Лекция 1

Введение в криптографию

(Конспект: В. Моргенштерн)

1.1 Постановка задачи

В криптографии существует множество задач, которые имеют широкую область применения. Это и закрытая передача данных по открытым каналам, и алгоритмы электронной подписи, и интерактивные доказательства с нулевым разглашением, и многие другие. Изучение криптографии мы начнем с так называемых “протоколов с открытым ключом”. Задача состоит в следующем. Имеется читатель A и писатель B. Проблема заключается в том, чтобы передать данные от B к A в зашифрованном виде, чтобы никто из тех, кто перехватит данные на пути из B к A, не смог расшифровать их. Кроме того, необходимо, чтобы перехватчик также не мог модифицировать зашифрованное сообщение и, таким образом, подсунуть A ложную информацию. В начале работы протокола у A и B нет никакой общей конфиденциальной информации. Канал между A и B все время открыт, то есть читать из него может кто угодно.

1.2 Криптосистема с открытым ключом[1]

Неформально, идея системы, которая бы позволила нам решить поставленную задачу, состоит в следующем. Первоначально читатель A создает пару ключей. Один ключ (`pub`) публикается. Он общедоступен и называется открытым ключом. Второй ключ (`pri`) читатель оставляет у себя и хранит в секрете. Этот ключ называется закрытым (личным, приватным). Алгоритм кодирования устроен так, что любой может закодировать сообщение, зная открытый ключ. Задача раскодирования сообщения легко решается, если известен закрытый ключ (`pri`). Если этот ключ врагу не известен, то раскодировать такое такое сообщение будет крайне трудно. Тем труднее, чем длиннее были выбраны ключи `pri` и `pub`. Существование подобной схемы целиком и полностью опирается на гипотезы теории сложности, о которых пойдет речь в дальнейшем.

Определение 1.1. Для формального построения криптосистемы нам понадобятся следующие алгоритмы.

[1] Public Key Encryption Scheme

Генератор ключей — вероятностный алгоритм с полиномиальным математическим ожиданием времени работы

$$G : 1^k \mapsto (\mathtt{pub}, \mathtt{pri}),$$

где $|\mathtt{pub}| = |\mathtt{pri}| = k$.

Шифровальщик — вероятностный алгоритм с полиномиальным математическим ожиданием времени работы

$$E : (m, \mathtt{pub}) \mapsto \mathtt{code}.$$

Дешифровщик — вероятностный алгоритм с полиномиальным математическим ожиданием времени работы

$$D : (\mathtt{code}, \mathtt{pri}) \mapsto m.$$

От шифровальщика и дешифровщика, мы, естественно, должны дополнительно потребовать малую вероятность ошибки: для любого сообщения m вероятность события $D(E(m, \mathtt{pub}), \mathtt{pri}) \neq m$ — константа (единая для всех m).

1.3 Односторонние функции

Определенные выше функции еще не дают нам никаких оснований полагать, что криптосистема будет надежной. Для того, чтобы создать надежную криптосистему, нам понадобится понятие семейства односторонних перестановок с секретом[2]. Чтобы дать определение такого семейства, мы сначала определим односторонние функции[3] и односторонние перестановки[4].

Определение 1.2. *Односторонней функцией* называется функция

$$f : X \to Y,$$

вычислимая за полиномиальное время, обратная к которой не может быть вычислена за полиномиальное время. (Обратная — любая функция $g : Y \to X$, такая, что $\forall y \in \mathbf{Im} f \; g(y) \in f^{-1}(y)$.)

Замечание 1.1. Вторая компонента ($\mathtt{pub}$) генератора ключей должна быть односторонней функцией от используемых случайных битов (в противном случае можно было бы вычислить по $\mathtt{pub}$ случайные биты, а по ним — $\mathtt{pri}$). Это одно необходимых (но вовсе не достаточных!) условий надежности криптосистемы. О достаточных мы будем говорить много позже. □

В дальнейшем одностороннюю функцию будет удобно понимать как последовательность булевых схем $f_i : X \cap \{0,1\}^i \to Y$, описания которых можно породить полиномиальным по времени алгоритмом: $1^i \mapsto f_i$.

[2] family of trapdoor permutations
[3] one-way function
[4] one-way permutation

Теорема 1.1. $\mathbf{P} \neq \mathbf{NP}$ *тогда и только тогда, когда существуют односторонние функции.*

Доказательство. $\boxed{\Rightarrow}$ Действительно, пусть F — формула в КНФ, а A — набор значений переменных. Определим одностороннюю функцию:

$$f(F, A) = (F, F[A]).$$

Если бы могли обратить эту функцию, мы могли бы найти выполняющий набор для любой выполнимой формулы (вычислив $f^{-1}(F, \texttt{True})$). Но мы знаем, что эта задача $\widetilde{\mathbf{NP}}$-полна.

$\boxed{\Leftarrow}$ Задача обращения функции, вычислимой за полиномиальное время, очевидно, принадлежит $\widetilde{\mathbf{NP}}$. □

Определение 1.3. *Односторонней перестановкой* называется односторонняя функция, которая является инъективной.

Легко доказать аналог предыдущей теоремы.

Теорема 1.2. $\mathbf{P} \neq \mathbf{UP}$ *тогда и только тогда, когда существуют односторонние перестановки.*

Замечание 1.2. Здесь $\mathbf{UP}$ — класс языков, принимаемых НМТ с ровно одним принимающим путем для слов из языка и нулем принимающих путей для слов не из языка.

Доказательство. Для НМТ M, задающей язык из $\mathbf{UP} \setminus \mathbf{P}$, определим

$$f(M, x, \text{путь вычислений } A \text{ на входе } x) = \begin{cases} (M, x, A), & A \text{ — отвергающий}, \\ (M, x), & A \text{ — принимающий}. \end{cases}$$

Обратное очевидно. □

Усовершенствуем понятие односторонней перестановки.

Определение 1.4. Семейством перестановок «с секретом»[5], сохраняющих длину, называется семейство булевых схем $\{f_i\}_{i \in I}$, где $\forall i \in I$ $f_i : D_i \to D_i$ является перестановкой; любая возрастающая (по $|i|$) последовательность этих перестановок является односторонней перестановкой, $D_i \subseteq \{0,1\}^{|i|}$, и существуют вероятностные алгоритмы $G : 1^k \mapsto (f_i, t_i)$ (где $|i| = k$) и $g^* : (c, t_i) \mapsto f_i^{-1}(c)$.

Замечание 1.3. Важно, что семейство содержит много функций f_i для одной длины входа $|i|$.

Замечание 1.4. Аналогично можно определить семейство односторонних функций «с секретом», а также семейство односторонних перестановок «с секретом», необязательно сохраняющих длину. □

[5] trapdoor permutations

Замечание 1.5. Из любой надежной криптосистемы можно получить семейство перестановок с секретом (возможно, не сохраняющих длину) следующим способом. Положим $I = \mathbf{Im}(G)$, $D_i = \{0,1\}^{|i|}$, для $m \in D_i$

$$f_i(m) = E(m,i) : D_i \to \{0,1\}^{\mathrm{poly}(|i|)}.$$

(Здесь использовано, что "надежная" криптосистема должна "надежно" шифровать хотя бы сообщения той же длины, что и используемый ключ.) □

Теперь мы можем попытаться построить криптосистему, предполагая, что такое семейство существует. Итак, генератор

$$G : 1^k \mapsto (f_i, t_i)$$

выдает нам пару (функция — публичный ключ, ее секрет — приватный ключ); шифровальщик

$$E : (m, f_i) \mapsto f_i(m)$$

применяет одностороннюю перестановку, а дешифровщик

$$D : (\mathtt{code}, t) \mapsto g^*(\mathtt{code}, t)$$

обращает ее, зная секрет.

Однако, в этой системе есть несколько неприятностей:

- алгоритм E — детерминированный; поэтому, однажды узнав $E(0, f_i)$, мы всегда сможем его отличить от $E(1, f_i)$ (если, конечно, не будем менять ключ каждый раз — но это крайне неэффективно, особенно если «писателей» — несколько);

  Замечание 1.6. Если же E — вероятностный, или мы действительно каждый раз используем новый ключ (что в некотором смысле — то же самое), то каждой криптосистеме соответствует естественным образом *promise problem*: в предположении ("promise"), что вход x — код нуля или единицы, ответить, кодом чего же именно является x.

  Вычислительная трудность этой проблемы является, очевидно, необходимым условием надежности побитного кодирования. Между тем, каждое из множеств "коды нуля" Z_0 и "коды единицы" Z_1 принадлежит **NP** (строкой подсказки являются случайные биты алгоритма кодирования и случайные биты ключа — подразумевается, что открытый ключ также входит в "код"). Если все строки являются кодами чего-либо (либо коды легко отличить от не-кодов), то эти множества (и, неформально говоря, задача раскодирования) оказываются в $\mathbf{NP} \cap \mathbf{co\text{-}NP}$. В противном случае задача раскодирования — это так называемая *дизъюнктная* **NP**-*пара*, а алгоритм взлома — *разделитель* для этой **NP**-пары (он должен правильно отвечать, какому из множеств принадлежит строка, но только если строка принадлежит $Z_0 \cup Z_1$; в противном случае он может выдавать что угодно). □

- мы требовали от односторонних функций трудности в наихудшем случае — но что, если их будет легко обратить как раз на интересующем нас сообщении (или можно будет вычислить какую-то полезную функцию от сообщения — первую его половину, например).

В дальнейшем мы будем с ними бороться.

1.4 Пример: RSA

В заключение, мы опишем самый простой и популярный алгоритм шифрования с открытым ключом, который называется RSA, по первым буквам фамилий его авторов (Rivest, Shamir, Adleman). В таком виде алгоритм, конечно, не является надежным, но это служит базой для многочисленных модификаций, которые на самом деле применяются на практике.

Итак выберем два очень больших простых числа p, q. Вычислим число $n = pq$. Выберем число e так, чтобы НОД$(e, p-1) =$ НОД$(e, q-1) = 1$, то есть НОД$(e, \phi(n)) = 1$. Далее, находим $d \in [1..n]$, такое, что $de \equiv 1 \pmod{\phi(n)}$. Оно, очевидно, существует и единственно в силу того, что НОД$(e, \phi(n)) = 1$.

Упражнение 1.1. А нужное e как найти?

Итак, публичный ключ `pub` $= (e, n)$; приватный ключ `pri` $= (d, n)$. Функция кодирования

$$f_{n,e}(x) = x^e \bmod n.$$

Для декодирования мы просто возводим код в степень d

$$g^*_d(y) = y^d \bmod n.$$

При этом по теореме Эйлера

$$g^*_d(x^e) \equiv x^{ed} \equiv (x^{\phi(n)})^k x \equiv x \pmod{n}.$$

Мы видим, что зная d, мы можем элементарно раскодировать сообщение за полиномиальное время, однако если d неизвестно, непонятно, как это можно было бы сделать.

Замечание 1.7. К сожалению, надежность модификаций этой системы (как и других) основывается не на общих гипотезах типа $\mathbf{P} = \mathbf{NP}$, а на том, что *конкретные* функции являются односторонними (в данном случае — $f_{n,e}$).