Post-Quantum-Kryptografie anschaulich erklärt

Die Schnecke im Salatfeld

Klaus Schmeh

Für Laien ist die komplexe Mathematik hinter Post-Quantum-Kryptografie kaum nachvollziehbar. Einfache Analogien können jedoch den Zugang enorm erleichtern.

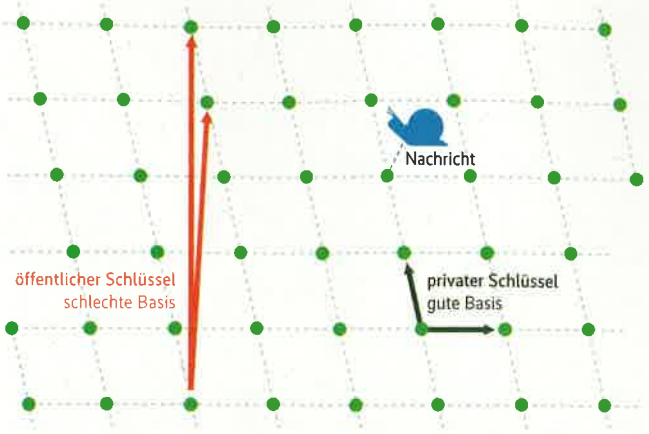
Trotz aller Fortschritte in der Informationstechnik seit Pionieren wie Konrad Zuse oder Alan Turing hat sich eines nie geändert: Computer arbeiten mit Nullen und Einsen – und damit mit Speicherbausteinen, die stets den einen oder anderen Wert annehmen. Das muss aber nicht unbedingt so bleiben, denn es gibt eine Technik, die dieses Konzept verändern könnte. Sie arbeitet auch mit binären Speicherbausteinen, die jedoch im Gegensatz zu herkömmlichen Bits die Werte null und eins gleichzeitig annehmen können. Die Rede ist von sogenannten Quantencomputern.

iX-TRACT

- Asymmetrische und gegenüber Quantencomputern nicht anfällige Kryptomethoden (Post-Quanten-Verfahren) sind mathematisch anspruchsvoll.
- Mithilfe einiger Analogien und Vereinfachungen können auch Nichtmathematiker Post-Quanten-Verfahren verstehen.
- Derzeit haben gitterbasierte Verfahren sowie die multivariate Kryptografie die größten Einsatzchancen.

Dabei handelt es sich um Rechner, die nach den Prinzipien der Quantenphysik arbeiten. Eine davon wird oft mit Schrödingers Katze veranschaulicht, die gleichzeitig tot und lebendig ist – so lange, bis man nachschaut. Ähnlich verhält es sich mit einem Quanten-Speicherbaustein. Ein solcher hat gleichzeitig den Wert null und eins – so lange, bis man ihn ausliest. Durch diese spezielle Eigenschaft können Quantencomputer mühelos Aufgaben lösen, für die herkömmliche Rechner astronomische Zeiträume benötigen würden. Insbesondere für das Knacken asymmetrischer Kryptoverfahren wären Quantencomputer äußerst nützlich. Praktisch alle eingesetzten Algorithmen dieser Art – inklusive RSA, Diffie-Hellman und Verfahren auf Basis elliptischer Kurven – müssten ausrangiert werden, sollte es eines Tages brauchbare Quantencomputer geben.

Wenn man in die Zukunft der IT schaut, darf man neben selbstfahrenden Autos, Haushaltsrobotern und dem Internet der Din-



Der private Schlüssel ist eine gute Basis, der öffentliche eine schlechte. Bob codiert eine Nachricht mit der Schnecke, die er in die Nähe eines Salatkopfs setzt. Alice kann mit ihrem privaten Schlüssel den nächsten Salatkopf bestimmen und somit die Nachricht entschlüsseln (Abb. 2).

ge die Quantenapokalypse nicht vergessen, die eines Tages eintreten könnte. Zahlreiche Kryptologen haben darauf längst reagiert und beschäftigen sich mit asymmetrischen Verfahren, denen Quantencomputer nichts anhaben können. Bei diesen sogenannten Post-Quanten-Verfahren handelt es sich um Methoden, die bisher in der Praxis kaum eine Rolle spielen und noch nicht ausreichend erforscht sind. Über einen derzeit von der US-Behörde NIST veranstalteten, mehrjährigen Wettbewerb, der Klarheit bringen soll, hat *iX* berichtet [1]. Von ursprünglich 69 Verfahren haben es 26 in die derzeit laufende zweite Runde geschafft.

Es wird immer komplizierter

Fast alle diese Verfahren lassen sich einem von sechs mathematischen Prinzipien zuordnen. Wer sich mühevoll die Grundlagen von RSA, Diffie-Hellman oder elliptischen Kurven angeeignet hat, sieht sich nun mit sechs weiteren Algorithmen-Familien konfrontiert, die mathematisch meist noch komplexer sind. Für Nichtmathematiker geeignete Literatur zum Thema gibt es kaum. Dabei kann man zumindest die Grundlagen der gängigen Post-Quantum-Verfahren durchaus anschaulich erklären.

Die wohl bedeutendste Familie ist die der gitterbasierten Verfahren. Nicht nur weil der englische Begriff „Lattice“ (Gitter) dem Wort „Lettuce“ (Salat) ähnelt, kann man sich ein Gitter als Salatfeld vorstellen (Abbildung 1), in dem die Salatköpfe (mathematisch als Gitterpunkte bezeichnet) auf parallelen Linien in gleichen Abständen angeordnet sind. Ein zweidimensionales Salatfeld lässt sich mithilfe zweier Vektoren definieren, die Basis heißen. Man unterscheidet zwischen einer guten Basis (die Vektoren stehen annähernd senkrecht zueinander) und einer schlechten (die Vektoren verlaufen annähernd parallel).

Die Frage ist nun: Wie findet eine Schnecke, die sich irgendwo im Salatfeld befindet, den nächsten Salatkopf? Im bisher betrachteten zweidimensionalen Fall ist es recht einfach – einer der vier benachbarten Salatköpfe ist eben der nächste. Deutlich schwieriger gestaltet sich die Sache, wenn sich die Schnecke im (beispielsweise) 250-dimensionalen Raum bewegt. Nun hat sie nicht mehr nur vier Salatköpfe zur Auswahl, sondern 2^{250} (eine 75-stellige Zahl), was das Durchprobieren aller Entfernungen unmöglich macht. Zwar gibt es Algorithmen, die den nächsten Salatkopf ermitteln, doch für sie gilt: Nur wenn eine gute Basis bekannt ist, wird die Pflanze schnell gefunden. Steht dagegen nur eine schlechte Basis zur Verfügung, dauert die Suche extrem lange.

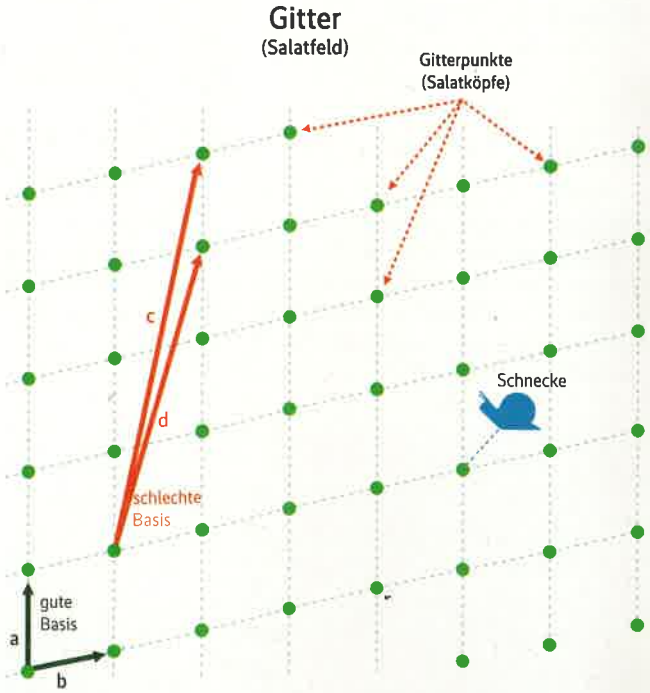
Aus der vieldimensionalen Salatkopfsuche lässt sich ein asymmetrisches Verschlüsselungsverfahren konstruieren. Es sieht zunächst vor, dass sich der potenzielle Empfänger einer Nachricht (in der Kryptografie meist Alice genannt) ausgehend von einer guten Basis ein Salatfeld (sprich: Gitter) im 250-dimensionalen Raum generiert. Die gute Basis ist Alices privater Schlüssel. Ihr öffentlicher Schlüssel ist eine schlechte Basis desselben Gitters (es ist recht einfach, aus einer guten Basis eine schlechte abzuleiten, umgekehrt gilt das allerdings nicht).

Weichtiere als Pfadfinder im Hyperraum

Will Bob (der Absender) eine Nachricht an Alice verschicken, setzt er eine Schnecke in die Nähe eines Salatkopfs. Er wählt also einen Punkt, der nahe an einem Gitterpunkt liegt. Die Differenz zwischen Schnecke und Salatkopf ist die Nachricht (da sich Bob im 250-dimensionalen Raum befindet, hat er 250 Komponenten zur Verfügung, um die Nachricht zu codieren). Weil Alice eine gute Basis besitzt, kann sie den nächsten Salatkopf schnell finden und die Nachricht damit entschlüsseln. Ein Angreifer hat dagegen nur eine schlechte Basis zur Verfügung und benötigt somit sehr lange für diese Aufgabe.

Ein Verfahren, das ziemlich genau nach dem beschriebenen Schema arbeitet, ist das nach Goldreich, Goldwasser und Halevi benannte GGH-Verfahren (Abbildung 2). Allerdings hat sich GGH als unsicher erwiesen und kommt damit als Post-Quantum-Methode nicht infrage.

Es gibt jedoch Gitterverfahren, die bisher noch als sicher gelten. Bei den meisten davon taucht der Begriff Gitter in der Beschreibung zwar nicht auf, doch das Problem, auf dem sie beruhen, ist gleichwertig mit einer Salatkopfsuche. Zu diesen Me-



In einem zweidimensionalen Gitter (oder Salatfeld) kann eine Schnecke den nächsten Salatkopf schnell finden. Im 250-dimensionalen Raum funktioniert es nur mit einer guten Basis (Abb. 1).

thoden gehören die LWE-Verfahren (Learning with Errors). Um sie zu verstehen, muss man sich zunächst mit linearen Gleichungssystemen beschäftigen, die einige Leser vielleicht noch aus der Schule kennen. Speziell geht es hier um Systeme, in denen modulo gerechnet wird (es gibt nur ganze Zahlen, die alle zwischen 0 und einer bestimmten Zahl, dem Modulus, liegen). Das folgende Gleichungssystem arbeitet mit dem Modulus 701:

```
604x + 123y + 251z = 562 (mod 701)
322x + 88y + 478z = 634 (mod 701)
602x + 67y + 176z = 698 (mod 701)
420x + 223y + 635z = 262 (mod 701)
189x + 618y + 77z = 252 (mod 701)
```

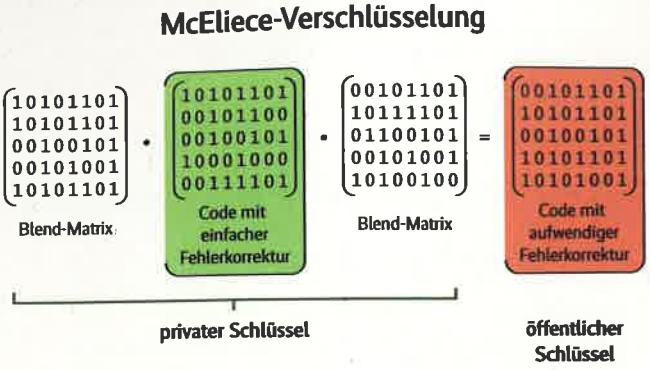
Da hier fünf Gleichungen, aber nur drei Variablen existieren, wäre zu erwarten, dass es keine Lösung gibt. Allerdings sind die Gleichungen so ausgewählt, dass eine Lösung doch existiert. Sie lautet: $x=33$, $y=402$, $z=128$. Der nächste Schritt führt auf der rechten Seite der Gleichung ein paar Fehler ein (aus 562 wird beispielsweise 563, aus 634 wird 633):

```
604x + 123y + 251z = 563 (mod 701)
322x + 88y + 478z = 633 (mod 701)
602x + 67y + 176z = 698 (mod 701)
420x + 223y + 635z = 264 (mod 701)
189x + 618y + 77z = 253 (mod 701)
```

Die Fehler sind stets klein im Vergleich zum Modulus (der Fehler kann auch null betragen). Wer die Lösung kennt, kann die fünf Fehler auf der rechten Seite schnell ermitteln. Die Frage ist nun: Ist es möglich, die Fehler auf der rechten Seite zu finden, ohne die Lösung zu kennen? Die Antwort lautet: Ja, das ist möglich, aber ziemlich schwierig. Wenn man es mit Hunderten von Variablen und Gleichungen zu tun und der Modulus einige Hundert Stellen hat, kann man die Fehler nicht mit realistischem Aufwand finden.

Durch fehlerhafte Gleichungen zum Ziel

Dieses Prinzip lässt sich für ein asymmetrisches Verschlüsselungsverfahren nutzen. Dabei dient das fehlerhafte Gleichungssystem als Alices öffentlicher Schlüssel. Den privaten Schlüssel bilden die Lösungen. Wenn Bob etwas für Alice verschlüsseln will, wählt er zufällig (beispielsweise durch Münzwurf) etwa die Hälfte der Gleichungen aus. In unserem Fall seien es die zweite, die dritte und die fünfte. Diese addiert er (durch eine Addition ändert sich die Lösung nicht):



Der Empfänger nutzt bei der McEliece-Verschlüsselung einen Code, der eine einfache Fehlerkorrektur ermöglicht, als privaten Schlüssel. Mithilfe zweier (geheimer) Matrizen wandelt er den Code in einen gewöhnlichen fehlerkorrigierenden um (Abb. 3).

```
322x + 88y + 478z = 633 (mod 701)
602x + 67y + 176z = 698 (mod 701)
189x + 618y + 77z = 253 (mod 701)
-------------------------------------------
412x + 72y + 622z = 182 (mod 701)
```

Nun kann Bob eine Null oder eine Eins codieren. Im ersten Fall zählt er zum Ergebnis der resultierenden Gleichung (182) eine kleine Zahl (beispielsweise 2) dazu und schickt dann $412x + 72y + 622z = 184 \pmod{701}$ an Alice. Im zweiten Fall addiert er etwa die Hälfte des Modulus (etwa die Zahl 350) zum Ergebnis und verschickt entsprechend $412x + 72y + 622z = 532 \pmod{701}$. Alice kennt die Lösung und kann damit leicht feststellen, ob das Ergebnis (bis auf einen kleinen Fehler) stimmt oder ob es grob danebenliegt. Im ersten Fall weiß sie, dass Bob eine Null übermittelt hat, im zweiten eine Eins. Ein Angreifer kann dagegen nicht wissen, ob die Gleichung nahezu korrekt oder kom-

FÜR PROFESSIONELLE INFORMATIONSTECHNIK

iX Special 2019 – IT heute

Postfach 61 04 07, 30604 Hannover; Karl-Wiechert-Allee 10, 30625 Hannover

Redaktion: Telefon: 0511 5352-387, Fax: 0511 5352-361, E-Mail: post@ix.de

Abonnements: Telefon: 0541 80009-120, Fax: 0541 80009-122, E-Mail: leserservice@heise.de

Herausgeber: Christian Heise, Ansgar Heise

Redaktion: Chefredakteure: Jürgen Seeger (js@ix.de) -386, Dr. Oliver Diedrich (odi@ix.de) -616

Ltd. Redakt.: Kersten Auel (ka@ix.de) -367, Markus Feilner (mfe@ix.de) -388, Alexander Neumann (ane@ix.de) -813, Bert Ungerer (un@ix.de) -368

Nicole Bechtel (nb@ix.de) -378, Björn Bohn (bbo@ix.de) -373, Jürgen Diercks (jd@ix.de) -379, Moritz Förster (fo@ix.de) -374, Alexandra Kleijn (akl@ix.de) -787, Rainald Menge-Sonnentag (rme@ix.de), Susanne Nolte (sun@ix.de) -689, Matthias Parbel (map@ix.de) -321, André von Raison (avr@ix.de) -377, Ute Roos (ur@ix.de) -535, Carina Schipper (csc@ix.de) -384

Redaktionsassistenz: Carmen Lehmann (cle@ix.de) -387, Michael Mentzel (mm@ix.de) -153

Layout und Satz: Madlen Grunert, Lisa Hemmerling, Kirsten Last, Steffi Martens, Marei Stade, Matthias Timm, Mai Wolters, Heise Medienwerk, Rostock

Chefin vom Dienst: Barbara Gückel

Korrektorat: Lydia M. Behnke, Barbara Gückel; Marei Stade, Ricardo Ulbricht, Heise Medienwerk, Rostock

Hergestellt und produziert mit Xpublisher: www.xpublisher.com

Xpublisher-Technik: Melanie Becker, Joana Hollasch

Fotografie: Martin Klauss Fotografie, Despetal/Barfelde

Titel: Idee: *iX*; Titel- und Aufmachergestaltung: Matthias Timm

Abbildungen: Titelmotiv: iStock.com/lena_serditova; Bildmotive des Preisrätsels S. 152/153: Shutterstock, sofern nicht anders angegeben

Verlag und Anzeigenverwaltung:

Heise Medien GmbH & Co. KG, Postfach 61 04 07, 30604 Hannover; Karl-Wiechert-Allee 10, 30625 Hannover; Telefon: 0511 5352-395, Fax: 0511 5352-129

Geschäftsführer: Ansgar Heise, Dr. Alfons Schräder

Mitglieder der Geschäftsleitung: Beate Gerold, Jörg Mühle

Verlagsleiter: Dr. Alfons Schräder

Anzeigenleitung: Michael Hanke -167, E-Mail: michael.hanke@heise.de, www.heise.de/mediadaten/ix

Leiter Vertrieb und Marketing: André Lux -299

Werbeleitung: Julia Conrades -156

Druck: Dierichs Druck + Media GmbH & Co. KG, Frankfurter Straße 168, 34121 Kassel

Sonderdruck-Service: Julia Conrades -156

Verantwortlich: Textteil: Dr. Oliver Diedrich; Anzeigenteil: Michael Hanke

***iX* Special 2019 – IT heute:** Einzelpreis: 12,90 €, Österreich 14,20 €, Schweiz 25,80 CHF, Luxemburg 14,80 €

Eine Haftung für die Richtigkeit der Veröffentlichungen kann trotz sorgfältiger Prüfung durch die Redaktion vom Herausgeber nicht übernommen werden. Die gewerbliche Nutzung abgedruckter Programme ist nur mit schriftlicher Genehmigung des Herausgebers zulässig.

Honorierte Arbeiten gehen in das Verfügungsrecht des Verlages über, Nachdruck nur mit Genehmigung des Verlages. Mit Übergabe der Manuskripte und Bilder an die Redaktion erteilt der Verfasser dem Verlag das Exklusivrecht zur Veröffentlichung. Für unverlangt eingesandte Manuskripte kann keine Haftung übernommen werden. Sämtliche Veröffentlichungen in *iX* erfolgen ohne Berücksichtigung eines eventuellen Patentschutzes. Warennamen werden ohne Gewährleistung einer freien Verwendung benutzt.

Printed in Germany

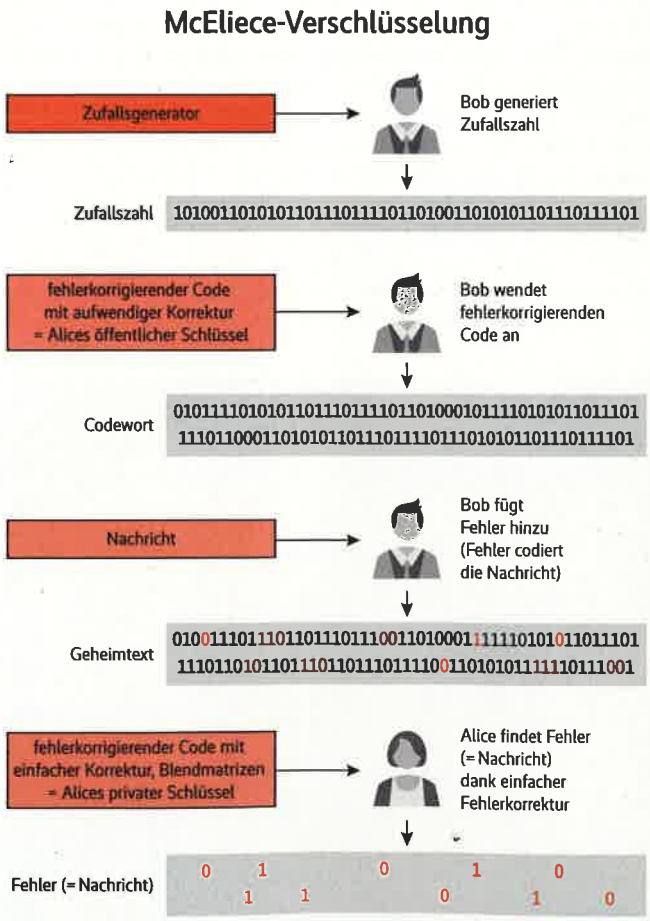
© Copyright by Heise Medien GmbH & Co. KG

plett falsch ist – da er die Lösung nicht mit realistischem Aufwand berechnen kann.

Viel Aufwand, wenig Ertrag

Die beschriebene Methode wird als Regev-Verschlüsselungsverfahren bezeichnet. Man kann zeigen, dass das Finden der richtigen Lösung dem Finden des nächsten Salatkopfs entspricht, weshalb Regev zu den gitterbasierten Verfahren gehört. Das gezeigte Prozedere ermöglicht es allerdings nur, ein einziges Bit zu übertragen – das ist sehr wenig im Vergleich zum getriebenen Aufwand. Es gibt jedoch effektivere LWE-Verfahren als dieses. Es ist außerdem möglich, statt mit ganzen Zahlen mit Elementen eines Rings zu rechnen (man spricht dann von Ring-LWE oder RLWE). Das bekannteste RLWE-Verfahren ist New Hope, das Google im Chrome-Browser implementiert hat.

Neben gitterbasierten Verfahren gehören auch codebasierte Algorithmen zur Post-Quantum-Kryptografie. Angenommen, eine Raumsonde ist auf dem Weg von der Erde zum Mars und sendet regelmäßig digitale Daten (also Folgen von Nullen und Einsen) zur Erde zurück. Da auf einer so langen Strecke Übertragungsfehler auftreten können, bietet es sich an, eine Prüfsumme zu verwenden, die beispielsweise aus Paritätsbits besteht: auf sieben Bit Nutzdaten folgt ein achtes, das so gesetzt wird, dass in jedem Acht-Bit-Block eine gerade Anzahl von Einsen steht.



Bei der McEliece-Verschlüsselung baut Bob in ein Codewort Fehler ein, die die zu verschlüsselnde Nachricht repräsentieren. Da Alice mit ihrem privaten Schlüssel den Fehler schnell finden kann, ist das Entschlüsseln für sie einfach (Abb. 4).

Aus 0110101 wird also 01101010. Paritätsbits sind ein Beispiel für einen fehlerdetektierenden Code.

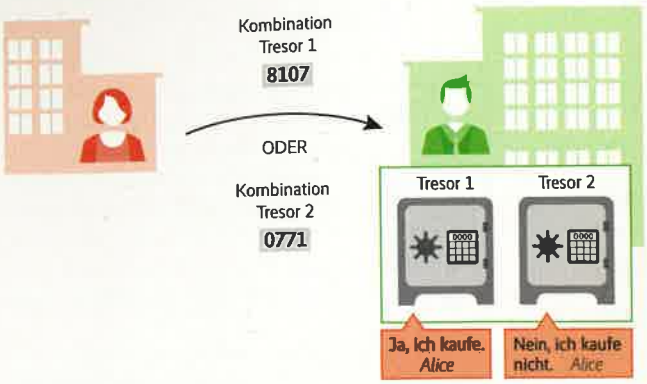
Alternativ ist es möglich, jeden Nutzdatenblock dreifach zu übertragen. Aus 01101010 wird nun 01101010 01101010 01101010. Der Vorteil dabei: Ein Fehler fällt nicht nur auf, sondern lässt sich auch gleich beheben. Das Dreifach-Senden ist damit ein Beispiel für einen fehlerkorrigierenden Code. Pro Codewort (im Beispiel hat dieses 24 Bit) lässt sich allerdings nur ein Fehler sicher korrigieren, während zwei Fehler schon zu einer falschen Decodierung führen können.

Jedes Bit dreimal zu übertragen, ist zudem recht aufwendig. Informatiker haben daher bessere fehlerkorrigierende Codes entwickelt. Einige davon sehen vor, dass beispielsweise ein 5-Bit-Nutzdatenblock durch eine Multiplikation mit einer 5×8-Matrix auf einen 8-Bit-Block ausgedehnt wird. Der ursprüngliche Block verlängert sich also um drei Bits. Es gibt Codes dieser Art (man spricht von einem linearen Code), die ein falsches Bit sicher korrigieren können.

Fehlerkorrekturen und ihre Tücken

Im Allgemeinen gilt: Ein linearer Code, der ein Nutzdatenwort um n Bits verlängert, kann maximal n/2 falsche Bits sicher korrigieren. Sind die Nutzdatenblöcke beispielsweise 60 Bit lang und werden per geeigneter Matrixmultiplikation jeweils auf einen 100-Bit-Block abgebildet, dann lassen sich (bei geeigneter Wahl der Matrix) bis zu 20 falsche Bits pro Block zweifelsfrei verbessern. Sind dagegen mehr als 20 falsche Bits enthalten, kann es passieren, dass man einige davon nicht entdeckt oder nicht vorhandene Fehler korrigiert.

Das Finden und Korrigieren von falschen Bits ist bei längeren Nutzdatenblöcken und größeren Werten von n recht aufwendig. Geht es beispielsweise um Nutzdatenblöcke der Länge 5000 Bit und Codewörter der Länge 7000 Bit (man kann damit maximal 1000 falsche Bits sicher korrigieren), geht selbst der stärkste Rechner in die Knie. Zum Glück gibt es jedoch bestimmte lineare Codes, für die deutlich schnellere Fehlerkorrektur-Algorithmen bekannt sind. Dazu gehören die sogenannten Goppa-Codes.



Alice hinterlässt dem Makler zwei Tresore – einen mit einer Zusage, einen weiteren mit einer Absage (Abb. 5).

Wie jeder lineare Code lässt sich auch ein Goppa-Code über die Matrix definieren, mit der die Nutzdaten multipliziert werden. Dabei ist es möglich, eine Goppa-Code-Matrix durch eine Multiplikation mit zwei anderen Matrizen (Blend-Matrizen) in einen gewöhnlichen Code umzuwandeln. Hält man die beiden Blend-Matrizen geheim, lässt sich der Prozess nicht ohne Weiteres umkehren.

Auf Basis dieser Überlegungen ist es möglich, das (codebasierte) McEliece-Verschlüsselungsverfahren zu definieren. Es sieht vor, dass Alice eine Goppa-Code-Matrix (eine übliche Größe wäre 5500 × 7000) generiert und diese durch eine Multiplikation mit zwei Blend-Matrizen in einen gewöhnlichen Code umwandelt (Abbildung 3). Die umgewandelte Matrix ist Alices öffentlicher Schlüssel. Will Bob etwas verschlüsseln, dann generiert er einen Nutzdatenblock (5000 Bit), wandelt ihn mit Alices öffentlichem Schlüssel in ein Codewort (7000 Bit) um und fügt diesem 128 Fehler (also falsche Bits) hinzu (Abbildung 4). Die falschen Bits codieren den Klartext, der in diesem Fall 128 Bit lang ist. Da Alice den zugehörigen Goppa-Code kennt, kann sie das fehlerhafte Codewort schnell decodieren und die Fehler (sprich: den Klartext) ermitteln. Ein Angreifer muss dagegen den langen Weg gehen und mit der gewöhnlichen Codematrix die Fehler suchen. Bei einem Code in dieser Größe ist dies nahezu aussichtslos.


digitronic®

Neu und für Großes geschaffen

Mit unserer Compliance-Lösung

HiCrypt™ Enterprise Services

kann das Audit kommen. Die neue Generation unserer Verschlüsselungslösung schafft Vertraulichkeit auf höchstem Niveau:

- zentrale Verwaltung aller verschlüsselten Shares
- gemanagte Vertraulichkeit für große IT-Infrastrukturen

QR-Code scannen und mehr erfahren:

www.digitronic.net I vertrieb@digitronic.net I +49 (0) 371 81539-0

digitronic® gratuliert zu **30 Jahren iX**!

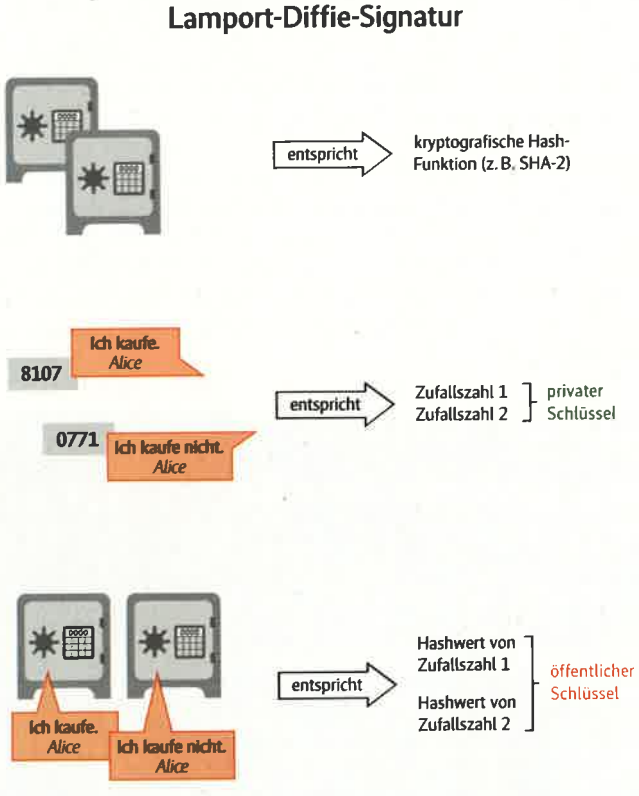
Kaum Relevanz für die Praxis

Das McEliece-Verfahren ist gut untersucht und gilt als sicher. Allerdings sind die Schlüssel extrem lang. Während man beispielsweise bei RSA mit 4 KByte auskommt, erreichen öffentliche McEliece-Schlüssel Längen in der Größenordnung von einem MByte. Die zahlreichen anderen codebasierten Verfahren haben ähnliche Nachteile und haben sich daher nie durchgesetzt. Obwohl sie als quantensicher gelten, werden sie sich wohl auch in Zukunft schwertun, sofern sich nicht andere Methoden als unsicher erweisen.

Hashbasierte Algorithmen bilden eine weitere Familie von Verfahren. Angenommen, Alice wolle ein Ferienhaus auf einer Südseeinsel kaufen. Ein Makler bietet ihr eins an, und so macht sich Alice auf die Reise. Nachdem sie das Haus ausgiebig begutachtet hat, teilt sie dem Makler per Telefon mit, ob sie es kaufen will oder nicht.

Dummerweise haben nun beide Parteien die Möglichkeit zu schummeln. Wenn Alice beispielsweise eine Zusage übermittelt, kann sie dies später bestreiten und behaupten, es sei eine Absage gewesen. Der Makler kann Alices Mitteilung ebenfalls ins Gegenteil verkehren. Um solche Betrügereien zu verhindern, einigen sich Alice und der Makler vorab auf folgendes Signaturschema (Abbildung 5): Alice überlässt dem Makler zwei mit einem Kombinationsschloss gesicherte Tresore. In einem befindet sich ein von Alice unterschriebener Brief mit Kaufzusage, im anderen liegt eine unterschriebene Absage. Wenn Alice sich entschieden hat, übermittelt sie die Kombination des Ja-Tresors oder des Nein-Tresors. Der Makler kann nun den entsprechenden Panzerschrank aufschließen und hat anschließend ein Schreiben in der Hand, das Alices Zu- oder Absage bestätigt.

Informationstechnisch gesehen hat Alice ein Bit übermittelt und es digital signiert. Ein ähnliches Verfahren lässt sich mithilfe der Kryptografie umsetzen (Abbildung 6). Hierzu nimmt Alice zwei zufällige 128-Bit-Folgen J und N. Mit einer kryptografischen Hashfunktion berechnet sie daraufhin deren Hashwerte H(J) und H(N). Diese beiden übergibt sie dem Makler vor der Abreise. Wenn sie dann per Telefon ihre Entscheidung übermittelt, nennt sie entweder J (Zusage) oder N (Absage). Der Makler kann nun durch Berechnen des Hashwerts aus J oder N verifizieren, ob tatsächlich H(J) oder H(N) herauskommt. Dieses Verfahren lässt sich leicht auf mehrere Bits erweitern, indem Alice für jedes Bit zwei 128-Bit-Werte (Kombinationen) und zugehörige Hashwerte (Tresore) hinterlässt.



Um ein Bit zu signieren, veröffentlicht Alice Zufallszahl 1 oder Zufallszahl 2.

Mit dem Lamport-Diffie-Signaturverfahren signiert der Nutzer nur ein Bit. Wenn es mehrere Bits sein sollen, muss er die Methode mehrfach anwenden. Jeden Schlüssel kann er nur einmal benutzen. Dieses ineffektive Verfahren lässt sich jedoch deutlich verbessern (Abb. 6).

Alice macht auch halbe Sachen

Die beschriebene Methode nennt sich Lamport-Diffie-Signaturverfahren. Die Hashwerte (entsprechen den Tresoren) bilden den öffentlichen Schlüssel, die zufälligen Bitfolgen J und N (entsprechen den Tresorkombinationen) den privaten. Das Besondere an diesem Algorithmus ist, dass Alice zum Signieren die Hälfte ihres privaten Schlüssels (J oder N) bekannt machen muss. Außerdem kann sie jedes Paar aus öffentlichem und privatem Schlüssel nur einmal nutzen. Bei den bisher verbreiteten Signaturverfahren (beispielsweise RSA) gibt es eine solche Beschränkung nicht.

Es dürfte klar sein, dass Lamport-Diffie-Signaturen kaum praxistauglich sind. Der Aufwand, den Alice und der Makler treiben müssen, um ein einziges Bit zu signieren, ist schlichtweg zu groß. Es gibt jedoch eine ganze Reihe von Tricks, mit denen sich dieses Verfahren deutlich effektiver gestalten lässt. Ein solches Verfahren ist das Extended Merkle Signature Scheme (XMSS), das unter deutscher Beteiligung als RFC 8391 standardisiert wurde (siehe Kasten „Post-Quantum made in Germany").

Doch allen Optimierungsmöglichkeiten zum Trotz bleiben hashbasierte Signaturen in wichtigen Parametern (beispielsweise der Länge der Signatur oder dem Aufwand für die Schlüsselgenerierung) stets um ein Vielfaches hinter RSA und anderen Verfahren zurück. Dass die Zahl der Signaturen begrenzt ist, ist zu verschmerzen, da es Möglichkeiten gibt, sie auf einen hohen Wert zu setzen. Man muss jedoch bei der Implementierung aufpassen, dass das gleiche Schlüsselmaterial nicht ein zweites Mal eingesetzt wird.

Manchmal geht es auch ohne komplexe Mathematik

Der Vorteil von hashbasierten Signaturen liegt vor allem darin, dass sie ohne komplexe Mathematik auskommen und beweisbar sicher sind. Methoden dieser Art sind daher vor allem dann interessant, wenn es weniger um Performanz als um langjährige Sicherheit geht, etwa bei Firmware-Updates auf eingebetteten Systemen.

Neben Gitter-, Code- und Hashverfahren gibt es drei weitere Familien, die als quantensicher gelten. Nichtkommutative Verfahren setzen darauf, dass man bestimmte mathematische Operationen in der Reihenfolge nicht vertauschen kann, was die

Post-Quantum made in Germany

Einer der ersten Standards für ein Post-Quantum-Verfahren kommt aus Deutschland. Forscher der TU Darmstadt und des IT-Sicherheitsunternehmens genua haben das hashbasierte Verfahren XMSS (Extended Merkle Signature Scheme) zur Praxistauglichkeit gebracht und in einer implementierbaren Form spezifiziert. Es basiert auf der gleichen Idee wie das Lamport-Diffie-Verfahren, ist jedoch aufgrund verschiedener Optimierungen deutlich effektiver.

Der private Schlüssel von XMSS besteht wie bei Lamport-Diffie aus Zufallswerten, die gehasht werden. Ein Pseudozufallsgenerator generiert die Werte aus einem Startwert (Seed). Anstatt jeden Zufallswert zu speichern, muss der Signierer nur den Seed ablegen, was den privaten Schlüssel deutlich verkürzt. Der öffentliche XMSS-Schlüssel besteht nicht etwa aus einem Hashwert pro Zufallswert, wie beim Lamport-Diffie-Verfahren, sondern aus nur einem Hashwert. Bei ihm handelt es sich um die Wurzel eines Hashbaums (Hashbäume sind eine gängige Methode, beliebig viele Datensätze mit demselben Hashwert vor unbemerkter Veränderung zu schützen).

Eine weitere Optimierung besteht darin, dass XMSS nicht bitweise arbeitet, sondern stets zwei oder vier Bits auf einmal signiert. Das jeweilige Bitmuster, beispielsweise 0110, wird dabei durch die Anzahl der Aufrufe der Hashfunktion codiert.

Insgesamt führen die Verbesserungen dazu, dass XMSS im Vergleich zur Lamport-Diffie-Methode mit deutlich kürzeren Schlüsseln und Signaturen auskommt. Trotz allem sind XMSS-Signaturen immer noch um ein Vielfaches länger als die herkömmlicher Verfahren wie RSA oder DSA. Außerdem ist das Generieren eines XMSS-Schlüsselpaars deutlich aufwendiger als der entsprechende Vorgang bei derzeit verbreiteten Signatur-Algorithmen.

Auf dem Weg zum Standard

Um XMSS zu verbreiten, reichte das Forscherteam in Kooperation mit Experten der TU Eindhoven einen Entwurf für einen Internetstandard ein. 2018 veröffentlichte die Internet Engineering Task Force (IETF) das Verfahren als RFC 8391. Es handelt sich dabei um einen Informational RFC, er hat also nicht den Standardisierungsprozess der IETF durchlaufen. Viele Informational RFCs haben jedoch große Bedeutung als Defacto-Standards erlangt. Genau dies erhoffen sich die Entwickler auch von RFC 8391.

Lösung bestimmter Gleichungen erschwert. Multivariate Kryptografie nutzt nichtlineare Gleichungssysteme, die im Allgemeinen schwer, in Spezialfällen aber einfach zu lösen sind. Das Leiterproblem (wie breit ist ein Raum, in dem zwei sich kreuzende Leitern stehen, abhängig von deren Länge und der Höhe der Überkreuzung?) führt zu so einem Gleichungssystem. Und schließlich gibt es die bisher noch vergleichsweise junge und kleine Familie der Isogenie-basierten Verfahren, die ausnutzen, dass in bestimmten Graphen (unter einem Graphen kann man sich ein Labyrinth vorstellen) verschiedene Wege zum gleichen Ziel führen können.

Welche Post-Quanten-Verfahren sich am Ende durchsetzen, ist noch nicht ausgemacht. Für nichtkommutative Methoden sieht es schlecht aus, fast alle davon wurden gebrochen. Im NIST-Wettbewerb war nur eines vertreten, und das hat es nicht in die zweite Runde geschafft. Code- und Hashverfahren gelten als sicher, haben aber lange Schlüssel oder lange Signaturen als Nachteil. Am vielversprechendsten sieht es derzeit für gitterbasierte und multivariate Kryptografie aus, die noch mit zwölf beziehungsweise vier Vertretern im NIST-Wettbewerb stehen. Sicherheitsberater, Softwareentwickler und Studenten in aller Welt werden sich also mit der Schnecke im 250-dimensionalen Salatfeld und Leiterproblemen beschäftigen müssen.

(jd@ix.de)

Quellen

Claus Diem, Klaus Schmeh; Kryptografie; Gegen die Apokalypse; Algorithmenwettbewerb zur Post-Quanten-Kryptografie, *iX* 6/2018, S. 116

Klaus Schmeh

ist Berater bei der Gelsenkirchener Firma cryptovision, Buchautor und Blogger (www.schmeh.org).

Artificial Intelligence und Hybrid IT

Die Band „Police" veröffentlichte 1981 ihr Album „Ghost in the Machine". Was als „Ghost in the Machine" Vision war, ist heute in ersten Implementierungen Realität. Lernende AI-Software, die uns helfen soll, eine komplexe Infrastruktur effizienter zu managen. Software steuert auch andere Software. Was technisch den strategischen Wechsel auf durchgängig „Software-definierte" Infrastrukturen und die Realisierung einer Private- oder besser gleich Hybrid Cloud mit sich bringt.

Alle IT-Abteilungen suchen nach Wegen, um mit gleichen Ressourcen neue Themen wie Edge Computing, DevOps, Blockchain oder Big Data schultern zu können, ohne dabei die Qualität ihrer bisherigen Arbeit zu gefährden. Auch bei dieser Transformation helfen Software-definierte Infrastrukturen.

Mit Digitalisierung und IOT wachsen die Datenmengen rasant und verteilt. Edge Computing umreißt, wie verteilt und teils hybrid sich das „Data Center" künftig darstellen wird. Die Daten und die darunterliegende Technologie müssen geordnet zusammenspielen.

Über all diese Herausforderungen wollen wir als Fujitsu Sie so ganzheitlich wie möglich aufklären. Auf unserer diesjährigen expert4you Roadshow versammeln wir Experten von Fujitsu, Microsoft, Nutanix, VMware und NetApp, um genau über diese Aspekte einer modernen Hybrid Cloud-Strategie zu sprechen.

Nutzen Sie diese Möglichkeit der Information und Diskussion: www.fujitsu.com/de/expert4you