# Inhaltsverzeichnis

# Kapitel 1

# Einführung

## 1.1 Inhalt

Übertragung (Speicherung) von Daten:
Schutz vor:

- zufälligen oder systematischen (physikalischen bedingten) Störungen
- Abhören, absichtliche Veränderung von Dritten (Kryptologie / Verschlüsselung)

Kryptologie:

- symmetrische Verfahren
- asymmetrische Verfahren (Public-Key Verfahren)
- Authentifizierung
- Signaturen

Codierungstheorie

- Fehlererkennung und Fehlerkorrektur
- lineare Blockcodes
- Decodierverfahren

# Kapitel 2

# Kryptologie

## 2.1 Grundbegriffe und einfache Verfahren

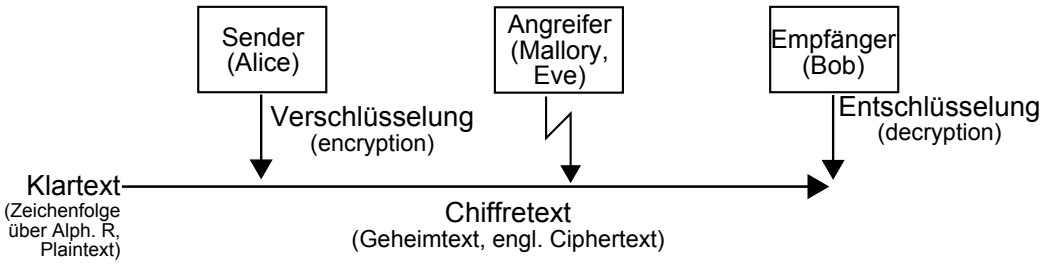


Abbildung 2.1: Schaubild der Kryptologie

### 2.1.1 Verschlüsselung erfordert

- Verschlüsselungsverfahren, Algorithmus (Funktion)
- Schlüssel $k_e$ (encryption key)

$E(m, k_e) = c$
$E$=Verschl.Fkt., $m$=Klartext, $c$=Chiffretext
$E(m_1, k_e) \neq E(m, k_e)$ für $m_1 \neq m_2$
$D(c, k_d) = m$
($k_d$ zu $k_e$ gehöriger Dechiffrierschlüssel!)

$k_d = k_e$ (oder $k_d$ leicht aus $k_e$ zu berechnen):
symmetrisches Verschl.verf., ansonsten asymm. Verschl.verf.. Ist $k_d$ nur sehr schwer (oder garnicht) zu $k_e$ berechenbar, so kann $k_e$ veröffentl. werden:
Public-Key-Verfahren.

### 2.1.2 Beispiel für (nicht sicheres) symm. Verfahren

a) $R = S = \{0, 1, \ldots, 25\}$
Verfahren: Verschiebechiffre
Schlüssel: $i \in \{0, 1, \ldots, 25\}$
Verfahren $x \in \mathbb{R} \longrightarrow x + i \, mod \, 26 = y$
$y \longmapsto y - i \, mod \, 26 = y$
$m = x_1 ... x_2 \longrightarrow c = (x_1 + i \, mod \, 26) \ldots (x_n + i \, mod \, 26)$, $E(m, i)$
Unsicher, weil Schlüsselmenge klein ist (Brute Force Angriff).

b) R,S, Schlüsselmenge=Menge aller Permutationen von $\{1, \ldots, 25\} = S_{26}$
Verschl.: Wähle Permuation $\pi$
$x \in \mathbb{R} \longrightarrow \pi(x) = y$
Entschl.: $y \longrightarrow \pi^{-1}(y) = x$
$m = x_1 \ldots x_r \rightarrow c = \pi(x_1) \ldots \pi(x_r)$
$\begin{pmatrix} 0 & 1 & 2 & \ldots & 25 \\ 3 & 17 & 4 & \ldots & 13 \end{pmatrix} \longrightarrow \pi(0) = 3$, u.s.w.
Anzahl der Permutationen: $|S_{26}| = 26! \approx 4 \cdot 10^{26} \longrightarrow$ Brute-Force Angriff nicht mehr möglich!
Warum? Man muss im Schnitt 50% der Permutationen testen. Angenommen man könnte $10^12$ Perm. pro Sekunde testen.
Aufwand: $2 \cdot 10^{14}$ Sekunden $\approx$ 6.000.000 Jahre
Trotzdem unsicher!
Grund: Charakteristiches Häufigkeitsverteilung von Buchstaben in natürlichspr. Texten.

Verfahren beinhalten viele Verschlüsselungsmöglichkeiten, abhängig von der Auswahl des Schlüssels.
Verfahren bekannt, aber Schlüssel $k_d$ geheim!

### 2.1.3 Prinzip von Kerkhoffs (1835-1903)

Sicherheit eines Verschlüsselungsverfahren darf nicht von der Geheimhaltung des Verfahrens, sondern nur von der Geheimhaltung des verwendeten Schlüssels abhängen!

Kryptologie besteht aus Kryptographie (Entwurf) und der Kryptoanalyse (Angriff).
Angriffserfolge:

- Schlüssel $k_d$ wird gefunden
- Eine zu der Dechiffrierfunktion $D(\cdot, k_d)$ äquivalente Funktion finden ohne Kenntnis von $k_d$
- gewisste Chiffretexte werden entschlüsselt

Arten von Angriffen

- Ciphertext-Only Angriff
- Known-Plaintext Angriff
- Chosen-Plaintext Angriff
- Chosen-Ciphertext Angriff

# Kapitel 3

# One-Time-Pad und perfekte Sicherheit

Lauftextverschlüsselung
Alphabet $\mathbb{Z}_k = \{0, 1, \ldots, k-1\}$
In $\mathbb{Z}_k$ kann man addieren und multiplizieren mit $mod\, k$.
Klartext $x_1, x_2, \ldots, x_n$
Schlüsselwort $k_1, k_2, \ldots, k_n$
$x_1 + k_1 \, mod\, k$, $x_n + k_n \, mod\, k \leftarrow$ Chiffretext
Mit natürlichsprachlichen Texten ist das Verfahren unsicher.
$\mathbb{Z}_2 = \{0, 1\}$, $1 \oplus 1 = 0 = 0 \oplus 0$, $0 \oplus 1 = 1 = 1 \oplus 0 \Rightarrow XOR$
Klartext in $\mathbb{Z}_2^n = \{(x_1, \ldots, x_n) : x_i \in \mathbb{Z}_2\}$ Schlüssel: Zufallsfolge über $\mathbb{Z}_2$ der Länge $n$. $m$ Klartext, $k$ Zufallsfolge (beide Länge $n$)
$c = m \oplus k$, $(x_1, \ldots, x_n) \oplus (k_1, \ldots, k_n) := (x_1 \oplus k_1, \ldots, x_n \oplus k_n)$

## 3.1 One-Time-Pad

Schlüssel $k$ darf nur einmal verwendet werden!

$$m_1 \oplus k = c_1, m_2 \oplus k = c_2, c_1 \oplus c_2 = m_1 \oplus k \oplus m_2 \oplus k = m_1 \oplus m_2$$

Wieder nur Lauftext → unsicher!
$m_1$ und $m_2$ lässt sich ermitteln.
Zufallsfolge der Länge $n$: eigentlich unsinniger Begriff. Da jedes Bit unabhängig von anderen mit Wahrscheinlichkeit $\frac{1}{2}$ erzeugt wird (Output einer binär symmetrischen Quelle)
Jede Folge der Länge $n$ ist gleich wahrscheinlich (Wahrscheinlichkeit $\frac{1}{2}n$
One-Time-Pad ist perfekt sicher.

## 3.2 Perfekte Sicherheit

Ein Verschlüsselungsverfahren ist perfekt sicher, falls gilt: Für jeden Klartext $m$ und jedem Chiffretext $c$ (der festen Länge $n$)
$pr(m|c) = pr(m)$
$pr(m|c) \rightarrow$ A-posteriori-Wahrscheinlichkeit (Wahrscheinlichkeit, dass $m$ Klartext, wenn $c$ empfangen wurde)
$pr(m) \rightarrow$ A-priori-Wahrscheinlichkeit
**Beispiel:** Substitutionschiffre aus Kapitel 2.
$n = 5, m = HALLO, pr(m) > 0$
Ang:$c = QITUA$ wird empfangen, $LL \neq TU \rightarrow pr(m|c) = 0$
nicht perfekt sicher.
One-Time-Pad ist perfekt sicher.
(Bayes'sche Formel) $m \oplus k$
Jede Folge $c$ lässt sich mit geeignetem $k$ in der Form $c = m \oplus k$ erhalten.
Wähle $k = m \oplus c$, $m \oplus k = m \oplus m \oplus c = c$
Bei gegebenem $m$ und zufällige gewählten Schlüssel $k$ ist jeder Chiffretext gleichwertig.

# Kapitel 4

# Symmetrische Blockchiffre

## 4.1 Blockchiffre

Zerlege Klartext in Blöcke (Strings) der Länge $n$. Jeder Block wird einzeln verschlüsselt (in der Regel wieder in einem Block der Länge $n$). Gleiche Blöcke werden gleich verschlüsselt.
Wieviele Blockchiffren der Länge $n$ gibt es?
Alphabet $\mathbb{Z}_2 = \{0, 1\}$
$|\{\underbrace{(0, \ldots, 0)}_{Block}, (0, \ldots, 1), \ldots, (1, \ldots, 1)\}| = 2^n$
Blockchiffre = Permuation der $2^n$ Blöcke.
$(2^n)!$ Blockchiffre
Wenn alle verwendet werden:
Schlüssel = Permuation der $2^n$ Blöcke
$(x_{1,1}, \ldots, x_{1,n}, x_{2,1}, \ldots, x_{2,n}, \ldots)$ $\boxed{n \cdot 2^n \text{ Bit}}$
Zur Speicherung eines Schlüssels werden $n \cdot 2^n$ Bit benötigt.
Zum Beispiel:
$n = 64,\ 64 \cdot 2^{64} = 2^{70} \approx 1$ ZetaByte $\approx 1$ Milliarde Festplatten à 1 TB
**Illusional!**
Konsequenz: Verwende Verfahren, wo nur ein kleiner Teil der Permutation als Schlüssel verwendet wird und so sich die Schlüssel dann in kürzerer Fom darstellt.

# Kapitel 5

# Affin-lineare Chiffre

## 5.1 Vorbemerkung

### 5.1.1 $n \times m$-Matrix

$$\begin{pmatrix} a_{11} & \dots & a_{1m} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nm} \end{pmatrix}$$

$1 \times n$ = Zeilenvektor = $(a_1, \dots, a_m)$

$n \times 1$ = Spaltenvektor = $\begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$

z.B. $a_{ij} \in \mathbb{R},\ a_{ij} \in \mathbb{Z}$ oder $a_{ij} \in R,\ R$ Ring

$n \times m$-Matrix A,B

$$\begin{pmatrix} a_{11} & \dots & a_{1m} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nm} \end{pmatrix} + \begin{pmatrix} b_{11} & \dots & b_{1m} \\ \vdots & & \vdots \\ b_{n1} & \dots & b_{nm} \end{pmatrix} := \begin{pmatrix} a_{11} + b_{11} & \dots & a_{1m} + b_{1m} \\ \vdots & & \vdots \\ a_{n1} + b_{n1} & \dots & a_{nm} + b_{nm} \end{pmatrix}$$

$$A = n \times m,\ B = m \times k,$$

$$A \cdot B \begin{pmatrix} c_{1l} & \dots & c_{1k} \\ \vdots & & \vdots \\ c_{m1} & \dots & c_{mk} \end{pmatrix} = n \times k$$

$$c_{1l} = (a_{i1} \cdot b_{ij}) + (a_{i2} \cdot b_{2j}) + \dots + (a_{im} \cdot b_{mj})$$

$$(A + B) \cdot C = A \cdot B + B \cot C$$

Im Allgemeinem: $A \cdot B \neq B \cdot A$

### 5.1.2 Quadritsche Matrix ($n \times n$)

$$E_n = \begin{pmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \end{pmatrix}$$

$$A = n \times n,\ A \cdot E_n = E_n \cdot A = A$$

$A$ $n \times n$-Matrix über kommutativen Ring R mit Eins.
Wann existiert Matrix $A^{-1}$(Inverse Matrix) mit $A^{-1} \cdot A = A \cdot A^{-1} = E_n$?
$det(A) \in R$ Determinante von A
$2 \times 2$-Matrix $det \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{21} \end{pmatrix} = a_{11} \cdot a_{22} - a_{12} \cdot a_{21}$
A besitzt inverse Matrix $\Leftrightarrow$ $det(A)$ in R ein inverses besitzt
(z.B. R Körper, $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}_p,\ det(A) \neq 0$

$$A^{-1} = \begin{pmatrix} \frac{1}{det(A)} \cdot b_{11} & \dots & \frac{1}{det(A)} \cdot b_{1m} \\ \vdots & & \vdots \\ \frac{1}{det(A)} \cdot b_{n1} & \dots & \frac{1}{det(A)} \cdot b_{nm} \end{pmatrix}$$

$b_{ij} = (-1)^{i+j}\ det(A_{ji})$
$A_{ji}$ = $(n-1) \times (n-1)$-Matrix, die aus $A$ durchstreichen der $j$-ten Zeile und $i$-ten Spalte entsteht.

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}\ A^{-1} = \begin{pmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{pmatrix}$$

$R = \mathbb{Z}_k\ \{0, 1, \dots, k\}$
Addition und Multiplikation in $\mathbb{Z}_k(\oplus, \odot)$
normale Add. und Mult. mit $mod\ k$

## 5.2 Affin-lineare Chiffren

Klartextalphabet = Chiffretextalphabet = $\mathbb{Z}_k$ ($k = 2,\ k = 26$)
Wähle $n \times n$-Matrix $A$ über $\mathbb{Z}_k$ und Zeilenvektor $b$ der Länge $n$ über $\mathbb{Z}_k$. Dies wird der Schlüssel sein für die Chiffrierung.
Blockchiffre der Länge $n$. Block = Zeilenvektor der Länge $n$ über $\mathbb{Z}_k$. Klartextblock $v$
Chiffretextblock $v \cdot A + b =: w$
$v \rightarrow v \cdot A + b =: w$ $w - b = v \cdot A$ benötigen: $A^{-1}$ existiert (d.h. $ggT(det(A), k) = 1$)
Dechiffrierung: $(w - b) \cdot A^{-1} = v \cdot A \cdot A^{-1} = v \cdot E_n = v$
(wenn immer b=0 gewählt wird, dann lineare Chiffren, Hill-Chiffren)
Beispiel:

$$A = \begin{pmatrix} 1 & 3 \\ 3 & 2 \end{pmatrix} \mathbb{Z}_6$$

Blockchiffre der Länge $n$ $det(A) = 1 \cdot 2 - 3 \cdot 3 = -7 = 5$ inverse in $\mathbb{Z}_6$

$$\frac{1}{det(A)} = det(A)^{-1} = 5$$

$$A^{-1} = 5 \cdot \begin{pmatrix} 2 & -3 \\ -3 & 1 \end{pmatrix} = \begin{pmatrix} 10 & -15 \\ -15 & 5 \end{pmatrix} = \begin{pmatrix} 4 & 3 \\ 3 & 5 \end{pmatrix}$$

Test:

$$A \cdot A^{-1} = \begin{pmatrix} 1 & 3 \\ 3 & 2 \end{pmatrix} \cdot \begin{pmatrix} 4 & 3 \\ 3 & 5 \end{pmatrix} = \begin{pmatrix} 4+9 & 3+15 \\ 12+6 & 9+10 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Verschlüsselung:

Schlüssel: $A = \begin{pmatrix} 1 & 3 \\ 3 & 2 \end{pmatrix} b = (3,5)$

Klartextblock: $(1,2)$

Chiffretextblock:

$$w = (1,2) \cdot \begin{pmatrix} 1 & 3 \\ 3 & 2 \end{pmatrix} + (3,5) = (1,1) + (3,5) = (4,0)$$

Entschlüsselung:

$$(w - b) \cdot A^{-1} = (1,1) \cdot \begin{pmatrix} 4 & 3 \\ 3 & 5 \end{pmatrix} = (1,2)$$

$\mathbb{Z}_2 : n^2 + n$ Bit zur Speicherung eines Schlüssels.

Wieviele inverse Matrizen über $\mathbb{Z}_2$ mit $n = 64$?

$(2^{64} - 1) \cdot (2^{64} - 2) \cdot \ldots \cdot (2^{64} - 2^{63}) \approx 0.29 \cdot 2^{4096}$

Verfahren ist unsicher gegenüber Known-Plaintext-Angriffe.

$(A, b)$ Schlüssel, A inverse $n \times n$-Matrix über $\mathbb{Z}_k, b \in \mathbb{Z}_k^n$

Angenommen Angreifer kennt $n+1$ Klartext/Chiffretextpaare verschlüsselt mit $(A, b)$, $v_0, v_1, \ldots, v_n$ $w_0, \ldots, w_n$

Dann kann er haufig $(A, b)$ bestimmen.

$V = \begin{pmatrix} v_1 - v_0 \\ v_2 - v_0 \\ \vdots \\ v_n - v_0 \end{pmatrix}$ $n \times n$-Matrix

Angenommen: V ist invertierbar. Setze $W = \begin{pmatrix} w_1 - w_0 \\ \vdots \\ w_n - w_0 \end{pmatrix}$

$$V \cdot A = \begin{pmatrix} (v_1 - v_0) \cdot A \\ \vdots \\ (v_n - v_0) \cdot A \end{pmatrix} = \begin{pmatrix} v_1 \cdot A + b - v_0 \cdot A + b \\ \vdots \\ v_n \cdot A + b - v_0 \cdot A + b \end{pmatrix} = \begin{pmatrix} w_1 - w_0 \\ \vdots \\ w_n - w_0 \end{pmatrix} = W$$

$V \cdot A$ bekannt, also auch $V^{-1}$:

$A = V^{-1} \cdot w$
$b = w_0 - v_0 \cdot A$
Beispiel: $n = 2,\ k = 25\ \ \{A, \ldots, Z\} = \{0, \ldots, 25\}$

HERBST $\longrightarrow$ NEBLIG

| | | | | | |
|---|---|---|---|---|---|
| H | 7 | $v_0$ | N | 13 | $w_0$ |
| E | 4 | | E | 4 | |
| R | 17 | $v_1$ | B | 1 | $w_1$ |
| B | 1 | | L | 11 | |
| S | 18 | $v_2$ | I | 8 | $w_2$ |
| T | 19 | | G | 6 | |

$$V = \begin{pmatrix} 10 & -3 \\ 11 & 15 \end{pmatrix} = \begin{pmatrix} 10 & 23 \\ 11 & 15 \end{pmatrix},\ W = \begin{pmatrix} 14 & 7 \\ 21 & 2 \end{pmatrix}$$

$$det(V) = 10 \cdot 15 + 33 = 183 \equiv 1 (mod\ 26)$$

$$V^{-1} = \begin{pmatrix} 15 & 3 \\ -11 & 10 \end{pmatrix} = \begin{pmatrix} 15 & 3 \\ 15 & 10 \end{pmatrix}$$

$$A = V^{-1} \cdot W = \begin{pmatrix} 15 & 3 \\ 15 & 10 \end{pmatrix} \cdot \begin{pmatrix} 14 & 7 \\ 21 & 2 \end{pmatrix} = \begin{pmatrix} 210 + 63 & 105 + 6 \\ 210 + 210 & 105 + 20 \end{pmatrix} = \begin{pmatrix} 13 & 7 \\ 4 & 21 \end{pmatrix}$$

$$b = w_0 - v_0 \cdot A = (13, 4) - (7, 4) \cdot \begin{pmatrix} 13 & 7 \\ 4 & 21 \end{pmatrix} = (10, 1)$$

Test:

$$v_1 \cdot A + b = w_1, v_2 \cdot A + b = w_2$$

# Kapitel 6

# Signaturen, Hashfunktionen, Authentifizierung

## 6.1 Anforderung an digitale Signaturen

**Identitätseigenschaft:** ID des Unterzeichners des Dokuments wird sichergestellt

**Echtheitseigenschaft:** des signiertem Dokument

**Verifikationseigenschaft:** Jeder Empfänger muss digitale Signatur verifizieren können.

## 6.2 RSA-Signatur (vereinfachte Version)

$A$ will Dokument $m$ signieren.
$A$ bestitzt öffentlichen RSA-Schlüssel $(n, e)$, geheimen Schlüssel $d$.
Signatur: $m^d \ mod \ n$ sendet $(m, \ m^d \ mod \ n)$ an $B$.
$(m^d \ mod \ n)^e = m^{e \cdot d} \ mod \ n = m \ (mod \ n)$
$m < n$
Wenn $m^{e \cdot d} \ mod \ n = m$, dann akzeptiert $B$ die Signatur.
$m > n \ \ m^d \ mod \ n \ B \ mod \ n$. Ist $m^{'} \ mod \ n = m \ mod \ n$, dann $(m^{'}, m^d \ mod \ n)$ gültige Signatur.

### 6.2.1 Wie lassen sich lange RSA-Signaturen vermeiden?

**Def:** Sei $R$ ein endliches Alphabet.
**Hashfunktion** $H : \mathbb{R}^* \rightarrow R^k (k \in \mathbb{N}$ fest ) soll effizient berechenbar sein.

## 6.3 RSA-Signatur mit HASH-Funktion

$H$ öffentlich bekannte Hashfunktion.
$A$ will Nachricht $m$ signieren.
Bildet $H(m)$ und signiert $H(m) : H(m)^d \ mod \ n$ sendet $(m, H(m)^d \ mod \ n)$
Verifikation durch $B$: $m \rightarrow H(m)$
$(H(m)^d \ mod \ n)^e \ mod \ n = H(m)$

### 6.3.1 Angriffsmöglichkeiten

- Angreifer kann $H(m)$ bestimmen wenn es ihm gelingt, $m' \neq m$ zu finden, so $(m', H(m)^d \ mod \ n)$ gültige Signatur von $m$ durch $A$.

- Angreife wählt zufällig $y$ und berechnet $y^e \mod n = z$
  Gelingt es ihm, $m$ zu finden mit $H(m) = z$, dann ist $(m, y)$ gütlige Signatur von $m$ durch $A$
  $H(m) \ \ y^e = H(m)$

**Def:** Eine **kryptographische Hashfunktion** ist eine Hashfunktion, die folgende Bedinungen erfüllt.

1. $H$ ist Einwegfunktion (um Angriffe des zweiten Typs zu vermeiden)

2. $H$ ist **schwach kollisionsresistent**, d.h. zu gegebenem $m \in R^*$, soll es effizient nicht möglich sein ein $m' \neq m$, mit $H(m) = H(m')$, zu finden. (um Angriffe des ersten Typs zu vermeiden)

Verschärfung von 2.

2′ $H$ ist **stark kollisions resistent**, wenn es effizient nicht möglich ist $m \neq m'$ zu finden, mit $H(m) = H(m')$.

Da $R^*$ unendlich und $\left|R^k\right| = |R|^k$ endlisch ist, existiert unendlich viele Paare $(m, m')$, $m \neq m'$ mit $H(m) = H(m')$.
(Bilde $|R|^k + 1$ viele Hashwerte: Kollision)
Kollisionen lassen sich nicht vermeiden, sie sollten aber nicht schnell herstellbar sein.

### 6.3.2 Satz: Geburtstagsparadoxon

Ein Merkmal komme in $m$ verschiedenen Ausprägungen vor. Jede Person besitze genau eine dieser Merkmalsausprägungen. Ist $c \geq \frac{1+\sqrt{1+8 \cdot m \cdot \ln 2}}{2} \approx 1.18\sqrt{m}$, so ist die Wahrscheinlichkeit, dass unter $l$ Personen zwei die gleiche Merkmalsausprägung haben, mindestens $\frac{1}{2}$ (Geburtstage: $m = 366, l = 23$).

**Beweis** $l$ Personen
Alle Möglichkeiten $(g_1, g_2, \ldots, g_l), g_i \in \{1, \ldots, m\}$ $m^l$ Möglichkeiten.
Alle Merkmalausprägungen verschieden: $m \cdot (m-1) \cdot (m-2) \cdot \ldots \cdot (m-(l-1))$
Wahrscheinlichkeit, dass $l$ Personen lauter verschiedene Geburtstage haben.
$q = \frac{m \cdot (m-1) \cdot (m-2) \cdot \ldots \cdot (m-(l-1)}{m^l} = \prod_0^{l-1} 1 - \frac{i}{m}$
Wann ist $q \leq \frac{1}{2}$?
$e^x \geq 1 + x$

$$\prod_0^{l-1} 1 - \frac{i}{m} \leq \prod_0^{l-1} e^{-\frac{i}{m}} = e^{\prod_0^{l-1} - \frac{i}{m}} = e^{-\frac{1}{m} \sum_0^{l-1} i} = e^{-\frac{1}{m} \cdot \frac{l \cdot (l-1)}{2}}$$

$$\ln a \leq -\frac{1}{m} \cdot \frac{l \cdot (l-1)}{2} = -\frac{l^2 - l}{2 \cdot m}$$

### 6.3.3 Hashfunktion

$H(m) = H(m'), m \neq m'$
$H : \mathbb{Z}_2^* \to \mathbb{Z}_2^n$ ($2^n$ Hashwerte)
Bei Erzeugung von $2^{\frac{n}{2}}$ Hashwerten ist Wahrscheinlichkeit, dass zwei gleich sind $\frac{1}{2}$
$n = 64 : 2^{32}$ Hashwerte ( 4 $cdot 10^9$) unsicher.
Weit verbreitet waren und sind:

MD5 (message digerst / Ron Rivest, 1991, 128 Bit)

SHA-1 (Secure Hash Algorithm, NSA, 1992/1993, 160 Bit)

## 6.4 Authentifizierung

Nachweise bzw. Überprüfung, dass jemand derjenige ist für den er sich ausgbit.
Möglichkeiten der Authentifizierung durch:

**Wissen**

Besitz

biometische Merkmale

gängiste Methode: Passwort
Im Allgemeinem: Passwort $w$ abgespeichert als $f(w)$ $f$ Einwegfunktion.
$w$ $f^n(w) = w_0 \overset{sicher}{\to}$ Id. überprüfer $f$ Einweg.

1. Auth. $w_1 = f^{n-1}(w) \to f(f^{n-1}(w)) = w_0$ ersetzt $w_0$ durch $w_1$
2. Auth. $w_2 = f^{n-2}(w) \to \ldots$

Passwortsicherheit: http://www.schneier.com/crypto-gram-0701.html

## 6.5 Challenge-Response-Authentifizierung

RSA-Verfahren $A \overset{auth.}{\longrightarrow} B$
Öffentlicher Schlüssel: $(n, e)$
geheimer Schlüssel: $d$
$A \overset{Zufallszahlr}{\longleftarrow} B,\ r < n$ `Challenge`
$A \overset{r^d \bmod n}{\longrightarrow} B$ überprüft, ob $r^{d^e} \bmod n = r$ `Response`

Damit $B$ sich sicher seien kann, dass es wirklich $A$ ist, kann $B$ so oft wie es für nötig hält neue $r$ schicken und dadurch die Chance verringern, dass $A$ nicht $A$ ist.

# Kapitel 7

# Secret Sharing

Geheimnis wird auf mehrere Teilnehmer verteilt (Teilgeheimnisse), so dass gewisse Teilmengen der Teilnehmer das Geheimnis mit ihren Teilgeheimnissen rekonstruieren können, die anderen nicht.

$T = \{ t_1, \ldots, t_n \}, k < n$ (T Menge der Teilnehmer)

Jede Teilmenge von $T$ mit mindestens $k$ Teilnehmer sollen Geheimnis rekonstruieren können, Teilmengen von $T$ mit weniger als $k$ Teilnehmer nicht.

## 7.1 $(k, n)$ - Schwellenwertsysteme

1979 Shamir (How to share a secret)

### 7.1.1 Konstruktion

Vereinbarung von großer Primzahl $p$, mindestens $p \geq n + 1$

$g \in \mathbb{Z}_p = \{0, \ldots, p - 1\}$

### 7.1.2 Verteilung der Teilgeheimnisse

Dealer wählt zufällig $a_1, \ldots, a_{k-1} \in \mathbb{Z}_p, a_{k-1} \neq 0, k$ = Schwelle
$f(x) = g + a_1 x + \ldots + a_{k-1} x^{k-1} \in \mathbb{Z}_p[x]$
($a_1, \ldots, a_{k-1}$ hält er geheim, natürlich auch g)

Dealer wählt zufällig $x_1, \ldots, x_n \in \mathbb{Z}_p$ (paarweise verschieden).
Teilnehmer $t_i$ erhält als Teilgeheimnis $(x_i, f(x_i))$ (Punkt auf Polynom)
Bei $x = 0$ hast du $g$.

### 7.1.3 Rekonstruktion(sversuch) des Geheimnisses

$k$ Teilnehmer $(x_{i_1}, f(x_{i_1})), \ldots, (x_{i_k}, f(x_{i_k}))$
Durch diese Punkte ist $f$ eindeutig bestimmt, z.B. durch Lagrange-Interpol.:
$f(x_{i_j}) = g_{i_j}$

$f(x) = \sum_{j=1}^{k} g_{i_j} \cdot \frac{(x-x_{i_1}),\ldots,(x-x_{i_{j-1}})(x-x_{i_{j+1}}),\ldots,(x-x_{i_k})}{(x_{i_j}-x_{i_1}),\ldots,(x_{i_j}-x_{i_{j-1}})(x_{i_j}-x_{i_{j+1}}),\ldots,((x_{i_j}-x_{i_k})}$
$f(0) = g$
$g = \sum_{j=1}^{k} g_{i_j} \prod_{l=j} \frac{x_{i_l}}{(x_{i_l}-x_{i_j})}$
Bei mehr als $k$ Teilnehmer selbe Ergebnis.
Weniger als $k$ Teilnehmer ($k'$): Anderes Polynom wegen weniger Punkte, also warscheinlich anderer $g$.
Erzeugen Polynom vom Grad $\leq k' - 1$
Für alle $k \in \mathbb{Z}_p$ existiert gleich viele Polynome vom Grad $\leq k' - 1$ durch die vorgegebene $k'$ Punkte, die bei $h$ durch $y$-Achse gehen.