# Kryptografie verständlich

Ein Fachbuch für
Studierende und Anwender

von
Christof Paar und Jan Pelzl

www.crypto-textbook.com


# Kapitel 6

## Einführung in die asymmetrische Kryptografie

(Version: 1. Dezember 2016)

# Übersicht


- Wiederholung zu symmetrischer Kryptografie
- Prinzip der asymmetrischen Kryptografie
- Praktische Aspekte der asymmetrischen Kryptografie
- Praxisrelevante asymmetrische Verfahren
- Hintergrund: Zahlentheorie für asymmetrische Verfahren

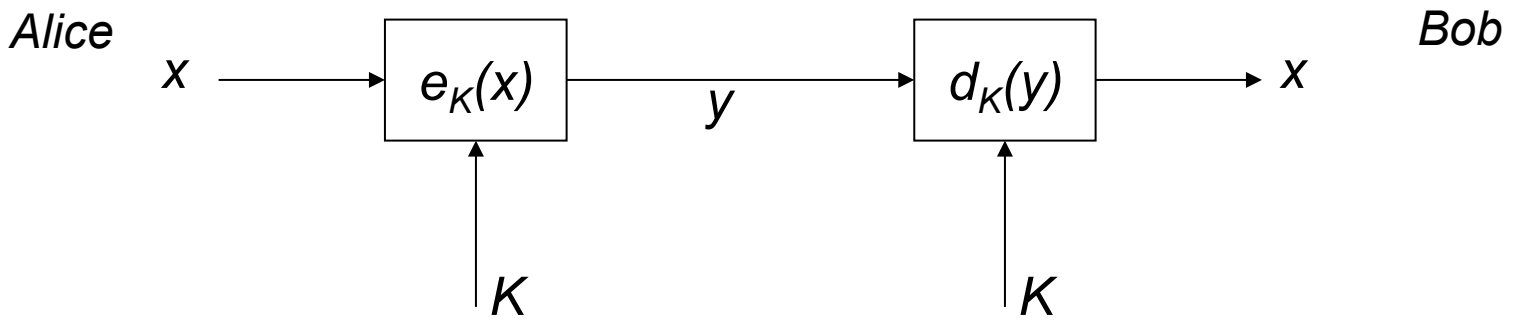
# Übersicht


- **Wiederholung zu symmetrischer Kryptografie**
- Prinzip der asymmetrischen Kryptografie
- Praktische Aspekte der asymmetrischen Kryptografie
- Praxisrelevante asymmetrische Verfahren
- Hintergrund: Zahlentheorie für asymmetrische Verfahren


# Einführung in die asymmetrische Kryptografie

## Wiederholung: Symmetrischen Kryptografie

Kurze Wiederholung der symmetrischen Kryptografie




Zwei Eigenschaften symmetrischer Verfahren:

- Ein und **derselbe geheime Schlüssel** wird für die Ver- und Entschlüsselung verwendet.
- Ver- und Entschlüsselung sind vom Prinzip her gleich (symmetrische Algorithmen).


# Einführung in die asymmetrische Kryptografie

## Wiederholung: Symmetrischen Kryptografie


- Safe mit starkem Schloss, zu dem nur Alice und Bob einen Schlüssel haben
- Alice schließt Nachricht mit Ihrem Schlüssel im Safe ein
- Bob öffnet den Safe mit seinem Schlüssel


# Einführung in die asymmetrische Kryptografie

## Probleme der symmetrischen Kryptografie

- Symmetrische Verfahren wie AES sind weitverbreitet und effizient, **ABER**:

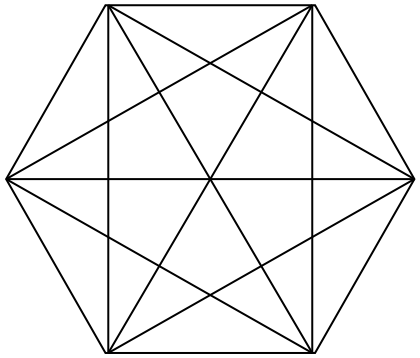
- Schlüsselverteilungsproblem: **Sicherer Transport** des Schlüssel
- Anzahl der Schlüssel: In einem Netzwerk benötigt jedes Paar von Nutzern einen eigenen Schlüssel

→ *n* Nutzer im Netz benötigen *(n-1)/2* Schlüssel, jeder Nutzer speichert *(n-1)* Schlüssel

**Beispiel:**

6 Nutzer (Ecken)

$$\frac{6 \cdot 5}{2} = 15$$ Schlüssel (Kanten)



- Alice oder Bob können sich gegenseitig betrügen, da sie identische Schlüssel haben
  **Beispiel**: Alice kann behaupten, dass sie niemals einen Fernseher online von Bob bestellt hat (er könnte Ihre Bestellung fingiert haben). Um dies zu vermeiden, hilft nur „Nichtzurückweisbarkeit“.

# Übersicht


- Wiederholung zu symmetrischer Kryptografie
- **Prinzip der asymmetrischen Kryptografie**
- Praktische Aspekte der asymmetrischen Kryptografie
- Praxisrelevante asymmetrische Verfahren
- Hintergrund: Zahlentheorie für asymmetrische Verfahren


# Einführung in die asymmetrische Kryptografie

## Neuartige Idee

Wende das Briefkasten-Prinzip auf einen Algorithmus an:

Jeder (Sender) kann Briefe in den Kasten hinein werfen.

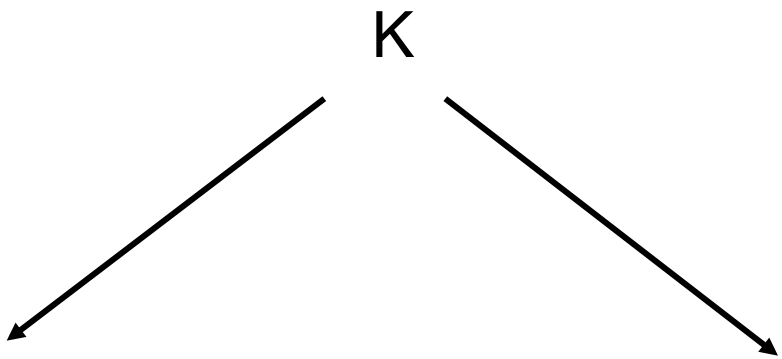
Nur der Empfänger hat den richtigen Schlüssel zum Öffnen des Kastens.

1976 wurde erstmals ein mathematischer Algorithmus für ein solches Verfahren publiziert [**Diffie, Hellman, Merkle**].


# Einführung in die asymmetrische Kryptografie

## Prinzip des Schlüsselpaars

Prinzip: “teile” den Schlüssel




Öffentlicher Teil ($K_{pub}$)
(Verschlüsselung)

Geheimer Teil ($K_{pr}$)
(Entschlüsselung)

→ Während der Schlüsselerzeugung wird ein **Schlüsselpaar** $K_{pub}$ und $K_{pr}$ berechnet

# Übersicht


- Wiederholung zu symmetrischer Kryptografie
- Prinzip der asymmetrischen Kryptografie
- **Praktische Aspekte der asymmetrischen Kryptografie**
- Praxisrelevante asymmetrische Verfahren
- Hintergrund: Zahlentheorie für asymmetrische Verfahren


# Einführung in die asymmetrische Kryptografie

## Einfaches Protokoll

Alice

Bob

$(K_{pubB}, K_{prB}) = K$

$K_{pubB}$

$x$

$y = e_{K_{pubB}}(x)$

$y$

$x = d_{K_{prB}}(y)$

→ Jetzt haben wir das Schlüsselaustauschproblem (vorerst) gelöst

# Einführung in die asymmetrische Kryptografie
Anwendungsfälle


Mechanismen, die mit asymmetrischen Verfahren realisiert werden können:

- **Schlüssel Einigung** (z.B. mit Diffie-Hellman Schlüsselaustausch)
- **Schlüssel Transport** (z.B. mit RSA) ohne zuvor ein Geheimnis (Schlüssel) ausgetauscht zu haben.
- **Digitale Signatur** (z.B. RSA, DSA oder ECDSA)
- **Verschlüsselung**
  Nachteil: Verschlüsselung großer Datenmengen mit asymmetrischen Verfahren ist sehr aufwendig und ca. 1000 mal langsamer als mit symmetrischen Algorithmen!


# Einführung in die asymmetrische Kryptografie

## Hybride Verfahren

In der Praxis verwendet man **hybride Systeme**, welche aus asymmetrischen und symmetrischen Algorithmen bestehen:

- **Schlüsselaustausch** (für symmetrische Verfahren) und **digitale Signaturen** werden mit den (langsamen) **asymmetrischen** Algorithmen durchgeführt.

- **Verschlüsselung** von Daten wird mit den (schnellen) **Block- oder Strom-Chiffren** durchgeführt.


# Einführung in die asymmetrische Kryptografie

Hybrides Verfahren

Alice

Bob

1) Asymmetrischer Algorithmus:

$y_1 = e_{K_{pubB}}(K)$ $\xrightarrow{y_1}$ $K = d_{K_{privB}}(y_1)$

2) Symm. Algorithmus:

$y_2 = e_K(x)$ $\xrightarrow{y_2}$ $x = d_K(y_2)$

**Schlüssel-Austausch**

**Verschlüsselung von Daten**

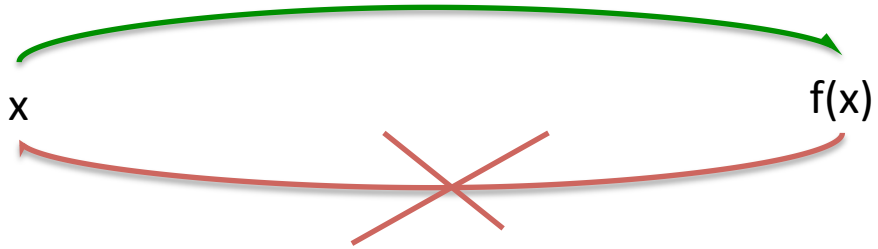
# Übersicht


- Wiederholung zu symmetrischer Kryptografie
- Prinzip der asymmetrischen Kryptografie
- Praktische Aspekte der asymmetrischen Kryptografie
- **Praxisrelevante asymmetrische Verfahren**
- Hintergrund: Zahlentheorie für asymmetrische Verfahren


# **Einführung in die asymmetrische Kryptografie**
## Aufbau asymmetrischer Verfahren

Asymmetrische Verfahren basieren auf einer **„Einwegfunktion“** ***f()***:

- Berechnung von $y = f(x)$ ist technisch einfach




- Berechnung von $x = f^{-1}(y)$ ist technisch unmöglich


# Einführung in die asymmetrische Kryptografie

## Aufbau asymmetrischer Verfahren

Einwegfunktionen basieren auf **mathematisch harten Problemen**.

- Es gibt drei bekannte Familien von harten Problemen:
  - **Faktorisierung ganzer Zahlen** (RSA, ...):
    Gegeben sei eine zusammengesetzte ganze Zahl *n*, finde deren Primfaktoren
    (Multiplikation von zwei Primzahlen ist einfach!)
  - **Diskreter Logarithmus** (Diffie-Hellman, Elgamal, DSA, …):
    Gegeben *a, y* und *m,* finde *x* mit $a^x = y \bmod m$
    (Potenzieren $a^x$ ist einfach!)
  - **Elliptische Kurven (EC)** (ECDH, ECDSA): Verallgemeinerung des diskreten Logarithmus

Anmerkung: Man nimmt an, dass diese Probleme schwierig sind, konnte es aber bisher nicht beweisen.


# Einführung in die asymmetrische Kryptografie

## Schlüssellängen und Sicherheitsniveau

| *symmetrisch* | *ECC* | *RSA, DL* | *Bemerkung* |
|---|---|---|---|
| 64 Bit | 128 Bit | ≈ 700 Bit | Nur kurze Sicherheit (mit Aufwand zu brechen) |
| 80 Bit | 160 Bit | ≈ 1024 Bit | Mittlere Sicherheit (ausgenommen Angriffe großer Behörden) |
| 128 Bit | 256 Bit | ≈ 2048 - 3072 Bit | Lange Sicherheit (ohne Quanten Computer) |

- Die genaue Komplexität von Angriffen auf RSA (Faktorisierung) und DL (Index-Calculus) ist schwer zu bestimmen.
- Die Existenz von Quantencomputern würde das Ende für ECC, RSA & DL bedeuten.
- http://www.keylength.com bietet eine gute Übersicht über aktuelle öffentliche Empfehlungen für Schlüssellängen

# Übersicht


- Wiederholung zu symmetrischer Kryptografie
- Prinzip der asymmetrischen Kryptografie
- Praktische Aspekte der asymmetrischen Kryptografie
- Praxisrelevante asymmetrische Verfahren
- **Hintergrund: Zahlentheorie für asymmetrische Verfahren**


# Zahlentheorie für asymmetrische Verfahren

## Euklidischer Algorithmus (1)

- Motivation: Berechnung des **größten gemeinsamen Teilers ggT($r_0$, $r_1$)** von zwei ganzen Zahlen $r_0$ und $r_1$

- ggT ist **einfach für kleine Zahlen**:
  1. Faktorisierung von $r_0$ und $r_1$
  2. ggT = größter gemeinsamer Faktor

- Beispiel:
  $r_0 = 84 = 2 \cdot 2 \cdot 3 \cdot 7$
  $r_1 = 30 = 2 \cdot 3 \cdot 5$

  → Der ggT ist das Produkt aller gemeinsamen Primfaktoren, d.h.
  $2 \cdot 3 = 6 = ggT\,(30,84)$

- **Aber:** Faktorisierung ist kompliziert (oder gar unmöglich) für große Zahlen


# Zahlentheorie für asymmetrische Verfahren

## Euklidischer Algorithmus (2)

- Beobachtung: $\boldsymbol{ggT\,(r_0, r_1) = ggT\,(r_0 - r_1, r_1)}$

→ Idee:

  - **Reduktion** des ursprünglichen ggT auf ein ggT mit kleineren Zahlen
  - **Rekursive Anwendung**, bis der engültige $\boldsymbol{ggT(r_i,0) = r_i}$ gefunden ist!

Beispiel: $ggT(r_0, r_1)$ für $r_0 = 27$ und $r_1 = 21$

| 21 | 6 |
|---|---|

$$ggT(27, 21) = ggT(1 \cdot 21+6, 21) = ggT(21, 6)$$

| 6 | 6 | 6 | 3 |
|---|---|---|---|

$$ggT(21, 6) = ggT(3 \cdot 6+3, 6) = ggT(6, 3)$$

| 3 | 3 |
|---|---|

$$ggT(6, 3) = ggT(2 \cdot 3+0, 3) = ggT(3, 0) = 3$$

- Anmerkung: Diese Methode ist sehr effizient bei großen Zahlen (Komplexität wächst **linear** mit der Anzahl der Bit)


# Zahlentheorie für asymmetrische Verfahren

## Erweiterter Euklidischer Algorithmus – EEA (1)

- Erweiterung des euklidischen Algorithmus zur Berechnung der **modularen Inversen** von $r_1$ mod $r_0$
- EEA berechnet $s$,$t$, und den gcd : $\gcd(r_0, r_1) = s \cdot r_0 + t \cdot r_1$
- Verwende die Gleichung **mod $r_0$**

$$s \cdot r_0 + t \cdot r_1 = 1$$
$$s \cdot 0 + t \cdot r_1 \equiv 1 \bmod r_0$$
$$r_1 \cdot t \equiv 1 \bmod r_0$$

→ Vergleich mit Def. der Inversen: **$t$ ist die Inverse von $r_1$ mod $r_0$**

- Anmerkung: Notwendige Bedingung zur Existenz der Inversen: $ggT(r_0, r_1) = 1$

- **Rekursive Formel** zur Berechnung von $s$ und $t$ in jedem Schritt → „magische Tabelle“ für $r$, $s$, $t$ und einem Quotienten $q$ zur Berechnung der Inversen mit Papier und Stift


# Zahlentheorie für asymmetrische Verfahren

## Erweiterter Euklidischer Algorithmus – EEA (2)

**Beispiel:**

- Berechnen Sie die modulare Inverse von 12 mod 67:
- Aus der Tabelle folgt: $-5 \cdot 67 + 28 \cdot 12 = 1$
- Daher ist **28 die Inverse** von 12 mod 67
- Check:

$$28 \cdot 12 = 336 \equiv 1 \bmod 67$$ ✓

| $i$ | $q_{i-1}$ | $r_i$ | $s_i$ | $t_i$ |
|---|---|---|---|---|
| 2 | 5 | 7 | 1 | -5 |
| 3 | 1 | 5 | -1 | 6 |
| 4 | 1 | 2 | 2 | -11 |
| 5 | 2 | 1 | -5 | **28** |


# Zahlentheorie für asymmetrische Verfahren

## Erweiterter Euklidischer Algorithmus – EEA (5)

**Erweiterter euklidischer Algorithmus (EEA)**

**Eingang**: Positive ganze Zahl $r_0$ und $r_1$ mit $r_0 > r_1$

**Ausgang**: $\text{ggT}(r_0, r_1)$ sowie $s$ und $t$ mit $\text{ggT}(r_0, r_1) = s \cdot r_0 + t \cdot r_1$.

**Initialisierung**:

$s_0 = 1 \quad t_0 = 0$

$s_1 = 0 \quad t_1 = 1$

$i = 1$

**Algorithmus**:

1 DO

1.1 $i = i + 1$

1.2 $r_i = r_{i-2} \bmod r_{i-1}$

1.3 $q_{i-1} = (r_{i-2} - r_i)/r_{i-1}$

1.4 $s_i = s_{i-2} - q_{i-1} \cdot s_{i-1}$

1.5 $t_i = t_{i-2} - q_{i-1} \cdot t_{i-1}$

WHILE $r_i \neq 0$

2 RETURN

$\text{ggT}(r_0, r_1) = r_{i-1}$

$s = s_{i-1}$

$t = t_{i-1}$


# Zahlentheorie für asymmetrische Verfahren

## Eulers Phi-Funktion (1)

- Motivation: Weiteres Problem, wichtig für asymmetrische Verfahren, wie z.B. RSA:
  Geben sei eine Menge an *m* ganzen Zahlen {0, 1, 2, …, *m* -1},
  **Wieviele** Zahlen der Menge sind **relativ prim zu *m*** ?
- Antwort: **Eulers Phi-Funktion *Φ(m)***
- **Beispiele** für die Mengen {0,1,2,3,4,5} (*m*=6) und {0,1,2,3,4} (*m*=5)

$$\gcd(0,6) = 6$$
$$\gcd(1,6) = 1 \leftarrow$$
$$\gcd(2,6) = 2$$
$$\gcd(3,6) = 3$$
$$\gcd(4,6) = 2$$
$$\gcd(5,6) = 1 \leftarrow$$

→ 1 und 5 teilerfremd zu *m*=6,
daher ist ***Φ*(6) = 2**

$$\gcd(0,5) = 5$$
$$\gcd(1,5) = 1 \leftarrow$$
$$\gcd(2,5) = 1 \leftarrow$$
$$\gcd(3,5) = 1 \leftarrow$$
$$\gcd(4,5) = 1 \leftarrow$$

→ ***Φ*(5) = 4**

- Eine ggT-Berechnung ist **sehr langsam für große *m*.**


# Zahlentheorie für asymmetrische Verfahren

## Eulers Phi-Funktion (2)

- **Wenn Faktorisierung von** $m$ bekannt: $m = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_n^{e_n}$
  (wobei $p_i$ Primzahlen und $e_i$ positive ganze Zahlen sind)
  **berechne** Phi durch

$$\Phi(m) = \prod_{i=1}^{n} (p_i^{e_i} - p_i^{e_i - 1})$$

- Berechnung einfach für $e_i = 1$, d.h., $m = p \cdot q$
  → $\Phi(m) = (p\text{-}1) \cdot (q\text{-}1)$

- **Beispiel:** $m = 899 = 29 \cdot 31$
  $\boldsymbol{\Phi(899)} = (29\text{-}1) \cdot (31\text{-}1) = 28 \cdot 30$ **= 840**

- **Anmerkung:** Berechnung von $\Phi(m)$ einfach, wenn **Faktorisierung von *m* bekannt**
  (ansonsten ist Berechnung von $\Phi(m)$ für große Zahlen praktisch unmöglich)


## Zahlentheorie für asymmetrische Verfahren
Kleiner Satz von Fermat

- Gegeben **Primzahl *p*** und ganze Zahl ***a***: $a^p \equiv a \pmod{p}$
  Kann auch als $a^{p-1} \equiv 1 \pmod{p}$ geschrieben werden

- **Anwendung:** Berechnung von **modularen Inversen**, wenn *p* prim:
  $$a \cdot a^{p-2} \equiv 1 \pmod{p}$$
  → Die modulare Inverse von *a* mod *p* ist: $a^{-1} \equiv a^{p-2} \pmod{p}$

**Beispiel:** *a* = 2, *p* = 7

$$a^{p-2} = 2^5 = 32 \equiv 4 \bmod 7$$

$$\text{verify: } 2 \cdot 4 \equiv 1 \bmod 7$$ ✓

- Der kleine Satz von Fermat gilt nur **modulo einer Primzahl *p***


# Zahlentheorie für asymmetrische Verfahren

## Satz von Euler

- **Verallgemeinerung** des kleinen Satzes von Fermat für zwei **beliebige, teilerfremde ganze Zahlen *a*** und ***m*** : $a^{\Phi(m)} \equiv 1 \pmod{m}$

- **Beispiel**: $m$=12, $a$=5
  1. Berechne Eulers Phi Function

$$\Phi(12) = \Phi(2^2 \cdot 3) = (2^2 - 2^1)(3^1 - 3^0) = (4-2)(3-1) = 4$$

  2. Prüfe Satz von Euler

$$5^{\Phi(12)} = 5^4 = 25^2 = 625 \equiv 1 \bmod 12$$

- Kleiner Satz von Fermat = Sonderfall von Eulers Satz für Primzahl **p**:

$$\Phi(p) = (p^1 - p^0) = p - 1$$

→ Fermat: $a^{\Phi(p)} = a^{p-1} \equiv 1 \pmod{p}$

# Lessons Learned


- Asymmetrische Algorithmen haben **Einsatzgebiete, die symmetrische Chiffren nicht haben**, im Besondern digitale Signaturen und Schlüsselaustausch.
- Asymmetrische Algorithmen sind **rechenintensiv** (=langsam) und deswegen **nicht geeignet für die Verschlüsselung großer Datenmengen**.
- Die meisten modernen Protokolle sind so genannte **Hybridprotokolle**, welche sowohl symmetrische als auch asymmetrische Algorithmen verwenden.
- Es gibt beträchtlich weniger etablierte asymmetrische Algorithmen als symmetrische Algorithmen.
- Der **erweiterte Euklidische Algorithmus** erlaubt uns die **effiziente Berechnung modularer Inverser**, was bedeutend für asymmetrische Verfahren ist
- **Eulers Phi-Funktion gibt uns die Anzahl der Elemente kleiner einer ganzen Zahl n, welche relativ prim zu n sind.** Dies ist bedeutend für das RSA Verfahren.