TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik | Institut Systemarchitektur | Professur Datenschutz und Datensicherheit

WS 2020/2021

Betriebssysteme und Sicherheit

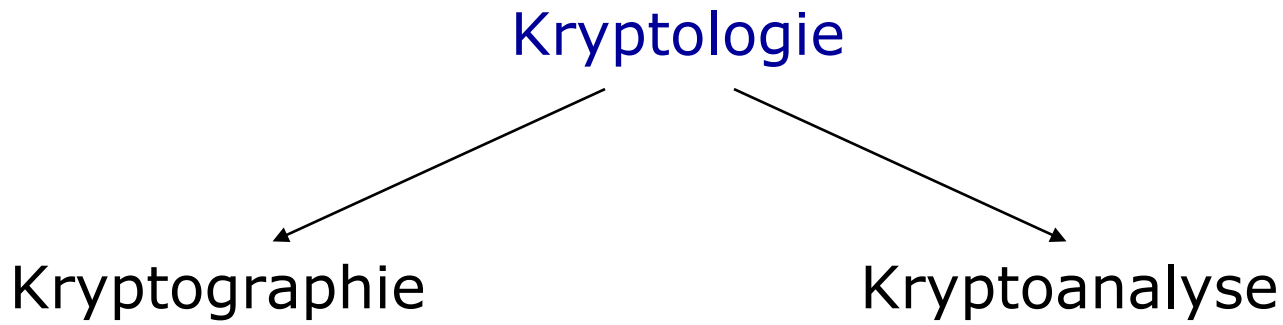
Einführung in Datenschutz und Datensicherheit

Kryptographische Grundlagen

Folien von: Elke Franz

Elke.Franz@tu-dresden.de

5 Kryptographische Grundlagen – Begriffe



Kryptographie (griech. „kryptos“+ „graphein“)

Wissenschaft von den Methoden der Ver- und Entschlüsselung von Informationen.

Kryptoanalyse (griech. „kryptos“+ „analyein“)

Wissenschaft vom Entschlüsseln von Nachrichten ohne Kenntnis dazu notwendiger geheimer Informationen.

Historische Verfahren

- Transpositionen
  Verwürfeln der Klartextzeichen, Permutation der Stellen des Klartextes (Permutationschiffren)

Beispiel: Skytala (Matrixtransposition)

transpositionschiffre →

t	r	a	n	s
p	o	s	i	t
i	o	n	s	c
h	i	f	f	r
e	x	y	z	x

↓

TPIHEROOIXASNFYNISFZSTCRX

5 Kryptographische Grundlagen – Historische Verfahren

- MM-Substitutionen (monoalphabetisch, monographisch)
  Ersetzen von Zeichen (monographisch: einzelner Zeichen)

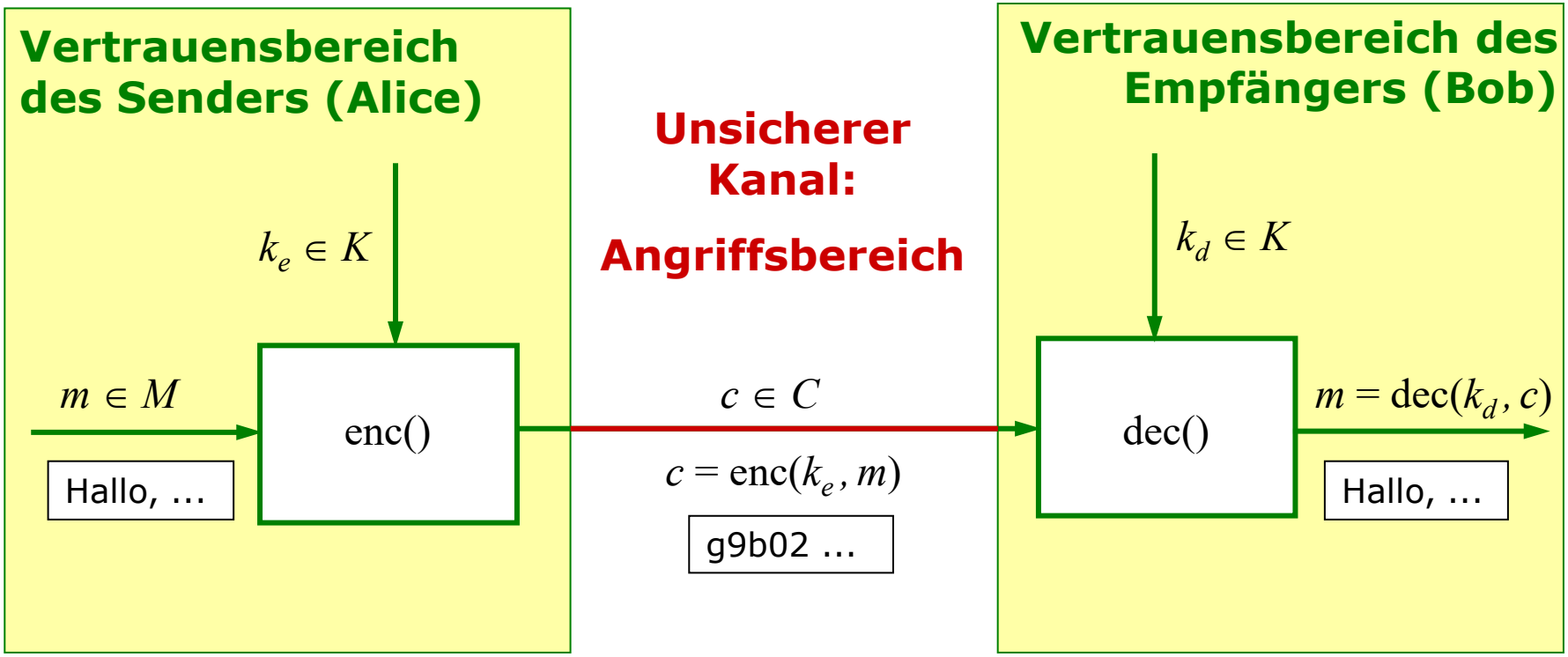
  Beispiel: Cäsarchiffre (Verschiebechiffre/Additive Chiffre)

Nachricht	a	b	c	d	e	f	g		…		x	y	z
Schlüsseltext	D	E	F	G	H	I	J		…		A	B	C

b e i s p i e l ⟶ E H L V S L H O

- PM-Substitutionen (polyalphabetisch, monographisch)
  Beispiel: Vigenère-Chiffre

5 Kryptographische Grundlagen – Systemtypen



m	Nachricht (*message*)
c	Schlüsseltext (*ciphertext*)
enc	Verschlüsselung (*encryption*)
dec	Entschlüsselung (*decryption*)

k	Schlüssel (*key*)
k_e	… zur Verschlüsselung (*encryption key*)
k_d	… zur Entschlüsselung (*decryption key*)
K/*M*/*C*	endliche Menge möglicher Schlüssel/ Nachrichten/Schlüsseltexte

Kerckhoffs-Prinzip

Die Sicherheit eines Verfahrens darf nicht von der Geheimhaltung des Verfahrens abhängen, sondern nur von der Geheimhaltung des Schlüssels.

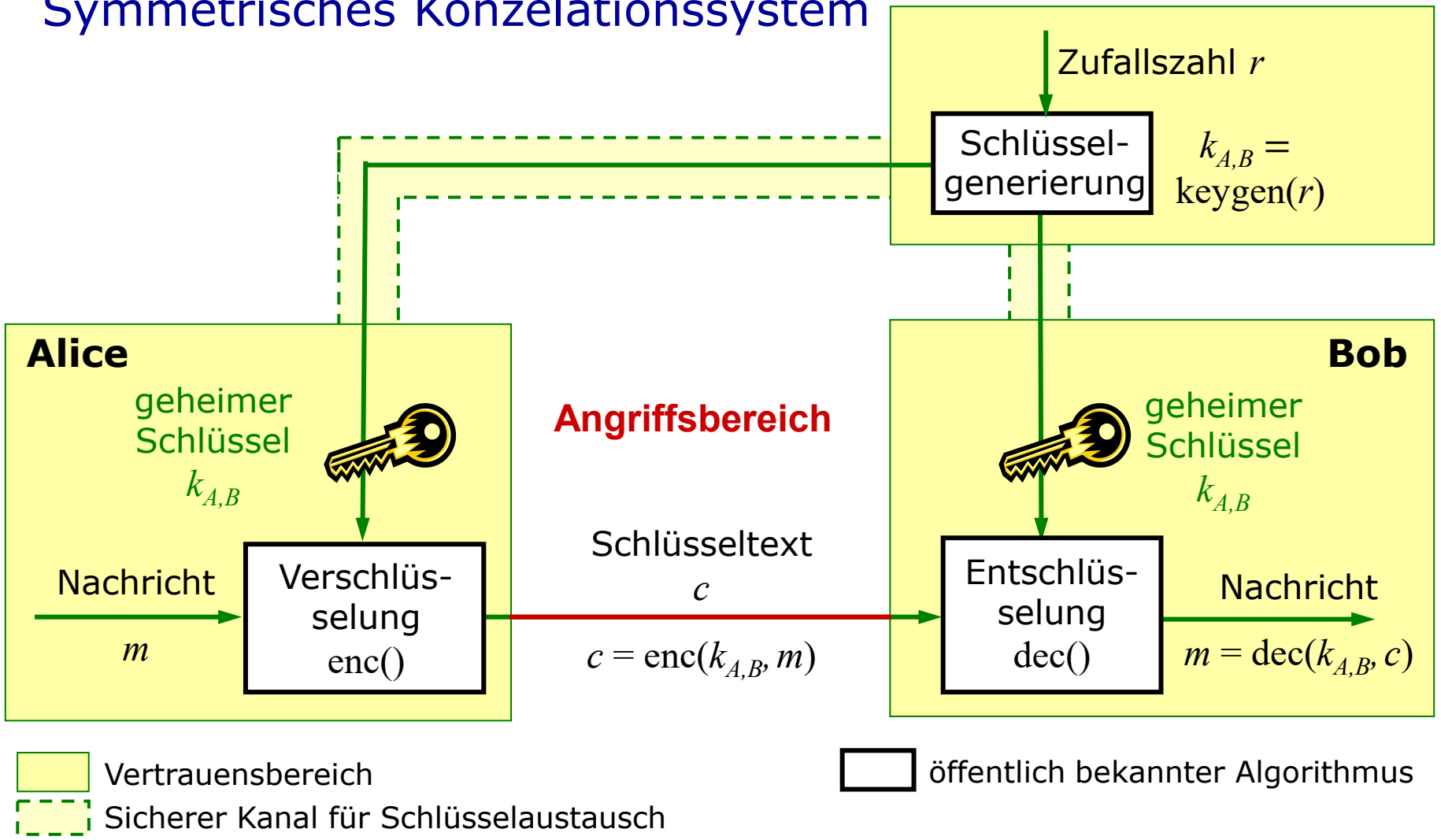
[Auguste Kerkhoffs: *La Cryptographie militaire.* Journal des Sciences Militaires, Januar 1883.]

- Keine „Security by Obscurity"
- Annahme: Angreifer kennt das Verfahren und die öffentlichen Parameter
- Sicherheit des Verfahrens begrenzt durch
  - Sicherheit der Schlüsselgenerierung und
  - Sicherheit des Schlüsselaustauschs

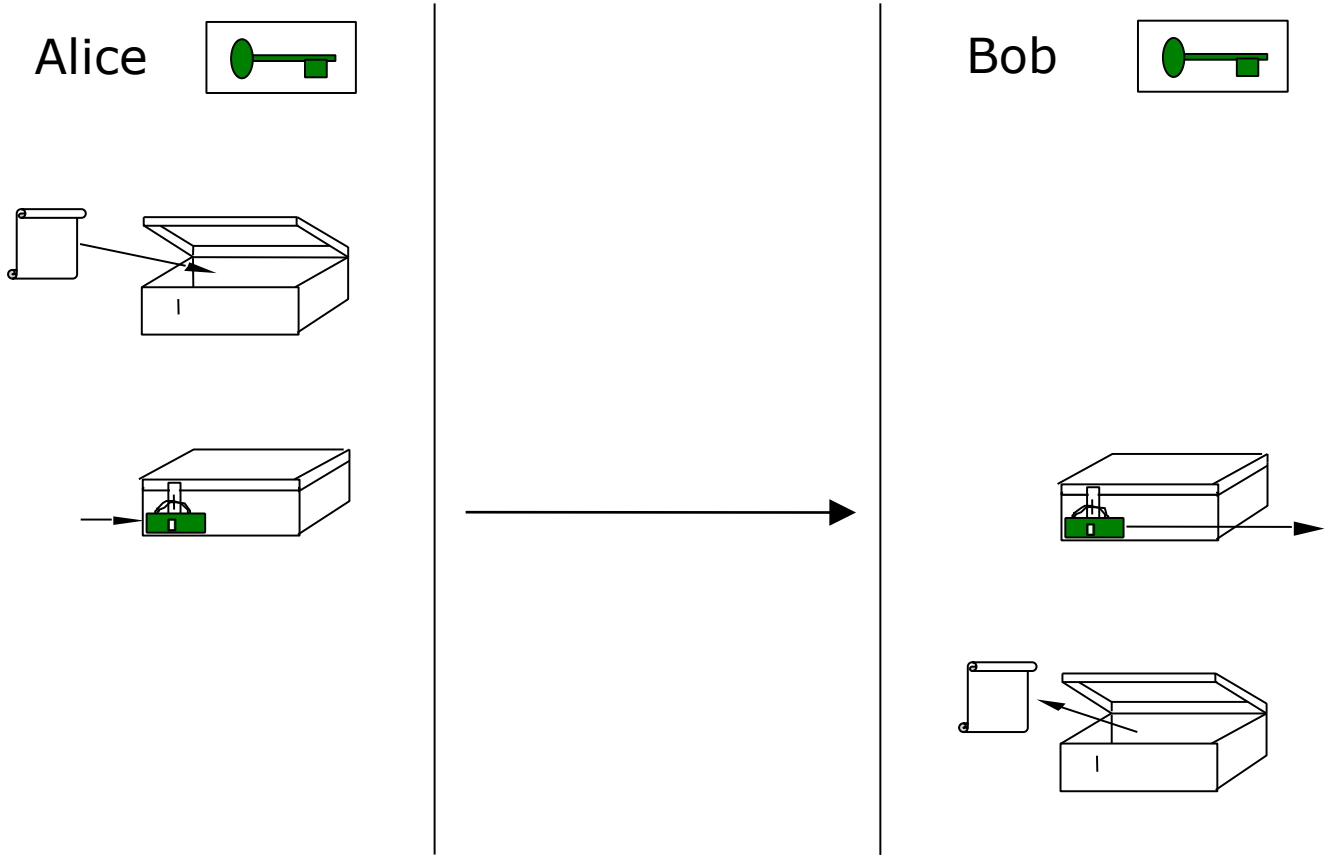
Kriterien für eine Einteilung

- Zweck
  - Konzelationssysteme
    Systeme zum Schutz der Vertraulichkeit der Daten
  - Authentikationssysteme
    Systeme zum Schutz der Integrität der Daten
    - digitale Signatursysteme (spezielle Authentikationssysteme)
      Systeme zur Realisierung von Zurechenbarkeit von Daten
- Schlüsselverteilung
  - Symmetrische Verfahren
    Sender und Empfänger arbeiten mit dem gleichen Schlüssel;
    ein Schlüssel pro Kommunikationsbeziehung
  - Asymmetrische Verfahren
    jeweils ein Schlüsselpaar pro Teilnehmer:
    öffentlicher und privater Schlüssel

Symmetrisches Konzelationssystem



Symmetrisches Konzelationssystem

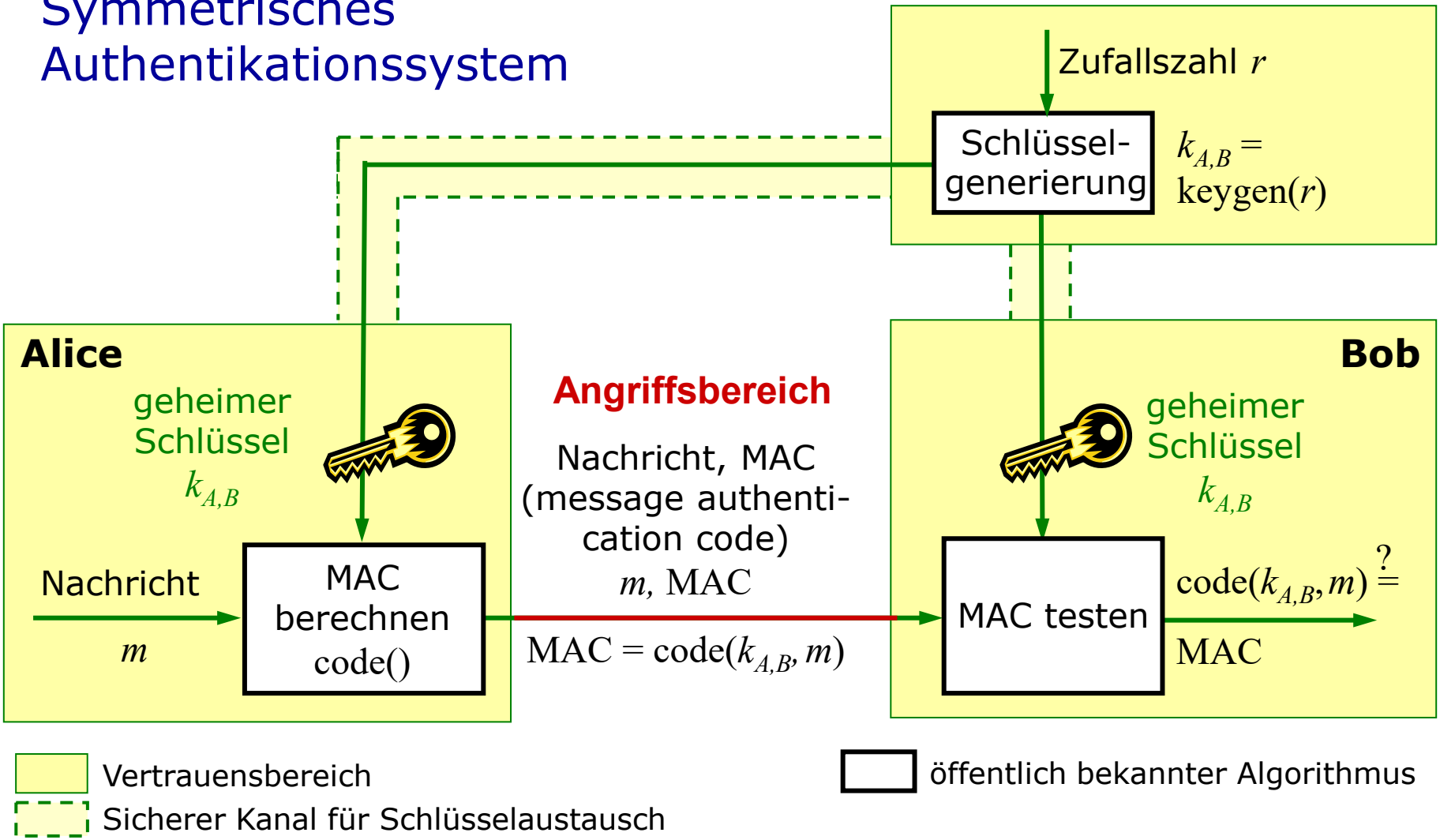


- Schlüsselaustausch
  - Notwendig: sicherer Kanal für Schlüsselaustausch
  - Offenes System: Sender und Empfänger können sich nicht vorab treffen

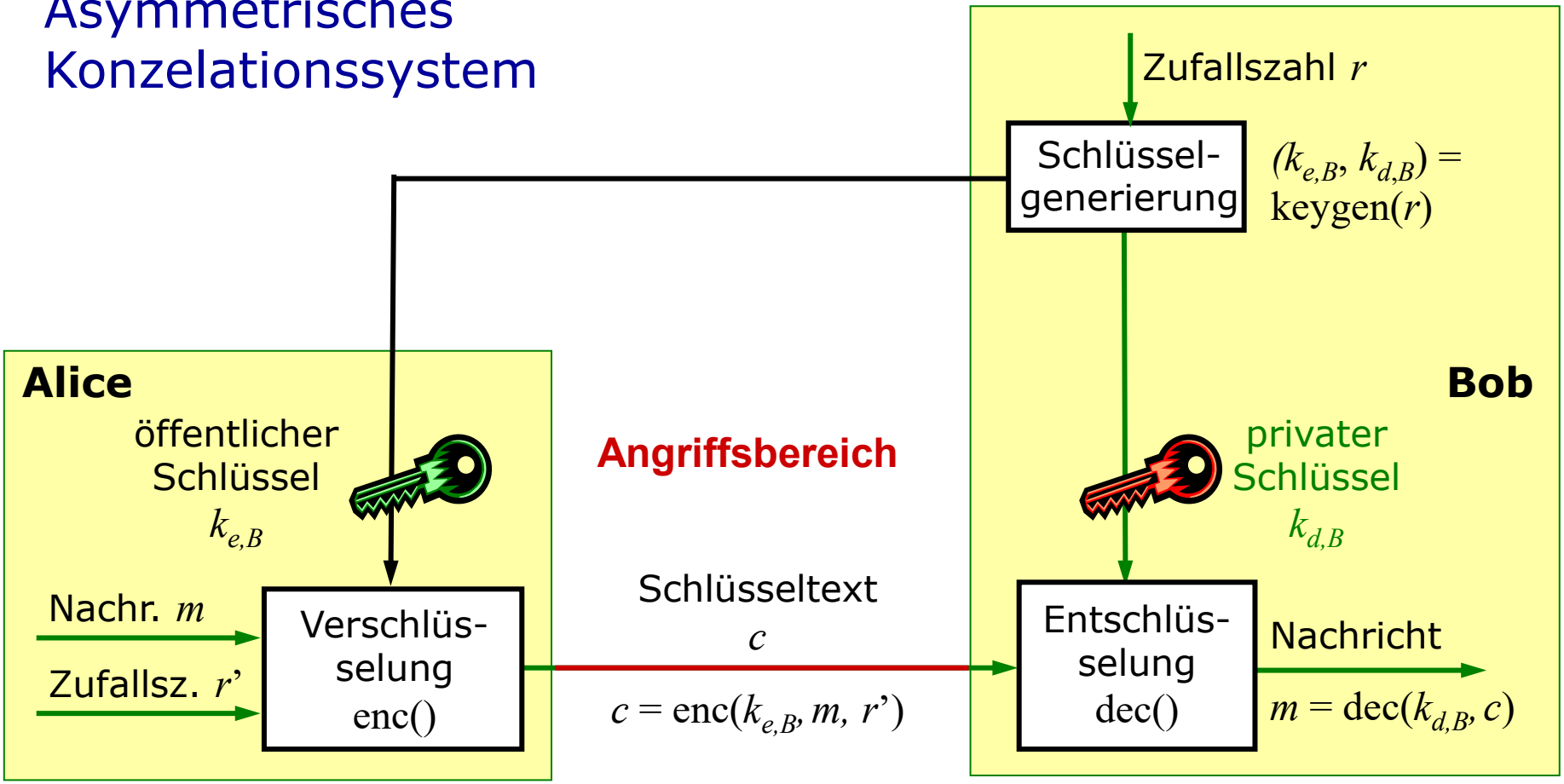
  → Lösung: Schlüsselverteilzentrale X
    - Jeder Teilnehmer (z.B. A) meldet sich an und tauscht einen geheimen Schlüssel $k_{A,X}$ mit X aus
    - Kommunikation mit Teilnehmer B: Anfrage an X nach geheimem Schlüssel $k_{A,B}$
    - X generiert Schlüssel $k_{A,B}$ und sendet ihn an A und B

  - Problem: X kann alle Nachrichten lesen
  - Verbesserung: verschiedene Schlüsselverteilzentralen verwenden und geheime Schlüssel lokal berechnen

Symmetrisches Authentikationssystem



Asymmetrisches Konzelationssystem



Vertrauensbereich

öffentlich bekannter Algorithmus

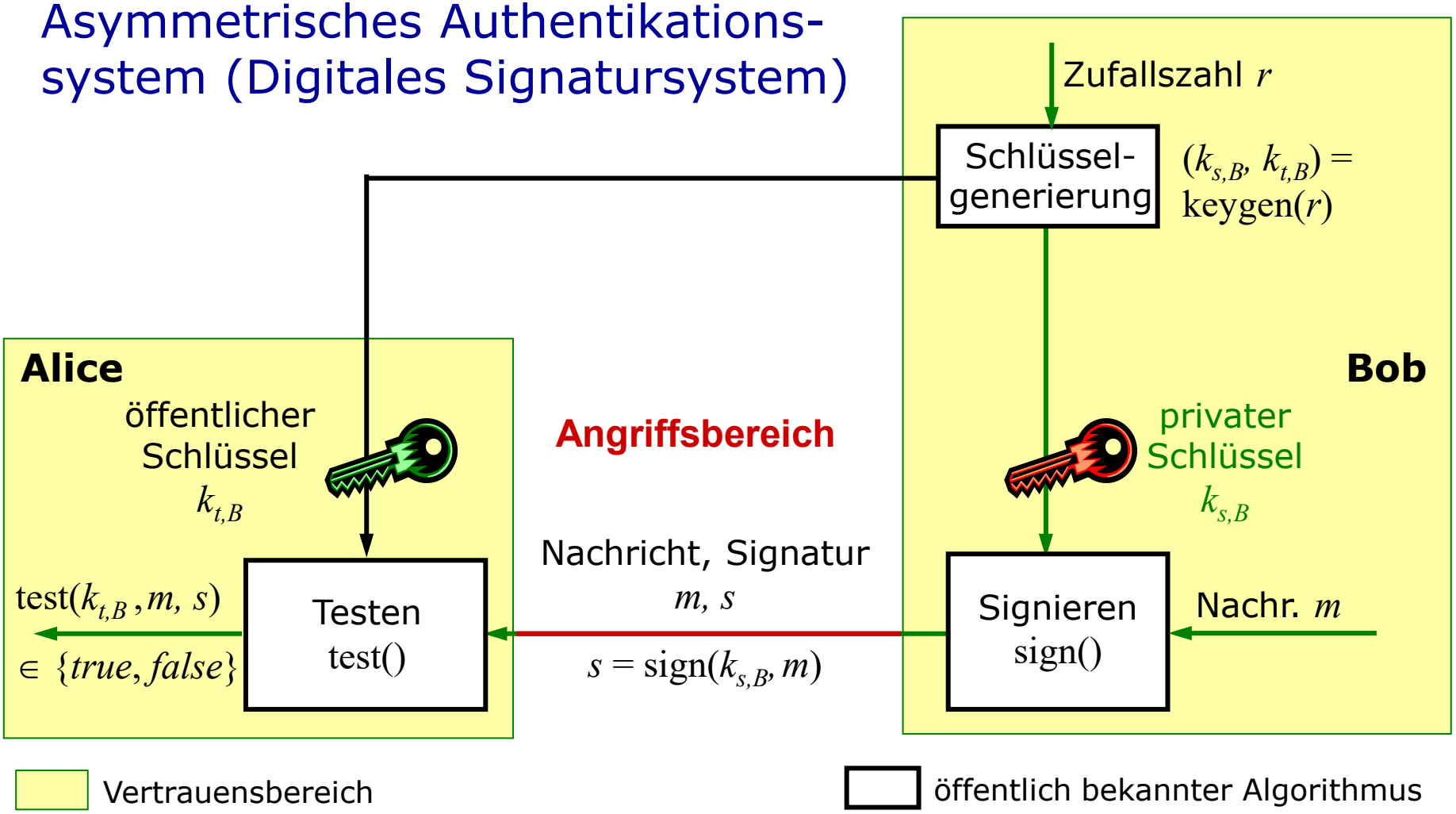
Zufallszahl r`: probabilistische bzw. indeterministische Verschlüsselung

- Schlüsselaustausch
  - Jeder Teilnehmer generiert eigenes Schlüsselpaar – kein (gegen Abhören) sicherer Kanal für Schlüsselaustausch notwendig
  - Verteilung der öffentlichen Schlüssel: veröffentlichen

  → Andere Möglichkeit: Öffentliches Schlüsselregister R
    - Jeder Teilnehmer (z.B. A) trägt seinen öffentlichen Schlüsel ein ($k_{e,A}$)
    - Teilnehmer B will mit A kommunizieren: bittet R um öffentlichen Schlüssel $k_{e,A}$ von A
    - B erhält $k_{e,A}$, beglaubigt durch die Signatur von R; R beglaubigt Zusammenhang zwischen A und $k_{e,A}$

  - Problem: einzelnes Register R hat Möglichkeit für aktiven Angriff
  - Verbesserung: verschiedene Register verwenden

Asymmetrisches Authentikations-system (Digitales Signatursystem)



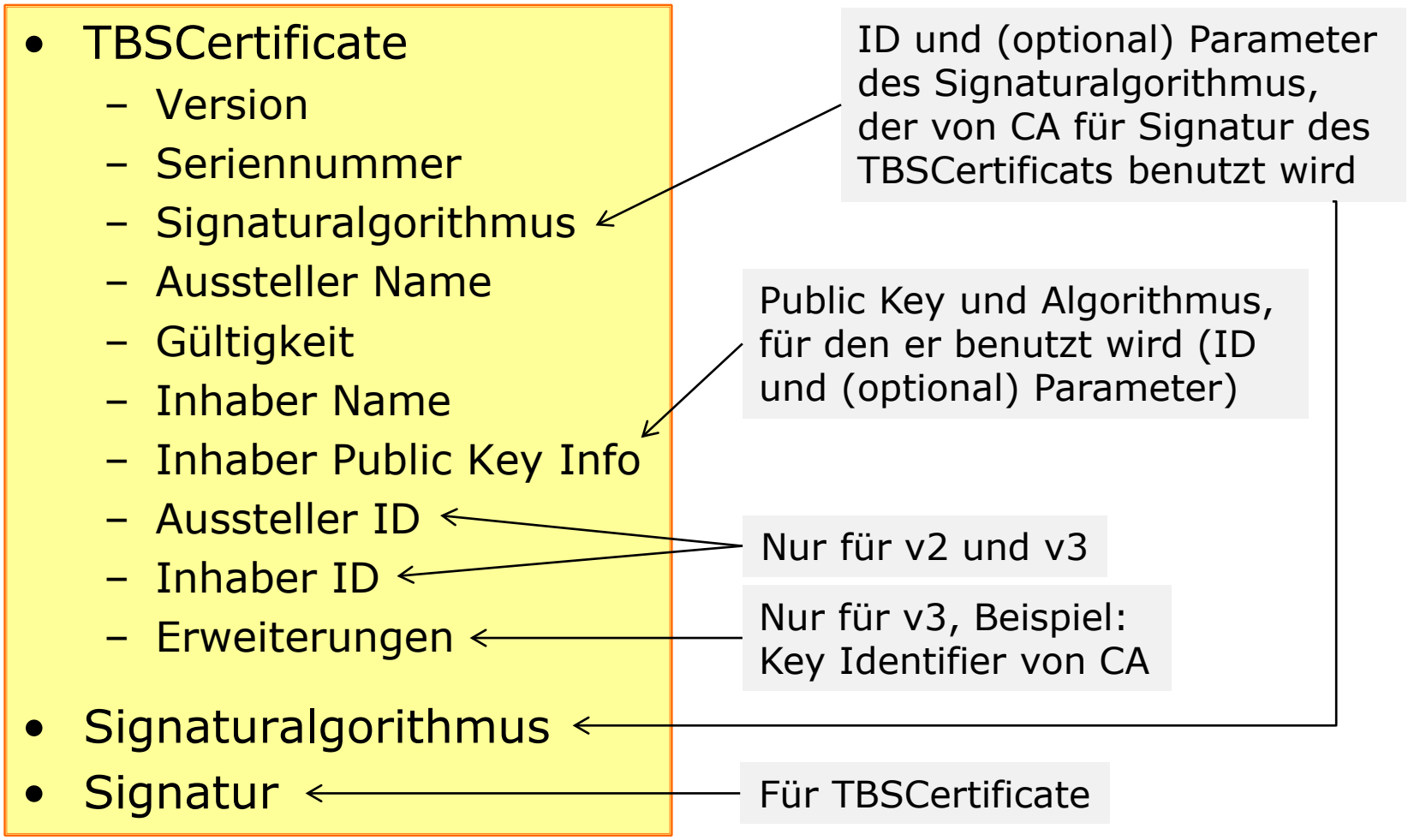
- Schlüsselaustausch
  - Notwendig für Zurechenbarkeit:

    Bestätigung der Zuordnung des öffentlichen Testschlüssels zum jeweiligen Teilnehmer mittels Schlüsselzertifikat, ausgestellt von Zertifizierungsinstanz (*certification authority CA*)

  - Verbreitetes Format für Schlüsselzertifikat: X.509
    - Erstmals 1988 veröffentlicht, aktuell in Version 3 (X.509v3)
    - Standard der ITU-T für Public-Key Infrastructure:

      http://www.itu.int/rec/T-REC-X.509
    - RFC 5280
    - Sperrlisten für Zertifikate

- Aufbau eines X.509-Zertifikats [RFC 5280]



5 Kryptographische Grundlagen – Systemtypen

	Symmetrische Authentikationssysteme (MAC)	Asymmetrische Authentikationssysteme (Digitale Signatursysteme)
Schlüssel	Geheimer Schlüssel, einem Paar von Kommunikationspartnern zugeordnet	Schlüsselpaar: privater Signaturschlüssel und öffentlicher Testschlüssel
Prüfung	MAC für empfangene Daten berechnen und mit empfangenem MAC vergleichen	Testalgorithmus erforderlich
Schutzziele	Integrität	Integrität Zurechenbarkeit

Anmerkungen zum Schlüsselaustausch

- Wem werden Schlüssel zugeordnet?
  - einzelnen Teilnehmern — asymmetrische Systeme
  - Paarbeziehungen — symmetrische Systeme
  - Gruppen — ---
- Wie viele Schlüssel werden benötigt?

  n Teilnehmer

  asymmetrische Systeme — je System *n* Schlüssel

  symmetrische Systeme — $n(n-1)/2$
- Wann Schlüssel generieren und austauschen?
- Sicherheit der Schlüsselgenerierung und des Schlüsselaustausch begrenzt die mit dem Kryptosystem erreichbare Sicherheit

  → mehrere Ur-Schlüsselaustausche durchführen

  → Schlüsselgenerierung abhängig von Zufallszahl (XOR verschiedener Zufallszahlen: Ergebnis zufällig und geheim, falls eine ZZ zufällig und geheim)

Was kann mit Kryptographie erreicht werden?

- Konzelationssysteme: Vertraulichkeit
  - Verschlüsselung der Klartexte mit einem geheimen oder privaten Schlüssel, d.h. Transformation der Klartexte in zufällig aussehenden Schlüsseltext
  - Angreifer darf nichts über Klartext (oder Schlüssel) herausfinden

  → Verlust der Vertraulichkeit wird verhindert

- Authentikationssysteme: Integrität / Zurechenbarkeit
  - Berechnung eines „Tags" (MAC oder digitale Signatur) in Abhängigkeit von des Daten und dem geheimen oder privaten Schlüssel
  - Angreifer darf nicht in der Lage sein, ein gültiges Tag für modifizierte oder selbst erzeugte Daten zu berechnen

  → Erkennbarkeit von Modifikationen wird gewährleistet

5 Kryptographische Grundlagen – Systemtypen

Symmetrische Systeme

- ☹ Sicherer Kanal für Schlüsselaustausch notwendig
- ☺ Performance symmetrischer Systeme sehr gut

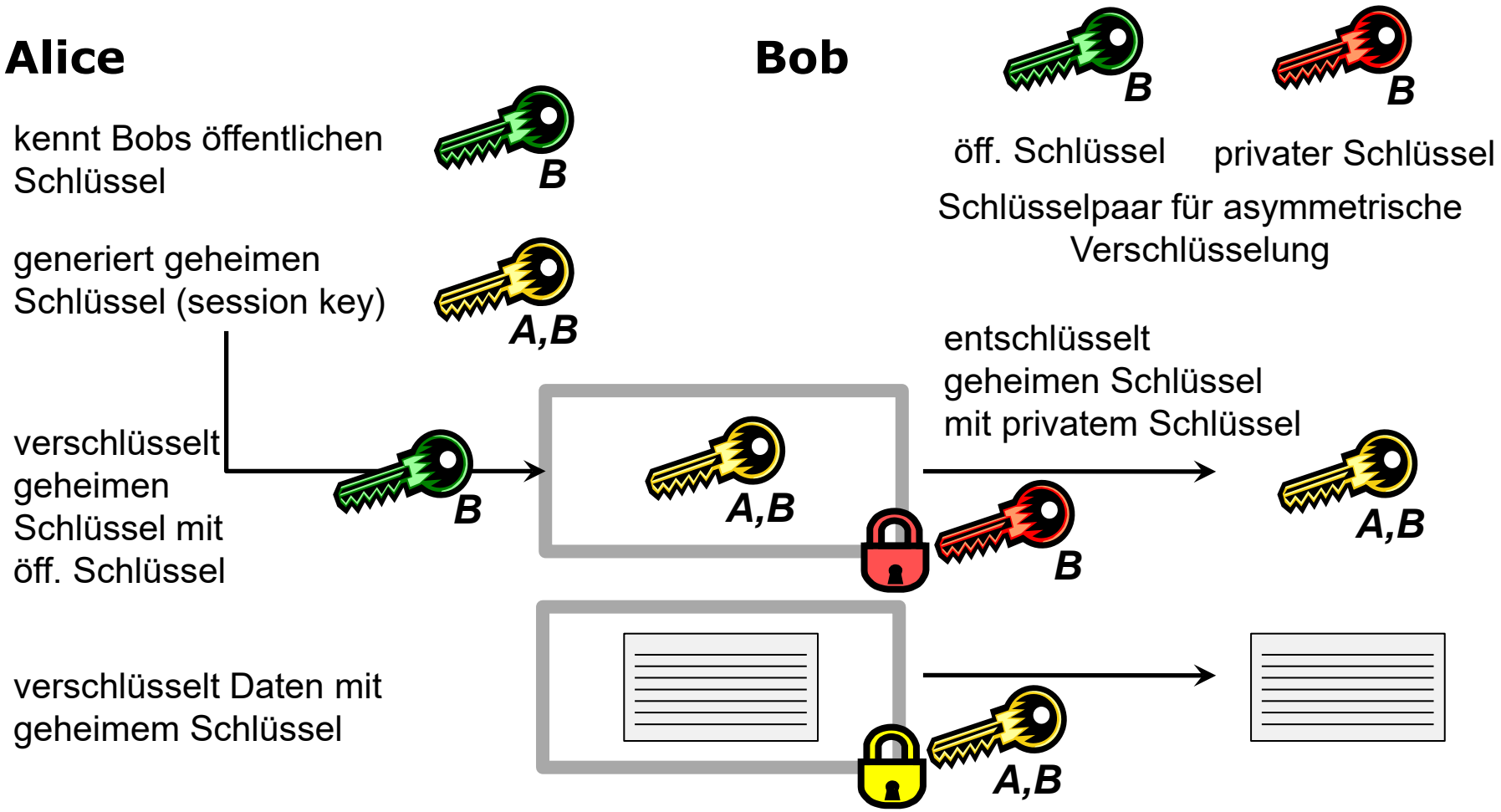
Asymmetrische Systeme

- ☺ Kein sicherer Kanal notwendig („nur" Zuordnung öffentlicher Schlüssel)
- ☹ Langsamer aufgrund komplexerer Operationen

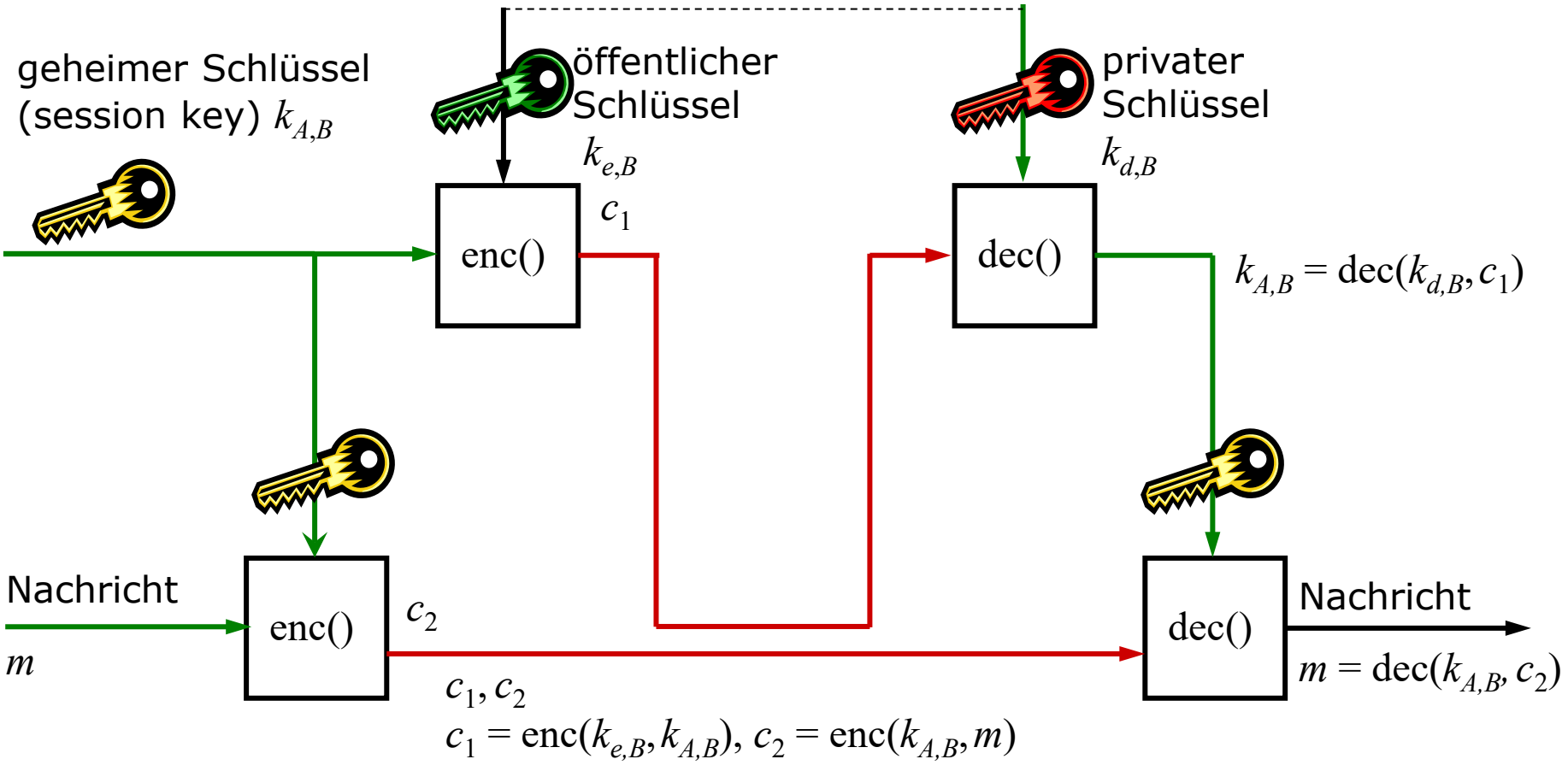
Hybride Systems: Kombination asymmetrischer Systeme (→ Schlüsselaustausch) und symmetrischer Systeme (→ bessere Performance)

Beispiel: Transport Layer Security (TLS)

Hybrides Konzelationssystem



Hybrides Konzelationssystem



Ziel und Erfolg von Angriffen

1. Finden des geheimen Schlüssels
   (vollständiges Brechen, *total break*)

2. Finden eines zum Schlüssel äquivalenten Verfahrens
   (universelles Brechen, *universal break*)

3. Brechen nur für manche Nachrichten
   (nachrichtenbezogenes Brechen, *message-dependent breaking*)
   a) für selbstgewählte Nachricht (*selective break*)
   b) für irgendeine Nachricht (*existential break*)

Konzelationssysteme:
komplette/partielle Entschlüsselung

Authentikationssysteme:
valider MAC/valide digitale Signatur für modifizierte oder neue Nachricht

Klassifizierung von Angriffen (Konzelationssysteme)

- Passive Angriffe
  - Reiner Schlüsseltext-Angriff (ciphertext-only attack)
  - Klartext-Schlüsseltext-Angriff (known-plaintext attack)
- Aktive Angriffe
  - Gewählter Klartext-Schlüsseltext-Angriff (chosen-plaintext attack CPA; „Verschlüsselungsorakel“)
  - Gewählter Schlüsseltext-Klartext-Angriff (chosen-ciphertext attack; „Entschlüsselungsorakel“)

    → CCA1 (auch als „lunchtime“, „lunch-break“ oder „midnight attack“ bezeichnet)
- Weiteres Kriterium: Adaptivität

  → CCA2

Klassifizierung von Kryptosystemen nach ihrer Sicherheit

- informationstheoretisch (perfekt) sicher
  Auch einem unbeschränkten Angreifer gelingt es nicht, das System zu brechen.
  („unconditional security“, „perfect secrecy“)
- beste erreichbare Sicherheit

- Verschiedene Begriffe zur Bewertung der Sicherheit der übrigen Systeme
- Annahmen über Möglichkeiten des Angreifers, Betrachtung der Sicherheit unter bestimmten Angriffen

Informationstheoretische (perfekte) Sicherheit

[Claude Shannon: *Communication Theory of Secrecy Systems.* Bell Systems Technical Journal, 28(1949), 656-715.]

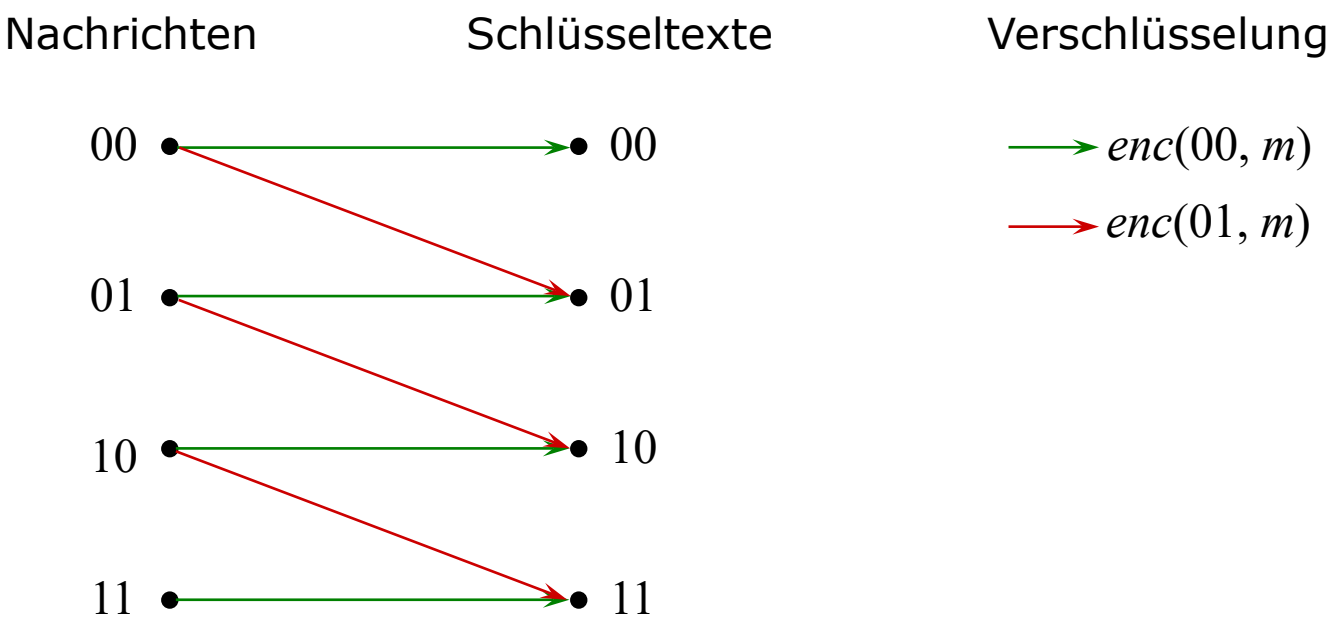
- Informelle Beschreibung (bzgl. Konzelationssystem):

> Selbst ein unbeschränkter Angreifer gewinnt aus seinen Beobachtungen keinerlei zusätzliche Informationen über Klartext oder Schlüssel.

- „unbeschränkt“: beliebiger Rechen- und Zeitaufwand
- „zusätzliche Informationen“: nicht besser als bloßes Raten

→ Nachrichten und Schlüsseltext müssen stochastisch unabhängig sein
→ Anforderungen an Schlüssel abgeleitet

Beispiel für die Anforderungen an die Schlüssel



→ nicht informationstheoretisch sicher
→ Beispiel: Anzahl der Schlüssel

Weitere Anforderungen:
- Schlüssel zufällig
- Schlüssel nur einmal verwenden

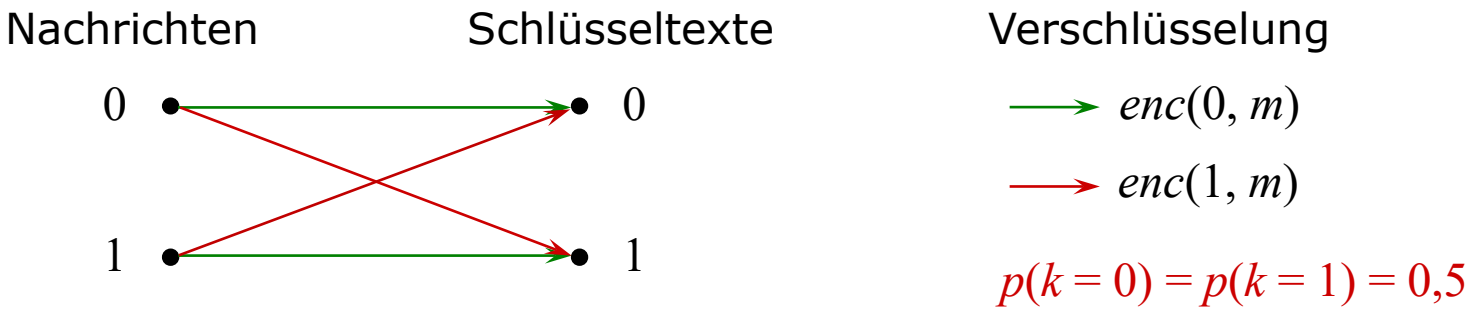
Vernam-Chiffre (one-time pad)

- Jeder Schlüssel wird nur einmal verwendet
- Schlüssellänge und Länge des Klartextes sind gleich
- Schlüssel sind zufällig

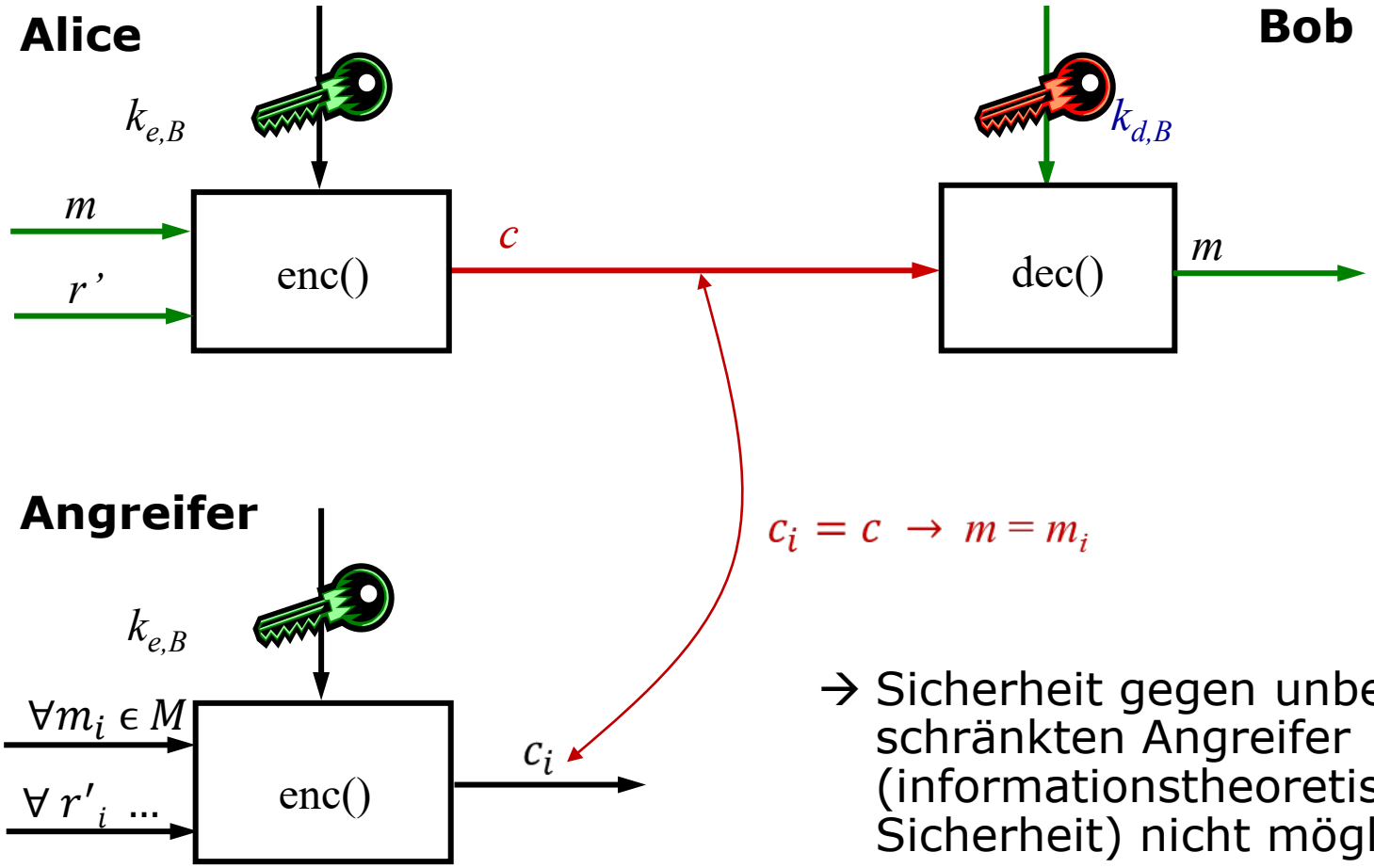
→ Einzige informationstheoretisch sichere Chiffre.

- Binäre Vernam-Chiffre

$$c = \mathrm{enc}(k, m) = m \oplus k \qquad m = \mathrm{dec}(k, c) = c \oplus k$$



Erreichbare Sicherheit asymmetrischer Systeme



→ Sicherheit gegen unbeschränkten Angreifer (informationstheoretische Sicherheit) nicht möglich.

Anmerkungen zur informationstheoretischen Sicherheit

- Aussagen gelten nur für den Algorithmus!
- Betrachtet wurde zunächst passiver Angriff (ciphertext only attack)
- Informationstheoretisch sicheres System ist jedoch auch gegen aktive Angriffe sicher

- Informationstheoretische Sicherheit kann nur von symmetrischen Systemen erbracht werden
  - Schlüsselmanagement problematisch
  - Zurechenbarkeit kann nicht mit symmetrischen Systemen erbracht werden

→ Verwendung von nicht informationstheoretisch sicheren Systemen notwendig

Stromchiffren

- „Verschlüsselung einzelner Zeichen“ unter Nutzung eines Stroms von Schlüsselzeichen
- Beispiele: Vernam-Chiffe, Pseudo-One-Time-Pad, Grain

Blockchiffren

- „Verschlüsselung von Blöcken von Zeichen“ (derselben Länge) unter Nutzung desselben Schlüssels
- Aufteilung längerer Nachrichten in Blöcke dieser Länge

$$m = m_0 m_1 \ldots m_{b\text{-}1}, \quad m_i = m_{i0} m_{i1} \ldots m_{i,l\text{-}1}, \quad m_{ij} \in \{0, 1\}$$

- Auffüllen kürzerer Blöcke (*padding*)
- Beispiele: DES, AES, IDEA
- Produktchiffren (Shannon): Kombination verschiedener Verschlüsselungsoperationen (Feistel-Chiffren, Substitutions-Permutations-Netzwerk)

- Iterierte Blockchiffren
  - Verschlüsselung geschieht in mehreren Runden
  - Anzahl der Runden relevant für Sicherheit
  - Anwendung einer Rundenfunktion f() unter Nutzung des entsprechenden Rundenschlüssels k_i
  - Rundenschlüssel werden aus dem Schlüssel k abgeleitet

  - Verschlüsselung:

    $$c = \mathrm{enc}(k, m) = \mathrm{f}_n(k_n, \mathrm{f}_{n-1}(k_{n-1}, \ldots \mathrm{f}_2(k_2, \mathrm{f}_1(k_1, m)) \ldots))$$

  - Entschlüsselung:

    $$m = \mathrm{dec}(k, c) = \mathrm{f}^{-1}{}_1(k_1, \mathrm{f}^{-1}{}_2(k_2, \ldots \mathrm{f}^{-1}{}_{n-1}(k_{n-1}, \mathrm{f}^{-1}{}_n(k_n, c)) \ldots))$$

  - Anmerkung: Feistel-Chiffre ist selbstinvers, d.h., es wird keine inverse Rundenfunktion benötigt, Ver- und Entschlüsselung arbeiten mit derselben Rundenfunktion f()

AES (Advanced Encryption Standard)

- 1997 Ausschreibung eines öffentlichen Wettbewerbs für die Einreichung eines kryptographischen Algorithmus “AES” als Nachfolger des DES durch das National Institute of Standards and Technology (NIST) der USA

- Sieger des Wettbewerbs:
  Rijndael (Vincent **Rij**men und Joan **Dae**men, Belgien)

- Publikation als Standard im Herbst 2001 (FIPS Standard „Specification for the Advanced Encryption Standard“, FIPS 197)
- 2002 trat AES in Kraft
- Einsatz z.B.: Verschlüsselungsstandard 802.1 für Wireless LAN bzw. für Wi-Fi WPA2, SSH, IPSec, 7-Zip, PGP

Überblick über den Algorithmus

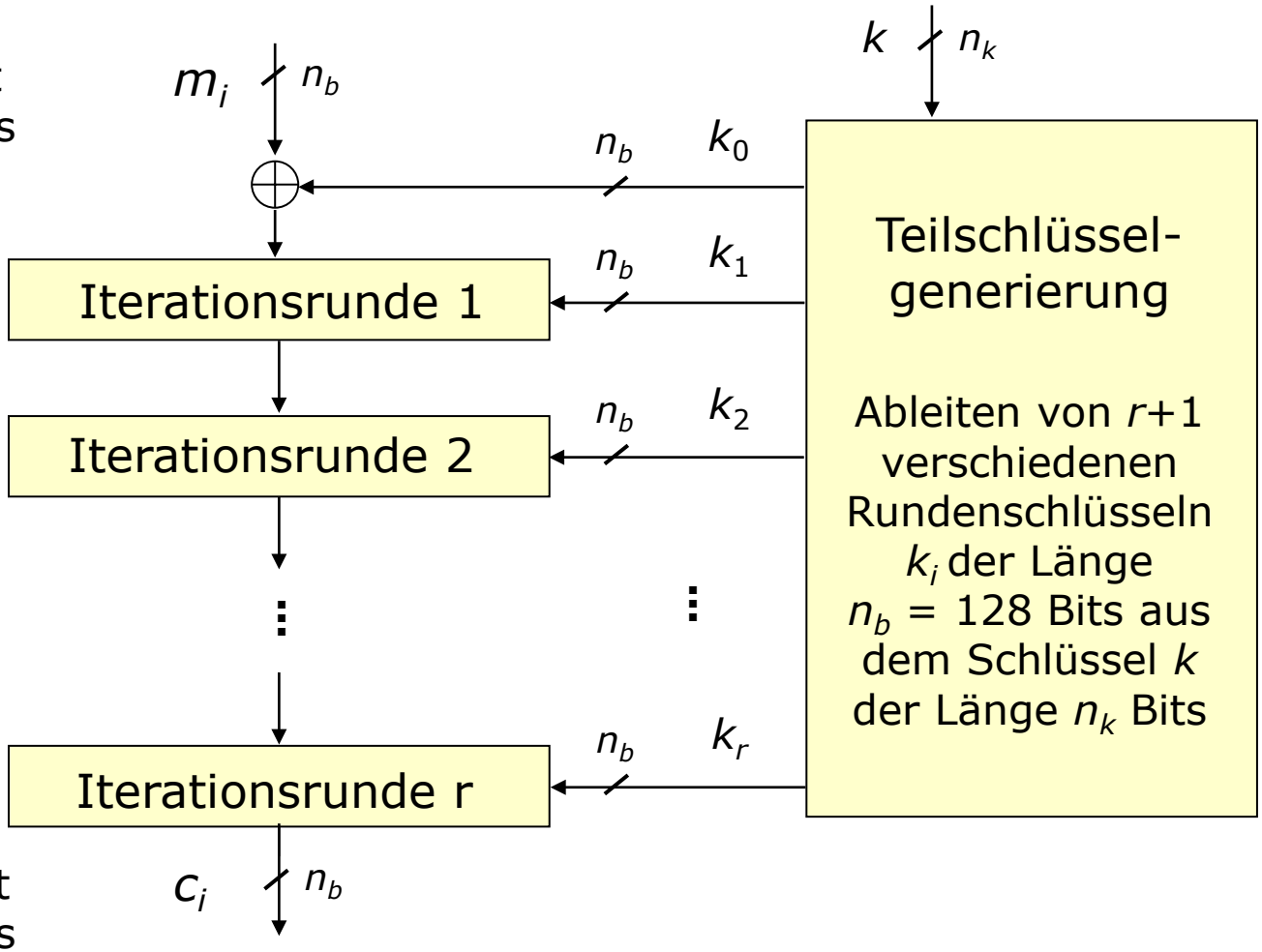
- Verschlüsselung von Klartextblöcken der Länge 128 Bit (vorgeschlagene Längen von 192 und 256 Bits nicht standardisiert)
- Schlüssellänge wahlweise 128, 192 oder 256 Bits
- Mehrere Runden, jeweils Substitutionen, Permutationen und Schlüsseladdition
- Rundenanzahl *r* hängt von Schlüssel- und Klartextlänge ab:

Schlüssel-länge n_k	Blocklänge des Klartextes n_b		
	128 Bit	192 Bit	256 Bit
128 Bit	10	12	14
192 Bit	12	12	14
256 Bit	14	14	14

AES
Rijndael

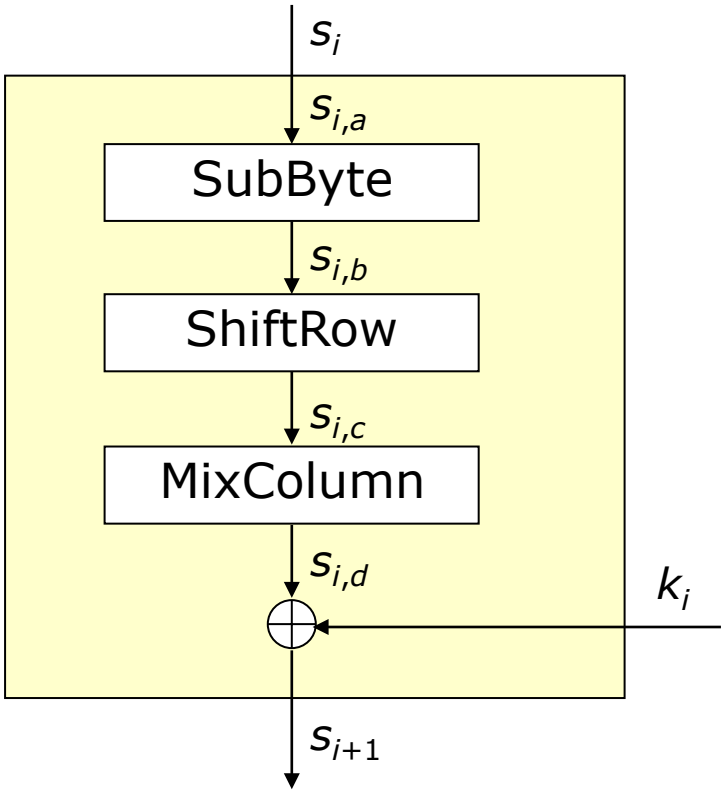
Struktur des AES

Nachrichtenblock m_i mit Blocklänge n_b = 128 Bits

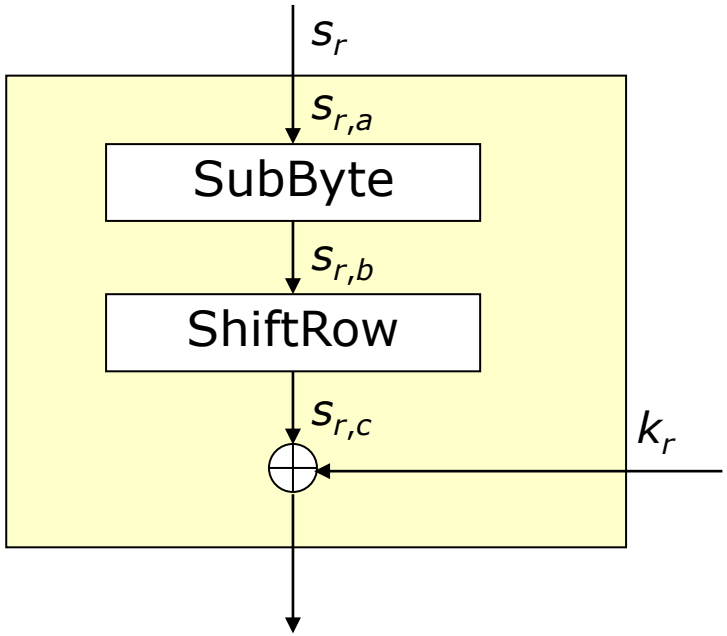


Schlüsseltextblock c_i mit Blocklänge n_b = 128 Bits

Struktur der Iterationsrunden



Runde i, i = 1, 2, …, r-1



Runde r

Notation

- Darstellung eines Bytes als Folge von Bits:

$$a = (a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0)_2, \quad a_i \in \{0,1\}$$

- Darstellung als Polynom:

$$a = \sum_{i=0}^{7} a_i x^i = a_7 x^7 + a_6 x^6 + a_5 x^5 + a_4 x^4 + a_3 x^3 + a_2 x^2 + a_1 x^1 + a_0 x^0$$

- Darstellung als Hexadezimalzahl

Darstellung der Operanden

Byte-Matrizen mit 4 Zeilen und N_b (N_k) Spalten
mit N_b (N_k): Blocklänge n_b (Schlüssellänge n_k) / 32

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	$a_{0,6}$	$a_{0,7}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	$a_{1,6}$	$a_{1,7}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	$a_{2,6}$	$a_{2,7}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$	$a_{3,6}$	$a_{3,7}$

Matrix (state) für Blocklänge

128, 192, 256 Bit

AES

Rijndael

Schlüssel für Schlüssellänge

128, 192, 256 Bit

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	$k_{0,6}$	$k_{0,7}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	$k_{1,6}$	$k_{1,7}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$	$k_{2,4}$	$k_{2,5}$	$k_{2,6}$	$k_{2,7}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$	$k_{3,4}$	$k_{3,5}$	$k_{3,6}$	$k_{3,7}$

Mathematische Grundlagen

- Alle Verschlüsselungsschritte basieren auf Operationen in endlichen Strukturen
- Bytes interpretiert als Elemente des endlichen Körpers GF(2^8)
  - Byte Addition $\oplus$:

    $a = \{a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0\}, b = \{b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0\}$

    $c = a \oplus b$ mit $c_i = a_i \oplus b_i$
  - Byte Multiplikation $\odot$:

    $c = a \odot b = a \cdot b \bmod m(x)$

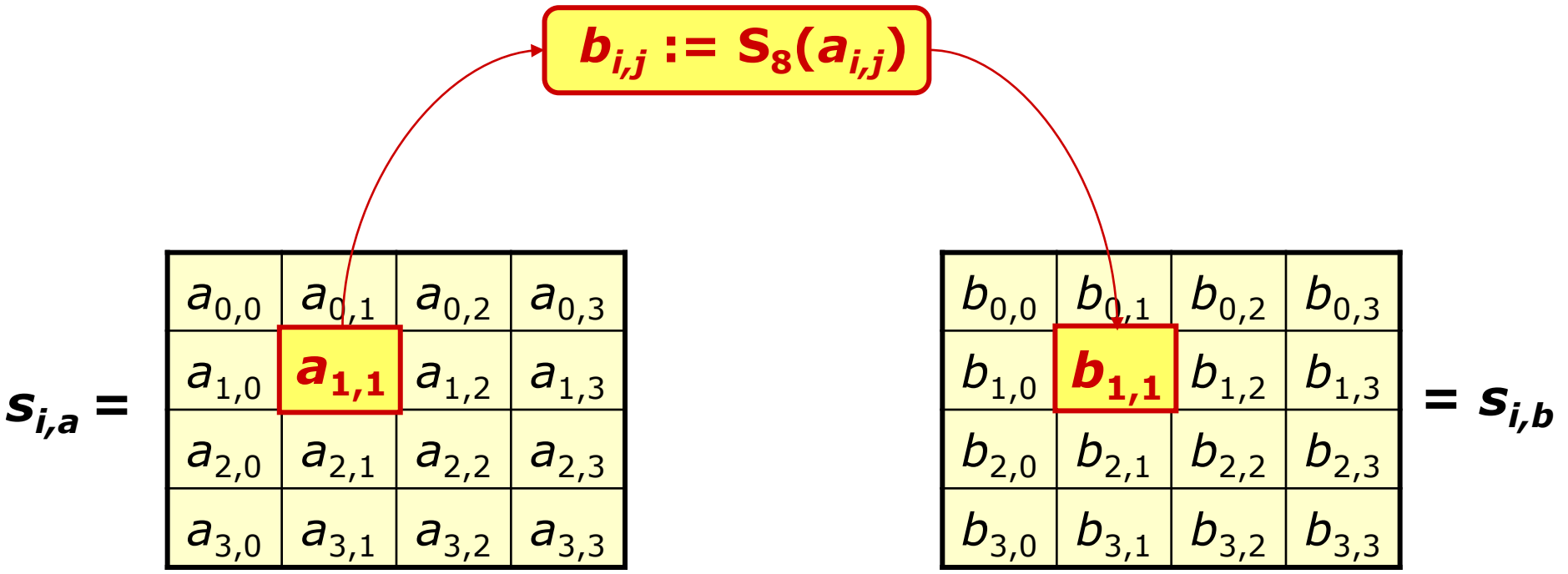
    mit $m(x) = x^8 + x^4 + x^3 + x + 1$ (irreducible polynomial)
- Operationen auf 4-Byte langen Spalten der Matrizen:

  $a(x) = a_3 x^3 + a_2 x^2 + a_1 x + a_0$ mit $a_i \in \mathrm{GF}(2^8)$

  Polynom für Reduktion: $x^4 + 1$

Schritt 1: SubBytes

- Alle Bytes einer Matrix werden unabhängig voneinander substituiert



Substitutionsbox $S_8(a_7a_6a_5a_4a_3a_2a_1a_0)$

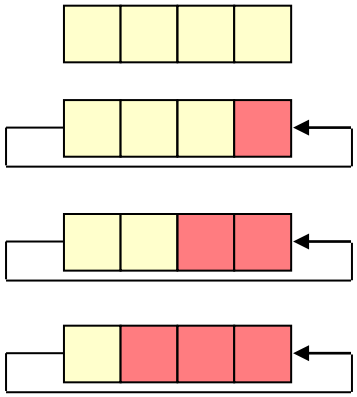
$a_7a_6a_5a_4$	$a_3a_2a_1a_0$															
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
⋮	⋮															
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Schritt 2: ShiftRow

- Zyklische Verschiebung der Zeilen nach links

Zeile	0	1	2	3
$N_b = 4$	0	1	2	3
$N_b = 6$	0	1	2	3
$N_b = 8$	0	1	3	4

AES
Rijndael



$s_{i,b} =$

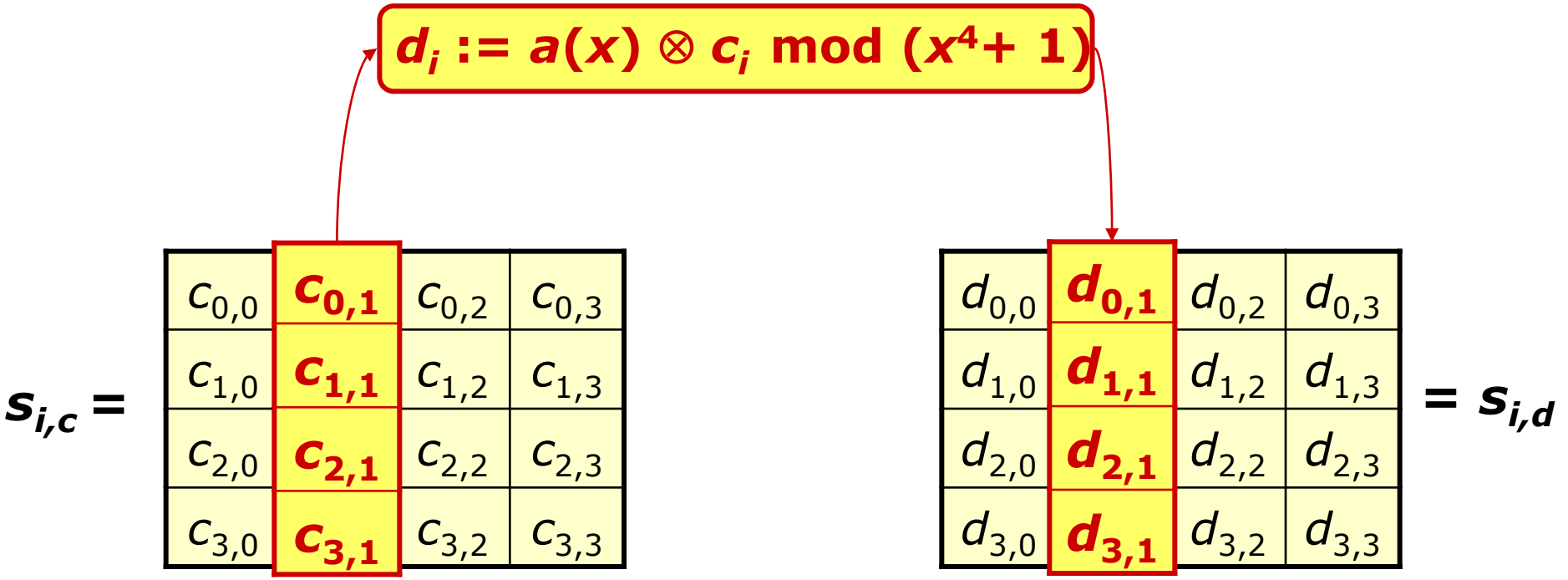
$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$
$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$
$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$

$c_{0,0}$	$c_{0,1}$	$c_{0,2}$	$c_{0,3}$
$c_{1,0}$	$c_{1,1}$	$c_{1,2}$	$c_{1,3}$
$c_{2,0}$	$c_{2,1}$	$c_{2,2}$	$c_{2,3}$
$c_{3,0}$	$c_{3,1}$	$c_{3,2}$	$c_{3,3}$

$= s_{i,c}$

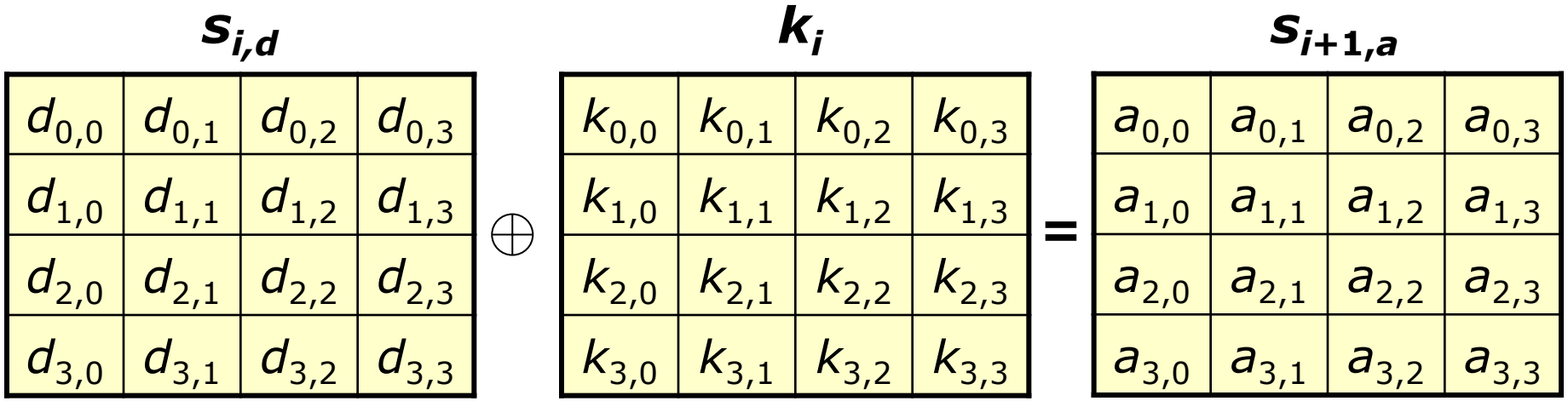
Schritt 3: MixColumn

- Operiert jeweils auf Spalten der Matrix (32-Bit Substitution)
- Verwendetes Polynom: $a(x) = 03x^3 + 01x^2 + 01x + 02$



Schritt 4: AddRoundKey

- Macht Iterationsrunden schlüsselabhängig
- Länge des Rundenschlüssels k_i: n_b



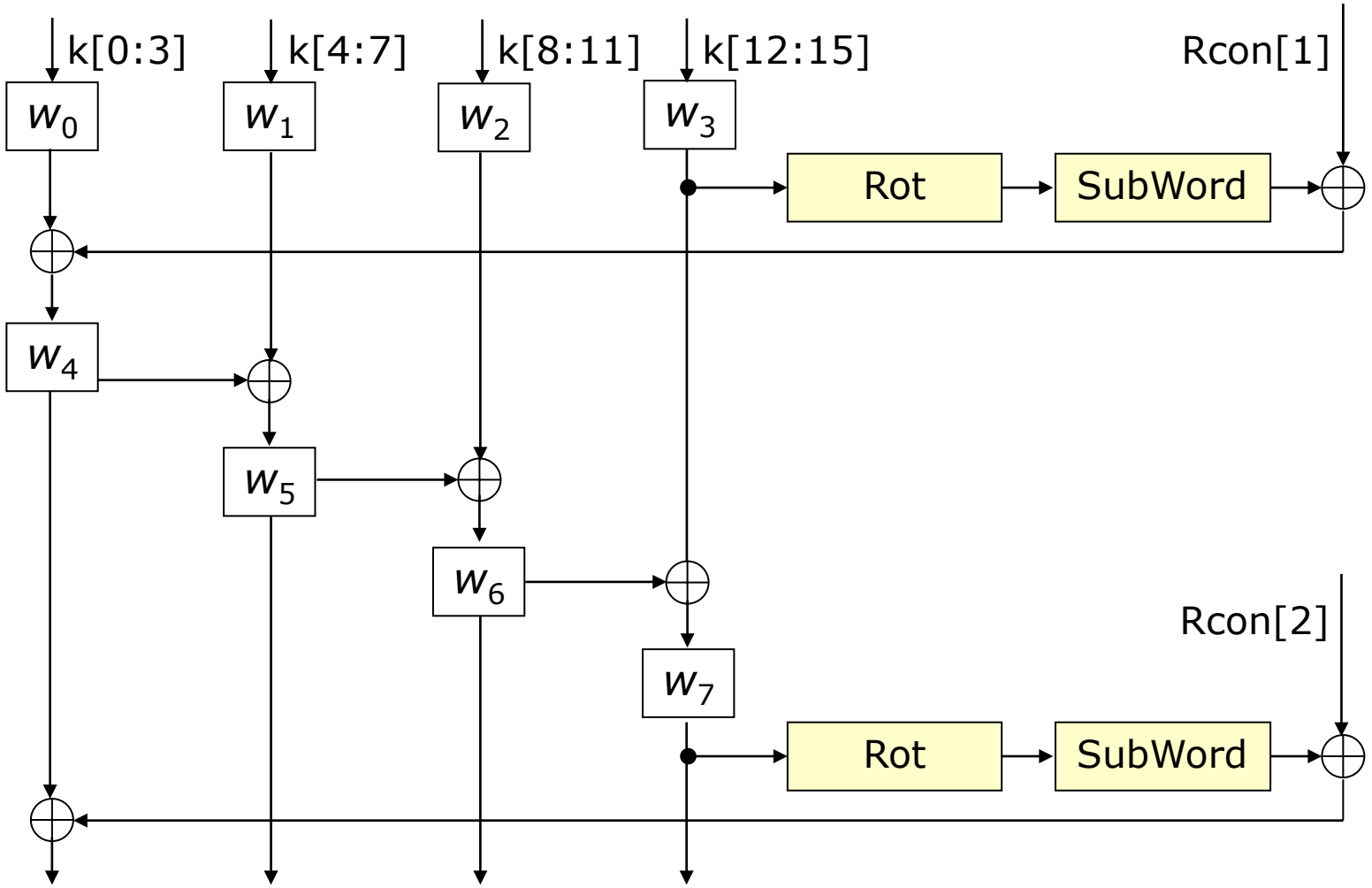
Teilschlüsselgenerierung

- Expansion des AES-Schlüssels, abhängig von n_b und n_k
- n_b bestimmt Länge der Rundenschlüssel
- n_b und n_k bestimmen Anzahl der Runden → Anzahl der Rundenschlüssel
- Expandierter Schlüssel: Folge von 4-Byte-Worten w_i (Spalten der Schlüsselmatrix)
- Länge des expandierten Schlüssels in Byte = $4N_b(r+1)$:

Schlüssel-länge n_k	Blocklänge des Klartextes n_b		
	128 Bit	192 Bit	256 Bit
128 Bit	16·11	24·13	32·15
192 Bit	16·13	24·13	32·15
256 Bit	16·15	24·15	32·15

AES
Rijndael

Schlüsselexpansion für $N_k = 4$



Entschlüsselung

- Inverse Funktionen notwendig
  - ShiftRow^{-1}: zyklische Verschiebung nach rechts
  - SubByte^{-1}: Anwendung der inversen Substitution
  - MixColumn^{-1}: Multiplikation mit dem multiplikativen Inversen des Polynoms $(a^{-1}(x))$ mod $(x^4 + 1)$

- Anwendung der Rundenschlüssel in inverser Reihenfolge

- Entschlüsselung mit äquivalenter Reihenfolge der einzelnen Schritte wie bei Verschlüsselung möglich

Betriebsarten

→ Verschlüsselung längerer Nachrichten, Authentikation (Berechnung von MACs), Vertraulichkeit *und* Integrität

Beispiele für Betriebsarten:

- Verschlüsselung:
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
  - Counter Mode (CTR)

  } 1981 für DES standardisiert (FIPS 81)
- Authentikation:
  - Cipher-based MAC (CMAC)
- Authentikation und Verschlüsselung:
  - Counter with CBC-MAC (CCM)
  - Galois/Counter Mode (GCM bzw. GMAC)

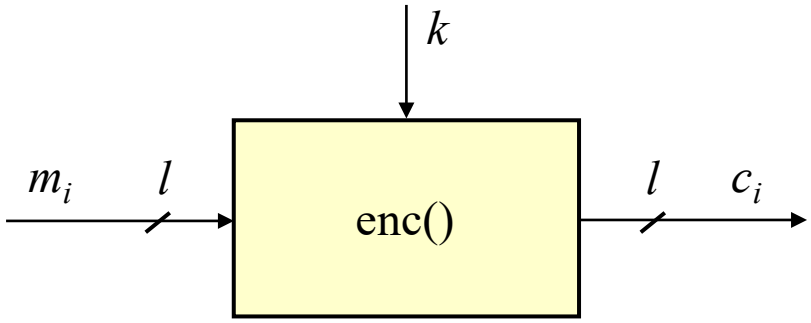
- Anwendung der Betriebsarten erlaubt die Konstruktion von synchronen oder selbstsynchronisierenden „Stromchiffren“ aus Blockchiffren
  (zugrunde liegendes Alphabet wird dabei teilweise gewechselt)
  - Synchrone Stromchiffre: Verschlüsselung eines Zeichens ist abhängig von der Position bzw. von vorhergehenden Klartext- oder Schlüsselzeichen
  - Selbstsynchronisierende Stromchiffre: Verschlüsselung ist nur von begrenzter Anzahl vorhergehender Zeichen abhängig

- ECB, CBC und CFB: selbstsynchronisierende Stromchiffre
- OFB, CTR: synchrone Stromchiffre

- Unterschiede zwischen den Betriebsarten bzgl. der Auswirkungen von Fehlern / Manipulationen, Verwendbarkeit für Authentikation, Effizienz …

Electronic Codebook (ECB)

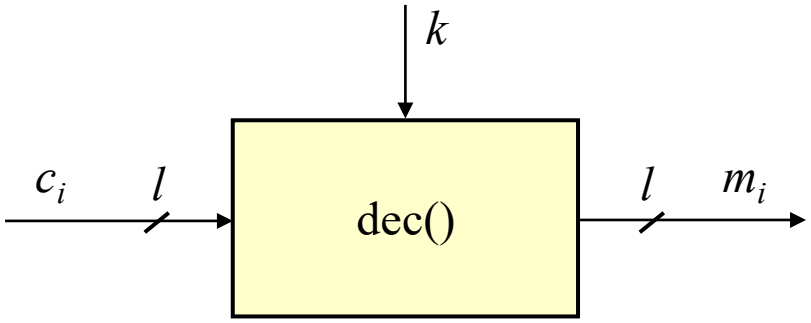
Verschlüsselung



$c_i = \text{enc}(k, m_i),\ 1 < i \leq n$

$c = \text{enc}(k, m_1)\ \text{enc}(k, m_2)\ \ldots\ \text{enc}(k, m_n)$

Entschlüsselung



$m_i = \text{dec}(k, c_i),\ 1 < i \leq n$

$m = \text{dec}(k, c_1)\ \text{dec}(k, c_2)\ \ldots\ \text{dec}(k, c_n)$

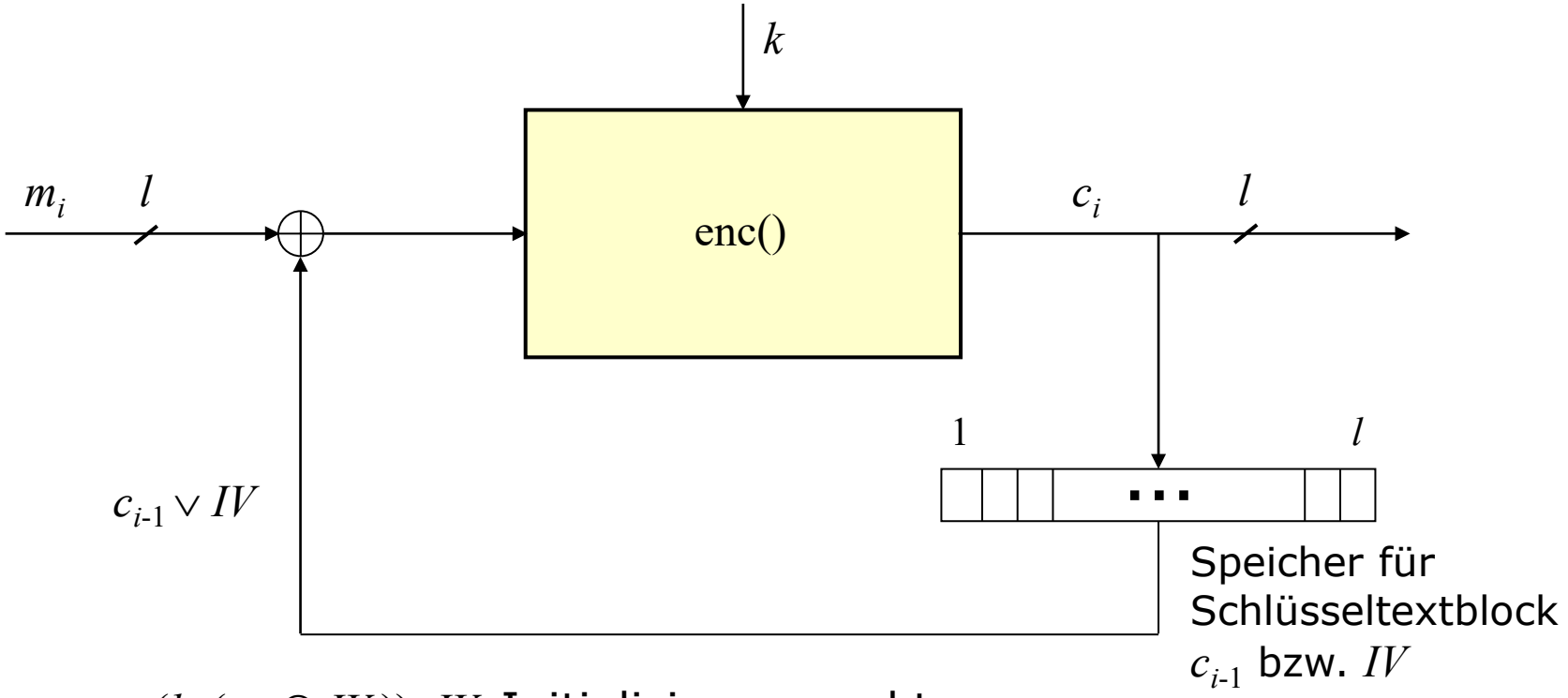
Electronic Codebook (ECB) – Eigenschaften

- Selbstsynchronisierend (Abhängigkeit von 0 Blöcken)
- Länge der verarbeiteten Einheiten: entsprechend Blockgröße der Blockchiffre (AES: l = 128 Bit)

- Keine Abhängigkeiten zwischen den Blöcken
  - Direktzugriff auf einzelne Schlüsseltextblöcke möglich
  - **Problem:** gleiche Klartextblöcke liefern gleiche Schlüsseltextblöcke
    → Kodebuchanalysen

- Problem wird durch Abhängigkeiten zwischen den Blöcken vermieden
  Beispiel: Cipher Block Chaining (CBC)

Cipher Block Chaining (CBC) – Verschlüsselung

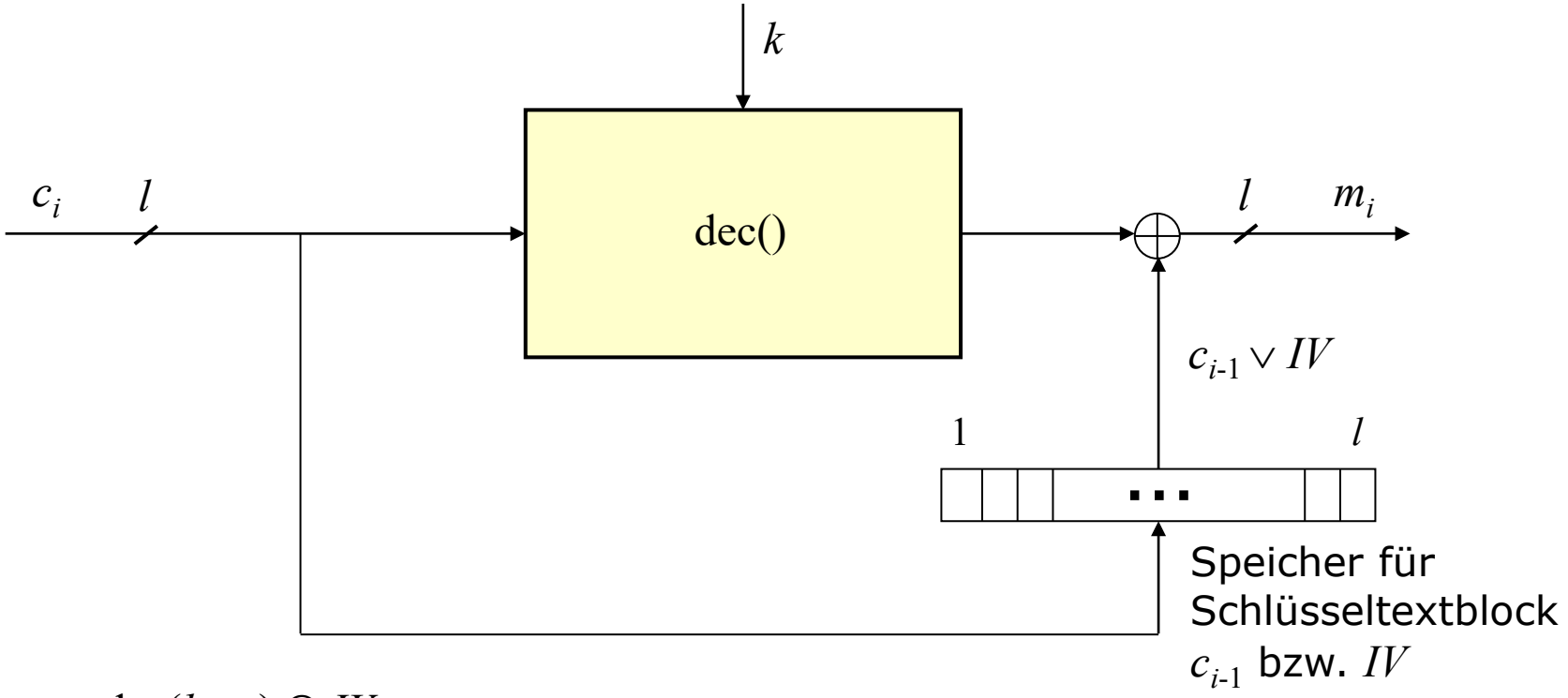


$c_1 = \text{enc}(k, (m_1 \oplus IV))$; IV: Initialisierungsvektor

$c_i = \text{enc}(k, (m_i \oplus c_{i\text{-}1}))$, $1 < i \leq n$

$c = \text{enc}(k, (m_1 \oplus IV))\ \ \text{enc}(k, (m_2 \oplus c_1))\ \ \text{enc}(k, (m_3 \oplus c_2))\ \ \ldots\ \ \text{enc}(k, (m_n \oplus c_{n\text{-}1}))$

Cipher Block Chaining (CBC) – Entschlüsselung



$$m_1 = \mathrm{dec}(k, c_1) \oplus IV$$

$$m_i = \mathrm{dec}(k, c_i) \oplus c_{i-1},\ 1 < i \leq n$$

$$m = \mathrm{dec}(k, c_1) \oplus IV \quad \mathrm{dec}(k, c_2) \oplus c_1 \quad \mathrm{dec}(k, c_3) \oplus c_2 \quad \ldots \quad \mathrm{dec}(k, c_n) \oplus c_{n-1}$$

Asymmetrische Verfahren

- Problematisch bei symmetrischen Systemen: Schlüsselverteilung, Anzahl von Schlüsseln
- Asymmetrische bzw. Public-Key-Systeme: Schlüsselpaare, bestehend aus privatem und öffentlichem Schlüssel
- Grundlage asymmetrischer Systeme: schwierige mathematische Probleme
- Es darf praktisch nicht möglich sein, den privaten Schlüssel aus dem öffentlichen Schlüssel zu ermitteln.

Grundlage: Trapdoor-Einwegfunktion

A, B Mengen

f: $A \rightarrow B$ heißt Einwegfunktion, wenn gilt:

f: $A \rightarrow B$ leicht berechenbar für alle $a \in A$, aber

f^{-1}: $B \rightarrow A$ schwierig oder nicht berechenbar für fast alle $b \in B$

Trapdoor-Eigenschaft:
Berechnung von $f^{-1}(b)$ durch Kenntnis bestimmter Zusatzparameter ebenfalls leicht berechenbar.

RSA

- Ronald L. **R**ivest, Adi **S**hamir, Leonhard M. **A**dleman: *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.* Communications of the ACM, vol. 21, no. 2, 1978, 120-126.
- Verwendung als Konzelations- und Signatursystem möglich
- Basiert auf der Faktorisierungsannahme
  - Geheimnis: 2 große, unabhängig und zufällig gewählte Primzahlen p und q
  - Öffentliche Information: $n = p \cdot q$
  - ➢ Es darf nicht möglich sein, *n* „effizient" zu faktorisieren.
  - ➢ Die Wahl von großen Primzahlen in vertretbarer Zeit muss möglich sein.
  - ➢ Notwendig: Funktionen zur Verarbeitung der Nachrichten (ver-/entschlüsseln, signieren und testen).

- Berechnungen in $\mathbb{Z}_n$
  - $\mathbb{Z}_n$: Restklassenring modulo *n*, $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$
  - Addition, Subtraktion und Multiplikation leicht
  - Berechnung von $y = a^x \bmod n$ ebenfalls effizient möglich:

    Square-and-Multiply-Algorithmus

    Binärdarstellung des Exponenten: $x_{10} = (x_{l-1}x_{l-2} \ldots x_1x_0)_2$

    $z = 1;$

    for $(i = l\text{-}1;\ i \geq 0;\ i\text{--})$ {

    $z = z^2;$

    if $(x_i == 1)$ then $z = z\ a \bmod n;$

    }

- Eulersche Φ-Funktion
  - Anzahl der zu n teilerfremden Zahlen kleiner n:
    $\Phi(n) = |\{a \in \mathbb{Z}_n | \text{ggt}(a,n) = 1\}|$ (= Ordnung der Gruppe $\mathbb{Z}_n^*$)
  - $\Phi(p) = p\text{-}1$ (p prim)
  - $n = p\,q$; p, q prim, $p \cdot q$: $\Phi(p\,q) = (p\text{-}1) \cdot (q\text{-}1)$

- Erweiterter Euklidscher Algorithmus (EEA)
  - Bestimmung von $\text{ggT}(a,b)$ und u, v mit:

    EEA(a,b) → $\text{ggT}(a,b) = u\,a + v\,b$

  - Bestimmung des multiplikativen Inversen a^{-1} von a in $\mathbb{Z}_n^*$:

    EEA(a,n) → $\text{ggT}(a,n) = u\,a + v\,n = 1$

    $u = a^{-1}$ mit $aa^{-1} \equiv 1 \bmod n$

EEA: $a,b \in \mathbb{N}$, $b > a$ → ggT(a,b), ggT(a,b) = $u\ a + v\ b$

Initialisierung

r	***q***	***s***	***t***
b		1	0
a		0	1
b mod a	b div a	$s_{-2} - q_0\ s_{-1}$	$t_{-2} - q_0\ t_{-1}$
…	…	…	…
r_{i-2} mod r_{i-1}	r_{i-2} div r_{i-1}	$s_{i-2} - q_i\ s_{i-1}$	$t_{i-2} - q_i\ t_{i-1}$
…	…	…	…
r_{k-1}	q_{k-1}	v	u
0	q_k		

Abbruch: $r_k = 0$

→ ggT(a,b) = r_{k-1}, u = t_{k-1}, v = s_{k-1}

- Spezialfall des Chinesischen Restsatzes:
  für $n = p \cdot q$ gilt

  $$a \equiv b \bmod n \;\Leftrightarrow\; a \equiv b \bmod p \wedge a \equiv b \bmod q$$

  d.h.: $n|(a-b) \;\Leftrightarrow\; p|(a-b) \wedge q|(a-b)$

- Effiziente Berechnung von $f(\mathrm{x}) \bmod n$ mit Hilfe der Kenntnis von p, q möglich ($y_p = f(\mathrm{x}) \bmod p$, $y_q = f(\mathrm{x}) \bmod q$):

  $$y \equiv f(x) \bmod n \;\Leftrightarrow y \equiv y_p \bmod p \wedge y \equiv y_q \bmod q$$

  Chinesischer Restealgorithmus (CRA):

  1. bestimme u, v mit $u \cdot p + v \cdot q = 1$ (mittels EEA)
  2. $y = \mathrm{CRA}(y_p, y_q) = u \cdot p \cdot y_q + v \cdot q \cdot y_p \bmod n$

Schlüsselgenerierung (Konzelationssystem)

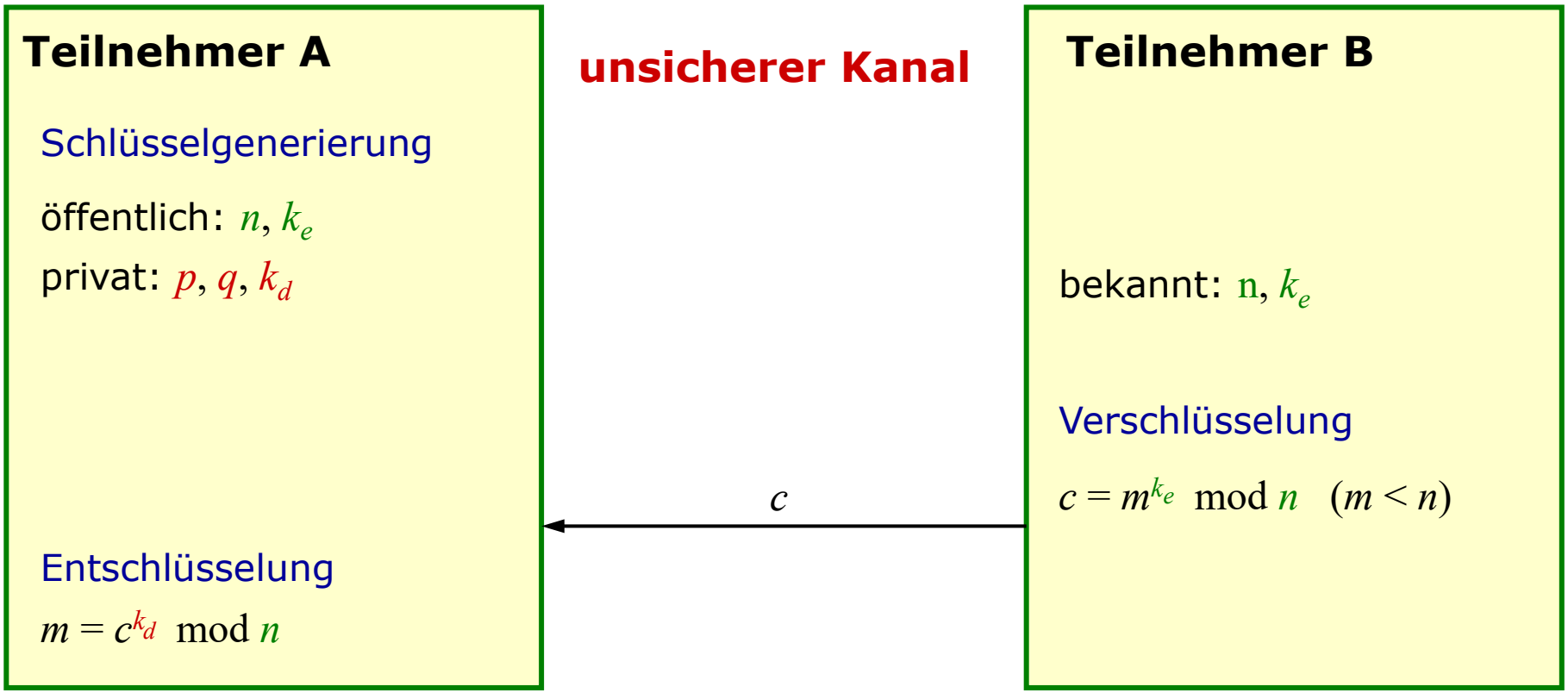
Jeder Teilnehmer

- wählt zufällig und unabhängig 2 verschiedene Primzahlen p, q ungefähr gleicher Länge
- berechnet $n = pq$
- wählt zufällige Zahl k_e mit $1 < k_e < \Phi(n)$, $\text{ggT}(k_e, \Phi(n)) = 1$
- berechnet $k_d = k_e^{-1} \bmod \Phi(n)$

- Öffentlicher Schlüssel: (n, k_e)
- Geheimer Schlüssel: (p, q, k_d)

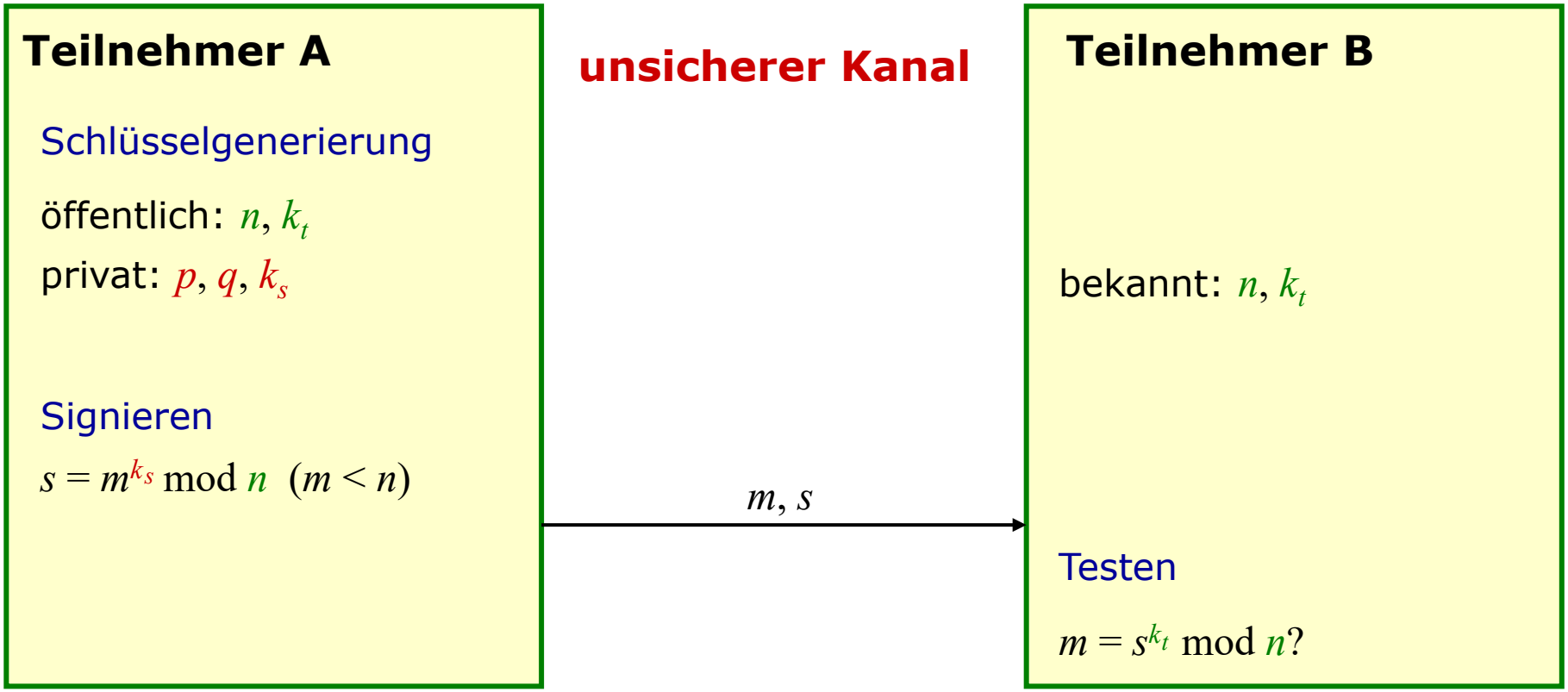
- Signatursystem: k_s statt k_d und k_t statt k_e

RSA als Konzelationssystem (unsichere Variante)



- Nachzuweisen: $\forall m \in Z_n : \left(m^{k_e}\right)^{k_d} \equiv m^{k_e \cdot k_d} \equiv m \bmod n$

RSA als Signatursystem (unsichere Variante)



Effiziente Berechnung der Entschlüsselung

- mit Hilfe der Kenntnis von p und q
- Statt Berechnung von $\mathrm{f}(x) \bmod n$:
  Berechnung von $y_p \bmod p$ und $y_q \bmod q$ und CRA(y_p, y_q)

- Einmal zu berechnen:
  $k_{d,p} = k_e^{-1} \bmod (p\text{-}1)$
  $k_{d,q} = k_e^{-1} \bmod (q\text{-}1)$

- Entschlüsselung eines Schlüsseltextes c:

$$\left.\begin{aligned} y_p &= c^{k_{d,p}} \bmod p \\ y_q &= c^{k_{d,q}} \bmod q \end{aligned}\right\} m = \mathrm{CRA}(y_p, y_q)$$

Sicherheit

- Parameterwahl
  - Primzahlen
    - Länge der verwendeten Primzahlen
    - Anforderungen an die Primzahlen aufgrund spezieller Faktorisierungsalgorithmen
  - Angriff auf RSA als Konzelationssystem bei zu kleinem öffentlichen Schlüssel
- sichere Verwendung
  - Verwendung unterschiedlicher Module für unterschiedliche Nutzer (z.B. Verhinderung der „Common Modulus Attack“)
  - Verhinderung passiver Angriffe notwendig
  - Verhinderung aktiver Angriffe notwendig

Passive Angriffe

- RSA arbeitet deterministisch
- Konzelationssystem
  - Angreifer: probeweise Verschlüsselung von Klartextblöcken und Vergleich mit beobachteten Schlüsseltextblöcken
  - Abhilfe: Hinzunahme einer Zufallszahl *r* → indeterministische Verschlüsselung der Nachrichten („Randomisierung", „Padding")

$$c = (r, m)^k \mod n$$

  - PKCS #1 v 1.5 (verwendet in SSL v 3.0): 1998 von Bleichenbacher gebrochen (gewählter Schlüsseltext-Klartext-Angriff)
  - PKCS #1 v 2.1 basierend auf OAEP (Optimal Asymmetric Encryption Padding, Bellare und Rogaway 1995)

Aktive Angriffe (am Beispiel des Signatursystems)

- Grundlage: RSA ist Homomorphismus bzgl. Multiplikation
- Angreifer
  - beobachtet Signaturen s_1, s_2 für Nachrichten m_1, m_2
  - berechnet Signatur $s_3 = s_1 s_2 \bmod n$ für Nachricht $m_3 = m_1 m_2 \bmod n$ (m_3 jedoch nicht frei wählbar)

Aktiver Angriff von Davida (selektiv)

- Ziel: Signatur für *gewählte* Nachricht m_3
- Angreifer
  - wählt m_1 und berechnet $m_1^{-1} \bmod n$
  - berechnet $m_2 = m_3 m_1^{-1} \bmod n$
  - lässt m_1 und m_2 signieren → erhält s_1, s_2
  - berechnet $s_3 = s_1 s_2 \bmod n$

Verbesserter aktiver Angriff von Moore (selektiv)

- Ziel: Signatur für *gewählte* Nachricht m_2
- Angreifer
  - wählt $r \in Z_n^*$, berechnet $r^{-1} \bmod n$
  - berechnet $m_1 = m_2 r^{k_t} \bmod n$
  - lässt m_1 signieren → erhält s_1
  - berechnet $s_2 = s_1 r^{-1} \bmod n$

- Anwendung der Angriffe auf RSA als Konzelationssystem möglich
- Nutzung des Angriffs von Moore zur Berechnung blinder Signaturen

Verhinderung der skizzierten Angriffe

Konzelationssystem:

- Hinzunahme einer Zufallszahl r
- Hinzufügen von Redundanz, die nach der Entschlüsselung geprüft wird: Anwendung einer kollisionsresistenten Hashfunktion $\mathrm{h}()$ auf Nachricht und Zufallszahl:

$$c = (r, m, \mathrm{h}(r, m))^k$$

Signatursystem:

- Anwendung einer kollisionsresistenten Hashfunktion $\mathrm{h}()$ auf die Nachricht:

$$s = (\mathrm{h}(m))^k$$