Crypotography

A Quick Introduction

Mohamed Barakat

TU-Kaiserslautern

Kaiserslautern, October 2010

Kryptologie

Die moderne Kryptologie besteht aus zwei Teilgebieten:

1. Die Kryptographie[1] (Entwicklung von Kryptosystemen): Verfahren zur Verschlüsselung (Chiffrieren[2]) von Informationen und Nachrichten.
2. Die Kryptoanalyse (Analyse von Kryptosystemen): Sicherheitsanalyse und Angriffsstrategien.

[1] Griechisch: $\kappa\rho\upsilon\pi\tau o\varsigma$, kryptós = „verborgen“, und $\gamma\rho\alpha\phi\epsilon\iota\nu$, gráphein = „schreiben“

[2] Arabisch: sifr = Null

Mögliche Ziele der Verschlüsselung

- Schutz gegen Abhören (Geheimcode gegen Lauscher)

Der klassische Einsatzbereich von Verschlüsselungsverfahren.

War der Schutz gegen unerlaubte Mithörer lange Zeit nur wichtig für das Militär und die Diplomatie, so ist es heute ein Problem für jedermann, z.B.

- Telefonieren mit dem Handy,
- Datenübertragung im Internet,
- Pay-TV
- ...

Mögliche Ziele der Verschlüsselung

- Schutz gegen Abhören (Geheimcode gegen Lauscher)
- Schutz gegen Veränderung (Authentifizierung)

Wer möchte schon, dass beim Homebanking die Kontonummer des Empfängers und/oder der überwiesene Betrag von einem Bösewicht verändert werden.

Mögliche Ziele der Verschlüsselung

- Schutz gegen Abhören (Geheimcode gegen Lauscher)
- Schutz gegen Veränderung (Authentifizierung)
- Beweis der Urheberschaft (elektronische Unterschrift)

Oftmals will man absolut sicher sein, dass eine Internetseite wirklich vom beabsichtigten Dienstleister stammt (z.B., wenn jemand nach Daten der Kreditkarte fragt).

„Definitionen“

Sei $\mathcal{B}$ die Menge aller **„Buchstaben“** (**Bits**, **Bytes**, ...)

„Definitionen“

Sei $\mathcal{B}$ die Menge aller **„Buchstaben“** (**Bits**, **Bytes**, ...) und

$$\mathcal{N} \subset \mathcal{B}^{\mathbb{N}}$$

die Menge aller endlichen Sequenzen (=Tupeln) von Buchstaben.

„Definitionen“

Sei $\mathcal{B}$ die Menge aller **„Buchstaben“** (**Bits**, **Bytes**, ...) und

$$\mathcal{N} \subset \mathcal{B}^{\mathbb{N}}$$

die Menge aller endlichen Sequenzen (=Tupeln) von Buchstaben. Wir nennen $\mathcal{N}$ die Menge der **„Nachrichten“**.

„Definitionen“

Sei $\mathcal{B}$ die Menge aller **„Buchstaben“** (**Bits**, **Bytes**, ...) und

$$\mathcal{N} \subset \mathcal{B}^{\mathbb{N}}$$

die Menge aller endlichen Sequenzen (=Tupeln) von Buchstaben. Wir nennen $\mathcal{N}$ die Menge der **„Nachrichten“**.

Ein **Verschlüsselungsalgorithmus** ist eine bijektive Abbildung

$$\chi : \mathcal{N} \to \mathcal{N}$$

(Permutation).

„Definitionen“

Sei $\mathcal{B}$ die Menge aller **„Buchstaben“** (**Bits**, **Bytes**, ...) und

$$\mathcal{N} \subset \mathcal{B}^{\mathbb{N}}$$

die Menge aller endlichen Sequenzen (=Tupeln) von Buchstaben. Wir nennen $\mathcal{N}$ die Menge der **„Nachrichten“**.

Ein **Verschlüsselungsalgorithmus** ist eine bijektive Abbildung

$$\chi : \mathcal{N} \to \mathcal{N}$$

(Permutation). Ihre inverse Abbildung $\chi^{-1} : \mathcal{N} \to \mathcal{N}$ nennt man **Entschlüsselungsalgorithmus**.

SKC

Symmetrisches Kryptosystem = Secret Key Cryptosystem (SKC)

Ein **symmetrisches Kryptosystem** ist eine Familie von Verschlüsselungsalgorithmen $(\chi_k)_{k\in\mathcal{K}}$, so daß für jedes $k \in \mathcal{K}$ ein eindeutiges $k^{-1} \in \mathcal{K}$ existiert mit ${\chi_k}^{-1} = \chi_{k^{-1}}$.

SKC

Symmetrisches Kryptosystem = Secret Key Cryptosystem (SKC)

Ein **symmetrisches Kryptosystem** ist eine Familie von Verschlüsselungsalgorithmen $(\chi_k)_{k\in\mathcal{K}}$, so daß für jedes $k \in \mathcal{K}$ ein eindeutiges $k^{-1} \in \mathcal{K}$ existiert mit ${\chi_k}^{-1} = \chi_{k^{-1}}$. Die Menge $\mathcal{K}$ nennt man den **Schlüsselraum**.

SKC

Symmetrisches Kryptosystem = Secret Key Cryptosystem (SKC)

Ein **symmetrisches Kryptosystem** ist eine Familie von Verschlüsselungsalgorithmen $(\chi_k)_{k\in\mathcal{K}}$, so daß für jedes $k \in \mathcal{K}$ ein eindeutiges $k^{-1} \in \mathcal{K}$ existiert mit ${\chi_k}^{-1} = \chi_{k^{-1}}$. Die Menge $\mathcal{K}$ nennt man den **Schlüsselraum**. Ihre Elemente nennt man **Schlüssel**.

SKC

Symmetrisches Kryptosystem = Secret Key Cryptosystem (SKC)

Ein **symmetrisches Kryptosystem** ist eine Familie von Verschlüsselungsalgorithmen $(\chi_k)_{k\in\mathcal{K}}$, so daß für jedes $k \in \mathcal{K}$ ein eindeutiges $k^{-1} \in \mathcal{K}$ existiert mit ${\chi_k}^{-1} = \chi_{k^{-1}}$. Die Menge $\mathcal{K}$ nennt man den **Schlüsselraum**. Ihre Elemente nennt man **Schlüssel**. Weiterhin verlangt man, daß k^{-1} aus k „leicht“ berechnet werden kann.

SKC

Symmetrisches Kryptosystem = Secret Key Cryptosystem (SKC)

Ein **symmetrisches Kryptosystem** ist eine Familie von Verschlüsselungsalgorithmen $(\chi_k)_{k\in\mathcal{K}}$, so daß für jedes $k \in \mathcal{K}$ ein eindeutiges $k^{-1} \in \mathcal{K}$ existiert mit ${\chi_k}^{-1} = \chi_{k^{-1}}$. Die Menge $\mathcal{K}$ nennt man den **Schlüsselraum**. Ihre Elemente nennt man **Schlüssel**. Weiterhin verlangt man, daß k^{-1} aus k „leicht“ berechnet werden kann. Ist $k^{-1} = k$ für alle $k \in \mathcal{K}$, so nennt man das Kryptosystem **involutorisch**.

Strom-Chiffren

Strom-Chiffre

Eine **Strom-Chiffre** ist ein symmetrisches Kryptosystem $(\chi_k)_{k \in \mathcal{K}}$ mit folgenden Eigenschaften:

Strom-Chiffren

Strom-Chiffre

Eine **Strom-Chiffre** ist ein symmetrisches Kryptosystem $(\chi_k)_{k\in\mathcal{K}}$ mit folgenden Eigenschaften:

1. Jedes $k \in \mathcal{K}$ ist ein Tupel von Permutationen $(\sigma_i)_{i=1}^{l}$, wobei $\sigma_i \in S_{\mathcal{B}}$ für alle $i = 1, \ldots, l$.

Strom-Chiffren

Strom-Chiffre

Eine **Strom-Chiffre** ist ein symmetrisches Kryptosystem $(\chi_k)_{k\in\mathcal{K}}$ mit folgenden Eigenschaften:

1. Jedes $k \in \mathcal{K}$ ist ein Tupel von Permutationen $(\sigma_i)_{i=1}^{l}$, wobei $\sigma_i \in S_{\mathcal{B}}$ für alle $i = 1, \ldots, l$.
2. Für ein $n = (n_j)_{j=1}^{h} \in \mathcal{N} \subset \mathcal{B}^{\mathbb{N}}$ der Länge h gilt $(\chi_k(n))_j = \sigma_{(j \mod l)}(n_j)$, für alle $j = 1, \ldots, h$.

Strom-Chiffren

Strom-Chiffre

Eine **Strom-Chiffre** ist ein symmetrisches Kryptosystem $(\chi_k)_{k\in\mathcal{K}}$ mit folgenden Eigenschaften:

1. Jedes $k \in \mathcal{K}$ ist ein Tupel von Permutationen $(\sigma_i)_{i=1}^{l}$, wobei $\sigma_i \in S_{\mathcal{B}}$ für alle $i = 1, \ldots, l$.
2. Für ein $n = (n_j)_{j=1}^{h} \in \mathcal{N} \subset \mathcal{B}^{\mathbb{N}}$ der Länge h gilt $(\chi_k(n))_j = \sigma_{(j \mod l)}(n_j)$, für alle $j = 1, \ldots, h$.

Die natürliche Zahl l nennt man die Länge des Schlüssels $k = (\sigma_i)_{i=1}^{l}$.

Strom-Chiffren

Strom-Chiffre

Eine **Strom-Chiffre** ist ein symmetrisches Kryptosystem $(\chi_k)_{k\in\mathcal{K}}$ mit folgenden Eigenschaften:

1. Jedes $k \in \mathcal{K}$ ist ein Tupel von Permutationen $(\sigma_i)_{i=1}^{l}$, wobei $\sigma_i \in S_{\mathcal{B}}$ für alle $i = 1, \ldots, l$.
2. Für ein $n = (n_j)_{j=1}^{h} \in \mathcal{N} \subset \mathcal{B}^{\mathbb{N}}$ der Länge h gilt $(\chi_k(n))_j = \sigma_{(j \mod l)}(n_j)$, für alle $j = 1, \ldots, h$.

Die natürliche Zahl l nennt man die Länge des Schlüssels $k = (\sigma_i)_{i=1}^{l}$.

Beispiel

Cäsar-Chiffre: $\mathcal{B} := \{A, B, C, \ldots, Z\}$, $l := 1$ und $\sigma_1 :=$„C“.

Caesar-Verschlüsselung

- Schlüssel ist Zahl s zwischen 1 und 25
- Verschlüsselung erfolgt durch Verschiebung des Alphabets um s Stellen, d.h. falls $s = 3$:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
DEFGHIJKLMNOPQRSTUVWXYZABC
```

Aus

```
VENI VIDI VICI
```

wird somit

```
YHQL YLGL YLFL
```

Caesar-Verschlüsselung

Mit Hilfe zweier gegeneinander drehbarer Scheiben kann man leicht eine Ver- und Entschlüsselungsmaschine bauen:

Caesar-Verschlüsselung

- Die Caesar-Verschiebung ist eine monoalphabetische Verschlüsselung: Jedem Klarbuchstaben entspricht genau ein Geheimbuchstabe.
- **Knacken:** Teste die 25 möglichen Verschiebungen.

Knacken der Caesar-Verschlüsselung

```
INJBZJWKJQXNSILJKFQQJS
JOKCAKXLKRYOTJMKLGRRKT
KPLDBLYMLSZPUKNLMHSSLU
LQMECMZNMTAQVLOMNITTMV
MRNFDNAONUBRWMPNOJUUNW
NSOGEOBPOVCSXNQOPKVVOX
OTPHFPCQPWDTYORPQLWWPY
PUQIGQDRQXEUZPSQRMXXQZ
QVRJHRESRYFVAQTRSNYYRA
RWSKISFTSZGWBRUSTOZZSB
SXTLJTGUTAHXCSVTUPAATC
TYUMKUHVUBIYDTWUVQBBUD
UZVNLVIWVCJZEUXVWRCCVE
```

```
VAWOMWJXWDKAFVYWXSDDWF
WBXPNXKYXELBGWZXYTEEXG
XCYQOYLZYFMCHXAYZUFFYH
YDZRPZMAZGNDIYBZAVGGZI
ZEASQANBAHOEJZCABWHHAJ
AFBTRBOCBIPFKADBCXIIBK
BGCUSCPDCJQGLBECDYJJCL
CHDVTDQEDKRHMCFDEZKKDM
DIEWUERFELSINDGEFALLEN
EJFXVFSGFMTJOEHFGBMMFO
FKGYWGTHGNUKPFIGHCNNGP
GLHZXHUIHOVLQGJHIDOOHQ
HMIAYIVJIPWMRHKIJEPPIR
```

Alternative zur Caesar-Verschlüsselung

- Statt nur zu verschieben, können wir die Buchstaben des Geheimalphabets auch komplizierter anordnen. Dann

  Schlüssel = "verwürfeltes" Alphabet.

  Insgesamt gibt es

  $$26 \cdot 25 \cdot \ldots \cdot 2 \cdot 1 = 403\,291\,461\,126\,605\,635\,584\,000\,000 \approx 2^{88}$$

  verwürfelte Alphabete. Also kann man nicht einfach alle ausprobieren, um den Code zu knacken. Aber bereits um 850 n. Chr. haben arabische Gelehrte gezeigt, wie man solche Botschaften entziffern kann....

Knacken Monoalphabet. Verschlüsselung

Man nutzt die Häufigkeitsverteilung der Buchstaben (z.B.: am häufigsten auftretender Buchstabe $\rightarrow$ E, ...)

Häufigkeitsverteilung der Einzelbuchstaben in deutscher Sprache									
A	6.51	F	1.66	K	1.21	P	0.79	U	4.35	Z	1.13
B	1.89	G	3.01	L	3.44	Q	0.02	V	0.67		
C	3.06	H	4.76	M	2.53	R	7.00	W	1.89		
D	5.08	I	7.55	N	9.78	S	7.27	X	0.03		
E	17.40	J	0.27	O	2.51	T	6.15	Y	0.04		

Häufigkeitsverteilung von Buchstabenpaaren in deutscher Sprache										
EN	ER	CH	TE	DE	ND	EI	IE	IN	ES	EA,ET
3.88	3.75	2.75	2.26	2.0	1.99	1.88	1.79	1.67	1.52	≤ 0.5

$\Longrightarrow$ **Ziel:** Verschleierung der Häufigkeiten !

Vigenère-Verschlüsselung

- Benannt nach Blaise de Vigenère (am Hofe Heinrichs III von Frankreich).
- benutzt verschiedene monoalphabetische Verschlüsselungen im Wechsel ("polyalphabetisches Verfahren").
- **Bestimmung des jeweils aktuellen Alphabets** erfolgt mit Hilfe eines Schlüsselwortes aus dem sogenannten Vigenère-Quadrat.

Vigenère-Verschlüsselung

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

Vigenère-Verschlüsselung

Mit dem Schlüssel LUTETIA wird Caesars

VENI VIDI VICI

zu:

	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
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
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
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
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
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
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
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
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
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
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
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
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

GYGM OQDT PBGB

Vigenère-Verschlüsselung

- Die Vigenère-Verschlüsselung ist mit einer naiven Häufigkeitsanalyse nicht zu knacken. Sie galt lange Zeit als absolut sicher.
- **Wesentliche Schwäche:** zyklischer Charakter.

  Als Erster hat dies Charles Babbage im 19-ten Jahrhundert erkannt: man kann Länge ℓ des Schlüsselworts ermitteln. Anschließend kommt man mit ℓ Häufigkeitsanalysen zum Ziel.

Knacken der Vigenère-Verschlüsselung

Angenommen, wir haben die folgende Vigenère-verschlüsselte Botschaft abgehört:

```
WUBEFIQLZURMVOFEHMYMWTIXCGTMPIFKRZUPMVOIRQMMWOZMPULMBNYVQQQMVMVJLE
YMHFEFNZPSDLPPSDLPEVQMWCXYMDAVQEEFIQCAYTQOWCXYMWMSEMEFCFWYEYQETRLI
QYCGMTWCWFBSMYFPLRXTQYEEXMRULUKSGWFPTLRQAERLEEXMRULUKSGWFPTLRQAERL
UVPMVYQYCXTWFQLMTELSFJPQEHMOZCIWCIWFPZSLMAEZIQVLQMZVPPXAWCSMZMORVG
VVQSZETRLQZPBJAZVQIYXEWWOICCGDWHQMMVOWSGNTJPFPPAYBIYBJUTWRLQKLLLMD
PYVACDCFQNZPIFPPKSDVPTIDGXMQQVEBMQALKEZMGCVKUZKIZBZLIUAMMVZ
```

Knacken der Vigenère-Verschlüsselung

Angenommen, wir haben die folgende Vigenère-verschlüsselte Botschaft abgehört:

```
WUBEFIQLZURMVOFEHMYMWTIXCGTMPIFKRZUPMVOIRQMMWOZMPULMBNYVQQQMVMVJLE
YMHFEFNZPSDLPPSDLPEVQMWCXYMDAVQEEFIQCAYTQOWCXYMWMSEMEFCFWYEYQETRLI
QYCGMTWCWFBSMYFPLRXTQYEEXMRULUKSGWFPTLRQAERLEEXMRULUKSGWFPTLRQAERL
UVPMVYQYCXTWFQLMTELSFJPQEHMOZCIWCIWFPZSLMAEZIQVLQMZVPPXAWCSMZMORVG
VVQSZETRLQZPBJAZVQIYXEWWOICCGDWHQMMVOWSGNTJPFPPAYBIYBJUTWRLQKLLLMD
PYVACDCFQNZPIFPPKSDVPTIDGXMQQVEBMQALKEZMGCVKUZKIZBZLIUAMMVZ
```

Wiederholt auftretende Zeichenfolgen:

EFIQ PSDLP WCXYM ETRL

Knacken der Vigenère-Verschlüsselung

Bestimmung der wahrscheinlichen Schlüssellänge:

Zeichen-Folge	Zwischen-Raum	Teiler = mögliche Schlüssellänge																	
		2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
EFIQ	95				x														x	
PSDLP	5				x															
WCXYM	20	x		x	x					x										x
ETRL	120	x	x	x	x	x		x		x		x			x					x

Ergebnis: 5

Block-Chiffren

Block-Chiffre

Bei einer Block-Chiffre wird auf Blöcken von Buchstaben, statt auf einzelnen Buchstaben operiert.

Block-Chiffren

Block-Chiffre

Bei einer Block-Chiffre wird auf Blöcken von Buchstaben, statt auf einzelnen Buchstaben operiert.

Beispiele

- Die spartanische Skytale.

Block-Chiffren

Block-Chiffre

Bei einer Block-Chiffre wird auf Blöcken von Buchstaben, statt auf einzelnen Buchstaben operiert.

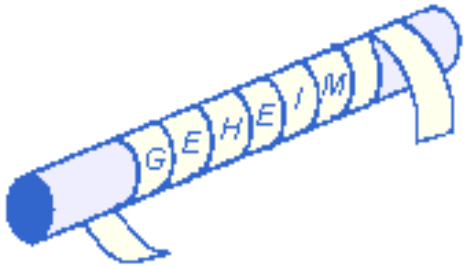
Beispiele

- Die spartanische Skytale.
- DES, 3DES, AES.

Die Skytala

$\longrightarrow$ verwendet von Spartanern (ca. 500 v. Chr.)

Verschlüsseln: Wähle einen Stab, wickele einen Papierstreifen mehrfach darum herum und schreibe dann den Text auf den Streifen, so dass jeder Buchstabe auf einer neuen Papierbahn liegt:



Entschlüsseln: Aufrollen auf Stab derselben Dicke.

Al-Kindi

Abū Yūsuf **Al-Kindī**
(Alkindus)
Universalgelehrter
801 – 873, Bagdad

hat, wie wir seit 1992 wissen, im 9. Jahrhundert sein Buch „Risalah fi Istikhraj al-Mu'amma“ (Manuskript über die Dechiffrierung kryptographischer Nachrichten) über die Kryptoanalyse von polyalphabetischen Ersetzungschiffren verfasst.

Al-Kindi

Abū Yūsuf **Al-Kindī**
(Alkindus)
Universalgelehrter
801 – 873, Bagdad

hat, wie wir seit 1992 wissen, im 9. Jahrhundert sein Buch „Risalah fi Istikhraj al-Mu'amma“ (Manuskript über die Dechiffrierung kryptographischer Nachrichten) über die Kryptoanalyse von polyalphabetischen Ersetzungschiffren verfasst. Er gilt heute als Erfinder der Häufigkeitsanalyse.

Al-Kindi

Abū Yūsuf **Al-Kindī**
(Alkindus)
Universalgelehrter
801 – 873, Bagdad

hat, wie wir seit 1992 wissen, im 9. Jahrhundert sein Buch „Risalah fi Istikhraj al-Mu'amma“ (Manuskript über die Dechiffrierung kryptographischer Nachrichten) über die Kryptoanalyse von polyalphabetischen Ersetzungschiffren verfasst. Er gilt heute als Erfinder der Häufigkeitsanalyse.
Der englische Mathematiker und Ingenieur Charles **Babbage** hat im Jahre 1846 eine Methode entwickelt, um polyalphabetische Ersetzungschiffren mit **kurzen** Schlüsseln ohne Kenntnis des Schlüssels zu dechiffrieren.

Al-Kindi

Abū Yūsuf **Al-Kindī**
(Alkindus)
Universalgelehrter
801 – 873, Bagdad

hat, wie wir seit 1992 wissen, im 9. Jahrhundert sein Buch „Risalah fi Istikhraj al-Mu'amma“ (Manuskript über die Dechiffrierung kryptographischer Nachrichten) über die Kryptoanalyse von polyalphabetischen Ersetzungschiffren verfasst. Er gilt heute als Erfinder der Häufigkeitsanalyse.
Der englische Mathematiker und Ingenieur Charles **Babbage** hat im Jahre 1846 eine Methode entwickelt, um polyalphabetische Ersetzungschiffren mit **kurzen** Schlüsseln ohne Kenntnis des Schlüssels zu dechiffrieren. Seine unveröffentlichte Methode wurde vom preußischen Offizier Friedrich **Kasiski** 1863 wiederentdeckt, und trägt seitdem den Namen „Kasiski-Test“.

Die Moderne Kryptographie

Auguste **Kerckhoffs**

1835 – 1903

ein niederländischer Linguist und Kryptograph, formulierte in einer Arbeit von 1883 ein Grundprinzip der modernen Kryptographie:

Die Moderne Kryptographie

Auguste **Kerckhoffs**

1835 – 1903

ein niederländischer Linguist und Kryptograph, formulierte in einer Arbeit von 1883 ein Grundprinzip der modernen Kryptographie:

Kerckhoffs' Prinzip

Die Sicherheit eines Verschlüsselungsverfahrens darf *nicht* auf der Geheimhaltung des Verschlüsselungsalgorithmus basieren, sondern auf der des Schlüssels.

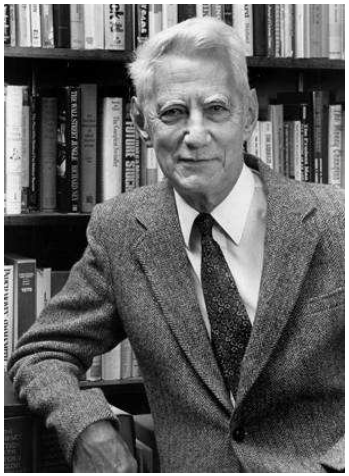
One-Time-Pad (Einmal-Block, Einmalverschlüsselung)

Ist die Schlüssellänge einer Strom-Chiffre mindestens genauso lang wie die der zu verschlüsselnden Nachricht, so redet man von einer Vernam-Chiffre.

One-Time-Pad (Einmal-Block, Einmalverschlüsselung)

Ist die Schlüssellänge einer Strom-Chiffre mindestens genauso lang wie die der zu verschlüsselnden Nachricht, so redet man von einer Vernam-Chiffre. Wird dieser Schlüssel nur einmal benutzt, so redet man von einem One-Time-Pad.

One-Time-Pad (Einmal-Block, Einmalverschlüsselung)

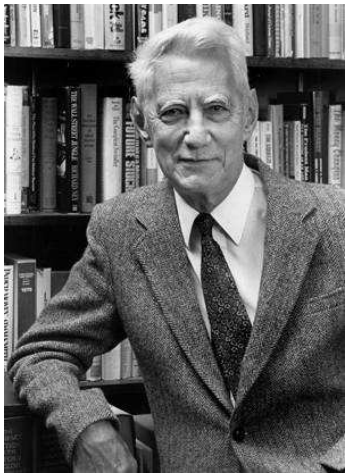


Claude **Shannon**

1916 – 2001, USA

Ist die Schlüssellänge einer Strom-Chiffre mindestens genauso lang wie die der zu verschlüsselnden Nachricht, so redet man von einer Vernam-Chiffre. Wird dieser Schlüssel nur einmal benutzt, so redet man von einem One-Time-Pad. Claude **Shannon**, der Vater der Informationstheorie, entdeckte 1945 die theoretische Bedeutung eines One-Time-Pad.

One-Time-Pad (Einmal-Block, Einmalverschlüsselung)

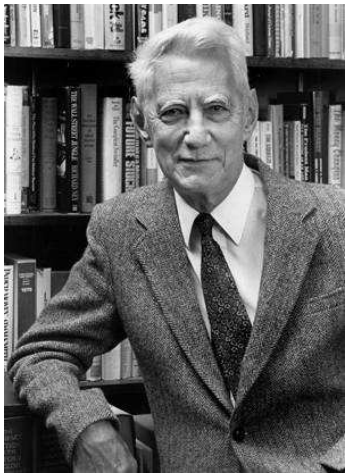


Claude **Shannon**

1916 – 2001, USA

Ist die Schlüssellänge einer Strom-Chiffre mindestens genauso lang wie die der zu verschlüsselnden Nachricht, so redet man von einer Vernam-Chiffre. Wird dieser Schlüssel nur einmal benutzt, so redet man von einem One-Time-Pad. Claude **Shannon**, der Vater der Informationstheorie, entdeckte 1945 die theoretische Bedeutung eines One-Time-Pad. Erst 1948 veröffentlichte er seinen Beweis:

One-Time-Pad (Einmal-Block, Einmalverschlüsselung)



Claude **Shannon**

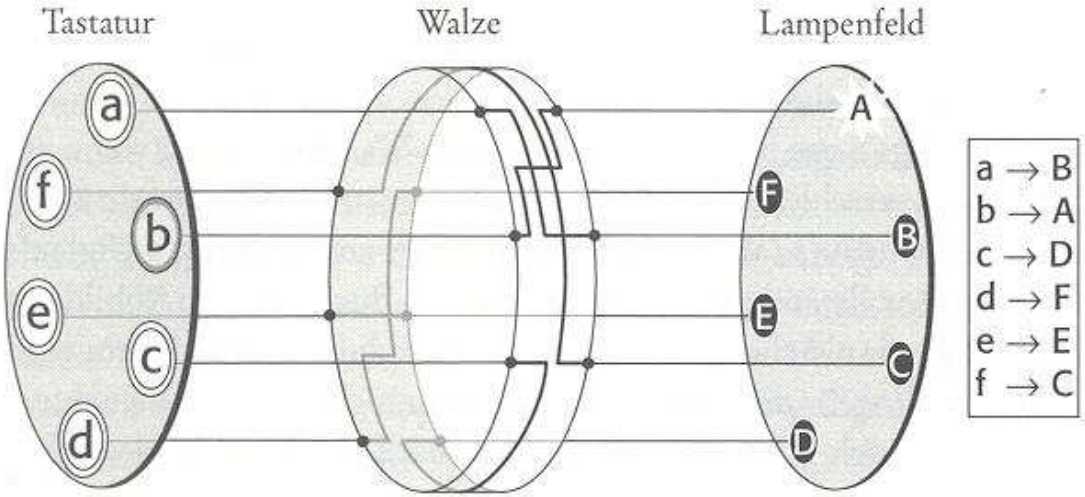
1916 – 2001, USA

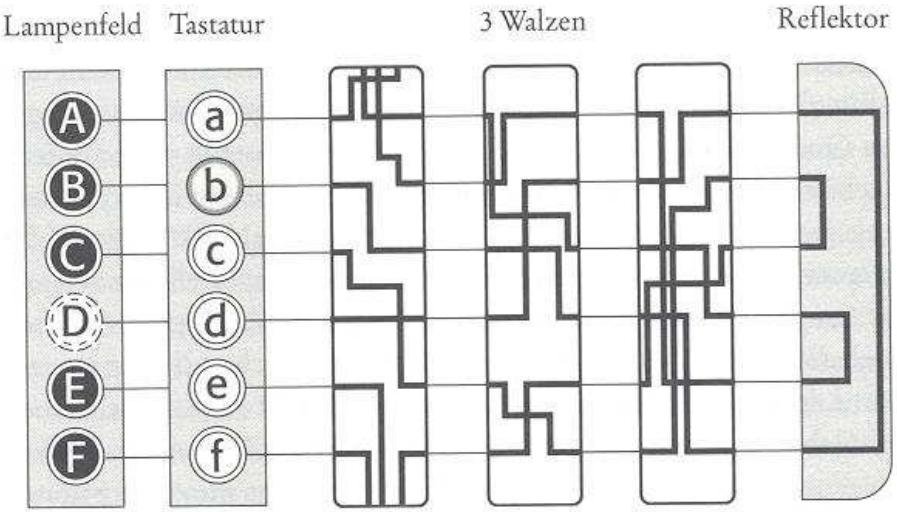
Ist die Schlüssellänge einer Strom-Chiffre mindestens genauso lang wie die der zu verschlüsselnden Nachricht, so redet man von einer Vernam-Chiffre. Wird dieser Schlüssel nur einmal benutzt, so redet man von einem One-Time-Pad. Claude **Shannon**, der Vater der Informationstheorie, entdeckte 1945 die theoretische Bedeutung eines One-Time-Pad. Erst 1948 veröffentlichte er seinen Beweis:

Perfekte Sicherheit

Das One-Time-Pad bietet eine perfekte kryptographische Sicherheit.

Die Enigma Maschine





Die Enigma Maschine

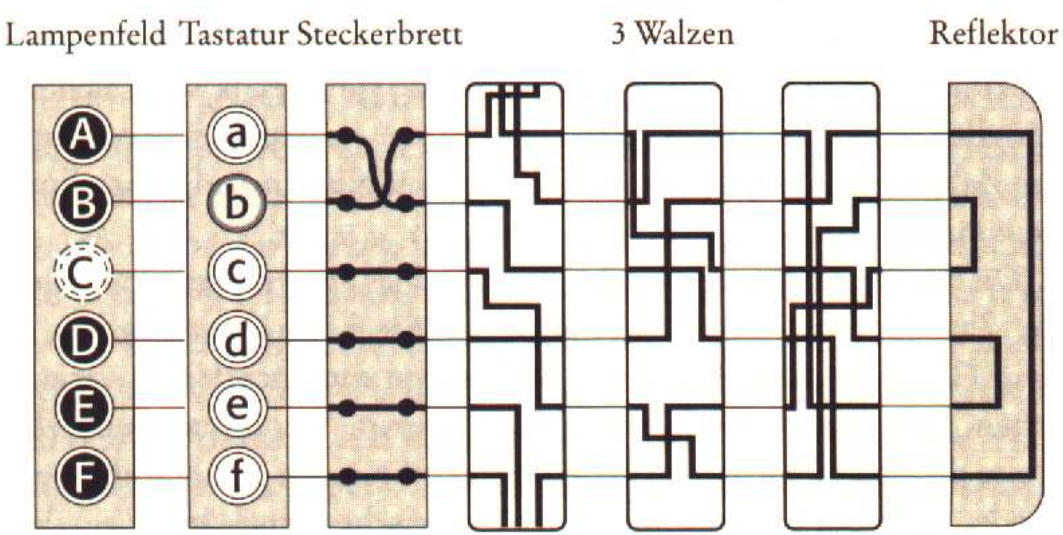
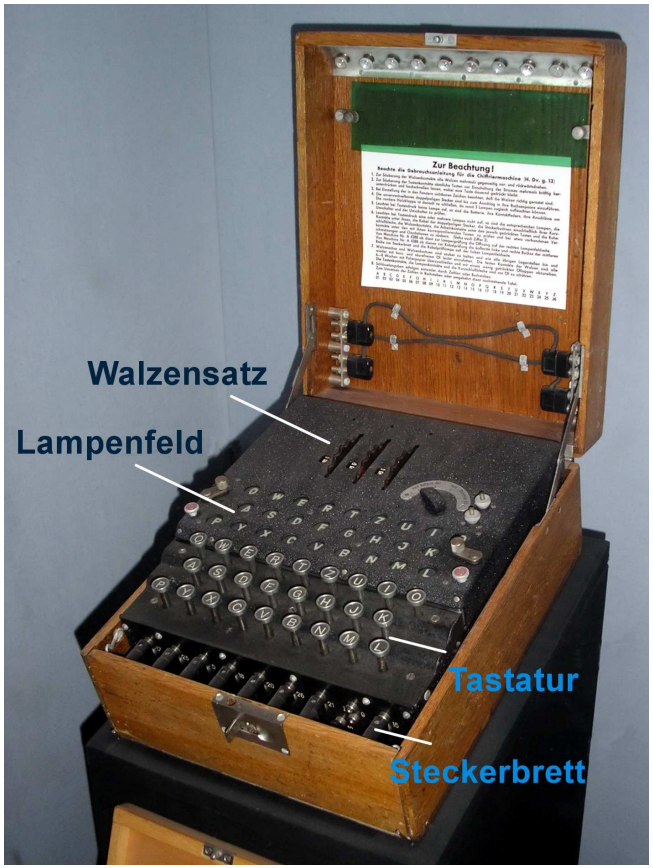


Abbildung 37: Das Steckerbrett sitzt zwischen Tastatur und erster Walze. Durch Kabelverbindungen ist es möglich, zwei Buchstaben miteinander zu vertauschen, in diesem Falle b und a. Jetzt wird b verschlüsselt, indem es dem Weg folgt, der ursprünglich für a vorgesehen war. In der echten 26-Buchstaben-Enigma hatte der Anwender sechs Kabel zur Verfügung, mit denen er sechs Buchstabenpaare vertauschen konnte.

Die Enigma



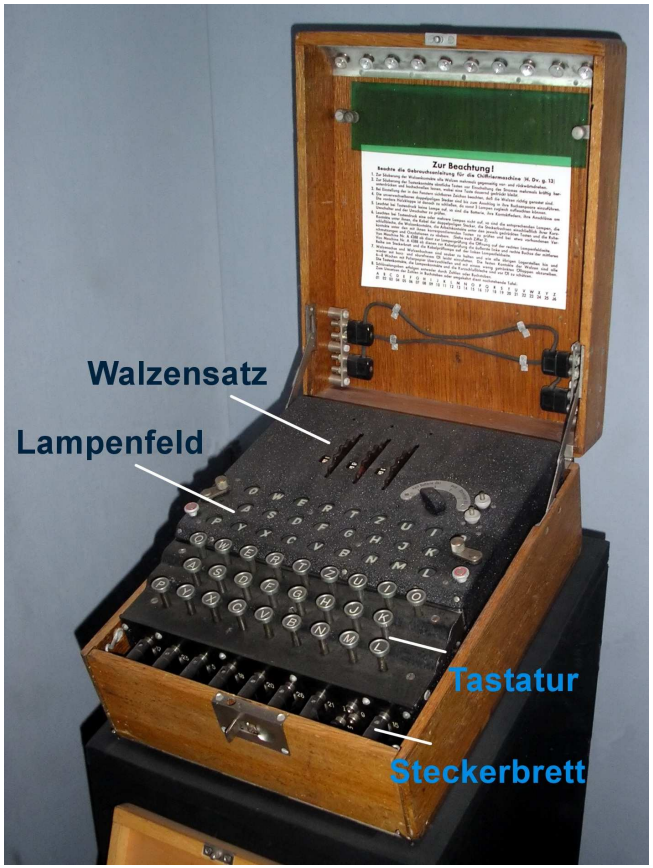
Die Enigma

Erfinder: Arthur **Scherbius**

Eckdaten:

- ca. $2 \cdot 10^{23} \approx 2^{7+9+14+47} = 2^{77} \approx |\mathcal{K}|$ Grundeinstellungen!

Die Enigma



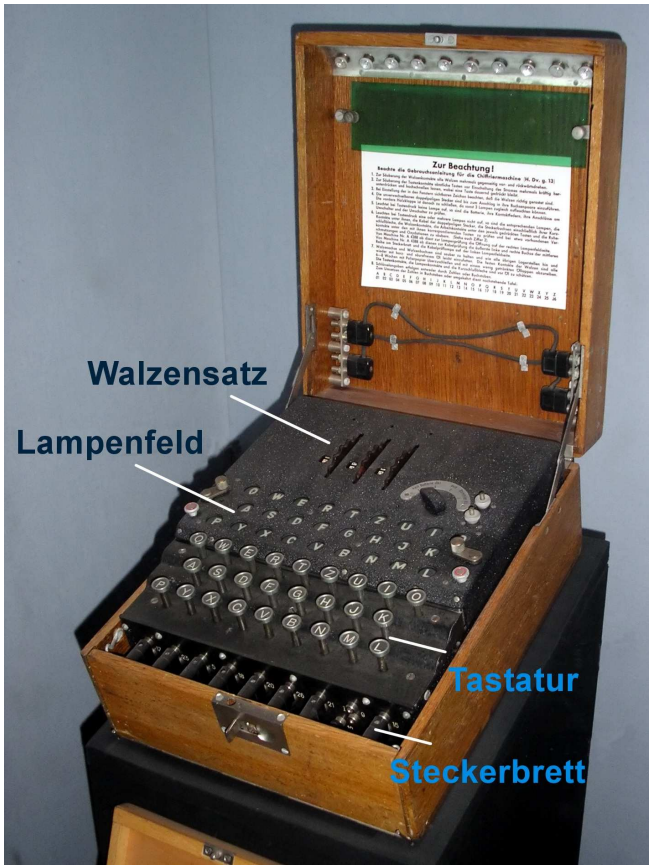
Die Enigma

Erfinder: Arthur **Scherbius**

Eckdaten:

- ca. $2 \cdot 10^{23} \approx 2^{7+9+14+47} = 2^{77} \approx |\mathcal{K}|$ Grundeinstellungen!
- $\mathcal{B} := \{A, B, C, \ldots, Z\}$ und $l := 26 \cdot 25 \cdot 26 = \mathbf{16900}$

Die Enigma



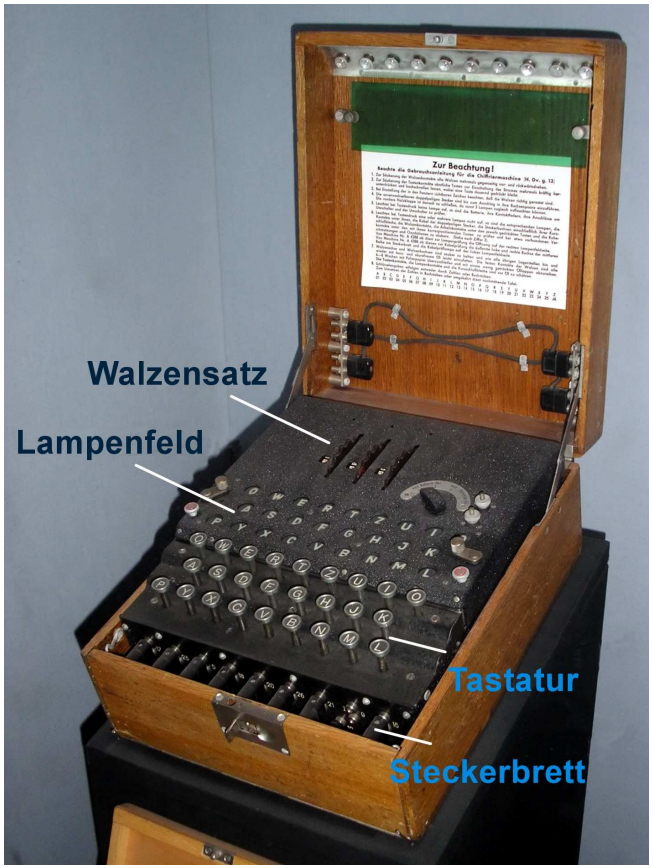
Die Enigma

Erfinder: Arthur **Scherbius**

Eckdaten:

- ca. $2 \cdot 10^{23} \approx 2^{7+9+14+47} = 2^{77} \approx |\mathcal{K}|$ Grundeinstellungen!
- $\mathcal{B} := \{A, B, C, \ldots, Z\}$ und $l := 26 \cdot 25 \cdot 26 = \mathbf{16900}$ (Kasiski-Test wertlos bei Nachrichten der Länge $h < 16900$)

Die Enigma



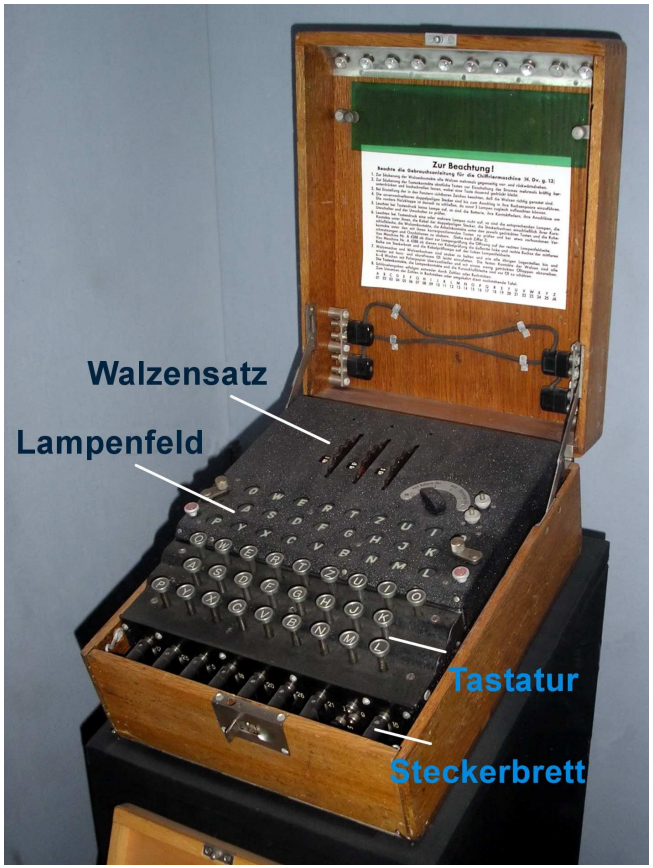
Die Enigma

Erfinder: Arthur **Scherbius**

Eckdaten:

- ca. $2 \cdot 10^{23} \approx 2^{7+9+14+47} = 2^{77} \approx |\mathcal{K}|$ Grundeinstellungen!
- $\mathcal{B} := \{A, B, C, \ldots, Z\}$ und $l := 26 \cdot 25 \cdot 26 = \mathbf{16900}$ (Kasiski-Test wertlos bei Nachrichten der Länge $h < 16900$)
- $\sigma_i = \sigma_i^{-1}$ für alle $i = 1, \ldots, l$ (involutorisch)

Die Enigma



Die Enigma

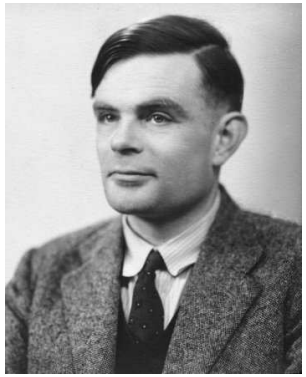
Erfinder: Arthur **Scherbius**

Eckdaten:

- ca. $2 \cdot 10^{23} \approx 2^{7+9+14+47} = 2^{77} \approx |\mathcal{K}|$ Grundeinstellungen!
- $\mathcal{B} := \{A, B, C, \ldots, Z\}$ und $l := 26 \cdot 25 \cdot 26 = \mathbf{16900}$ (Kasiski-Test wertlos bei Nachrichten der Länge $h < 16900$)
- $\sigma_i = \sigma_i^{-1}$ für alle $i = 1, \ldots, l$ (involutorisch)
- fixpunktfrei!

Rejewskis kryptologische Bomba

Marian **Rejewski**
1905 – 1980, Polen



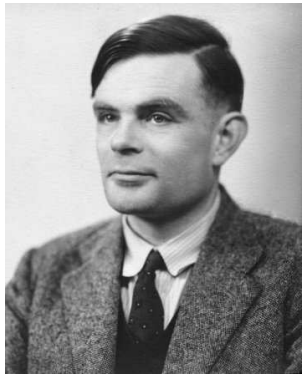
Alan **Turing**
1912 – 1954, GB

Kryptoanalysis: Poznań – Bomba – Bletchley park

- Die Enigma hatte strukturelle Schwächen, die es erlaubt haben, anhand von geratenen Nachrichtenstücken (cribs), Teile des Schlüssels und damit wiederum die Grundeinstellung zu bestimmen.

Rejewskis kryptologische Bomba

Marian **Rejewski**
1905 – 1980, Polen



Alan **Turing**
1912 – 1954, GB

Kryptoanalysis: Poznań – Bomba – Bletchley park

- Die Enigma hatte strukturelle Schwächen, die es erlaubt haben, anhand von geratenen Nachrichtenstücken (cribs), Teile des Schlüssels und damit wiederum die Grundeinstellung zu bestimmen.
- Die größte Schwäche waren aber die Bedienungsfehler der Funkoffiziere.

Wie kann man die geheimen Schlüssel sicher austauschen?

Trotz der hohen kryptographischen Sicherheit der Enigma-M3/M4 der deutschen Marine, zeigte die Erbeutung der Codebücher deutlich, daß die Geheimhaltung der Schlüssel schwer zu realisieren ist.

Wie kann man die geheimen Schlüssel sicher austauschen?

Trotz der hohen kryptographischen Sicherheit der Enigma-M3/M4 der deutschen Marine, zeigte die Erbeutung der Codebücher deutlich, daß die Geheimhaltung der Schlüssel schwer zu realisieren ist. Allein die Existenz von Codebüchern setzte Kerckhoffs Prinzip aufs Spiel.

Wie kann man die geheimen Schlüssel sicher austauschen?

Trotz der hohen kryptographischen Sicherheit der Enigma-M3/M4 der deutschen Marine, zeigte die Erbeutung der Codebücher deutlich, daß die Geheimhaltung der Schlüssel schwer zu realisieren ist. Allein die Existenz von Codebüchern setzte Kerckhoffs Prinzip aufs Spiel.

Eine Alternative mußte her!

Problemstellung

- Bob will eine geheime Nachricht an Alice schicken.

Problemstellung

- Bob will eine geheime Nachricht an Alice schicken.
- Alice und Bob haben keine Möglichkeit sich unter vier Augen zu treffen, um einen Geheimschlüssel für eine symmetrisch verschlüsselte Kommunikation auszutauschen.

Problemstellung

- Bob will eine geheime Nachricht an Alice schicken.
- Alice und Bob haben keine Möglichkeit sich unter vier Augen zu treffen, um einen Geheimschlüssel für eine symmetrisch verschlüsselte Kommunikation auszutauschen.
- Gesucht ist ein Verfahren, welches Alice und Bob ermöglicht, trotz einer öffentlichen Kommunikation den Geheimschlüssel vereinbaren zu können.

Problemstellung

- Bob will eine geheime Nachricht an Alice schicken.
- Alice und Bob haben keine Möglichkeit sich unter vier Augen zu treffen, um einen Geheimschlüssel für eine symmetrisch verschlüsselte Kommunikation auszutauschen.
- Gesucht ist ein Verfahren, welches Alice und Bob ermöglicht, trotz einer öffentlichen Kommunikation den Geheimschlüssel vereinbaren zu können.
- Eine Lauscherin Eve (engl. eavesdropper) darf anhand der öffentlichen Kommunikation von Alice und Bob den vereinbarten Geheimschlüssel nur mit einem *zu hohen* Rechenaufwand ermitteln können.

Problemstellung

- Bob will eine geheime Nachricht an Alice schicken.
- Alice und Bob haben keine Möglichkeit sich unter vier Augen zu treffen, um einen Geheimschlüssel für eine symmetrisch verschlüsselte Kommunikation auszutauschen.
- Gesucht ist ein Verfahren, welches Alice und Bob ermöglicht, trotz einer öffentlichen Kommunikation den Geheimschlüssel vereinbaren zu können.
- Eine Lauscherin Eve (engl. eavesdropper) darf anhand der öffentlichen Kommunikation von Alice und Bob den vereinbarten Geheimschlüssel nur mit einem *zu hohen* Rechenaufwand ermitteln können.

Wir setzen zunächst voraus, daß Eve die öffentliche Kommunikation zwischen Alice und Bob, die zur Schlüsselvereinbarung dient, nicht beeinflussen wird (keine man-in-the-middle Attacke).

Public-Key Kryptosystem

Public-Key Cryptography (PKC)

Man redet von einem Public-Key Kryptosystem (PKC), da die Vereinbarung des Geheimschlüssels öffentlich stattfindet bzw. öffentliche Schlüssel verwendet.

[3] GCHQ=Government Communications Headquarters

Public-Key Kryptosystem

Public-Key Cryptography (PKC)

Man redet von einem Public-Key Kryptosystem (PKC), da die Vereinbarung des Geheimschlüssels öffentlich stattfindet bzw. öffentliche Schlüssel verwendet.

James H. Ellis, Mitarbeiter des britischen Geheimdienstes GCHQ[3], beschrieb bereits 1970 in einer bis 1997 unter Verschluß gehaltenen Arbeit die Idee einer mathematischen „Einwegfunktion“ zur Realisierung eines Public-Key Kryptosystems.

[3] GCHQ=Government Communications Headquarters

Public-Key Kryptosystem

Public-Key Cryptography (PKC)

Man redet von einem Public-Key Kryptosystem (PKC), da die Vereinbarung des Geheimschlüssels öffentlich stattfindet bzw. öffentliche Schlüssel verwendet.

James H. Ellis, Mitarbeiter des britischen Geheimdienstes GCHQ[3], beschrieb bereits 1970 in einer bis 1997 unter Verschluß gehaltenen Arbeit die Idee einer mathematischen „Einwegfunktion“ zur Realisierung eines Public-Key Kryptosystems. Er war aber nicht in der Lage konkret ein solches Kryptosystem zu realisieren.

[3] GCHQ=Government Communications Headquarters

Einwegfunktionen

Einwegfunktionen:

Eine *Einwegfunktion* $f : M \to N$ ist eine Abbildung mit folgenden Eigenschaften:

1. Es existiert ein Algorithmus, der für jedes $m \in M$ den Bildpunkt $n = f(m)$ in *Polynomialzeit* ausrechnet.

Einwegfunktionen

Einwegfunktionen:

Eine *Einwegfunktion* $f : M \to N$ ist eine Abbildung mit folgenden Eigenschaften:

1. Es existiert ein Algorithmus, der für jedes $m \in M$ den Bildpunkt $n = f(m)$ in *Polynomialzeit* ausrechnet.
2. Es existiert kein (probabilistischer) Algorithmus, der für einen gegebenen Bildpunkt $n \in f(M)$ einen Urbildpunkt m in *Polynomialzeit* ausrechnet.

Einwegfunktionen

Einwegfunktionen:

Eine *Einwegfunktion* $f : M \to N$ ist eine Abbildung mit folgenden Eigenschaften:

1. Es existiert ein Algorithmus, der für jedes $m \in M$ den Bildpunkt $n = f(m)$ in *Polynomialzeit* ausrechnet.
2. Es existiert kein (probabilistischer) Algorithmus, der für einen gegebenen Bildpunkt $n \in f(M)$ einen Urbildpunkt m in *Polynomialzeit* ausrechnet.

Dabei ist „polynomial“ in Abhängigkeit von der „Größe“ der Eingabe zu verstehen.

Einwegfunktionen

Einwegfunktionen:

Eine *Einwegfunktion* $f : M \to N$ ist eine Abbildung mit folgenden Eigenschaften:

1. Es existiert ein Algorithmus, der für jedes $m \in M$ den Bildpunkt $n = f(m)$ in *Polynomialzeit* ausrechnet.
2. Es existiert kein (probabilistischer) Algorithmus, der für einen gegebenen Bildpunkt $n \in f(M)$ einen Urbildpunkt m in *Polynomialzeit* ausrechnet.

Dabei ist „polynomial“ in Abhängigkeit von der „Größe“ der Eingabe zu verstehen.

Man denke an das Papier-Telefonbuch.

Einwegfunktionen

Wunsch:

- Bild finden mit *polynomiellem Zeitaufwand* möglich.
- Urbild finden nur mit *exponentiellem Zeitaufwand* möglich.

Einwegfunktionen

Wunsch:

- Bild finden mit *polynomiellem Zeitaufwand* möglich.
- Urbild finden nur mit *exponentiellem Zeitaufwand* möglich.

$\exists$ Einwegfunktion $\Longrightarrow$ P $\neq$ NP

Die Existenz einer Einwegfunktion würde P $\neq$ NP implizieren:

Einwegfunktionen

Wunsch:

- Bild finden mit *polynomiellem Zeitaufwand* möglich.
- Urbild finden nur mit *exponentiellem Zeitaufwand* möglich.

$\exists$ Einwegfunktion $\Longrightarrow$ P $\neq$ NP

Die Existenz einer Einwegfunktion würde P $\neq$ NP implizieren: Das Finden von Urbildpunkten einer Einwegfunktion f beschreibt ein Problem, wo Lösungen nicht in polynomieller Zeit berechnet werden können, dagegen das Verifizieren einer Lösung in polynomieller Zeit erfolgen kann.

Einwegfunktionen

Wunsch:

- Bild finden mit *polynomiellem Zeitaufwand* möglich.
- Urbild finden nur mit *exponentiellem Zeitaufwand* möglich.

$\exists$ Einwegfunktion $\Longrightarrow$ P $\neq$ NP + US$ 1.000.000

Die Existenz einer Einwegfunktion würde P $\neq$ NP implizieren: Das Finden von Urbildpunkten einer Einwegfunktion f beschreibt ein Problem, wo Lösungen nicht in polynomieller Zeit berechnet werden können, dagegen das Verifizieren einer Lösung in polynomieller Zeit erfolgen kann.

Einwegfunktionen

Kandidaten für Einwegfunktionen (die Hoffnung stirbt zuletzt)

- Multiplizieren von zwei unterschiedlichen Primzahlen mit Faktorisierung als inverse Funktion.

Einwegfunktionen

Kandidaten für Einwegfunktionen (die Hoffnung stirbt zuletzt)

- Multiplizieren von zwei unterschiedlichen Primzahlen mit Faktorisierung als inverse Funktion.

Man spricht vom

- Faktorisierungsproblem großer Zahlen

Einwegfunktionen

Kandidaten für Einwegfunktionen (die Hoffnung stirbt zuletzt)

- Multiplizieren von zwei unterschiedlichen Primzahlen mit Faktorisierung als inverse Funktion.
- Potenzieren von Gruppenelementen mit Logarithmus als inverse Funktion.

Man spricht vom

- Faktorisierungsproblem großer Zahlen

Einwegfunktionen

Kandidaten für Einwegfunktionen (die Hoffnung stirbt zuletzt)

- Multiplizieren von zwei unterschiedlichen Primzahlen mit Faktorisierung als inverse Funktion.
- Potenzieren von Gruppenelementen mit Logarithmus als inverse Funktion.

Man spricht vom

- Faktorisierungsproblem großer Zahlen
- diskreten Logarithmus-Problem.

Einwegfunktionen

Kandidaten für Einwegfunktionen (die Hoffnung stirbt zuletzt)

- Multiplizieren von zwei unterschiedlichen Primzahlen mit Faktorisierung als inverse Funktion.
- Potenzieren von Gruppenelementen mit Logarithmus als inverse Funktion.

Man spricht vom

- Faktorisierungsproblem großer Zahlen
- diskreten Logarithmus-Problem.

Bislang sind beide Probleme nur mit einem *zu hohen* Rechenaufwand lösbar.

Diffie-Hellman Schlüsselaustausch Verfahren

Whitfield **Diffie**

CSO Sun Microsystems

Martin **Hellman**

Stanford University (Emeritus)

haben in einer gemeinsamen Arbeit von 1976 ein Public-Key Kryptosystem vorgestellt, das auf dem diskreten Logarithmus-Problem basiert.

Diffie-Hellman Schlüsselaustausch-Verfahren

Diffie-Hellman Schlüsselaustausch-Verfahren

1. Alice und Bob einigen sich öffentlich auf eine endliche zyklische Gruppe C der Ordnung q und einen Erzeuger g.

Diffie-Hellman Schlüsselaustausch-Verfahren

Diffie-Hellman Schlüsselaustausch-Verfahren

1. Alice und Bob einigen sich öffentlich auf eine endliche zyklische Gruppe C der Ordnung q und einen Erzeuger g.
2. Alice wählt einen *zufälligen geheimen* Exponenten $A < q$ und berechnet ihren *öffentlichen* Schlüssel $a := g^A \in C$.
   Bob wählt einen *zufälligen geheimen* Exponenten $B < q$ und berechnet seinen *öffentlichen* Schlüssel $b := g^B \in C$.

Diffie-Hellman Schlüsselaustausch-Verfahren

Diffie-Hellman Schlüsselaustausch-Verfahren

1. Alice und Bob einigen sich öffentlich auf eine endliche zyklische Gruppe C der Ordnung q und einen Erzeuger g.
2. Alice wählt einen *zufälligen geheimen* Exponenten $A < q$ und berechnet ihren *öffentlichen* Schlüssel $a := g^A \in C$.
   Bob wählt einen *zufälligen geheimen* Exponenten $B < q$ und berechnet seinen *öffentlichen* Schlüssel $b := g^B \in C$.
3. Alice und Bob tauschen ihre öffentlichen Schlüssel a und b aus.

Diffie-Hellman Schlüsselaustausch-Verfahren

Diffie-Hellman Schlüsselaustausch-Verfahren

1. Alice und Bob einigen sich öffentlich auf eine endliche zyklische Gruppe C der Ordnung q und einen Erzeuger g.
2. Alice wählt einen *zufälligen geheimen* Exponenten $A < q$ und berechnet ihren *öffentlichen* Schlüssel $a := g^A \in C$.
   Bob wählt einen *zufälligen geheimen* Exponenten $B < q$ und berechnet seinen *öffentlichen* Schlüssel $b := g^B \in C$.
3. Alice und Bob tauschen ihre öffentlichen Schlüssel a und b aus.
4. Alice berechnet den gemeinsamen *Geheim*schlüssel $k := b^A$.
   Bob berechnet den gemeinsamen *Geheim*schlüssel $k := a^B$.

Beweis: $b^A = (g^B)^A = k = (g^A)^B = a^B \in C$. □

Diffie-Hellman Schlüsselaustausch-Verfahren

Diffie-Hellman Schlüsselaustausch-Verfahren (Fortsetzung)

Weiterhin müssen Alice und Bob sowohl ein symmetrisches Kryptosystem festlegen

Diffie-Hellman Schlüsselaustausch-Verfahren

Diffie-Hellman Schlüsselaustausch-Verfahren (Fortsetzung)

Weiterhin müssen Alice und Bob sowohl ein symmetrisches Kryptosystem festlegen, als auch vereinbaren, wie die zugehörigen symmetrischen Schlüssel durch die Elemente $\ell \in C$ kodiert werden.

Diffie-Hellman Schlüsselaustausch-Verfahren

Diffie-Hellman Schlüsselaustausch-Verfahren (Fortsetzung)

Weiterhin müssen Alice und Bob sowohl ein symmetrisches Kryptosystem festlegen, als auch vereinbaren, wie die zugehörigen symmetrischen Schlüssel durch die Elemente $\ell \in C$ kodiert werden. Beides kann ebenfalls öffentlich geschehen.

Diffie-Hellman Schlüsselaustausch-Verfahren

Diffie-Hellman Schlüsselaustausch-Verfahren (Fortsetzung)

Weiterhin müssen Alice und Bob sowohl ein symmetrisches Kryptosystem festlegen, als auch vereinbaren, wie die zugehörigen symmetrischen Schlüssel durch die Elemente $\ell \in C$ kodiert werden. Beides kann ebenfalls öffentlich geschehen.

Beispiel für C

p eine „große“ Primzahl und $C := \mathbb{F}_p^* = (\mathbb{Z}/p\mathbb{Z} - \{\bar{0}\}, \cdot) \cong C_{p-1}$.

Diffie-Hellman Schlüsselaustausch-Verfahren

Bedienungsfehler vermeiden:

- Die Gruppe C, der Erzeuger g und die geheimen Exponenten A und B sind so zu „wählen“, daß Eve aus der Kenntnis von g, a und b die Exponenten $A = \log_g a$ und $B = \log_g b$ nur mit einem *zu hohen* Rechenaufwand ermitteln kann.

Diffie-Hellman Schlüsselaustausch-Verfahren

Bedienungsfehler vermeiden:

- Die Gruppe C, der Erzeuger g und die geheimen Exponenten A und B sind so zu „wählen“, daß Eve aus der Kenntnis von g, a und b die Exponenten $A = \log_g a$ und $B = \log_g b$ nur mit einem *zu hohen* Rechenaufwand ermitteln kann.
- A und B sind geheim zu halten.

Diffie-Hellman Schlüsselaustausch-Verfahren

Bedienungsfehler vermeiden:

- Die Gruppe C, der Erzeuger g und die geheimen Exponenten A und B sind so zu „wählen“, daß Eve aus der Kenntnis von g, a und b die Exponenten $A = \log_g a$ und $B = \log_g b$ nur mit einem *zu hohen* Rechenaufwand ermitteln kann.
- A und B sind geheim zu halten.
- k ist geheim zu halten.

Diffie-Hellman Schlüsselaustausch-Verfahren

Das Diffie-Hellman Schlüsselaustausch-Verfahren basiert auf der Annahme, daß zusätzlich zum diskreten Logarithmus-Problem folgende Probleme hart zu lösen sind:

Diffie-Hellman Schlüsselaustausch-Verfahren

Das Diffie-Hellman Schlüsselaustausch-Verfahren basiert auf der Annahme, daß zusätzlich zum diskreten Logarithmus-Problem folgende Probleme hart zu lösen sind:

Das Diffie-Hellman Problem (DH-Problem)

Gegeben g, $g^A =: a$ und $g^B =: b \in C$ berechne $g^{AB} =: k$.

Diffie-Hellman Schlüsselaustausch-Verfahren

Das Diffie-Hellman Schlüsselaustausch-Verfahren basiert auf der Annahme, daß zusätzlich zum diskreten Logarithmus-Problem folgende Probleme hart zu lösen sind:

Das Diffie-Hellman Problem (DH-Problem)

Gegeben g, $g^A =: a$ und $g^B =: b \in C$ berechne $g^{AB} =: k$.

Das Decision-Diffie-Hellman Problem (DDH-Problem)

Gegeben g, $g^A =: a$, $g^B =: b$ und ein weiteres Element $\ell \in C$ entscheide, ob $\ell = g^{AB}$.

Diffie-Hellman Schlüsselaustausch-Verfahren

Nachteile

- So wie dieses Verfahren beschrieben wurde, eignet es sich zunächst nur zum Austausch des Geheimschlüssels k.

Diffie-Hellman Schlüsselaustausch-Verfahren

Nachteile

- So wie dieses Verfahren beschrieben wurde, eignet es sich zunächst nur zum Austausch des Geheimschlüssels k.
- Möchte auch Ben eine geheime Nachricht an Alice schicken, so muß er mit ihr einen anderen Geheimschlüssel k' vereinbaren

Diffie-Hellman Schlüsselaustausch-Verfahren

Nachteile

- So wie dieses Verfahren beschrieben wurde, eignet es sich zunächst nur zum Austausch des Geheimschlüssels k.
- Möchte auch Ben eine geheime Nachricht an Alice schicken, so muß er mit ihr einen anderen Geheimschlüssel k' vereinbaren, d.h. Alice muß wieder aktiv werden.

Diffie-Hellman Schlüsselaustausch-Verfahren

Nachteile

- So wie dieses Verfahren beschrieben wurde, eignet es sich zunächst nur zum Austausch des Geheimschlüssels k.
- Möchte auch Ben eine geheime Nachricht an Alice schicken, so muß er mit ihr einen anderen Geheimschlüssel k' vereinbaren, d.h. Alice muß wieder aktiv werden. Dieses Verfahren, so wie es beschrieben wurde, eignet sich daher nicht für Einwegkommunikationen (Emails).

Problemstellung

- Alice möchte in der Lage sein geheime Nachrichten (etwa als Emails) zu erhalten, ohne mit jedem Sender einen separaten geheimen Schlüssel vereinbaren zu müssen.

Problemstellung

- Alice möchte in der Lage sein geheime Nachrichten (etwa als Emails) zu erhalten, ohne mit jedem Sender einen separaten geheimen Schlüssel vereinbaren zu müssen.
- Eine solche Nachricht könnte wieder ein Geheimschlüssel für eine symmetrisch verschlüsselte Kommunikation sein.

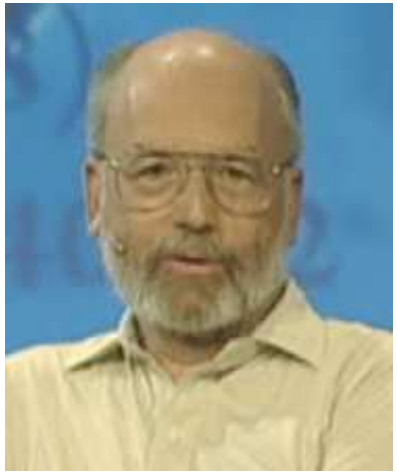
Problemstellung

- Alice möchte in der Lage sein geheime Nachrichten (etwa als Emails) zu erhalten, ohne mit jedem Sender einen separaten geheimen Schlüssel vereinbaren zu müssen.
- Eine solche Nachricht könnte wieder ein Geheimschlüssel für eine symmetrisch verschlüsselte Kommunikation sein.
- Idee: Alice erzeugt einen *öffentlichen* Schlüssel und einen im folgenden Sinne dazu passenden *privaten* geheimen Schlüssel: Der öffentliche Schlüssel kann von jedem dazu benutzt werden eine geheime Nachricht m an Alice vor dem Senden zu chiffrieren, und der private Schlüssel ermöglicht Alice wiederum die chiffrierte Nachricht $\check{m}$ zu dechiffrieren.

RSA-Kryptosystem

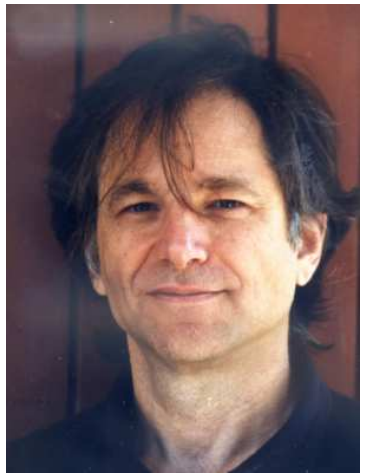
Ronald **Rivest**

MIT



Adi **Shamir**

Weizmann Institute



Leonard **Adleman**

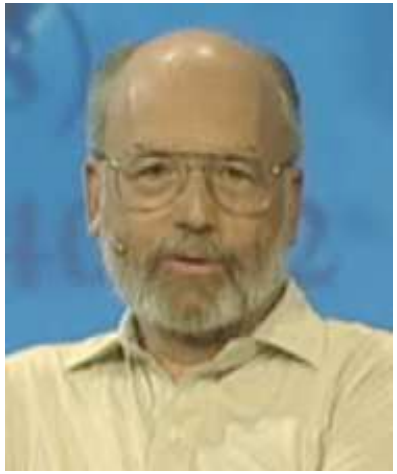
USC

haben in einer gemeinsamen Arbeit von 1978 ein Public-Key Kryptosystem vorgestellt, das auf dem Faktorisierungsproblem großer Zahlen basiert, und obige Idee umsetzt.

RSA-Kryptosystem

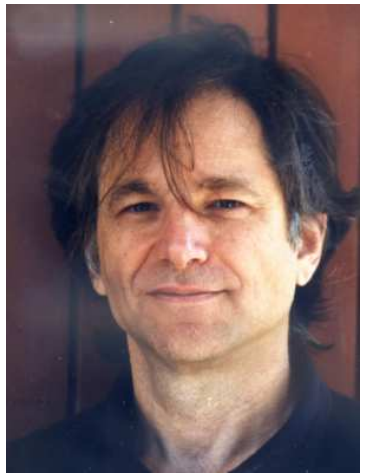
Ronald **Rivest**

MIT



Adi **Shamir**

Weizmann Institute



Leonard **Adleman**

USC

haben in einer gemeinsamen Arbeit von 1978 ein Public-Key Kryptosystem vorgestellt, das auf dem Faktorisierungsproblem großer Zahlen basiert, und obige Idee umsetzt. Sie gründeten 1982 ihre Firma RSA Data Security, die 2006 von EMC Corporation für 2.1 Milliarden US$ akquiriert wurde.

RSA-Kryptosystem

RSA: Erzeugung des Schlüsselpaars

1. Alice wählt zwei *zufällige unterschiedliche* Primzahlen p und q und berechnet ihr Produkt $n = pq$ und $\phi = (p-1)(q-1)$.

RSA-Kryptosystem

RSA: Erzeugung des Schlüsselpaars

1. Alice wählt zwei *zufällige unterschiedliche* Primzahlen p und q und berechnet ihr Produkt $n = pq$ und $\phi = (p-1)(q-1)$.
2. Alice wählt einen Exponenten $A > 1$, teilerfremd zu ϕ.

RSA-Kryptosystem

RSA: Erzeugung des Schlüsselpaars

1. Alice wählt zwei *zufällige unterschiedliche* Primzahlen p und q und berechnet ihr Produkt $n = pq$ und $\phi = (p-1)(q-1)$.
2. Alice wählt einen Exponenten $A > 1$, teilerfremd zu ϕ.
3. Alice veröffentlicht (n, A) als ihren *öffentlichen Schlüssel*.

RSA-Kryptosystem

RSA: Erzeugung des Schlüsselpaars

1. Alice wählt zwei *zufällige unterschiedliche* Primzahlen p und q und berechnet ihr Produkt $n = pq$ und $\phi = (p-1)(q-1)$.
2. Alice wählt einen Exponenten $A > 1$, teilerfremd zu ϕ.
3. Alice veröffentlicht (n, A) als ihren *öffentlichen Schlüssel*.
4. Alice berechnet mit Hilfe des erweiterten Euklidischen Algorithmus ihren *privaten Schlüssel* $K < \phi$ mit $AK \equiv 1 \mod \phi$.

RSA-Kryptosystem

RSA: Erzeugung des Schlüsselpaars

1. Alice wählt zwei *zufällige unterschiedliche* Primzahlen p und q und berechnet ihr Produkt $n = pq$ und $\phi = (p-1)(q-1)$.
2. Alice wählt einen Exponenten $A > 1$, teilerfremd zu ϕ.
3. Alice veröffentlicht (n, A) als ihren *öffentlichen Schlüssel*.
4. Alice berechnet mit Hilfe des erweiterten Euklidischen Algorithmus ihren *privaten Schlüssel* $K < \phi$ mit $AK \equiv 1 \mod \phi$.

RSA (Fortsetzung)

Alice muß noch öffentlich festlegen, wie Buchstaben bzw. Nachrichten durch die Elemente von $\mathbb{Z}/n\mathbb{Z} - X$ kodiert werden.

RSA-Kryptosystem

Jetzt kann Bob (wie jeder andere auch!) eine geheime Nachricht $m \in \mathbb{Z}/n\mathbb{Z} - X$ verfassen, sie durch $\boxed{\check{m} := m^A \mod n}$ chiffrieren und anschließend an Alice schicken.

RSA-Kryptosystem

Jetzt kann Bob (wie jeder andere auch!) eine geheime Nachricht $m \in \mathbb{Z}/n\mathbb{Z} - X$ verfassen, sie durch $\boxed{\check{m} := m^A \mod n}$ chiffrieren und anschließend an Alice schicken.

Alice kann Bobs geheime Nachricht m aus der chiffrierten Nachricht $\check{m}$ durch Potenzieren mit ihrem geheimen Schlüssel K rekonstruieren:

$$\boxed{m = \check{m}^K \mod n}$$

RSA-Kryptosystem

Jetzt kann Bob (wie jeder andere auch!) eine geheime Nachricht $m \in \mathbb{Z}/n\mathbb{Z} - X$ verfassen, sie durch $\boxed{\check{m} := m^A \mod n}$ chiffrieren und anschließend an Alice schicken.

Alice kann Bobs geheime Nachricht m aus der chiffrierten Nachricht $\check{m}$ durch Potenzieren mit ihrem geheimen Schlüssel K rekonstruieren:

$$\boxed{m = \check{m}^K \mod n}$$

Beweis: Für alle $a \in \mathbb{Z}/n\mathbb{Z}$ gilt $a = a^1 = a^{AK+\phi L} = (a^A)^K (a^\phi)^L = (a^A)^K$

RSA-Kryptosystem

Jetzt kann Bob (wie jeder andere auch!) eine geheime Nachricht $m \in \mathbb{Z}/n\mathbb{Z} - X$ verfassen, sie durch $\boxed{\check{m} := m^A \mod n}$ chiffrieren und anschließend an Alice schicken.

Alice kann Bobs geheime Nachricht m aus der chiffrierten Nachricht $\check{m}$ durch Potenzieren mit ihrem geheimen Schlüssel K rekonstruieren:

$$\boxed{m = \check{m}^K \mod n}$$

Beweis: Für alle $a \in \mathbb{Z}/n\mathbb{Z}$ gilt $a = a^1 = a^{AK+\phi L} = (a^A)^K (a^\phi)^L = (a^A)^K$, da nach dem chinesischen Restsatz $\mathbb{Z}/n\mathbb{Z} \cong \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$ und daher für alle $a = (a_1, a_2) \in \mathbb{Z}/n\mathbb{Z}$:
$a^\phi \in \{\bar{0} = (0,0), (1,0), (0,1), (1,1) = \bar{1}\}$. □

RSA-Kryptosystem

Bedienungsfehler vermeiden:

- Die Primzahlen p und q sind so zu „wählen“, daß Eve aus der Kenntnis von n die zwei Primfaktoren p und q nur mit einem *zu hohen* Rechenaufwand ermitteln kann. (Siehe RSA Factoring Challenge)

RSA-Kryptosystem

Bedienungsfehler vermeiden:

- Die Primzahlen p und q sind so zu „wählen“, daß Eve aus der Kenntnis von n die zwei Primfaktoren p und q nur mit einem *zu hohen* Rechenaufwand ermitteln kann. (Siehe RSA Factoring Challenge)
- Der öffentliche Schlüssel A ist so zu wählen, daß Eve aus der Kenntnis von A und n den privaten Schlüssel K nur mit einem *zu hohen* Rechenaufwand ermitteln kann.

RSA-Kryptosystem

Bedienungsfehler vermeiden:

- Die Primzahlen p und q sind so zu „wählen“, daß Eve aus der Kenntnis von n die zwei Primfaktoren p und q nur mit einem *zu hohen* Rechenaufwand ermitteln kann. (Siehe RSA Factoring Challenge)
- Der öffentliche Schlüssel A ist so zu wählen, daß Eve aus der Kenntnis von A und n den privaten Schlüssel K nur mit einem *zu hohen* Rechenaufwand ermitteln kann.
- Die Ausnahmemenge X muß mindestens $\{\bar{k} \mid k^A < n\}$ umfassen (üblich: $A \geq 65537$ oder $A = 3$).

RSA-Kryptosystem

Bedienungsfehler vermeiden:

- Die Primzahlen p und q sind so zu „wählen“, daß Eve aus der Kenntnis von n die zwei Primfaktoren p und q nur mit einem *zu hohen* Rechenaufwand ermitteln kann. (Siehe RSA Factoring Challenge)
- Der öffentliche Schlüssel A ist so zu wählen, daß Eve aus der Kenntnis von A und n den privaten Schlüssel K nur mit einem *zu hohen* Rechenaufwand ermitteln kann.
- Die Ausnahmemenge X muß mindestens $\{\bar{k} \mid k^A < n\}$ umfassen (üblich: $A \geq 65537$ oder $A = 3$).
- p, q, ϕ und K sind geheim zu halten.

RSA-Kryptosystem

Bedienungsfehler vermeiden:

- Die Primzahlen p und q sind so zu „wählen“, daß Eve aus der Kenntnis von n die zwei Primfaktoren p und q nur mit einem *zu hohen* Rechenaufwand ermitteln kann. (Siehe RSA Factoring Challenge)
- Der öffentliche Schlüssel A ist so zu wählen, daß Eve aus der Kenntnis von A und n den privaten Schlüssel K nur mit einem *zu hohen* Rechenaufwand ermitteln kann.
- Die Ausnahmemenge X muß mindestens $\{\bar{k} \mid k^A < n\}$ umfassen (üblich: $A \geq 65537$ oder $A = 3$).
- p, q, ϕ und K sind geheim zu halten.

Bei der OpenSSL-Implementation besteht der private Schlüssel aus $(n, A, K, p, q, K \mod (p-1), K \mod (q-1), q^{-1} \mod p)$.

Elgamal-Kryptosystem

Taher **Elgamal**

CTO Tumbleweed

(Ph.D.: Stanford)

hat in einer Arbeit von 1984/85 sowohl ein Public-Key Kryptosystem analog zu RSA als auch ein digitales Signierverfahren vorgestellt, das aber auf dem Diffie-Hellman Schlüsselaustausch-Verfahren und somit auf dem diskreten Logarithmus-Problem basiert.

Elgamal-Kryptosystem

Taher **Elgamal**

CTO Tumbleweed

(Ph.D.: Stanford)

hat in einer Arbeit von 1984/85 sowohl ein Public-Key Kryptosystem analog zu RSA als auch ein digitales Signierverfahren vorgestellt, das aber auf dem Diffie-Hellman Schlüsselaustausch-Verfahren und somit auf dem diskreten Logarithmus-Problem basiert. DSA, eine Variante seines Signierverfahrens, wurde im Jahr 1991 vom NSA-Ex-Mitarbeiter David Kravitz vorgeschlagen, und vom „National Institute of Standards and Technology“ (NIST) zum „Digital Signature Standard“ (DSS) deklariert.

Elgamal-Kryptosystem

Taher **Elgamal**

CTO Tumbleweed

(Ph.D.: Stanford)

hat in einer Arbeit von 1984/85 sowohl ein Public-Key Kryptosystem analog zu RSA als auch ein digitales Signierverfahren vorgestellt, das aber auf dem Diffie-Hellman Schlüsselaustausch-Verfahren und somit auf dem diskreten Logarithmus-Problem basiert. DSA, eine Variante seines Signierverfahrens, wurde im Jahr 1991 vom NSA-Ex-Mitarbeiter David Kravitz vorgeschlagen, und vom „National Institute of Standards and Technology“ (NIST) zum „Digital Signature Standard“ (DSS) deklariert.

Als Chef-Wissenschaftler bei Netscape (’95-’98) leitete Elgamal die Entwicklung des SSL-Protokolls (Secure Socket Layer).

Elgamal-Kryptosystem

Elgamal-Verschlüsselungsverfahren: Erzeugung des Schlüsselpaars

1. Alice wählt eine zyklische Gruppe C und einen Erzeuger g. Üblicherweise ist $q := |C|$ eine große Primzahl.

Elgamal-Kryptosystem

Elgamal-Verschlüsselungsverfahren: Erzeugung des Schlüsselpaars

1. Alice wählt eine zyklische Gruppe C und einen Erzeuger g. Üblicherweise ist $q := |C|$ eine große Primzahl.
2. Alice wählt einen *zufälligen geheimen* Exponenten $A < q$ als ihren *privaten* Schlüssel und berechnet $a := g^A$.

Elgamal-Kryptosystem

Elgamal-Verschlüsselungsverfahren: Erzeugung des Schlüsselpaars

1. Alice wählt eine zyklische Gruppe C und einen Erzeuger g. Üblicherweise ist $q := |C|$ eine große Primzahl.
2. Alice wählt einen *zufälligen geheimen* Exponenten $A < q$ als ihren *privaten* Schlüssel und berechnet $a := g^A$.
3. Alice veröffentlicht (C,g,a) als ihren *öffentlichen Schlüssel*.

Elgamal-Kryptosystem

Elgamal-Verschlüsselungsverfahren: Erzeugung des Schlüsselpaars

1. Alice wählt eine zyklische Gruppe C und einen Erzeuger g. Üblicherweise ist $q := |C|$ eine große Primzahl.
2. Alice wählt einen *zufälligen geheimen* Exponenten $A < q$ als ihren *privaten* Schlüssel und berechnet $a := g^A$.
3. Alice veröffentlicht (C,g,a) als ihren *öffentlichen Schlüssel*.

Elgamal-Verschlüsselungsverfahren (Fortsetzung)

Alice muß noch öffentlich festlegen, wie Buchstaben bzw. Nachrichten durch die Elemente von C kodiert werden.

Elgamal-Kryptosystem

1. Bob wählt für jede Nachricht, die er an Alice verschicken möchte, einen *zufälligen geheimen* Exponenten $B < q$ und berechnet $b := g^B$.

Elgamal-Kryptosystem

1. Bob wählt für jede Nachricht, die er an Alice verschicken möchte, einen *zufälligen geheimen* Exponenten $B < q$ und berechnet $b := g^B$.
2. Bob berechnet $k := a^B$.

Elgamal-Kryptosystem

1. Bob wählt für jede Nachricht, die er an Alice verschicken möchte, einen *zufälligen geheimen* Exponenten $B < q$ und berechnet $b := g^B$.
2. Bob berechnet $k := a^B$.

Die zwei Rechnungen kann Bob im Voraus machen.

Elgamal-Kryptosystem

1. Bob wählt für jede Nachricht, die er an Alice verschicken möchte, einen *zufälligen geheimen* Exponenten $B < q$ und berechnet $b := g^B$.
2. Bob berechnet $k := a^B$.

Die zwei Rechnungen kann Bob im Voraus machen.

3. Bob drückt seine geheime Nachricht an Alice als ein Element $m \in C$ aus.

Elgamal-Kryptosystem

1. Bob wählt für jede Nachricht, die er an Alice verschicken möchte, einen *zufälligen geheimen* Exponenten $B < q$ und berechnet $b := g^B$.
2. Bob berechnet $k := a^B$.

Die zwei Rechnungen kann Bob im Voraus machen.

3. Bob drückt seine geheime Nachricht an Alice als ein Element $m \in C$ aus.
4. Bob berechnet seine chiffrierte Nachricht $\boxed{\check{m} := km}$.

Elgamal-Kryptosystem

1. Bob wählt für jede Nachricht, die er an Alice verschicken möchte, einen *zufälligen geheimen* Exponenten $B < q$ und berechnet $b := g^B$.
2. Bob berechnet $k := a^B$.

Die zwei Rechnungen kann Bob im Voraus machen.

3. Bob drückt seine geheime Nachricht an Alice als ein Element $m \in C$ aus.
4. Bob berechnet seine chiffrierte Nachricht $\boxed{\check{m} := km}$.
5. Bob schickt das Paar $(b, \check{m})$ an Alice.

Elgamal-Kryptosystem

1. Bob wählt für jede Nachricht, die er an Alice verschicken möchte, einen *zufälligen geheimen* Exponenten $B < q$ und berechnet $b := g^B$.
2. Bob berechnet $k := a^B$.

Die zwei Rechnungen kann Bob im Voraus machen.

3. Bob drückt seine geheime Nachricht an Alice als ein Element $m \in C$ aus.
4. Bob berechnet seine chiffrierte Nachricht $\boxed{\check{m} := km}$.
5. Bob schickt das Paar $(b, \check{m})$ an Alice.

1. Alice rekonstruiert k durch $b^A = (g^B)^A = (g^A)^B = a^B =: k$.

Elgamal-Kryptosystem

1. Bob wählt für jede Nachricht, die er an Alice verschicken möchte, einen *zufälligen geheimen* Exponenten $B < q$ und berechnet $b := g^B$.
2. Bob berechnet $k := a^B$.

Die zwei Rechnungen kann Bob im Voraus machen.

3. Bob drückt seine geheime Nachricht an Alice als ein Element $m \in C$ aus.
4. Bob berechnet seine chiffrierte Nachricht $\boxed{\check{m} := km}$.
5. Bob schickt das Paar $(b, \check{m})$ an Alice.

1. Alice rekonstruiert k durch $b^A = (g^B)^A = (g^A)^B = a^B =: k$.
2. Alice rekonstruiert dann Bobs geheime Nachricht $\boxed{m = k^{-1}\check{m}}$.

Elgamal-Kryptosystem

Unterschied zum Diffie-Hellman Protokoll

- Bob schickt seinen öffentlichen Schlüssel b an Alice zusammen mit der chiffrierten Nachricht $\check{m}$

Elgamal-Kryptosystem

Unterschied zum Diffie-Hellman Protokoll

- Bob schickt seinen öffentlichen Schlüssel b an Alice zusammen mit der chiffrierten Nachricht $\check{m}$, d.h. „auf den letzten Drücker“.

Elgamal-Kryptosystem

Unterschied zum Diffie-Hellman Protokoll

- Bob schickt seinen öffentlichen Schlüssel b an Alice zusammen mit der chiffrierten Nachricht $\check{m}$, d.h. „auf den letzten Drücker“.
- Das Elgamal-Kryptosystem ist ein „Hybrid-Kryptosystem“, d.h. es kommt mit einem integrierten symmetrischen Verschlüsselungsverfahren.

Elgamal-Kryptosystem

Bedienungsfehler vermeiden:

- Die Bedienungsfehler des Diffie-Hellman Schlüsselaustausch-Verfahrens vermeiden.

Elgamal-Kryptosystem

Bedienungsfehler vermeiden:

- Die Bedienungsfehler des Diffie-Hellman Schlüsselaustausch-Verfahrens vermeiden.
- Es ist wichtig, daß Bob für jede Nachricht, die er an Alice schicken möchte, einen anderen zufälligen geheimen Exponenten B' benutzt:

Elgamal-Kryptosystem

Bedienungsfehler vermeiden:

- Die Bedienungsfehler des Diffie-Hellman Schlüsselaustausch-Verfahrens vermeiden.
- Es ist wichtig, daß Bob für jede Nachricht, die er an Alice schicken möchte, einen anderen zufälligen geheimen Exponenten B' benutzt: Verwendet Bob etwa für die Nachricht m' ebenfalls den Exponenten B so kriegt Eve das mit, da sich $b := g^B$ als erster Teil der Nachricht nicht verändern wird.

Elgamal-Kryptosystem

Bedienungsfehler vermeiden:

- Die Bedienungsfehler des Diffie-Hellman Schlüsselaustausch-Verfahrens vermeiden.
- Es ist wichtig, daß Bob für jede Nachricht, die er an Alice schicken möchte, einen anderen zufälligen geheimen Exponenten B' benutzt: Verwendet Bob etwa für die Nachricht m' ebenfalls den Exponenten B so kriegt Eve das mit, da sich $b := g^B$ als erster Teil der Nachricht nicht verändern wird. Wird die ursprüngliche Nachricht m irgendwann doch publik, so kann Eve den geheimen Schlüssel k als $k = \check{m} \cdot m^{-1}$ berechnen, und die neue geheime Nachricht m' als $m' = k^{-1}\check{m}'$ rekonstruieren.

Problemstellung

- Alice möchte in der Lage sein ihre Nachrichten zu signieren.

Problemstellung

- Alice möchte in der Lage sein ihre Nachrichten zu signieren.
- Jeder Empfänger soll verifizieren können, ob Alice die Verfasserin einer Nachricht ist oder nicht.

Problemstellung

- Alice möchte in der Lage sein ihre Nachrichten zu signieren.
- Jeder Empfänger soll verifizieren können, ob Alice die Verfasserin einer Nachricht ist oder nicht.
- Idee: Alice benutzt ihren geheimen Schlüssel um ihre Nachrichten zu signieren.

Problemstellung

- Alice möchte in der Lage sein ihre Nachrichten zu signieren.
- Jeder Empfänger soll verifizieren können, ob Alice die Verfasserin einer Nachricht ist oder nicht.
- Idee: Alice benutzt ihren geheimen Schlüssel um ihre Nachrichten zu signieren. Jeder Empfänger kann mit Hilfe von Alice öffentlichem Schlüssel verifizieren, daß Alice die Verfasserin ist.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur

1. Alice ist weiterhin im Besitz ihres öffentlichen Schlüssels (C, g, a) und des dazu passenden privaten Schlüssel A.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur

1. Alice ist weiterhin im Besitz ihres öffentlichen Schlüssels (C, g, a) und des dazu passenden privaten Schlüssel A.
2. Alice veröffentlicht ebenfalls eine Funktion $F : C \to \mathbb{Z}$.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur

1. Alice ist weiterhin im Besitz ihres öffentlichen Schlüssels (C, g, a) und des dazu passenden privaten Schlüssel A.
2. Alice veröffentlicht ebenfalls eine Funktion $F : C \to \mathbb{Z}$. Die einzige relevante Eigenschaft dieser Funktion ist, daß die Mächtigkeit ihrer Fasern „sehr klein“ ist.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur

1. Alice ist weiterhin im Besitz ihres öffentlichen Schlüssels (C, g, a) und des dazu passenden privaten Schlüssel A.
2. Alice veröffentlicht ebenfalls eine Funktion $F : C \to \mathbb{Z}$. Die einzige relevante Eigenschaft dieser Funktion ist, daß die Mächtigkeit ihrer Fasern „sehr klein“ ist.
3. Alice verfasst ihre Nachricht und drückt sie als eine ganze Zahl $M \in \mathbb{Z}$ aus.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur

1. Alice ist weiterhin im Besitz ihres öffentlichen Schlüssels (C, g, a) und des dazu passenden privaten Schlüssel A.
2. Alice veröffentlicht ebenfalls eine Funktion $F : C \to \mathbb{Z}$. Die einzige relevante Eigenschaft dieser Funktion ist, daß die Mächtigkeit ihrer Fasern „sehr klein“ ist.
3. Alice verfasst ihre Nachricht und drückt sie als eine ganze Zahl $M \in \mathbb{Z}$ aus.
4. Alice wählt öffentlich eine Hash-Funktion[a] $H : \mathbb{Z} \to \mathbb{N}$ und berechnet den Hash-Wert $H(M)$ ihrer Nachricht.

[a] Die prominentesten Hash-Funktionen sind MD5 und SHA.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur (Fortsetzung)

5. Alice wählt einen *zufälligen geheimen* Exponenten R, teilerfremd zu $q := |C|$ und berechnet $r := g^R$.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur (Fortsetzung)

5. Alice wählt einen *zufälligen geheimen* Exponenten R, teilerfremd zu $q := |C|$ und berechnet $r := g^R$.
6. Alice berechnet $S := R^{-1}[H(M) - AF(r)] \mod q$.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur (Fortsetzung)

5. Alice wählt einen *zufälligen geheimen* Exponenten R, teilerfremd zu $q := |C|$ und berechnet $r := g^R$.
6. Alice berechnet $S := R^{-1}[H(M) - AF(r)] \mod q$.
7. Alice signiert ihre Nachricht M mit (r, S).

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur (Fortsetzung)

5. Alice wählt einen *zufälligen geheimen* Exponenten R, teilerfremd zu $q := |C|$ und berechnet $r := g^R$.
6. Alice berechnet $S := R^{-1}[H(M) - AF(r)] \mod q$.
7. Alice signiert ihre Nachricht M mit (r, S).

Elgamal-Signierverfahren: Validierung der Signatur

Jeder kann überprüfen ob $S < q$ und ob $r^S a^{F(r)} = g^{H(M)} \in C$.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur (Fortsetzung)

5. Alice wählt einen *zufälligen geheimen* Exponenten R, teilerfremd zu $q := |C|$ und berechnet $r := g^R$.
6. Alice berechnet $S := R^{-1}[H(M) - AF(r)] \mod q$.
7. Alice signiert ihre Nachricht M mit (r, S).

Elgamal-Signierverfahren: Validierung der Signatur

Jeder kann überprüfen ob $S < q$ und ob $r^S a^{F(r)} = g^{H(M)} \in C$.
Alice ist „dann und nur dann“ die Verfasserin der Nachricht M, wenn die Gleichheit gilt.

Elgamal-Signierverfahren

Elgamal-Signierverfahren: Erzeugung der Signatur (Fortsetzung)

5. Alice wählt einen *zufälligen geheimen* Exponenten R, teilerfremd zu $q := |C|$ und berechnet $r := g^R$.
6. Alice berechnet $S := R^{-1}[H(M) - AF(r)] \mod q$.
7. Alice signiert ihre Nachricht M mit (r, S).

Elgamal-Signierverfahren: Validierung der Signatur

Jeder kann überprüfen ob $S < q$ und ob $r^S a^{F(r)} = g^{H(M)} \in C$.
Alice ist „dann und nur dann“ die Verfasserin der Nachricht M, wenn die Gleichheit gilt.

Beweis: $r^S a^{F(r)} = (g^R)^{R^{-1}[H(M)-AF(r)]}(g^A)^{F(r)} = g^{H(M)}$. □

Elgamal-Signierverfahren

Bedienungsfehler vermeiden:

- Der Exponent R ist geheim zu halten und sollte nach der Signierung vernichtet werden

Elgamal-Signierverfahren

Bedienungsfehler vermeiden:

- Der Exponent R ist geheim zu halten und sollte nach der Signierung vernichtet werden: Eve könnte aus der Kenntnis von R sogar Alice' privaten Schlüssel A mit Hilfe der definierenden Gleichung $\boxed{S = R^{-1}[H(M) - AF(r)] \mod q}$ leicht berechnen.

Elgamal-Signierverfahren

Bedienungsfehler vermeiden:

- Der Exponent R ist geheim zu halten und sollte nach der Signierung vernichtet werden: Eve könnte aus der Kenntnis von R sogar Alice' privaten Schlüssel A mit Hilfe der definierenden Gleichung $\boxed{S = R^{-1}[H(M) - AF(r)] \mod q}$ leicht berechnen. Damit kann Eve Nachrichten im Namen von Alice signieren.

Elgamal-Signierverfahren

Bedienungsfehler vermeiden:

- Der Exponent R ist geheim zu halten und sollte nach der Signierung vernichtet werden: Eve könnte aus der Kenntnis von R sogar Alice' privaten Schlüssel A mit Hilfe der definierenden Gleichung $\boxed{S = R^{-1}[H(M) - AF(r)] \mod q}$ leicht berechnen. Damit kann Eve Nachrichten im Namen von Alice signieren.
- Benutzt Alice denselben geheimen Exponenten R für zwei verschiedene Nachrichten M und M', so kriegt Eve das mit, da sich $r := g^R$ als erster Teil der Signatur nicht verändern wird.

Elgamal-Signierverfahren

Bedienungsfehler vermeiden:

- Der Exponent R ist geheim zu halten und sollte nach der Signierung vernichtet werden: Eve könnte aus der Kenntnis von R sogar Alice' privaten Schlüssel A mit Hilfe der definierenden Gleichung $\boxed{S = R^{-1}[H(M) - AF(r)] \mod q}$ leicht berechnen. Damit kann Eve Nachrichten im Namen von Alice signieren.
- Benutzt Alice denselben geheimen Exponenten R für zwei verschiedene Nachrichten M und M', so kriegt Eve das mit, da sich $r := g^R$ als erster Teil der Signatur nicht verändern wird. Eve kann dann aus den beiden definierenden Gleichungen $\boxed{RS = H(M) - AF(r) \mod q, \quad RS' = H(M') - AF(r) \mod q}$ sowohl R als auch den privaten Schlüssel A berechnen.

Elliptische Kurven

Jetzt machen wir uns auf die Suche nach Kandidaten für die Gruppe C (außer $C = \mathbb{F}_q^*$).

Elliptische Kurven

Jetzt machen wir uns auf die Suche nach Kandidaten für die Gruppe C (außer $C = \mathbb{F}_q^*$).

Elliptische Kurve

Sei k ein Körper.

Elliptische Kurven

Jetzt machen wir uns auf die Suche nach Kandidaten für die Gruppe C (außer $C = \mathbb{F}_q^*$).

Elliptische Kurve

Sei k ein Körper. Eine **elliptische Kurve** E in Weierstrass-Normalform ist eine kubische Gleichung der Form

$$E: \ y^2 = x^3 + Ax + B,$$

mit $A, B \in k$ und $4A^3 + 27B^2 \neq 0$.

Elliptische Kurven

Jetzt machen wir uns auf die Suche nach Kandidaten für die Gruppe C (außer $C = \mathbb{F}_q^*$).

Elliptische Kurve

Sei k ein Körper. Eine **elliptische Kurve** E in Weierstrass-Normalform ist eine kubische Gleichung der Form

$$E: \ y^2 = x^3 + Ax + B,$$

mit $A, B \in k$ und $4A^3 + 27B^2 \neq 0$.
Ist K ein Oberkörper von k, so bezeichnet $E(K)$ die Menge der Lösungen (x, y) von E in $K \times K$, vereinigt mit $\{\infty\}$.

Elliptische Kurven

Jetzt machen wir uns auf die Suche nach Kandidaten für die Gruppe C (außer $C = \mathbb{F}_q^*$).

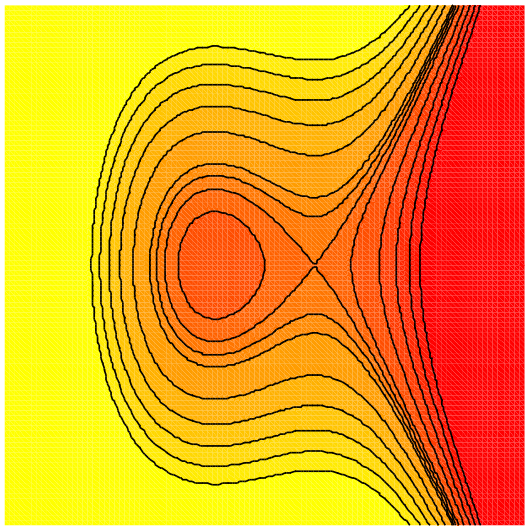
Elliptische Kurve

Sei k ein Körper. Eine **elliptische Kurve** E in Weierstrass-Normalform ist eine kubische Gleichung der Form

$$E: \ y^2 = x^3 + Ax + B,$$

mit $A, B \in k$ und $4A^3 + 27B^2 \neq 0$.
Ist K ein Oberkörper von k, so bezeichnet $E(K)$ die Menge der Lösungen (x, y) von E in $K \times K$, vereinigt mit $\{\infty\}$. ∞ nennt man den unendlich fernen Punkt.

Elliptische Kurven kommen in Familien

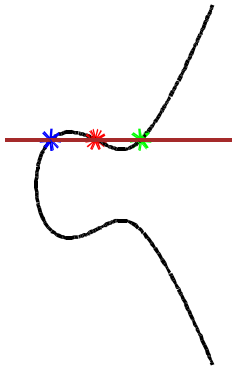


Eine Familie elliptischer Kurven über $\mathbb{R}$.

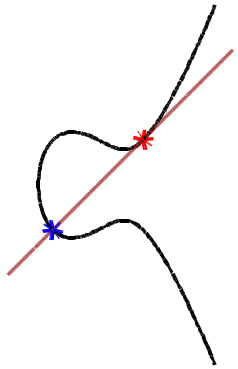
Elliptische Kurven als abelsche Gruppen

Abel

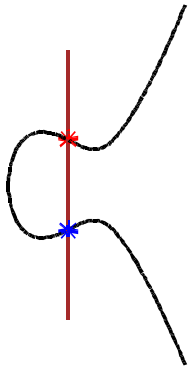
$E(K)$ ist mit folgender Verknüpfung eine abelsche Gruppe mit $\infty = 0$.



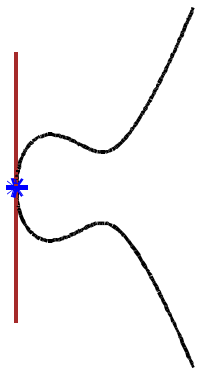
$P + Q + R = 0$



$P + Q + Q = 0$



$P + Q + 0 = 0$



$P + P + 0 = 0$

Die Gruppenstruktur auf der *elliptischen* Kurve $y^2 = x^3 - x + 1$.

Struktursatz

Struktursatz

Sei E eine elliptische Kurve über dem Körper k und sei n eine natürliche Zahl.

Struktursatz

Struktursatz

Sei E eine elliptische Kurve über dem Körper k und sei n eine natürliche Zahl. Wir bezeichnen mit $E[n]$ die Menge der n-Torsionspunkte von $E(\bar{k})$.

Struktursatz

Struktursatz

Sei E eine elliptische Kurve über dem Körper k und sei n eine natürliche Zahl. Wir bezeichnen mit $E[n]$ die Menge der n-Torsionspunkte von $E(\bar{k})$. Ist $\mathrm{char}(k) \nmid n$ dann gilt:

$$E[n] \cong \mathbb{Z}/n\mathbb{Z} \oplus \mathbb{Z}/n\mathbb{Z}.$$

Ist $p := \mathrm{char}(k) \mid n$, und $n = n'p^r$ mit $p \nmid n'$ so ist

$$E[n] \cong \mathbb{Z}/n'\mathbb{Z} \oplus \mathbb{Z}/n'\mathbb{Z}, \quad \text{oder} \quad E[n] \cong \mathbb{Z}/n\mathbb{Z} \oplus \mathbb{Z}/n'\mathbb{Z}.$$

Elliptische Kurven über endlichen Körpern

Hasse-Weil: Gruppenordnung über endlichen Körpern

Seien p eine Primzahl, $q = p^n$ und E eine elliptische Kurve über dem endlichen Körper $\mathbb{F}_q$.

Elliptische Kurven über endlichen Körpern

Hasse-Weil: Gruppenordnung über endlichen Körpern

Seien p eine Primzahl, $q = p^n$ und E eine elliptische Kurve über dem endlichen Körper $\mathbb{F}_q$. Ist $a := q + 1 - |E(\mathbb{F}_q)|$ und sind α und β die Wurzeln des ganzzahligen Polynoms $X^2 + aX + q$, so gilt

$$|E(\mathbb{F}_{q^m})| = q^m + 1 - (\alpha^m + \beta^m), \quad \text{für alle } m \in \mathbb{N}.$$

Elliptische Kurven über endlichen Körpern

Beispiel

Für die elliptische Kurve $E : y^2 + xy = x^3 + 1$ über $\mathbb{F}_2$ gilt

$$E(\mathbb{F}_2) = \{\infty, (0,1), (1,0), (1,1)\}.$$

Elliptische Kurven über endlichen Körpern

Beispiel

Für die elliptische Kurve $E : y^2 + xy = x^3 + 1$ über $\mathbb{F}_2$ gilt

$$E(\mathbb{F}_2) = \{\infty, (0,1), (1,0), (1,1)\}.$$

Daher ist

$$E(\mathbb{F}_{2^{101}}) = 2^{101} + 1 - \left[\left(\frac{-1+\sqrt{-7}}{2}\right)^{101} + \left(\frac{-1-\sqrt{-7}}{2}\right)^{101}\right]$$

Elliptische Kurven über endlichen Körpern

Beispiel

Für die elliptische Kurve $E : y^2 + xy = x^3 + 1$ über $\mathbb{F}_2$ gilt

$$E(\mathbb{F}_2) = \{\infty, (0,1), (1,0), (1,1)\}.$$

Daher ist

$$\begin{aligned} E(\mathbb{F}_{2^{101}}) &= 2^{101} + 1 - \left[\left(\frac{-1+\sqrt{-7}}{2} \right)^{101} + \left(\frac{-1-\sqrt{-7}}{2} \right)^{101} \right] \\ &= 2^{101} + 1 - 2969292210605269 \end{aligned}$$

Elliptische Kurven über endlichen Körpern

Beispiel

Für die elliptische Kurve $E : y^2 + xy = x^3 + 1$ über $\mathbb{F}_2$ gilt

$$E(\mathbb{F}_2) = \{\infty, (0,1), (1,0), (1,1)\}.$$

Daher ist

$$\begin{aligned} E(\mathbb{F}_{2^{101}}) &= 2^{101} + 1 - \left[\left(\frac{-1+\sqrt{-7}}{2}\right)^{101} + \left(\frac{-1-\sqrt{-7}}{2}\right)^{101}\right] \\ &= 2^{101} + 1 - 2969292210605269 \\ &= 2535301200456455833701195805484. \end{aligned}$$

Elliptische Kurven und Kryptographie

Elliptic Curve Cryptography (ECC)

Elliptische Kurven $C = E$ über $\mathbb{F}_q$ bieten eine Reihe von Vorteilen gegenüber der multiplikativen Gruppe $C = \mathbb{F}_q^*$:

Elliptische Kurven und Kryptographie

Elliptic Curve Cryptography (ECC)

Elliptische Kurven $C = E$ über $\mathbb{F}_q$ bieten eine Reihe von Vorteilen gegenüber der multiplikativen Gruppe $C = \mathbb{F}_q^*$:

1. Elliptische Kurven kommen in Familien.

Elliptische Kurven und Kryptographie

Elliptic Curve Cryptography (ECC)

Elliptische Kurven $C = E$ über $\mathbb{F}_q$ bieten eine Reihe von Vorteilen gegenüber der multiplikativen Gruppe $C = \mathbb{F}_q^*$:

1. Elliptische Kurven kommen in Familien.
2. Das diskrete Logarithmus-Problem ist in $\mathbb{F}_q^*$ mit Hilfe des „Index-Kalküls“ effizient angreifbar.

Elliptische Kurven und Kryptographie

Elliptic Curve Cryptography (ECC)

Elliptische Kurven $C = E$ über $\mathbb{F}_q$ bieten eine Reihe von Vorteilen gegenüber der multiplikativen Gruppe $C = \mathbb{F}_q^*$:

1. Elliptische Kurven kommen in Familien.
2. Das diskrete Logarithmus-Problem ist in $\mathbb{F}_q^*$ mit Hilfe des „Index-Kalküls“ effizient angreifbar.
3. Methoden zum Lösen des diskreten Logarithmus-Problems in elliptischen Kurven scheinen ihre Effizienz bei hoher Gruppenordnung sehr schnell zu verlieren.

Elliptische Kurven und Kryptographie

Elliptic Curve Cryptography (ECC)

Elliptische Kurven $C = E$ über $\mathbb{F}_q$ bieten eine Reihe von Vorteilen gegenüber der multiplikativen Gruppe $C = \mathbb{F}_q^*$:

1. Elliptische Kurven kommen in Familien.
2. Das diskrete Logarithmus-Problem ist in $\mathbb{F}_q^*$ mit Hilfe des „Index-Kalküls“ effizient angreifbar.
3. Methoden zum Lösen des diskreten Logarithmus-Problems in elliptischen Kurven scheinen ihre Effizienz bei hoher Gruppenordnung sehr schnell zu verlieren.

Bei den DL-PKCen empfiehlt die NSA den amerikanischen Unternehmen den schnellen Umstieg auf elliptische Kurven!

Die britischen Geheimdienstler

Obwohl der britische Geheimdienst-Mitarbeiter James H. Ellis seine revolutionäre Idee eines Public-Key-Systems mit Hilfe einer Einwegfunktion nicht umsetzen konnte, hatte seine Idee zwei seiner Kollegen am GCHQ stark inspiriert

Die britischen Geheimdienstler

Obwohl der britische Geheimdienst-Mitarbeiter James H. Ellis seine revolutionäre Idee eines Public-Key-Systems mit Hilfe einer Einwegfunktion nicht umsetzen konnte, hatte seine Idee zwei seiner Kollegen am GCHQ stark inspiriert:

- Clifford Cocks hat bereits 1973 das RSA-Kryptosystem entdeckt.

Die britischen Geheimdienstler

Obwohl der britische Geheimdienst-Mitarbeiter James H. Ellis seine revolutionäre Idee eines Public-Key-Systems mit Hilfe einer Einwegfunktion nicht umsetzen konnte, hatte seine Idee zwei seiner Kollegen am GCHQ stark inspiriert:

- Clifford Cocks hat bereits 1973 das RSA-Kryptosystem entdeckt.
- Malcolm J. Williamson hat bereits 1974 das Diffie-Hellman Schlüsselaustausch-Verfahren entdeckt.

Die britischen Geheimdienstler

Diffie hat Ellis bereits in den '70ern besucht

Die britischen Geheimdienstler

Diffie hat Ellis bereits in den '70ern besucht:

Diffie fragte ihn:
"Tell me how you invented public-key cryptography."

Die britischen Geheimdienstler

Diffie hat Ellis bereits in den '70ern besucht:

Diffie fragte ihn:
"Tell me how you invented public-key cryptography."
Nach einer langen Pause antwortete Ellis:

Die britischen Geheimdienstler

Diffie hat Ellis bereits in den '70ern besucht:

Diffie fragte ihn:
"Tell me how you invented public-key cryptography."
Nach einer langen Pause antwortete Ellis:
"Well, I don't know how much I should say. Let me just say that you people made much more of it than we did."

Die britischen Geheimdienstler

Diffie hat Ellis bereits in den '70ern besucht:

Diffie fragte ihn:
"Tell me how you invented public-key cryptography."
Nach einer langen Pause antwortete Ellis:
"Well, I don't know how much I should say. Let me just say that you people made much more of it than we did."

Am 18. Dezember 1997 hat Clifford Cocks einen öffentlichen Vortrag über die bislang unter Verschluß gebliebenen Beiträge der GCHQ-Mitarbeiter zur asymmetrischen Kryptographie gehalten.

Die britischen Geheimdienstler

Diffie hat Ellis bereits in den '70ern besucht:

Diffie fragte ihn:
"Tell me how you invented public-key cryptography."
Nach einer langen Pause antwortete Ellis:
"Well, I don't know how much I should say. Let me just say that you people made much more of it than we did."

Am 18. Dezember 1997 hat Clifford Cocks einen öffentlichen Vortrag über die bislang unter Verschluß gebliebenen Beiträge der GCHQ-Mitarbeiter zur asymmetrischen Kryptographie gehalten. James H. Ellis starb einen Monat zuvor, am 25. November 1997.

Der Faktor „Mensch“

Der Faktor „Mensch“

Die sogenannten Seitenkanal-Attacken, bedingt durch

Der Faktor „Mensch“

Die sogenannten Seitenkanal-Attacken, bedingt durch

1. **Implementationsfehler**

Der Faktor „Mensch“

Die sogenannten Seitenkanal-Attacken, bedingt durch

1. **Implementationsfehler**

1. http://digitaloffense.net/tools/debian-openssl/:
   „[...] All SSL and SSH keys generated on Debian-based systems (Ubuntu, Kubuntu, etc) between September 2006 and May 13th, 2008 may be affected. [...] When creating a new OpenSSH key, there are only 32,767 possible outcomes for a given architecture, key size, and key type.“

Der Faktor „Mensch“

Die sogenannten Seitenkanal-Attacken, bedingt durch

1. **Implementationsfehler** und
2. **Bedienungsfehler**

1. http://digitaloffense.net/tools/debian-openssl/:
   „[...] All SSL and SSH keys generated on Debian-based systems (Ubuntu, Kubuntu, etc) between September 2006 and May 13th, 2008 may be affected. [...] When creating a new OpenSSH key, there are only 32,767 possible outcomes for a given architecture, key size, and key type.“

Der Faktor „Mensch“

Die sogenannten Seitenkanal-Attacken, bedingt durch

1. **Implementationsfehler** und
2. **Bedienungsfehler**

1. http://digitaloffense.net/tools/debian-openssl/:
   „[...] All SSL and SSH keys generated on Debian-based systems (Ubuntu, Kubuntu, etc) between September 2006 and May 13th, 2008 may be affected. [...] When creating a new OpenSSH key, there are only 32,767 possible outcomes for a given architecture, key size, and key type.“
2. http://apcmag.com/hackers-break-into-pentagons-fighter-jet-project-.htm:
   APCMAG, 28 April 2009, 12:52 PM: „**Hackers break into stealth fighter jet project.** The Pentagon computer hack which gave access to the Joint Strike Fighter Project, could have been prevented. [...]“

Der Faktor „Mensch“

Die sogenannten Seitenkanal-Attacken, bedingt durch

1. **Implementationsfehler** und
2. **Bedienungsfehler**

sind die größten Sicherheitslücken

1. http://digitaloffense.net/tools/debian-openssl/:
   „[...] All SSL and SSH keys generated on Debian-based systems (Ubuntu, Kubuntu, etc) between September 2006 and May 13th, 2008 may be affected. [...] When creating a new OpenSSH key, there are only 32,767 possible outcomes for a given architecture, key size, and key type.“
2. http://apcmag.com/hackers-break-into-pentagons-fighter-jet-project-.htm:
   APCMAG, 28 April 2009, 12:52 PM: „**Hackers break into stealth fighter jet project.** The Pentagon computer hack which gave access to the Joint Strike Fighter Project, could have been prevented. [...]“

Der Faktor „Mensch“

Die sogenannten Seitenkanal-Attacken, bedingt durch

1. **Implementationsfehler** und
2. **Bedienungsfehler**

sind die größten Sicherheitslücken und werden es vermutlich bleiben.

1. http://digitaloffense.net/tools/debian-openssl/:
   „[...] All SSL and SSH keys generated on Debian-based systems (Ubuntu, Kubuntu, etc) between September 2006 and May 13th, 2008 may be affected. [...] When creating a new OpenSSH key, there are only 32,767 possible outcomes for a given architecture, key size, and key type.“
2. http://apcmag.com/hackers-break-into-pentagons-fighter-jet-project-.htm:
   APCMAG, 28 April 2009, 12:52 PM: „**Hackers break into stealth fighter jet project.** The Pentagon computer hack which gave access to the Joint Strike Fighter Project, could have been prevented. [...]“