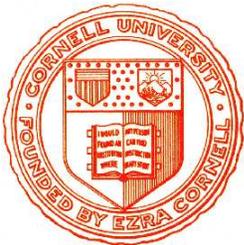


# Kryptographie und Komplexität

## Einheit 2



### Kryptoanalyse einfacher Verschlüsselungssysteme



1. Kryptosysteme & Sicherheit
2. Buchstabenorientierte Kryptosysteme
3. Blockbasierte Kryptosysteme
4. One-Time Pads und Stromchiffren

# KRYPTOGRAPHIE: TYPISCHES KOMMUNIKATIONSSZENARIO

- **Zwei Parteien wollen kommunizieren** (Alice & Bob)
  - Sicherheit: Dritte sollen nicht wissen, was übermittelt wird (Eve)
  - Übertragungskanal ist möglicherweise unsicher (Internet, Telephon,...)
- **Kommunikationsprotokoll**
  - Alice und Bob tauschen einen Schlüssel aus
    - Schlüssel muß über gesicherten Kanal übertragen werden
  - Alice **chiffriert ihre Klartext-Nachricht mit dem Schlüssel**
    - Chiffrierter Text kann auf unsicherem Kanal übertragen werden
    - **Eve kann die Originalnachricht nicht lesen**
  - Bob **dechiffriert die empfangene Nachricht mit dem Schlüssel** und erhält den ursprünglichen Klartext
- **Zentrale Fragen**
  - Kann die **Chiffrierung** von Eve **gebrochen** werden?
  - Wie kann man die **Schlüssel** einfach, schnell und sicher **austauschen**?
  - Kann das **Protokoll umgangen** werden (und gibt es Alternativen)?

- **Forschungsrichtung: sichere Kommunikation**

- Kryptographische **Einwegfunktionen**:

- schwer umzukehrende (mathematische) Verschlüsselungstechniken

- Kryptographische **Protokolle**:

- Vorschriften zu Auswahl, Austausch, Aufbewahrung von Schlüsseln

- **Forschungsrichtung: Angriffe**

- Angriffe auf Verschlüsselungstechniken (**Kryptoanalyse**)

- Algorithmen zur **Analyse von Chiffretexten**

- Algorithmen zur **Bestimmung von Schlüsseln**

- **Angriffe auf Protokolle**

- Man-in-the-middle Attacken, Phishing, Trojaner, gezieltes Raten, ...

- **Angriffe auf Hard- und Software**

- Timing, Temperatur, **Programmierfehler** (Heartbleed, Shellshock), ...

- **Beides braucht viel Mathematik**

- Zahlen- Informations-, Komplexitätstheorie, Algebra, Statistik, ...

## Mathematische Beschreibung von Kryptosystemen

- **Klartexte**

- Endliche Menge  $\mathcal{P}$  (“plaintexts”) von Wörtern eines Alphabets
- Nachrichten, die vom Sender zum Empfänger gehen sollen

- **Chiffretexte (Schlüsseltexte)**

- Endliche Menge  $\mathcal{C}$  (“ciphertexts”) von Wörtern eines Alphabets
- Texte, die über den unsicheren Kanal verschickt werden

- **Schlüssel**

- Endliche Menge  $\mathcal{K}$  (“keyspace”) möglicher Schlüssel
- Daten, die essentiell für Codierung und Decodierung sind

- **Ver- und Entschlüsselungsfunktionen**

- Funktionen  $e_K: \mathcal{P} \rightarrow \mathcal{C}$  und  $d_K: \mathcal{C} \rightarrow \mathcal{P}$  für jeden Schlüssel  $K \in \mathcal{K}$  mit der Eigenschaft  $d_K(e_K(x)) = x$  für jeden Klartext  $x \in \mathcal{P}$

# BEISPIEL: VERSCHIEBECHIFFRE ALS KRYPTOSYSTEM

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	B	C
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	X	Y	Z

- **Klartexte  $\mathcal{P}$**

- Wörter, die nur aus Großbuchstaben oder Leerzeichen bestehen
- Alphabet der Klartexte ist  $\Sigma = \{A, B, C, \dots, Z, ' '\}$

- **Chiffretexte  $\mathcal{C}$**

- Wörter über dem gleichen Alphabet  $\Sigma$

- **Schlüsselmenge  $\mathcal{K}$**

- Symbole aus dem Alphabet  $\Sigma$  (hier: Schlüssel X)

- **Verschlüsselungsfunktion  $e_K$**

- Funktion, die in jedem Wort  $x \in \mathcal{P}$  das Symbol A durch K ersetzt, das Symbol B durch den auf K (zyklisch) folgenden Buchstaben, usw.

- **Entschlüsselungsfunktionen  $d_K$**

- Funktion, die in jedem Wort  $y \in \mathcal{C}$  das Symbol K durch A ersetzt, den auf K (zyklisch) folgenden Buchstaben durch B, usw.
- Per Konstruktion gilt  $d_K(e_K(x)) = x$  für jeden Klartext  $x \in \mathcal{P}$

# KRYPTOSYSTEME: RAHMENBEDINGUNGEN

- **Jedes  $d_K$  muß zu  $e_K$  invers sein**
  - Mathematisch gesehen entspricht ein Kryptosystem einer Gruppe
- **$d_K$  und  $e_K$  müssen leicht zu berechnen sein**
  - Auch für große Datenmengen muß Ver-/Entschlüsselung schnell sein
- **Dechiffrierung von Nachrichten muß schwer sein**
  - Ohne Kenntnis von  $d_K$  darf Klartext nicht leicht zu bestimmen sein weder durch exakte Berechnung noch mit Wahrscheinlichkeitsanalysen
- **Brechen des Kryptosystems muß schwer sein**
  - Ohne Kenntnis von  $K$  darf  $d_K$  nicht (schnell) zu bestimmen sein selbst wenn das prinzipielle Verfahren bekannt ist

**$e_K$  muß eine Einwegfunktion sein**

# WANN IST EIN VERFAHREN “LEICHT” ODER “SCHWER”?

## Verwende komplexitätstheoretische Betrachtungen

- **Analysiere Rechenzeit und Platzbedarf** ↪ TI-2
  - Abschätzung: kleine Eingaben und Konstanten sind uninteressant
  - Notation:  $t \in \mathcal{O}(f)$ , falls  $t(n) \leq c * f(n)$  für ein  $c > 0$  und fast alle  $n$
  - Bezugsgröße  $n$  entspricht Größe von Schlüssel oder Nachricht
- **Nur polynomieller Aufwand ist “leicht”**
  - Zeit/Platzverbrauch beschränkt durch ein Polynom (“liegt in  $\mathcal{P}$ ”)
  - Berechnung von  $e_K(x) / d_K(y)$  muß polynomiell sein, wenn  $K$  bekannt
    - Wirklich effiziente Verfahren benötigen kleine Polynome
  - Bestimmung von  $d_K(y)$  darf nicht in  $\mathcal{P}$  liegen, wenn  $K$  unbekannt
- **Verschlüsselung ist ein  $\mathcal{NP}$ -Problem** (relativ zu  $|K|$ )
  - Sicherheit liegt in Schwierigkeit der Bestimmung des Schlüssels  $K$
  - Das Dechiffrierungsproblem kann “bestenfalls”  $\mathcal{NP}$ -vollständig sein

- **Private-Key Kryptographie** (symmetrisch)
  - Sender und Empfänger benutzen “den gleichen” geheimen Schlüssel
  - Details der Verschlüsselung müssen geheim gehalten werden da  $d_K=e_K$
  - Technisch kompliziert aber mathematisch (relativ) einfach
  - Effizient, da Hardwareunterstützung möglich (z.B. AES in WLAN Routern)
  - Problem: Schlüssel muß vorher auf sichere Art ausgetauscht werden
- **Public-Key Kryptographie** (asymmetrisch)
  - Ver- und Entschlüsselung benutzen verschiedene (inverse) Schlüssel
  - Empfänger kann Schlüssel  $e_K$  offenlegen:  $d_K$  aus  $e_K$  nicht zu bestimmen
    - Kein geheimer Schlüsselaustausch erforderlich
    - Jeder kann spontan sichere Kommunikation mit Empfänger aufbauen
  - Mathematisch aufwendig, aber sehr flexibel
- **Sichere Kommunikation großer Datenmengen**
  - Austausch eines geheimen Session-keys über Public-Key Verbindung
  - Dann Verwendung eines sicheren effizienten symmetrischen Verfahrens

## Untersuche Techniken zur Codeentschlüsselung

- **Annahme: Angreifer kennt Kryptosystem**
  - Man muß davon ausgehen, daß ein Verschlüsselungsverfahren nicht geheim bleiben kann und der Angreifer es kennt (**Kerckhoffs Prinzip**)
  - Ein Angreifer wird sicherlich alle gängigen Verfahren durchgehen
  - Kryptosysteme müssen auch unter dieser Annahme sicher sein
- **Mögliche Ziele eines Angreifers**
  - Decodierung einer Nachricht oder eines wichtigen Teils davon
  - Aufdecken des Schlüssels: zukünftige Nachrichten sind ungeschützt
  - Verfälschung, Vortäuschen einer anderen Identität, ...
- **Angriffsszenarien bestimmen Grad der Sicherheit**
  - Verfahren müssen auch dann noch sicher sein, wenn Angreifer relativ viel Informationen in der Hand hat
  - Es ist möglich, daß dem Angreifer einige Paare von Schlüsseltexten und zugehörigen Klartexten bekannt geworden sind

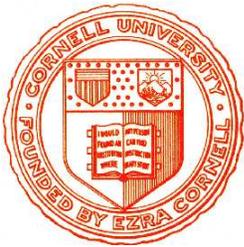
- **Ciphertext only**
  - Angreifer kennt nur die verschlüsselte Nachricht
  - Einfachster Angriff: **Brute Force** (Austesten aller Möglichkeiten)
  - Bessere Entschlüsselungsversuche setzen **statistische Analysen** ein
- **Known plaintext**
  - Angreifer kennt Paare von Klar-/Schlüsseltexten oder Fragmenten
  - Schlüssel könnte durch Analyse von Korrelationen abgeleitet werden
- **Chosen plaintext**
  - Angreifer hat (temporären) Zugang zum Verschlüsselungsverfahren  
z.B. auf eine Chipkarte oder ein Public-Key Verfahren
  - Liefert beliebig viele Klar-/Schlüsseltextpaare
- **Chosen ciphertext**
  - Angreifer hat temporären Zugang zum Entschlüsselungsverfahren  
(als Black Box) und versucht, den Schlüssel zu extrahieren

- **Perfekte (“absolute”) Sicherheit** (Unconditional Security)
  - Kein Angreifer kann das System jemals brechen,  
... selbst wenn beliebig viel Zeit und Rechnerkapazität bereitsteht
  - Schlüsseltext darf keinerlei Information über Klartext enthalten
  - Nur mit **One-Time Pads** erreichbar
- **Effiziente Sicherheit** (Computational Security)
  - Brechen des Kryptosystems erfordert **mindestens  $n$  Schritte** ( $n > 10^{100}$ )
    - Theoretisches Ziel, in dieser Form praktisch nicht nachweisbar
  - Realistische Variante: **Bestes bekanntes Verfahren** zum Brechen des Kryptosystems benötigt **mindestens  $n$  Schritte**
  - **Geläufigster aller Sicherheitsbegriffe** (“praktische Sicherheit”)
- **Beweisbare Sicherheit** (Provable Security)
  - Sicherheit des Systems basiert auf einem mathematischen Problem, das als schwer lösbar (z.B.  $\mathcal{NP}$ -vollständig) bekannt ist
  - Keine absolute Sicherheit, solange  $\mathcal{P} \neq \mathcal{NP}$  ungeklärt (aber nahe daran)

# Kryptographie und Komplexität

## Einheit 2.1

### Buchstabenorientierte Kryptosysteme Implementierung und Analyse



1. Verschiebechiffren
2. Affine Chiffren
3. Allgemeine Substitutionschiffre
4. Polyalphabetische Chiffren

# SUBSTITUTIONBASIERTE CHIFFREN

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

- **Buchstaben werden durch andere ersetzt**

Aus ENDE UM DREI UHR wird HTAHIMBIARHZIMGR

– Originaltext durch Rückwärtsersetzen ermittelbar

- **Unterschiedlich aufwendige Varianten**

– **Verschiebechiffre**: Buchstaben werden um festen Betrag verschoben  
Sehr einfache Ver-/Entschlüsselung, leicht zu merkender Schlüssel

– **Affine Chiffre**: Buchstaben werden um ein Vielfaches ihrer eigenen  
Position und einen konstanten Betrag verschoben

Ver-/Entschlüsselung aufwendiger, leicht zu merkender Schlüssel

– **Allgemeine Substitution**: Buchstaben im Alphabet werden permutiert  
Ver-/Entschlüsselung etwas mühsam, langer Schlüssel

# VERSCHIEBECHIFFREN

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	B	C
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	X	Y	Z

- **Eine der ältesten Verschlüsselungstechniken**
  - (Zyklische) Verschiebung der Buchstaben im Alphabet
  - Schon vor über 2000 Jahren von Julius Cäsar eingesetzt
- **Verschlüsselung durch Vorwärtsschieben**
  - Geheimer **Schlüssel** ist Buchstabe, der das A ersetzt (hier X)
  - **Schlüsseltext** wird durch Verschieben der Klartextbuchstaben erzeugt  
Aus INFORMATIK BRAUCHT MATHEMATIK  
wird **EJBKNIXPEGWYNXQZDPWIXPDAIXPEG**
- **Entschlüsselung durch Rückwärtsschieben**
  - Adressat muß den Schlüssel (und das Chiffrierverfahren) kennen
  - **Klartext** wird buchstabenweise wiederhergestellt
- **Zu leicht zu brechen da nur 27 mögliche Schlüssel**
  - Ausprobieren aller Schlüssel sogar von Hand durchführbar

# VERSCHIEBECHIFFREN AUS MATHEMATISCHER SICHT

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	B	C
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	X	Y	Z

- **Buchstaben können mit Zahlen identifiziert werden**
  - Bei  $n$  Symbolen erhält jeder Buchstabe eine Zahl zwischen 0 und  $n-1$
  - Schlüssel  $K$  wird ebenfalls eine Zahl zwischen 0 und  $n-1$
  - Im Beispiel war  $n = 27$  und  $K = 23$  (der Code von 'X')
- **Chiffrierverfahren entspricht Modulararithmetik**
  - Verschlüsselung durch Verschieben entspricht Addition modulo  $n$ :
$$e_K(x) = x + K \bmod n$$
  - Entschlüsselung entspricht Subtraktion modulo  $n$ :
$$d_K(y) = y - K \bmod n$$
- **Implementierung benötigt Restklassenarithmetik**
  - Arithmetische Operationen modulo einer Zahl  $n$

## ● Teilbarkeit

- $a|n$  ( $a$  teilt  $n$ ), falls  $n = a \cdot b$  für ein  $b \in \mathbb{Z}$
- Aus  $a|b$  folgt  $|a| \leq |b|$ , falls  $b \neq 0$
- Aus  $a|b$  und  $b|a$  folgt  $|a| = |b|$
- Aus  $a|b$  und  $b|c$  folgt  $a|c$
- Aus  $a|b$  folgt  $ac|bc$  für alle  $c \in \mathbb{Z}$

Transitivität

Multiplikativität

## ● Division mit Rest

- Für  $a \in \mathbb{Z}, b > 0$  gibt es eindeutige Zahlen  $q, r$  mit  $a = qb + r$  und  $0 \leq r < b$
- Bezeichnung:  $q = \lfloor a/b \rfloor$  und  $r = a \bmod b$

## ● Restklassen modulo $n$

- $a \equiv b \pmod{n}$  ( $a$  kongruent zu  $b$  modulo  $n$ ), falls  $n \mid b - a$
- Kongruenz modulo  $n$  ist eine Äquivalenzrelation
- $[a] = a + n\mathbb{Z} := \{b \mid b \equiv a \pmod{n}\}$  ist die Restklasse von  $a$  modulo  $n$
- Die Menge aller Restklassen modulo  $n$  wird mit  $\mathbb{Z}/n\mathbb{Z}$  bezeichnet
- $\mathbb{Z}_n = \{0, \dots, n-1\}$  ist der Standardvertreter für die Klassen in  $\mathbb{Z}/n\mathbb{Z}$

## • Addition und Multiplikation in $\mathbb{Z}/n\mathbb{Z}$

– Definiert auf Vertretern:  $[a] + [b] := [a + b]$   $[a] \cdot [b] := [a \cdot b]$

– Einfachere Darstellung in  $\mathbb{Z}_n$ :  $a+_nb := a+b \bmod n$   $a\cdot_nb := a\cdot b \bmod n$

Index  $n$  wird weggelassen, wenn aus dem Kontext ersichtlich

## • Rechengesetze: Für alle $a, b, c \in \mathbb{Z}_n$ gilt

–  $a + b \in \mathbb{Z}_n$

Abgeschlossenheit der Addition

–  $(a + b) + c = a + (b + c)$

Assoziativität der Addition

–  $a + 0 = 0 + a = a$

Neutrales Element der Addition

–  $a + (n-a) = (n-a) + a = 0$

Inverse Elemente der Addition

–  $a + b = b + a$

Kommutativität der Addition

–  $a \cdot b \in \mathbb{Z}_n$

Abgeschlossenheit der Multiplikation

–  $(a \cdot b) \cdot c = a \cdot (b \cdot c)$

Assoziativität der Multiplikation

–  $a \cdot 1 = 1 \cdot a = a$

Neutrales Element der Multiplikation

–  $a \cdot (b + c) = a \cdot b + a \cdot c$

Distributivität der Multiplikation

–  $a \cdot b = b \cdot a$

Kommutativität der Multiplikation

# KOMPLEXITÄT DER OPERATIONEN AUF RESTKLASSEN

- **Addition ganzer Zahlen** (ohne Reduktion modulo  $n$ )
  - Binäre Addition benötigt  $\mathcal{O}(1)$  Schritte pro Bit
  - Addition  $a + b$  erfordert Zeit  $\mathcal{O}(\max\{\|a\|, \|b\|\})$  ( $\|a\|$ : Anzahl der Bits von  $a$ )
- **Multiplikation ganzer Zahlen**
  - Schriftliche Multiplikation mit  $b$  benötigt  $\mathcal{O}(\|a\|)$  Schritte pro Bit von  $b$
  - Multiplikation  $a \cdot b$  erfordert Zeit  $\mathcal{O}(\|a\| \cdot \|b\|)$
- **Division mit Rest:**  $q = \lfloor a/b \rfloor$ ,  $r = a \bmod b$ 
  - Schriftliche Division durch  $b$  benötigt  $\mathcal{O}(\|b\|)$  Schritte pro Bit von  $q$
  - Bestimmung von  $r$  und  $q$  erfordert Zeit  $\mathcal{O}(\|q\| \cdot \|b\|)$  und Platz  $\mathcal{O}(\|a\| + \|b\|)$
- **Addition:**  $a +_n b := a + b \bmod n$ 
  - Wegen  $a, b < n$  und  $a + b < 2n$  erfordert  $a +_n b$  insgesamt Zeit  $\mathcal{O}(\|n\|)$
- **Subtraktion:**  $a -_n b := a +_n (n - b)$ 
  - Gleicher Aufwand wie Addition: Zeit  $\mathcal{O}(\|n\|)$
- **Multiplikation:**  $a \cdot_n b := a \cdot b \bmod n$ 
  - Wegen  $a \cdot b < n^2$  erfordert  $a \cdot_n b$  insgesamt Zeit  $\mathcal{O}(\|n\|^2)$

# PROGRAMMIERUNG DER VERSCHIEBECHIFFRE

- **Einlesen des Klartexts:**      INFORMATIK BRAUCHT MATHEMATIK

- **Umwandlung jedes Buchstaben in eine Zahl**

– Symbole des Alphabets sind durchnummeriert

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	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

– Buchstaben des Textes werden schrittweise durch ihre Zahl ersetzt  
Implementierbar als Tabellensuche oder fest verdrahtete Operation

– Ergebnis ist Liste von Zahlen in  $\mathbb{Z}_n$  ( $n =$  Größe des Alphabets)

[8;13;5;14;17;12;0;19;8;10;26;1;17;0;20;2;7;19;26;12;0;19;7;4;12;0;19;8;10]

- **Verschiebung durch Addition in  $\mathbb{Z}_n$**

– Für jede Zahl  $x$  der Liste berechne  $e_K(x) = x + K \bmod n$

[4;9;1;10;13;8;23;15;4;6;22;24;13;23;16;25;3;15;22;8;23;15;3;0;8;23;15;4;6]

– Für die Entschlüsselung berechne  $d_K(x) = x - K \bmod n$

- **Rückumwandlung der Zahlen in Buchstaben**

– Wieder Tabellensuche oder fest verdrahtete Operation

EJBKNIXPEGWYNXQZDPWIXPDAIXPEG

# BERECHNUNGS-AUFWAND DER VERSCHIEBECHIFFRE

- **Meßgröße ist Länge des Klartextes  $w$** 
  - Für Entschlüsselung: Länge des Schlüsseltextes
- **Umwandlung zwischen Buchstaben und Zahlen**
  - Pro Buchstaben linearer Zeitaufwand relativ zur Alphabetgröße  $n$   
Da  $n$  konstant (und klein) ist, gilt der Aufwand als konstant  $\mathcal{O}(1)$
  - **Gesamtaufwand** für Umwandlung eines Klartextes ist linear  $\mathcal{O}(|w|)$
- **Verschiebung innerhalb der Zahlenfolge**
  - Addition/Subtraktion in  $\mathbb{Z}_n$  benötigt Zeit  $\mathcal{O}(\|n\|)$   
Da  $n$  konstant (und klein) ist, liegt der Aufwand in  $\mathcal{O}(1)$
  - **Gesamtaufwand** für Verschiebung innerhalb der Liste ist linear  $\mathcal{O}(|w|)$

**Die Verschiebechiffre benötigt lineare Zeit**

## Ziel ist Bestimmung von Schlüssel und Klartext

- **Brute-force Attacken sind durchführbar**
  - Vollständige Schlüsselsuche: jeder mögliche Schlüssel wird probiert
  - Es werden alle möglichen Klartexte generiert
  - Der Originaltext ist einer der erzeugten Texte
  - Nur wenige erzeugte Texte sind sprachlich akzeptabel (Wörterbuch)
  - Akzeptable Texte können “von Hand” untersucht werden
- **Geringer Aufwand für “Ciphertext only” Attacke**
  - Aufwand zur Erzeugung eines möglichen Klartextes ist  $\mathcal{O}(|w|)$
  - Aufwand zur Überprüfung der Texte mit Wörterbuch ist konstant
  - Schlüssel der Verschiebechiffre sind die Buchstaben des Alphabets  
Die Anzahl möglicher Schlüssel ist fest (unabhängig von Textlänge)
  - **Verschiebechiffre kann in linearer Zeit gebrochen werden**
  - Selbst eine Dechiffrierung von Hand ist leicht  
Aus **MVLMHBUHLZMQHBPZ**  
wird **ENDE UM DREI UHR**

( $K=8$ )

# DIE AFFINE CHIFFRE

## Erschwere Brute-force Attacken durch größere Anzahl von Schlüsseln

- **Erweitere Verschiebung auf affine Funktionen**

- Neben der Addition wird eine Multiplikation eingesetzt
- Verschlüsselung von  $x$  wird  $e_K(x) = ax + b \pmod n$
- Entschlüsselung von  $y$  wird  $d_K(y) = a^{-1}(y - b) \pmod n$
- Schlüssel  $K$  ist ein Paar  $(a, b) \in \mathbb{Z}_n^2$  (Verschiebechiffre:  $a = 1$ )

- **Anwendungsbeispiel mit  $n = 27$  und  $K = (8, 3)$**

- Aus ENDE UM DREI UHR  $\hat{=} [4;13;3;4;26;20;12;26;3;17;4;8;26;20;7;17]$   
wird  $[8;26;0;8;22;1;18;22;0;4;11;13;22;1;5;4]$   
 $\hat{=} \text{"I AIWBSWAEINWBFE"}$

- Entschlüsselungsfunktion ist  $d_K(y) = 17(y - 3) \pmod{27}$

- **Nicht jede affine Funktion ist brauchbar**

- $K = (18, 2)$  bildet 3,6,9,12,15,18,21,24 jeweils auf die Zahl 2 ab
- $a = 18$  hat in  $\mathbb{Z}_{27}$  kein eindeutiges multiplikatives Inverses
- Eine eindeutige Entschlüsselung ist nicht möglich

# INVERTIERUNG IN $\mathbb{Z}_n$ : IST $a \cdot_n x = 1$ EINDEUTIG LÖSBAR?

- **Notwendig:  $a$  und  $n$  haben keine gemeinsamen Teiler**
  - Ist die Gleichung  $a \cdot_n x = 1$  ganzzahlig lösbar, so ist  $\gcd(a, n) = 1$   
 $\gcd(a, n)$  ist der größte gemeinsame Teiler, also die größte Zahl  $t \in \mathbb{N}$  mit  $t \mid a$  und  $t \mid n$
  - Beweis: Sei  $t \in \mathbb{N}$  ein Teiler von  $a$  und  $n$ , also  $a = t \cdot i$  und  $n = t \cdot j$   
Wenn  $a \cdot_n x = 1$ , so gibt es ein  $k$  mit  $t \cdot i \cdot x = a \cdot x = n \cdot k + 1 = t \cdot j \cdot k + 1$   
Es folgt  $t \cdot (i \cdot x - j \cdot k) = 1$  und damit  $t = 1$
- **Eindeutigkeit: es gibt maximal eine Lösung**
  - Gilt  $\gcd(a, n) = 1$  und  $a \cdot_n x = a \cdot_n y$  für  $x, y \in \mathbb{Z}_n$ , so ist  $x = y$
  - Beweis: Sei  $a \cdot x = a \cdot y + j \cdot n$  und o.B.d.A.  $y < x$   
Dann gilt  $a \cdot (x - y) = j \cdot n$  und  $n \mid x - y$  wegen  $\gcd(a, n) = 1$   
Wegen  $x, y \in \mathbb{Z}_n$  folgt  $x - y = 0$ , also  $x = y$
  - Sind  $x, y \in \mathbb{Z}_n$  Lösungen für  $a \cdot_n x = 1$  und  $a \cdot_n y = 1$ , so ist  $x = y$
- **Hinreichend:  $a$  und  $n$  sind teilerfremd**
  - Ist  $\gcd(a, n) = 1$ , so ist die Gleichung  $a \cdot_n x = 1$  ganzzahlig lösbar
  - Beweis: Jede Gleichung  $a \cdot_n x = z$  hat maximal eine Lösung in  $\mathbb{Z}_n$   
Also nimmt  $a \cdot_n x$  für jedes  $x \in \mathbb{Z}_n$  einen anderen Wert aus  $\mathbb{Z}_n$  an  
Somit muß jeder Wert  $z \in \mathbb{Z}_n$  erreicht werden, insbesondere auch  $z = 1$

# INVERTIERUNG IN $\mathbb{Z}_n$ - ALGORITHMISCH

- **Vollständige Suche ist prinzipiell möglich**
  - Affine Chiffre benötigt Invertierung nur einmal für Schlüsselerzeugung
  - Für kleine  $n$  findet Computer multiplikative Inverse sehr schnell
  - Lösung muß existieren, wenn  $\gcd(a, n) = 1$
- **Invertierung durch Suche ist ineffizient**
  - Von Hand auch für kleine Alphabete kaum noch durchzuführen
  - Modernen Verfahren benötigen Invertierung in  $\mathbb{Z}_n$  für extrem große  $n$
- **Elegantere Lösung mit euklidischem Algorithmus**
  - Euklidischer Algorithmus berechnet  $\gcd(a, n)$  sehr effizient
  - Ablauf des euklidischen Algorithmus enthält Informationen zur Lösung der Gleichung  $a \cdot x + n \cdot y = \gcd(a, n)$  für gegebenes  $a, n$

**Analysiere und erweitere euklidischen Algorithmus**

- **Grundlage: Satz über Eigenschaften von  $gcd$**

- Für  $a, n \in \mathbb{Z}$  ist  $gcd(a, n) = gcd(|a|, |n|)$
- Für  $n = 0$  ist  $gcd(a, n) = a$
- Für  $n > 0$  ist  $gcd(a, n) = gcd(n, a \bmod n)$

- **Einfache funktionale Implementierung**

```
let rec gcd' a n
  = if n=0 then a else gcd' n (a mod n);;
let gcd a n
  = gcd' (abs a) (abs n);;
```

- Korrektheit folgt unmittelbar aus obigem Theorem
- Terminierung folgt aus einfachem Monotonieargument  
oder expliziter Laufzeitanalyse (nächste Folie)
- Imperative Implementierung mit einfacher `while` Schleife möglich

# EUKLIDISCHER ALGORITHMUS: LAUFZEITANALYSE

- **Wichtig ist Anzahl der Aufrufe**

- In  $i$ -ten Aufruf wird  $n_{i+1} := a_i \bmod n_i$  berechnet  $\mathcal{O}(\lfloor a_i/n_i \rfloor \cdot \|n_i\|)$
- Der Folgeaufruf setzt  $a_{i+1} := n_i$  und  $n_{i+1}$  wie berechnet
- Algorithmus terminiert nach  $k+1$  Schritten, wenn  $n_{k+1} = 0$   
Terminierung ist sicher, da Folge der  $n_i$  streng monoton fallend

- **Modulare Reduktion verkürzt “exponentiell”**

- Zeige, daß Folge der  $n_i$  exponentiell fällt:  $n_i > \vartheta^{k-i}$  für ein  $\vartheta > 1$
- Aufgrund der Monotonie ist  $n_{k+1} = 0$ ,  $n_k \geq 1$  und  $n_{k-1} \geq 2$
- Sei  $n_j > \vartheta^{k-j}$  für alle  $j \geq i$  bewiesen
- Dann gilt: 
$$n_{i-1} = a_i = \lfloor a_i/n_i \rfloor \cdot n_i + n_{i+1} \geq n_i + n_{i+1} > \vartheta^{k-i} + \vartheta^{k-(i+1)} = \vartheta^{k-(i-1)}(1/\vartheta + 1/\vartheta^2)$$
- Zu bestimmen bleibt ein  $\vartheta > 1$  mit  $1/\vartheta + 1/\vartheta^2 \geq 1$   
Multiplikation mit  $\vartheta^2$  ergibt  $\vartheta + 1 > \vartheta^2$  bzw.  $1 \geq (\vartheta - 1/2)^2 - 1/4$   
Für  $\vartheta = 1/2 + \sqrt{5/4}$  gilt also immer  $n_i > \vartheta^{k-i}$

- **Es gibt nur logarithmisch viele Aufrufe**

- Wegen  $n_1 = n$  folgt  $k < \log n / (\log \vartheta) + 1 < 1.441 \log n + 1$

# ERWEITERTER EUKLIDISCHER ALGORITHMUS

Bestimme  $x, y \in \mathbb{Z}$  mit  $a \cdot x + n \cdot y = \gcd(a, n)$

- **Bestimme  $x, y$  während Berechnung des  $\gcd$**

- Euklidischer Algorithmus bestimmt  $\gcd$  durch Folge von Werten  $n_i$
- $\gcd(a, n)$  ist Wert von  $n_k$  im Schritt direkt vor der Terminierung
- Analog bestimme  $x, y$  durch Folge von Werten  $x_i, y_i$  mit  $a \cdot x_i + n \cdot y_i = n_i$

- **Herleitung des Algorithmus aus der Invariante**

- Es ist  $n_{i+1} = a_i \bmod n_i = a_i - \lfloor a_i/n_i \rfloor \cdot n_i = n_{i-1} - q_i \cdot n_i$   $q_i = \lfloor n_{i-1}/n_i \rfloor$   
 $= a \cdot x_{i-1} + n \cdot y_{i-1} - q_i \cdot a \cdot x_i - q_i \cdot n \cdot y_i$   
 $= a \cdot (x_{i-1} - q_i \cdot x_i) + n \cdot (y_{i-1} - q_i \cdot y_i) = a \cdot x_{i+1} + n \cdot y_{i+1}$

- Startwerte:  $n_1 = n = a \cdot 0 + n \cdot 1$  also  $x_1 := 0$  und  $y_1 := 1$

Der Einfachheit halber:  $n_0 = a = a \cdot 1 + n \cdot 0$  also  $x_0 := 1$  und  $y_0 := 0$

- **Funktionale Implementierung**

```
let rec egcd' a n x y x' y'  
  = if n=0 then (a,x',y')  
    else egcd' n (a mod n) (x'-a/n*x) (y'-a/n*y) x y;;  
let egcd a n = egcd' (abs a) (abs n) 0 1 1 0;;
```

# INVERTIERUNG IN $\mathbb{Z}_n$ : IMPLEMENTIERUNG & LAUFZEIT

## ● Instanz des erweiterten euklidischen Algorithmus

- $\text{egcd}(a, n)$  liefert  $g = \text{gcd}(a, n)$  sowie  $x, y \in \mathbb{Z}$  mit  $a \cdot x + n \cdot y = g$   
Es folgt  $a \cdot x \bmod n = g$
- Da  $x$  negativ sein kann, bestimme  $x' := x \bmod n$
- Dann ist  $x' \in \mathbb{Z}_n$  und  $a \cdot_n x' = g$  also  $x' = a^{-1}$ , falls  $\text{gcd}(a, n) = 1$

## ● Laufzeitanalyse (bei $k$ Iterationen)

- Bestimmung von  $n_{i+1} := a_i \bmod n_i$  benötigt Zeit  $\mathcal{O}(\|q_i\| \cdot \|n_i\|)$
- Bestimmung von  $x_{i+1} = x_{i-1} - q_i \cdot x_i$  und  $y_{i+1}$  benötigt Zeit  $\mathcal{O}(\|q_i\| \cdot \|n_i\|)$
- Wegen  $n_i < n$  benötigt der gesamte Algorithmus maximal  $\mathcal{O}(\sum_{i=1}^k \|q_i\| \cdot \|n\|)$
- Wegen  $\prod_{i=1}^k q_i < a$  (also  $\sum_{i=1}^k \|q_i\| < \|a\|$ ) und  $k \in \mathcal{O}(\|n\|)$   
liegt die Gesamtlaufzeit von  $\text{egcd}(a, n)$  in  $\mathcal{O}(\|a\| \cdot \|n\|)$
- Die Zeit für die modulare Reduktion von  $x$  ist hiergegen vernachlässigbar

**Invertierung in  $\mathbb{Z}_n$  benötigt eine Gesamtlaufzeit in  $\mathcal{O}(\|n\|^2)$**

# BERECHNUNGSaufWAND DER AFFINEN CHIFFRE

- **Aufwand für Auswahl des Schlüssels (einmalig)**
  - Alice wählt  $a \in \mathbb{Z}_n$  mit  $\gcd(a, n) = 1$  und  $b \in \mathbb{Z}_n$  pro test  $\mathcal{O}(\|n\|^2)$
  - Zahl der Tests hängt ab von Zahl der zu  $n$  teilerfremden Elemente
  - Bob bestimmt das multiplikativ Inverse  $a^{-1} \in \mathbb{Z}_n$   $\mathcal{O}(\|n\|^2)$
- **Aufwand für Ver- und Entschlüsselung**
  - Umwandlung zwischen Buchstaben und Zahlen  $\mathcal{O}(|w|)$
  - Berechnen der affinen Funktion  $(a \cdot_n x +_n b / a^{-1} \cdot_n (y -_n b))$   $\mathcal{O}(|w| \cdot \|n\|^2)$
  - Gesamtaufwand bei festem Alphabet ( $n$  klein) ist linear  $\mathcal{O}(|w|)$
- **Aufwand für Brute-Force Attacke**
  - Pro Versuch Aufwand  $\mathcal{O}(|w|)$  für die Entschlüsselung
  - Anzahl der möglichen Schlüssel ist  $\varphi(n) \cdot n$ , wobei  $\varphi(n)$  die Anzahl der zu  $n$  teilerfremden Elemente von  $\mathbb{Z}_n$  bezeichnet
  - Bei festem Alphabet ist der affine Chiffre in linearer Zeit zu brechen
  - Nicht sicher gegen Brute-Force Attacken mit einem Computer  
aber für eine Dechiffrierung von Hand gibt es zu viele Schlüssel  
( $18 \cdot 27 = 486$  Schlüssel bei nur 27 Symbolen)

# MATHEMATIK: DIE EULERSCHE $\varphi$ -FUNKTION

- $\varphi(n) = |\{a \in \mathbb{Z}_n \mid \gcd(a, n) = 1\}|$ 
  - Anzahl der Elemente von  $\mathbb{Z}_n$ , die zu  $n$  **relativ prim** (teilerfremd) sind
  - Für jede Primzahl  $p$  ist  $\varphi(p) = p-1$   
da mit Ausnahme der 0 jedes Element von  $\mathbb{Z}_p$  zu  $p$  teilerfremd ist
  - Für jede Primzahl  $p$  ist  $\varphi(p^e) = p^e - p^{e-1}$   
da alle Vielfachen von  $p$  in  $\mathbb{Z}_p$  gemeinsame Teiler mit  $p^e$  haben
  - Für die Primfaktorzerlegung  $p_1^{e_1} \dots p_k^{e_k}$  von  $n$  ist  $\varphi(n) = \prod_{i=1}^k (p_i^{e_i} - p_i^{e_i-1})$   
Alle Vielfachen jedes der  $p_i$  in  $\mathbb{Z}_n$  haben gemeinsame Teiler mit  $n$   
Gemeinsame Vielfache der  $p_i$  dürfen nicht mehrfach gezählt werden
- **Das ASCII Alphabet hat 95 echte Symbole**
  - Wegen  $\varphi(95) = 18 * 4 = 72$  hat die affine Chiffre 6840 Schlüssel in  $\mathbb{Z}_{95}$
- **Für jede Primzahl  $p$  ist  $\mathbb{Z}_p$  ein Zahlkörper**
  - Jede von Null verschiedene Zahl hat ein multiplikativ Inverses
  - Eigenschaft ist **wichtiger Ausgangspunkt für sichere Kryptosysteme**

## • Halbgruppen, Monoide und Gruppen

- **Halbgruppe**: Menge  $G$  mit assoziativer Abbildung  $\circ : G \times G \rightarrow G$
- **Monoid**: Halbgruppe mit neutralem Element (oft **Null** genannt)
- **Gruppe**: Monoid, bei dem jedes Element ein Inverses hat  
Die **Ordnung** einer Gruppe ist die Anzahl ihrer Elemente
- **abelsche (Halb-)Gruppe**: (Halb-)Gruppe mit kommutativer Abbildung  
 $(\mathbb{Z}_n, +_n)$  ist eine abelsche Gruppe,  $(\mathbb{Z}_n, \cdot_n)$  ist ein abelsches Monoid

## • Ringe und Körper

- **Ring**: Menge  $R$  mit Abbildungen  $+$  und  $\cdot$ , wobei  $(R, +)$  abelsche Gruppe,  $(R, \cdot)$  Halbgruppe und  $\cdot$  distributiv
- **Ring mit Einselement**: Multiplikation  $\cdot$  hat neutrales Element
- **Kommutativer Ring**: Ring  $(R, +, \cdot)$  mit kommutativer Multiplikation  
 $(\mathbb{Z}_n, +_n, \cdot_n)$  ist kommutativer Ring mit Einselement
- **Körper**: kommutativer Ring mit Einselement, bei dem jedes von Null verschiedene Element invertierbar (bzgl.  $\cdot$ ) ist

**Satz:**  $(\mathbb{Z}_n, +_n, \cdot_n)$  ist genau dann ein Körper wenn  $n$  Primzahl ist

- **Alternative zur Brute-Force Attacken**

- Brute-Force Attacken sind nur noch mit dem Computer möglich
- Dennoch ist eine von-Hand Analyse von Schlüsseltexten möglich

- **Betrachte Häufigkeiten von Buchstaben im Text**

- Klartextbuchstaben werden in feste Schlüsseltextsymbole umgewandelt
- Buchstabenhäufigkeiten in deutschen Texten sind **sehr verschieden**
  - Sehr häufig: **E**  $\hat{=}$  14.7%, **'**  $\hat{=}$  12%, **N**  $\hat{=}$  8.8%, **I, S, R, A, T, D, H, U**
  - Häufig: **B, C, F, G, K, L, M, O, W, Z**
  - Selten: **J, P, V, Y, Q**  $\hat{=}$  0.014%, **X**  $\hat{=}$  0.013%
- **Statistische Analysen liefern bei langen Texten wertvolle Informationen**

- **Analysiere je zwei mögliche Zuordnungen**

- Zuordnungen liefern zwei Gleichungen mit zwei Unbekannten
$$a \cdot_n x_1 +_n b = y_1 \quad \text{und} \quad a \cdot_n x_2 +_n b = y_2$$
- Gleichungssystem liefert Vorschlag für Schlüssel  $K = (a, b)$
- Im Mißerfolgsfall ( $\gcd(a, n) \neq 1$ ) versuche andere Zuordnungen

# KRYPTOANALYSE DER AFFINEN CHIFFRE

## • **Untersuche den (langen) Chiffretext**

HPOGNUEAIJSEEKYSOVSBG SGHPOEOYGZDYJOESXBGVPXHOLGNJEGOP  
XOGNYLGQSOXHJPAIOYGCJNSES YGDIXOGOPBOXOGSXLOYJNBOXGELNLL

– Bestimme die Häufigkeit der Buchstaben im Chiffretext

A	2	B	4	C	1	D	2	E	8	F	0	G	14	H	4	I	3	
J	6	K	1	L	6	M	0	N	6	O	16	P	6	Q	1	R	0	
S	9	T	0	U	1	V	2	W	0	X	9	Y	7	Z	1	'	'	1

## • **Versuche Zuordnung der häufigsten Buchstaben**

– Ordne O, G, S, X den Klartextbuchstaben E, ' ', N, I zu

– Erster Versuch:  $E \mapsto O$ ,  $' ' \mapsto G$  liefert  $e_K(4) = 14$  und  $e_K(26) = 6$

– Liefert Gleichungssystem  $4 \cdot_{27} a +_{27} b = 14$  und  $26 \cdot_{27} a +_{27} b = 6$

Lösung:  $22 \cdot_{27} a = 19$  also  $a = 7$  und  $b = 13$

Wegen  $\gcd(7, 27) = 1$  ist  $(7, 13)$  ein gültiger Schlüssel

– Es ist nicht immer so einfach: bei Mißerfolg teste andere Zuordnungen

## • **Entschlüssele Text mit $d_K(y) = 4 \cdot_{27}(y -_{27} 13)$**

DIE ABSCHLUSSPRUEFUNG ZU DIESER VORLESUNG FINDET ALS EINE  
ART MUENDLICHER KLAUSUR OHNE EIGENE UNTERLAGEN STATT

# ALLGEMEINE SUBSTITUTIONSCHIFFRE

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

- **Permutationstabelle für Ersetzung von Buchstaben**

- Chiffretext entsteht durch Austausch der Buchstaben gemäß Tabelle

Aus INFORMATIK BRAUCHT MATHEMATIK

wird ZTPSRBXVZ INRXMYGVIBXVDGHBVZ

- Originaltext durch Rückwärtsersetzung ermittelbar

- **Signifikante Erhöhung der Zahl möglicher Schlüssel**

- Komplexer als Affine Chiffre, daher nicht so leicht zu brechen

- **Mathematisch: Permutation  $\pi$  der Zahlen  $[0, \dots, n-1]$**

- Verschlüsselung:  $e_\pi(x) = \pi(x)$ , Entschlüsselung:  $d_\pi(y) = \pi^{-1}(y)$

- Im Beispiel ist  $\pi = [23;13;24;0;7;15;14;6;25;16;26;22;1;19;18;5;11;17;2;21;12;20;4;10;9;3;8]$

Aus ENDE UM DREI UHR  $\hat{=} [4;13;3;4;26;20;12;26;3;17;4;8;26;20;7;17]$

wird  $[7;19;0;7;8;12;1;8;0;17;7;25;8;12;6;17]$   $\hat{=} \text{HTAHIMBIARHZIMGR}$

Entschlüsselungspermutation ist "DMSZWPHE YXQUBGFJRONVTLACIK"

bzw.  $\pi^{-1} = [3;12;18;25;22;15;7;4;26;24;23;16;20;1;6;5;9;17;14;13;21;19;11;0;2;8;10]$

- Leicht zu programmieren und anzuwenden, langer Schlüssel ( $n$  Zahlen)

## ● Darstellung des Schlüssels als Permutation

- Liste der Buchstaben, die A . . Z, ' ' codieren
  - z.B. "XNYAHPOGZQ WBTSFLRCVMUEKJDI"
- Liste der zugehörigen Zahlen (d.h. Indizes der Buchstaben)
  - z.B. [23;13;24;0;7;15;14;6;25;16;26;22;1;19;18;5;11;17;2;21;12;20;4;10;9;3;8]
- Länge des Schlüssels entspricht Größe  $n$  des Alphabets

## ● Aufwand für Ver- und Entschlüsselung

- Umwandlung zwischen Buchstaben und Zahlen  $\mathcal{O}(|w|)$
- Ersetzen eines Buchstabens gemäß Permutation (Listensuche)  $\mathcal{O}(n)$
- **Gesamtaufwand bei festem Alphabet ( $n$  klein) ist linear**  $\mathcal{O}(|w|)$

## ● Aufwand für Brute-Force Attacke

- Pro Versuch Aufwand  $\mathcal{O}(|w|)$  für die Entschlüsselung
- Anzahl der möglichen Schlüssel ist  $n!$  (10<sup>28</sup> im Beispiel)
- **Bei mehr als 26 Buchstaben ist die Substitutionschiffre nicht zu brechen**
- Sicher gegen Brute-Force Attacken mit Computern
- Dennoch ist eine von-Hand Analyse von Schlüsseltexten möglich

- **Chiffre bleibt anfällig für statistische Analysen**

- Substitution ändert Buchstabenhäufigkeit nicht
- Häufigste Buchstaben sind (deutsch): E, ' ', N, I, S, R, A, T, D, H, U
- Es gibt auch häufige Doppel-/Dreifachkombinationen
  - EN ER CH ND EI DE IN ES TE IE
  - EIN ICH NDE DIE UND DER CHE END GEN SCH
- Liefert wertvolle Informationen bei Analyse längerer Texte

- **Erzeuge partiellen Klartext und ergänze Lücken**

- Häufigste Buchstaben und Bi-/Trigramme liefern möglichen Teiltext
- Menschen können Lücken in unvollständigen Texten oft leicht ergänzen

.R.P..ANA..SE IS. .UEHSA.

KRYPTOANALYSE IST MUEHSAM

- Computer können erste Vorschläge mit Wörterbüchern erzeugen

# KRYPTOANALYSE DER SUBSTITUTIONSCHIFFRE: BEISPIEL

## • Untersuche den (langen) Chiffretext

AZHCIZCVIHZTICHGRIWXTOHRIRHWXVZUIDMPXHWZWOIGZTOHCYGRZHNHTHRIVHKVI  
AHTIEZRIPMHRIZTHIAHYGZPPRZHRMTOIBZVIMTCHRHTIUHRPXGRHTINHTSHVZOHT

– Bestimme die Häufigkeit der Buchstaben im Chiffretext

A	3	B	1	C	5	D	1	E	1	F	0	G	5	H	24	I	17
J	0	K	1	L	0	M	4	N	2	O	5	P	5	Q	0	R	12
S	1	T	12	U	2	V	6	W	4	X	4	Y	2	Z	13	'	0

## • Versuche Zuordnung der häufigsten Buchstaben

– Ordne H I Z R T V den Klartextbuchstaben E, ' ', N, I, S, R zu

– Chiffrierung  $E \mapsto H$ ,  $' \mapsto I$  ist eindeutig, Häufigkeit von Z R T ähnlich

– Häufigste Bi-/Trigramme (ohne Chiffrierung I des Leerzeichens)

HR	6	HT	6	GR	3	OH	3	RH	3	TO	3	ZH	3
HZT	2	NHT	2	RHT	2	RZH	2	TOH	2				

– Mit  $E \mapsto H$  folgt  $EN \mapsto HR$  oder  $EN \mapsto HT$  also wahrscheinlich  $EIN \mapsto HZT$

– Folglich ist zu vermuten  $ER \mapsto HR$

## • Analyse liefert partiellen Klartext

.IE. I.. EIN .E.R ..N.ER RE...I. ....E..I. .IN.E...RIE.ENER .E..

.EN .IR ..ER EINE .E..I..RIER.N. .I..N.EREN .ER...REN .EN.E.I.EN

## Analysiere partiellen Klartext

### Vergleiche mit Chiffretext

AZHCIZCVIHZTICHGRIWXTOHRIRHWXVZUIDMPXHWZWOIGZTOHCYGRZHNHTHRIVHKVI  
AHTIEZRIPMHRIZTHIAHYGZPPRZHRMTOIBZVIMTCHRHTIUHRPXGRHTINHTSHVZOHT

DIES IST EIN SEHR LANGER RELATIV ZUFAELLIG HINGESCHRIEBENER TEXT  
DEN WIR FUER EINE DECHIFFRIERUNG MIT UNSEREN VERFAHREN BENOETIGEN

- Vermutlich beginnt der Satz mit DIES, also  $A \mapsto D$ ,  $C \mapsto S$
- Das zweite Wort muß IST, daß vierte SEHR sein, also  $V \mapsto T$ ,  $G \mapsto H$
- Nächste Vermutungen  $E \mapsto W$ ,  $O \mapsto G$ ,  $Y \mapsto C$ ,  $N \mapsto B$ ,  $S \mapsto O$ ,  $P \mapsto F$ ,  
 $M \mapsto U$ ,  $B \mapsto M$ ,  $U \mapsto V$ ,  $X \mapsto A$ ,  $K \mapsto X$ ,  $W \mapsto L$ ,  $D \mapsto Z$

# POLYALPHABETISCHE KRYPTOSYSTEME

- **Substitutionschiffren sind monoalphabetisch**
  - Jeder Buchstabe wird durch ein festes Symbol des Alphabets ersetzt
- **Polyalphabetische Systeme sind sicherer**
  - Starre Substitution der Buchstaben wird aufgeweicht
  - Buchstaben werden durch **verschiedene Symbole des Alphabets** ersetzt
  - Position innerhalb eines Textblockes bestimmt die Ersetzung
  - **Kryptoanalyse wird schwieriger**
- **Unterschiedlich aufwendige Varianten möglich**
  - **Vigenere Chiffre**: polyalphabetische Verschiebechiffre
    - Schlüsselwort verschiebt Elemente eines Textblocks unterschiedlich
    - Sehr einfache Ver-/Entschlüsselung, leicht zu merkender Schlüssel
  - **Polyalphabetische affine Chiffre** ebenfalls möglich
    - Spezialfall der affin-linearen Chiffren ↪ Einheit 2.2
  - Polyalphabetische Substitutionschiffren zu aufwendig (Nutzen gering)

# DIE VIGENERE CHIFFRE

- **Verschiebe Blöcke von  $m$  Buchstaben asynchron**

- Schlüssel verschiebt Elemente eines Buchstabenblocks unterschiedlich  
Aus **INFORMATIK BRAUCHT MATHEMATIK** mit Schlüssel **'A KEY'**  
wird **IMPSOM CMH AAERCGCDJASRIJASSOX** (Leerzeichen vervollständigt letzten Block)
- Originaltext durch Rückwärtsschieben der Blöcke ermittelbar

- **Programmierung ähnlich zur Verschiebechiffre**

- Schlüsselwort der Länge  $m$  entspricht Zahlentupel  $K = (k_1, \dots, k_m) \in \mathbb{Z}_n^m$
- (Umgewandelte) Texte werden aufgeteilt in Blöcke der Länge  $m$
- Ver-/Entschlüsselung durch simultane Addition/Subtraktion modulo  $n$ 
  - $e_K(x_1, \dots, x_m) = (x_1 + k_1, \dots, x_m + k_m) \bmod n,$
  - $d_K(y_1, \dots, y_m) = (y_1 - k_1, \dots, y_m - k_m) \bmod n$

- **Anwendungsbeispiel**

- Schlüssel "A KEY" entspricht Zahlentupel  $[0; 26; 10; 4; 24]$
- Aus **ENDE UM DREI UHR**  $\hat{=}$   $[4; 13; 3; 4; 26; 20; 12; 26; 3; 17; 4; 8; 26; 20; 7; 17; 26; 26; 26]$   
wird  $[4; 12; 13; 8; 23; 20; 11; 9; 7; 14; 4; 7; 9; 24; 4; 17; 25; 9; 3; 23]$   $\hat{=}$  **EMNIXULJHOEHJYERZJDX**

## Ähnlich zur Verschiebechiffre

- **Aufwand für Ver- und Entschlüsselung ist linear**
  - Umwandlung zwischen Buchstaben und Zahlen  $\mathcal{O}(|w|)$
  - Verschieben der Zahlenblöcke (Alphabet fest,  $n$  klein)  $\mathcal{O}(|w|)$
- **Aufwand für Brute Force Attacken ist höher**
  - Pro Versuch Aufwand  $\mathcal{O}(|w|)$  für die Entschlüsselung
  - Bei  $n$  Symbolen und Blockgröße  $m$  gibt es insgesamt  $n^m$  **Schlüssel**  
( $27^5 = 14.348.907$  im Beispiel)
  - Brute-Force Attacken ab Blockgröße 8 und Verwendung von  
62 alphanumerischen Symbolen undurchführbar  $62^8 = 2.2 * 10^{14}$
  - **Sicher gegen Brute-Force Attacken mit Computern**
- **Trotzdem nicht sicher**
  - Blockgröße kann bei lange Chiffretexten bestimmt werden
  - Wörterbuchattacken überprüfen einfache Schlüssel ( $\leq 500.000$  Tests)
  - Statistische Analysen ermöglichen Ermittlung von Schlüsselteilen

# KRYPTOANALYSE DES VIGENERE CHIFFRE

## ● Bestimme Blocklänge mit **Kasiski Test**

- Identische Klartextsegmente führen zum gleichen Schlüsseltextsegment, wenn Abstand ein Vielfaches der Blocklänge ist
- Suche identische Schlüsselsegmente und bestimme Abstände  
Segment sollte Mindestgröße drei haben und oft auftauchen
- Blocklänge muß Teiler der Abstände sein

## ● Anwendungsbeispiel

NRPOYBVXBBWSJZPHSSDXHXKCQWIJNUHVDLBQHYD WOQFASMVFVHXAJMB LYAZBVSODIXK  
AYS**FBS**A0JZJORJ0ZQTFNFVIHCESVDOOHAUBBS OTEPFWBTTETSHXUXTDSUJOCESMYBBAFMY  
BOCEHFQCZEKFEXXFHIQTESJPYWDEOZ WSSSDDTKSCPQOTHMYKXAXQ**FBS**POHBXVVODXKGN  
NZBGSASTEHS YVD AQRQTPJNFVFHXERZKOE0XKB DSJIVGLSWVXJXEFVK SXGJMRFW SXQ**FB**  
**S**RSVPTJNIVFHSJDHATFNQHLGKSJWLFLFMYOXLGOQ**FBS**ROHXATGRVJPLWBTETTFNUHVDLBQH  
YD WOQDTEOMYQOFBCMBBWWQVOKXWBVXVXFMYYSRVSPOBSNE WSADHXXENQVETA ODXSAS  
JUBILFMYBBS KTEHXAJHBVXAJXBQJODTEONBXQBXFSWQY KNODFVEOJSBZTAXJBBSFHIQT  
ENFVIR SBQAIJQRQAPKNDBQFTNBOPHX JXBZFOMAQOOHAUBOSN

- **FBS** an Position 73 194 278 320 (Abstände 121 205 247 84 126 42)
- **FMY** an Position 137 311 383 425 (Abstände 174 246 288 72 114 42)
- Erstes **FBS** vermutlich ein Ausreißer – andere Abstände durch 6 teilbar
- **Wahrscheinliche Blocklänge ist 6**

## ● Überprüfe wahrscheinlichste Schlüssel

- Bestimme häufigste Buchstaben an jeweils  $i$ -ter Position im Block
- Unter den ersten zwei oder drei ist wahrscheinlich die Codierung von E
- Teste alle Kombinationen der jeweils zugehörigen Teilschlüssel
  - Bei Blocklänge  $m$  sind  $2^m$  bzw.  $3^m$  Schlüssel zu prüfen
  - Falsche Schlüssel sind meist nach wenigen Blöcken identifiziert

## ● Analyse des Anwendungsbeispiels

- Häufigste zwei Buchstaben an jeweils  $i$ -ter Position eines Sechserblocks sind J/O, V/Q, B/X, O/T, S/X, S/N
- Zugehörige Teilschlüssel: F/K, R/M, Y/T, K/P, O/T, O/J
- Überprüfung der 64 möglichen Schlüssel auf die ersten 3 Sechserblöcke des Schlüsseltextes liefert als einzig sinnvollen Schlüssel: **KRYPTO**

## Aussagekräftige Informationen über lange Texte

- **Vigenerechiffre ist eine Verschiebechiffre**
  - Alle Buchstaben der erste, zweite,... Stelle jedes Blocks werden um dieselbe Konstante verschoben
  - Gleiche Buchstaben werden durch gleiche Buchstaben ersetzt
  - Statistische Eigenschaften der zugehörigen Klar- und Schlüsseltextfragmente sind identisch
- **Welche statistische Eigenschaft ist gut meßbar?**
  - Häufigkeit konkreter Buchstaben im Klartext kann schwanken
    - Auch in langen Texten kann ein E untypisch selten vorkommen
  - Aber insgesamt dominieren E, ' ', N, I, S, R, A, T, D, H, U jeden Text
  - Entsprechend müssen manche Symbole den Schlüsseltext dominieren
  - Der **Koinzidenzindex** ist ein Maß für diese Ungleichverteilung

## Wie wahrscheinlich ist es, daß zwei beliebige Symbole in einem Text gleich sind?

- **Sollwert für typische Texte ist**  $\sum_{i=0}^{26} p_i^2$ 
  - Summe aller Wahrscheinlichkeiten, daß erstes und zweites Symbol eines beliebigen Textes jeweils der  $i$ -te Buchstabe des Alphabets sind
    - $p_i$  Wahrscheinlichkeit, daß ein Symbol eines Textes der  $i$ -te Buchstabe ist
  - Bei Gleichverteilung:  $\sum_{i=0}^{26} p_i^2 = 1/27$  (0.0370) im Deutschen: 0.0762
- **Koinzidenzindex eines konkreten Wortes  $w$** 
  - Summe aller Wahrscheinlichkeiten, daß beide Symbole innerhalb von  $w$  jeweils der  $i$ -te Buchstabe des Alphabets sind
  - Hängt ab von tatsächlicher Häufigkeit der Symbole im Text
    - $I_c(w) = \sum_{i=0}^{26} f_i^w (f_i^w - 1) / (|w|(|w| - 1))$  ( $f_i^w$  Häufigkeit des  $i$ -ten Symbols in  $w$ )
  - Liegt bei längeren Texten nahe am Sollwert

# BESTIMME BLOCKLÄNGE MIT KOINZIDENZANALYSE

## ● Bestimme Koinzidenzindizes möglicher Blöcke

- Bei richtiger Blockaufteilung sind Koinzidenzindizes der jeweils ersten, zweiten, ... Elemente eines Blocks identisch mit denen des Klartextes
- Bei falscher Blockaufteilung ist Verteilung der Symbole im zugehörigen Schlüsseltextfragment und der Koinzidenzindex eher untypisch
- Korrekte Blocklänge führt zu Koinzidenzindizes nahe am Sollwert
- Überprüfe entsprechende Koinzidenzindizes für Blocklängen  $m=1, 2, 3, \dots$

## ● Anwendungsbeispiel

NRPOYBVXBBWSJZPHSSSDXHXKCQWIJNUHVDLBQHYD WOQFASMVFVHXAJMB LYAZBVSODIXK  
AYSFBSAOJZJORJOZQTFNFVIHCESVDOOHAUBBS OTEPFWB TETSHXUXTDSUJOCESMYBBAFMY  
BOCEHFQCZEKFEXXFHIQTESJPYWDEOZ WSSSDDTKSCPQOTHMYKXAXQFBSPOHXBVVODXKGN  
NZBGSASTEHS YVD AQRQTPJNFVFHXERZKOEOKB DSJIVGLSWVXJXEFVK SXGJMRFW SXQFB  
SRSVPTJNIVFHSJDHATFNQHLGKSJWLFLFMYOXLGOQFBSROHXATGRVJPLWBTETFNUHVDLBQH  
YD WOQDTEOMYQOFBCMBBWVQVOKXWBVXVXFMYWRSVPOBSNE WSADHXXENQVETA ODXSAS  
JUBILFMYBBS KTEHXA JHBVXA JXBQJODTEONBXQB XFSWQY KNODFVEOJSBZTAXJBBSFHIQT  
ENFVIR SBQAIJQRQAPKNDBQFTNBOPHX JXBZFOMA QOOHAUBOSN

- Indizes für Blocklängen 1 – 5 weichen stark vom Sollwert ab  
.0153; .0543,.0456; .0512,.0490,.0585; .0598,.0438,.0466,.0443; .0517,.0439,.0420,.0460,.0434
- Blocklänge **6** hat geringe Abweichung: .0619, .0736, .0861, .0606, .0798, .0761

# KOINZIDENZANALYSE AM BEISPIEL

## ● Bestimme Schlüssel mit Koinzidenzanalyse

- Berechne  $\sum_{i=0}^{26} p_i \cdot f_{i+k}^v / |v|$  für alle  $k \in \mathbb{Z}_{27}$  und alle Teilworte  $v$ , die sich aus den jeweils ersten, zweiten, ... Elemente des  $m$ -Blocks ergeben
- Bei korrektem Teilschlüssel  $k$  sollte dies nahe am Sollwert liegen, da  $f_{i+k}^v$  die Häufigkeit der tatsächlichen Codierung des  $i$ -ten Symbols ist

## ● Schlüsselbestimmung mit Koinzidenzanalyse

- Beste Werte für mögliche Teilschlüssel liefern  
**K** .0661, **R** .0720, **Y** .0771, **P** .0678, **T** .0752, **O** .0762
- Entschlüsselung mit Schlüssel **KRYPTO** liefert folgenden Klartext

DAS FOLGENDE IST EIN TEXT ZUR KRYPTOGRAPHIE IM ZWEITEN WELTKRIEG AUS WIKIPEDIA IM ZWEITEN WELTKRIEG WURDEN MECHANISCHE UND ELEKTROMECHANISCHE KRYPTOGRAPHIESYSTEME ZAHLREICH EINGESETZT AUCH WENN IN BEREICHEN WO DIES NICHT MOEGLICH WAR WEITERHIN MANUELLE SYSTEME VERWENDET WURDEN IN DIESER ZEIT WURDEN GROSSE FORTSCHRITTE IN DER MATHEMATISCHEN KRYPTOGRAPHIE GEMACHT NOTWENDIGERWEISE GESCHAH DIES JEDOCH NUR IM GEHEIMEN DIE DEUTSCHEN MACHTEN REGEN GEBRAUCH VON EINEM ALS ENIGMA BEKANNTEN SYSTEM WELCHES DURCH DAS ULTRA SYSTEM GEKNACKT WURDE