RSA

Luc Spachmann

Friedrich-Schiller-Universität Jena

03.12.2021

RSA

- Schlüssel (e, n) und (d, n) gegeben mit

$$ed = 1 \mod \varphi(n)$$

  und n Produkt zweier Primzahlen
- φ ist Eulersche φ-Funktion
- Für Produkt zweier Primzahlen pq gilt

$$\varphi(pq) = (p-1)(q-1)$$

- Text wird dargestellt als Folge von Zahlen $x_1, ..., x_n < n$
- Verschlüsselung jeder Zahl $y_i = x_i^e \mod n$
- Entschlüsselung $x_i = y_i^d \mod n$
- Frage: Effektive Berechnung von x^e

Quadrieren und Multiplizieren

- Effektiver Algorithmus für Potenzen Modulo n
- Ähnlich russischer Bauernmultiplikation
- Berechnen $x^m \mod n$
- Sei $m = b_0 + b_1 \cdot 2 + b_2 \cdot 2^2 + ... + b_r \cdot 2^r$ mit $b_i \in \{0, 1\}$

Pseudocode

Require: x, m, n

$y = 1$
for $i = 0, \ldots, r$ **do**
    if $b_i = 1$ **then**
        $y = y \cdot x \mod n$
    end if
    $x = x^2 \mod n$
end for
return y

Erweiterter euklidischer Algorithmus

- Berechnet $\mathrm{ggT}(a, b) = sa + tb = c$

Require: a, b

$k = 0, r_0 = a, r_1 = b, s_0 = 1, s_1 = 0, t_0 = 0, t_1 = 1$
repeat
    Erhöhe k um 1
    $q_k = r_{k-1} \div r_k$
    $r_{k+1} = r_{k-1} - q_k \cdot r_k$
    $s_{k+1} = s_{k-1} - q_k \cdot s_k$
    $t_{k+1} = t_{k-1} - q_k \cdot t_k$
until $r_{k+1} = 0$
return r_k, s_k, t_k

Aufgaben

- Implementiert RSA mithilfe Quadrieren und Multiplizieren
- Erlaubt Zahlen in Größenordnungen von bis zu 2^{2000}
- Implementiert den erweiterten euklidischen Algorithmus
- Beispiel Schlüssel zu verschlüsseln: $(e = 53, n = 77)$
- Es gilt $77 = 11 \cdot 7$
- Findet Schlüssel zu entschlüsseln d mit $ed = 1 \mod \varphi(n)$ mithilfe des eeA