2.8 Kleine Exponenten

Frage: Ist es beim RSA-Verfahren gefährlich, einen kleinen öffentlichen Exponenten e zu wählen?

Der Exponent $e = 1$ ist völlig unbrauchbar, da dann nicht verschlüsselt wird.

Der Exponent $e = 2$ ist für RSA nicht verwendbar, da er gerade und somit nicht teilerfremd zu $\lambda(n)$ ist. Es gibt allerdings eine verwandte Chiffre, das RABIN-Verfahren, das gerade mit $e = 2$ so arbeitet. Bei diesem muss der Empfänger Quadratwurzeln $\bmod n$ ziehen können, und das kann er, wenn er n faktorisieren kann (siehe später). (Er muss auch noch eine Möglichkeit haben, unter den verschiedenen Quadratwurzeln den richtigen Klartext zu erkennen.)

Gleiche Nachricht an mehrere Empfänger

Der kleinste für das RSA-Verfahren prinzipiell geeignete Exponent ist $e = 3$. Hier entsteht eine Schwäche, sobald jemand die gleiche Nachricht a an *drei* verschiedene Teilnehmer A, B und C schickt. Deren öffentliche Schlüssel seien $(n_\mathrm{A}, 3)$, $(n_\mathrm{B}, 3)$ und $(n_\mathrm{C}, 3)$. O. B. d. A. sind die Moduln n_A, n_B und n_C paarweise teilerfremd, sonst kann die Angreiferin mindestens zwei davon faktorisieren, also erst recht a lesen. Ansonsten hat sie (mit etwas Glück) drei Geheimtexte

$$c_\mathrm{A} = a^3 \bmod n_\mathrm{A}, \quad c_\mathrm{B} = a^3 \bmod n_\mathrm{B}, \quad c_\mathrm{C} = a^3 \bmod n_\mathrm{C}$$

mit $0 \le a < n_\mathrm{A}, n_\mathrm{B}, n_\mathrm{C}$, also $a^3 < n_\mathrm{A} n_\mathrm{B} n_\mathrm{C}$. Mit dem chinesischen Restalgorithmus konstruiert sie daraus eine Zahl $\tilde{c} \in \mathbb{Z}$ mit

$$0 \le \tilde{c} < n_\mathrm{A} n_\mathrm{B} n_\mathrm{C}$$

und

$$\tilde{c} \equiv c_\mathrm{X} \bmod n_\mathrm{X} \quad \text{für } \mathrm{X} = \mathrm{A}, \mathrm{B}, \mathrm{C}.$$

Wegen der Eindeutigkeit ist $\tilde{c} = a^3$ in $\mathbb{Z}$. Damit bestimmt sie $a = \sqrt[3]{\tilde{c}}$ durch Wurzelziehen in $\mathbb{Z}$; das ist effizient möglich. (Auch hier gelingt es ihr nicht, die geheimen Exponenten zu bestimmen.)

Die Verallgemeinerung dieses Angriffs auf einen beliebigen „kleinen“ gemeinsamen öffentlichen Exponenten ist offensichtlich: Wenn eine identische Nachricht an e verschiedene Teilnehmer geschickt wird, kann diese von jedem gelesen werden. So abwegig ist dieser Angriffspunkt nicht, wenn man sich etwa eine konstante „Protokollinformation“ am Beginn längerer Nachrichten vorstellt. In der Praxis wird der Exponent $e = 2^{16} + 1 = 65537$ meist als hinreichend sicher für „normale“ Anwendungen angesehen.

Stereotype Teilnachrichten

Seien (n, e, d) die aktuellen Schlüsselparameter. Wir stellen uns eine Situation eines Angriffs mit bekanntem Klartext vor. Der Klartext laute:

```
Der heutige Tagesschlüssel ist: ********
```

(Beispiel von Julia Dietrichs) mit bekanntem Teil (Stereotyp) „`Der heutige Tagesschlüssel ist: `“ und unbekanntem 8 Byte langem Teil „`********`“.

Diese Nachricht wird in dem (in Westeuropa üblichen) 8-Bit-Zeichensatz ISO-8859-1 zu einer Bitfolge aus 40 Bytes oder 320 Bits, die für das RSA-Verfahren als ganze Zahl $a \in [0 \dots n-1]$ interpretiert wird (wenn n mehr als 320 Bits hat). Diese lässt sich zerlegen in $a = b + x$, wobei b dem bekannten stereotypen Teil und x dem unbekannten Teil entspricht. Da dieser am Ende der Nachricht steht und aus 64 Bit besteht, ist $x < 2^{64}$. Durch die Verschlüsselung wird daraus der Geheimtext

$$c = a^e \bmod n = (b + x)^e \bmod n.$$

Das Geheimnis x ist also Nullstelle des Polynoms

$$(T + b)^e - c \in (\mathbb{Z}/n\mathbb{Z})[T].$$

Das ist zunächst nicht weiter aufregend, da es keine allgemeinen effizienten Algorithmen gibt, um solche Nullstellen zu bestimmen. Allerdings gibt es solche Algorithmen für Spezialfälle, z. B. den

> **Algorithmus von** Coppersmith
> Sei $f \in (\mathbb{Z}/n\mathbb{Z})[T]$ ein Polynom vom Grad r. Der Algorithmus findet alle Nullstellen x von f mit $0 \leq x < \sqrt[r]{n}$ (oder beweist, dass es keine solchen gibt).
> Die Laufzeit ist polynomial in $\log n$ und r.
> (Der Algorithmus verwendet den „LLL-Algorithmus“ zur Reduktion von Gitterbasen.)

Da in unserem Beispiel n mindestens 321 Bit lang und $e = 3$ ist, lässt sich x bestimmen, da $x^3 < 2^{192} < 2^{320} < n$.

Dieser Angriff ist nur ein erstes Beispiel einer ganzen Klasse von Angriffen, die in speziellen Situation auf eine Nullstellenbestimmung $\bmod n$ hinauslaufen.

Übungsaufgabe. Wie kann man den obigen Angriff für den Fall modifizieren, dass der unbekannte Klartext aus einigen zusammenhängenden Zeichen mitten zwischen bekanntem Klartext steht?