Vorlesungsskript

Einführung in die Kryptologie

Sommersemester 2020

Prof. Dr. Johannes Köbler
Humboldt-Universität zu Berlin
Lehrstuhl Komplexität und Kryptografie

6. Mai 2020

Inhaltsverzeichnis

1 Klassische Verfahren

1.1 Einführung

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.

Andererseits können durch kryptografische Verfahren weitere **Schutzziele** realisiert werden.

- *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

In das Umfeld der Kryptografie fallen auch die folgenden Begriffe.

Kryptografie: Lehre von der Geheimhaltung von Informationen durch die Verschlüsselung von Daten. 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 bzw. Schwächen.

Kryptologie: Wissenschaft vom Entwurf, der Anwendung und der Analyse von kryptografischen Verfahren (umfasst Kryptografie und Kryptoanalyse).

1.2 Kryptosysteme

Es ist wichtig, Kryptosysteme von Codesystemen zu unterscheiden.

Codesysteme

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

Beispiel 1 (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

Definition 2 (**Alphabet**)**.** *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* $A^* = \bigcup_{n \geq 0} A^n$*.*

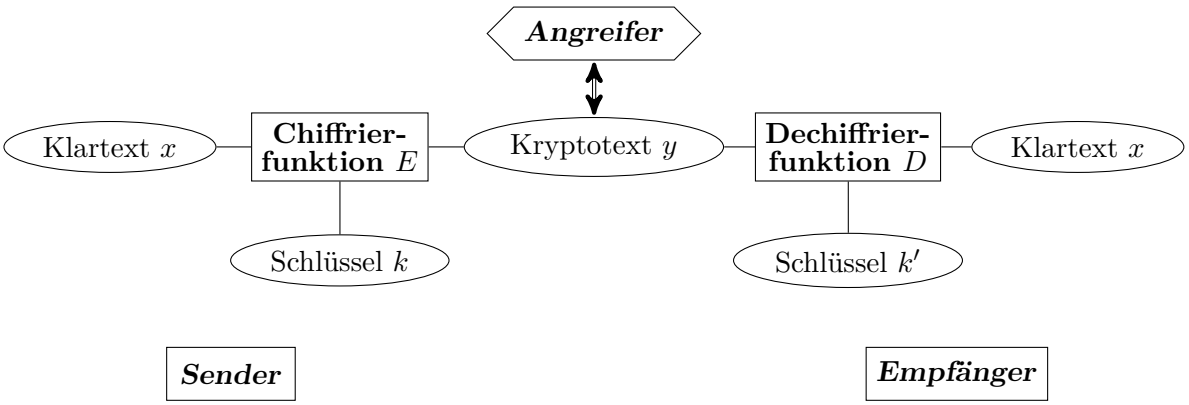
Beispiel 3. *Das* ***lateinische Alphabet*** A_{lat} *enthält die 26 Zeichen* `A,...,Z`*. Bei der Abfassung von Klartexten wurde meist auf den Gebrauch von Interpunktions- und Leerzeichen sowie auf Groß- und Kleinschreibung verzichtet (*$\rightsquigarrow$ *Verringerung der Redundanz im Klartext).* ◁

Definition 4 (**Kryptosystem**)**.** *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) und*
- $S \subseteq K \times K$*, eine Menge von Schlüsselpaaren* (k, k') *mit der Eigenschaft, dass für jeden Klartext* $x \in M$ *folgende Beziehung gilt:*

$$D(k', E(k, x)) = x \tag{1.1}$$

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



Zu jedem Schlüssel $k \in K$ korrespondiert also 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.)

Lemma 5. *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

$$x_1 \overset{(1.1)}{=} D(k', \underbrace{E(k, x_1)}_{E(k,x_2)}) = D(k', E(k, x_2)) \overset{(1.1)}{=} x_2$$

□

1.3 Die affine Chiffre

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

Definition 6 (**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*

$$a \equiv_m b \text{ oder } a \equiv b \pmod{m}$$

*(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 ganze Zahl $d \in \mathbb{Z}$ existiert mit $a = dm + r$. Sowohl r als auch d sind hierbei eindeutig bestimmt (siehe Übungen) und die Zahl d wird auch mit a div m bezeichnet.*

Die auf $\mathbb{Z}$ definierten Operationen

$$a \oplus_m b := (a + b) \bmod m \text{ und } a \odot_m b := ab \bmod m$$

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

Tabelle 1.1: Werte der additiven Chiffrierfunktion ROT13 (Schlüssel $k = 13$).

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*

$a \oplus_m -b$ 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.

Definition 7 (**Buchstabenrechnung**). *Sei $A = \{a_0, \ldots, a_{m-1}\}$ ein Alphabet. Für Indizes $i, j \in \{0, \ldots, m-1\}$ und eine ganze Zahl $z \in \mathbb{Z}$ ist*

$$\begin{aligned} a_i + a_j &= a_{i+j}, \quad a_i - a_j = a_{i-j}, \quad a_i a_j = \quad a_{ij}, \\ a_i + z &= a_{i+z}, \quad a_i - z = a_{i-z}, \quad z a_j = a_{zj \bmod m}. \end{aligned}$$

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

Definition 8 (**additive Chiffre**). *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*

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

Im Fall des lateinischen Alphabets führt der Schlüssel $k = 13$ auf eine interessante Chiffrierfunktion, die in UNIX-Umgebungen auch unter der Bezeichnung ROT13 bekannt ist (siehe Tabelle 1.1). 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 Buchstabenrechnung legt folgende Modifikation der Caesar-Chiffre nahe. Anstatt auf jedes Klartextzeichen den Schlüsselwert k zu addieren, können wir die Klartextzeichen 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 `A` als auch `N` auf das Zeichen $g(\mathtt{A}) = g(\mathtt{N}) = \mathtt{A}$ ab. Um eine hinreichende und notwendige Bedingung für die Zulässigkeit eines Schlüsselwerts k formulieren zu können, führen wir folgende Begriffe ein.

Definition 9 (**ggT, kgV, teilerfremd**). *Seien $a, b \in \mathbb{Z}$. Für $(a, b) \neq (0, 0)$ ist*

$$\mathrm{ggT}(a, b) = \max\{d \in \mathbb{Z} \mid d \text{ teilt die beiden Zahlen } a \text{ und } b\}$$

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

$$\mathrm{kgV}(a, b) = \min\{d \in \mathbb{Z} \mid d \geq 1 \text{ und die beiden Zahlen } a \text{ und } b \text{ teilen } d\}$$

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 10. *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. Jeder Teiler d von a und b ist auch ein 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. sei $a>b>0$. Bestimme die natürlichen Zahlen (durch Divsision mit Rest*):

$$r_0=a>r_1=b>r_2>\cdots>r_s>r_{s+1}=0 \text{ und } d_2,d_3,\ldots d_{s+1}$$

mit

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

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

Beispiel 11. *Für $a=693$ und $b=147$ erhalten wir*

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	$\cdot\ \mathbf{21}$	$+\ 0$

und damit $\mathrm{ggT}(693,147)=r_4=21$. ◁

Der Euklidsche Algorithmus lässt sich sowohl iterativ als auch rekursiv implementieren.

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)
```

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))
```

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

*Also: $d_{i+1}=r_{i-1}\ \mathrm{div}\ r_i$ und $r_{i+1}=r_{i-1} \bmod r_i$.

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

Durch Induktion über $i = s+1, s, \ldots, 0$ folgt $r_i \geq F_{s+1-i}$ und somit $a = r_0 \geq F_{s+1}$. Weiterhin lässt sich durch Induktion über $n \geq 0$ zeigen, dass $F_{n+1} \geq \phi^{n-1}$ ist, wobei $\phi = (1+\sqrt{5})/2$ der *goldene Schnitt* ist. Der Induktionsanfang ($n = 0$ oder 1) ist klar, da $F_2 = F_1 = 1 = \phi^0 \geq \phi^{-1}$ ist. Unter der Induktionsannahme $F_{i+1} \geq \phi^{i-1}$ 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. $s \leq 1 + \lfloor \log_\phi a \rfloor$.

Satz 12. *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* $\mathrm{ggT}(a, b)$ *durch. Dies führt auf eine Zeitkomplexität von $O(n^3)$, da jede Ganzzahldivision in Zeit $O(n^2)$ durchführbar ist.*

Erweiterter Euklidscher bzw. **Berlekamp-Algorithmus:** Der Euklidsche Algorithmus kann so modifiziert werden, dass er eine lineare Darstellung

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

des ggT liefert (Zeitkomplexität ebenfalls $O(n^3)$). Hierzu werden neben r_i und d_i weitere Zahlen

$$p_i = p_{i-2} - d_i p_{i-1} \text{ (mit } p_0 = 1 \text{ und } p_1 = 0)$$

und

$$q_i = q_{i-2} - d_i q_{i-1} \text{ (mit } q_0 = 0 \text{ und } q_1 = 1$$

für $i = 0, \ldots, s$ bestimmt. Dann gilt für $i = 0$ und $i = 1$,

$$ap_i + bq_i = r_i,$$

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 diese Gleichung auch für $i = s$ gilt:

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

Korollar 13 (Lemma von Bezout)**.** *Der größte gemeinsame Teiler von a und b ist in der Form*

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

darstellbar.

Beispiel 14. *Für* $a = 693$ *und* $b = 147$ *erhalten wir wegen*

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

$p_i\cdot 693 +$	$q_i\cdot 147 =$	r_i
$1\cdot 693 +$	$0\cdot 147 =$	693
$0\cdot 693 +$	$1\cdot 147 =$	147
$1\cdot 693 -$	$4\cdot 147 =$	105
$-1\cdot 693 +$	$5\cdot 147 =$	42
$3\cdot 693 -$	$14\cdot 147 =$	21

die lineare Darstellung $3 \cdot 693 - 14 \cdot 147 = 21$. ◁

Aus der linearen Darstellbarkeit des größten gemeinsamen Teilers ergeben sich eine Reihe von nützlichen Schlussfolgerungen.

Korollar 15. *Der größte gemeinsame Teiler von* a *und* b *wird von allen gemeinsamen Teilern von* a *und* b *geteilt,*

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

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

Korollar 16. $\operatorname{ggT}(a,b) = \min\{\lambda a + \mu b \geq 1 \mid \lambda, \mu \in \mathbb{Z}\}$.

Beweis. Sei $M = \{\lambda a + \mu b \geq 1 \mid \lambda, \mu \in \mathbb{Z}\}$, $m = \min M$ und $g = \operatorname{ggT}(a,b)$. Dann folgt $g \geq m$, da g in der Menge M enthalten ist, und $g \leq m$, da g jede Zahl in M teilt. □

Korollar 17. *Zwei Zahlen* a *und* b *sind genau dann zu einer Zahl* $m \in \mathbb{Z}$ *teilerfremd, wenn ihr Produkt* ab *teilerfremd zu* m *ist,*

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

Beweis. Da a und b teilerfremd zu m sind, existieren Zahlen $\mu, \lambda, \mu', \lambda' \in \mathbb{Z}$ mit $\mu a + \lambda m = \mu' b + \lambda' m = 1$. Somit ergibt sich aus der Darstellung

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

und Korollar 16, dass auch ab teilerfremd zu m ist.

Gilt umgekehrt $\operatorname{ggT}(ab,m) = 1$, so existieren Zahlen $\mu, \lambda \in \mathbb{Z}$ mit $\mu ab + \lambda m = 1$. Mit Korollar 16 folgt sofort $\operatorname{ggT}(a,m) = \operatorname{ggT}(b,m) = 1$. □

Korollar 18 (Lemma von Euklid)**.** *Sind* a *und* b *teilerfremd und teilt* a *das Produkt* bc*, so teilt* a *auch* c*,*

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

Beweis. Wegen $\operatorname{ggT}(a,b) = 1$ 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. □

Damit nun eine Abbildung $g : A \to A$ der Form $g(x) = bx$ auf einem Alphabet A injektiv (oder gleichbedeutend, surjektiv) ist, muss es zu jedem Zeichen $y \in A$ genau einen Zeichen $x \in A$ mit $bx = y$ geben. Wie der folgende Satz zeigt, ist dies genau dann der Fall, wenn b und m teilerfremd sind.

Satz 19. *Seien b, y, m ganze Zahlen mit $m \geq 1$. Die lineare Kongruenzgleichung $bx \equiv_m y$ besitzt genau dann eine eindeutige Lösung $x \in \{0, \ldots, m-1\}$, wenn $\mathrm{ggT}(b, m) = 1$ ist.*

Beweis. Angenommen, $\mathrm{ggT}(b, m) = g > 1$. Dann ist mit x auch $x' = x + m/g$ eine Lösung von $bx \equiv_m y$ mit $x \not\equiv_m x'$. Folglich ist die Kongruenz $bx \equiv_m y$ nicht eindeutig lösbar.
Gilt umgekehrt $\mathrm{ggT}(b, m) = 1$, so folgt aus den Kongruenzen

$$bx_1 \equiv_m y$$

und

$$bx_2 \equiv_m y$$

sofort $b(x_1 - x_2) \equiv_m 0$, also $m | b(x_1 - x_2)$. Wegen $\mathrm{ggT}(b, m) = 1$ folgt mit dem Lemma von Euklid $m | (x_1 - x_2)$, also $x_1 \equiv_m x_2$. Folglich hat die Kongruenz $bx \equiv_m y$ für jedes $y \in \mathbb{Z}_m$ höchstens eine Lösung $x \in \{0, \ldots, m-1\}$. Zudem folgt, dass die Abbildung $f : \mathbb{Z}_m \to \mathbb{Z}_m$ mit $f(x) = bx \bmod m$ injektiv ist. Da aber der Definitions- und der Wertebereich von f die gleiche Mächtigkeit haben, muss f dann auch surjektiv sein. Somit hat die Kongruenz $bx \equiv_m y$ für jedes $y \in \mathbb{Z}_m$ sogar genau eine Lösung $x \in \{0, \ldots, m-1\}$. □

Korollar 20. *Im Fall $\mathrm{ggT}(b, m) = 1$ hat die Kongruenz $bx \equiv_m 1$ genau eine Lösung, die das **multiplikative Inverse** von b modulo m genannt und mit $b^{-1} \bmod m$ (oder einfach mit b^{-1}) bezeichnet wird.*

Korollar 17 zeigt, dass die Menge

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

aller invertierbaren Elemente von $\mathbb{Z}_m$ unter der Operation $\odot_m$ abgeschlossen ist. Mit Korollar 20 folgt daher, dass $(\mathbb{Z}_m^*, \odot_m, 1)$ eine multiplikative Gruppe bildet. 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.
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 b^{-1} 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.
Nun lässt sich die additive Chiffre leicht zur affinen Chiffre erweitern.

Definition 21 (affine Chiffre). *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*

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

In diesem Fall liefert die Schlüsselkomponente $b = -1$ für jeden Wert von $c \in \mathbb{Z}_m$ eine involutorische Chiffrierfunktion $x \mapsto E_{(-1,c)}(x) = c - x$ (**verschobenes komplementäres Alphabet**). Wählen wir für c ebenfalls den Wert -1, so ergibt sich die Chiffrierfunktion $x \mapsto -x - 1$, die auch als **revertiertes Alphabet** bekannt ist. Offenbar ist diese Funktion genau dann echt involutorisch, wenn m gerade 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*

Als nächstes illustrieren wir die Ver- und Entschlüsselung mit der affinen Chiffre an einem kleinen Beispiel.

Beispiel 22 (affine Chiffre)**.** *Sei* $A = \{\mathsf{A}, \ldots, \mathsf{Z}\} = B$*, also* $m = 26$*. Weiter sei* $k = (9, 2)$*, also* $b = 9$ *und* $c = 2$*. Um das Klartextzeichen* $x = \mathsf{F}$ *zu verschlüsseln, berechnen wir*

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

da der Index von F *gleich* 5*, der von* V *gleich* 21 *und* $9 \cdot 5 + 2 = 47 \equiv_{26} 21$ *ist. Um ein Kryptotextzeichen wieder entschlüsseln zu können, benötigen wir das multiplikative Inverse von* $b = 9$*, das sich wegen*

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

zu $b^{-1} = q_3 = 3$ *ergibt. Damit erhalten wir für das Kryptotextzeichen* $y = \mathsf{V}$ *das ursprüngliche Klartextzeichen*

$$D(k, y) = b^{-1}(y - c) = 3(\mathsf{V} - 2) = \mathsf{F}$$

zurück, da $3 \cdot 19 = 57 \equiv_{26} 5$ *ist.* ◁

Zur Berechnung der Schlüsselzahl bei der multiplikativen und affinen Chiffre benötigen wir die Funktion

$$\varphi : \mathbb{N} \to \mathbb{N} \quad \text{mit } \varphi(m) = \|\mathbb{Z}_m^*\| = \|\{a \in \mathbb{Z}_m \mid \mathrm{ggT}(a, m) = 1\}\|,$$

die sogenannte *Eulersche* φ*-Funktion*. Die folgende Tabelle zeigt die Werte $\varphi(m)$ für $m = 1, \ldots, 10$ (für die Menge $\{1, \ldots, n\}$, $n \in \mathbb{N}$, schreiben wir auch kurz $[n]$).

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. 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 für beliebige Zahlen $n \in \mathbb{N}$ 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

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

Beispiel 23. *Sei $m = 5$ und $l = 6$. Dann 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 fällt auf, dass ein x-Wert genau dann in $\mathbb{Z}_{30}^*$ liegt, wenn der Funktionswert $f(x) = (y,z)$ zu $\mathbb{Z}_5^* \times \mathbb{Z}_6^*$ gehört (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 Argumente in $\mathbb{Z}_{30}^*$ bijektiv auf die Werte in $\mathbb{Z}_5^* \times \mathbb{Z}_6^*$ ab. Für f^{-1} erhalten wir somit folgende Tabelle:*

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**

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 $\mathbb{Z}_{30}^*$. ◁

Der Chinesische Restsatz, den wir im nächsten Abschnitt beweisen, besagt, dass f im Fall $\mathrm{ggT}(m,\ell) = 1$ bijektiv und damit invertierbar 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

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

Satz 24. *Die Eulersche φ-Funktion ist multiplikativ, d. h. für teilerfremde Zahlen m und ℓ gilt $\varphi(m\ell) = \varphi(m)\varphi(\ell)$.*

Korollar 25. *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

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

Der Chinesische Restsatz

Die beiden linearen Kongruenzen

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

besitzen je eine Lösung, es gibt aber kein x, das beide Kongruenzen gleichzeitig erfüllt. Der nächste Satz zeigt, dass unter bestimmten Voraussetzungen gemeinsame Lösungen existieren, und wie sie berechnet werden können.

Satz 26 (Chinesischer Restsatz (CRS))**.** *Falls $m_1, \ldots, m_k$ paarweise teilerfremd sind, dann hat das System*

$$\begin{aligned} x &\equiv_{m_1} b_1 \\ &\vdots \\ x &\equiv_{m_k} b_k \end{aligned} \tag{1.2}$$

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 $n_i = {}^{m}\!/_{m_i}$ existieren 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

$$x \equiv_{m_j} \begin{cases} 0, & j \neq i \quad (a) \\ 1, & j = i \quad (b) \end{cases} \tag{1.3}$$

Folglich gelten für $s = \sum_{i=1}^{k} b_i s_i$ die Kongruenzen $s \overset{(1.3a)}{\equiv}_{m_j} b_j s_j \overset{(1.3b)}{\equiv}_{m_j} b_j$, d.h. s löst das System (1.2). 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 (1.2) ist eindeutig lösbar. $\square$

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 Ring $(\mathbb{Z}_m, \oplus_m, \odot_m)$ 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 dadurch die Laufzeit von bestimmten Berechnungen im Ring $\mathbb{Z}_m$ deutlich reduzieren, sofern die Primzahlzerlegung von m bekannt ist.

1.4 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 $x \in A^l$. Als Schlüssel dient eine $(l \times l)$-Matrix $k = (k_{ij})$ mit Koeffizienten in $\mathbb{Z}_m$. Diese transformiert einen Klartext $x = x_1 \dots x_l \in A^l$ in den Kryptotext $y = y_1 \dots y_l$ mit $y_i = x_1 k_{1i} + \dots + x_l k_{li}$ für $i = 1, \dots, l$:

$$(y_1 \ \cdots \ y_l) = (x_1 \ \cdots \ x_l) \begin{pmatrix} k_{11} & \dots & k_{1l} \\ \vdots & \ddots & \vdots \\ k_{l1} & \dots & k_{ll} \end{pmatrix}$$

Wir bezeichnen die Menge aller $(l \times l)$-Matrizen (k_{ij}) mit Koeffizienten $k_{ij} \in \mathbb{Z}_m$ mit $\mathbb{Z}_m^{l \times l}$. Als Schlüssel können nur invertierbare Matrizen k benutzt werden, da sonst der Chiffriervorgang nicht injektiv ist. Ob eine Matrix $k \in \mathbb{Z}_m^{l \times l}$ invertierbar ist, lässt sich an ihrer Determinante erkennen.

Definition 27 (**Determinante**). *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 $A = (a_1, \dots, a_n) \in R^{n \times n}$ mit Spalten $a_1, \dots, a_n \in (R^n)^T$, jeden Spaltenvektor $b \in (R^n)^T$ und jedes $r \in R$ gilt*

$$f(a_1, \dots, ra_i + b, \dots, a_n) = rf(a_1, \dots, a_i, \dots, a_n) + f(a_1, \dots, b, \dots, a_n).$$

- *f ist **alternierend**, d.h. im Fall $a_i = a_j$ für $i \neq j$ gilt $f(a_1, \dots, a_n) = 0$.*
- *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 $det(A)$.

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 $i \in \{1, \cdots, n\}$ 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).

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

Definition 28 (**Hill-Chiffre**). *Sei $A = \{a_0, \dots, a_{m-1}\}$ 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*

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

Beispiel 29 (Hill-Chiffre)**.** *Benutzen wir zur Chiffrierung von Klartextblöcken der Länge $l = 4$ über dem lateinischen Alphabet A_{lat} die Schlüsselmatrix*

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

so erhalten wir beispielsweise für den Klartext HILL *wegen*

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

den Kryptotext $E(k, \mathtt{HILL}) = \mathit{NERX}$. Für die Entschlüsselung wird die inverse Matrix k^{-1} benötigt. Diese wird in den Übungen berechnet. ◁

1.5 Die Vigenère-Chiffre und andere Stromsysteme

Die nach dem Franzosen Blaise de Vigenère (1523–1596) benannte Chiffre ersetzt den Klartext zeichenweise, allerdings je nach Position im Klartext unterschiedlich.

Definition 30 (**Vigenère-Chiffre**)**.** *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*

$$E(k, x) = y_0 \dots y_{n-1} \quad \textit{mit} \quad y_i = x_i + k_{(i \bmod d)} \textit{ für } i = 1, \dots, n-1$$

und dechiffriert einen Kryptotext $y = y_0 \dots y_{n-1}$ zu

$$D(k, y) = x_0 \dots x_{n-1} \quad \textit{mit} \quad x_i = y_i - k_{(i \bmod d)} \textit{ für } i = 1, \dots, n-1.$$

Beispiel 31 (Vigenère-Chiffre)**.** *Verwenden wir das lateinische Alphabet A_{lat} als Klartextalpabet und wählen wir als Schlüssel das Wort $k = \mathit{WIE}$, so ergibt sich für den Klartext* VIGENERE *beispielsweise der Kryptotext*

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

◁

Um einen Klartext x zu verschlüsseln, wird also das Schlüsselwort $k = k_0 \dots k_{d-1}$ so oft wiederholt, bis der dabei entstehende **Schlüsselstrom** $\hat{k} = k_0 k_1 \dots k_{d-1} k_0 \dots$ 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.

Beispiel 32. *Vigenère-Chiffre*

Chiffrierung:		*Dechiffrierung:*	
VIGENERE	(*Klartext x*)	*RQKAVINM*	(*Kryptotext y*)
$+$ *WIEWIEWI*	(*Schlüsselstrom $\hat{k}$*)	$-$ *WIEWIEWI*	(*Schlüsselstrom $\hat{k}$*)
RQKAVINM	(*Kryptotext y*)	VIGENERE	(*Klartext x*)

◁

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

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 33 (**Beaufort-Chiffre**). *Verschlüsseln wir den Klartext* `BEAUFORT` *beispielsweise unter dem Schlüsselwort* $k =$ `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*)

◁

Bei den bisher betrachteten Chiffren wird aus einem Schlüsselwort $k = k_0 \dots k_{d-1}$ ein **periodischer Schlüsselstrom** $\hat{k} = \hat{k}_0 \dots \hat{k}_{n-1}$ erzeugt, das heißt, es gilt $\hat{k}_i = \hat{k}_{i+d}$ für alle $i = 0, \dots, n-d-1$. Da eine kleine Periode das Brechen der Chiffre erleichtert, sollte entweder ein Schlüsselstrom mit sehr großer Periode oder noch besser ein **fortlaufender Schlüsselstrom** zur Chiffrierung benutzt werden. Ein solcher nichtperiodischer Schlüsselstrom lässt sich beispielsweise ohne großen Aufwand erzeugen, indem man an das Schlüsselwort den Klartext oder den Kryptotext anhängt (sogenannte ***Autokey-Chiffrierung***).[†]

[†]Die Idee, den Schlüsselstrom durch Anhängen des Klartextes an ein Schlüsselwort zu bilden, stammt von Vigenère, während er mit der Erfindung der nach ihm benannten Vigenère-Chiffre „nichts zu tun“ hatte. Diese wird vielmehr Giovan Batista Belaso (1553) zugeschrieben.

Beispiel 34 (*Autokey*-Chiffre)**.** *Benutzen wir wieder das Schlüsselwort* **WIE***, um den Schlüsselstrom durch Anhängen des Klar- bzw. Kryptotextes zu erzeugen, so erhalten wir für den Klartext* VIGENERE *folgende Kryptotexte:*

Klartext-Schlüsselstrom:		*Kryptotext-Schlüsselstrom:*	
VIGENERE	(*Klartext*)	VIGENERE	(*Klartext*)
+ WIEVIGEN	(*Schlüsselstrom*)	+ WIERQKVD	(*Schlüsselstrom*)
RQKZVKVR	(*Kryptotext*)	RQKVDOMH	(*Kryptotext*)

◁

Auch die Dechiffrierung ist in beiden Fällen einfach. Bei der ersten Alternative kann der Empfänger durch Subtraktion des Schlüsselworts den Anfang des Klartextes bilden und gleichzeitig den Schlüsselstrom verlängern, so dass sich auf diese Weise Stück für Stück der gesamte Kryptotext entschlüsseln lässt. Noch einfacher gestaltet sich die Dechiffrierung im zweiten Fall, da sich hier der Schlüsselstrom vom Kryptotext nur durch das vorangestelle Schlüsselwort unterscheidet.

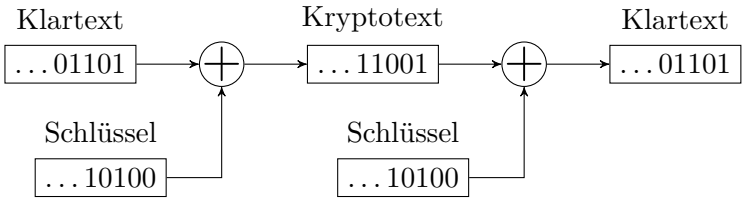
1.6 Der One-Time-Pad

Eine weitere Möglichkeit ist, eine Textstelle in einem Buch als Schlüssel zu vereinbaren und den dort beginnenden Text als aperiodischen Schlüsselstrom zu benutzen (Lauftextverschlüsselung). Besser ist es jedoch, mithilfe von Pseudozufallsgeneratoren 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, weshalb diese Chiffre nur wenig praktikabel ist.[§]

Beispiel 35 (One-time-pad)**.** *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.



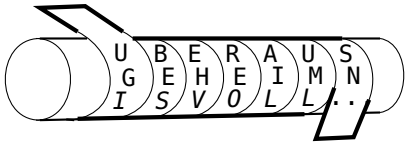
[‡] Diese Methode schlug der amerikanische Major Joseph O. Mauborgne im Jahr 1918 vor, nachdem ihm ein von Gilbert S. Vernam für den Fernschreibverkehr entwickeltes Chiffriersystem vorgestellt wurde.

[§] Diese Methode wurde beispielsweise beim „heißen Draht“, der 1963 eingerichteten, direkten Fernschreibverbindung zwischen dem Weißen Haus in Washington und dem Kreml in Moskau, angewandt.

1.7 Klassifikation von Kryptosystemen

Bei den bisher betrachteten Chiffrierfunktionen handelt es sich um **Substitutionen**, d.h. sie bilden den Kryptotext aus dem Klartext, indem sie Klartextzeichen – einzeln oder in Gruppen – durch Kryptotextzeichen *ersetzen*. Dagegen verändern **Transpositionen** lediglich die *Reihenfolge* der einzelnen Klartextzeichen.

Beispiel 36 (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 sogenannte* ***Skytale****) und beschreibt ihn in Längsrichtung mit der Geheimbotschaft.*



ÜBERAUS GEHEIMNISVOLL ...
↝ *ÜGI ... BES ... EHV ... REO ... AIL ... UML ... SN ...*

Besitzt der Empfänger eines auf diese Weise beschrifteten Papierstreifens einen Stab mit dem gleichen Umfang, so kann er ihn auf dieselbe Art wieder entziffern. ◁

Als Schlüssel fungiert hier also der Stabumfang bzw. die Anzahl k der Zeilen, mit denen der Stab beschrieben wird. Findet der gesamte Klartext x auf der Skytale Platz und beträgt seine Länge ein Vielfaches von k, so geht x bei der Chiffrierung in den Kryptotext

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

über. Dasselbe Resultat erhält man, wenn x zeilenweise in eine $k \times m$-Matrix geschrieben und spaltenweise wieder auslesen wird (sogenannte **Spaltentransposition**):

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

Ist die Klartextlänge kein Vielfaches von k, so kann der Klartext durch das Ein- bzw. Anfügen von sogenannten **Blendern** (Füllzeichen) verlängert werden. Damit der Empfänger diese Füllzeichen nach der Entschlüsselung wieder entfernen kann, ist lediglich darauf zu achten, dass sie im Klartext leicht als solche erkennbar sind.

Von der Methode, die letzte *Zeile* nur zum Teil zu füllen, ist dagegen abzuraten. In diesem Fall würden nämlich auf dem abgewickelten Papierstreifen Lücken entstehen, aus deren Anordnung man Schlüsse auf den benutzten Schlüssel k ziehen könnte. Andererseits ist nichts dagegen einzuwenden, dass der Sender die letzte *Spalte* der Skytale nur zum Teil beschriftet.

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

Beispiel 37. *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 kann als Schlüssel vereinbart werden.*

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

ZICKZACKLINIE $\rightsquigarrow$ *ZZLEIKAKIICCN*

◁

Bei einer Zick-Zack-Transposition werden Zeichen im vorderen Klartextbereich bis fast ans Ende des Kryptotextes verlagert und umgekehrt. Dies hat den Nachteil, dass für die Generierung des Kryptotextes der gesamte Klartext gepuffert werden muss. Daher werden meist **Blocktranspositionen** verwendet, bei denen die Zeichen nur innerhalb fester Blockgrenzen transponiert werden.

Definition 38 (Blocktranspositionschiffre). *Sei $A = B$ ein beliebiges Alphabet und für eine natürliche Zahl $l \geq 2$ sei $M = C = A^l$. Bei einer **Blocktranspositionschiffre** wird durch jeden Schlüssel $k \in K$ eine Permutation π beschrieben, so dass für alle Zeichenfolgen $x_1 \cdots x_l \in M$ und $y_1 \cdots y_l \in C$*

$$E(k, x_1 \cdots x_l) = x_{\pi(1)} \cdots x_{\pi(l)}$$

und

$$D(k, y_1 \cdots y_l) = y_{\pi^{-1}(1)} \cdots y_{\pi^{-1}(l)}$$

gilt.

Eine Blocktransposition mit Blocklänge l lässt sich durch eine Permutation $\pi \in S_l$ (also auf der Menge $\{1, \ldots, l\}$) beschreiben.

Beispiel 39. *Eine Skytale, die mit 4 Zeilen der Länge 6 beschrieben wird, realisiert beispielsweise folgende 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

◁

Für die Entschlüsselung muss die zu π **inverse Permutation** π^{-1} benutzt werden. Wird π durch eine Folge von Zyklen $(i_1\ i_2\ i_3\ \ldots\ i_n)$ dargestellt, wobei i_1 auf i_2, i_2 auf i_3 usw. und schließlich i_n auf i_1 abgebildet wird, so ist π^{-1} sehr leicht zu bestimmen.

Beispiel 40.

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

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

wenn, wie allgemein üblich, Einerzklen weggelassen werden. Daraus erhalten wir unmittelbar π^{-1} zu

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

wenn wir jeden Zyklus mit seinem kleinsten Element beginnen lassen und die Zyklen nach der Größe dieser Elemente anordnen. ◁

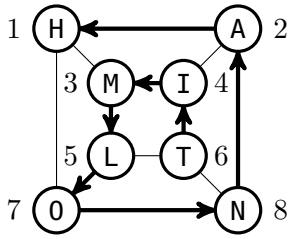
Beispiel 41. *Bei der **Matrix-Transposition** wird der Klartext zeilenweise in eine $k \times m$-Matrix eingelesen und der Kryptotext spaltenweise gemäß einer Spaltenpermutation π, die als Schlüssel dient, wieder ausgelesen. Für $\pi = (1\ 4\ 3)\ (2\ 6)$ wird also zuerst Spalte $\pi(1) = 4$, dann Spalte $\pi(2) = 6$ und danach Spalte $\pi(3) = 1$ usw. 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*

◁

Beispiel 42. *Bei der* ***Weg-Transposition*** *wird als Schlüssel eine Hamiltonlinie in einem Graphen mit den Knoten* $1, \ldots, l$ *benutzt. (Eine Hamiltonlinie ist eine Anordnung aller Knoten, in der je zwei aufeinanderfolgende Knoten durch eine Kante verbunden sind.) Der Klartextblock* $x_1 \cdots x_l$ *wird gemäß der Knotennumerierung in den Graphen eingelesen und der zugehörige 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, eine Hamiltonlinie als Schlüssel zu benutzen, besteht offenbar darin, dass man sich den Verlauf einer Hamiltonlinie bildhaft vorstellen und daher besser einprägen kann als eine Zahlenfolge.

Sehr beliebt ist auch die Methode, sich eine Permutationen in Form eines **Schlüsselworts** (oder einer aus mehreren Wörtern bestehenden **Schlüsselphrase**) ins Gedächtnis einzuprägen. Aus einem solchen Schlüsselwort lässt sich die zugehörige Permutation σ leicht rekonstruieren, indem man das Wort auf Papier schreibt und in der Zeile darunter für jedes einzelne Zeichen seine Position i innerhalb des Wortes vermerkt.

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

Die Werte $\sigma(i)$, die σ auf diesen Nummern annimmt, werden nun dadurch ermittelt, dass man die Schlüsselwort-Buchstaben in alphabetischer Reihenfolge durchzählt. Dabei werden mehrfach vorkommende Zeichen gemäß ihrer Position im Schlüsselwort an die Reihe genommen. Alternativ kann man auch alle im Schlüsselwort wiederholt vorkommenden Zeichen streichen, was im Fall des Schlüsselworts *CAESAR* auf eine Blocklänge von 5 führen würde.

Wir wenden uns nun der Klassifikation von Substitutionen zu. Ein wichtiges Unterscheidungsmerkmal ist z.B. die Länge der Klartexteinheiten, auf denen die Chiffre operiert.

Monografische Substitutionen ersetzen Einzelbuchstaben.

Polygrafische Substitutionen ersetzen dagegen aus mehreren Zeichen bestehende Klartextsegmente auf einmal.

Eine polygrafische Substitution, die auf Zeichenpaaren operiert, wird **digrafisch** genannt. Das älteste bekannte polygrafische Chiffrierverfahren wurde von Giovanni Porta im Jahr 1563 veröffentlicht. Dabei werden je zwei aufeinanderfolgende Klartextzeichen durch ein einzelnes Kryptotextzeichen ersetzt.

Beispiel 43. *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 *Klartextzeichen* $\mathtt{A}, \ldots, \mathtt{I}, \mathtt{L}, \ldots, \mathtt{T}, \mathtt{V}, \mathtt{Z}$ *indiziert sind. Zur Ersetzung des Zeichenpaars* $a_i a_j$ *wird das in Zeile* i *und Spalte* j *befindliche Kryptotextzeichen*

$$E(M, a_i a_j) = y_{ij}$$

benutzt. ◁

Eine Substitution heißt **monopartit**, falls sie die Klartextsegmente durch Einzelzeichen ersetzt, sonst **multipartit**. Wird der Kryptotext aus Zeichenpaaren zusammengesetzt, so spricht man von einer **bipartiten** Substitution. 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 ⇝ *30 24 21 43 01 13 24 33*

Die **Polybios-Chiffre** benutzt als Schlüssel eine 5×5-Matrix, die aus sämtlichen Klartextzeichen gebildet wird.[¶] Die Verschlüsselung des Klartextes erfolgt zeichenweise, indem man einen in Zeile i und Spalte j eingetragenen Klartextzeichen durch das Koordinatenpaar ij ersetzt. Der Kryptotextraum besteht also aus den Ziffernpaaren $\{\mathtt{00}, \mathtt{01}, \ldots, \mathtt{44}\}$.

Die Frage, ob bei der Ersetzung der einzelnen Segmente des Klartextes eine einheitliche Strategie verfolgt wird oder ob diese von Segment zu Segment verändert wird, führt uns auf ein weiteres wichtiges Unterscheidungsmerkmal bei Substitutionen.

Monoalphabetische Substitutionen ersetzen jedes einzelne Klartextsegment unabhängig von seiner Position im Klartext auf dieselbe Weise.

Polyalphabetische Substitutionen verwenden eine Ersetzungsregel, die in Abhängigkeit von den bereits verarbeiteten Klartextsegmenten varieren kann.

Die Bezeichnung „monoalphabetisch“ bringt zum Ausdruck, dass der Ersetzungsmechanismus im monografischen Fall für jeden Schlüssel auf einem festen Alphabet beruht. Die von Caesar benutzte Chiffriermethode mit dem Schlüssel $k = 3$ kann beispielsweise vollständig durch Angabe des Ersetzungsalphabets $\{\mathtt{D,E,F,G,W,...,Y,Z,A,B,C}\}$ beschrieben werden. Monoalphabetische Chiffrierverfahren ersetzen meist Texteinheiten einer festen Länge $l \geq 1$ durch Kryptotextsegmente derselben Länge.

Definition 44 (**Blockchiffre**). *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* $y \in C$,

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

Im Fall $\ell = 1$ *spricht man auch von einer* ***einfachen Substitutionschiffre****.*

[¶] Da nur 25 Plätze zur Verfügung stehen, muss bei Benutzung des lateinischen Alphabets entweder ein Buchstabe weggelassen oder ein Platz mit zwei Zeichen besetzt werden.

Polyalphabetische Substitutionen greifen im Wechsel auf verschiedene Ersetzungsalphabete zurück, so dass unterschiedliche Vorkommen eines Zeichens (oder einer Zeichenkette) auch auf unterschiedliche Art ersetzt werden können. Welches Ersetzungsalphabet wann an der Reihe ist, wird dabei in Abhängigkeit von der Länge oder der Gestalt des bereits verarbeiteten Klartextes bestimmt.

Fast alle polyalphabetischen Chiffrierverfahren operieren – genau wie monoalphabetische Substitutionen – auf Klartextblöcken einer festen Länge l, die sie in Kryptotextblöcke einer festen Länge l' überführen, wobei meist $l = l'$ ist. Da diese Blöcke jedoch vergleichsweise kurz sind, kann der Klartext der Chiffrierfunktion ungepuffert zugeführt werden. Man nennt die einzelnen Klartextblöcke in diesem Zusammenhang auch nicht ‚Blöcke' sondern ‚Zeichen' und spricht von **sequentiellen Chiffren** oder von **Stromchiffren**.

Definition 45 (**Stromchiffre**)**.** *Sei A ein beliebiges Alphabet und sei $M = C = A^l$ für eine natürliche Zahl $l \geq 1$. Weiterhin seien K und $\hat{K}$ Schlüsselräume. Eine* ***Stromchiffre*** *wird durch eine Verschlüsselungsfunktion $E : \hat{K} \times M \to C$ und einen Schlüsselstromgenerator $g : K \times A^* \to \hat{K}$ beschrieben. 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.

Der interne Schlüsselraum kann also wie bei der Blockchiffre eine maximale Größe von $(m^l)!$ annehmen (im häufigen Spezialfall $l = 1$ also $m!$). Die Aufgabe des Schlüsselstromgenerators g besteht darin, aus dem externen Schlüssel k und dem bereits verarbeiteten Klartext $x_0 \dots x_{i-1}$ den aktuellen internen Schlüssel $\hat{k}_i$ zu berechnen. Die bisher betrachteten Stromchiffren benutzen z.B. die folgenden Schlüsselstromgeneratoren.

Stromchiffre	Chiffrierfunktionen	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 Blöcke, aus denen der Kryptotext zusammengesetzt wird, eine einheitliche Länge. Es liegt nahe, einem Gegner

die unbefugte Rekonstruktion des Klartextes dadurch zu erschweren, dass man Blöcke unterschiedlicher Länge verwendet. Man spricht hierbei auch von einer **Spreizung** (*straddling*) des Kryptotextalphabets. Ein bekanntes Beispiel für diese Technik ist die sogenannte Spionage-Chiffre, die vorzugsweise von der ehemaligen sowjetischen Geheimpolizei NKWD (*Naródny Komissariàt Wnutrennich Del*; zu deutsch: Volkskommissariat des Innern) benutzt wurde.

Beispiel 46. *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` $\rightsquigarrow$ *274154795751*

◁

Man überzeugt sich leicht davon, dass sich die von der Spionage-Chiffre generierten Kryptotexte wieder eindeutig dechiffrieren lassen, da die Kryptotextsegmente 1, 2,. . . , 8, 01, 02, . . . , 08, 91, 92, . . . , 98, die für die Klartextzeichen eingesetzt werden, die **Fano-Bedingung** erfüllen: Keines von ihnen bildet den Anfang eines anderen. Da die Nummern 5 und 8 der beiden letzten Spalten der Matrix auch als Zeilennummern verwendet werden, liefert dies auch eine Erklärung dafür, warum keine Schlüsselwortzeichen in die beiden letzten Spalten eingetragen werden dürfen.

Verwendung von Blendern und Homophonen

Die Verwendung von gespreizten Chiffren zielt offenbar darauf ab, die „**Fuge**“ zwischen den einzelnen Kryptotextsegmenten, die von unterschiedlichen Klartextzeichen herrühren, zu verdecken, um dem Gegner eine unbefugte Dechiffrierung zu erschweren. Dennoch bietet die Spionage-Chiffre noch genügend Angriffsfläche, da im Klartext häufig vorkommende Wortmuster auch im Kryptotext zu Textwiederholungen führen.

Eine Möglichkeit, diese Muster aufzubrechen, besteht darin, Blender in den Klartext einzustreuen. Abgesehen davon, dass das Entfernen der Blender auch für den rechtmäßigen Empfänger mit Mühe verbunden ist, muss für den Zugewinn an Sicherheit auch mit einer Expansion des Kryptotextes bezahlt werden.

Ist man bereit, dies in Kauf zu nehmen, so gibt es auch noch eine wirksamere Methode, die Übertragung struktureller und statistischer Klartextmerkmale auf den Kryptotext abzumildern. Die Idee dabei ist, zur Chiffrierung der einzelnen Klartextzeichen a nicht nur jeweils eines, sondern eine Menge $H(a)$ von Chiffrezeichen vorzusehen, und daraus für jedes Vorkommen von a im Klartext eines auszuwählen (am besten zufällig). Da alle Zeichen in $H(a)$ für dasselbe Klartextzeichen stehen, werden sie auch **Homophone** genannt.

Definition 47 (**homophonen Substitutionschiffre**)**.** *Sei* A *ein Klartextalphabet und sei* $M = A$*. Weiter sei* C *ein Kryptotextraum der Größe* $\|C\| > \|A\| = m$*. In einer* ***homophonen Substitutionschiffre*** *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 aus der Menge $H(a)$ gewählt und für a eingesetzt.

Durch den Einsatz einer homophonen Substitution wird also erreicht, dass verschiedene Vorkommen eines Klartextzeichens auch auf unterschiedliche Weise ersetzt werden können. 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 durch eine Funktion beschreibbar ist, da derselbe Klartext x in mehrere verschiedene Kryptotexte y übergehen kann.

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

Beispiel 48 (homophone Substitution)**.** *Sei* $A = \{\mathsf{A}, \ldots, \mathsf{Z}\}$, $B = \{\mathtt{0}, \ldots, \mathtt{9}\}$ *und* $C = \{\mathtt{00}, \ldots, \mathtt{99}\}$.

	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 ansprechbar ist. Der Kryptotextraum wird durch* M *also in* 25 *Mengen* $H(a)$*,* $a \in A$*, mit je* 4 *Homophonen partitioniert.* ◁

Wie wir noch sehen werden, sind homophone Chiffrierungen auch deshalb schwerer zu brechen, weil durch sie die charakteristische Häufigkeitsverteilung der Klartextzeichen zerstört wird. Dieser Effekt kann dadurch noch verstärkt werden, dass man für häufig vorkommende Klartextzeichen a eine entsprechend größere Menge $H(a)$ an Homophonen vorsieht. Damit lässt sich erreichen, dass die Verteilung der im Geheimtext auftretenden Zeichen weitgehend nivelliert wird.

Beispiel 49 (homophone Substitution, verbesserte Version)**.** *Ist* $p(a)$ *die Wahrscheinlichkeit, mit der ein Zeichen* $a \in A$ *in der Klartextsprache auftritt, so sollte* $\|H(a)\| \approx 100 \cdot p(a)$ *sein.*

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, 43, 46, 47, 53, 61, 67, 69, 72, 80, 86, 90, 92, 97*}
⋮	⋮	⋮

Da der Buchstabe A *im Deutschen beispielsweise mit einer Wahrscheinlichkeit von* $p(\mathsf{A}) = 0.0647$ *auftritt, sind für ihn sechs verschiedene Homophone vorgesehen.* ◁

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.

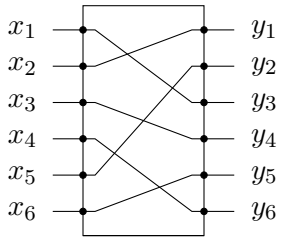
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*0*
1	N	E	C	S	A	O	D	X	I	N
2	R	G	S	N	N	A	U	C	H	Y
3	T	L	I	O	U	D	Z	M	N	E
4	H	R	E	A	N	E	E	S	I	T
5	N	I	E	T	P	H	S	L	A	R
6	E	U	M	F	R	J	E	N	E	D
7	N	E	K	S	C	T	I	T	A	A
8	H	N	I	B	R	E	U	G	V	E
9	T	E	L	S	D	R	E	O	S	E
0	B	D	W	E	Q	I	F	E	I	R

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.

1.8 Realisierung von Blocktranspositionen und einfachen Substitutionen

Abschließend möchten wir eine einfache elektronische Realisierungsmöglichkeit von Blocktranspositionen erwähnen, die auf binär kodierten Klartexten operieren (d.h. $A = \{0, 1\}$). Um einen Binärblock $x_1 \cdots x_l$ der Länge l zu permutieren, müssen die einzelnen Bits lediglich auf l Leitungen gelegt und diese gemäß π in einer sogenannten **Permutationsbox** (kurz **P-Box**) vertauscht werden.



Die Implementierung einer solchen P-Box kann beispielsweise auf einem VLSI-Chip erfolgen. Allerdings kann hierbei für größere Werte von l aufgrund der hohen Zahl von Überkreuzungspunkten ein hoher Flächenbedarf anfallen.

Blocktranspositionen können auch leicht durch Software als eine Folge von Zuweisungen

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

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

Von Alberti stammt die Idee, das Klartext- und Kryptotextalphabet auf zwei konzentrischen Scheiben unterschiedlichen Durchmessers anzuordnen. In Abbildung 1.1 ist gezeigt, wie sich mit einer solchen Drehscheibe beispielsweise die additive Chiffre realisieren lässt. Zur Einstellung des Schlüssels k müssen die Scheiben so gegeneinander verdreht werden, dass der Schlüsselbuchstabe a_k auf der inneren Scheibe mit dem Klartextzeichen $a_0 = \mathsf{A}$ auf der äußeren Scheibe zur Deckung kommt. Auf der Drehscheibe in Abbildung 1.1 ist

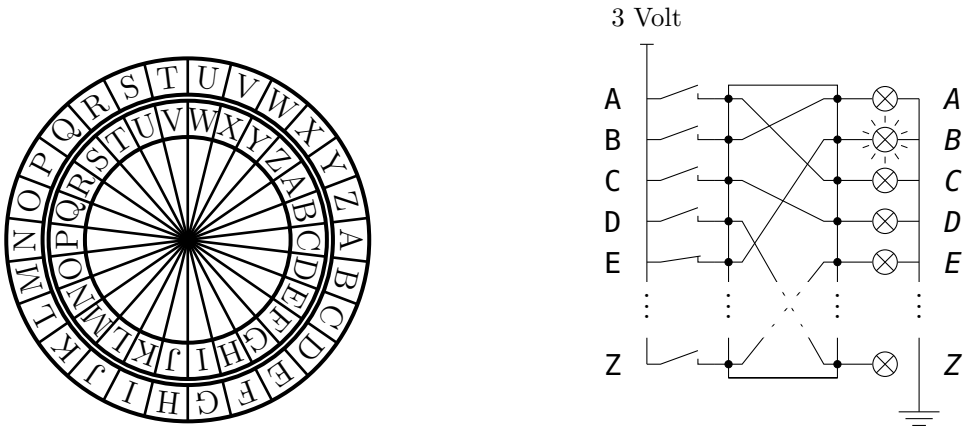


Abbildung 1.1: Realisierung von einfachen Substitutionen mit einer Drehscheibe und mit Hilfe von Steckverbindungen.

beispielsweise der Schlüssel $k = 2$ eingestellt, das heißt, $a_k = \mathsf{C}$. 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.

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 (siehe Abbildung 1.1). 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.

Schließlich lassen sich Substitutionen auch leicht durch Software realisieren. Hierzu wird ein Feld (*array*) deklariert, dessen Einträge über die Klartextzeichen $x \in A$ adressierbar sind. Das mit x indizierte Feldelement enthält das Kryptotextzeichen, durch welches x beim Chiffriervorgang zu ersetzen ist.

Ein Nachteil hierbei ist, dass das Feld nach jedem Schlüsselwechsel neu beschrieben werden muss. Um dies zu umgehen, kann ein zweidimensionales Feld deklariert werden, dessen Einträge zusätzlich über den aktuellen Schlüsselwert k adressierbar sind. Ist genügend Speicherplatz vorhanden, um für alle $x \in A$ und alle $k \in K$ die zugehörigen Kryptotextzeichen $E(k, x)$ abspeichern zu können, so muss das Feld nur einmal initialisiert und danach nicht mehr geändert werden.

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