6. Restklassenringe und RSA

Erinnerung: Kongruenzen und Restklassen

Definition

Ganze Zahlen $x, y \in \mathbb{Z}$ sind kongruent modulo m für eine natürliche Zahl $m \in \mathbb{N}$, falls x und y denselben Rest bei Division durch m haben und wir schreiben

$$x \equiv y \pmod{m} \qquad \Longleftrightarrow \qquad m \mid x - y\,.$$

Dies definiert eine Äquivalenzrelation und die Äquivalenzklasse

$$[x]_m := \{y \in \mathbb{Z} \colon x \equiv y \pmod{m}\}$$

ist die Restklasse von x modulo m.

Bemerkungen:

- $\mathbb{Z} = [0]_m \cup \ldots \cup [m-1]_m$ für jedes $m \in \mathbb{N}$
- Für die Menge der Restklassen (Faktormenge der Äquivalenzrelation kongruent modulo m) schreiben wir

$$\mathbb{Z}/m\mathbb{Z} := \big\{[0]_m\,,\ [1]_m\,,\ \ldots\,,\ [m-1]_m\big\}$$

Erinnerung: Modulare Arithmetik

- mit Restklassen kann man gut rechnen
- $x_1 \equiv y_1 \pmod m$ und $x_2 \equiv y_2 \pmod m \Rightarrow (x_1 + x_2) \equiv (y_1 + y_2) \pmod m$

$\Rightarrow$ $[z]_m \oplus [z']_m := [z + z']_m$ ist wohldefinierte Addition auf $\mathbb{Z}/m\mathbb{Z}$

  - Addition auf $\mathbb{Z}/m\mathbb{Z}$ ist assoziativ und kommutativ
  - $[0]_m$ ist neutrales Element der Addition auf $\mathbb{Z}/m\mathbb{Z}$
  - Subtraktion kann durch $[z]_m \ominus [z']_m := [z - z']_m$ definiert werden
  - $[-z]_m$ ist invers zu $[z]_m$, d. h. $-[z]_m = [-z]_m$
  - für $\ell \in \{0, \ldots, m-1\}$ gilt $-[\ell]_m = [-\ell]_m = [m - \ell]_m$

- $x_1 \equiv y_1 \pmod m$ und $x_2 \equiv y_2 \pmod m \Rightarrow (x_1 \cdot x_2) \equiv (y_1 \cdot y_2) \pmod m$

$\Rightarrow$ $[z]_m \odot [z']_m := [z \cdot z']_m$ ist wohldefinierte Multiplikation auf $\mathbb{Z}/m\mathbb{Z}$

  - Multiplikation auf $\mathbb{Z}/m\mathbb{Z}$ ist assoziativ und kommutativ
  - $[1]_m$ ist neutrales Element der Multiplikation auf $\mathbb{Z}/m\mathbb{Z}$
  - im Allgemeinen gibt es keine inversen Elemente für die Multiplikation:

$$[2]_4 \odot [0]_4 = [0]_4\,, \qquad [2]_4 \odot [1]_4 = [2]_4\,,$$
$$[2]_4 \odot [2]_4 = [4]_4 = [0]_4\,, \qquad [2]_4 \odot [3]_4 = [6]_4 = [2]_4$$

    $\Rightarrow [2]_4$ hat kein multiplikativ Inverses in $\mathbb{Z}/4\mathbb{Z}$
  - Addition und Multiplikation erfüllen das Distributivgesetz

Für jedes $m \in \mathbb{N}$ heißt $\mathbb{Z}/m\mathbb{Z}$ mit Verknüpfungen $\oplus$ und $\odot$ Restklassenring modulo m.

- $\mathbb{Z}/1\mathbb{Z} = \{[0]_1\} = \{\mathbb{Z}\}$ ist trivial (Nullring), aber $\mathbb{Z}/2\mathbb{Z} = \mathbb{F}_2$ ist sogar ein Körper

Wohldefiniertheit von $\oplus$ und $\odot$ in $\mathbb{Z}/m\mathbb{Z}$

Definition von $\oplus$: $[z]_m \oplus [z']_m := [z + z']_m$

- Es ist zu zeigen, dass diese Definition unabhängig von der Wahl der Repräsentanten der Restklassen ist. D. h. wenn y kongruent zu z und y' kongruent zu z' ist, dann muss auch $y + y'$ kongruent zu $z + z'$ sein.
- Sei also y kongruent zu z modulo $m \Rightarrow m \mid z - y$ und es gelte analog $m \mid z' - y'$.

$\Rightarrow$ $m \mid ((z - y) + (z' - y')) \Rightarrow m \mid ((z + z') - (y + y'))$

$\Rightarrow$ $y + y'$ und $z + z'$ sind kongruent modulo m ✓

Definition von $\odot$: $[z]_m \odot [z']_m := [z \cdot z']_m$

- Analog betrachte y und y' mit y kongruent z und y' kongruent z' modulo m.

$\Rightarrow$ $z = q_z m + r$ und $y = q_y m + r$, sowie
$z' = q_{z'} m + r'$ und $y' = q_{z'} m + r'$
mit $q_z,\ q_{z'},\ q_y,\ q_{y'} \in \mathbb{Z}$ und $r,\ r' \in \{0, \ldots, m - 1\}$

$\Rightarrow$ $zz' = m(q_z q_{z'} m + q_z r' + q_{y'} r) + rr' \Rightarrow zz' \in [rr']_m$
genauso rechnet man nach, dass $yy' \in [rr']_m$

$\Rightarrow$ yy' und zz' sind kongruent modulo m ✓

Rechenregeln in $\mathbb{Z}/m\mathbb{Z}$ vererben sich von $\mathbb{Z}$

Exemplarisch überprüfen wir das Distributivgesetz:

$$[x]_m \odot \big([y]_m \oplus [z]_m\big) = \big([x]_m \odot [y]_m\big) \oplus \big([x]_m \odot [z]_m\big)$$

für alle ganzen Zahlen x, y, $z \in \mathbb{Z}$ und $m \in \mathbb{N}$.

Beweis:

$$[x]_m \odot \big([y]_m \oplus [z]_m\big) \overset{\text{Def.}\oplus}{=} [x]_m \odot \big([y+z]_m\big) = [x]_m \odot [y+z]_m$$

$$\overset{\text{Def.}\odot}{=} [x \cdot (y+z)]_m \overset{\text{DG. in } \mathbb{Z}}{=} [x \cdot y + x \cdot z]_m \overset{\text{Def.}\oplus}{=} [x \cdot y]_m \oplus [x \cdot z]_m$$

und zwei weitere Anwendungen der Definition von $\odot$ liefern das Gewünschte:

$$[x \cdot y]_m \oplus [x \cdot z]_m \overset{\text{Def.}\odot}{=} \big([x]_m \odot [y]_m\big) \oplus \big([x]_m \odot [z]_m\big) .$$

Restklassenringe

Satz

Für alle natürlichen Zahlen $m \in \mathbb{N}$ sind die Operationen

$$\oplus\colon \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z} \to \mathbb{Z}/m\mathbb{Z} \text{ definiert durch } [a]_m \oplus [b]_m := [a+b]_m,$$
$$\odot\colon \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z} \to \mathbb{Z}/m\mathbb{Z} \text{ definiert durch } [a]_m \odot [b]_m := [a \cdot b]_m$$

wohldefiniert und für alle $[a]_m$, $[b]_m$, $[c]_m \in \mathbb{Z}/m\mathbb{Z}$ gelten

- die Assoziativgesetze:

$$[a]_m \oplus ([b]_m \oplus [c]_m) = ([a]_m \oplus [b]_m) \oplus [c]_m$$
$$\text{und} \quad [a]_m \odot ([b]_m \odot [c]_m) = ([a]_m \odot [b]_m) \odot [c]_m\,,$$

- die Kommutativgesetze:

$$[a]_m \oplus [b]_m = [b]_m \oplus [a]_m$$
$$\text{und} \quad [a]_m \odot [b]_m = [b]_m \odot [a]_m\,,$$

- das Distributivgesetz: $[a]_m \odot ([b]_m \oplus [c]_m) = ([a]_m \odot [b]_m) \oplus ([a]_m \odot [c]_m)$,
- die Existenz neutraler Elemente: $[a]_m \oplus [0]_m = [a]_m$ und $[1]_m \odot [a]_m = [a]_m$
- und die Existenz inverser Elemente für $\oplus$: $[a]_m \oplus [-a]_m = [0]_m$.

Wir benutzen vereinfachend von nun an $+$ und $\cdot$ an Stelle von $\oplus$ und $\odot$.

Prime Restklassengruppe

Definition

Für $m \geqslant 2$ heißt eine Restklasse $[a]_m \in \mathbb{Z}/m\mathbb{Z}$ multiplikativ invertierbar, falls es ein $[b]_m \in \mathbb{Z}/m\mathbb{Z}$ gibt, sodass

$$[a]_m \cdot [b]_m = [1]_m$$

und $[b]_m$ heißt (multiplikativ) Inverses von $[a]_m$. Die Menge invertierbarer Elemente

$$(\mathbb{Z}/m\mathbb{Z})^\times := \big\{[a]_m \in \mathbb{Z}/m\mathbb{Z} \colon [a]_m \text{ ist multiplikativ invertierbar}\big\}$$

heißt prime Restklassengruppe und die Elemente heißen Einheiten.

Bemerkungen:

- Es gibt höchstens ein multiplikativ Inverses für jedes $[a]_m \in \mathbb{Z}/m\mathbb{Z}$:
  Falls $[a]_m \cdot [b]_m = [1]_m$ und $[a]_m \cdot [b']_m = [1]_m$, dann gilt

$$[b]_m = [b]_m \cdot [1]_m = [b]_m \cdot ([a]_m \cdot [b']_m) = ([b]_m \cdot [a]_m) \cdot [b']_m = [1]_m \cdot [b']_m = [b']_m\,,$$

  d. h. $[b]_m = [b']_m$. ✓
- Wir bezeichnen somit das Inverse von $[a]_m$ (falls es existiert) mit $[a]_m^{-1}$.

Bemerkungen zu $(\mathbb{Z}/m\mathbb{Z})^\times$

- $[0]_m$ ist nicht multiplikativ invertierbar, da für alle $m \geqslant 2$ und $z \in \mathbb{Z}$ gilt $0 \cdot z = 0 \not\equiv 1 \pmod{m}$.

$\Rightarrow \left|(\mathbb{Z}/m\mathbb{Z})^\times\right| \leqslant m - 1$

- $(\mathbb{Z}/m\mathbb{Z})^\times$ ist unter Multiplikation abgeschlossen, d. h.

$$[a]_m, [b]_m \in (\mathbb{Z}/m\mathbb{Z})^\times \quad \Longrightarrow \quad [a]_m \cdot [b]_m \in (\mathbb{Z}/m\mathbb{Z})^\times$$

Beweis: Da $[a]_m$ und $[b]_m$ multiplaktiv invertierbar sind, gibt es $[y]_m := [b]_m^{-1} \cdot [a]_m^{-1} \in \mathbb{Z}/m\mathbb{Z}$ und es gilt

$$([a]_m \cdot [b]_m) \cdot ([b]_m^{-1} \cdot [a]_m^{-1}) = [a]_m \cdot ([b]_m \cdot [b]_m^{-1}) \cdot [a]_m^{-1} = [a]_m \cdot [1]_m \cdot [a]_m^{-1} = [1]_m .$$

und somit hat $[a]_m \cdot [b]_m$ multiplikativ Inverses $[y]_m$ und ist in $(\mathbb{Z}/m\mathbb{Z})^\times$. □

- Für alle $[a]_m \in (\mathbb{Z}/m\mathbb{Z})^\times$ ist $[x]_m \mapsto [a]_m \cdot [x]_m$ eine Bijektion auf $(\mathbb{Z}/m\mathbb{Z})^\times$:
  - Injektivität: $[a]_m \cdot [x]_m = [a]_m \cdot [y]_m$
    $\Rightarrow [a]_m^{-1} \cdot [a]_m \cdot [x]_m = [a]_m^{-1} \cdot [a]_m \cdot [y]_m \Rightarrow [x]_m = [y]_m$ ✓
  - Surjektivität: $[z]_m \in (\mathbb{Z}/m\mathbb{Z})^\times \Rightarrow [y]_m := [a]_m^{-1} \cdot [z]_m \in (\mathbb{Z}/m\mathbb{Z})^\times$
    $\Rightarrow [a]_m \cdot [y]_m = [z]_m$, d. h. $[z]_m$ ist im Bild der Abbildung ✓

Multiplikative Inverse

Beispiele:

- Wir hatten bereits gesehen, dass $[2]_4$ kein multiplikativ Inverses hat.
- $[2]_5^{-1} = [3]_5$, da $[2]_5 \cdot [3]_5 = [6]_5 = [1]_5$
- $[3]_4$ ist selbstinvers, da

$$[3]_4 \cdot [3]_4 = [9]_4 = [1]_4 \,,$$

d. h. $[3]_4^{-1} = [3]_4$

Satz

Ein Element $[a]_m \in \mathbb{Z}/m\mathbb{Z}$ ist genau dann multiplikativ invertierbar/eine Einheit, wenn $\mathrm{ggT}(a, m) = 1$ (d. h. wenn a und m teilerfremd sind).

Korollar

$(\mathbb{Z}/p\mathbb{Z}, +, \cdot, [0]_p, [1]_p)$ ist genau dann ein Körper, wenn p eine Primzahl ist.

$[a]_m$ multiplikativ invertierbar $\Longleftrightarrow$ ggT$(a, m) = 1$

Beweis:

„$\Longrightarrow$“ Sei $[a]_m$ multiplikativ invertierbar und $[b]_m = [a]_m^{-1}$.

$\Rightarrow$ $a \cdot b \equiv 1 \pmod{m}$ (Welches $\cdot$?)

$\Rightarrow$ es existiert $q \in \mathbb{Z}$ mit $a \cdot b = q \cdot m + 1$

$\Rightarrow$ $a \cdot b - q \cdot m = 1$

$\Rightarrow$ d. h. jeder Teiler von a und m teilt auch 1

$\Rightarrow$ ggT$(a, m) = 1$ ✓

„$\Longleftarrow$“ Sei ggT$(a, m) = 1$.

- Wegen dem Lemma von Bézout (siehe Elementare Zahlentheorie) gibt es $s,\ t \in \mathbb{Z}$, sodass

$$s \cdot a + t \cdot m = \text{ggT}(a, m) = 1 \quad \Longrightarrow \quad s \cdot a = (-t) \cdot m + 1\,.$$

$\Rightarrow$ $s \cdot a \equiv 1 \pmod{m}$

$\Rightarrow$ $[s]_m$ ist multiplikativ Inverses von $[a]_m$ ✓ □

Berechnung von multiplikativen Inversen

- Zur Erinnerung: Der erweiterterte EUKLIDische Algorithmus lieferte einen algorithmischen Beweis des Lemmas von Bézout.

$\Rightarrow$ $s,\ t \in \mathbb{Z}$ mit $s \cdot a + t \cdot m = \mathrm{ggT}(a, m)$ können mit dem erweiterterten EUKLIDischen Algorithmus effizient berechnet werden

$\Rightarrow$ Repräsentant $s \in [a]_m^{-1}$ kann effizient berechnet werden, falls ein multipliaktiv Inverses von $[a]_m$ existiert (d. h. genau dann, wenn $\mathrm{ggT}(a, m) = 1$)

Beispiel: $[13]_{2412}$ invertierbar?

$$2412 = 185 \cdot 13 + 7$$
$$13 = 1 \cdot 7 + 6$$
$$7 = 1 \cdot 6 + 1$$

$\Rightarrow$ $\mathrm{ggT}(13, 2412) = 1$ und Rückwärtseinsetzen ergibt:

$$1 = 7 - 1 \cdot 6 = 7 - 1 \cdot (13 - 1 \cdot 7) = 2 \cdot 7 - 1 \cdot 13 = 2 \cdot (2412 - 185 \cdot 13) - 1 \cdot 13$$

$\Rightarrow$ $-371 \cdot 13 + 2 \cdot 2412 = 1 \Rightarrow [-371]_{2412} = [2041]_{2412} = [13]_{2412}^{-1}$

Probe: $13 \cdot 2041 = 26533 = 11 \cdot 2412 + 1 \Rightarrow 13 \cdot 2412 \equiv 1 \pmod{2412}$

Kleiner Satz von Fermat

Satz (Fermat 1640)

Sei $a \in \mathbb{N}$ und p eine Primzahl mit $p \nmid a$. Dann gilt

$$a^{p-1} \equiv 1 \pmod{p}$$

und somit $[a^{p-1}]_p = [a]_p^{-1}$.

Beweis: Mit Induktion über a für eine feste Primzahl p zeigen wir $a^p \equiv a \pmod{p}$ für alle $a \in \mathbb{N}$. Der Satz folgt dann, da wir wegen der Voraussetzung ($\mathrm{ggT}(a, p) = 1$) auf beiden Seiten mit $b \in [a]_p^{-1}$ „kürzen“ können.

- Induktionsanfang für $a = 1$: klar, da $1^p = 1 \equiv 1 \pmod{p}$ für $p \geqslant 2$ ✓
- Induktionsschritt $a \to a + 1$: Mit dem binomischen Lehrsatz folgt

$$(a+1)^p = a^p + \sum_{i=1}^{p-1} \binom{p}{i} 1^i a^{p-i} + 1^p .$$

Da jeder der Summanden in $\sum_{i=1}^{p-1} \binom{p}{i} 1^i a^{p-i}$ wegen dem Binomialkoeffizienten $\binom{p}{i}$ durch p teilbar ist (p Primzahl $\Rightarrow \mathrm{ggT}(i!(p-i)!, p) = 1$ für $0 < i < p$), gilt

$$(a+1)^p \equiv a^p + 1 \pmod{p} .$$

Nach der Induktionsvoraussetzung gilt $a^p \equiv a \pmod{p}$

$\Rightarrow$ $(a+1)^p \equiv a + 1 \pmod{p}$ □

Bemerkungen zum kleinen Satz von Fermat

- Für a, p wie in dem Satz können wir für „große“ x bei Berechnungen der Form $a^x \pmod p$ die Rechnung vereinfachen, da

$$a^{p-1} \equiv 1 \,(\mathrm{mod}\, p) \quad \Longrightarrow \quad a^x \equiv a^{x-(p-1)} \,(\mathrm{mod}\, p) \equiv a^r \,(\mathrm{mod}\, p)$$

  für den Rest $r = \mathrm{mod}(x, p-1)$ der ganzzahligen Division x durch $p-1$.

Satz (Fermat und Euler)

Seien a, $m \in \mathbb{N}$ teilerfremd und sei $\varphi(m)$ die Anzahl der zu m teilerfremden natürlichen Zahlen kleiner m. Dann gilt

$$a^{\varphi(m)} \equiv 1 \pmod m,$$

Bemerkungen:

- $\varphi\colon \mathbb{N} \to \mathbb{N}$ heißt eulersche φ-Funktion
- für Primzahlen p ist $\varphi(p) = p-1 \Rightarrow$ Satz von Fermat und Euler verallgemeinert den kleinen Satz von Fermat

Beweis von Fermat–Euler

Beweis: Sei $\mathrm{ggT}(a, m) = 1$ und seien $x_1, \ldots, x_{\varphi(m)} \in \mathbb{N}$ die zu m teilerfremden natürlichen Zahlen kleiner m.

$\Rightarrow$ $(\mathbb{Z}/m\mathbb{Z})^\times = \{[x_1]_m, \ldots, [x_{\varphi(m)}]_m\}$ und $[a]_m \in (\mathbb{Z}/m\mathbb{Z})^\times$

- wir hatten bereits gesehen, dass $[x]_m \mapsto [a]_m \cdot [x]_m$ eine Bijektion auf $(\mathbb{Z}/m\mathbb{Z})^\times$ ist, d. h.

$$\{[ax_1]_m, \ldots, [ax_{\varphi(m)}]_m\} = \{[x_1]_m, \ldots, [x_{\varphi(m)}]_m\}$$

$\Rightarrow$

$$\prod_{i=1}^{\varphi(m)} [x_i]_m = \prod_{i=1}^{\varphi(m)} [ax_i]_m = \left[a^{\varphi(m)} \prod_{i=1}^{\varphi(m)} x_i\right]_m = [a^{\varphi(m)}]_m \prod_{i=1}^{\varphi(m)} [x_i]_m$$

- da $[x_1]_m, \ldots, [x_{\varphi(m)}]_m$ Einheiten sind, können wir auf beiden Seiten mit $\prod_{i=1}^{\varphi(m)} [x_i]_m^{-1}$ multiplizieren und erhalten

$$[1]_m = [a^{\varphi(m)}]_m \quad \Longrightarrow \quad a^{\varphi(m)} \equiv 1 \pmod{m}.$$

EULERsche φ-Funktion

Für jede natürliche Zahl $m \in \mathbb{N}$ definiert durch

$$\varphi(m) = \left|\{x \in \mathbb{N} \colon \text{ ggT}(x, m) = 1 \text{ und } 1 \leqslant x < m\}\right|.$$

- $\varphi(m) = \left|(\mathbb{Z}/m\mathbb{Z})^\times\right| \leqslant m - 1$
- $\varphi(p) = p - 1$ für Primzahlen p
- $\varphi(p \cdot q) = (p-1)(q-1) = \varphi(p)\varphi(q)$ für Primzahlen $p \neq q$:
  Beweis: Neben den trivialen Teilern teilen nur p und q das Produkt pq.
  $\Rightarrow$ alle $x < pq$ nicht teilerfremd zu pq sind Vielfache von p oder q
  diese Vielfachen sind: $p, 2p, \ldots, (q-1)p$ und $q, 2q, \ldots, (p-1)q$
  $\Rightarrow \varphi(pq) = pq - 1 - (q-1) - (p-1) = (p-1)(q-1)$ □
- **Aber:** Berechnung von $\varphi(n)$ für $n = pq$ mit Primzahlen $p \neq q$ ohne Kenntnis von p und q ist *schwer*
  $\longrightarrow$ so schwer, wie Berechnung der Primfaktorzerlegung von n als $n = pq$
  Beweis: $\varphi(n) = (p-1)(q-1) = pq + 1 - (p+q) = n + 1 - (p+q)$
  $\Rightarrow$ bekanntes $\varphi(n)$ liefert die Summe $p + q = n + 1 - \varphi(n)$
  $\Rightarrow$ mit $p = n/q$ erhält man quadratische Gleichung in einer Variable (in q)
  $\Rightarrow$ Lösung der quadratischen Gleichung ergibt q und dann p □
- kein effizienter Algorithmus bekannt

$\longrightarrow$ eine Grundlage des RSA-Verfahrens

A Method for Obtaining Digital Signatures and Public-Key Cryptosystems

R.L. Rivest, A. Shamir, and L. Adleman*

Abstract

An encryption method is presented with the novel property that publicly revealing an encryption key does not thereby reveal the corresponding decryption key. This has two important consequences:

1. Couriers or other secure means are not needed to transmit keys, since a message can be enciphered using an encryption key publicly revealed by the intended recipient. Only he can decipher the message, since only he knows the corresponding decryption key.
2. A message can be "signed" using a privately held decryption key. Anyone can verify this signature using the corresponding publicly revealed encryption key. Signatures cannot be forged, and a signer cannot later deny the validity of his signature. This has obvious applications in "electronic mail" and "electronic funds transfer" systems.

A message is encrypted by representing it as a number M, raising M to a publicly specified power e, and then taking the remainder when the result is divided by the publicly specified product, n, of two large secret prime numbers p and q. Decryption is similar; only a different, secret, power d is used, where $e \cdot d \equiv 1 \pmod{(p-1)\cdot(q-1)}$. The security of the system rests in part on the difficulty of factoring the published divisor, n.

RSA-Verfahren

- benannt nach den Erfindern Rivest, Shamir und Adleman
- Public-Key-Verschlüsselungverfahren von 1977
- basiert auf öffentlichen und (geheimen) privaten Schlüssel des Empfängers
- Sender verschlüsselt (engl. **e**ncrypt) Nachricht M mit öffentlichem Schlüssel
- verschlüsselte Nachricht C wird an den Empfänger geschickt
- Empfänger entschlüsselt (engl. **d**ecrypt) C und rekonstruiert so M
- Nachrichten werden hierbei als Zahlen codiert, d. h. o. B. d. A. ist $M \in \mathbb{N}$

RSA-Verfahren

1. Schlüsselgenerierung: Empfänger wählt zwei große Primzahlen p und q
   - berechnet $N = pq$ und $\varphi(N) = (p-1)(q-1)$
   - wählt e teilerfremd zu $\varphi(N)$ mit $1 < e < \varphi(N)$
   - berechnet $d \in [e]^{-1}_{\varphi(N)}$, d. h. $ed = r\varphi(N) + 1$ für ein $r \in \mathbb{Z}$
   - veröffentlicht (e, N) und speichert geheim (d, N)
2. Verschlüsselung: Sender berechnet $C \equiv M^e \pmod N$ für Nachricht $M < N$ und schickt Nachricht C an Empfänger
3. Entschlüsselung: Empfänger berechnet kanonisches $M' \equiv C^d \pmod N$

Fermat–Euler: $M' \equiv C^d \equiv (M^e)^d \equiv M^{r\varphi(N)+1} \equiv (M^{\varphi(N)})^r \cdot M \equiv 1^r \cdot M \equiv M \pmod N$

Korrektheit des RSA-Verfahrens — Wieso?

- die Kongruenz $M' \equiv M$ auf der letzten Folie verwendete den Satz von Fermat und Euler für

$$M^{\varphi(N)} \equiv 1 \pmod{N}$$

- der Satz benötigt aber auf der Annahme $\mathrm{ggT}(M, N) = 1$
- da $N = pq$, müssen wir die Fälle $p \mid M$ bzw. $q \mid M$ gesondert betrachten: Sei also $p \mid M$

$\Rightarrow$ $M \equiv 0 \pmod{p} \Longrightarrow M^{r\varphi(N)+1} \equiv M \pmod{p}$
wegen $p \mid M$ und $M < pq$ gilt in diesem Fall $q \nmid M$
$\Longrightarrow M^{q-1} \equiv 1 \pmod{q}$, wegen dem kleinen Satz von Fermat
$\Longrightarrow M^{r\varphi(N)+1} = (M^{q-1})^{r(p-1)} \cdot M \equiv 1^{r(p-1)} M \pmod{q} \equiv M \pmod{q}$

Schließlich zeigt man (Übung), dass für Primzahlen $p \neq q$ und $x, y \in \mathbb{Z}$ gilt:

$$x \equiv y \pmod{p} \quad \text{und} \quad x \equiv y \pmod{q} \quad \Longrightarrow \quad x \equiv y \pmod{pq}\,.$$

Somit folgt für $x = M^{r\varphi(N)+1}$ und $y = M$ auch das gewünschte

$$M^{r\varphi(N)+1} \equiv M \pmod{N}\,.$$

Beispiel: RSA-Verfahren

1. Bob wählt Primzahlen $p = 3$ und $q = 11$, berechnet

$$N = 3 \cdot 11 = 33\,, \quad \varphi(N) = 2 \cdot 10 = 20\,,$$

wählt $e = 7$ (teilerfremd zu $\varphi(N) = 20$) und berechnet mit erw. Euklidischem Algorithmus

$$d = 3$$

$\Rightarrow$ öffentlicher Schlüssel: $(7, 33)$ und privater Schlüssel: $(3, 33)$

2. Alice möchte $M = 4$ senden und berechnet

$$4^7 = 16384 = 496 \cdot 33 + 16$$

$\Rightarrow C = 16 \equiv 4^7 \pmod{33}$

3. Bob empfängt $C = 16$ und berechnet

$$16^3 = 4096 = 124 \cdot 33 + 4$$

$\Rightarrow M' = M = 4$

Beispiel aus dem Originalartikel

VIII A Small Example

Consider the case $p = 47, q = 59, n = p \cdot q = 47 \cdot 59 = 2773$, and $d = 157$. Then $\phi(2773) = 46 \cdot 58 = 2668$, and e can be computed as follows:

$$
\begin{array}{lll}
x_0 = 2668, & a_0 = 1, & b_0 = 0, \\
x_1 = 157, & a_1 = 0, & b_1 = 1, \\
x_2 = 156, & a_2 = 1, & b_2 = -16 \text{ (since } 2668 = 157 \cdot 16 + 156) \text{ ,} \\
x_3 = 1, & a_3 = -1, & b_3 = 17 \text{ (since } 157 = 1 \cdot 156 + 1) \text{ .}
\end{array}
$$

Therefore $e = 17$, the multiplicative inverse $\pmod{2668}$ of $d = 157$.

With $n = 2773$ we can encode two letters per block, substituting a two-digit number for each letter: blank = 00, A = 01, B = 02, ..., Z = 26. Thus the message

```
ITS ALL GREEK TO ME
```

(Julius Caesar, I, ii, 288, paraphrased) is encoded:

```
0920 1900 0112 1200 0718 0505 1100 2015 0013 0500
```

Since $e = 10001$ in binary, the first block ($M = 920$) is enciphered:

$$M^{17} = (((((1)^2 \cdot M)^2)^2)^2)^2 \cdot M = 948 \pmod{2773} .$$

Sicherheit von RSA

Nachricht M kann nur schwer aus $C \equiv M^e \pmod{N}$ mithilfe des öffentlichen Schlüssels (e, N) berechnet werden, da

- in $\mathbb{Z}/N\mathbb{Z}$ kein effizientes Verfahren zum „Wurzelziehen“ bekannt ist
  $\longrightarrow$ diskreter Logarithmus
- kein effizientes Verfahren zur Berechnung von $\varphi(N)$ bekannt ist
  $\longrightarrow$ so schwer wie Primfaktorisierung von N

Aber:

- für die praktische Anwendung sollten wichtige Nebenbedingungen für die Wahl von p, q und e beachtet werden
- vollständige Sicherheit gibt es nicht
- mit „sehr großer“ Rechenleistung kann jede RSA-verschlüsselte Nachricht entschlüsselt werden

RSA-Factoring Challenge

RSA Number	Decimal digits	Binary digits	Cash prize offered	Factored on	Factored by
RSA-100	100	330	US$1,000[4]	April 1, 1991[5]	Arjen K. Lenstra
RSA-110	110	364	US$4,429[4]	April 14, 1992[5]	Arjen K. Lenstra and M.S. Manasse
RSA-120	120	397	$5,898[4]	July 9, 1993[6]	T. Denny et al.
RSA-129 [**]	129	426	$100 USD	April 26, 1994[5]	Arjen K. Lenstra et al.
RSA-130	130	430	US$14,527[4]	April 10, 1996	Arjen K. Lenstra et al.
RSA-140	140	463	US$17,226	February 2, 1999	Herman te Riele et al.
RSA-150 [*] ?	150	496		April 16, 2004	Kazumaro Aoki et al.
RSA-155	155	512	$9,383[4]	August 22, 1999	Herman te Riele et al.
RSA-160	160	530		April 1, 2003	Jens Franke et al., University of Bonn
RSA-170 [*]	170	563		December 29, 2009	D. Bonenberger and M. Krone [***]
RSA-576	174	576	$10,000 USD	December 3, 2003	Jens Franke et al., University of Bonn
RSA-180 [*]	180	596		May 8, 2010	S. A. Danilov and I. A. Popovyan, Moscow State University[7]
RSA-190 [*]	190	629		November 8, 2010	A. Timofeev and I. A. Popovyan
RSA-640	193	640	$20,000 USD	November 2, 2005	Jens Franke et al., University of Bonn
RSA-200 [*] ?	200	663		May 9, 2005	Jens Franke et al., University of Bonn
RSA-210 [*]	210	696		September 26, 2013[8]	Ryan Propper
RSA-704 [*]	212	704	$30,000 USD	July 2, 2012	Shi Bai, Emmanuel Thomé and Paul Zimmermann
RSA-220	220	729		May 13, 2016	S. Bai, P. Gaudry, A. Kruppa, E. Thomé and P. Zimmermann
RSA-230	230	762			
RSA-232	232	768			
RSA-768 [*]	232	768	$50,000 USD	December 12, 2009	Thorsten Kleinjung et al.
RSA-240	240	795			
RSA-250	250	829			
RSA-260	260	862			
RSA-270	270	895			
RSA-896	270	896	$75,000 USD		
RSA-280	280	928			
RSA-290	290	962			
RSA-300	300	995			
RSA-309	309	1024			
RSA-1024	309	1024	$100,000 USD		

RSA-Factoring Challenge

RSA-768 IS FACTORED!

A six-institution research team led by T. Kleinjung has successfully factored the RSA-768 challenge number. While the RSA Factoring Challenge is no longer active, the factoring of RSA-768 represents a major milestone for the community. The factors were found on December 12, 2009 and reported shortly thereafter. The academic paper describing the work can be found at: http://eprint.iacr.org/2010/006.pdf.

The factors are:

33478071698956898786044169848212690817704 7
94983713768568912431388982883793878002287 6
14711652531743087737814467999489

and

36746043666799590428244633799627952632279 15
81643430876426760322838157396665112792333 73
41714339681027009279873630891 7

The effort took almost 2000 2.2GHz-Opteron-CPU years according to the submitters, just short of 3 years of calendar time.