§11. Das RSA–Verfahren und andere Verfahren

Eine konkrete Realisierung eines Public–Key–Kryptosystems ist das sog. **RSA–Verfahren**, das im Jahre 1978 von den drei Wissenschaftlern

Ron L.Rivest,

Adi Shamir und

Leonard Aldeman

veröffentlicht wurde. Der Titel ihrer Arbeit lautete "A Method for Obtaining Digital Signatures and Public–Key". Grundlage dafür ist der folgende Satz:

(11.1) SATZ: Die natürliche Zahl $n \in \mathbb{N}$ sei das Produkt zweier verschiedener Primzahlen p und q. Ferner sei $e \in \mathbb{N}$ eine zu $\varphi(n)$ teilerfremde Zahl. Dann gilt:

a) Es gibt eine natürliche Zahl $d \in \mathbb{N}$ mit

$$(e \cdot d) \bmod \varphi(n) = 1$$

b) Sei $\mathcal{R}_n = \{0, 1, 2, \ldots, n-1\}$. Dann sind die Abbildungen $E, D : \mathcal{R}_n \longrightarrow \mathcal{R}_n$, die durch die Vorschriften

$$E(x) := x^e \bmod n \quad (x \in \mathcal{R}_n)$$

$$D(x) := x^d \bmod n \quad (x \in \mathcal{R}_n)$$

definiert sind, beide bijektiv und zueinander invers, d.h.

$$D \circ E \;=\; \mathrm{id}_{\mathcal{R}_n} \;=\; E \circ D\,.$$

(11.2) Das RSA–Kryptosystem $(\mathcal{R}_n, \mathcal{R}_n, \mathcal{K}, E, D)$

Die natürliche Zahl $n \in \mathbb{N}$ sei das Produkt zweier verschiedener Primzahlen. Es sollen Zahlen aus $\mathcal{R}_n = \{0, 1, 2, \ldots, n-1\}$ verschlüsselt werden. Um Texte zu verschlüsseln, müssen die Buchstaben erst durch Zahlen ersetzt werden.

Klartextalphabet: $\mathcal{R}_n$

Geheimtextalphabet: $\mathcal{R}_n$

Schlüsselmenge: $\mathcal{K} \;=\; \{\,(n,e)\,,\,(n,d)\,\}$ mit $e, d \in \mathbb{N}\,,\ (e \cdot d) \bmod \varphi(n) = 1$

Verschlüsselungsabbildung: $E : \mathcal{R}_n \longrightarrow \mathcal{R}_n\,,\ E(x) := x^e \bmod n \quad (x \in \mathcal{R}_n)$

Entschlüsselungsabbildung: $D : \mathcal{R}_n \longrightarrow \mathcal{R}_n\,,\ D(x) := x^d \bmod n \quad (x \in \mathcal{R}_n)$

Es gilt die Entschlüsselungsbedingung

$$D(E(x)) \;=\; x \quad (\text{für alle } x \in \mathcal{R}_n)\,.$$

(11.3) Die praktische Durchführung des RSA–Verfahrens

Für eine (große) Zahl $n \in \mathbb{N}$, die Produkt von zwei (großen) verschiedenen Primzahlen ist, wähle man Zahlen $d, e \in \mathbb{N}$ mit $1 < d, e < \varphi(n)$, für die

$$(e \cdot d) \bmod \varphi(n) \;=\; 1$$

gilt. Dazu wählt man e teilerfremd zu $\varphi(n)$ und berechnet d als Inverses von e modulo $\varphi(n)$ mit Hilfe des erweiterten euklidischen Algorithmus. Mit dem **öffentlichen Schlüssel** (n, e) wird dann eine Nachricht, die in Form einer Zahl $x < n$ vorliegt, zu

$$y := x^e \bmod n$$

verschlüsselt. Mit dem **geheimen Schlüssel** (n, d), der nur dem Empfänger bekannt ist, kann y durch

$$y^d \bmod n = x$$

wieder entschlüsselt werden.

Ist die Zahl $x \geq n$, so wird x in Blöcke $x_1, x_2, \ldots, x_r$ mit $x_i < n$ aufgespalten. Dann werden die einzelnen Zahlen x_i zu $y_i := x_i^e \bmod n$ verschlüsselt. Die einzelnen Blöcke y_i können dann durch $y_i^d \bmod n$ wieder zu x_i entschlüsselt werden.

Für das folgende Beispiel benutzen wir die in der Tabelle (11.4) angegebene Ersetzung von Buchstaben und Zeichen durch zweiziffrige Zahlen. Dies ist natürlich nur eine von vielen Möglichkeiten.

(11.4) Tabelle

A	10	M	22	X	33
B	11	N	23	Y	34
C	12	O	24	Z	35
D	13	P	25	.	36
E	14	Q	26	,	37
F	15	R	27	!	38
G	16	S	28	?	39
H	17	T	29	blank	40
I	18	U	30	-	41
J	19	V	31	:	42
K	20	W	32	;	43
L	21				

(11.5) BEISPIEL für das RSA–Verfahren:

A) Schlüsselerzeugung

Wähle zwei vierstellige Primzahlen $\boldsymbol{p := 9587}$ und $\boldsymbol{q := 9811}$ und setze

$$\boldsymbol{n := p \cdot q = 9587 \cdot 9811 = 94.058.057\,.}$$

Dann ist $$\boldsymbol{\varphi(n) \;=\; 9586 \cdot 9810 \;=\; 94.038.660}\,.$$

Wähle $\boldsymbol{e}$ teilerfremd zu $\boldsymbol{p-1}$ und $\boldsymbol{q-1}$, etwa $\boldsymbol{e := 1723}$ ($\boldsymbol{e}$ ist hier sogar eine Primzahl, was aber nicht notwendig ist). Dann ist $\boldsymbol{e}$ auch teilerfremd zu dem Produkt

$$\boldsymbol{(p-1) \cdot (q-1) \;=\; \varphi(n)\,.}$$

Berechne nun das Inverse $\boldsymbol{d \in \mathcal{R}_n}$ von $\boldsymbol{e}$ modulo $\boldsymbol{\varphi(n)}$ mit Hilfe des erweiterten euklidischen Algorithmus. Man erhält

$$\boldsymbol{46.773.727 \cdot e + (-857) \cdot \varphi(n) \;=\; 1\,,}$$

also $\boldsymbol{d := 46.773.727}$. Jetzt ist $\boldsymbol{(n,e)}$ der öffentliche und $\boldsymbol{(n,d)}$ der geheime Schlüssel.

B) Durchführung

Klartext: RSA $\boldsymbol{\longleftrightarrow}$ $\boldsymbol{272810 =: x}$ (nach Tabelle (11.4))

Verschlüsselung mit $\boldsymbol{(n,e)}$: $\boldsymbol{x^e \bmod n \;=\; 87.272.685 =: y}$

Entschlüsselung mit $\boldsymbol{(n,d)}$: $\boldsymbol{y^d \bmod n \;=\; 272810 \;=\; x}$ $\boldsymbol{\longleftrightarrow}$ RSA .

Zur Frage nach der Sicherheit des RSA–Verfahrens

$\boldsymbol{(n,e)}$ sei der öffentliche Schlüssel. Wenn man auf irgendeine Weise $\boldsymbol{\varphi(n)}$ berechnen kann, so läßt sich das Inverse $\boldsymbol{d}$ von $\boldsymbol{e}$ modulo $\boldsymbol{n}$ bestimmen, und der geheime Schlüssel $\boldsymbol{(n,d)}$ ist geknackt.

Die **Sicherheit** des Verfahrens beruht also darauf, dass $\boldsymbol{\varphi(n)}$ nicht berechnet werden kann. Wie der nächste Satz zeigt, ist dies aber äquivalent dazu, dass man die Primfaktorzerlegung von $\boldsymbol{n}$ nicht berechnen kann.

(11.6) SATZ: Gegeben sei eine natürliche Zahl $\boldsymbol{n}$, von der bekannt ist, dass sie Produkt zweier verschiedener Primzahlen ist. Dann sind folgende Aussagen äquivalent:

a) $\boldsymbol{\varphi(n)}$ läßt sich berechnen

b) Die Primfaktorzerlegung von $\boldsymbol{n}$ läßt sich bestimmen.

(11.7) BEM: Die Sicherheit des RSA–Verfahrens ist also gewährleistet, wenn es unmöglich ist, die Primfaktorzerlegung von $\boldsymbol{n}$ zu berechnen.

Nach dem heutigen Stand der Computertechnik und der Faktorisierungsverfahren muss man für $\boldsymbol{p}$ und $\boldsymbol{q}$ zwei mindestens 100–stellige Primzahlen wählen. Dann läßt sich die etwa 200–stellige Zahl $\boldsymbol{n = p \cdot q}$ nicht faktorisieren oder es würde zumindest eine sehr sehr lange Zeit dauern.

Digitale Signatur mit dem RSA–Verfahren

Für das RSA–Verfahren gilt nicht nur die Entschlüsselungsbedingung $\boldsymbol{D(E(x)) = x}$ für alle $\boldsymbol{x \in \mathcal{R}_n}$, sondern es gilt nach (11.1) auch umgekehrt

$$\boldsymbol{E(D(x)) = x} \text{ für alle } \boldsymbol{x \in \mathcal{R}_n}\,.$$

Dies läßt sich benutzen, um mit dem RSA–Verfahren auch einen Text zu signieren.

(11.8) Digitale Signatur mit dem RSA–Verfahren:

Wenn $\boldsymbol{A}$ einen signierten Text verschlüsselt an $\boldsymbol{B}$ schicken möchte, verschlüsselt $\boldsymbol{A}$ den Text mit seinem **geheimen** Schlüssel. Wenn dann $\boldsymbol{B}$ den Geheimtext mit dem **öffentlichen** Schlüssel von $\boldsymbol{A}$ entschlüsseln kann, ist sichergestellt, dass der Text auch wirklich von $\boldsymbol{A}$ stammt.

Wenn $\boldsymbol{A}$ noch zusätzlich den Text mit dem **öffentlichen** Schlüssel von $\boldsymbol{B}$ verschlüsselt, ist $\boldsymbol{B}$ der einzige, der den Text entschlüsseln kann.

(11.9) ElGamal–Verschlüsselung (1985)

Eine Primzahl p und eine natürliche Zahl g mit $g < p-1$ sind (öffentlich) gegeben (p und g müssen in geeigneter Weise gewählt sein).

Teilnehmer A hat einen geheimen Schlüssel $a \in \mathcal{R}_{p-1}$ und einen öffentlichen Schlüssel

$$\alpha := g^a \bmod p\,.$$

Teilnehmer B hat einen geheimen Schlüssel $b \in \mathcal{R}_{p-1}$ und einen öffentlichen Schlüssel

$$\beta := g^b \bmod p\,.$$

A verschlüsselt eine Nachricht an B, die in Form einer Zahl $m \in \mathcal{R}_p$ vorliegt, mit dem Schlüssel

$$k := \beta^a \bmod p$$

für ein symmetrisches Verschlüsselungsverfahren E_k und schickt $(\alpha, E_k(m))$ an B.

B berechnet $\alpha^b \bmod p$ und erhält den ursprünglichen Schlüssel

$$\alpha^b \bmod p \;=\; k$$

zurück. Damit kann B jetzt den Geheimtext $E_k(m)$ entschlüsseln.

Die **Sicherheit** dieses Verfahrens ist gewährleistet, wenn man aus einem öffentlichen Schlüssel $y = g^x \bmod p$ nicht den Exponenten x – also den geheimen Schlüssel – bestimmen kann. x heißt in diesem Falle der **diskrete Logarithmus von y zur Basis g**. Die Bestimmung des diskreten Logarithmus ist i.a. genauso schwierig wie die Primfaktorzerlegung einer ganzen Zahl.

(11.10) Massey–Omura–Verschlüsselung

Bei diesem Verfahren hat jeder Teilnehmer zwei geheime Schlüssel.

Für alle Teilnehmer wird eine Primzahl p festgelegt. Jeder Teilnehmer T wählt zwei Zahlen $e_T < p-1$ und $d_T < p-1$ mit

$$(e_T \cdot d_T) \bmod (p-1) \;=\; 1\,,$$

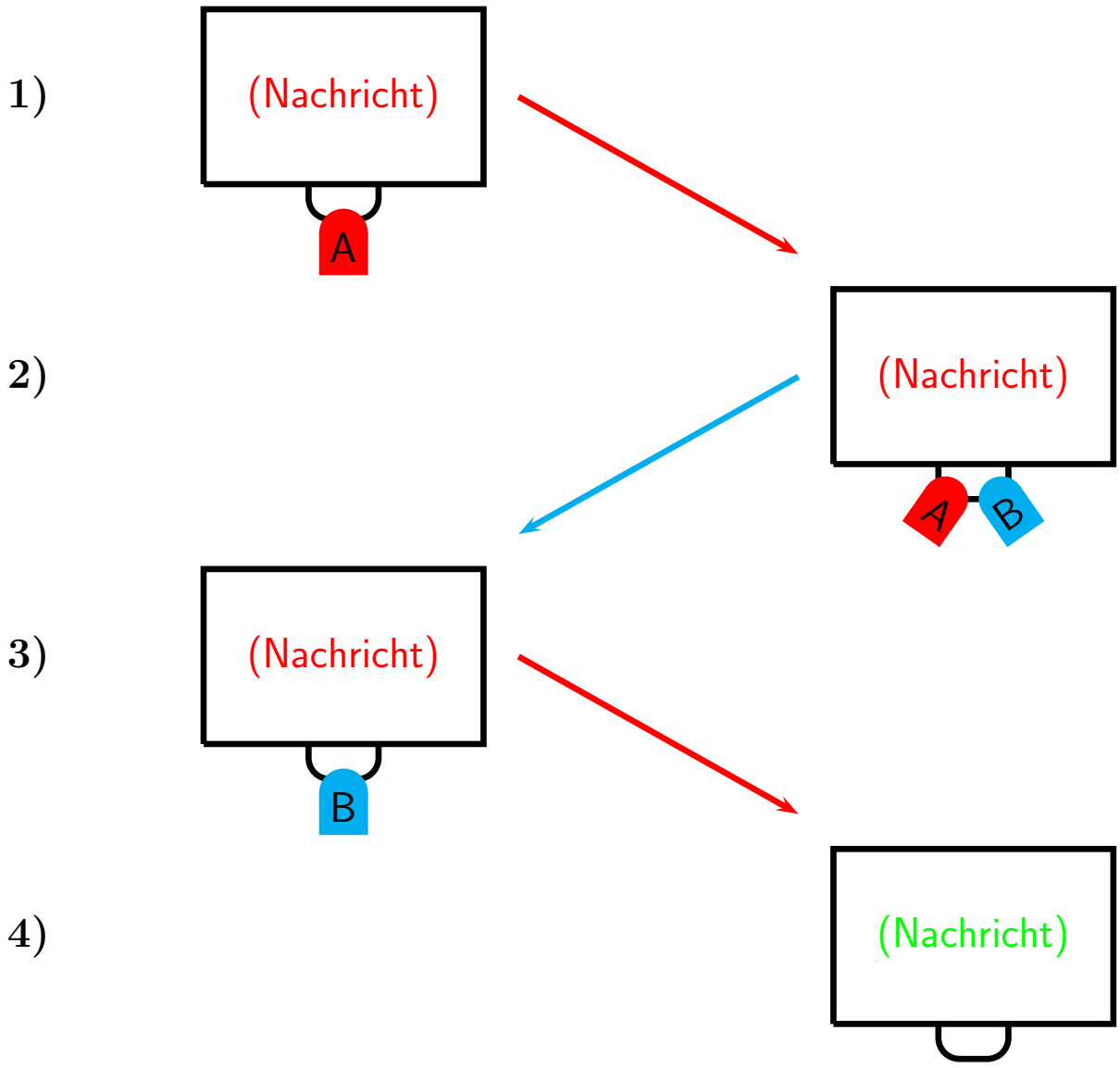
die beide geheimgehalten werden. Teilnehmer A möchte eine Nachricht, die in Form einer Zahl m mit $1 \le m < p$ vorliegt, verschlüsselt an einen Teilnehmer B übermitteln:

$$\begin{aligned}
A \text{ schickt} \quad & \alpha := m^{e_A} \bmod p \quad \text{an } B \\
B \text{ schickt} \quad & \beta := \alpha^{e_B} \bmod p \quad \text{an } A \\
A \text{ schickt} \quad & \gamma := \beta^{d_A} \bmod p \quad \text{an } B \\
B \text{ berechnet} \quad & \delta := \gamma^{d_B} \bmod p
\end{aligned}$$

Dann gilt $\delta = m$, d.h. B hat die Nachricht von A entschlüsselt.

Das Massey–Omura–Verfahren

(Shamir's no-key algorithm)



1) A legt seine Nachricht an B in einen Koffer und verschließt ihn mit einem Schloss, zu dem nur er einen Schlüssel hat. Dann schickt er den Koffer an B.

2) B verschließt den Koffer mit einem Schloss, zu dem nur er einen Schlüssel hat, und schickt ihn zurück an A.

3) A schließt sein Schloss auf und schickt den Koffer wieder zurück an B.

4) B kann jetzt sein Schloss aufschließen, den Koffer öffnen und die Nachricht von A aus dem Koffer herausnehmen.