Kryptologie am Voyage 200

Sinn der Verschlüsselung ist es, einen Text (Klartext) so zu verändern, dass nur ein autorisierter Empfänger in der Lage ist, den Klartext zu rekonstruieren.
Grundsätzlich unterscheidet man zwischen symmetrischen und asymmetrischen Verfahren zur Verschlüsselung.
Bei symmetrischen Verfahren wird jedes Zeichen oder jede Zeichenkette eines Textes mit Hilfe eines Schlüssels nach einer bestimmten Vorschrift umgewandelt. Der Empfänger benötigt zum Entschlüsseln den gleichen Schlüssel, wie er bei der Verschlüsselung verwendet wurde.
Bei asymmetrischen Verfahren erzeugt der Empfänger zwei Schlüssel, von denen er einen veröffentlicht (Public-Key) und den anderen geheim hält (Private-Key). Der Sender der Nachricht verschlüsselt nun den Text mit Hilfe des Public-Keys und nur der Besitzer des Private-Keys kann die Nachricht entschlüsseln.

1. Symmetrische Verfahren

Die symmetrischen Verfahren werden in zwei Vorgehensweisen, Methoden eingeteilt: Transposition und Substitution.
Bei der Transposition wird die Reihenfolge der Klartext-Zeichen; bei der Substitution werden die Zeichen an sich verändert.

Transposition

Ein einfaches Transpositionsverfahren ist die Spaltentransposition. Der Klartext wird in eine Matrix mit n Spalten und m Zeilen eingeschrieben (Die Anzahl der Zeilen ergibt sich aus der Länge des Textes.), dann die Matrix transponiert und schließlich die neue Nachricht wieder zeilenweise ausgelesen. Der Schlüssel ist dabei die Anzahl der Spalten n.
Beispiel: Der Text "Dieser Text ist geheim" (Länge des Textes: 22) wird bei einer zuvor festgelegten Spaltenzahl 4 (der Schlüssel) wie folgt verschlüsselt:

$$M = \begin{pmatrix} D & i & e & s \\ e & r & & T \\ e & x & t & \\ i & s & t & \\ g & e & h & e \\ i & m & X & X \end{pmatrix} \Rightarrow M^T = \begin{pmatrix} D & e & e & i & g & i \\ i & r & x & s & e & m \\ e & & t & t & h & X \\ s & T & & & e & X \end{pmatrix}$$

Nun muss der Text zeilenweise ausgelesen und in eine Zeile geschrieben werden:
„Deeigiirxseme tthXsT eX“

(Programm: transp()).

Bei der Programmierung ist wichtig, dass die Matrix vollständig aufgefüllt wird. Der verschlüsselte Text wird auf der Variable "out" abgespeichert. (man erspart sich dadurch die neuerliche Eingabe des verschlüsselten Textes!)

Entschlüsseln: (Programm: enttrans(out))

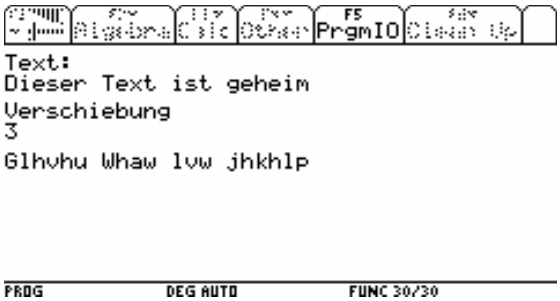
Der Empfänger kennt die Spaltenzahl (den Schlüssel) der Matrix M. Nun teilt er die Gesamtlänge des Geheimtextes (24) durch diese Zahl und erhält die Zahl der Spalten (6) der Matrix, die zu transponieren ist. Aus dieser transponierten Matrix kann er dann den ursprünglichen Text zeilenweise auslesen.

Cäsar – Code (Substitution)

Bei diesem Verschlüsselungsverfahren werden die Buchstaben des Klartextes um eine bestimmte Anzahl (m) von Stellen nach rechts verschoben; der Schlüssel lautet somit m.

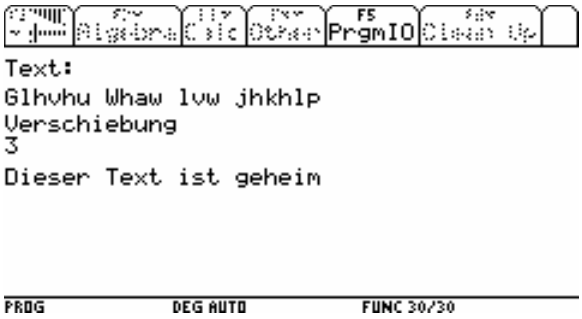
Beispiel (m = 3)

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	Klartextalphabet
x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	Geheimtextalphabet



(Programm: vercaes())

Beim Entschlüsseln muss der Text um die entsprechende Anzahl (Schlüssel m) nach links geschoben werden:



(Programm: entcaes(out))

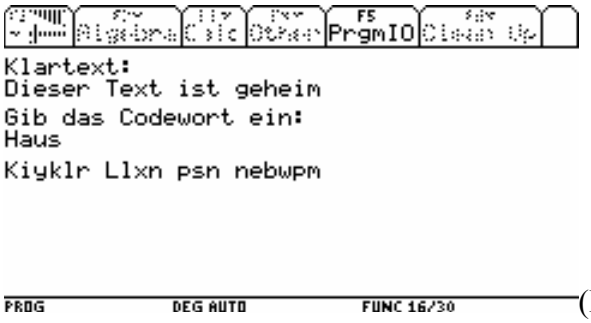
Vignere – Code (Substitution)

Beim Vignere – Code wird mit Hilfe eines Schlüsselwortes verschlüsselt. Dieses Schlüsselwort wird so oft hintereinander geschrieben, bis es die Länge des Textes hat, der verschlüsselt werden soll. Der entsprechende Buchstabe des Schlüsselwortes gibt nun an, wie weit der Buchstabe des Klartextes verschoben werden soll.

Beispiel: Schlüsselwort: Haus

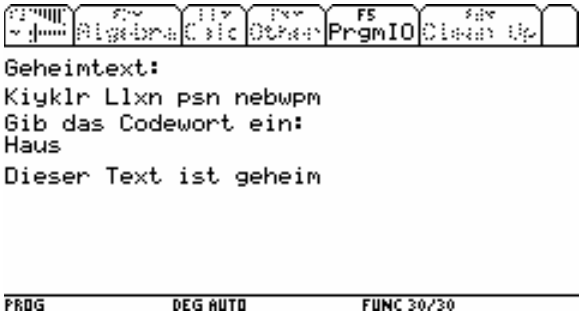
D	i	e	s	e	r		T	e	x	t		i	s	t		g	e	h	e	i	m
H	a	u	s	H	a	u	s	H	a	u	s	H	a	u	s	H	a	u	s	H	a
K	i	y	k	l	r		L	l	x	n		p	s	n		n	e	b	w	p	m

Klartext
Schlüsselwort
Geheimtext



(Programm: vign())

Beim Entschlüsseln wird der Geheimtext um den entsprechenden Buchstaben des Schlüsselwortes nach links verschoben.



(Programm : entvign(out))

Dieses Programm wurde so geschrieben, dass Groß- bzw. Kleinbuchstaben im Schlüsselwort keine Rolle spielen.

2. Asymmetrische Verfahern – RSA-Code

Das Manko der symmetrischen Verfahren liegt darin, dass der benötigte Schlüssel geheim bleiben muss, aber der berechtigte Empfänger der Nachricht ihn dennoch haben muss. Dies erfordert einen geheimen Transport des Schlüssels. Ferner ist eine geheime Kommunikation mit jedem einzelnen einer größeren Anzahl von Teilnehmern dadurch erschwert, dass eine Vielzahl von Schlüsseln erforderlich wäre. Bei drei Teilnehmern wären es noch (3*2)/2=3 Schlüssel (A-B, A-C, B-C); bei 10 Teilnehmern aber sind dies schon (10*9)/2=45 Schlüssel. Diese Probleme umgehen asymmetrische Verfahren, auch Public-Key-Verfahren genannt. Man kann die Einzelschritte zur Durchführung des RSA-Verfahrens folgendermaßen beschreiben. Schritt 1 bis 3 sind die Schlüsselerzeugung, Schritt 4 und 5 sind die Verschlüsselung, 6 und 7 die Entschlüsselung.

(Der allgemeinen Konvention folgend, werden im Folgenden die Kommunikationspartner einer verschlüsselten Konversation als Alice und Bob bezeichnet. Alice stellt den öffentlichen Schlüssel zur Verfügung und entschlüsselt mit ihrem privaten Schlüssel die Nachricht von Bob.)

1. Alice wählt zufällig 2 verschiedene Primzahlen p und q und berechne n = p : q. Der Wert n wird als RSA-Modul bezeichnet.
2. Alice sucht eine Zahl $d \in \{2, \cdots, n-1\}$, so dass gilt:
   d ist teilerfremd zu J(n) = (p − 1) . (q − 1) (Eulersche Funktion). Zum Beispiel kann man d so wählen, dass gilt: max(p, q) < d < J(n) − 1. Danach kann sie p und q ”wegwerfen”.
3. Alice bestimmt eine Zahl $e \in \{1, \cdots, n-1\}$ mit $e \cdot d \equiv 1 \bmod J(n)$, d.h. d ist die multiplikative Inverse zu d modulo J(n). Danach kann man J(n) ”wegwerfen”.
   → (n, e) ist der öffentliche Schlüssel P.
   → (n, d) ist der geheime Schlüssel S (Alice muss nur d geheim halten).
4. Zum Verschlüsseln bricht Bob die als (binäre) Zahl dargestellte Nachricht in Teile auf, so dass jede Teilzahl kleiner als n ist. (Auf diesen Schritt habe ich bei der Programmierung verzichtet.)
5. Bob verschlüsselt den Klartext (bzw. seine Teilstücke) $m \in \{1, \cdots, n-1\}$:
   $c = E((n, e);M) := m^e \bmod n$.
6. Alice bricht zum Entschlüsseln das binär als Zahl dargestellte Chiffrat in Teile auf, so dass jede Teilzahl kleiner als n ist.
7. Alice berechnet mit dem Chiffretext (bzw. seinen Teilstücke) $c \in \{1, \cdots, n-1\}$:
   $x = D((n, d);C) := c^d \bmod n$

(Die Zahlen d, e, n sind normalerweise sehr groß (z.B. d und e 300 Bit, n 600 Bit).).

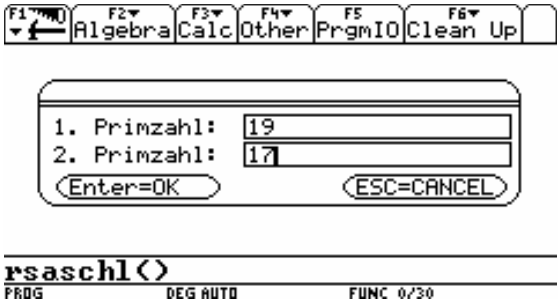
Bestimmung des Private-Keys: Für d wird eine Zufallszahl genommen, die prim ist und für die gilt: $\max(p, q) < d < J(n) - 1$.

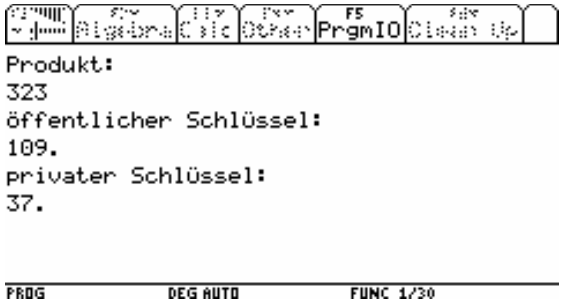
Bestimmung des Public-Keys: e so bestimmt werden, dass (mit $J(n) = r$) $e.d \equiv 1 \bmod r$ gilt. Diese Kongruenz lässt sich in Form einer Gleichung anschreiben:

$e . d = s . r + 1$ (mit $s \in Z$) (vgl.: $22 \equiv 2 \bmod 5 \Leftrightarrow 22 = 4 . 5 + 2$)

Gesucht sind nun ganzzahlige Lösungen für e und s dieser Gleichung (d und r sind teilerfremd!). Diese ganzzahligen Lösungen können mit dem euklidischen Algorithmus berechnet werden.

Zunächst muss man mit dem Programm rsaschl() die beiden Schlüssel berechnen. Dafür werden zwei Primzahlen eingegeben. Diese sollten nicht zu groß sein, da dann bei der Verschlüsselung bzw. Entschlüsselung die Rechnerkapazität überschritten wird (max Primzahlen deren Produkt kleiner als ungefähr 400 ist – leider sehr niedrig!).





Mit dem Programm rsa() wird ein Text verschlüsselt und mit entrsa(out) wieder entschlüsselt: (Im Home-Fenster kann man den gesamten verschlüsselten Text sehen.)