Post Quanten Kryptographie

Warum wir schon jetzt neue Verschlüsselungsalgorithmen brauchen

Thomas Jakkel
MatNr. 1001594
01jath1bif@hft-stuttgart.de

Lukas Reinke
MatNr. 1001213
01relu1bif@hft-stuttgart.de

***Zusammenfassung*—Quantenkryptographie wird als ernsthafte Gefahr für die aktuell gängigen Kryptographie-Verfahren gesehen. Speziell die, für das sichere Funktionieren der Netzwerk und Internet Kommunikation benötigten, asymmetrischen Kryptographie-Algorithmen wie beispielsweise *RSA* stehen in der Gefahr mithilfe eines funktionierenden Quantencomputers binnen weniger Minuten gebrochen zu werden. Ein solcher Computer birgt somit eine Gefahr für die gesamte heutige Informationssicherheit.**

Schon 1994 bewies der amerikanische Mathematiker *Peter Shor* in der Theorie, wie mithilfe eines Quantencomputers die Primfaktorzerlegung großer Zahlen in realer Zeit erfolgen kann. Mit einem klassischen, digitalen Computer würde eine solche Berechnung eine längere Zeit als die Existenz des Universums dauern. Dies stellt das zentrale Sicherheits-Prinzip asymmetrischer Kryptographie dar. Wegen dieser Bedenken hat das US National Institute of Standards and Technology (NIST) seit 2016 eine Ausschreibung zur Entwicklung eines quantensicheren Algorithmus aufgestellt. Der aktuelle Favorit, ein Verfahren auf basis mehrdimensionaler Vektorfelder, soll, soweit sich keine Schwachstellen herausstellen, schon in den nächsten Jahren in der Netzwerk Kryptographie etabliert werden und somit die Kommunikation schon vor der Existenz eines potentiellen Quantencomputers absichern. Es ist also gut möglich, dass die aktuellen Kryptographie Standards innerhalb der nächsten 5 Jahre ausgetauscht werden.

Abbildungsverzeichnis

Abkürzungsverzeichnis

QC Quantencomputer
QR Quantenregister
QT Quantenteilchen
ggT größter gemeinsamer Teiler
QFT Quanten-Fourier-Transformation
NIST National Institute of Standards and Technology
LWE Learning With Errors
SVP Shortest Vector Problem

I. Einleitung

Das Thema der *Quanten Supremacy*, der Zeitpunkt zu dem ein QC die Fähigkeit besitzt komplexe Probleme besser zu lösen als ein klassischer Computer, ist in den letzten Jahren immer wieder in einschlägigen Medien und diverse Fachpublikationen aufgetaucht. Google zum Beispiel behauptete schon wiederholt einen solchen QC zu besitzen, was jedoch in diversen Publikationen bezweifelt wurde [1]. Wenig Zweifel besteht jedoch, dass ein solcher Computer in absehbarer Zeit einsatzbereit sein wird. Im Folgenden wird beschrieben, wie ein QC funktioniert und mit welchen Algorithmen, die in realer Zeit laufen, QC die aktuellen, kryptographischen Verfahren gebrochen werden können. Außerdem

werden wir neu entwickelte, quantensichere Algorithmen betrachten und wie sie die Gefahr durch QC mitigieren werden.

II. Einführung in das Quanten Computing

Quanten Computing ist eine relativ junge Disziplin der Physik und Informatik. Obwohl die Entwicklung der beiden Bereiche unabhängig voneinander zu Beginn des 20. Jahrhunderts begann, wurden schon in den 1980er Jahren die ersten Versuche unternommen und theoretische Überlegungen angestellt, um Methoden der Informatik mithilfe von Quantenobjekten umzusetzen. Obwohl der QC erst seit kurzer Zeit existiert, hat er bereits bedeutende Fortschritte gemacht und besitzt ein enormes Potenzial für Entwicklungen zum Beispiel in den Bereichen Kryptographie, Optimierungsprobleme, Simulation chemischer Prozesse und maschinelles Lernen.

A. Ein theoretischer Quantencomputer

Ein der Informatik zugrunde liegendes Prinzip ist, dass [2, S122] Informationen auf unterschiedlichen Weisen dargestellt (codiert) werden können. So kann die Zahl *fünf* zum Beispiel binär (0101) oder als Unicode Zeichen (*U+0035*) dargestellt werden. Der Inhalt der Information ist in beiden Fällen jedoch der Gleiche.
Dieses Prinzip ist für die Informatik sehr wichtig. Es ermöglicht Maschinen komplexe Informationen zu speichern, mithilfe einfacher Operationen zu manipulieren und wieder in ein komplexes Format zu überführen, ohne dabei an Informationsgehalt zu verlieren oder Informationen unkenntlich zu machen. In einem Computer werden diese Informationen in Form von *bits* darstellt die zwei Zustände, repräsentiert durch *0* und *1*, annehmen können. In einem klassischen, digitalen Computer sind diese beispielsweise durch das fließen von Strom abgebildet. Ein QC repräsentiert Informationen in den Eigenschaften von Quanten-Objekten, beispielsweise dem Spin von Elektronen.
Um die Funktion eines QC zu erläutern muss erst ein kurzer Blick auf einige quantenphysikalische Phänomene geworfen werden.

1) Superpositionen: Eine Superposition ist das Fundamentale quantenphysische Phänomen das einem QC zugrunde Liegt. Es wird meistens mithilfe des Gedankenexperiments von [3, $5] *Schrödingers Katze* veranschaulicht: In einer Box befindet sich eine Katze und ein Gefäß mit Gift das zerstört wird und die Katze tötet, wenn ein radioaktiver Zerfall gemessen wird. Außerhalb der Box lässt sich nicht feststellen, ob der Mechanismus der das Gift freisetzt aktiviert wurde. Sie befindet sich, Quantenmechanisch gesehen, in einem Zustand in dem sie sowohl tot als auch lebendig ist. In dem Moment, in dem die Box geöffnet wird, kann festgestellt werden welcher der beiden Zustände eingetroffen ist.
Auf ein Quantenteilchen (QT) übertragen heißt das: Es kann sich in einem undefinierten Zustand befinden, der **Superposition** aus allen möglichen Zuständen. Erst wenn *nachgeschaut*, also der Zustand gemessen wird, kollabiert die Superposition in einen der möglichen Zustände. Eine Superposition kann durch die Wahrscheinlichkeit, mit der jeder der Zustände eintreten kann, beschrieben werden. Ein Elektron mit den Zuständen Spin up (75%) und Spin down (25%) kollabiert also, wenn der Spin gemessen wird zu $3/4$ der Fälle als Spin up und zu $1/4$ als Spin down. Mathematisch kann eine Superposition ψ als Linearkombination ihrer Zustände betrachtet werden.

$$|\psi\rangle = \alpha\,|1\rangle + \beta\,|0\rangle \tag{1}$$

Diese Linearkombination kann auch grafisch, als Vektor auf einer Kugel dargestellt werden (siehe Graphik 1). Es ist zu beachten, dass der Vektor z nicht den Spin, sondern die Linearkombination darstellt.

2) Quantenverschränkung: Das zweite quantenphysikalische Phänomen auf dem ein Quantencomputer basiert ist die **Quantenverschränkung**. Der Zustand verschränkter QT is voneinander abhängig. Verändert sich der Zustand eines einzelnen QTs Verändert sich der Zustand aller mit ihm verschränken QT. Das bedeutet auch: Wird der Zustand eines QTs gemessen und seine die Superposition kollabiert, so kollabiert die

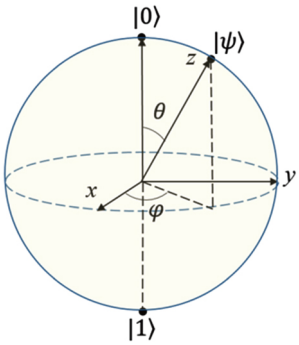

Abbildung 1. Spin eines Elektrons in Superposition [4]

Superposition aller verschränken QT [5].

B. Implementierung eines Quantencomputers

Mit den oben erläuterten Phänomenen sind wir nun in der Lage einen Quantencomputer zu konstruieren. Ein klassischer Computer kann in einem Bit immer nur eine Information auf ein mal darstellen. In zwei bit kann somit eine der Kombinationen aus dem Folgenden Set darstellen.

$$KlassischerComputer : \{00, 01, 10, 11\} \quad (2)$$

Ein QC hingegen kann mithilfe verschränkter QT das Set aus allen Möglichen Informationen repräsentieren.

$$Quantencomputer : |\psi_0\psi_1\rangle \quad (3)$$

Was alle Elemente aus dem folgenden Set repräsentiert.

$$\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\} \quad (4)$$

[2, S. 142]

Die Herausforderung ist nun die richtige Lösung aus der Menge aller möglicher Lösungen heraus zu extrahieren. Messen wir den Zustand eines QT so kollabieren die verschränken Superpositionen zu einer der möglichen Informationen, alle Informationen können nicht auf ein mal ausgelesen werden. Auch ist es nicht möglich ein mal gemessene QTs wieder in eine Superposition zu versetzen. Wie ist es nun möglich mit einem QC Berechnungen anzustellen?

1) Quantenregister: Ein Quantenregister (QR) ist ein grundlegender Bestandteil eines QC und repräsentiert den Speicher für Quanteninformationen. Es ist vergleichbar mit einem klassischen Register in einem herkömmlichen Computer.

Ein QR besteht aus einer Anordnung von Qubits und dient als Container, um diese zu speichern und sie miteinander zu manipulieren. Die Anzahl der Qubits in einem QR bestimmt die Größe und Komplexität der Berechnungen, die ein QC durchführen kann. Je größer das QR ist, desto mehr Qubits stehen zur Verfügung, um komplexe Berechnungen durchzuführen.

Durch die Anwendung von Quantengattern, wie beispielsweise Hadamard-Gattern, CNOT-Gattern und Phasengattern, können Operationen auf den Qubits im QR ausgeführt werden. Diese Gatter ermöglichen es, Qubits zu verschränken, Superpositionen zu erzeugen und gezielte Manipulationen der Quanteninformationen durchzuführen.

Das QR spielt eine zentrale Rolle in Quantenalgorithmen, da es die Plattform bietet, auf der die Quantenberechnungen durchgeführt werden. Die Größe und Qualität des QRs sind entscheidend für die Leistungsfähigkeit eines Quantencomputers. Daher ist die Entwicklung und Skalierung von QRn ein aktiver Bereich der Forschung, um die Möglichkeiten der Quantencomputertechnologie weiter zu verbessern [6].

III. RSA

IV. Shors-Algorithmus

Die Faktorisierung großer Zahlen [7, S189] ist ein fundamentales mathematisches Problem, bei dem eine gegebene Zahl in ihre Primfaktoren zerlegt wird. Dieses Problem ist von zentraler Bedeutung für die Kryptographie, da viele asymmetrische Verschlüsselungsverfahren, wie beispielsweise RSA, auf der Schwierigkeit der Faktorisierung großer Zahlen beruhen.

Das Problem der Primfaktorzerlegung besteht darin, eine öffentliche Zahl N in zwei oder mehr geheime Primzahlen p und q zu zerlegen, sodass:

$N = p * q$. Für kleine Zahlen kann die Faktorisierung durch Ausprobieren möglicher Primfaktoren relativ einfach sein. Allerdings wird die Faktorisierung bei großen Zahlen exponentiell schwieriger, da es keine effizienten klassischen Algorithmen gibt, welche dies in Polynomialzeit bewältigen können.

Die Sicherheit vieler asymmetrischer Verschlüsselungsverfahren, wie beispielsweise das RSA-Verfahren, beruht auf der Schwierigkeit der Faktorisierung großer Zahlen. Wenn ein Angreifer in der Lage wäre, die Primfaktoren einer Zahl zu finden, könnte er den geheimen Schlüssel einer Verschlüsselungsmethode berechnen und die Sicherheit des Systems kompromittieren.

Shors Algorithmus [8] gilt als einer der bedeutendsten Quantenalgorithmen, der die Faktorisierung großer Zahlen in Polynomialzeit ermöglicht. Auch wenn dieser Algorithmus einen QC mit vielen, stabilen Qubits voraussetzt, stellt diese revolutionäre Entdeckung bereits heute eine enorme Bedrohung für die moderne Kryptographie dar. Zum einen wird effektiv an QC geforscht, die Anzahl stabiler Qubits in QC steigen. Wenn der QC seinen Durchbruch in unserer Gsellschaft erreicht hat, müssen wir darauf vorbereitet sein und uns bereits heute mit den Konsequenzen auseinander setzen. Zum anderen werden verschlüsselten Daten bereits heute gespeichert, um sie in der Zukunft entschlüsseln zu können und im Nachinein an wichtige, vertrauliche Informationen zu gelangen. In der Kryptographie spricht man von dem Prinzip: „Safe now, decrypt later“.

Im folgenden [9][10, S. 4-5] werden wir an einem vereinfachten Beispiel näher aufzeigen, wie Shors Algorithmus eine effiziente Lösung für die Faktorisierung großer Zahlen auf einem QC bietet. Zunächst betrachten wir die öffentliche Zahl N, welche in ihre Primfaktoren p und q zerlegt werden soll. Für das vereinfachte Beispiel wählen wir $N = 77$. Also:

$$N = p * q \text{ mit } N = 77$$

Beachte das N in Realität eine riesige Zahl ist. Das vereinfachte Beispiel soll ledeglich den Prozess von Shors Algorithmus aufzeigen.

Nun stellen wir eine zweite Gleichung

$$g^r = mN + 1$$

auf, welche behauptet, dass man immer einen Exponenten r finden kann, sodass ein Vielfaches einer zufälligen Zahl $g < N$ gleich einem vielfachen m der Zahl $N + 1$ ist. Wir wählen zufällig die Zahl $g = 8$ und erhalten aus Abbildung 2 $r = 10$

g^x	$g^x/77$	Rest
g^0	0	1
g^1	0	8
g^2	0	64
g^3	6	50
g^4	53	15
g^5	425	43
g^6	3404	36
g^7	27235	57
g^8	217885	71
g^9	1733087	29
g^{10}	13944699	1

Abbildung 2. Zufallszahl g^x mit Rest

Wie extrahiert man aus der Funktion $g^{10} = mN + 1$ die Primfaktoren p, q? Hierfür schreiben wir unsere ursprüngliche Gleichung um und erhalten:

$$\begin{aligned} & g^r = mN + 1 \\ \Leftrightarrow\ & g^r - 1 = mN \\ \Leftrightarrow\ & g^r - 1 = mN \\ \Leftrightarrow\ & (g^{r/2} + 1)(g^{r/2} - 1) = mN \end{aligned}$$

Mit $r = 10$ erhalten wir:

$$\begin{aligned} (g^{r/2} + 1) = (8^5 + 1) \qquad &= 32.769 \\ (g^{r/2} - 1) = (8^5 - 1) \qquad &= 32.767 \end{aligned}$$

Damit haben wir aus einer geschätzten Zahl $g = 8$ zwei Zahlen gefunden, welche vermutlich Faktoren mit N teilen. Um diese Faktoren zu finden, wenden wir den euklidischen Algorithmus an, um den größter gemeinsamer Teiler (ggT) zu finden:

$$
\begin{aligned}
&32.769/77 && = 425R44\\
&77/44 && = 1R33\\
&44/33 && = 1R11\\
&33/11 && = 3R0\\
\Rightarrow\ & ggT(32.769, 77) = 11
\end{aligned}
$$

Wir erhalten somit als ersten Faktor $p = 11$. Für den zweiten Faktor kann der selbe Prozess nochmals mit der zweiten Zahl berechnet werden, oder wir erhalten durch $q = N/p \Leftrightarrow q = 7$. Damit haben wir erfolgreich die Zahl $N = 77$ in ihre Primfaktoren $p = 11$ und $q = 7$ zerteilt.

Nachfolgend fassen wir nochmal die Schritte zusammen, wie man eine eine öffentliche Zahl N, ein Produkt aus zwei Primzahlen faktorisiert.

1) Schätze eine Zahl $g < N$
2) Finde einen Exponenten r sodass $g^r = mN + 1$
3) Berechne $(g^{r/2} + 1)$ und $(g^{r/2} - 1)$
4) Nutze den euklidischen Algorithmus um den größter gemeinsamer Teiler (ggT) zu finden

Um diesen Algorithmus auszuführen bräuchte es noch keinen QC, jedoch wäre diese Methode auf einem klassischen Computer nicht schneller, als herkömmliche Methoden. Der Kernprozess, welcher ein QC beschleunigt ist Schritt 2. Um dies zu verstehen, betrachten wir Abbildung 2 nochmal genauer. Würde man die Tabelle fortführen, würde sich der Restanteil der Zahlen stets wiederholen. In unserem obrigen Beispiel mit der Periode 10, wie sich aus $g^0 Rest1$ und $g^{10} Rest1$ erkennen lässt. Dies gilt ebenfalls für die anderen Zahlen, beispielsweise würde $8^{13} Rest50$ haben. Dies bedeutet, dass sich Schritt 2 beschleunigen lässt indem man die Periode findet, mit der sich der Restanteil wiederholt. An dieser Stelle setzt der Quantenpart ein.

Für die Quantenberechnung [11, S.1-2] werden zwei QR benötigt. Ein Periodenregister n_p, um den Wert der Periode zu speichern und ein Ergebnisregister n_q um das Ergebnis der Berechnung zu speichern. Die Größe der beiden Register hängt von der Zahl N ab, welche faktorisiert werden soll. Das Periodenregister benötigt eine Anzahl von Qubits n_p, von etwa $\log_2(N^2) \leq n_p \leq log_2(2N^2)$. Das Ergebnisregister muss ledeglich groß genug sein, um die Zahl $N-1$ zu speichern, was $n_q = \log_2(N)$ Qubits benötigt. Zu Beginn wird das Periodenregister fortlaufend von 1 initialisiert:

$$|n_p\rangle = |0\rangle + |1\rangle + |2\rangle + ... + \left|10^{1234}\right\rangle$$

Diese Zahlen stellen den Exponenten r dar. Das Ergebnisregister wird vorerst mit 0 initialisiert:

$$|n_q\rangle = |0\rangle + |0\rangle + |0\rangle + ... + |0\rangle$$

Nun nehmen wir unsere geschätzte Zahl $g < N$ und potenzieren diese mit der Menge des Periodenregisters n_p, teilen diese durch die Zahl N und speichern den Rest R im Ergebnisregister n_q.

$$\left|\frac{g^0}{N}\right\rangle,\ \left|\frac{g^1}{N}\right\rangle,\ \left|\frac{g^2}{N}\right\rangle,\ ...$$

$$|R_0\rangle,\ |R_1\rangle,\ |R_2\rangle,\ ...$$

Im jetzigen Stadium haben wir das unveränderte Periodenregister n_p, welches die Zahlen r der Folge $1, 2, 3...$ beinhaltet und das Ergebnisregister n_q in welchem nun der Restanteil R von $\frac{g^r}{N}$ liegt.

$$|0\rangle |R_0\rangle + |1\rangle |R_1\rangle + |2\rangle |R_2\rangle + ...$$

Die Herausforderung besteht nun darin, aus der Superposition von allen möglichen Lösungen, die eine Richtige zu extrahieren. Der Trick ligt darin sich nur den Rest anzuschauen. Betrachten wir nochmal unser obriges Beispiel Abbildung 2. Sobald man eine mögliche Lösung misst, beispielsweise den Rest $R = 15$, wird dieser Rest periodisch erneut auftauchen, nämlich

$$|4\rangle |R_{15}\rangle + |14\rangle |R_{15}\rangle + |24\rangle |R_{15}\rangle + ...$$

Nachdem man also einen Restanteil misst, erhält man eine Menge von Superposition, welche den selben Restanteil besitzen. Und die Exponenten sind genau um die Periode r voneinander getrennt.

$$|i\rangle |R\rangle + |i+r\rangle |R\rangle + |i+2r\rangle |R\rangle + \ ...$$

Da der Restanteil hier immer der Gleiche ist, kann dieser vernachläsigt werden und man erhält somit eine periodische Superposition.

$$|i\rangle + |i+r\rangle + |i+2r\rangle + \ ...$$

Im letzten Schritt kann die Quanten-Fourier-Transformation (QFT) angewendet werden, welche es erlaubt die Frequenz einer periodischen Superposition zu ermitteln. [11, S. 2] Wenden wir also auf die obrige periodische Superposition die QFT an, können wir aus dem Ergebnis r auslesen.

V. Lattice-Kryptographie

Im Jahr 2016 hat das US-amerikanische NIST einen weltweit einheitlichen Standardisierungsprozess für Post-Quantum-Kryptographie gestartet [12]. 2022 wurden mehrere Algorithmen Ausgewählt. Die meisten dieser Algorithmen basieren auf mathematischen Gitter- (*engl. Lattice*) Problemen.

Lattice-Kryptographie [13] ist ein Bereich der post-quanten Kryptographie. Mit dieser werden Verschlüsselungsalgorithmen auf der Grundlage mathematischer Gitterstrukturen entwickelt. Ein Gitter ist eine diskrete, periodische Anordnung von Punkten im Raum. In der Kryptographie werden meistens Gitter in mehrdimensionalen Vektorräumen verwendet. Die Gitterpunkte werden durch Vektoren repräsentiert, die eine lineare Kombination von Basisvektoren des Gitters sind. Ein Beispiel für ein zweidimensionales Gitter ist die Ebene, in der die Basisvektoren (1, 0) und (0, 1) sind, und die Gitterpunkte alle ganzzahligen Kombinationen dieser Basisvektoren.

Ein grundlegendes Problem in der Lattice-Kryptographie ist das Shortest Vector Problem (SVP), bei dem man den kürzesten Vektor in einem Gitter finden muss. Dieses Problem gilt als NP-vollständig [14, Abs. 2.1]. Ein weiteres Problem ist das Learning With Errors (LWE), bei dem man die Lösung eines linearen Gleichungssystems mit fehlerbehafteten Variablen in einem Gitter finden muss. Die fehlerbehafteten Werte werden durch eine Fehlerfunktion zufällig erzeugt. Diese fügt den Gleichungen Rauschen hinzu und erschwert es, den geheimen Wert aus den Gleichungen zu erlernen.

Diese Probleme sind jedoch leicht zu lösen, wenn man eine gute Basis an Vektoren hat. In Abbildung 3 ist eine gute Basis B' in grün dargstellt. Diese besteht aus kurzen Vektoren, welche orthogonal zueinander stehen. Dadurch lässt sich jeder Gitterpunkt durch viele verschiedene Linearkombinationen einfach erreichen. Eine schlechte Basis hingegen besteht aus langen Vektoren, welche in ähnliche Richtungen zeigen. In Abbildung 3 ist die schlechte Basis B rot dargestellt. Will eine Person A, eine verschlüsselte Nachricht an Person B schicken, teilen sie das Gitter mit den schlechten Vektoren öffentlich. Person A und B besitzen jedoch das private, gute Vektorpaar. Person A wählt einen Punkt t in der nähe eines Gitterpunkts, in Abbildung 3 lila dargestellt. Person B muss nun den Gitterpunkt t' herausfinden, welcher am nächsten zum lilanen Punkt t liegt. Da Person B das gute Vektorpaar besitzt, ist das keine schwere Aufgabe. Eine andere Person müsste das jedoch mit dem schlechten Vektorpaar bewältigen, was bis heute als enorm schwierig gilt.

In diesem Beispiel gehen wir zur Veranschaulichung von einem 2D-Gitter aus. Diese Aufgabe wird jedoch exponentiell schwieriger, für jede Dimension welche hinzukommt. In einem 1000D-Gitter würde das selbst ein QC nicht schaffen [15, Abs. 3].

Nachfolgend wird das LWE-Problem genauer erläutert. Gegeben sei ein Vektor s, der den geheimen Schlüssel darstellt, und ein Vektor e, der den Fehler darstellt. Das LWE-Problem besteht darin, den geheimen Vektor s aus einer Reihe von Gleichungen der Form

$$c = (a * s + e) \mod q$$

zu erlernen, wobei c der Ciphertext und a ein öffentlicher Vektor ist. Das Modulo q repräsentiert

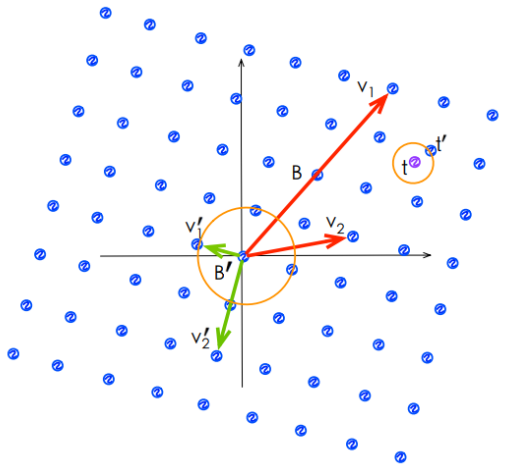

Abbildung 3. Darstellung eines 2D Gitters[15, Fig. 2]

eine endliche Gruppe, in der die Rechenoperationen durchgeführt werden.

Um das LWE-Problem zu lösen, werden eine Reihe von Gleichungen erzeugt. Der geheime Vektor s und der Fehlervektor e werden zufällig generiert. Der öffentliche Vektor a wird ebenfalls zufällig gewählt. Für jeden Gitterpunkt i wird die Gleichung

$$c_i = (a * s_i + e_i) \mod q$$

erstellt.

Der Angreifer erhält eine Menge von Gleichungen, bestehend aus den Ciphertexts c_i und den öffentlichen Vektoren a_i. Das Ziel des Angreifers ist es, den geheimen Vektor s_i zu erlernen, basierend auf den Ciphertexts und den öffentlichen Vektoren. Dies erfordert die Bestimmung der Werte des geheimen Vektors s_i trotz des Vorhandenseins von Fehlern e_i.

Die Schwierigkeit des LWE-Problems beruht darin, die Werte des geheimen Vektors s_i aus den gegebenen Gleichungen zu rekonstruieren. Der Angreifer muss die richtigen Werte des geheimen Vektors, von den fehlerbehafteten Gleichungen unterscheiden können. Daher ist es von zentraler Bedeutung, dass die fehlerbehafteten Gleichungen ausreichend komplex sind, um eine effektive Lösung des LWE-Problems zu verhindern.

Im Bereich Performance ist die Lattice-Kryptographie schneller als herkömmliche asymmetrische Verfahren, wie beispielsweise RSA. Das liegt u.a. daran, dass bei Gittern mit vergleichsweise kleinen Zahlen gearbeitet wird. Der aktuelle Stand der Technik in der gitterbasierten Kryptographie basiert hauptsächlich auf einfachen Operationen mit Matrizen und Vektoren in Ringen oder Feldern mit geringer Ordnung. So konnte bereits die Lattice-Kryptographie im Embedded Bereich auf einem 8 Bit AVR Mikrokontroller erfolgreich implementiert werden [14, Abs. 3].

VI. Leitfrage: Warum wir schon jetzt neue Verschlüsselungsalgorithmen brauchen

Nachdem wir uns in dieser Arbeit damit beschäftigt haben, wie QC die aktuelle asymmetrische Kryptographie brechen kann und wie mit diesem Problem auf einer technischen Ebene umgegangen wird, gilt es noch eine weitere Frage zu beantworten:
Warum müssen wir uns schon jetzt mit dieser potentiellen Gefahr auseinandersetzen?
Ein Großteil der vorgestellen Konzepte und Algorithmen sind nur theoretischer Natur und aktuelle QC sind nicht in der Lage zum Beispiel RSA zu decodieren. Warum also hat das NIST seine Ausschreibung für einen quantensicheren Algorithmus gestartet? Warum sollte die Gefahr eines QCs auf die Kryptographie schon jetzt ernst genommen werden?
Zum einen gilt es die Frage zu beantworten, wann es wahrscheinlich ist, dass ein QC in der Lage sein wird eine mit RSA-2048 Bit verschlüsselte Nachricht zu entschlüsseln. Das theoretische Minimum an benötigten Qubits hierfür liegt laut Abschätzungen der Schweizer IT-Sicherheitsfirma *Kudelski Security* bei 6190 logischen Qubits [16]. Dies berücksichtigt jedoch nicht Faktoren wie Implementierungs-Details und Fehlerkorrektur. Einer anderen Schätzung zufolge, würden eher um die 10.000 reale Qubits benötigt werden [17]. Wie Lange wird es dauern bis ein QC mit einer solchen

Anzahl an Qubits Realität ist?
Schaut man auf die *Timeline* der Firma IBM (siehe Grafik 5) so sieht man, dass der aktuelle Entwicklungsstand ein QC mit 433 Qubits ist. Dies stellt schon ein enormes Wachstum zu den 27 Qubits in 2019 dar. Der Plan ist es bis 2025 auf ungefähr 4000 Qubits zu gelangen und nach 2026 10.000 und mehr Qubits anzustreben [18].
Diese Vorhersage deckt sich auch mit den Ansichten führender Experten in dem Gebiet der Quantenkryptographie. Die Ergebnisse einer Befragung des *Global Risk Institutes* an der 47 Forschende teilnahmen, sind in der Abbildung 4 veranschaulicht. Es wird für sehr wahrscheinlich gehalten, dass in den nächsten 15 bis 20 Jahren ein QC in der Lage sein wird RSA-2048 zu brechen. Mindestens aber in 30 Jahren soll dies mit nahezu vollständiger Sicherheit möglich sein [19].

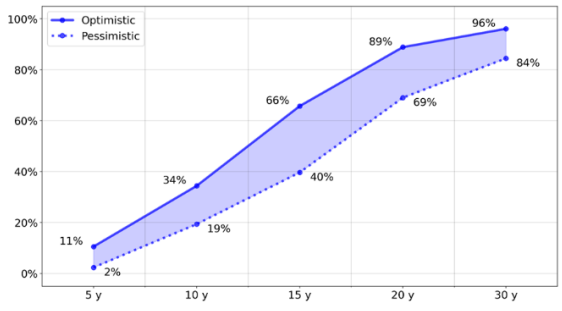

Abbildung 4. Wahrscheinlichkeit wann laut Experten des Quantencomputing RSA-2048 durch einen QC gebrochen sein kann [19]

Literaturverzeichnis

[1] a. cho adrian. „Ordinary computers can beat Google's quantum computer after all — Science — AAAS." (2. Aug. 2022), Adresse: https://www.science.org/content/article/ordinary-computers-can-beat-google-s-quantum-computer-after-all (besucht am 14. 04. 2023).

[2] A. Steane, „Quantum computing," *Reports on Progress in Physics*, Jg. 61, Nr. 2, S. 117, Feb. 1998, ISSN: 0034-4885. DOI: 10.1088/0034-4885/61/2/002. Adresse: https://dx.doi.org/10.1088/0034-4885/61/2/002 (besucht am 19. 03. 2023).

[3] E. Schrödinger, „Die gegenwärtige Situation in der Quantenmechanik," *Naturwissenschaften*, Jg. 23, Nr. 48, S. 807–812, 1. Nov. 1935, ISSN: 1432-1904. DOI: 10.1007/BF01491891. Adresse: https://doi.org/10.1007/BF01491891 (besucht am 10. 06. 2023).

[4] „cpb_27_9_090308_f8.jpg (JPEG Image, 462 × 527 pixels)." (), Adresse: https://cpb.iphy.ac.cn/article/2018/1953/cpb_27_9_090308/cpb_27_9_090308_f8.jpg (besucht am 10. 06. 2023).

[5] n. miller none, *Principles of Quantum Mechanics*. Adresse: https://www.astro.umd.edu/~miller/teaching/astr320/lecture21.pdf (besucht am 11. 06. 2023).

[6] D. P. DiVincenzo und IBM, „The physical implementation of quantum computation," *Fortschritte der Physik*, Jg. 48, Nr. 9, S. 771–783, Sep. 2000, ISSN: 00158208, 15213978. DOI: 10.1002/1521-3978(200009)48:9/11⟨771::AID-PROP771⟩3.0.CO;2-E. arXiv: quant-ph/0002077. Adresse: http://arxiv.org/abs/quant-ph/0002077 (besucht am 10. 06. 2023).

[7] D. J. Bernstein und T. Lange, „Post-quantum cryptography," *Nature*, Jg. 549, Nr. 7671, S. 188–194, Sep. 2017, Number: 7671 Publisher: Nature Publishing Group, ISSN: 1476-4687. DOI: 10.1038/nature23461. Adresse: https://www.nature.com/articles/nature23461 (besucht am 16. 04. 2023).

[8] P. Shor, „Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, Nov. 1994, S. 124–134. DOI: 10.1109/SFCS.1994.365700.

[9] A. Kak, „Lecture 12: Public-key cryptography and the RSA algorithm lecture notes on "computer and network security"",

[10] C. Gidney und M. Ekerå, „How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits," *Quantum*, Jg. 5, S. 433, 15. Apr. 2021, ISSN: 2521-327X. DOI: 10.22331/q-2021-04-15-433. arXiv: 1905.09749[quant-ph]. Adresse: http://arxiv.org/abs/1905.09749 (besucht am 16. 04. 2023).

[11] M. Amico, Z. H. Saleem und M. Kumph, „An Experimental Study of Shor's Factoring Algorithm on IBM Q," *Physical Review A*, Jg. 100, Nr. 1, S. 012 305, 8. Juli 2019, ISSN: 2469-9926, 2469-9934. DOI: 10.1103/PhysRevA.100.012305. arXiv: 1903.00768[quant-ph]. Adresse: http://arxiv.org/abs/1903.00768 (besucht am 16. 04. 2023).

[12] D. Moody, „Status report on the third round of the NIST post-quantum cryptography standardization process," National Institute of Standards und Technology, Gaithersburg, MD, NIST IR 8413-upd1, 2022, NIST IR 8413–upd1. DOI: 10.6028/NIST.IR.8413-upd1. Adresse: https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8413-upd1.pdf (besucht am 16. 04. 2023).

[13] D. Micciancio und O. Regev, „Lattice-based cryptography,"

[14] A. Wang, D. Xiao und Y. Yu, „Lattice-based cryptosystems in standardisation processes: A survey," *IET Information Security*, Jg. 17, Nr. 2, S. 227–243, 2023, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1049/ise2.12101, ISSN: 1751-8717. DOI: 10.1049/ise2.12101. Adresse: https://onlinelibrary.wiley.com/doi/abs/10.1049/ise2.12101 (besucht am 10. 06. 2023).

[15] R. Xu, C. Cheng, Y. Qin und T. Jiang, *Lighting the Way to a Smart World: Lattice-Based Cryptography for Internet of Things*, 13. Mai 2018. arXiv: 1805.04880[cs]. Adresse: http://arxiv.org/abs/1805.04880 (besucht am 10. 06. 2023).

[16] T. Gagliardoni. „Quantum attack resource estimate: Using shor's algorithm to break RSA vs DH/DSA VS ECC," Kudelski Security Research. (24. Aug. 2021), Adresse: https://research.kudelskisecurity.com/2021/08/24/quantum-attack-resource-estimate-using-shors-algorithm-to-break-rsa-vs-dh-dsa-vs-ecc/ (besucht am 11. 06. 2023).

[17] L. Ziegler, „Online security, cryptography, and quantum computing," *Forum Lectures*, 29. Jan. 2015. Adresse: https://digitalcommons.csbsju.edu/forum_lectures/119.

[18] „IBM Quantum Computing — Roadmap." (1. Okt. 2015), Adresse: https://www.ibm.com/quantum/www.ibm.com/quantum/roadmap (besucht am 11. 06. 2023).

[19] „2021 quantum threat timeline report: Global risk institute," Global Risk Institute. (), Adresse: https://globalriskinstitute.org/publication/2021-quantum-threat-timeline-report-global-risk-institute-global-risk-institute/ (besucht am 16. 04. 2023).

VII. Anhang

A. *IBM Quantencomputing roadmap*

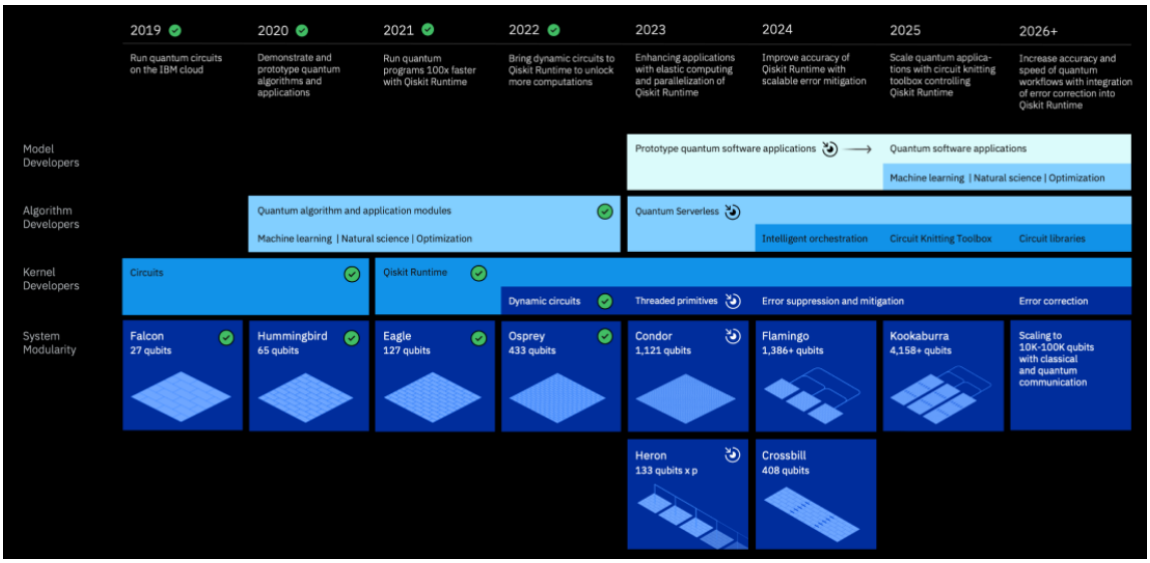

Abbildung 5. Rodmap der Firma IBM für ihren QC [18]