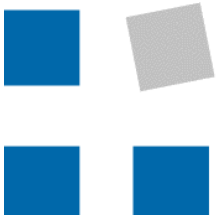


HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

Angewandte Kryptographie

3. Asymmetrische Verfahren

Netzwerksicherheit WS 2001/2002

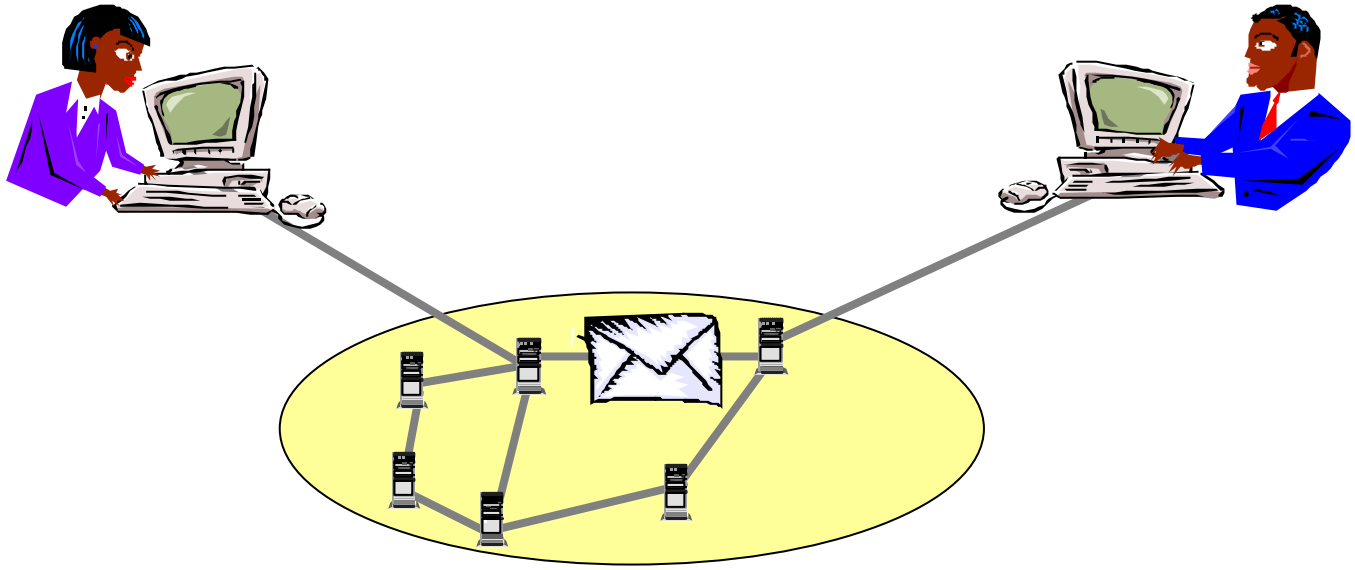
Jean-Marc Piveteau

1. Die „Public Key“-Revolution

Symmetrische Kryptographie: Die Bilanz

Aber: Key-Management nicht optimal gelöst

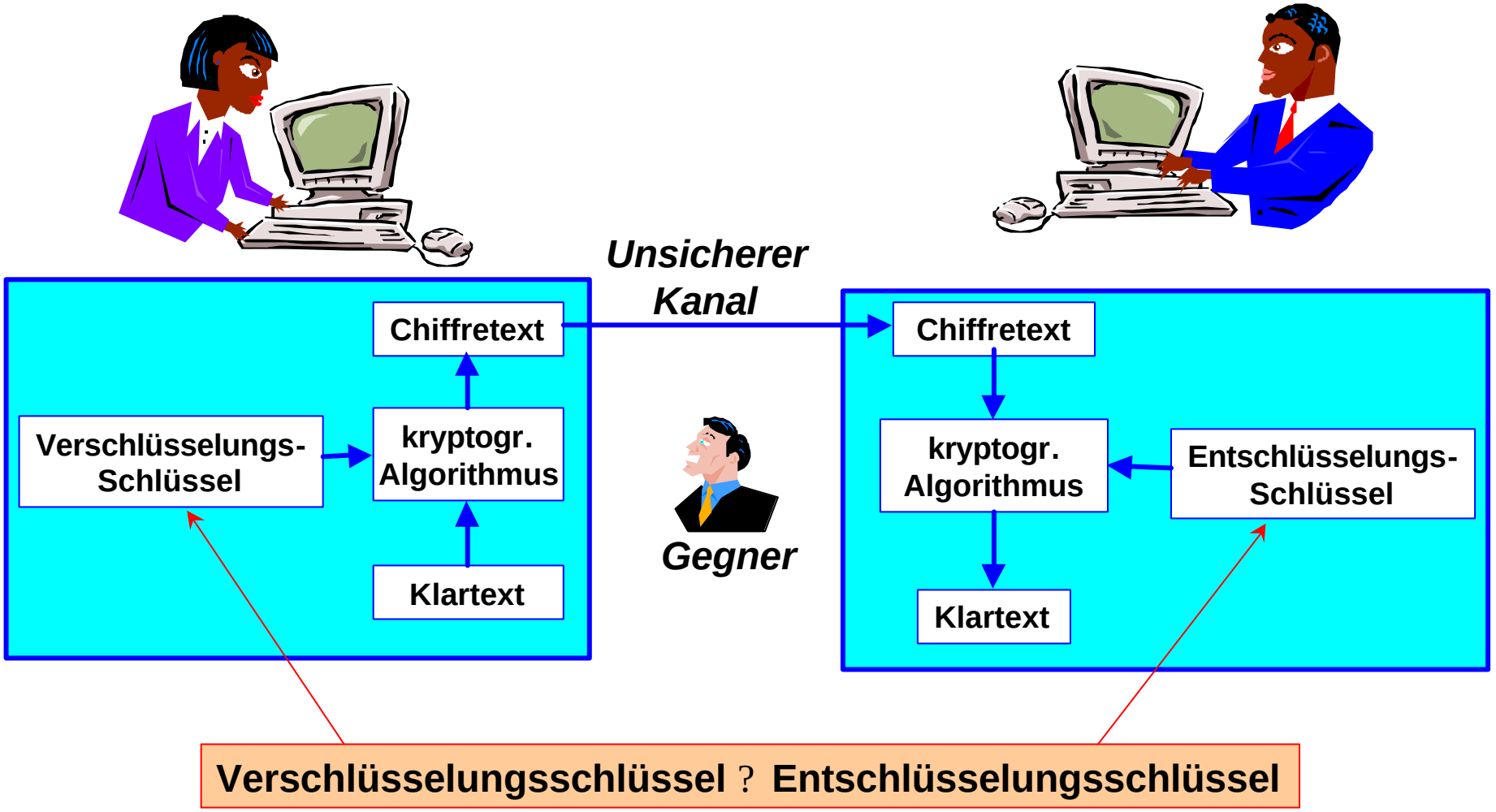
- **Vertraulichkeit**
- **Daten-Integrität**
- **Daten-Authentisierung**
- **Benutzer-Authentisierung**
- **Verbindlichkeit des Senders**



The Big Bang

W. Diffie, M. Hellman: *New Directions in cryptography*,
IEEE Transaction on Information Theory, **22** (1976), 644 - 654

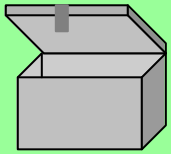
Asymmetrische-Verschlüsselung



Public Key-Verschlüsselung: Illustrative Analogie

Phase 1: Schlüsselvereinbarung

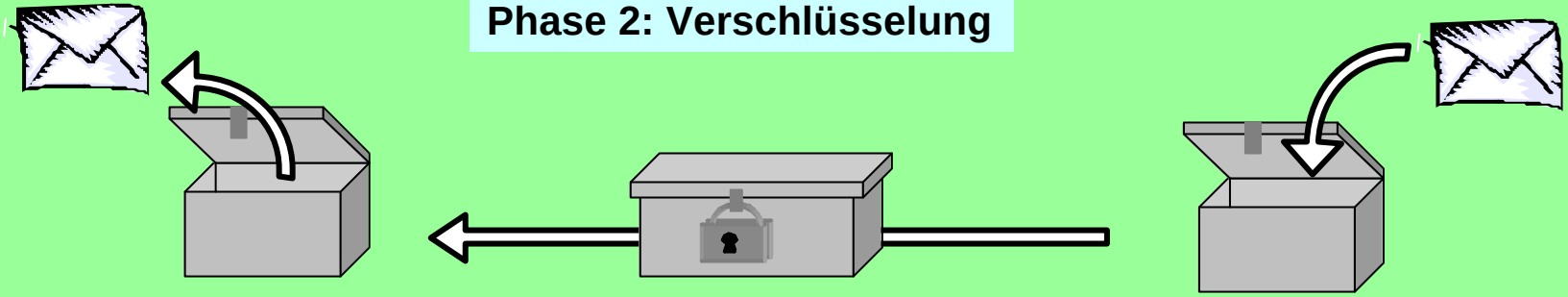
Private Key von Alice



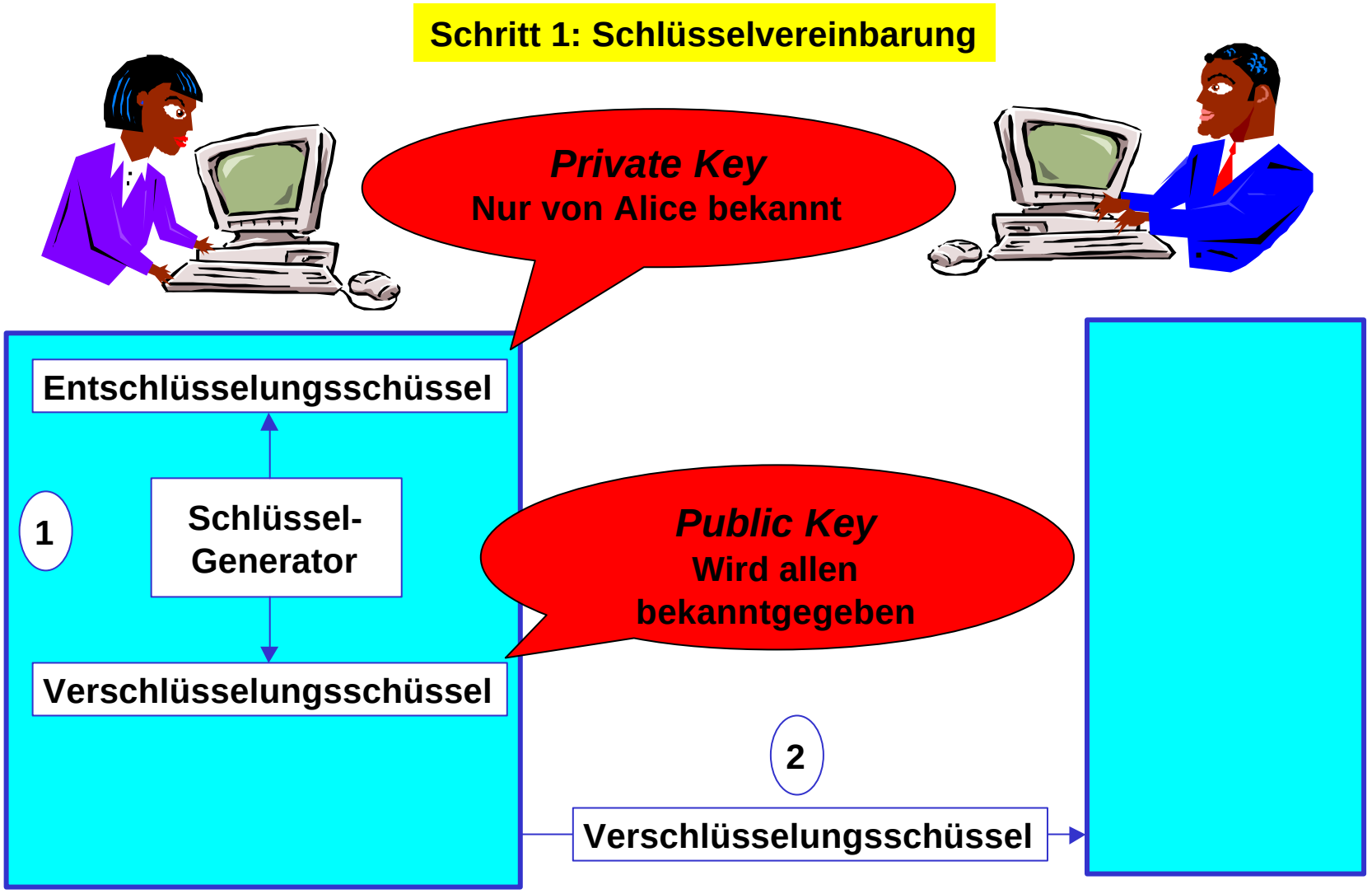
Kryptographischer Algorithmus

Public Key von Alice

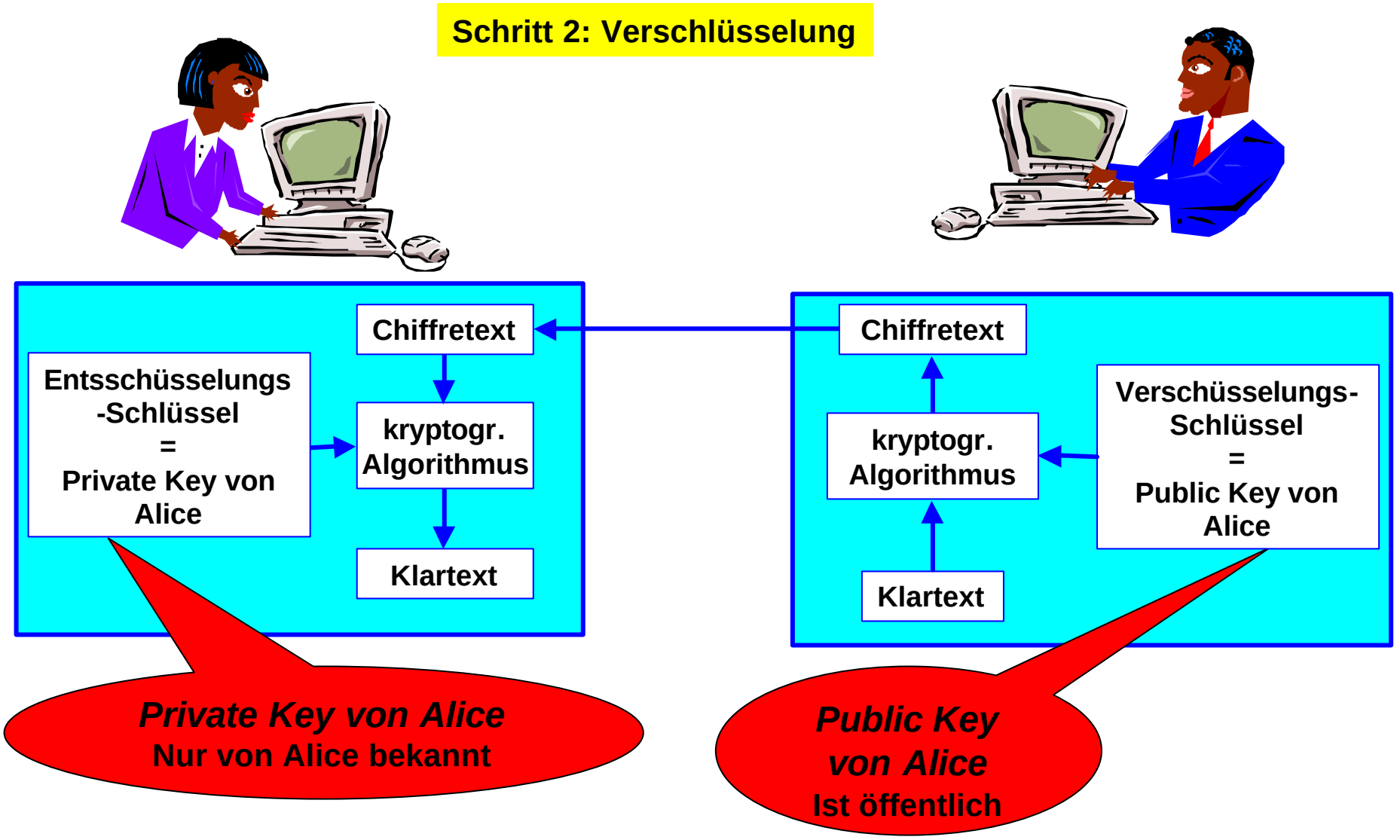
Phase 2: Verschlüsselung



Prinzip der Public-Key-Verschlüsselung (1/2)



Prinzip der Public-Key-Verschlüsselung (2/2)



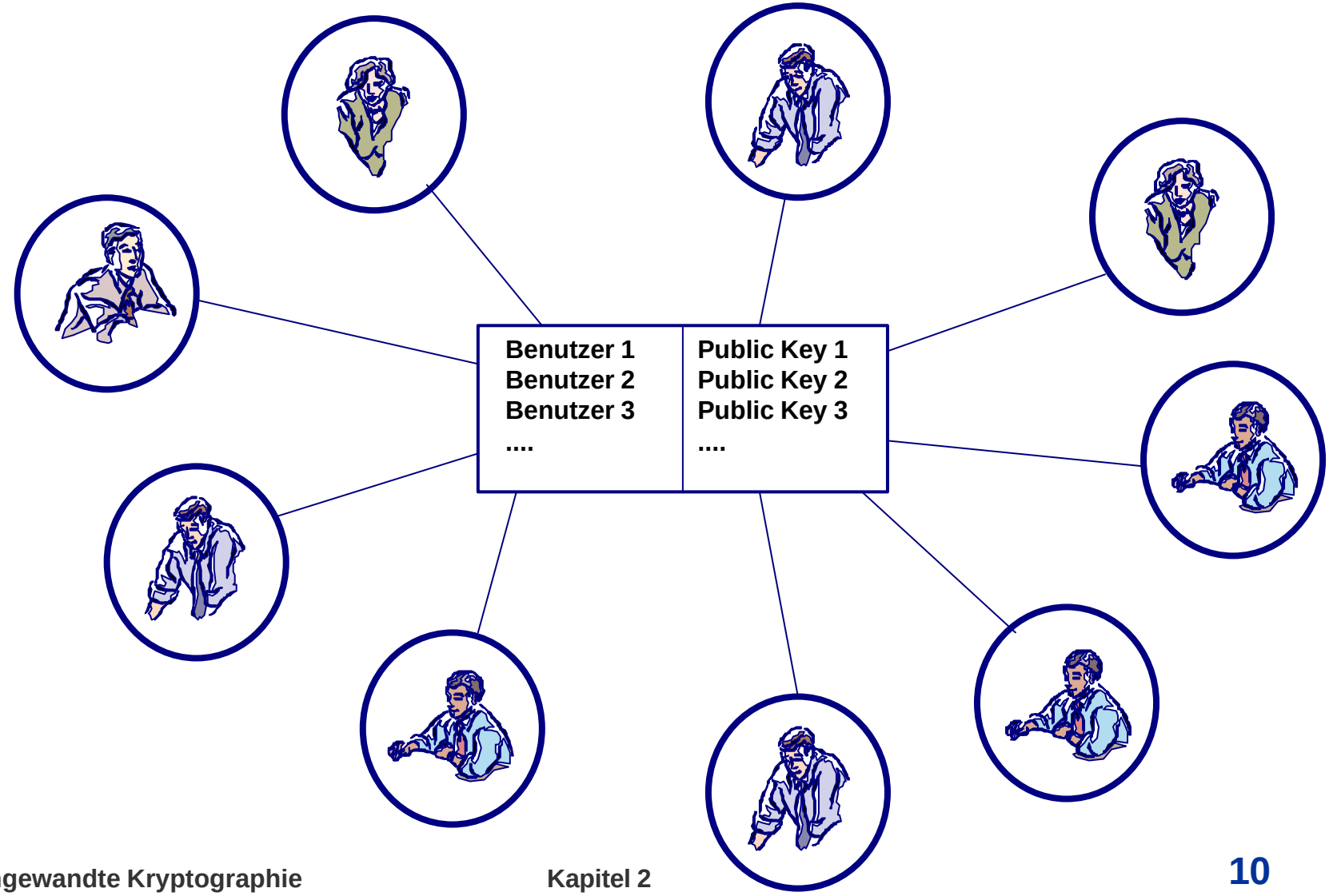
Sicherheit von Public Key-Verfahren

Die Sicherheit basiert auf die Annahme:

Es gibt (mathematische) Methoden, welche erlauben zwei Schlüssel, den Private Key (den privaten Schlüssel) und den Public Key (den öffentlichen Schlüssel) mit folgender Eigenschaft zu generieren:

- Der Public Key ist der Verschlüsselungsschlüssel, d.h. wer den Public Key kennt, kann verschüsseln
- Der Private Key ist der Entschlüsselungsschlüssel, d.h. wer den Private Key kennt, kann entschlüsseln
- Jemand der nur der Public Key kennt, kann die verschlüsselten Nachrichten nicht entschlüsseln. Insbesondere ist es praktisch nicht möglich, den Private Key zu berechnen, wenn nur der Public Key bekannt ist

Key-Management



2. RSA

RSA (Rivest-Shamir-Adleman)

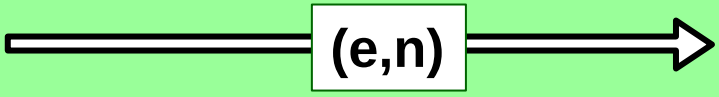
- **1978: Von R. Rivest, A. Shamir und L. Adleman erfunden**
- **Die Sicherheit von RSA basiert auf eine mathematische Annahme**
- **Im Gegensatz zu einem symmetrischen Schlüssel ist ein RSA-Schlüssel keine Zufallsfolge von Bits**
- **Sicherheit von RSA: Faust-Regel für die Schlüssel-Länge:**

  64 Bit symmetrisch ~ 512 Bit RSA
  128 Bit symmetrisch ~ 1024 Bit RSA
  256 Bit symmetrisch ~ 2048 Bit RSA

RSA (1/2)

Phase 1: Schlüsselvereinbarung

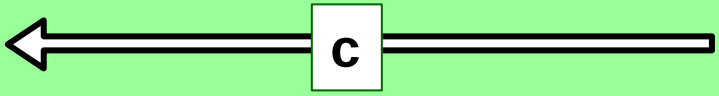
1. **Wähle zwei grosse Primzahlen p und q**
2. **Berechne n = p.q**
3. **Wähle eine kleine Zahl e mit ggT(e, ? (n)) = 1, wobei ? (n) = (p-1)(q-1)**
4. **Berechne d mit d·e=1 mod ? (n) mit Hilfe des erweiterten Euklidischen Algorithmus**
5. **Halte d als geheimen Schlüssel und gib (e,n) als öffentlichen Schlüssel bekannt**



RSA (2/2)

Phase 2: Verschlüsselung

1. **Nachricht: m ? {0,1, ..., n-1}**
2. **Verschlüsselte Nachricht:**
   $c = m^e \bmod N$



3. **Entschlüsselung:**
   $m = c^d \bmod N$

RSA-Problem

Gegeben:

- **n: Produkt von zwei grossen Primzahlen**
- **e: ganze Zahl mit ggT(e,? (n)) =1**
- **c ? {0,1, ..., n-1}**

Berechne: m mit $c = m^e \bmod N$ zu berechnen

RSA-Sicherheitsannahme

Es gibt keinen effizienten Algorithmus, um das RSA-Problem zu lösen

RSA und Faktorisierung

Wenn die Faktorisierung von n bekannt ist, kann m mit $c = m^e \bmod N$ sehr effizient berechnet werden

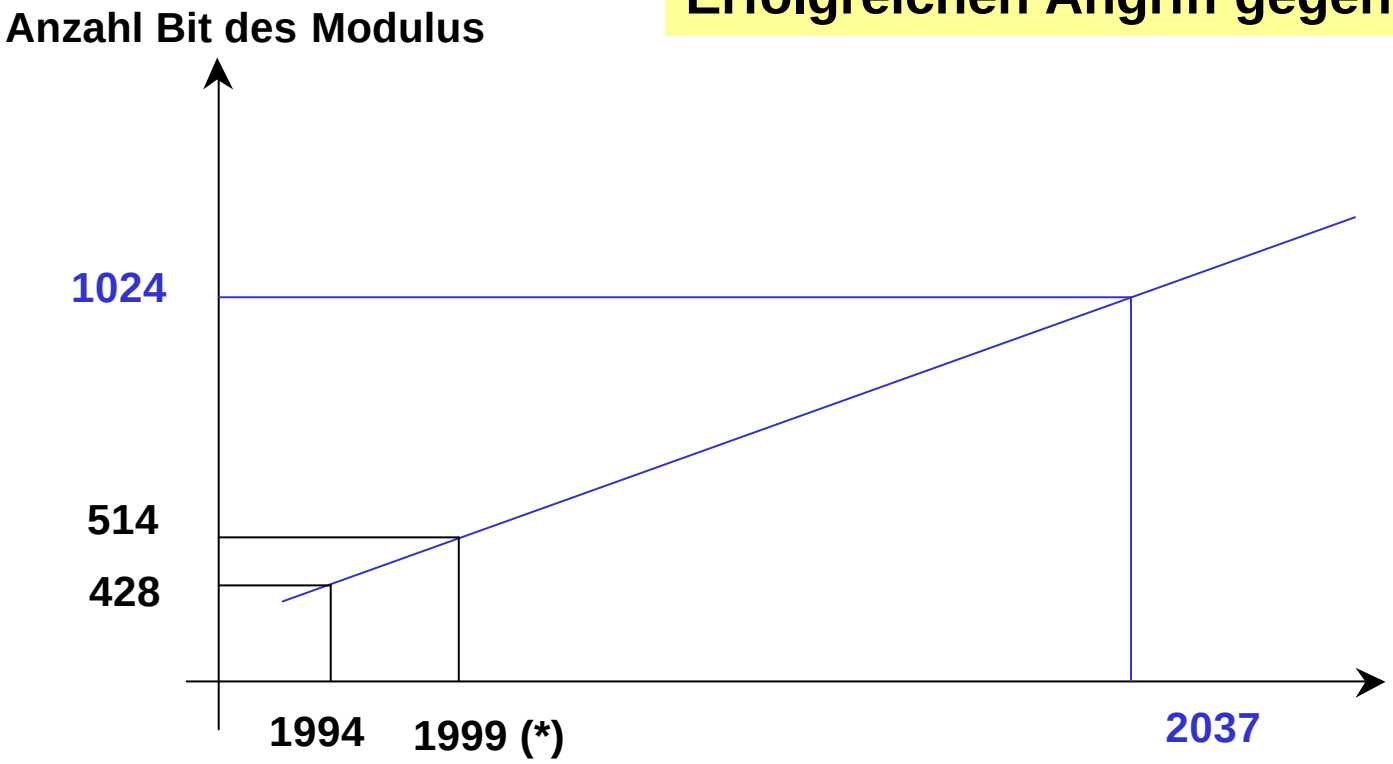
Faktorisierungsproblem für RSA

Berechne die Primfaktorzerlegung einer gegebenen ganzen Zahl N, welche das Produkt von zwei grossen Primzahlen ist

Algorithmus	**Anzahl von Operationen für eine Zahl von n Bit**
„Trial Division“	$< C\,(2^{n-1} + n)$ für eine Konstante C
Zahlkörpersieb	$< C\, e^{1.7\, n^{1/3}\, (\log n + 0.366)^{2/3}}$

RSA-Sicherheit

Erfolgreichen Angriff gegen RSA



(*) 292 PCs und Workstations, Taktfrequenz 400 MHz, vier Monate Rechenzeit