HANDOUT ZUM VORTRAG
"ASYMMETRISCHE CHIFFRIERVERFAHREN – RSA"

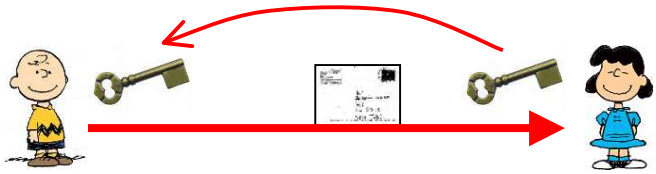
IM VORTRAG ERWÄHNTE CHIFFRIERVERFAHREN:

- Symmetrische Verfahren
  - monoalphabetische Verfahren
    Beispiel: • ***Cäsar***-Verfahren (extrem unsicher)
  - polyalphabetische Verfahren
    Beispiel: • ***Vigenère***-Verfahren (mit heutigen Rechnern genauso unsicher),
    • ***One-Time-Pad*** (absolut sicher bei zufälligem Schlüssel)
- Asymmetrische Verfahren
  Beispiel: • ***RSA***-Verfahren (absolut sicher bei Schlüssellänge ≥ 1024 Bit)

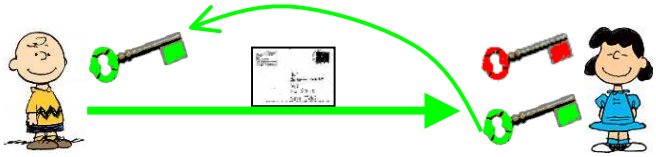
GRUNDPRINZIPIEN BEIDER VERFAHREN

(**rot markiert: unsicherer Bereich/Angriffsmöglichkeit** / **grün markiert: sicherer Bereich**)

symmetrische Chiffrierung:



asymmetrische Chiffrierung:



GRUNDIDEE DER ASYMMETRISCHEN CHIFFRIERUNG IST EINE EINWEGFUNKTION MIT FALLTÜR:

Eine Funktion $f: X \to Y$ heißt Einwegfunktion mit Falltür, wenn
- $y = f(x)$ mit einem **Algorithmus E** leicht zu berechnen ist,
- $x = f^{-1}(y)$ mit einem **Algorithmus D** leicht zu berechnen ist, aber
- die Bestimmung des Algorithmus D aus E nur sehr schwer möglich ist.

PARAMETER FÜR DIE ALGORITHMEN D UND E:

- Wähle als Falltürfunktion $f(p,q) = p \cdot q$, wobei p, q **große** Primzahlen sind
- Bilde $n = p \cdot q$. $\Rightarrow$ Rechnen im Restklassenring Z_n
- $\varphi(n) = (p-1) \cdot (q-1)$ sei die Eulersche φ-Funktion, dann gilt
- $m^{\varphi(n)} \bmod n = 1$ für m und n teilerfremd (*Satz von Euler*)
- Wähle e teilerfremd zu $\varphi(n)$, dann gibt es ein d mit $e \cdot d \bmod \varphi(n) = 1$
  (Berechnung mit Hilfe des erweiterten euklidischen Algorithmus)

ALGORITHMUS E:

Berechne $C = M^e \bmod n$

ALGORITHMUS D:

Berechne $M = C^d \bmod n$

BEWEIS DES RSA-VERFAHRENS:

$$
\begin{aligned}
C^d \bmod n &= (M^e \bmod n)^d \bmod n \\
&= M^{e \cdot d} \bmod n \\
&= M^{k \cdot j(n) + 1} \bmod n \\
&= M^{k \cdot j(n)} \cdot M \bmod n \\
&= (M^{j(n)})^k \cdot M \bmod n \\
&= 1^k \cdot M \bmod n \\
&= M
\end{aligned}
$$

FAZIT:

RSA nutzt aus, dass die Zerlegung einer Zahl in Primfaktoren ein Problem der Klasse NP ist. Würde man einen Algorithmus dieses Problems finden, der in der Klasse P läge, so wäre RSA nicht mehr sicher (und man hätte die seit langem erforschte Frage P ≠ NP beantwortet).

Weiterführendes Material, etc. auf www.gymnasium-odenthal.de/download/rsa/index.htm