Memo: RSA auf einen Blick

Alice | **Kommunikation zwischen Alice und Bob** | **Bob**

Alice wählt zwei verschiedene Primzahlen p und q, z.B. $p = 5$ und $q = 11$. Alice berechnet

$$n = pq = 55$$

sowie

$$m = (q-1)(p-1) = 40 \ .$$

Alice wählt eine Zahl a, die zu m teilerfremd ist, z.B.

$$a = 7 \ .$$

Alice gibt die Zahlen n und a als ihren öffentlichen Schlüssel bekannt:

$$\begin{aligned} n &= 55 \\ a &= 7 \end{aligned}$$

Bobs Nachricht ist eine Zahl x, die kleiner als n ist, z.B. $x = 8$. Bob verschlüsselt sie gemäß

$$\begin{aligned} y &= x^a \bmod n \\ &= 8^7 \bmod 55 = 2 \end{aligned}$$

Bob sendet y an Alice:

$$y = 2 \ .$$

Alice berechnet aus a und m das multiplikative Inverse von a modulo m:

$$\begin{aligned} b &= a^{-1} \bmod m \\ &= 7^{-1} \bmod 40 = 23 \ . \end{aligned}$$

Sie kann Bobs Nachricht mit der Formel

$$x = y^b \bmod n = 2^{23} \bmod 55 = 8$$

wieder entschlüsseln.

Bemerkung 3.2 *Details zu den in diesem Schaubild enthaltenen Rechnungen:*

$$\begin{aligned} 8^7 \bmod 55 &= 8^2 \cdot 8^2 \cdot 8^2 \cdot 8 \bmod 55 \\ &= 9 \cdot 9 \cdot 9 \cdot 8 \bmod 55 \\ &= 81 \cdot 72 \bmod 55 \\ &= 26 \cdot 17 \bmod 55 \\ &= 442 \bmod 55 \\ &= 8 \cdot 55 + 2 \bmod 55 \\ &= 2 \end{aligned}$$

Ferner gilt

$$7^{-1} \bmod 40 = 23$$

wegen

$$7 \cdot 23 \bmod 40 = 161 \bmod 40 = 4 \cdot 40 + 1 \bmod 40 = 1 \ .$$

Die multiplikative Inverse modulo 40 zur Zahl 7 ist also die Zahl 23.

Bemerkung 3.3 (RSA in der Praxis:) *In der Praxis wählt man die Primzahlen p und q sehr groß. In der Regel haben beide mehrere Hundert Stellen. Die zugehörigen Berechnungen werden mit Hilfe von Computerprogrammen erledigt.*

Ist die zu verschlüsselnde Nachricht größer als n, so teilt man die Nachricht in mehrere kleinere Nachrichten und verschlüsselt und sendet diese unabhängig voneinander.

3.4 Warum ist die RSA-Verschlüsselung sicher?

Der öffentliche Schlüssel von Alice besteht aus den beiden Zahlen n und a und ist allgemein bekannt. Zum Entschlüsseln benötigt Alice lediglich die Zahl m. Hiermit kann sie b (das Inverse von a modulo m) bestimmen und so durch modulares Potenzieren die Nachricht entschlüsseln.

Wenn es also gelingen würde, aus der Kenntnis von n die Zahl m zu ermitteln, wäre RSA nicht sicher. Wegen

$$n = pq$$

und

$$m = (p-1)(q-1)$$

wäre dies leicht möglich, wenn es gelänge, n in seine beiden Primfaktoren p und q zu zerlegen. Bisher ist jedoch kein Verfahren bekannt, mit dem dies für große Primzahlen p und q in weniger als vielen Monaten gelingt. Daher kann RSA für hinreichend große Primzahlen als eine sehr sichere Art der Verschlüsselung betrachtet werden.

Sollte jedoch eine Methode entdeckt werden, mit der man aus n seine beiden Primfaktoren p und q oder $m = (p-1)(q-1)$ relativ einfach berechnen könnte, wäre RSA kein sicheres Verschlüsselungsverfahren mehr.

3.5 Warum funktioniert die RSA-Verschlüsselung? – Die Mathematik hinter RSA

In Abschnitt 3.3 haben wir die Funktionsweise von RSA beschrieben. Wir fassen dies noch einmal kurz zusammen:

- Alice wählt zwei verschiedene Primzahlen p und q und berechnet

  $$n = pq$$

  sowie

  $$m = (q-1)(p-1) \ .$$

  Ferner wählt Alice eine Zahl a, die zu m teilerfremd ist.

- a und n bilden den öffentlichen Schlüssel von Alice.

- Bobs Nachricht ist eine Zahl x, die kleiner als n ist. Er verschlüsselt sie nach der Formel

  $$y = x^a \bmod n \ .$$

- Zur Entschlüsselung berechnet Alice aus a und m zunächst das multiplikative Inverse von a modulo m:

  $$b = a^{-1} \bmod m$$

  und entschlüsselt dann Bobs Nachricht mittels

  $$x = y^b \bmod n \ . \tag{1}$$

Zu verstehen, warum RSA funktioniert, bedeutet mathematisch nichts anderes als die Gültigkeit der Gleichung (1) zu beweisen, also zu zeigen, dass $y^b \bmod n$ gerade wieder x ergibt.

Für diesen Beweis benötigen wir den Satz von Euler-Fermat:

Satz 3.1 (Euler-Fermat) *Sind a und n teilerfremde Zahlen mit $1 < a < n$, so gilt*

$$a^{\varphi(n)} \equiv 1 \bmod n \ . \tag{2}$$

Hierbei ist $\varphi(n)$ die Anzahl der zu n teilerfremden Zahlen[1]*.*

Diesen Satz wollen wir hier nicht beweisen. Wir werden ihn aber gleich benutzen, um Gleichung (1) für $x \neq p$ und $x \neq q$ zu beweisen. (Für $x = q$ und $x = p$ kann Gleichung (1) mit Hilfe des Chinesischen Restsatzes weitgehend analog bewiesen werden. Hierzu verweisen wir auf [5].)

Bevor wir den Beweis durchführen, vermerken wir noch, dass für Primzahlen p, q und $n = pq$ gilt

$$\varphi(p) = p-1 \qquad \varphi(q) = q-1 \qquad \text{sowie} \qquad \varphi(n) = \varphi(pq) = (p-1)(q-1) \ . \tag{3}$$

[1] Zur Erinnerung: Zwei Zahlen sind teilerfremd, wenn ihr größter gemeinsamer Teiler 1 ist.

Hiermit folgt:

$$
\begin{aligned}
y^b \bmod n &= (x^a \bmod n)^b \bmod n \\
&= (x^a)^b \bmod n \\
&= x^{ab} \bmod n && b \text{ ist das multiplikative Inverse zu } a, \text{ also } a \cdot b \bmod m \equiv 1 \ . \\
& && \text{D.h. es gibt } k \in \mathbb{Z} \text{ so dass } a \cdot b = k \cdot m + 1 \\
&= x^{k \cdot m + 1} \bmod n \\
&= x^{k \cdot m} \cdot x \bmod n \\
&= (x^m)^k \cdot x \bmod n \\
&= ((x^m)^k \bmod n) \cdot (x \bmod n) \bmod n \\
&= ((x^m) \bmod n)^k \cdot (x \bmod n) \bmod n \\
&= (1^k \bmod n) \cdot (x \bmod n) \bmod n && \text{wegen des Satzes von Euler-Fermat (sofern } x \text{ und } n \text{ teilerfremd sind), da } m = (p-1)(q-1) = \varphi(n) \text{ gesetzt wurde} \\
&= 1 \cdot x \bmod n \\
&= x && \text{wegen } x < n
\end{aligned}
$$

Damit ist Gleichung (1) bewiesen für $x \neq q$ und $x \neq p$ (um den Satz von Fermat-Euler anwenden zu können, müssen x und $n = pq$ als teilerfremd vorausgesetzt werden). Die RSA-Entschlüsselung liefert also wieder die Ausgangszahl.

4 Details zu Berechnungen beim RSA-Verfahren

Wollen Schüler/-innen den RSA-Algorithmus per Hand durchführen, ist es sinnvoll, dass sie zunächst entdecken, wie man das modulare Potenzieren möglichst einfach durchführen kann (vgl. hierzu Abschnitt 4.1).

Greift man hingegen auf Rechner zurück, muss man lediglich beachten, dass ganze Zahlen auf dem Rechner in manchen Programmiersprachen nur bis zu einer maximal darstellbaren Zahl unterstützt werden. Da beim Potenzieren recht schnell sehr große Zahlen auftauchen, sollte man die Modulo-Funktion daher auch mit unter den Exponenten ziehen.

Arbeitsblatt: RSA-Verfahren

Beschrifte das folgende Schaubild mit den Variablen, die im Memo *RSA auf einen Blick* verwendet wurden.