Поточные шифры

Сергей Николенко

Криптография — АУ РАН, осень 2011

Outline

Одноразовый блокнот

- Теорема Шеннона: пусть K и M — случайные величины, соответствующие ключам и сообщениям. Чтобы система с закрытым ключом была безусловно надёжной, необходимо, чтобы $H(K) \geq H(M)$.
- В частности, если $|k| \geq |m|$, и ключи берутся из равномерного распределения, то это условие всегда выполнено.
- Но для этого нужно использовать ключ размером с сообщение и сразу его выбрасывать, что обычно невозможно.

Поточный шифр как одноразовый блокнот

- Но идея хороша: давайте действительно генерировать новый шифр поточным образом.
- Только он у нас будет не совсем случайный, а *псевдослучайный*.

Синхронные шифры

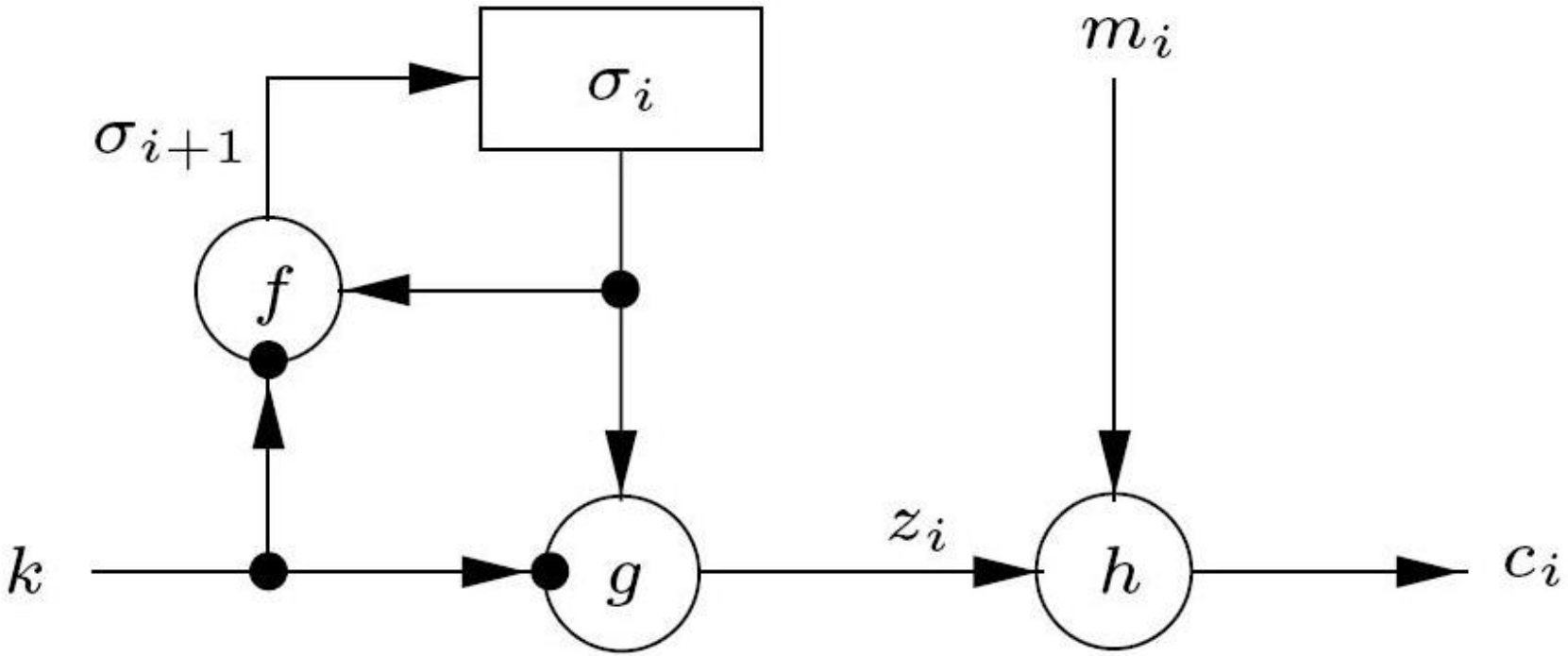
- В *синхронных* поточных шифрах поток ключа генерируется независимо от сообщения и кода.
- Кодирование синхронных шифров можно описать как

$$\begin{aligned} \sigma_{i+1} &= f(\sigma_i, k), \\ z_i &= g(\sigma_i, k), \\ c_i &= h_i(z_i, m_i). \end{aligned}$$

- σ_0 — начальное состояние, k — ключ, f — функция перехода между состояниями, g производит поток ключей z, а код c получается из этого потока и сообщения.

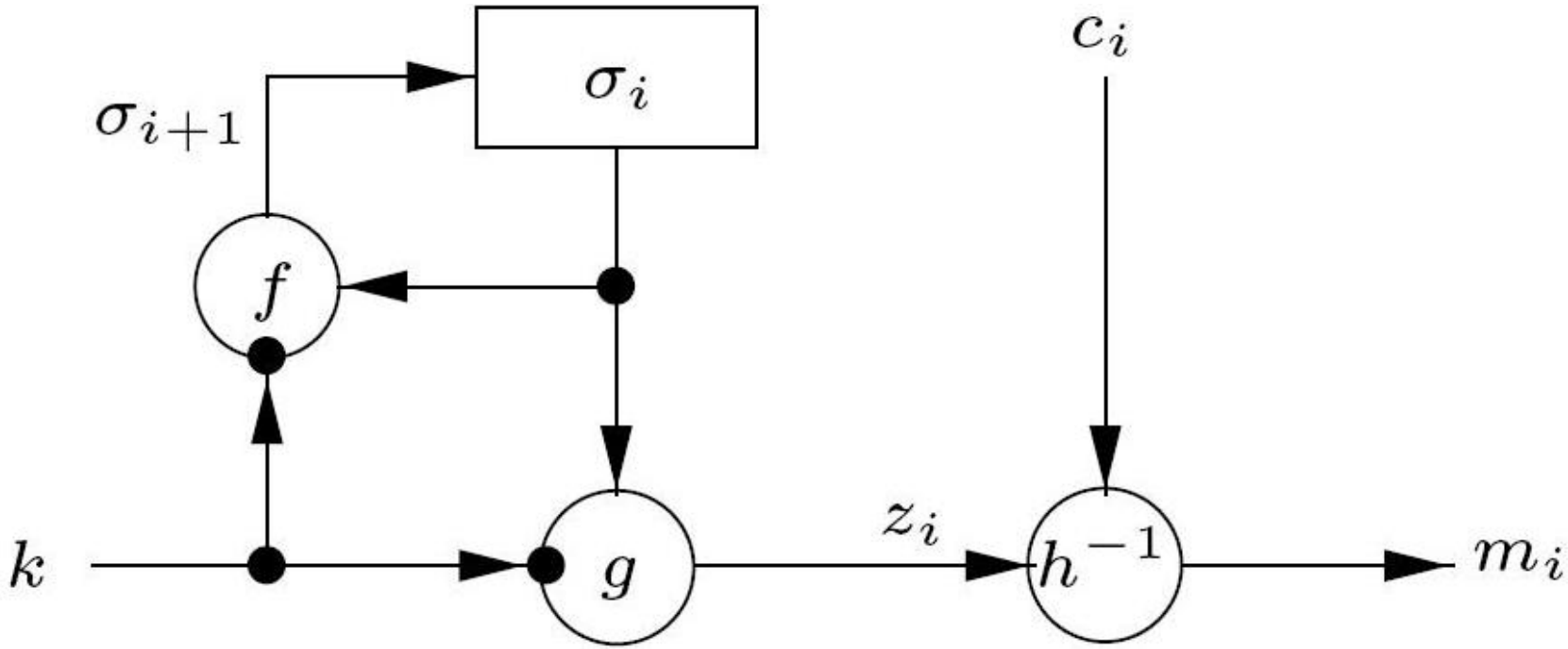
Синхронные шифры

- В *синхронных* поточных шифрах поток ключа генерируется независимо от сообщения и кода.
- Кодирование:



Синхронные шифры

- В *синхронных* поточных шифрах поток ключа генерируется независимо от сообщения и кода.
- Декодирование:



Синхронные шифры

- В *синхронных* поточных шифрах поток ключа генерируется независимо от сообщения и кода.
- Обычно рассматривают *линейные бинарные поточные шифры*.
- В них m_i и c_i — биты, и $h_i(z_i, m_i) = m_i \oplus z_i$.

Свойства синхронных шифров

- Нужно синхронизировать. Если вдруг в канале могут пропадать биты или появляться новые, нужны специальные методы синхронизации.
- Ошибка не распространяется дальше.
- Относительно атак:
  - из-за первого свойства атаки, связанные с новыми символами или удалением, легче заметить;
  - из-за второго свойства, если взломщик может менять код, он будет знать, как это повлияет на сообщение.

Самосинхронизирующиеся шифры

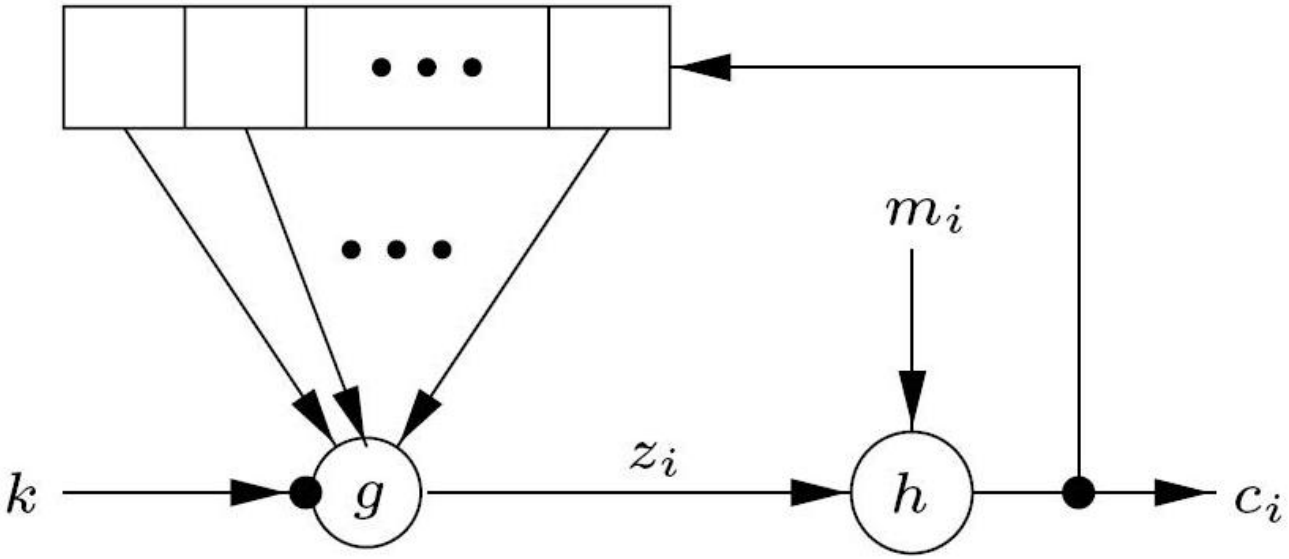
- В *самосинхронизирующихся* поточных шифрах поток ключа — функция сообщения и нескольких предшествующих битов кода.
- Кодирование самосинхронизирующихся шифров можно описать как

$$\begin{aligned}\sigma_i &= (c_{i-t}, c_{i-t+1}, \ldots, c_{i-1}),\\ z_i &= g(\sigma_i, k),\\ c_i &= h_i(z_i, m_i).\end{aligned}$$

- σ_0 — начальное состояние, k — ключ, f — функция перехода между состояниями, g производит поток ключей z, а код c получается из этого потока и сообщения.

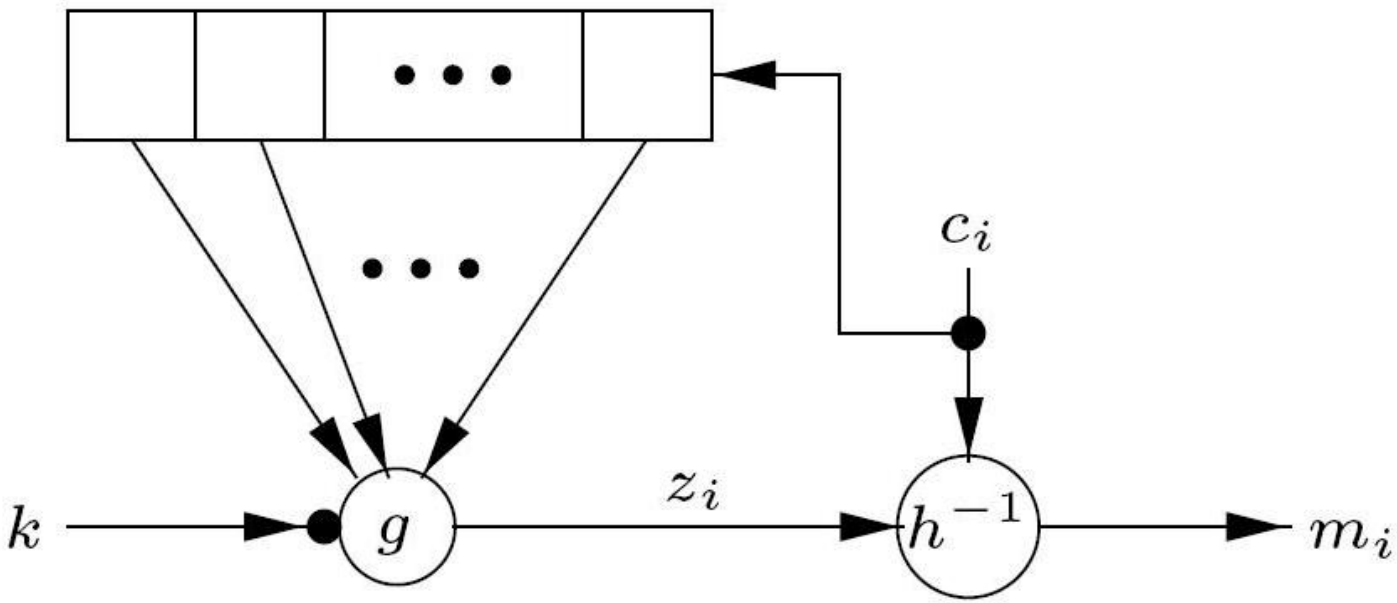
Самосинхронизирующиеся шифры

- В *самосинхронизирующихся* поточных шифрах поток ключа — функция сообщения и нескольких предшествующих битов кода.
- Кодирование:



Самосинхронизирующиеся шифры

- В *самосинхронизирующихся* поточных шифрах поток ключа — функция сообщения и нескольких предшествующих битов кода.
- Декодирование:



Свойства самосинхронизирующихся шифров

- Синхронизируются сами: каждая ошибка влияет только на конечное число последующих битов.
- Ошибка распространяется, но ограниченно.
- Относительно атак:
  - из-за первого свойства атаки, связанные с новыми символами или удалением, труднее заметить;
  - из-за второго свойства, если взломщик изменил код, это повлияет на следующие биты, что может позволить заметить вторжение.

Целеполагание

- Мы хотим получить надёжный поточный шифр.
- Для этого нам нужно получить надёжную последовательность ключей z_i, которая потом XOR'ится с сообщением.
- Что значит — «надёжная последовательность»?

Целеполагание

- Надёжная — значит, похожая на случайную, раз уж совсем случайной не будет.
- Иначе говоря, поточный шифр должен быть неплохим *псевдослучайным генератором*.
- Но заметим, что если наша булева схема, вычисляющая z_i, конечна, то последовательность будет периодической.
- Какими свойствами должна обладать периодическая последовательность, чтобы быть похожей на случайную?

Постулаты Голомба

- Соломон Голомб предложил следующие *необходимые* условия для N-периодичной последовательности.
  1. В цикле длины N число единиц отличается от числа нулей не более чем на 1.
  2. В цикле длины N не менее половины последовательностей одинаковых символов имеют длину 1, не менее четверти — длину 2, не менее $\frac{1}{8}$ — длину 3 и т.д. Для каждой длины последовательностей из нулей (почти) столько же, сколько из единиц.
  3. Автокорреляция принимает ровно два значения: $\exists K \in \mathbb{N}$

$$C(t) = \frac{1}{N}\sum_{i=0}^{N-1}(2s_i - 1)(2s_{i+t} - 1) = \begin{cases} 1, & \text{если } t = 0, \\ \frac{K}{N}, & \text{если } 1 \leq t \leq N-1. \end{cases}$$

- Последовательность, удовлетворяющая постулатам Голомба, называется *псевдошумной* (pn-последовательностью).

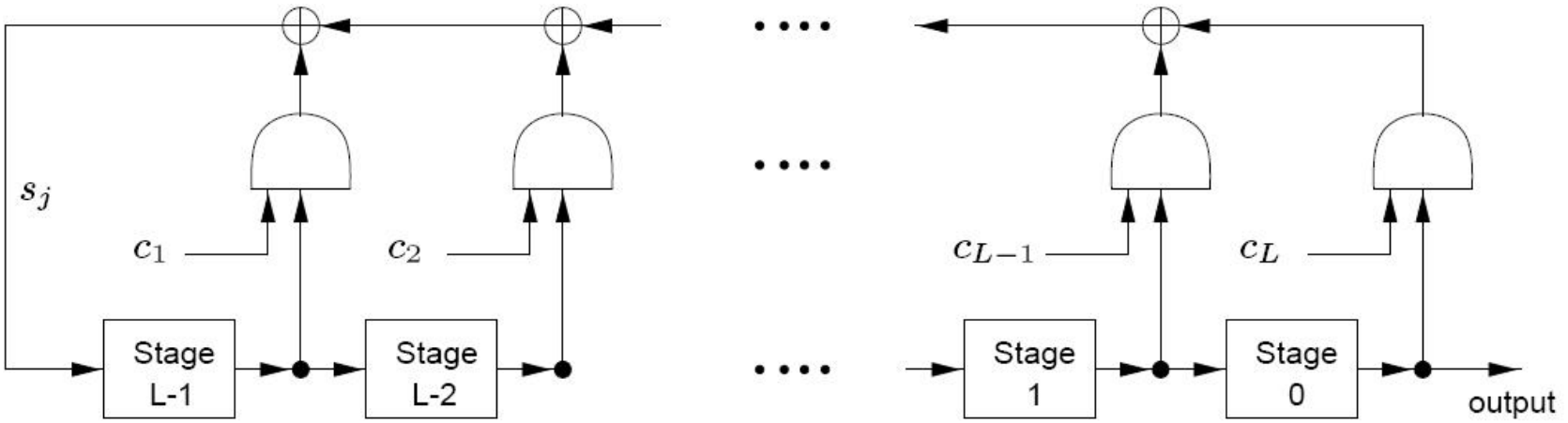
Статистические тесты

- Можно запускать статистические тесты, которые проверяют полезные гипотезы:
  - (monobit test) равно ли количество нулей и единиц;
  - (two-bit test) равно ли количество 00, 01, 10 и 11;
  - (poker test) равно ли количество разных последовательностей длины m;
  - (runs test) подходящее ли количество последовательностей идущих подряд нулей и единиц той или иной длины;
  - (autocorrelation test) одинаковая ли автокорреляция на разных сдвигах;
  - (Maurer test) хорошо ли получается сжать получившийся поток.
- Хороший генератор должен проходить все эти тесты.

Outline

Определение

- *Линейный регистр сдвига с обратной связью* (linear feedback shift register, LFSR) длины L состоит из L уровней, $S_0, \ldots, S_{L-1}$ каждый из которых хранит 1 бит, и счётчика времени. На каждом шаге содержимое S_0 выдаётся на выход, содержимое S_i передаётся на S_{i-1}, а в S_{L-1} поступает *бит обратной связи*, вычисленный как XOR некоторого фиксированного подмножества $S_0, \ldots, S_{L-1}$.



Ассоциированный многочлен

- LFSR обозначается как $\langle L, C(D)\rangle$, где $C(D) = 1 + c_1 D + \ldots + c_L D^L \in \mathbb{Z}_2[D]$ — его *характеристический многочлен*.
- LFSR *несингулярен*, если $c_l = 1$ (т.е. $\deg C(D) = L$). Будем предполагать несингулярность.
- LFSR производит периодическую последовательность.

Упражнение. Доказать, что период генерируемой последовательности равен наименьшему числу N, при котором многочлен $C(D)$ делит $1 + D^N$.

Ассоциированный многочлен

- Следовательно, LFSR выдаёт максимально возможный период $2^L - 1$ тогда и только тогда, когда $C(D)$ примитивен.
- (примитивный многочлен — минимальный многочлен для примитивного элемента соответствующего расширения $\mathbb{Z}_2$; например, примитивный многочлен степени 2 только один: $1 + x + x^2$; степени 3 – два многочлена: $1 + x + x^3$ и $1 + x^2 + x^3$; если не помните, ничего страшного, пользоваться сильно не будем)

Ассоциированный многочлен

- Пример: рассмотрим многочлен $C(D) = 1 + D + D^3$. Это значит, что

$$z_j = z_{j-3} \oplus z_{j-1}.$$

- Если начальное состояние $\sigma_0 = 001$, то дальше будет:

001	011
100	101
110	010
111	001

- Период $2^3 - 1 = 7$, как и обещали. На выход поступит 10011101.

Свойства LFSR

- Главное свойство — выход LFSR удовлетворяет постулатам Голомба.
- В частности, последовательностей подряд идущих нулей/единиц за период ровно 2^{L-1}, из них ровно половина — длины 1, ровно четверть — длины 2 и т.д., длины $L-1$ и длины L — по одной.
- Например, в примере выше за период получилось 1001110; $4 = 2^2$ последовательности, свойства все здесь.

Линейная сложность

- Мы выяснили, что LFSR — хорошие порождатели псевдослучайных последовательностей.
- Можно посмотреть и наоборот. LFSR *порождает* последовательность s, если для некоторого начального состояния он выдаёт s на выход.
- *Линейная сложность* (linear complexity) $L(s)$ последовательности s определяется так:
  - если $s = 00000\ldots$, то $L(s) = 0$;
  - если не существует LFSR, порождающего s, то $L(s) = \infty$;
  - иначе $L(s)$ — длина самого короткого LFSR, порождающего s.
- Для конечных — длина самого короткого LFSR, порождающего последовательность, начинающуюся с данной.

О линейной сложности

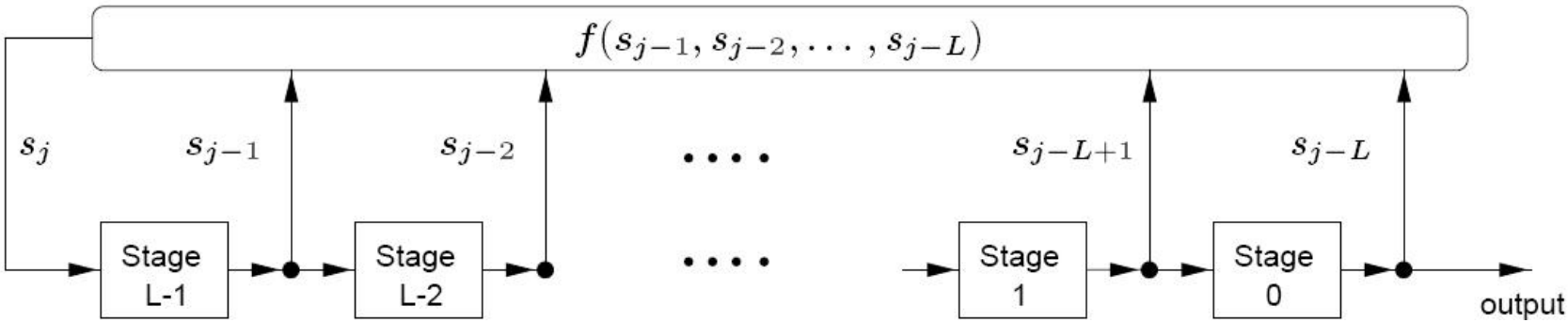
- О линейной сложности многое известно; она по сути похожа на колмогоровскую сложность или сложность Лемпеля-Зива: насколько сложный объект может породить эту последовательность.
- Но, в отличие от колмогоровской сложности, линейную легко вычислить.

Немного математики

1. Алгоритм Евклида над многочленами.
2. Реконструкция рациональных функций.
3. Реконструкция по ряду Лорана, коды Рида-Соломона.
4. Линейно порождённые последовательности, их реконструкция.

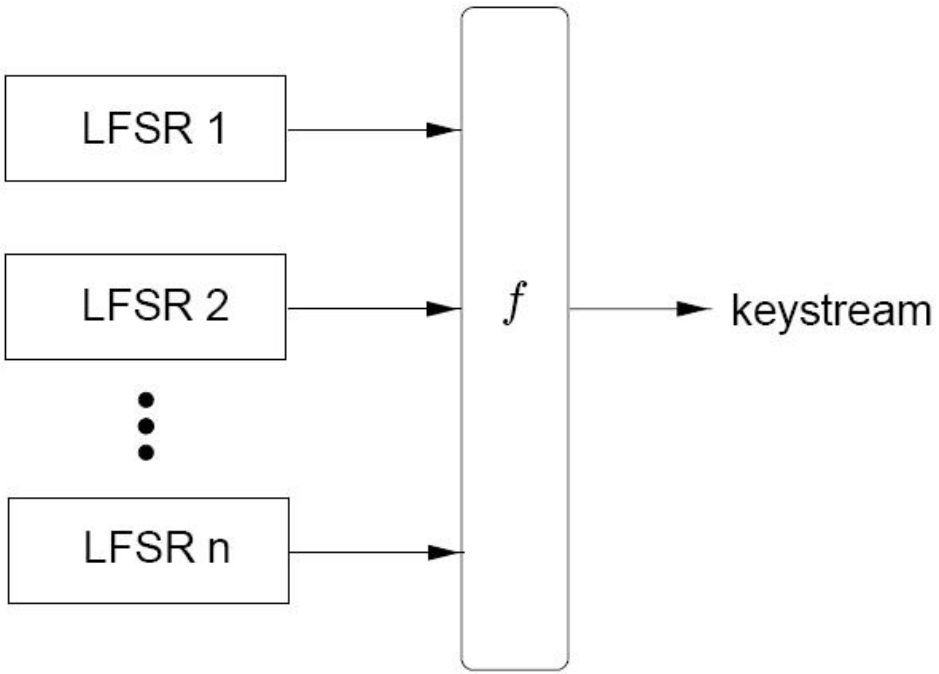
Нелинейные регистры сдвига

- Линейные хороши, но для криптографии сами по себе непригодны; вот, например, простой алгоритм определяет устройство LFSR по данным длиной всего $2L$.
- Нужно где-то ввести нелинейность.
- Нелинейный регистр сдвига:



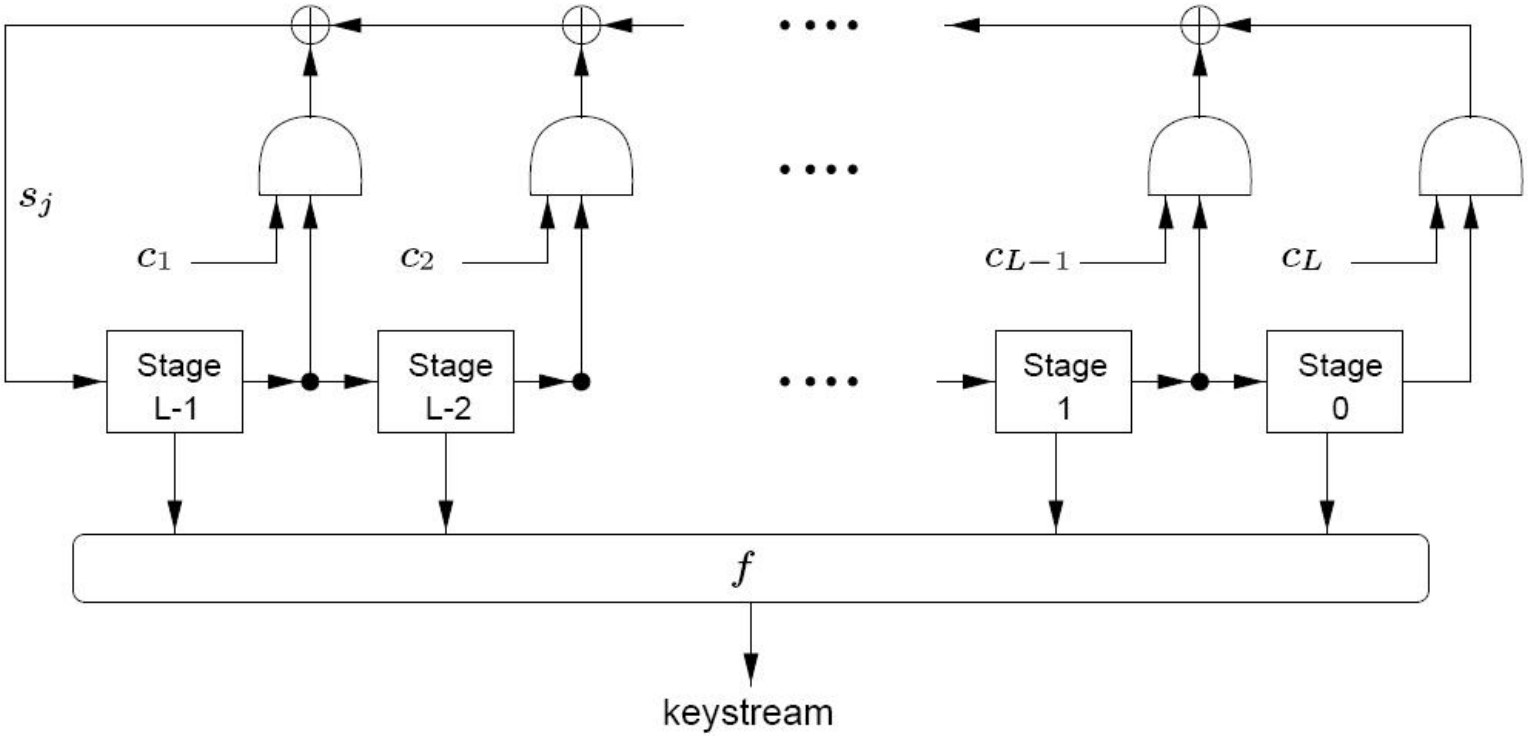
Нелинейные комбинации LFSR

- Можно объединить результаты нескольких LFSR нелинейной функцией:



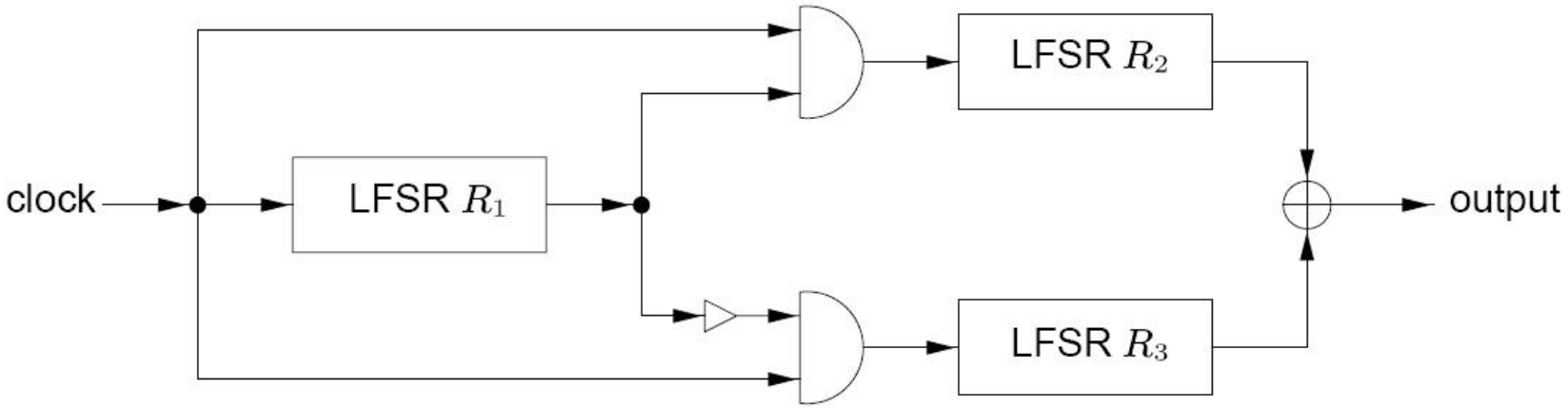
Нелинейные комбинации LFSR

- Можно генерировать поток как нелинейную функцию (фильтр) от состояний:



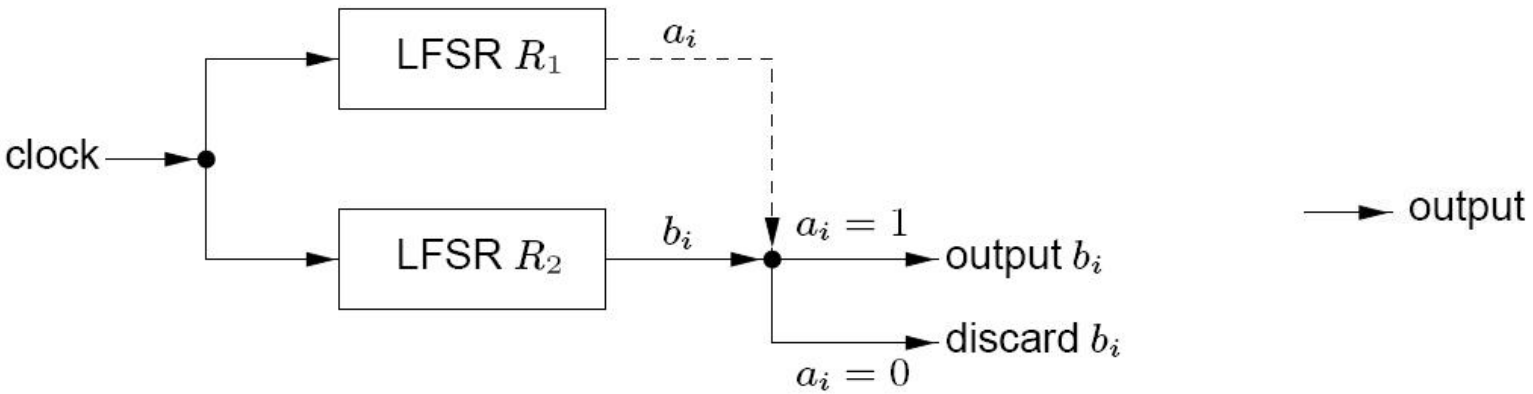
Нелинейные комбинации LFSR

- Можно использовать один LFSR как «часы» для других:



Нелинейные комбинации LFSR

- The shrinking generator:



Спасибо за внимание!

- Lecture notes и слайды будут появляться на моей homepage:

  `http://logic.pdmi.ras.ru/~sergey/`

- Присылайте любые замечания, решения упражнений, новые численные примеры и прочее по адресам: `sergey@logic.pdmi.ras.ru`, `snikolenko@gmail.com`.