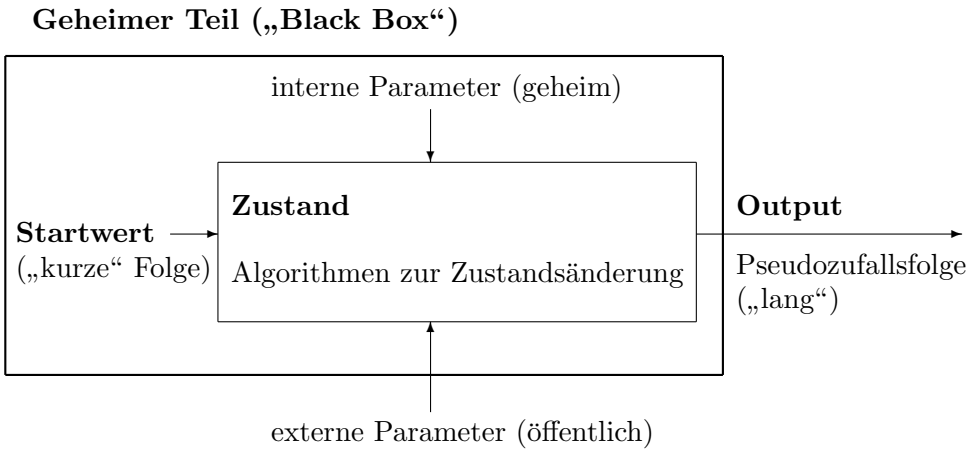
2 Kryptoanalyse von Zufallsgeneratoren

Schematisch sieht die Funktionsweise eines Zufallsgenerators so aus:



„Kryptoanalyse“ bedeutet für Zufallsgeneratoren, dass aus einem Teil ihres Outputs eine der folgenden Informationen bestimmt werden kann:

- die geheimen Parameter,
- der Startwert,
- weitere Teile des Outputs („Vorhersageproblem“).

2.1 Der allgemeine lineare Generator

Erinnern wir uns an die Beschreibung des allgemeinen linearen Generators: Gegeben sind

- als externe Parameter ein Ring R und ein R-Modul M,
- als interner Parameter eine lineare Abbildung $A\colon M \longrightarrow M$,
- als Zustand der Vektor $x_n \in M$,
- als Startwert der Vektor $x_0 \in M$,
- als Zustandsänderung die Rekursion $x_n = Ax_{n-1}$ für $n \geq 1$.

Bemerkung (Trivialfall): Falls A bekannt ist, ist aus jedem Folgeglied x_k die weitere Folge $(x_n)_{n\geq k}$ komplett vorhersagbar. Dieser Fall ist also kryptologisch völlig uninteressant. Die Rückwärtsberechnung von x_n mit $0 \leq n < k$ ist allerdings im allgemeinen nur möglich, wenn A injektiv ist. Das reicht natürlich nicht, um kryptologische Brauchbarkeit zu erreichen. Daher wird im folgenden meist nur das Problem der Vorwärtsberechnung behandelt und angenommen, dass ein Anfangsstück der Folge $x_0, \ldots, x_k$ bekannt ist. Trotzdem sollte man das Problem der Rückwärtsberechnung auch immer im Auge behalten.

Annahme also jetzt: R und M sind bekannt, A ist unbekannt, ein Anfangsstück $x_0, \ldots, x_k$ ist bekannt (o. B. d. A. $x_0 \neq 0$). Das *Vorhersageproblem* ist: Kann man daraus $x_{k+1}, x_{k+2}, \ldots$ bestimmen?

Man kann, wenn es einem gelingt, eine Linearkombination

$$x_k = c_1 x_{k-1} + \cdots + c_k x_0$$

zu bestimmen – also mit bekannten Koeffizienten $c_1, \ldots, c_k$. Dann ist nämlich

$$\begin{aligned} x_{k+1} &= Ax_k = c_1 Ax_{k-1} + \cdots + c_k Ax_0 \\ &= c_1 x_k + \cdots + c_k x_1 \\ &\vdots \\ x_n &= c_1 x_{n-1} + \cdots + c_k x_{n-k} \quad \text{für alle } n \geq k, \end{aligned}$$

die weitere Folge also komplett bestimmt – ohne dass man A kennt(!). Wie findet man eine solche Linearkombination?

Die Antwort liegt – natürlich – in der linearen Algebra. Im gegenwärtigen abstrakten Rahmen setzt man voraus, dass M noethersch ist (das ist die „richtige“ Verallgemeinerung von endlich-dimensionalen Vektorräumen); dann ist die aufsteigende Folge von Untermoduln

$$Rx_0 \subseteq Rx_0 + Rx_1 \subseteq \ldots \subseteq M$$

stationär, d. h., es gibt ein k mit $x_k \in Rx_0 + \cdots + Rx_{k-1}$: das ist die gesuchte lineare Relation. – Falls M endlich ist, wie wir es bei der Zufallserzeugung ja meist einrichten, ist die noethersche Eigenschaft selbstverständlich trivial. – Das erste solche k reicht, alle übrigen x_n, $n \geq k$, liegen dann auch in diesem Untermodul.

Wir haben also gezeigt:

Satz 1 (Noethersches Prinzip für lineare Generatoren) *Sei R ein Ring, M ein noetherscher R-Modul, $A\colon M \longrightarrow M$ linear und $(x_n)_{n\in\mathbb{N}}$ eine Folge in M mit $x_n = Ax_{n-1}$ für $n \geq 1$. Dann gibt es ein $k \geq 1$ und $c_1, \ldots, c_k \in R$ mit*

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} \quad \textit{für alle } n \geq k.$$

Jedes k mit $x_k \in Rx_0 + \cdots + Rx_{k-1}$ ist geeignet.

Wie bestimmt man aber den Index k und die Koeffizienten $c_1, \ldots, c_k$ praktisch? Dazu muss man natürlich in R und M rechnen können. Wir betrachten im folgenden zwei Beispiele: $R = K$ ein Körper oder $R = \mathbb{Z}/m\mathbb{Z}$ ein Restklassenring von ganzen Zahlen.

In beiden Fällen kann man von vornherein etwas darüber sagen, wie oft eine echte Zunahme in der Kette der Untermoduln vorkommen kann. Ist z. B. R ein Körper, so ist die Anzahl der echten Schritte durch die Vektorraum-Dimension $\operatorname{Dim} M$ beschränkt. Allgemein gilt:

Satz 2 (Krawczyk) *Sei M ein R-Modul und $0 \subset M_1 \subset \ldots \subset M_l \subseteq M$ eine echt aufsteigende Kette von Untermoduln. Dann ist $2^l \leq \#M$.*

Dieser Satz ist natürlich nur dann nützlich, wenn M endlich ist. Aber das ist ja derjenige Fall, der für die Vorhersage von Kongruenzgeneratoren am meisten interessiert. Man kann dann auch $l \leq {}^2\log(\#M)$ schreiben. Das ist nicht so viel schlechter als die Abschätzung im Fall Körper/Vektorraum, beides endlich: $l \leq \operatorname{Dim}(M) \leq {}^2\log(\#M)/\,{}^2\log(\#R)$.

Beweis. Sei $b_i \in M_i - M_{i-1}$ für $i = 1, \ldots, l$ (mit $M_0 = 0$). Dann besteht die Menge

$$U = \{c_1 b_1 + \cdots + c_l b_l \mid \text{alle } c_i = 0 \text{ oder } 1\} \subseteq M$$

aus 2^l verschiedenen Elementen. Wären nämlich zwei davon gleich, so wäre ihre Differenz (für ein t mit $1 \leq t \leq l$) von der Form

$$e_1 b_1 + \cdots + e_t b_t = 0 \text{ mit } e_i \in \{0, \pm 1\}, e_t \neq 0.$$

Da $e_t = \pm 1 \in R^\times$, folgte $b_t = -e_t^{-1}(e_1 b_1 + \cdots + e_{t-1} b_{t-1}) \in M_{t-1}$, Widerspruch. Also ist $\#M \geq \#U = 2^l$. $\diamond$

2.2 Lineare Generatoren über Körpern

Hier wird der Spezialfall betrachtet, dass $R = K$ ein Körper und M ein endlich-dimensionaler K-Vektorraum ist (also ein noetherscher K-Modul).

Dann muss man nur das minimale k finden mit

$$\mathrm{Dim}(Kx_0 + \cdots + Kx_k) = \mathrm{Dim}(Kx_0 + \cdots + Kx_{k-1})$$

– diese Zahl ist dann notwendig $= k$ – und dann die Linearkombination

$$x_k = c_1 x_{k-1} + \cdots + c_k x_0.$$

Das ist eine Standard-Aufgabe der linearen Algebra.

Zur konkreten Berechnung wählt man eine feste Basis $(e_1, \ldots, e_r)$ von M. Sei

$$x_n = \sum_{i=1}^{r} x_{in} e_i$$

die jeweilige Basis-Darstellung. Da $\mathrm{Rang}(x_0, \ldots, x_{k-1}) = k$, gibt es eine Indexmenge $I = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, r\}$ mit $\#I = k$, so dass die Matrix

$$X = (x_{ij})_{i \in I, 0 \le j < k} = \begin{pmatrix} x_{i_1 0} & \ldots & x_{i_1 k-1} \\ \vdots & & \vdots \\ x_{i_k 0} & \ldots & x_{i_k k-1} \end{pmatrix}$$

invertierbar ist. Die bisher noch unbekannten c_j gewinnt man aus dem Ansatz

$$x_k = \sum_{j=0}^{k-1} c_j x_j,$$

also

$$\sum_{i=1}^{r} x_{ik} e_i = \sum_{j=0}^{k-1} \sum_{i=1}^{r} c_j x_{ij} e_i,$$

also

$$x_{ik} = \sum_{j=0}^{k-1} x_{ij} c_j \quad \text{für alle } i \in I,$$

oder in Matrixschreibweise:

$$\bar{x} = (x_{ik})_{i \in I} = X \cdot c.$$

Die Lösung ist

$$c = X^{-1} \cdot \bar{x}.$$

Damit sind auch schon die ersten beiden Aussagen des folgenden Zusatzes zu Satz 1 bewiesen:

Satz 3 *Zusätzlich zu den Voraussetzungen von Satz 1 sei $R = K$ ein Körper. Dann gilt:*

(i) *Das minimale geeignete k ist das kleinste mit* $\mathrm{Dim}(Kx_0+\cdots+Kx_k) = k$*; es ist* $k \leq \mathrm{Dim}\, M =: r$.

(ii) *Die Koeffizienten $c_1, \ldots, c_k$ werden durch Lösung eines linearen Gleichungsystems mit invertierbarer quadratischer Koeffizientenmatrix bestimmt, deren Einträge aus Basiskoeffizienten von $x_0, \ldots, x_{k-1}$ bestehen.*

(iii) *Ist $k = r$, so ist A eindeutig aus den Basiskoeffizienten von $x_0, \ldots, x_k$ bestimmbar.*

Beweis. (iii) Seien

$$X_1 = (x_r, \ldots, x_1),\ X_0 = (x_{r-1}, \ldots, x_0) \in M_r(K).$$

Dann ist $X_1 = AX_0$ in Matrix-Darstellung bezüglich der Basis $(e_1, \ldots, e_r)$ von M. Da $\mathrm{Rang}\, X_0 = r$, ist X_0 invertierbar und

$$A = X_1 X_0^{-1},$$

wie behauptet. $\diamond$

Ist A invertierbar, so kann man analog die Folge (x_n) auch rückwärts berechnen, sobald man ein Stück $x_t, \ldots, x_{t+r}$ der Länge $r + 1$ mit $\mathrm{Rang}(x_t, \ldots, x_{t+r-1}) = r$ gefunden hat.

Beispiel.

Im Fall eines r-stufigen homogenen linearen Kongruenzgenerators $x_n = a_1 x_{n-1} + \cdots a_r x_{n-r}$ über $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$ mit p prim ist

$$A = \begin{pmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ a_r & \ldots & a_2 & a_1 \end{pmatrix}, \quad \mathrm{Det}\, A = (-1)^r a_r.$$

Hier ist A also genau dann invertierbar, wenn $a_r \neq 0$, und das kann man o. B. d. A. annehmen – sonst ist die Rekursionstiefe $< r$.

Für die Vorhersage der Folge benötigt man dann höchstens $r + 1$ Zustandsvektoren, also $2r$ Folgenglieder:

Korollar 1 *Ein r-stufiger homogener linearer Kongruenzgenerator mit Primzahlmodul ist aus den $2r$ Folgegliedern $x_0, \ldots, x_{2r-1}$ vorhersagbar.*

Korollar 2 *Ein lineares Schieberegister der Länge l ist aus den ersten $2l$ Bits vorhersagbar.*

Korollar 3 *Ein homogener linearer Kongruenzgenerator mit Primzahlmodul ist aus x_0, x_1, ein inhomogener aus x_0, x_1, x_2, x_3 vorhersagbar.*

Im nächsten Abschnitt wird u. a. gezeigt, dass bereits x_0, x_1, x_2 genügen.

Damit sind lineare Schieberegister als Quelle von Schlüsselbits für eine Bitstrom-Chiffre ein- für allemal kryptologisch erledigt. – Sollte die Länge zusätzlich geheim sein, kann der Kryptoanalytiker sie durch sukzessives Probieren bestimmen; das erhöht die Schwierigkeit des Angriffs nur unwesentlich.

Bei linearen Kongruenzgeneratoren könnte allerdings noch der Fall interessant sein, dass der Modul m geheimgehalten wird (und eventuell nicht prim ist). Dieser Fall wird im folgenden ebenfalls erledigt.

2.3 Lineare Kongruenzgeneratoren mit bekanntem Modul

Die Behandlung hier ist elementar ohne Benutzung der allgemeinen Theorie der vorhergehenden Abschnitte.

Die Parameter a und b des linearen Kongruenzgenerators $x_n = ax_{n-1} + b \bmod m$ seien unbekannt, bekannt hingegen sei zunächst der Modul m.

Für die Vorhersage reichen, auch wenn m nicht prim ist, 3 aufeinanderfolgende Folgenglieder x_0, x_1, x_2, wie im folgenden gezeigt wird. Aus der Relation

$$x_2 - x_1 \equiv a(x_1 - x_0) \pmod{m}$$

erhält man sofort (falls $x_1 - x_0$ zu m teilerfremd ist – das wird zunächst angenommen)

$$a = \frac{x_2 - x_1}{x_1 - x_0} \bmod m,$$

wobei die Division $\bmod m$ vorzunehmen ist (mit dem erweiterten Euklidischen Algorithmus). Das Inkrement b ergibt sich aus

$$b = x_1 - ax_0 \bmod m.$$

Damit ist das Bildungsgesetz bekannt und die Folge total vorhersagbar.

Typisch war schon in diesem einfachen Fall die Verwendung der **Differenzenfolge**

$$y_i = x_i - x_{i-1} \quad \text{für } i \geq 1.$$

Sie gehorcht dem Bildungsgesetz

$$y_{i+1} \equiv ay_i \pmod{m}.$$

Zu beachten ist, dass die y_i auch negativ sein können; sie liegen im Bereich $-m < y_i < m$. Da m bekannt ist, könnte man sie durch $y_i \bmod m$ ersetzen, aber das spielte, wie gesehen, keine Rolle, und bei unbekanntem m – später – geht es sowieso nicht.

Hilfssatz 1 (von der Differenzenfolge) *Die Folge (x_i) sei von dem linearen Kongruenzgenerator mit Modul m, Multiplikator a und Inkrement b erzeugt. Sei (y_i) ihre Differenzenfolge, $c = \mathrm{ggT}(m, a)$ und $d = \mathrm{ggT}(m, y_1)$. Dann gilt:*

(i) *Folgende Aussagen sind äquivalent:*
  (a) *Die Folge (x_i) ist konstant.*
  (b) $y_1 = 0$.
  (c) *Für alle i ist $y_i = 0$.*

(ii) $\mathrm{ggT}(m, y_i) | \mathrm{ggT}(m, y_{i+1})$ *für alle i.*

(iii) $d | y_i$ *für alle i.*

(iv) *Ist* $\mathrm{ggT}(y_1, \ldots, y_t) = 1$ *für ein $t \geq 1$, so $d = 1$.*

(v) $c | y_i$ *für alle $i \geq 2$.*

(vi) *Ist* $\mathrm{ggT}(y_2, \ldots, y_t) = 1$ *für ein $t \geq 2$, so $c = 1$.*

(vii) $m|y_iy_{i+2} - y_{i+1}^2$ *für alle* i.
(viii) *Sind* $\tilde{a}, \tilde{m}$ *ganze Zahlen,* $\tilde{m} \geq 1$, *mit* $y_i \equiv \tilde{a}y_{i-1} \pmod{\tilde{m}}$ *für* $i = 2, \ldots, r$, *so gilt* $x_i = \tilde{a}x_{i-1} + \tilde{b} \bmod \tilde{m}$ *für alle* $i = 1, \ldots, r$ *mit* $\tilde{b} = x_1 - \tilde{a}x_0 \bmod \tilde{m}$.

Beweis. (i) Es ist nur zu bemerken, dass mit einem y_i auch alle folgenden 0 sind.

(ii) Ist e Teiler von y_i und m, so wegen $y_{i+1} = ay_i + k_im$ auch Teiler von y_{i+1}.

(iii) ist ein Spezialfall von (ii).

(iv) gilt, weil $d|\operatorname{ggT}(y_1, \ldots, y_t)$ nach (iii).

(v) Sei $m = c\tilde{m}$ und $a = c\tilde{a}$. Dann ist $y_{i+1} = c\tilde{a}y_i + k_ic\tilde{m}$, also $c|y_{i+1}$ für $i \geq 1$.

(vi) gilt, weil $c|\operatorname{ggT}(y_2, \ldots, y_t)$ nach (v).

(vii) $y_iy_{i+2} - y_{i+1}^2 \equiv a^2y_i - a^2y_i \pmod{m}$.

(viii) durch Induktion: Für $i = 1$ ist die Behauptung die Definition von $\tilde{b}$. Für $i \geq 2$ folgt

$$x_i - \tilde{a}x_{i-1} - \tilde{b} \equiv x_i - \tilde{a}x_{i-1} - x_{i-1} + \tilde{a}x_{i-2} \equiv y_i - \tilde{a}y_{i-1} \equiv 0 \pmod{\tilde{m}},$$

wie behauptet. $\diamond$

Der triviale Fall der konstanten Folge braucht nicht weiter untersucht zu werden. Man erkennt an ihm aber, dass die Parameter eines linearen Kongruenzgenerators oft nicht eindeutig durch die erzeugte Folge bestimmt sind. Zum Beispiel kann man die konstante Folge mit einem beliebigen Modul m und einem beliebigen Multiplikator a erzeugen, wenn man nur das Inkrement $b = -(a-1)x_0 \bmod m$ setzt. Auch bei gegebenem m ist a dabei noch nicht eindeutig festgelegt, nicht einmal $a \bmod m$.

Im oben behandelten Fall war y_1 zu m teilerfremd und somit $a = y_2/y_1 \bmod m$. Im allgemeinen kann es allerdings passieren, dass die Division $\bmod m$ gar nicht eindeutig ist; genau dann trifft das zu, wenn m und y_1 nicht teilerfremd sind, also $d = \operatorname{ggT}(m, y_1) > 1$ ist. Die **reduzierte Differenzenfolge** $\bar{y}_i = y_i/d$ (vgl. (iii) in Hilfssatz 1) folgt dann der Rekursionsformel

$$\bar{y}_{i+1} \equiv \bar{a}\bar{y}_i \pmod{\bar{m}}$$

mit dem reduzierten Modul $\bar{m} = m/d$ und reduzierten Multiplikator $\bar{a} = a \bmod \bar{m}$, aus der sich $\bar{a} = \bar{y}_2/\bar{y}_1$ eindeutig bestimmen lässt. Setzt man $\tilde{a} = \bar{a} + k\bar{m}$ mit einer beliebigen ganzen Zahl k und $\tilde{b} = x_1 - \tilde{a}x_0 \bmod m$, so folgt nach Hilfssatz 1 (viii) auch $x_i = \tilde{a}x_{i-1} + \tilde{b} \bmod m$ für alle $i \geq 1$. Damit ist der folgende Satz gezeigt:

Satz 4 *Die Folge* (x_i) *sei von einem linearen Kongruenzgenerator mit bekanntem Modul* m, *aber unbekanntem Multiplikator* a *und Inkrement* b *erzeugt. Dann ist die gesamte Folge aus* x_0, x_1 *und* x_2 *bestimmbar. Falls die*

Folge (x_i) nicht konstant ist, ist der Multiplikator a genau bis auf ein Vielfaches des reduzierten Moduls $\bar{m}$ bestimmt.

Man muss sich also auch hier unter Umständen damit begnügen, die Folge vorherzusagen, ohne letzte Gewissheit über die wirklich verwendeten Parameter erlangen zu können. Wer ein ganz einfaches Zahlenbeispiel möchte: Für $m = 24$, $a = 2k + 1$ mit $k \in [0 \ldots 11]$ und $b = 12 - 2k \bmod 24$ wird aus dem Startwert $x_0 = 1$ stets die Folge $(1, 13, 1, 13, \ldots)$ erzeugt.

2.4 Lineare Kongruenzgeneratoren mit unbekanntem Modul

Schwieriger wird es natürlich, wenn der Modul m ebenfalls unbekannt ist und sich auch nicht leicht erraten lässt. Es wird angenommen, dass man nur ein Stück $x_0, x_1, \ldots$ der Folge zur Verfügung hat. Überraschenderweise ist es leichter, zuerst den Multiplikator zu bestimmen. Aus dem folgenden Satz erhält man in wenigen Schritten einen geeigneten Wert dafür, der dann später bei der Suche nach dem Modul hilft. Man erkennt den noetherschen Ansatz in der Form $y_{t+1} \in \mathbb{Z}y_1 + \cdots + \mathbb{Z}y_t$ wieder.

Satz 5 (Plumstead) *Sei (y_i) die Differenzenfolge des linearen Kongruenzgenerators mit erzeugender Funktion $s(x) = ax + b \bmod m$, $m \geq 2$, und Startwert x_0. Sei $y_1 \neq 0$ und t die kleinste Zahl mit $e = \mathrm{ggT}(y_1, \ldots, y_t) \,|\, y_{t+1}$. Dann gilt:*

(i) $t < 1 + {}^2\log m$.

(ii) *Ist $e = c_1y_1 + \cdots + c_ty_t$ mit $c_i \in \mathbb{Z}$ und $a' = (c_1y_2 + \cdots + c_ty_{t+1})/e$, so ($a' \in \mathbb{Z}$ und)*

$$y_{i+1} \equiv a'y_i \pmod{m} \quad \textit{für alle } i.$$

(iii) *Mit $b' = x_1 - a'x_0$ gilt*

$$x_i = a'x_{i-1} + b' \bmod m \quad \textit{für alle } i.$$

Beweis. (i) Ist $e_j = \mathrm{ggT}(y_1, \ldots, y_j)$ kein Teiler von y_{j+1}, so $e_{j+1} \leq e_j/2$. Da $e_1 = |y_1| < m$, folgt $e = e_t < m/2^{t-1}$, also $t - 1 < {}^2\log m$.

(ii) Es ist

$$ae = c_1ay_1 + \cdots + c_tay_t \equiv c_1y_2 + \cdots + c_ty_{t+1} = a'e \pmod{m}.$$

Der größte gemeinsame Teiler d von m und y_1 teilt e nach Hilfssatz 1, also ist auch $d = \mathrm{ggT}(m, e)$. Die Kongruenz wird zuerst durch d geteilt:

$$a\frac{e}{d} \equiv a'\frac{e}{d} \pmod{\bar{m}}$$

mit dem reduzierten Modul $\bar{m} = m/d$. Da e/d zu $\bar{m}$ teilerfremd ist, kann man es wegdividieren:

$$a \equiv a' \pmod{\bar{m}}, \;\; a = a' + k\bar{m}.$$

Also ist $y_{i+1} \equiv ay_i = a'y_i + ky_i\bar{m} \pmod{m}$. Da $d \,|\, y_i$, folgt $y_i \equiv 0$, also $y_{i+1} \equiv a'y_i \pmod{m}$.

(iii) folgt aus Hilfssatz 1 (viii). $\diamond$

Bemerkungen und Beispiele

1. Sei $m = 8397$, $a = 4381$ und $b = 7364$ [Reeds 1977]. Damit wird erzeugt

$$\begin{array}{lll} x_0 = 2134 & & \\ x_1 = 2160 & y_1 = 26 & e_1 = 26 \\ x_2 = 6905 & y_2 = 4745 & e_2 = 13 \\ x_3 = 3778 & y_3 = -3127 & e_3 = 1 \\ x_4 = 8295 & y_4 = 4517 & \end{array}$$

   Es folgt $c_1 = 87542$, $c_2 = -481$, $c_3 = -1$ und $a' = 416881843$.

2. Sei $m = 2^q + 1$, $a = 2^{q-1}$, $b = 2^q$ und $x_0 = 0$. Nach dem Korollar zum folgenden Hilfssatz 2 ist $y_i = (-1)^{i-1} \cdot 2^{q-i+1}$ für $i = 1, \ldots, q+1$ und daher $e_i = 2^{q-i+1}$. Damit ist $t = q+1$. Die Abschätzung für t im Satz 5 ist also scharf, und man braucht tatsächlich $q+3$ der Folgeglieder x_i, also x_0 bis x_{q+2}, um a' zu ermitteln.

Hilfssatz 2 *Die Folge (c_i) in $\mathbb{Z}$ sei durch $c_0 = 0$, $c_i = 2^{i-1} - c_{i-1}$ für $i \geq 1$, definiert. Dann ist*
(i) $c_i = \frac{1}{3} \cdot [2^i - (-1)^i]$ *für alle i,*
(ii) $c_i - 2c_{i-1} = (-1)^{i-1}$ *für alle $i \geq 1$.*

Beweis. (i) zeigt man durch Induktion und (ii) durch direkte Rechnung. ◇

Korollar 1 *Die Folge (x_i) sei von dem linearen Kongruenzgenerator mit Modul $m = 2^q + 1$, Multiplikator $a = 2^{q-1}$, Inkrement $b = 2^q$ und Startwert $x_0 = 0$ erzeugt; (y_i) sei ihre Differenzenfolge. Dann gilt:*
(i) $x_i = c_i \cdot 2^{q-i+1}$ *für $i = 0, \ldots, q+1$,*
(ii) $y_i = (-1)^{i-1} \cdot 2^{q-i+1}$ *für $i = 1, \ldots, q+1$.*

Ein „Ersatzmultiplikator“ a' läßt sich mit Hilfe von Satz 5 also effizient ermitteln. Nun fehlt noch ein Verfahren zur Ermittlung des Moduls m. Dieser wird durch „sukzessive Korrektur“ eingekesselt; im j-ten Schritt wird ein „Ersatzmodul“ m_j und ein „Ersatzmultiplikator“ a_j bestimmt:

- Im ersten Schritt setzt man $m_1 = \infty$ und $a_1 = a'$. [Rechnen mod ∞ soll einfach Rechnen mit ganzen Zahlen bedeuten, und $\mathrm{ggT}(c, \infty)$ soll c sein, wenn $c \neq 0$, und ∞, wenn $c = 0$.]

- Im j-ten Schritt, $j \geq 2$, sei $y'_j := a_{j-1} y_{j-1} \bmod m_{j-1}$. Dann setzt man $m_j = \mathrm{ggT}(m_{j-1}, y'_j - y_j)$ und $a_j = a_{j-1} \bmod m_j$.

Es wird also stets mit den aktuellen Ersatzwerten m_{j-1} und a_{j-1} für m und a eine Voraussage y'_j für y_j gemacht und diese mit dem tatsächlichen Wert y_j verglichen. Stimmen diese beiden Zahlen nicht überein, so unterscheiden sie sich um ein Vielfaches von m; dann werden die Ersatzwerte

korrigiert. Stets gilt $m \mid m_j$. Die j-te Korrektur ändert an der bisherigen Rechnung nichts, denn $y_i \equiv a_j y_{i-1} \pmod{m_j}$ für $i = 2, \ldots, j$, und auch $y_i \equiv a_j y_{i-1} \pmod{m}$ für alle $i \geq 2$. Auch die eigentliche Folge (x_i) erfüllt stets $x_i \equiv a_j x_{i-1} + b_j \pmod{m_j}$ für $i = 1, \ldots, j$ mit $b_j = x_1 - a_j x_0$ nach Hilfssatz 1 (viii).

Im oben gerechneten Beispiel 1 ist

$$\begin{array}{lll} & m_1 = \infty & a_1 = 416881843 \\ y_2' = 10838927918 & m_2 = 10838923173 & a_2 = 416881843 \\ y_3' = 5420327549 & m_3 = 8397 & a_3 = 4381 \end{array}$$

Der Wert m_3 errechnet sich als

$$\mathrm{ggT}(10838923173, 5420330676) = 8397.$$

Da $m_3 \leq 2x_2$, ist $m = m_3$, $a = a_3$ und $b = x_1 - ax_0 \bmod m = 7364$. Wir haben also die korrekten Werte mit zwei Korrekturen gefunden und dabei keine weiteren Folgenglieder gebraucht als die fünf, die schon zur Bestimmung von a' nötig waren. Auffallend sind die großen Zwischenergebnisse, so dass man mit der gewöhnlichen Ganzzahlarithmetik nicht mehr auskommt, sondern eine Arithmetik mit erweiterter Stellenzahl braucht.

Kommt das Verfahren stets zum Ziel? Spätestens wenn die Periode der Folge erreicht ist, also nach höchstens m Schritten, ist die gesamte Folge korrekt voraussagbar. Diese Schranke hat allerdings keinen praktischen Wert. Leider ist sie schon scharf: Bei beliebigem m sei $a = 1$, $b = 1$ und $x_0 = 0$. Dann ist $x_i = i$ und $y_i = 1$ für $i = 0, \ldots, m-1$. Der Startwert für den Ersatzmultiplikator ist $a' = 1$. Die erste falsche Voraussage ist $y_m' = 1$ statt des korrekten Werts $y_m = 1 - m$. Erst nach Auswertung von x_m ist das Verfahren beendet. Nun ist dieser schlechteste Fall leicht erkennbar und separat zu behandeln. Er erschwert aber das Auffinden guter allgemeingültiger Ergebnisse, und in der Tat sind keine solchen bekannt.

Ein etwas anderer Gesichtspunkt ergibt sich, wenn man die Anzahl der notwendigen Korrekturschritte zählt, also die Schritte, in denen der Ersatzmodul sich ändert. Ist nämlich $m_j \neq m_{j-1}$, so $m_j \leq m_{j-1}/2$. Sei $m^{(0)} = \infty > m^{(1)} > \ldots$ die Folge der *verschiedenen* Ersatzmoduln. Dann gilt

$$m^{(1)} = m_{j_1} = |y_{j_1}' - y_{j_1}| < a'|y_{j_1-1}| + m < m(a'+1),$$

$$m \leq m^{(j)} < \frac{m(a'+1)}{2^{j-1}},$$

also stets $j < 1 + {}^2\log(a'+1)$. Damit ist eine obere Schranke für die Anzahl der nötigen Korrekturen gefunden. Joan Plumstead-Boyar gab auch einen Algorithmus an, der zu einem eventuell kleineren Wert von a' und zu der oberen Schranke $2 + {}^2\log m$ für die Anzahl der Korrekturschritte führt. Allerdings wird diese Anzahl von Korrekturschritten in der Regel gar nicht erreicht, so dass die Schranke als Abbruchkriterium nichts nützt.

Hier scheint noch ein lohnendes Betätigungsfeld für die Suche nach theoretischen Ergebnissen offenzustehen. Lässt sich eine kleine Klasse von (vielleicht sowieso schlechten) linearen Kongruenzgeneratoren ausgrenzen, so dass für den großen Rest ein praktisch brauchbares Abbruchkriterium herleitbar ist? Das ist eigentlich zu erwarten. Lässt sich die Verteilung der nötigen Schrittzahl in den Griff bekommen? Wenigstens der Mittelwert? Jedenfalls reichen die vorliegenden Ergebnisse schon, um lineare Kongruenzgeneratoren endgültig als kryptologisch ungeeignet einzustufen.

2.5 Eine allgemeine Vorhersagemethode

Das Verfahren von PLUMSTEAD (die später unter dem Namen BOYAR publiziert hat) ist weitgehend durch das Verfahren von BOYAR/KRAWCZYK verallgemeinert worden: auf Rekursionsvorschriften, die sich durch eine Linearkombination *irgendwelcher* bekannten Funktionen ausdrücken lassen. Man beschreibt es zunächst wieder besonders passend in der Sprache der kommutativen Algebra, also durch Ringe und Moduln.

Sei also R ein kommutativer Ring (mit $1 \neq 0$), und X, Z seien R-Moduln. Gegeben sei eine Familie von Abbildungen

$$\Phi^{(i)} : X^i \longrightarrow Z \text{ für } i \geq h,$$

die wir uns als bekannt denken, und eine lineare Abbildung

$$\alpha : Z \longrightarrow X,$$

die als geheim angesehen wird (also als interner Parameter des zu beschreibenden Zufallsgenerators). Damit wird eine Folge $(x_n)_{n\in\mathbb{N}}$ in X erzeugt:

- $x_0, \ldots, x_{h-1} \in X$ werden als Startwerte gesetzt.
- Sind $x_0, \ldots, x_{n-1}$ schon erzeugt für $n \geq h$, so sei

$$\begin{aligned} z_n &:= \Phi^{(n)}(x_0, \ldots, x_{n-1}) \in Z, \\ x_n &:= \alpha(z_n) \in X. \end{aligned}$$

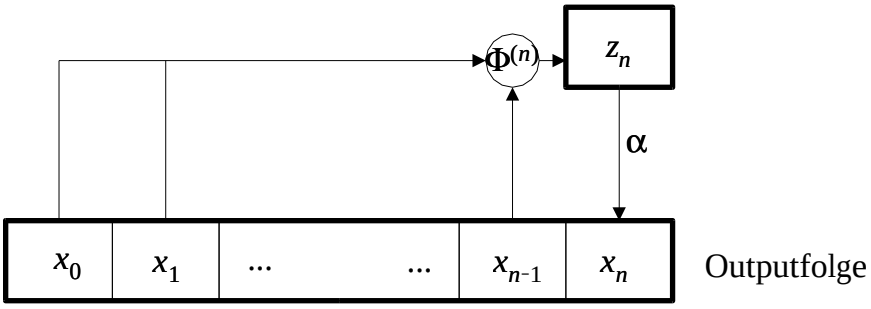


Abbildung 4: Ein allgemeiner Generator

Hier kann also, allgemeiner als bisher, jedes Folgenglied von *allen* vorhergehenden, also von der gesamten „Vergangenheit“ abhängen. Damit ein solches Verfahren sinnvoll zur Zufallserzeugung eingesetzt werden kann, müssen die $\Phi^{(i)}$ natürlich effizient berechenbar sein – im Fall $R = \mathbb{Z}/m\mathbb{Z}$ und $X = R^k$ etwa soll der Aufwand höchstens polynomial mit $\log(m)$ und k wachsen.

Beispiele

1. Der lineare Kongruenzgenerator: $R = \mathbb{Z}/m\mathbb{Z} = X$, $Z = R^2$, $h = 1$,

$$\Phi^{(i)}(x_0, \ldots, x_{i-1}) = \begin{pmatrix} x_{i-1} \\ 1 \end{pmatrix},$$

$$\alpha \begin{pmatrix} s \\ t \end{pmatrix} = as + bt.$$

2. Der linear-inversive Kongruenzgenerator: R, X, Z, h, α wie oben,

$$\Phi^{(i)}(x_0, \ldots, x_{i-1}) = \begin{pmatrix} x_{i-1}^{-1} \bmod m \\ 1 \end{pmatrix}.$$

3. Kongruenzgeneratoren höheren Grades: $R = \mathbb{Z}/m\mathbb{Z} = X$, $Z = R^{d+1}$, $h = 1$, $x_n = a_d x_{n-1}^d + \cdots + a_0$,

$$\Phi^{(i)}(x_0, \ldots, x_{i-1}) = \begin{pmatrix} x_{i-1}^d \\ \vdots \\ x_{i-1} \\ 1 \end{pmatrix},$$

$$\alpha \begin{pmatrix} t_0 \\ \vdots \\ t_d \end{pmatrix} = a_d t_0 + \cdots + a_0 t_d.$$

4. Beliebige Kongruenzgeneratoren: $R = \mathbb{Z}/m\mathbb{Z}$, $x_n = s(x_{n-1})$, $h = 1$. Ist m prim, so lässt sich jede Funktion $s : R \longrightarrow R$ als Polynom vom Grad $< m$ schreiben. Ist m zusammengesetzt, so verwendet man eben statt der Basis aus den Monomen die Basis $\{e_0, \ldots, e_{m-1}\}$ mit $e_i(j) = \delta_{ij}$ von R^R. Die Basis-Darstellung ist $s = \sum_{i=0}^{m-1} s(i)e_i$. Man nimmt $X = R$, $Z = R^m$ und

$$\Phi^{(i)}(x_0, \ldots, x_{i-1}) = \begin{pmatrix} e_0(x_{i-1}) \\ \vdots \\ e_{m-1}(x_{i-1}) \end{pmatrix},$$

$$\alpha \begin{pmatrix} t_0 \\ \vdots \\ t_{m-1} \end{pmatrix} = s(0)t_0 + \cdots + s(m-1)t_{m-1}.$$

Dass die $\Phi^{(i)}$ effizient berechenbar sein sollen, kann hier, egal welche Basis verwendet wird, nur bedeuten, dass eine Familie s_m von Funktionen auf $\mathbb{Z}/m\mathbb{Z}$ gegeben ist, die sich einheitlich als Linearkombination einer Teilmenge der Basis beschreiben lassen, die höchstens polynomial mit $\log(m)$ wächst.

5. Mehrstufige Kongruenzgeneratoren werden natürlich auch erfasst, wenn man h gleich der Rekursionstiefe nimmt.

6. Auch nichtlineare Schieberegister sind Beispiele, siehe den nächsten Abschnitt 2.6.

Für die Kryptoanalyse nimmt man wie gesagt an, dass die $\Phi^{(i)}$ bekannt sind, aber α unbekannt ist. (Später wird im Fall $R = \mathbb{Z}/m\mathbb{Z}$ auch noch m als unbekannt angenommen.) Die Frage ist: Kann man aus einem Anfangsstück $x_0, \ldots, x_{n-1}$ $(n \geq h)$ der Folge das nächste Glied x_n bestimmen?

Dazu betrachtet man die aufsteigende Kette $Z_h \subseteq Z_{h+1} \subseteq \ldots \subseteq Z$ von Untermoduln mit

$$Z_n = Rz_h + \cdots + Rz_n.$$

Falls $Z_n = Z_{n-1}$, ist $z_n = t_h z_h + \cdots + t_{n-1} z_{n-1}$ mit $t_h, \ldots, t_{n-1} \in R$ und daher

$$x_n = t_h x_h + \cdots + t_{n-1} x_{n-1}$$

bestimmbar ohne Verwendung von α. Ist Z ein noetherscher R-Modul, so wird nach endlich vielen Schritten der stationäre Zustand erreicht: $Z_n = Z_l$ für $n \geq l$. Ab dieser Stelle ist die Folge der x_n komplett vorhersagbar nach folgendem „Algorithmus“:

- Bilde $z_n = \Phi^{(n)}(x_0, \ldots, x_{n-1})$.
- Finde eine Linearkombination $z_n = t_h z_h + \cdots + t_{n-1} z_{n-1}$.
- Setze $x_n = t_h x_h + \cdots + t_{n-1} x_{n-1}$.

Damit aus dem „Algorithmus“ ein Algorithmus wird, muss das Verfahren im zweiten Schritt zum Finden einer Linearkombination algorithmisch durchführbar sein.

In unserem Standard-Beispiel mit (bekanntem) Modul $m = 8397$, $x_0 = 2134$, $x_1 = 2160$, $x_2 = 6905$ ist

$$z_1 = \begin{pmatrix} 2134 \\ 1 \end{pmatrix}, z_2 = \begin{pmatrix} 2160 \\ 1 \end{pmatrix}, z_3 = \begin{pmatrix} 6905 \\ 1 \end{pmatrix}.$$

Der Versuch, z_3 als Linearkombination $t_1 z_1 + t_2 z_2$ zu schreiben, führt auf das Gleichungssystem (in $R = \mathbb{Z}/8397\mathbb{Z}$)

$$\begin{aligned} 2134t_1 + 2160t_2 &= 6905, \\ t_1 + t_2 &= 1. \end{aligned} \tag{1}$$

Durch Elimination kommt man auf $26t_1 = -4745 = 3652$. Das Inverse von 26 mod 8397 ist 323, und daraus ergibt sich $t_1 = 4016$, $t_2 = 4382$. Damit wird $x_3 = 3778$ korrekt vorhergesagt.

Auch der Rest der Folge wird so korrekt vorhergesagt, denn es ist schon $Z_2 = Z$: Da $z_2 - z_1 = \binom{26}{0}$, ist $e_1 = \binom{1}{0} \in Z_2$ und $e_2 = \binom{0}{1} = z_1 - 2134 \cdot e_1 \in Z_2$.

Das Beispiel liefert auch eine Teilantwort auf die Frage, wann die Kette der Z_n stationär wird: Spätestens bei $Z_l = Z$, wenn das überhaupt vorkommt. Im allgemeinen kann man das nicht erwarten. Es folgt im allgemeinen auch nicht notwendig aus $Z_l = Z_{l+1}$, dass die Kette bei Z_l schon stationär ist – sie könnte später wieder ansteigen. Eine Schranke dafür, wie oft ein echter Anstieg möglich ist, gibt Satz 2.

In einem Schleifendurchlauf des Vorhersage-Algorithmus sind zwei Ereignisse möglich:

- $z_n \notin Z_{n-1}$. Dann ist keine Vorhersage für x_n möglich, aber Z_{n-1} wird zu $Z_n = Z_{n-1} + Rz_n$ erweitert, und zwar echt.
- $z_n \in Z_{n-1}$. Dann wird x_n korrekt vorhergesagt.

Der Satz besagt, dass das erste dieser Ereignisse höchstens ${}^2\log(\#Z)$-mal vorkommen kann. Bei jedem dieser Vorkommnisse braucht man dann den Zugriff auf das Folgeglied x_n, um weiter zu kommen. Das befriedigt nicht ganz, entspricht bei genauem Hinsehen aber der Situation des Kryptoanalytikers, der beim Brechen einer Verschlüsselung mit einem vermuteten Schlüssel weiterarbeitet, bis sinnloser Text entsteht, dann die nächsten Zeichen zu erraten versucht, seinen vermuteten Schlüssel korrigiert und damit weiter entschlüsselt. Im übrigen kennen wir diese Situation ja schon aus dem vorigen Abschnitt. Bemerkenswert ist, dass der neue Algorithmus recht einfach ist, aber sich auch ganz auf die Vorhersage konzentriert und nicht versucht, die unbekannten Parameter zu bestimmen.

2.6 Nichtlineare Schieberegister

Als weiteres Beispiel für die allgemeine Vorhersagemethode werden hier beliebige, nicht notwendig lineare, Schieberegister behandelt. Ein solches wird durch Abbildung 5 beschrieben.

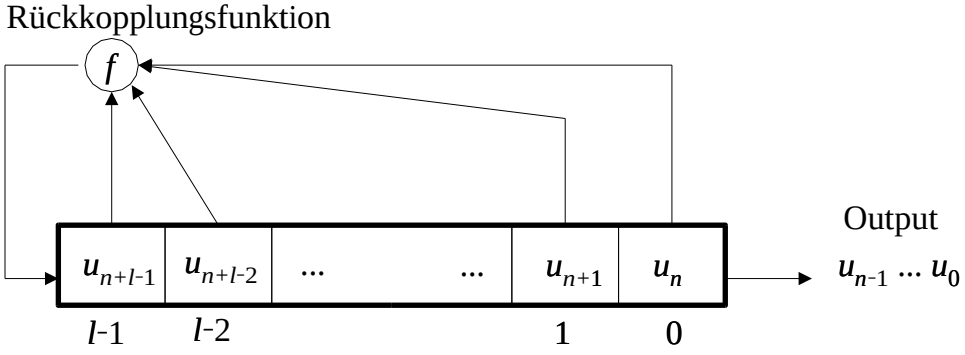


Abbildung 5: Ein Schieberegister der Länge l

Hierbei ist die Rückkopplungsfunktion $f : \mathbb{F}_2^l \longrightarrow \mathbb{F}_2$ eine beliebige Boolesche Funktion, die sich in algebraischer Normalform als Polynom

$$f(y_1, \ldots, y_l) = \sum_{I \subseteq \{1,\ldots,l\}} a_I y^I \qquad \text{mit } y^I = \prod_{j \in I} y_j$$

schreiben lässt.

Die Funktion f ist genau dann effizient (etwa durch ein Boolesches Schaltnetz) berechenbar, wenn „fast alle“ Koeffizienten $a_I = 0$ sind; d. h., es gibt ein Polynom $p \in \mathbb{N}[X]$ mit

$$\#\{I \mid a_I \neq 0\} \leq p(l).$$

Es ist dem Kryptoanalytiker allerdings nicht bekannt, welche $a_I \neq 0$ sind – vielmehr ist es eins seiner Ziele, das herauszubekommen.

Für die Anwendung der Vorhersagemethode wird $R = X = \mathbb{F}_2$, $h = l$, $Z = \mathbb{F}_2^{2^l}$ gesetzt. Für $i \geq l$ ist

$$\Phi^{(i)} : \mathbb{F}_2^i \longrightarrow Z$$

gegeben durch

$$z_i := \Phi^{(i)}(x_1, \ldots, x_i) = (y^I)_{I \subseteq \{1,\ldots,l\}} \qquad \text{mit } y = (x_{i-l+1}, \ldots, x_i).$$

Und schließlich ist

$$\alpha : Z \longrightarrow X, \quad \alpha((t^I)_{I \subseteq \{1,\ldots,l\}}) = \sum a_I t^I.$$

Zunächst zwei konkrete Beispiele für die Vorhersage:

Beispiele

1. $l = 2$, $f = T_1T_2 + T_1$. Aus den Startwerten $u_0 = 1$, $u_1 = 0$ wird die Folge
$$u_0 = 1,\ u_1 = 0,\ u_2 = 1,\ u_3 = 0,\ \ldots$$
erzeugt. Es ist
$$Z = \mathbb{F}_2^4, \qquad z_n = \begin{pmatrix} u_{n-1}u_{n-2} \\ u_{n-1} \\ u_{n-2} \\ 1 \end{pmatrix},$$
$$z_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}, \quad z_3 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}, \quad z_4 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} = z_2, \quad \ldots$$
Also vermutet der Kryptoanalytiker die lineare Rekursion
$$z_n = z_{n-2} = 0 \cdot z_{n-1} + 1 \cdot z_{n-2} \quad \text{für } n \geq 4$$
und sagt korrekt voraus
$$u_n = 0 \cdot u_{n-1} + 1 \cdot u_{n-2} = u_{n-2} \quad \text{für } n \geq 4.$$
Die Folge kann also auch durch ein *lineares* Schieberegister der Länge 2 erzeugt werden. Benötigt wurden u_0 bis u_3.

2. $l = 3$, $f = T_1T_3 + T_2$. Aus den Startwerten $u_0 = 0$, $u_1 = 1$, $u_2 = 1$ wird die weitere Folge
$$u_3 = 1,\ u_4 = 0,\ u_5 = 1,\ u_6 = 1,\ u_7 = 1,\ u_8 = 0,\ u_9 = 1,\ \ldots$$
erzeugt. Es ist
$$Z = \mathbb{F}_2^8, \qquad z_n = \begin{pmatrix} u_{n-1}u_{n-2}u_{n-3} \\ u_{n-1}u_{n-2} \\ u_{n-1}u_{n-3} \\ u_{n-2}u_{n-3} \\ u_{n-1} \\ u_{n-2} \\ u_{n-3} \\ 1 \end{pmatrix},$$
$$z_3 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}, \quad z_4 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}, \quad z_5 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}, \quad z_6 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}, \quad z_7 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} = z_3, \quad \ldots$$

Also ist die vermutete lineare Rekursion hier

$$z_n = z_{n-4} \quad \text{für } n \geq 4,$$

$$u_n = u_{n-4} \quad \text{für } n \geq 4,$$

und auch das ist wieder korrekt. Benötigt wurden u_0 bis u_6; und gefunden wurde ein „äquivalentes“ lineares Schieberegister der Länge 4.

Wegen der exponentiellen Zunahme der Dimension von Z sieht es zunächst so aus, als ob das Vorhersageverfahren bald an seine Grenzen stößt; der stationäre Zustand der aufsteigenden Unterräume, d. h., die gesuchte lineare Relation, wird womöglich erst nach 2^l Schritten erreicht; die Bitfolge kann also auch durch eine lineares Schieberegister der Länge $\leq 2^l$ erzeugt werden, das durch das Verfahren von Boyar/Krawczyk explizit gefunden wird. Immerhin ist dabei noch ein Schieberegister der Länge 32 mit linearer Algebra im 2^{32}-dimensionalen binären Vektorraum mit realistischem Aufwand vorhersagbar.

Im allgemeinen Fall kommt aber ein anderer Gesichtspunkt zum Tragen: Die Rückkopplungsfunktion f hängt ja von 2^l Parametern ab. Um zu einem handhabbaren Schlüsselraum zu kommen, muss man die möglichen Koeffizienten $\neq 0$ *von vorneherein* auf eine polynomiale Anzahl beschränken. Diese Auswahl ist aber Teil des Algorithmus – etwa des in Hardware realisierten Schieberegisters – und nicht Bestandteil des Schlüssels, wird also nach dem Kerckhoffs-Prinzip früher oder später dem Gegner bekannt sein. Die Notwendigkeit, eine effizient berechenbare Rückkopplungsfunktion zu wählen, führt also dazu, dass die Vorhersagemethode ebenfalls effizient wird. Daher kann man sagen:

Satz 6 *Jede durch ein Schieberegister mit effizient berechenbarer Rückkopplungsfunktion erzeugte Bitfolge ist vorhersagbar.*

Schieberegister, ob linear oder nichtlinear, sind somit zur Erzeugung kryptographisch brauchbarer Zufallsfolgen nicht geeignet – jedenfalls nicht bei direkter Verwendung. Das bedeutet nicht, dass das Verfahren von Boyar/Krawczyk eine Erfolgsgarantie für den Kryptoanalytiker liefert; allerdings kann der Kryptograph sich auch mit nichtlinearen Schieberegistern nicht sicher fühlen.

2.7 Der allgemeine Kongruenzgenerator

Etwas komplizierter, aber nicht entmutigend, wird das Vorhersageverfahren für Kongruenzgeneratoren, bei denen auch der Modul unbekannt ist. Hier bringt die allgemeine Sprache der kommutativen Algebra nicht mehr viel, da sehr spezielle Eigenschaften der Ringe $\mathbb{Z}$ und $\mathbb{Z}/m\mathbb{Z}$ verwendet werden, insbesondere das „kanonische“ Repräsentantensystem $\{0, \ldots, m-1\} \subseteq \mathbb{Z}$ von $\mathbb{Z}/m\mathbb{Z}$.

Sei $X = \mathbb{Z}^r$, $\bar{X} = (\mathbb{Z}/m\mathbb{Z})^r$, $Z = \mathbb{Z}^k$, $\bar{Z} = (\mathbb{Z}/m\mathbb{Z})^k$. Gegeben seien die Abbildungen

$$\Phi^{(i)} : X^i \longrightarrow Z \text{ für } i \geq h,$$

$$\alpha : \bar{Z} \longrightarrow \bar{X} \text{ linear,}$$

wobei α und m für die Kryptoanalyse als unbekannt behandelt werden. Mit Hilfe des kanonischen Repräsentantensystems wird $\bar{X}$ als Teilmenge $\{0, \ldots, m-1\}^r$ von X aufgefasst. Dann funktioniert die Erzeugung der Folge wie gehabt, und wir nennen das Verfahren einen **allgemeinen Kongruenzgenerator**, wenn die Berechnung aller $\Phi^{(i)}$ effizient möglich ist, d. h., mit einem Aufwand, der polynomial von r, k und $\log(m)$ abhängt. Insbesondere gibt es eine Schranke M für die Werte der $\Phi^{(i)}$ auf $\{0, \ldots, m-1\}^i$, die polynomial in r, k und $\log(m)$ ist.

Die Kryptoanalyse wird in zwei Phasen unterteilt. In der ersten Phase wird über dem Ring $\mathbb{Z}$ bzw. seinem Quotientenkörper $\mathbb{Q}$ gearbeitet und ein Vielfaches $\hat{m}$ des Moduls m bestimmt. In der zweiten Phase arbeitet man über dem Ring $\mathbb{Z}/\hat{m}\mathbb{Z}$. Bei der Voraussage von x_n sind jetzt drei Ereignisse möglich:

- $z_n \notin Z_{n-1}$; der ($\mathbb{Q}$- oder $\mathbb{Z}/m\mathbb{Z}$-) Modul Z_{n-1} muss zu Z_n erweitert werden, für x_n ist keine Vorhersage möglich. (D. h., es muss ein weiteres Folgenglied anderswie beschafft werden; für die Kryptoanalyse bedeutet das: Man braucht weiteren bekannten Klartext.)

- x_n wird korrekt vorhergesagt.

- x_n wird falsch vorhergesagt. Dann wird der Modul $\hat{m}$ korrigiert.

In der ersten Phase ist Z_{n-1} der $\mathbb{Q}$-Vektorraum, der von $z_h, \ldots, z_{n-1}$ aufgespannt wird, wobei man natürlich redundante z_i einfach weglässt.

1. Fall: $z_n \notin Z_{n-1}$. Dann wird $Z_n = Z_{n-1} + \mathbb{Q}z_n$ gesetzt und x_n nicht vorhergesagt. Dieser Fall kann höchstens k-mal auftreten.

2. Fall: [Die lineare Voraussageformel] $z_n = t_h z_h + \cdots + t_{n-1} z_{n-1}$. Dann wird $x_n = t_h x_h + \cdots + t_{n-1} x_{n-1}$ vorhergesagt (als Element von $\mathbb{Q}^r$). (Es treten höchstens k der z_i in der konstruierten Basis von Z_{n-1} auf, also auch höchstens k von 0 verschiedene Koeffizienten t_i.)

3. Fall: Genauso, aber $\hat{x}_n = t_h x_h + \cdots + t_{n-1} x_{n-1}$ stimmt nicht mit x_n überein. Sei dann $d \in \mathbb{N}$ der Hauptnenner von $t_h, \ldots, t_{n-1}$. Dann ist

$$d\hat{x}_n = \alpha(dt_h z_h + \cdots + dt_{n-1} z_{n-1}) = \alpha(dz_n) = dx_n$$

in $\bar{X}$, also mod m gerechnet. Damit ist gezeigt:

Hilfssatz 3 *(*BOYAR*) Der größte gemeinsame Teiler $\hat{m}$ der Komponenten von $d\hat{x}_n - dx_n$ im 3. Fall ist ein Vielfaches des Moduls m.*

Die erste Phase liefert also ein Vielfaches $\hat{m} \neq 0$ des Moduls m. Der Aufwand dafür beträgt:

- höchstens $k+1$ Versuche, ein lineares Gleichungssystem mit höchstens k Unbekannten über $\mathbb{Q}$ zu lösen,
- eine Bestimmung des größten gemeinsamen Teilers von r Zahlen.

Daneben wird eine unbestimmte Anzahl von Folgegliedern x_n korrekt vorhergesagt, was jeweils ebenfalls mit der Lösung eines solchen linearen Gleichungssystems bezahlt wird.

Wie groß kann $\hat{m}$ sein? Zur Abschätzung braucht man eine obere Schranke M für alle Komponenten aller $\Phi^{(i)}$ auf $\{0, \ldots, m-1\}^i \subseteq X^i$. Zur Herleitung wird die Ungleichung von HADAMARD verwendet: Für beliebige Vektoren $x_1, \ldots, x_k \in \mathbb{R}^k$ gilt

$$|\operatorname{Det}(x_1, \ldots, x_k)| \leq \|x_1\|_2 \cdots \|x_k\|_2$$

mit der euklidischen Norm $\| \bullet \|_2$.

Hilfssatz 4 *$\hat{m} \leq (k+1) \cdot m \cdot \sqrt{k^k} \cdot M^k$, insbesondere wächst $\log(\hat{m})$ höchstens polynomial mit k, $\log(m)$ und $\log(M)$.*

Beweis. Der Koeffizientenvektor t ist Lösung eines linearen Gleichungssystems aus höchstens k Gleichungen mit ebensovielen Unbekannten. Die Koeffizienten z_i dieses Gleichungssystems sind durch M beschränkt. Nach der Ungleichung von HADAMARD für die Determinante und der CRAMERschen Regel sind Zähler dt_i und Nenner d der Lösung durch

$$\prod_{i=1}^{k} \sqrt{\sum_{j=1}^{k} M^2} = \prod_{i=1}^{k} \sqrt{kM^2} = \sqrt{k^k} \cdot M^k$$

beschränkt. Die Komponenten von $d\hat{x}_n$ sind also durch

$$\|d\hat{x}_n\|_\infty = \|\sum dt_i x_i\|_\infty \leq \sqrt{k^k} \cdot M^k \cdot \sum \|x_i\|_\infty \leq km \cdot \sqrt{k^k} \cdot M^k$$

beschränkt, weil m eine Schranke für die Komponenten der x_i ist. Daraus folgt

$$\|d\hat{x}_n - dx_n\|_\infty \leq km \cdot \sqrt{k^k} \cdot M^k + \sqrt{k^k} \cdot M^k \cdot m = (k+1) \cdot m \cdot \sqrt{k^k} \cdot M^k,$$

wie behauptet. $\diamond$

Wie sieht das im Beispiel des gewöhnlichen linearen Kongruenzgenerators aus? Hier ist

$$z_1 = \begin{pmatrix} x_0 \\ 1 \end{pmatrix}, z_2 = \begin{pmatrix} x_1 \\ 1 \end{pmatrix}, z_3 = \begin{pmatrix} x_2 \\ 1 \end{pmatrix}, \ldots$$

Falls $x_1 = x_0$, sind wir im trivialen Fall der konstanten Folge. Andernfalls ist z_3 rationale Linearkombination $t_1 z_1 + t_2 z_2$: Die Lösung des Gleichungssystems

$$\begin{aligned} x_0 t_1 + x_1 t_2 &= x_2, \\ t_1 + t_2 &= 1 \end{aligned}$$

ist

$$t = \frac{1}{d} \cdot \begin{pmatrix} -x_2 + x_1 \\ x_2 - x_0 \end{pmatrix} \quad \text{mit } d = x_1 - x_0.$$

Vorhergesagt wird dann

$$\hat{x}_3 = t_1 x_1 + t_2 x_2 = \frac{-x_2 x_1 + x_1^2 + x_2^2 - x_2 x_0}{x_1 - x_0} = \frac{(x_2 - x_1)^2}{x_1 - x_0} + x_2.$$

Also ist $d(\hat{x}_3 - x_3) = (x_2 - x_1)^2 - (x_1 - x_0)(x_3 - x_2) = y_2^2 - y_1 y_3$ mit der Differenzenfolge (y_i). Falls $\hat{x}_3 = x_3$, müssen wir weiter machen. Sonst erhalten wir, wie aus Hilfssatz 1, $m|\hat{m} = |y_1 y_3 - y_2^2|$.

Im Standard-Beispiel $x_0 = 2134$, $x_1 = 2160$, $x_2 = 6905$, $x_3 = 3778$, also mit $y_1 = 26$, $y_2 = 4745$, $y_3 = -3127$, erhalten wir mit diesem allgemeinen Ansatz

$$\hat{m} = 4745^2 + 26 \cdot 3127 = 22596327.$$

Sieht man aber im konkreten Fall genauer hin und wendet Hilfssatz 3 direkt an, erhält man

$$t_1 = -\frac{365}{2},\ t_2 = \frac{367}{2},\ \hat{x}_3 = \frac{1745735}{2},\ \hat{m} = 2 \cdot (\hat{x}_3 - x_3) = 1738179.$$

In der zweiten Phase des Algorithmus wird das gleiche Verfahren, aber über dem Ring $\hat{R} = \mathbb{Z}/\hat{m}\mathbb{Z}$ durchgeführt. Da man die rationalen Ergebnisse aus der ersten Phase nicht einfach $\bmod \hat{m}$ reduzieren kann, startet man wieder neu bei z_h. Es gibt wieder drei Fälle für jeden Einzelschritt:

1. Fall: $z_n \notin \hat{Z}_{n-1} = \hat{R}z_h + \cdots + \hat{R}z_{n-1}$. Dann wird $\hat{Z}_n = \hat{Z}_{n-1} + \hat{R}z_n$ gesetzt (und dieser $\hat{R}$-Modul durch ein nicht-redundantes Erzeugendensystem

$\{z_{j_1}, \ldots, z_{j_l}\}$ repräsentiert, wobei $z_{j_l} = z_n$). Hier wird x_n nicht vorhergesagt (sondern muss anderswie beschafft werden).

2. Fall: $z_n = t_h z_h + \cdots + t_{n-1} z_{n-1}$. Dann wird $x_n = t_h x_h + \cdots + t_{n-1} x_{n-1}$ vorhergesagt (als Element von $\hat{X} = (\mathbb{Z}/\hat{m}\mathbb{Z})^r$). Die Voraussage sei korrekt.

3. Fall: Genauso, aber die Voraussage $\hat{x}_n = t_h x_h + \cdots + t_{n-1} x_{n-1}$ stimmt in $\hat{X}$ nicht mit x_n überein. Dann wird $\hat{x}_n - x_n$ als Element von $\mathbb{Z}^r$ betrachtet:

Hilfssatz 5 *Der größte gemeinsame Teiler der Koeffizienten von $\hat{x}_n - x_n$ im 3. Fall ist ein Vielfaches von m, aber kein Vielfaches von $\hat{m}$.*

Beweis. Er ist ein Vielfaches von m, weil $\hat{x}_n \bmod m = x_n$ sein muss. Er ist kein Vielfaches von $\hat{m}$, weil sonst ja $\hat{x}_n = x_n$ in $\hat{X}$ wäre. $\diamond$

Im 3. Fall wird $\hat{m}$ durch den ggT dieses größten gemeinsamen Teilers mit $\hat{m}$ ersetzt und die ganze Kette der bisherigen z_j (soweit sie nicht schon redundant waren) $\bmod \hat{m}$ reduziert. Wegen der zweiten Aussage im Hilfssatz ist dabei $\hat{m}$ echt kleiner geworden.

Wegen Hilfssatz 4 kann der dritte Fall insgesamt nicht zu oft auftreten; die Anzahl der Vorkommnisse ist polynomial in k, $\log(m)$ und $\log(M)$. Ist das richtige m erreicht, kann dieser Fall gar nicht mehr vorkommen. Der erste Fall kann in der zweiten Phase insgesamt wegen Satz 2 höchstens ${}^2\log(\#(\mathbb{Z}/\hat{m}\mathbb{Z})^k) = k \cdot {}^2\log(\hat{m})$ Mal vorkommen, und diese Schranke ist polynomial in k, $\log(m)$ und $\log(M)$.

> **Anmerkung.** Die Gemeinsamkeit von erster und zweiter Phase besteht darin, dass beide Male über dem vollen Quotientenring gerechnet wird: Der volle Quotientenring von $\mathbb{Z}$ ist der Quotientenkörper $\mathbb{Q}$. In einem Restklassenring $\mathbb{Z}/m\mathbb{Z}$ dagegen sind die Nicht-Nullteiler genau die zu m teilerfremden Elemente, also die Einheiten. Daher ist $\mathbb{Z}/m\mathbb{Z}$ sein eigener voller Quotientenring.

Im Standard-Beispiel haben wir nach der ersten Phase $\hat{m} = 1738179$ und müssen nun das lineare Gleichungssystem (1) $\bmod \hat{m}$ lösen. Da dessen Determinante -26 zu $\hat{m}$ teilerfremd sind, ist bereits $Z_2 = \hat{R}^2$, und Fall1 wird nicht mehr auftreten. Es ist $-26t_1 = 4745$, also $t_1 = 868907$, da das Inverse von -26 gleich 66853 ist (alles in $\mathbb{Z}/\hat{m}\mathbb{Z}$). Damit folgt $t_2 = 1 - t_1 = 869273$, und $\hat{x}_3 = 1_1 x_1 + t_2 x_2 = 3778$ wird korrekt vorausgesagt.

Im nächsten Schritt werden die neuen Koeffizienten t_1 und t_2 in der Linearkombination $z_4 = t_1 z_1 + t_2 z_2$ bestimmt. Zu lösen ist also (in $\mathbb{Z}/\hat{m}\mathbb{Z}$)

$$\begin{aligned} 2134t_1 + 2160t_2 &= 3778, \\ t_1 + t_2 &= 1. \end{aligned}$$

Elimination von t_2 ergibt $-26t_1 = 1618$, also $t_1 = 401056$, und daraus $t_2 = 1337124$ sowie $\hat{x}_4 = 1_1 x_1 + t_2 x_2 = 302190$. Da $x_4 = 8295$, sind wir im

3. Fall und haben $\hat{m}$ zu korrigieren:

$$\mathrm{ggT}(\hat{x}_4 - x_4, \hat{m}) = \mathrm{ggT}(293895, 1738179) = 8397.$$

Weil $\hat{m} < 2x_2$, wird von jetzt an nur noch der zweite Fall auftreten, d. h., der Rest der Folge wird korrekt vorhergesagt.

Ein **Vorhersageverfahren** für einen allgemeinen Kongruenzgenerator ist ein Algorithmus, der als Eingabe die Startwerte $x_0, \ldots, x_{h-1}$ erhält, dann Schätzungen für $x_h, x_{h+1}, \ldots$ auswirft und diese anschließend mit dem jeweiligen wahren Wert vergleicht; bei einer Fehlvorhersage werden unter Verwendung des wahren Werts die Parameter des Verfahrens adjustiert. Das Vorhersageverfahren ist **effizient**, wenn

(a) der Aufwand für die Vorhersage jedes x_n polynomial in r, k und $\log(m)$ ist,

(b) die Zahl der Fehlvorhersagen durch ein Polynom in r, k und $\log(m)$ beschränkt ist, ebenso der Aufwand für die Parameteradjustierung im Fall einer Fehlvorhersage.

Der Algorithmus von Boyar/Krawczyk, den wir in diesem Abschnitt behandelt haben, erfüllt (b). Er erfüllt auch (a), da das Lösen linearer Gleichungssysteme über Restklassenringen $\mathbb{Z}/m\mathbb{Z}$ effizient möglich ist, wie schon früher gezeigt. Damit ist bewiesen:

Hauptsatz 1 *Für einen beliebigen effizienten Kongruenzgenerator ist der Algorithmus von* Boyar/Krawczyk *ein effizientes Vorhersageverfahren.*

Die Anwendung auf nichtlineare Generatoren wird an einem weiteren einfachen Zahlenbeispiel gezeigt. Von einem quadratischen Generator der Form

$$x_n = ax_{n-1}^2 + bx_{n-1} + c \bmod m$$

sei die Zahlenfolge

$$x_0 = 63,\ x_1 = 96,\ x_2 = 17,\ x_3 = 32,\ x_4 = 37,\ x_5 = 72$$

erzeugt worden. Wir verwenden also $X = \mathbb{Z}$, $Z = \mathbb{Z}^3$, $h = 1$. In der ersten Phase spannen

$$z_1 = \begin{pmatrix} 3969 \\ 63 \\ 1 \end{pmatrix} z_2 = \begin{pmatrix} 9216 \\ 96 \\ 1 \end{pmatrix} z_3 = \begin{pmatrix} 289 \\ 17 \\ 1 \end{pmatrix}$$

schon ganz $\mathbb{Q}^3$ auf, denn die Koeffizientenmatrix ist die Vandermonde-Matrix mit Determinante 119922. Die Auflösung von

$$z_4 = \begin{pmatrix} 1024 \\ 32 \\ 1 \end{pmatrix} = t_1 z_1 + t_2 z_2 + t_3 z_3$$

ergibt $t_1 = \frac{160}{253}, t_2 = -\frac{155}{869}, t_3 = \frac{992}{1817}$ mit Hauptnenner $d = 11{\cdot}23{\cdot}79 = 19987$. Die Voraussage ist $\hat{x}_4 = \frac{1502019}{19987} \neq x_4$. Der erste geschätzte Modul ist also

$$\hat{m} = d\hat{x}_4 - dx_4 = 762500.$$

Damit ist die erste Phase abgeschlossen. Das gleiche lineare Gleichungssystem soll jetzt über $\mathbb{Z}/\hat{m}\mathbb{Z}$ aufgelöst werden, wo die Determinante allerdings ein Nullteiler ist, und ergibt zwei Lösungen, darunter

$$t_1 = 156720, t_2 = 719505, t_3 = 648776.$$

Vorausgesagt wird also der korrekte Wert

$$\hat{x}_4 = 156720 \cdot 96 + 719505 \cdot 17 + 648776 \cdot 32 \bmod 763500 = 37.$$

Daher sind wir im Fall 2 und versuchen, x_5 vorauszusagen:

$$z_5 = \begin{pmatrix} 1369 \\ 37 \\ 1 \end{pmatrix} = t_1 z_1 + t_2 z_2 + t_3 z_3$$

ergibt zwei Lösungen, darunter

$$t_1 = 2010, t_2 = 558640, t_3 = 201851,$$

also

$$\hat{x}_5 = 136572, \quad \hat{x}_5 - x_5 = 136500.$$

Also wird $\hat{m}$ korrigiert zu

$$\mathrm{ggT}(762500, 136500) = 500.$$

Damit sind die bekannten Werte erschöpft. Da alle z_i in $\hat{Z}_3 = \hat{R}z_1 + \hat{R}z_2 + \hat{R}z_3 \neq \hat{R}^3$ liegen, bleibt der erste Fall bei der weiteren Vorhersage möglich. Da $x_0, \ldots, x_5$ kleiner als die Hälfte des aktuellen Moduls $\hat{m}$ sind, bleibt auch der dritte Fall möglich, der Modul muss also eventuell noch weiter korrigiert werden.

Für die Vorhersage von x_6 ergibt sich (mod500)

$$t_1 = 240, t_2 = 285, t_3 = 476, x_6 = 1117.$$

Übungsaufgabe. Was passiert, wenn man im Standardbeispiel nach der ersten Phase mit dem Wert $\hat{m} = 22596327$ weiterrechnet?

2.8 Analyse bei gestutztem Output

Schwieriger wird die Kryptoanalyse, wenn der Zufallsgenerator nicht alle Bits der erzeugten Zahlen ausgibt. Das kann auf Absicht beruhen, aber auch als Nebeneffekt dadurch entstehen, dass die Zahlen ins reelle Intervall $[0, 1]$ oder sonstwie transformiert und dabei gerundet werden; besonders wenn der Modul m keine Zweierpotenz ist, muss man den Verlust einiger gering signifikanter Bits in Erwägung ziehen. Die Differenzenfolge ist dann natürlich auch nur ungefähr bekannt, die größten gemeinsamen Teiler sind nicht mehr zu ermitteln, und die Algorithmen von PLUMSTEAD-BOYAR und BOYAR/KRAWCZYK brechen zusammen.

Sind die Parameter des Zufallsgenerators bekannt, kann man es mit systematischem Probieren versuchen. Für die folgende Überlegung (die im übrigen nicht streng durchgeführt wird) muss der Zufallsgenerator nicht einmal notwendig linear sein. Nehmen wir an, es werden n-Bit-Zahlen erzeugt, aber jeweils nur q Bits ausgegeben und $n - q$ Bits zurückgehalten. Die ausgegebenen Bits stammen jeweils von festen, bekannten Positionen. Dann gibt es zu jedem ausgegebenen q-Bit-Fragment 2^{n-q} mögliche vollständige Werte; anders ausgedrückt, enthält eine zufällig gewählte n-Bit-Zahl mit der Wahrscheinlichkeit $1/2^q$ die vorgegebenen Bits an den richtigen Stellen.

Die weitere Überlegung wird hier nur exemplarisch für den Fall durchgeführt, dass die q ausgegebenen Bits die Leitbits sind. Man zerlegt also den Startwert x in $x = x_0 2^{n-q} + x_1$ mit $0 \le x_1 < 2^{n-q}$. Der Wert x_0, die ersten q Bits, ist bekannt. Der Angreifer startet eine Exhaustion über die 2^{n-q} verschiedenen möglichen Werte für x_1. Zu jeder Wahl von x_1 bildet er $x = x_0 2^{n-q} + x_1$ und $y = s(x)$ mit der erzeugenden Funktion s des betrachteten Zufallsgenerators. Diesen Wert y vergleicht er mit den ihm bekannten führenden q Bits des wahren Wertes. Ist der Zufallsgenerator statistisch gut, so ist die Wahrscheinlichkeit eines Treffers hierbei $1/2^q$. Das bedeutet, dass von den 2^{n-q} Werten für x_0 noch ungefähr 2^{n-2q} übrig bleiben. Falls $q \ge \frac{n}{2}$, können wir also genau einen Treffer erwarten. Ansonsten wird weitergemacht. Nach k Schritten ist die Trefferzahl ungefähr 2^{n-kq}. Die zu erwartende nötige Schrittzahl ist also $\ge k$ nur, wenn $kq \le n$, also $q \ge \frac{n}{k}$. Bei $q = \frac{1}{4}$ (zum Beispiel $n = 32$, $q = 8$, d. h., Ausgabe von 8 Bit einer 32-Bit-Zahl) reichen also vier q-Bit-Fragmente (wobei man im Beispiel allerdings schon 2^{24} Zahlen durchprobieren muss). Dieses Probierverfahren ist bei kleinem Modul m durchführbar; der Aufwand wächst aber exponentiell mit m (wenn der Anteil $r = \frac{q}{n}$ der ausgegebenen Bits gegen 1 beschränkt bleibt).

Für lineare Kongruenzgeneratoren haben FRIEZE/KENNAN/LAGARIAS, HÅSTAD/SHAMIR und J. STERN ein besseres (probabilistisches) Verfahren entwickelt, dessen erster Schritt im folgenden Satz resümiert wird (ohne Beweis).

Satz 7 (FRIEZE, KANNAN, LAGARIAS) *Sei* $0 \le r \le 1$ *und* p_n *die Wahr-*

scheinlichkeit, dass das Verfahren nicht den Modul m eines linearen Kongruenzgenerators bestimmt. Dann gilt für beliebiges $\varepsilon > 0$

$$p_n = \mathrm{O}(n^{\frac{5r-3}{2}+\varepsilon}).$$

Insbesondere $p_n \to 0$ mit $n \to \infty$, wenn $r > \frac{2}{5}$. *Es gelingt also mit großer Wahrscheinlichkeit, m zu finden, wenn mehr als $2/5$ der Leitbits ausgegeben werden.*

Im zweiten Schritt geht es darum, den Multiplikator a unter der Annahme zu bestimmen, dass der Modul m schon bekannt ist. Im dritten Schritt sind noch die vollen Zahlen x_i oder die Differenzen y_i zu bestimmen. Auch dies gelingt außer für eine vernachlässigbare Menge von Multiplikatoren, die mit wachsendem m immer kleiner gewählt werden kann, und für die „guten“ Multiplikatoren benötigt man nur noch etwas mehr als ein Drittel der Leitbits von x_0, x_1, x_2 und x_3, um die volle Bitinformation herzuleiten. Ähnliche, etwas schwächere Ergebnisse hat J. Stern auch für den Fall gefunden, dass statt der Leitbits „innere Bits“ der erzeugten Zahlen ausgegeben werden.

Die Kryptoanalyse der linearen Kongruenzgeneratoren hat also grundsätzliche Schwächen aufgedeckt, und zwar unabhängig davon, ob ein solcher Generator statistisch gute oder schlechte Eigenschaften hat.

Trotzdem sind die linearen Kongruenzgeneratoren für statistische Anwendungen durchaus brauchbar, denn es scheint extrem unwahrscheinlich, dass ein Anwendungsprogramm „aus Versehen“ die nötigen Schritte enthält, um einen linearen Kongruenzgenerator zu knacken und so seinen Determinismus aufzudecken. Für die kryptographische Anwendung sind die linearen Kongruenzgeneratoren auch bei gestutztem Output aber ein für allemal disqualifiziert. Offen ist allerdings, ob die Einwände auch zutreffen, wenn man nur „ganz wenige“ Bits ausgibt (etwa nur ein Viertel oder gar nur $\log\log(m)$ Bits).

Literaturverweise

- J. Stern: Secret linear congruential generators are not cryptographically secure. FOCS 28 (1987), 421–426.
- Frieze/Håstad/Kannan/Lagarias/Shamir: Reconstructing truncated integer variables satisfying linear congruences. SIAM J. Comput. 17 (1988), 262–280.
- J. Boyar: Inferring sequences produced by a linear congruential generator missing low-order bits. J. Cryptology 1 (1989), 177–184.

2.9 Resümee

Die Abschnitte 2.1 bis 2.7 ergaben ein Vorhersageverfahren, das so abläuft:

1. Der Kryptoanalytiker findet durch Klartextraten ein Stück der Schlüssel-Bitfolge, so lange, bis sich eine geeignete lineare Relation aufstellen lässt (NOETHERsches Prinzip).

2. Er sagt mit Hilfe dieser linearen Relation weitere Schlüsselbits voraus.

3. Erweisen sich vorausgesagte Bits als falsch (weil der Klartext an dieser Stelle aufhört, sinnvoll zu sein), muss der Kryptoanalytiker wieder etwas Klartext raten und damit die Parameter adjustieren; dann kann er weiter vorhersagen.

Dieses Verfahren ist für die „klassischen“ Zufallsgeneratoren effizient, also für Kongruenzgeneratoren – auch bei unbekanntem Modul – und für Schieberegister – auch nichtlineare. „Effizient“ bedeutet hier auch, dass die benötigte Menge von bekanntem oder erratenem Klartext klein ist.

Das Fazit daraus ist, dass für kryptographisch sichere Zufallserzeugung niemals der Zustand des Zufallsgenerators direkt als Output verwendet werden sollte; vielmehr ist eine Transformation dazwischen zu schalten. Abschnitt 2.8 zeigt exemplarisch, dass das schlichte Unterdrücken einiger Bits, das „Stutzen“ oder die „Dezimierung“, als Output-Transformation aber auch nicht ohne weiteres ausreicht. Bessere Output-Transformationen werden in den folgenden Abschnitten behandelt.

Die Grauzone zwischen dem, was dem Kryptoanalytiker Erfolg garantiert, und dem, was den Kryptologen ruhig schlafen lässt, ist freilich sehr breit. Auf jeden Fall sollten besser beide Prozesse

- Zustandsänderung,

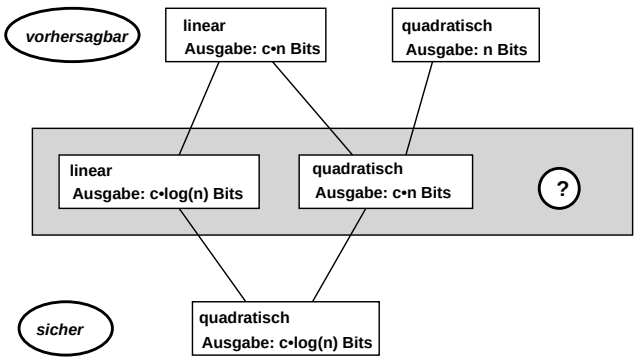
- Output-Transformation,

nichtlinear sein. In der Grauzone, wo keine nützlichen Aussagen über die Sicherheit bekannt sind, liegen unter anderem quadratische Kongruenzgeneratoren mit mäßig gestutztem Output.

Im Folgenden werden zwei Ansätze behandelt, zu sicheren Zufallsgeneratoren zu kommen:

- Kombination linearer Schieberegister mit nichtlinearer Output-Transformation,

- nichtlineare Kongruenzgeneratoren mit stark gestutztem Output.


vorhersagbar
linear
Ausgabe: c•n Bits
quadratisch
Ausgabe: n Bits
linear
Ausgabe: c•log(n) Bits
quadratisch
Ausgabe: c•n Bits
?
sicher
quadratisch
Ausgabe: c•log(n) Bits