Защита информации

Павел Юдаев

МГТУ им. Баумана, Кафедра ИУ-9

Москва, 2014

Раздел 6 - Шифры AES и RC4

TODO: в 2015 надо рассказать еще короче, буквально в двух словах. Т.к. анализировать AES мы не можем.

Кратко о шифре AES, он же Rijndael.

В 1997-2001 году был объявлен конкурс на новый стандарт симм. шифра AES, и Rijndael его выиграл.
Стандарт шифрования в США с 2001 года (FIPS 197).

Блочный шифр. Блок - 128 бит. Ключ - 128, 196 или 256 бит. (Мы рассм. 128 бит.)

10 раундов для ключа в 128 бит, 12 для 192, 14 для 256.

Использует не схему Фейстеля, а сеть замен и перестановок (substitution-permutation network): S-блоки (нелинейные замены на основе обращения элемента расширения конечного поля) и P-блоки (перестановки, линейные).

Как и у шифра DES, раундовые ключи AES для шифрования и расшифрования строятся по ключу шифра заранее и хранятся, пока используется этот ключ.

TODO: вот отсюда в 2015 не рассказывать. Сразу перейти к атакам на AES.

Шифрование:

00) Построение расписания ключей (key schedule) для 10+1 раундов. Раундовый ключ - 128 бит.

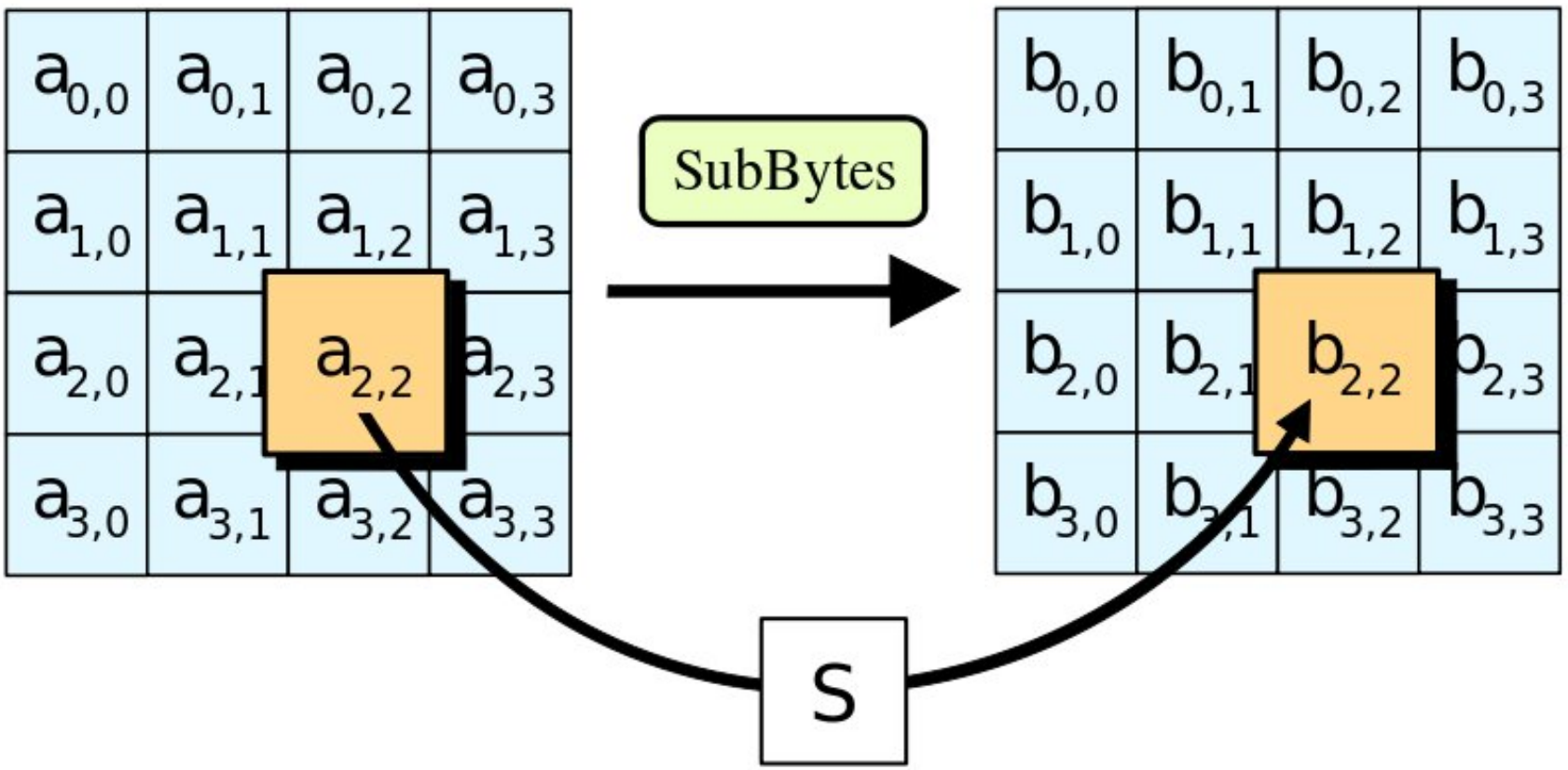
Блок представлен в виде таблицы 4x4, элемент таблицы - 1 байт. Итого 16 байтов, 128 бит. Его называют state, “состояние”. Индексы - позиции в таблице.

0) Начальный раунд, AddRoundKey. $state = m \oplus round\_key$.

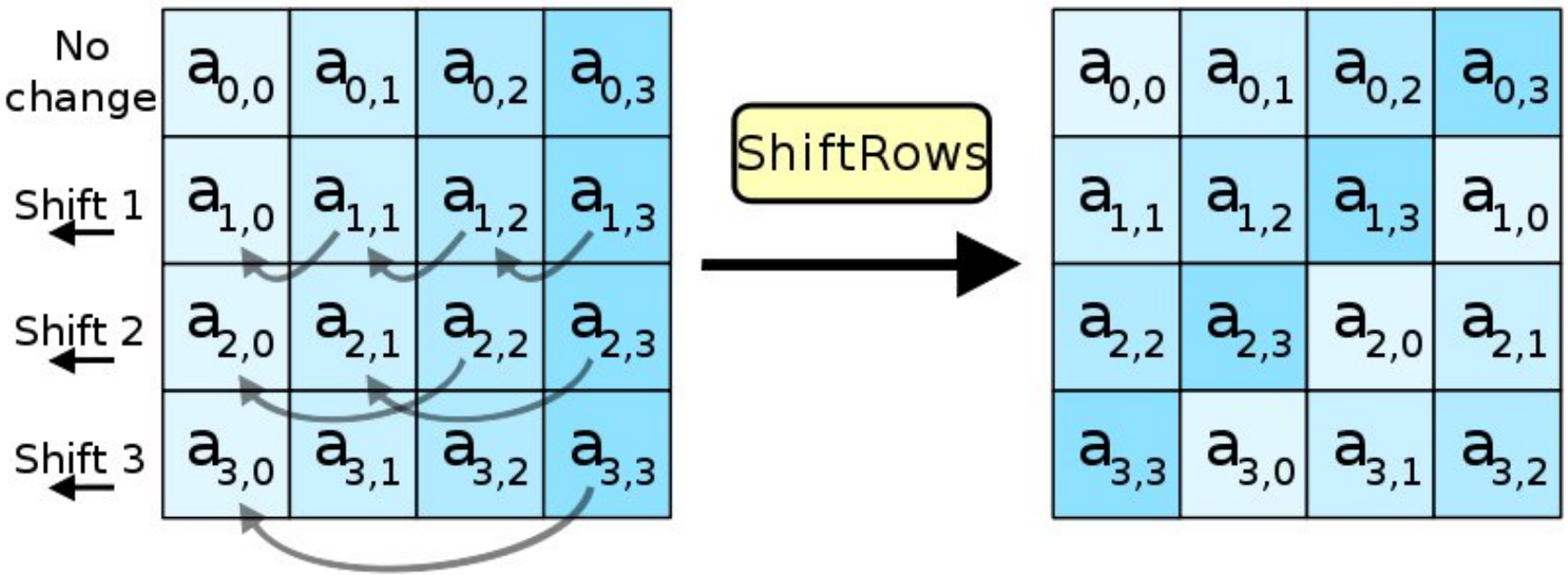
1)-9) К блоку в заданном порядке прим. 4 преобразования:
SubBytes
ShiftRows
MixColumns
AddRoundKey

10) Последний раунд
SubBytes
ShiftRows
AddRoundKey
(без MixColumns)

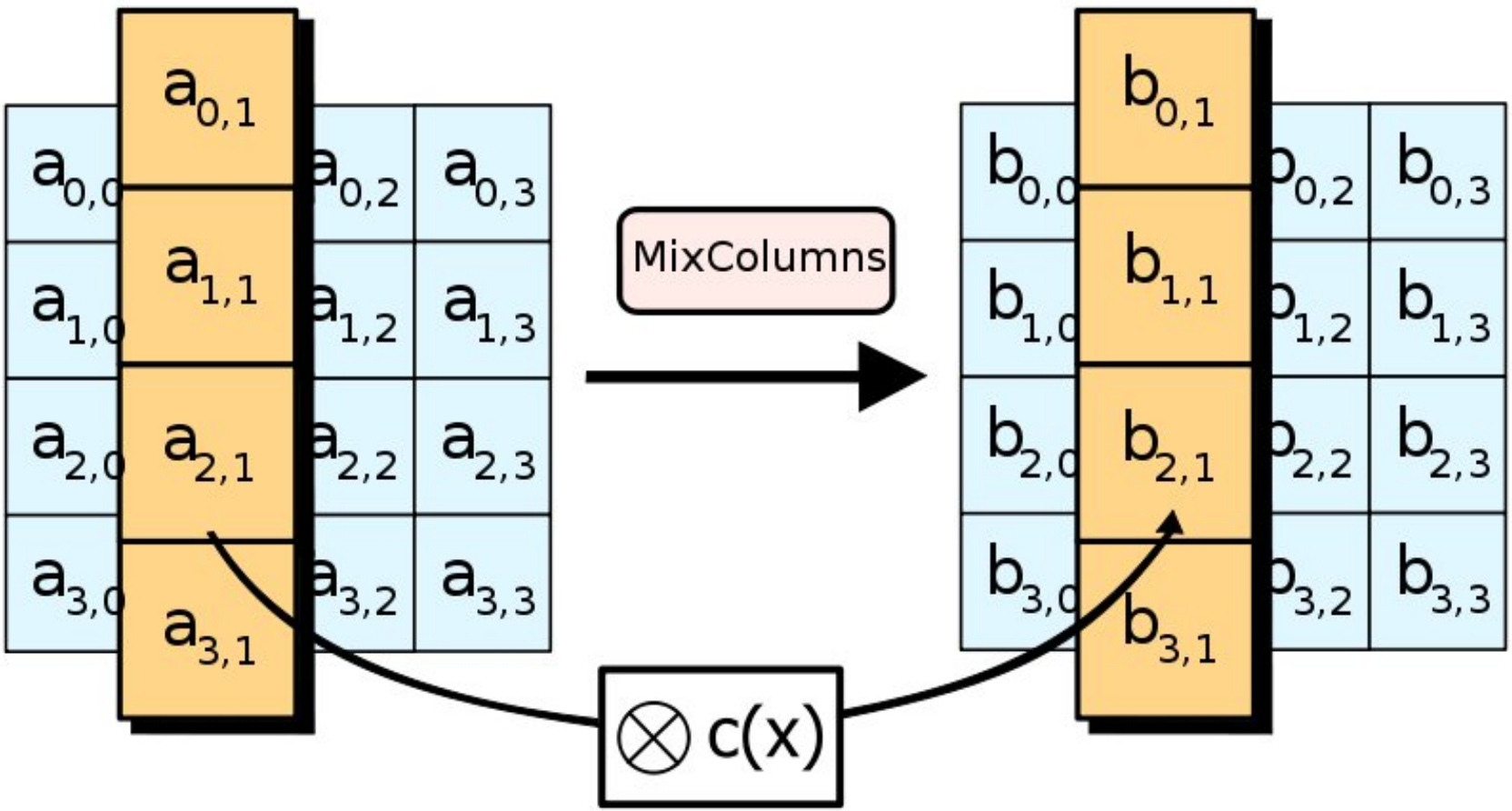
SubBytes. Нелинейная операция замены, Rijndael S-блок.
$state[i][j] = Sub(state[i][j])$
Комбинация обращения элемента конечного поля и обратимой линейной операции над ним.
Программная реализация - таблица.



ShiftRows. Каждый ряд матрицы 4x4 сдвигается на 0, 1, 2, 3 позиции соотв., ряды обрабат. независимо.
$state[i][\,] = Shift(state[i][\,], i)$



MixColumn. Для каждого столбца (независимо) применяется фиксированное обратимое линейное преобразование.
$state[\,][j] = Mix(state[\,][j], j)$



AddRoundKeys. $state = state \oplus round\_key$.

Свойства операций внутри одного раунда:

- AddRoundKey - это $\oplus$, она обратная к самой себе.
- SubBytes и ShiftRows коммутативны (первая работает с байтами целиком, вторая сдвигает байты целиком, не меняя их значений),т.е.
  $SubBytes(ShiftRows(state)) = ShiftRows(SubBytes(state))$
- MixColumns линейна относительно входных данных (столбцов), поэтому:
  $MixColumns^{-1}(state \oplus round\_key) =$
  $MixColumns^{-1}(state) \oplus MixColumns^{-1}(round\_key)$

Поэтому можно применить тот же порядок обратных операций при расшифровании, что и при шифровании, изменив раундовые ключи.

Шифрование:

0) AddRoundKey
1) SubBytes
ShiftRows
MixColumn
AddRoundKey
...
9) SubBytes
ShiftRows
MixColumn
AddRoundKey
10) SubBytes
ShiftRows
AddRoundKey

Расшифрование:

00) Расписание ключей - в обратном порядке, при этом на раундах 1-9 к раундовым ключам применена операция MixColumns^{-1}.

$$round\_key_{D,10} = round\_key_{E,10}$$

$$round\_key_{D,t} = MixColumns^{-1}(round\_key_{E,t}) \ \forall t = 1..9$$

$$round\_key_{D,0} = round\_key_{E,0}$$

0) *(10)* AddRoundKey. $state = state \oplus round\_key_{D,10}$

1) *(10)* SubBytes^{-1}
(10) ShiftRows^{-1}
(9) MixColumns^{-1}
(9) AddRoundKey. $state = state \oplus round\_key_{D,9}$

2) *(9)* SubBytes^{-1}
(9) ShiftRows^{-1}
(8) MixColumns^{-1}
(8) AddRoundKey. $state = state \oplus round\_key_{D,8}$

...

10) *(1)* SubBytes^{-1}
(1) ShiftRows^{-1}
(0) AddRoundKey. $state = state \oplus round\_key_{D,0}$

TODO: в 2015 отсюда продолжать рассказ.

Атаки на AES

В атаках стараются найти секретный ключ, а не просто “какую-то информацию” об открытом тексте.

Перестановка AES - односторонняя функция по k?

Перестановка AES - односторонняя функция с секретом по m?

Семантическая стойкость перестановки AES? (Шифр для 1 бл. с 1-р. кл.)

Перестановка AES - ПСП?

1)

Опр. (Атака на k связанных ключах)

Пусть можно по любому открытому тексту получать шифротекст при k не известных ключах, связанных известным соотношением.
Цель: найти первый ключ.

AES-256: По $2^{99.5}$ пар (m, c) на 4 связанных ключах можно найти ключ за время $2^{99.5}$.

В реальных сценариях нет связанных ключей.

2) Атака с выбранным открытым текстом без связанных ключей.

Ключ 128 бит - время $2^{126.1}$, 256 бит - время $2^{254.4}$.

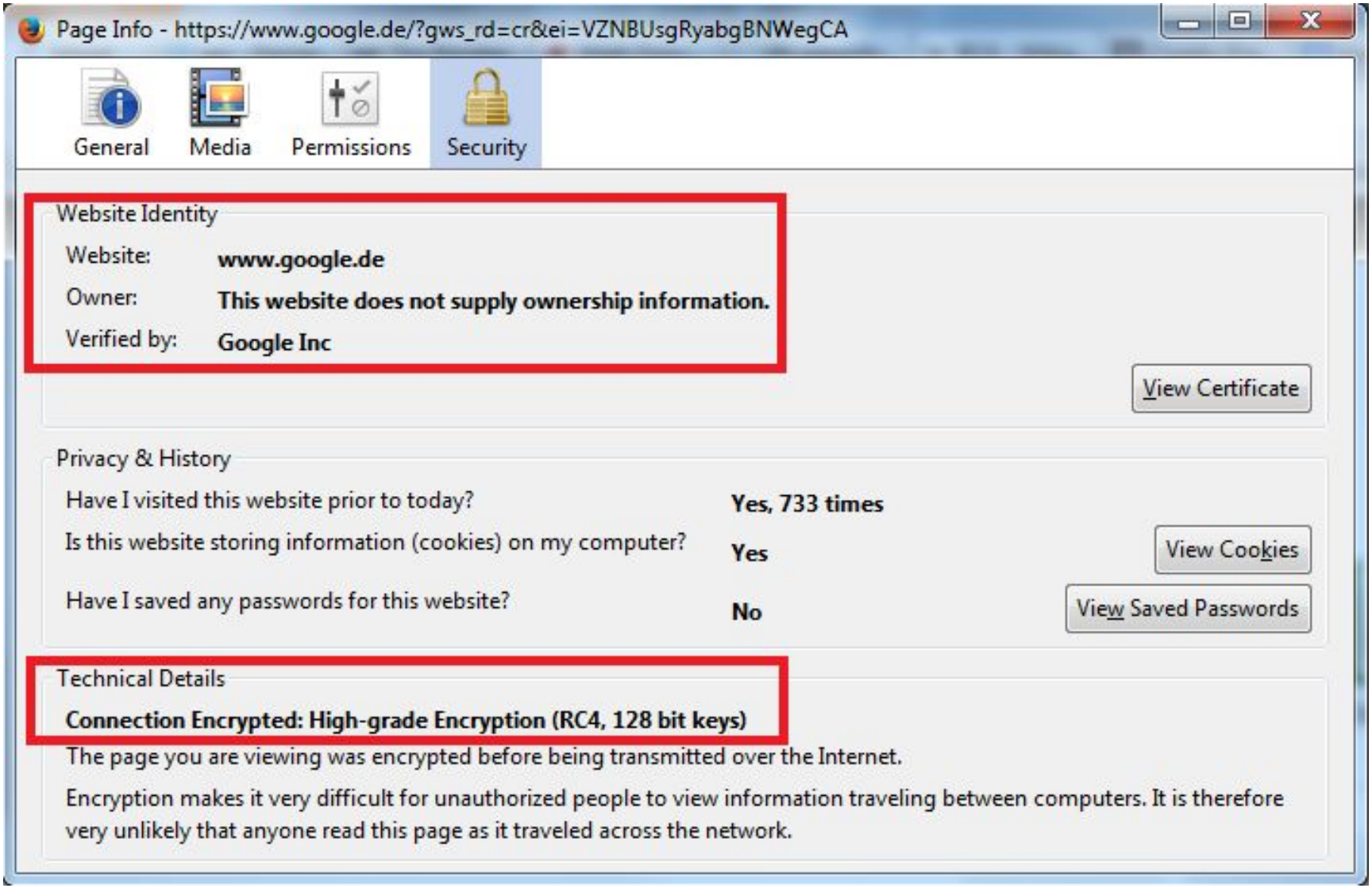
Раздел 6 - Шифры AES и RC4

1987, Ron Rivest.

В 1994 году его анонимно опубликовали в интернете.
(Правила Керкгоффса?)

Преимущество: скорость, нелинейность ГПСЧ.

2012 год:


Page Info - https://www.google.de/?gws_rd=cr&ei=VZNBUsgRyabgBNWegCA
General
Media
Permissions
Security
Website Identity
Website: www.google.de
Owner: This website does not supply ownership information.
Verified by: Google Inc
View Certificate
Privacy & History
Have I visited this website prior to today? Yes, 733 times
Is this website storing information (cookies) on my computer? Yes
View Cookies
Have I saved any passwords for this website? No
View Saved Passwords
Technical Details
Connection Encrypted: High-grade Encryption (RC4, 128 bit keys)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it very difficult for unauthorized people to view information traveling between computers. It is therefore very unlikely that anyone read this page as it traveled across the network.

ГПСЧ:

1) Создание начального значения S (seed)
Длина секретного ключа - от 1 до 256 байтов, типично - от 5 до 16.
S - массив из 256 байтов. Python:

```
S = range(0, 256)
j = 0
keylen = len(key)
for i in range(0, 256):
  j = (j + S[i] + key[i % keylen]) % 256
  S[i], S[j] = S[j], S[i]
```

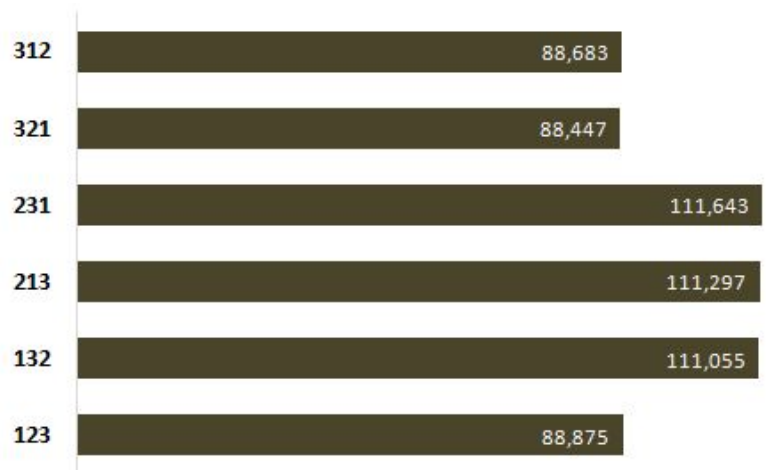
Есть неравномерности в вероятности разных значений S.

Наивный алгоритм перемешивания (см. [2]):

```
for (int i = 0; i < array.Length; ++i)
{
  int n = rand.Next(array.Length);
  Swap(array[i], array[n]);
}
```

Для массива длины 3 здесь $3^3 = 27$ возм. путей. Но разных комбинаций - всего 3!=6.

600000 испытаний - неравномерность результата:

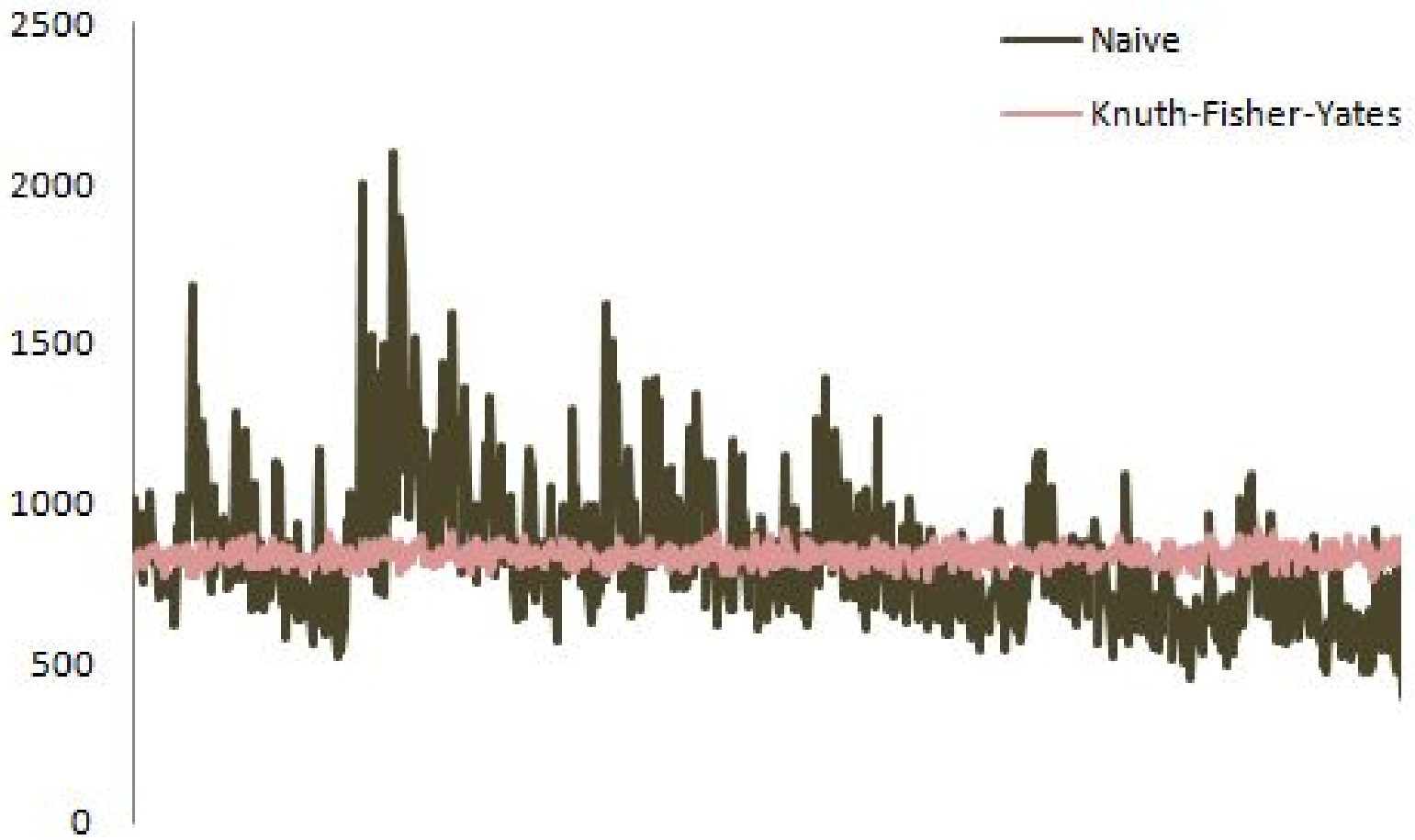


Алгоритм Knuth-Fisher-Yates:

```
for (int i = array.Length - 1; i > 0; --i)
{
  int n = rand.Next(i + 1);
  Swap(array[i], array[n]);
}
```

Ровно (array.Length)! возможных путей. При равномерном rand они равновероятны.

Массив из 6 элементов, 600000 испытаний:



Наивный алгоритм: $6^6 = 46656$ путей.

Кнут и реальность: $6! = 720$ комбинаций.

2) собственно ГПСЧ. S - состояние. Меняет его при работе.

```
i = 0
j = 0
while GeneratingOutput():
  i = (i + 1) % 256
  j = (j + S[i]) % 256
  S[i], S[j] = S[j], S[i] # перестановка двух байтов
  output S[(S[i] + S[j]) % 256]
```

(S-блок, i, j): $256! \cdot 256^2 \approx 2^{1700}$ состояний.
$\forall$ байт S-блока изменится через 256 итераций.

ГПСЧ:

- нелинейный, проходит стат. тесты на $output[i]$ при $i > 150$
- практически непредсказуемый при $i > 150$
- $P(output[1] = 0) = 2/256$ (второй байт)
- начало *output* неравновероятное! проблема!

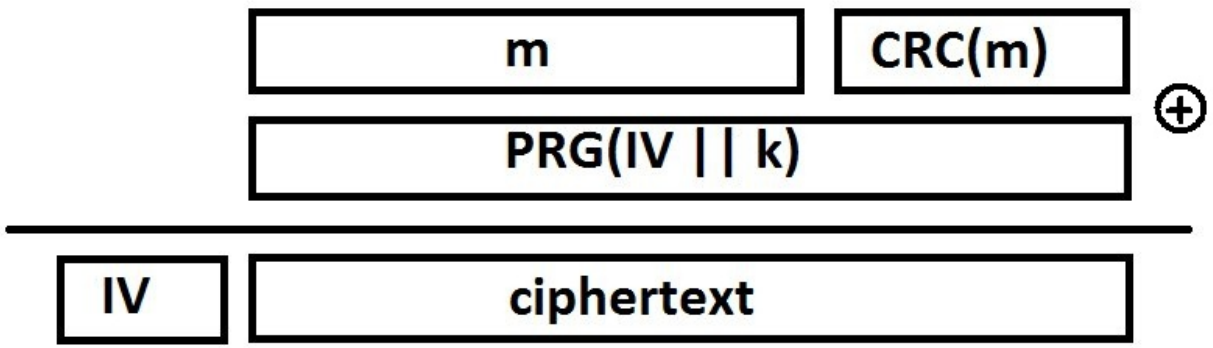
“RSA Security”: шифр RC4 при случайном выборе ключа устойчив к линейному и дифф. криптоанализу.

Шифр RC4:

- быстрый
- не требует IV, pad. Нет padding oracle, BEAST, Lucky13.
- детерминированный при фикс. ключе. Nonce - **как добавить?**

Атака на протокол WEP

IV - счетчик, nonce. $|m|$ - до 2300 Б. Но можно и 1 байт.



Добавили nonce: $G(IV||k)[0..t]$.

- Ключ - 104 бита. IV - 24 бита.
  IV повторяется после $2^{24} \approx 16M$ сообщений. На некоторых сетевых картах IV $= 0$ после перезагрузки.
- Неравновероятное *начало последовательности*
- Предсказуемо связанные ключи: $(1||k)$, $(2||k)$, ...
- Существует атака на связанных ключах, и др.
- $\Rightarrow$ становится известным секретный ключ k.
  2001 год: 10^6 фреймов.
  2005-2007: 60000 фреймов, 80% вер-ть узнать весь ключ.
  Время: 3 минуты.

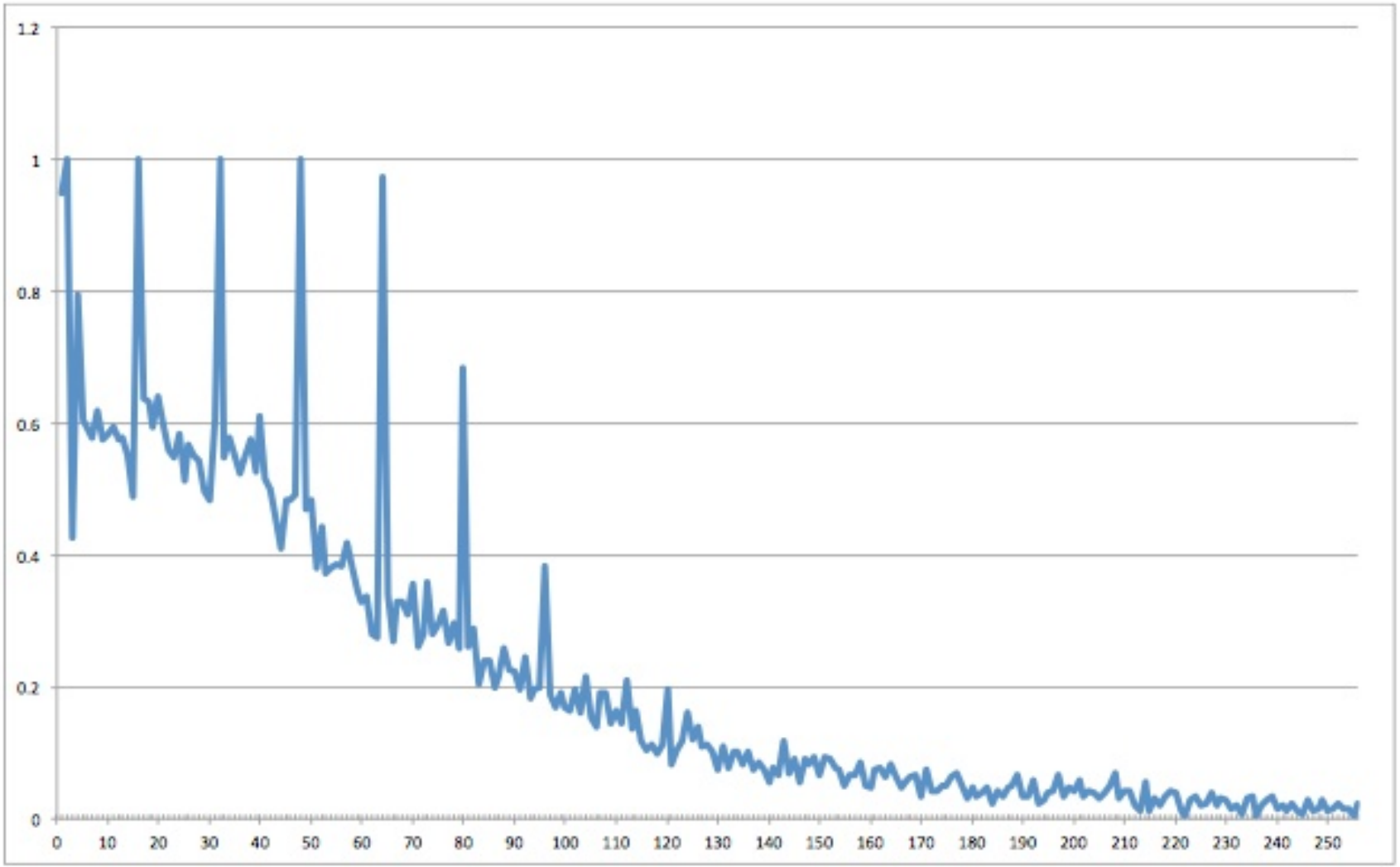
WEP можно было бы: $G(k) \Rightarrow k_1, k_2, ...$
k_i - псевдослучайный ключ для фрейма: $c = m \oplus G(k_i)$

Используйте WPA.

Атака на RC4 2013 года: (*)

Исслед. неравномерности (biases) в первых 256 байтах ГПСЧ.

Вер-ть узнать байт ГПСЧ по $2^{24} \approx 16M$ шифротекстов:



(*)

Пример (для атаки 2013 года)

Вход на сайт - cookie авторизации в начале запроса по HTTPS. Потом создается сессионный ключ, cookie не передается.

Новая сессия - снова cookie. Так миллион раз??

А так: Вредоносный javascript в другом фрейме устанавливает соединения с сайтом.

Атаки становятся только лучше.

Применение RC4:

WEP (взломан!)
BitTorrent protocol
Microsoft Point-to-Point Encryption
TLS / SSLayer (optionally)
Secure Shell (optionally)
Remote Desktop Protocol
Kerberos (optionally)
PDF

Gpcode.AK, вирус для Windows в июне 2008. Брал данные на диске в заложники, шифруя их шифрами RC4 и RSA-1024. Расшифрование - за плату.

Скорость работы поточных и блочных шифров
(AMD Opteron 2.2 GHz, Linux, crypto++ 5.6.0)

Шифр	Размер блока/ключа	Скорость, МБ/с
поточные шифры		
RC4 (1987)		126
Salsa20 (2005)		643
блочные шифры		
3DES (1977, 1997)	64/168	13
AES-128 (2000)	128/128	109

Литература к лекции:

1. RC4 is kind of broken,
http://blog.cryptographyengineering.com/2013/03/attack-of-week-rc4-is-kind-of-broken-in.html

2. Jeff Atwood, *The Danger of Naivete*,
`http://blog.codinghorror.com/the-danger-of-naivete/`

3*. Атака на AES с выбранным открытым текстом на 4-х связанных ключах.
http://eprint.iacr.org/2009/317.pdf

4*. Атака на AES с выбранным открытым текстом без связанных ключей.
http://research.microsoft.com/en-us/projects/cryptanalysis/aesbc.pdf