Содержание

Крипта ИСП

Disclaymer: доверять этому конспекту или нет выбирайте сами

1. О криптографии

Определение 1.1: **Криптографические средства защиты информации (КСЗИ)** – основанные на математических методах преобразования защищаемой информации.

Определение 1.2: **Теоретическая криптография (математическая криптография, криптология)** – раздел дискретной математики, изучающий математические модели КСЗИ с научной точки зрения.

Основной предмет теоритической криптографии – криптографический протокол. (о нём в следующей главе).

Пример: **Криптографические примитивы**:
- **Односторонняя функция** – эффективно вычислимая функция, задача инвертирования которой вычислительно трудна.
- **Псевдослучайный генератор** – эффективный алгоритм, генерирующий длинные последовательности, которые никакой эффективный алгоритм не отличит от чисто случайных.
- **Криптографическая хэш-функция** – эффективно вычислимое семейство функций, уменьшающих длину аргумента, для которого задача поиска коллизий вычислительно трудна.

Определение 1.3: **Атака** – совокупность предположений о возможностях противника, о том, какие действия ему доступны (помимо вычислений).

Определение 1.4: **Угроза** – цель противника, состоящая в нарушении одного или нескольких из трёх условий (задач) криптографического протокола.

2. О криптографических протоколах

Определение 2.1: **Криптографический протокол** – это протокол, решающий хотя бы одну из трёх задач:
- Обеспечение **конфиденциальности** данных
- Обеспечение **целостности** сообщений и системы в целом – гарантия отсутствия нежелательных последствий вмешательства противника
- Обеспечение **неотслеживаемости** – невозможность установления противником, кто из участников выполнил определённое действие

Пример: **Прикладные КП**:
- Системы шифрования
- Подбрасование монеты по телефону
- Схемы электронной подписи
- Протоколы аутентификации
- Системы электронных платежей

Пример: **Примитивные КП**:
- bit-commitment (схема обязательства)
- oblivious transfer (протокол с забыванием)

Определение 2.2: **Стойкость** – формализация понятия качества криптографического протокола, его способность решать поставленную перед ним задачу.

Замечание 2.1: Стойкость определяется **только** для конкретной модели противника, состоящей из трёх основных компонентов:
- Вычислительные ресурсы (включая модель вычислений)
- Атака
- Угроза

3. О теории сложности

Замечание 3.1: Задача кодируется множеством строк в некотором конечном алфавите Σ, $|\Sigma| \geq 2$. Без ограничения общности, будем рассматривать только $\Sigma = \{0, 1\} = \mathbb{B}$.

Определение 3.1: Σ^* – множество всех слов в алфавите Σ, то есть $\Sigma^* = \cup_{n=0}^{\infty} \Sigma^n$.

Определение 3.2: **Язык** – некоторое множество слов, то есть подмножество в Σ^*.

Определение 3.3: Модель вычислений, которую мы будем использовать в дальнейшем – **машина Тьюринга**

$$M = (Q, q_0, q_f, \Sigma, b, \sigma)$$

где:

- Q – множество состояний (конечное, непустое)
- $q_0, q_f \in Q$ – выделенные состояния: начальнео и конечное
- Σ – конечный алфавит
- b – специальный «пустой символ»
- $\sigma : \Sigma \times Q \to \Sigma \times Q \times \{-1, 0, 1\}$ – функция перехода (частично определённая, в общем случае многозначная)

Определение 3.4: С машиной Тьюринга M связаны отображения:

- **Вычисляемая машиной функция** $M(\cdot) : \mathbb{B}^* \to B^* \cup \{\perp\}$, где $M(w)$ – выход машины M, если на вход подана строка w. (Выдаёт $\perp$ если вычисление не закончено)
- **Время её работы** $T_M(\cdot) : \mathbb{B}^* \to \mathbb{N} \cup \{\infty\}$, где $T_M(w)$ – число тактов работы машины M при вычислении на входе w.
- **Используемая ею память** $S_M(\cdot) : \mathbb{B}^* \to \mathbb{N} \cup \{\infty\}$, где $S_M(w)$ – число ячеек ленты, задействованных в вычислении на входе w.

Определение 3.5: Введём $\operatorname{poly}(x)$ – обозначение для «**некоторого полинома**» от переменной x. Важен не сам полином, факт его существования.

Определение 3.6: Введём названия для некоторых видов машин Тьюринга:

- **Детерминированная машина Тьюринга** – функция перехода σ однозначна
- **Полиномиальная (детерминированная) машина Тьюринга M** – обладает свойством:
$$\forall w \in \mathbb{B}^* : T_M(w) \leq \text{poly}(|w|)$$
- **Недетерминированная машина Тьюринга** – функция перехода σ, вообще говоря, многозначна, выбор её значений в конкретном вычислении осуществляется с помощью строки «Недетерминированного выбора» $\psi \in \mathbb{B}^\infty$, записанной на специальную ленту
- **Полиномиальная недетерминированная машина Тьюринга M** – обладает свойством:
$$\forall w \in \mathbb{B}^* : \forall \psi \in \mathbb{B}^* : T_M(\psi; w) \leq \text{poly}(|w|)$$
- **Вероятностная машина Тьюринга** – функция перехода σ принимает случайные значения, $M(w)$ – случайная величина (при фиксированном w). Выбор значения функции перехода в каждом такте осуществляется с помощью случайной строки $\rho \in \mathbb{B}^\infty$, записанной на специальную ленту.
- **Полиномиальная вероятностная машина Тьюринга (п.в.м.Т.) M** – обладает свойством:
$$\forall w \in \mathbb{B}^* : \forall \rho \in \mathbb{B}^\infty : T_M(\rho; w) \leq \text{poly}(|w|)$$
- **Полиномиальная в среднем вероятностная машина Тьюринга M** – обладает свойством:
$$\exists \varepsilon > 0 : \forall n \in \mathbb{N} : \forall \rho \in \mathbb{B}^\infty : \forall w \in \mathbb{B}^n : \mathbb{E}(T_M(\rho; w))^\varepsilon \leq n$$

Определение 3.7: Класс сложностей **Bounded-error Probabilistic Polynomial time**:
$$\text{BPP} = \left\{ L \subseteq \mathbb{B}^* \mid \exists \text{ п.в.м.Т. } M \left\{ \begin{smallmatrix} w \in L \ \Rightarrow \mu(\{M(w)=1\}) \geq \frac{2}{3} \\ w \notin\in L \Rightarrow \mu(\{M(w)=1\}) \leq \frac{1}{3} \end{smallmatrix} \right\} \right.$$

Определение 3.8: Класс сложностей **Randomzed Polynomial time**:
$$\text{RP} = \left\{ L \subseteq \mid \exists \text{ п.в.м.Т. M} \left\{ \begin{smallmatrix} w \in L \ \Rightarrow \mu(\{M(w)=1\}) \geq \frac{2}{3} \\ w \notin\in L \Rightarrow \mu(\{M(w)=1\}) = 0 \end{smallmatrix} \right\} \right\}$$

Определение 3.9: **Однородной моделью вычислителя противника** называется полиномиальная вероятностная машина Тьюринга или полиномальная в среднем вероятностная машина Тьюринга.

Определение 3.10: **Булевой схемой** называется отображение $C : \mathbb{B}^n \to \mathbb{B}^m$, такое, что для каждой координаты образа существует логическая функция от входа, тождественно задающая её.

Размером булевой схемы называется размерность её выхода.

Пример: Булевая схема $C : \mathbb{B}^1 \to \mathbb{B}^3$ имеет размер 3:
$$C(x_1) = (x_1, \neg x_1, x_1 \vee \neg x_1)$$

Определение 3.11: **Неоднородной моделью вычислителя противника** называется семейство булевых схем полиномиального размера $C = \{C_n\}_{n=1}^{\infty}$:
$$\forall n : |C_n| \leq \mathrm{poly}(n)$$
причём для каждого размера входа $|w|$ выбирается $C_{|w|}$ схема.

4. Об односторонних функциях

Определение 4.1: Функция $\nu : \mathbb{N} \to \mathbb{R}^+$ называется **пренебрежимо малой**, если
$$\forall \text{ полинома } p : \exists n_0 \in \mathbb{N} : \forall n \geq n_0 : \nu(n) \leq \tfrac{1}{p(n)}$$
Обозначение: $\mathrm{negl}(n)$.

Определение 4.2: Функция $f : X \to Y; X, Y \in\subseteq \mathbb{B}^*$ называется **полиномиально вычислимой**, если существует полиномиальная (детерминированная) машина Тьюринга M такая, что
$$\forall x \in X : M(x) = f(x)$$

Замечание 4.1:
- $\mathcal{U}$ – равномерное распределение вероятностей
- $x \underset{\mathcal{U}}{\in} Z$ значит, что x выбран случайно из множества Z в соответствии с равномерным распределением вероятностей на этом множестве
- $y \leftarrow M(x)$ значит, что y – случайный выход в.м.Т. M, на вход которой был подан X.
- Под возведением в степень 0 или 1 имеется в виду декартово умножение

Определение 4.3: Функция $f : \cup_{n \in \mathbb{N}} \mathbb{B}^n \to \mathbb{B}^*$ называется **сильно односторонней**, если
1. f полиномиально вычислима
2.

$$\forall \text{ п.в.м.Т. } A : \mu_{\substack{x \in \mathbb{B}^n \\ \mathcal{U}}}\big(\{A(1^n, f(x)) \in f^{-1}(f(x))\}\big) = \mathrm{negl}(n)$$

Определение 4.4: Функция $f : \cup_{n\in\mathbb{N}} \mathbb{B}^n \to \mathbb{B}^*$ называется **слабо односторонней**, если
1. f полиномиально вычислима
2.

$$\exists \text{ полином } p : \forall \text{ п.в.м.Т. } A : \exists n_0 \in \mathbb{N} : \forall n \geq n_0 :$$
$$\mu_{\underset{\mathcal{U}}{x\in\mathbb{B}^n}}\left(\left\{A(1^n, f(x)) \in f^{-1}(f(x))\right\}\right) \leq 1 - \frac{1}{p(n)}$$

Лемма 4.1: Любую полиномиально вычислимую, а значит и (сильно/слабо) одностороннюю функцию можно преобразовать так, чтобы она сохраняла длину аргумента.

Доказательство:
- Выберем какой-нибудь полином m, существующий в силу полиномиальной вычислимости функции f:

$$\forall x : |f(x)| \leq m(|x|)$$

это верно, так как машина Тьюринга совершит не более некоторого полиномиального числа тактов, а за такт она может прибавить максимум 1 к длине вывода.
- Определим функцию h на множестве $\cup_{n\in\mathbb{N}} \mathbb{B}^{m(n)+1}$, для чего представим каждый x из этого множества в виде $x = x'x''$, где $x' \in \mathbb{B}^n, x'' \in \mathbb{B}^{m(n)+1-n}$, и положим

$$h(x) = f(x') \times 1 \times 0^{m(|x'|)-|f(x')|}$$

Заметим, что вывод теперь имеет такую же длину, как и вход. (Почему нужно добавить единицу, а не все нули?) □

Теорема 4.1 (Яо): Если существует слабо односторонняя функция, то существует и сильно односторонняя функция.

Доказательство: Пусть f – слабо односторонняя функция, БОО предположим, что мы уже преобразовали её к виду, сохраняющему длину входа, то есть

$$\forall n \in \mathbb{N} : f(\mathbb{B}^n) \subseteq \mathbb{B}^n$$

Зафиксируем некоторый полином p из определения слабой односторонности.

Для любой п.в.м.Т. A и для всех достаточно больших n:

$$\mu_{\underset{\mathcal{U}}{x\in\mathbb{B}^n}}\left(\left\{A(1^n; f(x)) \in f^{-1}(f(x))\right\}\right) \leq 1 - \frac{1}{p(n)}$$

Введём функцию

$$g(x_1, ..., x_t) \coloneqq (f(x_1), ..., f(x_t)); \quad x_i \in B^n, t = n \cdot p(n)$$

Предположим, что g – не односторонняя, тогда для произвольного полинома q существует п.в.м.Т. B и бесконечное множество $N \subseteq \mathbb{N}$, что

$$\forall n \in N : \mu_{\underset{U}{x \in \mathbb{B}^{nt}}}\left(\left\{B(1^{nt}; g(x)) \in g^{-1}(g(x))\right\}\right) > \frac{1}{q(nt)}$$

Определим вероятностную машину C_0 на входе $y \in \mathbb{B}^n$:

1. for i in [1..t]
2. let z = $B\big(f(x_1), ..., f(x_{i-1}), y, f(x_{i+1}), ..., f(x_t)\big)$
3. if $f(z_i) = y$: return z_i

Также определим вероятностный алгоритм C на входе y, выполняющий алгоритм C_0 на этом входе не более $k \coloneqq 2 \cdot n \cdot t \cdot q(n \cdot t)$ раз.

Если на некоторой итерации алгоритм C_0 что-то вернул, то это будет результатом C, иначе C заканчивает работу без выходного значения.

Для произвольного $n \in \mathbb{N}$ положим

$$E_n = \left\{x \in \mathbb{B}^n \mid \mu\left(\left\{C_0(1^n; f(x)) \in f^{-1}(f(x))\right\}\right) > \frac{n}{k}\right\}$$

Лемма 4.2:

$$\forall n \in \mathbb{N} : \forall x \in E^n : \mu\left(\left\{C(1^n; f(x)) \in f^{-1}(f(x))\right\}\right) > 1 - e^{-n}$$

Эта лемма показывает, что ограниченная на E^n f является сильно односторонней.

Доказательство: Зная, что:

- C – применение алгоритма C_0 k раз, а значит если C не угадал прообраз, то и k раз применённый C_0 тоже не угадал. (Оценка вероятности)
- Мы взяли $x \in E_n$, в котором вероятность угадать прообраз алгоритмом $C_0 > \frac{n}{k}$, а значит вероятность не угадать $< 1 - \frac{n}{k}$
- $\forall r : \ln r \leq r - 1$

получим:

$$\mu\left(\left\{C(1^n; f(x)) \notin f^{-1}(f(x))\right\}\right) < \left(1 - \frac{n}{k}\right)^k = e^{k \ln\left(1 - \frac{n}{k}\right)} \leq e^{-n}$$

□

Лемма 4.3:

$$\exists N_0 \in \mathbb{N} : \forall n > N_0 : \mu(E_n) > 1 - \frac{1}{2p(n)}$$

Этой леммой мы хотим показать, что с какого-то момента E_n достаточно большое.

Доказательство: Пока скип, большое □

Из доказанных лемм вытекает, что

$$\mu\left(\left\{C(1^n; f(x)) \in f^{-1}(f(x))\right\}\right) \geq$$

$$\mu\left(\left\{A(1^n; f(x)) \in f^{-1}(f(x))\right\} \mid E_n\right)\mu(E_n) > (1 - e^{-n})\left(1 - \frac{1}{2p(n)}\right)$$

Но если вспомним, что f слабо односторонняя, то получим неравенство:

$$1 - \frac{1}{p(n)} > (1 - e^{-n})\left(1 - \frac{1}{2p(n)}\right)$$

Раскрыв скобки в правой части получим, что

$$\frac{1}{p(n)} < e^{-n} + \frac{1}{2p(n)} - \frac{e^{-n}}{2p(n)} < e^{-n} + \frac{1}{2p(n)}$$

Что неверно при достаточно больших n, так как e^{-n} убывает быстрее $\frac{1}{2p(n)}$.

5. О трудных предикатах

Определение 5.1: Функция $\mathbb{B}^* \to \mathbb{B}$ называется **трудным предикатом** для функции $f : \mathbb{B}^* \to \mathbb{B}^*$, если

- b – полиномиально вычислимая функция
- $\forall$ п.в.м.Т. $A : \mu_{x \underset{U}{\in} \mathbb{B}^n}(\{A(1^n; f(x)) = b(x)\}) < \frac{1}{2} + \operatorname{negl}(n)$

Теорема 5.1 (Гольдрайха-Левина): Пусть f – односторонняя функция, определённая всюду и сохраняющая длину, и пусть для всех $x, r \in \mathbb{B}^* : |x| = |r|$, определены функции

$$g(x, r) = (f(x), r) \quad b(x, r) = \oplus_{i=1}^{|x|} x^{[i]} r^{[i]}$$

Тогда b – трудный предикат для функции g.

Доказательство: Предположим, что b не является трудным предикатом для функции g.

Это значит, что существуют полиномиальный вероятностный алгоритм A, полином p и бесконечное множество $N \subseteq \mathbb{N} \setminus \{0\}$ такие, что

$$\forall n \in N : \varepsilon(n) = \mu(\{A(1^{2n}, f(x), r) = b(x, r)\}) - \frac{1}{2} > \frac{1}{p(n)}$$

Пусть $n \in N$ и $x \in \mathbb{B}^n$. Положим

$$t(n, x) = \mu(\{A(1^{2n}, f(x), r) = b(x, r)\}) \quad E_n = \left\{x \in \mathbb{B}^n \mid t(x) \geq \frac{1}{2} + \frac{\varepsilon(n)}{2}\right\}$$

Тогда, заметив, что

- $\mathbb{E}_x(t(n, x)) = \varepsilon(n) + \frac{1}{2}$ – по определению
- Можно применить неравенство Чебышёва, так как $\frac{1}{2} - \frac{\varepsilon(n)}{2} > 0$.

$$\mu\left(\left\{t(x) < \frac{1}{2} + \frac{\varepsilon(n)}{2}\right\}\right) = \mu\left(\left\{1 - t(x) > \frac{1}{2} - \frac{\varepsilon(n)}{2}\right\}\right) \leq$$

$$\frac{\mathbb{E}_x(1 - t(n, x))}{\frac{1}{2} - \frac{\varepsilon(n)}{2}} = \frac{\frac{1}{2} - \varepsilon(n)}{\frac{1}{2} - \frac{\varepsilon(n)}{2}} = 1 - \frac{\varepsilon(n)}{1 - \varepsilon(n)} < 1 - \varepsilon(n)$$

Воспользовавшись отрицанием обеих частей неравенства, получим

$$\mu(E_n) = \mu\left(\left\{t(x) \geq \frac{1}{2} + \frac{\varepsilon(n)}{2}\right\}\right) > \varepsilon(n) > \frac{1}{p(n)}$$

Для завершения доказательства теоремы достаточно построить полиномиальный вероятностный алгоритм B, определённый для всех n и на $f(E_n)$, такой, что

$$\mu(\{B(1^n, f(x)) = x\}) \geq \frac{1}{\operatorname{poly}(n)}$$

Тогда этой вероятностью мы сможем оценить снизу вероятность угадать прообраз f, что будет противоречить односторонности f.

Введём обозначение $e_i \in \mathbb{B}^n$ – вектор с единицей на i-м месте.

Алгоритм B на входе $(1^n, f(x))$, где $n \in N$ и $x \in E_n$, будет искать каждый бит $x^{[i]}$ отдельно. Для этого алгоритм B:

- Выбирает случайные элементы $r_1, ..., r_{\pi(n)} \underset{\mathcal{U}}{\in} \mathbb{B}^n$, где π – некоторый полиномиальный параметр на N, принимающий лишь нечётные значения.
- Для каждого $j \in \{1, ..., \pi(n)\}$ вычисляет биты β_j, ρ_j, являющиеся предполагаемыми значениями $b\big(x, r_j \oplus e_i\big)$ и $b\big(x, r_j\big)$ соответственно
- Выбирает в качестве предпологаемого значения $x^{[i]}$ бит, который встречается в последовательности $\beta_j \oplus \rho_j; j \in \{1, ..., \pi(n)\}$ более $\frac{\pi(n)}{2}$ раз

Очевидно, если $\beta_j = b\big(x, r_j \oplus e_i\big)$ и $\rho_j = b\big(x, r_j\big)$ для более чем половины индексов $j \in \{1, ..., \pi(n)\}$, то $x^{[i]}$ будет найден правильно, так как

$$b\big(x, r_j \oplus e_i\big) \oplus b\big(x, r_j\big) = b(x, e_i) = x^{[i]}$$

Бит β_j вычисляется как $A\big(1^{2n}, f(x), r_j \oplus e_i\big)$. Мы не получим нужную оценку вероятности успеха алгоритма B, если будем вычислять ρ_j как $A\big(1^{2n}, f(x), r_j\big)$. Вместо этого алгоритм пытается угадать значение $b\big(x, r_j\big)$ для всех j.

Но если просто выбрать $\rho_j \underset{\mathcal{U}}{\in} \mathbb{B}$, то вероятность того, что $\rho_j = b\big(x, r_j\big)$ для всех $j \in \{1, ..., \pi(n)\}$ будет равна $\frac{1}{2^{\pi(n)}}$, а эта величина при нужном для нас росте $\pi(n)$ будет пренебрежимо малой, как функция от n. Чтобы обойти это препятствие, алгоритм B делает некую грязь. $\square$