

Содержание

Лекция 1. Односторонние функции	3
1 Введение	3
2 Односторонние функции	3
Лекция 2. Слабо и сильно односторонние функции	4
3 Построение сильно односторонней функции из слабой	4
4 Примеры «односторонних» функций	5
5 Генераторы псевдослучайных чисел	6
Лекция 3. Односторонняя перестановка и генератор псевдослучайных чисел	7
6 XOR-лемма Яо	7
7 Построение генератора любой длины	8
Лекция 4. Псевдослучайные функции	9
8 Псевдослучайные функции с неадаптивным отличителем	9
9 Адаптивные отличители	10
Лекция 5. Шифрование с открытым и закрытым ключом	11
10 Принципиальная схема шифрования	11
11 Шифрование с закрытым ключом	11
12 Шифрование с открытым ключом	12
Лекция 6. Интерактивный протокол бросания монетки	13
13 Общая схема задачи	13
14 Неинтерактивный протокол привязки к биту	14
15 Интерактивный протокол привязки к биту	15
16 Протокол бросания монетки	16
Лекция 8. Электронная подпись I	16
17 Электронная подпись с закрытым ключом	16

18 Электронная подпись с открытым ключом	17
19 Снятие ограничений по длине и одноразовости	18

Лекция 1. Односторонние функции

1 Введение

5 миров Импальяццо.

- Алгоритмика ($\mathbf{P} = \mathbf{NP}$).
- Эвристика ($\mathbf{P} \neq \mathbf{NP}$, но есть быстрый алгоритм в среднем).
- Pessiland ($\mathbf{P} \neq \mathbf{NP}$, нет ни быстрых алгоритмов, ни односторонней функции).
- Миникрипт (есть односторонние функции, но нет односторонних функций с секретом).
- Криптомания (есть односторонние функции с секретом).

В принципе, может стать еще что-то странное навроде $\mathbf{P} = \mathbf{NP}$, но на практике эти алгоритмы очень долгие или $\mathbf{P} \neq \mathbf{NP}$, но наоборот, есть какие-либо квазиполиномиальные быстрые алгоритмы.

Сначала будут обсуждаться примитивы, односторонние функции, доказательства с нулевым разглашением и прочее, потом на базе этого покажем, как построить криптографические протоколы.

Литература: конспект лекций Верещагина «Лекции по математической криптографии», черновик, Goldreich «Foundations of Cryptography», конспекты Goldwasser.

2 Односторонние функции

В криптографических задачах полиномиальность будет считаться от параметра безопасности n (неформально, длина ключа) для доказательства надёжности, и от длины шифруемого сообщения при шифровании.

Определение 1. $\{f_n\}_{n=1}^{\infty}$ — семейство односторонних функций, если:

- f регулярны по длине: $f_n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{l(n)}$.
- f вычислимы за полиномиальное время.
- f труднообратима (4 варианта: в сильном/слабом смысле, против равномерного/неравномерного обратителя).

Обратимость в слабом смысле: вероятность неудачи обратителя больше, чем некоторый обратный полином.

В сильном смысле: вероятность успеха асимптотически меньше, чем любой обратный полином.

Равномерный обратитель — полиномиальный вероятностный алгоритм.

Неравномерный обратитель — семейство схем полиномиального размера.

Задача обращения: по $f(x)$ найти $x' : f(x') = f(x)$.

Кванторная запись определения труднообратимой функции в сильном смысле: $\forall p(\cdot) \forall \{R_n\}_{n=1}^{\infty} \exists N : \forall n > N \rightarrow P_{x \sim U_k(n)} \{f(R(f(x))) = f(x)\} < \frac{1}{p(n)}$.

R в определении пробегает по всем семействам схем или вероятностным обратителям в зависимости от вида обратителя. Определение в слабом смысле отличается первым квантором.

Задача. Может ли семейство $f_n : |\text{Im} f_n| = \text{poly}(n) = s(n)$ быть труднообратимым в слабом смысле?

Неравномерный обратитель: можно «защитить» в схему по одному прообразу от каждого класса.

Равномерный: берёт случайный x , вычисляет $f(x)$, если $y = f(x)$, вернуть x . Можно подобрать такое число повторений N , чтобы вероятность ошибки была мала. Идея: классы бывают большие (размера $> 2^{l(n)} \cdot \varepsilon$), и маленькие. Вероятность неуспеха для больших классов не больше $(1-\varepsilon)^N$, а для маленького класса можно оценить единицей. Тогда общая вероятность ошибки для случайного x не больше $s(n) \cdot \varepsilon + (1-\varepsilon)^N$. Если ε взять как $\frac{1}{2s(n)q(n)}$, а $N = \frac{n}{\varepsilon}$, то сумма будет не больше $\frac{1}{q(n)}$ для любого полинома $q(n)$.

Задача. f — односторонняя функция. Верно ли, что $g(x) = f(x)f(x)$ тоже односторонняя? Верно ли, что $h(xy) = f(x)f(y)$ будет односторонней?

Если g односторонняя, то $\exists R_g$, которая обращает g . Тогда $R_f(y) = R_g(y)$ обращает f .

Если R_h обращает h , то можно построить такой обратитель f : берём случайный y , считаем $f(y)$ и возвращаем первую часть $R_h(f(x)f(y))$, если всё нормально, иначе нужно повторить процедуру.

Хорошие значения x — это те, для которых доля пар (x, y) больше или равна ε . Остальных значений x мало. Аналогичными предыдущей задаче рассуждениями можно получить оценку.

Лекция 2. Слабо и сильно односторонние функции

3 Построение сильно односторонней функции из слабой

Напоминание:

Определение 1. Слабо односторонняя функция $f(x)$ — это такая, что $\exists p(x) \geq 0 \forall \{C_n\}_{n=1}^{\infty} \exists N \forall n \geq N \rightarrow P(f(C_n(f(n))) = f(x)) < 1 - \frac{1}{p(n)}$, где C_n — семейство схем полиномиального размера, а $f(x)$ вычислима за полиномиальное время.

Определение 2. Сильно односторонняя функция $f(x)$ — это такая, что $\forall p(x) \geq 0 \forall \{C_n\}_{n=1}^{\infty} \exists N \forall n \geq N \rightarrow P(f(C_n(f(n))) = f(x)) < \frac{1}{p(n)}$.

Теорема 1. Если существует слабо односторонняя функция, то существует и сильно односторонняя.

Доказательство. Рассмотрим функцию $F(x_1, \dots, x_N) = (f(x_1), \dots, f(x_N))$. Ясно, что такая функция защищена от наивных обратителей, которые пытаются обратить каждую компоненту по отдельности. Однако, неясно, почему не существует более сложного и более эффективного обратителя.

Поэтому мы возьмем гипотетический обратитель R_F для F в обратитель R_f для f . Обратитель $R_f(y) = R_F(y, f(x_2), \dots, f(x_N))$ $|_1$ может не преуспеть, так как при фиксированной первой компоненте вероятность успеха может быть мала. Однако, мы можем запускать обратитель R_F много раз, поэтому сделаем так:

```
for (int i = 0; i < n; ++i) {
    for (int j = 0; j < K; ++j) {
        x_1, ..., x_n = gen_random(); // except i
        X = R_F(f(x_1), ..., y, ..., f(x_n)); // except i
        if (f(X) == y) {
            return X;
        }
    }
}
```

Для всех $i = 1, \dots, n$ K раз выберем случайные $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n$, и запустим $R_F(f(x_1), \dots, f(x_{i-1}), y, f(x_{i+1}), \dots, f(x_n))$ и выберем i -ю компоненту x . Если $f(x) = y$, вернем x .

Пусть $\rho_i(x) = P\{f(R_F(\dots)) = f(x)\}$, $\rho_{\max}(x) = \max_{i=1, \dots, N} \rho_i(x)$. x бывает двух видов: такой, что $\rho_{\max}(x) \geq \varepsilon$ и такой, что $\rho_{\max} < \varepsilon$, доля последних равна δ .

Вероятность неудачи в таком случае $R_f \leq \delta + (1 - \varepsilon)^K$. Если F — не сильно односторонняя, то вероятность успеха $R_F > \frac{1}{q(n)}$.

Для (x_1, \dots, x_n) вероятность, что все x_i хорошие $\leq (1 - \delta)^N$, а если хотя бы один x плохой, то условная вероятность обращения $R_F < \varepsilon$.

Тогда вероятность успеха $\frac{1}{q(n)} < R_F < \varepsilon + (1 - \delta)^N$. При $\varepsilon = \frac{1}{2q(n)}$ получается, что $(1 - \delta)^N > \frac{1}{2q(n)}$. При $N = np(n) \Rightarrow \delta < \frac{1}{2p(n)}$. За счёт выбора K можем сделать $K = nq(n)$ и тогда $(1 - \varepsilon)^K < \frac{1}{2p(n)}$. В итоге $\delta + (1 - \varepsilon)^K < \frac{1}{p(n)}$, что означает, что f не слабо односторонняя. \square

4 Примеры «односторонних» функций

Функция Рабина: $(x, y) \mapsto (x^2 \bmod y, y)$. $y = p \cdot q$, $0 \leq x < y$, притом p, q — простые числа вида $4k + 3$.

Функция RSA: $(x, y, z) \mapsto (x^z \bmod y, y, z)$.

$P\{f(R_n(f(x))) = f(x)\}$ определяется по всем $x \in D_n$, притом требование к области D_n таково, что нужно уметь порождать случайные элементы D_n , то есть существует полиномиальный вероятностный алгоритм, порождающий случайную величину, статистически близкую к равномерной на D_n (расстояние между любыми двумя событиями меньше любого обратного полинома).

Можно отметить, что у функции Рабина, например, есть так называемый «секрет» (разложение $y = p \cdot q$), благодаря которому можно расшифровать сообщение. Более формально определим

Определение 3. Семейство односторонних функций с секретом $\{f_\alpha\}_{\alpha \in A}$: $f_\alpha : D_\alpha \rightarrow R_\alpha$ это такие функции, что существуют 4 алгоритма:

- Генератор: $1^n \rightarrow (\alpha, \tau)$, генерирует ключ и секрет.
- Сэмплер: $\alpha \mapsto$ случайный элемент D_α (с точностью до статистической близости).
- Вычислитель: $(\alpha, x) \mapsto f_\alpha(x)$.
- Обратитель: $(\alpha, \tau, y) \mapsto f_\alpha^{-1}(y)$.

Притом $(\alpha, y) \mapsto f_\alpha^{-1}(y)$ труднообратимо в обычном смысле.

Улучшенная односторонняя перестановка с секретом: y выбирается как случайный элемент D_α , а обратитель помимо α и y получает случайные биты, использованные при порождении y (при этом они все равно ему не помогают).

5 Генераторы псевдослучайных чисел

Определение 4. G — генератор псевдослучайных чисел, если

- $G : \{0, 1\}^n \rightarrow \{0, 1\}^{p(n)}$.
- G вычислима за полином.
- $\forall \{D_n\}_{n=1}^\infty \forall q(\cdot) \rightarrow \exists N : \forall n > N \rightarrow |P_{x \sim U_n}(D_n(G(x)) = 1) - P_{y \sim U_{p(n)}}(D_n(y) = 1)| < \frac{1}{q(n)}$.

Ясно, что генератор должен быть односторонней функцией, так как иначе обратитель мог бы отличить вывод генератора от случайного вывода.

Теорема 2. Если существует односторонняя функция, то существует и генератор.

Мы докажем ослабленную версию этой теоремы:

Теорема 3. Если существует односторонняя перестановка, то существует и генератор.

Определение 5. Трудный бит. Схематически: $x \mapsto f(x), x \mapsto b(x)$ вычисляются легко, $|b(x)| = 1$. При этом по $f(x)$ сложно вычислить $b(x)$:

$$\forall q(\cdot) \forall \{P_n\}_{n=1}^\infty \exists N \forall n > N |P(P_n(f(x)) = b(x)) - \frac{1}{2}| < \frac{1}{q(n)}.$$

Схема доказательства теоремы такая:

- Односторонняя перестановка $f \mapsto$ односторонняя перестановка с трудным битом: $g(x, y) = (f(x), y), b(x, y) = x \odot y = \bigoplus_{i=1}^n x_i y_i$.
- Генератор $n \rightarrow n + 1$: $G(x) = g(x)b(x)$.
- Генератор $n \rightarrow p(n)$: $g(g(x))b(g(x))b(x), g(g(g(x)))b(g(g(x)))b(g(x))b(x), \dots$

Лекция 3. Односторонняя перестановка и генератор псевдослучайных чисел

6 XOR-лемма Яо

Теорема 1. Если существует односторонняя перестановка $p : D_n \rightarrow D_n, D_n \subset \{0, 1\}^{k(n)}$, то существует генератор псевдослучайных чисел.

Доказательство. Напоминание: схема доказательства теоремы:

- Односторонняя перестановка $f \mapsto$ односторонняя перестановка с трудным битом (декодирование списком кода Адамара и дерандомизации с помощью попарной независимости).
- Генератор $n \rightarrow n + 1$: $G(x) = g(x)b(x)$ (XOR-лемма Яо).
- Генератор $n \rightarrow p(n)$: $g(g(x))b(g(x))b(x), g(g(g(x)))b(g(g(x)))b(g(x))b(x), \dots$ (hybrid argument).

Сначала сделаем второй шаг.

Определение 1. $b(x)$ — трудный бит для $f(x)$, если он полиномиально вычислим и $\forall p(\cdot) \forall \{P_n\}_{n=1}^\infty \exists N : \forall n \geq N \rightarrow |P(P_n(f(x)) = b(x)) - \frac{1}{2}| < \frac{1}{p(n)}$.

Лемма. $b(x)$ — трудный бит для $f(x) \Rightarrow G(x) = f(x)b(x)$ — генератор псевдослучайных чисел.

Доказательство. Если существует отличитель для $G(x)$, то $\exists s(\cdot) \exists \{D_n\}_{n=1}^\infty \forall N \exists n > N : |P_x(D_n(f(x)b(x)) = 1) - P_y(D_n(y) = 1)| \geq \frac{1}{s(n)}$. Можно считать, что выражение под модулем положительно, так как для тех n , для которых это не так, можно инвертировать вывод D_n .

Рассмотрим варианты для $D(f(x)0) = \alpha, D(f(x)1) = \beta$.

- $\alpha = \beta \Rightarrow$ значение предсказателя случайно.
- $\alpha = 0, \beta = 1 \Rightarrow$ предсказатель возвращает 1.

- $\alpha = 1, \beta = 0 \Rightarrow$ предсказатель возвращает 0.

Обозначим A, B, C, D — события для 00, 01, 10, 11 соответственно. $A_0, A_1 \subset AB_0, B_1 \subset B \dots$ разбиения по значениям трудного бита, a_0, a_1, \dots — их вероятности.

$$P(D_n(f(x)b(x)) = 1) = b_1 + c_0 + d_0 + d_1,$$

$$P(D_n(y) = 1) = \frac{b_0+b_1}{2} + \frac{c_0+c_1}{2} + d_0 + d_1.$$

$$\text{Тогда разность } \Delta = \frac{b_0+b_1}{2} + \frac{c_0+c_1}{2} \geq \frac{1}{s(n)}.$$

$$\text{Успех предсказателя: } \frac{a_0+a_1}{2} + b_1 + c_0 + \frac{d_0+d_1}{2} = \frac{1}{2} + \Delta \geq \frac{1}{2} + \frac{1}{s(n)}. \quad \square$$

Почему XOR-лемма? Потому что $P(f(x)) = D(f(x)r) \oplus r \oplus 1$.

7 Построение генератора любой длины

Теперь сделаем генератор $n \rightarrow q(n)$. Для начала рассмотрим $G(x) = f(f(x))b(f(x))b(x)$, что должно быть вычислительно неотличимо от xr_1r_2 .

$xr_1r_2 \sim f(x)r_1r_2$, так как x и $f(x)$ одинаково распределены (так как f — перестановка). $f(x)r_1r_2 \sim f(x)b(x)r_2$ по определению G . Далее, $xr_2 \sim f(x)b(x) \Rightarrow xr_1r_2 \sim f(f(x))b(f(x))b(x)$.

Для любого константного увеличения можно сделать точно также. Для $n \rightarrow q(n)$ делаем так:

$$\begin{aligned} h_0(x) &= xr_1r_2 \dots r_{q(n)} \\ &\vdots \\ h_{q(n)}(x) &= f^{q(n)}(x)b(f^{q(n)-1}(x)) \dots b(x) \end{aligned}$$

Хотим доказать, что $h_{q(n)} \sim h_0(x)$. Если $P(D_n(h_{q(n)}(x)) = 1) - P(D_n(h_0(x)) = 1) \geq \frac{1}{s(n)}$, то $\exists m : P(D_n(h_m(x)) = 1) - P(D_n(h_{m-1}(x)) = 1) \geq \frac{1}{s(n)q(n)}$, что невозможно аналогично пункту $n \rightarrow n+2$.

Теорема 2 (Левин-Голдрайх). Пусть f — односторонняя перестановка, то $g(xy) = f(x)y$ тоже односторонняя перестановка, а $b(xy) = x \odot y = \bigoplus_{i=1}^n x_i y_i$ — трудный бит для g .

Доказательство. Первая часть очевидна, если f — односторонняя перестановка, то и g тоже перестановка, легко вычисляется и если g можно обратить, то обратить можно и f . Для доказательства второй части воспользуемся кодом Адамара.

Код Адамара: $x \mapsto (x \odot z)_{z \in \{0,1\}^n}$ слово длины n превращает в слово длины 2^n . Его можно воспринимать как значение всех линейных функций на входе x или как значение на всех входах линейной функции, заданной x .

Пусть $\hat{f}(z)$ совпадает с $f(z)$ на доле входов z равной $\frac{3}{4} + \varepsilon$. Тогда можно восстановить $f(z) = \hat{f}(z+r) + \hat{f}(r)$ и с вероятностью $> \frac{1}{2}$ мы восстановим $f(z)$. Повторив много раз, можем узнать $f(e_i) = x_i$.

Для доли повреждения $\frac{1}{2}$ декодировать уже не получится, но можно декодировать списком: имея доступ к $\hat{f}(z)$ как к оракулу, напечатать полиномиальный список в котором с вероятностью $\geq \frac{1}{2}$ находится вектор x , определяющий f .

Задача. В шаре с центром в любой точке и радиусом (в смысле расстояния Хемминга) $\frac{1}{2} - \varepsilon$ находится $\text{poly}\left(\frac{1}{\varepsilon}\right)$ кодовых слов.

Запишем равенство: $f(z) = \hat{f}(z + r) + f(r)$, которое должно быть выполнено в $\geq \frac{1}{2}$ случаев. Непонятно только, откуда взять $f(r)$.

Идея попарной независимости: проведем процедуру выше для некоторого числа попарно независимых случайных r . Возьмем случайные независимые в совокупности вектора u_1, \dots, u_l и вектора r_1, \dots, r_{2^l-1} построим как $r_a = a_1 u_1 + \dots + a_l u_l$. Тогда r_1, \dots, r_{2^l-1} попарно независимы. Алгоритм будет следующий:

```
u_1, ..., u_l := random()
for (f(u_1), ... f(u_l) in {{0,1}^n}^l) {
  for (int a = 1; a < 2^l - 1; ++a) {
    f(r_a) = a_1 f(u_1) + ... + a_l f(u_l) // linearity
    f(e_i) = f_hat(e_i + r_a) + f(r_a)
  }
  choose f(e_i) as majority for all a
  add f(e_1), ..., f(e_l) in list
}
```

Утверждается, что по неравенству Чебышёва при большом числе повторений с вероятностью больше, чем $\frac{1}{2}$ декодирование произведено верно.

Теперь, если g — это односторонняя функция, $h(xy) = g(x)y, b(xy) = x \odot y = f(y)$ и есть предсказатель b , то можно с его помощью построить $\hat{f}(y)$, совпадающую на доле $\frac{1}{2} + \varepsilon$, что можно декодировать списком x_1, \dots, x_m и каждый x проверить непосредственно.

Несмотря на то, что \hat{f} экспоненциально длинная, но нам нужно только значение в полиноме точек, которые мы и запомним (или можно относиться к \hat{f} как к оракулу).

□

□

Лекция 4. Псевдослучайные функции

8 Псевдослучайные функции с неадаптивным отличителем

Определение 1. Семейство функций $\{f_s^n\} : f_s^n : \{0,1\}^n \rightarrow \{0,1\}^n, s \in \{0,1\}^{p(n)}$ называется псевдослучайным, если:

- Существует полиномиальный алгоритм, который по s и x вычисляет $f_s^n(x)$.
- Надёжность против неадаптивных отличителей:

$$\forall q(\cdot) \forall \{D_n\} \forall \{x_1, \dots, x_{q(n)}\} \forall w(\cdot) \exists N : \forall n > N \rightarrow$$

$$|P_s(D_n(x_1, \dots, x_{q(n)}, f_s(x_1), \dots, f_s(x_{q(n)})) = 1) -$$

$$P_g(D_n(x_1, \dots, x_{q(n)}, g(x_1), \dots, g(x_{q(n)})) = 1)| < \frac{1}{w(n)}$$

Теорема 1. Если существует генератор псевдослучайных чисел из $\{0, 1\}^n$ в $\{0, 1\}^{2n}$, то существует и семейство псевдослучайных функций.

Доказательство. Конструкция такова: для некоторого x длины n делаем следующее:

- Считаем $G(s) = s_0 s_1$, если 1й бит x равен 1, то берем s_1 , иначе s_0 .
- Считаем $G(s_{x_1}) = s_{x_1 0} s_{x_1 1}$, выбираем одну из половин в зависимости от x_2 .
- Продолжаем аналогично.

Доказательство индукцией по дереву: нарисует бинарное дерево, которое является частью полного бинарного, содержащей $x_1, \dots, x_{q(n)}$. На каждом следующем уровне мы имеем s_{a_1}, \dots, s_{a_r} , однако в силу псевдослучайности мы можем вычислительно неотличимо заменить их на действительно случайные значения. Поскольку размер дерева полиномиален, то мы использовали вычислительную неотличимость полиномиально много раз, что делать можно.

Более формально: если $G(y) = G_0(y)G_1(y)$, то можно записать $f_s(x) = G_{x_n}(G_{x_{n-1}}(\dots G_{x_1}(s) \dots))$.
 $h_{i,t}(x) = G_{x_n}(G_{x_{n-1}}(\dots G_{x_{i+1}}(t_{x_1 \dots x_i}) \dots))$, $|t| = 2^{n+i}$.
 $h_{0,t}(x) = f_t(x)$, $h_{n,t}(x)$ — случайная функция. Цепочка эквивалентностей приводит к тому, что они вычислительно неотличимы. \square

9 Адаптивные отличители

Пример 1. Пример, когда адаптивный отличитель сильнее неадаптивного: пусть есть f , такая что:

$f(0 \dots 0) = v$ — случайное, $f(v) = 0 \dots 0$, все остальные слова случайны.

Адаптивный отличитель легко справится с такой задачей, а для неадаптивного отличителя вероятность найти нужное значение v очень мала.

Мы воспользуемся тем, что адаптивные алгоритмы — это то же самое, что алгоритмы с подсказкой и доступом к оракулу-функции. Алгоритм получает на вход 1^n и подсказку a_n длины $\text{poly}(n)$.

$A^g(1^n, a_n)$ — это результат работы такого алгоритма с функцией g в качестве оракула.

Определение 2. Систему псевдослучайных функций будем называть устойчивой относительно адаптивного отличителя, если $\forall A$ — отличителя $\forall q(\cdot) \exists N : \forall n > N \rightarrow |P_s(A^{f_s}(1^n, a_n) = 1) - P_g(A^g(1^n, a_n) = 1)| < \frac{1}{q(n)}$.

Утверждение 1. Построенная система функций устойчива относительно адаптивного отличителя.

Доказательство. Доказательство в целом такое же, только одного общего дерева нет, оно строится по ходу алгоритма. Однако в ходе рассуждений ничего особо не меняется. \square

Вариации с параметрами могут быть следующие:

- Уменьшение длины - легко, если уменьшить длину выхода, ничего не нарушится.
- $f_s : \{0, 1\}^* \rightarrow \{0, 1\}^{r(n)}, s \in \{0, 1\}^{p(n)}$. Используется генератор $G : \{0, 1\}^{p(n)} \rightarrow \{0, 1\}^{2p(n)+r(n)}$, $G(s) = \underbrace{G_0(s)}_{p(n)} \underbrace{G_1(s)}_{p(n)} \underbrace{G_2(s)}_{r(n)}$, а функции вычисляются так: $f_s(x) = G_2(G_{x_k}(G_{x_{k-1}}(\dots)))$.

Лекция 5. Шифрование с открытым и закрытым ключом

10 Принципиальная схема шифрования

Пока что рассмотрим только задачи одноразового шифрования.

Шифрование с закрытым ключом: есть $Encoder(m, d)$, который передает сообщение s полиномиальной длины $Decoder(d, c) \rightarrow m$. Нужно чтобы перехватчик $A(c)$ не мог восстановить m .

Шифрование с открытым ключом: $Encoder(m, e)$ передает s программе $Decoder(c, d) \rightarrow m$. Ключи e, d у них разные, и перехватчик $A(c, d)$ может пользоваться одним из них.

11 Шифрование с закрытым ключом

Более формально, есть полиномиальные алгоритмы, G — генератор ключей, E — шифратор, D — дешифратор с понятными условиями:

- Корректность — $P(D(d, E(d, m)) = m) = 1$.
- Надёжность — $E(d, m_1) \sim E(d, m_2)$ (вычислительно не отличимы) для $m_1 \neq m_2$ или, что тоже самое $E(d, m_1) \sim E(d, 0 \dots 0)$.

Для закрытого ключа есть идеальная, но довольно бесполезная процедура: передать $m \oplus d$, где d — случайная строка. Есть две проблемы: ключ по длине равен сообщению (если мы можем обмениваться такими ключами, то почему не можем обмениваться сообщениями?), но даже если предположить, что мы заранее договорились о закрытом ключе, то остается проблема того, что шифр одноразовый: если известно $m_1 \oplus d$ и $m_2 \oplus d$, то можно узнать $m_1 \oplus m_2$, что может быть полезной информацией.

Теорема 1. *Если существует генератор псевдослучайных чисел, то существует и схема шифрования с закрытым ключом для сообщений полиномиальной длины.*

Доказательство. Если параметр безопасности равен длине ключа, то схема описана выше.

Вторая идея состоит в том, чтобы шифровать «раздутым» ключом $c = m \oplus G(d)$. Корректность очевидна, надёжность следует из того, что $G(d) \sim r$, где r — случайная строка. \square

Для многократовой схемы нужно немного исправить условия:

- Корректность — $P(D(d, E(d, m)) = m) = 1$.
- $\forall m_1, \dots, m_k, m'_1, \dots, m'_k \rightarrow (E(d, m_1), \dots, E(d, m_k)) \sim (E(d, m'_1), \dots, E(d, m'_k))$.

Теорема 2. *Если существует семейство псевдослучайных функций, то существует схема многократового шифрования с закрытым ключом.*

Доказательство. Закрытый ключ d — индекс случайной функции из семейства.

E выбирает случайный аргумент z и посылает $c = (m \oplus f_d(z), z)$, $D(d, x, z) = x \oplus f_d(z)$. Корректность схемы очевидна.

Надёжность: с большой вероятностью все z_i различны. Тогда значения $f_d(z_1), \dots, f_d(z_k)$ вычислительно неотличимы от r_1, \dots, r_k и всё хорошо. \square

12 Шифрование с открытым ключом

Схема шифрования с открытым ключом подразумевает, что есть полиномиальные алгоритмы, K — генератор ключей, E — шифратор, D — дешифратор со следующими условиями:

- Корректность — $P(D(d, E(e, m)) = m) = 1$.
- Надёжность — $(E(e, m_1), e) \sim (E(e, m_2), e)$ для одноразовой схемы и аналогичное условие для многих перехваченных сообщений.

Определение 1. Проверяемая односторонняя перестановка с секретом — некоторое семейство перестановок (на разных областях определения), для которой:

- $\exists G$ — генератор (α, τ) .
- $\exists f_\alpha : D_\alpha \leftrightarrow D_\alpha$.
- $\exists S$ — сэмплер для почти равномерного распределения на D_α .
- $\exists \text{Forwarder } F : (\alpha, x \in D_\alpha) \mapsto f_\alpha(x)$.
- $\exists \text{Backwarder } B : (\alpha, \tau, y \in D_\alpha) \mapsto f_\alpha^{-1}(y)$.
- Любой обратитель обращает $(\alpha, y) \mapsto f_\alpha^{-1}(y)$ с вероятностью ≈ 0 .

Теорема 3. \exists односторонняя перестановка \Rightarrow односторонняя перестановка с секретом и трудным битом $(h_\alpha(x))$, такой что по $f_\alpha(x)$ и α трудно восстановить $h_\alpha(x) : (f_\alpha(x), \alpha, h_\alpha(x)) \sim (f_\alpha(x), \alpha b)$, b — случайный).

Доказательство. Аналогично теореме в предыдущих лекциях. \square

Шифрование одного бита: $m \mapsto (m \oplus h_\alpha(x), f_\alpha(x))$, $x \in_R D_\alpha$. Декодер, зная секрет τ , восстановит $x, h_\alpha(x)$ и расшифрует с вероятностью 1.

Надёжность: $(\alpha, f_\alpha(x), h_\alpha(x) \oplus m) \sim (\alpha, f_\alpha(x), b \oplus m) \sim (\alpha, f_\alpha(x), b)$ вне зависимости от m .

Много бит можно шифровать, генерируя каждый раз новый случайный x . Таким образом, обобщаем в теорему:

Теорема 4. Если существует односторонняя перестановка с секретом, то существует схема шифрования с открытым ключом.

Лекция 6. Интерактивный протокол бросания монетки

13 Общая схема задачи

Алиса и Боб ненавидят друг друга. Необходимо получить общий случайный бит в условиях полного недоверия. A и B представляют собой два рандомизированных полиномиальных алгоритма с независимыми случайными битами. Они общаются по протоколу и после завершения выдают по одному биту σ, τ .

Нужно, чтобы оказалось так, чтобы $\sigma = \tau$, $P(\sigma = 0) \approx \frac{1}{2}$. Для этого есть полиномиальный алгоритм J , который получает протокол и возвращает A, B, \perp_A, \perp_B — либо сторону-победителя, либо сторону, которая первая нарушила протокол.

Требуемые свойства:

- Корректность: если обе стороны используют предписанные алгоритмы, то $P(J = A) = P(J = B) = \frac{1}{2}$.
- Интересы Алисы: $\forall B^* \rightarrow P(J \in \{0, \perp_B\}) \approx \frac{1}{2}$.

- Интересы Боба: $\forall A^* \rightarrow P(J \in \{1, \perp_A\}) \approx \frac{1}{2}$.

Если бы можно было обмениваться сообщениями одновременно, то можно было бы каждому послать случайный бит и в качестве результата взять их \oplus . Однако, одновременных сообщений не предусмотрено, поэтому используется привязка к биту (bit commitment). Такие протоколы неформально «запечатывают» бит в конверт так, чтобы его уже нельзя было подменить, но и нельзя посмотреть, не вскрыв конверт.

14 Неинтерактивный протокол привязки к биту

Неинтерактивная версия протокола подразумевает следующее: σ — запечатанный бит, r — случайные биты, $c(\sigma, r)$ — привязка, $k(\sigma, r)$ — ключ, $d(c, k) \in \{0, 1, \perp\}$ — процедура вскрытия (все алгоритмы полиномиальны и детерминированы). Условия на протокол следующие:

- Корректность: $d(c(\sigma, r), k(\sigma, r)) = \sigma$.
- Неразглашение: $c(0, r) \sim c(1, r)$.
- Неподменяемость: $\nexists c, k_0, k_1 : d(c, k_0) = 0, d(c, k_1) = 1$.

Замечание. В условии неразглашения требовать статистическую неотличимость не получится, так как третье условие говорит о том, что привязка расширяется однозначно либо в 1, либо в 0, то есть $c(0, r)$ и $c(1, r)$ вообще распределены на разных множествах. Поэтому требуется вычислительная неотличимость, добиться которой можно.

Функция привязки строится на базе односторонней перестановки f , из которой по теореме Левина-Голдрайха получается односторонняя перестановка g с трудным битом h . А именно $c(\sigma, r) = (\sigma \oplus h(r), g(r))$.

Можно считать, что $k(\sigma, r) = \sigma r$, так как распаковщик сам может провести все нужные вычисления. Каноническая процедура вскрытия может просто вычислять $c(0, r)$, $c(1, r)$, сравнивать его с c и возвращать 0, 1 или \perp в зависимости от результата. То есть, формально $k(\sigma, r) = (\sigma, r)$, $d((\tau, s); (\sigma, r)) = \tau \oplus h(r)$ если $g(r) = s$ и $\sigma = h(r)$, иначе \perp . Корректность такого алгоритма очевидна.

Трудность бита h означает, что $(g(r), h(r)) \sim (g(r), \gamma)$, где γ — случайный бит. Отсюда $(g(r), 1 \oplus h(r)) \sim (g(r), 1 \oplus \gamma) \sim (g(r), \gamma)$, то есть $(g(r), h(r)) \sim (g(r), 1 \oplus h(r))$, откуда следует неразглашение.

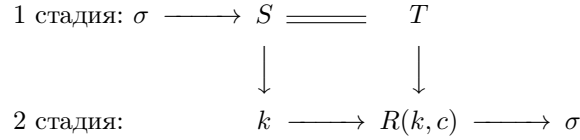
В свою очередь, так как g — инъекция, то $\forall s \exists \leq 1 r : g(r) = s$, то есть подменить ключ в самом деле не получится.

Замечание. Практически, одностороннюю перестановку удобно иметь не на $\{0, 1\}^*$, а на любой области D . В таком случае хочется сказать, что нужно уметь проверять условие $r \in D$, однако это не практично (например, если брать одностороннюю перестановку на базе RSA, мы не можем проверить,

является ли это число произведением больших простых чисел). Поэтому уточнение такое: нужно уметь генерировать случайную величину, которая либо равномерно распределена на D , либо принимает фиктивное значение (пустую строку, к примеру). В таком случае, мы можем пытаться открывать конверт, пока он не откроется, то есть либо в среднем за полиномиальное время, либо за гарантированный полином с маленькой вероятностью ошибки.

15 Интерактивный протокол привязки к биту

Такой протокол действует в две стадии, условно изображённые на схеме:



Где $c = c_{T,S(\sigma)}$ — протокол общения между S, T , а алгоритм верификации R проверяет его и возвращает что-то $\{0, 1, \perp_S, \perp_T\}$. Условия на протокол похожие с поправкой на то, что жульничать могут обе стороны:

- Корректность: $R(k_{T,S(\sigma)}, c_{T,S(\sigma)}) = \sigma$.
- Неразглашение: $\forall T^* \rightarrow c_{T^*,S(0)} \sim c_{T^*,S(1)}$.
- Неподменяемость: $\forall S^*$ с пренебрежимо малой вероятностью могут произойти события:
 - $\exists k_0, k_1 R(k_0, c_{T,S^*}) = 0, R(k_1, c_{T,S^*}) = 1$.
 - $\exists k_1 : R(k_1, c_{T,S^*}) = \perp_T$.

Конечно, неинтерактивный протокол можно без труда переделать в интерактивный. Однако, для существования интерактивного протокола будет достаточно меньшего предположения, а именно существования генератора $G : \{0, 1\}^n \rightarrow \{0, 1\}^{3n}$. Протокол устроен так:

- T посылается случайный вектор $t \in \{0, 1\}^{3n}$.
- S имеет собственный случайный вектор $s \in \{0, 1\}^n$ и посылает $m = G(s) \oplus (t \cdot \sigma)$.
- Верификатор $R(t, m, s)$ выдает:
 - \perp_T , если $|t| \neq 3n$.
 - \perp_S , если $|m| \neq 3n$ или $|s| \neq n$ или $m \oplus G(s) \notin \{t, 0^{3n}\}$.
 - 0, если $m \oplus G(s) = 0^{3n}$.
 - 1, иначе.

Корректность как всегда ясна (по модулю случая, когда $t = 0^{3n}$, его нужно либо запретить с небольшим перекосом в распределении, либо допустить малую вероятность некорректности).

$G(s)$ неотличимы от равномерных, значит какими бы ни были t , $G(s) \oplus t$ тоже неотличимы от равномерных, что даёт неразглашение.

С неподменяемостью нужно разобрать некоторые случаи. Вероятность \perp_T просто равна 0. Остается только первое условие: в нём множество плохих $t = G(s_0) \oplus G(s_1)$ имеет размер не больше 2^{2n} , значит вероятность выбрать такое t экспоненциально мала, даже для любой константы $2 + \varepsilon$ вместо 3.

16 Протокол бросания монетки

Протокол выглядит так:

- Алиса запускает протокол привязки к своему случайному биту σ . После этого у нее остается ключ k , а у Боба появляется привязка c .
- Боб отправляет случайный бит τ прямым текстом.
- Алиса отправляет ключ к привязке k .
- Судья проверяет корректность всех действий и выдаёт $\tau \oplus \sigma$.

Протокол очевидно корректен. Если B^* отклоняется от протокола, то он может использовать T^* и взять τ в зависимости от c . Однако, так как привязка к 0 и привязка к 1 вычислительно неотличимы, то он не может существенно повлиять на конечную вероятность, иначе он бы был отличителем привязок. Таким образом, интересы Алисы следуют из неразглашения.

Интересы Боба же следуют из неподменяемости: так как Боб посылает реально случайный бит τ , она не может жульничать с самим битом σ , единственное, что она может попытаться сделать — это привязаться одновременно к 0 и 1, что невозможно в силу неподменяемости.

Лекция 8. Электронная подпись I

17 Электронная подпись с закрытым ключом

Схема электронной подписи с закрытым ключом:

- K — генератор закрытого ключа d .
- $S(m, d) = s$ — подпись.
- $V(m, d, s) \in \{0, 1\}$ — верификатор.
- Корректность: $P(V(m, d, S(m, d)) = 1) \approx 1$.

- Атака происходит следующим образом: схема C несколько раз может сгенерировать сообщение x_i и получить подпись для него. После этого схема D пытается сгенерировать новое слово x , не встречавшееся среди запросов и подписать его так, чтобы верификатор принял подпись.

$$m_1 = C(), s_1 = S(m_1, d),$$

$$m_2 = C(s_2), s_2 = S(m_2, d),$$

$\dots,$

$$m_k = C(s_1, \dots, s_{k-1}), s_k = S(m_k, d),$$

$$m_{k+1} = C(s_1, \dots, s_k), s_{k+1} = D(s_1, \dots, s_k).$$

Условие неподделываемости: $P(m_{k+1} \notin \{m_1, \dots, m_k\}, V(m_{k+1}, d, s_{k+1}) = 1) \approx 0$.

Если существует протокол подписи, то его можно использовать для задачи идентификации: сервер посылает клиенту случайное сообщение m на подпись и проверяет подпись под ним. Несложно увидеть, что атака на такой протокол это и есть атака на электронную подпись.

С закрытым ключом алгоритм довольно простой: пусть d — идентификатор псевдослучайной функции. Положим $S(m, d) = f_d(m)$, $V(m, d, s) = I(s = f_d(m))$. Взломщик в таком случае получает значение псевдослучайной функции на полиномиальном числе входов. Однако семейство псевдослучайных функций вычислительно неотличимо от семейства всех функций, поэтому подписывая значение m_{k+1} он угадает с вероятностью, близкой к вероятности угадать значение случайной функции, которое не зависит от значений в предыдущих точках, то есть с экспоненциально малой вероятностью.

18 Электронная подпись с открытым ключом

- K — генератор закрытого и открытого ключей d, e .
- $S(m, d) = s$ — подпись.
- $V(m, e, s) \in \{0, 1\}$ — верификатор.
- Корректность: $P(V(m, e, S(m, d)) = 1) \approx 1$.
- Атака происходит таким же образом, однако схемы C, D дополнительно получают аргумент e .

Условие неподделываемости: $P(m_{k+1} \notin \{m_1, \dots, m_k\}, V(m_{k+1}, e, s_{k+1}) = 1) \approx 0$.

Подпись одного бита можно сделать на базе любой односторонней функции f . $d = (x_0, x_1)$, $e = (f(x_0), f(x_1))$ (нужно выбрать так, чтобы $f(x_0) \neq f(x_1)$).

$m_1 = \sigma = C(y_0, y_1)$, $s_1 = x_\sigma$, $m_2 = 1 - \sigma$, $s_2 = D(y_0, y_1, x_\sigma)$. Нужно показать, что $P(f(s_2) = y_{1-\sigma}) \approx 0$.

Пусть существует алгоритм, взламывающий такую подпись, тогда построим алгоритм, обращающий одностороннюю функцию. Обратитель f будет действовать так:

- Получает y .
- Выбирает случайный x и случайный бит τ .
- На всякий случай проверяет $f(x) \neq y$ (если равно, то обратил).
- $y' = f(x'), y_\tau = y', y_{1-\tau} = y$.
- $C(y_0, y_1) = \tau \Rightarrow$ можно вернуть $D(y_0, y_1, x')$.
- В противном случае вернуть что угодно.

Так как вероятность последнего условия $\frac{1}{2}$, то вероятность обращения не меньше $\frac{1}{2}$ вероятности подделки подписи.

Построим теперь подпись одного сообщения фиксированной длины. Для этого переделаем схему подписи одного бита (K, S, V) в $(\hat{K}, \hat{S}, \hat{V})$, так что \hat{K} запускает k раз алгоритм K , генерируя $(d_1, e_1), \dots, (d_k, e_k)$ то есть $d = (d_1, \dots, d_k), e = (e_1, \dots, e_k)$.

$\hat{S}(\sigma_1, \dots, \sigma_k) = S(\sigma_1, d_1) \dots S(\sigma_k, d_k)$. Верификатор устроен очевидным образом.

Такая подпись работает только для фиксированной длины (нельзя даже брать слова длины $\leq k$, так как после любого сообщения можно корректно подписать любой его префикс), притом подпись одноразовая, поскольку подпись сообщений $0 \dots 0$ и $1 \dots 1$ позволяет подделать любую подпись.

Часть проблем решает беспрефиксным кодированием: например удваиваем каждый бит и дописываем в конце 01. Так, вместо длины ровно k мы сможем подписывать сообщения длины $\leq k$.

19 Снятие ограничений по длине и одноразовости

Идея: подпись под хеш-значением.

Определение 1. Семейство хеш-функций с трудно обнаружимыми коллизиями — $h : \{0, 1\}^{p(n)} \times \{0, 1\}^* \rightarrow \{0, 1\}^{q(n)}$, такая что для схемы, ищущей коллизии $C(s) = \{x_1, x_2\}, x_1 \neq x_2, h_s(x_1) = h_s(x_2)$ вероятность успеха близка к 0 при случайно выбранном s .

Если такое семейство существует, то достаточно взять хеш-значение сообщения и подписать только его. Атакующий должен будет сделать одну из двух вещей — либо найти коллизию хеш-функции, либо использовать сообщение с другим хешом, но подделать его подпись. В обеих ситуациях вероятность успеха мала.

Идея для многоразовой подписи: S, V интерактивны и хранят всю историю, взломщик прослушивает канал и может потом попытаться подделать подпись. S вместе с подписью присылает новый открытый ключ для следующего раунда, притом подпись распространяется и на него.

Идею интерактивности можно использовать для подписи слова произвольной длины. $m = \sigma_1 \dots \sigma_k$, (d_0, e_0) ключ для однократной подписи k -битов $e = (e_0, l)$. Генерируются (d_1, e_1) , s_1 — подпись под $h_l(\sigma_1, e_1)$ при помощи d , (d_2, e_2) , s_2 — подпись под $h_l(\sigma_2, e_2)$ при помощи d_1 и так далее.

Определение 2. Универсальное семейство односторонних хеш-функций — семейство, такое что трудно обнаружить коллизию для заранее заданного x .

Общая идея для многократной электронной подписи без внутренней памяти: вместо следующего ключа подписываются 2 следующих ключа: какой из них используется зависит от битов сообщения.