Open Class – Sieben Wunder der Informatik

Prof. Dr. Juraj Hromkovič

Departement Informatik

Lösungsvorschläge für die Übungsaufgaben – Blatt 6

Zürich, 14. Dezember 2005

Lösung zu Aufgabe 17

Wir schreiben den Klartext als eine Folge von Nullen und Einsen und verwenden nun das in der Veranstaltung beschriebene XOR-Verfahren auf Blöcken z.B. der Länge $2^8 = 256$. Das heisst, dass Schlüssel ebenfalls 0-1-Folgen sind und die Länge 256 haben. Es gibt genau 2^{256} derartige Schlüssel.
Ferner gilt $2^{256} = 10^{256 \cdot \frac{\ln 2}{\ln 10}}$ und $\frac{\ln 2}{\ln 10} \approx 0.30103$; mithin $2^{256} \approx 10^{256 \cdot 0.30103} = 10^{77.06368}$.
Also gibt es mehr als 10^{77} mögliche Schlüssel. Dennoch ist dieses Kryptosystem unsicher im folgenden Sinne: Wenn man für einen Block von Krypto-Text den Klartext erschliessen kann, so kann man dies für jeden Block, denn aus Klar- und Krypto-Text kann man den Schlüssel ausrechnen.
Gute Kryptosysteme erfüllen die Anforderung, dass selbst wenn ein potenzieller Angreifer einen Klartext vorgeben darf ("chosen plaintext attack"), der vom rechtmässigen Schlüsselinhaber verschlüsselt wird, die Kenntnis des resultierenden Krypto-Texts ihm keinen Vorteil verschafft.
Folglich ist ein 256 Bit breiter Schlüssel alleine noch keine Garantie für Sicherheit.

Lösung zu Aufgabe 18

Es sei $\oplus$ die aus der Veranstaltung bekannte XOR-Funktion. Wenn nun der Krypto-Text mit einem Schlüssel K verschlüsselt wurde, dessen Bits die Folge $(k_1, \ldots, k_n)$ bilden, so wählt der Absender eine Zufallsfolge $(a_1, \ldots, a_n)$ und bildet die Folge $(b_1, \ldots, b_n)$ durch

$$b_i := k_i \oplus a_i \text{ für alle } i \in \{1, \ldots, n\} \quad .$$

Die beiden Folgen $(a_i)_{i=1}^n$ und $(b_i)_{i=1}^n$ werden nun an die beiden Empfänger versendet. Beide Empfänger müssten für jedes Bit des Schlüssels raten, solange sie nicht die Information des anderen Empfängers besitzen. Besitzen sie sie jedoch, so rekonstruieren sie

$$k_i := a_i \oplus b_i \text{ für alle } i \in \{1, \ldots, n\} \quad .$$

Lösung zu Bonus-Aufgabe 6

Wir nutzen erneut solche Einweg-Funktionen, für die gilt, dass man ihre Umkehrung effizient berechnen kann, wenn man ein Geheimnis von ihnen kennt. Wenn nämlich etwa ein

Teilnehmer A die Einweg-Funktion f_A so bestimmt hat, dass nur er effizient f_A^{-1} berechnen kann, so kann er, wie auch schon im Verschlüsselungsfall, f_A in ein öffentliches Verzeichnis (wie ein Telefonbuch) eintragen lassen.
Um nun sein Einverständnis mit dem Text T zu bekunden, verfährt A wie folgt:

- Wir betrachten T (was immer möglich ist) als ein potenzielles Argument und als einen potenziellen Wert von f_A, also in der Regel als eine Zahl.
- Von A wird nun der Wert $S := f_A^{-1}(T)$ berechnet und veröffentlicht. Die Veröffentlichung kann dabei etwa durch Versand an alle Teilnehmer geschehen oder, was pragmatischer ist, durch Anhängen der in S enthaltenen Information an T, wann immer T versandt wird.
- Um sich zu überzeugen, dass die Einverständniserklärung S auch wirklich zu T passt, berechnen daran interessierte Teilnehmer $f_A(S)$ und vergleichen diesen Wert mit T.

Zusätzliche Anmerkungen
Zwei Probleme treten auf. Einerseits können potenzielle Texte T relativ lang sein, wohingegen die Funktionen f_A nur auf relativ kleinen Zahlen definiert sind. Andererseits könnte man auf die Idee kommen, zu irgendeiner (erfundenen) Einverständniserklärung S' einfach einen Text T' durch $T' := f_A(S')$ zu berechnen, um zu behaupten, A sei mit T' einverstanden. In der Regel würde so ein Text T' zwar keine erkennbare Struktur aufweisen, aber damit geben wir uns nicht zufrieden.
In der Praxis verwendet man daher zunächst sogenannte (kryptographische) *Hash-Funktionen,* die mit Einweg-Funktionen gemeinsam haben, dass ihre Umkehrung nur sehr aufwändig zu berechnen ist. Darüber hinaus ist die Umkehrung im Falle von Hash-Funktionen nicht eindeutig, wodurch die Grösse ihrer Werte in der Regel beschränkt bleibt. Sei nun h eine solche Hash-Funktion[1]. Anstatt nun sein Einverständnis mit T zu bekunden, bekundet man einfach sein Einverständnis mit $h(T)$. Dadurch werden die oben beschriebenen Probleme gleichzeitig vermieden.
Wir illustrieren die Vorgehensweise in Analogie zum in der Veranstaltung vorgestellten Rabin-Verfahren zur Verschlüsselung. Zu Funktionen f_A erklären wir das Quadrieren *modulo einer grossen Zahl* n, die sich aus zwei Primzahlen zusammensetzt. Man geht davon aus, dass die Umkehrung, das Wurzelziehen (modulo n) ein schweres Problem ist, solange man die Faktorisierung von n nicht kennt. Deswegen verwenden wir die Faktorisierung als Geheimnis. Einverständniserklärungen werden also wie folgt geschehen:

- Der Teilnehmer A berechnet einen Wert[2] $S \in \{0, \ldots, n-1\}$ so, dass $S^2 \bmod n = h(T)$. Wir sagen, er *zieht die (Quadrat-) Wurzel modulo* n aus $h(T)$.
- Um ein behauptetes Einverständnis von A zu überprüfen, quadriert man einfach den Wert S modulo n und vergleicht mit $h(T)$.

[1] Häufig verwendet man die Funktionen SHA-1 (Secure Hash Algorithm) oder MD-5 (Message Digest).
[2] Wir haben hier unterschlagen, dass nicht alle Zahlen eine Wurzel modulo n besitzen. In der Praxis wählt man h so, dass die Werte von h allesamt Quadratzahlen modulo n sind.