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
ggT größter gemeinsamer Teiler
QFT Quanten-Fourier-Transformation
NIST National Institute of Standards and Technology

I. Einleitung

Das Thema der *Quanten Supremacy*, der Zeitpunkt zu dem ein Quantencomputer 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 Quantencomputer zu besitzen, was jedoch in diversen Publikationen bezweifelt wurde **cho˙ordinary˙2022**. Wenig Zweifel besteht jedoch, dass ein solcher Computer in absehbarer Zeit einsatzbereit sein wird.
Im Folgenden wird beschieden wie ein Quantencomputer funktioniert, mit welchen Algorithmen, die in realer Zeit laufen, Quantencomputer die aktuellen, kryptographischen Verfahren brechen können werden. Außerdem werden wir aktuelle, neu Entwickelte quantensichere Algorithmen betrachten und wie sie sie Gefahr durch Quantencomputer mittigern werden.

II. Einführung in das Quanten Computing

Quanten Computing ist eine relativ junge Disziplin der Physik und Informatik. Während sich die beiden Bereiche unabhängig von einader Anfang des 20. Jahrhundert entwickelt hatten, wurden die ersten Versuche und theoretischen Überlegungen, Methoden der Informatik mithilfe von Quantenobjekten umzusetzen um 1980 gestartet. Obwohl

das Quantencomputer erst seit kurzer Zeit existiert, hat es 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 **steane˙quantum˙1998** 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 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 Quantencomputer repräsentiert Informationen in den Eigenschaften von Quanten-Objekten, beispielsweise dem Spin von Elektronen.
Um die Funktion eines Quantencomputer zu erläutern muss erst ein kurzer Blick auf einige Quantenphysikalische Phänomene geworfen werden.

1) Superpositionen: Eine Superposition ist das Fundamentale quantenphysische Phänomen das einem Quantencomputer zugrunde Liegt. Es wird meistens mithilfe des Gedankenexperiments von **schrodinger˙gegenwartige˙1935** *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. Von 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 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 $1/4$ als Spin down. Mathematisch kann eine Superposition ψ also 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 **??**), es ist zu beachten, dass der Vector z nicht den Spin, sondern die Linearkombination darstellt.

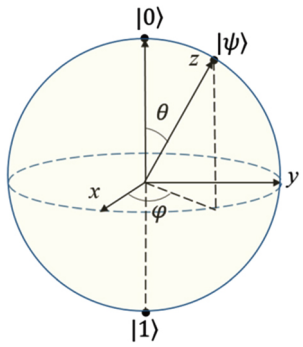

Abbildung 1. Spin eines Elektrons in Superposition **noauthor˙cpb˙27˙9˙090308˙f8jpg˙nodate**

2) Quantenverschränkung:

B. Implementierung eines Quantencomputers

III. RSA

IV. Shors-Algorithmus

Die Faktorisierung großer Zahlen **bernsteinPostquantumCryptography2017** 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 **shorAlgorithmsQuantumComputation1994** gilt als einer der bedeutendsten Quantenalgorithmen, der die Faktorisierung großer Zahlen in Polynomialzeit ermöglicht. Auch wenn dieser Algorithmus einen Quantencomputer 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!** (**qc!**) geforscht, die Anzahl stabiler Qubits in **qc!** steigen. Wenn der Quantencomputer 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 **kakLecture12PublicKeygidneyHowFactor20482021** werden wir an einem vereinfachten Beispiel näher aufzeigen, wie Shors Algorithmus eine effiziente Lösung für die Faktorisierung großer Zahlen auf einem Quantencomputer 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 **??** $r = 10$

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 **ggT!** (**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 **ggT!** (**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 **??** 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 **amicoExperimentalStudyShor2019** werden zwei Quantenregister 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 $\frac{q^r}{N}$ 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. Sobald man eine mögliche Lösung misst, z.B. aus **??** $R = 15.$ 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 **qft!** (**qft!**) angewendet werden, welche es erlaubt die Frequenz einer periodischen Superposition zu ermitteln. **amicoExperimentalStudyShor2019** 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!** (**nist!**) einen weltweit einheitlichen Standardisierungsprozess für Post-Quantum-Kryptografie gestartet **moodyStatusReportThird2022**. 2022 wurden mehrere Algorithmen Ausgewählt. Die meisten dieser Algorithmen basieren auf mathematischen Gitter- (*engl. Lattice*) Problemen.

Lattice-Kryptographie **micciancioLatticebasedCryptography** 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. **wangLatticebasedCryptosystemsStandardisation2023**

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.

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 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.

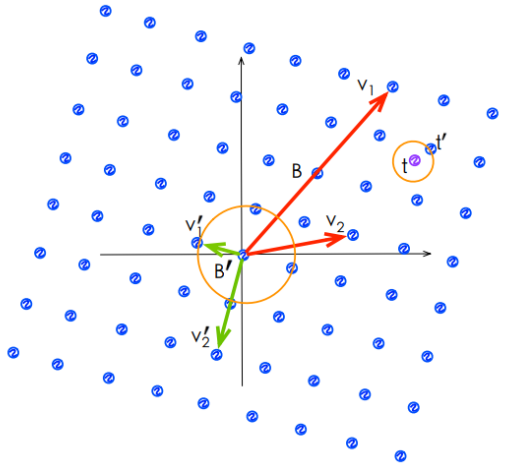

Abbildung 3. Darstellung eines 2D Gitters
xuLightingWaySmart2018

Hallo

VI. Leitfrage: Warum wir schon jetzt neue Verschlüsselungsalgorithmen brauchen

Nachdem wir uns in dieser Arbeit hauptsächlich mit dem WIE Quantencomputer die aktuelle asymmetrische Kryptografie brechen können und wie mit diesem Problem auf einer technischen Ebene umgegangen werden kann, 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 Quantencomputer sind nicht in der Lage zum Beispiel RSA zu decodieren. Warum also hat das NIST seine Ausschreibung für einen Quantensichern Algorithmus gestartet? Warum sollte die Gefahr eines Quantencomputers auf die Kryptographie schon jetzt ernst genommen werden?

Zum einen gilt es die Frage zu beantworten wann es wahrscheinlich es ist, dass ein Quantencomputer in der Lage ist eine mit RSA-2048 bit verschlüsselte Nachricht zu entschlüsseln. Das theoretische minimum an benötigte Qbits hierfür liegt laut Abschätzungen der Schweizer IT-Sicherheitsfirma *Kudelski Security* bei 6190 logischen Qbits **gagliardoni˙quantum˙2021**. Dies berücksichtigt jedoch Faktoren wie Implementierungs-Details und Fehlerkorrektur nicht. Einer anderen Schätzung zufolge würden eher um die 10.000 reale Qbits benötigt**ziegler˙online˙2015**. Wie Lange wird es dauern bis ein Quantencomputer mit einer solchen Anzahl an Qbits Realität ist?

Schaut man auf die *Timeline* der Firma IBM (siehe Grafik **??**) so sieht man, dass der aktuelle entwicklungsstand ein Quantencomputer mit 433 Qbits ist. Dies stellt schon ein enormes Wachstum zu den 27 Qbits in 2019 dar. Der Plan ist es bis 2025 auf ungefähr 4000 Qbits zu gelangen und nach 2026 10.000 und mehr Qbits anzupeilen **noauthor˙ibm˙2015**.

Diese Vorhersage deckt sich auch mit den Ansichten führende Experten in dem Gebiet der Quantenkryptografie. Bei einer Befragung des *Global Risk Intites* an der 47 Forschende teilnahmen ergab sich die folgenden Ergebnisse. Es wird für sehr wahrscheinlich gehalten, dass in den nächsten 15 bis 20 Jahren ein Quantencomputer in der Lage sein wird RSA-2048 zu brechen mindestens aber in 30 Jahren soll dies mit nahezu vollständiger Sicherheit möglich sein **noauthor˙2021˙nodate**.

Wenn

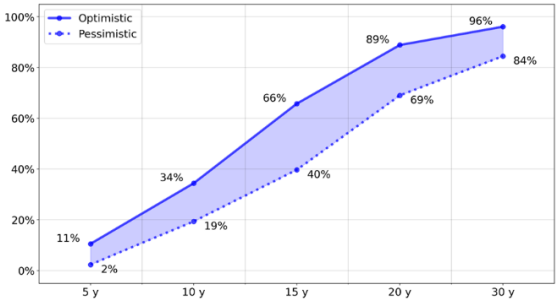

Abbildung 4. Wahrscheinlichkeit wann laut Experten des Quantencomputing RSA-2048 durch einen Quantencomputer gebrochen sein kann **noauthor˙2021˙nodate**

VII. Anhang

A. *IBM Quantencomputing roadmap*

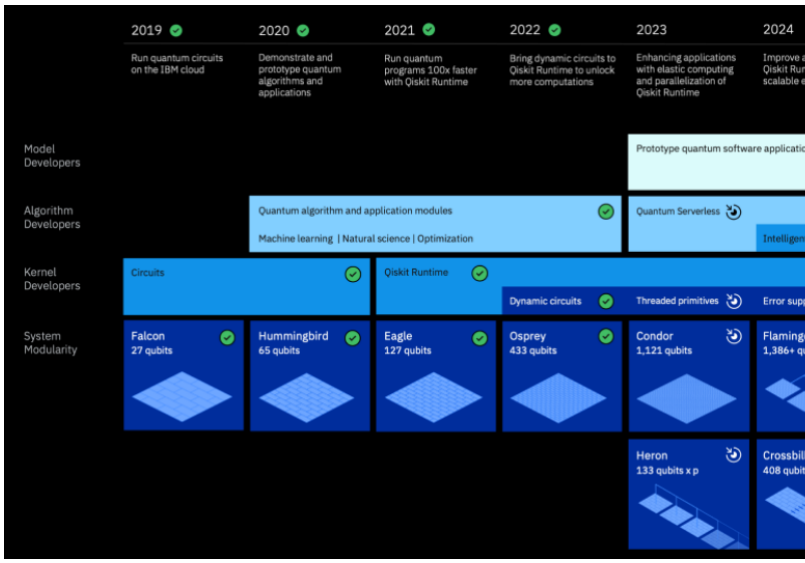

Abbildung 5. Rodmap der Firma IBM für ihren Quantencomputer **noauthor˙ibm˙2015**

Literaturverzeichnis