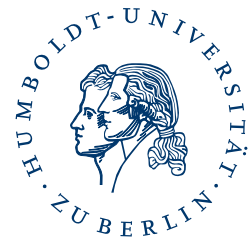
Einführung in die Kryptologie

Johannes Köbler



Institut für Informatik
Humboldt-Universität zu Berlin

SS 2020

Lernziele

- Kryptografische Verfahren schaffen Vertrauen in ungeschützten Umgebungen
- Sie ermöglichen sichere Kommunikation über unsichere Kanäle und können verhindern, dass sich ein Kommunikationspartner unfair verhält
- In unsicheren Umgebungen wie dem Internet können sie die aus direkter Interaktion gewohnte Sicherheit herstellen
- Und auch die Interaktion in sicheren Umgebungen wird um Möglichkeiten erweitert, die ohne Kryptografie nicht denkbar wären
- In diesem Modul werden wir uns mit den mathematischen Grundlagen von kryptografischen Verfahren beschäftigen, wobei (symmetrische und asymmetrische) Verschlüsselungsverfahren im Vordergrund stehen
- Im Mastermodul Kryptologie werden wir dann auch kryptografische Verfahren und Protokolle für andere Schutzziele betrachten wie z.B. Hashverfahren und digitale Signaturen sowie Pseudozufallsgeneratoren

Kryptosysteme

- Kryptosysteme (Verschlüsselungsverfahren) dienen der Geheimhaltung von Nachrichten bzw. Daten
- Hierzu gibt es auch andere Methoden wie z.B.
  - Physikalische Maßnahmen: Tresor etc.
  - Organisatorische Maßnahmen: einsamer Waldspaziergang etc.
  - Steganografische Maßnahmen: unsichtbare Tinte etc.

Überblick weiterer Schutzziele

Andererseits können durch kryptografische Verfahren weitere Schutzziele realisiert werden wie z.B.

- Vertraulichkeit
  - Geheimhaltung
  - Anonymität (z.B. Mobiltelefon)
  - Unbeobachtbarkeit (von Transaktionen)
- Integrität
  - von Nachrichten und Daten
- Zurechenbarkeit
  - Authentikation
  - Unabstreitbarkeit
  - Identifizierung
- Verfügbarkeit
  - von Daten
  - von Rechenressourcen
  - von Informationsdienstleistungen

Kryptologie

In das Umfeld der Kryptologie fallen die folgenden Begriffe

- Kryptografie:
  Lehre von der Geheimhaltung von Informationen durch Verschlüsselung
  Im weiteren Sinne: Wissenschaft von der Übermittlung, Speicherung und Verarbeitung von Daten in einer von potentiellen Gegnern bedrohten Umgebung
- Kryptoanalysis:
  Erforschung der Methoden eines unbefugten Angriffs gegen ein Kryptoverfahren
  Zweck: Vereitelung der mit seinem Einsatz verfolgten Ziele
- Kryptoanalyse:
  Analyse eines Kryptoverfahrens zum Zweck der Bewertung seiner kryptografischen Stärken und Schwächen
- Kryptologie:
  Wissenschaft vom Entwurf, der Anwendung und der Analyse von kryptografischen Verfahren (umfasst Kryptografie und Kryptoanalyse)

Abgrenzung zwischen Krypto- und Codesystemen

Codesysteme

- operieren auf semantischen Einheiten
- starre Festlegung, welche Zeichenfolge wie zu ersetzen ist

Beispiel (Ausschnitt aus einem Codebuch der deutschen Luftwaffe)

`xve`	**Bis auf weiteres Wettermeldung gemäß Funkbefehl testen**
`yde`	**Frage**
`sLk`	**Befehl**
`fin`	**beendet**
`eom`	**eigene Maschinen**

Kryptosysteme

- operieren auf syntaktischen Einheiten
- flexibler Mechanismus durch Schlüsselvereinbarung

Alphabet

Definition

- Ein ***Alphabet*** $A = \{a_0, \ldots, a_{m-1}\}$ ist eine geordnete endliche Menge von ***Zeichen*** a_i
- Eine Folge $x = x_1 \ldots x_n \in A^n$ heißt ***Wort*** (der ***Länge*** n)
- Die Menge aller Wörter über dem Alphabet A ist $A^* = \bigcup_{n \geq 0} A^n$

Beispiel

- Das ***lateinische Alphabet*** A_{lat} enthält die 26 Buchstaben `A`, ..., `Z`
- Bei Klartexten wurde meist auf den Gebrauch von Interpunktions- und Leerzeichen sowie auf Groß- und Kleinschreibung verzichtet
  $\rightsquigarrow$ Verringerung der Redundanz im Klartext

Mathematische Beschreibung eines Kryptosystems

Definition

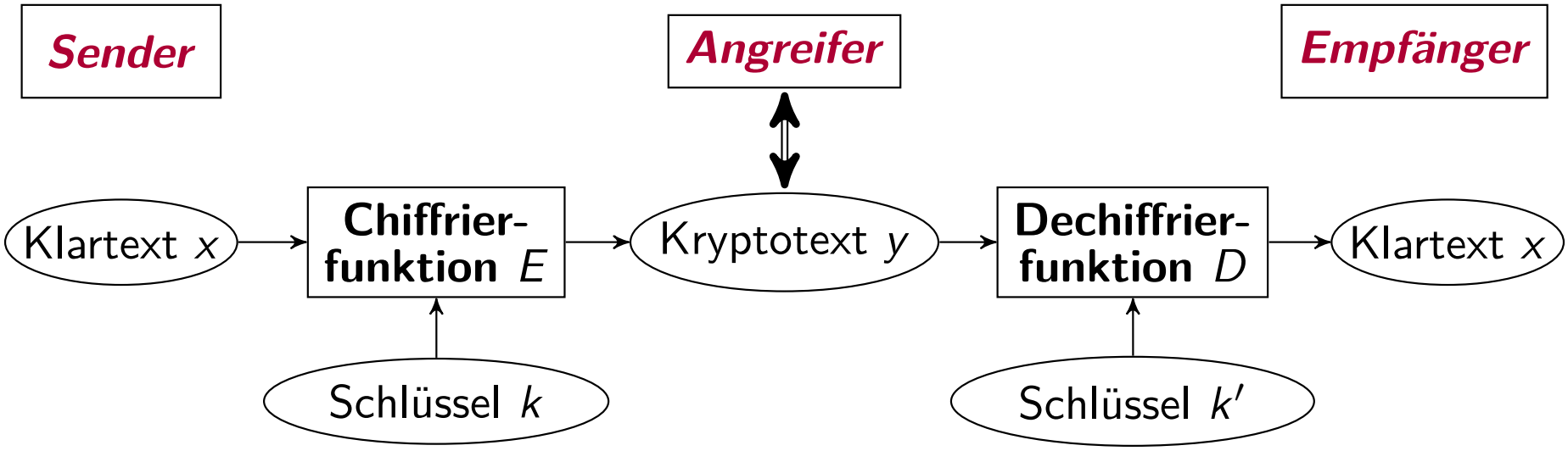
Ein ***Kryptosystem*** wird durch folgende Komponenten beschrieben:

- A, das ***Klartextalphabet***,
- B, das ***Kryptotextalphabet***,
- K, der ***Schlüsselraum*** (key space),
- $M \subseteq A^*$, der ***Klartextraum*** (message space),
- $C \subseteq B^*$, der ***Kryptotextraum*** (ciphertext space),
- $E : K \times M \to C$, die ***Verschlüsselungsfunktion*** (encryption function),
- $D : K \times C \to M$, die ***Entschlüsselungsfunktion*** (decryption function)
- $S \subseteq K \times K$, eine Menge von Schlüsselpaaren (k, k') mit der Eigenschaft, dass für jeden Klartext $x \in M$ folgende Beziehung gilt:

$$D(k', E(k, x)) = x \qquad (*)$$

Bei symmetrischen Kryptosystemen ist $S = \{(k, k) \mid k \in K\}$, weshalb wir in diesem Fall auf die Angabe von S verzichten können

Ver- und Entschlüsseln mit einem Kryptosystem



- Zu jedem Schlüssel $k \in K$ korrespondiert eine
  - ***Chiffrierfunktion*** $E_k : x \mapsto E(k, x)$ und eine
  - ***Dechiffrierfunktion*** $D_k : y \mapsto D(k, y)$
- Die Gesamtheit dieser Abbildungen wird auch ***Chiffre*** (englisch cipher) genannt
- Daneben wird der Begriff „Chiffre“ auch als Bezeichnung für einzelne Kryptotextzeichen oder kleinere Kryptotextsequenzen verwendet

Notwendigkeit der Injektivität von Chiffrierfunktionen

Lemma

Für jedes Paar $(k, k') \in S$ ist die Chiffrierfunktion E_k injektiv

Beweis

- Angenommen, für zwei Klartexte x_1 und x_2 gilt $E(k, x_1) = E(k, x_2)$
- Dann folgt

$$x_1 \overset{(*)}{=} D(k', \underbrace{E(k, x_1)}_{E(k,x_2)}) = D(k', E(k, x_2)) \overset{(*)}{=} x_2$$

□

Moduloarithmetik

Die Moduloarithmetik erlaubt es uns, das Klartextalphabet mit einer Addition und Multiplikation auszustatten

Definition (*teilt-Relation, modulare Kongruenz*)

Seien a, b, m ganze Zahlen mit $m \geq 1$

- Die Zahl a ***teilt*** b (kurz: $a|b$), falls ein $d \in \mathbb{Z}$ existiert mit $b = ad$
- Teilt m die Differenz $a - b$, so schreiben wir hierfür $a \equiv_m b$
  (in Worten: a ist ***kongruent*** zu b modulo m)
- Weiterhin bezeichne

$$a \bmod m = \min\{a - dm \geq 0 \mid d \in \mathbb{Z}\}$$

  den bei der Ganzzahldivision von a durch m auftretenden ***Rest***
  (also diejenige ganze Zahl $r \in \{0, \ldots, m-1\}$, für die eine Zahl $d \in \mathbb{Z}$ existiert mit $a = dm + r$)

Addition und Multiplikation von Buchstaben

- Die auf $\mathbb{Z}$ definierten Operationen

  $$a \oplus_m b := (a + b) \bmod m$$

  und

  $$a \odot_m b := ab \bmod m$$

  sind abgeschlossen auf $\mathbb{Z}_m = \{0, \ldots, m-1\}$ und
- bilden auf dieser Menge einen kommutativen Ring mit Einselement, den sogenannten ***Restklassenring*** modulo m
- Für $a \oplus_m -b$ schreiben wir auch $a \ominus_m b$
- Wenn aus dem Kontext klar ist, dass $a, b \in \mathbb{Z}_m$ sind, schreiben wir anstelle von $a \oplus_m b$, $a \ominus_m b$ und $a \odot_m b$ auch einfach $a + b$, $a - b$ bzw. ab
- Durch Identifikation der Zeichen a_i eines Alphabets $A = \{a_0, \ldots, a_{m-1}\}$ mit ihren Indizes können wir die auf $\mathbb{Z}_m$ definierten Rechenoperationen auf Buchstaben übertragen

Die additive Chiffre

Definition (*Buchstabenrechnung*)

Sei $A = \{a_0, \ldots, a_{m-1}\}$ ein Alphabet. Für Indizes $i, j \in \{0, \ldots, m-1\}$ und eine ganze Zahl $z \in \mathbb{Z}$ definieren wir

$$a_i + a_j = a_{i \oplus_m j}, \quad a_i - a_j = a_{i \ominus_m j}, \quad a_i a_j = a_{i \odot_m j},$$
$$a_i + z = a_{i \oplus_m z}, \quad a_i - z = a_{i \ominus_m z}, \quad z a_j = a_{z \odot_m j}.$$

Mit Hilfe dieser Notation lässt sich die Verschiebechiffre, die auch als additive Chiffre bezeichnet wird, leicht beschreiben

Definition

- Bei der ***additiven Chiffre*** ist $A = B = M = C$ ein beliebiges Alphabet mit $m := \|A\|$ und $K = \{0, \ldots, m-1\}$
- Für $k \in K$, $x \in M$ und $y \in C$ gilt

$$E(k, x) = x + k \quad \text{und} \quad D(k, y) = y - k$$

Die ROT13 Verschlüsselung

- Im Fall des lateinischen Alphabets führt der Schlüssel $k = 13$ auf eine interessante Chiffrierfunktion

x	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
$E(13, x)$	*N*	*O*	*P*	*Q*	*R*	*S*	*T*	*U*	*V*	*W*	*X*	*Y*	*Z*	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*	*K*	*L*	*M*

- Diese ist in UNIX-Umgebungen unter der Bezeichnung ROT13 bekannt
- Natürlich kann mit dieser Substitution nicht ernsthaft die Vertraulichkeit von Nachrichten gewahrt werden
- Vielmehr soll durch sie ein unbeabsichtigtes Mitlesen – etwa von Rätsellösungen – verhindert werden
- ROT13 ist eine ***involutorische*** (also zu sich selbst inverse) Abbildung, d.h. für alle $x \in A$ gilt $\mathrm{ROT13}(\mathrm{ROT13}(x)) = x$
- Da ROT13 zudem keinen Buchstaben auf sich selbst abbildet, ist sie sogar ***echt involutorisch***

Die multiplikative Chiffre

- Die Buchstabenrechnung legt folgende Modifikation der Caesar-Chiffre nahe
- Anstatt auf jeden Klartextbuchstaben den Schlüsselwert k zu addieren, können wir die Klartextbuchstaben auch mit k multiplizieren
- Allerdings erhalten wir hierbei nicht für jeden Wert von k eine injektive Chiffrierfunktion
- So bildet etwa die Funktion $g : A_{lat} \to A_{lat}$ mit $g(x) = 2x$ sowohl $\mathtt{A}$ als auch $\mathtt{N}$ auf den Buchstaben $g(\mathtt{A}) = g(\mathtt{N}) = \mathit{A}$ ab
- Um eine hinreichende und notwendige Bedingung für die Zulässigkeit eines Schlüsselwerts k formulieren zu können, führen wir zunächst eine Reihe von zahlentheoretischen Begriffen ein

Begriffe der elementaren Zahlentheorie

Definition (*ggT, kgV, teilerfremd*)

Seien $a, b \in \mathbb{Z}$. Für $(a, b) \neq (0, 0)$ ist

$$\mathrm{ggT}(a, b) = \max\{d \in \mathbb{Z} \mid d|a \wedge d|b\}$$

der ***größte gemeinsame Teiler*** von a und b und für $a \neq 0, b \neq 0$ ist

$$\mathrm{kgV}(a, b) = \min\{d \in \mathbb{Z} \mid d \geq 1 \wedge a|d \wedge b|d\}$$

das ***kleinste gemeinsame Vielfache*** von a und b. Ist $\mathrm{ggT}(a, b) = 1$, so nennt man a und b ***teilerfremd*** oder man sagt, a ist ***relativ prim*** zu b

Lemma. Seien $a, b, c \in \mathbb{Z}$ mit $(a, b) \neq (0, 0)$. Dann gilt

- $\mathrm{ggT}(a, b) = \mathrm{ggT}(b, a + bc)$ und somit
- $\mathrm{ggT}(a, b) = \mathrm{ggT}(b, a \bmod b)$, falls $b \geq 1$ ist

Beweis.

Ein Teiler von a und b ist auch Teiler von b und $a + bc$ und umgekehrt □

Euklidscher Algorithmus

- Der größte gemeinsame Teiler zweier Zahlen a und b lässt sich wie folgt bestimmen (o. B. d. A. gelte $a > b > 0$)
- Bestimme durch Divsision mit Rest natürliche Zahlen
  $r_0 = a > r_1 = b > r_2 > \cdots > r_s > r_{s+1} = 0$ und $d_2, d_3, \ldots d_{s+1}$ mit

  $$r_{i-1} = d_{i+1} r_i + r_{i+1} \text{ für } i = 1, \ldots, s$$

  (also $d_{i+1} = r_{i-1}$ div r_i und $r_{i+1} = r_{i-1}$ mod r_i)
- Hierzu sind s Divisionsschritte erforderlich
- Wegen

  $$\mathrm{ggT}(r_{i-1}, r_i) = \mathrm{ggT}(r_i, \underbrace{r_{i-1} - d_{i+1} r_i}_{r_{i+1}})$$

  folgt $\mathrm{ggT}(a, b) = \mathrm{ggT}(r_s, r_{s+1}) = r_s$

Euklidscher Algorithmus

Der Euklidsche Alg. kann iterativ oder rekursiv implementiert werden

Prozedur $\texttt{Euklid}_{\texttt{rek}}(a, b)$

```
1 if b = 0 then
2   return(a)
3 else
4   return(Euklid_rek(b, a mod b))
```

Prozedur $\texttt{Euklid}_{\texttt{it}}(a, b)$

```
1 repeat
2   r := a mod b
3   a := b
4   b := r
5 until r = 0
6 return(a)
```

Beispiel

Für $a = 693$ und $b = 147$ erhalten wir $\text{ggT}(693, 147) = r_4 = 21$

i	r_{i-1}	$=$	d_{i+1}	$\cdot$	r_i	$+$	r_{i+1}
1	693	$=$	4	$\cdot$	147	$+$	105
2	147	$=$	1	$\cdot$	105	$+$	42
3	105	$=$	2	$\cdot$	42	$+$	21
4	42	$=$	2	$\cdot$	**21**	$+$	0

$\triangleleft$

Laufzeit des Euklidschen Algorithmus

- Zur Abschätzung von s verwenden wir die Folge der Fibonacci-Zahlen

$$F_n = \begin{cases} 0, & \text{falls } n = 0 \\ 1, & \text{falls } n = 1 \\ F_{n-1} + F_{n-2}, & \text{falls } n \geq 2 \end{cases}$$

- Induktiv über $i = s+1, s, \ldots, 0$ folgt $r_i \geq F_{s+1-i}$ und somit $a = r_0 \geq F_{s+1}$
- Induktiv über $n \geq 0$ folgt $F_{n+1} \geq \phi^{n-1}$ für $\phi = \frac{1+\sqrt{5}}{2}$ (goldener Schnitt):
  - Der Induktionsanfang ($n = 0$ oder 1) ist klar:
    $F_2 = F_1 = 1 = \phi^0 \geq \phi^{-1}$
  - Unter der Induktionsvoraussetzung (IV) $F_{i+1} \geq \phi^{i-1}$ für $i \leq n-1$ folgt wegen $\phi^2 = \phi + 1$

$$F_{n+1} = F_n + F_{n-1} \geq \phi^{n-2} + \phi^{n-3} = \phi^{n-3}(\phi + 1) = \phi^{n-1}$$

- Somit ist $a \geq \phi^{s-1}$, d. h. $s \leq 1 + \lfloor \log_\phi a \rfloor$

Laufzeit des Euklidschen Algorithmus

Als Folgerung erhalten wir folgende Laufzeitabschätzung

Satz

- Seien $a > b > 0$ ganze Zahlen und sei n die Länge von a in Binärdarstellung
- Dann führt der Euklidsche Algorithmus $O(n)$ Divisionsschritte zur Berechnung von ggT(a, b) durch
- Dies führt auf eine Zeitkomplexität von $O(n^3)$, da jede Ganzzahldivision in Zeit $O(n^2)$ durchführbar ist

Erweiterter Euklidscher Algorithmus

- Der Euklidsche Algorithmus kann so modifiziert werden, dass er eine lineare Darstellung des ggT liefert:

  $$\mathrm{ggT}(a,b) = \lambda a + \mu b \text{ mit } \lambda, \mu \in \mathbb{Z} \quad \text{(ebenfalls in Zeit } O(n^3)\text{)}$$

- Hierzu werden für $i = 0, \ldots, s$ neben r_i und d_i weitere Zahlen bestimmt:

  $$\begin{aligned} p_i &= p_{i-2} - d_i p_{i-1} \text{ (mit } p_0 = 1 \text{ und } p_1 = 0\text{) und} \\ q_i &= q_{i-2} - d_i q_{i-1} \text{ (mit } q_0 = 0 \text{ und } q_1 = 1\text{)} \end{aligned}$$

- Dann gilt die Gleichung $ap_i + bq_i = r_i$ für $i = 0$ und $i = 1$ und wegen

  $$\begin{aligned} ap_{i+1} + bq_{i+1} &= a(p_{i-1} - d_{i+1}p_i) + b(q_{i-1} - d_{i+1}q_i) \\ &= ap_{i-1} + bq_{i-1} - d_{i+1}(ap_i + bq_i) \\ &= (r_{i-1} - d_{i+1}r_i) \\ &= r_{i+1} \end{aligned}$$

  folgt induktiv über $i = 2, \ldots, s$, dass sie auch für $i = s$ gilt:

  $$ap_s + bq_s = r_s = \mathrm{ggT}(a,b)$$

Erweiterter Euklidscher Algorithmus

Korollar (Lemma von Bezout)

Der größte gemeinsame Teiler von a und b ist in der Form

$$\text{ggT}(a,b) = \lambda a + \mu b \text{ mit } \lambda, \mu \in \mathbb{Z}$$

darstellbar

Beispiel

Für $g = \text{ggT}(693, 147)$ ergibt sich die Darstellung $g = 3 \cdot 693 - 14 \cdot 147 = 21$:

i	$r_{i-1} = d_{i+1} \cdot r_i + r_{i+1}$	p_i	q_i	$p_i \cdot 693 + q_i \cdot 147 = r_i$
0		1	0	$1 \cdot 693 + 0 \cdot 147 = 693$
1	$693 = 4 \cdot 147 + 105$	0	1	$0 \cdot 693 + 1 \cdot 147 = 147$
2	$147 = 1 \cdot 105 + 42$	1	-4	$1 \cdot 693 - 4 \cdot 147 = 105$
3	$105 = 2 \cdot 42 + 21$	-1	5	$-1 \cdot 693 + 5 \cdot 147 = 42$
4	$42 = 2 \cdot \mathbf{21} + 0$	**3**	$\mathbf{-14}$	$3 \cdot 693 - 14 \cdot 147 = 21$

◁

Schlussfolgerungen aus dem Lemma von Bezout

Korollar

Der $\mathrm{ggT}(a, b)$ wird von allen gemeinsamen Teilern von a und b geteilt:

$$x|a \ \wedge \ x|b \quad \Rightarrow \quad x|\,\mathrm{ggT}(a, b).$$

Beweis

Seien $\mu, \lambda \in \mathbb{Z}$ mit $\mu a + \lambda b = \mathrm{ggT}(a, b)$. Falls x sowohl a als auch b teilt, dann teilt x auch die Produkte μa und λb und somit auch deren Summe $\square$

Korollar

$\mathrm{ggT}(a, b) = \min\{\lambda a + \mu b \geq 1 \mid \lambda, \mu \in \mathbb{Z}\}$

Beweis

Sei $M = \{\lambda a + \mu b \geq 1 \mid \lambda, \mu \in \mathbb{Z}\}$, $m = \min M$ und $g = \mathrm{ggT}(a, b)$. Wegen

- $g \in M$ folgt dann $g \geq m$ und da
- g jede Zahl in M teilt, folgt auch $g \leq m$ $\square$

Schlussfolgerungen aus dem Lemma von Bezout

Korollar

$\mathrm{ggT}(a,b) = \min\{\lambda a + \mu b \geq 1 \mid \lambda, \mu \in \mathbb{Z}\}$ $(*)$

Korollar

$\mathrm{ggT}(a,m) = \mathrm{ggT}(b,m) = 1 \quad \Leftrightarrow \quad \mathrm{ggT}(ab,m) = 1$

Beweis.

- Da a und b teilerfremd zu m sind, existieren Zahlen $\mu, \lambda, \mu', \lambda' \in \mathbb{Z}$ mit $\mu a + \lambda m = 1 = \mu' b + \lambda' m$. Wegen

$$1 = (\mu a + \lambda m)(\mu' b + \lambda' m) = \underbrace{\mu\mu'}_{\mu''} ab + \underbrace{(\mu a \lambda' + \mu' b \lambda + \lambda\lambda' m)}_{\lambda''} m$$

  folgt dann $\mathrm{ggT}(ab,m) = 1$ aus Gleichung $(*)$

- Gilt umgekehrt $\mathrm{ggT}(ab,m) = 1$, so existieren Zahlen $\mu, \lambda \in \mathbb{Z}$ mit $\mu ab + \lambda m = 1$. Mit $(*)$ folgt nun sofort $\mathrm{ggT}(a,m) = \mathrm{ggT}(b,m) = 1$ $\square$

Schlussfolgerungen aus dem Lemma von Bezout

Korollar (Lemma von Euklid)

Sind a und b teilerfremd und teilt a das Produkt bc, so teilt a auch c:

$$\text{ggT}(a,b) = 1 \ \wedge \ a|bc \quad \Rightarrow \quad a|c$$

Beweis

- Wegen $\text{ggT}(a,b) = 1$ existieren Zahlen $\mu, \lambda \in \mathbb{Z}$ mit $\mu a + \lambda b = 1$
- Falls a das Produkt bc teilt, muss a auch die Zahl $\mu ac + \lambda bc = c$ teilen

□

Zulässige Schlüssel für die multiplikative Chiffre

- Sei A ein Alphabet und sei $g : A \to A$ eine Abbildung der Form $g(x) = bx$
- Damit g injektiv (oder gleichbedeutend, surjektiv) ist, muss es zu jedem Zeichen $y \in A$ genau einen Zeichen $x \in A$ mit $bx = y$ geben
- Der folgende Satz gibt hierfür eine notwendige und hinreichende Bedingung

Satz. Seien b, y, m ganze Zahlen mit $m \geq 1$.

Dann besitzt die Kongruenz $bx \equiv_m y$ genau im Fall $\mathrm{ggT}(b, m) = 1$ eine eindeutige Lösung $x \in \{0, \ldots, m-1\}$.

Zulässige Schlüssel für die multiplikative Chiffre

Satz. Seien b, y, m ganze Zahlen mit $m \geq 1$.

Dann besitzt die Kongruenz $bx \equiv_m y$ genau im Fall $\mathrm{ggT}(b, m) = 1$ eine eindeutige Lösung $x \in \{0, \ldots, m-1\}$.

Beweis.

- Im Fall $\mathrm{ggT}(b, m) = g > 1$ ist mit x auch $x' = x + m/g \not\equiv_m x$ eine Lösung von $bx \equiv_m y$ und somit die Kongruenz $bx \equiv_m y$ nicht eindeutig lösbar
- Umgekehrt folgt im Fall $\mathrm{ggT}(b, m) = 1$ aus den beiden Kongruenzen

  $$bx_1 \equiv_m y \text{ und } bx_2 \equiv_m y$$

  sofort $b(x_1 - x_2) \equiv_m 0$, also $m | b(x_1 - x_2)$
- Mit dem Lemma von Euklid folgt $m | (x_1 - x_2)$, d.h. $x_1 \equiv_m x_2$
- Folglich ist die Abbildung $f : \mathbb{Z}_m \to \mathbb{Z}_m$ mit $f(x) = bx \bmod m$ injektiv und somit auch surjektiv (da Definitions- und Wertebereich übereinstimmen), d.h. die Kongruenz $bx \equiv_m y$ hat genau eine Lösung in $\mathbb{Z}_m$ $\square$

Zulässige Schlüssel für die multiplikative Chiffre

Korollar

Im Fall $\text{ggT}(b, m) = 1$ hat die Kongruenz $bx \equiv_m 1$ genau eine Lösung, die das ***multiplikative Inverse*** von b modulo m genannt und mit b^{-1} mod m (oder einfach mit b^{-1}) bezeichnet wird

- Wegen $\text{ggT}(a, m) = \text{ggT}(b, m) = 1 \Rightarrow \text{ggT}(ab, m) = 1$ ist die Menge

  $$\mathbb{Z}_m^* = \{b \in \mathbb{Z}_m \mid \text{ggT}(b, m) = 1\}$$

  aller invertierbaren Elemente von $\mathbb{Z}_m$ unter der Operation $\odot_m$ abgeschlossen, d.h. $(\mathbb{Z}_m^*, \odot_m, 1)$ bildet eine multiplikative Gruppe
- Allgemeiner zeigt man, dass die Multiplikation eines beliebigen Rings $(R, +, \cdot, 0, 1)$ mit Eins auf der Menge

  $$R^* = \{a \in R \mid \exists b \in R : ab = 1 = ba\}$$

  aller Einheiten von R eine Gruppe bildet (siehe Übungen)
- Diese Gruppe $(R^*, \cdot, 1)$ wird als Einheitengruppe von R bezeichnet

Berechnung von multiplikativen Inversen

- Das multiplikative Inverse von b modulo m ergibt sich aus der linearen Darstellung

  $$\lambda b + \mu m = \mathrm{ggT}(b, m) = 1$$

  zu $b^{-1} = \lambda \bmod m$
- Die folgende Tabelle gibt für jedes $b \in \mathbb{Z}_{26}^*$ das multiplikative Inverse an:

b	1	3	5	7	9	11	15	17	19	21	23	25
b^{-1}	1	9	21	15	3	19	7	23	11	5	17	25

- Bei Kenntnis von b^{-1} kann die Kongruenz $bx \equiv_m y$ leicht zu $x = yb^{-1} \bmod m$ gelöst werden

Die affine Chiffre

Definition

Bei der ***affinen Chiffre*** ist $A = B = M = C$ ein beliebiges Alphabet mit $m := \|A\|$ und $K = \mathbb{Z}_m^* \times \mathbb{Z}_m$. Für $k = (b, c) \in K$, $x \in M$ und $y \in C$ gilt

$$E(k, x) = bx + c \quad \text{und} \quad D(k, y) = b^{-1}(y - c)$$

- Hierbei liefert die Schlüsselkomponente $b = -1$ für jeden Wert von $c \in \mathbb{Z}_m$ eine involutorische Chiffrierfunktion $x \mapsto E_{(-1,c)}(x) = c - x$
- Wählen wir für c ebenfalls den Wert -1, so ergibt sich die Funktion $E_{(-1,-1)} : x \mapsto -x - 1$, die auch als ***revertiertes Alphabet*** bekannt ist

x	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
$-x$	*A*	*Z*	*Y*	*X*	*W*	*V*	*U*	*T*	*S*	*R*	*Q*	*P*	*O*	*N*	*M*	*L*	*K*	*J*	*I*	*H*	*G*	*F*	*E*	*D*	*C*	*B*
$-x-1$	*Z*	*Y*	*X*	*W*	*V*	*U*	*T*	*S*	*R*	*Q*	*P*	*O*	*N*	*M*	*L*	*K*	*J*	*I*	*H*	*G*	*F*	*E*	*D*	*C*	*B*	*A*

- Offenbar ist $E_{(-1,-1)}$ genau dann echt involutorisch, wenn m gerade ist

Die affine Chiffre

Beispiel (affine Chiffre)

- Sei $A = \{\mathtt{A}, \ldots, \mathtt{Z}\} = B$, also $m = 26$, und sei $k = (b, c) = (9, 2)$
- Um den Klartextbuchstaben $x = \mathtt{F}$ zu verschlüsseln, berechnen wir

$$E(k, x) = bx + c = 9\mathtt{F} + 2 = V,$$

da $\mathtt{F}$ den Index 5 und V den Index 21 hat und $9 \cdot 5 + 2 = 47 \equiv_{26} 21$ ist

- Für die Entschlüsselung benötigen wir das Inverse $b^{-1} = q_3 = 3$:

i	r_{i-1}	$=$	$d_{i+1} \cdot r_i + r_{i+1}$	$p_i \cdot 26 +$	$q_i \cdot 9 =$	r_i
0				$1 \cdot 26 +$	$0 \cdot 9 =$	26
1	26	$=$	$2 \cdot 9 + 8$	$0 \cdot 26 +$	$1 \cdot 9 =$	9
2	9	$=$	$1 \cdot 8 + 1$	$1 \cdot 26 +$	$(-2) \cdot 9 =$	8
3	8	$=$	$8 \cdot \mathbf{1} + 0$	$(-1) \cdot 26 +$	$\mathbf{3} \cdot 9 =$	1

- Damit erhalten wir den ursprünglichen Klartextbuchstaben zurück:

$$D(k, y) = b^{-1}(y - c) = 3(V - 2) = \mathtt{F}, \text{ da } 3 \cdot 19 = 57 \equiv_{26} 5 \text{ ist} \quad \triangleleft$$

Die Eulersche Phi-Funktion

- Zur Berechnung der Schlüsselzahl bei der multiplikativen und affinen Chiffre benötigen wir die Eulersche Phi-Funktion $\varphi(m) = \|\mathbb{Z}_m^*\|$

m	1	2	3	4	5	6	7	8	9	10
$\mathbb{Z}_m^*$	$\{0\}$	$\{1\}$	$[2]$	$\{1,3\}$	$[4]$	$\{1,5\}$	$[6]$	$\{1,3,5,7\}$	$\{1,2,4,5,7,8\}$	$\{1,3,7,9\}$
$\varphi(m)$	1	1	2	2	4	2	6	4	6	4

- Für primes p gilt offensichtlich $\varphi(p) = p - 1$, da $\mathbb{Z}_p^* = [p-1]$ ist (für die Menge $\{1, \ldots, n\}$, $n \in \mathbb{N}$, schreiben wir auch kurz $[n]$)
- Wegen $\mathbb{Z}_{p^k} - \mathbb{Z}_{p^k}^* = \{0, p, 2p, \ldots, (p^{k-1} - 1)p\}$ folgt zudem

$$\varphi(p^k) = p^k - p^{k-1} = p^{k-1}(p-1) \text{ für } k \geq 1$$

- Um hieraus eine Formel für $\varphi(n)$ zu erhalten, genügt es, $\varphi(ml)$ im Fall $\mathrm{ggT}(m, l) = 1$ in Abhängigkeit von $\varphi(m)$ und $\varphi(l)$ zu bestimmen
- Hierzu betrachten wir die Abbildung $f : \mathbb{Z}_{ml} \to \mathbb{Z}_m \times \mathbb{Z}_l$ mit

$$f(x) = (x \bmod m, x \bmod l)$$

Eine Berechnungsformel für die Eulersche Phi-Funktion

Beispiel

- Für $m = 5$ und $l = 6$ erhalten wir die Funktion $f : \mathbb{Z}_{30} \to \mathbb{Z}_5 \times \mathbb{Z}_6$ mit

x	0	**1**	2	3	4	5	6	**7**	8	9
$f(x)$	$(0,0)$	$(\mathbf{1},\mathbf{1})$	$(\mathbf{2},2)$	$(\mathbf{3},3)$	$(\mathbf{4},4)$	$(0,\mathbf{5})$	$(\mathbf{1},0)$	$(\mathbf{2},\mathbf{1})$	$(\mathbf{3},2)$	$(\mathbf{4},3)$

x	10	**11**	12	**13**	14	15	16	**17**	18	**19**
$f(x)$	$(0,4)$	$(\mathbf{1},\mathbf{5})$	$(\mathbf{2},0)$	$(\mathbf{3},\mathbf{1})$	$(\mathbf{4},2)$	$(0,3)$	$(\mathbf{1},4)$	$(\mathbf{2},\mathbf{5})$	$(\mathbf{3},0)$	$(\mathbf{4},\mathbf{1})$

x	20	21	22	**23**	24	25	26	27	28	**29**
$f(x)$	$(0,2)$	$(\mathbf{1},3)$	$(\mathbf{2},4)$	$(\mathbf{3},\mathbf{5})$	$(\mathbf{4},0)$	$(0,\mathbf{1})$	$(\mathbf{1},2)$	$(\mathbf{2},3)$	$(\mathbf{3},4)$	$(\mathbf{4},\mathbf{5})$

- Man beachte, dass f eine Bijektion zwischen $\mathbb{Z}_{30}$ und $\mathbb{Z}_5 \times \mathbb{Z}_6$ ist
- Zudem liegt $f(x) = (y, z)$ genau dann in $\mathbb{Z}_5^* \times \mathbb{Z}_6^*$, wenn $x \in \mathbb{Z}_{30}^*$ ist (die Werte $x \in \mathbb{Z}_{30}^*$, $y \in \mathbb{Z}_5^*$ und $z \in \mathbb{Z}_6^*$ sind ***fett*** gedruckt)
- Folglich bildet f die x-Werte in $\mathbb{Z}_{30}^*$ bijektiv auf die Werte in $\mathbb{Z}_5^* \times \mathbb{Z}_6^*$ ab

Eine Berechnungsformel für die Eulersche Phi-Funktion

Beispiel (Schluss)

- Für die Umkehrfunktion $f^{-1} : \mathbb{Z}_5 \times \mathbb{Z}_6 \to \mathbb{Z}_{30}$ erhalten wir nebenstehende Tabelle
- Die fett gedruckten Einträge bilden dann die Tabelle der Einschränkung $\hat{f}^{-1}$ von f^{-1} auf die Menge $\mathbb{Z}_5^* \times \mathbb{Z}_6^*$
- Das Bild dieser Einschränkung ist genau die Menge $\mathbb{Z}_{30}^*$

f^{-1}	0	**1**	2	3	4	**5**
0	0	25	20	15	10	5
1	6	**1**	26	21	16	**11**
2	12	**7**	2	27	22	**17**
3	18	**13**	8	3	28	**23**
4	24	**19**	14	9	4	**29**

◁

Eine Berechnungsformel für die Eulersche Phi-Funktion

- Der Chinesische Restsatz (den wir in Kürze beweisen) besagt, dass die Funktion $f : \mathbb{Z}_{m\ell} \to \mathbb{Z}_m \times \mathbb{Z}_\ell$ mit $f(x) = (x \bmod m, x \bmod \ell)$ im Fall $\mathrm{ggT}(m, \ell) = 1$ bijektiv ist
- Wegen

$$\begin{aligned} \mathrm{ggT}(x, m\ell) = 1 &\Leftrightarrow \mathrm{ggT}(x, m) = \mathrm{ggT}(x, \ell) = 1 \\ &\Leftrightarrow \mathrm{ggT}(x \bmod m, m) = \mathrm{ggT}(x \bmod \ell, \ell) = 1 \end{aligned}$$

ist daher die Einschränkung $\hat{f}$ von f auf den Bereich $\mathbb{Z}^*_{m\ell}$ eine Bijektion zwischen $\mathbb{Z}^*_{m\ell}$ und $\mathbb{Z}^*_m \times \mathbb{Z}^*_\ell$, d.h. es gilt

$$\varphi(m\ell) = \|\mathbb{Z}^*_{m\ell}\| = \|\mathbb{Z}^*_m \times \mathbb{Z}^*_\ell\| = \|\mathbb{Z}^*_m\| \cdot \|\mathbb{Z}^*_\ell\| = \varphi(m)\varphi(\ell)$$

Eine Berechnungsformel für die Eulersche Phi-Funktion

- Als Folgerung aus dem Chinesischen Restsatz erhalten wir nun leicht eine Berechnungsformel für die Phi-Funktion

Satz

Die Eulersche φ-Funktion ist multiplikativ, d. h. für teilerfremde Zahlen m und ℓ gilt $\varphi(m\ell) = \varphi(m)\varphi(\ell)$

Korollar

Sei $m = \prod_{i=1}^{\ell} p_i^{k_i}$ die Primfaktorzerlegung von m. Dann gilt

$$\varphi(m) = \prod_{i=1}^{\ell} p_i^{k_i-1}(p_i - 1) = m \prod_{i=1}^{\ell} (p_i - 1)/p_i$$

Beweis.

Es gilt

$$\varphi(\textstyle\prod_{i=1}^{\ell} p_i^{k_i}) = \prod_{i=1}^{\ell} \varphi(p_i^{k_i}) = \prod_{i=1}^{\ell} (p_i^{k_i} - p_i^{k_i-1}) = \prod_{i=1}^{\ell} p_i^{k_i-1}(p_i - 1)$$ □

Der Chinesische Restsatz

- Jede der beiden linearen Kongruenzen

$$\begin{aligned} x &\equiv_3 0 \\ x &\equiv_6 1 \end{aligned}$$

  ist lösbar
- Es gibt aber kein x, das beide Kongruenzen erfüllt
- Der nächste Satz zeigt, dass unter bestimmten Voraussetzungen gemeinsame Lösungen existieren, und wie sie berechnet werden können

Satz (Chinesischer Restsatz)

Falls $m_1, \ldots, m_k$ paarweise teilerfremd sind, dann hat das System

$$\begin{aligned} x &\equiv_{m_1} b_1 \\ &\vdots \qquad\qquad (*) \\ x &\equiv_{m_k} b_k \end{aligned}$$

für beliebige Zahlen $b_1, \ldots, b_k \in \mathbb{Z}$ genau eine Lösung modulo $m = \prod_{i=1}^{k} m_i$

Der Chinesische Restsatz

Satz (Chinesischer Restsatz)

Falls $m_1, \ldots, m_k$ paarweise teilerfremd sind, dann hat das System

$$\begin{array}{lll} x & \equiv_{m_1} & b_1 \\ & \vdots & \qquad (*) \\ x & \equiv_{m_k} & b_k \end{array}$$

für beliebige Zahlen $b_1, \ldots, b_k \in \mathbb{Z}$ genau eine Lösung modulo $m = \prod_{i=1}^{k} m_i$

Beweis.

- Zu jeder Zahl $n_i = {}^{m}/_{m_i}$ ex. wegen $\mathrm{ggT}(n_i, m_i) = 1$ Zahlen μ_i und λ_i mit
$$\mu_i n_i + \lambda_i m_i = \mathrm{ggT}(n_i, m_i) = 1$$
- Für $i = 1, \ldots, k$ löst daher die Zahl $s_i = \mu_i n_i$ das System
$$x \equiv_{m_j} \begin{cases} 0, & j \neq i \quad (1) \\ 1, & j = i \quad (2) \end{cases}$$

Beweis des Chinesischen Restsatzes

- Zu jeder Zahl $n_i = {}^{m}/_{m_i}$ ex. wegen $\mathrm{ggT}(n_i, m_i) = 1$ Zahlen μ_i und λ_i mit
$$\mu_i n_i + \lambda_i m_i = \mathrm{ggT}(n_i, m_i) = 1$$
- Für $i = 1, \ldots, k$ löst daher die Zahl $s_i = \mu_i n_i$ das System
$$x \equiv_{m_j} \begin{cases} 0, & j \neq i \quad (1) \\ 1, & j = i \quad (2) \end{cases}$$
- Folglich erfüllt $s = \sum_{i=1}^{k} b_i s_i$ für $j = 1, \ldots, k$ die Kongruenz
$$s \overset{(1)}{\equiv}_{m_j} b_j s_j \overset{(2)}{\equiv}_{m_j} b_j,$$
d.h. s löst das System $(*)$
- Dies zeigt, dass die Funktion
$$f : \mathbb{Z}_m \to \mathbb{Z}_{m_1} \times \cdots \times \mathbb{Z}_{m_k} \text{ mit } f(x) = (x \bmod m_1, \ldots, x \bmod m_k)$$
surjektiv ist
- Da der Definitions- und der Wertebereich von f gleich groß sind, muss f auch injektiv sein und $(*)$ ist eindeutig lösbar □

Der Chinesische Restsatz

- Man beachte, dass der Beweis des Chinesischen Restsatzes konstruktiv ist und die Lösung x unter Verwendung des erweiterten Euklidschen Algorithmus' effizient berechnet werden kann
- Man verifiziert auch leicht, dass f ein Isomorphismus zwischen dem Restklassenring $(\mathbb{Z}_m, \oplus_m, \odot_m)$ und dem direkten Produkt der Ringe $(\mathbb{Z}_{m_i}, \oplus_{m_i}, \odot_{m_i})$, $1 \leq i \leq k$, ist
- Dies ist nicht nur für theoretische Überlegungen nützlich, sondern hat auch praktische Konsequenzen
- Beispielsweise lässt sich so die Laufzeit von bestimmten Berechnungen im Ring $\mathbb{Z}_m$ deutlich reduzieren, sofern die Primzahlzerlegung von m bekannt ist

Die Hill-Chiffre

- Die von Hill im Jahr 1929 publizierte Chiffre ist eine Erweiterung der multiplikativen Chiffre auf Buchstabenblöcke
- Der Klartext wird also nicht zeichen- sondern blockweise verarbeitet
- Die Blöcke haben eine feste Länge ℓ und sowohl Klar- als auch Kryptotextraum bestehen aus allen Wörtern $x \in A^\ell$
- Als Schlüssel dient eine $(\ell \times \ell)$-Matrix $k = (k_{ij})$ mit Koeffizienten in $\mathbb{Z}_m$
- Diese transformiert einen Klartext $x = x_1 \dots x_\ell \in A^\ell$ in den Kryptotext $y = y_1 \dots y_\ell$ mit $y_i = x_1 k_{1i} + \dots + x_\ell k_{\ell i}$ für $i = 1, \dots, \ell$:

$$(y_1 \ \cdots \ y_\ell) = (x_1 \ \cdots \ x_\ell) \begin{pmatrix} k_{11} & \dots & k_{1\ell} \\ \vdots & \ddots & \vdots \\ k_{\ell 1} & \dots & k_{\ell\ell} \end{pmatrix}$$

- Wir bezeichnen die Menge aller $(\ell \times \ell)$-Matrizen (k_{ij}) mit Koeffizienten $k_{ij} \in \mathbb{Z}_m$ mit $\mathbb{Z}_m^{\ell \times \ell}$

Die Hill-Chiffre

- Damit der Chiffriervorgang injektiv ist, muss k invertierbar sein
- Dies lässt sich an der Determinante von k erkennen

Definition

- Sei R ein kommutativer Ring mit Eins und sei $A = (a_{ij}) \in R^{n \times n}$
- Eine Funktion $f : R^{n \times n} \to R$ heißt Determinantenfunktion, falls sie folgende drei Eigenschaften erfüllt:
  - f ist ***multilinear***, d.h. für jede Matrix $A = (a_1, \ldots, a_n) \in R^{n \times n}$ mit Spalten $a_1, \ldots, a_n \in (R^n)^T$, jeden Spaltenvektor $b \in (R^n)^T$ und jedes Element $r \in R$ gilt

$$f(a_1 \ldots r \cdot a_i + b \ldots a_n) = r \cdot f(a_1 \ldots a_i \ldots a_n) + f(a_1 \ldots b \ldots a_n)$$

  - f ist ***alternierend***, d.h. im Fall $a_i = a_j$ für $i \neq j$ gilt $f(a_1, \ldots, a_n) = 0$
  - f ist ***normiert***, d.h. $f(E) = 1$, wobei E die Einheitsmatrix ist

- Tatsächlich ist f durch diese drei Eigenschaften eindeutig festgelegt und wir bezeichnen $f(A)$ wie üblich mit $\det(A)$

Die Hill-Chiffre

- Eine explizite Darstellung für die Determinantenfunktion liefert der laplacesche Entwicklungssatz
- Für $1 \leq i, j \leq n$ sei A_{ij} die durch Streichen der i-ten Zeile und j-ten Spalte aus A hervorgehende Matrix
- Dann ist $\det(A) = a_{11}$, falls $n = 1$, und für $n > 1$ ist

$$\det(A) = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det(A_{ij}),$$

  wobei $i \in [n]$ beliebig wählbar ist (Entwicklung nach der i-ten Zeile)
- Das Produkt $(-1)^{i+j} \det(A_{ij})$ wird Kofaktor genannt und mit $\tilde{a}_{ij}$ bezeichnet
- Aus dieser Formel lässt sich zwar ein Algorithmus zur Berechnung der Determinante ableiten, allerdings hat dieser eine exponentielle Laufzeit
- Das Gauß-Verfahren führt dagegen auf eine effiziente Berechnungsmethode für die Determinante (siehe Übungen)

Die Hill-Chiffre

- Für die Dechiffrierung eines mit der Schlüsselmatrix k berechneten Kryptotextes wird die inverse Matrix k^{-1} benötigt
- Invertierbare Matrizen werden auch als regulär bezeichnet
- Eine Matrix $k \in \mathbb{Z}_m^{\ell \times \ell}$ ist genau dann regulär, wenn $\mathrm{ggT}(\det(k), m) = 1$ ist (siehe Übungen)
- In diesem Fall lässt sich k^{-1} mit dem Gauß-Jordan-Algorithmus effizient berechnen (siehe Übungen)

Definition

- Sei $A = \{a_0, \ldots, a_{m-1}\}$ ein beliebiges Alphabet und für eine natürliche Zahl $\ell \geq 2$ sei $M = C = A^\ell$
- Bei der ***Hill-Chiffre*** ist $K = \{k \in \mathbb{Z}_m^{\ell \times \ell} \mid \mathrm{ggT}(\det(k), m) = 1\}$ und es gilt

$$E(k, x) = xk \quad \text{und} \quad D(k, y) = yk^{-1}$$

Die Hill-Chiffre

Beispiel

- Wir benutzen zur Chiffrierung von Klartextblöcken der Länge $\ell = 4$ über dem lateinischen Alphabet A_{lat} die Schlüsselmatrix

$$k = \begin{pmatrix} 11 & 13 & 8 & 21 \\ 24 & 17 & 3 & 25 \\ 18 & 12 & 23 & 17 \\ 6 & 15 & 2 & 15 \end{pmatrix}$$

- Diese überführt den Klartext $\mathtt{HILL}$ in den Kryptotext $E_k(\mathtt{HILL}) = \mathit{NERX}$:

$$(\mathtt{H\,I\,L\,L}) \begin{pmatrix} 11 & 13 & 8 & 21 \\ 24 & 17 & 3 & 25 \\ 18 & 12 & 23 & 17 \\ 6 & 15 & 2 & 15 \end{pmatrix} = (\mathit{N\,E\,R\,X}) \quad \text{bzw.} \quad \begin{aligned} 11\,\mathtt{H} + 24\,\mathtt{I} + 18\,\mathtt{L} + 6\,\mathtt{L} &= \mathit{N} \\ 13\,\mathtt{H} + 17\,\mathtt{I} + 12\,\mathtt{L} + 15\,\mathtt{L} &= \mathit{E} \\ 8\,\mathtt{H} + 3\,\mathtt{I} + 23\,\mathtt{L} + 2\,\mathtt{L} &= \mathit{R} \\ 21\,\mathtt{H} + 25\,\mathtt{I} + 17\,\mathtt{L} + 15\,\mathtt{L} &= \mathit{X} \end{aligned}$$

- Für die Entschlüsselung wird die inverse Matrix k^{-1} benötigt, die wir in den Übungen berechnen ◁

Die Vigenère-Chiffre

- Die Vigenère-Chiffre ersetzt den Klartext zeichenweise, allerdings je nach Position im Klartext unterschiedlich
- Sie ist nach dem Franzosen Blaise de Vigenère (1523–1596) benannt

Definition

- Sei $A = B$ ein beliebiges Alphabet
- Die ***Vigenère-Chiffre*** chiffriert unter einem Schlüssel $k = k_0 \dots k_{d-1}$ in $K = A^*$ einen Klartext $x = x_0 \dots x_{n-1}$ beliebiger Länge zu
  - $E(k, x) = y_0 \dots y_{n-1}$ mit $y_i = x_i + k_{(i \bmod d)}$ für $i = 1, \dots, n-1$ und dechiffriert einen Kryptotext $y = y_0 \dots y_{n-1}$ zu
  - $D(k, y) = x_0 \dots x_{n-1}$ mit $x_i = y_i - k_{(i \bmod d)}$ für $i = 1, \dots, n-1$

Die Vigenère-Chiffre

Beispiel

Das Schlüsselwort $k = \mathit{WIE}$ überführt den Klartext VIGENERE beispielsweise der Kryptotext

$$E(\mathit{WIE}, \mathtt{VIGENERE}) = \underbrace{\mathtt{V}+\mathit{W}}_{R}\ \underbrace{\mathtt{I}+\mathit{I}}_{Q}\ \underbrace{\mathtt{G}+\mathit{E}}_{K}\ \underbrace{\mathtt{E}+\mathit{W}}_{A}\ \underbrace{\mathtt{N}+\mathit{I}}_{V}\ \underbrace{\mathtt{E}+\mathit{E}}_{I}\ \underbrace{\mathtt{R}+\mathit{W}}_{N}\ \underbrace{\mathtt{E}+\mathit{I}}_{M}$$

$$= \mathit{RQKAVINM}$$

◁

- Um einen Klartext x zu verschlüsseln, wird also das Schlüsselwort $k = k_0 \ldots k_{d-1}$ so oft wiederholt, bis der dabei entstehende Schlüsselstrom $\hat{k} = k_0 k_1 \ldots k_{d-1} k_0 k_1 \ldots k_{d-1} k_0 k_1 \ldots$ die Länge von x erreicht
- Dann werden x und $\hat{k}$ zeichenweise addiert, um den zugehörigen Kryptotext y zu bilden
- Aus diesem kann der ursprüngliche Klartext x zurückgewonnen werden, indem man den Schlüsselstrom $\hat{k}$ wieder subtrahiert

Die Vigenère-Chiffre

Beispiel

Chiffrierung:

$$\begin{array}{rll} & \texttt{VIGENERE} & \text{(Klartext } x\text{)} \\ + & \underline{\textit{WIEWIEWI}} & \text{(Schlüsselstrom } \hat{k}\text{)} \\ & \textit{RQKAVINM} & \text{(Kryptotext } y\text{)} \end{array}$$

Dechiffrierung:

$$\begin{array}{rll} & \textit{RQKAVINM} & \text{(Kryptotext } y\text{)} \\ - & \underline{\textit{WIEWIEWI}} & \text{(Schlüsselstrom } \hat{k}\text{)} \\ & \texttt{VIGENERE} & \text{(Klartext } x\text{)} \end{array}$$

◁

Die Chiffrierarbeit lässt sich durch Benutzung einer Additionstabelle erleichtern (auch als Vigenère-Tableau bekannt):

+	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
⋮																										
X	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
Y	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	X
Z	Z	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

Die Beaufort-Chiffre

- Um eine involutorische Chiffre zu erhalten, schlug Sir Francis Beaufort, ein Admiral der britischen Marine, vor, den Schlüsselstrom nicht auf den Klartext zu addieren, sondern letzteren von ersterem zu subtrahieren

Beispiel

Verschlüsseln wir den Klartext `BEAUFORT` beispielsweise unter dem Schlüsselwort $k = \mathit{WIE}$, so erhalten wir den Kryptotext *XMEQNSNB*. Eine erneute Verschlüsselung liefert wieder den Klartext `BEAUFORT`.

Chiffrierung:

	WIEWIEWI	(Schlüsselstrom)
−	BEAUFORT	(Klartext)
	VEECDQFP	(Kryptotext)

Dechiffrierung:

	WIEWIEWI	(Schlüsselstrom)
−	*VEECDQFP*	(Kryptotext)
	BEAUFORT	(Klartext)

◁

Die Autokey-Chiffre

- Die bisher betrachteten Chiffren erzeugen einen ***periodischen Schlüsselstrom*** $\hat{k} = \hat{k}_0 \dots \hat{k}_{n-1}$, das heißt, es gilt $\hat{k}_i = \hat{k}_{i+d}$ für eine feste Zahl d
- Da dadurch Angriffe erleichtert werden, sollte entweder eine sehr große Periode oder besser ein ***fortlaufender Schlüsselstrom*** benutzt werden
- Ein Möglichkeit besteht darin, an das Schlüsselwort den Klartext oder den Kryptotext anzuhängen (sogenannte Autokey-Chiffrierung)

Beispiel

Die Autokey-Chiffre erzeugt mit dem Schlüsselwort *WIE* aus dem Klartext VIGENERE folgende Kryptotexte.

Klartext-Schlüsselstrom:

	VIGENERE	(Klartext)
+	*WIEVIGEN*	(Schlüsselstrom)
	RQKZVKVR	(Kryptotext)

Kryptotext-Schlüsselstrom:

	VIGENERE	(Klartext)
+	*WIERQKVD*	(Schlüsselstrom)
	RQKVDOMH	(Kryptotext)

◁

Die Autokey-Chiffre

Beispiel

Die Autokey-Chiffre erzeugt mit dem Schlüsselwort *WIE* aus dem Klartext VIGENERE folgende Kryptotexte.

Klartext-Schlüsselstrom:

	VIGENERE	(Klartext)
+	*WIEVIGEN*	(Schlüsselstrom)
	RQKZVKVR	(Kryptotext)

Kryptotext-Schlüsselstrom:

	VIGENERE	(Klartext)
+	*WIERQKVD*	(Schlüsselstrom)
	RQKVDOMH	(Kryptotext)

◁

Auch die Dechiffrierung ist in beiden Fällen einfach:

- Im ersten Fall kann der Empfänger durch Subtraktion des Schlüsselworts den Anfang des Klartextes bilden und so den Schlüsselstrom verlängern
- Noch einfacher ist die Dechiffrierung im zweiten Fall, da sich hier der Schlüsselstrom vom Kryptotext nur durch das vorangestelle Schlüsselwort unterscheidet

Der One-Time-Pad

- Man kann auch einen zuvor vereinbarten Text aus einem Buch als aperiodischen Schlüsselstrom verwenden (Lauftextverschlüsselung)
- Besser ist es jedoch, mit einem Pseudozufallsgenerator aus einem relativ kurzen Schlüssel einen deutlich längeren Schlüsselstrom zu erzeugen
- Noch besser ist es, den Schlüsselstrom wirklich zufällig zu erzeugen.
- Dies führt auf eine absolut sichere Verschlüsselung, sofern der Schlüsselstrom nicht mehrmals benutzt wird
- Ein solcher „Wegwerfschlüssel“ (engl. One-Time-Pad oder kurz OTP; im Deutschen auch als individueller Schlüssel bezeichnet) lässt sich für längere Klartexte allerdings nur mit großem Aufwand generieren und auf einem sicheren Kanal zwischen Sender und Empfänger verteilen
- Der OTP wurde beispielsweise beim „heißen Draht“, der 1963 eingerichteten, direkten Fernschreibverbindung zwischen dem Weißen Haus in Washington und dem Kreml in Moskau, angewandt

Der One-Time-Pad

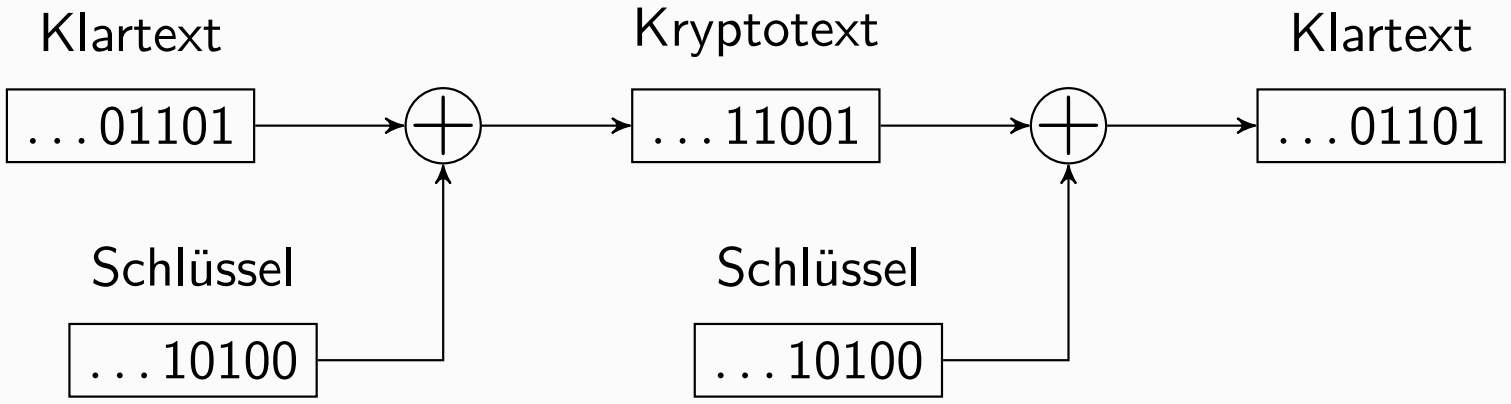
Beispiel

Sei $A = \{a_0, \ldots, a_{m-1}\}$ ein beliebiges Klartextalphabet. Um einen Klartext $x = x_0 \ldots x_{n-1}$ zu verschlüsseln, wird auf jedes Klartextzeichen x_i ein neuer, zufällig generierter Schlüsselbuchstabe k_i addiert,

$$y = y_0 \ldots y_{n-1}, \text{ wobei } y_i = x_i + k_i$$

◁

- Der Klartext wird also wie bei einer additiven Chiffre verschlüsselt, nur dass der Schlüssel nach einmaligem Gebrauch gewechselt wird
- Wie diese ist der One-Time-Pad im Binärfall involutorisch

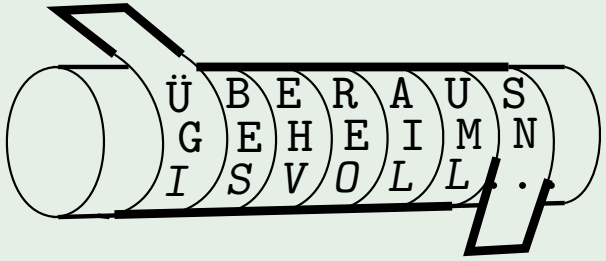


Klassifikation von Kryptosystemen

- Bei den bisher betrachteten Chiffren handelt es sich um Substitutionen
- Diese ersetzen Klartextzeichen – einzeln oder blockweise – durch Kryptotextsegmente
- Dagegen verändern Transpositionen lediglich die *Reihenfolge* der einzelnen Klartextzeichen

Beispiel (Skytale-Chiffre)

- Die älteste bekannte Verschlüsselungstechnik stammt aus der Antike und wurde im 5. Jahrhundert v. Chr. von den Spartanern entwickelt
- Der Sender wickelt einen Papierstreifen spiralförmig um einen Holzstab (die sog. ***Skytale***) und beschreibt ihn mit der Geheimbotschaft:



ÜBERAUS GEHEIMNISVOLL ...

$\rightsquigarrow$ *ÜGI ... BES ... EHV ... REO ... AIL ... UML ... SN ...*

- Der Empfänger benötigt einen Stab mit dem gleichen Umfang, um den Kryptotext auf dem Papierstreifen wieder zu entziffern ◁

Skytale und Spaltentransposition

- Als Schlüssel fungiert hier also der Stabumfang bzw. die Anzahl k der Zeilen, mit denen der Stab beschrieben wird
- Ist die Länge des Klartextes x ein Vielfaches von k, so gilt

$$E(k, x_1 \cdots x_{km}) = x_1 x_{m+1} \cdots x_{(k-1)m+1}\ x_2 x_{m+2} \cdots x_{(k-1)m+2}\ \cdots\ x_m x_{2m} \cdots x_{km}$$

- Dasselbe Resultat ergibt sich, wenn man x zeilenweise in eine $(k \times m)$-Matrix schreibt und spaltenweise ausliest (sog. ***Spaltentransposition***):

$$\begin{array}{cccc} x_1 & x_2 & \cdots & x_m \\ x_{m+1} & x_{m+2} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(k-1)m+1} & x_{(k-1)m+2} & \cdots & x_{km} \end{array}$$

Skytale und Spaltentransposition

- Ist die Länge von x kein Vielfaches von k, so kann man x durch Ein- bzw. Anfügen von sog. ***Blendern*** (Füllzeichen) verlängern
- Damit der Empfänger diese Füllzeichen nach der Entschlüsselung wieder entfernen kann, ist lediglich darauf zu achten, dass sie nach der Entschlüsselung als solche erkennbar sind
- Von der Methode, die letzte Zeile nur teilweise zu beschriften, ist dagegen abzuraten
- In diesem Fall würden nämlich auf dem abgewickelten Papierstreifen Lücken entstehen, die Rückschlüsse auf den benutzten Schlüssel erlauben würden
- Sicherer ist es, wenn der Sender die letzte Spalte auf der Skytale nur teilweise beschriftet

Die Zick-Zack-Transposition

- Eng verwandt mit der Skytale-Chiffre ist die Zick-Zack-Transposition

Beispiel

Bei Ausführung einer ***Zick-Zack-Transposition*** wird der Klartext in eine Zick-Zack-Linie geschrieben und horizontal wieder ausgelesen. Die Höhe der Zick-Zack-Linie dient als Schlüssel.

```
Z     Z     L     E
  I K   A K   I I
    C     C     N
```

ZICKZACKLINIE $\rightsquigarrow$ *ZZLEIKAKIICCN*

◁

- Hierbei werden Zeichen im vorderen Klartextbereich bis fast ans Ende des Kryptotextes verschoben und umgekehrt
- Ein Nachteil hiervon ist, dass bei der Erzeugung des Kryptotextes der gesamte Klartext gepuffert werden muss
- Daher werden meist Blocktranspositionen verwendet, die Klartextzeichen nur innerhalb fester Blockgrenzen transponieren

Die Blocktransposition

Definition

- Sei $A = B$ ein Alphabet und für eine Zahl $\ell \geq 2$ sei $M = C = A^\ell$
- Eine ***Blocktransposition*** ordnet jedem Schlüssel $k \in K$ eine Permutation $\pi \in S_\ell$ zu, so dass für alle $x_1 \cdots x_\ell \in M$ und $y_1 \cdots y_\ell \in C$ gilt:

$$E(k, x_1 \cdots x_\ell) = x_{\pi(1)} \cdots x_{\pi(\ell)} \text{ und } D(k, y_1 \cdots y_\ell) = y_{\pi^{-1}(1)} \cdots y_{\pi^{-1}(\ell)}$$

Beispiel

Eine Skytale, die mit 4 Zeilen der Länge 6 beschrieben wird, realisiert beispielsweise die Blocktransposition

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
$\pi(i)$	1	7	13	19	2	8	14	20	3	9	15	21	4	10	16	22	5	11	17	23	6	12	18	24

◁

Die Blocktransposition

- Für die Entschlüsselung wird die ***inverse Permutation*** π^{-1} benutzt
- Diese lässt sich leicht berechnen, wenn π durch eine Folge von Zyklen $(i_1\ i_2\ i_3\ \ldots\ i_n)$ dargestellt wird; dies bedeutet, dass i_1 auf i_2, i_2 auf i_3, $\ldots$ und schließlich i_n auf i_1 abgebildet wird
- Dabei werden meist Einerzyklen weggelassen, die Zyklen nach der Größe ihrer kleinsten Elemente sortiert und letztere an den Anfang gesetzt

Beispiel

i	1	2	3	4	5	6
$\pi(i)$	4	6	1	3	5	2

i	1	2	3	4	5	6
$\pi^{-1}(i)$	3	6	4	1	5	2

- Obiges π hat beispielsweise die Zyklendarstellung

  $$\pi = (1\ 4\ 3)\,(2\ 6)\,(5) \text{ oder } \pi = (1\ 4\ 3)\,(2\ 6)$$

- Daraus erhalten wir unmittelbar

  $$\pi^{-1} = (3\ 4\ 1)\,(6\ 2) \text{ oder } (1\ 3\ 4)\,(2\ 6)$$

◁

Die Matrix-Transposition

Beispiel

- Bei der ***Matrix-Transposition*** wird der Klartext zeilenweise in eine $k \times l$-Matrix eingelesen und der Kryptotext spaltenweise gemäß einer Spaltenpermutation $\pi \in S_l$, die als Schlüssel dient, wieder ausgelesen
- Für $\pi = (1\ 4\ 3)\,(2\ 6)$ wird also zuerst Spalte $\pi(1) = 4$, dann Spalte $\pi(2) = 6$ und zuletzt Spalte $\pi(6) = 2$ ausgelesen:

3	6	4	1	5	2
D	I	E	S	E	R
K	L	A	R	T	E
X	T	I	S	T	N
I	C	H	T	S	E
H	R	L	A	N	G

DIESER KLARTEXT IST NICHT SEHR LANG

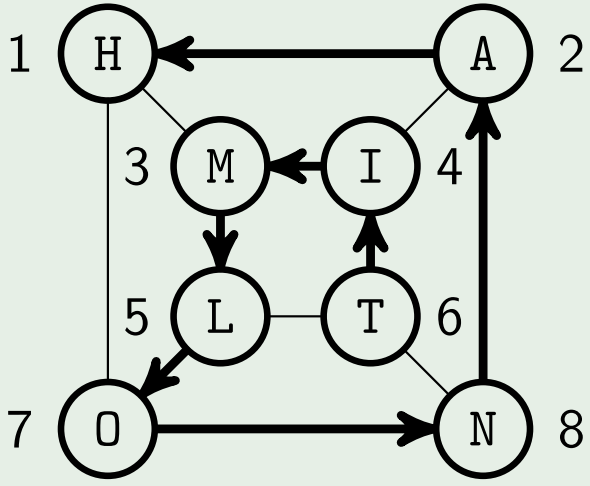
$\rightsquigarrow$ *SRSTA RENEG DKXIH EAIHL ETTSN ILTCR*

◁

Die Weg-Transposition

Beispiel

- Bei der ***Weg-Transposition*** wird als Schlüssel eine Hamiltonlinie in einem Graphen mit den Knoten $1, \ldots, \ell$ benutzt
- Der Klartext $x_1 \cdots x_\ell$ wird gemäß der Knotennumerierung eingelesen und der Kryptotext entlang der Hamiltonlinie wieder ausgelesen:



HAMILTON $\rightsquigarrow$ *TIMLONAH*

- Es ist leicht zu sehen, dass sich jede Blocktransposition durch eine Hamiltonlinie in einem geeigneten Graphen realisieren lässt
- Der Vorteil ist, dass man sich den Verlauf einer Hamiltonlinie in einem Graphen besser einprägen kann als eine Zahlenfolge ◁

Schlüsselphrasen

- Sehr beliebt ist auch die Methode, sich eine Permutation in Form eines ***Schlüsselworts*** (oder einer ***Schlüsselphrase***) zu merken
- Die zugehörige Permutation σ erhält man, indem man das Wort auf Papier schreibt und in der Zeile darunter die Zeichen abzählt:

Schlüsselwort für σ	C	A	E	S	A	R
i	1	2	3	4	5	6
$\sigma(i)$	3	1	4	6	2	5
Zyklendarstellung von σ	(1 3 4 6 5 2)					

DIE BLOCKLAENGE IST SECHS $\rightsquigarrow$
EDBOIL LCANKE IGSSET EXCSYH

- In der nächsten Zeile werden dann die Zeichen gemäß ihrer alphabetischen Reihenfolge abgezählt
- Auf diese Weise erhält man die Wertetabelle von σ
- Mehrfach vorkommende Zeichen können entweder gestrichen oder gemäß ihrer Position im Schlüsselwort hochgezählt werden
- Im ersten Fall erhält man eine entsprechend kleinere Blocklänge; in obigem Beispiel wäre dann $\ell = 5$ und $\sigma = (1\,2)(4\,5)$

Klassifikation von Substitutionen

- Ein wichtiges Unterscheidungsmerkmal ist die Länge der Klartext- und Kryptotexteinheiten, mit denen eine Substitution operiert

Definition

- Eine Substitution, die Einzelzeichen ersetzt, heißt ***monografisch***, andernfalls ***polygrafisch***
- Eine Substitution, die Klartextsegmente durch Einzelzeichen ersetzt, heißt ***monopartit***, andernfalls ***multipartit***

- Operiert eine Substitution auf Zeichenpaaren, heißt sie ***digrafisch***
- Wird der Kryptotext aus Zeichenpaaren gebildet, heißt sie ***bipartit***

Die Porta-Chiffre

- Das älteste bekannte polygrafische Chiffrierverfahren wurde von Giovanni Porta im Jahr 1563 veröffentlicht
- Dabei werden je zwei aufeinanderfolgende Klartextzeichen durch ein einzelnes Kryptotextzeichen ersetzt (d.h. die Chiffre ist monopartit)

Beispiel

- Bei der ***Porta-Chiffre*** werden 400 (!) unterschiedliche von Porta für diesen Zweck entworfene Kryptotextzeichen verwendet
- Diese sind in einer (20×20)-Matrix $M = (y_{ij})$ angeordnet, deren Zeilen und Spalten mit den 20 Buchstaben $\mathtt{A}, \ldots, \mathtt{I}, \mathtt{L}, \ldots, \mathtt{T}, \mathtt{V}, \mathtt{Z}$ indiziert sind
- Zur Ersetzung eines Zeichenpaars $a_i a_j$ im Klartext wird das in Zeile i und Spalte j befindliche Kryptotextzeichen $E(M, a_i a_j) = y_{ij}$ benutzt ◁

Die Polybios-Chiffre

- Ein frühes (monografisches) Beispiel einer bipartiten Chiffriermethode geht auf Polybios (circa 200 – 120 v. Chr.) zurück:

	0	*1*	*2*	*3*	*4*
0	A	B	C	D	E
1	F	G	H	I	J
2	K	L	M	N	O
3	P	Q	R	S	T
4	U	V	W	X/Y	Z

POLYBIOS
$\rightsquigarrow$ *30 24 21 43 01 13 24 33*

- Die ***Polybios-Chiffre*** benutzt als Schlüssel eine 5×5-Matrix, deren Einträge aus den Klartextzeichen bestehen
- Jedes Zeichen des Klartextes wird durch sein Koordinatenpaar ij in der Matrix ersetzt
- Der Kryptotextraum besteht also aus den Ziffernpaaren $\{00, 01, \ldots, 44\}$

Klassifikation von Substitutionen

- Ob bei der Ersetzung der Klartextsegmente eine fixe oder variable Strategie verfolgt wird, führt auf ein weiteres Unterscheidungsmerkmal

Definition

- Ersetzt eine Substitution Segmente unabhängig von ihrer Position im Klartext, so heißt sie ***monooalphabetisch***, sonst ***polyalphabetisch***

- Die Ersetzungsregel einer polyalphabetischen Substitution variiert also in Abhängigkeit von den bereits verarbeiteten Klartextsegmenten
- Die Bezeichnung „monoalphabetisch“ ist darauf zurückzuführen, dass sich die Ersetzungsregel im monografischen Fall für jeden Schlüssel durch ein einzelnes Alphabet charakterisieren lässt
- So wird etwa die Caesarchiffre mit dem Schlüssel $k = 3$ durch das Alphabet $\{D, E, F, G, W, \ldots, Y, Z, A, B, C\}$ beschrieben
- Dagegen kommen bei der Vigenère-Chiffre periodisch mehrere verschiedene Ersetzungsalphabete zur Anwendung

Blockchiffren und einfache Substitutionschiffren

- Monoalphabetische Chiffrierverfahren ersetzen meist Texteinheiten einer festen Länge $\ell \geq 1$ durch Kryptotextsegmente der Länge ℓ

Definition

- Sei A ein beliebiges Alphabet und es gelte $M = C = A^\ell$, $\ell \geq 1$
- Eine ***Blockchiffre*** realisiert für jeden Schlüssel $k \in K$ eine bijektive Abbildung g auf A^ℓ und es gilt für alle $x \in M$ und $y \in C$,

$$E(k, x) = g(x) \quad \text{und} \quad D(k, y) = g^{-1}(y).$$

- Im Fall $\ell = 1$ spricht man auch von einer ***einfachen Substitution***

Stromchiffren

- Auch polyalphabetische Chiffren verwenden oft eine feste Blocklänge ℓ
- Da diese jedoch meist relativ klein ist (meist $\ell = 1$), nennt man die einzelnen Segmente nicht ‚Blöcke' sondern ‚Zeichen' und spricht von ***sequentiellen Chiffren*** oder ***Stromchiffren***

Definition

- Sei A ein beliebiges Alphabet und sei $M = C = A^\ell$ für eine Zahl $\ell \geq 1$
- Weiterhin seien K und $\hat{K}$ Schlüsselräume
- Eine ***Stromchiffre*** besteht aus einer Verschlüsselungsfunktion $E : \hat{K} \times M \to C$ und einem ***Schlüsselstromgenerator*** $g : K \times A^* \to \hat{K}$
- Der Generator g erzeugt aus einem ***externen*** Schlüssel $k \in K$ für einen Klartext $x = x_0 \dots x_{n-1}$, $x_i \in M$, eine Folge $\hat{k}_0, \dots, \hat{k}_{n-1}$ von ***internen*** Schlüsseln $\hat{k}_i = g(k, x_0 \dots x_{i-1}) \in \hat{K}$, unter denen x in den Kryptotext

$$E_g(k, x) = E(\hat{k}_0, x_0) \dots E(\hat{k}_{n-1}, x_{n-1})$$

überführt wird

Stromchiffren

- Der interne Schlüsselraum $\hat{K}$ einer Stromchiffre kann also wie der Schlüsselraum einer Blockchiffre eine maximale Größe von $(m^\ell)!$ annehmen
- Im häufigen Spezialfall $\ell = 1$ hat $\hat{K}$ also die maximale Größe $m!$
- Die Aufgabe des Schlüsselstromgenerators g ist es, aus dem externen Schlüssel $k \in K$ und dem bereits verarbeiteten Klartext $x_0 \dots x_{i-1}$ den aktuellen internen Schlüssel $\hat{k}_i$ zu berechnen

Bisher betrachtete Stromchiffren

Stromchiffre	Chiffrierfunktion	Schlüsselstromgenerator
Vigenère	$E(\hat{k}, x) = x + \hat{k}$	$g(k_0 \dots k_{d-1}, x_0 \dots x_{i-1}) = k_{(i \bmod m)}$
Beaufort	$E(\hat{k}, x) = \hat{k} - x$	$g(k_0 \dots k_{d-1}, x_0 \dots x_{i-1}) = k_{(i \bmod m)}$
Autokey mit Klartext-Schlüsselstrom	$E(\hat{k}, x) = x + \hat{k}$	$g(k_0 \dots k_{d-1}, x_0 \dots x_{i-1}) = \begin{cases} k_i, & i < d \\ x_{i-d}, & i \geq d \end{cases}$
Autokey mit Kryptotext-Schlüsselstrom	$E(\hat{k}, x) = x + \hat{k}$	$g(k_0 \dots k_{d-1}, x_0 \dots x_{i-1}) = \begin{cases} k_i, & i < d \\ y_{i-d}, & i \geq d \end{cases}$ $= k_{(i \bmod d)} + \sum_{j=1}^{\lfloor i/d \rfloor} x_{i-jd}$

- Bei der Vigenère- und Beaufortchiffre hängt der Schlüsselstrom nicht vom Klartext, sondern nur vom externen Schlüssel k ab, d.h. sie sind ***synchron***
- Die *Autokey*-Chiffren sind dagegen ***asynchron*** (und aperiodisch)

Gespreizte Substitutionen

- Bei den bisher betrachteten Substitutionen haben die einzelnen Segmente, aus denen der Kryptotext gebildet wird, die gleiche Länge
- Es liegt nahe, einen Angriff zu erschweren, indem man die Länge der Kryptotextsegmente variiert (auch ***Spreizung*** oder ***straddling*** genannt)
- Diese Technik wurde etwa von der ehemaligen sowjetischen Geheimpolizei NKWD benutzt

Beispiel

- Bei der ***Spionage-Chiffre*** wird in die erste Zeile einer 3×10-Matrix ein Schlüsselwort w geschrieben, welches kein Zeichen mehrfach enthält und eine Länge von 6 bis 8 Zeichen hat (also zum Beispiel *SPIONAGE*)
- Danach werden die anderen beiden Zeilen der Matrix mit den restlichen Klartextzeichen (etwa in alphabetischer Reihenfolge) gefüllt

	4	1	9	6	0	3	2	7	5	8
	S	*P*	*I*	*O*	*N*	*A*	*G*	*E*		
8	B	C	D	F	H	J	K	L	M	Q
5	R	T	U	V	W	X	Y	Z		

GESPREIZT
$\rightsquigarrow$ *2741547957 51*

Gespreizte Substitutionen

- Man überzeugt sich leicht davon, dass sich die Kryptotexte der Spionage-Chiffre wieder eindeutig dechiffrieren lassen
- Die verwendeten Kryptotextsegmente 1, 2,. . . , 8, 01, 02, . . . , 08, 91, 92, . . . , 98 erfüllen nämlich die

  Fano-Bedingung: Keines ist das Anfangsstück eines anderen
- Da die Nummern 5 und 8 der beiden letzten Spalten der Matrix auch als Zeilennummern verwendet werden, erklärt dies auch, warum die letzten beiden Einträge der ersten Zeile leer bleiben müssen

Verwendung von Blendern und Homophonen

- Die Verwendung von gespreizten Chiffren zielt offenbar darauf ab, die „***Fuge***“ zwischen den einzelnen Kryptotextsegmenten zu verdecken
- Dennoch bietet die Spionage-Chiffre noch genügend Angriffsfläche für eine unbefugte Dechiffrierung, da im Klartext häufig vorkommende Wortmuster auch im Kryptotext zu Textwiederholungen führen
- Diese Muster können durch das Einstreuen von ***Blendern*** in den Klartext aufgebrochen werden
- Abgesehen davon, dass das Entfernen der Blender auch für den Empfänger mit Mühe verbunden ist, führt diese Methode auch zu einer Expansion des Kryptotextes
- Ist man bereit, dies in Kauf zu nehmen, so gibt es noch eine wirksamere Methode, die Übertragung struktureller und statistischer Klartextmerkmale auf den Kryptotext abzumildern, nämlich die Verwendung von ***Homophonen***

Verwendung von Homophonen

- Die Idee dabei ist, für die Ersetzung eines Klartextzeichens a nicht nur eines, sondern eine Menge $H(a)$ von Chiffrezeichen vorzusehen, und jeweils eines auszuwählen (am besten zufällig)
- Da die Zeichen in $H(a)$ für dasselbe Klartextzeichen stehen, werden sie auch ***Homophone*** genannt

Definition

- Sei A ein Klartextalphabet und sei $M = A$
- Weiter sei C ein Kryptotextraum der Größe $\|C\| > \|A\| = m$
- In einer ***homophonen Substitution*** beschreibt jeder Schlüssel $k \in K$ eine Zerlegung von C in m disjunkte Mengen $H(a)$, $a \in A$
- Um ein Zeichen $a \in A$ unter k zu chiffrieren, wird nach einer bestimmten Methode ein Homophon $y \in H(a)$ gewählt und für a eingesetzt

Verwendung von Homophonen

- Durch den Einsatz einer homophonen Substitution wird also erreicht, dass verschiedene Vorkommen eines Klartextzeichens auch auf unterschiedliche Weise ersetzt werden können
  (ähnlich wie bei einer polyalphabetischen Chiffre)
- Damit der Empfänger den Kryptotext auch wieder eindeutig dechiffrieren kann, dürfen sich die Homophonmengen zweier verschiedener Klartextzeichen aber nicht überlappen
- Daher kann es nicht vorkommen, dass zwei verschiedene Klartextzeichen durch dasselbe Geheimtextzeichen ersetzt werden
- Man beachte, dass der Chiffriervorgang $x \mapsto E(k, x)$ nicht funktional ist, da derselbe Klartext x in mehrere verschiedene Kryptotexte y übergehen kann

Verwendung von Homophonen

- Durch eine geringfügige Modifikation der Polybios-Chiffre lässt sich die folgende bipartite homophone Chiffre erhalten

Beispiel (homophone Substitution)

- Sei $A = \{\mathtt{A}, \ldots, \mathtt{Z}\}$, $B = \{0, \ldots, 9\}$ und $C = \{00, \ldots, 99\}$

	1,0	*2,9*	*3,8*	*4,7*	*5,6*
1,6	A	F	K	P	U
2,7	B	G	L	Q	V
3,8	C	H	M	R	W
4,9	D	I	N	S	X/Y
5,0	E	J	O	T	Z

HOMOPHON
$\rightsquigarrow$ *82 03 88 53 17 32 08 98*

- Genau wie bei Polybios wird eine 5×5-Matrix M als Schlüssel benutzt
- Die Zeilen und Spalten von M sind jedoch nicht nur mit jeweils einer, sondern mit zwei Ziffern versehen, so dass jeder Klartextbuchstabe x über vier verschiedene Koordinatenpaare verfügt
- Der Kryptotextraum wird durch M also in 25 Mengen $H(a)$, $a \in A$, mit je 4 Homophonen partitioniert

◁

Verwendung von Homophonen

- Wie wir noch sehen werden, sind homophone Chiffren auch deshalb schwerer zu brechen, weil durch sie die charakteristische Häufigkeitsverteilung der Klartextzeichen zerstört wird
- Diesen Effekt kann man noch verstärken, indem man für häufigere Klartextzeichen a eine größere Menge $H(a)$ von Homophonen vorsieht
- Damit lässt sich erreichen, dass die Verteilung der Zeichen im Geheimtext weitestgehend nivelliert wird

Beispiel (homophone Substitution, verbesserte Version)

- Tritt ein Zeichen $a \in A$ im Klartext mit Wahrscheinlichkeit $p(a)$ auf, so sollte $\|H(a)\| \approx 100 \cdot p(a)$ sein
- Da der Buchstabe `A` im Deutschen bspw. mit der Wahrscheinlichkeit $p(\texttt{A}) = 0.0647$ auftritt, werden für ihn sechs Homophone verwendet

a	$p(a)$	$H(a)$
A	0.0647	$\{15, 26, 44, 59, 70, 79\}$
B	0.0193	$\{01, 84\}$
C	0.0268	$\{13, 28, 75\}$
D	0.0483	$\{02, 17, 36, 60, 95\}$
E	0.1748	$\{04, 08, 12, 30, \ldots\}$
$\vdots$	$\vdots$	$\vdots$

◁

Verwendung von Homophonen

- Um den Suchaufwand bei der Dechiffrierung zu reduzieren, empfiehlt es sich, eine 10×10-Matrix anzulegen, in der jeder Klartextbuchstabe a an allen Stellen vorkommt, deren Koordinaten in $H(a)$ enthalten sind

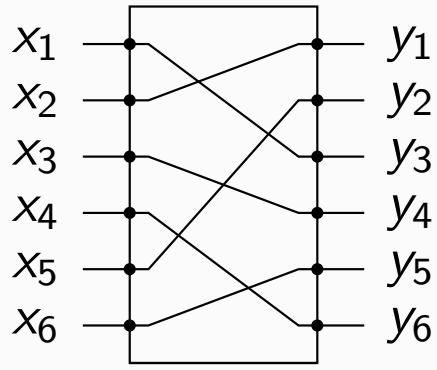
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*0*
1	N	E	C	S	A	O	D	X	I	N
2	R	G	S	N	N	A	U	C	H	Y
3	T	L	I	O	U	D	Z	M	N	E
4	H	R	E	A	N	E	E	S	I	T
5	N	I	E	T	P	H	S	L	A	R
6	E	U	M	F	R	J	E	N	E	D
7	N	E	K	S	C	T	I	T	A	A
8	H	N	I	B	R	E	U	G	V	E
9	T	E	L	S	D	R	E	O	S	E
0	B	D	W	E	Q	I	F	E	I	R

HOMOPHON
$\rightsquigarrow$ *56 98 63 34 55 29 16 68*

- Offenbar kann man diese Matrix auch zur Chiffrierung benutzen, was sogar den positiven Nebeneffekt hat, dass dadurch eine zufällige Wahl der Homophone begünstigt wird

Realisierung von binären Blocktranspositionen

- Binäre Blocktranspositionen (d.h. $A = \{0, 1\}$) lassen sich wie folgt elektronisch realisieren
- Um einen Binärblock $x_1 \cdots x_\ell$ zu permutieren, werden die einzelnen Bits auf ℓ Leitungen gelegt und diese gemäß π in einer sog. ***Permutations-box*** (kurz ***P-Box***) vertauscht
- Die Implementierung kann beispielsweise auf einem VLSI-Chip erfolgen
- Allerdings kann hierbei für große Werte von ℓ aufgrund der hohen Zahl von Überkreuzungspunkten ein hoher Flächenbedarf anfallen



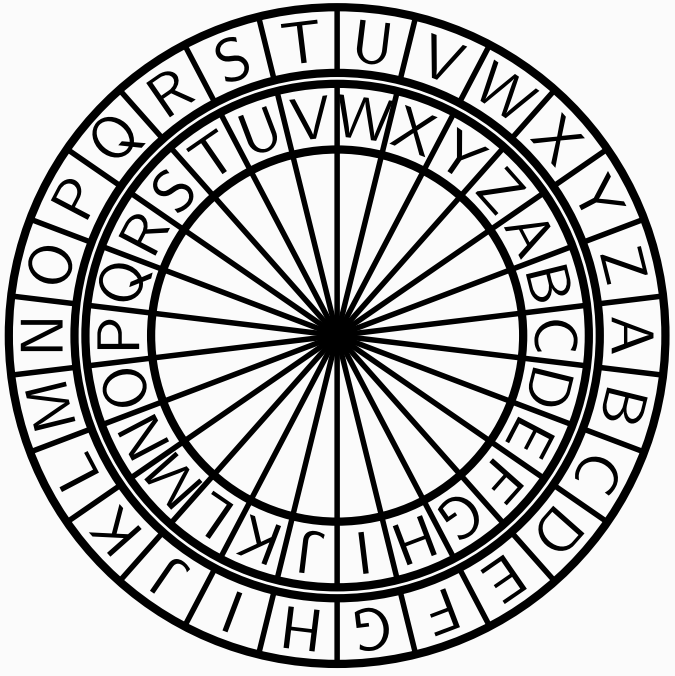
- Blocktranspositionen können auch leicht in Software als eine Folge von Zuweisungen implementiert werden:

$$Y1 := X2; \quad Y2 := X5; \quad \ldots \quad Y6 := X4;$$

- Bei großer Blocklänge und sequentieller Abarbeitung erfordert diese Art der Implementierung jedoch einen relativ hohen Zeitaufwand

Realisierung von einfachen Substitutionen

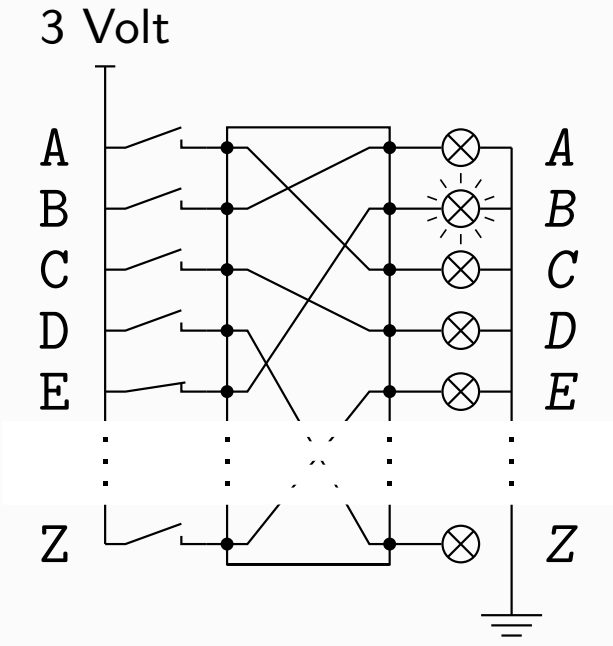
- Von Alberti stammt die Idee, das Klartext- und Kryptotextalphabet auf zwei konzentrischen Scheiben anzuordnen
- Damit lässt sich beispielsweise die additive Chiffre realisieren
- Zur Einstellung des Schlüssels k müssen die Scheiben so gegeneinander verdreht werden, dass der Schlüsselbuchstabe a_k auf der inneren Scheibe mit dem Klartextzeichen $a_0 = \mathtt{A}$ auf der äußeren Scheibe zur Deckung kommt



- Die Verschlüsselung geschieht nun durch bloßes Ablesen der zugehörigen Kryptotextzeichen auf der inneren Scheibe, so dass von der Drehfunktion der Scheiben nur bei einem Schlüsselwechsel Gebrauch gemacht wird

Realisierung von einfachen Substitutionen

- Aufgrund ihrer engen Verwandtschaft mit der Klasse der Blocktranspositionen lassen sich einfache Substitutionen auch mit Hilfe einer P-Box realisieren
- Hierfür können beispielsweise zwei Steckkontaktleisten verwendet werden
- Der aktuelle Schlüssel wird in diesem Fall durch Verbinden der entsprechenden Kontakte mit elektrischen Kabeln eingestellt



- Um etwa das Klartextzeichen `E` zu verschlüsseln, drückt man auf die entsprechende Taste, und das zugehörige Kryptotextzeichen B wird im selben Moment durch ein aufleuchtendes Lämpchen signalisiert

Realisierung von einfachen Substitutionen

- Zudem lassen sich Substitutionen auch leicht in Software realisieren
- Hierzu wird ein Feld a (*array*) deklariert, dessen Einträge $a[x]$ über die Klartextzeichen $x \in A$ adressierbar sind
- Das mit x indizierte Feldelement $a[x]$ enthält das Kryptotextzeichen, durch welches x beim Chiffriervorgang zu ersetzen ist
- Damit das Feld nicht nach jedem Schlüsselwechsel neu beschrieben werden muss, kann auch ein zweidimensionales Feld $a[k, x]$ deklariert werden, in dem für jeden Schlüssel $k \in K$ und jedes Zeichen $x \in A$ das Zeichen $E(k, x)$ gespeichert wird

Schlüssel-wert	**Klartextzeichen**			
	A	B	...	Z
0	*U*	*H*	...	*C*
1	*E*	*H*	...	*A*
⋮	⋮	⋮	⋱	⋮
63	*Y*	*F*	...	*W*