Verschlüsselung von Texten durch RSA

n wird sehr groß. Wir haben $\mathcal{P} = \mathcal{C} = \mathbb{Z}_n$ gewählt. Aber in Realität kann unser Alphabet $\mathcal{A}$ die Mächtigkeit N haben, $N << n$, $\mathcal{A} = \{0, \dots, N-1\} \subseteq \mathbb{Z}_n$.

Wir wählen dann $\mathcal{P} = \mathcal{A}^\infty = \mathcal{C}$, indem wir

Einen beliebig langen Text zerlegen wir in Blöcke der Länge k, $k = \lfloor \log_N n \rfloor$. *1 Ein solcher Block $a_1 \dots a_k$, $a_i \in \mathcal{A}$, wird dann geschrieben als

$$m = \sum_{i=1}^{k} a_i N^{k-i}. \quad *2$$

Dann ist $0 \le m \le (N-1)\left(\sum_{i=1}^{k} N^{k-i}\right) = N^k - 1 < n$.

Daher können wir m durch $c = e_B(m) = m^e \bmod n$ verschlüsseln. Die Zahl c kann wieder mittels der N-adischen Entwicklung geschrieben werden:

$$c = \sum_{i=0}^{k} c_i N^{k-i}, \quad c_i \in \mathcal{A}$$

/ Definition & Bsp. der N-adischen Entwicklung

$c_0 \dots c_k$ wird dann gesendet.

Beispiel $p = 11$, $q = 23$, $n = 253$, $\varphi(n) = (p-1)(q-1) = 10 \cdot 22 = 4 \cdot 5 \cdot 11$. Wähle $e = 3$. Es folgt $d = 147$ $3 \cdot 147 = 441 \equiv$

$A = \{0, a, b, c\} = \{0, 1, 2, 3\} \Rightarrow N = 4$

$k = \lfloor \log_4 253 \rfloor = 3 \quad (253 = 4^{3,\ldots})$.

Verschlüsselabb $\sim$ 122

$\Rightarrow m = 1 \cdot N^{3-1} + 2 \cdot N^{3-2} + 2 \cdot N^{3-3} = N^2 + 2N + 2$

$= 16 + 8 + 2 = 26$

$c = m^e = 26^3 \equiv 119 \mod 253$.

Es ist $c = 1 \cdot 4^3 + 3 \cdot 4^2 + 1 \cdot 4 + 3 \cdot 1$.

Der Schlüsseltextblock (Chiffretext) ist

also $1313 \sim acac$.

Effizienz

Wie berechnen wir $m^e \bmod n$? Wir bestimmen die 2-adische (binäre) Entwicklung von e:

$$e = \sum_{i=0}^{r} e_i \cdot 2^i \quad , e_i \in \{0, 1\},$$

und quadrieren m r-mal modulo n.

Also wir berechnen $m^2 \bmod n$, $m^4 \bmod n$, $\ldots$, $m^{2^r} \bmod n$.
$(m^2)^2 \bmod n$ $(m^{2^{r-1}})^2 \bmod n$

Es ist

$$m^e = m^{\sum_{i=0}^{r} e_i 2^i} = \prod_{i=0}^{r} m^{e_i 2^i} = \prod_{i=0}^{r} \left(m^{2^i}\right)^{e_i},$$

wobei $m^{2^i} = \left(m^{2^{i-1}}\right)^2$ ist.

Beispiel $m = 13, e = 7, n = 15 \quad e = 2^2 + 2 + 1$

$m^2 = 13^2 \equiv (-2)^2 = 4 \bmod 15, \quad m^4 \equiv 4^2 = 16 \equiv 1 \bmod 15 \Rightarrow$
$m^7 = m^4 m^2 m \equiv 1 \cdot 4 \cdot 13 \equiv 4(-2) = -8 \equiv 7 \bmod 15.$

Um m^7 zu berechnen, muß also m 2 mal quadriert werden.

Im Allgemeinen ist e relativ klein und d relativ groß. Falls l die Bitlänge von d ist, dann brauchen wir zur Entschlüsselung – also zur Berechnung von $c^d \bmod n$ – l Quadrierungen modulo n und, falls in der Binärentwicklung von d die Hälfte der Bits den Wert 1 hat, $l/2$ Multiplikationen modulo n. $l = 1024 \Rightarrow$ 1024 Quadrierungen und 526 Multiplikationen. Das ist aufwendig!

Daher wird der chinesische Restsatz benutzt um das Verfahren zu beschleunigen:

(Wir müssen auch p und q kennen!!!)
um den chin. Restsatz anwenden zu können

$$m_p \equiv c^d \bmod p \qquad (n = p \cdot q)$$

$$m_q \equiv c^d \bmod q$$

Da $p, q << n$ (wesentlich kleiner als n), sind m_p und m_q wesentlich leichter als $c^d \bmod n$ zu berechnen. m löst dann die beiden Kongruenzen:

$$m \equiv m_p \bmod p$$

$$m \equiv m_q \bmod q$$

m erhalten wir also durch:

Ist $x_1 p + x_2 q = 1$, dann folgt

$$m = (m_p x_2 q + m_q x_1 p) \bmod n$$

$x_2 q \bmod n$ und $x_1 p \bmod n$ können vorher berechnet und gespeichert werden.

Beispiel $n = 133 = 7 \cdot 19 \Rightarrow \varphi(n) = 6 \cdot 18 = 108$

$\Rightarrow e = 5$ ist ein zulässiger Schlüssel

$(ggT(e, \varphi(n)) = ggT(5, 108) = 1)$.

$\Rightarrow d = 65$

Alice hat von Bob $c = 48$ erhalten.

Sie muß nun $48^{65} \bmod 133$ berechnen. Dafür geht sie wie oben beschrieben vor:

$65 = 64 + 1 = 2^6 + 1$. Es muß also 6mal quadriert werden.

Sie rechnet modulo 7 und 19:

$-1 \equiv 48 \bmod 7 \quad \Rightarrow \quad m_7 = -1$

$10 \equiv 48 \bmod 19 \qquad 10^2 \equiv 5 \bmod 19$

$10^4 \equiv 5^2 \equiv 6 \bmod 19$

$10^8 \equiv 6^2 \equiv -2 \bmod 19$

$10^{16} \equiv (-2)^2 \equiv 4 \bmod 19$

$10^{32} \equiv 4^2 \equiv 16 \bmod 19$

$10^{64} \equiv 16^2 \equiv (-3)^2 = 9 \bmod 19$

$48^{65} = 48^{2^6} \cdot 48$

$\equiv 9 \cdot 10 = 90 \equiv -5 \bmod 19$

$\Rightarrow m_{19} = -5$

$3 \cdot 19 - 8 \cdot 7 = 1 \quad \Rightarrow \quad x_1 = -8, \; x_2 = 3$

$m = (m_7 \, x_2 \, 19 + m_{19} \, x_1 \, 7) \bmod 133$

$= -1 \cdot 3 \cdot 19 + (-5) \cdot (-8) \cdot 7 = -57 + 280 = 227 \equiv 94 \bmod 133$

$\Rightarrow m = 94$

Der geheime Schlüssel d

d muss unbedingt geheim gehalten werden, da mit d entschlüsselt werden kann.

Der Angreifer versucht d zu berechnen.

Falls die Faktorisierung $n = p \cdot q$ bekannt ist, dann kann $\varphi(n)$ und daher auch d mit dem erweiterten euklidischen Algorithmus berechnet werden. Wichtig ist also auch p und q geheim zu halten. d kann schon ~~aus~~ ~~$\varphi(n)$~~ berechnet werden, wenn $\varphi(n)$ bekannt ist. Aus $\varphi(n)$ ergeben sich auch die Primfaktoren p und q:

p und q sind die Nullstellen von

$x^2 - (n+1-\varphi(n))\,x + n$, da

$$x^2 - (n+1-\varphi(n))\,x + n = x^2 - (p\cdot q + 1 - (p-1)(q-1))\,x + p\cdot q = x^2 - (pq+1-pq+p+q-1)\,x + p\cdot q$$

$$= x^2 - (p+q)\,x + p\cdot q$$

$$= (x-p)(x-q).$$

Mit großer Wahrscheinlichkeit können sogar p und q aus d bestimmt werden $\Rightarrow$ es ist ungefähr

genauso schwer n in $p \cdot q$ zu faktorisieren wie d zu bestimmen. (und wie $\varphi(n)$ zu bestimmen)

7. Primzahltests

Um RSA benutzen zu können, müssen große Primzahlen gefunden werden. Es gibt genügend davon, wie wir wissen, aber wie können wir sie finden? Die Strategie ist eine beliebige genügend große ungerade Zahl n auszuwählen und dann zu testen, ob sie eine Primzahl ist. Sei $\mathbb{P}$ die Menge der Primzahlen. Wir hatten definiert $\pi(x) = |\{ p \in \mathbb{P} \mid p \leq x \}|$ für $x \in \mathbb{R}^+$.
Nach dem Primzahlsatz ist

$$\pi(x) \sim \frac{x}{\log x}.$$

Daher $\frac{\pi(n)}{n} \sim \frac{1}{\log n}$. Wenn wir also eine beliebige Zahl n wählen (in der richtigen Größe) dann ist in einer $\frac{\Delta}{\log n}$ - Umgebung sehr

wahrscheinlich eine Primzahl.

Also ist $n \approx 2^{512}$, dann ist in einer $\log n \approx 355$ Umgebung von n eine Primzahl. Es sind also nicht viele Zahlen zu überprüfen.

Wie können wir überprüfen?

7.1 Probedivision

Falls n durch keine Primzahl $p \leq \sqrt{n}$ teilbar ist, dann ist $n \in \mathbb{P}$.

Das zu überprüfen ist sehr aufwendig!

In der Praxis wird eine Liste der kleinen Primzahlen gespeichert und n durch diese geteilt, bevor ein weiterer Test durchgeführt wird.

Beispiel Teile $n = 187$ nacheinander durch $3, 5, 7, 11$ und erhalte so $n = 11 \cdot 17$.

Ein anderer Test ist

7.2 Der Fermat-Test

Ist p eine Primzahl und $1 \leq a < p$, dann gilt nach dem Kleinen Satz von Fermat

$$a^{p-1} \equiv 1 \bmod p.$$

Auf dieser Aussage beruht der Fermat-Test:

Sei $n \in \mathbb{N}$ und $1 \neq a < n$.

Ist $a^{n-1} \not\equiv 1 \bmod n \Rightarrow n \notin \mathbb{P}$.

Falls $a^{n-1} \not\equiv 1 \bmod n$, dann sagen wir, dass a ein Zeuge für die Zusammengesetztheit von n ist.

Beispiel $n = 6 \quad a = 3 \Rightarrow 3^5 \equiv 9^2 \cdot 3 \equiv 3^2 \cdot 3$

$$\equiv 3 \cdot 3 \equiv 3 \bmod 6$$

also $a = 3$ ist Zeuge der Zusammengesetztheit von $n = 6$.

Falls aber $a^{n-1} \equiv 1 \bmod n$ gilt, obwohl n keine Primzahl ist, dann heißt n Pseudoprimzahl zur Basis a.

Beispiel

$n = 341 \quad a = 2 \Rightarrow a^{340} = 2^{340} \equiv 1 \bmod 341$

Aber $341 = 11 \cdot 31$.

Also 341 ist Pseudoprimzahl zur Basis $a = 2$.