Zur RSA-Verschlüsselung

Rivest, **S**hamir und **A**delmann haben in ihrem Aufsatz „A Method for Obtaining Digital Signatures and Public-Key Cryprosystems“, 21, Seiten 120-126 Communications of the ACM, im Februar 1978 ein asymmetrisches Verfahren mit öffentlichem und geheimen Schlüssel beschrieben. Obwohl der faszinierende Grundgedanke hier nicht zum ersten Mal auftaucht, hat sich der vorgeschlagene Weg im Einsatz bewährt und durchgesetzt. Es folgt eine Kurzbeschreibung.

Das Verfahren selbst benötigt folgende Vorbereitung, die Hochzahlen verweisen auf die nachfolgenden Erläuterungen:

Primzahlen [1)]	p,q	**geheim**
Beispiel	**p=11, q=23**	

Modul	$n=p\cdot q$	**öffentlich**
	253	

Eulerzahl [2)]	$\Phi(n)=(p-1)\cdot(q-1)$	**geheim**
	220	

öffentlicher Schlüssel [3)]	$e \rightarrow$ Zufallszahl mit $ggT(e,\Phi(n))=1,\ e<\Phi(n)$	**öffentlich**
	27	

geheimer Schlüssel [4)]	$d \rightarrow$ mit $(d\cdot e)\ MOD\ \Phi(n)=1$ d ist die modulare inverse Zahl zu e bezüglich $\Phi(n)$	**geheim**
	163	

Die **Sicherheit** gegen Angriffe liegt darin, dass der Modul n **nicht** auf schnellem Wege in die beiden Primfaktoren p und q zerlegt werden kann, die man aber zur Bestimmung der Eulerzahl braucht.

Zur Erhaltung der Übersichtlichkeit sind für das Beispiel kleine Primzahlen gewählt, im praktischen Einsatz liegen Sie zur Zeit aber im Bereich von über 100 Dezimalstellen.

Die Ver- und Entschlüsselung des Klartextes K erfolgt nach diesen Vorschriften:

Klartext [5)]	K mit K<n
Beispiel	**8**

Geheimtext G [6)]	$G = K^e\ MOD\ n$
	200

Entschlüsselung zu K [7)]	$K = G^d\ MOD\ n$
	8

Bemerkungen zu den Einzelschritten:

1) **Primzahlen:** Es gibt bis heute für praktische Anforderungen kein „schnelles“ Verfahren zur Berechnung großer Primzahlen. Allerdings kann man Zufallszahlen erzeugen und diese auf Primzahleneigenschaft testen (z. B. Miller-Rabin-Test, siehe z. B. Dankmeier, Grundkurs Codierung, Vieweg 2006, Kapitel 5.5). Der Test läuft in einer endlichen Anzahl von Teilschritten ab. Diese Anzahl ist bei großen Zufallszahlen jedoch selbst so groß, dass in praktikabler Zeit nicht alle Teilschritte durchlaufen werden können.

- Ist der Test in einem der Teilschritte negativ, liegt garantiert **keine** Primzahl vor und man erzeugt mit dem Zufallszahlengenerator einen neuen Kandidaten.
- Ist der Test bis zum i-ten Teilschritt positiv, so liegt mit der Wahrscheinlichkeit

$$1-\left(\frac{1}{4}\right)^{i}$$

eine Primzahl vor. Mit steigender Anzahl positiver Teilschritte wächst also die Wahrscheinlichkeit , dass man eine Primzahl gefunden hat, sicher ist es dennoch nicht. Für den praktischen Gebrauch reicht es aber aus.

2) **Eulerzahl:** Dies ist die Anzahl der zur Zahl s teilerfremden Zahlen kleiner als s einschließlich der „1“. Beispiel: s = 6 → $\Phi(6)=2$, da nur 1 und 5 teilerfremd zu 6 sind.

Da Primzahlen p keine Teiler außer 1 und sich selbst besitzen, gilt

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

3) **öffentlicher Schlüssel e:** Eine Zufallszahl, welche die Bedingung $e<\Phi(n)$ erfüllen soll und zu $\Phi(n)$ teilerfremd sein **muss**. Man sagt auch, das der größte gemeinsame Teiler (= ggT) von e und $\Phi(n)$ gleich 1 sein muss. Diese Eigenschaft kann mit dem **Euklidischen Algorithmus** (=EA) überprüft werden. Mit der Startbedingung

$$\Phi_0=\Phi(n) \quad \text{und} \quad \Phi_1=e$$

läuft der EA in folgenden Schritten ab (linker Bereich im Diagramm):

Φ_0	=	a_1	·	Φ_1	+	Φ_2
220		8		27		4
Φ_1	=	a_2	·	Φ_2	+	Φ_3
27		6		4		3
Φ_2	=	a_3	·	Φ_3	+	Φ_4
4		1		3		1
Φ_3	=	a_4	·	Φ_4	+	Φ_5
3		3		1		0

$= ggT(\Phi_0,\Phi_1) = ggT(\Phi(n),e)$

→ der EA endet immer mit „0“

m	=	4	= Index von Rest 1

Rekursion:

z_i	=	$-a_{m-i}$	·	z_{i-1}	+	z_{i-2}

mit i = 2, 3, …, m-1 = **2,3**

Startwerte:

z_0 = 1

z_1 = $-a_{m-1}$ = $-a_3$ = **-1**

Durchführung der Rekursion:

z_2	=	$-a_2$	·	z_1	+	z_0
7		-6		-1		1

z_3	=	$-a_1$	·	z_2	+	z_1
-57		-8		7		-1

z_3 ist die modulare inverse Zahl d zu e

Wegen $d = d \text{ MOD } \Phi(n)$

ist auch $d = -57 \text{ MOD } 220 = +163$

Der EA endet immer mit dem Rest 0. Im vorhergehenden Schritt ist der Rest der gesuchte ggT (hier = 1). Die Koeffizienten ai werden zur Bestimmung des geheimen Schlüssels d benötigt.

4) **geheimer Schlüssel d:** Mit der Rückwärtsrechnung des EA lässt sich über die im oben angegebenen Diagramm rechts dargestellte Rekursion der geheime Schlüssel d bestimmen. Hier ist d = - 57 bzw. +163 (siehe z. B. Dankmeier, Grundkurs Codierung, Vieweg 2006, Kapitel 5.3)

5) **Verschlüsselung:** Die Ausführung der in $G = K^e \text{ MOD } n$ erscheinenden Potenzierung ist wegen der Eigenschaft der Assoziativität (Vertauschung der Reihenfolgen) der Modulo-Operation schrittweise in kleinen „Portionen“ möglich, so dass in den Teilschritten nur Modulo-Operanden kleiner als $[\Phi(n)]^2$ auftreten. Dazu zerlegt man den Exponenten e in eine Summe von Zweier-Potenzen, hier

$e = 27 = 16 + 8 + 2 + 1$ und erhält $G = 8^{16+8+2+1} \text{ MOD } 253 = \left[8^{16} \cdot 8^8 \cdot 8^2 \cdot 8\right] \text{MOD } 253$.

Nun wird die Folge

$$8 \text{ MOD } 253 = 8$$

$$8^2 \text{ MOD } 253 = 64$$

$$8^4 \text{ MOD } 253 = \left(8^2\right)^2 \text{ MOD } 253 = \left(8^2 \text{MOD } 253\right)^2 \text{ MOD } 253 = 64^2 \text{ MOD } 253 = 48$$

$$8^8 \text{ MOD } 253 = \left(8^4\right)^2 \text{ MOD } 253 = \left(8^4 \text{MOD } 253\right)^2 \text{ MOD } 253 = 48^2 \text{ MOD } 253 = 27$$

$$8^{16} \text{ MOD } 253 = \left(8^8\right)^2 \text{ MOD } 253 = \left(8^8 \text{MOD } 253\right)^2 \text{ MOD } 253 = 27^2 \text{ MOD } 253 = 223$$

für **alle** Zweipotenzen bestimmt, auch für die im Exponenten e Fehlenden und nacheinander

$$G = \left[8^{16} \cdot 8^8 \cdot 8^2 \cdot 8\right] \text{MOD } 253 = \left[\left(\left(8^{16} \cdot 8^8 \cdot 8^2\right) \text{ MOD } 253\right) \cdot 8\right] \text{ MOD } 253$$

$$G = \left[\left(\left(8^{16} \cdot 8^8 \cdot 8^2\right) \text{ MOD } 253\right) \cdot 8\right] \text{ MOD } 253 = \left[\left(\left(8^{16} \cdot 8^8\right) \text{ MOD } 253\right) \cdot 64 \cdot 8\right] \text{ MOD } 253$$

usw. gebildet. Man erhält

$G = \left[223 \cdot 27 \cdot 64 \cdot 8\right] \text{ MOD } 253$ (48 fehlt, da die Zweierpotenz 8^4 nicht auftritt)

und wertet den Ausdruck schrittweise ebenfalls mit Hilfe des Assoziativgesetzes aus. Ergebnis:

$G = \left[223 \cdot 27 \cdot 64 \cdot 8\right] \text{ MOD } 253 = 200$.

Die einzige Bedingung an das Rechenprogramm ist also, dass die Modulo-Operation für Operanden < n^2 mit voller Genauigkeit durchgeführt werden kann, was sich immer sicherstellen lässt.

6) **Entschlüsselung:** Diese wird mit $K = G^d \text{ MOD } n$ wie bei der Verschlüsselung ausgeführt. Dabei ist mit Hilfe der Assoziativität der Modulo-Operation

$$K = \left(K^e \text{ MOD } n\right)^d \text{ MOD } n = K^{e \cdot d} \text{ MOD } n = K$$

Das dies funktioniert, liegt an der speziellen Abhängigkeit im Schlüsselpaar:

$$(d \cdot e) \text{ MOD } \Phi(n) = 1$$

Einen Beweis hierfür findet man z. B. bei Beutelsbacher, „Kryptografie in Theorie und Praxis“, Vieweg 2010, Kapitel 10.3.

Hier erhalten wir

$$K = 200^{163} \text{ MOD } 253 = 200^{128+32+2+1} \text{ MOD } 253 = 8$$

Dafür müssen die 7 Modulo-Operationen

$$200 \text{ MOD } 253 = 200$$

$$200^2 \text{ MOD } 253 = 26$$

$$200^4 \text{ MOD } 253 = \left[200^2 \text{ MOD } 253\right]^2 \text{ MOD } 253 = 26^2 \text{ MOD } 253 = 170$$

usw. ausgeführt werden. Der Ablauf ist z. B. auch bei Dankmeier „Grundkurs Codierung, Vieweg 2006“, Kapitel 5.4 beschrieben und wird in vielen weiteren asymmetrischen Verschlüsselungsverfahren benötigt. Man nennt es **„Square an Multiply“.**

Eine bedeutende Eigenheit des RSA-Algorithmus liegt darin, dass das Schlüsselpaar (d, e) bei jedem Teilnehmer vor Ort erzeugt werden kann und der geheime Schlüssel d **nicht** übergeben werden muss. Wenn ein Sender einem Empfänger eine verschlüsselte Nachricht schicken will, verschlüsselt er diese mit dem öffentlichen Schlüssel des Empfängers. Ein Abhörer kann über $G = K^e \text{ MOD } n$ aus den ihm zugänglichen Informationen G, e und n **nicht** auf den Klartext K schließen. Es bleibt ihm (beinahe) nur der Brute Force-Angriff.

Eine weitere nützliche Eigenschaft ist die Vertauschbarkeit von Verschlüsselung und Entschlüsselung: Wenn ein Sender einen Klartext K an die „Welt“ schicken will, verschlüsselt er diesen mit seinem geheimen Schlüssel d

$$G = K^d \text{ MOD } n$$

und stellt G öffentlich zur Verfügung. Jeder kann G mit dem öffentlichen Schlüssel e des Senders über

$$K = G^e \text{ MOD } n \rightarrow K = \left(K^d \text{MOD } n\right)^e \text{ MOD } n \rightarrow K = \left(K^d \text{MOD } n\right)^e \text{ MOD } n \ .$$

entschlüsseln. Dabei hilft wieder die Assoziativität der Modulo-Operation:

$$K = K^{d \cdot e} \text{ MOD } n = K^{e \cdot d} \text{ MOD } n = K \ .$$

Die Nachricht des Senders selbst ist bei dieser Variante zwar nicht geheim, die Empfänger können aber sicher sein, dass er der Sender ist, denn kein anderer wird den passenden geheimen Schlüssel e haben (Frage: Ist das wirklich ganz sicher?)