RSA & Co. in der Schule

Moderne Kryptologie, alte Mathematik, raffinierte Protokolle

Teil 3: Flußchiffren, perfekte Sicherheit und Zufall per Computer

von Helmut Witten, Irmgard Letzner und Ralph-Hardo Schulz

In der letzten Folge dieser Reihe (vgl. LOG IN Heft 5'98, S. 31-39) haben wir gesehen, daß polyalphabetische Chiffrierungen nach dem Vigenère-Verfahren leicht zu dechiffrieren sind, wenn das Schlüsselwort kurz im Verhältnis zum Geheimtext ist. In gewissem Sinn ist die Dechiffrierung in diesem Fall sogar leichter als bei monoalphabetischen Chiffrierungen, weil allein die Kenntnis der Schlüsselwort*länge* ausreicht, um die Chiffrierung zu „knacken". Wenn man z. B. herausgefunden hat, daß die Schlüsselwortlänge fünf beträgt, kann der Text in fünf jeweils nach Caesar chiffrierte Teiltexte zerlegt werden, indem man jeweils den 1., 6., 11. usw., dann den 2., 7., 12. usw. Buchstaben zusammenfaßt. Bei den so erhaltenen Teiltexten reicht es dann, den häufigsten Buchstaben zu ermitteln, um die Chiffrierung des Buchstabens E und damit den Schlüsselbuchstaben zu finden. Alle Schlüsselbuchstaben zusammen ergeben dann das Schlüsselwort.

Mit dem Kasiski- und dem Friedman-Test zur Bestimmung der Schlüsselwortlänge (der Achillesferse beim Vigenère-Verfahren) stehen gleich zwei wirkungsvolle Verfahren zur Kryptoanalyse zur Verfügung (vgl. Teil 2 in LOG IN Heft 5'98, S. 33 ff.).

Diese Möglichkeiten legen den Gedanken nahe, das Schlüsselwort so lang wie den Klartext zu wählen. Man erhält damit (unter gewissen Bedingungen, s. Kasten „Perfekte Sicherheit", nächste Seite) ein Verfahren, dessen „perfekte" Sicherheit sich sogar mathematisch beweisen läßt. Die Sicherheit des Verfahrens läßt sich aber auch ganz einfach plausibel machen:

Ist zum Beispiel die Chiffre
`ICFQDBQDEYYNIGTR`, dann kann der Klartext
`SCHULEMACHTSPASS` oder
`MATHEMATIKISTGUT`, aber auch
`FERIENSINDBESSER` gewesen sein.

Die passenden Schlüsseltexte dazu sind
`QAYWSXEDCRFVTGBZ`
`WCMJZPQKWOQVPAZY` und
`DYOIZOYVRVXJQOPA`. [...]

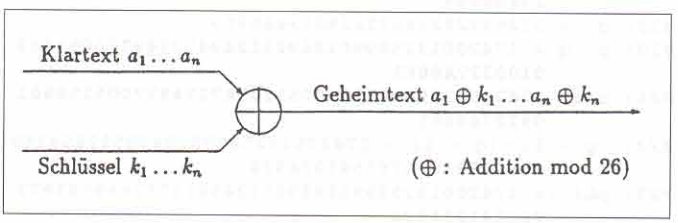
Aufgabe: Prüfen Sie das jetzt gleich nach. (Giessmann, 1995, Teil II, S. 9).

Somit beruht die Sicherheit auf der Tatsache, daß sich aus der Kenntnis des Geheimtextes keinerlei Information über den Klartext gewinnen läßt.

Man braucht für dieses Verfahren einen Schlüssel, der genauso lang ist wie der Klartext. Bei der Chiffrierung läßt man dann Klartext und Schlüssel durch eine Addition modulo 26 „zusammenfließen", für die Dechiffrierung muß der Schlüssel wieder subtrahiert werden. Aus diesem Grund spricht man auch von „Flußchiffren" (Bild 1), weitere Bezeichnungen sind „Stromchiffren", „Vernam-Verfahren" und „one-time-pad" (zur Erklärung der beiden letzten Namen s. u.). Das Verfahren bietet neben der Sicherheit noch weitere Vorteile: Es ist außerordentlich schnell und einfach mit dem Computer zu realisieren. Flußchiffren werden in der Praxis häufig verwendet, z. B. um den Funkverkehr zwischen Handys und Feststationen vor dem Abhören zu schützen (Weis, 1998, S. 39 f.).

Um wirklich perfekte Sicherheit (s. Kasten, nächste Seite) zu erreichen, muß jede mögliche Schlüsselfolge die gleiche Wahrscheinlichkeit haben. Das ist aber nur schwer zu erreichen, so daß diese Voraussetzung häufig abgeschwächt werden muß. In den Handys z. B. werden keine echten Zufallszahlen verwendet, sondern sog. Pseudozufallszahlen, die von einem Chip erzeugt werden. Damit wird aber auch das Sicherheitskonzept durchlöchert: „Geheimdienste können [...] jede Menge Daten von Handygesprächen speichern und diese zumindest offline ohne große Probleme entschlüsseln"

Bild 1: Verschlüsselung mit einer Flußchiffre.



Perfekte Sicherheit

Ein Chiffriersystem bietet perfekte Sicherheit, falls für *jedes Chiffrat* c aus der Menge C der möglichen Geheimtexte und *jeden Klartext* m aus der Menge M der möglichen Klartexte gilt:

$$p(m) = p(m \mid c);$$

hierbei ist p(m) die „a priori"-Wahrscheinlichkeit von m und p(m | c) die „a posteriori"-Wahrscheinlichkeit, nämlich daß bei bekanntem Geheimtext c dieser vom Klartext m herkommt.
Also: Wenn der Geheimtext c bekannt wird, so soll sich dadurch keine zusätzliche Information über m ergeben.

Wir zeigen folgenden Satz (Shannon, 1949):

Das one-time-pad-Verfahren besitzt perfekte Sicherheit.

Beweis des Satzes (nach A. Sgarro):

Zunächst berechnen wir in zwei Schritten ((i) und (ii)) die Wahrscheinlichkeit eines Geheimtext-Wortes; dabei ist jedes Ergebnis des Wahrscheinlichkeitsraumes von der Form (m, k, c), wobei c = m + k durch Klartext m und Schlüsselwort k bestimmt ist.

(i) $p(c = c_1 \dots c_n \wedge k = k_1 \dots k_n)$
$= p(m = c_1 - k_1 \dots c_n - k_n \wedge k = k_1 \dots k_n)$
$= p(m = c_1 - k_1 \dots c_n - k_n) \cdot p(k = k_1 \dots k_n)$
(wegen der Unabhängigkeit der Schlüssel vom Klartext)
$= p(m = c_1 - k_1 \dots c_n - k_n) \cdot \frac{1}{|K|}$
(wegen der Gleichverteilung der Schlüssel).

(ii) Summiert man nun über alle $k_1 \dots k_n \in K$, so deckt $c_1 - k_1 \dots c_n - k_n$ für festes c alle Klartexte genau einmal ab. Man erhält:

$$p(c = c_1 \dots c_n) = \sum_{k_1 \dots k_n \in K} p(c_1 = c_1 \dots c_n \wedge k = k_1 \dots k_n)$$
$$\overset{\text{(i)}}{=} \frac{1}{|K|} \sum_{k \in K} p(m = c - k)$$
$$= \frac{1}{|K|} \sum_{m \in M} p(m) = \frac{1}{|K|}.$$

Insgesamt ergibt sich

(iii) $p(m = a_1 \dots a_n \mid c = c_1 \dots c_n)$
$= \frac{p(m = a_1 \dots a_n \wedge c = c_1 \dots c_n)}{p(c = c_1 \dots c_n)}$
(nach Definition der bedingten Wahrscheinlichkeit)
$= \frac{p(m = a_1 \dots a_n \wedge k = c_1 - a_1 \dots c_n - a_n)}{p(c = c_1 \dots c_n)}$
$= \frac{p(m = a_1 \dots a_n) \cdot p(k = c_1 - a_1 \dots c_n - a_n)}{p(c = c_1 - c_n)}$
(wegen der Unabhängigkeit des Schlüssels vom Klartext)
$= \frac{p(m = a_1 \dots a_n) \cdot \frac{1}{|K|}}{\frac{1}{|K|}}$
(wegen (ii), und da die Schlüssel gleichwahrscheinlich sind)
$= p(m = a_1 \dots a_n).$

Gemäß der Definition bietet das Verfahren (bei Einhaltung der Bedingungen) perfekte Sicherheit. □

(Weis, 1998, S. 40). Die Verschlüsselung der Handygespräche verhindert aber immerhin, daß Funkamateure mithören.

Für eine absolut sichere Chiffrierung ist es daher notwendig, eine echte Zufallsfolge als Schlüsseltext (vorher) zu übermitteln. Dieses aufwendige Verfahren ist in diesem Jahrhundert tatsächlich mehrfach mit Erfolg zum Einsatz gekommen. Zunächst wollen wir aber die Erfinder der Flußchiffrierung vorstellen.

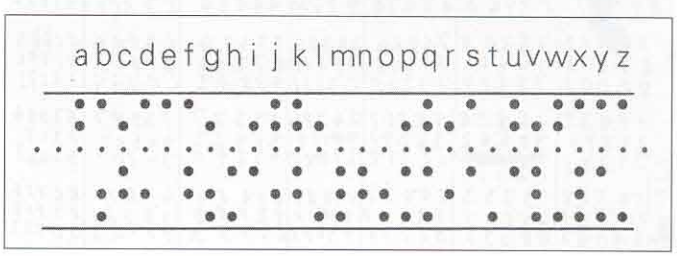
Vernam, Lochstreifen und geheime Notizblöcke

Gilbert S. Vernam (1890-1960) von AT&T entwickelte 1917 zusammen mit Major Joseph Mauborgne einen Chiffrierzusatz für die damals gerade erfundenen Fernschreiber. Vernam ließ sich diese bahnbrechende Erfindung 1918 patentieren, konnte aber leider – wie viele Kryptologen vor und nach ihm – keinen kommerziellen Erfolg damit erzielen (Bauer, 1995, S. 109; Schneier, 1996, S. 17 ff.). Er hat auch nicht mehr erleben können, daß 1963 eine Variante des von ihm erfundenen Gerätes beim sog. „heißen Draht" zwischen Washington und Moskau zu höchsten Ehren kam.

Zur Aufzeichnung der Nachrichten wurden bei Fernschreibern Lochstreifen verwendet; die Buchstaben wurden dabei meistens im sog. Baudot-Code dargestellt, der für jedes Zeichen fünf Bits verwendet. So wird z. B. der Buchstabe a durch das Bitmuster 11 000, z dagegen durch 10 001 beschrieben (Bild 2; Kippenhahn, 1997, S. 257; zum Baudot-Code vgl. Witten, 1994).

Die Verschüsselungs-Maschine von Vernam verknüpfte den Lochstreifen des Textes und einen Schlüssel-Lochstreifen zu einem dritten Lochstreifen mit dem Chiffrat. Dabei wurden die einzelnen Bits im Zweiersystem ohne Übertrag addiert, also nach dem bekannten Schema 0 + 1 = 1 + 0 = 1 bzw. 0 + 0 = 1 + 1 = 0. Wenn

Bild 2: Buchstaben, Ziffern und Zeichen werden bei Fernschreibern durch Löcher und ungelochte Stellen auf einem Papierstreifen dargestellt, die fortlaufende Reihe mit kleinen Löchern dient zum Transport des Streifens. Elektrisch leitende Fühler tasten den Streifen ab; sobald sie auf ein Loch treffen, wird ein Kontakt geschlossen.



Quelle: Kippenhahn, 1997, S. 257

der Klartext *geheim* mit dem Schlüsselwort *vernam* verschlüsselt wird ergibt sich also:

Klartext:	01 011 10 000 00 101 10 000 01 100 00 111
Schlüssel:	*01 111 10 000 01 010 00 110 11 000 00 111*
Geheimtext:	00 100 00 000 01 111 10 110 10 100 00 000

Anmerkung: Da das Bitmuster beim Schlüssel nicht zufällig gewählt wurde und in der ersten und zweiten Zeile zweimal die gleichen Buchstaben aufeinanderfallen, ergeben sich im Geheimtext auffallend viele Nullen!

Zum Entschlüsseln muß der Empfänger im Besitz des zum Chiffrieren verwendeten Schlüsselstreifens sein; die Addition des Schlüssels zum Geheimtext führt wieder auf den Klartext. Zum Verschlüsseln kann daher das gleiche Gerät wie zum Entschlüsseln benutzt werden. Es handelt sich im Kern um die sog. XOR-Schaltung (exklusives Oder), die technisch leicht realisiert werden kann.

Im Anfang klebte die Gruppe um Vernam die Schlüsselstreifen zu einem Ring zusammen, so daß der Klartext länger als der Schüssel sein konnte. Da in diesem Fall der Schlüssel zyklisch wiederverwendet wird, handelt es sich bei dieser Variante um ein bitweises Vigenère-Verfahren. Die Chiffrierung läßt sich daher mit den gleichen Methoden „knacken", die wir in der letzten Folge im Zusammenhang mit dem Vigenère-Verfahren kennengelernt haben. Bruce Schneier weist darauf hin, daß dieses unsichere Verfahren immer noch in kommerziellen Software-Paketen aus der MS-DOS- und Macintosh-Welt anzutreffen ist. „Leider ist die Wahrscheinlichkeit groß, daß es sich um eine Variante dieses Algorithmus handelt, wenn ein Sicherheitssoftwarepaket für sich beansprucht, einen ‚eigenen' Verschlüsselungsalgorithmus zu verwenden, der deutlich schneller als DES ist" (Schneier, 1996, S. 16).

Bild 3: Kryptogramm von Ché Guevara.

Quelle: Spektrum der Wissenschaft, 15 (1992), H. 12, S. 98

Vernam experimentierte später mit extrem langen Schlüsselstreifen (Kippenhahn, 1997, S. 258). Um die gleiche Zeit (nach dem 1. Weltkrieg) wurde von den deutschen Kryptologen Werner Kunze, Rudolf Schaffler und Erich Langlotz ein Verfahren entwickelt, das später auch in anderen Ländern verwendet wurde und im englischsprachigen Raum den Namen *one time pad* erhielt. Der Grundgedanke war, daß der Klartext zunächst in eine Zahlenfolge verwandelt wird. Der Schlüssel besteht aus einer langen Reihe von Ziffern, die ohne Übertrag zu der Klartextziffernfolge addiert wird; zum Entschlüsseln muß die Schlüsselfolge wieder subtrahiert werden. (Nur im Dualsystem stimmen Addition und Subtraktion überein, wenn ohne Übertrag gerechnet wird.)

Der Name *one time pad* erklärt sich daraus, daß der „Ziffernwurm" auf einem Notizblock verzeichnet war. In Deutschland wurde der diplomatische Dienst in der Weimarer Republik mit Blöcken von je fünfzig Seiten mit jeweils fünfundachtzig Fünfergruppen aus zufällig gewählten Ziffern ausgestattet. „Keine zwei Seiten waren gleich, und jede durfte nur einmal verwendet werden. Nachdem eine Nachricht verschlüsselt worden war, mußte das benutzte Blatt vernichtet werden" (Kippenhahn, 1997, S. 165). Natürlich mußten Sender und Empfänger mit dem gleichen Block ausgestattet werden. Dieses Verfahren wird vermutlich noch heute eingesetzt. Im folgenden bringen wir einige Beispiele aus der Geschichte dieses Jahrhunderts.

Perfekte (und nicht ganz perfekte) Sicherheit in der Praxis

„Als Soldaten der bolivianischen Armee, 1967 den Revolutionär Ché Guevara gefangennahmen und töteten, fanden sie bei ihm ein Papier, auf dem er eine Nachricht an den kubanischen Präsidenten Fidel Castro chiffriert hatte. [...] Zunächst setzte er die Buchstaben seines (spanischen) Klartextes nach einem festen Schema in Zahlen um [...]. Im nächsten Schritt notierte er die Ziffernfolge, in handliche Fünfergruppen aufgeteilt, auf das Papier; sie bildet die oberste Zeile jeder Dreizeilengruppe. Darunter schrieb er den Schlüsseltext, eine zufällig bestimmte Zahlenfolge, die nur Castro und er selbst kannten. Dann addierte er an jeder Position die Ziffer des Klar- und die des Schlüsseltextes und schrieb das Ergebnis (ohne Zehnerübertrag) darunter" (Bennet u. a., 1992, S. 98, s. Bild 3). Diese Nachricht konnte dann über Kurzwellenfunk nach Havanna geschickt werden und mit dem dort ebenfalls vorhandenen Schlüsseltext durch Subtraktion ohne Zehnerübertrag wieder entschlüsselt werden. Die Chiffre wurde nicht gebrochen. Trotzdem gelang es Ché Guevara (Bild 4, nächste Seite) bekanntlich nicht, in Bolivien eine Revolution zu entfesseln.

Über ein anderes Beispiel berichtet Rudolf Kippenhahn: „Am 20. Juni 1953 wurde im New Yorker Staats-

Bild 4:
Ché Guevara liebte Zigarren aus Havanna und perfekte Sicherheit beim Chiffrieren.

Foto: R. Burri/Magnum

gefängnis Sing-Sing das amerikanische Ehepaar Ethel und Julius Rosenberg hingerichtet. Zwei Jahre zuvor waren sie wegen Verrats von Atomgeheimnissen zum Tode verurteilt worden. Sie hatten für die Sowjetunion spioniert. Eine Panne ermöglichte ihre Enttarnung. Die Sowjets hatten den gleichen Schüsseltext mehrfach benutzt. Auch der verantwortliche Offizier mußte diesen Fehler mit dem Leben bezahlen" (Kippenhahn, 1997, S. 166).

Fehler dieser Art wurden aber nicht häufig begangen. „Zahlreiche Nachrichten sowjetischer Spione wurden mit One-Time-Pads chiffriert. Diese Mitteilungen sind bis heute geschützt und werden es bis in alle Ewigkeit bleiben. Supercomputer können end- und erfolglos mit diesem Problem beschäftigt werden. Selbst die Aliens von Andromeda, die mit riesigen Raumschiffen und unvorstellbarer Rechenleistung vielleicht einmal auf der Erde landen, werden die sowjetischen, mit One-Time-Pads chiffrierten Botschaften nicht entschlüsseln können (falls sie nicht eine kleine Reise in die Vergangenheit unternehmen und sich die One-Time-Pads besorgen)" (Schneier, 1996, S. 20).

Die bislang erfolgreichste Anwendung des One-Time-Pad-Verfahrens wurde erst Mitte der siebziger Jahre bekannt: die Übermittlung der entzifferten Enigma-Funksprüche.

Das deutsche Militär setzte im zweiten Weltkrieg zur Verschlüsselung der Funksprüche in großem Umfang die Enigma ein, deren Code als absolut sicher galt (s. Batzer, 1996; Marian Kassovic hat ein Programm für DOS-Rechner geschrieben, in dem die Enigma mit der Original-Walzenbelegung simuliert wird. Es kann unter: ftp://ftp.uni-hamburg.de/pub/virus/crypt/enigma/simulators/enigma22.exe heruntergeladen werden. Das selbstentpackende Archiv enthält auch eine Dokumentation zur Enigma).

Bereits zu Beginn der 30er Jahre waren jedoch polnischen Kryptologen Einbrüche in den Enigma-Code gelungen. Darauf aufbauend konnten nach Kriegsausbruch in einem kleinen englischen Ort zwischen Oxford und Cambrigde (Bletchley Park) durch ein Team von Mathematikern und Linguisten sehr viele Funksprüche dechiffriert werden. Dieses für die Alliierten äußerst wichtige Projekt lief unter dem Namen „Ultra" (Lewin, 1981).

Die Geschichte von der erfolgreichen Dechiffrierung in Blechtley Park ist inzwischen so bekannt, daß wir uns in diesem Artikel auf Literaturhinweise und einige weniger bekannte Details beschränken können. Sie wurde inzwischen von Richard Harris zu einem spannenden Roman verarbeitet, der demnächst auch verfilmt werden soll (Harris, 1995). In dem Roman wird das Ultra-System anschaulich und detailgetreu beschrieben.

Erstaunlicherweise ist es den Briten und Amerikanern gelungen, das Projekt Ultra 30 Jahre lange geheimzuhalten, obwohl neben den führenden Politikern, Militärs und Wissenschaftlern wie Alan Turing Tausende von Fernmeldesoldaten, Verschlüßlern und Hilfskräften eingeweiht waren (allerdings drohten bei Verrat des Geheimnisses auch drakonische Strafen). Welche Rolle Alan Turing, der Mathematiker und Stammvater der Informatik, als Kryptologe in Bletchley Park spielte, wird sehr genau in seiner lesenswerten, von Andrew Hodges verfaßten Biographie geschildert (Hodges, 1994).

Im Projekt Ultra konnten Auswirkungen auf den Verlauf der Kämpfe im 2. Weltkrieg nur erzielt werden, wenn die entschlüsselten Funksprüche schnell und sicher an die Front übermittelt wurden. Dabei mußte sichergestellt werden, daß die Deutschen und Italiener auf keinen Fall erfuhren, daß der Enigma-Code gebrochen worden war. Zur Übertragung nutzten die Alliierten deshalb die perfekte Sicherheit des One-Time-Pad-Verfahrens. Zur Übermittlung der Erkenntnisse von Bletchley Park an die Front erhielten nur ausgewählte, den einzelnen Truppeneinheiten zugeordnete Nach-

Bild 5:
Der Verbrennungsofen im Königlichen Palast in Caserta, in dem die Ultra-Funksprüche an den Oberbefehlshaber Mittelmeer, General Alexander, nach Gebrauch verbrannt wurden.

Quelle:
Lewin, 1981, vor S. 241

Modulares Rechnen ohne modulare Verwirrung

Von Carl Friedrich Gauß stammt die heute noch übliche Schreibweise mit Kongruenzen als Verallgemeinerung der Gleichheit:

$$a \equiv b \pmod{n}.$$

Hierbei steht n für eine natürliche Zahl, während a und b zwei ganze Zahlen sein können, gesprochen „a kongruent b modulo n". Damit ist gemeint, daß a und b bei der Division durch n den gleichen Rest R haben ($0 \le R \le n-1$) oder (gleichbedeutend dazu), daß n ein Teiler von (a – b) ist (d. h., es existiert eine ganze Zahl t mit $a - b = t \cdot n$).

Unglücklicherweise bezeichnet mod in vielen Programmiersprachen (z. B. PASCAL, MODULA, OBERON) demgegenüber eine Funktion: a mod n liefert den Rest bei der Division von a durch n. A. Beutelspacher verwendet in seinem Kryptologiebuch (Beutelspacher, 1993) mod (kleingeschrieben) für die Gaußsche Kongruenzschreibweise, MOD (großgeschrieben) als Rest-Operator. Da hier Verwirrung vorprogrammiert ist, taucht die Gaußsche Schreibweise in seinem neuesten Buch „Geheimsprachen" (Beutelspacher, 1997) gar nicht mehr auf, dafür steht dort mod (kleingeschrieben) für den Restoperator. Schneier verwendet mod (kleingeschrieben) sowohl für die Gauß-Schreibweise als auch für den Restoperator, die jeweilige Bedeutung muß aus dem Kontext erschlossen werden (Schneier, 1996, S. 284 f.). Das mag für den normalen Leserkreis dieses Standardwerkes zur Kryptologie vertretbar sein, für Lernende sollte eine „Doppelbelegung" von mod wegen der Gefahr kognitiver Interferenzen vermieden werden.

Ein weiterer Faktor der Verwirrung ist der folgende Unterschied: Während in der Mathematik mit dem Rest R üblicherweise eine natürliche Zahl ($0 \le R \le n-1$) gemeint ist, liefert a mod n z. B. in PASCAL eine negative Zahl, falls $a < 0$ ist. Daher berechnet der mod-Operator nur für nichtnegative Werte von a den korrekten Rest, andernfalls muß zum Ergebnis noch einmal n addiert werden. Der Rest-Operator in den Sprachen mit C-ähnlicher Syntax (C, C++, JAVA) wird durch das Prozentzeichen % dargestellt und liefert u. U. ebenfalls negative Werte.

Aus diesem Grund verwenden wir in unserer Artikelserie neben der Gauß-Schreibweise die programmiersprachenunabhängige Bezeichnung $R_n(a)$ für den Rest von a modulo n mit $(0 \le R_n(a) \le n-1)$. Eine Implementierung könnte z. B. in der Syntax von PASCAL so aussehen:

```
FUNCTION R(a, n: longint):longint;
BEGIN
  IF a >= 0
    THEN R := a mod n
    ELSE R := (a mod n) + n
END {R};
```

Bei dem Verfahren von Vernam (wie auch beim Caesar- und Vigenère-Verfahren) können wir uns auf die Addition bzw. Subtraktion modulo einer Zahl beschränken. Die Multiplikation, Division und vor allem das Potenzieren mit den Umkehrungen Logarithmieren und Wurzelziehen wird uns in den nächsten Folgen beschäftigen.

Man rechnet leicht nach, daß für vier ganze Zahlen a, b, c und d gilt:

$$a \equiv b \pmod{n} \wedge c \equiv d \pmod{n} \Rightarrow a \pm c \equiv b \pm d \pmod{n} \text{ bzw.}$$
$$R_n(a) = R_n(b) \wedge R_n(c) = R_n(d) \Rightarrow R_n(a \pm c) = R_n(b \pm d).$$

Bei der Vertauschung von der Restoperation mit Addition bzw. Subtraktion muß beachtet werden, daß eine weitere Restbildung erforderlich sein kann, da $0 \le R_n(a) \le n-1$ gelten muß:

$$R_n(a \pm b) = R_n(R_n(a) \pm R_n(b)).$$

Wie beim normalen Rechnen kann man die Subtraktion auf die Addition der entsprechenden inversen Zahl zurückführen, die aber – im Gegensatz zu den „normalen" Zahlen – nicht negativ sein muß, da

$$R_n(-a) = R_n(n-a) \text{ gilt.}$$

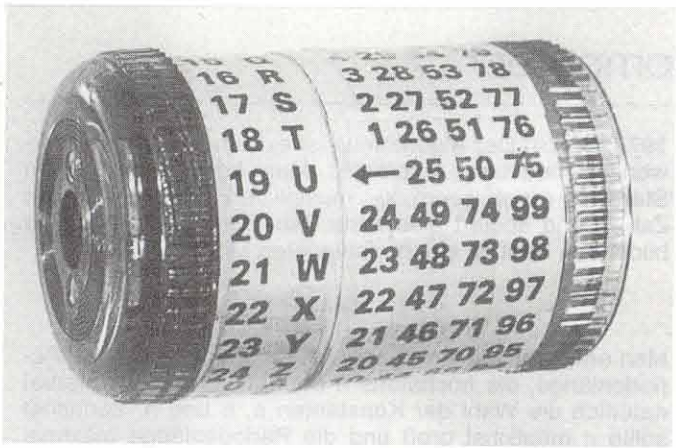
(Im Caesar-Verfahren z. B. kann die Dechiffrierung durch die Subtraktion von 3 oder die Addition von 23 erfolgen, falls ein Alphabet mit 26 Buchstaben verwendet wird.)

richtenoffiziere der „Special Liaison Units" die „Ultra"-Funksprüche, die nach Gebrauch verbrannt werden mußten (Bild 5, vorige Seite; zu Einzelheiten s. Lewin, 1981, S. 164 ff.).

Wir wollen damit die Geschichte des *one time pad* verlassen. Diese Chiffriermethode ist das bislang einzige bekannte Verfahren mit *beweisbarer* Sicherheit. Allerdings hat diese Sicherheit auch ihren Preis: Es müssen sehr viele Zufallszahlen mit guten statistischen Eigenschaften erzeugt und *vor* der eigentlichen Kommunikation ausgetauscht werden. Dieser Aufwand war in den vorgestellten Fällen sicherlich angemessen und auch realisierbar, wenn man sich die amerikanischen und sowjetischen Diplomaten mit den One-Time-Pad-Lochstreifen im Koffer vorstellt. Für einen breiten Einsatz bei der geschützten Kommunikation für Millionen von Menschen (man denke nur an die Unzahl von Handygesprächen) kann ein solcher Aufwand sicherlich nicht getrieben werden. (Wir kommen weiter unten auf dieses Problem zurück.)

Hinweise für den Unterricht

Wie in den letzten beiden Folgen sollen jetzt didaktische Probleme bei der Behandlung dieses Chiffrierverfahrens im Mathematik-, ITG- und Informatikunterricht diskutiert und Anregungen für den Unterricht gegeben werden. Inhaltlich geht es zunächst um das modulare Rechnen, das für die ganze moderne Kryptologie sehr wichtig ist und uns auch in den nächsten Folgen beschäftigen wird.



Quelle: Kippenhahn, 1997, S. 336

Bild 6: Eine selbstgebastelte Chiffriermaschine.

Ein didaktisches Problem ergibt sich schon bei der Schreibweise bzw. der Terminologie. Die Division mit Rest und elementare Teilbarkeitsregeln werden nur in der Grundschule behandelt, für den Mathematikunterricht der Oberschulen im Pflichtbereich ist das z. Z. (leider!) kein Thema mehr. Die Lernenden kennen daher die gebräuchlichen Schreibweisen nicht. Hinzu kommt, daß die in der Literatur verwendeten Bezeichnungen in verwirrender Weise unterschiedlich und teilweise inkonsistent sind (s. Kasten, vorige Seite: „Modulares Rechnen ohne modulare Verwirrung").

Man kann ganz einfach beginnen, denn modulares Rechnen ist uns aus dem Alltag vertraut. So werden beispielsweise Uhrzeiten modulo 12 oder modulo 24 angegeben, Wochentage modulo 7, zurückgelegte Kilometer bei Zählern in Kraftfahrzeugen modulo 100 000. Zur Übung können Aufgaben der folgenden Art dienen:

▷ Es ist jetzt 11 Uhr.
Wieviel Uhr ist es nach 100 Stunden?
▷ Heute ist Donnerstag.
Welcher Wochentag ist nach 47 Tagen?
▷ Es ist der Monat August.
Welcher Monat ist nach 50 Monaten?

Das modulare Rechnen ist bei den Moduln 2 und 10 besonders einfach. Wie wir bei dem Verfahren von Vernam und dem Kryptogramm von Ché Guevara gesehen haben, läßt man lediglich den „Übertrag" weg. Daher könnten diese Methoden zur Chiffrierung schon in der Grundschule vorgestellt werden, ohne daß das modulare Rechnen explizit problematisiert werden müßte.

Bei den in der letzten Folge (LOG IN Heft 5'98, Teil 2, S. 31-39) vorgestellten Verfahren nach Caesar bzw. Vigenère werden zum Chiffrieren ebenfalls modulare Additionen und zum Dechiffrieren modulare Subtraktionen durchgeführt, in diesem Fall aber mit dem Modul 26. Die vorgestellten Hilfsmittel (Alberti- oder Caesar-Scheibe und Vigenère-Tafel) ersparen die Umsetzung von Buchstaben in Zahlen und umgekehrt. Außerdem erledigen Scheibe bzw. Tafel – sozusagen versteckt – die korrekten Rechnungen modulo 26. Natürlich kann man mit diesen Geräten auch das Vernam-Verfahren realisieren, wenn man einen zufällig gewählten „Buchstabenwurm" als Schlüssel mit dem Klartext verknüpft. Die Schüler kennen häufig das Verfahren, als Schüssel z. B. einen Romantext oder eine Zeitungsmeldung zu verwenden. Da in diesem Fall die Schlüsselbuchstaben nicht zufällig verteilt sind und die Zeichenhäufigkeit des Schlüsseltextes „durchscheint", ist dieses Verfahren nicht wirklich sicher.

Das Rechnen „per Hand" wird leichter, wenn man I und J zu einem Buchstaben zusammenfaßt. Damit bleibt der Klartext verständlich, und die Zahl der Buchstaben reduziert sich auf 25, was das modulare Rechnen erheblich vereinfacht. In diesem Fall müssen die Buchstaben nach einem festen Schema in die Zahlen 0 bis 24 umgewandelt werden. Im Buch von Kippenhahn findet sich eine Bastelanleitung für ein einfaches Rotorgerät, das die modulare Addition bzw. Subtraktion modulo 25 mechanisiert: „Der Londoner Journalist Robert Matthews veröffentlichte 1989 eine Bau- und Benutzeranleitung für ein einfaches Verschlüsselungsgerät. Es besteht aus zwei Papierstreifen, die am oberen und unteren Rand zusammengeklebt werden und auf einem Zylinder gegeneinander verdreht werden können" (Kippenhahn, 1997, S. 335). Matthews hatte vorgeschlagen, die Papierstreifen auf eine Filmdose zu kleben, Kippenhahn verwendete einen Bleistiftanspitzer (Bild 6).

Zum Selberbasteln muß die in Bild 7 wiedergegebene Tabelle mit einem Kopierer so vergrößert oder verkleinert werden, daß sie genau auf den Zylinder paßt. (Statt Filmdosen und Bleistiftanspitzern können z. B. auch zwei ineinander geschobene Pappkerne von Küchenrollen verwendet werden.) Beide Streifen müssen so auf den Zylinder geklebt werden, daß sie sich gegeneinander verdrehen lassen.

0	A	1	26	51	76
1	B	←	25	50	75
2	C	24	49	74	99
3	D	23	48	73	98
4	E	22	47	72	97
5	F	21	46	71	96
6	G	20	45	70	95
7	H	19	44	69	94
8	I	18	43	68	93
9	K	17	42	67	92
10	L	16	41	66	91
11	M	15	40	65	90
12	N	14	39	64	89
13	O	13	38	63	88
14	P	12	37	62	87
15	Q	11	36	61	86
16	R	10	35	60	85
17	S	9	34	59	84
18	T	8	33	58	83
19	U	7	32	57	82
20	V	6	31	56	81
21	W	5	30	55	80
22	X	4	29	54	79
23	Y	3	28	53	78
24	Z	2	27	52	77

Die Funktionsweise dieser Maschine soll an einem einfachen Beispiel erläutert werden: Der Schlüssel beginnt mit den Zahlen 07 19 12 03, das erste Wort des Klartextes sei *rose*. Dann werden die Streifen so verdreht, daß 07 auf das r zeigt. Beim Pfeil liest man dann als ersten Buchstaben des Geheimtextes y ab. Ent-

Bild 7: Kopiervorlage zum Selbstbau einer Chiffriermaschine.

Quelle: Kippenhahn, 1997, S. 337

Zufall per Computer?

Computer sollten sich deterministisch verhalten (obwohl man manchmal den Eindruck gewinnen kann, daß die Messung der Zeiträume zwischen zwei Abstürzen eines Windows-Rechners einen hervorragenden Zufallsgenerator ergeben würde). Schon in der Frühzeit der Computer hat man Algorithmen gesucht, die zwar deterministisch arbeiten, aber eine Ausgabe erzeugen, die zufällig erscheint. Solche Programme nennt man (Pseudo-)Zufallsgeneratoren.

Der erste dieser Generatoren geht auf einen Vorschlag von John von Neumann zurück und wurde im Jahr 1946 veröffentlicht (Engel, 1977, S. 33). Man nimmt z. B. das Quadrat einer sechsstelligen Zahl, schneidet vorne und hinten gleich viele Ziffern ab, so daß wieder eine sechsstellige Zahl entsteht, die wiederum quadriert wird usw. Hierzu ein einfaches Beispiel:

$(123456)^2 = 015241383936$
$(241383)^2 = 058265752689$
$(265752)^2 = 070624125504$
$(624125)^2 = 389532015625$
$(532015)^2 = \ldots$

Leider passiert es in kaum vorhersagbarer Weise, daß die „zufällige" Folge nur allzu deterministisch wird:

$(333167)^2 = 111000249889$
$(000249)^2 = 000000062001$
$(000062)^2 = 000000003844$
$(000003)^2 = 000000000009$
$(000000)^2 = \ldots$

Heute arbeiten praktisch alle in Programmbibliotheken mitgelieferten Zufallsgeneratoren nach der „Linearen Kongruenz Methode" (D. H. Lehmer, 1948; s. Engel, 1977, S. 33). Der Algorithmus ist extrem einfach und verwendet modulare Arithmetik. Man beginnt mit einem Startwert (engl. seed) x_0, multipliziert mit einer festen Zahl a und addiert eine feste Zahl b zum Ergebnis und bildet den Rest zu einem geeigneten Modul n:

$$x_{i+1} = R_n(ax_i + b).$$

Man erhält mit diesem Generator eine Folge mit einer Periodenlänge, die höchstens n beträgt. Kritisch ist hierbei natürlich die Wahl der Konstanten a, b und n. Zunächst sollte n möglichst groß und die Periodenlänge maximal sein. Man kann hierzu Bedingungen formulieren, die recht einfach sind, falls n eine Zweierpotenz ist: a muß bei der Division durch 4 den Rest 1 haben und b muß ungerade sein. Wie man am Beispiel a = 1 und b = 1 sieht, reichen diese Bedingungen jedoch nicht aus, um einen guten Zufallsgenerator zu erhalten, weitere statistische Tests müssen bestanden werden (Forster, 1996, S. 72 ff.; Kuntze, 1998). Eine Tabelle mit geeigneten Konstanten für lineare Kongruenzgeneratoren findet sich z. B. im Buch von Schneier (Schneier, 1996, S. 426).

Diese Zufallsgeneratoren haben mit gut gewählten Konstanten hervorragende statistische Eigenschaften und sind daher für viele Simulationsprogramme geeignet, nicht aber für den ernsthaften Einsatz in der Kryptographie, da die Konstanten schon aus wenigen Folgengliedern berechnet werden können (Stallings, 1995, S. 133 f.). Für den Einsatz in der Kryptologie werden daher andere Zufallsgeneratoren verwendet, z. B. Schieberegister (vgl. z. B. Beutelspacher, 1993, S. 66 ff.; Kuntze, 1998; Schneier, 1996, S. 425 ff.; Stallings, 1995, S. 129 ff., dort werden diese und andere Pseudozufallsgeneratoren beschrieben und weiterführende Literarurhinweise gegeben).

sprechend erhält man für die anderen Buchstaben *heh* als Chiffrat. Zur Dechiffrierung geht man umgekehrt vor: Der Pfeil wird auf den ersten Geheimbuchstaben y eingestellt, unter 07 kann dann der Klartextbuchstabe r abgelesen werden usw.

Dieses Gerät können Schüler der Sekundarstufe I einfach selbst herstellen und mit einem geeigneten „Zahlenwurm" (z. B. aus einem Telefonbuch) nahezu perfekte Sicherheit beim Chiffrieren erlangen.

Mit dem Computer ist das Rechnen mit jedem beliebigen Modul einfach und unkompliziert durchzuführen. Im Informatikunterricht kann das Vernam-Verfahren bereits im ersten Lernjahr von den Schülern programmiert werden. Wenn man den Klartext, den Schlüssel und das Chiffrat jeweils in Textdateien ablegt, ist das Ergebnis leicht zu überprüfen.

An diese Aufgabe schließt sich in natürlicher Weise die Frage an, wie man zu der Schlüsseldatei mit Zufallszahlen (oder Zufallstexten) kommen kann. Dabei stellt sich heraus, daß „echter" Zufall nicht einfach zu haben ist: „Zeichnen Sie auf, was ein unzuverlässiger Geigerzähler während einer Fahrt über holprige Straßen von einer radioaktiven Probe im Fahrzeug mißt, und überlagern Sie diesen Datenstrom mit dem digitalisierten Rauschen eines Wasserfalls sowie dem Blöken eines Schafes: Da gibt jeder Geheimdienst auf" (Wobst, 1997, S. 53; vgl. Schneier, 1996, S. 482 ff.).

Nun könnte man die Hoffnung haben, daß z. B. der lineare Kongruenzgenerator mit $x_{i+1} = R_n(ax_i + b)$ (s. Kasten auf dieser Seite: „Zufall per Computer?") das Problem der Zufallszahlen für die Flußchiffren in eleganter Weise lösen könnte: Man muß nur noch den Startwert x_0 sowie a, b und n übermitteln, um mit perfekter Sicherheit chiffrieren zu können. Das ist natürlich zu schön, um wahr zu sein. Diese Zufallsfolge ist noch viel zu deterministisch, um wirklich sicher zu sein. Man kann schon aus wenigen Werten die Parameter a, b und n berechnen und damit alle weiteren Folgenglieder bestimmen (Schneier, 1996, S. 425; Stallings, 1995, S. 132 ff.). Im Kasten „Zufall per Computer?" sind Literaturhinweise zu anderen, kryptologisch geeigneteren Zufallsgeneratoren angegeben.

Pseudozufallsgeneratoren sind wegen ihrer großen praktischen Bedeutung auch außerhalb der Kryptolo-

gie für den Unterricht interessant (vgl. z. B. Engel, 1977, S. 33 ff.; Niedersächsisches Kultusministerium, 1990, S. 160 ff.; Engel, 1991, S. 101 ff.).

Wie geht es weiter?

Modulare Arithmetik und Zufallsgeneratoren spielen auch in der modernen, asymmetrischen Kryptographie eine grundlegende Rolle. In der nächsten Folge werden wir uns mit den Möglichkeiten der unterrichtlichen Behandlung am Beispiel des berühmten RSA-Algorithmus beschäftigen.

(Fortsetzung folgt)

StD Helmut Witten
Landesbildstelle Berlin
BICS
Wikingerufer 7
10555 Berlin

E-Mail: witten@bics.be.schule.de

OStR Irmgard Letzner
Fritz-Karsen-Schule
Onkel-Bräsig-Straße 76-78
12359 Berlin

E-Mail: letzner@math.fu-berlin.de

Prof. Dr. Ralph-Hardo Schulz
Freie Universität Berlin
Fachbereich Mathematik und Informatik
Institut für Mathematik II
Arnimallee 3
14195 Berlin

E-Mail: schulz@math.fu-berlin.de

Literatur

Batzer, P.: Die ENIGMA – Grundlagen zu einer Unterrichtssequenz. In: LOG IN, 16 (1996), H. 5/6, S. 44-51.

Bauer, F. L.: Entzifferte Geheimnisse – Methoden und Maximen der Kryptologie. Berlin u. a.: Springer, 1995.

Bennet, C. H.; Brassard, G.; Ekert, A. K.: Quanten-Kryptographie. In: Spektrum der Wissenschaft, 15 (1992), H. 12, S. 96-104.

Beutelspacher, A.: Kryptologie – Eine Einführung in die Wissenschaft vom Verschlüsseln, Verbergen und Verheimlichen. Braunschweig: Vieweg, ^{3}1993.

Beutelspacher, A.: Geheimsprachen – Geschichte und Techniken. München: Beck, 1997.

Engel, A.: Elementarmathematik vom algorithmischen Standpunkt. Stuttgart: Klett, 1977.

Engel, A.: Mathematisches Experimentieren mit dem PC. Stuttgart: Klett, 1991.

Forster, O.: Algorithmische Zahlentheorie. Braunschweig/Wiesbaden: Vieweg, 1996.

Giessmann, E. G.: Sichere Verschlüsselungen – geht das überhaupt? In alpha, 29 (1995), H. 6, S. 10-11 (Teil I) und alpha, 29 (1995), H. 7, S. 8-10 (Teil II).

Harris, R.: Enigma. München: Heyne-Verlag, 1995.

Hodges, A.: Alan Turing – Enigma. Wien: Springer-Verlag, 1994.

Kippenhahn, R.: Verschlüsselte Botschaften – Geheimschrift, Enigma und Chipkarte. Reinbek bei Hamburg: Rowohlt, 1997.

Kuntze, R.: Wetten am Computer. In: PC Magazin Spezial, 1998, H. 5, S. 30-34.

Lewin, R: Entschied ULTRA den Krieg? Allierte Funkaufklärung im zweiten Weltkrieg. Koblenz/Bonn: Verlag Wehr & Wissen, 1981.

Niedersächsisches Kultusministerium (Hrsg.): Neue Technologien und Allgemeinbildung – Mathematik. Hannover: Berenberg, 1990.

Schneier, B.: Angewandte Kryptographie – Protokolle, Algorithmen und Sourcecode in C. Bonn u. a.: Addison-Wesley, 1996.

Sgarro, A; Würmli, M.: Geheimschriften – Verschlüsseln und Enträtseln von Geheimtexten. Augsburg: Weltbild Verlag, 1991.

Stallings, W: Sicherheit im Datennetz. München u. a.: Prentice Hall, 1995.

Weis, R: Chiffrieren am laufenden Band. In: Kryptographie und Netzwerksicherheit, PC Magazin Spezial, 1998, H. 5, S. 39-41.

Witten, H.; Letzner, I.; Schulz, R.-H.: RSA & Co. in der Schule. Teil 2: Von Cäsar über Vigenère zu Friedman. In: LOG IN, 18 (1998), H. 5, S. 31-39.

Witten, H.: Wege aus dem 8-bit-Chaos – Eine kleine Geschichte binärer Block-Codes (Teil 1). In: LOG IN, 14 (1994), H. 5/6, S. 83-85.

Wobst, R.: Abenteuer Kryptologie – Methoden, Risiken und Nutzen der Datenverschlüsselung. Bonn u. a.: Addison-Wesley, 1997.