htw saar

Studiengang Kommunikationsinformatik (Master)
Studiengang Praktische Informatik (Master)
Prof. Dr.–Ing. Damian Weber

Sicherheit und Kryptographie – Übung 4

Wählen Sie aus den folgenden Aufgaben eine Aufgabe aus, je nachdem, ob Sie die bestehende Python-Implementierung von ECM weiterentwickeln möchten, oder in einer von Ihnen gewählten Programmiersprache die Arithmetik der elliptischen Kurven implementieren möchten.

Aufgabe 1 (Elliptische Kurven Implementierung)

(Falls Sie nicht Python als Programmiersprache wählen.)

Sie können die Python-Implementierung für ECM aus Aufgabe 2 als Vorlage benutzen.

a) Implementieren Sie für die elliptischen Kurven mod p eine Arithmetik, die für zwei Punkte P, Q den Wert $P + Q$ berechnet.

b) Die Implementierung soll die Gültigkeit der elliptischen Kurve mittels

$$4a^3 + 27b^2 \not\equiv 0 \ mod \ p$$

prüfen.

c) Die Implementierung soll durch eine `for`–Schleife für x von 0 bis $p - 1$ die Anzahl der Punkte $|E|$ berechnen. Hierbei wird die rechte Seite der Kurvengleichung ausgewertet und festgestellt, ob diese ein Quadrat ergibt.

Wie groß können Sie p wählen, wenn Ihre Implementierung den Wert $|E|$ innerhalb von 10 Minuten berechnen können soll. Sie könnten diesen Teil verteilt in mehreren Prozessen oder Threads berechnen, indem der Bereich für x auf die Prozessoren aufgeteilt wird.

d) Implementieren Sie durch fortgesetztes Verdoppeln des Punktes eine schnelle Multiplikation eines Punktes P mit k, d.h. berechnen Sie

$$k \cdot P$$

für gegebene $k \in \mathbf{N}$, $P \in E$.

Ist Ihre Implementierung effizient genug, um dies für k bzw. $p \approx 160$ Bit zu tun?

Aufgabe 2 (Elliptische Kurven Faktorisierungsmethode (verteilt))

(Falls Sie Python als Programmiersprache wählen.)

Nehmen Sie die unter

`http://www-crypto.htw-saarland.de/weber/teaching/15_ws_sk/ecm-h-opt.py`

vorhandene und in der Vorlesung erläuterte Implementierung der ECM-Methode als Grundlage, um eine verteilte Version der Faktorisierungsmethode zu erzeugen.

Hierzu kontaktiert jeder Client `ecm-h-opt.py` einen Server, um die zu zerlegende Zahl und die Parameter B_1, B_2, *n_curves* zu erfahren. Wenn ein Client einen Faktor t gefunden hat, kontaktiert er den Server. Der Server berechnet den noch zu zerlegenden Rest $n' = n/t$ und sorgt dafür

a) falls n' eine Primzahl ist (siehe Übung 2), dass sich alle Clients beenden

b) dass alle Clients mit n' an Stelle von n weiterrechnen, mit den richtigen Parametern für n'