

# LU04.A05 - RSA-Verschlüsselungsverfahren

## Lernziele

- Ich kann das RSA-Verschlüsselungsverfahren anhand einer mathematischen Aufgabe nachvollziehen.
- Ich kann öffentliche und private Schlüssel generieren und verwenden, um Nachrichten zu verschlüsseln und zu entschlüsseln.

## Rahmenbedingungen

- **Zeitbudget:** 45 Minuten
- **Sozialform:** Einzelarbeit
- **Hilfsmittel:** Taschenrechner oder Computer mit Zugriff auf eine Programmierumgebung
- **Erwartetes Ergebnis:** Ausgefülltes Arbeitsblatt mit Ihren Berechnungen und Erklärungen

## Ausgangslage

Das RSA-Verschlüsselungsverfahren ist ein weit verbreitetes asymmetrisches Kryptosystem, das für sichere Datenübertragungen verwendet wird.

## Arbeitsauftrag

- **Schlüsselgenerierung:**
  1. Wählen Sie zwei unterschiedliche Primzahlen  $p$  und  $q$ .
  2. Berechnen Sie das Produkt  $n = p * q$ , welches als Modulus für die Schlüssel dient.
  3. Ermitteln Sie die totient Funktion  $\phi(n) = (p-1)(q-1)$ .
  4. Wählen Sie eine ganze Zahl  $e$ , die zu  $\phi(n)$  teilerfremd ist und kleiner als  $\phi(n)$  ist.
  5. Berechnen Sie den privaten Exponenten  $d$ , sodass  $e * d \equiv 1 \pmod{\phi(n)}$ .
- **Öffentlicher Schlüssel (Public Key):** Der öffentliche Schlüssel besteht aus dem Paar  $(n, e)$ .
- **Privater Schlüssel (Private Key):** Der private Schlüssel besteht aus dem Paar  $(n, d)$ .
- **Verschlüsselung:**
  1. Verschlüsseln Sie die Nachricht  $m = 123$ , indem Sie  $c = m^e \pmod{n}$  berechnen. Verwenden Sie dafür den öffentlichen Schlüssel.
- **Entschlüsselung:**
  1. Entschlüsseln Sie den Chiffretext  $c$ , indem Sie  $m = c^d \pmod{n}$  berechnen. Verwenden Sie dafür den privaten Schlüssel.
- **Verifikation:**
  1. Überprüfen Sie, dass der entschlüsselte Text mit der ursprünglichen Nachricht  $m$  übereinstimmt.

## Verwenden eines Simulators

Verwenden Sie CrypTools um die Schritte zu veranschaulichen:

<https://www.cryptool.org/en/cto/rsa-step-by-step>

## Theorie: Berechnung von Kongruenzen

Um die Kongruenz  $a \equiv b \pmod{m}$  zu berechnen, folgen Sie diesen Schritten:

- **Modulo-Operation durchführen:**

1. Berechnen Sie den Rest der Division von  $a$  durch  $m$ , bezeichnet als  $a \bmod m$ .
2. Berechnen Sie den Rest der Division von  $b$  durch  $m$ , bezeichnet als  $b \bmod m$ .

- **Vergleich der Reste:**

1. Wenn die Reste gleich sind, d.h.,  $a \bmod m$  ist gleich  $b \bmod m$ , dann gilt die Kongruenz  $a \equiv b \pmod{m}$ .

**Beispiel:** - Um  $17 \equiv x \pmod{5}$  zu berechnen, bestimmen Sie den Rest von 17 geteilt durch 5. Da  $17 \bmod 5 = 2$ , suchen Sie nach einem Wert von  $x$ , der ebenfalls einen Rest von 2 ergibt, wenn er durch 5 geteilt wird. Jede Zahl, die um ein Vielfaches von 5 plus 2 ist (z.B. 7, 12, 22, ...), würde diese Bedingung erfüllen.

## Theorie: Sicherheit des RSA-Algorithmus

Die Sicherheit des RSA-Algorithmus basiert nicht auf der Schwierigkeit der Kongruenzberechnung, sondern auf dem Problem der Faktorisierung großer Zahlen.

- **Faktorisierungsproblem:**

1. Im RSA-Algorithmus wird das Produkt zweier großer Primzahlen  $p$  und  $q$  verwendet, um  $n = p \times q$  zu bilden. Der öffentliche Schlüssel enthält  $n$  und einen Exponenten  $e$ , während der private Schlüssel aus einem anderen Exponenten  $d$  besteht.
2. Der private Schlüssel  $d$  wird durch die Berechnung von  $e^{-1} \bmod \varphi(n)$  ermittelt, wobei  $\varphi(n) = (p-1) \times (q-1)$ . Um  $\varphi(n)$  zu berechnen, muss man  $n$  faktorisieren.
3. Das Problem der RSA-Sicherheit basiert auf der Schwierigkeit, eine große Zahl  $n$  in ihre Primfaktoren  $p$  und  $q$  zu zerlegen. Für große Zahlen wird dies extrem schwierig und zeitaufwändig, selbst mit leistungsfähigen Computern.

- **Kongruenzberechnung im RSA:**

1. Die Kongruenzberechnung kommt ins Spiel, wenn Nachrichten verschlüsselt oder entschlüsselt werden (z.B.  $c = m^e \bmod n$  für die Verschlüsselung und  $m = c^d \bmod n$  für die Entschlüsselung). Diese Operationen sind selbst für sehr große Zahlen effizient durchführbar.

Zusammenfassend beruht die Sicherheit des RSA-Algorithmus auf der Schwierigkeit, große Zahlen zu faktorisieren, nicht auf der Schwierigkeit der Kongruenzberechnung. Die Komplexität und Sicherheit des RSA-Algorithmus erhöht sich mit der Länge der verwendeten Schlüssel.

From:

<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:

<https://wiki.bzz.ch/modul/m183/learningunits/lu04/aufgaben/05?rev=1752152602>

Last update: **2025/07/10 15:03**

