei der jede Zelle
die Größe von 1 Byte hat. Der Klartext

$$m(a_0, a_1, a_2, a_3, b_0, b_1, b_2, b_3, c_0, c_1, c_2, c_3, d_0, d_1, d_2, d_3)$$

wird in eine Matrix Form überführt

$$\begin{pmatrix} a_0 & b_0 & c_0 & d_0 \\ a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \end{pmatrix}. \quad (14)$$

Das AES Verfahren besteht aus vier Funktionen, die jeweils mehrfach durch-
laufen werden, und kurz vorgestellt werden sollen.

SubBytes()

Diese Funktion realisiert, wie im DES, die S-Boxen für die nichtlineare Substi-
tution des Klartext-Blocks und somit die Konfusion.

ShiftRow()

In diesem Schritt werden die Zeilen verschoben und durchgemischt. Hierdurch
wird der Block permutiert, was für die Anforderung der Diffusion sorgt.

$$\begin{pmatrix} a_0 & b_0 & c_0 & d_0 \\ a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \end{pmatrix} \rightarrow \begin{pmatrix} a_0 & b_0 & c_0 & d_0 \\ b_1 & c_1 & d_1 & a_1 \\ c_2 & d_2 & a_2 & b_2 \\ d_3 & a_3 & b_3 & c_3 \end{pmatrix}.$$

MixColumn()

Auch diese Funktion ist linear, permutiert und sorgt für Diffusion. Hierbei liegt der Fokus aber auf dem komplexeren Durchmischen von Spalten. Hierbei werden die Bytes der Spalten mit den Konstanten 1, 2 oder 3 multipliziert. Für ein Byte b definieren sich die Multiplikationen mit den Konstanten wie folgt

$$b \cdot 1 = b,$$

$$b \cdot 2 \Rightarrow \text{Linksshift von } b, \text{ falls } b < 128 \text{ Bit,}$$

$$\Rightarrow (\text{Linksshift von } b) \oplus 00011011, \text{ falls } b \geq 128 \text{ Bit,}$$

$$b \cdot 3 = (b \cdot 2) \oplus b.$$

Die Spalten werden dann wie folgt neu berechnet und jeder Wert b_i in den neuen Wert b_i' überführt

$$b_0' = (b_0 \cdot 2) \oplus (b_1 \cdot 3) \oplus b_2 \oplus b_3,$$

$$b_1' = b_0 \oplus (b_1 \cdot 2) \oplus (b_2 \cdot 3) \oplus b_3,$$

$$b_2' = b_0 \oplus b_1 \oplus (b_2 \cdot 2) \oplus (b_3 \cdot 3),$$

$$b_3' = (b_0 \cdot 2) \oplus b_1 \oplus b_2 \oplus (b_3 \cdot 3).$$

AddRoundKey()

In der einzigen Funktion, die vom Schlüssel abhängt, wird ein Subschlüssel zu der 4x4-Matrix bitweise Exklusiv-Oder-Verknüpft. Dies geschieht in jeder Runde sowie in der Vorrunde für jedes Byte b der Zeile i , Spalte j

$$b_{ij}' = b_{ij} \oplus k_{ij}.$$

Neben diesen Funktionen müssen aus dem 128-Bit Schlüssel noch für die Rundenzahl r insgesamt k_{r+1} Teilschlüssel generiert werden. Das geschieht

in der Schlüsselaufbereitung. Hierbei werden vor der ersten Runde und für jede durchlaufende Runde die Operationen `SubWord()`, `RotWord` sowie die Verrechnung einer Konstanten `Rcon` durchgeführt, um alle $r + 1$ Teilschlüssel zu generieren. Auf dieser Grundlage wird AES nach folgendem Ablauf durchgeführt:

1. Vorrunde
 - a. `AddRoundKey()` mit dem ersten Schlüssel k_1 , welcher dem Initialschlüssel entspricht.
2. Runde r ($r = \{1, \dots, r - 1\}$ in Abhängigkeit der Blockgröße 128-, 192-, 256-Bit)
 - a. `SubBytes()`
 - b. `ShiftRow()`
 - c. `MixColumn()`
 - d. `AddRoundKey(k_{r+1})`
3. Abschlussrunde r
 - a. `SubBytes()`
 - b. `ShiftRow()`
 - c. `AddRoundKey(k_{11})`, [5, p. 120].

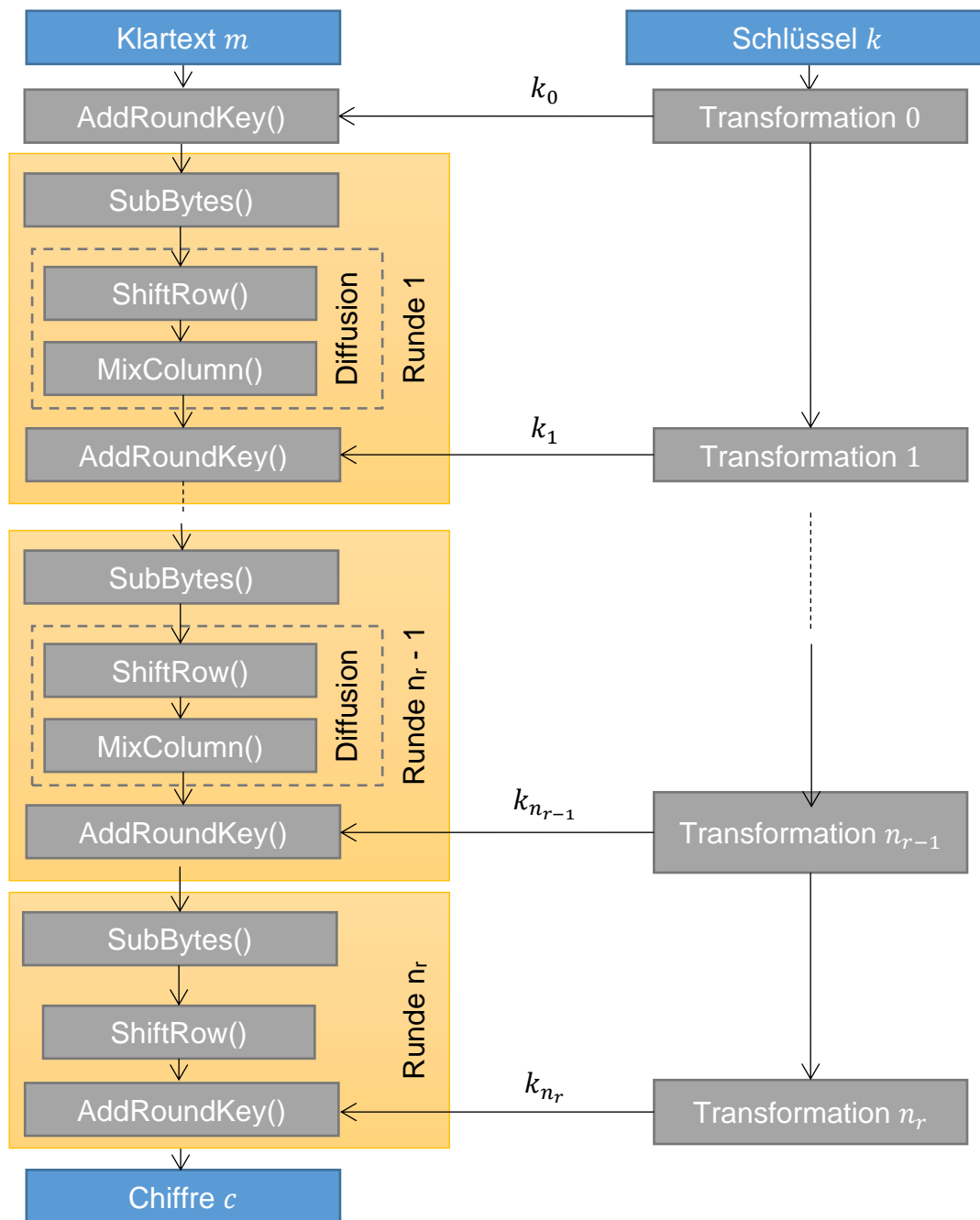


Bild 9 Blockdiagramm der AES-Verschlüsselung [9, p. 107].

A10 DLP und Computational-Diffie-Hellman-Problem

Die Sicherheit bzw. das Problem eines Angreifers einen ausgehandelten Schlüssel zu berechnen, (Computational-) Diffie-Hellman-Problem CDH, ist eng verwandt mit dem Diskreten-Logarithmus-Problem DLP. Der diskrete Logarithmus ist die Umkehrung der diskreten Exponential-Funktion

$$f(x) = a^x \bmod p.$$

Das Paar bildet eine Einwegfunktion: Die Berechnung großer Potenzen ist durchführbar. Das Multiplizieren vergrößert einen Exponenten um +1, das Quadrieren verdoppelt einen Exponenten:

$$a^e * a = a^{e+1} \text{ sowie } a^e * a^e = a^{2*e}.$$

Mit diesem Vorgehen kann man effizient große Potenzen berechnen, was sich insbesondere für duale Zahlen auf klassischen Rechnersystemen eignet. Die Verdoppelung eines Exponenten entspricht dem Aufschieben um die Ziffer 0, die erneute Multiplikation mit der Basis vergrößert den Exponenten um 1, aus der niederwertigsten Stelle 0 wird so als Beispiel eine 1. So wird bei dem Verfahren der Exponent aufgebaut. Beispiel-Algorithmus zum Potenzieren von a^{13} :

Zielwert	a^{13}	=	a^{1011}
Startwert			a^1
Quadrierung			a^{10}
Quadrierung			a^{100}
Multiplikation			a^{101}
Quadrierung			a^{1010}
Multiplikation			a^{1011} = Zielwert

Wenn der Exponent e eine Länge von 1_e Binärstellen hat, dann sind $1_e - 1$ Quadrierungen und im Mittel $(1_e - 1)/2$ Multiplikationen erforderlich (für jede Stelle mit dem Wert 1, außer der ersten Stelle). Bei einem Exponenten mit

1000 Binärstellen sind damit etwa 1000 bis 2000 Quadrierungen oder Multiplikationen erforderlich.

Die Umkehrung, der diskrete Logarithmus, ist für große Zahlenbereiche praktisch nicht durchführbar. Diskrete Logarithmen finden Anwendung beim Schlüsselaustausch nach Diffie-Hellman, beim ElGamal-Verfahren und in der Kryptographie mit elliptischen Kurven [7]. Sind die Zahlen a, b und n gegeben, so ist der diskrete Logarithmus die Zahl x , für die gilt:

$$a^x = b \text{ mod } n.$$

Ein Beispiel zur Veranschaulichung der Problematik:

Sei das Modul p die Primzahl $p = 37$,

sei der Generator die Basis $a = 2$,

$$f(x) = 2^x \text{ mod } 37,$$

so sind die Funktionswerte $f(x)$ für x einfach zu berechnen und darzustellen.

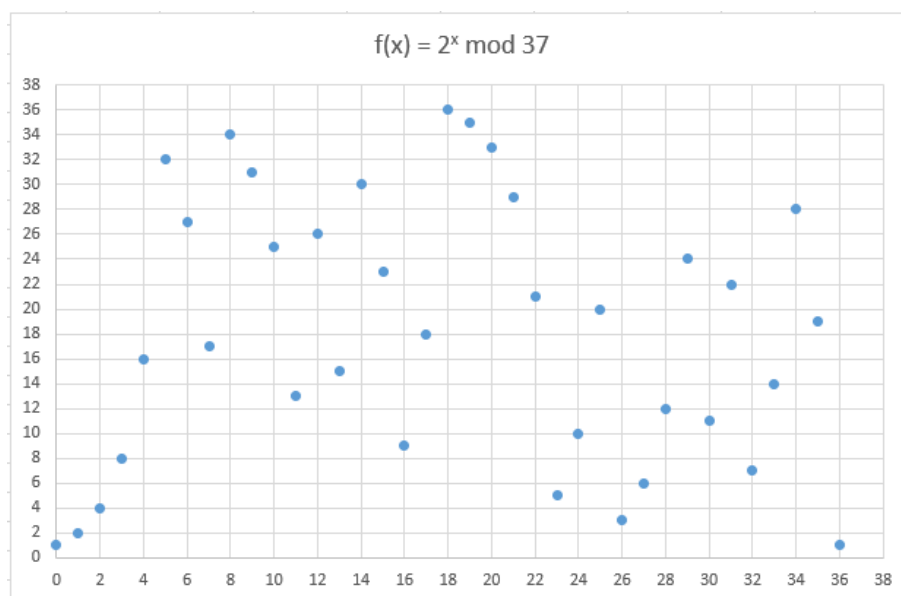


Bild 10 Darstellung der Funktion $f(x) = 2^x \text{ mod } 37$. Die Darstellung der Werte lässt kein Muster erkennen und so eine Prognose treffen.

Der Aufwand für die Berechnung eines Punktes steigt nur linear mit der Stellenlänge 1_p des Moduls p (s. diskreten Exponential-Funktion). Für die Werte $x > 5$ ergibt sich jedoch kein Muster. Will man die Gleichung $14 = 2^x \bmod 37$ lösen, dann hat man in der Folge trotz des „nahen“ Funktionswertes $13 = 2^{11} \bmod 37$ und $15 = 2^{13} \bmod 37$ keinen Hinweis, dass sich die Lösung erst mit $14 = 2^{33} \bmod 37$ einstellen wird [7].

Es sind bisher keine schnellen Algorithmen zur Berechnung des diskreten Logarithmus bekannt. Es gibt aber Algorithmen, die die Lösung gezielter finden als bloßes Ausprobieren bzw. Brute-Force. Die Laufzeit für den einfachsten Algorithmus steigt mit der Ordnung $\mathcal{O}(p)$. Aufgrund des hohen Laufzeitverhaltens und der in der Kryptografie üblichen Größenordnungen (mehrere hundert Dezimalstellen als Modul und Basis) spielen sie praktisch aber keine Rolle.

Tabelle 6 Rekorde bei der Berechnung des diskreten Logarithmus aus dem Jahr 2008 [10, p. 95].

Bit	Jahr	Entwicklergruppe	Algorithmus
193	1991	Lamacchia-Odlyzko	Gaußsche Methode
216	1995	Weber	Zahlkörpersieb
282	1996	Weber	Gaußsche Methode
299	1998	Joux-Lercier	Gaußsche Methode
332	1999	Joux-Lercier	Zahlkörpersieb
365	2000	Joux-Lercier	Zahlkörpersieb
398	2001	Joux-Lercier	Zahlkörpersieb
432	2005	Joux-Lercier	Zahlkörpersieb

A11 Der Diffie-Hellman-Schlüsselaustausch

Der Diffie-Hellman-Schlüsselaustausch, DH-Verfahren, nutzt das Problem des diskreten Logarithmus DLP für einen Schlüsselaustausch zwischen Alice und Bob aus. Dieses Verfahren, von Whitfield Diffie und Martin Hellman, legte den Grundstein für die asymmetrische Kryptographie und ist das Erste seiner Art. Die Bedeutung dieser Entwicklung wird auch mit der kopernikanischen Wende verglichen: „*Die Entwicklung der asymmetrischen Verschlüsselung hat für die Kryptographie eine vergleichbare Bedeutung wie die Kopernikanische Wende für die Astronomie und stellt neben der Digitalisierung der Informationen und dem Internet als Kommunikationsplattform ein Fundament des dritten Jahrtausends dar* [11, p. 53]“.

In DH-Verfahren handeln Alice und Bob über ein unsicheres Netz einen gemeinsamen Schlüssel unter Austausch von Parametern aus, die theoretisch alle von Mallory gelesen werden dürfen. Alice und Bob behalten lediglich ein Geheimnis für sich. Alice und Bob einigen sich auf eine gemeinsame Basis p ihrer Gruppe. Eine möglichst große Primzahl, die den endlichen Zahlenkörper definiert. Darüber hinaus wählen beide eine gemeinsame, natürliche Zahl g . Für die Voraussetzung einer Gruppe G gilt:

- Es ist eine Verknüpfung definiert, bei der man zwei Elemente der Gruppe miteinander verknüpfen kann und so ein weiteres Element dieser Gruppe erhält.
- Die Verknüpfung ist assoziativ, $a \cdot (b \cdot c) = (a \cdot b) \cdot c$.
- Es gibt ein neutrales Element e welches bei Verknüpfung mit einem anderen Element a genau dieses wieder gibt $a \cdot e = a$.
- Zu jedem Element a gibt es ein inverses Element a' für das gilt $a \cdot a' = e$, [12], vgl. A1.

Generatoren

Durch die Verknüpfung mittels Modulo-Multiplikation und der Primzahl p sind diese Bedingungen erfüllt. Die 0 ist, ohne ein inverses Element, von dieser Gruppe ausgeschlossen. Selbiges gilt für die Modulo-Addition, hier mit der 0

als neutrales Element. Man spricht in dem Zusammenhang vom Galois-Feld $GF(p)$ oder auch von einem Körper der Größe p . Für die natürliche Zahl g gilt

$$g < p.$$

Die Zahlen g und p werden von Alice und Bob offen gewählt und kommuniziert. Mallory ist also in Kenntnis beider Zahlen. Darüber hinaus wählen Alice und Bob weitere natürliche Zahlen x_{Alice} und y_{Bob} mit

$$x_{Alice} < p,$$

$$y_{Bob} < p.$$

Diese Zahlen behalten Alice und Bob jeweils für sich.

Alice berechnet: $a = g^x \bmod p$

Bob berechnet: $b = g^y \bmod p$

Beide Ergebnisse werden untereinander ausgetauscht (Auch diese Werte sind vor Mallory nicht geheim zuhalten) und für eine weitere Rechnung verwendet.

Alice berechnet: $k_{Alice} = b^x \bmod p$

Bob berechnet: $k_{Bob} = a^y \bmod p$

Es gilt:

$$\begin{aligned} k_{Alice} &= b^x \bmod p \\ &= (g^y \bmod p)^x \bmod p \\ &= (g^y)^x \bmod p \\ &= g^{yx} \bmod p \\ k_{Bob} &= a^y \bmod p \\ &= (g^x \bmod p)^y \bmod p \\ &= (g^x)^y \bmod p \end{aligned}$$

$$= g^{xy} \bmod p$$

Da die Multiplikation von Exponenten kommutativ ist gilt somit

$$k_{Alice} = k_{Bob} = k.$$

k ist der gemeinsame Schlüssel von Alice und Bob, der nun für eine symmetrisch verschlüsselte Kommunikation verwendet werden kann. Mallory kann diesen nicht berechnen, obwohl ihm in seiner Gleichung lediglich einer der geheimen Werte fehlt. Um k zu berechnen müsste Mallory den diskreten Logarithmus lösen wofür es jedoch keinen effizienten Algorithmus unter Verwendung klassischer Rechensysteme zu geben scheint.

Ein Kriterium für die Sicherheit symmetrischer Verfahren war, dass die effektivste Methode zum Brechen der Verschlüsselung die vollständige Schlüsselsuche sein muss. Für das asymmetrische Diffie-Hellman-Verfahren gilt das nicht, da es bessere Algorithmen gibt als die vollständige Schlüsselsuche. Einige dieser Möglichkeiten werden nachfolgend beschrieben. Um dem entgegenzuwirken werden von Alice und Bob entsprechend größere Parameter gewählt als es bei symmetrischen Verfahren der Fall ist.

Das DH-Verfahren sieht von sich aus zunächst keine Authentisierung der Partner vor, daher ist das gesamte Verfahren anfällig für Man-In-The-Middle-Angriffe. Bei korrekter Implementierung erreicht das Verfahren von Diffie-Hellman Perfect Forward Secrecy, PFS [13]. Grund hierfür ist, dass der Sitzungsschlüssel einer Kommunikation niemals übertragen wird. Er wird auf jeder Seite berechnet. Sämtliche Parameter hierfür werden unmittelbar im Anschluss verworfen und ggf. stetig ein neuer Sitzungsschlüssel generiert, so dass ein Angreifer selbst bei Berechnung eines Sitzungsschlüssels nie sofort die gesamte Kommunikation entschlüsseln könnte, sondern mehrfach vor diesem Problem steht.

Die Sicherheit des Diffie-Hellman-Schlüsselaustausch basiert neben der Größe der gewählten Parameter auch auf der Wahl der öffentlich zugänglichen

Zahlen. Der diskrete Logarithmus für eine Primzahl p darf nicht effizient berechnet werden können. Je größer die Zahl p , je ineffizienter die Algorithmen, desto aufwändiger ist jedoch auch das Austauschverfahren selber (Das BSI empfiehlt 3000 Bit Schlüssellängen spätestens ab dem Jahr 2023 [14, p. 14]). Für eine starke Primzahl p gilt

$$p - 1 = 2 * q \text{ mit } q: \text{Primzahl}$$

Starke Bsp.: $p = 11$, da $11 - 1 = 5 * 2$, $q = 5$: Primzahl

Schwache Bsp.: $p = 13$, da $13 - 1 = 2 * 6$, $q = 6$: kleine Primzahl, [15].

Dann sind nur 2 und q echte Teiler von $p - 1$ und es gibt nur zwei nicht-triviale Untergruppen – eine der Ordnung 2 und eine der Ordnung q . Die Untergruppe der Ordnung 2 ist $\{1, p - 1\}$, die restlichen Elemente sind je zur Hälfte Elemente der Untergruppe der Ordnung q und erzeugende Elemente der gesamten Gruppe.

Beispiel: Sei $p = 11$. Offenbar ist p eine starke Primzahl. Die Menge ganzer Zahlen $Z_p^* = Z_{11}^*$ (auch $Z(p, \cdot) = Z(11, \cdot)$) besteht dann aus folgenden Elementen:

$$\text{Ganze Zahlen } Z_{11}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}.$$

Die Untergruppe der Ordnung 2 besteht aus den Elementen 1 und 10, die Untergruppe der Ordnung 5 besteht aus den Elementen 1, 3, 4, 5, 9. Die Elemente 2, 6, 7, 8 sind erzeugende Elemente von ganzen Zahlen Z_{11}^* .

Wenn also $p - 1 = 2q$ ist, findet man ein erzeugendes Element von ganzen Zahlen Z_p^* , indem man zufällig ein Element g mit $1 < g < p - 1$ wählt und prüft, ob $g^q = 1 \text{ mod } p$ ist. Ist dies der Fall, so verwirft man g , denn dann ist g kein erzeugendes Element der gesamten Gruppe, sondern Element der Untergruppe der Ordnung q . Man wählt dann ein neues Element usw. Die Wahrscheinlichkeit, dass man auch nach n Versuchen noch kein erzeugendes Element gefunden hat, beträgt lediglich $\frac{1}{2}$ [16].

Neben der Primzahl p spielt auch die Wahl der natürlichen Zahl g eine große Rolle im Diffie-Hellman-Schlüsselaustausch. Sie kann nicht beliebig gewählt werden. g muss die Ordnung $p - 1$ haben. Wie zuvor genannt lauten die drei Gruppenaxiome, dass die Verknüpfung assoziativ ist, ein neutrales Element und ein inverses Element besitzt.

Nimmt man ein beliebiges Element a aus der Gruppe Z_p^* und betrachtet die Menge $\{a^0, a^1, a^2, a^3, \dots, a^{p-1} \text{ mod } p\}$, dann erhält man eine Untergruppe. Das Element a heißt dann Generator dieser Untergruppe. Jede Untergruppe von Z_p^* hat mindestens einen Generator und damit auch Z_p^* selbst.

Tabelle 7 Beispiel der Generatoren 1 bis 4 der Untergruppe Z_5 .

1			
2	4	3	1
3	4	2	1
4	1		

Beispiel für den Generator $g = 1 \text{ mod } 5$:

$$1^0 = 1 \quad 1^1 = 1 \quad 1^2 = 1 \quad 1^3 = 1 \quad 1^4 = 1$$

Beispiel für den Generator $g = 2 \text{ mod } 5$:

$$2^0 = 1 \quad 2^1 = 2 \quad 2^2 = 4 \quad 2^3 = 3 \quad 2^4 = 1$$

Beispiel für den Generator $g = 3 \text{ mod } 5$:

$$3^0 = 1 \quad 3^1 = 3 \quad 3^2 = 4 \quad 3^3 = 2 \quad 3^4 = 1$$

Beispiel für den Generator $g = 4 \text{ mod } 5$:

$$4^0 = 1 \quad 4^1 = 4 \quad 4^2 = 1 \quad 4^3 = 4 \quad 4^4 = 1$$

Tabelle 8 Beispiel der Generatoren 1 bis 12 der Untergruppe Z_{13} .

1											
2	4	8	3	6	12	11	9	5	10	7	1
3	9	1									

4	3	12	9	10	1						
5	12	8	1								
6	10	8	9	2	12	7	3	5	4	11	1
7	10	5	9	11	12	6	3	8	4	2	1
8	12	5	1								
9	3	1									
10	9	12	3	4	1						
11	4	5	3	7	12	2	9	8	10	6	1
12	1										

Punktuelles Beispiel für den Generator $g = 8 \bmod 13$:

$$\begin{array}{lllll}
 8^0 = 1 & 8^1 = 8 & 8^2 = 12 & 8^3 = 5 & 8^4 = 1 \\
 8^5 = 8 & 8^6 = 12 & 8^7 = 5 & 8^8 = 1 & 8^9 = 8 \\
 8^{11} = 5 & 8^{12} = 1 & & &
 \end{array}$$

Die Zahl g sollte so gewählt werden, dass alle Zahlen zwischen 1 und $p - 1$ als Resultat aus $g^a \bmod p$ in Frage kommen. Für das erste aufgeführtes Beispiel wären das die Generatoren 2 und 3, für das zweite Beispiel die Generatoren 2, 6, 7, 11. Idealerweise ist also die Zahl g eine Primitivwurzel zum Modul p , d.h. ein Erzeuger der Gruppe Z_p^* . Eine Primitivwurzel erzeugt jedes Element der primen Restklassengruppe.

Der Extremfall für eine falsche Wahl von g wäre $g = 1$. Die von $g = 1$ erzeugte Untergruppe ist $\{1\}$, d.h. als Ergebnis für k ist nur 1 möglich.

Auch $g = p - 1$ ist eine schlechte Wahl, denn die von $p - 1$ erzeugte Untergruppe ist $\{1, p - 1\}$, sodass als mögliche Ergebnisse für k nur 1 und $p - 1$ in Betracht kommen (s. Beispieltabelle). Ein Angreifer braucht also in diesem Fall keine diskreten Logarithmen zu berechnen, sondern nur die Schlüssel $k = 1$ und $k = p - 1$ auszuprobieren.

A12 Rivest, Shamir und Adleman, RSA

Ron Rivest, Adi Shamir und Leonard Adleman entwickelten ein kryptographisches Verfahren, das nach den Nachnamen der Erfinder als RSA bezeichnet wird. Es handelt sich dabei nicht nur um eines der ältesten, sondern um das mit Abstand wichtigste und bekannteste asymmetrische Verschlüsselungsverfahren [17]. Es kann sowohl für Verschlüsselung als auch für digitale Signaturen verwendet werden.

RSA ist ein asymmetrisches Public-Key-Kryptoverfahren, bei dem ein öffentlicher Schlüssel $K_{pub} = (n, e)$ und ein privater Schlüssel $K_{priv} = (d)$ zum Einsatz kommen. Die Public-Key-Kryptographie wurde von Diffie und Hellman in ihrer Veröffentlichung *Whitfield Diffie, Martin E. Hellman: New Directions in Cryptography* diskutiert. Im Abschnitt III beschreiben beide das Problem der Schlüsselverteilung. Auf Basis der Arbeit von Diffie und Hellman entstand 1977 das RSA-Verfahren.

Die Sicherheit dieses Verfahrens basiert auf dem Faktorisierungsproblem. Das Multiplizieren zweier Primzahlen p und q zur Zahl n ist einfach, kann mit Computern sehr effizient berechnet werden und stellt Rechner auch im Falle der Multiplikation großer Zahlen vor keinerlei Probleme. Für die Umkehrung ist jedoch kein effizientes Verfahren bekannt, bei dem man aus dem Produkt zweier Primzahlen n die beiden Faktoren p und q berechnen kann. Daher spricht man auch hier von einer Einwegfunktion. Für diese Einwegfunktionen gibt es sogenannte Falltüren, die die Rückwärtsrechnung mit Hilfe einer Zusatzfunktion ermöglichen.

Eine Umkehrung der Modulo-Exponentiation ist das Modulo-Wurzelziehen. Zu gegebenen a, b und n wird eine Zahl x gesucht, für die gilt:

$$x^a = b \bmod n. \quad (1)$$

x heißt die a -te Wurzel von $b \bmod n$. Um die Frage zu beantworten wann eine Modulo-Wurzel existiert, benötigt man die φ -Funktion. Sie gibt an wie

viele natürliche Zahlen größer als 0, kleiner als eine Zahl n sind und zu n teilerfremd sind. Es gilt:

$$\varphi(n) = (p - 1) * (q - 1),$$

wenn n das Produkt zweier Primzahlen p, q ist, sonst gilt

$$\varphi(n) = n - 1.$$

Sind a und n natürliche Zahlen mit $a < n$, dann gilt:

$$a^{1 \pmod{\varphi(n)}} = a \pmod{n}. \quad (2)$$

Zurück zu gesuchtem x aus (1). Sind a und $\varphi(n)$ teilerfremd, also $\text{ggT}(a, \varphi(n)) = 1$, dann kann man mit dem euklidischen Algorithmus die Zahl

$$c = a^{-1 \pmod{\varphi(n)}}$$

berechnen. Beide Seiten der Gleichung $x^a = b \pmod{n}$ werden nun mit dem Wert c bei $\varphi(n)$ exponiert:

$$(x^a)^{c \pmod{\varphi(n)}} = b^{c \pmod{\varphi(n)}} \pmod{n}.$$

So vereinfacht sich die Gleichung nach:

$$(x^a)^{c \pmod{\varphi(n)}} = b^c \pmod{n}.$$

Da $a * c = 1 \pmod{\varphi(n)}$:

$$x^{1 \pmod{\varphi(n)}} = b^c \pmod{n}.$$

Nach Anwendung von (2) vereinfacht sich die Gleichung:

$$x = b^c \text{ mod } n. \quad (37)$$

Somit ist die a -te Wurzel von b der Wert x wenn die genannte Bedingung $ggT(a, \varphi(n)) = 1$ eingehalten wurde. Dieses Verfahren ist einfach durchzuführen, wenn man $\varphi(n)$ kennt. Dieses zu berechnen ist einfach, wenn man die beiden Primzahlen p und q kennt mit denen das Modul n generiert wurde. Wie zuvor geschildert ist es jedoch schwer aus dem Produkt n die Primfaktoren p und q zu berechnen. Somit lässt sich auch die Wurzel nicht berechnen. Nach heutigem Wissensstand gibt es keine andere Methode der Wurzelberechnung [5]. Die Falltürfunktion ist die Faktorisierung des Moduls n .

Für eine gesicherte Kommunikation zwischen Alice und Bob werden ein öffentlicher und privater Schlüssel erzeugt.

1. Alice wählt zwei große, verschiedene Primzahlen p und q und berechnet das RSA-Modul $n = p * q$
2. Alice berechnet die Eulersche φ -Funktion von n mit $\varphi(n = p * q) = (p - 1) * (q - 1)$
3. Alice wählt eine natürliche Zahl e , die teilerfremd zu $\varphi(n)$ ist, e mit $ggT(e, \varphi(n)) = 1$
 $1 < e < \varphi(n)$.

Alice hat nun einen öffentlichen Schlüssel $k_{pub} = (n, e)$.

1. Alice berechnet ihren privaten Schlüssel $d = k_{priv}$ als
 $d = e^{-1} \text{ mod } \varphi(n)$, es gilt

$$d \odot e \equiv 1 \text{ mod } n.$$

2. Nach Berechnung der Schlüssel sind p, q und $\varphi(n)$ nicht mehr nötig und können verworfen werden.

Kennt Bob Alice öffentlichen Schlüssel e , dann kann er damit die Nachricht m verschlüsseln und an Alice schicken. Dazu berechnet er für den Klartext m den Geheimtext c mit

$$c = m^e \bmod n.$$

Die Verschlüsselung entspricht der Modulo-Exponentiation. Alice kann diese Nachricht nicht auf gleiche Weise (symmetrisch) entschlüsseln. Sie benötigt dafür ihren anderen, privaten Schlüssel (asymmetrisch). Für die erneute Berechnung des Klartextes m aus dem Geheimtext c berechnet Alice

$$m = c^d \bmod n.$$

Die Entschlüsselung entspricht dem Modulo-Wurzelziehen. Sie zieht die e -te Wurzel aus c in dem sie die d -te Potenz berechnet (Falltürfunktion, das Ziehen der e -ten Wurzel). Mallory hingegen kann d nicht ermitteln, da sie die Faktorisierung von n nicht kennt (Faktorisierungsproblem bzw. RSA-Problem RSAP).

$$c^d = (m^e)^d = (\sqrt[e]{m^e}) = m^{1(\bmod \varphi(n))} = m \bmod n \quad (3)$$

(3) zeigt, dass wenn eine Nachricht m mit $e * d \bmod n$ potenziert wird, die ursprüngliche Nachricht m gewonnen werden kann. Statt dies zeitgleich zu tun wird lediglich ein Exponent verwendet. Es ist dabei unerheblich welcher Exponent verwendet wird (hier als privat oder öffentlich bezeichnet). Wichtig ist, dass nur einer der beiden öffentlich bekannt wird während der andere geheim bleibt.

A13 Sicherheit von RSA

Der private RSA-Schlüssel d kann aus dem öffentlichen Schlüssel (e, n) berechnet werden, falls entweder $\varphi(n)$ oder die Faktorisierung $n = p \cdot q$ aus n ermittelt werden könnte.

Man könnte meinen, dass $\varphi(n) = (p - 1) * (q - 1)$ nahe an $n = p * q$ liegt und somit durch Probieren gefunden werden kann. Das täuscht jedoch, denn der Abstand ist

$$n - \varphi(n) = p + q - 1.$$

Für den Fall, dass q eine Zahl von $1_q \approx 522$ Bit ist, dann müssten ca. 2^{522} Zahlen durchprobiert werden, was aktuell nicht durchführbar ist.

Die Sicherheit hängt davon ab, dass Mallory nicht die Primfaktoren p oder q aus dem Modul n in realistischer Zeit berechnen kann. Ob der Aufwand für das Faktorisieren oberhalb einer gewissen Schranke liegt, konnte bisher nicht bewiesen werden. Es ist also nicht auszuschließen, dass ein Mathematiker einen Algorithmus findet, der die Faktorisierung auch für sehr große Zahlen löst bzw. in überschaubarer Zeit bewerkstelligt. Geschieht das, dann wäre der RSA Algorithmus nicht mehr verwendbar. Die Sicherheit des RSA-Algorithmus hängt also von einem offenen mathematischen Problem ab [7].

Der RSA-Algorithmus wird zwar nicht durch klassische Systeme bedroht, jedoch ist er mit implementiertem Algorithmus von Shor auf ausreichend starken Quantencomputern nicht mehr sicher.

A14 Vergleich von Angriffen mit dem Ziel der Faktorisierung

Vollständige Schlüsselsuche

Mit der Public-Key-Only-Attacke kann Mallory mithilfe von Alice öffentlichen Schlüsseln beliebig Geheimtexte erstellen und durch Ausprobieren sämtlicher Schlüssel den geheimen Schlüssel k_{priv} finden. Dieses Vorgehen entspricht dem bereits bei der symmetrischen Verschlüsselung vorgestellten vollständigen Schlüsselsuche. Es wurde gezeigt, dass bereits 256 Bit mehr Berechnungen nach sich ziehen als heute durchführbar sind.

Allerdings sind 256 Bit-Schlüssel in der asymmetrischen Verschlüsselung nicht zeitgemäß, da es deutlich bessere Methoden eines Angriffes als die vollständige Schlüsselsuche gibt. Aktuelle Empfehlungen für Schlüsselgrößen liegen aufgrund dieser besseren Strategien bei mindestens 2048 Bit oder höher.

Faktorisierungsangriffe

Die Sicherheit des RSA-Verfahren hängt eng mit der Schwierigkeit zusammen, natürliche Zahlen in ihre Primfaktoren zu zerlegen. Es ist nicht bekannt, ob das Faktorisierungsproblem für natürliche Zahlen leicht oder schwer ist, dennoch wurden in den letzten Jahren Fortschritte bei der Suche nach effizienteren Methoden gemacht. Jedoch sind auch diese Methoden noch fern ab von anwendbaren Lösungen, so dass das RSA Verfahren immer noch Sicherheit bietet. Im folgenden Abschnitt werden punktuell Verfahren skizziert, die das Problem der Faktorisierung darstellen und zu lösen probieren. Dadurch soll ein besseres Verständnis für die Sicherheit von Verfahren entstehen, welche auf dem Faktorisierungsproblem beruhen [18].

Probedivision

Die Probedivision ist ein einfaches Verfahren, um die Primfaktoren einer Zahl n zu erhalten. Die Zahl n ist zusammengesetzt und es gilt

$$n = a * b \text{ mit}$$

$$\begin{aligned}
 a &> 1, b > 1 \\
 a &\leq \sqrt{n} \text{ oder} \\
 b &\leq \sqrt{n}, \text{ da sonst} \\
 n &= a * b > \sqrt{n} * \sqrt{n} = n.
 \end{aligned}$$

Um festzustellen ob n eine Primzahl ist braucht man also „nur“ für alle Primzahlen p , die nicht größer als \sqrt{n} sind, zu testen, ob sie n teilen.

$$n/p \text{ für alle } 2 \leq p \leq \lfloor \sqrt{n} \rfloor \text{ ohne Rest teilbar und } p \in \mathbb{P}.$$

Gibt es einen Teiler $a > \sqrt{n}$ von n , dann ist $p = n/a$ der Komplementärteiler und $b < \sqrt{n}$. Es reicht somit aus die Suche nach Primfaktoren bis \sqrt{n} auszuführen. Sollte bis zu dieser Grenze keine Zahl gefunden worden sein, so ist n selber eine Primzahl oder 1. Da das Vorhandensein kleiner Faktoren sehr viel wahrscheinlicher ist als das von großen Faktoren und Divisionen schnell durchgeführt sind, hat das Verfahren seine Daseinsberechtigung. Es ist sinnvoll für Zahlen kleiner 106 [18, p. 156].

Tabelle 9 Beispiel einer Probedivision anhand der Zahl $n = 1.270$.

$$n = 1.270, \lfloor \sqrt{n} \rfloor = 35$$

$2 \leq p \leq \lfloor \sqrt{n} \rfloor, p \in \mathbb{P}$	n/p	Primfaktor
2	$1.270 / 2 = 635$	2
3	-	-
4	$635 / 5 = 127$	5

Da 127 selbst eine Primzahl ist, wird die Probedivision an der Stelle beendet und nicht bis 31 fortgeführt. Das Ergebnis lautet $n = 1.270 = 2 * 5 * 127$.

Für ein Ergebnis benötigt man wenigstens $\frac{\sqrt{n}}{\log_2(\sqrt{n})}$ -Probefaktorisierungen. Für eine Primzahl der Größenordnung 10^{75} würde man demnach $\frac{10^{\frac{75}{2}}}{\log_2(\sqrt{10^{\frac{75}{2}}})} > 3,7 * 10^{35}$ Probefaktorisierungen durchführen müssen.

Im RSA-Verfahren werden jedoch Primzahlen der Größenordnung 10^{154} verwendet. Daher ist die Probefaktorisierung für einen Angriff auf heutige RSA-Verfahren nicht sinnvoll [18, p. 157].

Faktorisierungsmethode von Fermat

Die meisten modernen Faktorisierungsmethoden verwenden die Darstellung eines Produktes aus der Differenz zweier Quadrate a, b und der 3. Binomischen Formel:

$$n = a * b \text{ mit}$$

$$a = x + y, b = x - y$$

$$n = x^2 - y^2$$

$$= (x + y) * (x - y)$$

$$= (x - \sqrt{f(x)}) * (x + \sqrt{f(x)}) \text{ mit}$$

$$y^2 = x^2 - n \text{ oder}$$

$$f(x) = x^2 - n \text{ (Kraitchik's quadratisches Polynom)}$$

Zunächst wählt man für a den nächst größeren oder gleichen ganzzahligen Wert von \sqrt{n} . Anschließend inkrementiert man diesen so lange um 1 bis man ein Quadrat gefunden hat:

$$a = \lceil \sqrt{n} \rceil$$

$$a = \lceil \sqrt{n} \rceil + 1$$

$$a = \lceil \sqrt{n} \rceil + 2$$

...

Tabelle 10 Beispiel der Faktorisierungsmethode von Fermat anhand $n = 703$.

$n = 703$		
	Runde 1	Runde 2
a	$27 (= \lceil \sqrt{703} \rceil)$	28
a^2	729	784
$f(a) = a^2 - n$	26	$81 = 9^2$

Die Faktoren der Zahl $n = 703 = (a^2 - f(a)^2) = (28 - 9)(28 + 9) = 19 * 37$.

Tabelle 11 Beispiel der Faktorisierungsmethode von Fermat anhand $n = 15.229$.

$n = 15.229$				
	Runde 1	Runde 2	Runde 3	Runde 4
a	$124 (= \lceil \sqrt{15.229} \rceil)$	125	126	127
a^2	15.376	15.625	15.876	16.129
$f(a) = a^2 - n$	147	396	647	$900 = 30^2$

Die Faktoren der Zahl $n = 15.229 = (a^2 - f(a)^2) = (127 - 30)(127 + 30) = 97 * 157$.

Die Fermat-Faktorisierung funktioniert dann gut, wenn p und q dicht beieinander liegen. Wählt man die Primzahlen jedoch mit auseinander Abstand, so wirkt man der Fermat-Faktorisierung entgegen. Im schlechtesten Fall benötigt das Verfahren für $n = 3p$ mit der Primzahl p dann $\Omega(n)$ Iterationen. Damit ist die (maximale) Laufzeit des Algorithmus mindestens der Größenordnung von n , also exponentiell bzgl. der Stellenzahl von n . In Fall $n = 3p$ ist die Brute-Force Faktorisierung mit Probedivisionen schneller [19].

Dieses Verfahren lässt sich mittels Faktorisierungsmethode von Russell Sherman Lehman auf eine Laufzeit von $O(\sqrt[3]{n})$ bzw. $O(\sqrt[3]{n} \log \log n)$, je nach angenommenen Aufwand für das Berechnen von Wurzeln, optimieren.

Beide Verfahren sind deterministisch und können auf klassischen Systemen implementiert werden.

Pollard'sche Rho-Methode

Der Mathematiker John Pollard entwickelte 1975 ein probabilistisches Verfahren zur Primfaktorzerlegung einer zusammengesetzten Zahl n für $n \geq 4$, bei dem die Laufzeit von der Größe der Primfaktoren abhängt.

Die Zahl n ist zusammengesetzt und es wird angenommen, dass p ein unbekannter Primfaktor von n ist, so sucht man nach $x, y \in \mathbb{Z}$ mit

$$x \equiv y \pmod{p} \tag{4}$$

$$x \not\equiv y \pmod{n}. \tag{5}$$

Aus (4) und (5) folgt: $1 < \text{ggT}(x - y, n) < n$ [20]. Die Zahl n , als Produkt mit einem Faktor p , ist ein Vielfaches von p . Ebenso ist $(x - y)$ ein Vielfaches der Zahl p , somit liefert der $\text{ggT}(x - y, n)$ einen echten Teiler der Zahl n [21].

Das Verfahren beruht auf einer Folge von Pseudozufallszahlen. Sei $f(x)$ eine ganzzahlige Polynomfunktion, so ist die Folge von Pseudozufallszahlen

$$x_0 = S, x_{i+1} = f(x_i) \pmod{n}, \text{ (Pollard-Sequenz)}$$

schließlich periodisch. Die Folge startet mit dem Wert x_0 , weitere Werte werden iterativ berechnet mit:

$$x_i = f(x_{i-1})$$

wobei sich

$$x_i = x_{i-1}^2 + c \tag{6}$$

als Polynom zur Erzeugung der Zahlen eignet. Mit dieser Methode ergibt sich eine Folge (Vorperiode) $x = (x_0, x_1, x_2, \dots, x_{k-1})$, deren Glieder der Bedingung

der Pollard-Sequenz genügen. Wegen der Endlichkeit der auftretenden Reste gibt es in jeder solchen Pollard-Sequenz ein Index-Paar $k > j$, so dass $x_k = x_j$ gilt, d. h. nach endlich vielen Schritten wiederholen sich Folgenglieder und sind gleich.

Nach der Bildungsvorschrift gilt dann auch $x_{k+1} = x_{j+1}$ für $i \geq 0$. Der Umstand, dass sich nach einer Vorperiode die Glieder wiederholen, kann grafisch dargestellt werden und gab der Methode letztendlich auch ihren Namen.

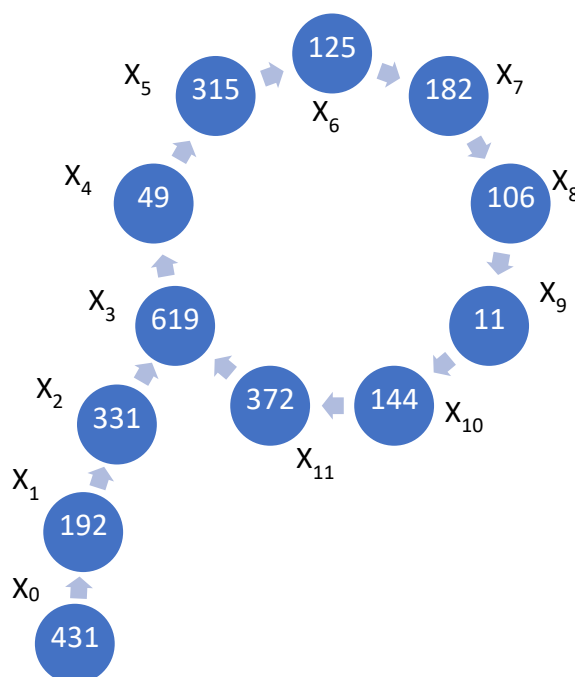


Bild 11 Das charakteristische Rho-Symbol, welches die Vorperiode und Periode zeigt und der Methode ihren Namen gab.

So ergibt sich aus den Perioden $x_k = x_j$ dann $x_k = x_{2k}$ bzw. $x_k \equiv x_{2k} \pmod p$. Haben zwei Werte x und $y \pmod p$ den gleichen Wert, so ergibt sich der

$$ggT(x_k - x_{2k}, n) \text{ bzw. } ggT(x - y, n)$$

ein Vielfaches von p und oftmals einen echten Teiler von n [21]. Jedoch ist der Vergleich sämtlicher Zahlenpaare (x_i, x_j) sehr aufwändig. Mit Hilfe des Algorithmus von Floyd zum Auffinden von Schleifen, auch Hase-Igel-Algorithmus,

wird so gewählt, dass x_i stets die nächste und x_j die übernächste Zahl betrachtet. Das funktioniert aufgrund von $x_i \equiv x_j \pmod p$, da der Unterschied zwischen x_i und x_j lediglich ein Vielfaches der Zahl p ist.

$$y_0 = S, y_i + 1 = f(f(y_i) \pmod n) \pmod n$$

Es ergeben sich so die zwei Folgen

$$x = (x_1, x_2, x_3, x_4, \dots) \text{ sowie}$$

$$y = (y_1, y_2, y_3, y_4, \dots) = (x_2, x_4, x_6, x_8, \dots).$$

Es genügt den $ggT(x_i - y_i, n)$ bzw. $ggT(x_i - x_{2*i}, n)$ zu berechnen. Zur Berechnung der Zahlenfolge kann eine Funktion der Form

$$f(x) = x^2 + c \text{ mit}$$

$$c \not\equiv 0 \pmod n \quad \text{und}$$

$$c \not\equiv 2 \pmod n$$

nach (6) verwendet werden.

Tabelle 12 Faktorisierung der Zahl 74.539 nach Pollard's Rho-Methode mit den angegebenen ersten vier Schritten (1.), (2.), (3.) und (4.).

$n = 74.539, f(x_i) = (x_i - 12 + 1) \pmod n, \text{Startwert } x_0 = 2$			
i	$x_i = f(x_{i-1})$	$y_i = x_{2i}f(f(y_{i-1}))$	$ggT(x_i - y_i, n)$
1	(1.) $x_1 = (x_0^2 + 1) \pmod n$ $(2^2 + 1) \pmod{74.539} =$ 5	(3.) $y_1 = x_2$ 26	(4.) $ggT(x_1 - y_1 , n)$ 1
2	(2.) $x_2 = (x_1^2 + 1) \pmod n$ $(5^2 + 1) \pmod{74.539} =$ 26	11.096	1
3	677	536	1
4	11.096	71.723	1
5	57.328	13.078	1
6	536	3.880	1

7	63.680	23.332	131
---	--------	--------	-----

Somit ergibt die Primfaktorzerlegung

$$n = 74.539 = 131 * (74.539/131) = 131 * 569.$$

Basierend auf Erfahrungen endet die Pollard's-Rho-Methode nach $\mathcal{O}(\sqrt{p}) \leq \mathcal{O}(n^{\frac{1}{4}})$ Schritten mit einer Wahrscheinlichkeit von $\frac{1}{2}$, da nicht alle Berechnungen für p notwendig sind (vgl. Geburtstagsparadox) [22]. Die verbesserte Laufzeit von bis zu $\mathcal{O}(\sqrt{p})$ gelingt also nur unter Annahme der Erfolgsgarantie und den Einsatz einer so genannten "Monte-Carlo-Methode" [9].

Das quadratische Sieb

Einer der effizientesten Algorithmen zur Faktorisierung ist das Quadratische Sieb von Carl Pomerance aus dem Jahr 1981. Sei auch hier die Zahl n eine natürliche Zahl, zusammengesetzt aus den ganzen Zahlen a und b mit

$$n = a * b,$$

$$a, b \in \mathbb{p},$$

$$a^2 \equiv b^2 \pmod{n},$$

$$a \not\equiv \pm b \pmod{n}$$

Die Zahl N ist ein Teiler von $a^2 - b^2$, nach der 3. Binomischen Formel $a^2 - b^2 = (a-b) * (a+b)$, teilt aber weder $(a-b)$ noch $(a+b)$. Es gilt

$$N \mid a^2 - b^2 \text{ und somit}$$

$$N \mid (a - b) * (a + b),$$

$$ggT(a - b, N),$$

$$ggT(a + b, N),$$

und man hat mit a und b zwei Faktoren von n gefunden. Es ist Ziel des quadratischen Siebs genau diese zwei Zahlen zu finden [19], wobei es sehr aufwendig ist diese Zahlen zu suchen.

Es ist effektiver diese Zahlen zu generieren. Hierfür werden B –glatte Zahlen einer Faktorbasis F verwendet. B –glatte sind die Zahlen, deren einzelne Primfaktoren kleiner oder gleich dieser Schranke B sind, d.h.

$$F_B = \{p \leq B \mid p \in \mathbb{P}\}.$$

Auch diese Menge muss nicht vollständig betrachtet werden. Es genügt n quadratischer Rest Modulo p , d.h. die Faktorbasis lautet nun

$$F_B = \{p \leq B \mid p \in \mathbb{P} \text{ und } n \text{ quadratischer Rest Modulo } p\}.$$

Eine Zahl, z.B. a heißt genau dann quadratischer Rest, bezogen auf ein Modul m , wenn sie zu m teilerfremd ist $\text{ggT}(a, m) = 1$ und es eine Zahl x gibt, für die die Kongruenz

$$x^2 \equiv a \pmod{m}$$

gilt. Gibt es keine Zahl x für die das zutrifft, so ist a quadratischer Nichtrest modulo m .

Beispiel

Es wird die natürliche Zahl $n = 22.213$ für die Faktorisierung betrachtet. Als Schranke B wird 19 gewählt. Man erhält die Faktorbasis F mit

$$F_B = F_{19} = \{p \leq 19 \mid p \in \mathbb{P}\}.$$

Nun wird aus F die explizite Faktorbasis F_B bestimmt, in dem geprüft wird für welche Primzahlen p aus F die Zahl n quadratischer Rest $x^2 \equiv n \pmod{p}$ ist.

Tabelle 13 Bestimmung der expliziten Faktorbasis.

B-glatt								
p (Modulo)	2	3	5	7	11	13	17	19
$n \bmod p$	1	1	3	2	4	9	11	2
$x^2 \bmod p$	1	1	-	9	4	9	-	-
x	± 1	± 1	-	± 3	± 2	± 3	-	-

Alternativ kann man mit Hilfe des Satzes

$$n^{\frac{(p-1)}{2}} \bmod p \tag{61}$$

untersuchen ob $n = 1$ quadratischer Rest $\bmod p$ ist oder $n = -1$ quadratischer Nichtrest ist. Dieser Satz gilt jedoch nur für ungerade Primzahlen und berücksichtigt daher nicht die 2.

Tabelle 14 Bestimmung der Faktorbasis mit Hilfe der Restberechnung.

B-glatt								
p (Modulo)	-	3	5	7	11	13	17	19
$n \bmod p$	-	1	3	2	4	9	11	2
$n^{(p-1)/2} \bmod p$	-	1	$4 \equiv -1$	9	4	9	$16 \equiv -1$	$18 \equiv -1$

Bei der Durchführung des quadratischen Sieb-Algorithmus, ausgehend von einem bestimmten Wert x , soll auch eine Möglichkeit existieren rückwärts zu gehen. Daher wird zur Faktorbasis der Wert (-1) ergänzt. Daraus resultiert die Faktorbasis

$$F_B = \{ p \leq B \mid p \in \mathbb{p} \text{ und } n \text{ quadratischer Rest modulo } p \} \cup \{ -1 \} \text{ mit} \\ = \{ -1, 2, 3, 7, 11, 13 \}.$$

Damit haben die Exponentenvektoren die Länge 6. In der Hoffnung, dass genügend Werte $q(x)$ mit $q(x) = x^2 - n$ glatt sein werden und linear abhängige Exponentenvektoren haben, verwenden man das Siebintervall $S = \{ x \in \mathbb{N} \mid 150 - 7 \leq m \leq 150 + 7 \} = \{ 143, 144, 145, 146, 147, \dots, 157 \}$. Es ist üblich

die Werte für x im Bereich $m = \sqrt{n}$ zu wählen, was in diesem Beispiel den Wert $x = 150$ entspricht.

Tabelle 15 Anwenden des Siebintervalls S und Untersuchung der Faktorbasis.

x	$q(x) = x^2 - 22.213$	Primfaktoren von F_B	
143	-1.764	$(-1) * 2^2 * 3^2 * 7^2$	glatt
144	-1.477	$(-1) * 7 * 211$	$211 \notin F_B$
145	-1.188	$(-1) * 2^2 * 3^3 * 11$	glatt
146	-897	$(-1) * 3 * 13 * 23$	$23 \notin F_B$
147	-604	$(-1) * 2^2 * 151$	$211 \notin F_B$
148	-309	$(-1) * 3 * 103$	$103 \notin F_B$
149	-12	$(-1) * 2^2 * 3$	glatt
150	287	$7 * 41$	$41 \notin F_B$
151	588	$2^2 * 3 * 7^2$	glatt
152	891	$3^4 * 11$	glatt
153	1.196	$2^2 * 13 * 23$	$23 \notin F_B$
154	1.503	$3^2 * 167$	$167 \notin F_B$
155	1.812	$2^2 * 3 * 151$	$151 \notin F_B$
156	2.123	$11 * 193$	$193 \notin F_B$
157	2.436	$2^2 * 3 * 7 * 29$	$29 \notin F_B$

Es verbleiben nach dieser Berechnung nur noch die folgenden x –Werte mit ihren Faktoren als glatte Zahlen. So ergeben sich die binären Exponentenvektoren (1 = ungerade, 0 = gerade).

Tabelle 16 Suche nach einer Linearkombination, die den Nullvektor ergibt aus den zuvor gefundenen glatten Primfaktoren (Gerader oder ungerader Exponent).

Primfaktorzerlegung mod 2						
x/p	-1	2	3	7	11	13
143	1	0	0	0	0	0
145	1	0	1	0	1	0

A14 Vergleich von Angriffen mit dem Ziel der Faktorisierung

149	1	0	1	0	0	0
151	0	0	1	0	0	0
152	0	0	0	0	1	0
145 +	0	0	0	0	0	0
149 +						
152 =						

Gesucht ist eine Linearkombination der Zeilen, die den Nullvektor ergeben.

Eine Möglichkeit ist die Kombination aus $x_1 = 145$, $x_2 = 149$, $x_3 = 152$.

Für die Faktoren x , y ergeben sich somit

$$\begin{aligned}
 x &= x_1 * x_2 * x_3 = 145 * 149 * 152 \equiv 18.649 \pmod{22.213}, \\
 y &= \sqrt{((-1) * 2^2 * 3^3 * 11 * (-1) * 2^2 * 3 * 3^4 * 11)} \\
 &= \sqrt{((-1)^2 * 2^4 * 3^8 * 11^2)} \\
 &= 2^2 * 3^4 * 11 \\
 &= 3.564 \\
 &\equiv 3.564 \pmod{22.213}.
 \end{aligned}$$

Kontrolle auf die Bedingungen

$$\begin{aligned}
 x^2 &\equiv y^2 \pmod{N} &\rightarrow & 18.473 \equiv 18.473 \pmod{22.213} \\
 x &\not\equiv \pm y \pmod{N} &\rightarrow & 18.649 \not\equiv \pm 3.564 \pmod{22.213},
 \end{aligned}$$

jedoch ist

$$\begin{aligned}
 &ggT(x + y, N), \\
 &ggT(18.649 + 3.564, 22.213) = ggT(22.213, 22.213)
 \end{aligned}$$

verletzt.

Problem an dem vorgestellten QS-Algorithmus ist, dass mit steigendem x auch $q(x)$ wachsen wird. Dies führt zur Schwierigkeit ein glattes $q(x)$ über eine zunehmend größer werdende Faktorbasis zu finden. An diesem Punkt setzt das Multiple Polynomial Quadratic Sieve (MPQS) an. Wie der Name vermuten lässt, verwendet dieses Siebverfahren mehrere Polynome mit der Absicht sowohl die Faktorbasis, das Siebintervall und $q(x)$ kleiner zu halten [23].

Effizienter als das Wiederholen des Vorgangs mit einer Erweiterung des Siebintervalls und Vergrößerung der Faktorbasis bietet sich das Sieben mit mehrfachen Polynomen an. Das Multiple Polynomial Quadratic Sieve MPQS wurde erstmalig von Peter Montgomery vorgeschlagen. Polynome haben hier die Form

$$q(x) = ax^2 + 2bx + c \quad \text{mit}$$

$$b^2 - ac = N,$$

$0 \leq b < a$, und a eine Quadratzahl ist.

$$b^2 \equiv n \pmod{a}, \text{ nur dann, wenn}$$

n eine Quadratzahl \pmod{q} für jede Primzahl $q|a$.

Ziel ist es das Siebintervall klein zu halten damit die Werte für $q(x)$ kleiner sind sowie möglichst viele Faktoren innerhalb der kleinen Faktorbasis bleiben. Für die Koeffizienten müssen bestimmte Auswahlen getroffen werden [24]. Es folgt

$$q(x) = ax^2 + 2bx + c \text{ mit } a, b, c \in \mathbb{Z}$$

$$a * q(x) = (ax)^2 + 2abx + ac$$

$$a * q(x) = (ax)^2 + 2abx + b^2 - N$$

$$a * q(x) = (ax + b)^2 - N$$

$$\rightarrow a * q(x) \equiv (ax + b)^2 \pmod{N} \text{ [23].}$$

Alle Werte m, F_B aus dem vorhergehenden Ansatz werden übernommen. Die Werte

$$a = 4, \quad (2^2)$$

$$b^2 \equiv 22.213 \pmod{4}$$

$$\equiv 1 \pmod{4}$$

$$b = \pm 1 \pmod{4}. \quad (0 \leq b < a)$$

Damit die Funktionswerte für $q_g(x)$ nicht zu groß werden, werden die Parameter entsprechend gesucht.

$$(ax + b)^2 - N \text{ mit } x \text{ -Werten nahe } \sqrt{n},$$

$$(4 * 150 + b)^2 \approx N$$

$$b \approx \sqrt{N} - 600 \approx -450 \rightarrow b = -451, \text{ gibt das Polynom}$$

$$q_g(x) = (4x - 451)^2 - 22.213.$$

Als Siebintervall dient Siebintervall $S = \{x \in \mathbb{N} | 150 - 10 \leq m \leq 150 + 10\} = \{140, 141, 142, 143, 144, 145, 146, 147, \dots, 157, 158, 159, 160\}$. Die Berechnung der Werte erfolgt analog zum Schritt vorher. Folgende Tabelle ist auf die glatten Zahlen gekürzt.

Tabelle 17 Gekürzte Tabelle der Untersuchung der Primfaktoren des gewählten Polynoms. Dargestellt sind lediglich die glatten Zahlen sowie Zahlen, die zur Faktorbasis hinzugenommen werden sollen.

x	$ax + b$	$q_g(x)$	Primfaktoren von F_B
140	109	-10.332	$(-1) * 2^2 * 3^2 * 7 * 41$ $F_B \cup \{41\}$
146	133	-5.572	$(-1) * 2^2 * 3 * 13 * 29$ $F_B \cup \{29\}$
147	137	-3.444	$(-1) * 2^2 * 3 * 7 * 41$ $F_B \cup \{41\}$
149	145	-1.188	$(-1) * 2^2 * 3^3 * 11$
150	149	-12	$(-1) * 2^2 * 3$
151	153	1.196	$2^2 * 13 * 23$
152	157	2.436	$2^2 * 3 * 7 * 29$
155	169	6.348	$2^2 * 3 * 23^2$ $F_B \cup \{23\}$
159	185	12.012	$2^2 * 3 * 7 * 11 * 13$ $F_B \cup \{29\}$

Für die Zerlegung in die Primfaktoren hat sich eine Erweiterung der Faktorbasis F_B auf

$$F = F_B \cup \{23, 29, 41\}$$

angeboten, so dass nun auch diese Werte berücksichtigt werden. Betrachtet man für die Suche nach Linearkombinationen nur wieder die Primfaktorzerlegung $\text{mod } 2$ ergibt sich die Liste

Tabelle 18 Suche nach einer Linearkombination, die den Nullvektor ergibt.

x	$ax + b$	-1	2	3	7	11	13	23	29	41
140	109	1	0	0	1	0	0	0	0	1
146	133	1	0	1	0	0	1	0	1	0
147	137	1	0	1	1	0	0	0	0	1
149	145	1	0	1	0	1	0	0	0	0
150	149	1	0	1	0	0	0	0	0	0
151	153	0	0	0	0	0	1	1	0	0
152	157	0	0	1	1	0	0	0	1	0
155	169	0	0	1	0	0	0	0	0	0
159	185	0	0	1	1	1	1	0	0	0

Gesucht ist eine Linearkombination der Zeilen, die den Nullvektor ergeben. Eine Möglichkeit ist die Kombination aus $x_1 = 140$, $x_2 = 147$, $x_3 = 155$. Für die Faktoren x, y ergeben sich somit

$$\begin{aligned}
 x &= (ax_1 + b) * (ax_2 + b) * (ax_3 + b) = 109 * 137 * 169 \\
 &\equiv 13.608 \text{ mod } 22.213, \\
 y &= \sqrt{((-1) * 2^2 * 3^2 * 7 * 41 * (-1) * 2^2 * 3 * 7 * 41 * 2^2 * 3 * 23^2)} \\
 &= \sqrt{((-1)^2 * 2^6 * 3^4 * 7^2 * 23^2 * 41^2)} \\
 &= \sqrt{(2^3 * 3^2 * 7 * 23 * 41)^2} \\
 &= 2^3 * 3^2 * 7 * 23 * 41 \\
 &\equiv 8.799 \text{ mod } 22.213.
 \end{aligned}$$

Abschließend erfolgt eine Kontrolle auf die Bedingungen

$$\begin{aligned}
 x^2 &\equiv y^2 \text{ mod } n && \rightarrow 10.096 \equiv 10.096 \text{ mod } 22.213 \\
 x &\not\equiv \pm y \text{ mod } n && \rightarrow 13.608 \not\equiv \pm 8.799 \text{ mod } 22.213, \\
 \text{ggT}(22.407, 22.213) &&& \rightarrow 97,
 \end{aligned}$$

$$ggT(4.809, 22.213) \rightarrow 229.$$

Somit ist die Zahl $n = 22.213$ mit $22.213 = 229 * 97$ erfolgreich faktorisiert.

Das quadratische Sieb ist eines der schnellsten Verfahren für die Faktorisierung großer natürlicher Zahlen und so ein Maßstab für die Sicherheit des RSA verfahrens. Wählt man B von der Größenordnung

$$e^{\sqrt{\frac{1}{2} \log_2 n \log_2 \log_2 n}}$$

und S von etwa der gleichen Größenordnung, so ergibt sich die Laufzeit

$$O(n) = e^{\sqrt{\log_2 n \log_2 \log_2 n}}.$$

Dies ist subexponentiell jedoch immer noch superpolynomiell [25].

Tabelle 19 Beispiele für die verschiedenen Größenordnungen von Laufzeiten [26].

Fakul- tät	>	Exponenti- ell	>	Polynomi- ell	>	Logarith- misch
$x!$		e^x		$x\sqrt{(1+x^2)}$		$\ln(\ln(x))$
		$\cosh x$		$x^2 + 1$		$(\ln(x))^3$
		$3^{\sqrt{x}}$		\sqrt{x}		$\ln(x)$
		2^x				

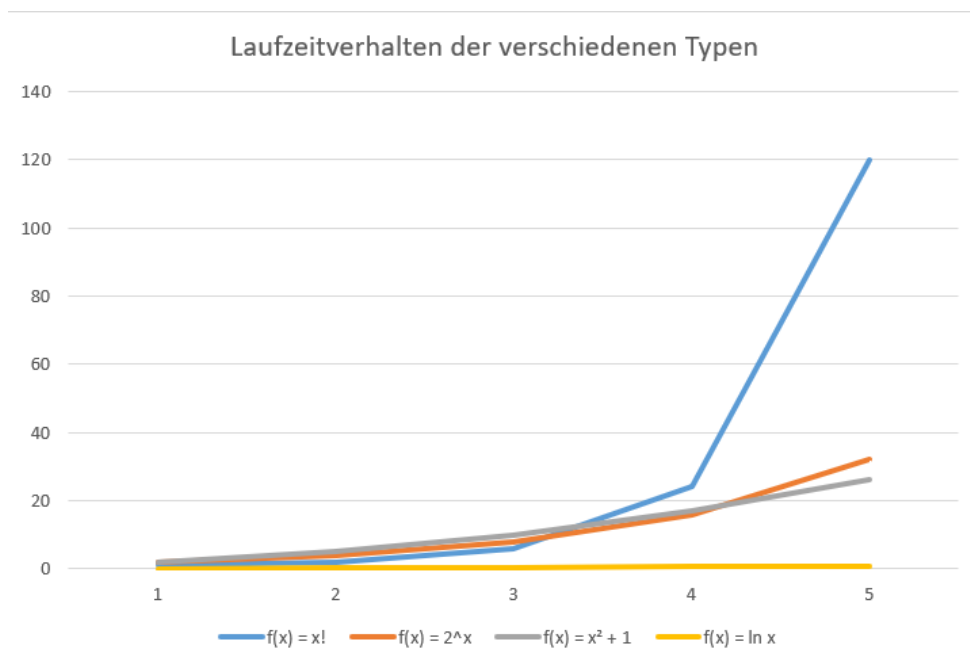


Bild 12 Darstellung des Aufwands für die ersten fünf Elemente aus x.

Auch das quadratische Sieb wird noch einmal durch das Zahlkörpersieb übertroffen. Jedoch sind auch hier ähnliche Grenzen gesetzt, so dass nach heutigem Wissensstand die Faktorisierung großer Zahlen, wie sie derzeit in RSA eingesetzt werden, nicht akut bedroht sind.

Tabelle 20 Stand der RSA Faktorisierungsrekorde 2008 [31, p. 89].

Zahl	Bits	Datum	Gruppe	Algorithmus
RSA-100	332	04. 1991	Lenstra	Quadr. Sieb
RSA-110	365	04. 1992	Lenstra	Quadr. Sieb
RSA-120	399	06. 1993	Denny	Quadr. Sieb
RSA-129	429	04. 1994	Lenstra	Quadr. Sieb
RSA-130	432	04. 1996	Lenstra	Zahlkörpersieb
RSA-140	465	02. 1999	te Riele	Zahlkörpersieb
RSA-155	512	08. 1999	te Riele	Zahlkörpersieb
RSA-160	532	04. 2003	Franke	Zahlkörpersieb
RSA-576	576	12. 2003	Franke	Zahlkörpersieb
RSA-640	640	11. 2005	Franke	Zahlkörpersieb
RSA-1024	1024	offen		

A15 Hybride Verschlüsselungsverfahren

Symmetrische und asymmetrische Verschlüsselung verfolgen dasselbe Ziel. Beide verschlüsseln Informationen, die zwischen Alice und Bob kommuniziert werden sollen. Obwohl sie ähnliche Zwecke verfolgen gehen sie jedoch unterschiedlich vor und nutzen dabei verschiedene mathematische Eigenschaften aus.

Symmetrische Verfahren sind relativ einfach. Ihre Sicherheit liegt nicht im Verfahren selbst, sondern im Schlüssel. Asymmetrische Verfahren sind in ihrer Berechnung komplexer und es gibt bessere Wege die Verschlüsselung zu brechen als die vollständige Schlüsselsuche. Daher kommen, im Vergleich zu symmetrischer Verschlüsselung, große Schlüssel zum Einsatz. Zwar lösen asymmetrische Verfahren das Problem des Schlüsselaustauschs, sie kosten jedoch auch wesentlich mehr Rechenzeit.

Vergleicht man die Verfahren RSA und AES, so ist das RSA Verfahren ungefähr um den Faktor 1.000 langsamer [5, p. 172].

Ein Vorteil von RSA ist jedoch, dass die Parameter, die Module und Schlüssellängen, frei in der Größe wählbar sind und sie, gemessen am technischen Stand, immer angepasst werden können in dem z.B. die Schlüsselgröße verdoppelt wird. Wird kein neues mathematisches Verfahren für das Problem des diskreten Logarithmus oder die Faktorisierung großer Zahlen gefunden, so kann jeder Zeit der Schlüssel vergrößert werden, sollten die technischen Mittel Fortschritte machen.

Die großen Geschwindigkeitsnachteile von asymmetrischen Verfahren verhindern den Einsatz bei großen Datenmengen. Kleine Daten, idealerweise Zahlen, wie sie bei Schlüsseln vorkommen, können jedoch ohne große Auswirkung ver- und entschlüsselt werden.

Kombiniert man symmetrische und asymmetrische Verfahren, so erhält man ein hybrides Kryptoschema, bei dem Alice einen symmetrischen Schlüssel für

eine Kommunikation mit Bob wählt, den sogenannten Session-Key, auch Einmalschlüssel. Der Session-Key wird asymmetrisch, mit Hilfe des öffentlichen Schlüssels von Bob, übertragen. Anschließend kann eine symmetrisch verschlüsselte Kommunikation zwischen Alice und Bob stattfinden.

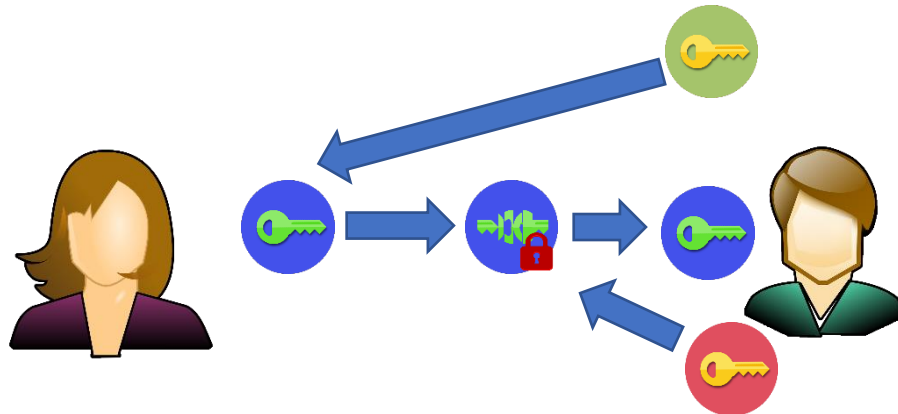


Bild 13 Alice generiert einen Session-Key (Blau) und verschlüsselt diesen mit Bobs öffentlichem Schlüssel (Grün), so kann nur Bob diesen Session-Key mit seinem privaten Schlüssel (Rot) wieder entschlüsseln.

Die Implementierung solcher Systeme hat, neben dem Vorteil der gelösten Schlüsselverteilung, auch Nachteile. Beide Verfahren müssen implementiert werden. Die Gesamtsicherheit des Verfahrens hängt nun von mehr Annahmen ab, nämlich von allen Annahmen, auf denen das symmetrische und das asymmetrische Verfahren fußen [27, p. 175].

Ein asymmetrisches Kryptoschema beschreibt sich mit

$$S_{asym.} = (m, K(k_{public}, k_{privat}), g, e, d)_{asym.}$$

m Klartext

K Menge der Schlüssel der asymmetrischen Verfahren

g Algorithmus der Schlüsselgenerierung (k_{public}, k_{privat})

e Verschlüsselungsalgorithmus (x, k_{public})

d Entschlüsselungsalgorithmus ($e(x), k_{privat}$)

Ein symmetrisches Kryptoschema beschreibt sich mit

$$S_{sym.} = (m, k, e, d)_{sym.}$$

m	Klartext
k	Schlüssel $k_{sym.}$
e	Verschlüsselungsalgorithmus $(x, k_{sym.})$
d	Entschlüsselungsalgorithmus $(e(x), k_{sym.})$

Ein hybride Kryptoschema beschreibt sich mit

$H(S_{asym.}, S_{sym.})$

$= (m, K, g, e_{hybrid}(m, e_{asym.}(k_{sym.}, k_{public})), d_{hybrid}(c, d_{asym.}(k_{cipher}, k_{privat})))$

m Klartext

K Menge der Schlüssel bei asymmetrischen Verfahren

g Algorithmus der Schlüsselgenerierung (k_{public}, k_{privat})

e_{hybrid} Verschlüsselungsalgorithmus

d_{hybrid} Entschlüsselungsalgorithmus

c Ciphertext mit $c = e_{hybrid}(m, e_{asym.}(k_{sym.}, k_{public}))$

k_{cipher} Verschlüsselter Session-Key mit $e_{asym.}(k_{sym.}, k_{public})$, [27, p.

176]

Je nach Implementierung kann ein Sender den symmetrischen Session-Key beibehalten oder stetig neu generieren. Ein Empfänger sollte das nicht tun. Es lässt sich für den Empfänger Bob nicht feststellen ob ein Absender auch wirklich Alice ist und nicht z.B. Mallory. Jeder hat Zugriff auf den öffentlichen Schlüssel von Bob, so dass ein Man-in-the-middle-Angriff bei nicht Austausch des Schlüssels erfolgreich durchgeführt werden könnte. Bob ist gezwungen für eine Antwort die Prozedur zu wiederholen.

Um dieses Problem zu lösen werden Signaturverfahren, abgeleitet aus asymmetrischen Kryptoverfahren, verwendet.

A16 Signaturverfahren

Digitale Signaturverfahren dienen der Überprüfung von Absendern. Hierbei handelt es sich in der Regel um schlüsselabhängige Prüfsummen, die fälschungssicher sind, auf Echtheit überprüft werden können, die nicht in einer anderen Kommunikation betreffend einer anderen Nachricht oder Dokument wiederverwendet werden können und sie verhindern, dass eine zugehörige Nachricht unbemerkt verändert wird.

Signaturverfahren bedienen sich dabei demselben Prinzip wie asymmetrische Verschlüsselungsverfahren. Während für die Verschlüsselung ein öffentlicher Schlüssel genommen wird, so dass nur ein geheimer privater Schlüssel die Nachricht entschlüsselt, ist es jedoch hier exakt anders herum.

Verwendet man einen privaten Schlüssel zum Verschlüsseln, so kann nur der passende öffentliche Schlüssel dieses rückgängig machen.

Alice verwendet für die Verschlüsselung einer Nachricht m ihren privaten Schlüssel k_{privat} . Die Unterschrift bzw. digitale Signatur wird als Funktion u mit der resultierenden Signatur s mit

$$s = u(m, k_{privat})$$

betrachtet. Um die Signatur s zu verifizieren verwendet der Empfänger Bob Funktion v sowie Alice öffentlichen Schlüssel k_{public} und errechnet damit den ursprünglichen Wert der Nachricht m mit

$$m' = v(s, k_{public}).$$

Stimmen die Nachrichten m und m' überein, so kann die empfangene Nachricht nur von Alice stammen, vorausgesetzt ihr Schlüssel wurde nicht korrumpiert.

Die hier beschriebene digitale Signatur basiert auf dem zuvor vorgestellten RSA Verfahren unter Austausch der Schlüssel, d.h. es kommen die voneinander abhängigen Schlüssel k_{public} und k_{privat} zum Einsatz die zum Ver- und Entschlüsseln konträr eingesetzt werden müssen. Es gilt

$$k_{private} = e$$

$$k_{public} = d$$

$$(m^d)^e \equiv (m^e)^d \equiv m \text{ mod } n$$

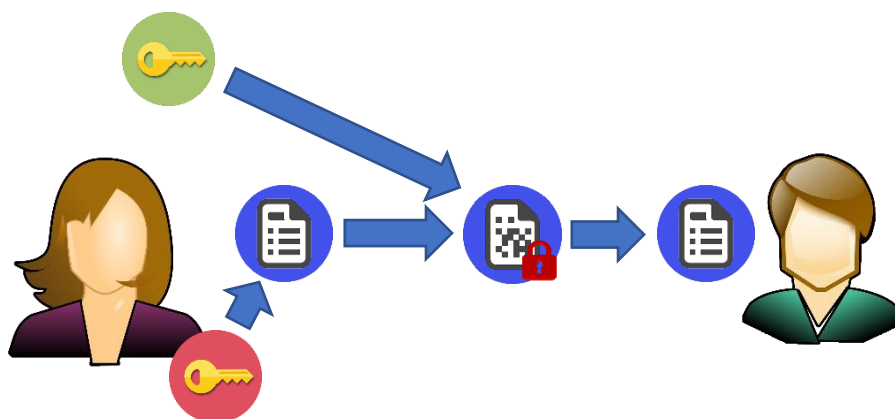


Bild 14 Alice verschlüsselt z.B. mit einem File zusammen dessen Hashwert. Sie verwendet dafür ihren privaten Schlüssel. Die Entschlüsselung kann nur mit ihrem öffentlichen Schlüssel erfolgen, so dass Bob sich sicher sein kann, dass die Information von Alice stammt.

Genau wie bei der Verschlüsselung durch RSA sind bei der Signatur mit RSA die vollständige Schlüsselsuche und der Versuch der Faktorisierung Möglichkeiten von Angriffen. Beide sind jedoch bei der zuvor vorgestellten hohen Laufzeit bei großen Schlüsseln aussichtslos. Die Signatur s der Nachricht m erfolgt, analog zur Verschlüsselung mit

$$s = m^d \text{ mod } n,$$

wobei für die Wahl der Schlüssel d, e sowie des Moduls n dieselben Regeln gelten wie zuvor in der Verschlüsselung mittels RSA erläutert. Die Verifikation der Signatur zur Nachricht m erfolgt analog mit

$$m \equiv s^e \pmod{n}.$$

Liegt hier eine Kongruenz vor, so ist die Signatur gültig und die Nachricht verifiziert.

Neben dem Signaturverfahren, basierend auf dem Problem der Faktorisierung großer Zahlen, gibt es auch, wie zuvor im Diffie-Hellman-Schlüsselaustausch, Signaturverfahren basierend auf dem diskreten Logarithmus, sogenannte DLSSs (Discrete Logarithm Signature Systems). Zwei bekannte Beispiele sind das ElGamal-Verfahren sowie der Digital Signature Algorithm DSA.

Bei dieser Art Verfahren sind Ver- und Entschlüsselung bzw. das Signieren nicht vertauschbar. Die Schlüsselerzeugung für die Signatur erfolgt analog zum ElGamal-Verschlüsselungsverfahren. Alice wählt eine große Primzahl p , für die $(p - 1)/2$ auch eine Primzahl ist, sowie eine Zahl g , die ein Generator von Z_p^* ist. Darüber hinaus benötigt Alice einen geheimen Schlüssel $k_{private} = d$ mit $d < p$ aus dem sie ihren öffentlichen Schlüssel k_{public} mittels

$$e = g^d \pmod{p},$$

(p, g, e) öffentlicher Schlüssel k_{public}

d privater Schlüssel $k_{private}$

berechnet. Für die Signatur einer Nachricht m (das kann auch der Hashwert $h(m)$ sein) wählt Alice eine zufällige Zahl k mit

$$k \in \{1, 2, 3, \dots, p - 2\},$$

$ggT(k, p - 1) = 1$ und berechnet

$$r = g^k \pmod{p}.$$

Anschließend stellt sie folgende Gleichungen auf

$$m = d * r + b * s \text{ mod}(p - 1),$$
$$s = k^{-1}(m - d * r) \text{ mod}(p - 1),$$

wobei k^{-1} das Inverse von $k \text{ mod}(p - 1)$ ist und s eine in Abhängigkeit von m, a, k, g stehende Variable. Das Zahlenpaar (r, s) bilden zur Nachricht m die digitale Signatur.

Bei der Verifikation durch Bob sind ihm der öffentliche Schlüssel von Alice (p, g, e) sowie die Signatur (r, s) zur Nachricht m bekannt. So kann Bob überprüfen ob die Kongruenz

$$e^r * r^s \equiv g^m \text{ mod } p$$

erfüllt ist.

Beispiel

Alice wählt die Zahlen $p = 23, g = 7, d = 6$ und berechnet ihren öffentlichen Schlüssel

$$e = g^d \text{ mod } p$$

$$e = 7^6 \text{ mod } 23$$

→ $k_{\text{public}} (p = 23, g = 7, e = 4)$, privater Schlüssel $k_{\text{private}} d = 6$.

Sie will die Nachricht $m = 7$ signieren und wählt $k = 5$. Sie berechnet:

$$r = g^k \text{ mod } p$$

$$= 7^5 \text{ mod } 23$$

$$r = 17$$

$$k^{-1} = 9 \text{ mod } (p - 1)$$

$$s = k^{-1} (m - d * r) \text{ mod } (p - 1)$$

$$= 9 (7 - 6 * 17) \text{ mod } (22)$$

$$s = 3$$

Signatur (17, 3).

Für die Verifikation mit der Nachricht m mit der Signatur s und dem öffentlichen Schlüssel (p, g, e) von Alice berechnet Bob

$$e^r * r^s \equiv g^m \pmod{p}$$

$$41^7 * 17^3 \equiv 7^7 \pmod{23}$$

$$5 \equiv 5 \pmod{23},$$

was erfüllt ist. Somit ist die Nachricht verifiziert.

Wer den diskreten Logarithmus berechnen kann ist in der Lage den geheimen Schlüssel d von Alice zu ermitteln und damit Signaturen zu fälschen. Dies ist die einzige bekannte allgemeine Methode ElGamal-Signaturen zu erzeugen. Der Wert p muss derart groß wählen, um eine Berechnung des diskreten Logarithmus unmöglich zu machen. Es handelt sich um eine Einwegfunktion. Es sollte vermieden werden, dass $p \equiv 3 \pmod{4}$, die Primitivwurzel g ein Teiler von $p - 1$ und die Berechnung des diskreten Logarithmus in der Untergruppe von $(\mathbb{Z}/p\mathbb{Z})^*$ der Ordnung g möglich ist [18, pp. 257-259] [17]. Das geht genau dann, wenn g nicht groß ist. Ferner ist die Bedingung $p \equiv 3 \pmod{4}$ häufig dann erfüllt, wenn die Primzahl p so gewählt wird, dass

$$p - 1 = q * 2$$

$$q \in \mathbb{P}, \text{ ungerade.}$$

Beispiel

$$p = 23,$$

$$p - 1 = 22,$$

$$q = 11 \rightarrow$$

$$p \equiv 3 \pmod{4}$$

$$23 \equiv 3 \pmod{4}$$

$$3 \equiv 3 \pmod{4},$$

daher wählt man die Primitivwurzel g so, dass diese $(p - 1)$ nicht teilt [18]. Auch die zufällige Zahl k sollte bei jedem Signaturverfahren neu gewählt werden, da anderweitig die Signaturen s_1 sowie s_2 für die Nachrichten m_1 und m_2 denselben Wert $r = g^k \bmod p$ haben,

$$s_1 - s_2 \equiv k^{-1}(m_1 - m_2) \bmod (p - 1).$$

Ist $(m_1 - m_2)$ invertierbar $\bmod (p - 1)$, so lässt sich die Zahl k bestimmen. Aus k, s_1, r, m_1 lässt sich dann der geheime Schlüssel $k_{private} = d$ bestimmen.

$$s_1 = k^{-1} (m_1 - d * r) \bmod (p - 1)$$

$$d \equiv r^{-1}(m_1 - k * s_1) \bmod (p - 1)$$

Zur Vereinfachung wurde die Nachricht m nicht näher betrachtet. Um existentielle Fälschungen zu vermeiden ist es jedoch erforderlich, dass nicht die eigentliche Nachricht, sondern ein Hash verwendet wird. Ein Hashwert ist die Abbildung einer großen Eingabemenge auf eine kleinere Menge, auf die hier jedoch nicht eingegangen werden soll. Eine Nachricht wird mit

$$e^r * r^s \equiv g^m \bmod p$$

verifiziert. Man wählt nun u, v mit $ggT(v, p - 1) = 1$ und setzt

$$r = g^u * e^v \bmod p,$$

$$s = -r * v^{-1} \bmod (p - 1),$$

$$m = s * u \bmod (p - 1).$$

So gilt die Kongruenz

$$e^r * r^s \equiv e^r * g^{su} * e^{sv} \equiv e^r * g^{su} * e^{-r} \equiv g^m \bmod p.$$

Beispiel

Es werden die Werte aus dem vorherigen Beispiel verwendet. Die Signatur $(r, s) = (17, 3)$ wurde zur Nachricht $m = 7$ generiert. Der öffentliche Schlüssel $K_{public}(p = 23, g = 7, e = 4)$ ist bekannt. Sei $m' = 3$ nun eine weitere Nachricht, die signiert werden soll.

$$\begin{aligned} u &= m' * m^{-1} \text{ mod } (p - 1) \\ &= 3 * (7)^{-1} \text{ mod } 22 \\ &= 3 * 19 \text{ mod } 22 \\ &= 13 \text{ mod } 22. \end{aligned}$$

$$\begin{aligned} s' &= s * u \text{ mod } (p - 1) \\ &= 3 * 13 \text{ mod } 22 \\ &= 17 \text{ mod } 22. \end{aligned}$$

$$\begin{aligned} r' &\equiv r * u \text{ mod } (p - 1), \\ & \\ r' &\equiv r \text{ mod } p \\ &\equiv 17 * 13 \text{ mod } 22 \\ &\equiv 1 \text{ mod } 22, \end{aligned}$$

d. h.

$$\begin{aligned} r' &\equiv 1 \text{ mod } 22, \\ r' &\equiv 17 \text{ mod } 23. \end{aligned}$$

Mit Hilfe des chinesischen Restsatzes wird nun $r' = 155 \text{ mod } 506$ berechnet. Somit lautet die Signatur dann für die neue Nachricht

$$\begin{aligned} m' &= 3 \\ (r', s') &= (155, 17). \end{aligned}$$

Verifikation dieser Signatur mit dem öffentlichen Schlüssel von Alice:

$$e^{r'} * r'^{s'} \equiv g^{m'} \text{ mod } p$$

$$4^{155} * 155^{17} \equiv 7^3 \text{ mod } 23$$

$$21 \equiv 21 \pmod{23},$$

was erfüllt ist. Somit ist die Nachricht m' mit Hilfe einer bekannten Signatur (r, s) ohne Kenntnis des privaten Schlüssels $k_{private} = d$ gültig signiert worden. Durch die Verwendung einer kollisionsresistenten Hashfunktion kann das verhindert werden [18, p. 259]. Vielfach wird deshalb die Signatur nicht aus der Nachricht und dem privaten Schlüssel gebildet, sondern es wird der Hash-Wert der Nachricht (z.B. mit SHA-1) für die Berechnung der Signatur herangezogen. Hierbei muss die Hash-Funktion kollisionsresistent sein, d. h., es muss praktisch unmöglich sein, zwei unterschiedliche Nachrichten zu finden, deren Hash-Wert identisch ist [21].

A17 Das Doppelspaltexperiment

Die Physik ging davon aus, dass sich die Welt aus Teilchen zusammensetzt, wie z.B. Licht aus Photonen besteht. Das wurde 1899 durch den Physiker Philip Lenard nachgewiesen. Jedoch konnte man durch Beobachten der Interferenz des Lichtes hinter einem Doppelspalt sehen, dass sich diese Licht-Teilchen auch wie Wellen verhalten. Fallen Licht-Teilchen auf einen doppelten Spalt, so würde man dahinter die Abbildung beider Spalten erwarten.

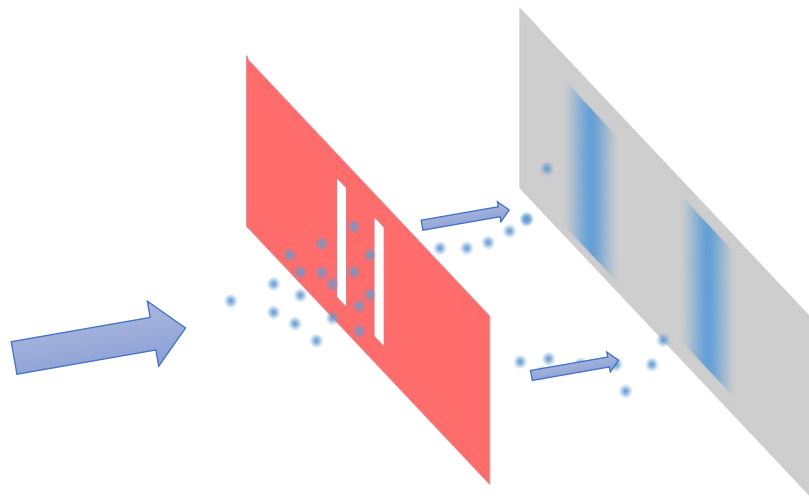


Bild 15 Doppelspaltexperiment, bei dem Teilchen (Photonen) aus einer Quelle durch einen Doppelspalt geschossen werden und ein Muster auf der Projektionsfläche dahinter abbilden.

Führt man das Experiment jedoch durch ergibt sich ein Muster, bei dem die Intensität der Projektion in der Mitte am stärksten ist und zu den Seiten hin abnimmt.

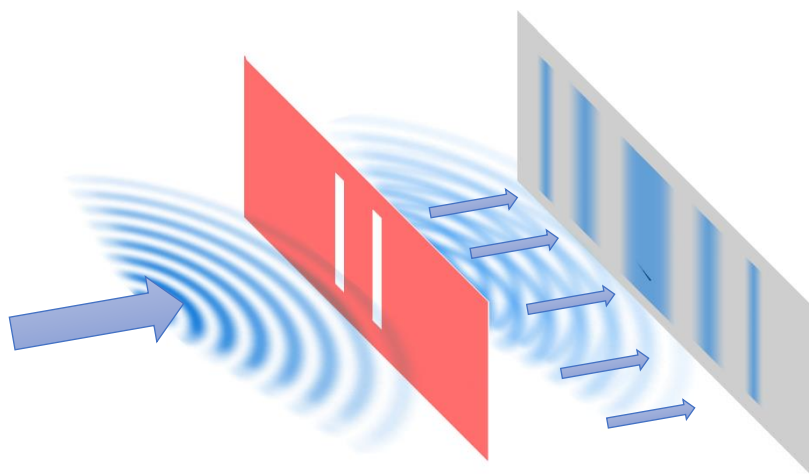


Bild 16 Nach der Passage des Doppelspalts bildet sich ein Interferenzmuster, welches an Wellen erinnert und nicht an Teilchen.

Verhält sich das Licht wie eine Welle, so können sich diese Wellen überlagern. Treffen zwei Wellenberge aufeinander verstärken sie sich (positive Interferenz). Trifft ein Wellenberg auf ein Wellental, so neutralisieren sie sich. Das ist im Ergebnis des Experimentes zu sehen. Auch bei Elektronen ist das festzustellen. Elektronen kann man eine Masse und eine Ladung zuordnen. Daher sollten es Teilchen sein und es ist die erste Abbildung zu erwarten. Dennoch zeigen auch Elektronen das Interferenzmuster der zweiten Abbildung. Man spricht hier von der Welle-Teilchen-Dualität in der Quantenmechanik. Manchmal verhalten sie sich wie ein Teilchen, manchmal wie eine Welle. Bei einer Messung werden jedoch stets Teilchen gemessen. Die höchsten Werte werden da gemessen, wo nach der Wellentheorie (Wellenfunktion Ψ) die höchste Intensitäten zu erwarten sind [28]. Das passiert auch dann, wenn man die Energie so weit reduziert, dass sich einzelne Teilchen im System befinden. Nach und nach baut sich das Interferenzmuster aus Abbildung 16 auf.

Das Muster zeigt die Wahrscheinlichkeit wo dieses Teilchen auftritt. Dabei geht die „Kopenhagener Interpretation“ davon aus, dass das Teilchen erst zum Zeitpunkt des Auftreffens entsteht [28, p. 21]. Die Wahrscheinlichkeit des Ortes des Auftreffens breitet sich wie eine Welle aus [29, p. 38]. Es ist nicht möglich präzise zu sagen wo das Teilchen auf die Platte treffen wird. Innerhalb der

Wahrscheinlichkeiten ist es überall möglich. Erst im Moment der Messung, d.h. beim Auftreffen auf die Platte, ist die Position eindeutig bestimmt. Die Wellenfunktion kollabiert und es ist genau eine Position übrig. Das Teilchen geht sowohl durch Spalt eins als auch durch Spalt zwei, ein Vorgang, der in der klassischen Physik nicht möglich ist, da es sich um ein Elementarteilchen handelt. Hier jedoch befindet sich das Teilchen in Superposition, was in der Quantenphysik durchaus normal ist (Vgl. *Schrödingers Katze*).

Eine Messung zerstört eine solche Superposition und weist einen eindeutigen Wert zu. Der Begriff Messung ist gleichzusetzen mit der Entnahme einer Information aus einem System in einer beliebigen Form. Wird ein Detektor an z.B. dem linken Spalt des Doppelspaltexperimentes angebracht, um zu registrieren ob das Teilchen durch diesen gelange, wird eine Messung vorgenommen. Diese Messung am Spalt zerstört die Superposition und beeinflusst das Interferenzmuster. Das geschieht auch dann, wenn der Detektor am linken Spalt das Teilchen gar nicht misst, da es durch den rechten Spalt ging. Auch in diesem Fall wird die Superposition zerstört. Das Teilchen muss nicht mal in der Nähe des Detektors sein. Die Nicht-Messung des Teilchens am linken Spalt bedeutet, dass es durch den rechten Spalt ging, was der Entnahme einer Information aus einem System gleichkommt und somit wieder einer Messung ist. Dieses Problem wird als Dekohärenz-Problem bezeichnet und stellt ein Problem beim Bau von Quantencomputern da. Es bedeutet, dass jedes Abfließen von Informationen die Superpositionen des Systems zerstören würde. Die Kohärenzzeit bezeichnet dabei die Zeit, die ein Quantencomputer für seine Berechnungen hat, bevor die Superpositionen zerstört werden [29].

A18 Quanten NOT-Gatter

Das NOT-Gatter ist ein Ein-Qubit-Gatter und verhält sich analog zum klassischen NOT-Gate, indem es den Zustand eines Qubits invertiert,

$$N|0\rangle = |1\rangle,$$

$$N|1\rangle = |0\rangle.$$

Die Invertierung eines Zustandes kann z.B. durch einen abgestimmten Radioimpuls bezüglich des Spins eines Teilchens erreicht werden. Das Schaltsymbol für ein NOT-Gatter ist

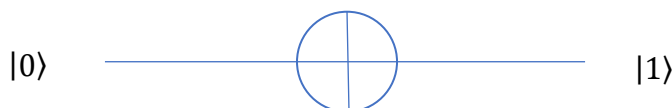


Bild 17 Schaltsymbol für ein NOT-Quantengatter.

Für Qubits in Superposition bedeutet das eine Invertierung nach

$$\alpha |0\rangle + \beta |1\rangle \rightarrow \alpha |1\rangle + \beta |0\rangle.$$

Das NOT-Gatter wird als folgende Matrix repräsentiert

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Beispiel: Invertieren des Qubit $|0\rangle$ mit Hilfe des NOT-Gatter.

$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle.$$

A19 Hadamard-Gatter

Das Hadamard-Gatter ist ein Ein-Qubit-Gatter und wird verwendet, um ein Qubit in Superposition zu bringen. Es versetzt ein Qubit genau „zwischen“ die beiden Basiszustände [30]. Es scheint als würde das Hadamard-Gatter so jede Information aus dem Qubit auslöschen. Wiederholt man die Anwendung des Hadamard-Gatters, so hat das Qubit jedoch wieder den Ausgangswert. Die Information wäre also nicht gelöscht worden. Das zeigt, dass die Zustände, in denen Qubits rein zufällige Messergebnisse ergeben, sehr genau definierte Zustände sind. Das heißt aber auch, dass sich zwischen der ersten und der zweiten Anwendung des Hadamard-Gatters nichts am Zustand des Qubits verändern darf (siehe Folgebeispiel). Wenn sich doch etwas verändert hat, kann das mit dem Hadamard-Gatter aufgedeckt werden, selbst wenn die Änderung gut versteckt war [31]. Für ein einzelnes Qubit $|0\rangle$ oder $|1\rangle$ liefert die Transformation folgende Möglichkeiten [32]

$$H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle),$$

$$H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

Daraus ergibt sich die Matrix-Darstellung

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Für die Amplituden beider Teile bedeutet das

$$\begin{aligned} \alpha|0\rangle + \beta|1\rangle &\rightarrow \alpha \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right) + \beta \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \\ &= \frac{\alpha + \beta}{\sqrt{2}} |0\rangle + \frac{\alpha - \beta}{\sqrt{2}} |1\rangle. \end{aligned}$$

Durch die Normierung der Wellenfunktion ergibt sich die Wurzel (s. Superposition). In der Regel beginnt jede Berechnung auf einem Quantencomputer damit, das Hadamard-Gatter auf alle n -Qubits anzuwenden, um sie so in Superposition zu bringen, die dann zu 2^n Zuständen führt [28]. Nachfolgendes Beispiel zeigt die Überführung der Qubits $|ABC\rangle$ mit $|000\rangle$ in die Superposition. Nach Anwendung des Hadamard-Gatters folgt

$$H(A) = |000\rangle + |100\rangle,$$

$$H(B) = (|000\rangle + |010\rangle) + (|100\rangle + |110\rangle),$$

$$H(C) = (|000\rangle + |001\rangle) + (|010\rangle + |011\rangle) + (|100\rangle + |101\rangle) + (|110\rangle + |111\rangle) [33].$$

Das NOT- und das Hadamard-Gatter sind die am häufigsten eingesetzten Ein-Qubit-Gatter [28]. Darüber hinaus gibt es weiter, die hier jedoch nicht erwähnt sein sollen.

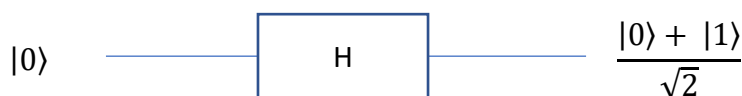


Bild 18 Schaltsymbol für ein Hadamard-Quantengatter.

Wie bereits angesprochen führt die doppelte Anwendung des Hadamard-Gatters wieder zum ursprünglichen Zustand. Aufbauend auf Abbildung 18 bedeutet das



Bild 19 Schaltsymbol für eine doppelte Hadamard-Quantengatter Anwendung.

mit

$$|0\rangle \rightarrow \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right) \rightarrow \frac{1}{\sqrt{2}} \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} + \frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) = |0\rangle.$$

A20 Controlled NOT-Gatter

Das Controlled NOT-Gatter, CNOT-Gatter, ist ein NOT-Gatter, welches von dem Zustand eines zweiten Qubits abhängt. Es ermöglicht so die Interaktion von zwei Qubits in einem Quantenregister. Steht das kontrollierende Qubit x auf $|1\rangle$ so wird das Ziel-Qubit y invertiert. Steht es auf $|0\rangle$ so bleibt es unverändert [32]. Notiert werden beide Qubit-Zustände in einer Klammer, wobei das erste das kontrollierende Qubit x und das zweite das Ziel-Qubit y darstellt,

z. B. $|01\rangle$.

Für die Notation der beiden Qubits in Superposition und ihrer Amplituden erfolgt die Schreibweise analog als

$$|\psi\rangle = \alpha |00\rangle + \beta |01\rangle + \gamma |10\rangle + \delta |11\rangle, \text{ mit}$$

$$|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1.$$

Beispiel für kontroll- und Ziel-Qubit vor und nach einem CNOT-Gatter [28]:

Vor dem CNOT-Gatter	→	Nach dem CNOT-Gatter
$ 00\rangle$	→	$ 00\rangle$,
$ 01\rangle$	→	$ 01\rangle$,
$ 10\rangle$	→	$ 11\rangle$,
$ 11\rangle$	→	$ 10\rangle$,

wodurch sich für das CNOT-Gatter folgende Matrixdarstellung ergibt

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

$$CNOT: |x, y\rangle \rightarrow |x, x \oplus y\rangle$$

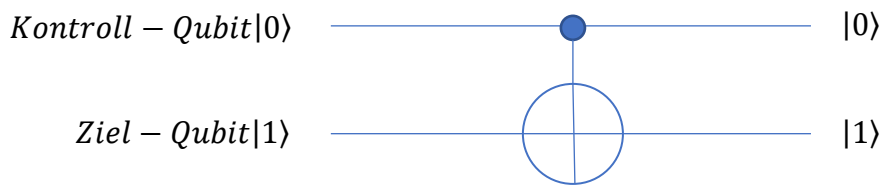


Bild 20 Schaltsymbol für ein CNOT-Quantengatter [34].

Die Ergebnisse nach einem CNOT-Gatter entsprechen einem XOR-Gatter aus klassischen Rechnersystemen, bei denen das Eingangs-Bit aufbewahrt wird (unitäre Variante des Exklusiv-Oder [35]). Das CNOT-Gatter wird verwendet, um beide Qubits miteinander zu verschränken oder die Verschränkung wieder aufzuheben. Wenn das kontrollierende Qubit selbst nur eine 30-Prozent-Chance hat im Zustand $|1\rangle$ zu sein, dann wird das zweite Qubit auch nur mit einer 30-Prozent-Chance in sein Gegenteil verkehrt. Welchen Zustand das zweite Qubit am Ende hat, ist jetzt also vollkommen abhängig davon, was der tatsächliche Zustand des ersten Qubits war. Die Zustände der beiden Qubits werden verschränkt. Somit können die beiden Qubits, auch wenn sie nicht mehr in unmittelbarer Nähe sind, miteinander interagieren [31].

Beispiel: Ein Hadamard-Gatter erzeugt ein Qubit in Superposition, was anschließend durch ein CNOT-Gatter führt.

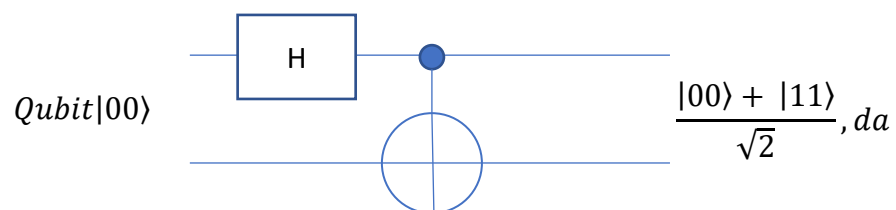


Bild 21 Beispiel für ein Hadamard- und CNOT-Gatter Schaltbild.

$$H|00\rangle = \frac{|00\rangle + |10\rangle}{\sqrt{2}},$$

$$CNOT \frac{|00\rangle + |10\rangle}{\sqrt{2}} = \frac{|00\rangle + |11\rangle}{\sqrt{2}}.$$

Das Hadamard-Gatter erzeugt die Superposition. Das CNOT-Gatter ändert aufgrund des gesetzten Kontroll-Qubit das Ziel-Qubit auf 1, ansonsten nicht.

Wie bereits erläutert kann man mit den Ein-Qubit-Gattern und dem Zwei-Qubit-Gatter CNOT einen universellen Quantencomputer bauen, analog zu den Gattern AND und NOT bei klassischen Computern. Es gibt weitere Gatter, wie das NAND-Gatter, jedoch sind diese Konstellationen äquivalent und können sich untereinander ersetzen.

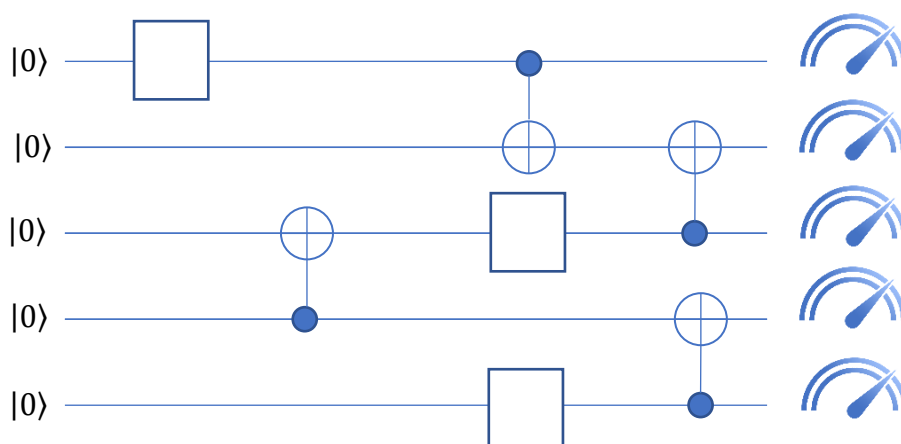


Bild 22 Beispielschaltbild einer Berechnung, bei der nicht näher spezifizierte Gatter zum Einsatz kommen.

Bild 22 zeigt eine beliebige Berechnung auf einem Quantencomputer. Das Schaltbild besteht aus einer Reihe angeordneter Ein- und Zwei-Qubit-Gattern. Am Ende der Berechnung erfolgt eine Messung der Qubits, welches die Superpositionen zerstört und ein Ergebnis mit einer bestimmbarer Wahrscheinlichkeit liefert.

Der Unterschied zu einem klassischen Rechner besteht nun da drin, dass das Ergebnis eben nicht präzise vorhersagbar ist. Das bedeutet, dass man das gewünschte Ergebnis erhalten hat oder aber ggf. auch ein ganz anderes. Es handelt sich keinesfalls um eine exakte Berechnung. Quantencomputer führen keine seriellen Berechnungen aus, bei denen jedes Ergebnis und jeder Zwi-

schensschritt exakt bestimmt sind (Determinismus). Aufgrund der Wahrscheinlichkeiten sind einige Ergebnisse möglicherweise falsch. Somit sind Quantencomputer ungeeignet exakte Rechnungen durchzuführen.

Der Vorteil liegt jedoch in den enormen Möglichkeiten bezüglich paralleler Rechnungen und Speichermöglichkeiten. Werden Rechnungen wiederholt durchgeführt und Wahrscheinlichkeiten verstärkt, desto wahrscheinlicher ist es dann auch ein korrektes Ergebnis zu erhalten. Die Faktorisierung von Zahlen ist ein NP-Problem, welches mit einem Quantencomputer in polynomieller Zeit gelöst werden kann. Das erhaltene wahrscheinlich richtige Ergebnis kann dann durch ein klassisches Rechnersystem polynomiell verifiziert werden. So kann sichergestellt werden ob ein Quantencomputer ein korrektes Ergebnis geliefert hat oder nicht [28]. Was ein Quantencomputer berechnet hängt vom Schaltbild bzw. vom implementierten Algorithmus ab.

A21 Quanten-Fouriertransformation

Die Quanten-Fouriertransformation QFT ist ein Quantenalgorithmus und ein wesentlicher Bestandteil des Shor-Algorithmus. Peter Shor's Algorithmus ist in der Lage das Faktorisierungsproblem, auf das RSA seine Sicherheit stützt, effizient zu lösen. Für ein besseres Verständnis soll die Quanten-Fouriertransformation erläutert werden. Sie führt die Superpositionsstruktur eines Quantenregisters in eine andere Superpositionsstruktur über.

Das Schaltbild der Berechnung besteht aus Hadamard-Gattern sowie einem Controlled Phase-Gatter bzw. kontrolliertem Rotationsgatter. Nach Durchlauf des Algorithmus liegt das Ergebnis bzw. die einzelnen Qubits in umgekehrter Reihenfolge vor. Durch eine Swap Transformation wird diese korrigiert. Diese ist aber für das Verständnis nicht erforderlich und soll an dieser Stelle nur erwähnt sein. Das kontrollierte Rotationsgatter rotiert die Quanten-Phase der Qubits um die Z-Achse.

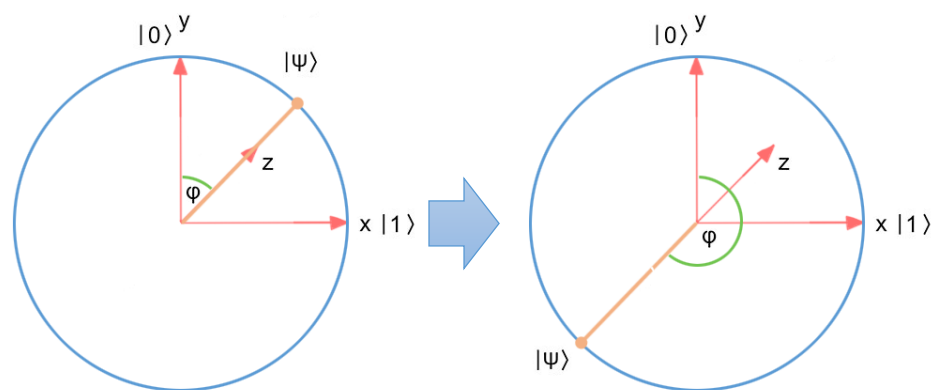


Bild 23 Beispielhafte Darstellung der Rotation der Phase um die Z-Achse.

Ein einfaches Schaltbild für die Quanten-Fouriertransformation mit drei Qubits sieht wie folgt aus

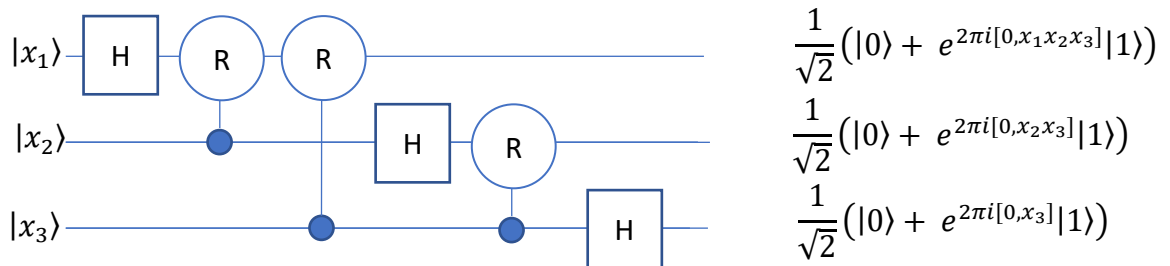


Bild 24 Exemplarisches Schaltbild einer QFT für 3 Qubits [36].⁵⁶

wobei der Aufbau für jedes weitere Qubit einfach nach diesem Schema ergänzt werden kann. Es ist zu sehen, dass jedes Qubit in Superposition in eine andere Superpositionsstruktur versetzt wird. Jedes Qubit wird unterschiedlich häufig rotiert. Das niederwertigste Qubit $|x_3\rangle$ erfährt keine Rotation (Bitte beachten, dass der Swap im Schaltbild nicht dargestellt ist), während das höchstwertige Qubit $|x_1\rangle$ in derselben Zeit zweimal rotiert wird und Interferenzen verwendet, bei denen sich Zustände verstärken bzw. gegenseitig neutralisieren.

Sie wird als lineare Operation F , der auf einem Zustand einer orthonormalen Basis, welche wiederum aus N Zuständen aufgebaut wird, wirkt, verstanden. Eine $N = 2^n$ dimensionale Basis enthält n Qubits. Im Falle eines Zustands $|j\rangle$ kann dies

$$QFT_N: |j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i[0,x_n]} |k\rangle$$

geschrieben werden [37].

Beispiel

Betrachtet wird folgendes 3-Qubit Register mit Dezimal-, Binär- und Vektorschreibweise

⁵⁶ Ein Simulator für die QFT für ein bis drei Qubits findet man unter <http://demonstrations.wolfram.com/QuantumFourierTransformCircuit/>

$$\text{Bsp. } |0\rangle = |000\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, |3\rangle = |011\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \text{etc.}$$

Die Rotation bzw. Phase für zwei Qubits entspricht dann einer $\frac{1}{4}$ - Rotation, also 90° . Für drei Qubits ist es eine Rotation um $\frac{1}{8}$ bzw. 45° . In einem vier Qubit-System wird um $\frac{1}{16}$ rotiert, etc. Die Ausgabe für den 0-Zustand lautet nach der QFT dann wie folgt:

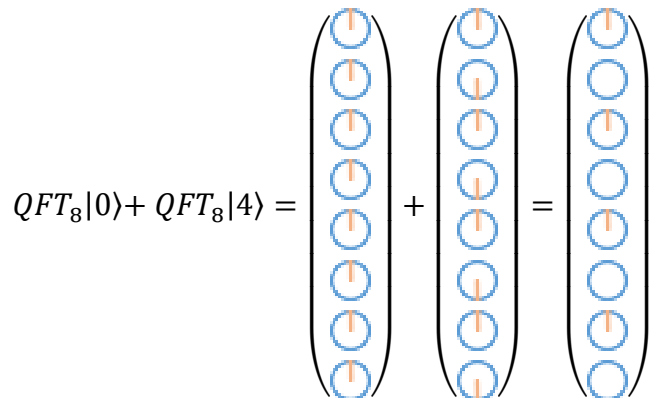
$$QFT_8|0\rangle = QFT_8 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \end{pmatrix}.$$

Exemplarisch wird die Anwendung der QFT für die Werte $|1\rangle$, $|2\rangle$, $|3\rangle$ und $|4\rangle$ dargestellt.

$$QFT_8|1\rangle = \begin{pmatrix} \uparrow \\ \nearrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \end{pmatrix}, QFT_8|2\rangle = \begin{pmatrix} \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \end{pmatrix}, QFT_8|3\rangle = \begin{pmatrix} \uparrow \\ \nearrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \end{pmatrix}, QFT_8|4\rangle = \begin{pmatrix} \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \end{pmatrix}.$$

Analog verhält es sich zu den weiteren Werten. Deutlich zu sehen sind die gestaffelten 45° Sprünge der Phasen. Addiert man nun zwei Werte und führt die QFT durch, so verhält sich das wie folgt:

$$QFT_8(|0\rangle + |4\rangle) = QFT_8|0\rangle + QFT_8|4\rangle \text{ bzw.}$$



Man beachte, dass sich entgegengesetzte Phasen neutralisieren (Interferenz) [38]. Die Werte 1, 3, 5 und 7 haben keine möglichen Messwerte. Die QFT bewirkt also, dass diese Zustände überhaupt nicht mehr auftreten. Sie verwendet hierfür die positiven und negativen Phasen. Mithilfe der QFT können periodische Funktionen erkannt werden.

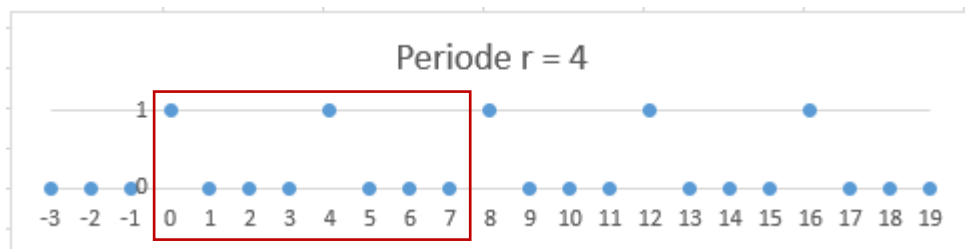


Bild 25 Darstellung der diskreten Funktion $f(x)$ mit den Werten $|0\rangle$ und $|4\rangle$ auf der Zahlengerade im Zahlenraum $N = 2^3 = 8$.

Nach Durchführung zuvor genannter QFT mit $QFT_8|0\rangle + QFT_8|4\rangle$ ergibt sich das Messergebnis

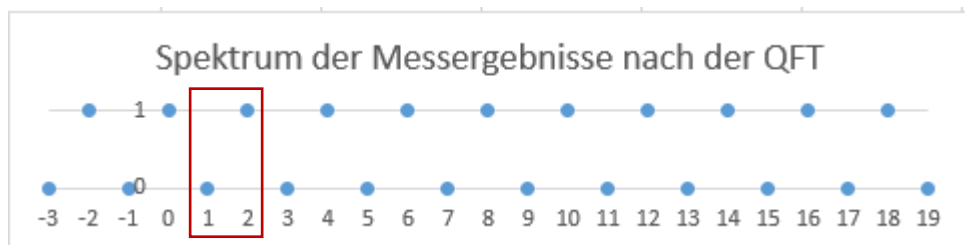


Bild 26 Frequenzraum $N = 2^3 = 8$ mit der Grundfrequenz von 2.

Die Periode r lässt sich hierbei zurückrechnen mit

$$\text{Periode } r = \frac{\text{Zahlenraum}}{\text{Grundfrequenz}} = \frac{8}{2} = 4.$$

Summiert man alle gefundenen Frequenzen zur Grundfrequenz, so erhält man die ursprüngliche diskrete Funktion wieder zurück.

A22 Shor-Algorithmus

Der Shor-Algorithmus bedroht sämtliche Verschlüsselungsverfahren, die auf der Primfaktorzerlegung und dem Problem des diskreten Logarithmus beruhen. Das betrifft insbesondere asymmetrischen Verschlüsselungsverfahren, dem Schlüsselaustausch und digitale Signaturen. Peter Shor zeigte 1997, dass es mit polynomieller Laufzeit möglich ist diese Probleme zu lösen während klassische Angriffe superpolynomielle Zeit benötigen (z.B. Quadratisches Sieb). Der Shor Algorithmus benutzt einen von Leonard Euler erkannten Zusammenhang aus der Zahlentheorie, der das Problem der Primzahlenfindung auf das Finden einer Ordnung respektive Periodenbestimmung der Funktion $f_{a,N}(x) = a^x \bmod N$ zurückführt [28]. Sie bildet die ganzen Zahlen der Menge $\{0, \dots, N - 1\}$ ab.

Die Ordnung r von a in Z_N^* ist die kleinste Zahl, für die gilt

$$f_{a,N}(x + r) = f_{a,N} \text{ für alle } k,$$

$$f_{a,N}(x + r) = a^{x+r} \bmod N,$$

$$f_{a,N}(x + r) = a^x a^r \bmod N,$$

$$f_{a,N}(x + r) = a^x \bmod N \text{ genau dann, wenn:}$$

$$a^r \equiv 1 \bmod N.$$

In diesem Fall ist r die Periode von $f_{a,N}(x) = a^x \bmod N$ [35]. Gesucht wird das x für ein zufällig gewähltes a mit $a \in \{2, \dots, N - 1\}$ und

$$x = a^{r/2} \rightarrow x^2 = a^r \equiv 1 \bmod N,$$

nur dann, wenn r eine gerade Zahl oder $a^{r/2} \equiv \pm 1$ ist. Anschließend können die Primfaktoren mit Hilfe des Euklidischen Algorithmus bestimmt werden

$$ggT\left(N, a^{\frac{r}{2}} + 1\right),$$

$$ggT\left(N, a^{\frac{r}{2}} - 1\right).$$

Einzig die Periode r muss für die Lösung bei der Suche nach den Primfaktoren bestimmt werden [35] [39].

Beispiel: Betrachtet man das Vielfache von 2 mit $2^0, 2^1, 2^2, 2^3$ usw. für das Modul $n = 15$, dann erhält man folgende Elemente der Periode $r \in [1, 2, 4, 8, 1, 2, 4, 8, 1, 2, 4, 8, 1, 2, 4, 8, \dots]$, d.h. Periode lautet $r = 4$. Euler hat gezeigt, dass für eine Funktion f mit

$$a^x \bmod n \rightarrow a \bmod n, a^2 \bmod n, a^3 \bmod n, \dots$$

gilt, dass wenn n ein Produkt aus zwei Primzahlen p, q ist und wenn x nicht durch p oder q teilbar ist, sich das Muster mit einer Periode wiederholt, die ein glatter Teiler von $(p - 1)(q - 1)$ ist. Für das Beispiel $n = 15$ mit $p = 3, q = 5$ und dem Produkt $(p - 1)(q - 1) = 8$ ist die Periode $r = 4$ und somit ein glatter Teiler von 8. Problematisch ist jedoch herauszufinden ab wann die oben gezeigte Periode eintritt und die Sequenz sich wiederholt. Für große n scheitern klassische Rechner [28]. Es werden zwei Quantenregister $|a\rangle, |b\rangle$ benötigt der Länge $\log_2 n$ Qubits [39]. Für die Funktion wird ein Quantenorakel gebaut mit $U_f: |a\rangle|b\rangle \rightarrow |a\rangle|b \oplus f(a)\rangle$. [40] [35].

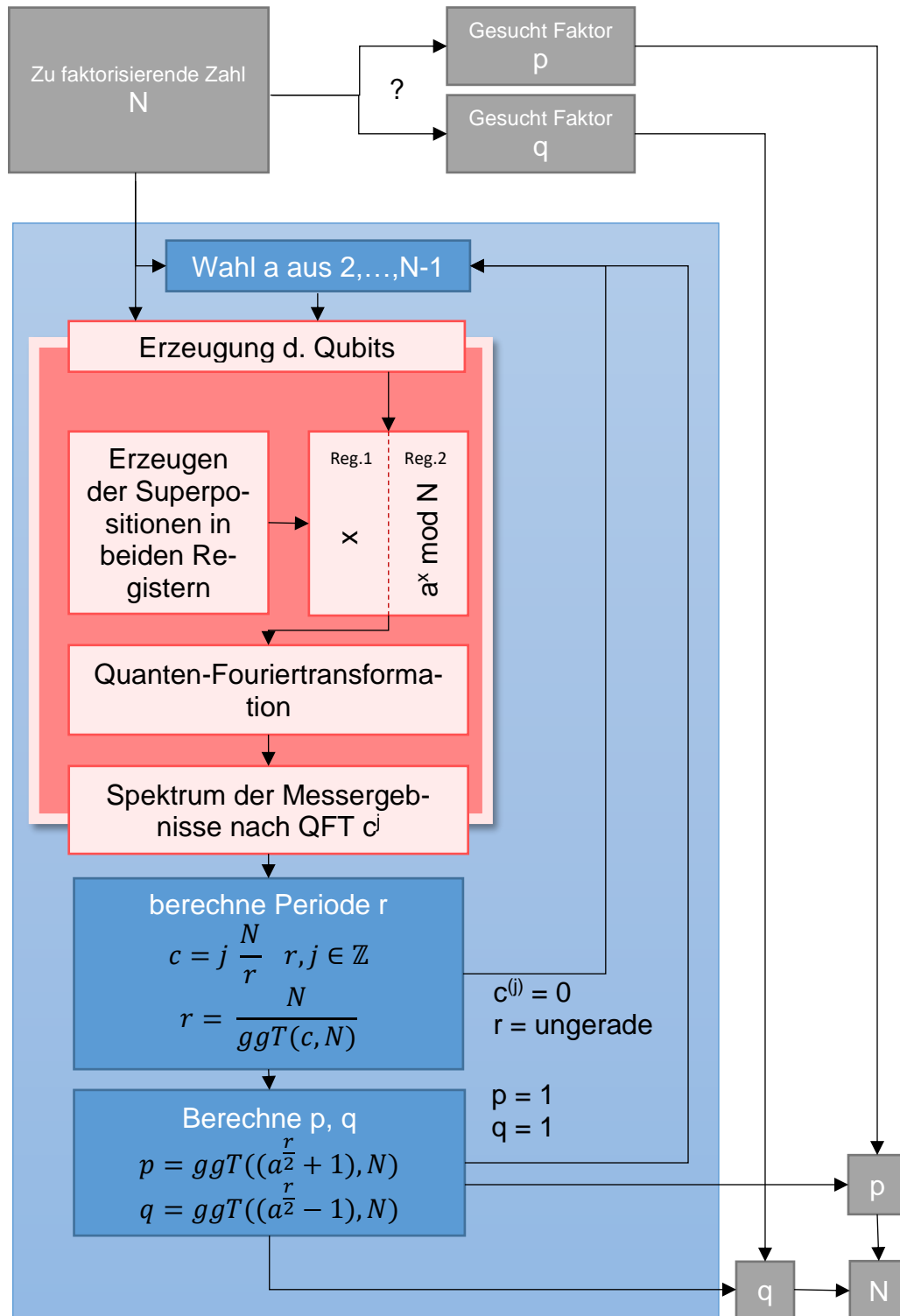


Bild 27 Schema des Shor-Algorithmus mit einem klassischen Teil (blau) und einer Berechnung auf Quantencomputern (rot).

Zunächst werden die Register $|a\rangle, |b\rangle$ initialisiert und anschließend miteinander verschränkt

1. $R = |a\rangle|b\rangle \rightarrow |0..0\rangle|0..0\rangle,$
2. $H|a\rangle \rightarrow R \rightarrow \frac{1}{\sqrt{N}} \sum_{x=0, \dots, N-1} |x\rangle |0..0\rangle.$

Anschließend werden die Exponenten x in das erste Register geladen, also die Werte $x = 0, 1, 2, 3, 4, 5, \dots$ etc. Die Werte der Funktion $f(x)$ werden berechnet und im zweiten Register abgelegt. Die Funktion lautet $f(x) = a^x \bmod n$. Dabei wird die Eingabemenge auf $\{0, \dots, N - 1\}$ beschränkt für eine Zweierpotenz N der Größenordnung n^2 , hier $n = 4 \rightarrow n^2 = N = 16$. Es wird zufällig $a = 7$ gewählt. Durch die Verschränkung der Qubits erfolgt die Berechnung zeitgleich

3. $R \rightarrow U_f R = \frac{1}{\sqrt{N}} \sum_{x=0, \dots, N-1} |x\rangle |f(x)\rangle.$

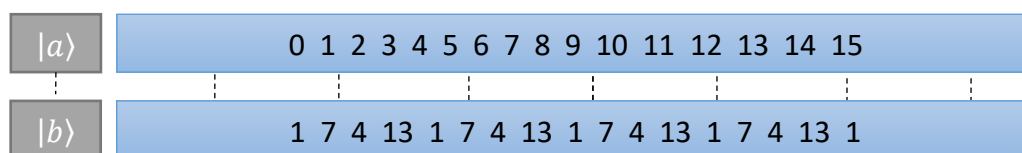


Bild 28 Exemplarische Darstellung der beiden Register a und b mit den Exponenten und den Ergebnissen der Funktion $f(x)$ für alle Exponenten in a.

Anschließend erfolgt die Messung von Register $|b\rangle$, die einen zufälliger Wert k aus $|b\rangle$ ermittelt,

4. $Messe|b\rangle \rightarrow k = a^x \bmod N.$

Dabei ist unklar welcher Wert gemessen wird. Durch die Messung verfallen alle nicht gemessenen Werte. Aufgrund der Verschränkung beider Register nimmt jedoch $|a\rangle$ einen kompatiblen Zustand zu $|b\rangle$ an, d.h. die zugehörigen Exponenten zum gemessenen Wert, z.B. 1, bleiben erhalten.

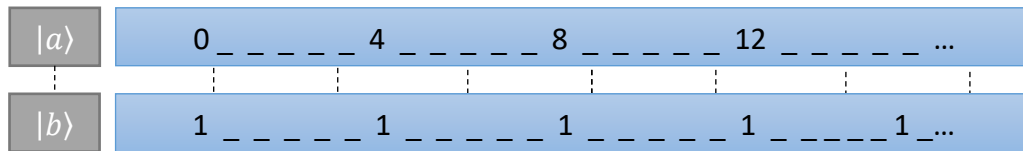


Bild 29 Nach Messen von Register **b** bleiben im Register **a** nach Kollaps der Wellenfunktion nur noch die passenden Exponenten stehen.

Im ersten Register der Exponenten stellen sich nun die Abstände der gesuchten Periode r , beginnend mit einem Offset o

$$o, o + r, o + 2r, o + 3r, o + 4r, \text{ usw.}$$

ein. Anschließend erfolgt nun auf Register $|a\rangle$ die Durchführung der QFT (s. Quanten-Fouriertransformation)

$$5. \quad |a\rangle \rightarrow QFT_Z |a\rangle.$$

Die QFT bewirkt nun, dass die periodische Funktion mit der Periode r und dem Offset o auf eine periodische Funktion mit der Periode $m \approx N/r$ abgebildet wird ohne den störenden Offset o , also ein Vielfaches j von $1/r$. Im nächsten Schritt wird ein Spektralwert c aus Register $|a\rangle$ gelesen [35] [28] [39]. Das Register $|b\rangle$ ist zu diesem Zeitpunkt nicht mehr von Relevanz.



Bild 30 Zustand des Registers **a** nach Durchführung der QFT, welche den Offset o entfernt hat und nur noch ein Vielfaches der Periode r als $1/r$ im Register belässt.

Alle hier vorhandenen Werte werden nun mit gleicher Wahrscheinlichkeit als Wert c gemessen, mit

$$c = j \frac{N}{r},$$

$$\rightarrow \frac{c}{N} = j \frac{1}{r}.$$

Ist $ggT(j, r) = 1$, dann kann r konkret bestimmt werden durch

$$r = \frac{N}{ggT(c, N)},$$

andernfalls muss die Berechnung und Messung erneut durchgeführt werden.

Mit der gefundenen Periode können nun die Faktoren p, q mit

$$ggT(N, a^{\frac{r}{2}} \pm 1)$$

berechnet werden [28] [35] [39] [40].

Beispiele: Shor Algorithmus

Shors Algorithmus wurde im Jahr 2000 auf einem experimentellen Quantenrechner ausgeführt. Es gelang damit die Zahl 15 zu faktorisieren [35]. Dieses Beispiel soll durchgeführt werden. Es soll die Zahl $n = 15$ in seine Primfaktoren $p = 3, q = 5$ zerlegt werden. Gesucht wird die Periode r für die Funktion $f_{a,n}(x) = a^x \bmod n$, mit dem zufällig gewählten Wert $a \in \{2, \dots, n - 1\}$. Es wird $a = 7$ gewählt. Die Eingabemenge wird auf $\{0, \dots, N - 1\}$ beschränkt für eine Zweierpotenz N der Größenordnung n^2 . N wird mit $N = 16$ festgelegt.

Schritt 1: Initialisieren der Register $|a\rangle, |b\rangle$.

$$|a\rangle|b\rangle = |0\rangle|0\rangle$$

Schritt 2: Anwenden der Hadamard-Transformation auf Register $|a\rangle$ und Herbeiführung der Superposition.

$$|a\rangle|b\rangle = \frac{1}{\sqrt{N}} \sum_{x=0, \dots, N-1} |x\rangle|0\rangle,$$

$$|a\rangle|b\rangle = \frac{1}{\sqrt{16}} \sum_{x=0, \dots, 15} |x\rangle|0\rangle,$$

$$|a\rangle|b\rangle = \frac{1}{4} (|0\rangle|0\rangle + |1\rangle|0\rangle + \dots + |15\rangle|0\rangle) \text{ bzw.}$$

$$|a\rangle|b\rangle = \frac{1}{4} \left(\begin{array}{l} |0\rangle|0\rangle + |1\rangle|0\rangle + |2\rangle|0\rangle + |3\rangle|0\rangle + \\ |4\rangle|0\rangle + |5\rangle|0\rangle + |6\rangle|0\rangle + |7\rangle|0\rangle + \\ |8\rangle|0\rangle + |9\rangle|0\rangle + |10\rangle|0\rangle + |11\rangle|0\rangle + \\ |12\rangle|0\rangle + |13\rangle|0\rangle + |14\rangle|0\rangle + |15\rangle|0\rangle \end{array} \right).$$

Schritt 3: Durchführung der Berechnung der Funktion $f(x)$ für alle im ersten Register abgelegten Werte („Orakel“) und schreiben des Ergebnisses in das Register $|b\rangle$.

$$|a\rangle|b\rangle = \frac{1}{\sqrt{N}} \sum_{x=0, \dots, N-1} |x\rangle|f(x)\rangle,$$

$$|a\rangle|b\rangle = \frac{1}{4} \left(\begin{array}{l} |0\rangle|1\rangle + |1\rangle|7\rangle + |2\rangle|4\rangle + |3\rangle|13\rangle + \\ |4\rangle|1\rangle + |5\rangle|7\rangle + |6\rangle|4\rangle + |7\rangle|13\rangle + \\ |8\rangle|1\rangle + |9\rangle|7\rangle + |10\rangle|4\rangle + |11\rangle|13\rangle + \\ |12\rangle|1\rangle + |13\rangle|7\rangle + |14\rangle|4\rangle + |15\rangle|13\rangle \end{array} \right).$$

Schritt 4: Es wird eine Messung in Register $|b\rangle$ vorgenommen. Dabei wird ein beliebiger Wert $k \in \{1, 7, 4, 13\}$ gemessen. Alle nicht gemessenen Werte kollabieren. Durch die Verschränkung zu Register $|a\rangle$ kollabiert auch hier die Superposition. Lediglich die Exponenten zu k bleiben bestehen. Sei $k = 7$, so verbleibt der Teil der Register mit

$$|a\rangle|b\rangle = \frac{1}{4} (|1\rangle + |5\rangle + |9\rangle + |13\rangle)|7\rangle.$$

Die Periode kann nun abgelesen werden. Eine simple Messung würde jedoch nur einen beliebigen, nicht weiter zu verwendendem, Wert liefern.

Schritt 5: Anwenden der Quanten-Fouriertransformation auf Register $|a\rangle$, was den Offset $o = 1$ eliminiert.

$$QFT_N \rightarrow |a\rangle = \frac{1}{\sqrt{N}} \sum_{c=0}^{N-1} e^{\frac{2\pi i c a}{N}} |c\rangle,$$

$e^{\frac{2\pi i}{N}}$ ist die N -te Einheitswurzel mit $N = 4$,

$$QFT_4 \rightarrow |a\rangle = \frac{1}{2} \sum_{c=0}^3 e^{\frac{2\pi i c a}{4}} |c\rangle,$$

$$QFT_{16} \rightarrow |a\rangle = \frac{1}{2} \left(|0\rangle + e^{\frac{2\pi i}{4}} |4\rangle + e^{\frac{2\pi i 2}{4}} |8\rangle + e^{\frac{2\pi i 3}{4}} |12\rangle \right), \text{ bzw.}$$

$$QFT(|a_1\rangle)|b\rangle = \frac{1}{4} (|0\rangle + |4\rangle + |8\rangle + |12\rangle)|1\rangle,$$

$$QFT(|a_2\rangle)|b\rangle = \frac{1}{4} (|0\rangle + i|4\rangle - |8\rangle - i|12\rangle)|7\rangle,$$

$$QFT(|a_3\rangle)|b\rangle = \frac{1}{4} (|0\rangle - |4\rangle + |8\rangle - |12\rangle)|4\rangle,$$

$$QFT(|a_4\rangle)|b\rangle = \frac{1}{4} (|0\rangle - i|4\rangle - |8\rangle + i|12\rangle)|13\rangle.$$

Schritt 6: Messung des Registers $|a\rangle$ mit dem Wert $c \in \{0, 4, 8, 12\}$ als ein Vielfaches j von $N/r = 16/4 = 4$ (gesuchte Periode) mit

$$c = j \frac{N}{r}, j \in \{0, 1, 2, 3\}.$$

Schritt 7: Angenommen es wurde $c = 12$ gemessen. Berechnung der Periode r mit

$$r = \frac{N}{ggT(c, N)},$$

$$r = \frac{16}{ggT(12, 16)} = \frac{16}{4} = 4.$$

Schritt 8: Berechnung der Faktoren p, q von n mit

$$p = ggT\left(\left(a^{\frac{r}{2}} + 1\right), n\right),$$

$$p = ggT\left(\left(7^{\frac{4}{2}} + 1\right), 15\right),$$

$$p = ggT((49 + 1), 15),$$

$$p = ggT(50, 15) = 5.$$

$$q = ggT\left(\left(a^{\frac{r}{2}} - 1\right), n\right),$$

$$p = ggT\left(\left(7^{\frac{4}{2}} - 1\right), 15\right),$$

$$p = ggT((49 - 1), 15),$$

$$p = ggT(48, 15) = 3.$$

Lösung: Damit wurden $p = 5$ und $q = 3$ für $n = 15$ berechnet.

Ein weiteres Beispiel soll den negativen Verlauf des Algorithmus mit anschließender Wiederholung demonstrieren. Das Verfahren soll aufgrund des Umfangs gekürzt dargestellt und vereinfacht durchgeführt werden. Dabei werden die Wahrscheinlichkeiten von Zuständen, die für alle gleich sind, nicht betrachtet.

Zu faktorisieren ist die Zahl $n = 35$ mit $p = 7$ und $q = 5$. Gewählt werden $a = 3, N = 36$. Damit ergibt sich für die Register $|a\rangle, |b\rangle$ für die Funktion $f(|a\rangle) = a^{|a\rangle} \bmod n$:

$$|a\rangle|b\rangle = \dots \left(\begin{array}{l} |0\rangle|1\rangle + |1\rangle|3\rangle + |2\rangle|9\rangle + |3\rangle|27\rangle + |4\rangle|11\rangle + |5\rangle|33\rangle + \\ |6\rangle|29\rangle + |7\rangle|17\rangle + |8\rangle|16\rangle + |9\rangle|13\rangle + |10\rangle|4\rangle + |11\rangle|12\rangle + \\ |12\rangle|1\rangle + |13\rangle|3\rangle + |14\rangle|9\rangle + |15\rangle|27\rangle + |16\rangle|11\rangle + |17\rangle|33\rangle + \\ |18\rangle|3\rangle + |19\rangle|17\rangle + |20\rangle|16\rangle + |21\rangle|13\rangle + |22\rangle|4\rangle + |23\rangle|12\rangle + \\ |24\rangle|1\rangle + |25\rangle|3\rangle + |26\rangle|9\rangle + |27\rangle|27\rangle + |28\rangle|11\rangle + |29\rangle|33\rangle + \\ |30\rangle|3\rangle + |31\rangle|17\rangle + |32\rangle|16\rangle + |33\rangle|13\rangle + |34\rangle|4\rangle + |35\rangle|12\rangle \end{array} \right).$$

Nach einer Messung im Register $|b\rangle$ ergibt sich für $k = 9$. Es verbleibt im Register $|a\rangle$

$$|a\rangle|b\rangle = \dots (|2\rangle + |14\rangle + |26\rangle)|9\rangle.$$

Es erfolgt die QFT auf $|a\rangle$

$$QFT|a\rangle = \dots (|0\rangle + |12\rangle + |24\rangle).$$

Eine Messung ergibt $c = 12$ worauf hin die Periode mit

$$r = \frac{N}{ggT(c, N)} = \frac{36}{ggT(12, 36)} = \frac{36}{12} = 3$$

berechnet wird. Somit ist die gefundene Periode r ungerade, was dazu führt, dass der Algorithmus erneut durchgeführt und a neu gewählt werden muss, da

$$p = ggT((a^{\frac{r}{2}} + 1), n)$$

$$q = ggT((a^{\frac{r}{2}} - 1), n)$$

nicht berechnet werden kann.

Sei im zweiten Durchgang nun $a = 4$. Unter Beibehaltung aller anderen Parameter ergeben sich die Register

$$|a\rangle|b\rangle = \dots \left(\begin{array}{l} |0\rangle|1\rangle + |1\rangle|4\rangle + |2\rangle|16\rangle + |3\rangle|29\rangle + |4\rangle|11\rangle + |5\rangle|9\rangle + \\ |6\rangle|1\rangle + |7\rangle|4\rangle + |8\rangle|16\rangle + |9\rangle|29\rangle + |10\rangle|11\rangle + |11\rangle|9\rangle + \\ |12\rangle|1\rangle + |13\rangle|4\rangle + |14\rangle|16\rangle + |15\rangle|29\rangle + |16\rangle|11\rangle + |17\rangle|9\rangle + \\ |18\rangle|1\rangle + |19\rangle|4\rangle + |20\rangle|16\rangle + |21\rangle|29\rangle + |22\rangle|11\rangle + |23\rangle|9\rangle + \\ |24\rangle|1\rangle + |25\rangle|4\rangle + |26\rangle|16\rangle + |27\rangle|29\rangle + |28\rangle|11\rangle + |29\rangle|9\rangle + \\ |30\rangle|1\rangle + |31\rangle|4\rangle + |32\rangle|16\rangle + |33\rangle|29\rangle + |34\rangle|11\rangle + |35\rangle|9\rangle \end{array} \right).$$

Nach einer Messung im Register $|b\rangle$ ergibt sich für $k = 4$. Es verbleibt im Register $|a\rangle$

$$|a\rangle|b\rangle = \dots (|1\rangle + |7\rangle + |13\rangle + |19\rangle + |25\rangle + |31\rangle)|4\rangle.$$

Es erfolgt die QFT auf $|a\rangle$

$$QFT|a\rangle = \dots (|0\rangle + |6\rangle + |12\rangle + |18\rangle + |24\rangle + |30\rangle).$$

Eine Messung ergibt $c = 30$ worauf hin die Periode mit

$$r = \frac{N}{\text{ggT}(c, N)} = \frac{36}{\text{ggT}(30, 36)} = \frac{36}{6} = 6$$

berechnet wird. Für p, q ergeben sich so die Werte

$$p = \text{ggT}\left(\left(a^{\frac{r}{2}} + 1\right), n\right) = \text{ggT}((64 + 1), 35) = 5,$$

$$q = \text{ggT}\left(\left(a^{\frac{r}{2}} - 1\right), n\right) = \text{ggT}((64 - 1), 35) = 7.$$

Lösung: Damit wurden $p = 5$ und $q = 7$ für $n = 35$ berechnet.

Laufzeitbetrachtung vom Shor-Algorithmus

Wie einleitend bei der Vorstellung des Shor-Algorithmus beschrieben werden zwei Register mit je $\log_2 n$ Qubits für die Faktorisierung einer Zahl n benötigt.

Die Hadamard-Transformation ist mit $2(\log_2 n)$ realisierbar.

Die Berechnung der Funktion $f(x) = x^a \bmod n$ für das Register $|b\rangle$ ist mit $O((\log_2 n)^3)$ durchführbar.

Die Messung von Register $|b\rangle$ benötigt $\log_2 n$ Messungen.

Die Quanten-Fouriertransformation ist realisierbar mit $O((\log_2 n)^2)$ Quantengattern.

Die Messung von Register $|a\rangle$ benötigt $2(\log_2 n)$ Messungen.

Der gesamte Quantenalgorithmus ist daher in $O((\log_2 n)^3)$ Schritten durchführbar [36].

Der klassische Teil von Shors Algorithmus benötigt $O(\log_2 n)$ Multiplikationen und hat in Summe eine erwartete Laufzeit von $O((\log_2 n)^4)$ [35] und ist somit polynomiell.

Im Vergleich zum Algorithmus von Shor war die Laufzeit einer der besten klassischen Algorithmen, das Quadratische Sieb, $O(e^{\sqrt{\log_2 n \log_2 \log_2 n}})$ superpolynomiell, mit dem zur Zeit schnellsten bekannten Verfahren, dem Zahlenkörpersieb, mit $O(e^{(\log_2 n)^{\frac{1}{3}} (\log_2 \log_2 n)^{\frac{2}{3}}})$. Der Ausdruck für die Laufzeit ist nicht exakt bewiesen, sondern nur unter einigen Annahmen hergeleitet, die nicht sicher sind, aber zumindest als wahrscheinlich gelten [41].

A23 Grovers Algorithmus

Eine zentrale Aufgabe von Computersystemen ist das Suchen nach Lösungen. Die meisten Menschen verwenden Suchmaschinen, um Informationen in Sekundenschnelle zu finden und nutzen zu können. Suchmaschinen verwenden ihrerseits spezielle Algorithmen, um diese Ergebnisse zu finden. Für gewöhnlich arbeiten diese Suchalgorithmen jedoch nicht auf ungeordneten Daten. Die Daten sind indiziert und vorsortiert, um ein Auffinden zu beschleunigen. So ist beispielsweise das Telefonbuch alphabetisch nach Familiennamen vorsortiert. Würden diese Informationen willkürlich abgedruckt, so würde eine Suche erheblich länger dauern. Neben der Suche nach Datenbankeinträgen gibt es weitere, auch bereits erwähnte Suchbeispiele wie die Suche nach der kürzesten Route des Handlungsreisenden, das Finden von optimierten Lösungen aber auch Suchen nach passenden Schlüsseln für den Zugang in Systeme (vollständige Schlüsselsuche).

Ein Beispiel wäre, bei beliebig vielen Versuchen, die Suche nach einem 4-stelligen PIN einer EC-Karte. Ohne Anhaltspunkt und bei $N = 10.000$ Möglichkeiten würde man den richtigen PIN im Mittel nach $N/2$ -Versuchen, also 5.000 Eingaben finden. Ähnliches würde für ein unsortiertes Telefonbuch mit $N = 1.000.000$ Einträgen bedeuten oder wenn nur die Nummer und nicht der Name einer Person vorliegt (vgl. „Ziehen ohne Zurücklegen“).

Der Informatiker Lov Grover entwickelte 1996 einen Suchalgorithmus für Quantencomputer, der eine Lösung in $O(\sqrt{N})$ -Schritten auf einer solchen Datenbasis findet. Für die Größenordnung von $N = 10.000$ fände er eine Lösung in 100 Schritten. Der Vorteil wächst mit steigender Größenordnung von N [28] [35].

Der Quantencomputer löst diese Aufgabe, ausgehend von einer Superposition. Er versucht die Amplitude der Wellenfunktion zur korrekten Lösung zu verstärken und liefert somit wahrscheinlich das korrekte Ergebnis bei einer Messung.

Beispiel: Es werden $n = 3$ Qubits betrachtet mit denen $N = 2^3 = 8$ Zustände dargestellt werden können. Sei dies eine unstrukturierte Datenbank, bei der mittels einer Funktion $f(x')$ mit genau einem Eingang x' ein Wert gesucht und ausgegeben wird. Das Such-Register gestaltet sich zu Beginn als

$$|s\rangle = |000\rangle.$$

Anschließend wird $|s\rangle$ mit dem Hadamard-Operator in Superposition gebracht

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle, \text{ es folgt}$$

$$|x\rangle = \frac{1}{\sqrt{8}} (|000\rangle + |001\rangle + |010\rangle + |011\rangle + \dots + |111\rangle).$$

Damit sind in der unstrukturierten Datenbank bei einer Messung alle Lösungen gleich wahrscheinlich, was für die Darstellung der Wahrscheinlichkeitsamplituden Folgendes bedeutet. Sei der gesuchte Eintrag $x' = |011\rangle$.

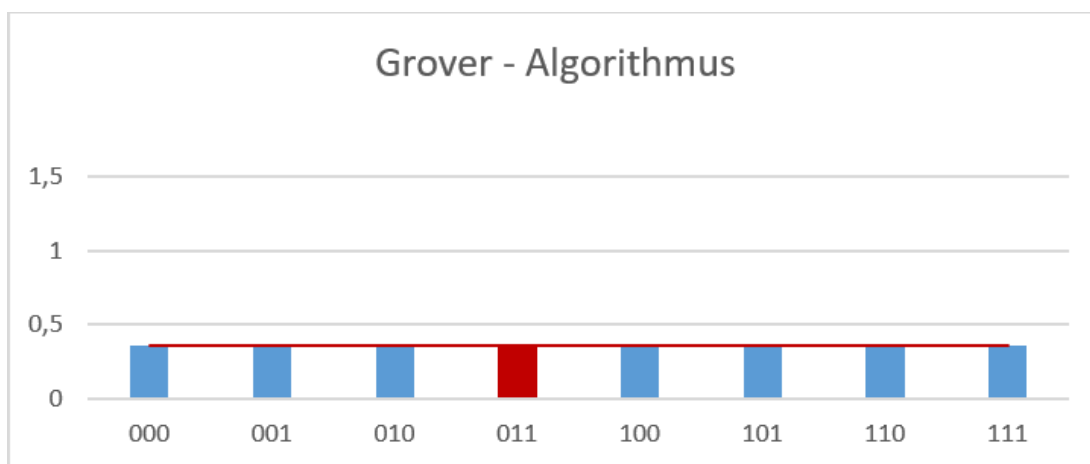


Bild 31 Unstrukturierte Datenbank in Superposition und der gleichen Amplitude für alle Einträge sowie eingezeichnetem Mittelwert.

Der Mittelwert aller Amplituden lautet

$$m = \frac{1}{8} \left(\frac{8}{\sqrt{8}} \right).$$

Es ist möglich genau diesen Zustand herauszuheben, indem die gesuchte Amplitude invertiert wird. Das hat zunächst keinen Einfluss auf die Messwahrscheinlichkeit, ist aber mit einem CNOT-Gatter einfach möglich. Die Invertierung erfolgt durch die Funktion $f(x)$ als

$$\text{CNOT}: |x, y\rangle \rightarrow |x, x \oplus y\rangle.$$

Nach Anwendung des Gatters auf das Register ergibt sich folgender Zustand

$$\text{CNOT}|x\rangle = \frac{1}{\sqrt{8}} (|000\rangle + |001\rangle + |010\rangle - |011\rangle + \dots + |111\rangle),$$

mit einer neuen graphischen Interpretation.

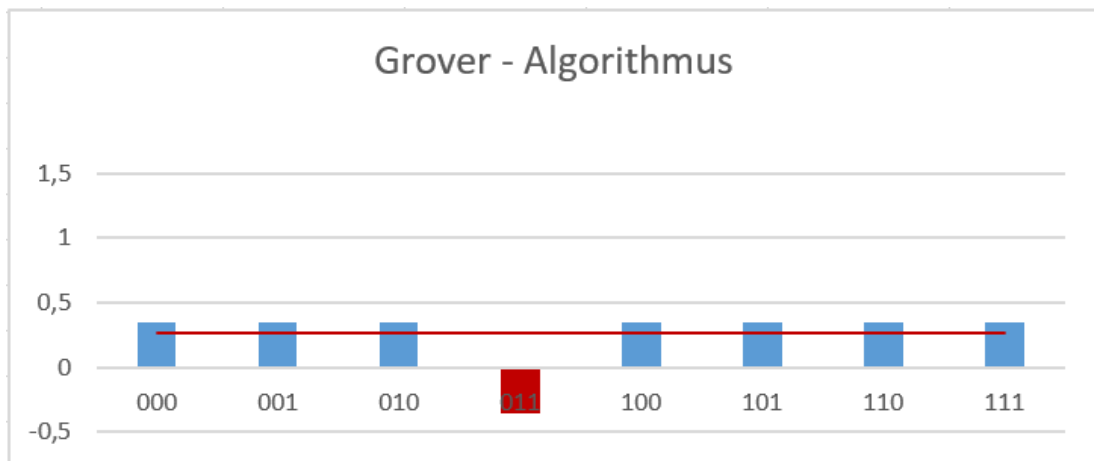


Bild 32 Unstrukturierte Datenbank in Superposition nach der CNOT-Operation sowie eingezeichnetem Mittelwert. Die Ausschläge der Amplituden sind im Betrag dieselben.

Genau hier setzt der Trick von Lov Grover an. Alle Werte verfügen über die Amplitude $\alpha = \frac{1}{\sqrt{8}} = \frac{1}{2\sqrt{2}}$ mit Ausnahme von $|011\rangle$. Der neue Mittelwert lautet

$$m = \frac{1}{8} \left(\frac{7}{\sqrt{8}} - \frac{1}{\sqrt{8}} \right).$$

Werden nun die Amplituden am Mittelwert m mit $\alpha' = m + (m - \alpha) = 2m - \alpha$ gespiegelt erhält die neue Amplitude α

$$\left(1 + \frac{6}{4}\right) \frac{1}{\sqrt{8}} = \frac{5}{2\sqrt{8}}$$

während alle anderen Amplituden a_i auf

$$\left(-1 + \frac{6}{4}\right) \frac{1}{\sqrt{8}} = \frac{1}{2\sqrt{8}}$$

verringert werden.

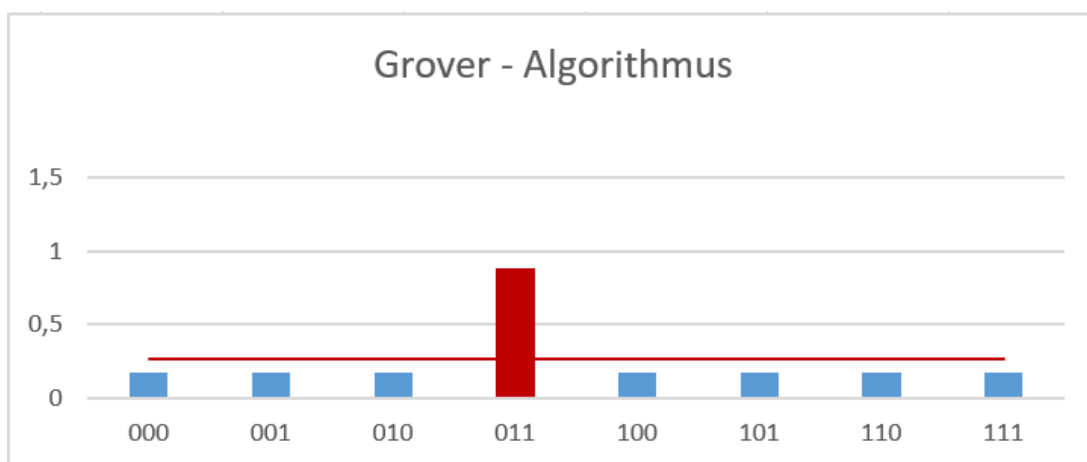


Bild 33 Nach der Spiegelung um den Mittelwert m hat der gesuchte Eintrag eine deutlich höhere Messwahrscheinlichkeit als die verbliebenen.

Auf diese Weise ist es möglich die gesuchte Amplitude aus dem Hintergrundrauschen ansteigen zu lassen und eine Messung in der Wahrscheinlichkeit dahingehend positiv zu beeinflussen. Das Gewicht der Amplitude α für x' wächst (fast) um $\frac{2}{\sqrt{N}}$ in jeder Iteration während der Mittelwert m im Wesentlichen bei $\frac{1}{\sqrt{N}}$ verweilt [42]. Mit jeder weiteren Anwendung des Algorithmus verstärkt sich so die Wahrscheinlichkeit für eine Messung [35] [28]. Der Algorithmus wird so oft wiederholt bis die Amplitude α für x' über $\frac{1}{2}$ liegt. Dafür werden $\frac{\sqrt{N}}{2}$ Iterationen benötigt. Allgemein bedeutet das für die Wahrscheinlichkeiten

$$x': \text{Wahrscheinlichkeit} = \frac{1}{4},$$

$$x_i \neq x': \text{Wahrscheinlichkeit} = \frac{\binom{1-\frac{1}{4}}{N-1}}{\binom{3}{4 \cdot 2^n}} \approx \frac{3}{4 \cdot 2^n}, [42].$$

Die Schritte der Grover-Iteration können mit $O(\log_2 N)$ Quantengattern realisiert werden, während der Algorithmus $O(\sqrt{N})$ Schritte benötigt. Die Größe des ganzen Schaltkreises ist

$$O(\log_2 N \sqrt{N}).$$

Die Suche gegenüber einer klassischen Datenbank wird quadratisch beschleunigt [35] [43].

Die Problematik am Algorithmus von Grover ist, dass man zu Beginn den gesuchten Zustand invertieren muss, damit er überhaupt starten kann. Lov Grover ließ die Frage offen und ging in seiner Arbeit davon aus, dass es eine solche Funktion, auch „Orakel“ genannt, existiert. Im verwendeten Beispiel nutzt man ein Hilfs-Qubit, welches mittels CNOT-Gatter dann im Such-Register invertiert wurde⁵⁷ [28]. Darüber hinaus würde die Schwierigkeit eines Angriffs auf ein symmetrisches Verschlüsselungsverfahren auf die Quadratwurzel reduziert werden. Ein Schlüssel der Größe 256 hätte damit nur noch die Wirkung eines Schlüssels der Größe 128. Dieser Verlust an Sicherheit kann leicht durch eine erneute Verdoppelung des Schlüsselraums kompensiert werden [44] [45].

⁵⁷ Auf der 7. internationalen Konferenz PQCrypto 2016 wurde eine Implementierung von AES auf einem Quantensystem vorgestellt. Die Veranstaltung findet sich unter <https://www.youtube.com/watch?v=3dla-l0CYHg>

A24 Quantenteleportation

Die Quantenteleportation findet in der Quantenkryptographie Anwendung. Der Begriff Teleportation beschreibt die Überwindung von Raum ohne das Zurücklegen von Weg und Zeitaufwand. In der Quantenteleportation wird aber kein Teilchen von einem Ort zu einem anderen transportiert. Es wird lediglich der Zustand übertragen. In der Quanteninformatik spricht man von verschränkten Qubits. Bei dem Transport ist der Zustand des zu transportierenden Qubits unbekannt. Es ist nicht möglich ein Qubit zu messen und dann zu übertragen. Ein derartiger Transport von Informationen ohne Zeitverlust würde gegen die Relativitätstheorie verstoßen (Transport von Informationen mit Überlichtgeschwindigkeit).

Die Übertragung erfolgt über einen Quantenkanal. Dieser ist in der Lage ein Qubit zu transportieren (z.B. Glasfaser für den Transport von Photonen). Während des Transportes darf die Superposition nicht beeinflusst werden (Kohärenzzeit) [35]. Am Ziel liegt bereits vor der Teleportation ein Qubit vor, welches im verschränkten Zustand zum Qubit an der Quelle ist. Diese Konstellation ist Vorbedingung für eine Teleportation.

Angenommen Alice will Bob den Zustand $|\psi\rangle$ eines Qubits $|x\rangle$ teleportieren. Alice kennt $|\psi\rangle$ nicht,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle.$$

Alice besitzt nun zwei Qubits $|a\rangle, |b\rangle$ und verschränkt (einer von vier Bell-Zuständen) diese wobei sie das Qubit $|b\rangle$ Bob gibt, damit die Eigenschaft von $|\psi\rangle$ von $|x\rangle$ auf $|b\rangle$ über $|a\rangle$ übertragen werden kann.

$$|ab\rangle = \frac{1}{\sqrt{2}}(|1_a 0_b\rangle - |0_a 1_b\rangle).$$

Alice wendet nun auf ihr Qubit $|a\rangle$ das CNOT-Gatter an. Das Kontrollierende Qubit ist in dem Fall $|x\rangle$,

$$CNOT|x, a\rangle \rightarrow |x\rangle|a \oplus x\rangle .$$

Anschließend wendet Alice auf das zu übermittelnde Qubit $|x\rangle$ das Hadamard-Gatter an und bringt es in Superposition. Alice misst die Qubits $|x\rangle, |a\rangle$ womit sie beide Superpositionen zerstört. Sie erhält eine von vier gleichgewichtigen Messergebnissen $|00\rangle, |01\rangle, |10\rangle, |11\rangle$. Das Qubit von Bob geht in einen an $(|a\rangle, |x\rangle)$ gekoppelten Zustand über. Alice teilt die Ergebnisse der Messung Bob über einen konventionellen Kanal mit (das erzwingt eine Kommunikation unter Lichtgeschwindigkeit und ist somit konform mit der Relativitätstheorie, da die Information Bob nicht schneller erreichen kann) [35] [28].

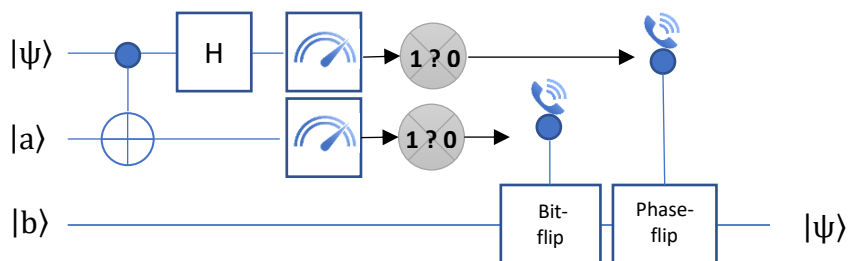


Bild 34 Bei der Quantenteleportation kommen drei Qubits zum Einsatz, bei dem eines bei Bob vorliegt. Dieses wird in Abhängigkeit der Messergebnisse mit einem Bit- oder Phasenflip bearbeitet. Anschließend hat $|b\rangle$ den Zustand $|\psi\rangle$ angenommen [35, p. 128].

Ein Bitflip X ist definiert mit

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$



Bild 35 Schaltbild eines Bitflips.

Ein Phasenflip Z definiert sich mit

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

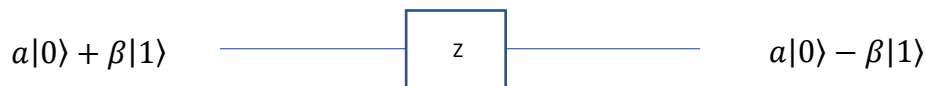


Bild 36 Schaltbild eines Phasenflips.

Beispiel: Alice möchte den Zustand eines Qubits $|x\rangle$ zu Bob teleportieren. Das Qubit definiert sich mit

$$|x\rangle = \alpha|0\rangle_x + \beta|1\rangle_x, \alpha, \beta \in \mathbb{C}.$$

Es könnte sich dabei um Polarisation handeln in der Stellung horizontal (0) und vertikal (1). Darüber hinaus gibt es die Qubits $|a\rangle, |b\rangle$ im verschränkten Zustand mit

$$|\phi\rangle_{a,b} = \frac{1}{\sqrt{2}}(|0\rangle_a|0\rangle_b + |1\rangle_a|1\rangle_b).$$

Das System befindet sich in einem Gesamtzustand

$$|\varphi\rangle_{x,a,b} = |x\rangle|\phi\rangle_{a,b},$$

$$|\varphi\rangle_{x,a,b} = (\alpha|0\rangle_x + \beta|1\rangle_x) \left(\frac{1}{\sqrt{2}}(|0\rangle_a|0\rangle_b + |1\rangle_a|1\rangle_b) \right),$$

$$|\varphi\rangle_{x,a,b} = \frac{\alpha}{\sqrt{2}}(|0\rangle_x|0\rangle_a|0\rangle_b + |0\rangle_x|1\rangle_a|1\rangle_b) + \frac{\beta}{\sqrt{2}}(|1\rangle_x|0\rangle_a|0\rangle_b + |1\rangle_x|1\rangle_a|1\rangle_b).$$

Anschließend erfolgt die Anwendung des CNOT-Gatters. $|x\rangle$ ist das kontrollierende-Qubit, $|a\rangle$ das Ziel-Qubit. Lediglich im zweiten Teil steht $|x\rangle$ auf $|1\rangle$ und hat somit Auswirkungen auf $|a\rangle$,

$$|\varphi\rangle_{x,a,b} = \frac{\alpha}{\sqrt{2}}(|0\rangle_x|0\rangle_a|0\rangle_b + |0\rangle_x|1\rangle_a|1\rangle_b) + \frac{\beta}{\sqrt{2}}(|1\rangle_x|1\rangle_a|0\rangle_b + |1\rangle_x|0\rangle_a|1\rangle_b).$$

Nun wird das Hadamard-Gatter auf $|x\rangle$ ausgeführt. Die Qubits $|a\rangle, |b\rangle$ sind von der Operation nicht betroffen,

$$|\varphi\rangle_{x,a,b} = \frac{\alpha}{\sqrt{2}} \left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \right)_x (|0\rangle_a|0\rangle_b + |1\rangle_a|1\rangle_b) + \frac{\beta}{\sqrt{2}} \left(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right)_x (|1\rangle_a|0\rangle_b + |0\rangle_a|1\rangle_b),$$

$$|\varphi\rangle_{x,a,b} = \frac{\alpha}{2}(|0\rangle_x|0\rangle_a|0\rangle_b + |0\rangle_x|1\rangle_a|1\rangle_b + |1\rangle_x|0\rangle_a|0\rangle_b + |1\rangle_x|1\rangle_a|1\rangle_b) + \frac{\beta}{2}(|0\rangle_x|1\rangle_a|0\rangle_b + |0\rangle_x|0\rangle_a|1\rangle_b - |1\rangle_x|1\rangle_a|0\rangle_b - |1\rangle_x|0\rangle_a|1\rangle_b).$$

Es erfolgt nun die Messung. Die Messung wird konkrete Ergebnisse für $|x\rangle, |a\rangle$ liefern. Die Gleichung wird umgestellt, so dass alle Kombinationen der ersten beiden Qubits einfach abgelesen werden können.

$$|\varphi\rangle_{x,a,b} = \frac{1}{2}(|00\rangle_{xa}(\alpha|0\rangle_b + \beta|1\rangle_b) + |01\rangle_{xa}(\beta|0\rangle_b + \alpha|1\rangle_b) + |10\rangle_{xa}(\alpha|0\rangle_b - \beta|1\rangle_b) - |11\rangle_{xa}(\beta|0\rangle_b - \alpha|1\rangle_b)).$$

Tabelle 21 Mögliche Zielzustände des Qubits von Bob in Abhängigkeit der Messwerte bei Alice.

Kombination der Qubits $ x\rangle, a\rangle$ bei der Messung bei Alice	Mögliche Zustände des Qubits $ b\rangle$ von Bob
$ 00\rangle_{xa}$	$\alpha 0\rangle_b + \beta 1\rangle_b$
$ 01\rangle_{xa}$	$\alpha 1\rangle_b + \beta 0\rangle_b$
$ 10\rangle_{xa}$	$\alpha 0\rangle_b - \beta 1\rangle_b$
$ 11\rangle_{xa}$	$\alpha 1\rangle_b - \beta 0\rangle_b$

Angenommen Alice misst den Zustand $|11\rangle_{xa}$. Sie gibt den gemessenen Zustand für $|x\rangle, |a\rangle$ an Bob auf einem üblichen Kanal durch. Bob multipliziert die

Matrizen für den Bit- und Phasenflip und übt die Operation auf sein Qubit $|b\rangle$ aus. Anschließend hat sein Qubit $|b\rangle$ den Zustand von $|x\rangle = \alpha|0\rangle_x + \beta|1\rangle_x$ übernommen [46] [28].

Tabelle 22 Konkrete Operationen, die Bob durchführt für sämtliche Kombinationen gemessener Zustände [35] [47].

$ x\rangle, a\rangle$	$ b\rangle$	Berechnungsvorschrift	Resultat bei Bob
$ 00\rangle_{xa}$	$\alpha 0\rangle_b + \beta 1\rangle_b$	$\begin{pmatrix} \alpha 0\rangle_b \\ \beta 1\rangle_b \end{pmatrix}$, keine Berechnung	$\alpha 0\rangle_b + \beta 1\rangle_b = x\rangle$
$ 01\rangle_{xa}$	$\alpha 1\rangle_b + \beta 0\rangle_b$	$\begin{pmatrix} \alpha 1\rangle_b \\ \beta 0\rangle_b \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} \alpha 0\rangle_b \\ \beta 1\rangle_b \end{pmatrix}$	$\alpha 0\rangle_b + \beta 1\rangle_b = x\rangle$
$ 10\rangle_{xa}$	$\alpha 0\rangle_b - \beta 1\rangle_b$	$\begin{pmatrix} \alpha 0\rangle_b \\ -\beta 1\rangle_b \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} \alpha 0\rangle_b \\ \beta 1\rangle_b \end{pmatrix}$	$\alpha 0\rangle_b + \beta 1\rangle_b = x\rangle$
$ 11\rangle_{xa}$	$\alpha 1\rangle_b - \beta 0\rangle_b$	$\begin{pmatrix} \alpha 1\rangle_b \\ -\beta 0\rangle_b \end{pmatrix} \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} \alpha 0\rangle_b \\ \beta 1\rangle_b \end{pmatrix}$	$\alpha 0\rangle_b + \beta 1\rangle_b = x\rangle$

A25 No-Cloning-Theorem

Bei einem Angriffsversuch auf BB84 und E91 mittels Verschränkung könnte man annehmen, dass Mallory einfach ein Duplikat bzw. einen Klon eines übertragenen Photons oder Qubit erstellen könnte. Eine Pretty Good Quantum Cloning Machine PGQCM kann eine Kopie-Genauigkeit von über 80% erreichen. Wünschenswert wäre jedoch ein identischer Klon. Bei diesem wird der Zustand eines Qubit $|a\rangle$ auf ein Qubit $|m\rangle$ kopiert, ohne dass das Original wie bei der Teleportation verloren geht. Es käme nicht zu den verräterischen Änderungen aufgrund der Verschränkung oder der Änderung aufgrund einer Messung. Alice und Bob könnten das Abhören durch Mallory nicht nachweisen. Mallory könnte die Messmethoden der Qubits auf dem klassischen Kanal abhören und anschließend entsprechend selbst anwenden. Mallory hat für die Operation ein Qubit

$$|m\rangle = |0\rangle$$

zur Verfügung und fängt das Qubit

$$|a\rangle = \alpha|0\rangle_a + \beta|1\rangle_a$$

ab. Es soll nun eine Operation k , das Klonen, auf $|a\rangle|m\rangle$ angewendet werden mit dem Ergebnis

$$k|a\rangle|m\rangle = |a\rangle|a\rangle_m,$$

wobei $|a\rangle_m$ das ursprüngliche Qubit von Mallory kenntlich machen soll. Dieser Vorgang ist einfach, wenn man ein vertikal polarisiertes Photon $|0\rangle$ bzw. ein horizontal polarisiertes Photon $|1\rangle$ hat. Das gilt nur dann, wenn Mallory den Zustand kennt. Mallorys Ziel muss es hingegen sein ein Qubit in unbekanntem Zustand zu kopieren, ohne dafür eine Messung vorzunehmen. Für die linke Seite der Gleichung gilt

$$k|a\rangle|m\rangle = k(\alpha|0\rangle_a + \beta|1\rangle_a)|m\rangle,$$

$$k|a\rangle|m\rangle = \alpha k|0\rangle_a|m\rangle + \beta k|1\rangle_a|m\rangle,$$

$$k|a\rangle|m\rangle = \alpha |0\rangle_a|0\rangle_m + \beta |1\rangle_a|1\rangle_m.$$

Für die rechte Seite der Gleichung gilt

$$|a\rangle|a\rangle_m = (\alpha|0\rangle_a + \beta|1\rangle_a)(\alpha|0\rangle_m + \beta|1\rangle_m),$$

$$|a\rangle|a\rangle_m = \alpha^2|0\rangle_a|0\rangle_m + \alpha\beta|0\rangle_a|1\rangle_m + \alpha\beta|1\rangle_a|0\rangle_m \\ + \beta^2|1\rangle_a|1\rangle_m.$$

Für alle beliebigen Zustände sind die Ergebnisse jedoch ungleich,

$$k|a\rangle|m\rangle \neq |a\rangle|a\rangle_m.$$

Es existiert kein linearer und unitärer Operator, mit dem man jeden Zustand $|a\rangle$ auf einen Zustand $|m\rangle$ klonen kann. Man nennt dieses Ergebnis das No-Cloning-Theorem [35], [28].

Bei diesem Versuch ein Qubit zu klonen hätte man hypothetisch zwei Qubits in genau denselben Zuständen. Der Unterschied zur zuvor vorgestellten Quantenteleportation ist, dass dort das Original mit einem weiteren Qubit verschränkt wird. Anschließend existiert nur noch die Verschränkung und es kann keine Aussage über das ursprüngliche Qubit mehr getroffen werden. Das Original wurde zerstört und existiert nicht mehr. Es existiert nur noch die Kopie. Somit verstößt die Quantenteleportation nicht gegen das No-Cloning-Theorem [28]. Mallory ist nicht in der Lage das gewünschte Ergebnis zu erreichen. Perfektes Quanten-Klonen ist unmöglich [48, p. 7] und stellt somit keine Angriffsmöglichkeit auf die Protokolle BB84 und E91 dar.

A26 Zwei-Teilchen-Systeme, Verschränkung, Bell und EPR

Im Protokoll von Artur Ekert werden zwei verschränkte Teilchen verwendet, welche durch eine EPR-Quelle generiert werden. Bei den Teilchen handelt es sich um Photonen. Die Photonen $|a\rangle, |b\rangle$ können horizontal $|H\rangle$, vertikal $|V\rangle$ oder in Superposition polarisiert sein, also einem überlagernden Zustand $|\phi\rangle$ dazwischen, z.B.

$$|\phi\rangle = |H\rangle_a |V\rangle_b.$$

Somit kann sich ein Zwei-Teilchen-System in Superposition mit allen Basiszuständen

$$|\phi\rangle = \alpha |H\rangle_a |H\rangle_b + \beta |V\rangle_a |V\rangle_b + \gamma |H\rangle_a |V\rangle_b + \delta |V\rangle_a |H\rangle_b \text{ mit}$$

$$|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$$

beschreiben lassen. Der Zustand der beiden Photonen $|a\rangle, |b\rangle$ ist in dem Fall nicht verschränkt. Es handelt sich um das Produkt aus

$$|\phi\rangle = (\alpha |H\rangle_a + \beta |V\rangle_a)(\gamma |H\rangle_b + \delta |V\rangle_b).$$

Es gibt vier Bell-Zustände. Die Bell-Zustände finden Anwendung im Ekert-Protokoll. Alle vier Bell-Zustände lassen sich nicht in einen analogen Produktzustand überführen [49].

$$|\psi^+\rangle = \frac{1}{\sqrt{2}}(|V\rangle|H\rangle + |H\rangle|V\rangle),$$

$$|\psi^-\rangle = \frac{1}{\sqrt{2}}(|V\rangle|H\rangle - |H\rangle|V\rangle),$$

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|H\rangle|H\rangle + |V\rangle|V\rangle),$$

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|H\rangle|H\rangle - |V\rangle|V\rangle).$$

Die Zustände, benannt nach dem Physiker John Bell, bilden zusammen eine Orthonormalbasis des Zustandsraumes der zwei Photonen $|a\rangle, |b\rangle$ (Qubits), wobei $|\psi^+\rangle, |\psi^-\rangle$ grundsätzlich entgegengesetzte Messergebnisse für die beiden Photonen liefern (antikorrelierend). Man bezeichnet zwei Qubits im Zustand $|\psi^\pm\rangle$ oder $|\Phi^\pm\rangle$ auch als EPR-Paare, benannt nach Einstein, Podolsky und Rosen [35, p. 55].

Man verschränkt zwei Qubits mittels Anwendung des Hadamard- und CNOT-Gatter.

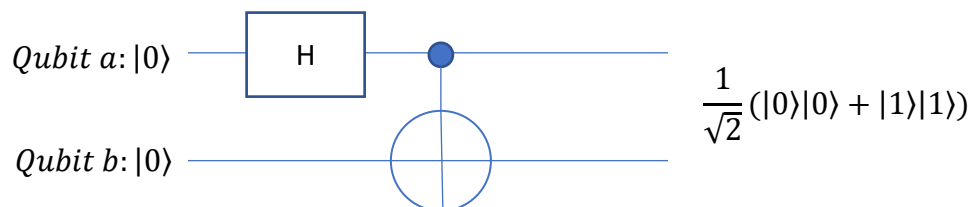


Bild 37 Beispiel: Verschränkung der Qubits a, b im Ausgangszustand $|0\rangle|0\rangle$.

Trennt man nun beide Qubits $|a\rangle, |b\rangle$ räumlich, wobei der Raum auch Lichtjahre betragen kann, und misst Alice das Qubit $|a\rangle$ (sei hier das Erste der beiden Qubits $|a\rangle$) des Gesamtzustands

$$\frac{1}{\sqrt{2}}(|0\rangle_a|0\rangle_b + |1\rangle_a|1\rangle_b)$$

und misst sie $|a\rangle = |0\rangle$, dann verfällt der zweite Teil und Bob erhält für seine Messung zwangsläufig $|b\rangle = |0\rangle$. Dabei spielt die Reihenfolge, wer zuerst die Messung durchführt, keine Rolle. Die Messung des ersten Qubits gibt die zweite Messung vor, sie wird nicht lokal beeinflusst. Diese Kopplung wurde 1935 von Albert Einstein als „*spukhafte Fernwirkung (engl. spooky action at a distance)*“ bezeichnet. Es widerspricht dem Lokalitätsprinzip. Dieses fordert, dass wenn zwei Objekte aufeinander wirken, diese in klassischer Auffassung in Verbindung stehen müssen (z.B. Direktkontakt, Medium, Feld, etc.). Sind

zwei Objekte vollständig voneinander getrennt, dann können sie sich nicht gegenseitig beeinflussen. Einstein war überzeugt, dass es nicht möglich sei, dass diese Information instant übertragen wird. Er ging von „verborgenen Variablen“ (*LHV – local hidden variables*) aus, die die Messung vorherbestimmen würde noch bevor beide Qubits getrennt werden. Diese „verborgenen Variablen“ werden „bei der Geburt“ mitgegeben [50].

John Bell beschäftigte sich mit der Frage wie eine vollständige Beschreibung der Wirklichkeit aussehen könnte, die lokal-realistisch ist und zugleich die quantenmechanischen Phänomene beschreibt [35]. Er zeigte, dass wenn man dem Konzept der „verborgenen Variablen“ folgt, Messergebnisse in jeder möglichen lokal-realistischen Theorie eine Ungleichung erfüllen müssen. Die Quantenmechanik steht jedoch im Widerspruch zu dieser Ungleichung und kann sie in bestimmten Fällen verletzen [35] [49] [51]. Clauser, Horne, Shimony und Holt verallgemeinerten die Ungleichung von Bell und schufen die CHSH-Ungleichung. Diese ist für reale Experimente angepasst und findet Anwendung in der Quantenkryptographie. Durch Verletzung dieser Ungleichung kann die Nicht-lokalität, also die Verschränkung, nachgewiesen werden. Es wird nun ein Photonenpaar im Bell-Zustand z.B. $|\psi^-\rangle$ durch eine Quelle mit

$$|\psi^-\rangle = \frac{1}{\sqrt{2}}(|V\rangle_a|H\rangle_b - |H\rangle_a|V\rangle_b)$$

erzeugt und an Alice und Bob geschickt. Beide werden, bei gleichen Filtern, bei der Messung gegensätzliche Messergebnisse erhalten. Alice verfügt über die Filtereinstellungen

$$a_1 = 0^\circ, a_2 = 45^\circ, a_3 = 22,5^\circ.$$

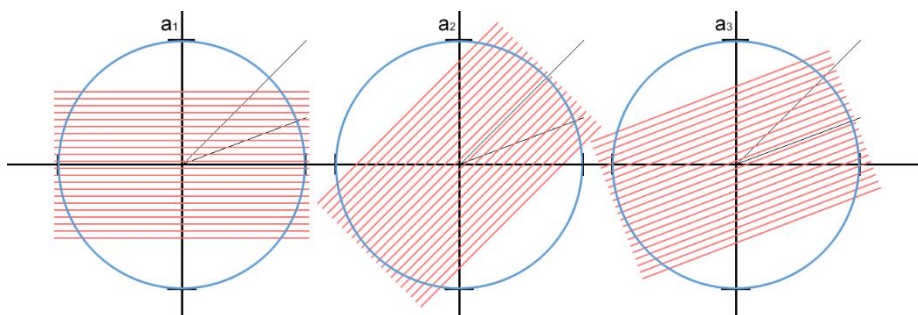


Bild 38 Orientierung der Polarisationsfilter von Alice in der CHSH-Ungleichung.

Bob stehen

$$b_1 = 0^\circ, b_2 = -22,5^\circ, b_3 = 22,5^\circ$$

zur Verfügung.

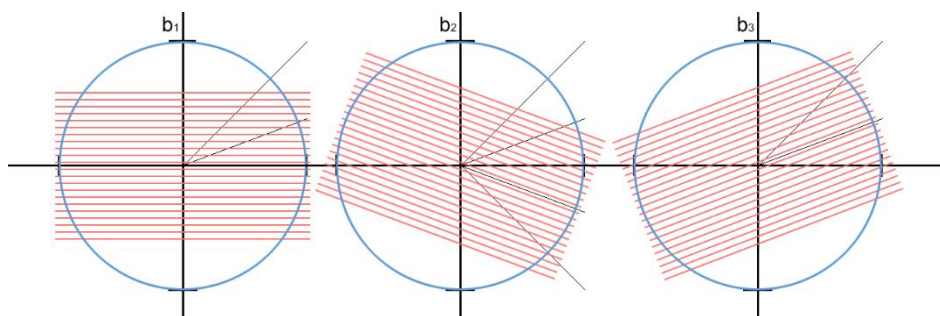


Bild 39 Orientierung der Polarisationsfilter von Bob in der CHSH-Ungleichung.

Messen Alice und Bob jeweils in den Filtern a_1 und b_1 wird einer ein Photon detektieren, der andere nicht. Für die Darstellung ob je Filter ein Photon detektiert wird (+) oder nicht detektiert wird (-) erfolgt die Notation mit

$$a_i \pm, b_j \pm; i, j \in \{1, 2, 3\}.$$

Daneben wird eine Zufallsvariable eingeführt, die jedem Ergebnis einen Wert ± 1 zuschreibt. Beispielsweise beschreibt das Ereignis $a_2+ \rightarrow a_2 = +1$ das Detektieren eines Photons mit der Filterorientierung $a_2 = 45^\circ$ bei Alice. Auf diese Weise lässt sich der Erwartungswert

$$\mathbb{E}(X) = \sum X \cdot P(X)$$

bezüglich der Wahrscheinlichkeit P beider Filterorientierungen a_i, b_j berechnen. Dieser Korrelationskoeffizient errechnet sich mit

$$\begin{aligned}\mathbb{E}(a_i, b_j) &= 1 \cdot P(a_i +) \cdot 1 \cdot P(b_j +) + (-1) \cdot P(a_i -) \cdot (-1) \\ &\quad \cdot P(b_j -) + 1 \cdot P(a_i +) \cdot (-1) \cdot P(b_j -) \\ &\quad + (-1) \cdot P(a_i -) \cdot 1 \cdot P(b_j +)\end{aligned}$$

$$\begin{aligned}\mathbb{E}(a_i, b_j) &= P(a_i+, b_j +) + P(a_i-, b_j -) - P(a_i+, b_j -) \\ &\quad - P(a_i-, b_j +)\end{aligned}$$

$$\mathbb{E}(a_i, b_j) = -\cos(2(a_i - b_j)).$$

Bei zuvor genannten verschränkten Paar von Qubits erhalten Alice und Bob bei gleicher Filtereinstellung perfekt antikorrelierte Ergebnisse

$$\mathbb{E}(a_1, b_1) = \mathbb{E}(a_3, b_3) = -1.$$

Mit Korrelationskoeffizienten unterschiedlicher Orientierung der Filter lässt sich nun die Größe S definieren als

$$S = \mathbb{E}(a_1, b_3) + \mathbb{E}(a_1, b_2) + \mathbb{E}(a_2, b_3) - \mathbb{E}(a_2, b_2).$$

Es handelt sich um die Kontrollgröße S aus der CHSH Ungleichung und kann mittels Filterorientierungen berechnet werden mit

$$\begin{aligned}S &= -\cos(2(a_1 - b_3)) - \cos(2(a_1 - b_2)) - \cos(2(a_2 - b_3)) \\ &\quad + \cos(2(a_2 - b_2))\end{aligned}$$

$$\begin{aligned}S &= -\cos(2(0^\circ - 22,5^\circ)) - \cos(2(0^\circ - -22,5^\circ)) - \cos(2(45^\circ - 22,5^\circ)) \\ &\quad + \cos(2(45^\circ - -22,5^\circ))\end{aligned}$$

$$S = -\cos(-45^\circ) - \cos(45^\circ) - \cos(45^\circ) + \cos(135^\circ)$$

$$S = -\frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2}$$

$$S = -4 \left(\frac{\sqrt{2}}{2} \right)$$

$$S_{max} = -2\sqrt{2}.$$

Die CHSH-Ungleichung sagt unter Annahme verborgener Variablen jedoch

$$-2 \leq S_{CHSH} \leq 2$$

für alle möglichen LHV Theorien aus [52, p. 308]. Das Ergebnis liegt im Widerspruch zu den zuvor gewählten Filtereinstellungen und widerlegt somit die Existenz verborgener Variablen [49] [51]. Es lag eine Verschränkung vor.

Angriff mittels Verschränkung

Bob erhält ein Qubit $|a\rangle = |1\rangle$ von Alice. Während der Übertragung hatte Mallory Zugang zum Quantenkanal. Er ist in der Lage ein eigenes Qubit $|m\rangle = |0\rangle$ mittels CNOT-Gatter mit dem Qubit $|a\rangle$ von Alice an Bob zu verschränken und ändert so $|m\rangle$ auf den Wert $|1\rangle$,

$$CNOT(|a\rangle, |m\rangle) = |1\rangle_a |1\rangle_m.$$

Misst Bob nun in der richtigen Basis, so erfährt Mallory das durch abhören des ungesicherten Kanals und misst mit derselben Methode wie Bob. Er erfährt das Ergebnis $|1\rangle$ und blieb dabei unentdeckt.

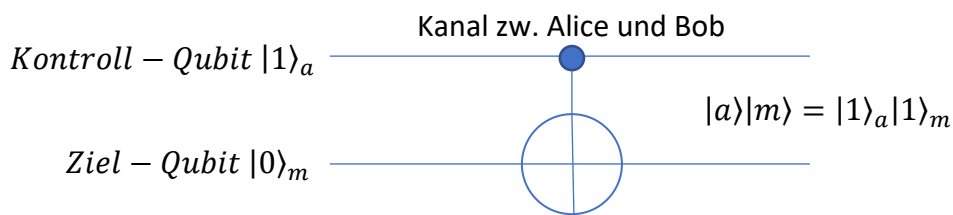


Bild 40 Mallory verschränkt die Übertragung mit seinem eigenen Qubit.

Abhängig vom Zufallswert a' wendet Alice jedoch vor dem Senden das Hadamard-Gatter auf $|a\rangle$ an und überführt das Qubit ggf. in Superposition

$$|a\rangle = \frac{1}{\sqrt{2}}(|0\rangle_a - |1\rangle_a).$$

Anschließend erfolgt die erneute Anwendung des CNOT-Gatters durch Mallory auf $|a\rangle$.

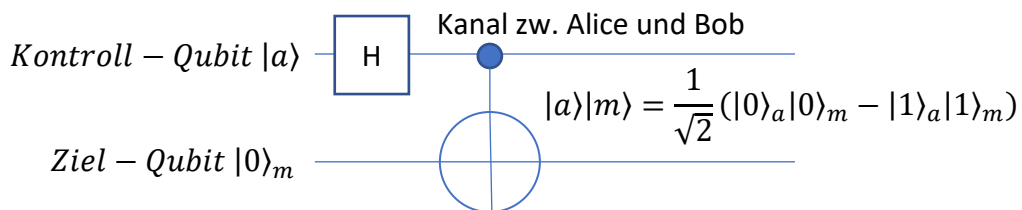


Bild 41 Mallory verschränkt die Übertragung mit seinem eigenen Qubit, diesmal jedoch auf ein Qubit in Superposition.

Messen Bob und Mallory nun beide in der Standardbasis erhalten sie beide dasselbe Ergebnis. Bob würde jedoch das Ergebnis nicht verwenden, da sich bei der späteren Kommunikation mit Alice herausstellen würde, dass sie die Hadamardbasis verwendet hat.

Wird jedoch wie bei Alice ebenfalls ein Zufallswert b' auf Seiten von Bob für eine Messbasis verwendet und gibt diese ebenfalls vor die Hadamardbasis anwenden, so fliegt Mallory in 50% der Fälle auf.

$$H(|a\rangle)|m\rangle = \frac{1}{\sqrt{2}}(H(|0\rangle_a)|0\rangle_m - H(|1\rangle_a)|1\rangle_m),$$

$$|a\rangle|m\rangle = \frac{1}{2}(|0\rangle_a|0\rangle_m - |0\rangle_a|1\rangle_m + |1\rangle_a|0\rangle_m + |1\rangle_a|1\rangle_m).$$

In der Hälfte der Fälle werden die Messungen von Alice und Bob nicht übereinstimmen obwohl beide feststellen, dass sie dieselbe Basis verwendet haben. Die Verschränkung beeinflusst die Übertragung genauso wie der Versuch einer Messung [35]. Je häufiger eine solche Messung durchgeführt wird, desto unwahrscheinlicher ist es, dass Mallory immer Glück hat.

Das BB84-Protokoll wird heute bereits verwendet. Es bietet die Möglichkeit der Anwendung des One-Time-Pad verfahren, welches sicher ist und nachweislich nicht gebrochen werden kann. Im Gegensatz zu klassischen Systemen werden im QKD keine pseudozufälligen Zahlen generiert. Es handelt sich um echte Zufälle, die nicht prognostiziert werden können.

A27 Gedanken zur QKD

Die Protokolle BB84 und E91 ermöglichen den Einsatz des One-Time-Pad OTP und sind somit als sicher zu betrachten, sollten die Verfahren technisch korrekt implementiert werden [53]. Erstmals wurde ein solches Verfahren 1991 eingesetzt, um einen Schlüssel über eine Distanz von 30 cm zu übertragen. Damals war es jedoch noch nicht möglich einzelne Photonen zu generieren. Die Folge waren Lichtblitze, welche mehrere Photonen erzeugten. Mallory bietet sich auf diese Weise neue Angriffsmöglichkeiten, um den Schlüssel zu ermitteln und während des Abhörens unbemerkt zu bleiben [35]. Im Jahr 1995 gelang es an der Universität in Genf eine 23 km lange quantenkryptographische Glasfaserverbindung zwischen Genf und Nyon zu bauen [54]. Im Jahr 2004 fand mittels Quantenkryptographie in Wien eine erste Überweisung statt um eine Spende von 3.000 Euro von der Stadt Wien an die Universität Wien zu transferieren [28]. Die Quantenkryptographie kann als absolut sicher betrachtet werden, eine perfekte technische Implementierung vorausgesetzt. Mallory ist es unmöglich einen ausgehandelten Schlüssel sowie das One-Time-Pad zu brechen. Mallory ist nicht einmal in der Lage unbemerkt in den Schlüsselaustausch aktiv oder passiv einzugreifen, ohne dass dies von Alice und Bob bemerkt wird [54].

Dennoch gibt es technische und logistische Hürden, die einen vollumfänglichen Einsatz der Quantenkryptographie verhindern bzw. erschweren. Zum einen ist es zwingend erforderlich zuverlässig perfekte, einzelne Photonen zu generieren und diese auch nach Möglichkeit 100% korrekt zu detektieren. Auch wurden die Quantenkanäle in den vorgestellten Verfahren stets als perfekt angenommen. Rauschen und Absorption wurden ignoriert. Die Ausrichtung der Filter muss perfekt präzise sein, um Photonen sicher feststellen zu können [55].

Reichweite und Übertragungsgeschwindigkeiten stellen weitere Grenzen in den Einsatzmöglichkeiten da. Die Reichweite über Glasfaser liegt heute bei

ca. 200 km⁵⁸, unter Laborbedingungen bei 300 km [29]. Im Jahr 2016 hat China den ersten Satelliten für Quantenkommunikation in den Orbit gebracht. Dieser soll 500 km über der Erde kreisen und verschränkte Photonen zur Erde senden (Eine dünner werdende Atmosphäre kann hier die Reichweite erhöhen). Es soll so eine Quantenverschlüsselung zwischen Nanshan am Xinjiang Astronomical Observatory in Westchina und dem Xinglong Observatory in Yanshan, etwa 200 km südlich von Nanshan Peking, etabliert werden. Beide Orte liegen 2.000 km weit auseinander, was es notwendig macht den Schlüssel für ca. 2 Stunden zwischen zu speichern [55], bedingt durch die niedrige Höhe des Satelliten sowie der Erdkrümmung (Sichtkontakt). Auch ist die Übertragungsrate aufgrund von Luftturbolenzen in der Atmosphäre mit ca. 600 Bit pro Sekunde gering [29]. China plant eine Reihe ähnlicher Satelliten zu starten, um bis 2030 ein Quantenkommunikationsnetzwerk zu schaffen. Es gibt jedoch technische Größen, wie das Einrichten eines stabilen Quantenkanals und eine präzise Verfolgung des Satelliten, die das Vorhaben vor große Herausforderungen stellt [56].

Neben der satellitengestützten Quantenkryptographie verfügt China ebenfalls über eine 2.000 km lange Verbindung zwischen Peking und Shanghai. Diese Strecke besteht aus 32 Knoten und hat eine Übertragungsrate von 20 kbit/s [55]. Hier besteht das Problem, dass es aktuell noch keine guten Quantenrepeater (verwendet Quantenteleportation unter Nicht-Verletzung des No-Cloning-Theorems) gibt und auch diese lediglich im Labor. Aufgrund von Dekohärenz gehen Informationen während der Übertragung verloren. Um das zu verhindern müssen ideale Bedingungen (sehr kalte Temperaturen und Vakuum) existieren. Derartige technische Geräte existieren noch nicht unter normalen Bedingungen. Um eine Strecke von 2.000 km über 32 Knoten zu betreiben müssen die Informationen von Station zu Station ent- und erneut verschlüsselt werden. Das zwingt zum Einsatz von sogenannten „Trusted Nodes“, d.h. nur sicherheitsüberprüfte Personen mit einer Vertrauensstufe wie die zu übertragenden Nachrichten dürfen Zugang zu den Geräten an gesicherten

⁵⁸ <https://arxiv.org/pdf/0908.4063.pdf>

Standorten haben. Das verhindert den internationalen Einsatz derartiger Geräte und beschränkt ihn maximal auf das Hoheitsgebiet eines Staates, insbesondere seit dem Bekanntwerden der Informationen durch den Whistleblower Edward Snowden [29]. All diese Herausforderungen machen Quantenkryptographie heute zu einer kostspieligen Angelegenheit bei geringen Datenraten, beschränkter Reichweite und fehlender Ende-zu-Ende-Verschlüsselung. Technisch komplexe Geräte machen die neuen und unerprobten Techniken anfällig für Seitenkanalangriffe und Konfigurationsfehler. Auch können bereits existierende Bedrohungen, wie Denial of Service Angriffe, eine erneute Qualität für QKD haben [55]. Dennoch geht auch in diesem Bereich der Fortschritt stetig voran.

A28 Lamport-Diffie One Time Signature, LD-OTS

Das Lamport-Diffie One Time Signature LD-OTS ist ein Schema für digitale Signaturen, welches 1979 von Leslie Lamport entwickelt wurde. Es verwendet kryptographische Hashfunktionen (Einwegfunktionen) die prinzipiell bei Entdecken von Sicherheitsproblemen ausgetauscht werden könnte [18]. Es verwendet ein Schlüsselpaar bestehend aus einem privaten Schlüssel k_{privat} und einem öffentlichen Schlüssel k_{public} . Beide stehen im Zusammenhang. Darüber hinaus gibt es die Einwegfunktion $h(x) = y$. Wendet Alice diese Funktion auf x an, dann ist sie die einzige Person, die x kennt. y hingegen kann veröffentlicht werden [57].

Zunächst generiert Alice das Schlüsselpaar k_{privat}, k_{public} . Dafür generiert sie eine Reihe an zufälligen Zahlen

$$k_{privat} = (x_{1,0}, x_{1,1}, x_{2,0}, x_{2,1} \dots x_{n,0}, x_{n,1}) \in \{0,1\}^{(n,2n)}$$

wobei Alice ihre Nachricht m in n Bit einteilt (Nachricht m mit der Bit-länge n)

$$m = (m_1, m_2, m_3, \dots m_n) \in \{0,1\}^n.$$

Alice hat nun also für jedes mögliche Bit $\{0, 1\}$ einen Schlüssel k_{privat} . Ihren öffentlichen Schlüssel generiert sie aus das Einwegfunktion h angewendet auf ihre privaten Schlüssel k_{privat}

$$k_{public} = (y_{1,0} = h(x_{1,0}), (y_{1,1} = h(x_{1,1})), \dots = h(x_{n,0}), (y_{n,1} = h(x_{n,1}))).$$

Somit bestehen der Signier- und Verifikationsschlüssel aus $2n = k$ Bitstrings der Länge n . Die Signatur s der Nachricht m ist dann

$$s = (s_1, \dots, s_n) = sig(m_1, m_2, m_3, \dots m_n) = (x_{1,m_1}, x_{n,m_n})$$

wobei, wenn $i \in \{1, \dots, n\}$, dann

entweder $m_i = 0 \rightarrow x_{i,0}$

oder $m_i = 1 \rightarrow x_{i,1}$.

Somit besteht die Signatur aus der Hälfte des geheimen Schlüssels k_{privat} .

Beispiel: Ist die zu signierende Nachricht $m = 001$, dann ist $n = 3, k = 6$, die Einwegfunktion h eine für das Beispiel triviale Funktion, die die Reihenfolge von m invertiert, z.B.

$$h(001) = 100 \text{ oder } h(100) = 001.$$

Der zufällig generierte Signaturschlüssel k_{privat} lautet (n, k)

$$k_{privat} = (x_{1,0}, x_{1,1}, x_{2,0}, x_{2,1}, x_{3,0}, x_{3,1}) = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Für die Funktion h , mit der beschriebenen Eigenschaft, ergibt sich dann für den privaten Verifikationsschlüssel k_{public}

$$k_{public} = (h(x_{1,0}), h(x_{1,1}), h(x_{2,0}), h(x_{2,1}), h(x_{3,0}), h(x_{3,1}))$$

$$k_{public} = (y_{1,0}, y_{1,1}, y_{2,0}, y_{2,1}, y_{3,0}, y_{3,1}) = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Nun wird der Signaturschlüssel k_{privat} auf die Nachricht $m = 001$ angewendet.

$$s = (s_1, s_2, s_3) = sig(m_1 = 0, m_2 = 0, m_3 = 1)$$

$$s = (x_{1,0}, x_{2,0}, x_{3,1}) = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix};$$

wie bereits erwähnt wird nur die Hälfte aus k_{privat} verwendet.

Für eine Verifikation kennt Bob in diesem Beispiel sowohl Alice öffentlichen Verifikationsschlüssel k_{public} als auch die eingesetzte Einwegfunktion h . Er verfügt über die Signatur $s = (s_1, s_2, s_3)$ und die Nachricht $m = (m_1 = 0, m_2 = 0, m_3 = 1)$. Er akzeptiert die Signatur, wenn

$$(h(s_1), \dots, h(s_n)) = (y(1, m_1), \dots, y(n, m_n)).$$

$$\begin{aligned} k_{public} &= (y_{1,0}, y_{1,1}, y_{2,0}, y_{2,1}, y_{3,0}, y_{3,1}) \\ &= \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}, \end{aligned}$$

$$s = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix},$$

Bob prüft die Signatur mit der Berechnung

$$(h(s_1), h(s_2), h(s_3)) = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix},$$

was ihn die Signatur zur Nachricht akzeptieren lässt [18, p. 247 ff.].

Hashbasierte Signaturverfahren sind sogenannte One-Time-Signatures und können lediglich einmal verwendet werden, da sonst das Schlüsselpaar $k_{public}, k_{private}$ berechnet werden kann, zumindest Anteile. So könnten Signaturen gefälscht werden [58]. Die Erzeugung der Schlüssel ist effizient, jedoch sind die Signaturen im LD-OPS sehr groß, was das Verfahren nicht praktikabel macht [59, p. 38]. Zusätzlich ist ein Schlüsselmanagement notwendig [60]. Ralph Merkle schlug eine Verbesserung des Verfahrens unter dem Namen W-OTS vor, benannt nach Robert Winternitz. Das Signaturverfahren verwendet relativ kleine Schlüssel- und Signaturgrößen und ist darüber hinaus Robust gegen den Einsatz von Quantencomputer.

A29 Winternitz One Time Signature, W-OTS

Das Verfahren initialisiert mit der Erstellung von 32x 256-Bit zufällig generierten Nummern, dem Feld privater Schlüssel $k_{private}[32]$. Auf jeden dieser Teilschlüssel $k_{private}[i], i \in \{0, \dots, 31\}$ wird die Hashfunktion h in Summe 256-mal angewendet. Das Ergebnis ist der öffentliche Schlüssel $k_{public}[32]$.

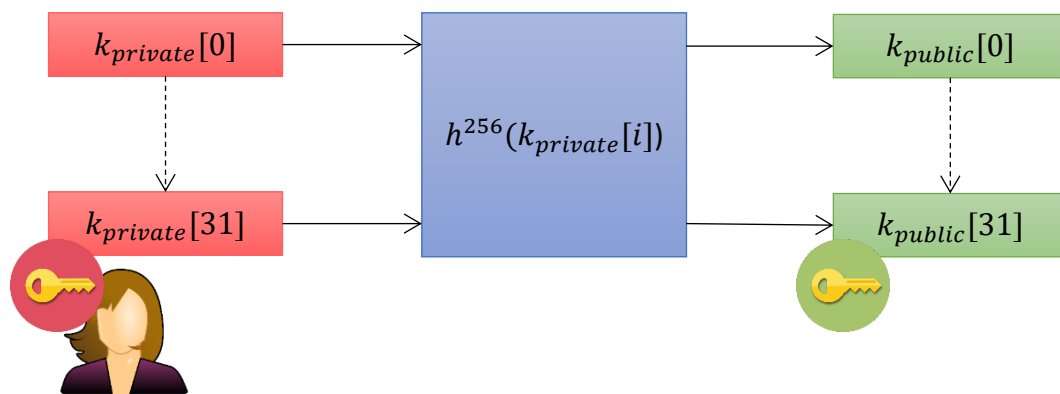


Bild 42 Erstellung eines Schlüsselpaares im W-OTS Verfahren. Jeder Schlüssel hat eine Länge von 256 Bit.

Anschließend wird die Hash-Funktion SHA-256 auf die zu signierende Nachricht M angewendet. SHA-256 erzeugt dabei 32x 8-Bit Werte $M[j], j \in \{1, \dots, 32\}$. Für eine Signatur wird nun für jeden 8-Bit Wert $M[j]$ für $s[i]$ mit dem privaten Schlüssel $k_{private}[i]$ mit $s[i] = h^{256-M[j]}(k_{private}[i])$ berechnet.

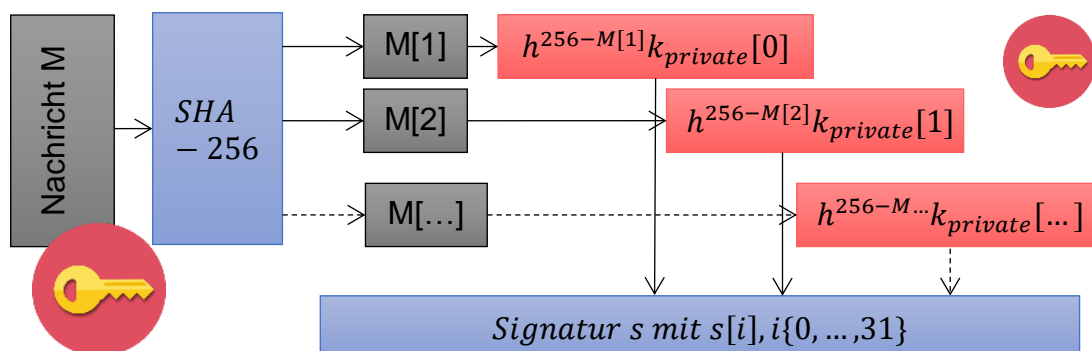


Bild 43 Alice signiert ihre Nachricht m mit ihrem privaten Schlüssel.

Um die Herkunft der Nachricht M zu verifizieren überführt Bob die Nachrichten in dieselben 32x 8-Bit Werte $M[j], j \in \{1, \dots, 32\}$ mittels der bekannten Hash-Funktion SHA-256. Anschließend werden diese Werte entsprechend $M[j]$ -

fach gehashed und das Ergebnis mit dem öffentlichen Schlüssel $k_{public}[i]$ verglichen.

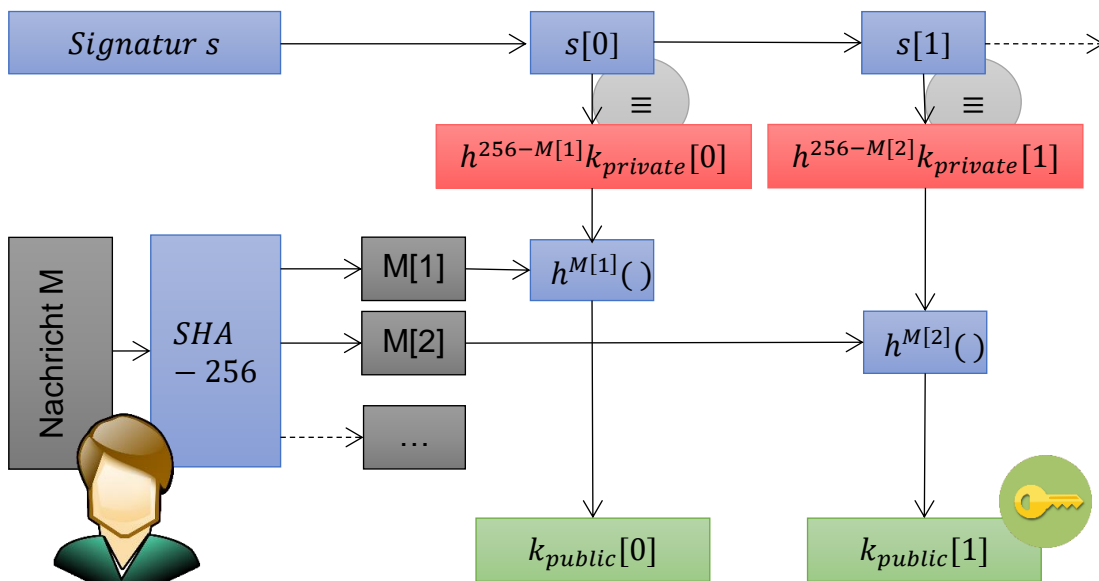


Bild 44 Bob erhält sowohl Nachricht als auch Signatur von Alice und berechnet aus den Angaben den zu erwartenden öffentlichen Schlüssel, den er mit Alice Schlüssel abgleicht.

Sind die berechneten Werte $h^{M[j]} = k_{public}[i]$ dieselben, dann ist die Signatur gültig und Bob kann sich sicher sein, dass Alice die Daten verschickt hat [61]. Auch dieses Verfahren ist sicher solange die unterliegende Hash-Funktion kryptographisch sicher ist [59].

A30 eXtended Merkle Signature Scheme, XMSS

Das XMSS-Verfahren, eXtended Merkle Signature Scheme, wurde von der TU-Darmstadt entwickelt und gemeinsam mit dem IT-Sicherheitsunternehmen genua GmbH zur Praxistauglichkeit gebracht. Das Verfahren wurde über einen Zeitraum von 15 Jahren entwickelt und nun, gemeinsam mit der TU Eindhoven, als Standard eingereicht (IETF-Spezifikation). XMSS wurde im Jahr 2018 unter dem defacto Standard RFC 8391 veröffentlicht (Request for Comments).

XMSS stellt dabei einen Container dar, in dem eine Hash-Funktion eingesetzt wird. Ist diese nicht sicher, so kann sie einfach ausgetauscht werden. Damit kommt XMSS ohne zusätzliche mathematische Hürde aus und die Sicherheit hängt nur von der Hash-Funktion ab. Auf diese Weise ist XMSS unabhängig und läuft nicht Gefahr, wie beispielsweise das diskrete Logarithmus Problem, zu einem späteren Zeitpunkt gelöst zu werden [62] [63].

Um dem Schlüsselmanagement in OTS-Verfahren entgegen zu wirken, in denen Schlüsselpaare nur einmal verwendet werden können, schlug Ralph Merkle den Einsatz von Hash-Bäumen vor. Der von Merkle vorgeschlagene binäre Hash-Baum kann eine feste Anzahl von Verifizierungsschlüsseln auf Basis eines einzelnen, gültigen und öffentlichen Schlüssel generieren. Dabei handelt es sich um den „root“ des Hash-Baumes (= Binärbaum, d.h. ein Knoten hat zwei Kind-Knoten). In einem solchen Baum wird eine Nachricht m mit einem OTS signiert (Blatt des Baumes). Die Signatur besteht nun aus der Einmalsignatur der Nachricht m und den Pfad im Merkle-Baum bis zum öffentlichen Schlüssel, dem root-Element. Eltern-Knoten, d.h. die Knoten von *Level* 0 bis hoch zum root-Element bzw. *Level* h (Höhe des Baums) setzen sich aus der Konkatenation (oder anderer Verrechnung) beider Kinder-Knoten zusammen $Node_{i+1} = H(Node_{links,i} || Node_{rechts,i})$ mit $0 \leq i < h$.

$$Signatur\ s = (OTS, Path_0, Path_2, \dots, Path_{h-1}).$$

Für die Verifikation wird der Einmalschlüssel zur Nachricht m berechnet und abgeglichen. Anschließend wird mit den benötigten Hashverfahren H und dem in der Signatur angegebenen Pfad der öffentliche Schlüssel des Merkle-Baums berechnet. Es sei angemerkt, dass im Gegensatz zum LD-OTS noch ein Index i für das Ausgangsblatt in der Signatur mitgegeben werden muss.

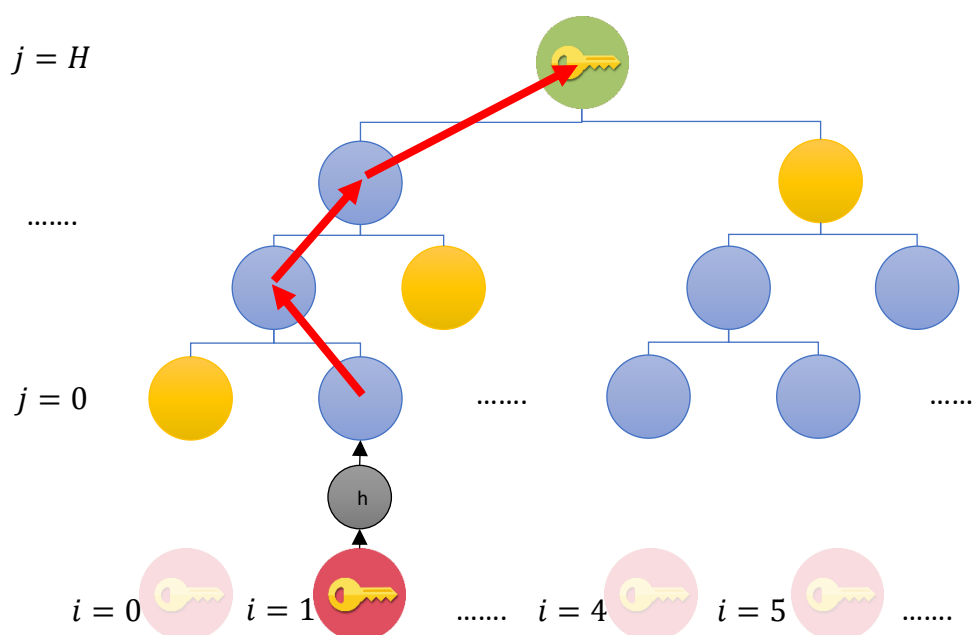


Bild 45 Verifikation in einem Merkle-Baum. Die gelb eingefärbten Knoten sind die notwendigen Pfadangaben, um den öffentlichen Schlüssel zu bestätigen.

Dieses Verfahren hat einige Nachteile. Die Erzeugung ist bei großen Bäumen sehr zeitaufwendig, der private Schlüssel besteht aus sämtlichen OTS-Schlüsseln, also allen Blättern des Baumes, und ist somit ebenfalls sehr groß. Ebenfalls ist die Signatur entsprechend groß (mit steigender Höhe des Baumes wird die Beschreibung des Pfades komplexer). All diese Komponenten spielen bei der Verifikation ebenfalls eine Rolle [64] [59].

Das eXtended Merkle Signature Schema XMSS arbeitet ebenfalls mit kryptographischen Hash-Funktionen und dem Sicherheitsparameter $n \in \mathbb{N}$. Die Länge erhaltende Funktion \mathfrak{S} , welche einen n -Bit String und einen n -Bit Schlüssel auf eine n -Bit Ausgabe abbildet (pseudozufällig)

$$\mathfrak{S}_n = \{F_K: \{0,1\}^n \rightarrow \{0,1\}^n \mid K \in \{0,1\}^n\}, K_n \text{ ist der Schlüsselraum}$$

sowie die second pre-Image resistente Hash-Funktions-Familie

$$\mathcal{H}_n = \{H_K: \{0,1\}^{2n} \rightarrow \{0,1\}^n \mid K \in \{0,1\}^n\}.$$

XMSS verwendet das Winternitz OTS⁺ Schema anstelle des LD-OTS Schema.
Weitere Parameter sind

$\omega \in \mathbb{N}, \omega > 1$, Winternitz Paramert für den Zeit und Speicher Kompromiss,

$n \in \mathbb{N}$, Sicherheitsparameter,

$m \in \mathbb{N}$, Nachrichtenlänge,

$h \in \mathbb{N}$, Höhe des Baumes,

$\ell \in \{0,1\}^n$, zufällige Strings x wird für die $W - OTS^+$ Schlüssel, mit

$$\ell_1 = \left\lceil \frac{m}{\log_2(w)} \right\rceil,$$

$$\ell_2 = \left\lceil \frac{\log_2(\ell_1(w-1))}{\log_2(w)} \right\rceil + 1,$$

$$\ell = \ell_1 + \ell_2,$$

Geheimer Schlüssel als zufälliger Seed aus $\{0,1\}^n$ aus \mathfrak{S}_n .

Ferner wird eine Funktionskette (*chaining function*) wie folgt definiert

$$c_k^i(x, r), \text{ mit}$$

$x \in \{0,1\}^n$, Eingabewert,

$i \in \mathbb{N}$, Iterator,

$k \in K$, Schlüssel

$r = (r_1, \dots, r_j) \in \{0, 1\}^{n \times j}$, mit $j \geq i$, zufälliges Element.

Wenn $i = 0$

$$c_k^0(x, r) = x.$$

Wenn $i > 0$

$$c_k^i(x, r) = f_k(c_k^{i-1}(x, r) \oplus r_i).$$

Schlüsselerstellung, Signatur und Verifikation im W-OTS⁺

Der geheime Schlüssel $sk = (sk_1, \dots, sk_\ell)$ besteht aus den ersten, zufälligen ℓ -Bit Strings. Für das zufällige Element $r = (r_1, \dots, r_j)$ werden $\omega - 1$ - Bit Strings gebildet und auf c angewendet. Mit Hilfe eines zufälligen Funktionsschlüssels k aus K wird der öffentliche Schlüssel berechnet mit

$$\begin{aligned} pk &= (pk_0, pk_1, \dots, pk_\ell) \\ &= \left((r, k), c_k^{\omega-1}(sk_1, r), \dots, c_k^{\omega-1}(sk_\ell, r) \right). \end{aligned}$$

Ein Algorithmus für die Erstellung der Schlüssel hängt vom Sicherheitsparameter n ab sowie dem Konfigurationswert ω .

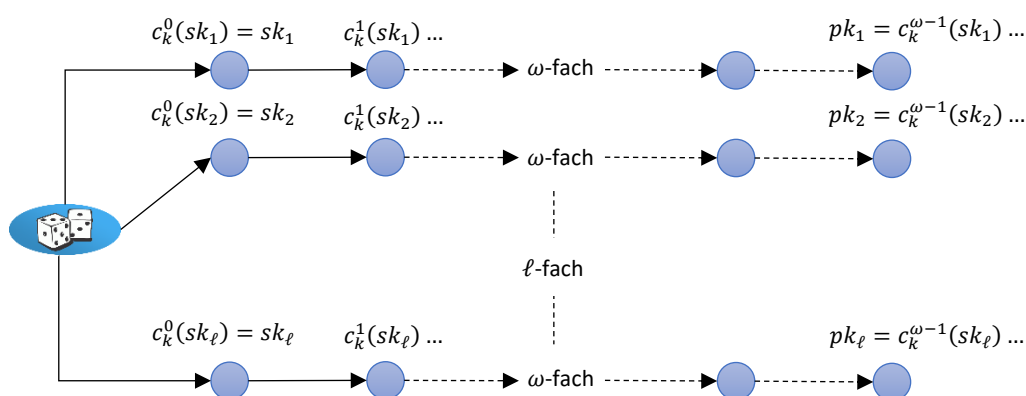


Bild 46 Schema der Schlüsselerstellung mittels chaining function [65].

Um eine Nachricht mit dem W-OTS⁺ zu signieren werden die Parameter Nachricht M , der geheime Schlüssel sk , sowie das zufällige Element r benötigt. Anschließend wird eine Repräsentation für M mit

$$M: M(M_1 \dots M_{\ell_1}), M_i \in \{0, \dots, \omega - 1\}$$

berechnet. Nun wird für M die Prüfsumme C berechnet

$$C = \sum_{i=1}^{\ell_1} (\omega - 1 - M_i), C = (C_1, \dots, C_{\ell_2}).$$

Die Konkatenation von C und M wird als B berechnet

$$B = (b_1, \dots, b_\ell) = M || C.$$

Anschließend ergibt sich eine Signatur

$$\sigma = (\sigma_1, \dots, \sigma_\ell) = (c_k^{b_1}(sk_1, r), \dots, c_k^{b_\ell}(sk_\ell, r)).$$

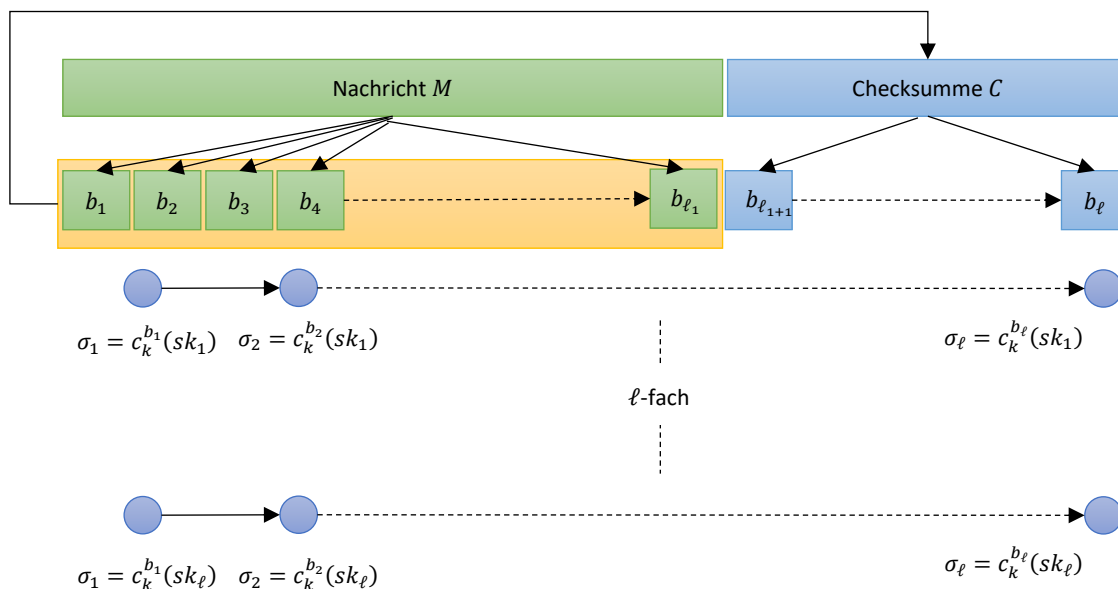


Bild 47 Erstellung der Signatur im W-OTS⁺ nach [65].

Für die Verifikation im W-OTS⁺ erfolgt zunächst erneut die Berechnung der Repräsentation von M , sowie der benötigten Werte von b_i . Anschließend kann ein Abgleich mit dem öffentlichen Schlüssel pk erfolgen

$$pk = (pk_0, pk_1, \dots, pk_\ell) = \left((r, k), c_k^{\omega-1}(sk_1, r), \dots, c_k^{\omega-1}(sk_\ell, r) \right)$$

$$=?$$

$$\left((r, k), c_k^{\omega-1-b_1}(\sigma_1, r_{b_1+1, \omega-1}), \dots, c_k^{\omega-1-b_\ell}(\sigma_\ell, r_{b_\ell+1, \omega-1}) \right).$$

Ist der Vergleich erfolgreich so ist die Verifikation gültig [66].

XMSS verwendet einen Baum als Variante des Merkle-Baums. In Abhängigkeit der Höhe h können 2^h Nachrichten der Länge m -Bits mit einem Schlüsselpaar signieren werden. Hierbei handelt es sich um einen wichtigen Unterschied zu aktuellen Signaturverfahren. Mittels RSA können mithilfe des Schlüsselpaars $k_{public}, k_{private}$ beliebig viele Dokumente signiert werden. Der öffentliche Schlüssel ist bei dieser Art Baum der „root“-Knoten. Die Blätter des Baumes liegen auf Level 0, „root“ liegt auf Level h . Es gibt somit $h + 1$ Level. Die Knoten auf Level $j, 0 \leq j \leq h$ werden als

$$Node_{i,j}, 0 \leq i < 2^{h-j}$$

bezeichnet. Der allgemeine Aufbau des Baums gestaltet sich wie in der folgenden Abbildung dargestellt und gleicht dem Merkle-Baum.

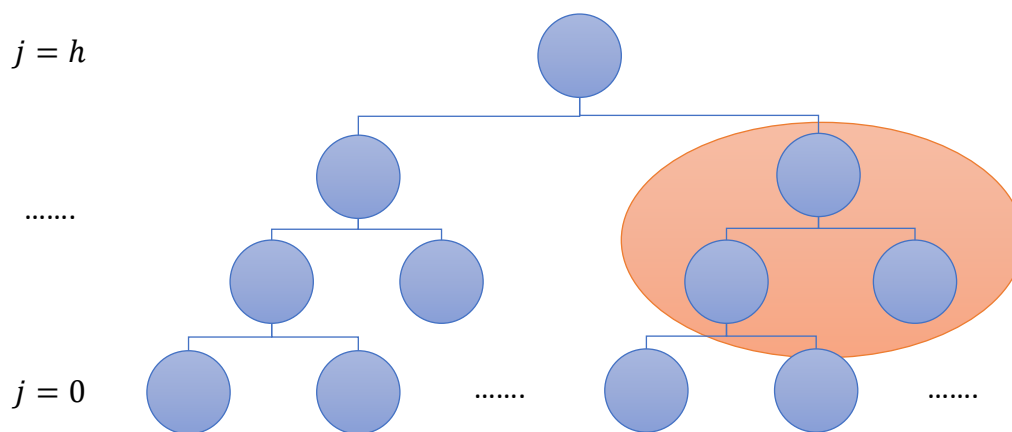


Bild 48 Schematische Darstellung eines XMSS-Baums, angelehnt an den Merkle Hash-Baum von den Blättern auf Level $j=0$ bis zur Höhe h . Hervorhebung des Aufbaus von Knoten in der Hierarchie des Baumes.

Die Knoten der oberen Level j mit $0 < j \leq h$ werden mithilfe der Bit-Maske $Q_j = (b_{l,j} || b_{r,j}) \in \{0, 1\}^{2n}$ nach dem Zufallsprinzip gewählt.

$$Node_{i,j} = h_K((Node_{2i,j-1} \oplus b_{l,j}) || (Node_{2i+1,j-1} \oplus b_{r,j}))$$

$$Node_{i,j} = h_K((Node_{2i,j-1} || Node_{2i+1,j-1}) \oplus Q_j)$$

Wie im Merkle-Baum werden die einzelnen Elemente aller Ebenen von links nach rechts durchgezählt, so dass ganz links $i = 0$ ist und j die Ebene bezeichnet (linkes Blatt/Knoten $Node_{2i}$, rechtes Blatt/Knoten $Node_{2i+1}$).

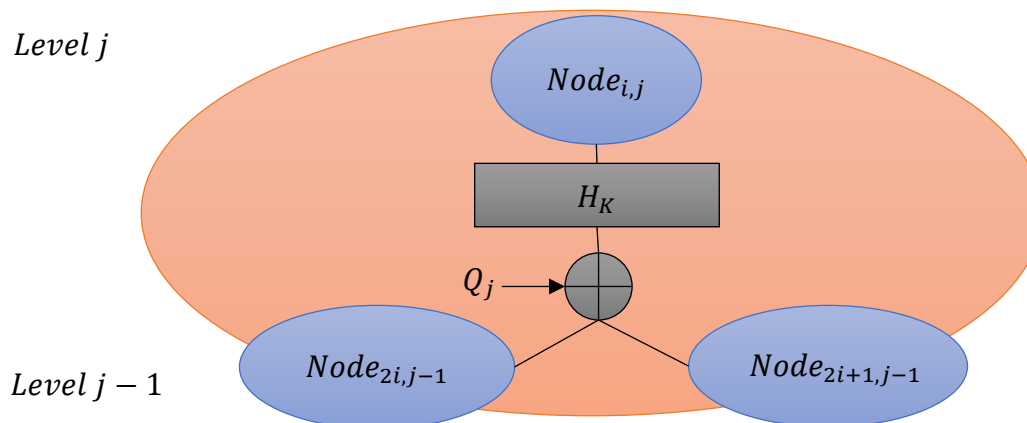


Bild 49 Darstellung des Merkle-Baumes im XMSS Verfahren mit Anwendung der Bit-Maske basierend auf Bild 48.

Die Anwendung der Bit-Maske erlaubt die Sicherheit des Baumes von der Kollisionsresistenz auf die second pre-image-Resistenz zu erhöhen [64]. Anwendung für den Authentifikations-Pfad im Baum findet der Buchmann-Dahmen-Schneider-Algorithmus, BDS⁵⁹, welcher einen Zeit- und Speicherkompromiss in Abhängigkeit eines Parameters ω ermöglicht (vgl. W-OTS⁺). Ein Index zeigt auf das Blatt des Baumes und somit den nächsten ungenutzten W-OTS⁺ privaten Schlüssel. Bei der Initialisierung steht dieser Index auf 0.

Für die Signatur wird das W-OTS⁺ Verfahren verwendet. Die Signatur besteht aus dem Index i , der W-OTS⁺-Signatur σ sowie einem Authentifizierungspfad $auth = (auth_0, \dots, auth_{h-1})$, ausgehend vom Knoten bzw. Blatt $node_{0,i}$ (vgl. Verifikation im Merkle-Baum) [67].

⁵⁹ http://www.horst-goertz.de/hgs-wordpress/wp-content/uploads/2013/09/2_Preis_2008.pdf

A31 Aufbau und Eigenschaften von Gittern

Gitter stellen eine endliche Menge an Punkten bzw. linear unabhängige Vektoren $G = \{g_1, g_2, \dots, g_k\}, k \in \mathbb{N}_0$, in einem n -Dimensionalen Raum, als eine diskrete Untergruppen aus \mathbb{R}^n , dar und verfügen über eine periodische Struktur, $n \in \mathbb{N}_0$. Ein Gitter wird bezeichnet als $\Lambda \subset \mathbb{R}^n$ mit $G \subset \Lambda$ [68],

$$\begin{aligned} \text{Definition } \Lambda = \Lambda(G) = G\mathbb{Z}^k &:= \left\{ \sum_{i=1}^k z_i g_i \mid z_i \in \mathbb{Z} \right\} \\ &= \mathbb{Z}g_1 \oplus \dots \oplus \mathbb{Z}g_k. \end{aligned}$$

$G \in \mathbb{R}^{n \times m}$ ist eine Matrix mit den Spaltenvektoren g_1, g_2, \dots, g_k . Somit ist $\Lambda(G)$ ein Gitter mit dem k -Tupel g_1, \dots, g_k als Basis. Beträgt der Rang eines Gitters mindestens 2, dann besitzt das Gitter unendlich viele Basen, wenn eine unimodulare Matrix $U \in \mathbb{Z}^{n \times n}$ existiert, $G' = GU$. Die Basis ist nicht durch das Gitter definiert, jedoch ist jede Basis eines Gitters vom selben Rang k .

Man kann zwischen "guten Basen" und „schlechten Basen“ unterscheiden. Bei einer guten Basis stehen zwei Vektoren annähernd senkrecht zueinander, z.B. (g_1, g_2) , bei einer schlechten verlaufen sie annähernd parallel, z.B. (h_1, h_2) [63].

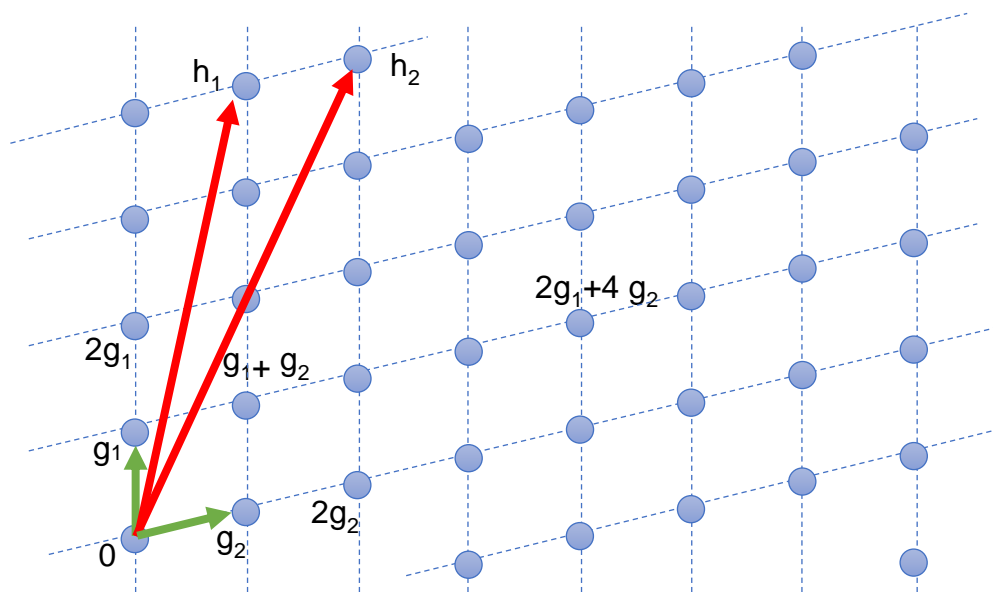


Bild 50 2-Dimensionales Beispielgitter mit der Basis (g_1, \dots, g_i) und (h_1, \dots, h_i) als vereinfachte Darstellung für das Verstehen annähernd senkrechter und parallel verlaufenden Vektoren.

Basen sind dabei nicht einzigartig für Gitter. Verschiedene Basen können dasselbe Gitter beschreiben aber auch ganz verschiedene Gitter.

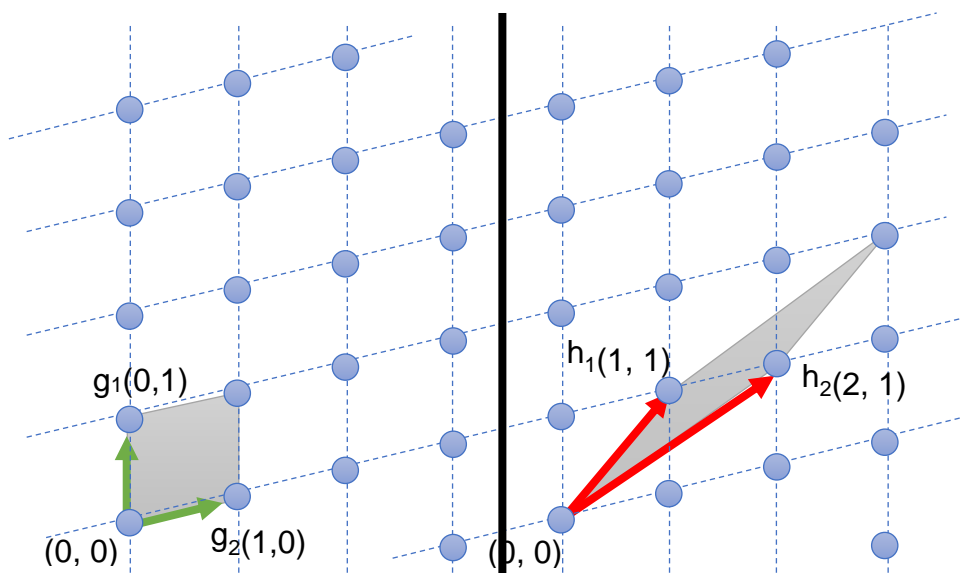


Bild 51 Die Basen G (links) und H (rechts) beschreiben dasselbe Gitter. Es ist offensichtlich, dass man mit den Vektoren in G jeden Gitterpunkt erreichen kann. Für H gilt dasselbe. Beide spannen eine bestimmte Fläche innerhalb des Gitters auf (Parallelepiped).

Mit

$$P_{1,2}(B) = B \left[-\frac{1}{2}, \frac{1}{2} \right)^k = \left\{ \sum_{i \in k} c_i b_i \mid c_i \in \left[-\frac{1}{2}, \frac{1}{2} \right) \right\}$$

kann das fundamentale Parallelepiped um den Null-Punkt ausgerichtet werden.

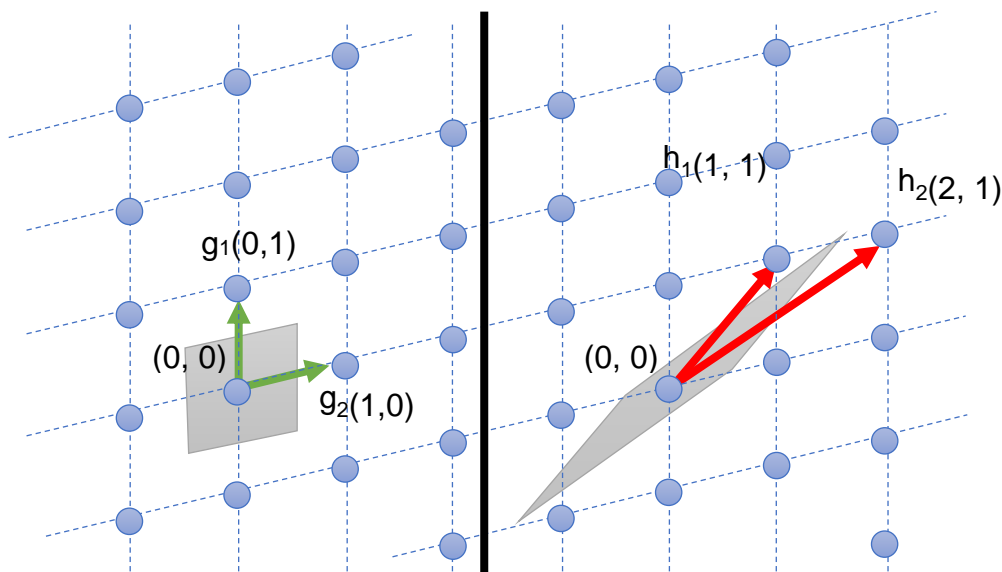


Bild 52 Die beiden fundamentalen Parallelepiped, ausgerichtet um den Nullpunkt [69].

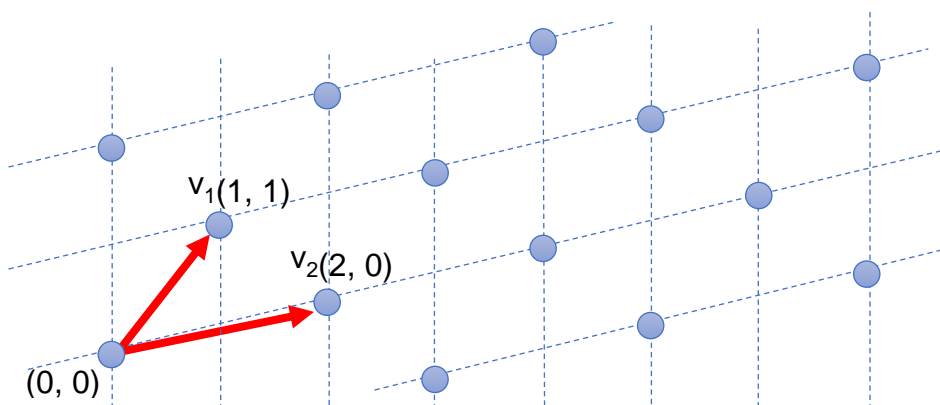


Bild 53 Die Basis V bildet mit seinen Vektoren ein ganz anderes Gitter.

Zwei Basen erzeugen dann dasselbe Gitter, wenn man

- die Vektoren eindeutig permutieren kann, $v_i \leftrightarrow v_j$,

- wenn ein Vektor $v_i \rightarrow -v_i$ negiert werden kann,
- wenn man einen Vektor v_j mit einem Integer k multiplizieren kann und diesen auf einen Vektor v_i addiert, $kv_j + v_i \rightarrow v_i$, $k \in \mathbb{Z}$,

was der zuvor erwähnten unimodulare Matrix U mit den Determinanten ± 1 entspricht. Eine Funktion auf so einem Gitter ist periodisch. Wiederholungen in einer Periode können überall im Gitter wiedergefunden werden.

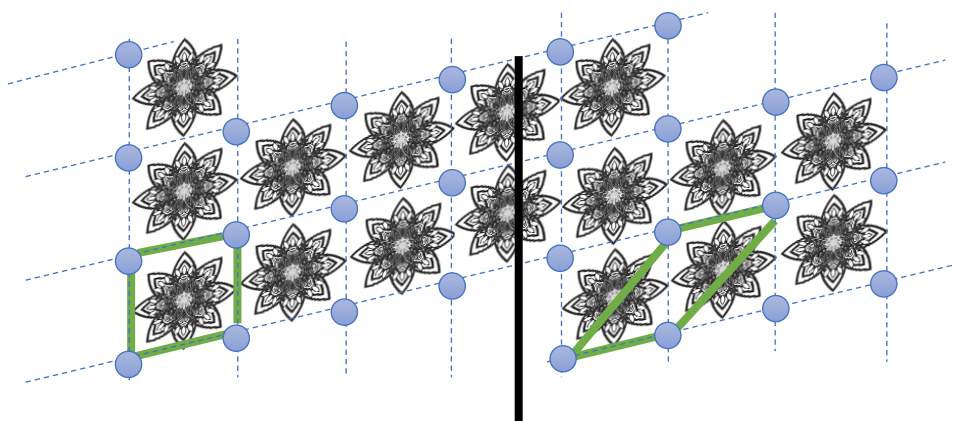


Bild 54 Einfache Beispiele von Periode innerhalb desselben Gitters mit verschiedenen Basen.

Dabei ist es auch möglich, dass die Periode anders aufgebaut ist und dennoch dieselbe Fähigkeit aufweist. Solange jeder Punkt stets dargestellt wird, wird dasselbe Gitter beschrieben trotz unterschiedlicher Basis.

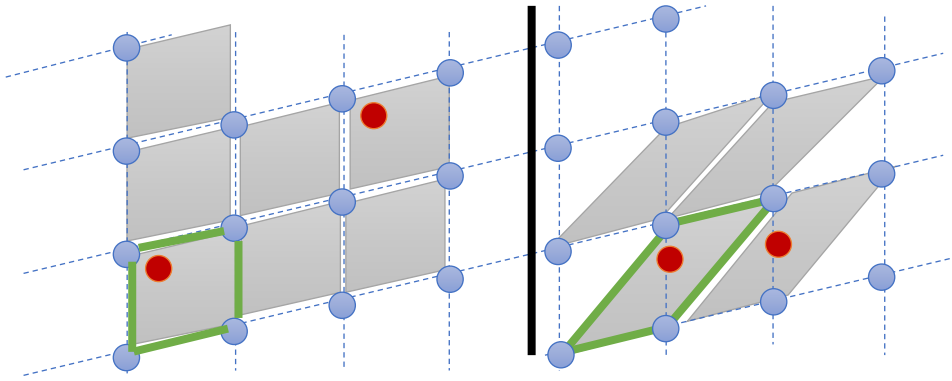


Bild 55 Alternative Betrachtungsweise einer Periode, die dennoch das gesamte Gitter, genauso wie im Beispiel zuvor, repräsentieren kann, angedeutet durch die aufgespannten Flächen. Der rote Punkt innerhalb der Gitter wiederholt sich periodisch in jedem reduzierten Teilabschnitt.

Die Berechnung einer Periode erfolgt durch die Modulo-Operation z.B.

$$\Lambda(G) = \{z_1 g_1 + \dots + z_k g_k \mid z_i \in [0,1)\}$$

So kann ein Punkt $x = z_1 g_1 + \dots + z_k g_k$ gefunden werden als

$$x \bmod \Lambda(G) = (z_1 \bmod 1)g_1 + \dots + (z_k \bmod 1)g_k.$$

Die Determinante eines Gitters $\Lambda(G)$ lautet $\det(\Lambda) = |\det(G)|$ mit

$$\det(U) = \pm 1$$

$$|\det(GU)| = |\det(G) \det(U)| = |\det(G)|$$

und beschreibt das Volumen des Parallelepipeds, definiert durch die Basis G [69]. Der Rang eines Gitters wird durch die Anzahl der Vektoren in seiner Basis definiert. Hat ein Gitter Λ in \mathbb{R}^n den Rang n , dann handelt es sich um ein full rank Gitter

$$\text{span}(\Lambda) = \mathbb{R}^n.$$

Eine spezielle Basistransformation ist die Gitterreduktion. Dabei versucht man aus einer gegebenen Basis G eine reduzierte Basis G' desselben Gitters zu erzeugen. Dabei heißt reduziert, dass

- die Basisvektoren weitestgehend orthogonal sind und
- die Länge der Basisvektoren minimal ist.

Derartiges kann man mit Hilfe von z.B. dem LLL-Algorithmus erreichen. Jedoch gibt es bisher noch keinen Algorithmus, der gleichzeitig effizient und genau ist [70, p. 101].

Auf dieser Art von Gittern gibt es schwer zu lösende Probleme, sogar für Quantencomputer. Sie sind geometrischer Natur und werden in die Kategorien

- Shortest Vector Problem, SVP,
- Closest Vector Problem, CVP,

eingeteilt. Für beide Probleme sind keine effizienten Algorithmen bekannt. Das CVP ist NP-hart. Ob SVP ebenfalls NP-hart ist, ist eine offene Frage [71, p. 35].

Beim Closest Vector Problem ist ein Vektor a gegeben, der nicht Bestandteil eines Gitters ist. Ziel ist es den am nächsten gelegenen Gitterpunkt u zu finden, so dass $|u - a|$ minimal ist.

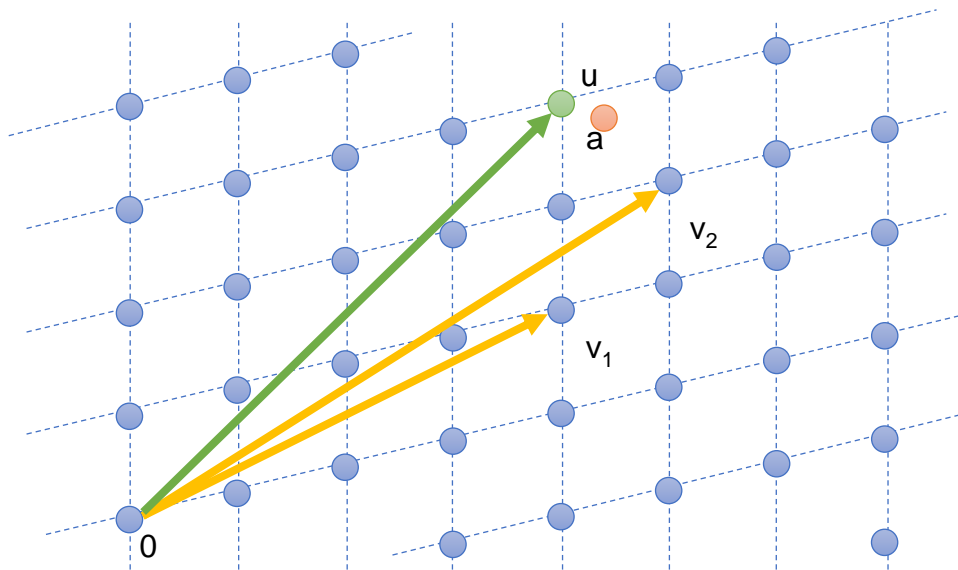


Bild 56 Beispielhafte Darstellung des Closest Vector Problem. Der Closest Vector ist grün dargestellt und Bestandteil des Gitters. Der Vector a muss das nicht zwingend sein.

Beim Shortest Vector Problem sucht man einen Gitterpunkt $v \neq 0$, der den kürzesten Abstand bezüglich einer bestimmten Norm vom Nullpunkt hat. Beispielsweise sein eine Basis V für ein Gitter mit „sehr langen“ Vektoren gegeben. Ziel ist es nun den kürzesten Vektor im Gitter zu dieser Basis zu finden. Dabei sind alle Kombinationen von v_1, \dots, v_n zu betrachten, was im zweidimensionalen Beispiel trivial ist, jedoch bei einer hohen Dimension ein Problem darstellt. Der LLL-Algorithmus berechnet eine $2^{(d-1)/2}$ -Approximation für kürzeste Vektoren in Gittern der Dimension d . Es hat sich jedoch empirisch herausgestellt, dass der LLL-Algorithmus für Gitter geringer Dimensionen, etwa $1 \leq d \leq 100$, deutlich bessere Approximationen liefert [71].

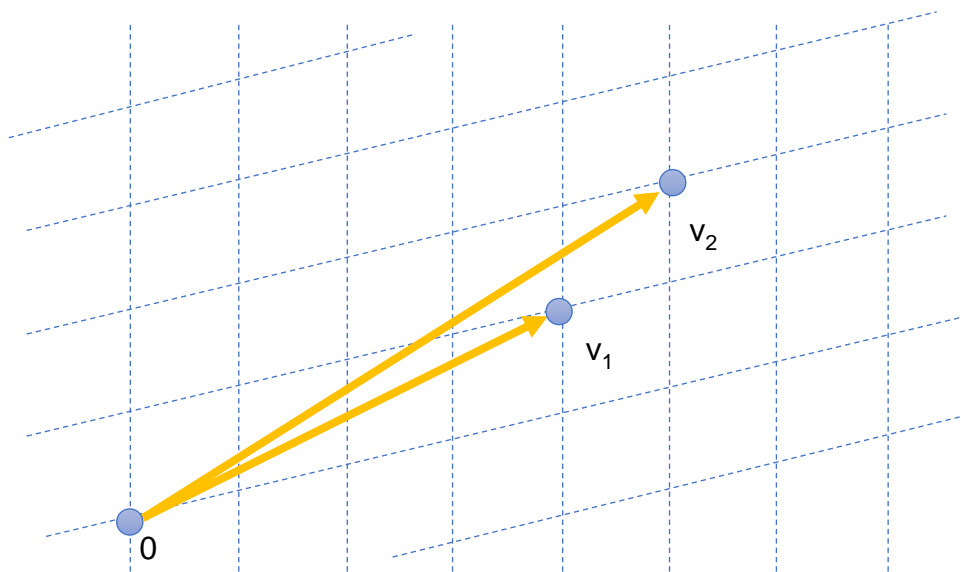


Bild 57 Problemstellung: Triviales Beispiel des Shortest Vector Problem im zweidimensionalen Raum.

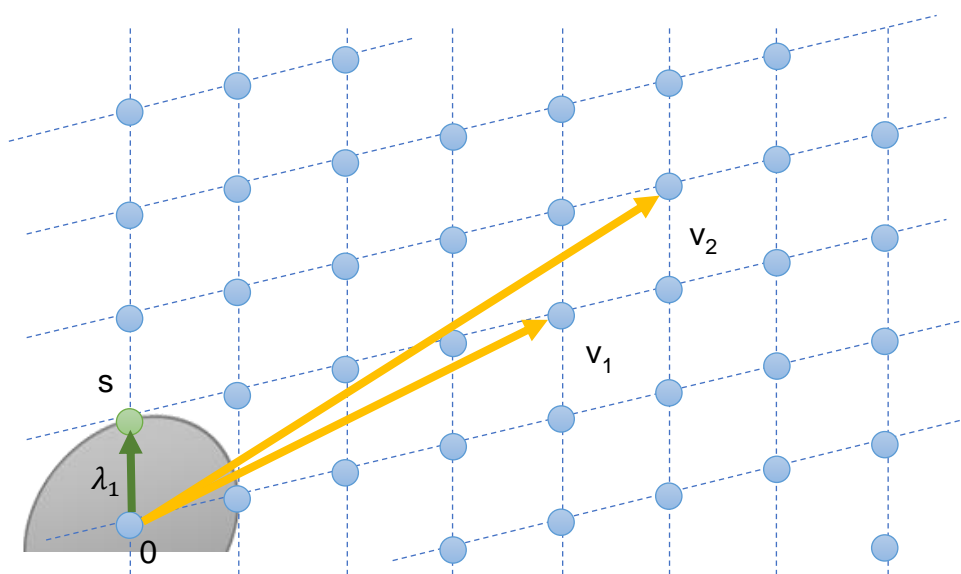


Bild 58 Lösung: Triviales Beispiel des Shortest Vector Problem im zweidimensionalen Raum mit einer eingezeichneten, möglichen minimalen Distanz, die nicht 0 ist.

Im gezeigten Beispiel stellt der Vektor s das Successive Minima dar, die minimale Distanz im beschriebene Gitter mit

$$\lambda_1(\Lambda) = \min\{\|s\|, 0 \neq s \in \Lambda.\}$$

Diese beiden schwierig zu lösenden Problemen innerhalb von mehrdimensionalen Gittern können als Basis für ein kryptographisches Verfahren verwendet werden.

A32 Babai's Closest Vector-Algorithmus

Babai's Closest Vector-Algorithmus versucht Probleme in Gittern zu lösen. Das SVP kann dabei als ein Spezialfall des CVP betrachtet werden, bei dem der gesuchte Punkt bzw. Vektor $\vec{w} = \vec{0}$ gesetzt wird um anschließend den nächsten Gitterpunkt zu suchen. Angenommen es ist zu genau diesem Punkt \vec{w} , der nicht zwangsläufig Bestandteil eines Gitters sein muss, der nächstgelegene Gitterpunkt gesucht. Ferner sei eine annähernd orthogonale Basis V mit v_1, v_2, \dots, v_k gegeben.

Es wird nun eine Linearkombination mit $a_i \in \mathbb{R}$ gesucht für

$$w = a_1 v_1 + a_2 v_2 + \dots + a_k v_k.$$

Da w nicht Bestandteil des Gitters ist muss es sich bei den Koeffizienten nicht um ganzzahlige Werte handeln. Es wird nun eine Kombination gesucht um in V einen Gitterpunkt v_i zu finden, der am nächsten an w liegt. Es wird dann, für jedes i , ein ganzzahliges b_i gesucht welches möglichst nahe an diesem a_i liegt um den entsprechenden Gitterpunkt v zu finden,

$$v = b_1 v_1 + b_2 v_2 + \dots + b_k v_k.$$

Das soll an einem zweidimensionalen Gitter beispielhaft gezeigt werden. Sei die Basis V mit $v_1 = (5,1), v_2 = (-2,8)$. Es soll der nächste Vektor zu $w = (27,8)$ gefunden werden.



Bild 59 Gesucht ist der nächste Gitterpunkt zu w der Basis V .

Dafür kann der Winkel zwischen den beiden Vektoren v_1, v_2 mit

$$\begin{aligned} \cos \theta &= \frac{v_1 v_2}{|v_1| |v_2|} \\ &= \frac{5(-2) + 1 \cdot 8}{\sqrt{5^2 + 1^2} \cdot \sqrt{(-2)^2 + 8^2}} \\ &= \frac{-2}{\sqrt{26} \sqrt{68}} \\ &\approx -0,05 \end{aligned}$$

berechnet werden. Das Ergebnis zeigt, dass die Basis nahezu orthogonal ist (vgl. Abbildung). Die Anwendung des Babai-Algorithmus findet also gute Bedingungen vor.

Nun kann die Gleichung

$$\begin{aligned} w &= a_1 v_1 + a_2 v_2 \\ (27,8) &= a_1(5,1) + a_2(-2,8) \end{aligned}$$

in einem Gleichungssystem

$$5a_1 - 2a_2 = 27,$$

$$a_1 + 8a_2 = 8$$

gelöst werden, was folgende Werte ergibt:

$$a_1 = \frac{232}{42}, \text{ gerundet } b_1 = 6$$

$$a_2 = \frac{13}{42}, \text{ gerundet } b_2 = 0.$$

Nun wird der gesuchte Gitterpunkt v mit den Werten für b berechnet:

$$v = b_1 v_1 + b_2 v_2$$

$$v = 6(5,1) + 0(-2,8)$$

$$v = (30,6),$$

was ein Vektor des Gitters, angenähert an $w = (27,8)$ ist.

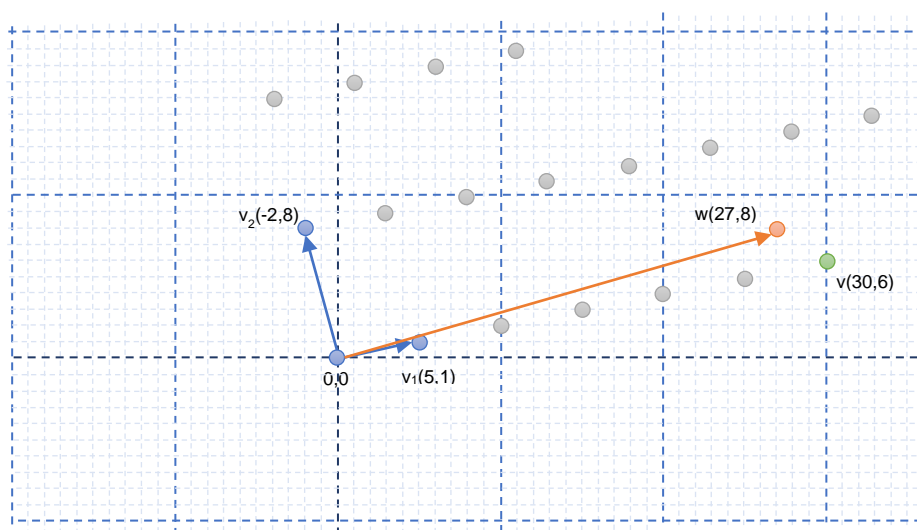


Bild 60 Angenäherte Lösung zu dem gegebenen Punkt w im Gitter.

Problematisch wird es, sobald die Basis nicht mehr annähernd orthogonal gegeben ist.

Sei diesmal die Basis U mit $u_1 = (37,41), u_2 = (103,113)$ für dasselbe Gitter. Es soll der nächste Vektor zu $w = (27,8)$ gefunden werden.

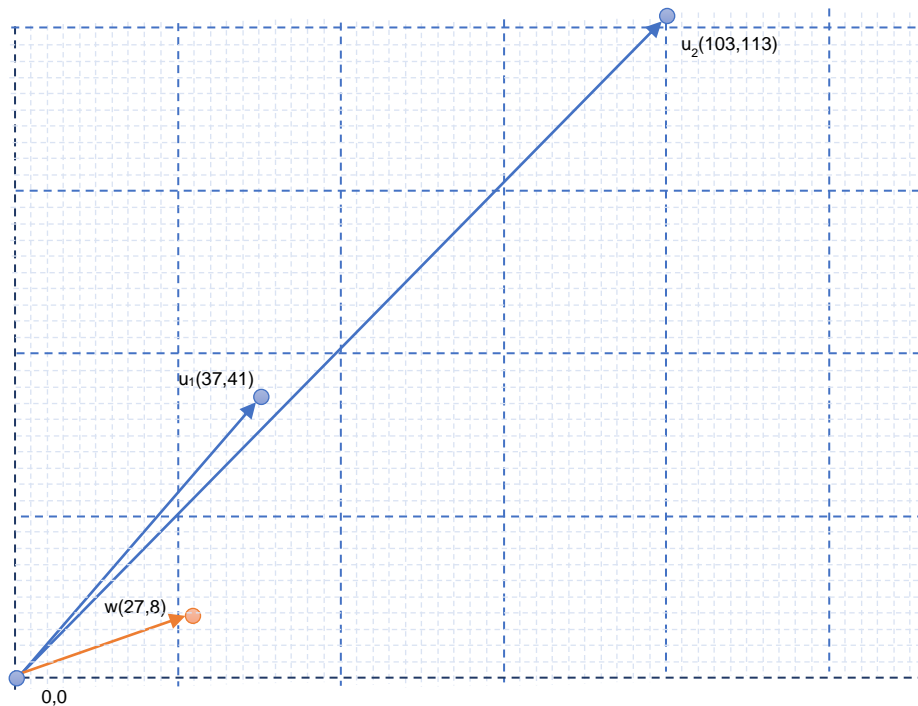


Bild 61 Darstellung desselben Gitters jedoch mit der Basis U , die nicht annähernd orthogonal ist.

Erneut wird die Gleichung berechnet

$$w = a_1 u + a_2 u$$

$$(27,8) = a_1(37,41) + a_2(103,113), \text{ gibt}$$

$$a_1 \approx -53,$$

$$a_2 \approx 19.$$

$$x = -53(37,41) + 19(103,113)$$

$$x = (-4, -26).$$

Dieser Punkt x ist jedoch weiter entfernt von w als der zuvor gefundene Punkt v . Zur Erinnerung, zwei Basen, hier U und V , beschreiben dasselbe Gitter L ,

wenn eine Matrix M mit $\det(M) = \pm 1$ existiert, d.h. $V = MB$. Das bedeutet, dass es eine Linearkombination der „alten“ Basis V gibt um U zu berechnen,

$$u_k = a_{k1}v_1 + \dots + a_{kn}v_n.$$

$$V = \begin{pmatrix} 5 & 1 \\ -2 & 8 \end{pmatrix}$$

Ziel ist es nun eine Matrix M zu finden, bei dem die Determinante $\det(M) = \pm 1$ ist und V in U überführt.

$$\begin{pmatrix} 9 & 4 \\ 25 & 11 \end{pmatrix} \begin{pmatrix} 5 & 1 \\ -2 & 8 \end{pmatrix} = \begin{pmatrix} 37 & 41 \\ 103 & 113 \end{pmatrix} = U$$

Somit ist $M = \begin{pmatrix} 9 & 4 \\ 25 & 11 \end{pmatrix}$ die gesuchte Matrix mit

$$\begin{aligned} \det(M) &= (9 \cdot 11 - 4 \cdot 25) \\ &= (99 - 100) \\ &= -1, \end{aligned}$$

was die Bedingung für M erfüllt (Beispiele nach [72]). Darauf aufbauend entwickelten Oded Goldreich, Shafri Goldwasser und Shai Halevi im Jahr 1997 das nach ihnen benannte GGH-Kryptosystem.

A33 Das GGH-Kryptosystem

Alice wählt eine Basis V mit k - nahezu orthogonalen Vektoren und behält diese Basis geheim. Des Weiteren formt Alice eine weitere Basis U , diesmal jedoch mit nahezu parallel verlaufenden Vektoren und veröffentlicht diese Basis als ihren öffentlichen Schlüssel. Sie hat nun ein Gitter L , welches durch diese beiden Basen beschrieben wird,

$$U = MV.$$

Um Alice nun eine Nachricht $M = (m_1, \dots, m_k)$ zu schicken verwendet Bob m_1, \dots, m_k als Koeffizienten zusammen mit der von Alice veröffentlichten Basis U . Darüber hinaus wählt Bob einen zufälligen Vektor r , auch als Rauschen, Noise oder Error bezeichnet, und sendet $M' = M + r$ an Alice,

$$M' = \sum_{i=0}^k (m_i)u_i,$$

$$M' = M \cdot U.$$

Die verschlüsselte Nachricht C definiert sich dann als

$$C = M' + r \text{ bzw.}$$

$$C = M \cdot U + r.$$

Alice verfügt über die „gute“, nahezu senkrechte Basis V und ist so in der Lage die Nachricht M wiederherzustellen. Ein Angreifer Mallory steht jedoch nur schlechte Basis U zur Verfügung. Wie im sehr einfachen Beispiel vorher gezeigt, ist Mallory, insbesondere bei der Wahl großer Dimensionen, nicht in der Lage dieses Problem effizient zu lösen. Er wird sehr wahrscheinlich eine Nachricht M'' berechnen, die keine Ähnlichkeit mit M hat.

Beispiel: Gewählt wird die Basis V des Gitters L mit $v_1 = (1,45)$ und $v_2 = (45, -1)$. Der Winkel zwischen den beiden Vektoren v_1, v_2 lautet

$$\begin{aligned} \cos \theta &= \frac{v_1 v_2}{|v_1||v_2|} \\ &= \frac{1(45) + 45(-1)}{\sqrt{1^2 + 45^2} \cdot \sqrt{(45)^2 + (-1)^2}} \\ &= 0, \end{aligned}$$

somit stehen die Vektoren orthogonal zueinander. Innerhalb des Gitters L soll nun ein solcher Punkt r bestimmt werden. Das geschieht ausgehend von einem Gitterpunkt P .

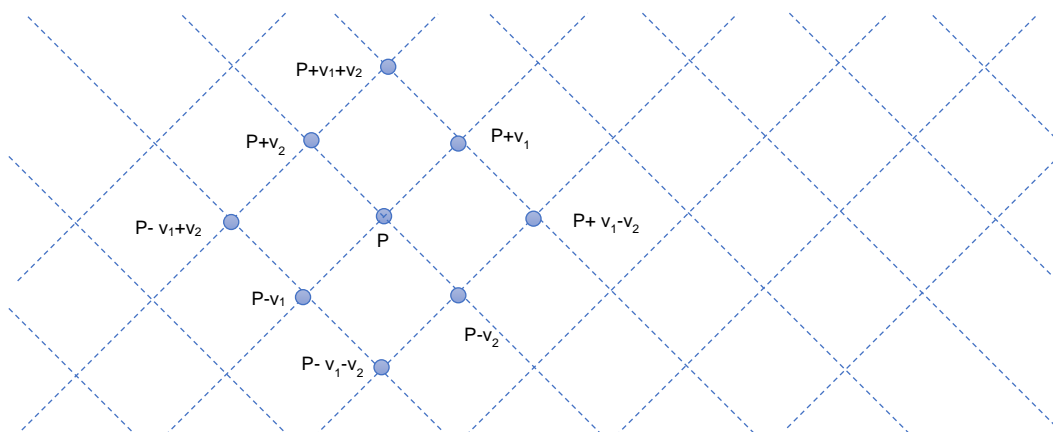


Bild 62 Exemplarische Darstellung des Gitters L sowie dem Gitterpunkt P und seinen nächsten Nachbarn.

Der Punkt r soll nun so gewählt werden, dass er P näher ist als jedem anderen Punkt im Gitter. Aufgrund der Basis V bietet es sich an, dass

$$|r| < 20$$

gewählt wird. Somit ist er stets näher an P als an jedem anderen Punkt im Gitter.

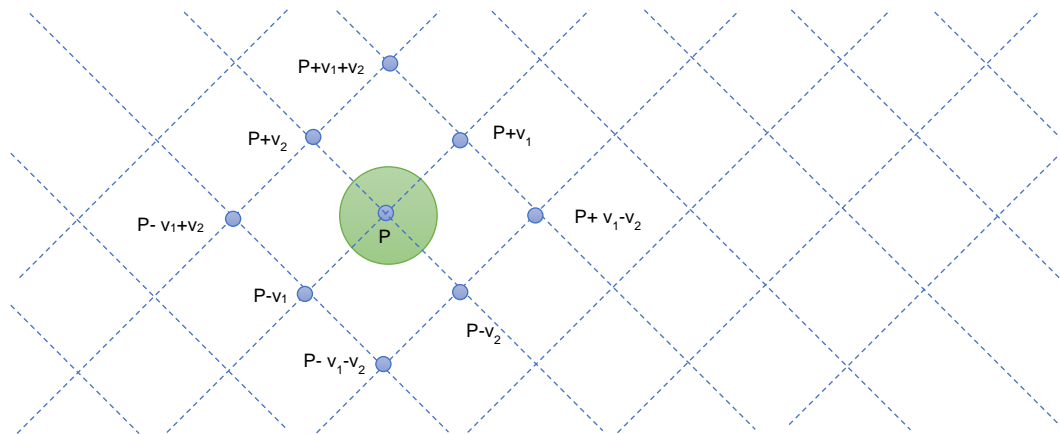


Bild 63 Mit $|r| < 20$ befindet sich r innerhalb des grünen Kreises und somit P am nächsten.

Ein mögliches $r(r_x, r_y)$ mit $-10 \leq r_x, r_y \leq 10$ lautet $r = (-9, 2)$.

Als nächstes wird eine neue, möglichst parallele Basis U gesucht. Dafür wird eine Matrix B mit $\det(M) = \pm 1$ geformt.

$$\begin{pmatrix} 1 & 5 \\ 8 & 41 \end{pmatrix} \begin{pmatrix} 1 & 45 \\ 45 & -1 \end{pmatrix} = \begin{pmatrix} 226 & 40 \\ 1853 & 319 \end{pmatrix} = U$$

Der Winkel zwischen beiden Vektoren lautet

$$\cos \theta_{u_1 u_2} \approx 0,999989,$$

was nahezu 1° entspricht. Somit verlaufen die Vektoren parallel. So ergibt sich $u_1 = (226, 40), u_2 = (1853, 319)$ als öffentliche Basis.

Die zu verschlüsselnde Nachricht M lautet $M = (35, 27)$. Erneut wird sie als Linearkombination mit der öffentlichen Basis betrachtet

$$m = 35u_1 + 27u_2,$$

$$m = 35(226, 40) + 27(1853, 319),$$

$$m = (57941, 10013),$$

was nun der Nachrichten-Vektor ist. Nun wird das Rauschen r auf die Nachricht addiert,

$$c = (57941,10013) + (-9,2) = (57932,10015).$$

Alice kennt die orthogonale Basis V und kann somit unter idealen Bedingungen den Babai-Algorithmus anwenden, um den nächstgelegenen Punkt im Gitter zu finden.

$$a_1(1,45) + a_2(45,-1) = (57932,10015),$$

$$a_1 \approx 251,04$$

$$a_2 \approx 1281,799$$

gibt

$$251(1,45) + 1282(45,-1) = (57941,10013) = m.$$

Alice besitzt sowohl den privaten als auch den öffentlichen Schlüssel und kann nun die Nachricht M gewinnen.

$$m_1(226,40) + m_2(1853,319) = (57941,10013)$$

$$m_1 = 35$$

$$m_2 = 27$$

$$M = (35,27)$$

Da Mallory nicht in der Lage ist das Rauschen r mit Hilfe der „guten“ Basis rauszurechnen muss er den Babai-Algorithmus auf dem Verschlüsselten Wert c anwenden.

$$m_1(226,40) + m_2(1853,319) = (57932,10015)$$

$$m_1 \approx 38,26$$

$$m_2 \approx 26,599$$

$$M' = (38,27) \neq M$$

Im hier gezeigten 2-dimensionalen Raum ist es einfach durch probieren den nächstgelegenen Gitterpunkt zu finden. Bei 250 Dimensionen wären es jedoch 2^{250} Nachbar-Punkte und es wäre nicht mehr möglich durch Probieren den nächsten Nachbarn zu finden. Das hier gezeigte GGH-Verfahren hat sich jedoch als unsicher erwiesen und kommt als PQC-Methode nicht in Frage. Dennoch vermittelt es einen Eindruck wie derartige Probleme, beruhend auf Gittern, kryptographisch verwendet werden können [63].

A34 NTRUEncrypt

Gitterbasierte Verfahren unterstützen sowohl den Schlüsselaustausch als auch das Signieren von Nachrichten. Ein Verfahren aus beiden Bereichen ist das patentierte Verfahren NTRU (n-truncated, dt. *n-abgeschnitten*), von der gleichnamigen Firma. Das Patent läuft am 01. Dezember 2023 aus⁶⁰. Das Verfahren wurde von Jeffrey Hoffstein, Jill Pipher und Joseph Silvermann entwickelt und liegt als praxistaugliche Implementierung am Markt vor. Noch steht es in der Verbreitung deutlich hinter RSA, DL- und ECC-Verfahren hinten an, was sich jedoch mit der wachsenden Bedrohung durch Quantencomputer wohl künftig verändern wird. NTRUEncrypt ist bisher nicht durch den Einsatz mittels Quantencomputer-Algorithmus gebrochen worden.

Tabelle 23 Auswahl des Status aktueller kryptographischer Verfahren bezüglich des Quantencomputers nach [59, p. 16].

Cryptosystem	Gebrochen durch QC-Algorithmus?
RSA Public Key Encryption	Gebrochen
DH-Schlüsselaustausch	Gebrochen
Elliptic Curve Cryptography ECC	Gebrochen
McEliece Public Key Encryption	Bisher nicht gebrochen
NTRU Public Key Encryption	Bisher nicht gebrochen
Lattice-based Public Key Encrypt.	Bisher nicht gebrochen

Das Verfahren splittet sich in NTRUEncrypt, was im Bereich der asymmetrischen Verschlüsselung zum Einsatz kommt, und NTRU-Sign, ein Signaturverfahren. Beide Verfahren weisen jedoch deutliche Unterschiede auf. Das Verfahren ist ein ringbasiertes Kryptosystem und kann auch über die Gittertheorie beschrieben werden, auf dem es lose basiert. NTRU verwendet Polynome der Form $f(x) = a_{N-1}x^{N-1} + a_{N-2}x^{N-2} + \dots + a_1x^1 + a_0$. Es handelt sich um einen

⁶⁰ <https://patents.google.com/patent/US7031468>

Ring R , dessen Grad des Polynoms niemals N übersteigt. Die Berechnungen erfolgen im Ring

$$R_{N,q} = \frac{Z_q[x]}{x^{N-1}}.$$

Die Teilung durch das Polynom x^{N-1} verhindert, dass bei der Multiplikation zweier Polynome ein Grad größer als N entstehen kann (Dieses Vorgehen gibt dem Verfahren seinen Namen) [5]. Das Verfahren wird mit p, q und $ggT(p, q) = 1$ parametrisiert, wobei q deutlich größer ist als p und beide nicht notwendigerweise Primzahlen sein müssen. Typische Werte sind $q = 2^8 = 256, p = 3, N = 503$, was dem höchsten Sicherheitslevel entspricht. Für NTRUEncrypt werden Mengen der Form $L(d_1, d_2)$ verwendet, bei denen ein Polynom in R über $d_1 = 1$ sowie $d_2 = -1$ verfügt. Alle restlichen Koeffizienten sind Null.

Für die Erstellung eines zufälligen privaten Schlüssels, Polynom f mit $f = [f_0, \dots, f_{N-1}]$, ist es nötig, dass die inversen Polynome $f_p^{-1}, f_q^{-1} \bmod p$ und $\bmod q$ existieren, sodass $f \cdot f_p^{-1} = 1$ und $f \cdot f_q^{-1} = 1$ gilt. Können weder f_p^{-1} noch f_q^{-1} erzeugt werden muss ein anders Polynom f generiert werden.

$$f \in L(d_f, d_f - 1)$$

Der öffentliche Schlüssel h benötigt ebenfalls ein zufälliges Polynom g mit $g = [g_0, \dots, g_{N-1}]$ und wird mit

$$h \equiv p f_q^{-1} * g \pmod{q},$$

$$g \in L(d_g, d_g)$$

berechnet. Diese Einwegfunktion macht das Verfahren sicher. Jedoch ist diese Eigenschaft nicht bewiesen und nur eine begründete Vermutung [5].

Privater Schlüssel: (f, f_p^{-1})

öffentlicher Schlüssel: $(1, h)$

Beispiel

Alice wählt die öffentlichen Parameter $(N, p, q, d) = (7, 3, 41, 2)$. Nun wählt sie zufällig $f \in L_f$ als

$$\text{Privater Schlüssel: } f(x) = x^6 - x^4 + x^3 + x^2 - 1$$

sowie $g \in L_g$ als

$$g(x) = x^6 + x^4 - x^2 - x.$$

Sie berechnet die Inversen von f bezüglich p, q

$$f_q^{-1}(x) \bmod q = 8x^6 + 26x^5 + 31x^4 + 21x^3 + 40x^2 + 2x + 37 \in R_q,$$

$$\text{Privater Schlüssel: } f_p^{-1}(x) \bmod p = x^6 + 2x^5 + x^3 + x^2 + x + 1 \in R_p,$$

$$\rightarrow \text{Privater Schlüssel: } (f(x), f_p^{-1}(x)).$$

Berechnen des öffentlichen Schlüssels mit

$$h(x) \equiv p f_q^{-1}(x) \pmod{q} * g \pmod{q},$$

$$h(x) = 19x^6 + 38x^5 + 6x^4 + 32x^3 + 24x^2 + 37x + 8 \in R_q.$$

Eine von Bob verschickte Nachricht m wird aus einer Menge an Klartexten L_m gewählt. L_m besteht aus allen Polynomen in R mit Koeffizienten zwischen $-(p-1)/2$ und $(p-1)/2$ als $L(d_m, d_m)$. Darüber hinaus generiert Bob ein zufälliges Polynom r mit $r \in L(d_r, d_r)$, auch „Blendpolynom“ [5, p. 192]. Er berechnet so den Geheimtext c mit

$$c \equiv r * ph + m \pmod{q}.$$

Die zu verschickende Nachricht m , dargestellt als Polynom $m(x)$, lautet

$$m(x) = -x^5 + x^3 + x^2 - x + 1.$$

Das Blendpolynom r lautet

$$r(x) = x^6 - x^5 + x - 1.$$

Damit ist der Ciphertext c

$$c(x) = 31x^6 + 19x^5 + 4x^4 + 2x^3 + 40x^2 + 3x + 25 \pmod{41}$$

Aufgrund des verwendeten Zufalls können aus demselben Eingabetext verschiedene Ciphertexte generiert werden.

Für die Entschlüsselung von c wird zunächst in einem Zwischenschritt a berechnet:

$$a \equiv f * c \pmod{q}$$

$$b \equiv a \pmod{p}$$

Alice berechnet somit

$$a \equiv x^6 + 10x^5 - 8x^4 - x^3 - x^2 + x - 40 \pmod{41}$$

Anschließend wird b berechnet

$$b \equiv a \pmod{p} \equiv x^6 + 10x^5 - 8x^4 - x^3 - x^2 + x - 40 \pmod{3}.$$

Die Koeffizienten von a werden normalerweise im Intervall $[-q/2, q/2)$ gewählt. Anschließend wird der Klartext m mit

$$m \equiv f_p^{-1} * b \pmod{p}$$

berechnet.

Nun wird $b(x)$ um *modulo* p reduziert. Mit

$$c \equiv f_p^{-1}(x) * b(x) \equiv 2x^5 + x^3 + x^2 + 2x + 1 \pmod{3}$$

kann nun das Klartext-Polynom

$$m(x) = -x^5 + x^3 + x^2 - x + 1$$

berechnet werden (Beispiel nach [73]). Das Ganze funktioniert, da

$$a \equiv f * c \pmod{q}$$

$$a \equiv f * (r * h + m) \pmod{q}$$

$$a \equiv f * (r * pf_q^{-1} * g + m) \pmod{q}$$

$$a \equiv f * (r * pf_q^{-1} * g) + f * m \pmod{q}$$

$$a \equiv pr * g + f * m \pmod{q} \in R$$

Die Koeffizienten von r, g, f sind sehr klein gewählt und viele sind Null. Auch ist der Parameter p klein gewählt. Die Koeffizienten im Polynom für a liegen wahrscheinlich im Intervall $[-q/2, q/2)$. $pr * g$ ist ein Vielfaches von p und somit $pr * g \pmod{p} = 0$. Daher kann mit $f_p^{-1} * a$ die Nachricht m wiederhergestellt werden [74], [68], [75], [5], [59].

Tabelle 24 Empfohlene Parameterwahl für verschiedene Stufen der Sicherheit. Die Schlüsselgrößen sind in Bit dargestellt, nach [74, p. 17].

Niveau	N	p	q	d _f	d _g	d _r	f + f _p ⁻¹	h
Moderat	167	3	128	61	20	18	530	1169
Standard	263	3	128	50	24	16	834	1841
Hoch	503	3	256	216	72	55	1595	3521

A35 Codierung mit linearen Codes

Codebasierte kryptographische Verfahren sind ebenfalls im Bereich der Verschlüsselung und des Schlüsselaustausches angesiedelt. Sie sind der älteste Zweig der PQC-Verfahren. Beispielsweise wurde das McEliece Kryptosystem bereits im Jahr 1978 vorgestellt. Die zugrundeliegende Idee ist, dass es für einige spezielle lineare Codes effiziente Fehlerkorrekturalgorithmen gibt, die als kryptographische Primitive (mit zugrundeliegende Einweg-Funktion) genutzt werden können. Das Dekodieren von fehlerbehafteten Codes, ohne das zugehörige Geheimnis, ist im allgemeinen Fall nicht lösbar. Für einen Angreifer ist es daher nicht möglich die Nachricht, ohne den privaten Schlüssel zu decodieren. Eine Nachricht m ist somit im Allgemeinen ein Codewort c mit bestimmtem Fehler e [76]. Das Codewort c wird mittels Generator G erzeugt.

Historisch kommen fehlerkorrigierende Codes aus der Untersuchung ob Übertragungsfehler bezüglich Informationen auf einem stör anfälligen Kanal vorliegen und ob diese korrigiert werden können. Hier wurden Nachrichten mit speziellen Codes codiert, die bestimmte Eigenschaften erfüllten. Durch das Ausnutzen von Eigenschaften der Codierung können Fehler erkannt und korrigiert werden. Im Jahr 1978 wurde der Versuch des Entfernens von Fehler einer kodierten Nachricht ohne Kenntnis des zu Grunde liegenden Codes als NP-vollständig bewiesen. Zeitgleich zum RSA-Verfahren konnte das Decodierungsproblem als Falltür-Funktion im McEliece-Verfahren verwendet werden. Bis heute wurde kein existentieller Angriff dagegen gefunden. Das Verfahren hat eine starke Sicherheitsbeziehung und eine hohe Effizienz, Ver- und Entschlüsselungsvorgänge sind mit niedrigem Rechenaufwand zu bewerkstelligen. Dennoch hat es aktuell keine nennenswerte Relevanz aufgrund von sehr großen öffentlichen Schlüsseln, die mittels (binärem) Goppa-Code ca. 1 Megabyte Größe haben [77]. Goppa-Code ist ein Typ von linearem Code, die u. a. die Eigenschaft der Fehlererkennung und -korrektur ausweisen. Bekannte Vertreter der codebasierten Kryptographie sind das McEliece- und das Nie-

derreiter-Kryptosystem. Durch besser strukturierte Codes konnten die Schlüsselgrößen reduziert werden z.B. im QC-MDPC. Das geschah jedoch auf Kosten der Sicherheit [78].

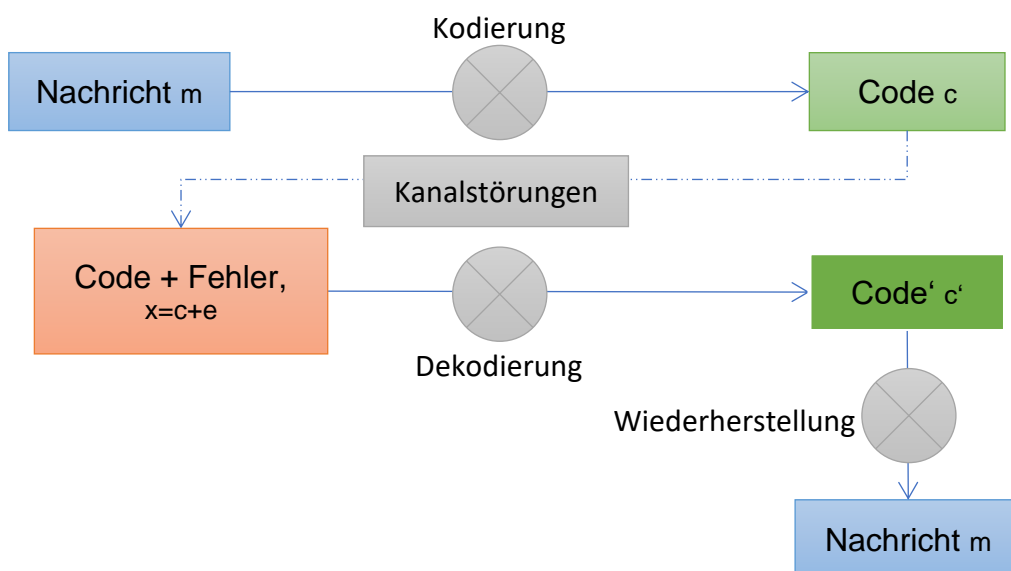


Bild 64 Schematische Darstellung der Übertragung einer Nachricht über einen fehlererzeugenden, rauschenden Kanal. Ziel ist die Decodierung und Wiederherstellung der ursprünglichen Nachricht. Typischerweise wird das durch das Hinzufügen von Redundanz erreicht.

Ein simples Beispiel für das Hinzufügen von Redundanzen sind Paritätsbits. Auf sieben Bits folgt ein achttes Bit, das dafür sorgt, dass im übertragenen Block eine gerade Anzahl an Einsen stehen, z.B.

$$100101 \rightarrow 100101 1$$

$$111100 \rightarrow 111100 0.$$

Derartige Paritäten detektieren jedoch einen Fehler nur, sie korrigieren ihn nicht.

Auch an codebasierte kryptographische Verfahren zum Verschlüsseln von Informationen als asymmetrisches Verschlüsselungsverfahren gibt es die drei Anforderungen der Erzeugung beider Schlüssel, einen Algorithmus zum Ver-

und Entschlüsseln. Selbstverständlich darf eine Entschlüsselung ohne einen entsprechenden privaten Schlüssel nicht möglich sein.

Lineare Codes sind einfach zu codieren und zu decodieren. Ihr Codealphabet ist ein endlicher Körper \mathbb{F} . Codewörter sind ein Element aus dem Vektorraum \mathbb{F}^n , wobei n für die Länge des Codewortes steht.

Tabelle 25 Vier verschiedene Codierungen von u mit Codewörtern der Teilmenge des Vektorraums \mathbb{Z}_2^3 . Die dargestellten Beispiele erzeugen eine gerade Anzahl an Einsen in ihren Codewörtern bezüglich des Ausgangswertes u .

u	$c_1(u)$	$c_2(u)$	$c_3(u)$	$c_4(u)$
00	000	000	110	000
01	011	101	101	001
10	101	011	011	010
11	110	110	000	100

$$\left\{ \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \right\}$$

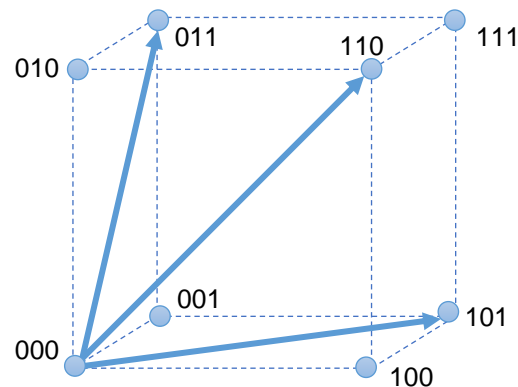


Bild 65 Die von c_1, c_2, c_3 erzeugten Codes sind ihrerseits ein eigener Untervektorraum. Diese Eigenschaft nennt man lineare Codierung [79, pp. 188-189].

Ein linearer (n, k, d) -Code C über einen endlichen Vektorraum \mathbb{F} ist ein k -dimensionaler Unterraum von \mathbb{F}^n mit einer minimalen Distanz $d = \min_{x \neq y \in C} \text{dist}(x, y)$ wenn dist die Hamming-Distanz ist, wobei ein (n, k, d) -Code t -Fehler mit $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ Fehlern beheben kann (es können $d - 1$ Fehler erkannt werden) [80, p. 7].

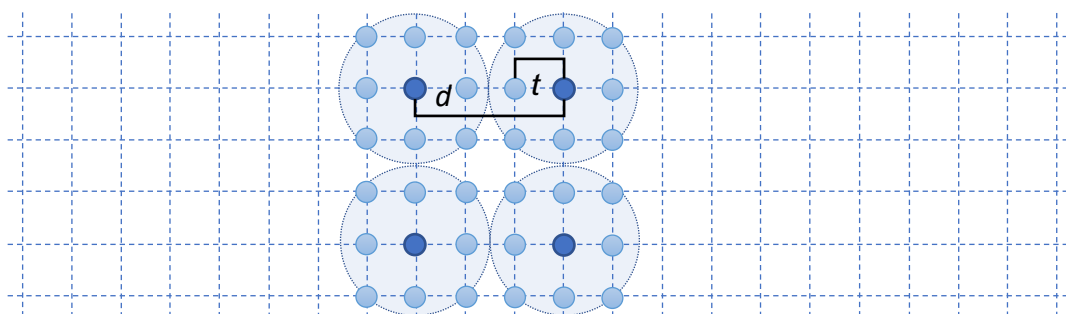


Bild 66 Schematische Darstellung linearer Codes im zweidimensionalen Raum mit der Distanz d und der möglichen Fehlerkorrektur t nach [80].

Untervektorräume lassen sich mit Hilfe von Generatormatrizen erzeugen. Eine Generatormatrix G wird durch zeilenweise Notation der Vektoren der Basis $\{b_1, \dots, b_k\}$ erstellt mit der Länge der Codewörter n . Die daraus resultierende Matrix G verfügt über k Zeilen und n Spalten.

Die Matrix $G \in \mathbb{F}^{k \times n}$ ist eine Generatormatrix für einen linearen (n, k, d) –Code C , wenn $C = \langle G \rangle$.

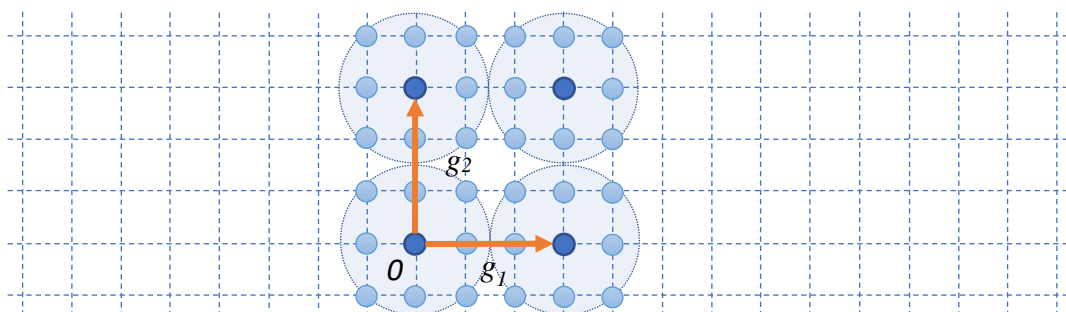


Bild 67 Darstellung der Generatormatrix G im zweidimensionalen Raum mit der Basis für alle markieren Codewörter (dunkelblau).

Eine Nachricht $m \in \mathbb{F}^k$ wird als $c = mG \in \mathbb{F}^n$ codiert.

Beispiel: Für die Codierung der Nachricht u in das Codewort c_1 aus dem zuvor aufgeführten Beispiel wird eine Generatormatrix G mit

$$G_1 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

benötigt und berechnet, so dass für die Nachrichten u (00), (01), (10), (11) die Codewörter $c_1(u)$

$$(00) \cdot \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = (000),$$

$$(01) \cdot \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = (011),$$

$$(10) \cdot \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = (101),$$

$$(11) \cdot \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = (110)$$

gebildet werden. Dasselbe gilt für die Codewörter $c_2(u), c_3(u)$. Es ist jedoch nicht möglich für $c_4(u)$ eine Generatormatrix G_4 zu finden.

Tritt nun bei der Übertragung einer Nachricht m als Codewort $c(m)$ eine Störung im Kanal auf, so wird ein Codewort $c(m)$ durch einen Fehler e verändert,

$$x = c(m) + e.$$

Der Fehler e darf höchstens t sein, wenn x noch eindeutig bestimmbar sein soll.

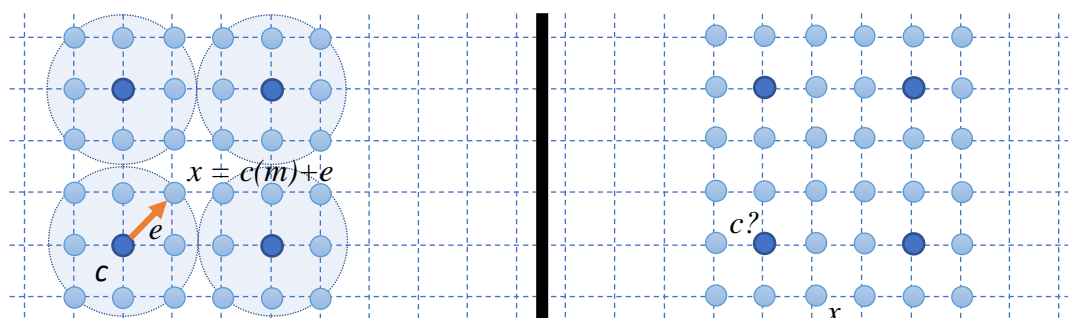


Bild 68 Der Punkt x ist eine Abfälschung des eigentlichen Wertes c um den Fehler e .

Das Problem der (allgemeinen) Decodierung ist NP-hart [80].

Beispiel: Sei $G \in \mathbb{F}^{k \times n}$ eine Generatormatrix und $H \in \mathbb{F}^{(k-n) \times k}$ eine Matrix für einen Paritätscheck mit $(n, k, d) = (7, 4, 3)$. I ist die Einheitsmatrix, P eine angefügte Matrix.

$$G = (I^k | P_2^{(k \times (n-k))}) = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix},$$

$$H = (-P^t | I^{(n-k)}) = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

$G \in \mathbb{F}_2^{4 \times 7}$ ist Generator des Codes c , $H \in \mathbb{F}_2^{3 \times 7}$ für den Paritätscheck $GH^t = 0$. Die Nachricht lautet $m = (0101) \in \mathbb{F}_2^{k=4}$.

$$\begin{aligned} c = mG &= (0101) \cdot \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \\ &= (0101110) \end{aligned}$$

$cH^t = 0$ genau dann, wenn $c \in C$.

Während der Übertragung von c kommt es zu einem Ein-Bit Fehler $e = (0001000)$, welcher c zu x abfälscht mit $x = c + e$,

$$x = c + e = (0100110), \text{ dann}$$

Den Vektor e bezeichnet man als Fehlervektor und enthält genau dort die Ziffer 1 wo der Fehler in der Übertragung aufgetreten ist. Durch Addition von e kann das Ursprüngliche c wiederhergestellt werden. Jedoch sind zunächst weder c noch e bekannt. Der empfangene Wert für x muss ein Codewort aus C sein, denn der Code C ist als Untervektorraum die Lösungsmenge. Der Paritätscheck mit der Prüfungsmatrix H und $GH^t = 0$ wird hierfür verwendet. Um auf einen Fehler festzustellen wird geprüft ob

$$x \in C \leftrightarrow xH^t = 0,$$

d.h. die Multiplikation bringt den Nullvektor hervor.

Um den Fehler zu korrigieren werden alle möglichen Fehler (hier Ein-Bit Fehler) eH^t berechnet und mit xH^t verglichen. Für die Fehlerkorrektur wird ein Fehlersyndrom s von x ermittelt. Für s gilt

$$s = xH^t \in \mathbb{F}_2^{n-k}.$$

Aus $x = c + e$ folgt

$$xH^t = cH^t + eH^t = eH^t = s, \text{ da}$$

$$cH^t = 0 \text{ genau dann, wenn } c \in C.$$

Löst man das lineare Gleichungssystem

$$eH^t = s$$

kann c ermittelt werden [81]. Ist das Syndrom der Nullvektor, dann wurde $x = c$ korrekt übertragen. Es werden nun alle möglichen Syndrome (3-Bit Kombinationen) in einer reduzierten Fehlerkorrekturtabelle (Syndromtabelle) betrachtet. Ferner gilt auch für den Fehler e multipliziert mit der transponierten Paritätsmatrix $xH^t = eH^t = s$. Beispiel-Berechnungen für die Fehler $e_1(1,0,0,0,0,0,0), e_2(0,1,0,0,0,0,0)$ lauten

$$s = e_1H^t = (1000000) \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = (110),$$

$$s = e_2 H^t = (0100000) \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = (011).$$

Für alle weiteren Fehler ergibt sich folgende Kombination tabellarisch dargestellt.

Tabelle 26 Syndromtabelle: Relation möglicher Ein-Bit Fehler und Syndrome tabellarisch dargestellt.

$s = xH^t, eH^t$	Fehler e_i	x	$c = x + e$
000	(0,0,0,0,0,0,0)	Übertragung erfolgte ohne Fehler	
001	(0,0,0,0,0,0,1)		
010	(0,0,0,0,0,1,0)		
011	(0,1,0,0,0,0,0)		
100	(0,0,0,0,1,0,0)		
101	(0,0,0,1,0,0,0)		
110	(1,0,0,0,0,0,0)		
111	(0,0,1,0,0,0,0)		

Im obenstehenden Beispiel wurde nach Auftreten von $e = (0001000)$ der Wert $x = c + e = (0100110)$ empfangen. Das Syndrom $s(x)$ berechnet sich als

$$s = xH^t = (0100110) \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = (101)$$

und weist somit in der Tabelle denselben Fehler auf, wie der wesentlich einberechnete. Durch eine anschließende erneute Addition des Fehlers wird dieser korrigiert und der ursprüngliche Code c wiederhergestellt.

Tabelle 27 Syndromtabelle: Relation möglicher Ein-Bit Fehler und Syndrome tabellarisch dargestellt.

$s = xH^t, eH^t$	Fehler e_i	x	$c = x + e$
000	(0,0,0,0,0,0,0)	Übertragung erfolgte ohne Fehler	
001	(0,0,0,0,0,0,1)		
010	(0,0,0,0,0,1,0)		
011	(0,1,0,0,0,0,0)		
100	(0,0,0,0,1,0,0)		
101	(0,0,0,1,0,0,0)	(0,1,0,0,1,1,0)	(0,1,0,1,1,1,0)
110	(1,0,0,0,0,0,0)		
111	(0,0,1,0,0,0,0)		

Damit gilt $cH^t = 0$ da nun $c \in C$ und $m = (0101)$.

A36 McEliece-Kryptosystem

Das McEliece-Kryptosystem ist ein asymmetrisches Verfahren von dem ausgegangen wird, dass es sicher gegenüber dem Einsatz von Quantencomputern ist. Es basiert ebenfalls weder auf dem Faktorisierungs- noch auf dem diskreten Logarithmus-Problem. Es wurde 1978 von Robert J. McEliece vorgestellt. Auch dieses Verfahren arbeitet mit fehlerkorrigierenden Codes, dem Goppa-Code. Diese sind einfach zu decodieren und werden im Verfahren auf einen linearen Code transformiert. Die Decodierung von allgemeinen linearen Codes ist ein NP-vollständiges Problem. Das System wird formal durch das Tripel (n, k, d) mit

- n : Blocklänge des Ciphertextblocks
- k : Blocklänge des Klartextblocks
- d : Minimaler Hamming-Abstand des Codes C

beschrieben. Auch hier wird die Anzahl der zu korrigierenden Fehler t des Codes C mit $t = \lfloor \frac{d-1}{2} \rfloor$ bestimmt.

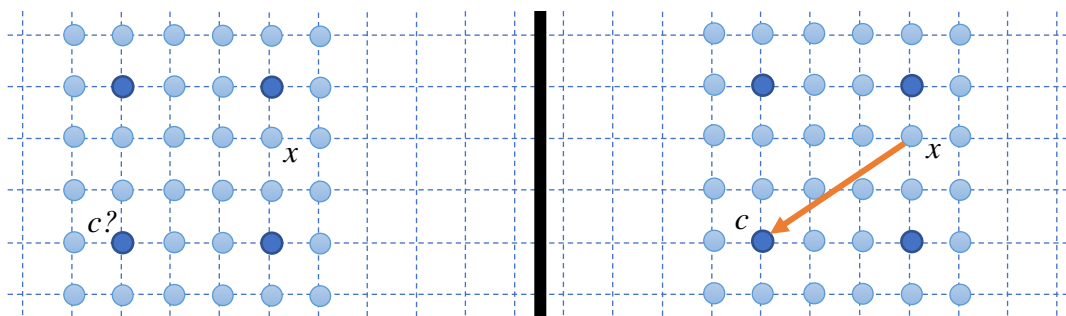


Bild 69 Schematische Darstellung eines Codes, bei dem x nicht aus dem nächsten Codewort c abgebildet ist. Eine solche Information kann als Falltür für eine Verschlüsselung verwendet werden. Im Allgemeinen ist das Decodierungsproblem NP-vollständig [80].

Da McEliece ein asymmetrisches Verfahren ist werden sowohl privater als auch öffentlicher Schlüssel in der Schlüsselerzeugung generiert. Der private Schlüssel besteht aus

- Generatormatrix $G \in \mathbb{F}_2^{k \times n}$, basierend auf dem Tripel (n, k, d) , das maximal t Fehler korrigieren kann und den Code C erzeugt. G muss invertierbar sein mit $GG^{-1} = I$, was der Einheitsmatrix entspricht (Goppa-Code Matrix),
- Permutationsmatrix $P \in \mathbb{F}_2^{n \times n}$,
- invertierbare Matrix $S \in \mathbb{F}_2^{k \times k}$ („Scrambler“ Matrix).

Der öffentliche Schlüssel G' leitet sich aus den drei Matrizen ab als

$$G' = SGP \in \mathbb{F}_2^{k \times n}.$$

Eine zu verschlüsselnde Nachricht m ist erneut ein binärer Vektor der Länge k über die Werte $\{0, 1\}$ als $m \in \mathbb{F}_2^k$. Für längere Nachrichten muss diese erst in verschiedene Blöcke eingeteilt werden. Für die Verschlüsselung wird zusätzlich eine Gewichtung e benötigt, die t nicht überschreitet (maximale Summe an Einsen),

$$e \in \mathbb{F}_2^n,$$

$$x = mG' + e \text{ mit}$$

$$x \in \mathbb{F}_2^n.$$

Die Entschlüsselung erfolgt durch die Multiplikation der inversen Matrix von P^{-1} auf x womit jedoch zunächst ein x' berechnet wird. Durch die Decodierung von x' erhält man m' .

$$xP^{-1} = (mG' + e)P^{-1}$$

$$(mG' + e)P^{-1} = mG'P^{-1} + eP^{-1}$$

Wobei der zweite Summand weiterhin einfach ein Fehler e , in dem Fall dann e' , bleibt. G' ist das Produkt aus $G' = SGP$ und somit kann P^{-1} eliminiert werden.

$$mG'P^{-1} + e' = mSG + e'$$

Aufgrund des Goppa-Codes können t Fehler korrigiert werden. So wird e' eliminiert. Bedingung für die Schlüssel war die Invertierbarkeit der Matrizen. Somit kann auf das bestehende Produkt das Inverse von S und G multipliziert werden, um m zu gewinnen.

$$mSGS^{-1}G^{-1} = m.$$

Beide Operationen sind effizient [80, p. 12], [82].

Beispiel

Betrachtet wird das McEliece-System $(n, k, d) = (7, 4, 3)$ mit $G \in \mathbb{F}_2^{k \times n}$

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix},$$

die zufällige und invertierbare Matrix $S \in \mathbb{F}_2^{k \times k}$

$$S = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

und die Permutationsmatrix $P \in \mathbb{F}_2^{n \times n}$

$$P = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Durch die Multiplikation der drei Matrizen wird der öffentliche Schlüssel G' mit

$$G' = SGP = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

berechnet. Sei nun die Nachricht $m \in \mathbb{F}_2^n$

$$m = (0101),$$

dann berechnet sich der Code c mit

$$c = mG' = (1011010).$$

Zusätzlich wird der Ein-Bit Fehler $e = (0001000)$ zu c addiert um den Geheimtext x zu berechnen

$$x = c + e = (1010010).$$

Für die Entschlüsselung werden die inversen Matrizen S^{-1}, P^{-1} benötigt,

$$S = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} \rightarrow S^{-1} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix},$$

$$P = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \rightarrow P^{-1} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Zunächst wird der empfangene Wert x zu c' dekodiert

$$c' = xP^{-1} = (0000111)$$

und die sich daraus ableitende Nachricht m' mit

$$m' = mS = (0010).$$

Mit der inversen Matrix S^{-1} kann nun m rekonstruiert werden

$$m'S^{-1} = m = (0101).$$

Ursprünglich wurde das System von McEliece mit den Werten $n = 1024$, $t \approx 50$ (Fehler) und das maximal mögliche k , für das ein Goppa-Code mit diesen Parametern existiert $k = 524$. Durch diese Werte wird die geheime Permutationsmatrix auf ca. 2^{64} Möglichkeiten festgelegt. Durch entsprechend größerer Werte ist das Durchführen einer vollständigen Schlüsselsuche nicht mehr effizient möglich.

A37 Das Matsumoto-Imai Kryptosystem C*

Es gab in der Kryptographie mehrere Versuche Felder nutzbar zu machen, da diese schnell zu berechnen sind und auch einfach in Hardware-Form implementiert werden können. Dennoch gab es immer wieder Probleme. Insbesondere wenn die Dimensionen kleiner als 1.000 war konnte erfolgreich Kryptoanalyse betrieben werden. Das Matsumoto-Imai Schema, auch C*, ist in der Lage deutlich schneller Signaturen zu ermöglichen als das RSA Schema. Auch besteht die Möglichkeit zu verschlüsseln.

Sei K nun $K = \mathbb{F}_{q^m}$ mit $q = 2$, dann ist $L = K^n = (\mathbb{F}_{2^m})^n \cong \mathbb{F}_{2^{mn}}$. Es gibt den Isomorphismus (umkehrbar eindeutig, bijektiv abbildbar) $\phi: L \rightarrow K^n$ mit einer K -Basis $\gamma_1, \dots, \gamma_n$ von L sowie ein System von m quadratischen Gleichungen $P = p_1, \dots, p_m$ mit n unbekanntem x_1, \dots, x_n .

Die versteckte Funktion φ im C*-Verfahren ist eine monomiale Abbildung

$$\varphi: \begin{array}{l} L \rightarrow L \\ x \rightarrow x^{1+q^\theta} \end{array}$$

mit $\theta \in \{1, \dots, n-1\}$, so dass $\text{ggT}(1+q^\theta, q^n-1) = 1$. Daher existiert ein $h \in \{1, \dots, q^n-1\}$ mit $(1+q^\theta) \cdot h \equiv 1 \pmod{q^n-1}$. Es gilt für alle $x \in L$

$$(\varphi(x))^h = x^{(1+q^\theta) \cdot h} = x.$$

Somit ist φ bijektiv. Daraus folgt auch, dass P bijektiv ist. Dann gibt es zu jedem Ciphertext genau einen Klartext.

Mit dem öffentlichen Schlüssel P kann nun eine Nachricht x signiert werden,

$$P: \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix} \rightarrow \begin{pmatrix} p_1(x_1, \dots, x_n) \\ \dots \\ p_n(x_1, \dots, x_n) \end{pmatrix}$$

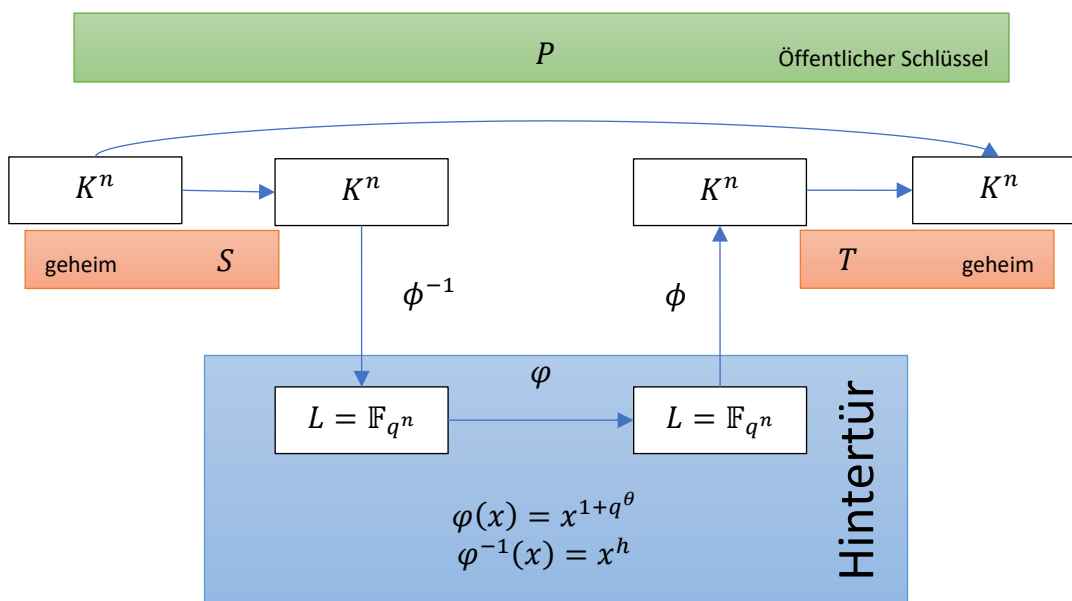


Bild 70 Darstellung des Matsumoto-Imai-Schema C* mit geheimen und öffentlichen Schlüsseln sowie der versteckten Hintertür.

Angemerkt sei, dass Jacques Patarin auf der Eurocrypt⁶¹ 1995 zeigte, dass das gewählte ϕ nicht sicher ist. Er war in der Lage das System zu brechen. Er leitete jedoch ebenfalls Vorschläge für eine Verbesserung ab. Eine Umkehrung von P ist auch ohne Kenntnis von S, T mit Hilfe einer bestimmten Relation möglich, womit P keine Einwegfunktion ist. Er konnte zeigen, dass zwischen einem Ciphertext $P(x_1, \dots, x_n)$ und (x_1, \dots, x_n) immer bestimmte Relation liegt.

$$\sum \sum \gamma_{ij} x_i y_j + \sum a_i x_i + \sum \beta_i y_i + \delta_0 = 0$$

für

$$(x_1, \dots, x_n) \rightarrow p_1(x_1, \dots, x_n) = (y_1, \dots, y_n).$$

Durch ausreichend viele Klar- und Ciphertexte kann das Gleichungssystem gelöst werden. Somit wird S, T umgangen [83].

⁶¹ <https://eurocrypt.iacr.org>

A38 Hidden Field Equations HFE

Im Zuge des Brechens des Matsumoto-Imai Verfahren durch Jacques Patarin stellte er das HFE-Verfahren auf der Konferenz Eurocrypt 1996 vor. Es basiert auf der Idee von Matsumoto und kann als „Reparatur“ von C^* betrachtet werden. Dabei bleibt P in seiner Form als multivariates quadratisches System erhalten wobei aber φ nunmehr ein beliebiges Polynom $U(\varphi) := \varphi \in U_{n,2}$ und so möglichst allgemein sein soll, $\varphi: L \rightarrow L$ mit

$$\begin{aligned} \varphi(x) &= \sum_{i,j=0}^{n-1} \alpha_{ij} x^{q^i+q^j} + \sum_{i=0}^{n-1} \beta_i x^{q^i} + \gamma_0 \in L[x] \\ &= \mathbb{F}_{q^n}[x] \text{ vom Grad } d. \end{aligned}$$

Der Grad d stellt den Sicherheitsparameter dar, der mit steigender Größe mehr Sicherheit bietet auf Kosten der Performance. Für eine Hintertür im HFE, die Umkehrung von $\varphi(x)$, wird der Grad der Funktion beschränkt mit

$$q^i + q^j \leq d, q^i \leq d,$$

das Gewicht des Polynoms ist 2. $\varphi(x)$ umzukehren, also zu $\varphi(a) = b$ das a zu berechnen, bedeutet das Finden der Nullstellen von $\varphi(a) - b = 0$ und $\varphi(a) - b$ zu faktorisieren. φ ist nicht bijektiv. Daraus folgt auch, dass P nicht bijektiv ist. Somit gibt es keine Garantie zu jedem Ciphertext genau einen Klartext zu finden.

Mit dem öffentlichen Schlüssel P kann nun eine Nachricht x signiert werden,

$$P: \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix} \rightarrow \begin{pmatrix} p_1(x_1, \dots, x_n) \\ \dots \\ p_n(x_1, \dots, x_n) \end{pmatrix}$$

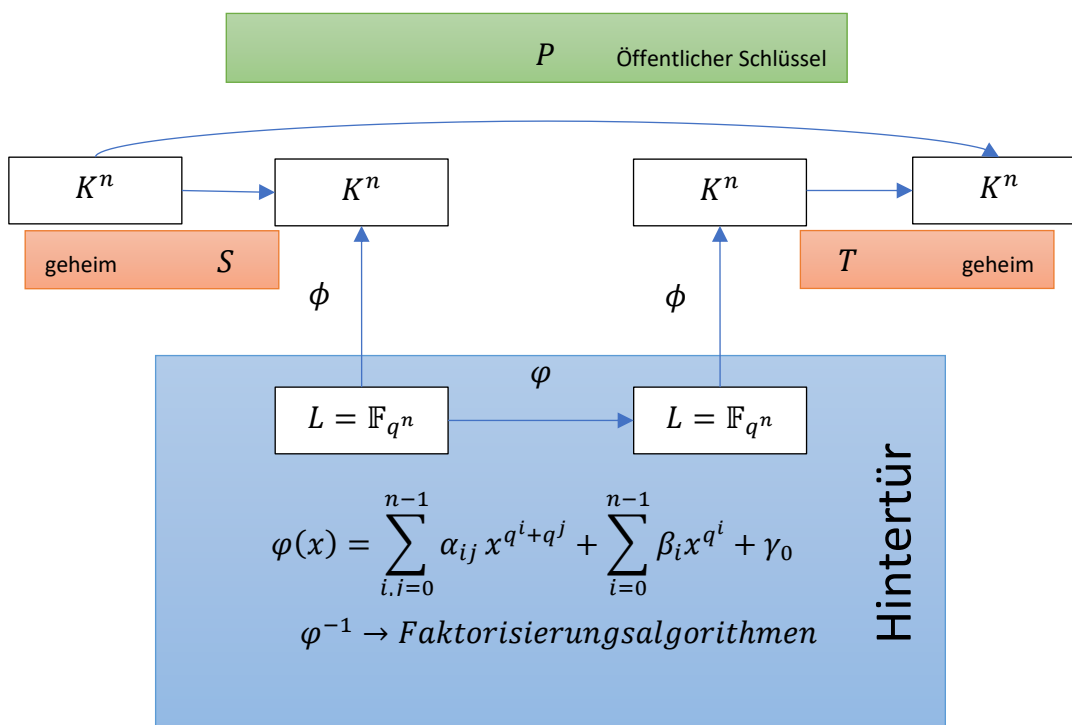


Bild 71 Darstellung des HFE-Schemas mit geheimen und öffentlichen Schlüssel sowie der Hintertür. Die Faktorisierung ist bei nicht allzu großem Grad d möglich [84].

Die fehlende Bijektivität von P und ϕ wird durch die Anwendung zusätzlicher, redundanter Informationen im Geheimtext kompensiert. Beispielsweise kann ein Hashwert der Nachricht m der Nachricht im Klartext oder als Ciphertext hinzugefügt werden,

$$m \rightarrow P(m)||h(m).$$

Ist P auch nicht surjektiv, dann existiert nicht immer eine Signatur $P(m)$ zu einer Nachricht m .

Öffentliche Parameter im HFE sind der Grundkörper K , die Anzahl der Elemente q sowie die Charakteristik p , die Einwegfunktion P durch die multivariate Darstellung sowie die Art wie mit der Redundanz umgegangen wird. Privat bleiben nach wie vor die Abbildungen S, T , die weiterhin bijektiv affin sind. Ob die Körper L bzw. K^n sowie die Funktion ϕ veröffentlicht werden oder nicht spielt für das Verfahren keine Rolle. Eine Nichtveröffentlichung erhöht höchst-

tens die Sicherheit während ein öffentliches ϕ einer Community die Möglichkeit böte ein besonders sicheres zu finden. Ein möglicher Angriff stellt die vollständige Suche von Urbildern $x \in K^n$ zu einem zugehörigen $P(x) = y \in K^n$ dar. Der Aufwand $|K|^n = q^n$ sollte ein $q^n \geq 2^{80}$ betragen während das Lösen polynomieller Gleichungssysteme einen Aufwand von ca. $\mathcal{O}(2^{2,7n})$ hat und so ein $n \geq 30$ gewählt werden sollte. Ein ähnlicher Angriff wie im C*-Verfahren kann durch Steigerung der Komplexität durch das Anheben von d vorgebeugt werden. Eine mögliche Konfiguration lautet $d \geq 128$. Bei entsprechender Wahl der Parameter bedeutet das für die Schlüssel des geheimen Schlüssels (S und T)

$$2n^2 \lceil \log_2(q) \rceil = 2 \cdot 80^2 (\log_2(2)) \approx 1,6 \text{ kByte}$$

und dem öffentlichen Schlüssel P

$$\frac{n^2(n-1)}{2} \lceil \log_2(q) \rceil \approx 32 \text{ kByte}, [83].$$

A39 Elliptic Curve Cryptography, ECC

Im Jahr 1986 wurden elliptische Kurven erstmalig für den Einsatz in der Kryptographie vorgestellt und werden seither für Verschlüsselungsverfahren und für den Schlüsselaustausch verwendet z.B. Elliptic Curve Diffie-Hellman, ECDH. ECC-Verfahren beziehen ihre Sicherheit durch das Problem des Lösens des diskreten Logarithmus.

Eine elliptische Kurve E ist eine affine Kurve über einem Körper K der Charakteristika $\text{char}(K) \notin \{2,3\}$, beispielsweise dargestellt in der Weierstraß-Gleichung

$$y^2 + a_1xy + a_2y = x^3 + a_3x^2 + a_4x^1 + a_5x^0, a_i \in K,$$

über dem Körper der reellen Zahlen \mathbb{R}

$$y^2 = x^3 + ax + b$$

mit

$$4a^3 + 27b^2 \neq 0$$

und dem Ausschluss von Singularitäten. Zusätzlich gibt es einen Punkt \mathcal{O} , der im Unendlichen liegt. Es ist der Nullpunkt, was jedoch nicht den Ursprung des Koordinatensystems bezeichnet [21], [5]. Elliptische Kurven haben die Eigenschaft, dass wenn sie von einer Geraden geschnitten werden, es drei Schnittpunkte gibt, wobei drei unterscheidbaren Fällen betrachtet werden:

- Die Gerade verläuft parallel zur Y-Achse, so ist einer der Schnittpunkt der Nullpunkt,
- die Gerade berührt die Kurve (tangiert), dann ist der Berührungspunkt ein doppelter Schnittpunkt,
- bei allen anderen Geraden sind drei Schnittpunkte offensichtlich.

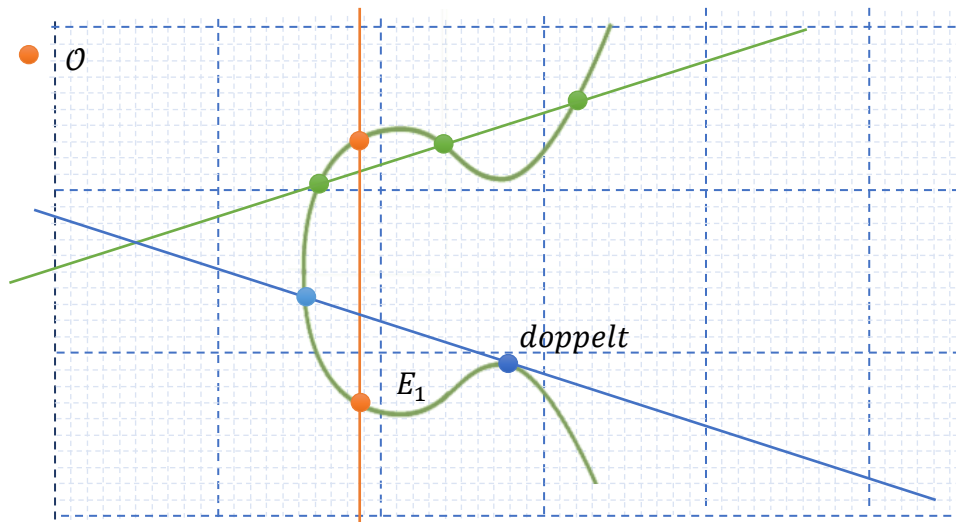


Bild 72 Die Menge der Schnittpunkte mit einer elliptischen Kurve ist stets 3.

Mit Hilfe der elliptischen Kurve und den genannten Eigenschaften lässt sich eine Gruppe $E(K)$ definieren, deren Elemente die Punkte auf der Kurve, inkl. dem Nullpunkt, sind. Die Addition von zwei Punkten erfolgt nun als Gerade durch die beiden Punkte. Handelt es sich um eine Tangente gilt die Verdopplung. Gemäß der drei Schnittpunkten gibt es nun einen dritten Punkt auf der Geraden. Dieser ist das Inverse des Ergebnisses der Addition der ursprünglichen zwei Punkte. Durch eine Spiegelung an der X-Achse erhält man das Ergebnis der Addition. Der Punkt $-R$ ist das additiv inverse zu R in der Gruppe und 0 das additiv neutrale Element mit $R + (-R) = 0$ und $R + 0 = R$ für alle R .

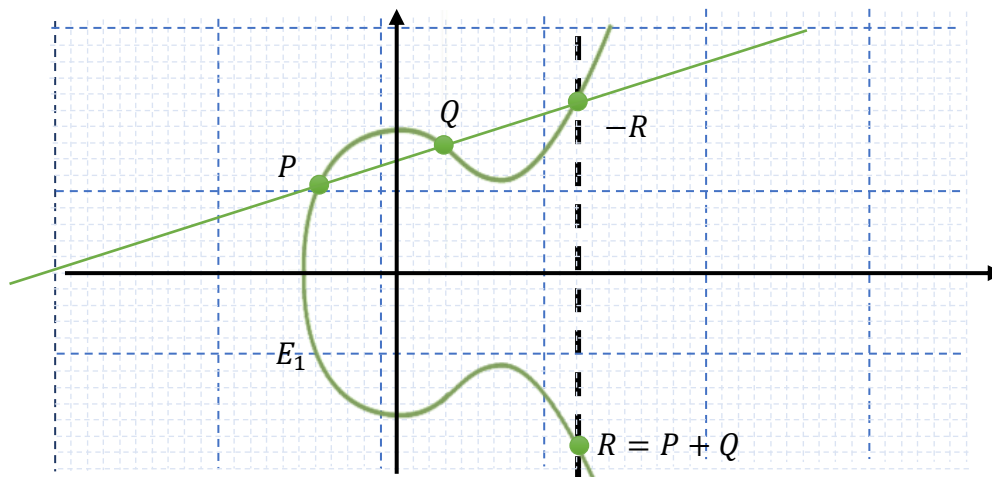


Bild 73 Graphische Interpretation der Addition der Punkte P, Q auf einer elliptischen Kurve.

Die Abbildung 73 zeigt beispielhaft die Addition $P + Q = R$ auf einer elliptischen Kurve E_1 sowie der Geraden

$$f(x) = y = mx + b,$$

$$m = \frac{Q_y - P_y}{Q_x - P_x}, P, Q, R \in E_1,$$

wobei mit einem der beiden Punkte P, Q anschließend b berechnet werden kann, z.B.

$$b = P_y - mP_x.$$

Im Falle der Punktverdopplung berechnet sich die Steigung m mit

$$m = \frac{3P_x^2 + a}{2P_y},$$

wobei a aus der Kurvengleichung

$$y^2 = x^3 + ax + b$$

bezogen wird.

Sollte die Gerade die elliptische Kurve in einem Punkt P berühren entspräche das der Punktverdoppelung

$$P + P = 2P = R.$$

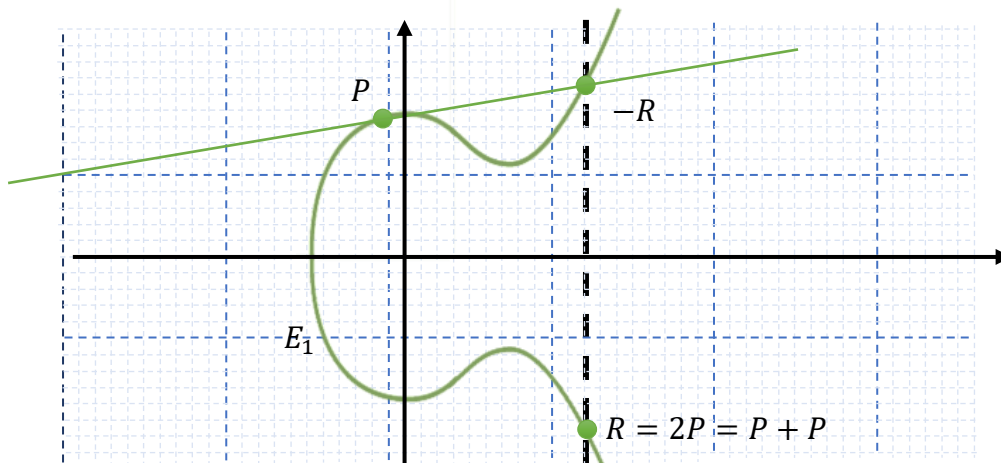


Bild 74 Graphische Interpretation der Punktverdoppelung P .

Die Tatsache, dass elliptische Kurven auch über endlichen Körpern definiert werden können, wird in der ECC ausgenutzt, wobei die Anzahl der Elemente innerhalb des Körpers seine Ordnung bestimmen. Die Verknüpfung der Elemente erfolgt mit der Punktaddition. Eine Multiplikation kann durch eine mehrfache Addition von Punkten realisiert werden und wird als Skalarmultiplikation bezeichnet (die n -fache Addition desselben Punktes mit sich entspricht der Multiplikation des Punktes mit dem Skalar n). Damit ist für die mehrfache Addition der Begriff Potenzfunktion ebenfalls gerechtfertigt. Somit würde die Umkehrung der Funktion dem Logarithmus entsprechen. Während eine Potenzfunktion effektiv mittels Algorithmus realisiert werden kann ist dies für die Umkehrung, den Logarithmus, nicht möglich.

In einem endlichen Körper ist die Anzahl der Elemente beschränkt. Im Körper werden die Ergebnisse Modulo einer Zahl reduziert. Die Division für die Berechnung der Steigung wird mit Hilfe des multiplikativ inversen Elements des Moduls gelöst. Außerdem sind auch die Punkte auf der elliptischen Kurve begrenzt („Punktewolke“). Durch die Abgeschlossenheit der Gruppe scheint die

Addition von zwei Punkten einem zufälligen Springen innerhalb der Menge der Elemente gleichzukommen.

Diffie-Hellman auf einer elliptischen Kurve

Alice und Bob müssen sich bei der Generierung eines gemeinsamen Schlüssels K auf eine elliptische Kurve E sowie einen Punkt P auf ihr verständigen. Alice und Bob wählen nun jeweils ein geheimes Skalar a_k bzw. b_k und berechnen zusammen mit P einen zu veröffentlichen Punkt $A_k = a_k P$ bzw. $B_k = b_k P$. Analog zum ursprünglichen Diffie-Hellman verfahren tauschen Alice und Bob dieses Ergebnis untereinander aus und berechnen im Anschluss ihren gemeinsamen Schlüssel

$$Alice \rightarrow K = a_k b_k P = a_k B_k = b_k A_k = b_k a_k P = K \leftarrow Bob.$$

Das Verfahren beruht auf dem Problem des Lösens des diskreten Logarithmus und ist daher aktuell sicher. Die Komplexität des diskreten Logarithmus in $E(GF(m))$ steigt mit m linear. Eine Schlüssellänge von 1.024 Bit kann so auf 200 Bit gekürzt werden ohne Reduktion der Sicherheit. Die eingesparte Rechenzeit kann ca. mit dem Faktor 10 bemessen werden. Zusätzlich werden Geheimtexte bzw. Signaturen gekürzt. Durch das Einsparen von Ressourcen sind elliptische Kurven interessant für den Einsatz bei z.B. Smartcards [5], [21], [85].

Tabelle 28 Vergleich der Sicherheitsniveaus bei entsprechenden Schlüsselgrößen von RSA und ECC-Verfahren⁶².

Sicherheitsniveau Bit	Schlüsselgröße Diffie-Hellman in Bit	Schlüsselgröße ECDH in Bit
80	1.248	160
112	2.432	224
128	3.248	256
192	7.936	384

⁶² <https://cordis.europa.eu/docs/projects/cnect/6/216676/080/deliverables/002-DSPA20.pdf>, Seite 30

256	15.424	512
------------	--------	-----

Die Sicherheit dieser Verfahren ist solange gegeben bis Quantencomputer ausreichender Kapazität zur Verfügung stehen. Ironischerweise sind ECC-Verfahren aufgrund ihrer geringen Schlüssellänge eher bedroht als RSA-Verfahren, da zum Brechen weniger Qubits erforderlich sind. Schätzungen gehen davon aus, dass 1.600 Qubits ausreichen, um das ECC-Verfahren Curve25519 zu brechen, während ca. 6.100 Qubits benötigt werden, um RSA-3072 zu brechen [86].

A40 Supersingular Isogeny Diffie-Hellman, SIDH

SIDH ist ein gegen Quantencomputer robustes Verfahren und zeigt dabei Verwandtschaft zur Diffie-Hellman-Schlüsselaustausch. Die Schlüsselgrößen, im Vergleich zu anderen Post-Quantenverfahren, sind klein. Eine Größe von 330 Byte entsprechen einem Sicherheitsniveau von 128 Bit [87]. Ein Vorteil gegenüber anderen Post-Quantenverfahren ist, dass SIDH die Perfect Forward Security, PFS, unterstützt. Selbst wenn ein Sitzungsschlüssel kompromittiert werden sollte, so kann ein Angreifer maximal einen kleinen Teil der Kommunikation entschlüsseln, da PFS regelmäßig die Parameter während einer Sitzung ändert. Weder z.B. NTRU noch Ring-LWS unterstützen diese Art der Sicherheit. Schlüssel können in ca. 200 ms gebildet werden.

Das Verfahren beruht auf dem Konzept, dass zwei elliptische Kurven E_1 und E_2 über eine Funktion bzw. Abbildung φ verfügen, die einen Punkt P_1 auf E_1 auf E_2 als Q_1 abbildet. Diese Art von Funktion wird als Isogenie bezeichnet und ist Namensgeber des Verfahren, d.h. $\varphi: E_1 \rightarrow E_2$ erfüllt den Homomorphismus für die beiden abelschen Varietäten E_1 und E_2 . Dabei ist φ surjektiv und ein Q_i besitzt mindestens ein Urbild auf E_1 . Darüber hinaus verfügt φ über einen endlichen Kern. Mit dieser Isogenie kann nun jeder Punkt von E_1 auf E_2 abgebildet werden. Die Isogenie stellt dann den privaten Schlüssel dar, die elliptische Kurve den öffentlichen Schlüssel.

Tabelle 29 Tabellarischer Vergleich der Diffie-Hellman-Schlüsselaustausch-Implementierungen.

	Diffie-Hellman	Elliptic Curve Diffie-Hellman	Supersingular Isogeny Diffie-Hellman
Elemente	Zahlen g mod Prime	Punkte P auf Kurve	Kurven E einer Isogenie-Klasse
Geheimnis	Exponent x	Skalar k	Isogenie φ
Berechnung	$g, x \rightarrow g^x$	$k, P \rightarrow [k]P$	$\varphi, E \rightarrow \varphi(E)$
Bekannt	g, g^x	$P, [k]P$	$E, \varphi(E)$

Schwierigkeit	Ermitteln von x	Ermitteln von k	Ermitteln von φ
---------------	----------------------	-------------------	-------------------------

Das Problem unterscheidet sich somit von ECC-Verfahren dahin gehend, dass in ECC eine versteckte Beziehung zwischen zwei Punkten auf derselben elliptischen Kurve beruht während nun eine Beziehung zwischen zwei elliptischen Kurven in einem großen Pool vorliegt.

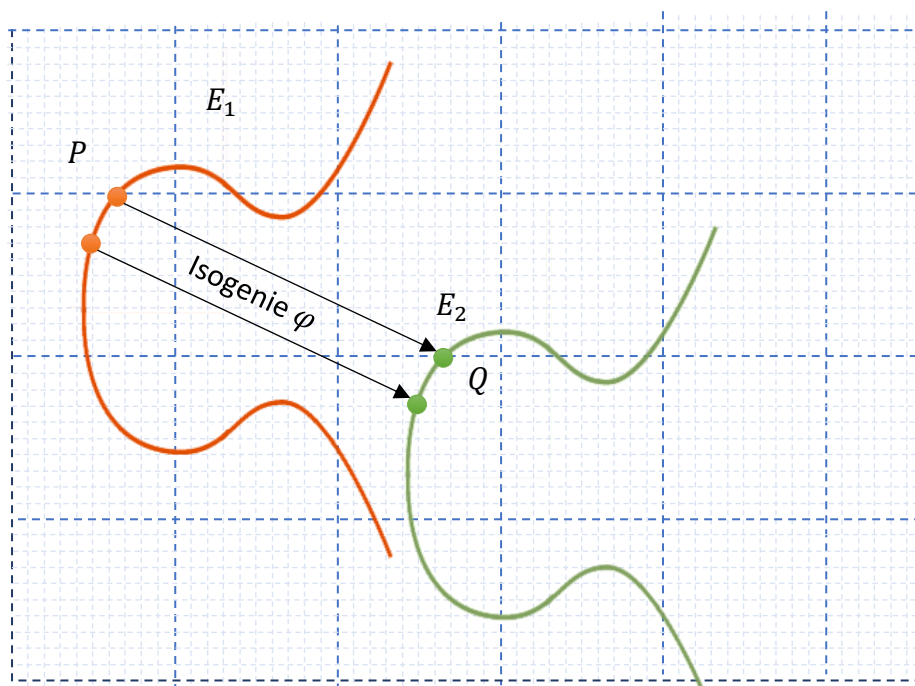


Bild 75 Exemplarische Darstellung der Isogenie zw. den beiden elliptischen Kurven E_1 und E_2 .

Unter dem Pool versteht man nun einen großen Graphen dessen Knoten E_i elliptische Kurven darstellen, wobei die Kanten zwischen den Knoten die Isogenien φ darstellen.

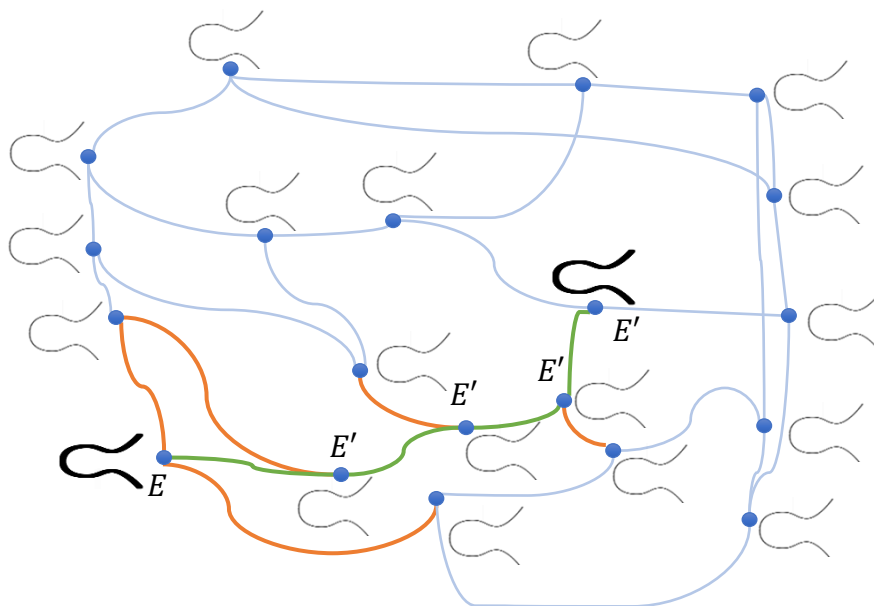


Bild 76 Beispiel-Graph elliptischer Kurven E_i verbunden über jeweils drei Kanten (Grad 2).

Durch das zufällige Wählen von Kanten φ gelangt man von einer Ausgangskurve E zu einer anderen Kurve E' . Selbiges geht auch über einen Graphen mit vier Kanten φ (Grad 3) zwischen den elliptischen Kurven E_i . Diese Art von Pool nennt man Isogenieklasse, die aus supersingularen elliptischen Kurven über einem endlichen Körper \mathbb{F}_{p^2} mit p^2 Elementen, wobei p eine Primzahl darstellt.

Der Diffie-Hellman-Schlüsselaustausch ist nun so implementiert, dass Alice und Bob sich auf einen gemeinsamen Pool von Kurven als Graph einigen sowie einer Ausgangskurve E . Alice wählt nun zufällig einen Pfad durch den Graphen, ausgehend von drei anliegenden Kanten bis zum Erreichen einer Kurve E_A . Bob seinerseits wählt analog einen zufälligen Pfad durch den Graphen, jedoch ausgehend von vier anliegenden Kanten je Knoten und erreicht so die elliptische Kurve E_B .

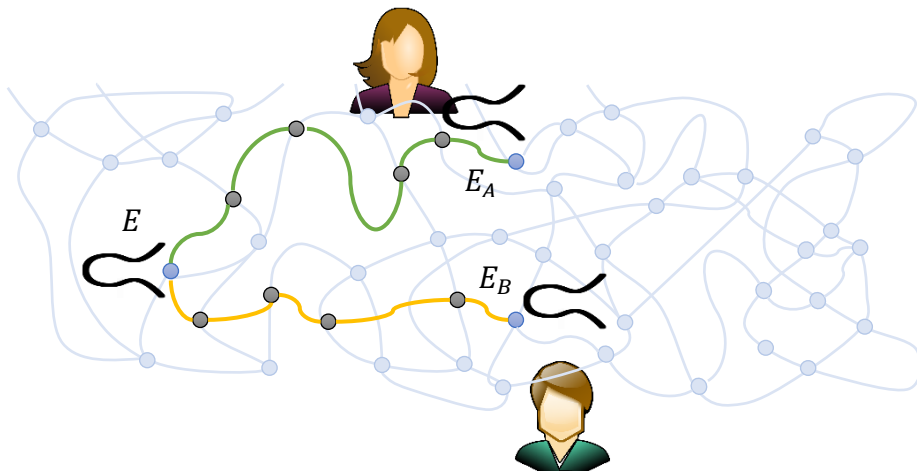


Bild 77 Alice und Bob wählen jeweils einen zufälligen Pfad durch den Graphen zu den elliptischen Kurven E_A und E_B .

Alice und Bob tauschen nun die Ergebnisse $E_A = \varphi_A(E)$, $E_B = \varphi_B(E)$ untereinander aus. Alice wendet anschließend ihre Folge, ausgehend von der von Bob erhaltenen elliptischen Kurve E_B erneut an. Bob wiederum macht dasselbe analog. Auf diese Weise berechnen Alice und Bob dieselbe elliptische Kurve E_{AB} (Die Verschiebung von Alice gewählten Pfad auf Bobs übermittelten Ausgangspunkt E_B ist mittels zusätzlicher Informationen möglich).

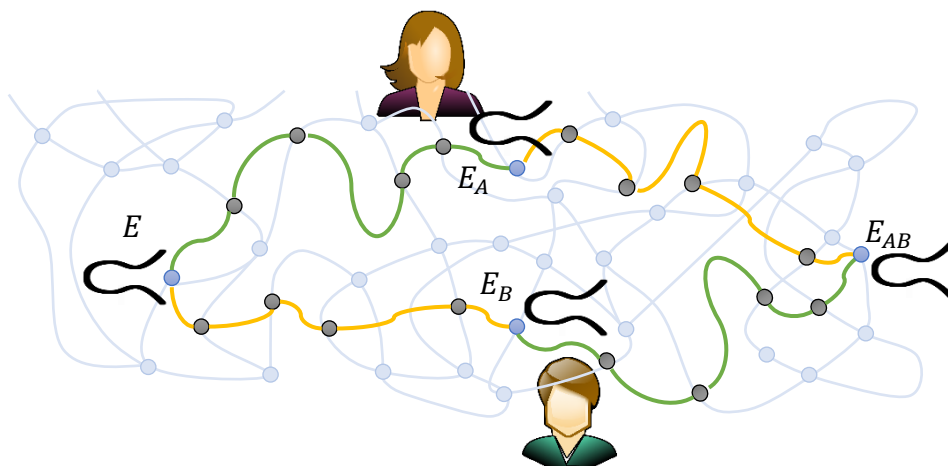


Bild 78 Berechnung der gemeinsamen Kurve E_{AB} analog zum Diffie-Hellman-Schlüsselaustausch ohne die Verwendung des diskreten Logarithmusproblems.

Benötigt werden Isogenien ausgehend von einer elliptischen Kurve E . Hierfür reicht es Untergruppen $G \subset E$ zu betrachten. Mit der Formel von Vélu können

diese Isogenien explizit berechnet werden. Dabei lassen sich Isogenien von kleinem Grad effizient berechnen. Supersingularität für E liegt genau dann vor, wenn $\#E(\mathbb{F}_q) \equiv 1 \pmod{p}$ gilt. Jede supersinguläre Kurve über \mathbb{F}_{p^k} ist isomorph zu einer supersingulären Kurve über \mathbb{F}_{p^2} , $p > 3$, $k \in \mathbb{N}$. Alle elliptischen Kurven über \mathbb{F}_{p^2} befinden sich in derselben Isogenieklasse. Interessant ist die Menge S_{p^2} der supersingulären j -Invarianten über \mathbb{F}_{p^2} . Sie lässt sich berechnen mit

$$\#S_{p^2} = \left\lfloor \frac{p}{12} \right\rfloor + \varepsilon, \varepsilon \in \{0,1,2\}.$$

Der für oben vorgestellte ℓ -Isogeniegraph mit $\ell < p$ trägt als Knoten die j -Invarianten über \mathbb{F}_{p^2} mit den ungerichteten Isogenien als Kanten vom Grad ℓ . Die j -Invarianten einer Weierstraß-Kurve E berechnet sich mittels

$$j(E) = j(a, b) = 1728 \frac{4a^3}{4a^3 + 27b^2}.$$

Beispiel über den Körper \mathbb{F}_{13} und den beiden Kurven E_1, E_2 mit

$$E_1: y^2 = x^3 + 9x + 8,$$

$$E_2: y^2 = x^3 + 3x + 5$$

als Weierstraß-Gleichung

$$y^2 = x^3 + ax + b$$

gibt

$$j(E_1) = 1728 \frac{4 \cdot 9^3}{4 \cdot 9^3 + 27 \cdot 8^2} = 3 = 1728 \frac{4 \cdot 3^3}{4 \cdot 3^3 + 27 \cdot 5^2} = j(E_2)$$

und somit sind die zwei Kurven E_1, E_2 über \mathbb{F}_{13} isomorph.

Für jede Primzahl ℓ , die p nicht teilt $\ell \nmid p$, gehen von jedem Knoten des Graphen $X(S_{p^2}, \ell)$ dann $\ell + 1$ Kanten ab, welche ungerichtet sind. Als Isogeniegrad wird aus Effizienzgründen $\ell_A = 2$ sowie $\ell_B = 3$ gewählt, d.h. in den Isogeniegraphen von supersingulären Kurven gibt es den Grad 2 mit 3 Kanten je Knoten und Grad 3 mit 4 Kanten je Knoten.

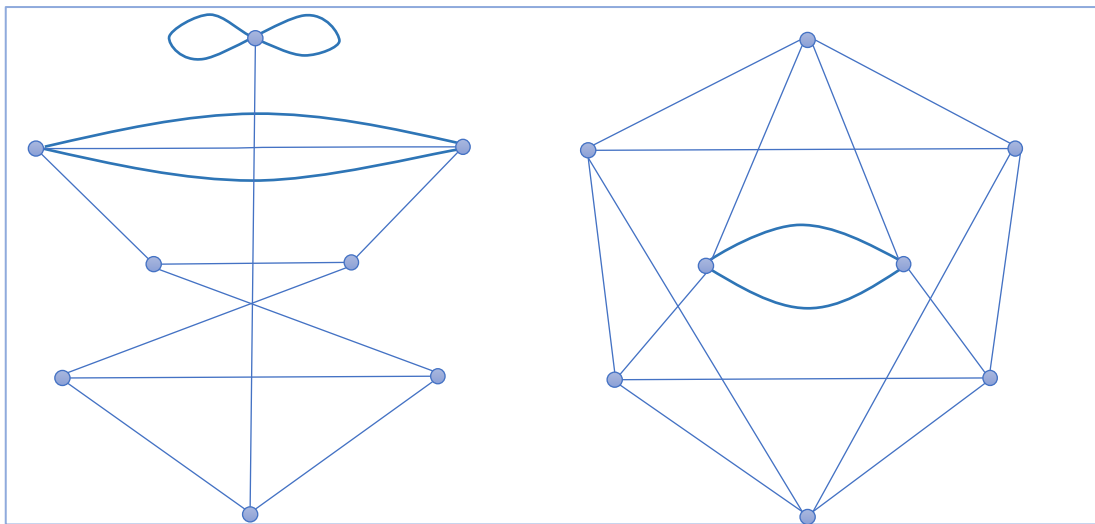


Bild 79 Beispiel-Isogeniegraphen von supersingulären Kurven im Grad 2, links, und Graph 3, rechts, über dem Körper \mathbb{F}_{97^2} mit $\#\mathcal{S}_{97^2}=8$ nach [85].

Für die Durchführung des Diffie-Hellman-Verfahrens auf supersingulären isogenen elliptischen Kurven im Graphen berechnet Alice nun einen zufälligen Punkt R_A mit den zwei Zufallszahlen $m_A, n_A \in \ell^{e_A}$ (ℓ_A kleine Primzahl, $e_A \rightarrow$ Alice geheime Isogenie als Komposition vom Grad ℓ_A),

$$R_A = [m_A]P_A + [n_A]Q_A.$$

Die Punkte P, Q erzeugen die sogenannte Torsionsgruppe $E[\ell^{e_A}]$. Somit erzeugt R_A eine zufällige Untergruppe der Ordnung ℓ^{e_A} . Hieraus lässt sich nun die Komposition e_A erstellen. Bob verfährt analog und beide erhalten ihre Isogenie

$$\varphi_A: E \rightarrow E_A,$$

$$\varphi_B: E \rightarrow E_B.$$

Sie hängen von den zufällig gewählten Zahlen m_A, n_A bzw. m_B, n_B ab. Sowohl Alice als auch Bob machen einen „Random Walk“ der Länge e_A bzw. e_B durch den ℓ_A bzw. ℓ_B –Isogeniegraphen.

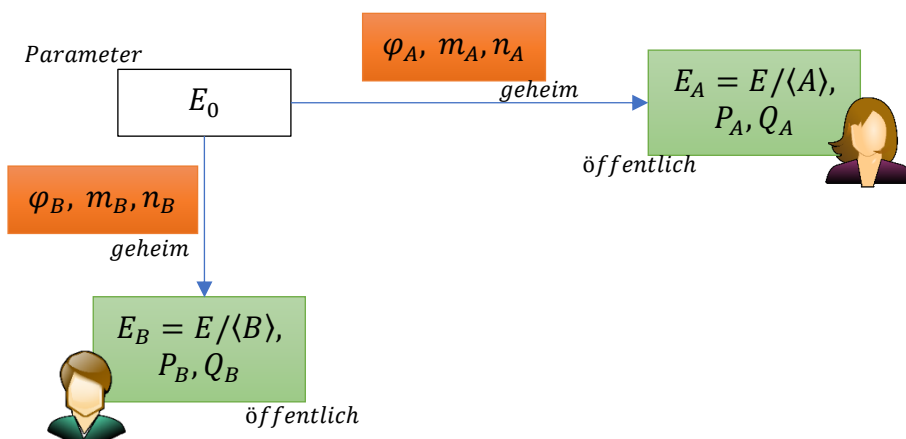


Bild 80 Erstellen der öffentlichen Schlüssel von Alice und Bob.

Alice und Bob Generatorpunkte P, Q liegen nun jedoch auf E_0 . Nach Austausch der öffentlichen Schlüssel sind die Ausgangskurven aber nun E_A bzw. E_B . Hierfür müssen Alice und Bob ihre Generatorpunkte mit ihrer geheimen Isogenie φ gegenseitig auf E_A bzw. E_B abbilden und ebenfalls bereitstellen. Dabei handelt es sich um die Werte

$$\varphi_A(P_B), \varphi_A(Q_B)$$

sowie

$$\varphi_B(P_A), \varphi_B(Q_A).$$

Alice und Bob verwenden nun ihre geheimen Zufallszahlen m, n und wenden diese, ausgehend von ihren neu abgebildeten Startpunkten z.B. $\varphi_A(P_B), \varphi_A(Q_B)$, an. Alice kann nun den Punkt R_{BA} mit den vorhergehenden zwei Zufallszahlen $m_A, n_A \in \ell^{e_A}$ mit

$$R_{BA} = [m_A]\varphi_B(P_A) + [n_A]\varphi_B(Q_A)$$

berechnen. So ist sie in der Lage die Isogenie $\varphi_{BA}: E_B \rightarrow E_{BA}$ zu erstellen. Bob verfährt analog. Auf diese Weise erreichen Alice und Bob die Kurven E_{BA}, E_{AB} , die nicht zwingend dieselben sein müssen, jedoch isomorph zueinander sind. Es gilt

$$j(E_{BA}) = j(E_{AB}).$$

Dieser Wert kann als gemeinsames Geheimnis verwendet werden [87], [85].

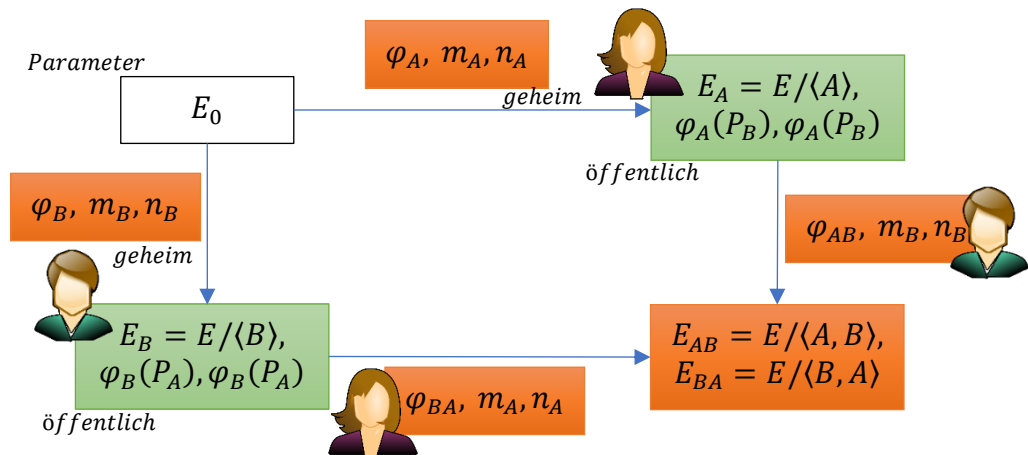


Bild 81 Alice und Bob berechnen ihren gemeinsamen, geheimen Schlüssel in Anlehnung an das Diffie-Hellman-Verfahren.

Literaturverzeichnis

- [1] J. Draeger, „Klassische und Quanten Turing Maschinen, Teil I,“ 25 06 2017. [Online]. Available: https://www.researchgate.net/publication/317869909_Klassische_und_Quanten_Turing_Maschinen_Teil_I. [Zugriff am 21 03 2019].
- [2] H. U. Simon, „Das Rechenmodell namens "Turing-Maschine",“ Ruhr-Universität Bochum, Germany, 2011/12.
- [3] Universität Ulm, Formale Grundlagen der Informatik, „Universität Ulm,“ [Online]. Available: https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.040/Formale_Methoden_der_Informatik/C3%9Cbungen/blatt08-zusatz_2.pdf. [Zugriff am 29 03 2019].
- [4] Divers, „Wikipedia - Elektrizität/Tabellen und Grafiken,“ Wikipedia - Die freie Enzyklopädie, 27 04 2019. [Online]. Available: https://de.wikipedia.org/wiki/Elektrizit%C3%A4t/Tabellen_und_Grafiken. [Zugriff am 30 04 2019].
- [5] K. Schmeih, Kryptographie. Verfahren, Protokolle, Infrastruktur, dpunkt Verlag, 2007.
- [6] C. Karpfinger und K. Hubert, Kryptologie. Algebraische Methoden und Algorithmen., Wiesbaden: Vieweg+Teubner, 2010.
- [7] S. Spitz, M. Pramateftakis und J. Swoboda, Kryptographie und IT-Sicherheit. Grundlagen und Anwendungen, Wiesbaden: Vieweg+Teubner Verlag, Springer Fachmedien Wiesbaden GmbH, 2008.
- [8] U. Baumann, E. Franz und A. Pfitzmann, Kryptographische Systeme, Berlin, Heidelberg: Springer-Verlag, 2014.
- [9] C. Paar und J. Pelzl, Kryptografie verständlich. Ein Lehrbuch für Studierende und Anwender, Berlin, Heidelberg: Springer-Verla, 2016.
- [10] A. May, „Public Key Kryptoanalyse,“ Ruhr Universität Bochum. Alexander May., Bochum, 2008.
- [11] A. Badach und E. Hoffmann, Technik der IP-Netze, München: Hanser, 2015.
- [12] H. W. Lang, „Hochschule Flensburg. University of Applied Sciences.,“ Hochschule Flensburg. University of Applied Sciences., 04 06 2018. [Online]. Available: <http://www.inf.fh-flensburg.de/lang/algorithmen/grundlagen/gruppe.htm>. [Zugriff am 20 04 2018].
- [13] M. Bartosch, „heise Security,“ Heise Medien GmbH & Co. KG, 27 05 2008. [Online]. Available: <https://www.heise.de/security/artikel/Diffie-Hellman-Verfahren-270980.html>. [Zugriff am 30 04 30].
- [14] Bundesamt für Sicherheit in der Informationstechnik., „Bundesamt für Sicherheit in der Informationstechnik, Technische Richtlinie TR-02102-3 Kryptographische Verfahren: Empfehlungen und Schlüssellängen,“ 11 02 2019. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR02102/BSI-TR-02102-3.pdf?__blob=publicationFile&v=6. [Zugriff am 28 04 2019].
- [15] H. W. Lang, „Diffie-Hellman-Schlüsselvereinbarung,“ Hochschule Flensburg, 14 03 2019. [Online]. Available: <http://www.inf.fh-flensburg.de/lang/krypto/protokolle/diffie.htm>. [Zugriff am 03 04 2019].
- [16] H. Lang, „Hochschule Flensburg. University of Applied Sciences.,“ <http://www.iti.fh-flensburg.de/lang/krypto/grund/gruppezn.htm>, 18 10 2018. [Online]. Available: <http://www.iti.fh-flensburg.de/lang/krypto/grund/gruppezn.htm>. [Zugriff am 17 05 2019].

- [17] A. Prof. Dr.-Ing. Ahrens, „Kryptografische Methoden und Anwendungen, Studienbrief, IT-Sicherheit und Forensik.“ Wings - Wismar International Graduation Service GmbH, Wismar, 2019.
- [18] J. Buchmann, Einführung in die Kryptographie, Berlin, Heidelberg: Springer-Verlag, 2016.
- [19] R.-H. Schulz und H. Witten, „Faktorisieren mit dem Quadratischen Sieb. Ein Beitrag zur Didaktik der Algebra und Kryptologie.“ Inst. f. Mathematik der FU Berlin, Berlin, 2011.
- [20] J. Döcker, „Pollards Rho-Methode zur Faktorisierung. Bachelorarbeit.“ Carl von Ossietzky Universität. Janosch Döcker., Oldenburg, 2011.
- [21] A. Prof. Dr.-Ing. Ahrens, „Kryptoanalyse, Studienbrief, IT-Sicherheit und Forensik.“ Wings - Wismar International Graduation Service GmbH, Wismar, 2018.
- [22] M. Goemans, „Factoring. 18.310 lecture notes.“ University of Michigan, Michigan, 2015.
- [23] M. Schneider, „Beschreibung des MPQS-Algorithmus und Implementierung eines Softwarepaketes als Beispiel für verteiltes Rechnens.“ OTH-Amberg, Weiden, 2017.
- [24] E. Landquist, „The Quadratic Sieve Factoring Algorithm-“, University of Virginia, Charlottesville, 2001.
- [25] C. Pomerance, „A Tale of Two Sieves“, University of Georgia, Athens, 1996.
- [26] R. Ghrist, „Orders of Growth.“ 20 09 2016. [Online]. Available: <http://calculus.seas.upenn.edu/?n=Main.OrdersOfGrowth>. [Zugriff am 12 04 2019].
- [27] R. Küsters und T. Wilke, Moderne Kryptographie. Eine Einführung., Wiesbaden: Vieweg+Teubner, 2011.
- [28] M. Kurt, Vom Bit zum Qubit. Eine kleine Einführung in die Quantencomputer., München: Red Horse, 2018.
- [29] C. J. Meier, Eine kurze Geschichte des Quantencomputers. Wie bizarre Quantenphysik eine neue Technologie erschafft., Hannover: Heise Zeitschriften Verlag GmbH & Co KG, 2015.
- [30] J. Rödling, „Unkonventionelle Computer. Quantengatter und Quantenschaltkreise.“, Jan Rödling. Technische Universität Braunschweig, Braunschweig, 2001.
- [31] F. Wunderlich-Pfeiffer, „CNOT-Gatter verschränken Qubits“, Golem Media GmbH, 25 07 2017. [Online]. Available: <https://www.golem.de/news/quantengatter-die-bauteile-des-quantencomputers-1707-128276-2.html>. [Zugriff am 09 05 2019].
- [32] G. Benenti, G. Casati und G. Strini, Principles of Quantum Computation and Information., Singapore: World Scientific Publishing Co. Pte. Ltd., 2005.
- [33] T. Bluher, „Introduction to Quantum Computing“, NSA, Math Research Group, USA, 2018.
- [34] D. Bacon, „CSE 599d - Quantum Computing. Winter 2006. One qubit, Two qubit (update 1/10/06)“, 01 10 2006. [Online]. Available: <https://courses.cs.washington.edu/courses/cse599d/06wi/>. [Zugriff am 10 05 2019].
- [35] M. Homeister, Quantum Computing verstehen. Grundlagen, Anwendungen, Perspektiven., Wiesbaden: Springer Vieweg, 2018.
- [36] W. Riedl, „Prinzipien des Quantencomputers und der Algorithmus von Shor. Diplomarbeit.“, Universität Bayreuth., Bayreuth., 2011.
- [37] L. Gasser, „Quanten Fourier Transformation: Simulation eines Quantenalgorithmus in Matlab. Bachelorarbeit.“, Insitut für Theorerische Physik. Computational Physics. TU Graz, Graz, 2018.
- [38] B. Trenwith, „Understanding Quantum Computers - 3.3 - Quantum Fourier Transform“, Keio University, Tokio, 2018.
- [39] T. Zoppke und C. Paul, „Shor's Algorithm. Faktorisierung großer Zahlen mit einem Quantencomputer.“, FU-Berlin. Zoppke, Till; Paul, Christian., Berlin, 2002.

- [40] E. Begemann, „Morituri te salutant!(1).“, <kes> *Die Zeitschrift für Informations-Sicherheit*, pp. 24-32, 02 2019.
- [41] J. Mandrysch, „Über den Shor-Algorithmus und Quantencomputer. Wissenschaftliche Arbeit.“, Wilhelms-Universität, Münster., Münster, 2016.
- [42] K. Melhorn und A. Neumann, „Quantenrechner. Grovers Algorithmus und technische Realisierung.“, Max Planck Institut Informatik., München, 2014.
- [43] H. Dr. Hagemeyer, S. Dr. Kousidis, M. Dr. Lochter und R. K. V. u. Entwicklung, „Public-Key-Kryptographie zukunftsicher machen.“ *BSI-Magazin "Mit Sicherheit"*, Bd. 02, Nr. 2018/02, pp. 22-23, 2018.
- [44] H. Böck, „Verschlüsselung. Was noch sicher ist.“, Golem Media GmbH, 09 09 2013. [Online]. Available: <https://www.golem.de/news/verschlueselung-was-noch-sicher-ist-1309-101457-9.html>. [Zugriff am 15 05 2019].
- [45] B. f. S. i. d. Informationstechnik, „Entwicklungsstand Quantencomputer. Deutsche Zusammenfassung.“, Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2018.
- [46] D. W. Hoffmann, *Theoretische Informatik*, München: Carl Hanser Fachbuchverlag, 2011.
- [47] M. Wiecha und M. Hanefeld, „Quantenteleportation.“, Goethe Universität. Frankfurt am Main. *Theoretische Physik.*, Frankfurt am Main., 2014.
- [48] N. Gisin und B. Huttner, „Quantum Cloning, Eavesdropping and Bell's inequality“, University of Geneva, Geneva, Switzerland., 1996.
- [49] B. Klein, „Quantenkryptographie. Theoretische Grundlagen und praktische Implikationen für die Schule.“, Technische Universität Kaiserslautern, Kaiserslautern, 2012.
- [50] P. Erker und M. Maiwöger, „Ekert-Protokoll.“, Universität Wien., Wien., 2011.
- [51] M. Zepfmeisel, „Einführung in die Grundlagen der Quantenkryptographie“, Ludwig-Maximilians-Universität München, München, 2006.
- [52] M. Fox, *Quantum Optics. An Introduction*. Oxford Master Series in Physics., New York: Oxford University Press, 2006.
- [53] U. Seyfarth, „Quantenkryptographie. Akademisches Hirngespinnst oder bald schon verbreitete Realität?“, <kes>. *Die Zeitschrift für Informations-Sicherheit*, pp. 16-24, 08 2017.
- [54] S. Singh, *Geheime Botschaften. Die Kunst der Verschlüsselung von der Antike bis in die Zeit des Internets.*, München: Deutscher Taschenbuch Verlag GmbH & Co. KG, 2001.
- [55] B. Preneel, „Challenges and opportunities. Business Cases for Quantum Key Distribution.“, COSIC KU Leuven and imec, Belgium., 2018.
- [56] L. Xin, „China launches world's first quantum science satellite“, *physicsworld*, 16 08 2016. [Online]. Available: <https://physicsworld.com/a/china-launches-worlds-first-quantum-science-satellite/>. [Zugriff am 27 05 2019].
- [57] R. C. Merkle, „A Certified Digital Signature.“, BNR Inc., Palo Alto, California., 1979.
- [58] g. GmbH, „Practical Hash-based Signatures“, genua. A Bundesdruckerei Company., [Online]. Available: <http://www.pqsignatures.org/index/hbs.html>. [Zugriff am 2019 06 05].
- [59] D. J. Bernstein, J. Buchmann und E. Dahmen, *Post-Quantum Cryptography.*, Berlin Heidelberg: Springer-Verlag, 2009.
- [60] D. Bernstein, D. Hopwood, A. Hülsing, T. Lange, R. Niederhagen, L. Papachristodoulou, P. Schwabe und Z. Wilcox O'Hearn, „SPHINCS: practical stateless hash-based signatures.“, Technische Universität Eindhoven., Eindhoven., 2018.
- [61] B. Prof. Buchanan, „a security site“, The Cyber Academy, 24 03 2018. [Online]. Available: <https://asecuritysite.com/encryption/wint>. [Zugriff am 06 06 2019].

-
- [62] B. Lauer, „XMSS: Post-Quanten-Verfahren steht bereit,“ Ebner Media Group GmbH & Co. KG, 11 06 2018. [Online]. Available: <https://www.dotnetpro.de/diverses/xmss-post-quanten-verfahren-steht-bereit-1544960.html>. [Zugriff am 06 06 2019].
- [63] K. Schmech, „Die Schnecke im Salatfeld. Post-Quantum-Kryptographie anschaulich erklärt,“ *iX Special 2019 - IT heute*, pp. 109-115, 2019.
- [64] J. Buchmann und A. Hülsing, „Johannes Buchmann & Andreas Hülsing - Hash-Based Signatures. Institute for Quantum Computing,“ 10 2014. [Online]. Available: <https://www.youtube.com/watch?v=MecexfUT4OQ>. [Zugriff am 07 06 2019].
- [65] A. Hülsing, E. Dahmen und J. Buchmann, „Practical Forward Secure Signatures using Minimal Security Assumptions,“ TU Darmstadt., Darmstadt, 2013.
- [66] A. Hülsing, „W-OTS+– Shorter Signatures for Hash-Based Signature Schemes,“ *Cryptography and Computer Algebra*, Department of Computer Science, TU Darmstadt., Darmstadt, 2017.
- [67] J. Buchmann, E. Dahmen und A. Hülsing, „XMSS – A Practical Forward Secure Signature Scheme based on Minimal Security Assumptions,“ *Cryptography and Computer Algebra*, Department of Computer Science, TU Darmstadt, Darmstadt, 2011.
- [68] J. Buchmann, J. Krämer, N. A. Alkadri, N. Bindel, R. El Bansarkhari, F. Göpfert und T. Wunderer, „Bewertung gitterbasierter kryptografischer Verfahren,“ Bundesamt für Sicherheit in der Informationstechnik 2017, Bonn., 2017.
- [69] E. Sanou, „Post-Quantum Cryptography: Lattice-based encryption,“ Universität Politecnica de Catalunya., Barcelona, 2016.
- [70] F. von Eye, „Die Entwicklung eines neuen Kryptosystems am Beispiel NTRU - Probleme und Chancen,“ Universität Augsburg, Augsburg, 2009.
- [71] F. Vallentin, „Zur Komplexität des "Shortest Vector Problem" und seine Anwendungen in der Kryptographie,“ Universität Dortmund, Dortmund, 1999.
- [72] J. Suzuki, „CVP and SVP. An introduction to the fundamental problems surrounding lattice cryptography: the closest vector problem, and the shortest vector problem,“ Brooklyn College, New York, 2015.
- [73] H. B. Nguyen, „An overview of the NTRU cryptographic system,“ San Diego State University, San Diego, 2014.
- [74] D. Rosenberg, „NTRUEncrypt and Lattice Attacks,“ KTH Numerical Analysis and Computer Science., Stockholm, Sweden., 2004.
- [75] V. Lyubashevsky, „Lattice Cryptography in the NIST Standardization Process,“ IBM Research – Zurich, Herakleon., 2018.
- [76] L. Bruckert und J.-M. Schmidt, „<kes>. Post-Quanten-Kryptografie. Sicherheit in Zeiten des Quantencomputers,“ <kes> online, 01 02 2018. [Online]. Available: <https://www.kes.info/archiv/leseproben/2018/post-quanten-kryptografie/>. [Zugriff am 18 06 2019].
- [77] Z. Zimmer, „Post-Quantum Kryptographie für IPsec,“ Universität Hamburg, Hamburg, 2014.
- [78] T. Dr. Pöppelman, J. Dr. Haid und P. Schmitz, „Security Insider. Post-Quantum-Kryptographie. Die Zukunft der Kryptographie im Zeitalter der Quanten,“ Vogel IT-Medien GmbH, 26 09 2017. [Online]. Available: <https://www.security-insider.de/die-zukunft-der-kryptographie-im-zeitalter-der-quanten-a-645548/>. [Zugriff am 18 06 2019].
- [79] D. W. Hoffmann, *Einführung in die Informations- und Codierungstheorie*, Berlin Heidelberg: Springer-Verlag, 2014.
- [80] D. Loebenberg, „Code-based Cryptography,“ genua. A Bundesdruckerei Company, Herakleon, 2018.
- [81] W. Globke, „Lineare Codes,“ Universität Karlsruhe, Karlsruhe, 2015.
-

- [82] W. Geiselmann, J.-M. Bohli und S. Röhrich, „Public Key Kryptographie. Vorlesungsskript 2006/2007,“ Universität Karlsruhe, Karlsruhe, 2006/2007.
- [83] M. Daum, „Das Kryptosystem HFE und quadratische Gleichungssysteme über endliche Körper,“ Universität Dortmund, Dortmund, 2001.
- [84] B. Preneel und C. Wolf, „Taxonomy of Public Key Schemes based on the problem of Multivariate Quadratic equations,“ K.U.Leuven, ESAT-COSIC, Leuven-Heverlee, Belgium, 2005.
- [85] M. Meyer und S. Reith, „Elliptische Kurven in der Post-Quantum-Kryptographie,“ Springer-Verlag GmbH Deutschland, Wiesbaden, Deutschland, 2018.
- [86] W. Castryck, „Ku Leuven. Research Group Cosic, Ku Leuven. Elliptic curves are quantum dead, long live elliptic curves,“ KU Leuven, 31 05 2017. [Online]. Available: <https://www.esat.kuleuven.be/cosic/tag/wouter-castryck/>. [Zugriff am 02 07 2019].
- [87] C. Costello, D. Jao, P. Longa, M. Naehrig, J. Renes und D. Urbanik, „Efficient compression of SIDH public keys,“ Microsoft/University of Waterloo/Radboud University, Redmond, USA; Waterloo, Canada; Nijmegen, Netherlands., 2017.
- [88] K. Freiermuth, J. Hromkovic, L. Keller und B. Steffen, Einführung in die Kryptologie, Zürich, Schweiz: Springer Vieweg, 2014.
- [89] T. Haigh, „Von-Neumann-Architektur, Speicherprogrammierung und modernes Code-Paradigma. Drei Leitbilder früher Rechenanlagen,“ *zfm Zeitschrift für Medienwissenschaft*, Bd. Jg. 12 (2015) Nr. 1, 2015.
- [90] J. von Neumann, "First Draft of a Report on EDVAC," University of Pennsylvania, 1945.
- [91] I. Wegener, Komplexitätstheorie. Grenzen der Effizienz von Algorithmen, Berlin Heidelberg: Springer-Verlag , 2003.
- [92] A. Beutelspracher, J. Schwenk und K.-D. Wolfenstetter, Moderne Verfahren der Kryptographie. Von RSA zu Zero-Knowledge., Wiesbaden: Springer Fachmedien, 2015.
- [93] Prashant, „The Church-Turing-Deutsch Principle and its Fundamental Ramifications,“ Universite de Montreal, Quebec, Canada., Montreal, 2006.
- [94] D. Deutsch, "Quantum theory, the Church-Turing principle and the universal quantum computer,," Centre for Quantum Computation, Oxford, 1984.
- [95] A. Cordel und S. Dr. Kousidis, „Quantencomputer. Eine BSI-Studie zum Entwicklungsstand,“ *BSI-Magazin 2018/02. Mit Sicherheit. Verschlüsselung als Grundlage für eine sichere Digitalisierung.*, pp. 22-25, 02 2018.
- [96] C. Dr. Seidel, F. Dr. Neukart und G. Dr. Compostella, „Quantum Computing. Durch die Barrieren. Verkehrsflussoptimierung mit einem D-Wave Quantum Annealer,“ *iX. MAGAZIN für professionelle Informationstechnik*, pp. 94-99, 02 02 2018.
- [97] K. Rainer, „Spektrum.de. Quantenphysik: Einsteins unbemerkte Revolution,“ Spektrum der Wissenschaft Verlagsgesellschaft mbH, 02 04 2012. [Online]. Available: <https://www.spektrum.de/news/einsteins-unbemerkte-revolution/1147602>. [Zugriff am 09 05 2019].
- [98] G. Dr. Kalbe, „EU investment in Quantum Technologies. ENISA Summer School, 26. September 2018,“ DG CNECT, European Commission, Heraklion, 2018.
- [99] T. Pöppelmann, „Bridging 1st PQC-functions and principles with the smart card world,“ Infineon., Heraklion, 2018.
- [100] N. Dr. Polemi, „Policies in the Quantum era,“ European Commission DG CNECT/H1, Heraklion, 2018.
- [101] N. Hines, „Then And Now: The World's Very First Computer Vs. Today's Supercomputers,“ *ati.*, 09 03 2016. [Online]. Available: <https://allthatsinteresting.com/first-computer>. [Zugriff am 16 05 2019].

-
- [102] W. Zeitung, „Quantencomputer rücken näher,“ Wiener Zeitung, 15 05 2019. [Online]. Available: <https://www.wienerzeitung.at/nachrichten/wissen/technologie/2009485-Quantencomputer-ruecken-naeher.html>. [Zugriff am 16 05 2019].
- [103] T. Lange, „Cryptology, cryptography, cryptanalysis. Definitions, meanings, requirements, and current challenges.,“ PQCrypto ICT-645622, Heraklion, 2018.
- [104] W. Dr. Fumy, „Datenschutz und Datensicherheit: Quantencomputer und die Zukunft der Kryptographie,“ Springer Fachmedien Wiesbaden GmbH, Wiesbaden, 2017.
- [105] L. Bruckert und J.-M. Schmidt, „Post-Quanten-Kryptographie. Sicherheit in Zeiten des Quantencomputer.,“ <kes>. *Die Zeitschrift für Informations-Sicherheit.*, pp. 54-59, 01 02 2018.
- [106] I. Kabanov, R. Yunusov, K. Y.V. und A. Fedorov, „Practical Cryptographic Strategies in the Post-Quantum Era,“ Cornell University, Cambridge, Massachusetts, 2018.
- [107] T. Lange, „Cryptology, cryptography, cryptanalysis. Definitions, meanings, requirements, and current challenges.,“ Technische Universiteit Eindhoven., Herakleon. Kreta., 2018.
- [108] S. Aaronson, „Three Questions about Quantum Computing.,“ University of Texas., Austin, USA., 2018.
- [109] J. Buchmann, „Post-Quantum Cryptography - an overview.,“ Technische Universität Darmstadt., Hitotsubashi Hall, Tokyo, 2015.
- [110] T. Pöppelmann, „Bridging 1st PQC-functions and principles with the smart card world.,“ infineon., Herakleon, Griechenland., 2018.
- [111] S.-L. Gazdag, „Hash-Based Signatures.,“ Genua. A Bundesdruckerei Company., Herakleon, Greek., 2018.
- [112] M. Sjöberg, „Post-quantum algorithms for digital signing in Public Key Infrastructures,“ TH Royal Institute of Technology., Stockhol, Schweden, 2017.
- [113] M. O'Neill, „Practical Implementation of Lattice-based cryptography.,“ SAFEcrypto, Herakleon, Griechenland., 2018.
- [114] R. Niebuhr, M. Mezzani, S. Bulygin und J. Buchmann, Selecting parameters for secure McEliece-based cryptosystems., Darmstadt: Springer-Verlag, 2012.
- [115] P. Luttmann, „Kryptologische Methoden und Alternativen.,“ Leibniz Universität Hannover., Hannover, 2015.
- [116] R. Misoczki, J.-P. Tillich und N. B. P. S. L. M. Sendrier, „MDPC-McEliece: New McEliece Variants from Moderate Density Parity-Check Codes.,“ Project SECRET/Escola Polit´ecnica, Universidade de Sao Paulo, France/Brazil, 2012.
- [117] J. Ding und D. Schmidt, „Multivariable public-key cryptosystems,“ University of Cincinnati, Cincinnati, USA, 2004.
- [118] S. Heyse, „Post Quantum Cryptography: Implementing alternative public key schemes on embedded devices. Preparing for the rise of Quantum Computers.,“ Ruhr-University Bochum, Bochum, Germany., 2013.
- [119] M. Pierrakea, „Supersingular isogeny key-exchange.,“ Univeristé de Bordeaux, Bordeaux, 2016/17.
- [120] C. Costello, „An introduction to supersingular isogeny-based cryptography (Netherlands).,“ Microsoft Research, Nijmegen, The Netherlands, 2017.
- [121] C. Costello, „An introduction to supersingular isogeny-based cryptography (Croatia).,“ Microsoft Research, Sibenik, Croatia., 2017.
- [122] A. Vahrner und M. Christler, „Quantencomputer: Innsbrucker Physiker schaffen Durchbruch,“ Tiroler Tageszeitung., 15 05 2019. [Online]. Available: <https://www.tt.com/panorama/wissen/15645518/quantencomputer-innsbrucker-physiker-schaffen-durchbruch>. [Zugriff am 16 05 2019].
- [123] H. Bröck, „Golem.de,“ Golem Media GmbH, 08 07 2016. [Online]. Available: <https://www.golem.de/news/new-hope-google-testet-post-quanten-algorithmus-1607-121989.html>. [Zugriff am 17 06 2019].
-

- [124] C. Gartenberg, „D-Wave is now shipping its new \$15 million, 10-foot tall quantum computer,“ The Verge. Vox Media, Inc., 25 01 2017. [Online]. Available: <https://www.theverge.com/circuitbreaker/2017/1/25/14390182/d-wave-q2000-quantum-computer-price-release-date>. [Zugriff am 16 05 2019].
- [125] A. Randall 5th, „Q&A: A lost interview with ENIAC co-inventor J. Presper Eckert. On the the 60th anniversary of the unveiling of ENIAC, a newly discovered interview with "Pres" Eckert explodes some ENIAC myths.,“ Computerworld, 14 02 2006. [Online]. Available: <https://www.computerworld.com/article/2561813/q-a-a-lost-interview-with-eniac-co-inventor-j--presper-eckert.html>. [Zugriff am 16 05 2019].
- [126] U. Ostler, „Vorglühen im Quantencomputing. D-Wave knackt mit 2.048-Qubit Prozessor bisher unlösbares Magnetismusproblem.,“ DataCenter Insider. Vogel IT-Medien GmbH., 25 07 2018. [Online]. Available: <https://www.datacenter-insider.de/d-wave-knackt-mit-2048-qubit-prozessor-bisher-unloesbares-magnetismusproblem-a-736032/>. [Zugriff am 16 05 2019].
- [127] A. Tillemans, „wissenschaft.de,“ Konradin Medien GmbH, 15 02 2001. [Online]. Available: <https://www.wissenschaft.de/technik-digitales/einsteins-spukhafte-fernwirkung-ist-realitaet/>. [Zugriff am 09 05 2019].
- [128] H. Wiseman, "nature. International weekly journal of science. Physics: Bell's theorem still reverberates,“ Nature, Springer Nature., 19 06 2014. [Online]. Available: <http://www.nature.com/news/physics-bell-s-theorem-still-reverberates-1.15435>. [Accessed 09 05 2019].
- [129] N. Wolchover, „Spektrum.de. Teilchenphysik. Wie real die die Quantenverschränkung?,“ Spektrum der Wissenschaft Verlagsgesellschaft mbH, 11 04 2017. [Online]. Available: <https://www.spektrum.de/news/wie-real-ist-die-quantenverschraenkung/1445463>. [Zugriff am 09 05 2019].
- [130] CNSS Secretariat (I42) . National Security Agency., „FACT SHEET - CNSS Policy No. 15, Fact Sheet No. 1. National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Security Information,“ 06 2003. [Online]. Available: <https://csrc.nist.gov/csrc/media/projects/cryptographic-module-validation-program/documents/cnss15fs.pdf>. [Zugriff am 14 04 2019].
- [131] R. Dr. Niederhagen und M. Prof. Dr. Waidner, „White Paper - Practical Post-Quantum Cryptography.,“ Fraunhofer Institute for secure information technology., Darmstadt., 2017.
- [132] F.-I. f. S. Informationstechnologie, „CRISP - Fraunhofer SIT - Forschung für die Kernfragen der Cybersicherheit.,“ Fraunhofer SIT, [Online]. Available: <https://www.sit.fraunhofer.de/de/angebote/gremien-netzwerke/crisp/>. [Zugriff am 10 07 2019].
- [133] L. Dr. Chen, „Post-Quantum Cryptography. Round 2 Submissions.,“ NIST. National Institute of Standards and Technology., 08 07 2019. [Online]. Available: <https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-2-Submissions>. [Zugriff am 11 07 2019].
- [134] C. Diem und K. Schmeh, „Gegen die Apokalypse. Algorithmenwettbewerb zur Post-Quanten-Kryptographie.,“ *iX. Magazin für professionelle Informationstechnik.*, Nr. 06/2018, pp. 116-120, 2018.
- [135] Internet Engineering Task Force (IETF) , „RFC7296: Internet Key Exchange Protocol Version 2 (IKEv2),“ [Online]. Available: <https://tools.ietf.org/html/rfc7296>. [Zugriff am 15 07 2019].
- [136] P. Véron, „Code based cryptography and steganography,“ Springer Verlag, Porquerolles, France, 2013.
- [137] Internet Engineering Task Force (IETF) , „Internet Key Exchange Protocol Version 2 (IKEv2),“ Internet Engineering Task Force (IETF) , October 2014. [Online]. Available: <https://tools.ietf.org/html/rfc7296>. [Zugriff am 16 07 2019].

- [138] E. Dipl.-Inf. Zimmer, „Universität Hamburg.“, 20 11 2014. [Online]. Available: <https://svs.informatik.uni-hamburg.de/publications/2014/2014-11-20-Zimmer-CAST-PQC-fuer-IPsec.pdf>. [Zugriff am 19 07 2019].
- [139] A. Steffen und M. Willi, „VPNs mit Strongswan-IKEv2,“ Linux Magazin Online., 02 2009. [Online]. Available: <https://www.linux-magazin.de/ausgaben/2009/02/tunnel-design-2/>. [Zugriff am 19 07 2019].
- [140] V. Lynch, „Google’s Post-Quantum Cryptography Experiment Successful,“ hashedout. The SSL Store., 30 11 2016. [Online]. Available: <https://www.thesslstore.com/blog/googles-post-quantum-cryptography-experiment-successful/>. [Zugriff am 19 07 2019].
- [141] B. Huttner und J. Melia, „Applied quantum-safe security: Quantum-resistant algorithms and quantum key distribution. Cloud Security Alliance.,“ 2017. [Online]. Available: <https://downloads.cloudsecurityalliance.org/assets/research/quantum-safe-security/applied-quantum-safe-security.pdf>. [Zugriff am 22 07 2019].
- [142] W. Whyte, Z. Zhang, S. Fluhrer und O. Garcia-Morchon, „Quantum-Safe Hybrid (QSH) Key Exchange for Transport Layer Security (TLS) version 1.3.,“ 31 03 2017. [Online]. Available: <https://tools.ietf.org/html/draft-whyte-qsh-tls13-04>. [Zugriff am 20 07 2019].
- [143] PQCRIPTO, „Initial recommendations of long-term secure postquantum systems.,“ 07 09 2015. [Online]. Available: <http://pqcrypto.eu.org/docs/initial-recommendations.pdf>. [Zugriff am 22 07 2019].
- [144] J. A. S. Velázquez, „Practical Implementations of Quantum-Resistant Cryptography,“ 18 12 2017. [Online]. Available: https://courses.cs.ut.ee/MTAT.07.022/2017_fall/uploads/Main/antonio-report-f17.pdf. [Zugriff am 22 07 2019].
- [145] P. Kampanakis, P. Panburana, E. Daw und D. Van Geest, „The Viability of Post-Quantum X.509 Certificates.,“ 2018. [Online]. Available: <https://eprint.iacr.org/2018/063.pdf>. [Zugriff am 22 07 2019].
- [146] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," AT&T Research, 1996.
- [147] D. J. Bernstein, J. Buchmann and E. Dahmen, Post-Quantum Cryptography, Springer, 2009.
- [148] A. Cordel und S. Dr. Kousidis, „Quantencomputer,“ *BSI-Magazin, Referat Kryptographische Vorgaben und Entwicklungen*, Bd. 02, Nr. 2018/02, p. 24, 2018.
- [149] . E. Conover, „Quantum computers take a step forward with a 50-qubit prototype,“ *ScienceNews - Magazine of the society for science & the public*, Bd. No. 6, Nr. Vol. 193, 2018.
- [150] T. Pöppelmann, „Bridging 1st PQC-functions and principles with the smart card world by Infineon,“ in *NIS Summer School 2018*, Herakleon, Kreta, Griechenland, 2018.
- [151] PQCRIPTO ICT-645622, „libpqcrypto. Version 20180314.,“ PQCRIPTO ICT-645622, 23 03 2018. [Online]. Available: <https://libpqcrypto.org/>. [Zugriff am 17 07 2019].
- [152] An, Hyeongcheol; Choi, Rakyong; Lee, Jeeun; Kim, Kwangjo, „Performance Evaluation of liboqs in Open Quantum Safe Project (Part I).,“ 23 01 2018. [Online]. Available: <https://pdfs.semanticscholar.org/7ef2/2ef120fd8dcc64a7e1865b99b02e37ad3a48.pdf>. [Zugriff am 22 07 2019].

