Ruhr-Universität Bochum
Lehrstuhl für Kryptologie und IT-Sicherheit
Prof. Dr. Alexander May
Elena Kirshanova, Ilya Ozerov

Hausübungen zur Vorlesung

Kryptanalyse

SS 2014

Blatt 2 / 05. Mai 2014 (Update 4c)
Abgabe: 08. Mai 2014, 14.00 Uhr, Kasten NA/02

AUFGABE 1 (5 Punkte)**:**

Sei $N = pq$, $p > q$ das Produkt zweier starker Primzahlen, d.h. $p = 2p' + 1$ und $q = 2q' + 1$ mit $p', q' > 2$ prim.

(a) Zeigen Sie, dass es in $\mathbb{Z}_N^*$ keine Elemente der Ordnung $4p'q'$ gibt.

(b) Zeigen Sie, dass es in $\mathbb{Z}_N^*$ genau $(p'-1)(q'-1)$ Elemente der Ordnung $p'q'$ gibt.

(c) Zeigen Sie, dass es in $\mathbb{Z}_N^*$ genau $3(p'-1)(q'-1)$ Elemente der Ordnung $2p'q'$ gibt.

(d) Zeigen Sie, dass für zufällig, gleichverteilt aus $\mathbb{Z}_N^*$ gewählte Elemente m gilt, dass

$$\mathsf{Ws}[\mathsf{ord}(m) \in \{p'q', 2p'q'\}] \geq 1 - \frac{2}{q'}$$

Hinweis: Sie dürfen für (b) und (c) verwenden, dass es in $\mathbb{Z}_p^*$ (analog $\mathbb{Z}_q^*$) genau $p'-1$ Elemente der Ordnung p' und genau $p'-1$ Elemente der Ordnung $2p'$ gibt.

AUFGABE 2 (5 Punkte)**:**

Seien p, q_1, q_2 prim, $r, B \in \mathbb{N}$, $q_1 > q_2 > p$ und $r \leq B < p$. Wie in der Präsenzübung ist das "approximative ggT Problem" wie folgt definiert: Gegeben sind $N_1 = p \cdot q_1$ und $N_2 = p \cdot q_2 + r$, gesucht ist p.

Implementieren Sie die Bruteforce-Angriffe aus der Präsenzübung in sage. In der ersten Version soll *in jedem Schritt* ein ggT berechnet werden. In der zweiten Version soll *insgesamt lediglich eine* ggT-Berechnung erfolgen. Verwenden Sie Ihre Implementierung, um die Instanz in der Datei `ggt.txt` anzugreifen. Wählen Sie dabei selbst einen geeigneten Wert für B. Was ist das gesuchte p? Vergleichen Sie die Laufzeiten Ihrer Implementierungen (Befehl time). Geben Sie den Quelltext Ihres Programms mit ab.

Bitte wenden!

AUFGABE 3 (7 Punkte)**:**

Seien p, q_1, q_2 prim, $r, B \in \mathbb{N}$, $q_1 > q_2 > p$ und $r \leq B < p$. Sei zudem B eine Quadratzahl, d.h. $\sqrt{B} \in \mathbb{N}$. Wieder ist das "approximative ggT Problem" wie folgt definiert: Gegeben sind $N_1 = p \cdot q_1$ und $N_2 = p \cdot q_2 + r$, gesucht ist p.

(a) Zeigen Sie, dass $\prod_{i=1}^{B}(N_2 - i) = \prod_{i=1}^{\sqrt{B}} \prod_{j=0}^{\sqrt{B}-1}(N_2 - i - j\sqrt{B}) \bmod N_1$.

(b) Sei $p = \text{ggT}\left(N_1, \prod_{i=1}^{B}(N_2 - i) \bmod N_1\right)$. Beschreiben Sie einen Algorithmus, der das Problem in Zeit (und Platz) $\widetilde{\mathcal{O}}(\sqrt{B})$ löst. Verwenden Sie Satz 25 aus der Vorlesung, der in der nachfolgenden Aufgabe bewiesen wird. Zeigen Sie Korrektheit und Laufzeit.

(c) Modifizieren Sie Ihren Algorithmus für den Fall $p \neq \text{ggT}\left(N_1, \prod_{i=1}^{B}(N_2 - i) \bmod N_1\right)$.

AUFGABE 4 (8 Punkte)**:**

Sei $f(x)$ ein Polynom vom Grad $n-1$, wobei n eine Zweierpotenz ist. Im Folgenden bezeichne mod das Modulo auf dem Ring der Polynome in x. Wir setzen voraus, dass man $g(x) \bmod h(x)$ für beliebige Polynome des Grads höchtens $n-1$ in Zeit $\mathcal{O}(n \log n)$ berechnen kann. Gegeben seien $f(x)$ und n Stellen $x_0, \ldots, x_{n-1}$. Zu berechnen ist $f(x_i)$ für alle $0 \leq i \leq n-1$. Für $0 \leq i \leq j \leq n-1$ definieren wir $p_{ij}(x) = \prod_{m=i}^{j}(x - x_m)$ und $q_{ij}(x) = f(x) \bmod p_{ij}(x)$.

(a) Zeigen Sie, dass $f(x) \bmod (x - z) = f(z)$ für alle z.

(b) Zeigen Sie, dass $q_{kk}(x) = f(x_k)$ und $q_{0,n-1}(x) = f(x)$.

(c) Zeigen Sie, dass $q_{ik}(x) = q_{ij}(x) \bmod p_{ik}(x)$ und $q_{kj}(x) = q_{ij}(x) \bmod p_{kj}(x)$ für alle $i \leq k \leq j$ gilt.

(d) Konstruieren Sie einen Algorithmus, der in Zeit $\mathcal{O}(n \log^2 n)$ die Werte $f(x_0), \ldots, f(x_{n-1})$ berechnet. Beweisen Sie Korrektheit und Laufzeit.

Hinweis: Realisieren Sie eine Didive-and-Conquer-Strategie. Benutzen Sie (a) bis (c).

AUFGABE 5 (5 Punkte)**:**

Sei (p, α, β) ein öffentlicher Schlüssel des ElGamal Verschlüsselungsverfahrens. In einer Open Source Implementierung wird beim Verschlüsseln aufgrund eines Programmierfehlers ein zu kleiner Zufallswert r verwendet. Aus einer Analyse des Codes entnehmen Sie, dass r maximal 32 Bit groß werden kann. Sie fangen eine verschlüsselte Nachricht (γ, δ) ab. Alle öffentlichen Daten liegen in der Datei `elgamal.txt`.

(a) Beschreiben Sie einen Meet-in-the-Middle Angriff auf dieses Problem. Zeigen Sie Laufzeit und Korrektheit.

(b) Implementieren Sie den Algorithmus in sage und bestimmen Sie die Nachricht m.

Hinweis: Resultate einer Reduzierung modulo p sollten vor der binären Suche mit `int` in natürliche Zahlen konvertiert werden. (Nur) für die binäre Suche dürfen Sie Python-Implementierungen aus dem Internet verwenden.