Security

- LV 4121 und 4241 -

Schlüsselmittelherstellung

Kap. 7: Schlüsselmittelherstellung

Teil 1: Erzeugung von Zufallszahlen

- Zufallszahlengeneratoren
- Neumann-Filter

Erzeugung von Zufallszahlen

- Die **Sicherheit** aller kryptographischen Verfahren basiert auf der Schwierigkeit, einen geheimen Schlüsselparameter zu erraten oder anderweitig zu beschaffen.
- Im Zusammenhang mit kryptographischen Schlüsselparameter spielt das Erzeugen von **Zufallszahlen** (möglichst zufällig, hinreichend groß, besondere Eigenschaften etc.) eine zentrale Rolle.
- Ein **Pseudozufallszahlengenerator** ist ein Algorithmus, der nach Eingabe von gewissen Initialisierungsdaten (sogenannten **seed numbers**) eine Zufallsfolge deterministisch erzeugt.
- Einen solchen randomisierten Algorithmus, der in Form eines Simulations- oder Rechenprogramms lediglich eine pseudozufällige Bitfolge liefert, nennt man **pseudo random number generator (PRNG)**.

Erzeugung von echten Zufallszahlen

Der echte Zufallszahlengenerator:

Definition:

A **random bit generator** is a device that is designed to output a sequenz of statistically independent and symmetrically distributed **binary random variables**, i. e., that is designed to be the implementation of a so-called **binary symmetric source (BSS)**.

- Das Wissen der ersten n Bits einer zufälligen Folge liefert keine Information über das n + 1-te Bit.
- Eine gute Zufallsquelle stützt sich auf physikalische Zufallsereignisse wie zum Beispiel thermisches Rauschen oder radioaktiver Zerfall ab.
- Den zugehörigen Prozess nennt man **real random number generator (RRNG)**.

Nachbehandlung echter Zufallsfolgen:

Auch wenn die Zufallszahlen aus einem physikalischen Prozess stammen, muss untersucht werden, ob der zugrunde liegende physikalische Prozess **echt** zufällig ist und im Falle einer **statistisch unabhängigen** Zahlenfolge diese eine symmetrische Verteilung bezüglich der Werte „0“ und „1“ aufweist.

Der Neumann-Filter:

Der Informatikpionier **John von Neumann** schlug 1951 eine sehr effiktive Funktion **f** zur Beseitigung der Asymmetrie in einer Bitfolge vor:

$$\mathbf{f} : \{0,1\}^m \to \{0,1\}^n \text{ mit } 00 \to \varepsilon,\ 11 \to \varepsilon,\ 01 \to 0,\ 10 \to 1,$$

wobei sich **f** auf zwei aufeinanderfolgende Bits (nicht überlappende Bit-Paare) bezieht und ε für die leere Zeichenkette steht.

Eigenschaften nach Anwendung des Neumann-Filters:

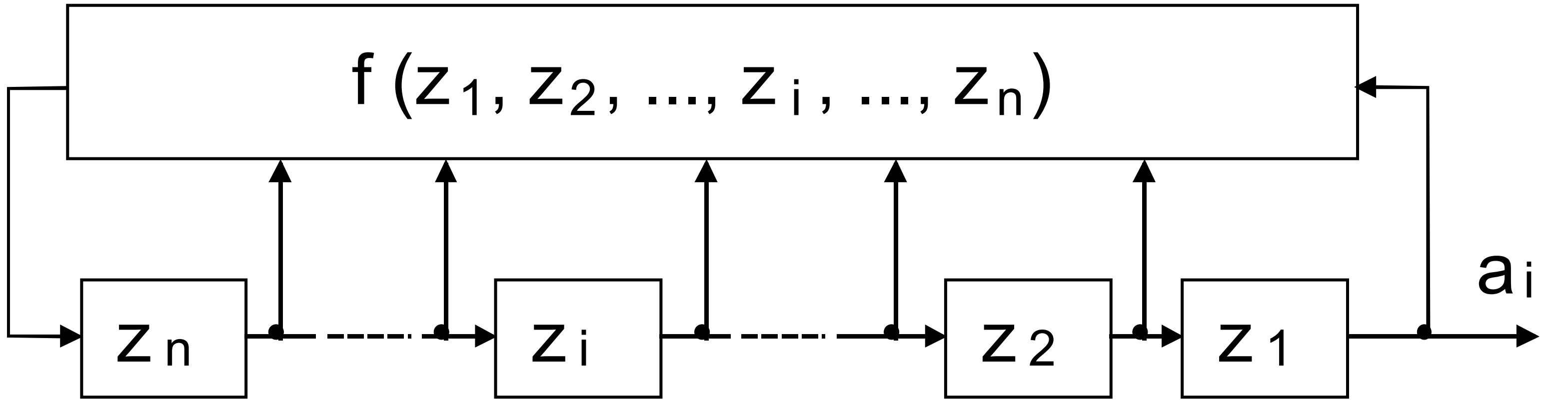
- Wenn in einer Bitfolge **$a_i \rightarrow \{0,1\}^n$** aufeinanderfolgende Bits **statistisch unabhängig** sind und den Wert „**1**" mit der Wahrscheinlichkeit **p** annehmen, so verkürzt sich die Länge der Bit-Folge durch die Filterung um den Faktor **p (1 – p)**.
- Im Falle **p = 1/2** gehen dann etwa **3/4** aller ursprünglichen Bits verloren und für alle anderen Werte von **p** ist der **Verlust** noch höher (dies ist der Preis für die Verbesserung der Zufälligkeit).
- Da die Wahrscheinlichkeit für ein Paar „**01**" bzw. „**10**" in der ursprünglichen Bitfolge gleich **p (1 – p)** ist, ergibt sich für die Wahrscheinlichkeit **p_0** und **p_1** für den Wert **1** bzw. **0** nach der Filterung der Wert **1/2**.

Kap. : Schlüsselmittelherstellung

Teil 2: Pseudozufallszahlengeneratoren

- Lineare Schieberegister mit Rückkopplung
- Primzahlhäufigkeit und Primzahldichtefunktion

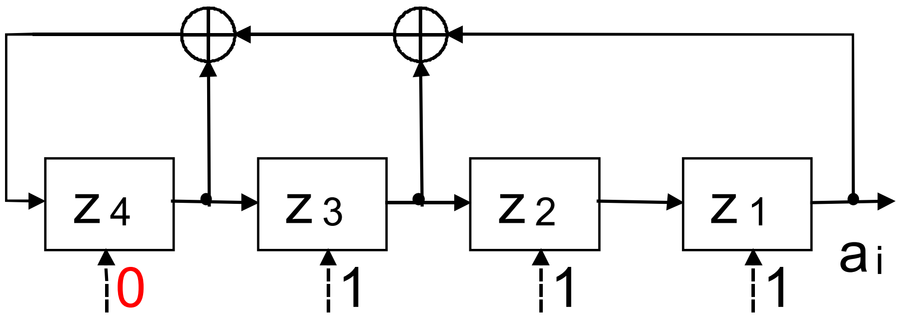
Prinzip des Linear Feedback Shift Register (LFSR)



- In den Zellen z_1 bis z_n des n-stufigen LFSR können die Binärwerte 0 oder 1 gespeichert werden.
- Bei jedem Berechnungsschritt werden die Inhalte der Zellen z_n bis z_2 nach rechts geschoben.

- Der Zelleninhalt z_n wird dabei durch den Wert der binärwertigen Funktion $f(z_1, z_2, ..., z_i, ..., z_n)$ ersetzt.
- Der Zelleninhalt z_1 geht verloren und kann als binäre Pseudozufalls-ziffer a_i betrachtet werden.
- Damit die maximale Periodenlänge erreicht wird, wird bei LFSR die Rückkopplung durch speziell ausgewählte Zelleninhalte realisiert.
- Die Verknüpfung der rückgekoppelten Zelleninhalte geschieht durch XOR-Bildung bzw. Addition modulo 2.
- Liegt bei einem n-stufigen LFSR eine Ausgabefolge von 2n Bit vor, so lässt sich das Rückkopplungsnetzwerk rekonstruieren.
- Das Finden eines n-stufigen LFSR mit **maximaler Periode** lässt sich zurückführen auf das Finden eines primitiven Polynoms vom Grad n.

Beispiel:



→ Lineare Transformation

$$T = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

LFSR:

- 1. Spalte von T bestimmt die Positionen der Rückkopplung (hier 4, 3 und 1).
- Restliche Spalten beschreiben Verschiebung um eine Position nach rechts (Einheitsmatrix!).

Startvektor: x = (0, 1, 1, 1) ⇒

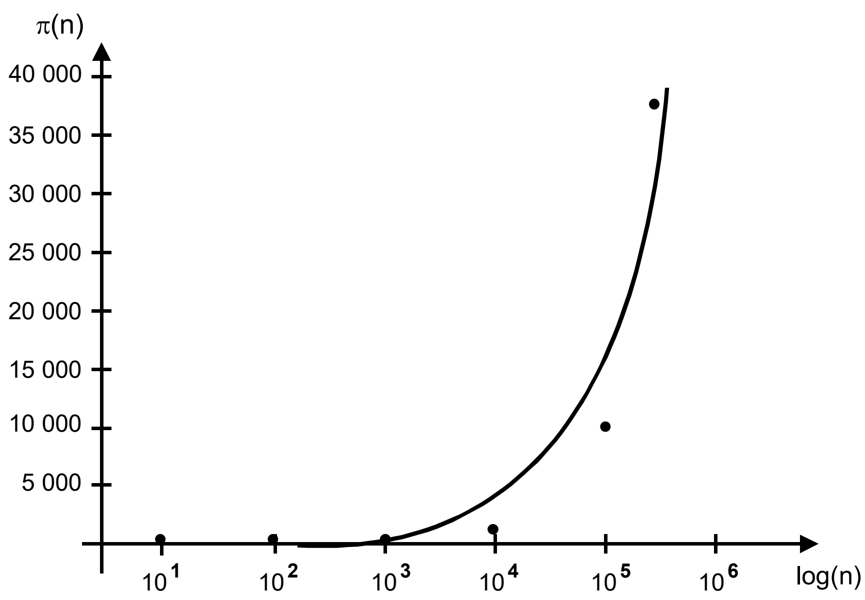
Folgevektoren: (0, 0, 1, 1), (1, 0, 0, 1), (0, 1, 0, 0), (1, 0, 1, 0), (1, 1, 0, 1), (1, 1, 1, 0), (0, 1, 1, 1), (0, 0, 1, 1), ...

⇒ a_i = {1, 1, 1, 0, 0, 1, 0, ...}, d = 7.

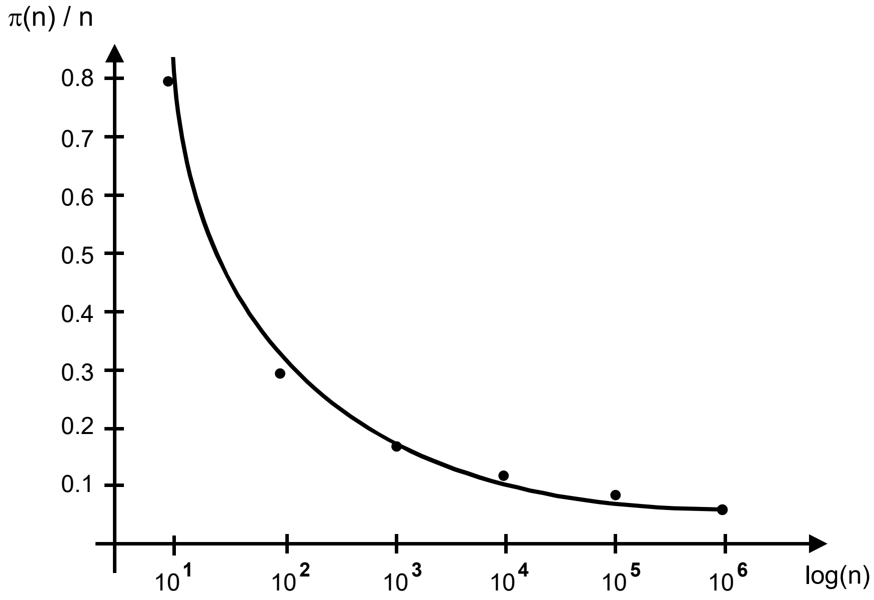
Diskussuion des Beispiels:

- Die Zustandsmenge **X** eines LFSR der Länge n lässt sich durch 0-1-Vekoren x der Form $x = (b_n, b_{n-1}, ..., b_2, b_1)$ darstellen.
- Die Funktion f kann als linaere Tranformation $f(x) = x \cdot T$ aufgefasst werden, wobei T ist eine binäre nxn-Matrix ist.
- Alle anfallenden Operationen werden modulo 2 ausgeführt – dies entspricht einer binären XOR-Verknüpfung.
- Bezeichnet x den Initialvektor des LFSR, so wird die Zustandsfolge x, $x \cdot T$, $x \cdot T^2$, $x \cdot T^3$, ... generiert.
- Die maximal erreichbare Periodenlänge beträgt $d = 2^n – 1$.

Primzahlhäufigkeit



Primzahldichtefunktion



$$\pi(n) = n \,/\, (\ln(n) - a_0) \quad \text{mit} \quad a_0 = 1.08366$$