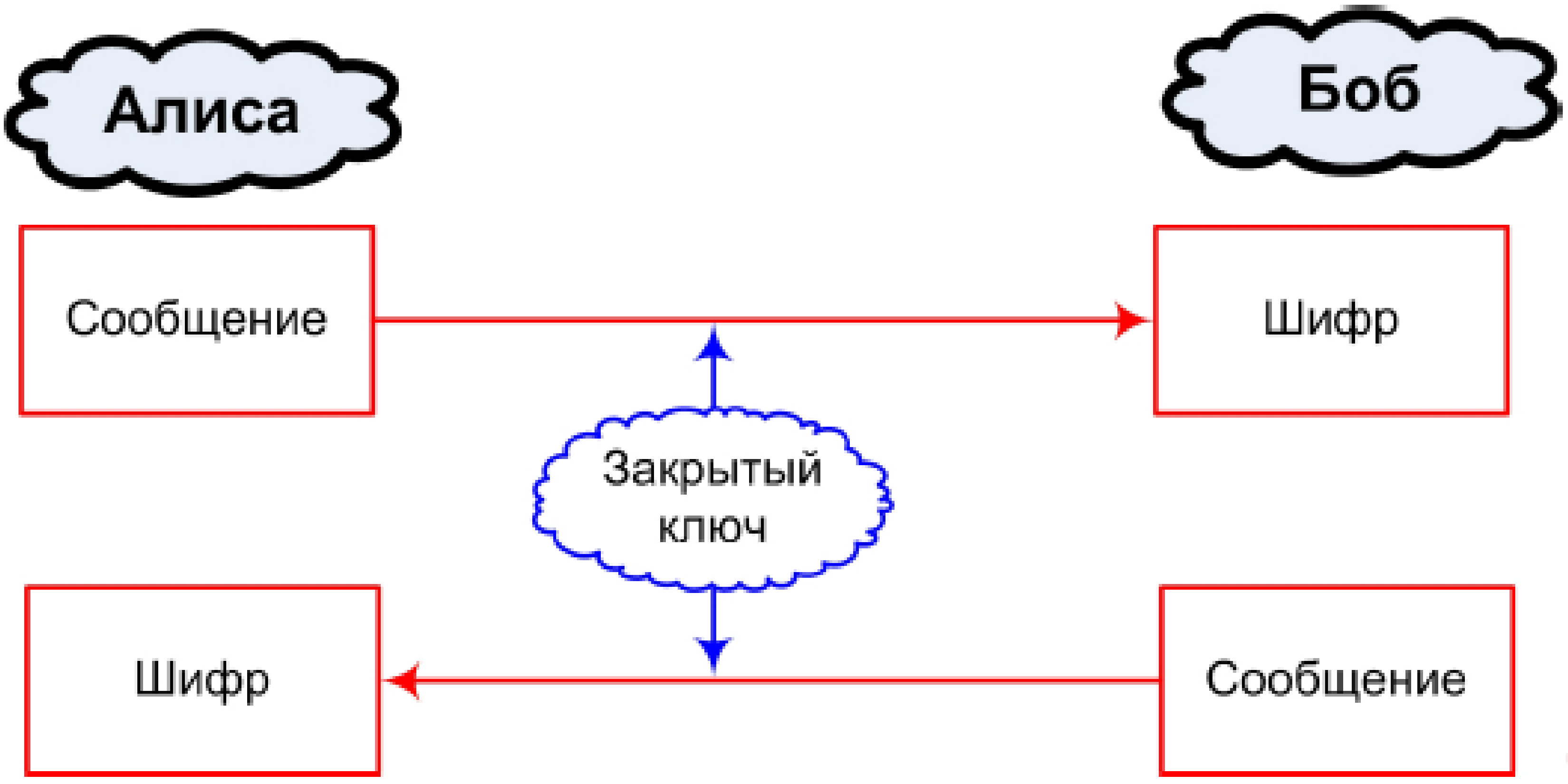
СИММЕТРИЧНЫЕ КРИПТОСИСТЕМЫ

Потоковые шифры

Симметричные криптоситемы



Формальное определение

- Симметричная криптосистема над $\{\mathcal{M}, \mathcal{K}, \mathcal{C}\}$ -- это пара эффективных алгоритмов **E** и **D** **:**

- **E:** $\mathcal{M}$ x $\mathcal{K}$ -> $\mathcal{C}$ **D:** $\mathcal{C}$ x $\mathcal{K}$ -> $\mathcal{M}$
- $\forall\ m \in \mathcal{M},\ k \in \mathcal{K}$: **D**(k, **E**(m,k))=m
- **E** – может быть рандомизирован
- **D** – строго детерминирован

Одноразовый блокнот

- 1917 год Гилберт Вернам
- **E**$(m,k) = m \oplus k$
- **D**$(c,k) = c \oplus k$

msg:	0	1	1	0	1	1	1	$\oplus$
key:	1	0	1	1	0	1	0	
CT:								

- $\mathcal{M} = \mathcal{C} = \mathcal{K} = \{0,1\}^n$

Вопрос?

- Если Вам дана пара: открытый текст (*m*) и зашифрованное сообщение (*c*), возможно ли нахождения одноразового ключа (k)?
  - Только половину *k*
  - Зависит от значений *m* и *c*
  - Да
  - Нет

Одноразовый блокнот

- Достоинства?
- Недостатки?
- Безопасность?

Что такое безопасность шифра?

- Возможности атакующего:
  - Атака по известному шифротексту(*c*)
- Требования безопасности:
  - Атакующий не может найти ключ
  - Атакующий не может раскрыть часть сообщения
  - К. Шеннон «Теория связи в секретных системах»(1949): Никакая информация о m не может быть раскрыта

Теоретико-информационная стойкость

- Def: Криптосистема (**E**, **D**) над набором $\{\mathcal{M}, \mathcal{K}, \mathcal{C}\}$ называется совершенно стойкой (**perfect secrecy**), если для любой пары исходных сообщений $m_0, m_1 \in \mathcal{M}$ равной длины и произвольного $c \in \mathcal{C}$ верно

  - $Pr\{\mathbf{E}(k, m_0) = \mathrm{c}\} = Pr\{\mathbf{E}(k, m_1) = \mathrm{c}\}$
  - $k \overset{R}{\leftarrow} \mathcal{K}$

- Лемма: Одноразовый блокнот – совершенно стойкая система.

- Доказательство:

  - Чему равна вероятность $Pr\{\mathbf{E}(k, m \quad) = \mathrm{c}\}$ для пары (m, c)?

  - Сколько k : $\mathbf{E}(k, m \quad) = \mathrm{c}$?

Основной недостаток

- Теорема Шеннона: пусть K и M случайные величины, соответствующие ключам и сообщениям. Чтобы система с закрытым ключом была безусловно надежной, 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

- Идея для нового подхода: давайте действительно генерировать новый шифр поточным образом.
- Только он у нас будет не совсем случайный, а **псевдослучайный**.

Потоковый шифр

- G(k)
- Шифрование:

$$\mathbf{E}(k,m) = m \oplus G(k) = c$$

- Дешифрование:

$$\boldsymbol{D}(k,c) = c \oplus G(k) = m$$

- Может ли потоковый шифр быть совершенно надежным?
  - Да, благодаря особым свойствам G(k)
  - Нет, не существует совершенно надежных систем
  - Нет, пока ключ короче сообщения

Новое определение безопасности

- G(k) – должна быть непредсказуемой

- Def: Будем называть G(k)-> $\{0,1\}^n$ предсказуемой, если

$\exists$ эффект. алгоритм A :

- $Pr\{A(G(k))|_{1,2,\ldots i} = G(k)|_{i+1}\} = \frac{1}{2} + \varepsilon$
- ε – пренебрежимо малая величина

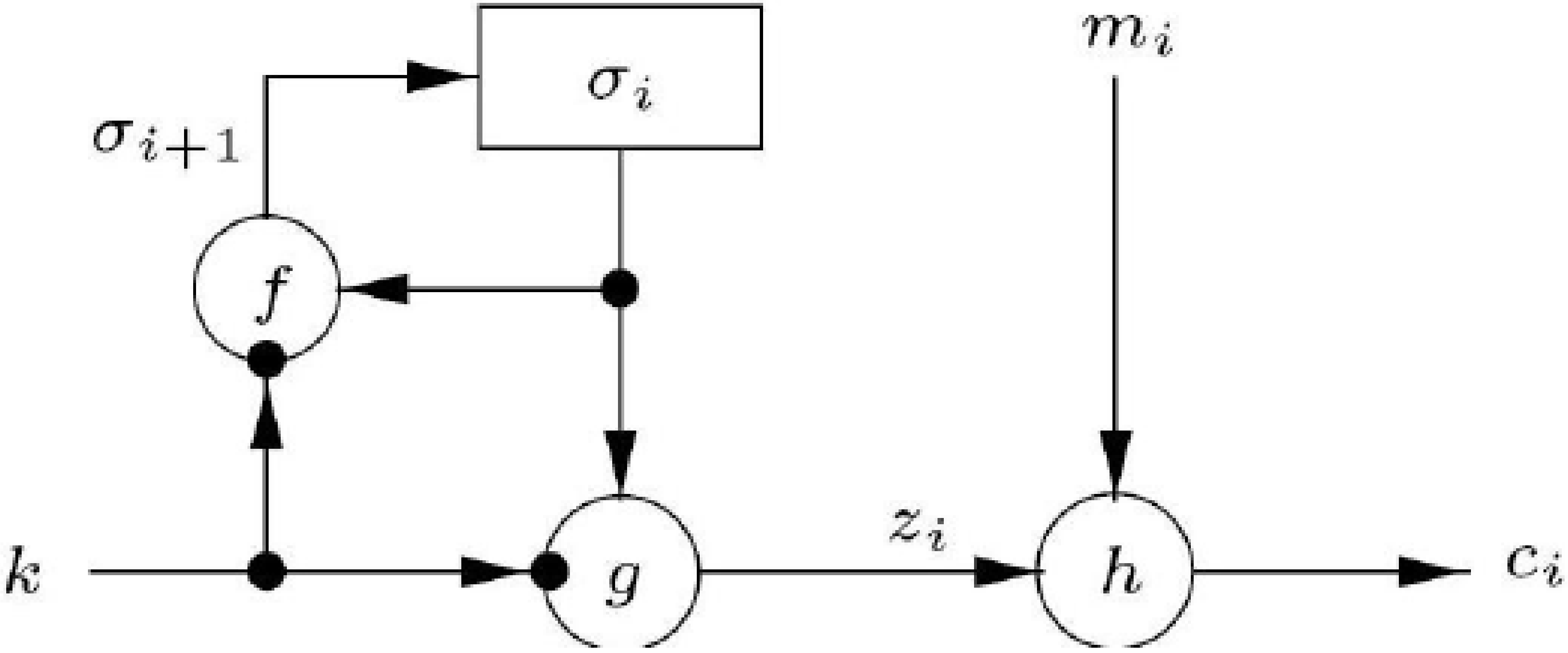
Синхронные шифры

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

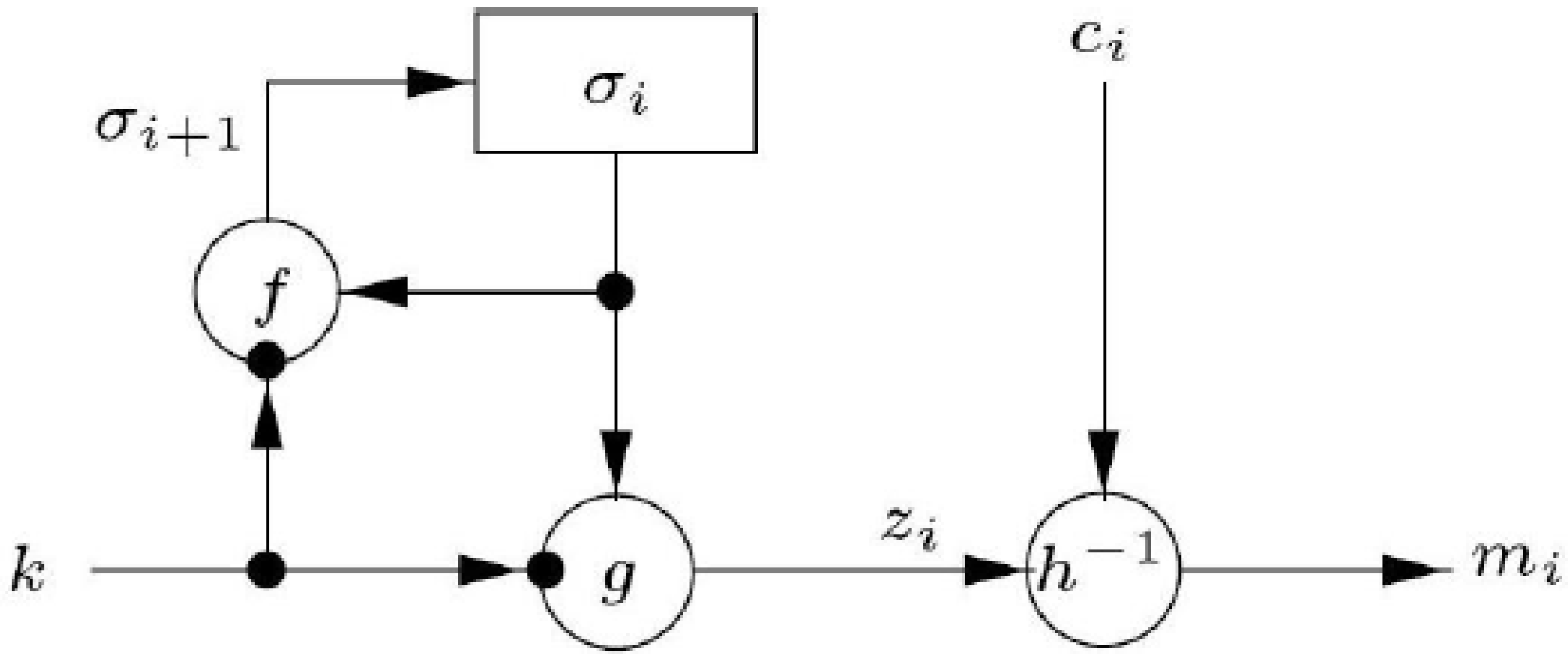
$$
\begin{aligned}
\sigma_{i+1} &= f(\sigma_i, k), \\
z_i &= g(\sigma_i, k), \\
c_i &= h_i(z_i, m_i).
\end{aligned}
$$

- 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.

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



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



- 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 $h_i(z_i, m_i) = m_i \oplus z_i$.

- 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.

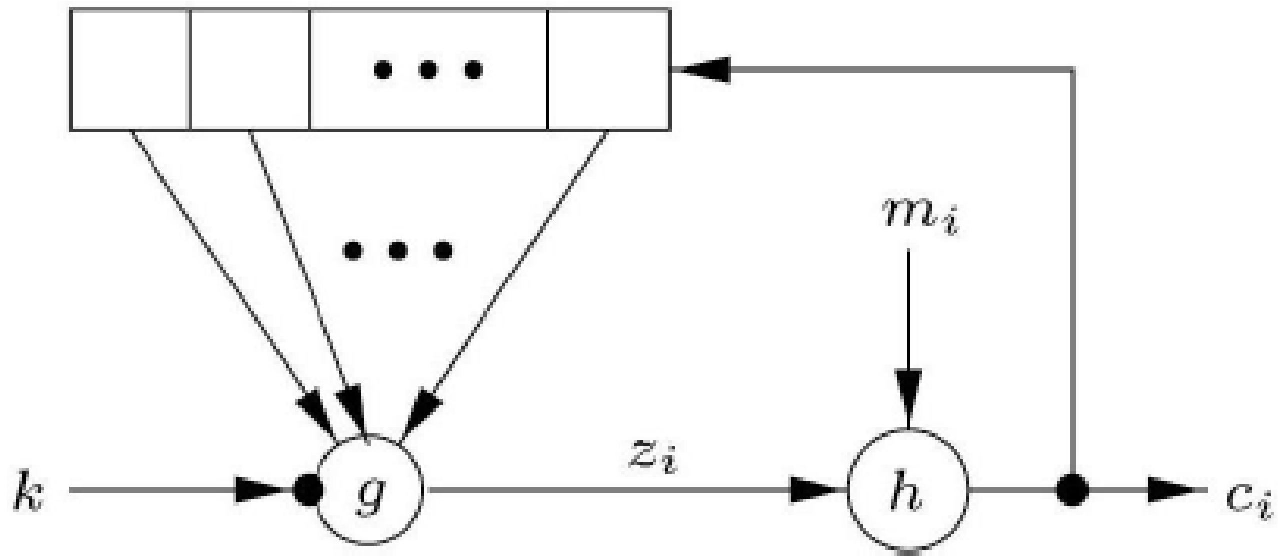
Самосинхронизирующиеся шифры

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

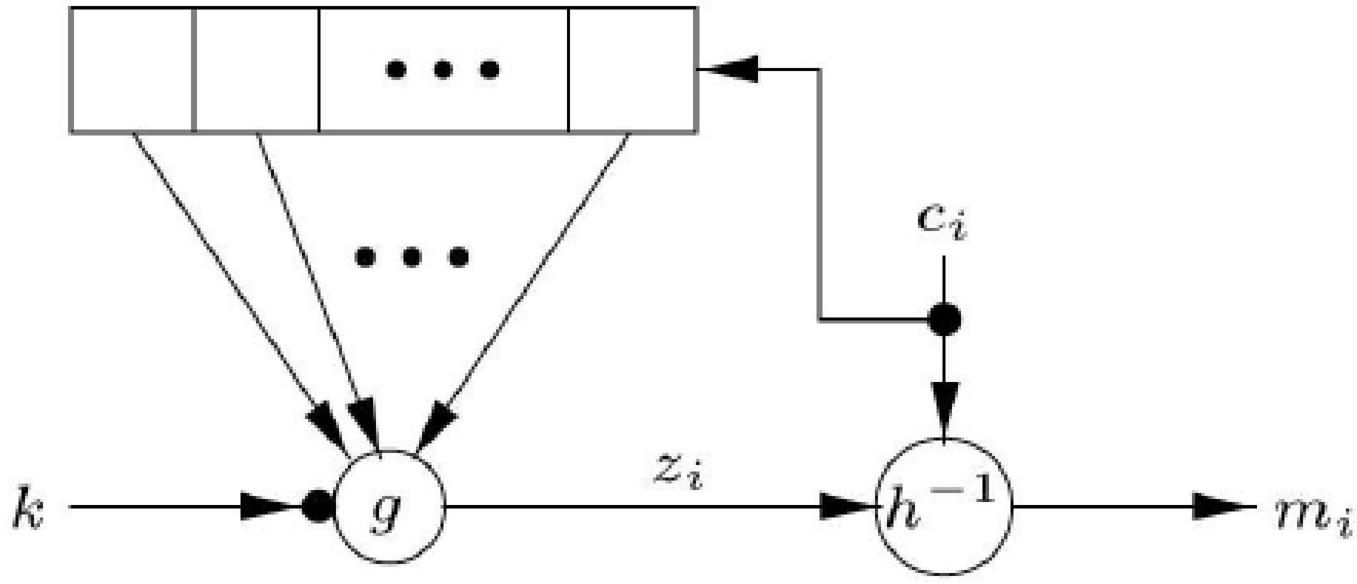
$$
\begin{aligned}
\sigma_i &= (c_{i-t}, c_{i-t+1}, \ldots, c_{i-1}),\\
z_i &= g(\sigma_i, k),\\
c_i &= h_i(z_i, m_i).
\end{aligned}
$$

- 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.

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



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



- 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.

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: $\exists K \in \mathbb{N}$

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

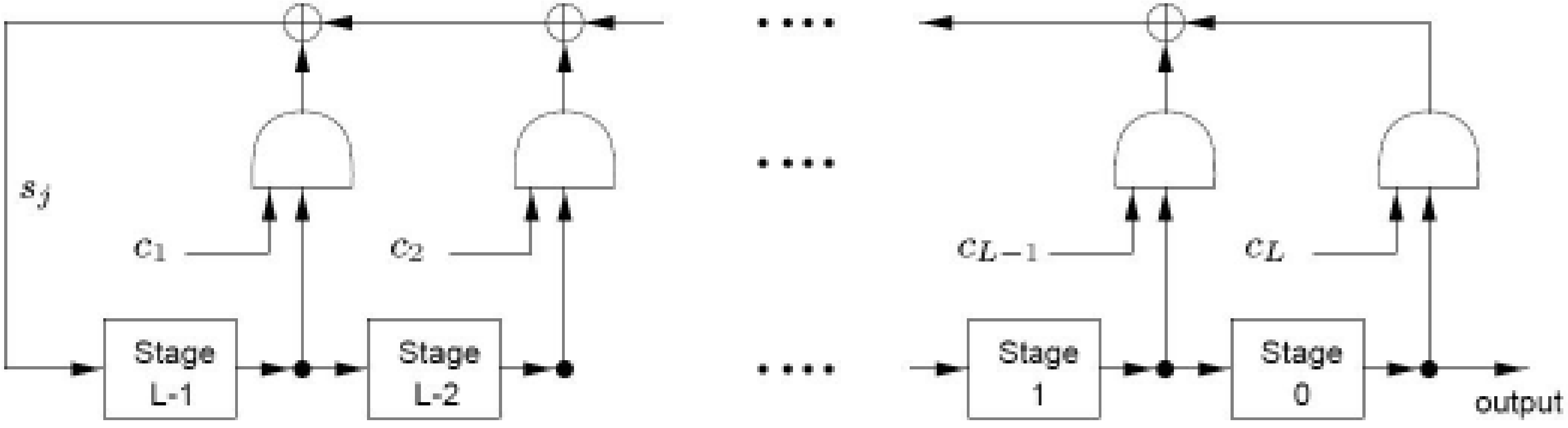
- Последовательность, удовлетворяющая постулатам Голомба, называется *псевдошумной*

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.

Построение генераторов

- *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.*
- 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 $\mathbb{Z}_2$;

Пример

- 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:
$$\begin{array}{cc} 001 & 011 \\ 100 & 101 \\ 110 & 010 \\ 111 & 001 \end{array}$$
- 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.

Алгоритм Берликемпа-Месси

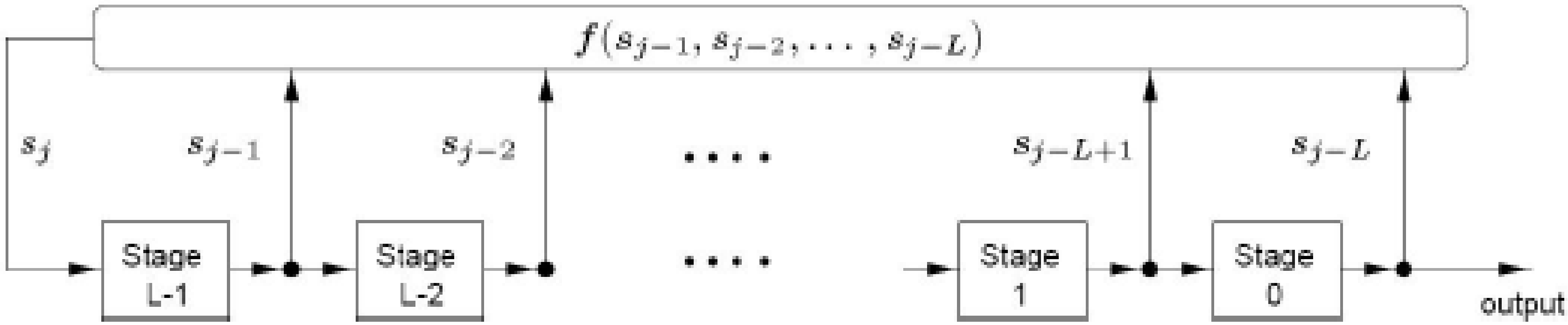
- Рассмотрим последовательность s длины $n+1$: $s^{n+1} = s_0 s_1 \ldots s_{n-1} s_n$.
- Пусть $\langle L, C(D) \rangle$, $C(D) = 1 + c_1 D + \ldots + c_L D^L$, порождает $s^n = s_0 s_1 \ldots s_{n-1}$.
- Рассмотрим разницу

$$d_n = s_n \oplus \sum_{i=1}^{L} c_i s_{N-i}.$$

- Если $d_n = 0$, всё хорошо, берём $L(s^{n+1}) = L$.
- Если $d_n = 1$, рассмотрим предыдущий LFSR, который отличался, т.е. максимальное такое $m < n$, что $L(s^m) < L(s^n)$. Пусть это был $\langle L(s^m), B(D) \rangle$.
- Теперь, если $L > n/2$, то $L' = L$, а если $L \leq n/2$, то $L' = n + 1 - L$.
- И результат — $\langle L', C'(D) \rangle$, $C'(D) = C(D) + B(D) \cdot D^{n-m}$.

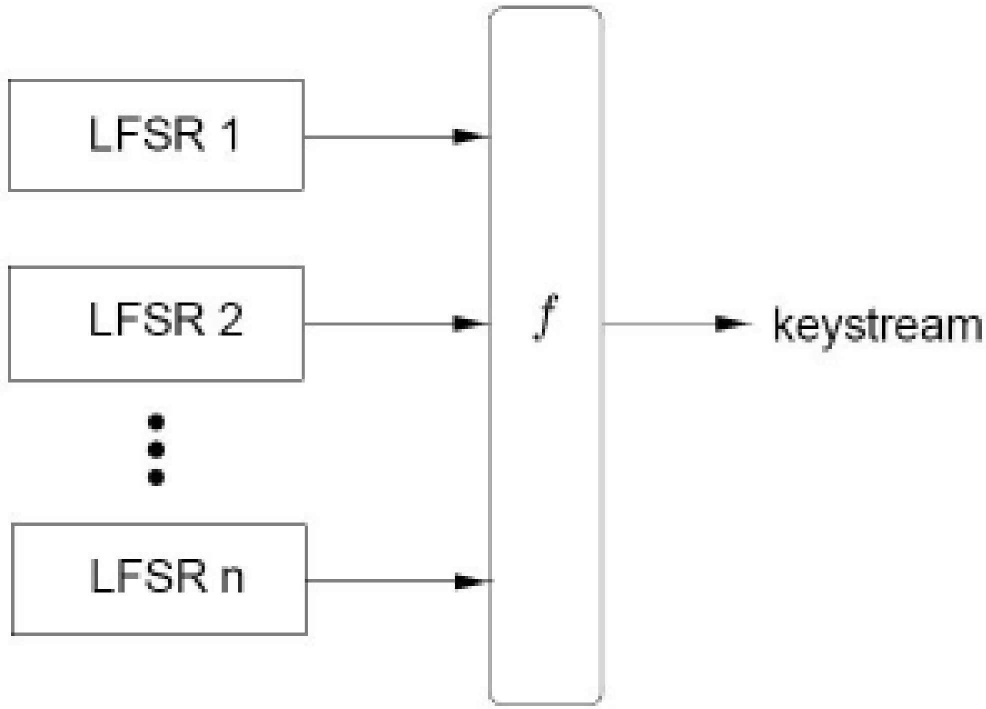
Нелинейные регистры сдвига

- 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:

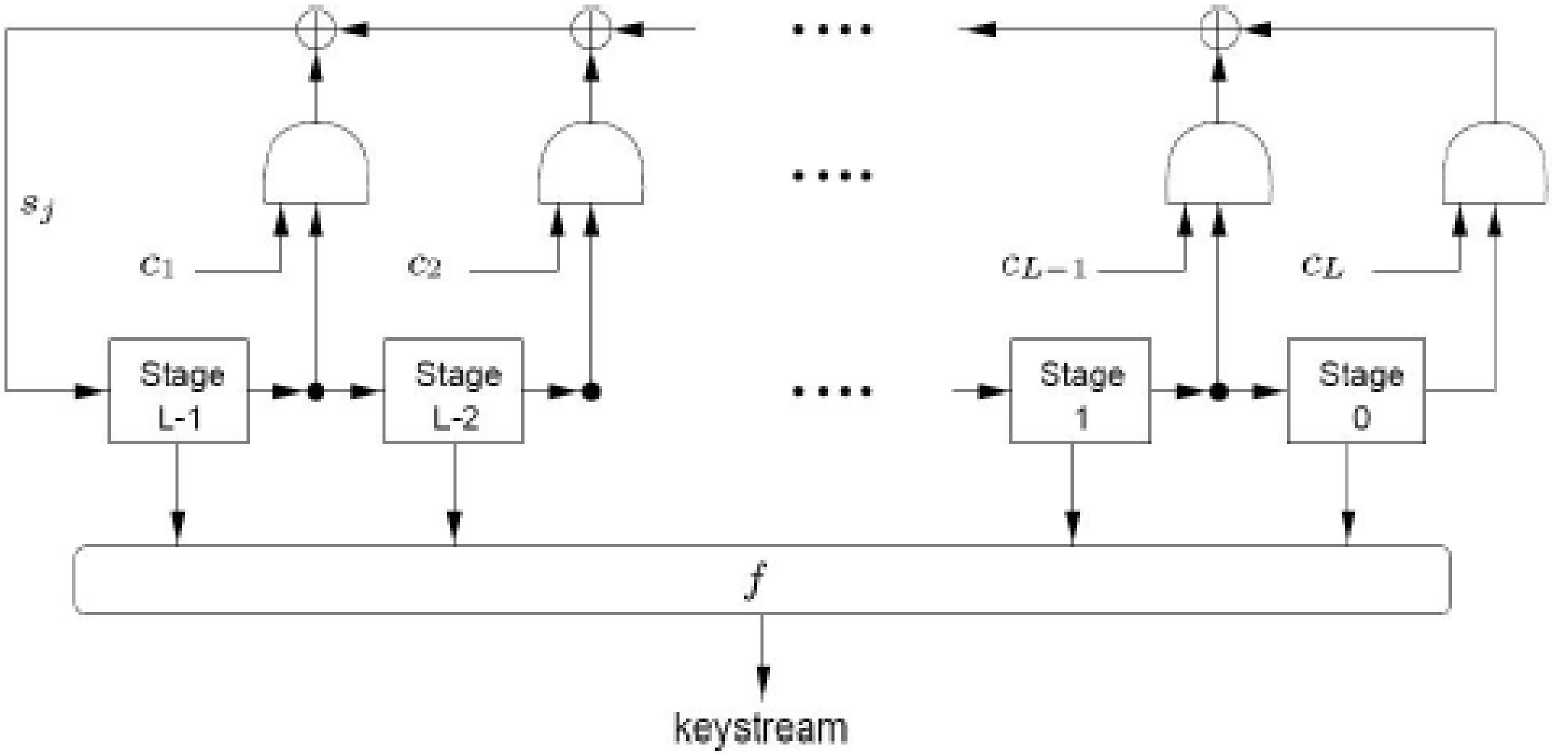


Нелинейные комбинации LFSR

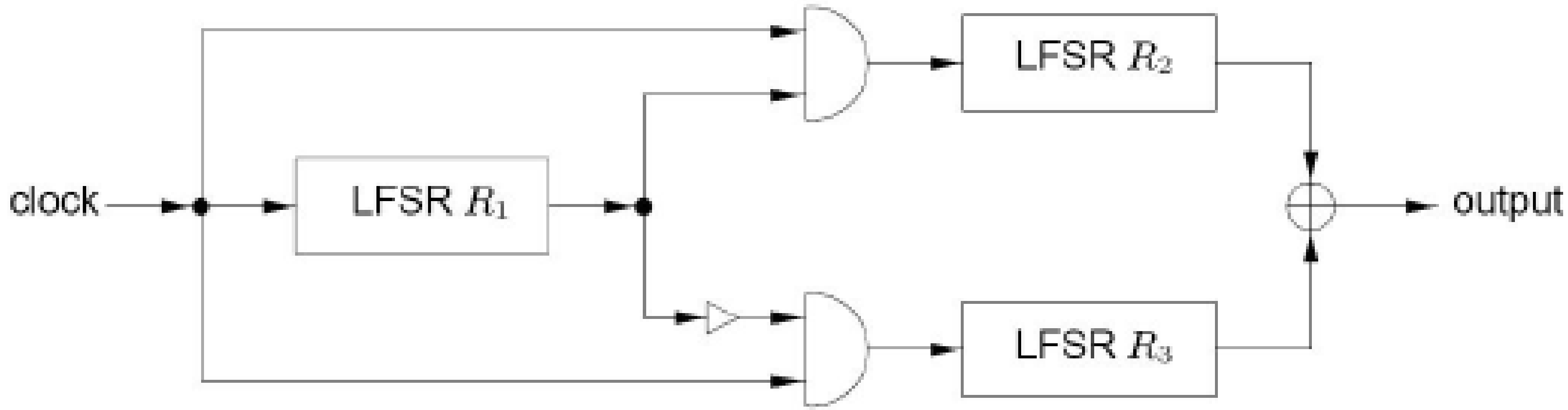
- Можно объединить результаты нескольких LFSR нелинейной функцией:



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



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



Атаки на одноразовый блокнот

- Повторное использование ключей
- Нарушение целостности сообщения.

Атака 1

- Ни в коем случае не используйте одну шифрующую гамму дважды!
  - $C_1 = m_1 \oplus PRG(k)$
  - $C_2 = m_2 \oplus PRG(k)$

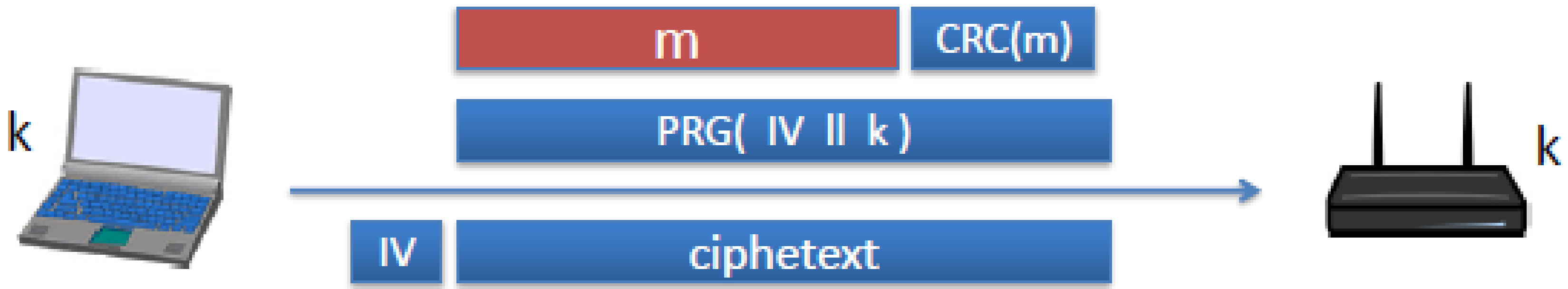
- Атакующий вычислит:
  - $C_1 \oplus C_2 = m_1 \oplus m_2$

- Избыточность языка и кодировок:
  - $m_1 \oplus m_2 \Rightarrow m_1$ и m_2

Атака 1. Исторические примеры

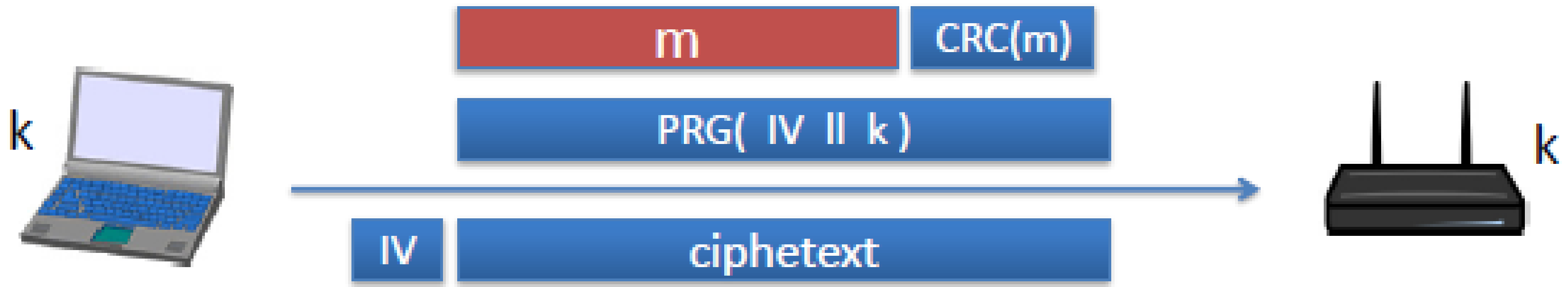
- Проект Venona

- 802.11b WEP



IV – 24 бита: простой счетчик
k – никогда не меняется

WEP



key for frame #1: (1 ll k)

key for frame #2: (2 ll k)

⋮ 24 bits 1024 bits

Fluhrer, Mantin and Shamir 2001:

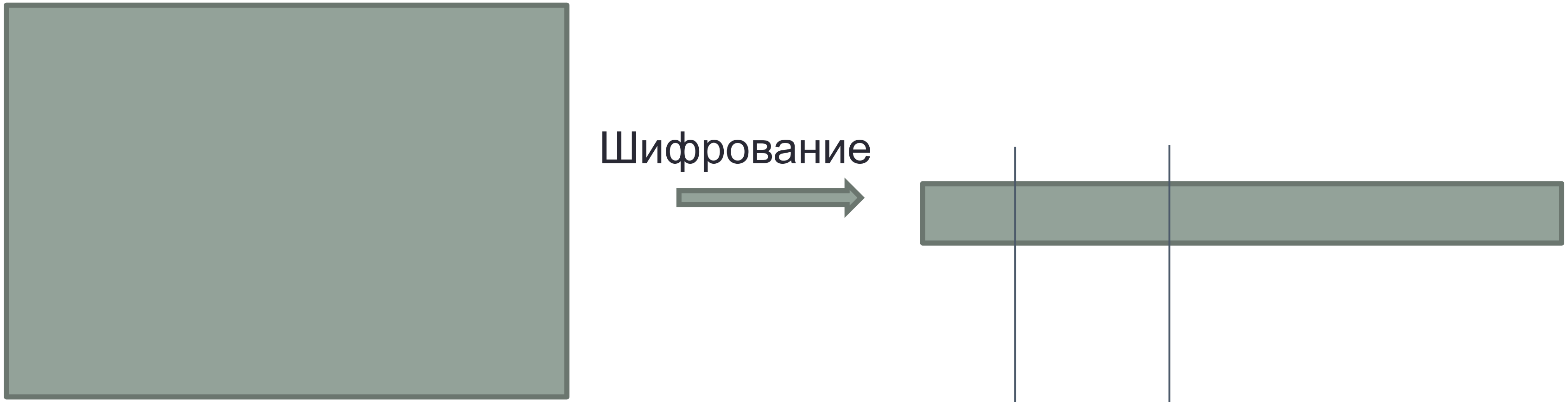
RC4 → k выделяется за 10^6 пакетов

Лучшее решение

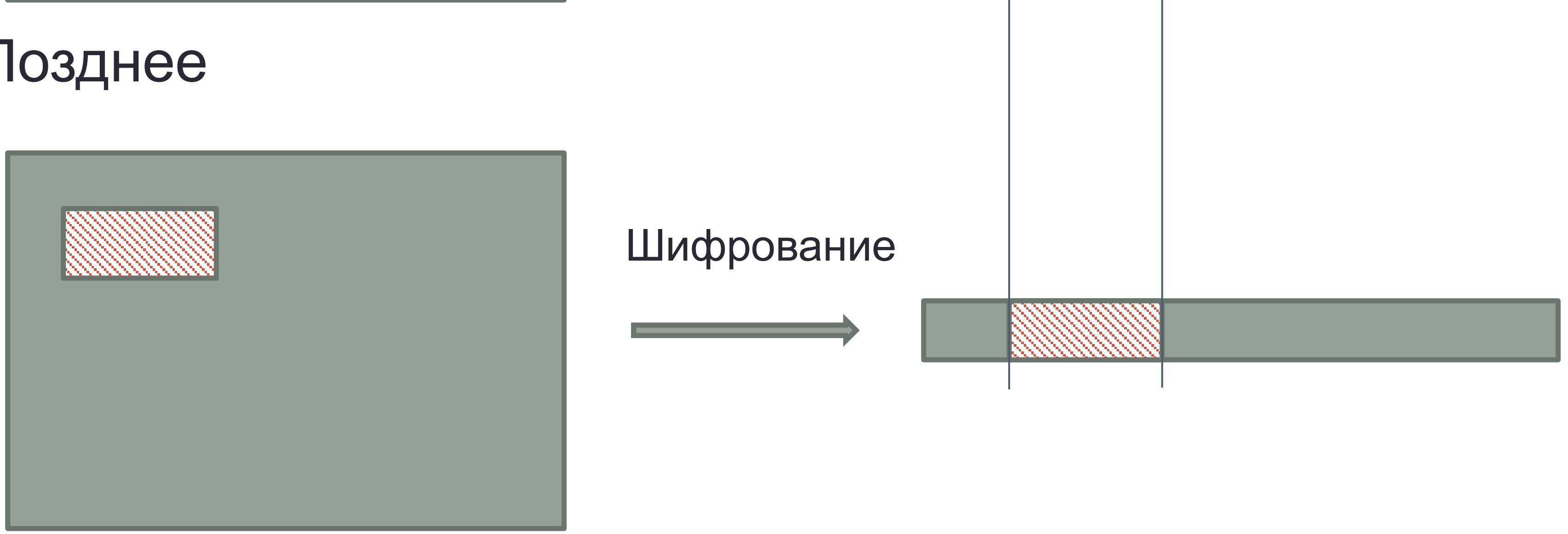
Для каждого пакета отдельный псевдослучайный ключ

Лучшее решение: использовать более надежный алгоритм шифрования

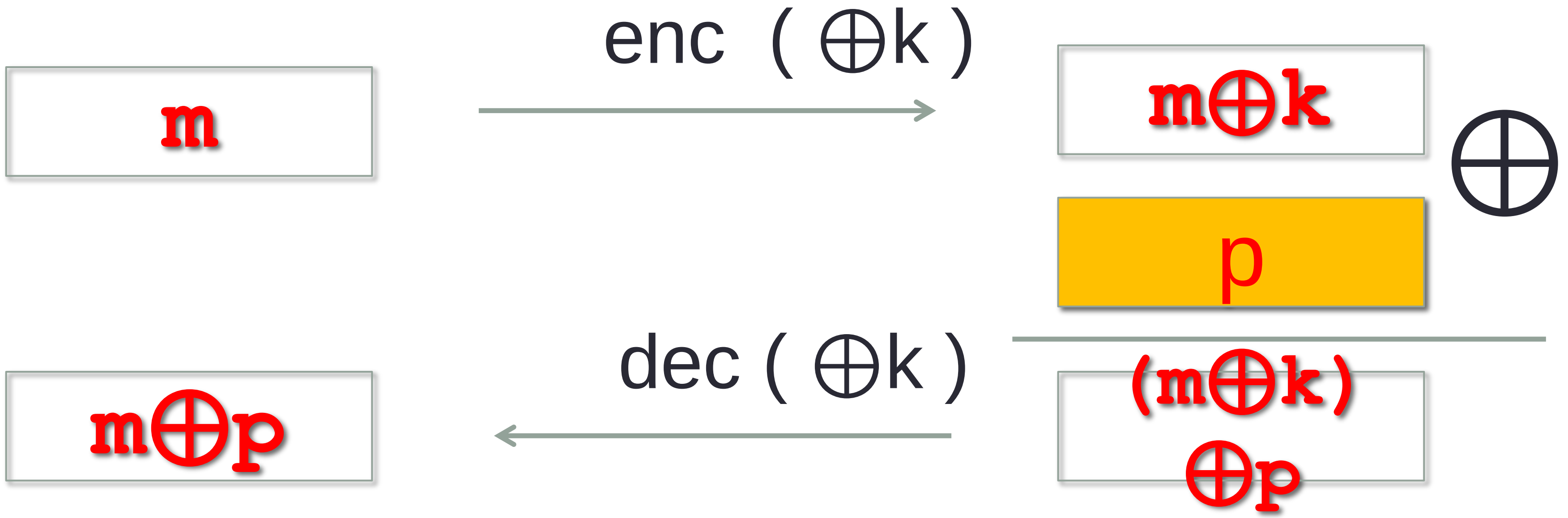
Шифрование дисков



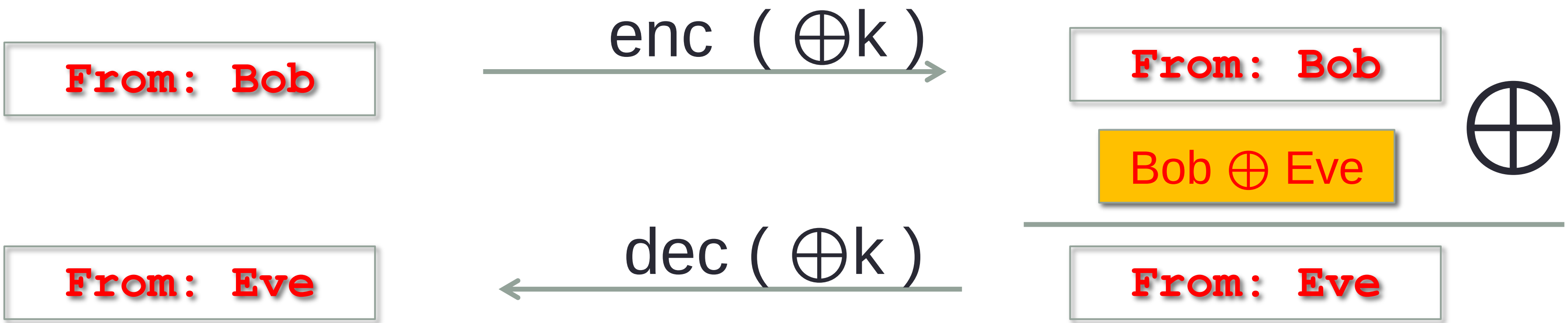
- Позднее



Атака 2. Нарушение целостности



Атака 2. Пример



Активная атака приводит к успешному подлогу!!!

Улучшение схемы 1

- Использование самосинхронизирующихся потоковых генераторов
  - Зависимость от зашифрованного текста

From: Bob

$$\mathrm{enc}\ (\oplus k(c_1, c_2, c_3 \ldots))$$

From: Bob

Bob ⊕ Eve

$\oplus$

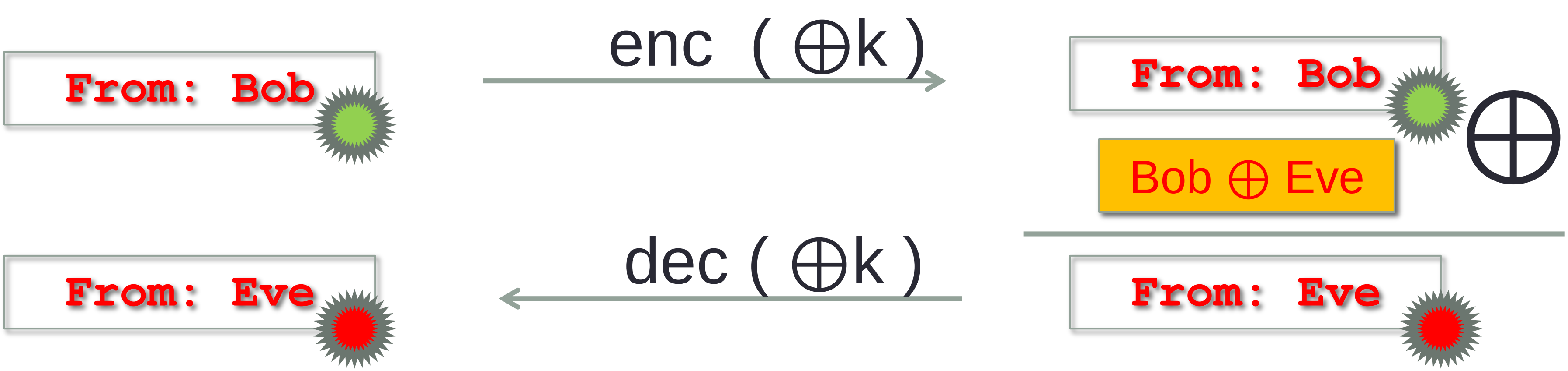
$$\mathrm{dec}\ (\oplus k\ (c'_1, c'_2, c'_3\ \ldots))$$

From: ???

From: Eve

Улучшение схемы 2

- Применение внутри сообщения проверочных сумм
  - CRC
  - HMAC
  - Цифровая подпись

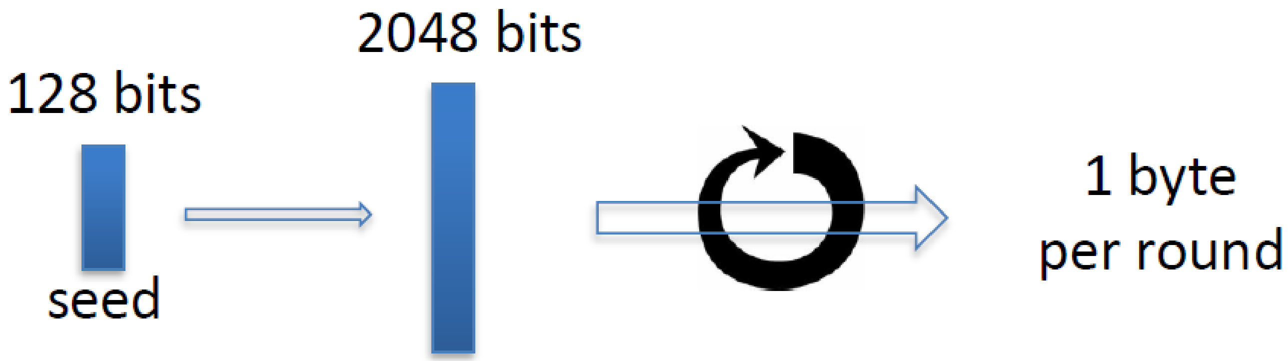


Атаки. Выводы

- Даже абсолютно надежно одноразовый блокнот не гарантирует безопасность!
  - Никогда не используйте одну последовательность дважды
    - При передаче данных: Один ключ → одна сессия
    - При хранении данных: Не использовать потоковые шифры
  - Шифрование $\neq$ Целостность

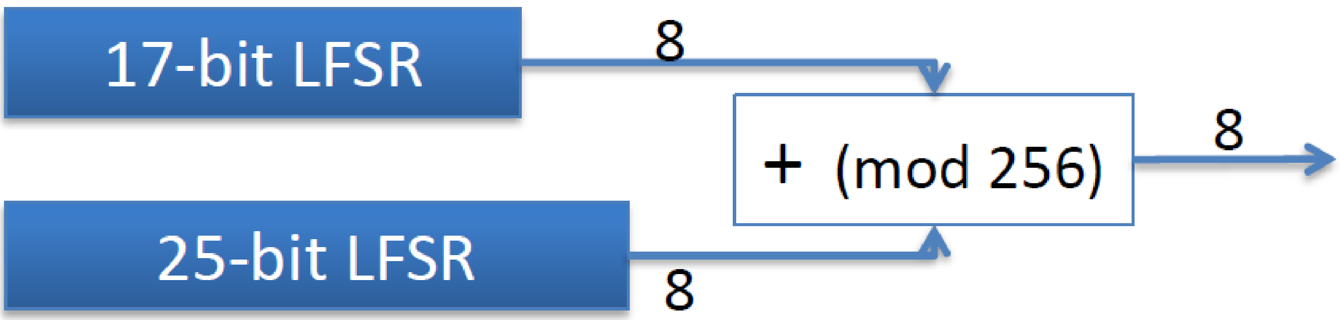
Потоковые шифры в реальном мире

- RC4 – 1987 год

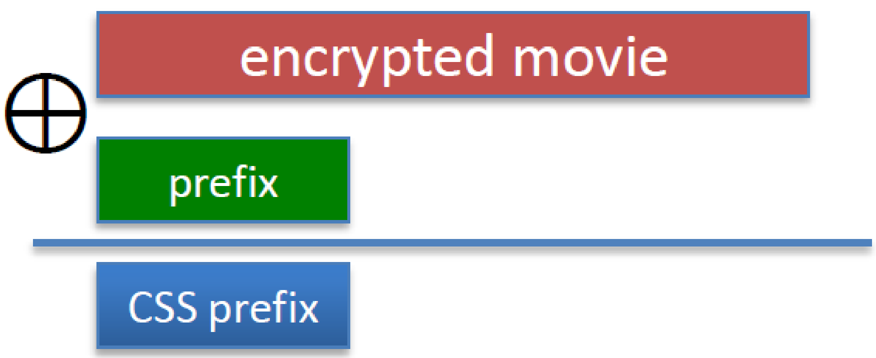


- Недостатки:
  - Смещение распределения вероятностей выходных байт
  - Успешные атаки на алгоритм формирования ключа

Content Scramble System(CSS)



- Успешно вскрыт перебором по 17-битному регистру



Другие алгоритмы схемы на LFSR

- GSM стандарт шифрования A5
  - Формирует битовую последовательность на 3 регистрах
  - Вскрыт

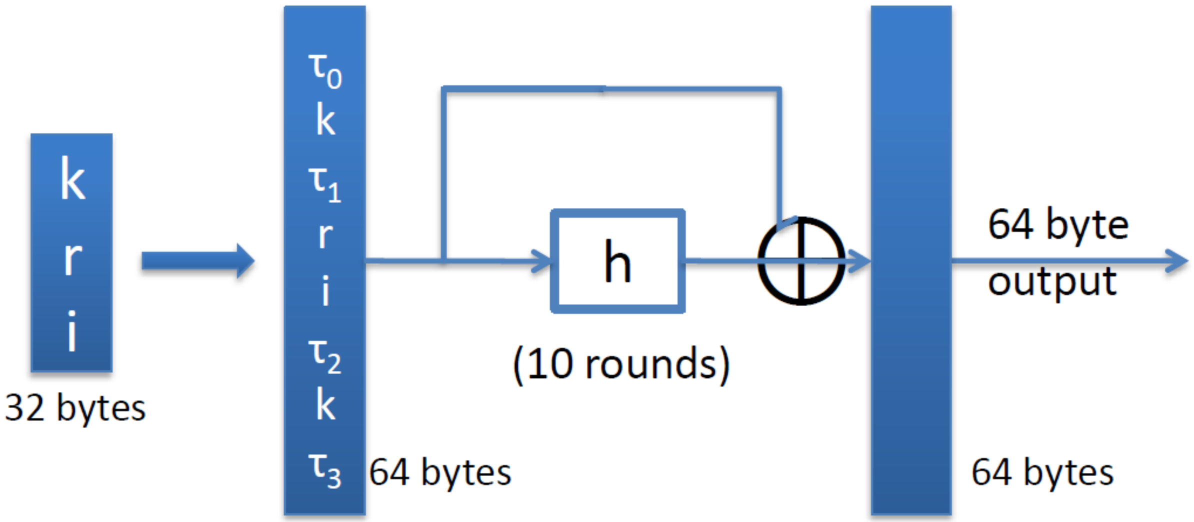
- Bluetooth (E0)
  - Использует 4 регистра
  - Вскрыт

eStream: Salsa

nonce

Salsa20: $\{0,1\}^{128 \text{ or } 256} \times \{0,1\}^{64} \longrightarrow \{0,1\}^{n}$ (max n = 2^{73} bits)

Salsa20(k ; r) := H(k , (r, 0)) || H(k , (r, 1)) || ...



Является ли Salsa безопасной?

- НЕИЗВЕСТНО:
  - Не существует **доказано** не предсказуемых PRG функций
- На сегодняшний день:
  - Не предложено ни одной успешной атаки на алгоритм, кроме исчерпывающего поиска

Производительность

AMD Opteron, 2.2 GHz (Linux)

	PRG	Speed (MB/sec)
	RC4	126
eStream	Salsa20/12	643
eStream	Sosemanuk	727

Генераторы случайных значений

- Для чего:
  - IV
  - Ключи
  - Метки
- Откуда:
  - Аппаратные датчики
  - Различные счетчики времени(клавиатура, мышь…)

Определение безопасности для PRG

- Пусть G: K→$\{0,1\}^n$ PRG

- Определим, что обозначает неотличимость

- $\left[k \overset{R}{\leftarrow} K,\ \text{для } G(k)\right]$
- от
- $\left[r \overset{R}{\leftarrow} \{0,1\}^n,\ \text{для } r\right]$

Статистические тесты

- Статистический тест над $\{0,1\}^n$: это такой алгоритм A, который для любого x возвращает значение A(x) =
  - 0 (если число не случайно)
  - 1 (если число случайно)
- Например:

$$(1) \quad A(x)=1 \quad \text{iff} \quad \left| \#0(x) - \#1(x) \right| \le 10 \cdot \sqrt{n}$$

$$(2) \quad A(x)=1 \quad \text{iff} \quad \left| \#00(x) - \frac{n}{4} \right| \le 10 \cdot \sqrt{n}$$

Количественная оценка стойкости

- Пусть G: K→$\{0,1\}^n$ PRG и A Статистический тест над $\{0,1\}^n$, тогда

- $Adv_{PRG}[A,G] = \left| \underbrace{Pr}_{k \overset{R}{\leftarrow} K} [A(G(k)) = 1] - \underbrace{Pr}_{r \overset{R}{\leftarrow} \{0,1\}} [A(r) = 1] \right|$

$A(x) = 0 \Rightarrow Adv_{PRG}[A,G] = \ ?$

Secure PRG

- Определение:

- Пусть $G: K \rightarrow \{0,1\}^n$ PRG, тогда будем называть ее стойкой, если для любого эффективного стат. теста A
- Adv[A,G] – пренебрежимо мала
-

- Существуют ли доказано стойкие PRG?

Yao 1982

- Теорема:
- Непредсказуемый PRG – является стойким.
- Пусть G: K→$\{0,1\}^n$ PRG, тогда будем называть ее стойкой, если для любого i от 1 до n-1 G – непредсказуема

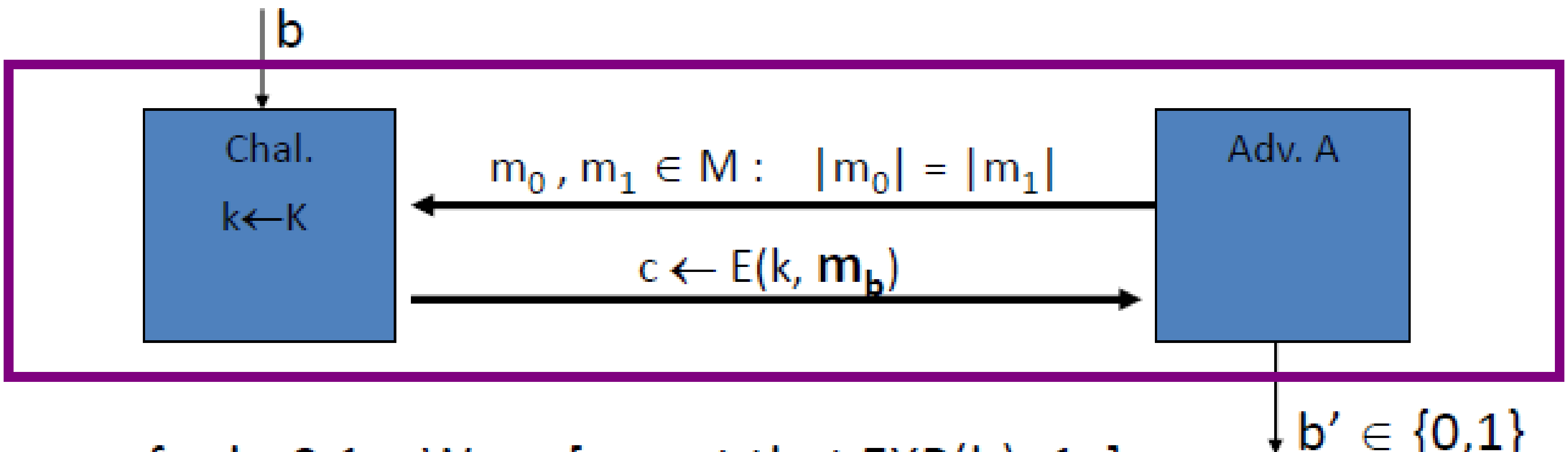
Обобщение

- Два распределения P1 и P2 над $\{0,1\}^n$ статистически неразличимы, если для любого стат теста A выполняется

$$\left| \Pr_{\underbrace{x \xleftarrow{R} P1}} [A(x) = 1] - \Pr_{\underbrace{x \xleftarrow{R} P2}} [A(x) = 1] \right|$$ -- пренебрежимо мала

Например: $\{ k \xleftarrow{R} K :\ G(k) \} \approx_p \text{uniform}(\{0,1\}^n)$

Семантическая стойкость



for b=0,1: W_b := [event that EXP(b)=1]

$$\mathrm{Adv}_{SS}[A,E] := \left| \Pr[W_0] - \Pr[W_1] \right| \in [0,1]$$

Adv[A, E] – пренебрежимо мала