RUHR
UNIVERSITÄT
BOCHUM

RUB

Lehrstuhl für Kryptologie und IT-Sicherheit
Prof. Dr. Alexander May
Alexander Meurer, Ilya Ozerov

Hausübungen zur Vorlesung

Kryptanalyse

WS 2011/2012

Blatt 7 / 30. November 2011 / Abgabe bis spätestens 7. Dezember 2011, 10 Uhr in dem Kasten auf NA 02

AUFGABE 1 (4 Punkte)**:**
Sei ein Algorithmus A gegeben, der bei Eingabe N einen nicht-trivialen Faktor von N in Zeit polynomiell in $\log N$ berechnet. Zeigen Sie, dass dann die komplette Primfaktorzerlegung von N in Zeit polynomiell in $\log N$ berechnet werden kann.

AUFGABE 2 (5 Punkte)**:**

(a) Seien $\mathbf{v}_1, \ldots, \mathbf{v}_j \in \mathbb{Z}_2^n$ linear unabhängig. Dann ist die Wahrscheinlichkeit, dass ein zufällig aus $\mathbb{Z}_2^n$ gezogener Vektor zu $\mathbf{v}_1, \ldots, \mathbf{v}_j$ linear unabhängig ist, durch $1 - 2^{j-n}$ gegeben.

(b) Seien $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{Z}_2^n, k \leq n$ zufällig gewählte Vektoren. Zeigen Sie, dass diese Vektoren mit Wahrscheinlichkeit

$$\prod_{i=0}^{k-1}(1 - 2^{i-n})$$

linear unabhängig sind.

Hinweis: Führen Sie einen Beweis per Induktion über k.

AUFGABE 3 (5 Punkte)**:**
Konstruieren Sie einen Algorithmus, der in Zeit $\tilde{\mathcal{O}}(B)$ eine Faktorbasis

$$F_B = \{p \in \mathbb{N} : p \leq B \text{ und } p \text{ prim}\} \cup \{-1\}$$

zur Schrnake $B \in \mathbb{N}$ konstruiert.

Hinweis: Überlegen Sie, ob der naive Siebalgorithmus aus Aufgabe 3 der Präsenzübung verbessert werden kann, indem in der zweiten Schleife weniger Elemente durchlaufen werden. Sie dürfen in der Laufzeit-Analyse die Abschätzung $\log(B) \geq \sum_{i=2}^{B} \frac{1}{i}$, $B \geq 2$ benutzen.

AUFGABE 4 (5 Punkte)**:**
Implementieren Sie den Faktorisierungs-Algorithmus mittels Faktorbasen (Skript Seite 87). Bestimmen Sie die *vollständige* Faktorisierung von `N=276509251861918637`. Verwenden Sie eine Faktorbasis F_B für die Glattheitsschranke $B = 800$. Geben Sie den Quellcode mit ab.

Hinweis: Zur Erzeugung der Faktorbasis ist der Befehl `prime_range(n)` hilfreich, welche alle Primzahlen zwischen $2, \ldots, n$ liefert. Eine naive Implementierung in sage kann zur Lösung einige Minuten in Anspruch nehmen.
Sie dürfen gerne eigentständige Verbesserungen benutzen (z.B. *Siebtechniken* zur effizienteren Bestimmungen B-glatter Elemente).