Vorlesung Nr. 7

Kryptologie II - Datum: 22.10.2018

- Sicherheitsmodelle (Teil 2)
- Primzahleigenschaften (Teil 2)
- Zufall (Teil 1)

Buch der Woche

Titel: Kryptographie in C und C++ (2. Auflage 2001)

Autor(en): Michael Welschenbach

Verlag: Springer

Umfang: ca. 400 Seiten

Hinweise zum Inhalt:

- Dieses Buch ist eine anspruchsvolle Einführung in mathematische Kryptographie, auch wenn der Buchtitel auf ein Programmierlehrbuch schließen lässt.
- Modulare Arithmetik und Zahlentheorie werden ausführlich erläutert
- Zahlreiche Hinweise und Programmierbeispiele zur möglichst effizienten Implementierung werden geliefert
- Da sich weder die Mathematik noch die Programmiersprache nennenswert verändert haben, ist das Buch trotz seines Erscheinungsdatums weiterhin aktuell und sehr nützlich für alle, die Krypto-Algorithmen implementieren wollen.

Sicherheitsmodelle (Fortsetzung)

Security Goals: Non-Malleability NM

Sicherheitsziele können in der Kryptographie sehr mathematisch und formal betrachtet werden. Wir verzichten hier auf Formeln und geben eine anschauliche Erklärung für eine gängige Forderung, die sogenannte

Non-Malleability NM: angenommen wir haben einen Chiffretext $C_1 = E(K, P_1)$. Dann soll es nicht möglich sein, einen anderen Chiffretext $C_2 = E(K, P_2)$ zu konstruieren, so dass die zugehörigen Klartexte P_1 und P_2 in einer "sinnvollen", mathematisch beschreibbaren Relation zueinander stehen (beispielsweise so, dass $P_2 = P_1 \oplus S$ gilt für einen bestimmten Bitstring S, etwa S = "000…01").

Beispiel: betrachten wir das One-Time Pad und nehmen an, es sei $C_2 = C_1 \oplus S$.

Dann gilt $P_2 = C_2 \oplus K = C_1 \oplus S \oplus K = C_1 \oplus K \oplus S = P_1 \oplus S$ für jedes gegebene S.

Unser One-Time Pad erfüllt also nicht Non-Malleability NM.

Anmerkung: streng genommen ist das kein ideales Beispiel, da wir ja vom One-Time Pad gerade gefordert hatten, dass kein Schlüssel K mehrfach verwendet werden darf...

Security Goals: Indistinguishability IND

Ein in der kryptographischen Forschung ebenfalls häufig formulierte Anforderung an ein Verschlüsselungsverfahren ist sogenannte ***Indistinguishability IND*** (deutsch: Ununterscheidbarkeit). Diese besagt (umgangssprachlich):

Chiffretexte sollten nicht unterscheidbar sein von Zufallsstrings.

Übersetzt man dies in eine Spielregel, so gilt:

Angenommen ein Angreifer wählt zwei Klartexte aus und erhält sodann den zugehörigen Chiffretext zu einem dieser beiden Klartexte. Dann soll er nicht in der Lage sein zu erkennen, welcher der beiden Klartexte mit dem erhaltenen Chiffretext korrespondiert.

Dies soll sogar gelten unter der Annahme, dass CPA-Angriffe zulässig sind, das heisst der Angreifer darf beliebige Klartexte verschlüsseln (auch die beiden zuvor oben gewählten) und erhält das jeweilige Resultat.

Anmerkung: wie man sieht, ist dies eine sehr weitreichende Forderung, der ein "normales" Verschlüsselungsverfahren nicht gerecht würde, denn wenn man seine beiden gewählten Klartexte verschlüsseln darf, so sieht man sofort, welches der beiden Ergebnisse mit dem vorgegebenen Chiffretext übereinstimmt. Gelegentlich wird obige Anforderung sogar auf CCA statt CPA ausgeweitet.

Security Notions

Betrachtet man nur entweder Security goals oder Attack models, so erhält man noch immer keine umfängliche Auffassung dessen, was eine Verschlüsselung zu leisten imstande ist bzw. sein soll. Erst in der Kombination der beiden können wir konkrete Aussagen formulieren, ob ein Verfahren ein bestimmtes Sicherheitsziel erfüllt, wenn man eine vorgegebene Menge von Angriffsarten zugrunde legt.

Diese Kombination aus Security goals und Attack model bezeichnet man als

Security Notion (notion = Vorstellung, Auffassung, Begriff).

Wir haben Security goals und Attack models auch deshalb mit Abkürzungen versehen, um diese handlicher als Kombination darzustellen, wie folgt:

- NM-COA, NM-KPA, NM-CPA, NM-CCA
- IND-COA, IND-KPA, IND-CPA, IND-CCA

Beispiel: ein Verschlüsselungsverfahren erfüllt beispielsweise IND-CPA, wenn es Indistinguishability leistet für den Fall von Chosen plaintext attacks.

IND-CPA: Semantic Security

Ein in der Kryptographie häufig geforderte Security notion ist die Kombination IND-CPA. Für diese wurde deshalb ein gesonderter Begriff vorgesehen:

IND-CPA nennt man ***Semantic Security***.

Gibt es überhaupt Verfahren, die Semantic security bieten?

Zunächst sehen wir schnell, dass ein solches Verfahren einen Plaintext bei wiederholter Anwendung nicht erneut auf denselben Ciphertext abbilden darf, da sonst keine Indistinguishability gegeben sein kann (siehe Anmerkung zu IND).

Damit dies gelingen kann, müssen wir einen zusätzlichen Zufallsstring in unseren Verschlüsselungsprozess einbauen. Das bedeutet, der Chiffretext muss länger sein als der zugehörige Klartext.

Die Entschlüsselung muss dabei natürlich trotzdem stets wieder den “einen”, ursprünglichen Klartext liefern, unabhängig von den in die Verschlüsselung eingebrachten Zufallsbits.

Nachfolgend schauen wir uns ein Beispiel eines solchen Verfahrens an.

Ein Semantisch Sicheres Verfahren (1)

Wir können auf einfache Weise ein Verschlüsselungsverfahren konstruieren, das IND-CPA bzw. Sematic Security erfüllt: hierzu benötigen wir lediglich einen beliebigen deterministischen Zufallsbitgenerator DRBG. Als weitere Zutat benötigen wir einen Zufallsstring R, der bei jedem Verschlüsselungsvorgang neu gewählt wird.

Die Verschlüsselung E erhält dann als Input den Plaintext P, den Zufallsstring R, und den Key K, wie folgt:

$E(P, K, R) = (DRBG(K \| R) \oplus P, R)$

Der Zufallsstring R wird unverschlüsselt beigefügt, da ansonsten keine Entschlüsselung mehr möglich wäre.

Ähnlich zum One-Time Pad wird auch hier der Plaintext mittels XOR verknüpft. Der Schlüssel K ist diesmal jedoch kürzer und dient nur als Input bzw. zur Initialisierung des Zufallsgenerators.

Ein Semantisch Sicheres Verfahren (2)

Es bleibt zu zeigen, dass $E(P, K, R) = (DRBG(K \| R) \oplus P, R)$ IND-CPA erfüllt.

Hierfür darf $DRBG(K \| R) \oplus P$ nicht von einem Zufallsstring unterscheidbar sein.

Nach Voraussetzung ist $DRBG(K \| R)$ nicht von einem Zufallsstring unterscheidbar.

Sei $S = DRBG(K \| R) = (s_1, \ldots, s_n)$ und sei $P = (p_1, \ldots, p_n)$.

Nach Annahme ist die Wahrscheinlichkeit, dass ein $s_i = 0$ ist jeweils gleich $\frac{1}{2}$.

Damit beträgt auch die Wahrscheinlichkeit, dass $s_i \oplus p_i = 0$ jeweils $\frac{1}{2}$.

Folglich ist $DRBG(K \| R) \oplus P$ nicht von einem Zufallsstring unterscheidbar.

Nun erhält ein Angreifer zwei Plaintexte P_1 und P_2, und für ein $i \in \{1,2\}$ den zugehörigen Ciphertext $DRBG(K \| R_i) \oplus P_i, R$. Die Aufgabe des Angreifers besteht darin zu bestimmen, ob $i = 1$ oder $i = 2$.

Wäre der Angreifer hierzu in der Lage, so müsste er den Ciphertext von einem Zufallsstring unterscheiden können, im Widerspruch zu unserer Annahme. Somit ist das Verfahren semantisch sicher.

Security Notions: Verhältnis zueinander

Es stellt sich sofort die interessante Frage, welche Abhängigkeiten zwischen den einzelnen Security notions bestehen. Folgt eines aus dem anderen? Sind manche äquivalent zueinander?

Da CCA stärker ist als CPA, gilt trivialerweise

IND-CCA $\Rightarrow$ IND-CPA und **NM-CCA $\Rightarrow$ NM-CPA**

Doch wie verhält es sich beispielsweise, wenn wir unterschiedliche Security goals zueinander in Relation setzen? Wie steht zum Beispiel IND-CPA zu NM-CPA?

Betrachten wir hierzu unser oben dargestelltes semantisch sicheres Verfahren: falls dieses nicht auch NM-CPA erfüllt, so haben wir ein Gegenbeispiel, aus dem folgt, dass **IND-CPA $\not\Rightarrow$ NM-CPA**.

Sei $C = DRBG(K \| R) \oplus P$, dann erhalten wir $C \oplus 1 = DRBG(K \| R) \oplus P \oplus 1$.

Die Verschlüsselung von $P \oplus 1$ liefert also $C \oplus 1$, im Widerspruch zu NM-CPA.

Umgekehrt gilt jedoch (ohne Beweis, denn dieser ist sehr technisch):

NM-CPA $\Rightarrow$ IND-CPA

Primzahleigenschaften (Fortsetzung)

Vorbemerkung: Zahlentheorie und Beweise

Bitte durchhalten, es dauert nicht mehr lang...

Wir werden in Kürze übergehen zu Symmetrischen Verfahren, Blockchiffren, Betriebsmodi, und vielem mehr. Von modularer Arithmetik haben wir also bald eine Pause. Im späteren Verlauf der Vorlesung wird uns modulare Arithmetik wieder begegnen, aber die wichtigsten Grundlagen haben wir dann bereits zur Hand.

Primzahleigenschaften (2) (Wiederholung aus letzter VL)

Vorbemerkung: ein ***Lemma*** ist ein Sachverhalt bzw. ein Hilfssatz, dessen Aussage für sich betrachtet keine besondere Bedeutung hat, der jedoch im Beweis eines später zu beweisenden Satzes verwendet bzw. zitiert werden kann.

Lemma 1: sei p Primzahl und $a \in \mathbb{N}, 0 < a < p$. Dann gilt

$$a^2 \bmod p = 1 \iff a \bmod p = \pm 1$$

Beweis:

"$\Leftarrow$": sei $a \bmod p = \pm 1$, dann folgt $1 = (\pm 1)^2 = (a \bmod p)^2 = a^2 \bmod p$.

"$\Rightarrow$": sei $a^2 \bmod p = 1$, dann folgt $1 = a^2 \bmod p = (a \bmod p)^2 \bmod p$.

Aus $1 = |1| = |a \bmod p| \cdot |a \bmod p|$ folgt $a \bmod p = \pm 1$.

Primzahleigenschaften (3)

Lemma 2: sei p > 2 eine Primzahl und a $\in \mathbb{N}$, $1 < a < p$. Sei ferner $p-1 = 2^k q$ für geeignetes $k \geq 1$, so dass q ungerade. Betrachten wir nun die Folge

a^q, a^{2q}, a^{4q}, …, $a^{2^{k-1}q}$, $a^{2^k q}$, so gilt:

Entweder sind alle Folgenglieder kongruent zu 1 mod p, oder eines der Folgenglieder ist kongruent zu -1 mod p.

Beweis: nach dem Satz von Fermat wissen wir, dass $a^{2^k q} = a^{n-1} = 1 \pmod p$.

Das letzte Glied in obiger Folge ist also kongruent 1 mod p.

Beachte: jedes Folgenglied ist das Quadrat des vorherigen Elements.

Nach Lemma 1 folgt für das vorletzte Folgenglied: dieses ist $\pm 1 \pmod p$.

Ist es gleich -1 (mod p), so gilt die Behauptung. Ist es indes gleich 1 (mod p), so betrachten wir mit demselben Argument wieder ein Folgenglied davor und dieses ist wieder wahlweise $\pm 1 \pmod p$.

Auf diese Weise erhalten wir entweder irgendwann eine -1 (mod p), oder alle Folgenglieder bis einschließlich dem ersten sind kongruent 1 (mod p).

Miller-Rabin Primzahltest (1)

Angenommen, wir möchten eine Zahl n auf Primzahleigenschaft testen. Hierzu können wir die soeben bewiesenen Eigenschaften nutzen. Erfüllt eine natürliche Zahl n diese Eigenschaften nicht, so kann sie keine Primzahl sein.

Ein erster trivialer Test: n muss ungerade sein.

Nun stellen wir n-1 wieder dar in der Form $n-1 = 2^k q$, mit q ungerade, $k \geq 1$.

Wir wählen ein beliebiges $a \in \mathbb{N}$, $1 < a < n-1$, und betrachten erneut die Folge a^q, a^{2q}, a^{4q}, ..., $a^{2^{k-1}q}$, $a^{2^k q}$.

Ist n tatsächlich Primzahl, so wissen wir nach Lemma 2:

- entweder ist bereits das erste Element der Folge $a^q = 1 \pmod n$
- oder es ist eines der Elemente $a^{2^{j-1}q} = -1 \pmod n$ für $1 \leq j \leq k$.

Sind wir beim vorletzten Element $a^{2^{k-1}q}$ angelangt und dieses ist noch immer nicht $\pm 1 \pmod n$, so kann n keine Primzahl sein.

Miller-Rabin Primzahltest (2)

Besteht eine Zahl diesen Miller-Rabin Test NICHT, so kann sie keine Primzahl sein.

Wissen wir umgekehrt, dass eine Zahl Primzahl ist, wenn sie den Test besteht?

Nein! Das Bestehen des Tests ist zwar notwendig, aber nicht hinreichend für die Primzahleigenschaft.

Beispiel (aus Stallings, Cryptography and Network Security): betrachte

$n = 2047 = 23 \cdot 89$. Es ist $n-1 = 2046 = 2^1 \cdot q$ mit $q = 1023$.

In der Tat erhalten wir für $a = 2$: $a^q \pmod n = 2^{1023} \pmod{2047} = 1 \pmod{2047}$.

Für $a = 2$ erfüllt n also unseren Primzahltest, ist jedoch keine Primzahl.

Wie oft passiert es, dass eine Nicht-Primzahl den Miller-Rabin Test besteht?

Es lässt sich zeigen, dass dies für ungerades n und beliebiges $a \in \mathbb{N}, 1 < a < n-1$, höchstens mit Wahrscheinlichkeit $\frac{1}{4}$ geschieht. Führt man den Test durch für m verschiedene Werte $a_1, a_2, \ldots, a_m \in \mathbb{N}, 1 < a_i < n-1$, sinkt die Wahrscheinlichkeit, dass der Test dennoch für alle a_i bestanden wird, auf $(\frac{1}{4})^m = 2^{-2m}$. Wir müssen den Test also nur "oft genug" durchführen.

Beispiel: besteht eine natürliche Zahl n den Rabin-Miller Test für zehn verschiedene Werte von a, so ist sie Primzahl mit einer Wahrscheinlichkeit $\geq 2^{-20}$, also etwa eins zu einer Million.

Primzahlhäufigkeit

Angenommen, wir möchten mithilfe eines (Pseudo-)Zufallsgenerators sehr große natürliche Zahlen erzeugen und diese sodann auf ihre Tauglichkeit als Parameter zur RSA-Verschlüsselung überprüfen. Für RSA benötigen wir zwei Primzahlen p_1 und p_2 mit einer Länge von jeweils $\geq$ 2000 Bit.

Nehmen wir an, wir haben ein zufälliges (ungerades) n erzeugt. Wie hoch ist die Wahrscheinlichkeit, dass n Primzahl ist. Anders gefragt: wieviele n müssen wir im Durchschnitt ausprobieren, bis wir eine Primzahl gefunden haben?

Zum Glück liefert die Zahlentheorie eine für uns brauchbare Aussage:

Satz: In der Umgebung der Zahl $n \in \mathbb{N}$ ist etwa jede log(n)-te Zahl eine Primzahl.

Anmerkung: gemeint ist hier der natürliche Logarithmus, nicht jener zur Basis 2. Für eine grobe Abschätzung können wir diesen Unterschied vernachlässigen...

Wir müssen also nur wenige tausend (ungerade) Zahlen n, n+2, n+4, ..., durchprobieren, bis wir auf eine Primzahl stoßen.

Der Chinesische Restsatz (1)

Der Chinesische Restsatz ist ein nützliches Hilfsmittel, wenn wir mit sehr großen Zahlen modulare Arithmetik betreiben.

Haben wir eine aus zwei oder mehreren Primfaktoren p_1, p_2, ..., p_k zusammengesetzte Zahl n, so erlaubt uns der Chinesische Restsatz, Berechnungen nicht mit dem Modul n, sondern mit den kleineren Moduln p_i durchzuführen und die Teilergebnisse erst am Ende wieder zusammenzusetzen.

Dies ermöglicht effizientere (Speicher- und Zeitbedarf) Berechnungen, beispielsweise bei der Verwendung des RSA.

Der Chinesische Restsatz wird historisch in China verortet und soll von mehr als 2000 Jahren entdeckt worden sein. Einzelne Überlieferungen ordnen ihn dem Militärstrategen und Philosophen Sun Tzu zu, doch letzteres ist eher Spekulation.

Der Chinesische Restsatz (2)

Der Chinesische Restsatz (CRT) findet sich in der Literatur in unterschiedlichen (doch äquivalenten) Varianten. Die gängige Formulierung lautet wie folgt:

Chinesischer Restsatz:

Seien $m_1, m_2, ..., m_k$ paarweise teilerfremde natürliche Zahlen, und bezeichne M das Produkt dieser m_i: $M = \prod_{i=1}^{k} m_i$. Dann hat das nachfolgende Gleichungssystem eine eindeutige Lösung:

$x = a_1 \pmod{m_1}$

$x = a_2 \pmod{m_2}$

...

$x = a_k \pmod{m_k}$

Anmerkung: im Zusammenhang mit dem RSA-Verfahren benötigen wir insbesondere den Fall k = 2, wobei $m_1 = p$, $m_2 = q$, und der RSA-Modul $n = pq$.

Der Chinesische Restsatz (3)

Wir geben noch eine andere Formulierung des CRT. Wieder seien m_1, m_2, ..., m_k paarweise teilerfremde natürliche Zahlen und M deren Produkt: $M = \prod_{i=1}^{k} m_i$.

Betrachten wir eine Zahl $A \in \mathbb{Z}_M$, so stellen wir A dar als k-Tupel wie folgt:

$A \leftrightarrow (A \bmod m_1, A \bmod m_2, \ldots, A \bmod m_k)$.

Behauptung:

(i) obige Zuordnung zwischen $A \in \mathbb{Z}_M$ und dem jeweiligen k-Tupel ist eine Bijektion zwischen $\mathbb{Z}_M$ und dem kartesischen Produkt $\mathbb{Z}_{m_1} \times \ldots \times \mathbb{Z}_{m_k}$.

(ii) Rechenoperationen können statt auf $\mathbb{Z}_M$ ebensogut auf dem jeweiligen k-Tupel ausgeführt werden, durch Berechnungen auf den einzelnen Moduln m_i.

Der Chinesische Restsatz (4)

Betrachten wir die Zuordnung $A \leftrightarrow (A \bmod m_1, A \bmod m_2, \ldots, A \bmod m_k)$.

"→": die Abbildung von $A \rightarrow (A \bmod m_1, A \bmod m_2, \ldots, A \bmod m_k)$ ist offensichtlich eindeutig möglich, hier ist nichts weiter zu zeigen.

"←": angenommen wir haben das Tupel $(a_1, a_2, \ldots, a_k)$, mit $a_i = A \bmod m_i$. Wie kommen wir von diesem wieder auf unser A?

Setze $M_i := \frac{M}{m_i}$ für alle i = 1, ..., k. Es ist also $M_i = m_1 \cdot m_2 \cdot \ldots \cdot m_{1i-1} \cdot m_{1i+1} \cdot \ldots \cdot m_k$.

Es gilt $M_i = 0 \pmod{m_j}$ für alle $i \neq j$.

M_i ist teilerfremd zu m_i , besitzt also ein multiplikatives Inverses $M_i^{-1} \bmod m_i$.

$A := \sum_{i=1}^{k} [(A \bmod m_i) \cdot Mi \cdot Mi^{-1} \bmod mi] \pmod M$

Der Chinesische Restsatz (5)

Teil (ii) bedeutet folgendes: Seien

$A \leftrightarrow (A \bmod m_1, A \bmod m_2, \ldots, A \bmod m_k)$,

$B \leftrightarrow (B \bmod m_1, B \bmod m_2, \ldots, B \bmod m_k)$,

dann können wir wie folgt rechnen:

$A + B \pmod M \leftrightarrow (A+B \bmod m_1, A+B \bmod m_2, \ldots, A+B \bmod m_k)$,

$A - B \pmod M \leftrightarrow (A-B \bmod m_1, A-B \bmod m_2, \ldots, A-B \bmod m_k)$,

$A \cdot B \pmod M \leftrightarrow (A \cdot B \bmod m_1, A \cdot B \bmod m_2, \ldots, A \cdot B \bmod m_k)$.

Anmerkung: aus obiger Darstellung des CRT ist unmittelbar ersichtlich, wie wir Rechenoperationen vom “großen” Modul M herunterbrechen können auf die kleineren Moduln m_i.

Zufall

Gibt es Zufall überhaupt???

Frage in die Runde:

wer ist der Meinung, es gebe gar keinen Zufall und alles sei deterministisch?!

…und falls ja: gilt dies nur für “wesentliche” Zukunftsfragen, oder auch für jeden kleinen Münzwurf?

Der Zufallsbegriff (1)

Wir zitieren einen Auszug aus Wikipedia zum Begriff ***Zufall***:

Wenn von Zufall gesprochen wird, kann konkret gemeint sein:

1. Ein Ereignis geschieht objektiv ohne Ursache.

   Fall 1 ist in der makroskopischen Welt bisher nicht beobachtet worden und dürfte prinzipiell nicht nachweisbar sein. In der Quantenmechanik wird die Existenz des objektiven Zufalls im Rahmen ihrer verschiedenen Interpretationen diskutiert. So ist der Zeitpunkt des Zerfalls des nächsten radioaktiven Atoms aus einer Stoffmenge nicht vorhersagbar.

2. Ein Ereignis geschieht, ohne dass eine Ursache erkennbar wäre.

   Fall 2 bedeutet, dass die Kausalkette oder die Einflussfaktoren nicht lückenlos nachgewiesen sind, aber ihr Vorhandensein zu vermuten ist. *Beispiele:*

   - Warum hat der Baum gerade hier einen Ast ausgebildet, im Gegensatz zum benachbarten Baum?

   - Bei der geschlechtlichen Vermehrung werden die Erbinformationen der Eltern neu kombiniert und zwar in einer Weise, die nicht vorherbestimmbar ist.

Der Zufallsbegriff (2)

3. Ein Ereignis geschieht, bei dem man zwar die Einflussfaktoren kennt, sie aber nicht messen oder steuern kann, so dass das Ergebnis nicht vorhersehbar ist („empirisch-pragmatischer Zufall“).

   Fall 3 setzt eine gewisse Komplexität voraus. *Beispiele:*

   Nicht manipulierte Glücksspielsituationen: Warum eine Roulette-Kugel gerade auf eine bestimmte Zahl gefallen ist, ist nicht vorhersehbar, weil in der Ausgangssituation (Wurf der Kugel) kleinste, nicht willentlich beeinflussbare Variationen großen Einfluss auf das Ergebnis haben. – Bei einem idealen Würfel kann für jeden Wurf ein Wert von 1 bis 6 auftreten. Vor dem Werfen kann nicht vorhergesagt werden, welches Ereignis eintritt. Es gibt keine Erklärung für das Auftreten einer bestimmten Zahl.

   Zwei – einander unbekannte – Menschen waren gleichzeitig im selben Eisenbahn-Abteil und kamen durch irgendein beobachtetes Ereignis ins Gespräch; bald darauf haben sie geheiratet und Kinder bekommen.

4. Zwei Ereignisse stehen in keinem (bekannten) kausalen Zusammenhang.

   Fall 4 ist der Versuch, voneinander unabhängige Dinge in Verbindung zu bringen. (Das ist eine der Formen magischen Denkens.) *Beispiel:* Zwei Menschen haben jeweils eine Telefonnummer. Ob der ältere oder jüngere die größere Nummer hat, ist „Zufall“.

Münzwurf: Zufall oder Pseudozufall?

In der Wahrscheinlichkeitstheorie muss der Münzwurf regelmäßig herhalten als Beispiel für ein Zufallsereignis. Wir gehen davon aus, dass niemand das Ergebnis eines Münzwurfs vorhersehen kann und bezeichnen dieses als echt zufällig.

Wie verhält es sich jedoch, wenn die Münze bereits in die Luft geworfen wurde, aber noch nicht gelandet ist? Lässt sich das Ergebnis vielleicht mithilfe noch nicht vorhandener, aber grundsätzlich möglicher, supergenauer Messgeräte vorhersagen?

Müssten wir einen schon “gestarteten” Münzwurf daher nicht eher als pseudozufällig bezeichnen?

Frage in die Runde: kennen Sie vergleichbare “Zufallsereignisse”, deren Zufallseigenschaft ggf. durch physikalische Messungen in Frage gestellt werden könnte?

Pseudozufall versus Zufall

- Wenn etwas nach Zufall aussieht, in Wirklichkeit jedoch berechenbar ist, bezeichnen wir es als ***Pseudozufall***.
- ***Pseudozufallszahlengeneratoren*** (***Pseudo random number generators PRNG***) liefern Zahlenwerte, die obiger Definition gerecht werden. Für kryptographische Zwecke ist uns dies aber noch nicht genug.
- Eine andere Erklärung bezeichnet etwas als pseudozufällig, wenn es aus der Perspektive des Betrachters nicht von echtem Zufall unterscheidbar ist.
- In der Praxis unterscheiden wir zwischen Pseudozufallszahlengeneratoren und kryptographisch sicheren Pseudozufallszahlengeneratoren.
- Wo liegt der Unterschied? Welche zusätzlichen Forderungen müssen wir stellen für kryptographische Sicherheit: Pseudozufallszahlen müssen (nur) statistischen Anforderungen genügen, können aber dennoch vorhersagbar sein. Letzteres müssen wir bei Kryptoschlüsseln jedoch vermeiden.

Wofür benötigen wir Zufall?

- In der Berechenbarkeitstheorie unterscheiden wir regelmäßig zwischen deterministischen und nichtdeterministischen Modellen (z.B. endliche Automaten, Kellerautomaten, Turingmaschinen etc.). Fragen für zuhause: Was würde es für diese Modelle bedeuten, falls es gar keinen echten Zufall gäbe? Würde Pseudozufall in irgendeiner Form aus diesem Dilemma helfen?
- In der Kryptographie benötigen wir sichere Schlüssel. Für diese verwendet man gemeinhin zufällig oder pseudozufällig erzeugte Bitstrings einer Länge n. Dabei ist von zentraler Bedeutung, dass ein Angreifer nicht in der Lage ist, den Suchraum $\{0,1\}^n$ in irgendeiner Weise einzuschränken oder potentiellen Schlüsseln unterschiedliche Wahrscheinlichkeiten zuzuordnen.
- Diese Eigenschaft wird in Frage gestellt, sobald wir nicht mehr mit echten Zufallswerten arbeiten, sondern mit Pseudozufallswerten.
- Aus diesem Grund ist beispielsweise das One-Time Pad nur dann beweisbar sicher, wenn der Schlüssel aus echten Zufallswerten erzeugt wurde.
- Frage in die Runde: definieren Sie “sicherer” (Verschlüsselungs-)Schlüssel…