Komplexitätstheorie

Sommersemester 2020

Mario Holldack, M. Sc.
Prof. Dr. Georg Schnitger
Hannes Seiwert, M. Sc.

GOETHE UNIVERSITÄT
FRANKFURT AM MAIN

Institut für Informatik
AG Theoretische Informatik

Übungsblatt 6

Ausgabe: 01.06.2020
Abgabe: Di, 08.06.2020

Die Abgabe erfolgt per E-Mail an Mario Holldack (`holldack@em.uni-frankfurt.de`).

Aufgabe 6.1 *Blockchiffrierung und AES* $((1+1+2+2)+(2+2) = 10$ Punkte)

Der *Rijndael-Algorithmus* ist der im AES (Advanced Encryption Standard) spezifizierte Verschlüsselungsalgorithmus. Es handelt sich um eine Blockchiffrierung, d. h. der zu verschlüsselnde Klartext wird in Blöcke von jeweils 128 Bits aufgeteilt. Anschließend werden die Blöcke einzeln verschlüsselt. In dieser Aufgabe verschaffen wir uns einen ersten Eindruck über dieses in der Praxis weit verbreitete Verschlüsselungsverfahren und zeigen eine interessante Verbindung zwischen Algebra und superschnellen Berechnungen auf.

a) Lesen Sie den Abschnitt 4.5 in S. Goldwasser und M. Bellare: Lecture Notes on Cryptography.
   i) Woher stammt der Name des Rijndael-Algorithmus?
   ii) Handelt es sich um ein Private-Key- oder um ein Public-Key-Verschlüsselungsverfahren?
   iii) Beschreiben Sie kurz die Schritte des Rijndael-Algorithmus. Sie müssen nicht auf Implementierungsdetails eingehen. Konzentrieren Sie sich auf die wesentlichen Ideen.
   iv) Wie kann der Klartext wiederhergestellt werden? Auch hier genügt eine kurze Beschreibung der wesentlichen Ideen.

b) Wir betrachten die Implementierung des Rijndael-Algorithmus nun aus mathematischer Perspektive. Jedes Byte $a = a_7a_6 \dots a_0 \in \{0,1\}^8$ assoziieren wir mit dem univariaten Polynom

$$p_a(x) := \sum_{i=0}^{7} a_i x^i \mod (x^8 + x^4 + x^3 + x + 1)$$

über dem endlichen Körper $\mathrm{GF}(2^8)$ mit 2^8 Elementen. (Die Elemente von $\mathrm{GF}(2^8)$ sind also Polynome vom Grad höchstens 7 mit $\{0,1\}$-wertigen Koeffizienten.)

   i) Multiplizieren Sie das Byte $a_7a_6 \dots a_0$ mit dem Byte 0000 0010, d. h. berechnen Sie

$$\left(\sum_{i=0}^{7} a_i x^i\right) \cdot x \mod (x^8 + x^4 + x^3 + x + 1).$$

   Geben Sie das Ergebnis als Byte an. Erläutern Sie kurz, wie die Multiplikation mit 0000 0010 durch Bitshifts und bitweises XOR implementiert werden kann.
   Hinweis: Unterscheiden Sie die Fälle $a_7 = 0$ und $a_7 = 1$ und zeigen Sie

$$x^8 \equiv x^4 + x^3 + x + 1 \mod (x^8 + x^4 + x^3 + x + 1).$$

   ii) Geben Sie das Produkt der Bytes 1001 1010 und 0010 1001 an.
   Hinweis: Nutzen Sie Teil i) und $0010\,1001 = 0010\,0000 \oplus 0000\,1000 \oplus 0000\,0001$.

Fazit: Die Rechenoperationen des Rijndael-Algorithmus können superschnell durch Bitshifts und bitweises XOR bzw. durch Lookup-Tabellen implementiert werden.

Aufgabe 6.2 *Der Generator der polynomiellen Kongruenzen* (6 Punkte)

Für eine Primzahl p und Koeffizienten $a = (a_0, \ldots, a_d) \in \mathbb{Z}_p^{d+1}$ sei $G_{a,p}(x) := \sum_{i=0}^{d} a_i x^i \bmod p$ der Generator der polynomiellen Kongruenzen vom Grad d, der die Folge

$$x_0 := s$$
$$x_{k+1} := \sum_{i=0}^{d} a_i x_k^i \bmod p$$

mit $s \not\equiv 0 \bmod p$ und $a \neq \mathbf{0} \in \mathbb{Z}_p^{d+1}$ erzeugt.

Sei $k = \Theta(d)$ mit $k \geq d+1$ beliebig. Nach k-facher Anwendung von $G_{a,p}$ möchten wir x_{k+1} ohne direkten Zugriff auf den Generator *vorhersagen*. Einen Algorithmus, der x_{k+1} vorhersagt, bezeichnen wir als *effizient*, wenn seine Laufzeit polynomiell in d und $\log p$ ist.

Angenommen, wir kennen die Primzahl p und den Grad d, aber nicht die Koeffizienten $a_0, \ldots, a_d$. Geben Sie einen effizienten deterministischen Algorithmus an, der x_{k+1} mithilfe der Folgenglieder $x_k, \ldots, x_0$ vorhersagt.

Hinweis: Bestimmen Sie die Koeffizienten $a_0, \ldots, a_d$.

Aufgabe 6.3 *El-Gamal-Verfahren und Diffie-Hellman-Vermutung* (4 + 4 = 8 Punkte)

Wir betrachten in dieser Aufgabe Sicherheitsgarantien für das El-Gamal-Verfahren (siehe Folie 51).

a) Wir nehmen an, dass Bob zur Verschlüsselung zweier Nachrichten x und y zweimal denselben Exponenten $b \in \{1, \ldots, q-1\}$ verwendet hat, d. h. es gelte

$$\mathrm{Enc}(x) = (g^b, h^b \cdot x) \text{ und } \mathrm{Enc}(y) = (g^b, h^b \cdot y).$$

Zeigen Sie: Falls Dave die verschlüsselten Nachrichten $\mathrm{Enc}(x)$ und $\mathrm{Enc}(y)$ sowie die Klartextnachricht y kennt, dann kann er die Klartextnachricht x bestimmen, ohne den privaten Schlüssel a zu besitzen.

Hinweis: Sei $z \in R$. Dann kann das multiplikative Inverse $z^{-1} \in R$ von z effizient mit dem erweiterten euklidischen Algorithmus bestimmt werden.

Fazit: Das El-Gamal-Verfahren ist anfällig gegen Known-Plaintext-Angriffe[1].

b) Seien das erzeugende Element g und der Modulus p gegeben. Die Diffie-Hellman-Vermutung besagt, dass effiziente, nicht-uniforme Algorithmen die Tripel (g^x, g^y, g^{xy}) und (g^x, g^y, g^z), jeweils modulo p, für zufällige $x, y, z \in \mathbb{Z}_q$ nur mit vernachlässigbarem Vorteil unterscheiden können.

Zeigen Sie:

Wenn die Diffie-Hellman-Vermutung gilt, dann ist das El-Gamal-Verfahren IND-CPA-sicher.

[1] https://en.wikipedia.org/wiki/Known-plaintext_attack