Die Faktorisierung von RSA-768

Thorsten Kleinjung
(École Polytechnique Fédérale de Lausanne)

thorsten.kleinjung@epfl.ch

Einleitung

Ein altes Problem der algorithmischen Zahlentheorie ist die Frage, wie die Primfaktorzerlegung einer natürlichen Zahl möglichst effizient gefunden werden kann. Dieses Problem spaltet sich in zwei Teilprobleme, nämlich eine Zahl auf Primalität zu testen und eine zusammengesetzte Zahl N in ein nicht triviales Produkt zweier natürlicher Zahlen zu zerlegen (rekursiv kann dann die Primfaktorzerlegung gefunden werden). Das erste Problem ist seit kurzer Zeit durch einen deterministischen Polynomialzeitalgorithmus gelöst, und schon seit längerem gibt es probabilistische Primalitätstests, die in der Praxis schneller sind.

Das zweite Problem – N nicht trivial zu zerlegen – ist wesentlich schwieriger, und es gibt eine Vielzahl von Algorithmen dafür, aber bisher noch keinen Polynomialzeitalgorithmus. Beispielsweise gibt es Algorithmen, deren Laufzeit hauptsächlich von der Größe des kleinsten Primfaktors p von N abhängt. Mit solchen Algorithmen kann man kleine Primfaktoren recht schnell finden. Wenn der kleinste Primfaktor aber relativ groß ist, müssen andere Algorithmen benutzt werden, z. B. das Zahlkörpersieb, das momentan der beste bekannte Algorithmus für solche Zahlen ist.

Seit einigen Jahrzehnten sorgt auch die Kryptologie für verstärktes Interesse an Faktorisierungsalgorithmen, da ein bekanntes und vielbenutztes Verschlüsselungsverfahren (RSA) auf der Schwierigkeit des Faktorisierungsproblems basiert. Die bei diesem Verfahren benutzten Zahlen sind Produkte von zwei etwa gleich großen Primzahlen, so dass sie (ab einer gewissen Größe) sinnvollerweise nur mit dem Zahlkörpersieb faktorisiert werden können. Eine dieser Zahlen, RSA-768, die aus einer alten „challenge list“ der Firma RSA stammt, haben wir kürzlich im Rahmen eines zweieinhalb Jahre dauernden Projektes faktorisiert[1]. An dem Projekt waren Institutionen aus vielen Ländern beteiligt: BSI und Universität Bonn (Deutschland), INRIA (Frankreich), NTT (Japan), CWI (Niederlande), EPFL (Schweiz) und einige weitere Personen, die Rechenzeit zur Verfügung gestellt haben. Diese Faktorisierung stellt einen neuen Rekord dar, der den vorherigen Rekord von 2005 (Faktorisierung einer 200-stelligen Zahl) um 32 Dezimalstellen übertrifft. Die Rechenzeit hierfür betrug fast 2000 Jahre auf einem Prozessorkern, und da der Aufwand für die nächste kryptologisch interessante Größe von 1024 Bit (309 Dezimalstellen) etwa tausendmal höher liegt, ist mit einer solchen Faktorisierung erst in etwa fünf bis fünfzehn Jahren zu rechnen.

Neben der Beschreibung des Zahlkörpersiebes soll hier auch auf den neuen Rekord eingegangen werden. Zuvor werden aber erst einige einfachere Faktorisierungsalgorithmen behandelt, die schon wichtige Elemente des Zahlkörpersiebes enthalten. Auf historisch wichtige Algorithmen wie z. B. den Kettenbruchalgorithmus oder Schroeppels lineares Sieb gehen wir nicht ein.

Moderne Faktorisierungsalgorithmen

Viele moderne Faktorisierungsalgorithmen konstruieren eine „zufällige“ Lösung $x, y \in \mathbb{Z}$ von

$$x^2 \equiv y^2 \pmod{N}, \tag{1}$$

woraus man für $\mathrm{ggT}(N, 2xy) = 1$ wegen

$$N \mid (x+y)(x-y)$$

die Zerlegung

$$N = \mathrm{ggT}(N, x+y) \cdot \mathrm{ggT}(N, x-y)$$

bekommt. Ein Primteiler $p \mid N$ kommt mit Wahrscheinlichkeit $\frac{1}{2}$ im ersten Faktor vor, wenn x und y nicht durch p teilbar sind und zufällig modulo p gewählt wurden. Wenn N also mindestens zwei verschiedene Primfaktoren enthält, bekommen wir auf diese Weise mit

[1] Ein Artikel darüber findet sich unter http://eprint.iacr.org/2010/006.

Wahrscheinlichkeit $\geq \frac{1}{2}$ eine Zerlegung von N. In der Art und Weise, wie solche Lösungen von (1) konstruiert werden, unterscheiden sich die im Folgenden beschriebenen Algorithmen.

In Dixons Algorithmus werden Relationen der Form

$$x_i^2 \equiv r_i = \prod_{p \leq B} p^{v_p(r_i)} \pmod{N}$$

konstruiert, indem zufällige Zahlen x_i modulo N erzeugt werden. Diese liefern eine Relation, sofern der Rest r_i von x_i^2 bei Division durch N eine B-glatte Zahl ergibt, d. h. in Primfaktoren $\leq B$ zerfällt. Wenn mehr als $\pi(B)$ (das ist die Anzahl der Primzahlen $\leq B$) Relationen $x_i^2 \equiv r_i$ erzeugt wurden, kann man eine Teilmenge I der Relationen finden, für die $2 \mid \sum_{i \in I} v_p(r_i)$ für alle Primzahlen p gilt. Damit ist

$$\left(\prod_{i \in I} x_i\right)^2 \equiv \left(\prod_{p \leq B} p^{\frac{1}{2}(\sum_{i \in I} v_p(r_i))}\right)^2 \pmod{N},$$

was wie oben beschreiben mit Wahrscheinlichkeit $\geq \frac{1}{2}$ zu einer Zerlegung von N führt (wir setzen voraus, dass N keinen Primfaktor $\leq B$ hat und dass alle x_i zu N teilerfremd sind). Obige Bedingung $2 \mid \sum_{i \in I} v_p(r_i)$ für die Teilmenge I kann leicht in ein lineares Gleichungssystem über $\mathbb{F}_2 = \mathbb{Z}/2\mathbb{Z}$ übersetzt werden. Die Grenze B darf dabei nicht zu klein gewählt werden, damit die Wahrscheinlichkeit, aus einem x_i eine Relation zu bekommen, nicht zu gering wird, aber auch nicht zu groß, da sonst zu viele Relationen erzeugt werden müssen und das lineare Gleichungssystem zu groß wird. Dixons Algorithmus besteht also aus zwei wesentlichen Schritten: der Erzeugung von Relationen und dem Lösen eines lineares Gleichungssystems über $\mathbb{F}_2$.

Im quadratischen Sieb werden die x_i nahe $\sqrt{N}$ gewählt, womit $x_i^2 - N$ von der Größenordnung $\sqrt{N}$ ist, also viel häufiger B-glatt ist. Außerdem gilt für eine Primzahl p, dass aus $p \mid x^2 - N$ auch $p \mid (x+p)^2 - N$ folgt, was die Anwendung einer Siebmethode zulässt (vergleichbar dem Sieb des Erathostenes, aber man merkt sich, welche Primzahl an welcher Stelle etwas bewirkt).

Das Zahlkörpersieb

Bei den oben beschriebenen Algorithmen sind die Zahlen, die auf Glattheit getesten werden müssen, mindestens von der Größenordnung $N^{\frac{1}{2}}$. Diese Größenordnung wird beim Zahlkörpersieb auf N^c mit $c \to 0$ für $N \to \infty$ gesenkt. Dies geschieht dadurch, dass zunächst zwei irreduzible Polynome $f_1, f_2 \in \mathbb{Z}[X]$ mit gemeinsamer Nullstelle m modulo N gewählt werden. Zur Vereinfachung der folgenden Darstellung nehmen wir an, dass die beiden Polynome normiert sind; für führende Koeffizienten ungleich 1 können die folgenden Schritte leicht angepasst werden. Diese Polynome liefern Homomorphismen

$$\begin{aligned} \phi_j : \mathbb{Z}[X]/(f_j) &\to \mathbb{Z}/N\mathbb{Z}, \\ X \bmod f_j &\mapsto m \bmod N, \end{aligned}$$

und wir bekommen die Kongruenzen

$$\phi_1(a - bX) \equiv \phi_2(a - bX) \pmod{N}$$

für $a, b \in \mathbb{Z}$ (um die Notation zu vereinfachen, bezeichnen wir die induzierten Homomorphismen

$$\mathbb{Z}[X] \to \mathbb{Z}[X]/(f_j) \to \mathbb{Z}/N\mathbb{Z}$$

auch mit ϕ_j). Das Ziel ist es nun, ein Produkt

$$g = \prod_{i \in I} a_i - b_i X$$

zu finden, so dass $g \bmod f_j$ für $j = 1, 2$ ein Quadrat in $\mathbb{Z}[X]/(f_j)$ ist. Das ist nach einiger Überlegung im Wesentlichen der Fall, wenn die Produkte

$$\prod_{i \in I} f_j(\frac{a_i}{b_i}) \cdot b_i^{\deg(f_j)}$$

Quadrate sind. Deshalb bezeichnen wir $a - bX \bmod f_j$ als B-glatt, wenn $f_j(\frac{a}{b}) \cdot b^{\deg(f_j)}$ B-glatt ist.

Bemerkung: Eigentlich betrachten wir oben die Zahlkörper $K_j = \mathbb{Q}[X]/(f_j)$, daher auch der Name des Verfahrens. Dann ist $f_j(\frac{a}{b}) \cdot b^{\deg(f_j)}$ die Norm des Bildes von $a - bX$ in K_j, und wir suchen ein Produkt solcher Elemente, das ein Quadrat in K_j ist.

Die einzelnen Schritte des Zahlkörpersiebes werden nun etwas genauer beschrieben, wobei jeweils auf die Faktorisierung von RSA-768 eingegangen wird. Die für die Faktorisierung benötigten Programme haben wir in C und Assembler programmiert, wobei an einigen Stellen GMP (für die Langzahlarithmetik) und Pari (für einige Berechnungen in Zahlkörpern) benutzt wurden. Für die parallelen Programme (Matrixschritt) war MPI wesentlich.

Polynomwahl:
Hier müssen die beiden Polynome $f_1, f_2 \in \mathbb{Z}[X]$ mit gemeinsamer Nullstelle m modulo N gewählt werden. Da es viele mögliche Wahlen für solche Polynompaare gibt, soll dasjenige gewählt werden, für das die nachfolgenden Schritte am schnellsten durchgeführt werden können.

Für RSA-768 hatten die Polynome f_1 und f_2 Grad 6 und Grad 1. Dieses Polynompaar war das Ergebnis einer umfangreichen Polynomsuche (ca. 40 Jahre auf einem Prozessorkern), bei der viele mögliche Polynompaare konstruiert und bewertet wurden.

Siebschritt:
In dieser Phase müssen genügend viele Paare $(a, b) \in \mathbb{Z} \times \mathbb{N}$ mit $\mathrm{ggT}(a, b) = 1$ gefunden werden, für die sowohl $f_1(\frac{a}{b}) \cdot b^6$ als auch $f_2(\frac{a}{b}) \cdot b$ B-glatt sind. Dies kann wie beim quadratischen Sieb durch ein Siebverfahren geschehen, wobei es auch hier einige Tricks gibt, um das Sieben zu beschleunigen (z. B. das Gittersieben,

das nur gewisse Paare (a, b) betrachtet, für die einer der Polynomwerte durch eine große Primzahl teilbar ist).

Im Fall von RSA-768 wurden etwa 10^{18} Paare (a, b) betrachtet, von denen 47 Milliarden Relationen lieferten, d. h. die beiden Polynomwerte $f_j(\frac{a}{b}) \cdot b^{\deg(f_j)}$ waren 2^{40}-glatt. Diese Phase dauerte fast zwei Jahre, und die eingesetzte Rechenleistung entsprach etwa 1500 Jahren auf einem modernen Prozessorkern.

Matrixschritt:
Aus den Relationen kann eine Matrix über $\mathbb{F}_2$ erstellt werden, die dann gelöst werden muss. Da die Matrix dünnbesetzt ist, können hier spezielle Verfahren, wie z. B. der Block-Wiedemann-Algorithmus, benutzt werden. Die Laufzeit dieser Verfahren hängt vom Produkt der Dimension und der Anzahl der nicht trivialen Einträge der Matrix ab, ist also bei dünnbesetzten Matrizen viel geringer als Gaußelimination. Außerdem liefern solche Verfahren im Allgemeinen nicht den gesamten Lösungsraum, sondern nur einen Teil davon, z. B. einen 512-dimensionalen Teilraum, was aber für diese Zwecke ausreicht.

Vor dem Lösen der Matrix sollte sie aber noch durch einige elementare Operationen (Entfernen von Zeilen und Spalten mit keinem oder einem Eintrag, Addition von Spalten) verkleinert werden. Damit kann der Aufwand für die Lösung des Gleichungssystems stark reduziert werden. Dabei muss aber darauf geachtet werden, dass die Matrix dünnbesetzt bleibt. Wenn die Matrix stark überbestimmt ist, können auch Spalten entfernt werden, um eine weitere Reduktion zu erreichen.

Die 47 Milliarden Relationen hätten ohne die Reduktion eine Matrix mit 47 Milliarden Spalten und 35 Milliarden Zeilen geliefert. Da diese stark überbestimmt war, konnte sie auf eine Matrix mit 193 Millionen Spalten und Zeilen und etwa 28 Milliarden nicht trivialen Einträgen reduziert werden.

Da im Wesentlichen die Positionen der nicht trivialen Einträge gespeichert werden müssen, konnten Cluster mit Speicher ab etwa 200 GB zur Lösung der Matrix beitragen. Die Matrix wurde innerhalb von 4 Monaten mit Clustern in Frankreich, Japan und der Schweiz gelöst, was etwa 10% des Siebaufwandes entsprach.

Im Gegensatz zum Siebschritt, der leicht auf viele kleine Computer verteilt werden kann, ist die Parallelisierung beim Matrixschritt schwieriger, und es werden auch Cluster mit großem Speicher benötigt.

Das Finale:
Jede Lösung der Matrix liefert durch Aufmultiplizieren der entsprechenden Relationen eine Kongruenz der Form

$$\phi_1\Big(\prod_{i\in I} a_i - b_i X\Big) \equiv \phi_2\Big(\prod_{i\in I} a_i - b_i X\Big) \pmod N.$$

Es gibt eine endliche Anzahl von Obstruktionen dagegen, dass die Produkte

$$\prod_{i\in I} a_i - b_i X \bmod f_j$$

Quadrate in $\mathbb{Z}[X]/(f_j)$ sind (als Elemente der beiden Zahlkörper betrachtet sind ihre Bewertungen zwar an allen unverzweigten Primidealen gerade, aber die Einheitengruppen und die 2-Anteile der Idealklassengruppen liefern noch Obstruktionen). Da die Anzahl der Obstruktionen normalerweise kleiner als die Dimension des Lösungsraumes ist, kann durch Multiplikation von obigen Kongruenzen erreicht werden, dass beide Produkte Quadrate sind, also:

$$\prod_{i\in I} a_i - b_i X = h_j^2 \pmod{f_j}$$

mit $h_j \in \mathbb{Z}[X]/(f_j)$. Dann ergibt sich

$$\phi_1(h_1)^2 \equiv \phi_1(h_1^2) \equiv \phi_2(h_2^2) \equiv \phi_2(h_2)^2 \pmod N.$$

Dies führt mit $x = \phi_1(h_1)$ und $y = \phi_2(h_2)$ zu einer Lösung der anfangs erwähnten Kongruenz

$$x^2 \equiv y^2 \pmod N$$

und damit einer $\geq 50\%$-Chance auf eine Zerlegung von N.

Die Laufzeit dieses Schrittes ist gegenüber der der anderen Schritte vernachlässigbar. Bei RSA-768 könnte dieser Schritt in wenigen Stunden auf mehreren Rechnern durchgeführt werden; da aber zwei Fehler in den Programmen aufgetaucht sind, hat er vier Tage gedauert.