Лекция №1

Часть 1. О курсе

Елена Киршанова

Курс “Основы криптографии”

Админ

- Встречаемся в Webinar по вторникам в 17:10 по Калининграду
- Лабораторные принимаются после лекций
- Страница курса
  `https://crypto-kantiana.com/elena.kirshanova/teaching/info_sec_2023.html`
- Для прохождения курса: сдача всех лабораторных + тест в день зачёта

Лабораторные работы

- После каждой лекции будет опубликовано задание к лабораторной работе, например `https://crypto-kantiana.com/elena.kirshanova/teaching/info_sec2022/Lab1.txt`
- Реализовывать алгоритмы с помощью библиотеки crypto++ (обе C++)
- Лабораторные работы выполняются индивидуально

Структура и план курса

I. Симметрическая криптография

- 07/09 Псевдослучайные генераторы
- 14/09 Блок-шифры
- 21/09 Хэш-функции. Коды аутентификации сообщений

II. Асимметрическая криптография

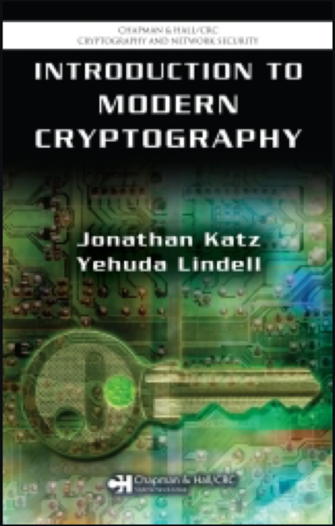
- 28/09 Обмен ключами
- 05/10 Цифровые подписи
- 12/10 TLS и мессенджеры

Чего не будет

Мы не будем говорить о

- блокчейнах
- программировании / реверс инжиниринге
- хакерстве
- квантовой и пост-квантовой криптографии

Литература



A Graduate Course in Applied Cryptography

Dan Boneh, Victor Shoup

`https://toc.cryptobook.us/book.pdf`

Комментарии

- Подразумеваем знания элементарных алгоритмов, тер. вера, линейной алгебры
- Будет много англоязычных слов!
- Опечатки неизбежны
- Комментарии/замечания/пожелания/недовольства можно отправить по почте

  elenakirshanova [at] gmail [dot] com

Часть II

Принципы криптографии

Определения I.

Принятая модель вычислений – машина Тьюринга

Полиномиальное время

Алгоритм $\mathcal{A}$ работает за *полиномиальное время*, если, получая на вход данные размера n бит, $\mathcal{A}$ терминирует за время $\mathcal{O}(n^k)$ для константы k.

Примеры:

- умножение двух n-битных чисел: $\mathcal{O}(n \log n)$ – полиномиальное время
- факторизация n-битного числа: $\exp(\mathcal{O}(n^{1/3} \cdot (\log n)^{2/3})$ – субэкспоненциальное время

Алгоритм $\mathcal{A}$ называется *вероятностным полиномиальным* (ppt), если он работает за полиномиальное время и использует случайные биты.

Определения II.

Пренебрежимо малая функция

Функция $f : \mathbb{N} \to \mathbb{R}$ *пренебрежимо мала* (negl), если для всех многочленов p существует $N \in \mathbb{N}$, такое что для любого $n \geq N$

$$f(n) < \frac{1}{p(n)}.$$

Примеры:

- negl:

$$\frac{1}{2^n}, \ \frac{1}{2^{\sqrt{n}}}, \ \frac{1}{2^{\log^2(n)}}$$

- non-negl:

$$\frac{1}{\log n}, \ \frac{1}{n^2}, \ \frac{1}{2^{\mathcal{O}(\log n)}}$$

Формальное описание шифра

Шифр-схема $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$

включает в себя *ppt* алгоритмы KeyGen, Enc, Dec и множества

$$\mathcal{K} \text{ — множество ключей}$$
$$\mathcal{M} \text{ — множество открытых текстов}$$
$$\mathcal{C} \text{ — множество шифр-текстов}$$

такими, что для

$$k \leftarrow \mathsf{KeyGen}(1^\lambda)$$
$$c \leftarrow \mathsf{Enc}(k, m)$$
$$m' = \mathsf{Dec}(k, c)$$

схема корректна: $\mathsf{Dec}(k, \mathsf{Enc}(k, m)) == m \quad \forall k \in \mathcal{K}, m \in \mathcal{M}$

Шифр-схема (свойства)

Формально: множества $\mathcal{K}, \mathcal{M}, \mathcal{C}$ зависят от пар-ра безопасности λ.

Параметризация шифр-схемы Param(λ) – ppt алгоритм, принимающий на вход пар-р безопасности λ, и выдающий битовую строку $\Lambda = \mathrm{poly}(\lambda)$, задающую параметры шифр-схемы.

Пример: Для криптографической хэш-функции SHA-256 с $\lambda = 128$, Param(λ) выдаст

$$\mathcal{K}_{128} = \{0,1\}^{512} \qquad \mathcal{M}_{128} = \mathcal{C}_{128} = \{0,1\}^{256}$$

Основные принципы современной криптографии

Принцип Керкгоффса (Kerckhoffs' principle)[1]:

`Криптосистема должна оставаться безопасной, если злоумышленнику известно всё, кроме секретного ключа`

Алгоритмы $\mathsf{Enc}, \mathsf{Dec}, \mathsf{Param}$ являются открытыми и подлежат открытым научным исследованиям

[1] Auguste Kerckhoffs, «La Cryptographie Militaire», 1883

Часть III

Абсолютная криптографическая стойкость. Одноразовый блокнот

Шифр Шеннона (Shannon's cipher)

Положим $\mathcal{K}, \mathcal{M}, \mathcal{C}$ – множества ключей, открытых текстов, шифр-текстов

Шифр Шеннона –

это тройка функций $\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}$:

$$\mathsf{Enc} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$$
$$\mathsf{Enc}(k, m) = c$$

$$\mathsf{Dec} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$$
$$\mathsf{Dec}(k, c) = m,$$

для которых выполняется

$$\mathsf{Dec}(k, \mathsf{Enc}(k, m)) == m \qquad \forall k \leftarrow \mathsf{KeyGen}, m \in \mathcal{M}$$

Абсолютная криптографическая стойкость (perfect secrecy)

- На каждом из этих множеств зададим распределение: $\Pr[M = m]$ – вероятность выбора $m \in \mathcal{M}$.
- Аналогично для $K \in \mathcal{K}, C \in \mathcal{C}$.

Абсолютная криптографическая стойкость

Шифр-схема $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ обладает *абсолютной криптографической стойкостью*, если для любого распределения над $\mathcal{M}$

$$\Pr\left[M = m | C = c\right] = \Pr\left[M = m\right] \quad \forall m \in \mathcal{M}, c \in \mathcal{C}.$$

Интуиция: шифр-текст c не содержит никакой информации об открытом тексте m.

Эквивалентные определения

Шифр-схема $\Pi = (\mathsf{Enc}, \mathsf{Dec})$ определена над $\mathcal{K}, \mathcal{M}, \mathcal{C}$

1. Шифр-текст не зависит от открытого текста:

$$\Pr[C = c \mid M = m] = \Pr[C = c] \qquad \forall m \in \mathcal{M}, c \in \mathcal{C}.$$

2. Шифр-тексты не отличимы друг от друга: для любых $m_0, m_1 \in \mathcal{M}$ выполняется

$$\Pr[C = c \mid M = m_0] = \Pr[C = c \mid M = m_1]$$

Одноразовый блокнот (One-time pad) или шифр Вернама

Одноразовый блокнот

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

- $\mathsf{KeyGen}(1^\lambda) : k \leftarrow \{0,1\}^n$
- $\mathsf{Enc}(k, m \in \{0,1\}^n) : c = k \oplus m$
- $\mathsf{Dec}(k, c \in \{0,1\}^n) : m = k \oplus c$

Теорема. Одноразовый блокнот является абсолютно стойким.

Недостаток абсолютной стойкости

Теорема. Положим $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ – абсолютно стойкая шифр-схема. Тогда $|\mathcal{K}| \geq |\mathcal{M}|$

Интуиция: Абсолютно стойкие схемы неэффективны.

Теорема Шэннона (1949)

Положим $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ – шифр-схема с $|\mathcal{K}| = |\mathcal{M}| = |\mathcal{C}|$.
Тогда Π – абсолютно стойкая тогда и только тогда, когда

1. KeyGen выбирает $k \in \mathcal{K}$ с вероятностью $\frac{1}{|\mathcal{K}|}$ для всех k
2. $\forall m \in \mathcal{M}, c \in \mathcal{C}$ существует единственный $k \in \mathcal{K} : c = \mathsf{Enc}(k, m)$.

Одноразовый блокнот на практике

- Правительственная «горячая линия» между Вашингтоном и Москвой в 60-х
  `https://en.wikipedia.org/wiki/Moscow%E2%80%93Washington_hotline`
- Вьетнамские войны
  `https://eprint.iacr.org/2016/1136.pdf`

Часть IV

Семантическая стойкость

Информационно-теоретическая vs. семантическая стойкость

OTP	**Вычислительный шифр**
любые атакующие	вычислительно ограниченные атакующие
Большие ключи $\lvert\mathcal{K}\rvert = \lvert\mathcal{M}\rvert$	несколько сотен бит
Фиксированная длина m	Любая длина m

Семантическая безопасность: формальное определение

$$\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$$

Челленджер $\mathcal{C}$		Атакующий $\mathcal{A}$
$k \leftarrow \mathsf{KeyGen}(1^\lambda)$	$\xrightarrow{\lambda}$	
	$\xleftarrow{m_0, m_1 \in \mathcal{M}}$	$m_0, m_1 \leftarrow \mathcal{M}$
$b \xleftarrow{\$} \{0, 1\}$		
$c \leftarrow \mathsf{Enc}(k, m_b)$		
	$\xrightarrow{c}$	
	$\xleftarrow{\hat{b}}$	

$\mathtt{W}_{\Pi,\mathcal{A}}$ – событие $b == \hat{b}$.

$\mathtt{SSAdv} = \left|\Pr[\mathtt{W}_{\Pi,\mathcal{A}}] - \frac{1}{2}\right|$ -выигрыш $\mathcal{A}$

Схема Π – семантически безопасна, если для любого ppt $\mathcal{A}$:

$$\mathtt{SSAdv} = \mathrm{negl}(\lambda).$$

Семантическая безопасность OTP

Теорема. Для абсолютно стойкой схемы (OTP) и для всех атакующих $\mathcal{A}$ выполняется

$$\Pr[\mathtt{W}_{\Pi,\mathcal{A}}] = \frac{1}{2}.$$

Эквивалентно

$$\mathtt{SSAdv} = |\Pr[\mathtt{W}_{\Pi,\mathcal{A}}] - 1/2| = 0.$$

“Взлом” абсолютно стойкой схемы эквивалентен угадываю ключа.

Следствия семантической безопасности

Теорема.
$\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ – семантически стойкая схема. Тогда $\forall$ ppt атакующего $\mathcal{A}$ существует $\mathcal{A}'$:

$$|\Pr[\mathcal{A}(\lambda, \mathsf{Enc}(k, m)) \to f(m)] - \Pr[\mathcal{A}'(\lambda) \to f(m)]| \leq \mathrm{negl}(\lambda).$$

То есть семантически безопасная схема стойка к вычислению *любой эффективной* функции $f(m)$.

Часть V

Псевдослучайные генераторы

Одноразовый блокнот

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

- KeyGen$(1^\lambda) : k \leftarrow \{0,1\}^n$
- Enc$(k, m \in \{0,1\}^n) : c = k \oplus m$
- Dec$(k, c \in \{0,1\}^n) : m = k \oplus c$

Проблема: большие ключи.

Решение: “растянуть” случайные ℓ-бит в $L > \ell$ псевдо-случайных бит.

$$G : \{0,1\}^\ell \to \{0,1\}^L : \\ s \mapsto G(s)$$

Интуиция: ppt $\mathcal{A}$ не может отличить $G(s)$ от $r \xleftarrow{\$} \{0,1\}^L$.

Псевдослучайный генератор (PRG)

PRG G – эффективный детерминированный алгоритм, получающий на вход начальное значение (seed) $s \in \mathcal{S} = \{0,1\}^\ell$ и вычисляющий $r \in \mathcal{R} = \{0,1\}^L$

Эксперимент 0

$\mathcal{C}$

$s \leftarrow \mathcal{S}$

$r \leftarrow G(s)$

$\xrightarrow{r}$

$\xleftarrow{\hat{b}}$

Атакующий $\mathcal{A}$

Эксперимент 1:

$\mathcal{C}$

$r \xleftarrow{\$} \{0,1\}^L$

$\xrightarrow{r}$

$\xleftarrow{\hat{b}}$

Атакующий $\mathcal{A}$

W_b событие "$\mathcal{A}$ возвращает b"

Выигрыш $\mathcal{A}$'s : PRGadv $[\mathcal{A}, G] = |\Pr[W_0] - \Pr[W_1]|$.

PRG G – **безопасный**, если PRGadv $= \mathrm{negl}(\cdot)$ для всех ppt $\mathcal{A}$.

Атаки на PRG

$$G:\{0,1\}^\ell \to \{0,1\}^L:$$
$$s \mapsto G(s)$$

Существует атакующий $\mathcal{A}$ для G со сложностью 2^ℓ.

Кроме этого, Π **статистический тест** для $\{0,1\}^\ell$ – алгоритм A, выдающий 0 (= “не случайное”) или 1 (= “случайное”)

1. $A(x) = 1 \quad |\#0(x) - \#1(x)| \le 10\sqrt{n}$
2. $A(x) = 1 \quad \max \mathsf{len}\{1...1(x)\} \le 10 \log n$

Примеры реализаций тестов:

- Diehard tests
- TestU01
- Тесты NIST

Откуда берётся начальное значение s?

Действительно случайный бит дорог!

Начальное значение s – результат работы Random Number Generator (RNG).

Реализации RNG:

- Hardware Security Module (для серверов)
- Trusted Platform Module – процессор для генерации ключей
- Встроенные чипы в процессоры ("'Bull Mountain"' в Intel)
- Движения мыши, события клавиатуры
- Сетевые события, глитчи
- Неинициализированная память

Часть VI

Потоковый шифр

Потоковый шифр = OTP + PRG

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

$$G : \{0,1\}^\ell \to \{0,1\}^n - \text{ PRG}$$

- KeyGen(1^ℓ) :
  $s \xleftarrow{\$} \{0,1\}^\ell$
  $k = G(s)$
- Enc$(k, m \in \{0,1\}^n) : c = k \oplus m$
- Dec$(k, c \in \{0,1\}^n) : m = k \oplus c$

Безопасность потокового шифра

$$\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}) \text{ — потоковый шифр с PRG } G.$$

Теорема. G – криптографически безопасный псевдослучайный генератор $\implies$ Π – семантически стойкий шифр.

Для любого ppt $\mathcal{A}$, атакующего семантическую стойкость Π, найдется ppt алгоритм $\mathcal{B}$, атакующий G.

Челленджер $\mathcal{C}$

$s \xleftarrow{\$} \{0,1\}^\ell$

$k = G(s)$

либо

$k \xleftarrow{\$} \{0,1\}^n$

$\xrightarrow{k}$

$\mathcal{B}$

$b \xleftarrow{\$} \{0,1\}$

$c = \mathsf{Enc}(k, m_b)$

$b == \hat{b} \; ? \; 0 : 1$

$\xrightarrow{1^n}$

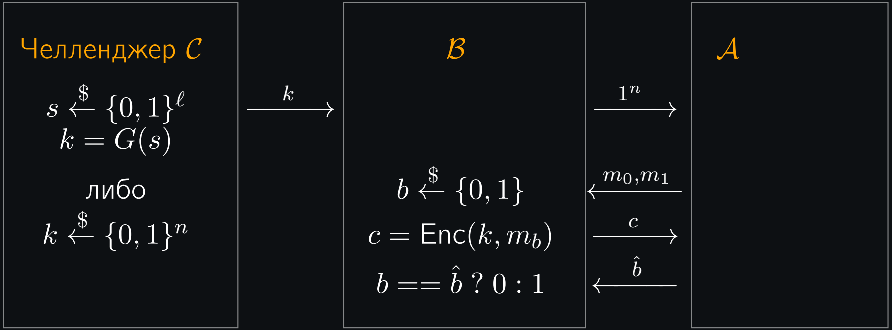
$\xleftarrow{m_0, m_1}$

$\xrightarrow{c}$

$\xleftarrow{\hat{b}}$

$\mathcal{A}$

Доказательство безопасности Π



1. Случай $k \xleftarrow{\$} \{0,1\}^n$: $\mathsf{Enc}(k, m_b)$ – OTP $\implies \Pr[W_0, \mathcal{B}] = \Pr[\mathtt{W}_{\Pi,\mathcal{A}}] = \frac{1}{2}$.
2. Случай $k = G(s)$: $\Pr[W_1, \mathcal{B}] = \Pr[\mathtt{W}_{\Pi,\mathcal{A}}] = 1/2 + \varepsilon(n)$.

В итоге,

$$\mathsf{PRGadv}\,[\mathcal{B}, G] = |\Pr[W_0, \mathcal{B}] - \Pr[W_1, \mathcal{B}]| = \varepsilon(n).$$

Потоковый шифр = OTP + PRG

$$\mathcal{M}, \mathcal{C} = \{0,1\}^n,\ \mathcal{K} = \{0,1\}^\ell$$
$$G : \{0,1\}^\ell \to \{0,1\}^n - \text{ PRG}$$

- KeyGen(1^ℓ) :
  $s \xleftarrow{\$} \{0,1\}^\ell$
  $k = G(s)$
- Enc$(k, m \in \{0,1\}^n) : c = k \oplus m$
- Dec$(k, c \in \{0,1\}^n) : m = k \oplus c$

Проблема: фиксированный размер открытых текстов.

Расширение области значений G

$$\text{Дано} \quad G : \{0,1\}^\ell \to \{0,1\}^n - \text{ PRG}$$
$$\text{Построить } G' : \{0,1\}^L \to \{0,1\}^N,\ N > n$$

Два метода композиции G:

1. Параллельная конструкция

$$G'(s_1, \ldots, s_n) = (G(s_1), \ldots, G(s_n))$$

2. Последовательная конструкция (Blum-Micali)

$G'(s) =$

$s_0 = s$

for $i = 1$ to k

$\quad (r_i, s_i) = G(s_{i-1})$

return $(r_1, \ldots, r_k, s_k)$

Современные PRG: Salsa and ChaCha

- Salsa20,ChaCha20: предложены Д.Бернштайном в 2005, 2008
- один из предложенных к использованию PRG в портфолио eStream
- используется в интернет протоколах (TLS)
- Вход: 256-битное нач. значение и параметр L
- Выход: $(256 \cdot L)$-битная псевдослучайная строка
- Детали алгоритма `https://cr.yp.to/chacha.html`

ChaCha PRG (упрощенная версия)

Два компонента:

1. функция $\texttt{pad}(s, j) : \{0,1\}^{256+64} \to \{0,1\}^{512}$
2. фиксированная перестановка $\pi : \{0,1\}^{512} \to \{0,1\}^{512}$

Алгоритм:

1. for $j = 0$ to $L - 1$
2. $\quad h_j = \texttt{pad}(s, j)$
3. $\quad r_j = \pi(h_j) \oplus h_j$
4. Выход $(r_0, \ldots r_{L-1})$

(Частично) Взломанные PRG

1. **Линейный конгруэнтный метод**
   - использовался в glibc, Microsoft Visual Basic, Java
   - печально известный RANDU
   - **не является криптографическим PRG!**

2. RC4
   - предложен Р.Ривестом в 1987
   - использовался TLS, 802.11b WEP
   - **не является криптографическим PRG!**

3. **Регистр сдвига с линейной обратной связью**
   - использовался для защиты данных DVD дисков
   - пример: Trivium (eStream)