НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

Криптография, основанная на помехоустойчивом кодировании.

Лекция 1. Введение и основные классические результаты

Иванов Ф. И.
к.ф.-м.н., доцент

Национальный исследовательский университет
«Высшая школа экономики»

9 августа 2020 г.

Основные "контрагенты"в криптографии

Рассмотрим задачу передачи некоторого сообщения **m**, информация в котором представляет некоторый "интерес"и не должна быть доступна подсторонним. Пусть в процессе участвуют:

- Отправитель ("Боб") - хочет переслать сообщение Алисе так, чтобы только Алиса могла его прочесть
- Получатель ("Алиса") - легитимный получатель сообщения
- Аналитик/злоумышленник ("Ева") - нелегитимный участник, который хочет получить информацию из сообщения.

Основные элементы криптографии

Предположим, что передача сообщения $\mathbf{m}$ осуществляется через некоторый (в общем случае открытый) канал передачи $\mathcal{C}$. Тогда Бобу нужно спрятать (зашифровать) информацию в сообщении $\mathbf{m}$ так, чтобы его могла легко прочесть (дешифровать) Алиса, но для Евы эта задача (атака) была бы сложной!

- Боб применяет к $\mathbf{m}$ некоторую функцию шифрования $f(\mathbf{m})$ и получает зашифрованное сообщение $\mathbf{c}$
- Алиса получает $\mathbf{c}$ и применяет обратное преобразование f^{-1} и получает $f^{-1}(\mathbf{c}) = f^{-1}(f(\mathbf{m})) = \mathbf{m}$
- Предполагается, что сам алгоритм шифрования известен. В чем же тогда сложность для Евы?

Криптографические ключи

- Для того, чтобы осуществлять защищенную передачу данных, необходимо ввести в шифрование/дешифрование некоторый секрет (секретный ключ) K, который бы затруднил для Евы извлечение информации из передаваемого сообщения.
- Сделаем допущение: *пусть существует секретный ключ K, известный Алисе (и, возможно, Бобу), но неизвестный Еве. Предполагается, что без знания этого ключа Еве потребутся значительные усилия для извлечения информации* $\mathbf{m}$ *из* $f(\mathbf{m})$
- Вопрос только в том, кому доступен секретный ключ?

Симметричная криптография

Секретный ключ K доступен и Бобу, и Алисе (один и тот же ключ используется для шифрования и дешифрования):

1. Боб использует алгоритм шифрования f и секретный ключ K для преобразования $\mathbf{m}$ в шифртекст $\mathbf{c}$: $\mathbf{c} = f(\mathbf{m}, K)$
2. Алиса получает $\mathbf{c}$ и имея $\mathbf{K}$ вычисляет $f^{-1}(f(\mathbf{m}, K), K) = \mathbf{m}$
3. Предполагается, что из $f(\mathbf{m}, K)$ даже при условии знания того, как устроен алгоритм f без знания K очень сложно получить как K, так и $\mathbf{m}$.

Шифр Вернама

- Информация $\mathbf{m}$ представляет из себя двоичный вектор длины n: $\mathbf{m} = (m_1, m_2, ..., m_n)$.
- Ключ шифрования $\mathbf{K}$ представляет из себя случайный двоичный вектор длины n
- Тогда шифртекст имеет вид: $\mathbf{c} = \mathbf{m} \bigoplus \mathbf{K}$
- Ясно, что $\mathbf{c} \bigoplus \mathbf{K} = \mathbf{m}$ (расшифрование).

В 1945 году К. Шеннон доказал абсолютную стойкость этого шифра. Но какой ценой она достается?

- Должно задаваться равномерное распределение на множестве V^n - всех двоичный последовательностей длины n
- Длина ключа равна длине сообщения
- Ключ одноразовый

Доказательство абсолютной стойкости

Пусть $\mathbf{m} = (m_1, ..., m_n)$ - передаваемое сообщение, $\mathbf{K} = (k_1, ..., k_n)$ - ключ, причем $P_k(\mathbf{K}) = 2^{-n}$. Распределение шифртекстов:

$$P(\mathbf{m} \oplus K = \mathbf{c}) = \sum_{\mathbf{m}} P(\mathbf{m})P(\mathbf{m} \oplus K|m) = \sum_{\mathbf{m}} P(\mathbf{m})2^{-n} = 2^{-n}$$

Совместное распределение открытых текстов и шифртекстов:

$$P(\mathbf{m} = \mathbf{a},\mathbf{c} = \mathbf{b}) = P(\mathbf{m} = \mathbf{a})P(\mathbf{c} = \mathbf{b}|\mathbf{m} = \mathbf{a}),$$

где

$$P(\mathbf{c} = \mathbf{b}|\mathbf{m} = \mathbf{a}) = P(\mathbf{m} = \mathbf{a})P(\mathbf{m} \oplus K = \mathbf{b}|\mathbf{m} = \mathbf{a}) = P(\mathbf{K} = \mathbf{b} \oplus \mathbf{a}|\mathbf{m} = \mathbf{a}),$$

$$P(\mathbf{K} = \mathbf{b} \oplus \mathbf{a}|\mathbf{m} = \mathbf{a}) = P(\mathbf{K} = \mathbf{b} \oplus \mathbf{a}) = 2^{-n}.$$

Итого:

$$P(\mathbf{m} = \mathbf{a},\mathbf{c} = \mathbf{b}) = P(\mathbf{m} = \mathbf{a})2^{-n}$$

Симметричная криптография - выводы

- Один ключ для шифрования и расшифрования
- Проблема надежной передачи ключа через открытый канал

Примеры шифров:

- AES 128/192/256 (длина блока - 128 бит)
- ГОСТ 28147-89 256 (длина блока - 64 бита)
- DES 56 (длина блока - 64 бита)
- 2-3 DES 56*2, 56*3 (длина блока - 64 бита)
- Blowfish 32-448 (длина блока - 64 бита)

Асимметричная криптография

Отличие заключается в том, что только Алиса является обладателем секретного ключа!
В асимметричных системах используется пара ключей $(K_{open}, K_{private})$, причем K_{open} - доступен всем и используется для шифрования, а $K_{private}$ - это секретный ключ, используемый для расшифрования сообщений.

1. Боб использует открытый ключ и функцию шифрования и посылает Алисе сообщение в виде $\mathbf{c} = f(K_{open}, \mathbf{m})$
2. Алиса использует $K_{private}$ для расшифрования сообщения:

$$f^{-1}(f(K_{open}, \mathbf{m}), K_{private}) = \mathbf{m}$$

3. Предполагается, что Ева, располагая $f(K_{open}, \mathbf{m})$ и K_{open}, но не имея $K_{private}$ не сможет извлечь $\mathbf{m}$ из $\mathbf{c}$ за "разумное"время

Односторонние функции

Криптографические системы с открытым ключом используют так называемые односторонние функции, которые обладают следующим свойством:

- **если известно x, то вычислить $f(x)$ относительно просто**
- **если известно $y = f(x)$, то для вычисления x нет простого (эффективного) пути.**

Под односторонностью понимается не теоретическая однонаправленность, а практическая невозможность вычислить обратное значение, используя современные вычислительные средства, за обозримый интервал времени.

Предпосылки - протокол Диффи-Хэллмана

Криптографический протокол, позволяющий двум и более сторонам получить общий секретный ключ, используя незащищенный от прослушивания канал связи. Полученный ключ используется для шифрования дальнейшего обмена с помощью алгоритмов симметричного шифрования.
Элементы:

1. Алиса и Боб - хотят создать защищенный сеанс передачи, обмениваясь данными по открытому каналу для получения сеансового ключа
2. Общедоступные данные: два числа g и p - простое.
3. Секретные данные Боба - большое число a.
4. Секретные данные Алисы - большое число b.

Описание протокола DH

1. Боб передает Алисе значение выражения $A = g^a \mod p$
2. Алиса передает Бобу значение выражения $B = g^b \mod p$
3. Боб вычисляет $K_b = B^a \mod p = g^{ab} \mod p$
4. Алиса вычисляет $K_a = A^b \mod p = g^{ba} \mod p$

Легко заметить, что $K_a = K_b$ и общий сеансовый ключ получен.

Для получения этого ключа третьей стороне (Еве) требуется решить сложную задачу: по известным g, p и $g^b \mod p$ найти b (задача дискретного логарифмирования).

Общие принципы построения криптосистем с открытым ключом

- Начинаем с трудной задачи P. Она должна решаться сложно в смысле теории: не должно быть алгоритма, с помощью которого можно было бы перебрать все варианты решения задачи P за полиномиальное время относительно размера задачи. Более правильно сказать: не должно быть известного полиномиального алгоритма, решающего данную задачу — так как ни для одной задачи ещё пока не доказано, что для неё подходящего алгоритма нет в принципе.
- Можно выделить легкую подзадачу P' из P. Она должна решаться за полиномиальное время и лучше, если за линейное.
- «Перетасовываем и взбалтываем» P', чтобы получить задачу P'', совершенно не похожую на первоначальную. Задача P'' должна по крайней мере выглядеть как оригинальная труднорешаемая задача P
- P'' открывается с описанием, как она может быть использована в роли ключа зашифрования. Как из P'' получить P', держится в секрете как секретная лазейка.
- Криптосистема организована так, что алгоритмы расшифрования для легального пользователя и криптоаналитика существенно различны. В то время как второй решает P''-задачу, первый использует секретную лазейку и решает P'-задачу.

Задачи, лежащие в основе асимметричного шифрования

1. Дискретное логарифмирование
2. Факторизация числа на простые сомножители
3. Задача упаковки рюкзака
4. Задача решения систем нелинейных уравнений над конечным полем
5. Проблема вычисления коротких векторов целочисленных решеток
6. Проблема декодирования линейного кода, не имеющего видимой алгебраической, комбинаторной или иной структуры

Криптосистема RSA - генерация ключей

Задача факторизации больших целых чисел.

1. выбираются два различных случайных простых числа p и q заданной длины
2. Вычисляется *модуль* $n = pq$.
3. Вычисляется значение функции Эйлера $\phi(n) = (p-1)(q-1)$
4. Выбирается *открытая экспонента* $1 < e < \phi(n)$.
5. Вычисляется *секретная экспонента* d: $de \equiv 1 \mod \phi(n)$
6. Открытый ключ: (e,n), секретный ключ (d,n).

RSA - шифрование/расшифрование

Предположим, Боб хочет послать Алисе сообщение m.
Сообщениями являются целые числа: $m \in \mathbb{Z}_n$. **Шифрование:**

1. Боб берет открытый текст m
2. Боб берет открытый ключ (e,n)
3. Боб вычисляет $E(m) = c = m^e \mod n$ и отправляет c Алисе

Расшифрование:

1. Алиса принимает зашифрованный текст c
2. Алиса использует секретную экспоненту d
3. Алиса вычисляет $D(c) = m = c^d \mod n$

Доказательство корректности RSA

Действительно $\forall n \in \mathbb{Z}_m \quad D(E(m)) = E(D(m)) = m^{ed} \mod n$.
Покажем что $\forall n \in \mathbb{Z}_m \quad m^{ed} = m \mod p$. Пусть $m \neq 0 \mod p$.
Поскольку числа e и d являются взаимно обратными относительно умножения по модулю $\varphi(n) = (p-1)(q-1)$:

$$k \in \mathbb{Z} : ed = 1 + k(p-1)(q-1).$$

Тогда

$$m^{ed} = m^{1+k(p-1)(q-1)} \mod p =$$

$$= m \left(m^{p-1} \mod p\right)^{k(q-1)} \mod p =$$

$$= m \cdot 1^{k(q-1)} = m.$$

Аналогично

$$m^{ed} \mod q = m \mod q.$$

Из китайской теоремы об остатках следует

$$m^{ed} \mod n = m \mod n.$$

Криптоанализ RSA

Существует множество алгоритмов для нахождения простых сомножителей, так называемой факторизации, самый быстрый из которых на сегодняшний день — общий метод решета числового поля, скорость которого для k-битного целого числа составляет:

$$\exp\left((c + o(1))k^{1/3}\log^{2/3} k\right), c < 2.$$

На сегодняшний день используют ключи RSA длиной от 1024 бит и более.

Algorithms for Quantum Computation: Discrete Logarithms and Factoring

Peter W. Shor
AT&T Bell Labs
Room 2D-149
600 Mountain Ave.
Murray Hill, NJ 07974, USA

Abstract

A computer is generally considered to be a universal computational device; i.e., it is believed able to simulate any physical computational device with a cost in computation time of at most a polynomial factor. It is not clear whether this is still true when quantum mechanics is taken into consideration. Several researchers, starting with David Deutsch, have developed models for quantum mechanical computers and have investigated their computational properties. This paper gives Las Vegas algorithms for finding discrete logarithms and factoring integers on a quantum computer that take a number of steps which is polynomial in the input size, e.g., the number of digits of the integer to be factored. These two problems are generally considered hard on a classical computer and have been used as the basis of several proposed cryptosystems. (We

[1, 2]. Although he did not ask whether quantum mechanics conferred extra power to computation, he did show that a Turing machine could be simulated by the reversible unitary evolution of a quantum process, which is a necessary prerequisite for quantum computation. Deutsch [9, 10] was the first to give an explicit model of quantum computation. He defined both quantum Turing machines and quantum circuits and investigated some of their properties.

The next part of this paper discusses how quantum computation relates to classical complexity classes. We will thus first give a brief intuitive discussion of complexity classes for those readers who do not have this background. There are generally two resources which limit the ability of computers to solve large problems: time and space (i.e., memory). The field of analysis of algorithms considers the asymptotic demands that algorithms make for these resources as a function of the problem size. Theoretical

Сложность алгоритма Шора

Шор показал, что его алгоритм решает задачу факторизации l-битного числа n с вероятностью $1 - \epsilon$ за N прогонов (итераций) базового алгоритма:

$$N \geq \frac{2 \log 1/\epsilon}{\alpha\beta} l^2,$$

где константы α и β не зависят от n.
Сложность одной итерации:

$$O(l^2 \log l \log \log l).$$

Постквантовая криптография

Постквантовая криптография строится из предположения о том, что злоумышленник (Ева) обладает мощным квантовым компьютером.
В настоящее время ведутся активные работы в области создания квантового компьютера:

https : *//en.wikipedia.org/wiki/Timeline_of_quantum_computing*.

Как только он будет представлен RSA, ECC, DH схемы перестанут быть надежными.

Кто может выжить?

- Кодовая криптография и ЭЦП
- Криптография на решетках
- Изогенное шифрование
- Криптография на квадратичных формах от нескольких переменных

Этот список основан на наилучших известных атаках. Это категории математических задач; отдельные системы могут быть совершенно небезопасными, если проблема не используется правильно.

Кодовая криптография - краткий обзор

1. 1971 - открытие кодов Гоппы.
2. 1978 - первая кодовая криптосистема Мак-Элиса. Использование кодов Гоппы в криптографии с открытым ключом
   - 2.1 Параметры Мак-Элиса выбраны для уровня защиты 2^{64}
   - 2.2 В 2008 году *Bernstein – Lange – Peters* предложили атаку за 2^{60} циклов
   - 2.3 Система легко масштабируема по уровню защиты
3. 1986 - криптосистема Нидеррайтера (упрощение криптосистемы Мак-Элиса)
4. Декодирование *Prange*: необходимо по крайней мере $(c_0 + o(1))l^2(\log_{10} l)^2$ бит для уровня защиты 2^l, где $c_0 \approx 0.7419$

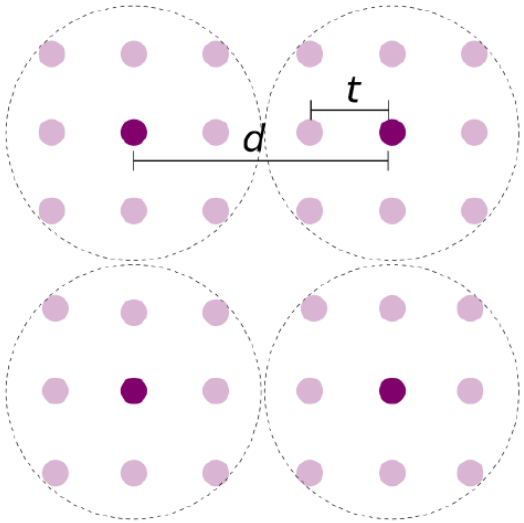
Базовые обозначения

Определение

Линейным (n, k, d)*-кодом* C *над конечным полем* F *называют* k*-мерное подпространство такое, что расстояние между векторами этого подпространства равно* $d = \min_{\mathbf{x} \neq \mathbf{y} \in C} dist(\mathbf{x}, \mathbf{y})$.

Теорема

Линейный (n, k, d)*-код* C *исправляет любые ошибки кратности до* $\frac{d-1}{2}$.



Базисные матрицы кода

Каждый код C может быть представлен через либо матрицу $\mathbf{G}$ (размера $k \times n$) задающую базис C:

$$\mathbf{G} = \left(\mathbf{g}_1^T, \mathbf{g}_2^T, \ldots, \mathbf{g}_k^T\right)^T,$$

где $\mathbf{g}_i$, $i = 1..k$ задает базис C, или через проверочную матрицу $\mathbf{H}$ размера $(n-k) \times n$, задающую базис ортогонального подпространства C.
Таким образом, код C можно определить как:

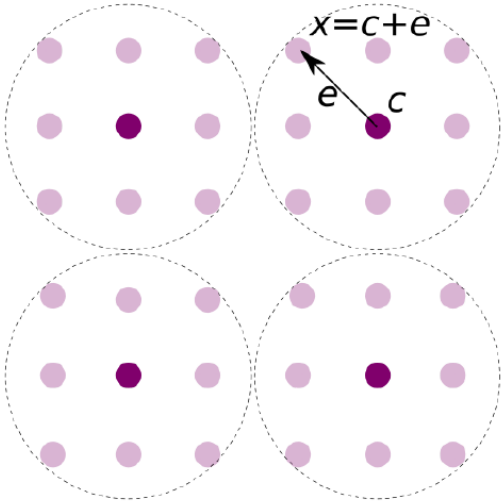
$$C = \{\mathbf{c} \in V : \mathbf{c} = \mathbf{uG}, \forall \mathbf{u} \in \mathbb{F}_q^k\}.$$

или:

$$C = \{\mathbf{c} \in V : \mathbf{cH}^T = \mathbf{0}\}.$$

Проблема декодирования

Для заданного $\mathbf{x} \in F^n$ найти такой вектор $\mathbf{c} \in C$, что $dist(\mathbf{x},\mathbf{c})$ наименьшая. Очевидно, если $\mathbf{x} = \mathbf{c} + \mathbf{e}$ и $wt(\mathbf{e}) \leq t$, то решение задачи единственно.



Теорема

Задача декодирования произвольного линейного кода является NP-трудной.

Двойственная задача

Входные данные: $n, r = n - k, t$, проверочная матрица $\mathbf{H}$ размера $(n - k) \times n$ и вектор $\mathbf{s}$ длины $n - k$.
Вопрос: найдется ли вектор $\mathbf{e}$ веса до t такой, что

$$\mathbf{H}\mathbf{e}^T = \mathbf{s}.$$

Декодирование линейного кода - сложность

Сложность исправления t ошибок в (n,k) двоичном коде имеет сложность $O(2^{n\alpha(k/n,t/n)})$:

Author(s)	Year	$\max\limits_{R,\tau} \alpha(R,\tau)$
Prange	1962	0.1207
Stern	1988	0.1164
Dumer	1991	0.1162
Bernstein, Lange, Peters	2011	
May, Meurer and Thomae	2011	0.1114
Becker, Joux, May, Meurer	2012	0.1019
May, Ozerov	2015	0.0966
Both, May	2017	0.0953
Both, May	2018	0.0885

Криптосистема Мак-Элиса - основная идея

- Общая проблема декодирования - сложная задача (P)
- Тем не менее для некоторых кодов известно простое декодирование (P')
- Одним из таких примеров являются коды Гоппы
- Необходимо "спрятать"такие коды, замаскировав их под общий случай и использовать его для шифрования (P'')
- Информация о том, как код "спрятан"используется для дешифрования ($P'' \mapsto P'$)
- Криптоаналитик в это же время вынужден решать задачу декодирования произвольного кода ($P'' \approx P$)

Криптосистема Мак-Элиса - генерация ключей

- Выберем секретную матрицу порождающую матрицу $\mathbf{G}$ (n,k,d) кода C, для которого известен простой алгоритм ϕ исправления t ошибок
- Выберем секретную случайную $k \times k$ обратимую матрицу $\mathbf{S}$
- Выберем секретную случайную $n \times n$ перестановку $\mathbf{P}$
- Секретный ключ: $(\mathbf{S}, \mathbf{G}, \mathbf{P})$. Открытый ключ: $(\mathbf{G}' = \mathbf{SGP}, t)$

Криптосистема Мак-Элиса - шифрование/дешифрование

Шифрование:

- Необходимо передать вектор $\mathbf{m}$ длины k.
- Выбрать случайный вектор $\mathbf{e}$ веса t.
- Отправить $\mathbf{x} = \mathbf{mG}' + \mathbf{e}$

Дешифрование:

- Вычислить $\mathbf{y} = \mathbf{xP}^{-1} = (\mathbf{mS})\mathbf{G} + \mathbf{eP}^{-1}$
- Декодировать слово $\mathbf{y}$, используя алгоритм ϕ, получим $\mathbf{mS}$
- $\mathbf{m} = \mathbf{mSS}^{-1}$

Пример шифрования

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \quad S = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \quad P = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$G' = SGP = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$\mathbf{m} = (0101)$, $c = \mathbf{mG}' = (1011010)$. Добавляем одну ошибку $\mathbf{e} = (0001000)$, тогда $\mathbf{x} = \mathbf{c} + \mathbf{e} = (1010010)$.

Дешифрование

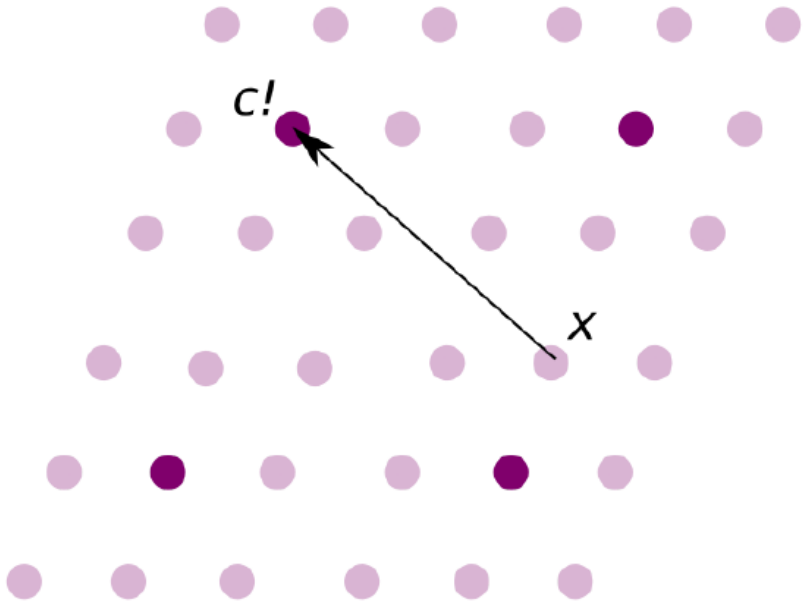
$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \quad S^{-1} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad P^{-1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Принимаем $\mathbf{x} = (1010010)$ и вычисляем $y = \mathbf{x}\mathbf{P}^{-1} = (0000111)$. Декодируем код Хэмминга $\mathbf{G}$: $\mathbf{m}' = \mathbf{m}\mathbf{S} = (0010)$. Умножаем результат на $\mathbf{S}^{-1}$: $\mathbf{m}'\mathbf{S}^{-1} = (0101)$.

Уровень защиты - классический

Проблема Мак-Элиса:

Для заданного открытого ключа $\mathbf{G}'$ и шифртекста $\mathbf{x}$ найти единственный вектор $\mathbf{m}$ такой, что $dist(\mathbf{mG}', \mathbf{x}) = t$



Факт: если вы можете решить общую проблему декодирования, то вы также можете решить проблему Мак-Элиса. Обратное неверно!

Уровень защиты - квантовый компьютер

Алгоритм Гровера: квантовый компьютер может найти неупорядоченное множество мощности I за $\sqrt{I}$ операций.

- Если применить алгоритм Гровера для задачи декодирования, можно получить квадратичный прирост в скорости, но сложность все еще экспоненциальна! Предположение состоит в том, что это максимальный прирост в скорости, которого можно добиться при помощи квантового компьютера.

Параметры классических криптосистем

Оригинальные параметры Мак-Элиса: (1024,524,101) код Гоппы. Уровень защиты: 50 бит.
В NIST были предложены следующие параметры:

- kem/mceliece6960119: $k = 5413$, $n = 6960$, $t = 119$. Длина открытого текста: 677 байт, длина зашифрованного текста: 870 байт, длина публичного ключа: 4.5 МБайт, длина секретного ключа 13.75 МБайт. Уровень защиты - 256 бит.
- kem/mceliece8192128: $k = 6528$, $n = 8192$, $t = 128$. Длина открытого текста: 870 байт, длина зашифрованного текста: 1024 байт, длина публичного ключа: 6.4 МБайт, длина секретного ключа 19.45 МБайт. Уровень защиты - 256 бит.

Оптимизация структуры - общие идеи

- Если $k > n - k$ — использовать проверочную матрицу вместо порождающей

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \Longrightarrow H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

- Хранить перестановку в виде вектора:

$$P = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \Longrightarrow P = (7, 2, 6, 3, 1, 5, 4)$$

Эффект от оптимизации:

- kem/mceliece6960119: $k = 5413$, $n = 6960$, $t = 119$. Публичный ключ: 4.5 МБ => 1.3 МБ, секретный ключ: 13.75 МБ => 4.8 МБ.
- kem/mceliece8192128: $k = 6528$, $n = 8192$, $t = 128$. Публичный ключ: 6.4 МБ => 1.6 МБ, секретный ключ: 19.45 МБ => 6.7 МБ.

Криптосистема Нидеррайтера - 1986 (I)

- Использовать проверочную матрицу $\mathbf{H}$ вместо порождающей матрицы $\mathbf{G}$
- Также используем секретную случайную $k \times k$ обратимую матрицу $\mathbf{S}$ и секретную случайную $n \times n$ перестановку $\mathbf{P}$.
- Публичный ключ: $\mathbf{H}' = \mathbf{SHP}$.
- Шифрование: открытый текст - это вектор ошибки $\mathbf{e}$ веса до t. Шифртекст имеет вид $\mathbf{s} = \mathbf{H}'\mathbf{e}$.
- Дешифрование: Найти вектор $\mathbf{e}$, такой что $wt(\mathbf{e}) = t$ и $\mathbf{s} = \mathbf{H}'\mathbf{e}$, используя алгоритм декодирования кода с проверочной матрицей $\mathbf{H}$

Криптосистема Нидеррайтера - 1986 (II)

- Публичный ключ: $\mathbf{H}' = \mathbf{SHP}$
- Шифрование: открытый текст - это вектор ошибки $\mathbf{e}$ веса до t. Шифртекст имеет вид $\mathbf{s} = \mathbf{H}'\mathbf{e}$
- Дешифрование:
  1. $\mathbf{S}^{-1}\mathbf{s} = \mathbf{S}^{-1}\mathbf{H}'\mathbf{e} = \mathbf{H}(\mathbf{Pe})$
  2. Легко видеть, что $wt(\mathbf{Pe}) = wt(\mathbf{e}) = t$
  3. Применяем эффективный синдромный декодер для $\mathbf{H}$ и находим $\mathbf{e}' = \mathbf{Pe}$
  4. $\mathbf{e} = \mathbf{P}^{-1}\mathbf{e}'$

Нидеррайтер предложил использовать коды Рида-Соломона в своей схеме. Эта криптосистема была взломана в 1992 году Сидельниковым и Шостаковым.

Кроме того, следующие коды оказались уязвимыми: каскадные коды, коды Рида-Маллера, некоторые алгеброгеометрические коды, коды Габидуллина, некоторые LDPC коды, циклические коды.

Примеры криптосистем:

- QC-MDPC
- Двоичные коды Гоппы
- Квазициклические коды
- Коды БЧХ
- Коды в ранговой метрике
- Ранговые квазициклические коды
- QC-LDPC
- Выколотые коды Рида-Маллера
- Полярные коды
- Подкоды обобщенных PC-кодов
- Алгеброгеометрические коды
- Сверточные коды

Атака по информационным совокупностям

Показала себя одной из лучших для криптоанализа кодовых систем.
Обозначим через $\mathcal{J}$ некоторое подмножество $\{1,2,\ldots,n\}$, а через $\mathbf{G}_{\mathcal{J}}$ матрицу, содержащую только столбцы с индексами из $\mathcal{J}$ в $\mathbf{G}$. Точно так же через $\mathbf{e}_{\mathcal{J}}$ мы обозначаем вектор $|\mathcal{J}|$ – длины с элементами $\mathbf{e}$, которые помещаются в индексы из $\mathcal{J}$.
Для получения $\mathbf{m}$ используют следующий алгоритм:

1. Выберем случайную информационную совокупность $\mathcal{J} \subset \{1,2,\ldots,n\}$.
2. Если на $\mathbf{x}_{\mathcal{J}}$ нет ошибок, то: $\mathbf{x}_{\mathcal{J}} = \mathbf{m}_{\mathcal{J}}\mathbf{G}_{\mathcal{J}} + \mathbf{e}_{\mathcal{J}}$.
3. Если $wt\left(\mathbf{x} + \mathbf{x}_{\mathcal{J}}\mathbf{G}_{\mathcal{J}}^{-1}\mathbf{G}\right) = t$ то нет ошибок на $\mathbf{x}_{\mathcal{J}}$ а поэтому $wt(\mathbf{e}_{\mathcal{J}}) = 0$. В этом случае $\mathbf{m} = \mathbf{x}_{\mathcal{J}}\mathbf{G}_{\mathcal{J}}^{-1}$. Иначе вернуться к шагу 1.

Сложность атаки

Легко заметить, что вероятность того, что выбранные k индексов свободны от ошибок, равна:

$$P_s = \frac{\binom{n-t}{k}}{\binom{n}{k}} = \frac{\binom{n-k}{t}}{\binom{n}{t}}.$$

Тогда среднее число попыток декодирования τ равно

$$\tau = \frac{1}{P_s} = \frac{\binom{n}{t}}{\binom{n-k}{t}}.$$

Это значение τ можно рассматривать как сложность ISD алгоритма.
Например, для (1024,524,101) кода Гоппы: $\tau \approx 2^{53}$

Атака Штерна

Зашифрованное сообщение $\mathbf{c} = \mathbf{mG}' + \mathbf{e}$ находится на расстоянии t от кодовых слов кода C'. В то же время, минимальное расстояние кода C' $2t + 1$. Построим порождающую матрицу A кода C_A:

$$\mathbf{A} = \begin{pmatrix} \mathbf{G}' \\ \mathbf{c} \end{pmatrix}$$

Код C_A имеет расстояние t, генерируемое единственным вектором веса t. Этот вектор и есть неизвестный вектор ошибки $\mathbf{e}$. Атака направлена на то, чтобы по известной проверочной матрице кода C_A найти t таких ее столбцов, сумма которых равна нулевому вектору. А как это делать - простор для оптимизации!

Итогое резюме

- Основное достоинство кодовой криптографии - масштабируемость и стойкость (по крайней мере, пока) к квантовым вычислениям
- Основная проблема - очень длинные ключи, необходимость которых обуславливается требованиями криптографической стойкости
- Как уменьшить длину ключа?
  1. Искать коды, которые можно описать компактно
  2. Учиться декодировать за t хотя бы некоторые множества ошибок
  3. В идеале - хотелось бы построить систему, для которой атака по информационным совокупностям/атака Штерна неприменимы.