Betriebssysteme und Sicherheit

Asymmetrische Kryptographie

WS 2012/2012

Dr.-Ing. Elke Franz
Elke.Franz@tu-dresden.de

Überblick

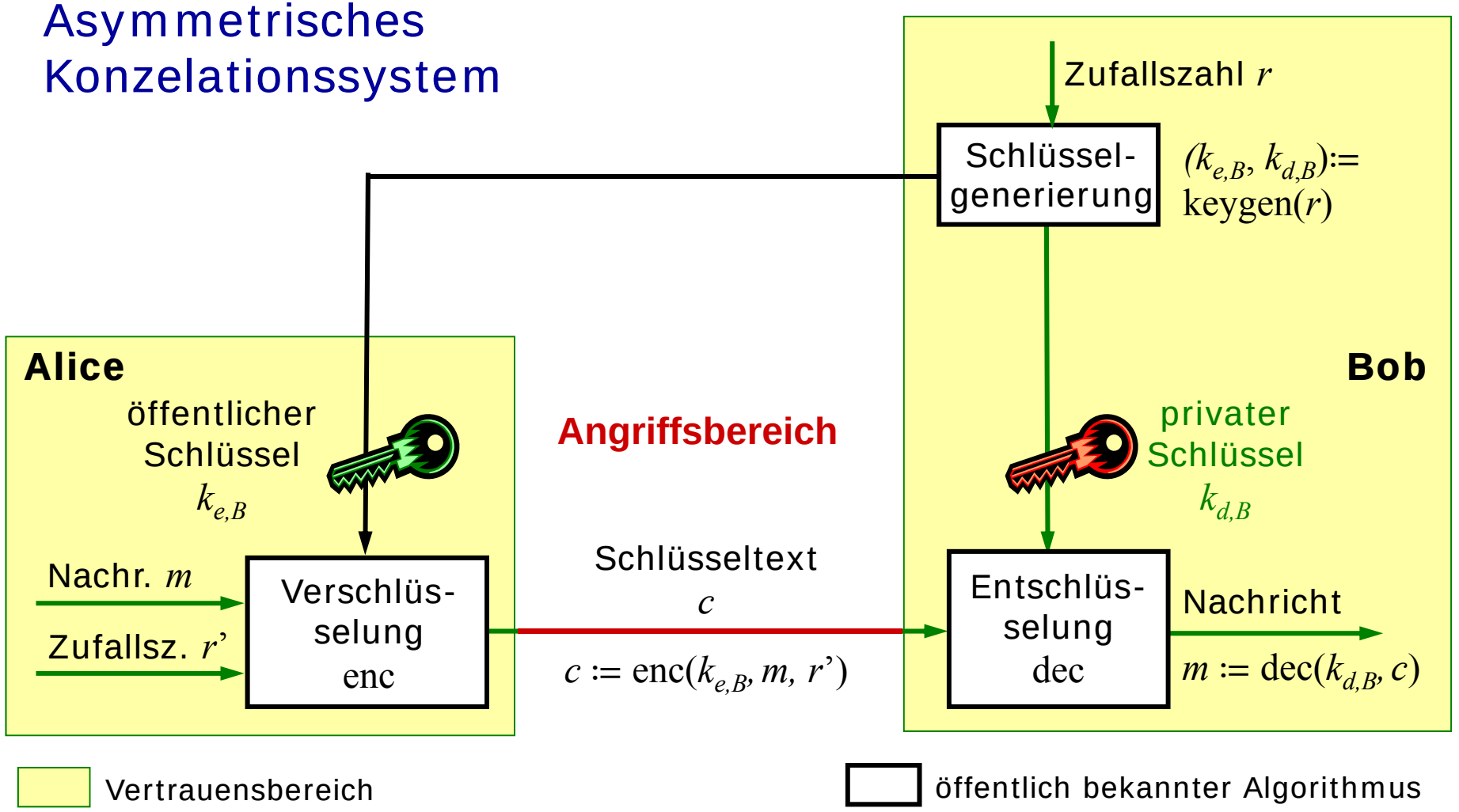
1 Prinzip asymmetrischer (Konzelations-)Systeme

2 Mathematische Grundlagen

3 Beispiel: Diffie-Hellman-Schlüsselaustausch

4 Beispiel: RSA zur Konzelation

Asymmetrisches Konzelationssystem



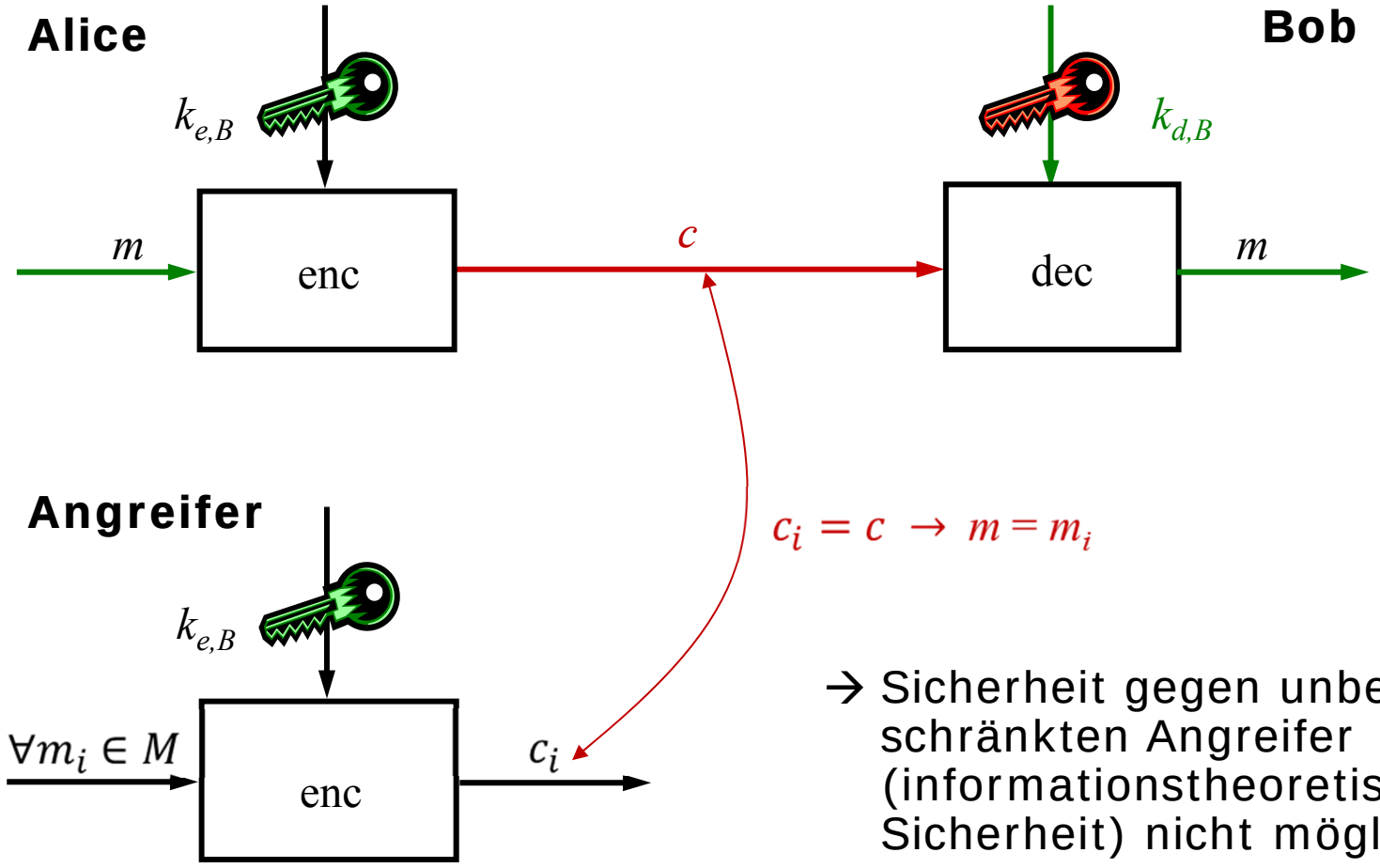
1 Prinzip asymmetrischer (Konzelations-)Systeme

- 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

- Erreichbare Sicherheit asymmetrischer Systeme



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

1 Prinzip asymmetrischer (Konzelations-)Systeme

Symmetrische Systeme

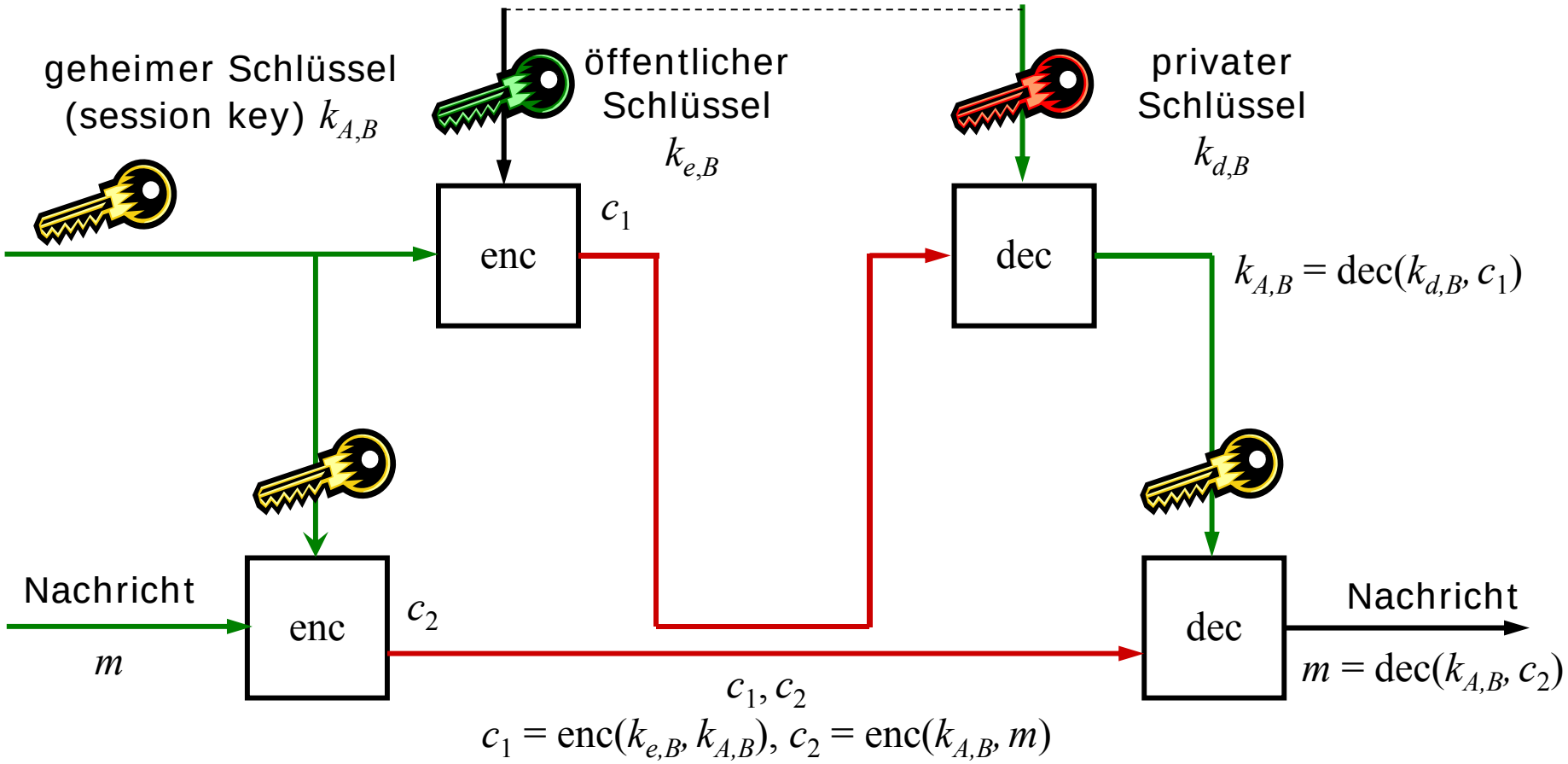
- Sicherer Kanal für Schlüsselaustausch erforderlich
- Gute Performance

Asymmetrische Systeme

- Vertraulichkeit des öffentlichen Schlüssels nicht erforderlich
- Rechenoperationen aufwändiger

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

Hybrides Konzelationssystem



Grundlage asymmetrischer Verfahren

- Es darf praktisch nicht möglich sein, den privaten Schlüssel aus dem öffentlichen Schlüssel zu ermitteln.
- Die geheime Operation darf ohne Kenntnis des privaten Schlüssels nicht (effizient) durchführbar sein.

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.

Ausnutzen schwieriger mathematischer Probleme, für deren Lösung kein effizienter Algorithmus bekannt ist.

- Erzeugung von Primzahlen
  1. Wahl einer Zufallszahl p als Kandidat für die Primzahl
  2. Test, ob p prim ist
  3. Wiederholung von 1. und 2., bis Primzahl gefunden

- Primzahltest nach Rabin-Miller für $p \equiv 3 \bmod 4$

  p prim: $\forall a \in Z_p^*: a^{\frac{p-1}{2}} \equiv \pm 1 \bmod p$

  falls p nicht prim, gilt dies für höchstens ¼ der möglichen a
- Test für l zufällig gewählte Werte a durchführen
  - Ergebnis einmal $\neq \pm 1$: p nicht prim
  - Ergebnis bei allen l Versuchen $= \pm 1$: p prim mit Wahrscheinlichkeit $\geq 1 - 4^{-l}$

Algebraische Strukturen

- Endliche Strukturen (z.B. Gruppen), basierend z.B. auf den natürlichen oder ganzen Zahlen

- $\mathbb{Z}_n = \{0,1,2, \ldots, n\text{-}1\}$ Restklassenring modulo n

- Kongruenz

  $a, b \in \mathbb{Z}; n \in \mathbb{Z} - \{0\}: n|(a\text{-}b) \rightarrow a, b$ kongruent

  $$a \equiv b \bmod n$$

- Restklasse $\bar{a}$ zu jedem $a \in \mathbb{Z}$: $\bar{a} := \{b \in \mathbb{Z} \mid a \equiv b \bmod n\}$

- $\mathbb{Z}_n^*$: multiplikative Gruppe, $\mathbb{Z}_n^* := \{a \in \mathbb{Z}_n \mid \text{ggT}(a, n) = 1\}$

- Zyklische Gruppe
  - alle Elemente der Gruppe G lassen sich aus einem Element
    $g \in G$ (erzeugendes Element oder Generator)
    durch Potenzieren von g erzeugen: $G = <g>$
  - Ordnung von Gruppenelementen $a \in G$: $\text{order}_G\, a$ bzw. $\text{order}\, a$
    kleinste natürliche Zahl e mit $a^e = 1$
    zyklische Gruppe: $\text{order}\, g = |G|$ (Ordnung von G)
  - Multiplikative Gruppe $Z_p^* = \{1, 2, \ldots, p\text{-}1\}$ (p prim) ist zyklisch:
    $Z_p^* = <g> = \{g^i \bmod p \mid i = 0, 1, \ldots, \Phi(p)\text{-}1\}$

$Z_7^* = \{1,2,3,4,5,6\}$

$3^0 = 1$

$3^1 = 3^0 \bullet 3 = 3$

$3^2 = 3^1 \bullet 3 = 9 \equiv 2 \bmod 7$

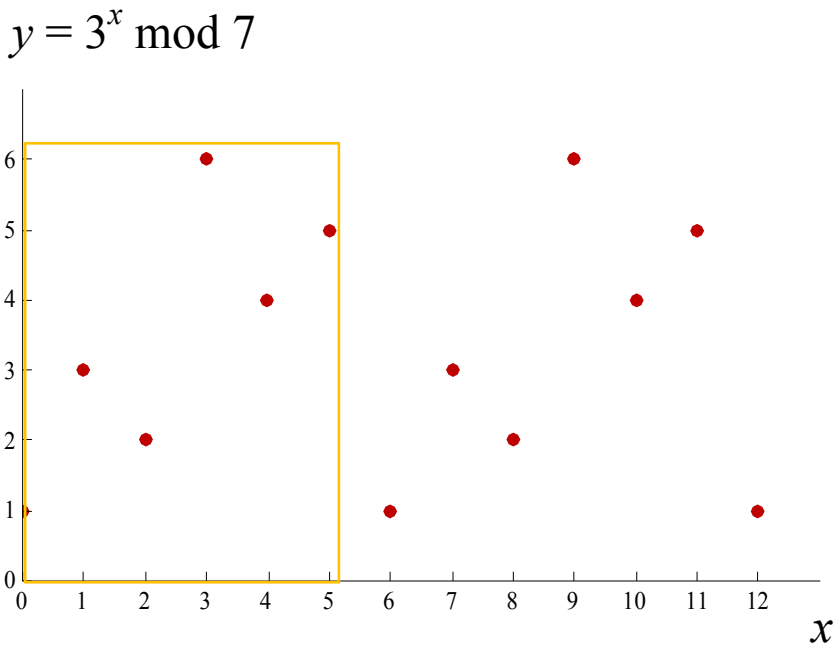
$3^3 = 3^2 \bullet 3 = 6$

$3^4 = 3^3 \bullet 3 = 18 \equiv 4 \bmod 7$

$3^5 = 3^4 \bullet 3 = 12 \equiv 5 \bmod 7$

$3^6 = 3^5 \bullet 3 = 15 \equiv 1 \bmod 7$

…



$3^6 \equiv 1 \bmod 7 \rightarrow$ order 3 = 6 $\rightarrow$ 3 ist Generator von Z_7^*

$Z_7^* = \{1,2,3,4,5,6\}$

$2^0 = 1$

$2^1 = 2^0 \bullet 2 = 2$

$2^2 = 2^1 \bullet 2 = 4$

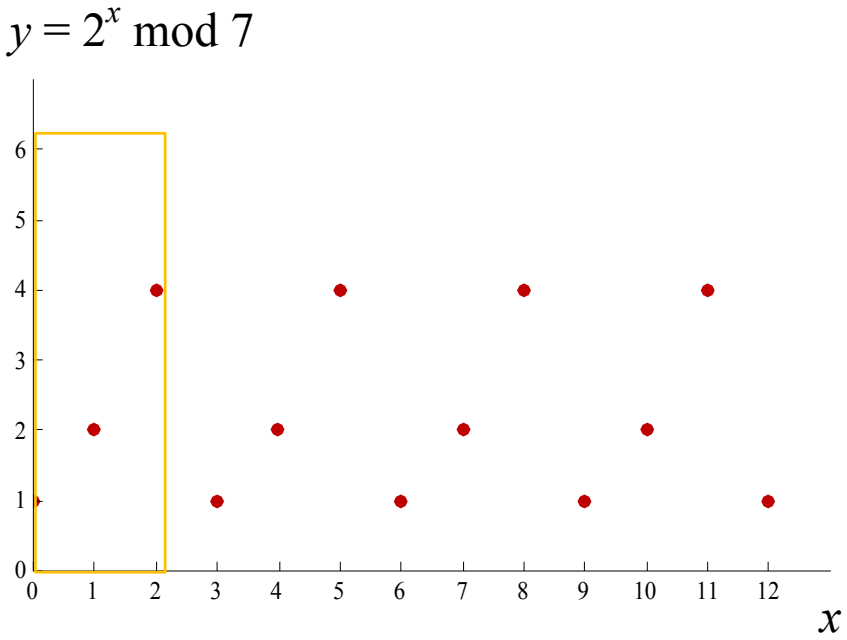
$2^3 = 2^2 \bullet 2 = 8 \equiv 1 \bmod 7$

$2^4 = 2^3 \bullet 2 = 2$

$2^5 = 2^4 \bullet 2 = 4$

$2^6 = 2^5 \bullet 2 = 8 \equiv 1 \bmod 7$

…



$2^3 \equiv 1 \bmod 7$ → order 2 = 3 → 2 ist kein Generator von Z_7^*

3 Beispiel: Diffie-Hellman-Schlüsselaustausch

- Diskreter Logarithmus
  - Für jede Zahl $y \in Z_p^*$ gibt es einen Exponenten x mit $0 \leq x \leq p\text{-}2$, so dass gilt:

$$y = g^x \bmod p.$$

  - Der Exponent x wird diskreter Logarithmus von y zur Basis g modulo p genannt:

$$x = \log_g y \bmod p$$

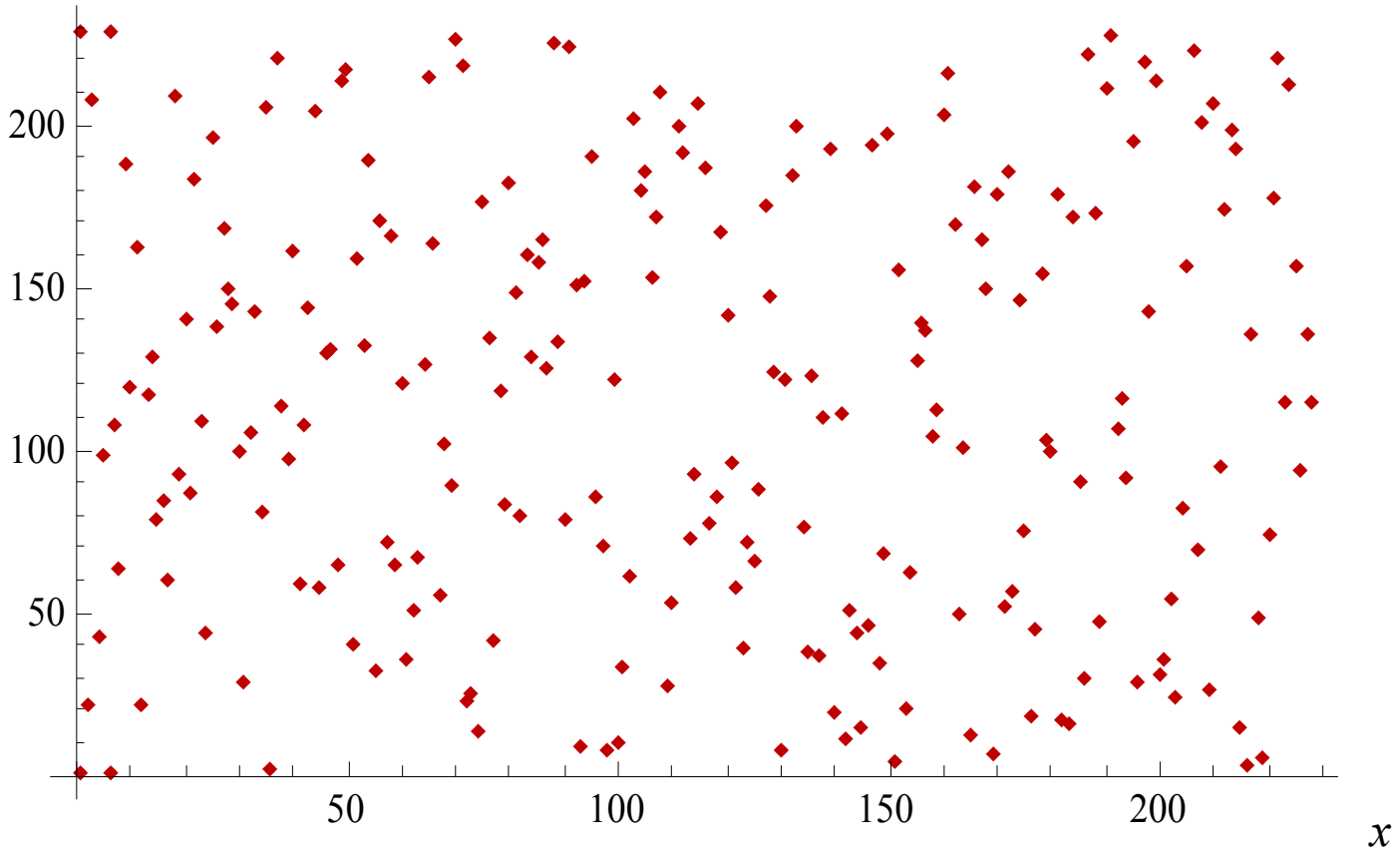
    Bestimmen von x: Diskreter-Logarithmus-Problem

- kein Algorithmus zur *effizienten* Berechnung des diskreten Logarithmus in beliebigen Gruppen bekannt
- verschiedene Algorithmen wie z.B. Babystep-Giantstep-Algorithmus von Shanks (nicht praktikabel für größere Gruppen ab (ca. ab $|G| > 2^{160}$) oder
  Pohlig-Hellman-Algorithmus (falls $p\text{-}1$ kleine Primteiler hat)

3 Beispiel: Diffie-Hellman-Schlüsselaustausch

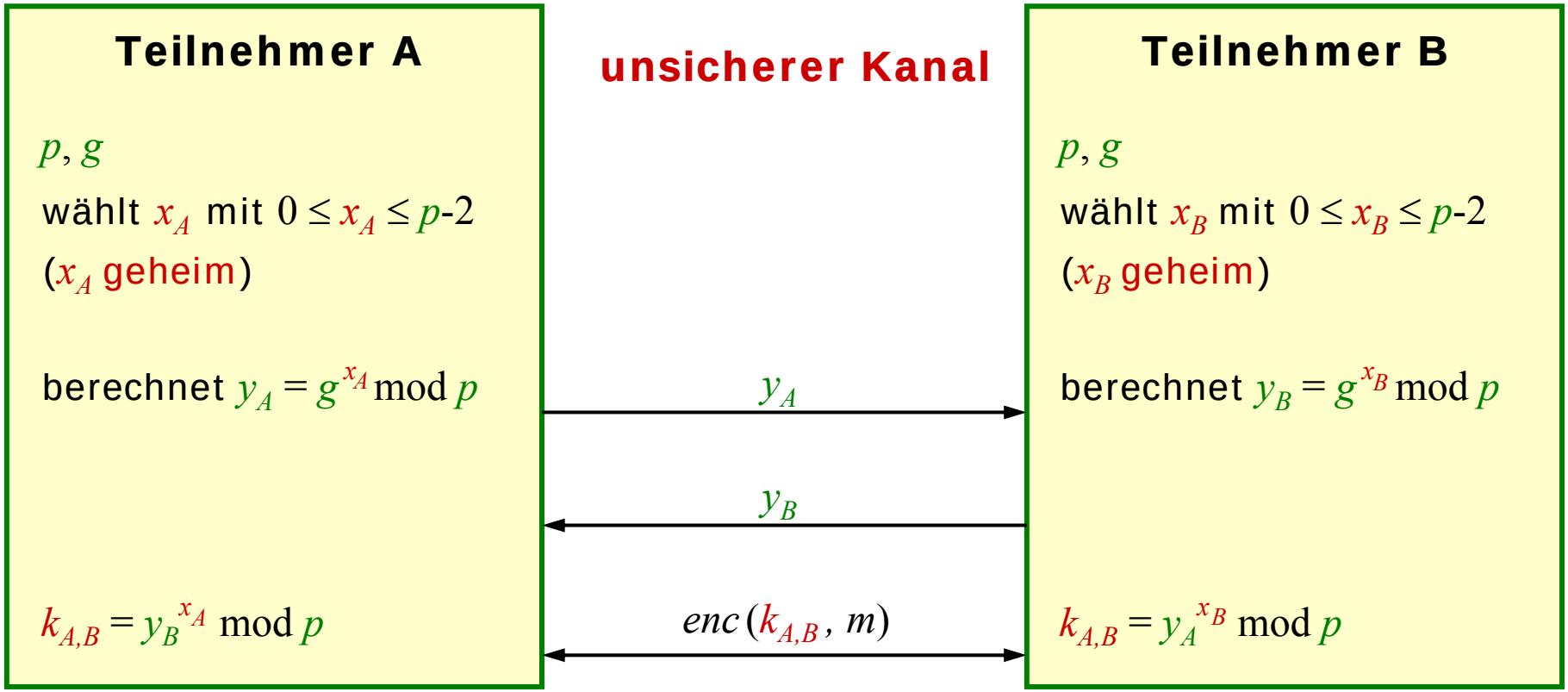
- Beispiel: Diskreter Logarithmus für $p = 229, g = 6$

$y = \log_6 x \bmod 229$



Diffie-Hellman-Schlüsselaustausch

Öffentlich bekannt: Primzahl p, Generator g



Sicherheit des Diffie-Hellman-Schlüsselaustauschs

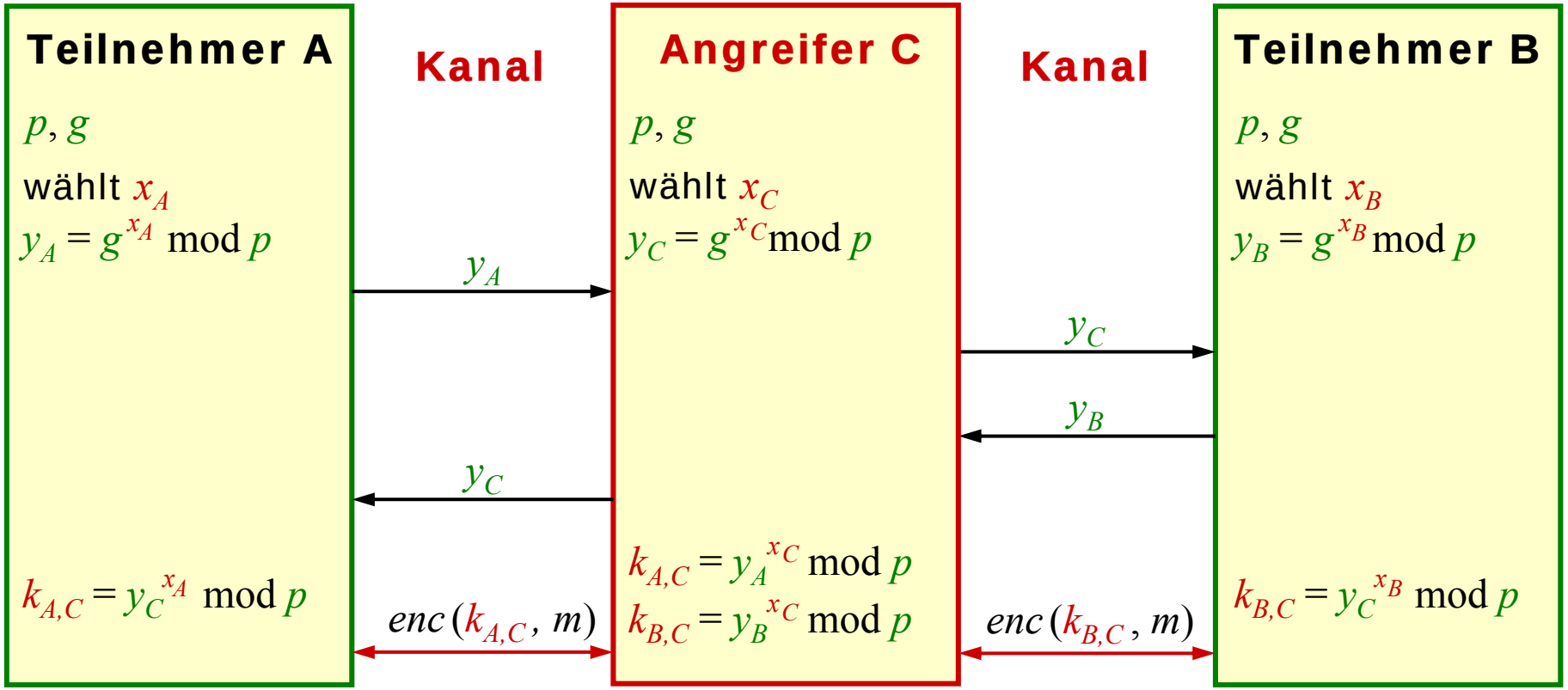
- Diffie-Hellman-Problem:
  Geg.: p, g, $y_A = g^{x_A} \bmod p$ und $y_B = g^{x_B} \bmod p$
  Problem: bestimme $g^{x_A x_B} \bmod p$

- Angreifer beobachtet y_A, y_B, kann er
  $x_A = \log_g y_A \bmod p$ oder
  $x_B = \log_g y_B \bmod p$
  berechnen, kann er das Geheimnis $k_{AB} = g^{x_A x_B} \bmod p$ ermitteln.

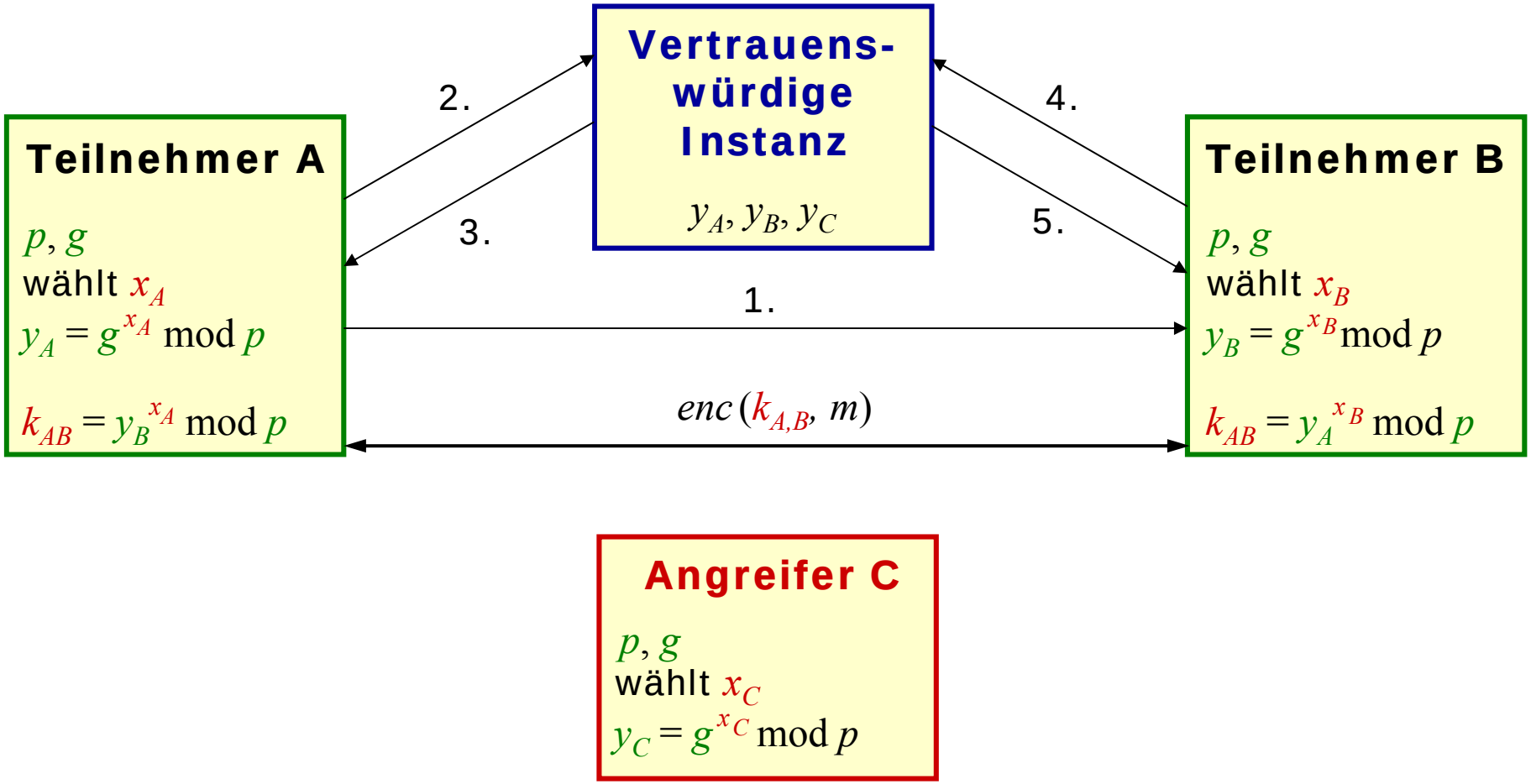
- sicher gegen passive Angriffe
- aber: unsicher gegen aktive Angriffe

Aktiver Angriff (Man-in-the-Middle-Angriff)

C gibt sich gegenüber A als B und gegenüber B als A aus.



Abhilfe: Vertrauenswürdige Instanz



RSA

- Ronald L. Rivest, Adi Shamir, Leonhard M. Adleman: *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.* Communications of the ACM, vol. 21, no. 2, 1978, 120-126.
- Basiert auf der Faktorisierungsannahme

  (kein Beweis)
- Verwendung als Konzelations- und Signatursystem

Mathematische Grundlagen

- Berechnung von $n = pq$ leicht, aber Faktorisierung von n schwer
- Faktorisierungsalgorithmen
  - Spezielle Algorithmen: Anforderungen an die Faktoren von n
  - Allgemeine Algorithmen: hängen nur von der Größe von n ab
- Beispiel für einen speziellen Algorithmus

  Faktorisierung nach Fermat (1643)

  → n ungerade, p, $q \sim \sqrt{n}$
  - Zerlegung $n = x^2 - y^2 = (x+y)(x-y)$
  - Start mit $x = \lceil \sqrt{n} \rceil$
  - Berechnung von $x^2 - n, (x+1)^2 - n, (x+2)^2 - n, \ldots$

    bis Ergebnis eine Quadratzahl ist $(y^2 = x^2 - n)$
  - Faktoren von n: $p = x + y, q = x - y$

4 Beispiel: RSA

- Eulersche Φ-Funktion
  - Anzahl der zu n teilerfremden Zahlen kleiner n:
    $\Phi(n) := |\{a \in \mathbb{Z}_n \mid \text{ggT}(a,n) = 1\}|$
  - Ordnung der Gruppe $\mathbb{Z}_n^*$ (bzw. $\mathbb{Z}_p^*$); $\mathbb{Z}_n^* := \{a \in \mathbb{Z}_n \mid \text{ggT}(a,n) = 1\}$
    (Anzahl der Elemente dieser Gruppe)

  - $\Phi(p) = p-1$ (p prim)

  - $n = p \bullet q$; p, q prim, $p \neq q$: $\Phi(p \bullet q) = (p-1)(q-1)$

4 Beispiel: RSA

- Erweiterter Euklidscher Algorithmus (EEA)
  - Bestimmung von $ggT(a,b)$ und seiner Linearkombinationsdarstellung:

    EEA(a,b) → $ggT(a,b) = u \bullet a + v \bullet b$

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

    EEA(a,n) → $ggT(a,n) = u \bullet a + v \bullet n = 1$

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

4 Beispiel: RSA

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

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

Initialisierung

Abbruch: $r_k = 0$

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

4 Beispiel: RSA

- Spezialfall des Chinesischen Restsatzes (benötigt für Nachweis der RSA-Entschlüsselung):

  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. $y = \mathrm{CRA}(y_p, y_q) = u \cdot p \cdot y_q + v \cdot q \cdot y_p \bmod n$

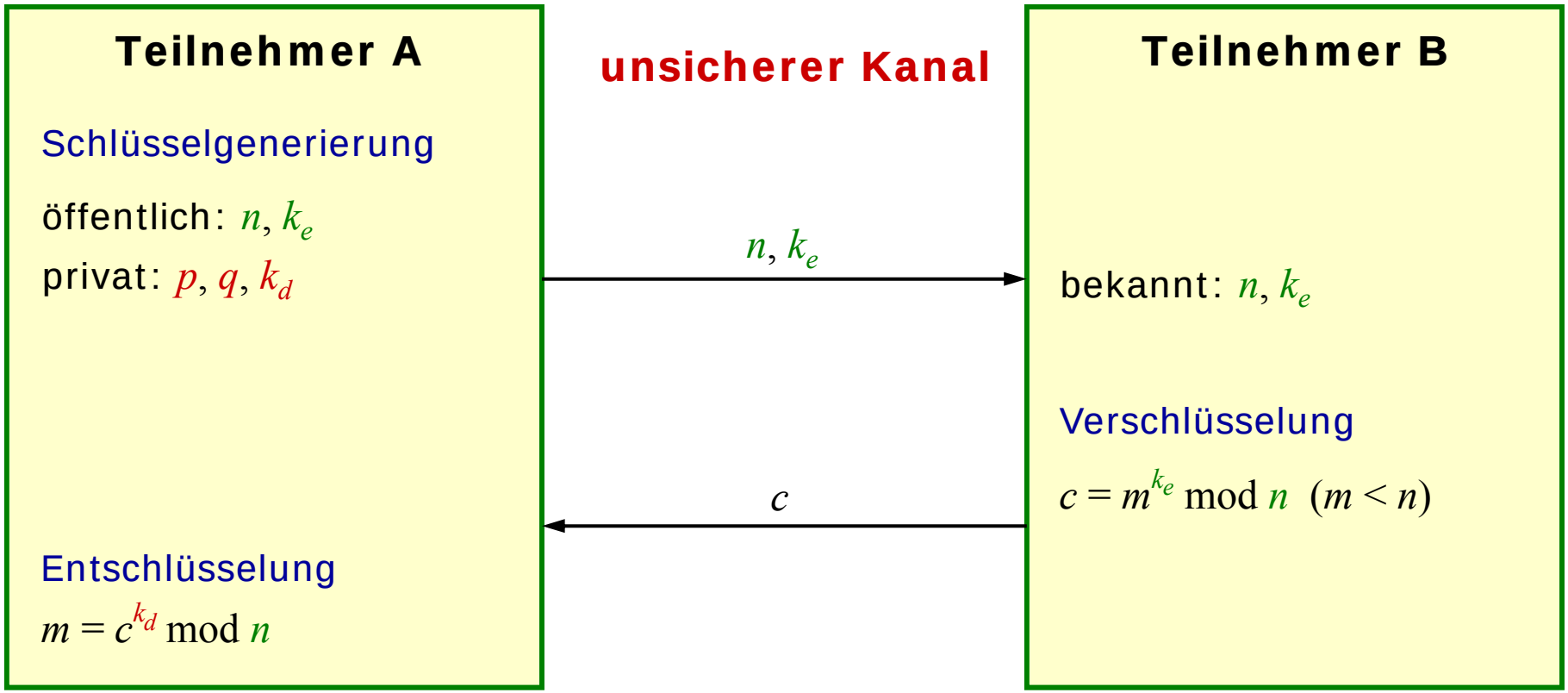
Schlüsselgenerierung

Jeder Teilnehmer

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

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

RSA als Konzelationssystem (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:
  $k_{d,p} = k_e^{-1} \bmod (p\text{-}1) \rightarrow (c^{k_{d,p}})^{k_e} \equiv c \bmod p$
  $k_{d,q} = k_e^{-1} \bmod (q\text{-}1) \rightarrow (c^{k_{d,q}})^{k_e} \equiv c \bmod q$

- Entschlüsselung eines Schlüsseltextes c:

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

Sicherheit

- Parameterwahl
  - Primzahlen
    - Länge der verwendeten Primzahlen
    - Anforderungen an die Primzahlen aufgrund spezieller Algorithmen
  - Angriff auf RSA als Konzelationssystem bei zu kleinem öffentlichen Schlüssel
- sichere Verwendung
  - Verwendung unterschiedlicher Module für unterschiedliche Nutzer (Verhinderung der „Common Modulus Attack“)
  - Verhinderung passiver Angriffe durch indeterministische Verschlüsselung
  - Verhinderung aktiver Angriffe durch Hinzufügen von Redundanz

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“)

$$c = (r, m)^{k_e} \bmod n$$

  - 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)
  - Details: http://www.rsa.com/rsalabs/pkcs/