ETH EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE ZÜRICH

Berichte über Mathematik und Unterricht
Herausgeber: U. Kirchgraber

Public Key - Kryptographie

RSA-Programm für den TI-92

Florian Blättler und Reto Sutter

1. Vorwort

Im September 1997 hatten wir die Gelegenheit, an einer Mathematik-Sommerakademie der Schweizerischen Studienstiftung unter der Leitung von Prof. Dr. Urs Kirchgraber von der ETH Zürich und Prof. Dr. Hanspeter Kraft von der Uni Basel teilzunehmen. Wir befassten uns mit Kryptologie und entwickelten in diesem Rahmen ein Programm für den TI-92.
Ziel dieses Berichts *ist es, MittelschullehrerInnen und MittelschülerInnen den Zugang zur Kryptologie und zugleich zum Programmieren auf dem TI-92 zu erleichtern.* Der Bericht ist möglichst einfach gehalten, ohne aber wichtige Schritte zu überspringen.

Zusammenfassung. Kryptologie ist die Wissenschaft der geheimen Nachrichtenübertragung. Dieser Bericht beleuchtet die Thematik der öffentlichen Kryptosysteme, insbesondere des RSA-Systems und erklärt dessen Anwendung anhand eines Programmes für den TI-92.

Quellen

Grosse Teile unseres Berichts sind folgendem Werk entnommen:
- Hanspeter Kraft: Kodierungstheorie und Kryptologie, Aufzeichnungen einer Vortragsreihe an der Universität Basel 1989/90

Empfehlenswerte Literatur:
- Ueli Maurer: Informationssicherheit und Kryptologie, Vorlesungsausarbeitung ETH Zürich 1995/96

Dank

Ein grosses Dankeschön geht an Herrn Prof. Dr. Hanspeter Kraft von der Universität Basel. Er hat uns während der Studienwoche betreut und uns in das Gebiet der Kryptographie eingeführt. Ebenso danken wir ganz herzlich Herrn Prof. Dr. Urs Kirchgraber von der ETH Zürich, der uns zu diesem Bericht motiviert hat. Für die Korrektur des Berichtes danken wir Herrn Prof. Dr. Hanspeter Kraft, Herrn Prof. Dr. Jürg Waldvogel und Herrn Dr. Peter Thurnheer.

Was die Schülerinnen und Schüler wissen sollten

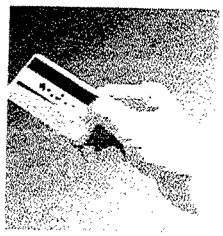
Um unseren Bericht durcharbeiten zu können, braucht es keine Vorkenntnisse in der Kryptologie und nicht unbedingt viele im Programmieren. Wir wollen uns im Bericht nicht auf ein bestimmtes mathematisches Niveau konzentrieren, angesprochen sind vor allem MittelschülerInnen.

Zwar sollte der Bericht in einer Art verfasst sein, die es den SchülerInnen ermöglicht, ihn *selbständig zu lesen und zu verstehen.* Es ist jedoch von Vorteil, wenn für allfällige Fragen eine kompetente Person zur Verfügung steht.

2. Einleitung

2.1. Wozu Kryptologie?

Auch heute bezahlen wir vieles mit Bargeld, wenn wir z. B. in der Bäckerei frische Brötchen oder im Warenhaus ein neues T-Shirt kaufen. Jedoch gebrauchen wir immer öfters die *elektronischen Zahlungsmittel*. Bargeld kann man klauen, eine EC-Karte nützt hingegen einem Dieb ohne den dazugehörigen Code nicht viel, also scheint die elektronische Methode sicherer zu sein.



In einem Zeitalter, in dem schon 17jährige Hacker in Rechner des amerikanischen Militärs eindringen können, muss man sich allerdings fragen, ob dieser elektronische Weg denn wirklich so sicher sei. Kann sich denn nicht irgend jemand in die Leitung zwischen zwei Banken oder zwischen Bankomat und Rechenzentrum schalten und sieht so bequem alles was läuft? - Kann er nicht! Wieso? Die Daten sind auf trickreiche Art verschlüsselt, so dass niemand ausser den befugten Personen sie lesen kann. In dieser Dokumentation versuchen wir, Ihnen einige Grundideen der Kryptologie näherzubringen.

Kryptologie ist die Wissenschaft der geheimen Nachrichtenübertragung.

Teilgebiete davon sind:

- *Kryptographie*: Absichern von Daten in Kommunikationssystemen
- *Kryptoanalysis*: Methoden zur Entschlüsselung verschlüsselter Informationen

Ursprünglich wurde die Kryptographie vor allem für *militärische Zwecke* eingesetzt. Der Gegner soll nicht wissen, was man für Nachrichten übermittelt. Umgekehrt ist es aber hilfreich, verschlüsselte Nachrichten des Gegners zu entziffern oder gar zu fälschen und sie so zu seinem eigenen Vorteil auszunützen.

Ein Übermittlungskanal kann also gestört werden durch:

- *Passive Beeinträchtigung*: Abhören
- *Aktive Beeinträchtigung*: Fälschen

2.2. Kleine Geschichte der Kryptologie

Erfunden wurde die Kryptologie von Leuten, die Spionage betrieben und zwar nicht erst in der Neuzeit. Schon die alten Römer waren froh, ihre Nachrichten verschlüsseln zu können, wenn auch die Methoden ungleich einfacher waren.

Antike: Caesar

Dieses Verfahren soll der römische Feldherr Gaius Julius Caesar erfunden und selbst benutzt haben. Dabei wird jeder Buchstabe durch einen anderen oder durch ein Zeichen ersetzt.

BEISPIEL: Das Alphabet wird um 1 nach rechts geschoben.

AVECAESAR

wird zu

BWFDBFTBS

AUFGABE: Verschlüsseln Sie folgenden Text:

KRYPTOGRAPHIE IST NUETZLICH

wird zu

.....

Diese Methode, die auch Substitutions-Methode genannt wird, ist sehr unsicher. In einem längeren Text kann man mittels Häufigkeitsuntersuchungen Zeichen leicht erkennen. Ein deutscher Text besteht z. B. aus 15% E, 11% N, 8% I, ...

Absolutismus: Rossignol

Der Kryptoanalytiker des französischen Königs Louis XIV, Antoine Rossignol, entschlüsselt einen codierten Hilferuf der belagerten französischen Stadt Hesdin und sendet die Antwort im gleichen Code zurück, worauf Hesdin kapituliert.

1. Weltkrieg: Zimmermann-Telegramm

Zimmermann, der deutsche Aussenminister, offeriert am 17. Januar 1917 den Mexikanern die „verlorenen" Gebiete Texas, New Mexico und Arizona, falls sie gegen die USA in den Krieg ziehen. Die Engländer fangen dieses chiffrierte Telegramm ab und leiten es weiter an den US-Präsidenten Wilson. Dieser veröffentlicht es und trägt so dazu bei, dass die öffentliche Meinung umschlägt und die Amerikaner den Krieg erklären.

2. Weltkrieg: ENIGMA

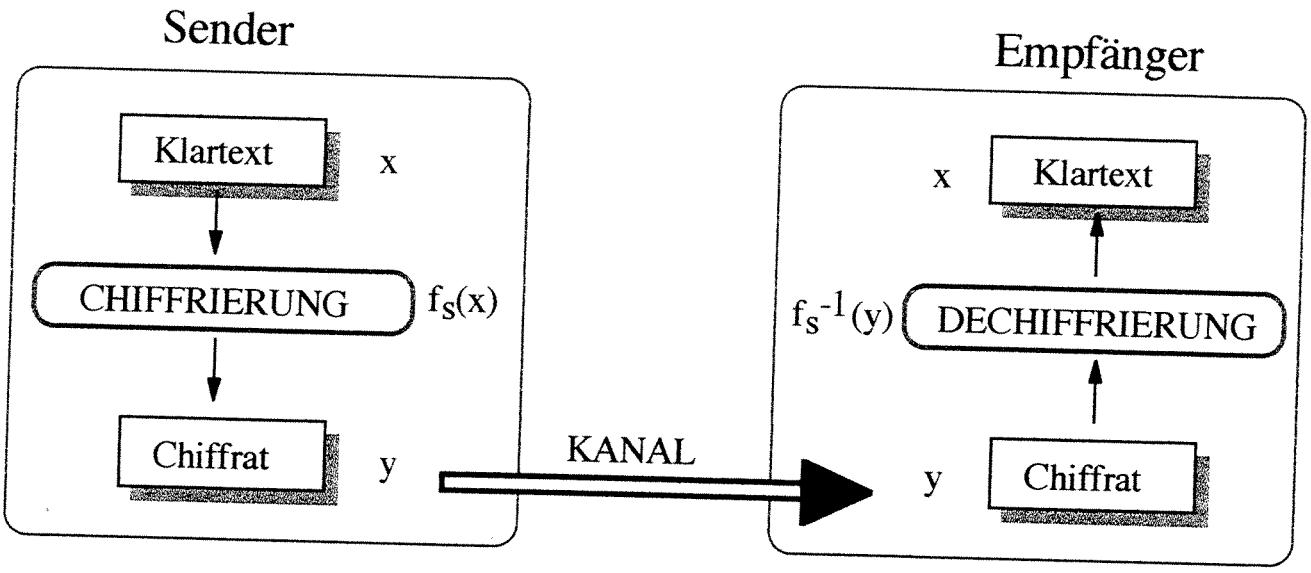
Die deutsche Wehrmacht benutzt die mechanische Rotorchiffriermaschine ENIGMA, um ihre Meldungen aus dem Felde zu verschlüsseln. Dieses System galt als äusserst raffiniert und sicher. Die Briten knackten jedoch den Mechanismus und konnten mit Hilfe der Dechiffriermaschine ULTRA die Nachrichten lesen.

Ende des 20. Jahrhunderts

Bis die Computer aufkamen, waren die Kryptoanalytiker den Kryptographen oft überlegen. In einer Zeit, in der sich die Computertechnologie so rasant entwickelt, sind die Chancen etwas gleichmässiger verteilt. Trotzdem hören wir regelmässig, dass Hacker hier und dort unbemerkt in ein als sicher geltendes Netzwerk gelangen konnten. Der Kampf um die Verschlüsselung geht also unvermindert weiter.

2.3. Wie läuft eine Chiffrierung ab?

Ein Klartext wird mittels einer Chiffrierung verschlüsselt, also in ein Chiffrat verwandelt. Dieses wird dann über den Kanal gesendet und mittels einer Dechiffrierung wieder in den ursprünglichen Klartext zurückübersetzt.



In der Abbildung haben wir bereits die Notation eingeführt. Im ganzen Bericht gelten folgende Bezeichnungen:

x Klartext
y Chiffrat = Verschlüsselter Text

$f_s(x)$ Chiffrierung = Funktion um Klartext zu verschlüsseln (Schlüsselfunktion)
$f_s^{-1}(y)$ Dechiffrierung = Funktion um Chiffrat zu entschlüsseln (Umgekehrte Schlüsselfunktion)
s Schlüssel

2.4. Vigenère

Die Notation, die wir oben kennengelernt haben, wollen wir nun beim einfachen Kryptosystem von Vigenère anwenden. Hier wird zum Klartext eine „Schlüsselfolge" addiert, welche aus einem Schlüsselwort besteht. Dieses Schlüsselwort wiederholt sich immer wieder.

BEISPIEL:

Klartext x	KRYPTOGRAPHIE IST RAFFINIERT
+	
Schlüsselfolge s	WORTWORTWORTWORTWORTWORTWO...
=	
Chiffrat y	GFPIPCWKWDYBAWJMNOWYECZXNH

Verschlüsseln
Um das Chiffrat zu berechnen, werden die Buchstaben von 0 bis 25 durchnumeriert.
A=0, B=1, C=2, D=3, E=4, F=5, G=6, H=7, I=8, J=9, K=10, L=11, M=12, N=13, O=14, P=15, Q=16, R=17, S=18, T=19, U=20, V=21, W=22, X=23, Y=24, Z=25

Nun werden die beiden ersten Buchstaben zueinander addiert und modulo 26 gerechnet*.
$f_s(x) = x + s \bmod 26$

$K = 10, W = 22$

$K + W = 10 + 22 = 32 \bmod 26 = 6 = G$

Nun kann man mit dem ganzen Text so weiterfahren.

* [Falls das Modulo-Rechnen noch nicht bekannt ist, kann man dies im Kapitel 2.5. nachlesen.]

Entschlüsseln

Um das Chiffrat wieder lesen zu können, wird einfach der Buchstabe des Schlüssels subtrahiert und modulo 26 gerechnet.

$f_s^{-1}(y) = y - s \bmod 26$

$G = 6, W = 22$

$G - W = (6 - 22) \bmod 26 = -16 \bmod 26 = 10 = K$

2.5. Modulo - Rechnen

Modulo ist Rechnen mit Resten. Wenn man a und b durch d dividiert und dabei denselben Rest r erhält, so schreibt man a≡b (mod d).

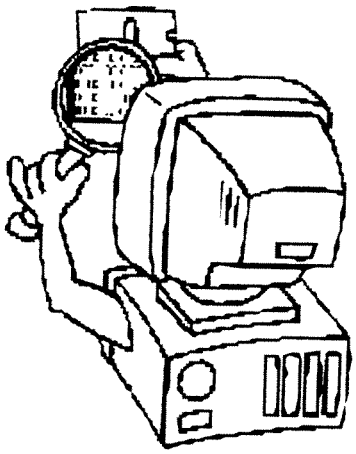
Beispiel: 17≡32 (mod 5)

Aufgabe: Berechnen Sie je eine Lösung für folgende Moduli x, so dass folgende Beziehungen gelten:

a) 32≡38 (mod x) x = ...

b) 67≡23 (mod x) x = ...

2.6. Neuere Verfahren



Ende des 20. Jahrhunderts benutzen wir natürlich nicht mehr Buchstabenfolgen zum Übermitteln von geheimen Botschaften, sondern es werden bit-Folgen, d.h. Folgen von 0 und 1 (z.B. 011010100) übertragen, wie sie in jedem Computer vorkommen. Diese enthalten codiert die Buchstaben und Zahlen.

Das Vigenère-Verfahren, das wir in 2.4. kennengelernt haben, ist mit Häufigkeitsanalysen recht einfach zu entziffern, da sich die Schlüsselfolge immer wiederholt, d.h. periodisch ist.
Im Jahre 1926 hat deshalb G.S. Vernam vorgeschlagen, eine Schlüsselfolge zu wählen, die unendlich lange ist und sich nirgends wiederholt. Wenn nun diese Schlüsselfolge eine "Zufallsfolge" wäre und nur einmal benützt würde, wäre dieses Verfahren absolut sicher. In der Praxis verwendet man dazu *Pseudozufallsgeneratoren*, auf die wir hier nicht näher eingehen. Für einen Pseudozufallsgenerator braucht man jedoch einen weiteren Schlüssel. Damit können Schlüsselfolgen produziert werden, die sich erst nach sehr vielen Stellen wiederholen. Das wesentliche ist, dass die Verfahren symmetrisch sind, die für die Verschlüsselung und die Entschlüsselung denselben Schlüssel benutzen. Völlig sicher sind sie nur, falls jedes Benutzer-Paar einen eigenen, geheimen Schlüssel hat. Dies ist jedoch wegen der hohen Zahl der Benutzer technisch schwer zu bewerkstelligen.

Das Problem ist also heute nicht die Länge der Schlüsselfolge, sondern die *Übermittlung* und *Verwaltung* des Schlüssels.

3. Öffentliche Kryptosysteme

3.1. Unmöglich? Einführung Public Key

Stellen Sie sich vor, 15 Personen sitzen in einem Kreis und alle nehmen wahr, wer mit wem und auf welche Art kommuniziert. Ist es möglich, dass zwei zufällig ausgewählte Testpersonen sich eine Botschaft übermitteln können, die nur sie zwei verstehen können, obwohl alle anderen jegliche Kommunikation zwischen den Testpersonen mitbekommen und vor dem Experiment keine Vorbereitungen und Absprachen getroffen worden sind?
Auf gar keinen Fall, das würden nur Zauberkünstler mit irgendwelchen Tricks zustandebringen - falsch! Mit ein wenig Mathematik ist es möglich, dass unsere zwei Testpersonen kommunizieren können, ohne dass die anderen eine Chance hätten, die Mitteilungen zu entziffern.
Noch erstaunlicher: Jede Person im Kreis kann mit jeder anderen auf dieselbe Weise kommunizieren. Alle sehen und hören alles und trotzdem wissen immer nur die richtigen zwei Leute, was die Nachricht enthält.

Im Kapitel 3.2. werden wir die mathematischen Voraussetzungen für das obenstehende Experiment behandeln. In 3.3. werden wir dann genau auf unser Kreis-Experiment eingehen.
Statt eines Kreises mit 15 Personen könnten wir uns auch unsere vernetzte Computerwelt vorstellen. Natürlich können hier nicht alle Teilnehmer mit jedem anderen Teilnehmer der Welt einen privaten Schlüssel vereinbaren, dies wäre zu kompliziert. Eine Lösung dazu ist der Öffentliche Schlüssel, der Public Key, den wir im Kapitel 3.4. behandeln.

3.2. Einweg- und Falltür-Funktionen

W. Diffie und M.E. Hellman schlugen 1976 vor, Verfahren zu entwickeln, bei denen zwar der Chiffrierschlüssel und das Verfahren bekannt sind, es aber trotzdem nicht möglich ist, daraus einen Algorithmus zum Dechiffrieren herzuleiten.
Dafür braucht man sogenannte *Einweg-Funktionen*. Diese lassen sich leicht berechnen, die Umkehrfunktion (also der Rückweg) kann jedoch nicht in kurzer Zeit bestimmt werden.
Eine *Falltür-Funktion* ist noch raffinierter. Mit einer nur dem Empfänger bekannten zusätzlichen Information ist es möglich, eine Funktion, die eigentlich eine Einweg-Funktion ist, trotzdem umzukehren.

Beispiel einer Einweg-Funktion:

Wir potenzieren (modulo einer grossen Primzahl p) $f(x)=s^x \bmod p$.
Dies erfordert bei einer 100-stelligen Primzahlen einen *Rechenaufwand* von log p Operationen.
Der *Supercomputer NEC SX3* arbeitet mit etwa 100'000 Mips, d.h. 10^{11} Bitoperationen pro Sekunde. Er bräuchte für diese Berechnung (wenn alle Zahlen 100-stellig sind) weniger als 0,000000005 s.

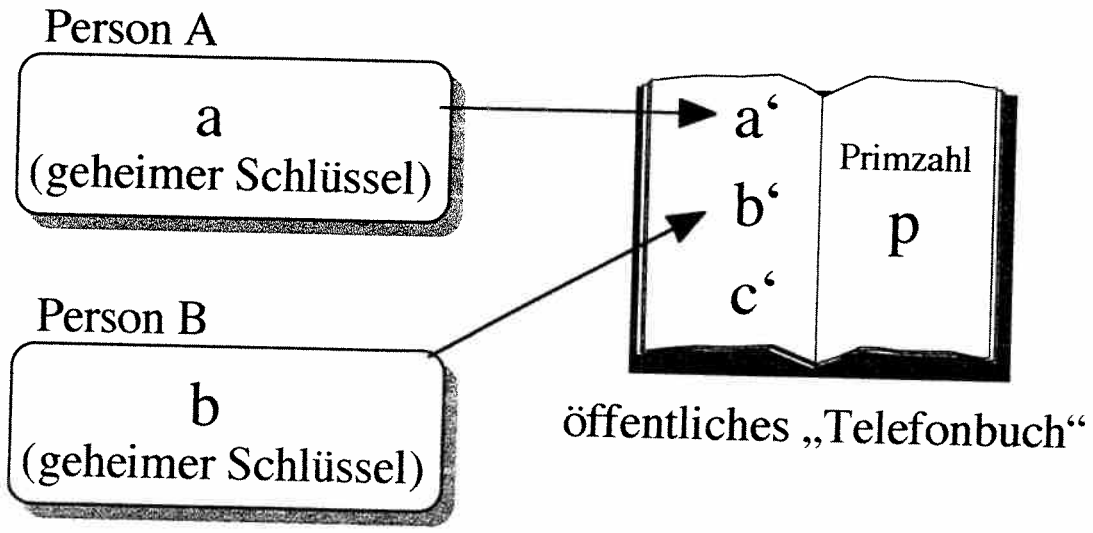
Das Potenzieren geht so schnell, weil man aus vielen Multiplikationen wenige machen kann. Zum Beispiel braucht die Operation 13^{1024} 1024 Multiplikationen.
Die Operation $13^{1024} = (((((((((13^2)^2)^2)^2)^2)^2)^2)^2)^2)^2$ hingegen braucht nur 10 Multiplikationen (Man multipliziert 13 mit 13 und das Resultat noch neunmal mit sich selbst. Dies ergibt ebenfalls 13^{1024})

Die *Umkehrfunktion*, der sog. diskrete Logarithmus, lautet $x = \log_s y \bmod p$. Der beste Algorithmus, der bis jetzt dafür bekannt ist, benötigt bei den oben erwähnten Zahlen einen *Rechenaufwand* von $\sqrt{p}$ Operationen. Der NEC SX3 bräuchte dafür über $3 \cdot 10^{81}$ Jahre (das ist eine 3 mit 81 Nullen!)

Schon bei kleinen Zahlen ist $\log p < \sqrt{p}$, bei Zahlen mit 50 Stellen hingegen sind $\log p$ Operationen kein Problem, $\sqrt{p}$ Operationen jedoch praktisch nicht mehr machbar.
Zur besseren Verdeutlichung ein kleines Zahlenbeispiel, das zeigt, wie gross der Unterschied ist:
$\log 10^{50}=115$, $\sqrt{10^{50}}=10^{25}$

3.3. Schlüsselaustausch

Da Einwegfunktionen auch für Eingeweihte nur in die eine Richtung auf einfache Weise zu berechnen sind, eignen sie sich nicht gut als Kryptosystem. Nützlich sind sie jedoch beim Austausch von Schlüsseln im Kreis-Experiment, wie wir zeigen wollen.
Dafür wird zunächst eine (grosse) Primzahl p öffentlich bekannt gegeben. Dann wählt Teilnehmer A einen geheimen Schlüssel a und teilt der Schlüsselbibliothek den öffentlichen Schlüssel $a' = s^a \bmod p$ mit. Der Teilnehmer B macht dasselbe.



Wenn nun die beiden miteinander kommunizieren wollen, so stellen sie aus ihrem geheimen Schlüssel und dem öffentlichen Schlüssel des anderen einen gemeinsamen Schlüssel her.

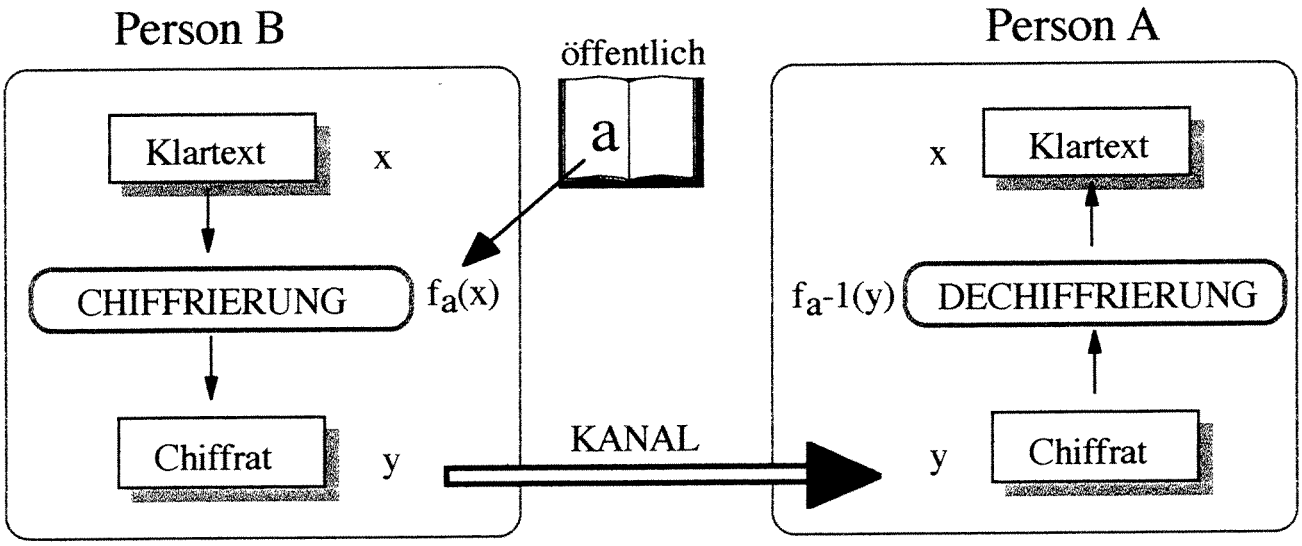
$$s_{A,B}=a'^{b}=s^{ab}=b'^{a} \pmod p$$

Nun haben sie den Schlüssel s^{ab}, den nur die beiden kennen, alle anderen aus unserem Kreis-Experiment jedoch nicht. Sie können ihn jetzt ungestört einsetzen, um miteinander zu kommunizieren.

3.4. Öffentliche Kryptosysteme (Public Key - Kryptosysteme)

Wie bereits erwähnt, ist es sehr mühsam, mit allen Computerbenutzern auf der ganzen Welt einzeln Schlüssel festzulegen. Deshalb gibt es die öffentlichen Kryptosysteme, die mit den Falltür-Funktionen arbeiten.
Das System, welches wir jetzt vorstellen wollen, ist kein symmetrisches (A und B haben den gleichen Schlüssel zum Verschlüsseln und Entschlüsseln) sondern ein asymmetrisches, d.h. man arbeitet mit einem öffentlichen Schlüssel f_a zur Verschlüsselung und einem geheimen Schlüssel f_a^{-1} zur Entschlüsselung.

Der Teilnehmer A wählt einen Schlüssel a aus und gibt ihn öffentlich (etwa im Telefonbuch) bekannt. Will nun B eine Meldung x an A senden, verschlüsselt er ihn mittels des Chiffrierverfahrens f_a (das Verfahren ist allgemein bekannt und wird mit dem öffentlichen Schlüssel a angewendet). B berechnet also das Chiffrat $y=f_a(x)$, welches er über jeden öffentlichen Kanal (z.B. das Internet) schicken kann, ohne dass jemand es lesen kann.
A benützt nun den Dechiffriermechanismus $x=f_a^{-1}(y)$, welchen nur er kennt, um die Botschaft zu entschlüsseln.



Die Realisierung eines solchen Verfahrens benutzt Falltürfunktionen, wie wir es im RSA - Kryptosytem zeigen.

4. Das RSA-Kryptosystem

4.1. Geschichte

Das RSA-Kryptosystem ist das älteste öffentliche Kryptosystem; es wurde 1978 von R. L. ***R***ivest, A. ***S***hamir und L. M. ***A***dleman vorgeschlagen. Das Verfahren beruht auf einem Satz der elementaren Zahlentheorie und der Tatsache, dass das Faktorisieren von grossen Zahlen sehr schwierig und mit grossem Zeitaufwand verbunden ist. Es ist heute sehr verbreitet, und man kann es in integrierter Form als RSA-Chip kaufen.

4.2. Die RSA-Methode

Wie bei dem bereits vorgestellten Kryptosystemen von Diffie und Hellman bildet das Potenzieren modulo einer grossen Zahl auch hier wieder die Basis der Verschlüsselung. Das ganze System ist jedoch komplexer, da nur der Schlüssel des Empfängers (A) genutzt wird.
Dieser gibt einen öffentlichen Schlüssel $S_A = (n,d)$ bekannt, währenddem er noch seinen geheimen Schlüssel (e) besitzt. Will ihm nun jemand eine Nachricht x (in Form einer Zahl) schicken, so verschlüsselt er diese folgendermassen:

$$\mathbf{y = f_A(x) = x^d \bmod n}$$

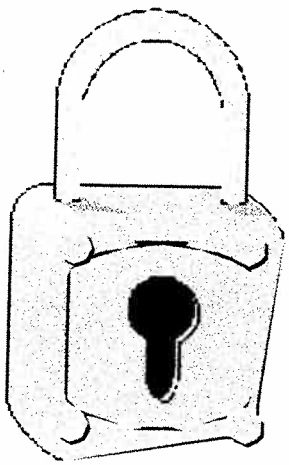
Den Empfänger erreicht die verschlüsselte Nachricht y. Nun ist e so konstruiert, dass man damit leicht entschlüsslen kann:

$$\mathbf{x = f_A^{-1}(y) = y^e \bmod n}$$

Bevor wir nun zur mathematischen Erklärung des Systems kommen, möchten wir zuerst den Schlüssel erstellen.

4.3. Die Schlüsselsuche

Der Benutzer (A) wählt zwei grosse Primzahlen p und q (etwa 100 Stellen) und berechnet $\mathbf{n = pq}$ und $\mathbf{\varphi(n) = (p - 1)(q - 1)}$. Weiter wählt er eine beliebige Zahl d ($< \varphi(n)$),welche zu $\varphi(n)$ teilerfremd ist und bestimmt e mit $\mathbf{de \equiv 1 \bmod \varphi(n)}$. **Nun gibt A öffentlich den Schlüssel $S_A = (n,d)$ bekannt.** Die Zahl e ist sein geheimer Schlüssel.



4.4. Mathematische Erklärungen

Ein wichtiger Bestandteil ist die Eulerfunktion $\varphi(n)$ (n sei eine positive ganze Zahl). $\varphi(n)$ ist die Anzahl der positiven ganzen Zahlen $m < n$, welche zu n teilerfremd sind:

$$\varphi(n) = \#\{m \mid 0 < m < n, \; ggT(n,m) = 1\}.$$

Beispiele: Ist $n = p$ prim, so ist $\varphi(p) = p-1$.

Ist $n = pq$ mit p, q prim und verschieden, so ist $\varphi(n) = (p - 1)(q - 1)$.

Lemma. Sei n ein Produkt von verschiedenen Primzahlen. Ist $s \equiv 1 \bmod \varphi(n)$ so gilt

$$a^s \equiv a \bmod n$$

für jede ganze Zahl a.

Dieses Lemma beruht auf einem Satz von Euler, welcher den bekannten "Kleinen Fermat" verallgemeinert.

Kommen wir nun zurück zu unserer Verschlüsselungsmethode. Der Absender erstellt das Chiffrat y der Nachricht x mit der Funktion

$$y = f_A(x) = x^d \bmod n.$$

Der Empfänger erhält das Chiffrat y und kann es mit Hilfe der Funktion

$$f_A^{-1}(y) = y^e \bmod n$$

wieder lesbar machen, denn es gilt:

$$f_A^{-1}(y) = f_A^{-1}(f_A(x)) = f_A^{-1}(x^d \bmod n) = (x^d \bmod n)^e \bmod n = x^{de} \bmod n$$

Nun wurde e so gewählt, dass $de \equiv 1 \bmod \varphi(n)$. Mit Hilfe des Lemmas sehen wir, dass somit

$$x^{de} \equiv x \bmod n$$

ist, womit wir wieder die ursprüngliche Nachricht x erhalten.

4.5. Die Sicherheit des RSA

Die einzige heute bekannte Methode zur Berechnung der Umkehrfunktion f_A^{-1}, besteht in der Bestimmung von e, wozu man wiederum $\varphi(n)$ benötigt. Wegen $\varphi(n) = n + 1 - (p + q)$ ist es leicht, aus $\varphi(n)$ (und n) die beiden Primzahlen p und q zu berechnen und umgekehrt. Dies bedeutet:

Die einzige heute bekannte Methode zum Brechen des RSA ist die Zerlegung der Zahl n in die beiden Primzahlen p und q.

Der Rechenaufwand zur Faktorisierung grosser Zahlen (n-stellig) liegt mit dem heute schnellsten bekannten Verfahren bei

$$e^{(1+\varepsilon)\sqrt[3]{\log n \log\log n}}$$

Rechenoperationen, wobei ε nach 0 geht für grosses n. Bei 50-stelligen Zahlen sind dies etwa 10^{11} Bitoperationen, bei 100-stelligen etwa 10^{16} und bei 200-stelligen etwa 10^{24}. (Der im Kapitel 3.2. erwähnte NEC SX3 hätte also etwa 300'000 Jahre)

5. RSA-Programm auf dem TI-92

5.1. Einleitung

Dieser Bericht entstand aus der Sommerakademie "Mathematik zu zweit" der Schweizerischen Studienstiftung im Herbst 1997. Für unsern abschliessenden Vortrag über die Kryptographie beschlossen wir, ein kleines Demonstationsprogramm auf dem TI - 92 zu entwickeln. Herr Prof. Dr. U. Kirchgraber animierte uns schliesslich, diese Programme in einem Bericht vorzustellen. Vorher möchten wir noch erwähnen, dass die Programme alles andere als perfekt sind (sie sind im Laufe eines Nachmittags und einer Nacht entstanden) und dass klar zu erkennen ist, dass der Programmierer vom Basic her kommt. Nachfolgend abgedruckt und erläutert sind je ein Programm zur Chiffrierung und Dechiffrierung sowie ein Programm zur Schlüsselsuche mit einem Unterprogramm. (Die Zeilennummerierung gehört nicht zu den Programmen.)

5.2. Chiffrierung

```
1   ( )
2   Prgm
3   Dialog
4   Text "Bitte zu verschluesselnde Nachricht x (x<√n )"
5   Request "eingeben",x
6   EndDlog
7   Request "Bitte n eingeben",n
8   Request "Bitte d eingeben",r
9   expr(x)→x
10  expr(n)→n
```

```
11  expr(r)→r
12  0→t
13  1→y
14
15  Loop
16  If 2^(t+1)>r Then
17  Goto fak
18  EndIf
19  t+1→t
20  EndLoop
21
22  Lbl fak
23  For b,t,1,-1
24  If 2^b<r Then
25  mod(mod(y^2,n)*mod(x^2,n),n) →y
26  r-2^b→r
27  Else
28  mod(y^2,n) →y
29  EndIf
30  EndFor
31
32  If r=1 Then
33  mod(y*x,n) →y
34  EndIf
35  Dialog
36  Text "Die zu uebermittelnde Botschaft y ist"
37  Text string(y)
38  EndDlog
39
40  EndPrgm
```

Erläuterungen:

Zeilen 3-8: Eingabe des Klartextes (Zahl, die kleiner ist als die kleinere Primzahl) und des öffentlichen Schlüssels
Zeilen 9-11: Umwandlung der Daten von Ziffern zu Zahlen
Zeilen 12-13: Zurücksetzen von Variablen
Zeilen 15-20: Berechnung der Anzahl Binärstellen t von d
Sobald 2^{t+1} grösser als d ist, wird die Schlaufe verlassen
Zeilen 22-30: Wie schon früher erklärt, wird das Potenzieren 2-adisch zerlegt. Die Modulo-funktion wird nach jedem Rechenschritt benutzt, um immer mit der kleinstmöglichen Zahl zu rechnen.
Zeilen 32-34: Falls die letzte Ziffer von d (binär) eine 1 ist, muss nach dem letzten quadrieren noch mit x multiplitziert werden.
Zeilen 35-38: Ausgabe des Chiffrates

5.3. Dechiffrierung

```
1   ( )
2   Prgm
3   Dialog
4   Text "Bitte uebermittelte Zahl y"
5   Request "eingeben",y
6   EndDlog
7   Request "Bitte n eingeben",n
8   Request "Bitte e eingeben",s
9   expr(y) →y
10  expr(n) →n
11  expr(s) →s
12  0→t
13  1→x
14
15  Loop
16  If 2^ (t+1)>s Then
17  Goto fak
18  EndIf
19  t+1→t
20  EndLoop
21
22  Lbl fak
23  For b,t,1,-1
24  If 2^b<s Then
25  mod(mod(x^2,n)*mod(y^2,n),n) →x
26  s-2^b→s
27  Else
28  mod(x^2,n) →x
29  EndIf
30  EndFor
31
32  If s=1 Then
33  mod(x*y,n) →x
34  EndIf
35  Dialog
36  Text "Die Meldung x heisst"
37  Text string(x)
38  EndDlog
39
40  EndPrgm
```

Erläuterungen:

Das Dechiffrieren läuft beim RSA mit der gleichen Funktion ab, wie das Chiffrieren, mit Ausnahme der Variablen. x und y werden vertauscht und anstatt einem d jeweils ein e.

5.4. Schlüsselsuche

5.4.1. Hauptprogramm

```
1   ( )
2   Prgm
3   Request "Primzahl 1 eingeben (p<3100)",p
4   Request "Primzahl 2 eingeben (q<3100)",q
5   expr(p) →p
6   expr(q) →q
7
8   p*q→n
9   (p-1)*(q-1) →Φ
10
11  Lbl test
12  Dialog
13  Text "Bitte Zahl d,"
14  Text "teilerfremd und kleiner zu/als"
15  Text string(Φ)
16  Request "eingeben",r
17  EndDlog
18  expr(r) →r
19  If gcd(r,Φ)≠1 or r≥Φ Then
20  Goto test
21  EndIf
22
23  euclid(Φ,r)
24
25  Dialog
26  Text "Ihre Geheimzahl e betraegt"
27  Text string(s)
28  Text "Der oeffentliche Schluessel n,d"
29  Text "betraegt"
30  Text string(n)
31  Text string(r)
32  EndDlog
33
34  EndPrgm
```

Zeilen 3-4: Eingabe der beiden zuvor ermittelten Primzahlen (diese sollten kleiner als 3100, da sonst beim TI-92 Ungenauigkeiten auftreten können)
Zeilen 5-6: Umwandlung der Daten von Ziffern in Zahlen
Zeilen 8-9: Berechnung von n und $\varphi(n)$
Zeilen 12-18: Eingabe von d und Umwandlung in eine Zahl
Zeilen 19-21: Prüfung, ob d kleiner als $\varphi(n)$ ist und teilerfremd
Zeile 23: Aufruf des Unterprogrammes zur Berechnung von e
Zeilen 25-32: Ausgabe des öffentlichen Schlüssels und der Geheimzahl

5.4.2. Unterprogramm zur Berechnung von e (euclid)

Das Problem, welches dieses Unterprogramm lösen sollte, ist die Bestimmung von e, mit Hilfe der Gleichung

$$\mathbf{de \equiv 1 \bmod \varphi(n).}$$

Dafür wollen wir sie zuerst einmal umformen zu

$$\mathbf{1 = de - x\varphi(n).}$$

Mit Hilfe des euklidschen Algorithmus lassen sich e und x nun sehr einfach berechnen. Anhand eines Beispieles wollen wir den Algorithmus erklären.

$\varphi(n) = 231$ und $d = 131$ seien gegeben.

$231 = 1 * 131 + 100$
$131 = 1 * 100 + 31$
$100 = 3 * 31 + 7$
$31 = 4 * 7 + 3$
$7 = 2 * 3 + 1$

Soweit die Zerlegung, nun müssen wir das ganze wieder zusammenfügen.

$1 = 7 - 2 * 3$
$= 7 - 2 * (31 - 4 * 7)$
$= 9 * 7 - 2 * 31$
$= 9 * (100 - 3 * 31) - 2 * 31$
$= 9 * 100 - 29 * 31$
$= 9 * 100 - 29 * (131 - 100)$
$= 38 * 100 - 29 * 131$
$= 38 * (231 - 131) - 29 * 131$
$= 38 * 231 - 67 * 131$
$= -y * \varphi(n) + e * d$

Nun fügen wir die zweitletzte Gleichung ein, vereinfachen, fügen die drittletzte ein, ...

Das erhaltene Ergebnis $e = -67$ muss natürlich immer modulo 231 betrachtet werden. Deshalb ist **e = 131** und wir erhalten

$$\mathbf{131 * 164 \equiv 1 \bmod 231}$$

Wenn wir den Algorithmus nun verallgemeinert anschauen, so sehen wir:

$\varphi(n) = f_1 * d + r_1$
$d = f_2 * r_1 + r_2$

$r_1 = f_3 * r_2 + r_3$
$r_3 = ...$
$r_{n-2} = f_n * r_{n-1} + r_n$
$r_{n-1} = f_{n+1} * r_n + r_{n+1}$
$r_n = f_{n+2} * r_{n+1} + 1$

Bei der Auflösung entdecken wir dann schlussendlich eine Wechselwirkung zwischen x und e:

(Von den linksstehenden zu den rechtsstehenden Termen werden die oben genannten Gleichungen eingesetzt;
von den rechtsstehenden zu den linksstehenden Termen wird umgeformt.)

$1 = r_n - r_{n+1} * f_{n+2}$

$= r_n * (1 + f_{n+1} * f_{n+2}) - r_{n-1} * f_{n+2}$

$= r_n - (r_{n-1} - f_{n+1} * r_n) * f_{n+2}$

$= r_{n-2} * (1 + f_{n+1} * f_{n+2}) - r_{n-1} * (f_{n+2} + f_n * (1 + f_{n+1} * f_{n+2}))$

$= (r_{n-2} - f_n * r_{n-1}) * (1 + f_{n+1} * f_{n+2}) - r_{n-1} * f_{n+2}$

$= r_{n-2} * (1 + f_{n+1} * f_{n+2}) - (r_{n-3} - f_{n-1} * r_{n-2}) * (f_{n+2} + f_n * (1 + f_{n+1} * f_{n+2}))$

$= r_{n-2} * \{ (1 + f_{n+1} * f_{n+2}) + f_{n-1} * (f_{n+2} + f_n * [1 + f_{n+1} * f_{n+2}]) \} - r_{n-3} * (f_{n+2} + f_n * [1 + f_{n+1} * f_{n+2}])$

Dieser Algorithmus entwickelt sich in die Richtung, dass $r_0 = d$ und $r_{-1} = \varphi(n)$ und aus den Vorfaktor (Klammerinhalten) entstehen x und e (da $1 = d * e - \varphi(n) * x$). Verfolgen wir nun die Vorfaktoren, so sehen wir, dass der neue Vorfaktor jeweils der alte plus der Faktor (f) aus der Zerlegung multipliziert mit dem andern Vorfaktor[1)] ist, währenddem der andere unverändert bleibt und im nächsten Schritt dann gleich transformiert wird.

[1)] Da die Summanden entgegengesetzte Vorzeichen haben, muss im Programm jeweils mit dem negativen Vorfaktor multipliziert werden

```
1   (Φ,r)
2   Prgm
3   1→c
4   Φ→famo[1]
5   r→famo[2]
6   1→fact[1]
7
8   Loop
9   mod(famo[c],famo[c+1]) →famo[c+2]
10  (famo[c]-famo[c+2])/famo[c+1] →fact[c+1]
11  If famo[c+2]=1 Then
```

```
12  Goto zus
13  EndIf
14  c+1→c
15  EndLoop
16
17  Lbl zus
18  1→vf1
19  -fact[c+1]→vf2
20  For I,c,1,-2
21  -vf2*fact[I]+vf1→vf1
22  If I≤2 Then
23  Goto end
24  EndIf
25  -vf1*fact[I-1]+vf2→vf2
26  EndFor
27
28  Lbl end
29  If I=1 Then
30  vf2+Φ→s
31  Else
32  vf1→s
33  EndIf
34
35  EndPrgm
```

Erläuterungen:

Zeile 1: Entgegennahme von $\varphi(n)$ und d
Zeilen 3-6: Zurücksetzen und Zuweisen von Variablen
Zeilen 8-15: Zerlegung
Zeile 9: Berechnung des Restes
Zeile 10: Berechnung des Faktors
Zeilen 17-26: Auflösung des Algorithmus
Zeilen 18-19: Der erste Vorfaktor beträgt 1, der zweite gleichviel wie der letzte Faktor
Zeilen 20-26: Berechnug der Vorfaktoren mit Hilfe ihrer Wechselwirkung
Zeilen 28-35: Ausgabe von e an das übergeordnete Programm

5.5. Variablenliste

Da der TI - 92 nicht alle Zeichen als Variablen erlaubt (z.B. d, e..), konnten wir nicht immer die gleichen verwenden wie in unserem Bericht. Deshalb hier eine Übersicht über ale Variablen:

Variable im Programm	*Variable im Bericht*
x	x
n	n
r	d
t	Zählvariable
y	y
b	Zählvariable
s	e
p	p
q	q
Φ	φ(n)
c	Zählvariable
famo[...]	"Rest"[Dimensionierung]
fact[...]	Faktor[Dimensionierung]
vf1 / vf2	Vorfaktoren
l	Zählvariable

5.6. Liste der von uns verwendeten Befehle

Befehl	*Bedeutung*
Prgm / EndPrgm	Start und Ende eines Programmes
Dialog / EndDlog	Was dazwischen steht, wird in **einer** Dialogbox ausgegeben
Text "a"	Gibt einen Text a aus
Request "a",b	Erwartet auf die Frage a eine Antwort b
Expr(a) →a	Wandelt Ziffern in eine Zahl um
Loop / EndLoop	Was dazwischen steht wird unendlich wiederholt
If [Behauptung]/ Then / Else / Endif	Wenn die Behauptung stimmt, dann (Then) macht der TI - 92, was dannach steht bis zum Else oder Endif, falls die Behauptung jedoch nicht stimmt, wird das gemacht, was nach Else steht
Goto a / Lbl a	Er geht vom Goto zum Lbl
For a,b,c,d / EndFor	Für a zählt er von b nach c in der Schrittweite d
mod(a,b)	a mod b
String(a)	Wandelt eine Zahl a in Ziffern um

gcd(a,b)	greatest common divisor of a and b (gcd) (grösster gemeinsamer Teiler von a und b (ggT))

6. Lösungen

Lösung zur Aufgabe 2.2. LSZQUPHSBQIJF JTU OVFUAMJDI

Lösung zur Aufgabe 2.5. a) x = 1,2,3,6 b) x = 1,2,4,11,22,44

Inhaltsverzeichnis

Prof. U. Kirchgraber
Eidg. Technische Hochschule Zürich
Mathematik
8092 Zürich

Berichte über Mathematik und Unterricht

89–01	H. Walser	Fraktale
89–02	H.R. Schneebeli	Zwei Fallstudien zur Geometrie
89–03	W. Büchi	Astronomie im Mathematikunterricht
89–04	M. Adelmeyer	Theorem von Sarkovskii
90–01	U. Kirchgraber	Von Mathematik und Mathematikunterricht
90–02	A. Kirsch	Das Paradoxon von Hausdorff, Banach und Tarski: Kann man es "verstehen"?
91–01	A. Barth	Formalisierung und künstliche Intelligenz – eine mögliche Behandlung in der Schule
91–02	U. Kirchgraber	Smale's Beweis des Fundamentalsatzes
91–03	M. Federer	Preistheorie
91–04	M. Gauglhofer	Zur Theorie der sozialen Entscheidungen: Das Arrow-Paradoxon bei Abstimmungen über mehrere Alternativen
92–01	U. Kirchgraber	Chaotisches Verhalten in einfachen Systemen
93–01	M. Huber, U. Manz, H. Walser	Annäherung an den Goldenen Schnitt
93–01(I)	M. Huber, U. Manz, H. Walser	Approccio alla Sezione Aurea
93–02	P. Gallin, H. Keller, H. Stocker	Perspektive und Axonometrie
93–02(I)	P. Gallin, H. Keller, H. Stocker	Prospettiva e Assonometria
93–03	H.R. Schneebeli, N. Sigrist, F. Spirig	Verzweigungsphänomene
93–03(I)	H.R. Schneebeli, N. Sigrist, F. Spirig	Fenomeni di Biforcazione
93–04	H. Biner, H.P. Dreyer, W. Hartmann, A. Moretti	Der Fallschirmspringer
93–04(I)	H. Biner, H.P. Dreyer, W. Hartmann, A. Moretti	Il Paracadutista
93–05	H.R. Schneebeli	Alles fliesst – Mit dem Graphikrechner zu den Quellen der Analysis
93–06	H. Biner	Kongruenzabbildungen und Symmetrien im Euklidischen Raum
93–07	U. Kirchgraber	Hundert Jahre Störungstheorie – Differentialgleichungen von Poincaré bis Nekhoroshev
94–01	U. Maurer	Kryptologie: Mathematik zwischen Anwendung und Ästhetik
94–02	H. Klemenz	Computergestützte Raumgeometrie
94–03	F. Barth	Erstens kommt es anders und zweitens als man denkt - Paradoxien im Umfeld der bedingten Wahrscheinlichkeit
94–04	W. Henn	Auto und Verkehr – Beispiele aus der Analysis zum realitätsnahen Mathematikunterricht
95–01	N. Sigrist	Auf der Kippe

95–02	U.Kirchgraber U. Kirchgraber, N. Sigrist	Als Poincaré, Hadamard und Perron die Invarianten Mannigfaltigkeiten entdeckten. Feigenbaum-Universalität: Beschreibung und Beweisskizze
95–03	A. Gächter	Infinitesimalgeometrie - am Beispiel der Kreisevolvente
95–04	P. Gallin	Grund- und Aufrissmethode in der Wahrscheinlichkeitsrechnung
95–05	P. Bolli	The unreasonable effectiveness of mathematics
95–06	G. Schierscher	Verfolgungsprobleme
96–01	W. Burgherr	Schwimmende Prismen mit Schlagseite
96–02	M. Struwe	Sattelpunkte oder Variationsprinzipien in Geometrie und Mechanik
96–03	M. Huber	Warum denn ist $\exp(x^2)$ nicht elementar integrierbar?
97–01	B. Eicke, E. Holzherr	Analysis – mit dem Computer-Algebra-System des TI-92 (Preis: Fr. 8.- inkl. MWSt)
97–02	C. Blatter	Notizen zu einer Mathematik fürs Leben
97–03	F. Spirig	Elemente zur Kugelgeometrie
98–01	U. Kortenkamp, J. Richter-Gebert	Geometry and Education in the Internet Age
98–02	M. Adelmeyer	KS Flight Simulator
98–03	H. Struve	Geometrie aus Schülersicht: Charakteristika und Probleme
98–04	M. Ludwig	Platonische Durchdringungen – ein Projekt im Mathematikunterricht der Klasse 10 (Preis: Fr. 5.- inkl. MWSt)
98–05	M. Adelmeyer,W. Durandi, M. Kopp	Schullotto – ein Projekt im Mathematikunterricht (Preis: Fr. 4.- inkl. MWSt)
98–06	J.-P. David, P. Hänsli, M. Leupp	Parabeln – ein Projekt im Mathematikunterricht (Preis: Fr. 4.- inkl. MWSt)
98–07	B. Dzung Wong, H.-W. Henn	Der Regenbogen – ein Projekt im Mathematikunterricht (Preis: Fr. 4.- inkl. MWSt)
98–08	M. Bettinaglio, F. Lehmann	Mathematisches Billard – ein Projekt im Mathematikunterricht (Preis: Fr. 4.- inkl. MWSt)
99–01	F. Blättler, R. Sutter	Public Key – Kryptographie RSA-programm für den TI-92