

Таганрогский радиотехнический университет

Кафедра РТС

Лабораторная работа

Исследование алгоритмов шифрования

Таганрог
2006

1.1 Симметричные (одноключевые) криптографические алгоритмы

1.1.1 Общая характеристика симметричных криптоалгоритмов

Одноключевые криптографические алгоритмы являются классическими алгоритмами защиты информации. Для шифрования и расшифрования текста информационных сообщений в них используется один и тот же секретный ключ Z_c , сохранение которого в тайне обеспечивает надежность защиты информации.

Процесс шифрования и расшифрования формально можно представить следующим образом:

$$Y = F_{z_c}(X), \\ X = D_{z_c}(Y) = D_{z_c}(F_{z_c}(X)),$$

где X – исходный и расшифрованный тексты;

Y – криптотекст;

Z_c – ключ шифра;

F_{z_c} – алгоритм (функция) шифрования;

D_{z_c} – алгоритм (функция) расшифрования.

Для взаимной однозначности шифрования и расшифрования должно выполняться равенство: $F_{z_c} D_{z_c} = E$, где E – единичное преобразование. В криптографии алгоритмы шифрования часто называют шифрами.

Одноключевые алгоритмы (шифры) подразделяются на блочные (блоковые), поточные и комбинированные.

При применении блочных шифров открытый текст сообщения разбивается на блоки фиксированной длины. Тексты этих блоков шифруются отдельно и независимо друг от друга.

Известно три вида одноключевых блочных алгоритмов: алгоритмы перестановок, замены (подстановки) и комбинированные (составные).

1.1.2 Алгоритмы перестановок

При использовании алгоритмов перестановок символы открытого текста переставляются по некоторому правилу (ключу) в пределах заданного блока. В результате этого нарушается нормальный порядок их следования и сам смысл информационного сообщения. Различают шифры простой и сложной перестановки.

Алгоритм простой перестановки переупорядочивает группу букв текста регулярным образом в соответствии с выбранным ключом (правилом) перестановки. Пример простейшего шифра перестановки представлен в табл. 1.1.

Таблица 1.1

Простейший шифр перестановки

<i>м</i>	<i>д</i>	<i>щ</i>	<i>и</i>	<i>р</i>	<i>и</i>
<i>е</i>	<i>ы</i>	<i>и</i>	<i>н</i>	<i>м</i>	<i>и</i>
<i>т</i>	<i>з</i>	<i>т</i>	<i>ф</i>	<i>а</i>	<i>ы</i>
<i>о</i>	<i>а</i>	<i>ы</i>	<i>о</i>	<i>ц</i>	<i>й</i>

Исходный текст: МЕТОДЫ ЗАЩИТЫ ИНФОРМАЦИИ

Ключ. Исходный текст записать по столбцам в таблицу размером 6 столбцов на 4 строки, начиная с левого верхнего угла; дополнить свободные клетки таблицы произвольными символами. Криптотекст считывается по строкам таблицы, начиная с левого верхнего угла. Текст разбивается на пятисимвольные группы. Последняя группа дополняется до необходимого размера произвольными символами.

Криптотекст: МДЩИР ИЕЫИН МИТЗТ ФАЫЮА АЫЮЦЙ

Шифруемое сообщение «Методы защиты информации» записано в виде таблицы, состоящей из 4 строк и 6 столбцов, начиная с верхнего левого угла. Текст сообщения за-

писывается по столбцам, пробелы исключаются. Если последний столбец оказывается неполным, он заполняется любыми символами. Символы зашифрованного сообщения считываются из таблицы построчно (слева направо) и записываются группами, например, по 5 цифр. Последняя процедура к алгоритму шифрования не относится и делается только для удобства записи текста, лишенного всякого смысла. Для расшифровки криптотекст записывается в аналогичную таблицу, а исходный текст считывается из нее по столбцам.

Рассмотренный метод шифрования обладает невысокой криптоустойчивостью. Для расшифровки сообщения достаточно установить размер таблицы, что при длине криптотекста в 25 символов сделать совсем нетрудно. Возможные комбинации: 2x12, 3x8, 4x6, 5x5, 12x2, 8x3, 6x4. Если отбросить две заведомо непригодные комбинации (2x12 и 12x2), то перебрав по очереди четыре оставшиеся, легко восстановить исходный текст.

1.1.3 Криптоустойчивость алгоритма можно повысить, если строки или столбцы таблицы переставить по ключу (слову, фразе или набору чисел) длиной соответственно в строку или столбец таблицы.

В этом случае размер блока равен длине ключа слова, в котором не должно быть повторяющихся символов. Символы ключа, пронумерованные в порядке их расположения в алфавите, задают правило перестановки. Исходный текст последовательными строками записывается под символами ключа. Криптотекст выписывается колонками в той последовательности, в которой располагаются в алфавите буквы ключа или в порядке следования цифр в натуральном ряде, если ключ цифровой.

Пример шифрования с использованием шифра простой перестановки с использованием ключа шифрования 251634 (ключевое слово КРЕСЛО) представлен в табл.1.2 и табл. 1.3. Шифруемый текст (ВЫХОД НА СВЯЗЬ ВТОРНИК ШЕСТЬ ЧАСОВ) записывается без пробелов в строки исходной таблицы, количество столбцов которой равно длине ключа. В соответствии с длиной ключа шифруемое сообщение делится на блоки, состоящие из 6 символов. Эти блоки записываются в строки исходной таблицы, последняя строка которой дополнена до полной длины символом "Г". Затем столбцы таблицы переставляются в соответствии с ключом шифрования. В рассматриваемом случае первое место ставится второй столбец, на второе - пятый, на третье – первый и т.д. В результате будет получена таблица перестановок. Криптотекст считывается из строк таблицы перестановок, начиная, например, с левого верхнего угла.

Таблица 1.2

Исходная таблица					
1	2	3	4	5	6
в	ы	х	о	д	н
а	с	в	я	з	ь
в	т	о	р	н	и
к	ш	е	с	т	ь
ч	а	с	о	в	г

Таблица 1.3

Таблица перестановок					
2	5	1	6	3	4
ы	д	в	н	х	о
с	з	а	ь	в	я
т	н	в	и	о	р
ш	т	к	ь	е	с
а	в	ч	г	с	о

Исходный текст: ВЫХОД НА СВЯЗЬ ВТОРНИК ШЕСТЬ ЧАСОВ

Ключ: 251634 (КРЕСЛО)

Криптотекст: ЫДВНХ ОСЗАЬ ВЯТНВ ИОРШТ КЪЕСА ВЧГСО

Такой алгоритм также обладает невысокой криптостойкостью. Разложив число символов зашифрованного текста на множители можно определить вероятную длину кодового слова, которое использовалось при шифровании, и подобрать ключ

Для повышения криптостойкости полученный криптотекст можно зашифровать его еще раз с использованием таблицы с другой размерностью (длины строк и столбцов подбираются другими). Кроме того, в одной таблице можно переставлять строки, а в другой

столбцы. Заполнять таблицу исходным текстом можно разными способами: зигзагом, змейкой, по спирали и т. п. Этот способ шифрования известен под названием двойной перестановки. При использовании шифра сложной перестановки группы символов открытого текста подвергаются перестановкам не только по строкам (как в шифре простой перестановки), но и по столбцам. В таблицу каким-либо из вариантов записывается исходное сообщение. Относительно двух сторон таблицы записывается ключ, в соответствии с которым переставляются сначала столбцы, а затем строки таблицы. Число вариантов N двойной перестановки определяется размерностью шифровальной таблицы.

$$N=m!k!,$$

где m и k – число строк и столбцов в шифровальной таблице.

К шифрам перестановки можно отнести и метод шифрования с использованием квадратных решеток, трафаретов или палеток, у которых, у которых четверть ячеек (квадратиков) вырезана таким образом, чтобы после 4 поворотов они покрывали весь квадрат

1.1.3 Алгоритмы замены (подстановки)

Алгоритмы замены (подстановки) образуются с помощью замены знаков открытого текста другими знаками или символами в соответствии с определенным правилом (ключом). Шифрование на основе замены использует принцип шифроалфавита – перечня эквивалентов, применяемых для преобразования открытого текста в зашифрованный. В том случае, когда для шифрования используется всего один шифроалфавит, шифр называется одноалфавитным (моноалфавитным). Когда же используются два и более шифроалфавитов, шифр называется многоалфавитным (полиалфавитным).

В машинных методах шифрования используется цифровое представление текстовой информации, при котором символы текста заменяются некоторыми цифровыми эквивалентами или представляются в виде двоичного кода. При шифровании, символы исходного текста последовательно складываются с символами некоторой ключевой последовательности. Ключ на исходный текст накладывают двумя способами.

При первом способе символы исходного текста, замененные цифровыми эквивалентами, (например, А – 24, Б – 12, В – 31 и т. д.), складываются с символами ключа по модулю K , где K – число символов в алфавите.

При втором способе символы исходного открытого текста и ключа, представляются в виде двоичного кода и складываются поразрядно друг с другом по модулю 2 (рис. 1.1). Пусть необходимо зашифровать слово «КРЫША», каждая буква которого представлена неким двоичным кодом. Выберем в качестве ключа двоичную последовательность 1001 и сложим ее по модулю 2 с двоичными кодами символов исходного сообщения. В результате получится криптотекст в виде последовательности двоичных символов. Чтобы восстановить исходный текст достаточно сложить по модулю 2 символы криптотекста с символами ключа (рис 1.1б).

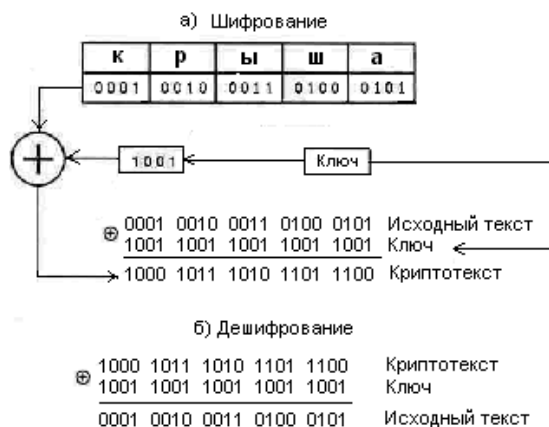


Рис. 1.1. Алгоритм замены с использованием двоичного кода

Вместо сложения по модулю 2 можно использовать любые другие однозначные логические функции, например, преобразование исходного текста и ключевого слова по правилам логической эквивалентности.

Примером одноалфавитного шифра является шифр Цезаря, в котором каждая буква исходного открытого текста заменялась третьей по счету буквой латинского алфавита с применением циклического сдвига. Вскрытие такого шифра осуществляется путем перебора возможных ключей, в качестве которых используется величина сдвига букв сообщения в алфавите до появления смыслового текста.

При простой моноалфавитной подстановке каждый символ исходного текста m_n , принадлежащий алфавиту А, заменяется соответствующим символом h_n , принадлежащим к алфавиту В криптотекста. Соответствие между символами алфавитов А и В задается с помощью кодовой таблицы или выражения. Например, для обобщенного шифра Цезаря выражение, устанавливающее связь между алфавитами А и В, имеет вид

$$F(h_i) = (F(m_i) + h) \bmod K,$$

где K – количество знаков в алфавитах;

h – постоянная величина сдвига.

Чтобы получить криптотекст номер символа в алфавите исходного текста суммирования с некоторым постоянным числом h . Шифрование данным способом эквивалентно сдвигу алфавита на фиксированное число позиций h . Если сдвиг $h=1$, то, например, для русского алфавита буква А заменяется на букву Б, буква Б – на букву В, буква Я – на букву А и т. д.

В качестве примера зашифруем вышерассмотренное информационное сообщение
ВЫХОД НА СВЯЗЬ ВТОРНИК ШЕСТЬ ЧАСОВ

После шифрования получим новое сообщение, которое имеет вид

ГЫЦПЕ ОБ ТГАИЭ ДУПСОЙЛ ЩЁТУЭ ШБТПГ

Увеличить надежность такого шифра можно путем использования перемешанного алфавита (табл.5.2). Исходное сообщение после применения такого алфавита будет выглядеть следующим образом:

ИЕЩТК СЖ ХИОБЁ ИЦОФСЭЯ ВЛХЦЁ БЖХТИ

Таблица 1.4.

Пример шифра моноалфавитной подстановки

1	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
2	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А
3	Ж	З	И	Й	К	Л	Ъ	Ы	Ь	Э	Ю	Я	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	А	Б	В	Г	Д	Е	Ё	М	Н	О

Примечание. 1 – исходный алфавит;

2 – рабочий алфавит (исходный алфавит, циклически сдвинутый на одну позицию вправо);

3 – перемешанный алфавит.

Хотя количество возможных перестановок букв алфавита очень велико (для русского алфавита 33!), шифры моноалфавитной замены не обладают высокой криптостойкостью.

В алгоритмах многоалфавитной замены для шифрования применяются несколько перемешанных алфавитов, поочередно используемых при замене букв исходного шифруемого сообщения. К многоалфавитным алгоритмам относятся шифр Вижинера, шифр «Энигма», цилиндр Джефферсона и др.

Шифр Вижинера организован следующим образом. Из выбранного исходного алфавита путем циклических сдвигов формируется множество вторичных алфавитов. Размер этого множества равен числу символов исходного алфавита.

Таблица 1.5.

Шифровальная таблица Вижинера для русского алфавита

А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А
В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б
Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В
Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г
Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д
Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е
Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё
З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж
И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З
Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И
К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й
Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К
М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л
Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М
О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н
П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О
Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П
С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р
Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С
У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т
Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У
Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф
Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х
Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц
Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч
Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш
Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ
Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ
Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы
Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь
Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э
Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю

Совокупность всех полученных алфавитов, сведенных в одну таблицу, образует так называемую шифровальную таблицу Вижинера. Например, для русского алфавита рассматриваемое множество состоит из 33 циклически сдвинутых исходных алфавитов (табл.1.5). При шифровании в этом случае применяется кодовое слово, буквы которого определяют выбор конкретного алфавита, используемого при замене символов исходного текста. Процедура шифрования может быть представлена как суммирование номеров соответствующих символов открытого текста и ключевого слова по модулю 33.

Рассмотрим пример шифрования сообщения «ВЫХОД НА СВЯЗЬ ВТОРНИК ШЕСТЬ ЧАСОВ» с использованием ключевого слова «КРЕСЛО» (длина ключа 6). Исходное сообщение разбивается на блоки длиной 6 символов, над которыми записываются символы ключа (рис. 1.2).

ВЫХОД НА СВЯЗЬ ВТОРНИК ШЕСТЬ ЧАСОВ - исходное сообщение

КРЕСЛО КРЕСЛО КРЕСЛО КРЕСЛО КРЕСЛО - ключевая строка

ВЫХОДН АСВЯЗЬ ВТОРНИ КШЕСТЬ ЧАСОВС - блоки исходного сообщения

МЛЪАПЬ КВЖРУК МГУВЩЧ ХИЙГЮК ВРЦАНА - криптотекст

К Р Е С Л О К Р Е С Л О К Р Е С Л О К Р Е С Л О К Р Е С Л О																																
В Ы Х О Д Н А С В Я З Ь В Т О Р Н И К Ш Е С Т Ь Ч А С О В С																																
А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й
Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П
Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д
С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р
Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К
О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н
М Л Ё А П Ъ К В Ж Р У К М Г У В Щ Ч Х И Й Г Ю К В Р Ц А Н А																																
К Р Е С Л О К Р Е С Л О К Р Е С Л О К Р Е С Л О К Р Е С Л О																																

Рис. 1.2 Пример шифрования с помощью таблицы Вижинера

Символы криптотекста выбирают на пересечении строки, задаваемой ключевой строкой и столбца, задаваемого символом исходного сообщения. Таким образом первым символом криптотекста для рассматриваемого примера будет символ М, стоящий на пересечении строки К и столбца В таблицы Вижинера (табл. 1.5). Вторым символом будет символ Л, стоящий на пересечении строки Р и столбца Ы и т.д. Для облегчения процесса шифрования на рис. 1.2 из таблицы Вижинера выбраны строки, первые символы которых соответствуют ключевому слову. Над ними записана ключевая строка и строка блоков исходного сообщения. В результате шифрования получится следующий криптотекст:

МЛЪАПЬ КВЖРУК МГУВЩЧ ХИЙГЮК ВРЦАНА

Для дешифровки криптотекста необходимо под или над таблицей записать ключевую строку и строку криптотекста (рис.1.2). Символы исходного сообщения получают из столбца первой строки таблицы, соответствующего символу криптотекста, задаваемому символом ключевого слова. Например, первым символом исходного текста будет символ В первой строки таблицы, стоящий в столбце М (первый символ криптотекста) в строке К, задаваемой символом ключевой строки, соответствующего данному символу криптотекста.

1.1.3 Составные шифры

Для повышения криптостойкости используют два общих принципа шифрования: рассеивание и перемешивание. Принцип рассеивания заключается в распространении влияния одного символа исходного текста на несколько символов криптотекста. Этим достигается маскировка статистических свойств исходного текста. В развитие этого принципа можно распространить влияние одного символа ключа на ряд символов криптотекста, что затрудняет восстановление ключа по частям. Принцип перемешивания заключается в выборе таких шифрующих преобразований, которые не позволяют установить взаимосвязь статистических свойств исходного и криптотекста.

Хорошее рассеивание и перемешивание достигается с использованием составного шифра. Такой шифр строят на основе совместного использования простых шифров замены и перестановки, каждый из которых вносит вклад суммарное рассеивание и перемешивание.

Примером составного шифра, разработанного в соответствии с принципами рассеивания и перемешивания, является стандарт шифрования данных DES, принятый в 1977 году Национальным бюро стандартов США. До настоящего времени не найдено уязвимых мест этого криптоалгоритма, которые позволили бы использовать метод криптоанализа, существенно лучший полного перебора ключей. В России аналогичный криптоалгоритм введен ГОСТ 28147-89.

1.2 Несимметричные (двухключевые) алгоритмы шифрования

1.2.1. Двухключевые криптографические системы используют два ключа: открытый (несекретный) и закрытый (секретный). Структурная схема системы шифрования двухключевой системы шифрования приведена на рис.1.3.

Двухключевые системы в зависимости от вариантов применения открытого и секретного ключей дают возможность получить две разновидности шифрования.

В первой для шифрования используется открытый ключ, а для дешифрования закрытый, - такую систему называют системой шифрования с открытым ключом. В этом случае зашифровать текст может каждый владелец открытого ключа, а расшифровать его - только владелец закрытого ключа. Такой способ используется, например, в системах сотовой подвижной связи стандарта GSM.



Рис. 1.3 Двухключевая система шифрования

Во второй для шифрования используется закрытый ключ, а для дешифрования открытый. Такую систему называют системой с закрытым ключом. В данном случае только владелец закрытого ключа может правильно зашифровать текст (подписать его), а расшифровать (проверить подпись) - любой пользователь системы, имеющий в своем распоряжении открытый ключ. Примером такой системы является система электронной цифровой подписи.

Процесс шифрования и расшифрования в двухключевых криптосистемах можно описать формально следующим образом.

$Y = E_{Z_0}(X)$, $X = D_{Z_3}(Y) = D_{Z_3}(E_{Z_0}(X))$ для систем с открытым ключом;

$Y = E_{Z_3}(X)$, $X = D_{Z_0}(Y) = D_{Z_0}(E_{Z_3}(X))$ для систем с закрытым ключом.

Здесь X – открытый текст;

Y – зашифрованный текст;

Z_0 – открытый ключ;

Z_3 – закрытый ключ;

E_{Z_0} – функция шифрования;

D_{Z_3} – функция расшифрования.

Для взаимной однозначности процедур шифрования и дешифрования с открытым и секретным ключами должны выполняться условия

$$E_{Z_0} D_{Z_3} = E_{Z_3} D_{Z_0} = e,$$

где e – единичное преобразование.

1.2.2 В двухключевых криптосистемах используются некоторые алгоритмы преобразования данных и два разных, но взаимосвязанных друг с другом ключа: один открытый, доступный любому лицу, другой закрытый, доступный только одному лицу.

Стойкость двухключевых криптосистем определяется вычислительной сложностью алгоритмов шифрования: сообщение не сможет быть дешифровано (зашифровано) без ключа за требуемое время из-за большого объема необходимых вычислений.

В двухключевых криптосистемах в алгоритмах шифрования и расшифрования используются разные ключи, один из которых нельзя получить из другого с приемлемыми вычислительными затратами. Следовательно, рассматриваемые методы шифрования должны обладать как минимум двумя свойствами:

законный пользователь сможет выполнить прямое или обратное криптопреобразование текста сообщения;

злоумышленник или криптоаналитик противника не сможет восстановить исходное сообщение по криптотексту или получить криптотекст по исходному сообщению с приемлемыми затратами времени и средств.

Двухключевые криптосистемы используют так называемые односторонние функции. Под односторонней функцией понимают некоторую функцию $f(x)$, значение которой для заданного аргумента x вычисляется легко, тогда как обратная вычислительная задача - нахождение x из $f(x)$ – относится к классу трудновычислимых (в смысле теории сложности).

Например, несложно перемножить два простых 100-значных числа, однако для разложения на множители получившегося 200-значного числа потребуются десятки лет непрерывной работы мощного компьютера. Криптоалгоритм на основе умножения простых чисел использует, например, алгоритм RSA. Открытый ключ в таких системах предоставляется широкому кругу пользова-

телей без опасения его огласки, а закрытый ключ, который используется для шифрования или дешифровки сообщений. В системах с открытым ключом, используя открытый ключ, отправитель может зашифровать открытый текст и отправить его получателю, который сформировал данный ключ. В системах с закрытым ключом сообщение (цифровая подпись) шифруется закрытым ключом, а дешифруется открытым. Все алгоритмы этого процесса являются общедоступными.

Односторонняя функция должна иметь «потайную дверь», то есть должен существовать эффективный способ ее вычисления в обоих направлениях, при этом знание прямого преобразования не должно позволять легко найти обратное преобразование. Нахождение x из $f(x)$ трудновычислимо только для криптоаналитика. Законный получатель информации знает лазейку - «потайную дверь». Такие односторонние функции называют криптографическими.

Не известен ни один пример строго криптографической односторонней функции. Однако известно много криптографических функций $f(x)$, удовлетворяющих следующим условиям:

- 1) по заданному x легко вычисляется $f(x)$;
- 2) нахождение x по известному значению $f(x)$, вероятно, является трудновычислимой задачей.

Широко используются следующие методы построения двухключевых криптографических алгоритмов:

- умножение простых чисел;
- дискретное возведение в степень;
- упаковка множества положительных чисел (задача об укладке рюкзака).

Недостатком этих методов является большой размер шифруемого блока. А это, наряду с низкой скоростью шифрования, препятствует использованию алгоритмов с открытым ключом в потоковых шифрах. В настоящее время такие криптосистемы используются, в основном, для управления ключами и цифровой подписи.

1.2.3 Известно несколько методов шифрования на основе дискретного возведения в степень. Наиболее распространенным из них является метод дискретного возведения в степень в конечных полях. В качестве функции с «потайной дверью» можно использовать модульное возведение в степень с фиксированным показателем степени m и модулем n :

$$f(x) = x^m \bmod n.$$

Эффективный алгоритм для обратной операции – извлечения корня m -ой степени по модулю n для произвольных x^m неизвестен. Это хорошо известная задача дискретного логарифмирования для больших чисел. Для ее решения можно использовать известный алгоритм извлечения корня при разложении числа n на простые множители, набор которых и является «потайной дверью», т.е. ключом шифра. Это позволяет отнести функцию вида $f(x) = x^m \bmod n$ к классу односторонних функций с «потайной дверью».

Реализацией задачи об упаковке множества положительных чисел (укладке рюкзака) является криптоалгоритм Меркле и Хелмана. Рассмотрим его на примере. Пусть задано множество чисел

$$A = (a_1, a_2, \dots, a_n),$$

включающее n разных положительных целых чисел и еще одно положительное целое число k . Задачей является нахождение такого подмножества чисел a_i , сумма которых (если это возможно) равна числу k .

Выберем, например, число $k=3231$ и набор из 10 целых чисел

$$43, 129, 215, 473, 903, 302, 561, 1165, 697, 1523.$$

Заметим, что число k получается при сложении только чисел:

$$3231 = 129 + 473 + 903 + 561 + 1165.$$

Таким образом, сложив эти числа, мы нашли решение, то есть заполнили рюкзак.

В принципе решение всегда может быть найдено полным перебором подмножеств множества A и проверкой, какая из сумм равна числу k . В рассматриваемом случае это означает перебор $2^{10}=1024$ подмножеств (включая при этом и пустое множество), что вполне осуществимо. Если же взять, к примеру, 100 чисел, то число подмножеств будет составлять уже 2^{100} . Вероятность нахождения правильного решения в этом случае составит

$10^{-30} \cong 2^{-100}$. Криптоустойчивость шифра определяется тем, что неизвестны алгоритмы решения рассматриваемой задачи, имеющие существенно меньшую сложность по сравнению с алгоритмом полного перебора.

1.2.4 Криптоалгоритм RSA, разработан в 1977 году и получил название в честь ее создателей: Рона Ривеста, Ади Шамира и Леонарда Эйдельмана.

В основу алгоритма положены легкость вычисления больших простых чисел и практическая невыполнимость разложения на множители произведения двух таких чисел. Доказано (теорема Рабина), что раскрытие шифра RSA эквивалентно такому разложению. Поэтому для любой длины ключа можно дать нижнюю оценку числа операций для раскрытия шифра, а с учетом производительности современных компьютеров оценить и необходимое на это время.

Возможность гарантированно оценить защищенность алгоритма RSA стала одной из причин популярности этой системы. Алгоритм RSA используется в банковских компьютерных сетях, особенно для работы с удаленными клиентами (обслуживание кредитных карточек). В настоящее время алгоритм RSA используется во многих стандартах, среди которых SSL, S-HTTP, S-MIME, S/WAN, STT и PCT.

Математическую основу алгоритма составляют следующие результаты:

Теорема 1. (Малая теорема Ферма.) Если p - простое число, то $x^{p-1} \bmod p = 1$ для любого x , простого относительно p , и $x^p \bmod p = x$ для любого x .

Теорема 2. Если $n=pq$, (p и q - отличные друг от друга простые числа), то $\varphi(n)=(p-1)(q-1)$, где $\varphi(n)$ - функция Эйлера - число положительных целых, меньших n и простых относительно n .

Теорема 3. Если $n=pq$, (p и q - отличные друг от друга простые числа) и x - простое относительно p и q , то $x^{\varphi(n)} = 1 \pmod{n}$.

Следствие. Если $n=pq$, (p и q - отличные друг от друга простые числа) и e простое относительно $\varphi(n)$, то отображение $E_{e,n}: x \rightarrow x^e \pmod{n}$ является взаимно однозначным на Z_n .

Очевиден и тот факт, что если e - простое относительно $\varphi(n)$, то существует целое d , такое, что $ed = 1 \pmod{\varphi(n)}$

Пусть $n=pq$, где p и q - различные простые числа. Если e и d удовлетворяют уравнению $ed = 1 \pmod{\varphi(n)}$, то отображения $E_{e,n}$ и $E_{d,n}$ являются инверсиями на Z_n . Как $E_{e,n}$, так и $E_{d,n}$ легко рассчитываются, когда известны e , d , p , q . Если известны e и n , но p и q неизвестны, то $E_{e,n}$ представляет собой одностороннюю функцию; нахождение $E_{d,n}$ по заданному n равносильно разложению n . Если p и q достаточно большие простые числа, то разложение n практически не осуществимо. Это и заложено в основу системы шифрования RSA.

Пользователь i выбирает пару различных простых p_i и q_i и рассчитывает пару целых чисел (e_i, d_i) , которые являются простыми относительно $\varphi(n_i)$, где $n_i=p_i q_i$. Открытые ключи $\{(e_i, n_i)\}$.

Предположим, что исходный текст

$$x = (x_0, x_1, \dots, x_{n-1}), x \in Z_n, 0 \leq i < n,$$

сначала представлен по основанию n_i :

$$N = c_0 + c_1 n_i + \dots$$

Пользователь i зашифровывает текст при передаче его пользователю j , применяя к n отображение Ed_i, n_i :

$$N \rightarrow Ed_i, n_i \ n = n'.$$

Пользователь j расшифровывает n' , применяя к нему Ee_i, n_i :

$$N' \rightarrow Ee_i, n_i \ n' = Ee_i, n_i \ Ed_i, n_i \ n = n.$$

Очевидно, для того чтобы найти инверсию $E d_i, n_i$ по отношению к $E e_i, n_i$, требуется знание множителей $n = p_i q_i$. Время выполнения наилучших из известных алгоритмов разложения при $n = 10^{100}$ на сегодняшний день выходит за пределы современных технологических возможностей.

Рассмотрим небольшой пример, иллюстрирующий применение алгоритма RSA.

Пример Зашифруем сообщение “СAB”. Для простоты будем использовать маленькие числа (на практике применяются гораздо большие).

1. Выберем $p=3$ и $q=11$.
2. Определим $n=3*11=33$.
3. Найдем $(p-1)(q-1)=20$. Возьмем в качестве d , взаимно простое с 20 число, например, $d=3$.
4. Выберем число e . В качестве него может быть взято любое число, для которого удовлетворяется соотношение $(e*3) \pmod{20} = 1$, например 7.
5. Представим шифруемое сообщение как последовательность целых чисел с помощью отображения: $A \rightarrow 1, B \rightarrow 2, C \rightarrow 3$. Тогда сообщение принимает вид (3,1,2). Зашифруем сообщение с помощью ключа $\{7,33\}$.

$$\text{ШТ1} = (3^7) \pmod{33} = 2187 \pmod{33} = 9,$$

$$\text{ШТ2} = (1^7) \pmod{33} = 1 \pmod{33} = 1,$$

$$\text{ШТ3} = (2^7) \pmod{33} = 128 \pmod{33} = 29.$$

6. Расшифруем полученное зашифрованное сообщение (9,1,29) на основе закрытого ключа $\{3,33\}$:

$$\text{ИТ1} = (9^3) \pmod{33} = 729 \pmod{33} = 3,$$

$$\text{ИТ2} = (1^3) \pmod{33} = 1 \pmod{33} = 1,$$

$$\text{ИТ3} = (29^3) \pmod{33} = 24389 \pmod{33} = 2.$$

В реальных системах алгоритм RSA реализуется следующим образом: каждый пользователь выбирает два больших простых числа, и в соответствии с описанным выше алгоритмом выбирает два простых числа e и d . Как результат умножения первых двух чисел (p и q) устанавливается n .

$\{e, n\}$ образует открытый ключ, а $\{d, n\}$ - закрытый (хотя можно взять и наоборот).

Открытый ключ публикуется и доступен каждому, кто желает послать владельцу ключа сообщение, которое зашифровывается указанным алгоритмом. После шифрования, сообщение невозможно раскрыть с помощью открытого ключа. Владелец же закрытого ключа без труда может расшифровать принятое сообщение.

Алгоритм RSA активно реализуется как в виде самостоятельных криптографических продуктов, так и в качестве встроенных средств в популярных приложениях.

Проблемой практической реализации является генерация больших простых чисел. Решение задачи «в лоб» - генерация случайного большого числа n (нечетного) и проверка его делимости на множители от 3 вплоть до $n^{0.5}$. В случае неудачи берется $n+2$ и так далее.

В принципе в качестве p и q можно использовать «почти» простые числа, то есть числа для которых вероятность того, что они простые, стремится к 1. Однако, если использовано составное число, а не простое, криптостойкость RSA падает. Известны алгоритмы, которые позволяют генерировать «почти» простые числа с уровнем доверия 2^{-100} .

Другой проблемой является выбор длины ключей? Для практической реализации алгоритмов RSA полезно знать оценки трудоемкости разложения простых чисел различной длины, сделанные Шроппелем (табл.1.6). В конце 1995 года удалось практически реализовать раскрытие шифра RSA для 500-значного ключа. Для этого с помощью сети Интернет было задействовано 1600 компьютеров.

Авторы RSA рекомендуют использовать следующие размеры модуля n :

- 768 бит - для частных лиц;
- 1024 бит - для коммерческой информации;

- 2048 бит - для особо секретной информации.

Таблица 1.6

$\log_{10} n$	Число операций	Примечания
50	$1.4 \cdot 10^{10}$	Раскрываем на суперкомпьютерах
100	$2.3 \cdot 10^{15}$	На пределе современных технологий
200	$1.2 \cdot 10^{23}$	За пределами современных технологий
400	$2.7 \cdot 10^{34}$	Требуем существенных изменений в технологии
800	$1.3 \cdot 10^{51}$	Не раскрываем

Третий немаловажный аспект реализации RSA - *вычислительный*. Ведь приходится использовать аппарат длинной арифметики. Если используется ключ длиной k бит, то для операций по открытому ключу требуется $O(k^2)$ операций, по закрытому ключу - $O(k^3)$ операций, а для генерации новых ключей требуется $O(k^4)$ операций.

1.3 Алгоритмы электронной цифровой подписи

1.3.1 Технологическое применение электронной цифровой подписи (ЭЦП) предполагает наличие сети абонентов, посылающих друг другу подписанные электронные документы. Для каждого абонента генерируется пара ключей: секретный и открытый. Секретный ключ хранится абонентом в тайне и используется им для формирования ЭЦП. Открытый ключ известен всем остальным пользователям и предназначен для проверки ЭЦП получателем подписанного электронного документа. Открытый ключ не позволяет вычислить секретный ключ.

Для генерации пары ключей (секретного и открытого) в алгоритмах ЭЦП, как и в асимметричных системах шифрования, используются разные математические схемы, основанные на применении однонаправленных функций. Эти схемы разделяются на две группы. В основе такого разделения лежат известные сложные вычислительные задачи:

- задача факторизации (разложение на множители) больших целых чисел;
- задача дискретного логарифмирования.

1.3.2. При реализации ЭЦП для сокращения времени подписывания и размера подписи в качестве источника подписи служат не само исходное сообщение M произвольной длины, а некоторая функция от него фиксированной длины, которую называют хеш-функцией. Хеш-функция $h(\cdot)$ использует в качестве аргумента сообщение (документ) M произвольной длины и возвращает хеш-значение $h(M) = H$ фиксированной длины. Обычно хешированная информация является сжатым двоичным представлением основного сообщения произвольной длины. Следует отметить, что значение хеш-функции $h(M)$ сложным образом зависит от документа M и не позволяет восстановить сам документ M .

Хеш-функция должна удовлетворять ряду условий:

- она должна быть чувствительна к всевозможным изменениям в тексте M , таким как вставки, выбросы, перестановки и т. п.;
- она должна обладать свойством необратимости, то есть задача подбора документа M' , который обладал бы требуемым значением хеш-функции, должна быть вычислительно неразрешима;
- вероятность того, что значения хеш-функций двух различных документов (вне зависимости от их длины) совпадут, должна быть ничтожно мала.

Большинство хеш-функций строится на основе однонаправленной функции $f(\cdot)$, которая образует выходное значение длиной n при задании двух входных значений длиной n . Этими входами являются блок исходного текста M_i и хеш-значение H_{i-1} предыдущего блока текста (рис. 1.4).

$$H_i = f(M_i, H_{i-1}).$$

Хеш-значение, вычисляемое при вводе последнего блока текста, становится хеш-значением всего сообщения M .

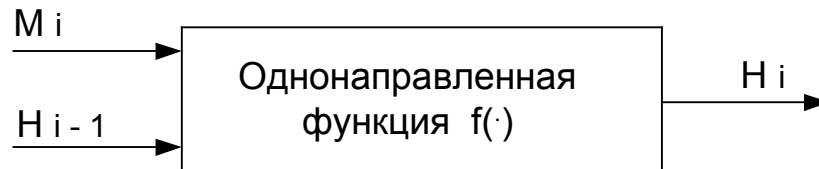


Рис. 1.4

Однонаправленная хеш-функция всегда формирует число фиксированной длины n независимо от длины входного текста.

1.3.3 Алгоритм цифровой подписи DSA (Digital Signature Algorithm) предложен в 1991 г. в НИСТ США для использования в стандарте цифровой подписи DSS (Digital Signature Standard). Он является развитием алгоритмов цифровой подписи Эль Гамала и К.Шнорра.

Отправитель и получатель электронного документа используют при вычислении большие целые числа: G и P - простые числа, L бит каждое ($512 \leq L \leq 1024$); q - простое число длиной 160 бит (делитель числа $(P - 1)$).

Причем G выбирается таким образом, что

$$G = h^{(p-1)/q} \bmod P.$$

Где h -любое число, меньшее $P-1$, такое, что $h^{(p-1)/q} \bmod P > 1$.

Числа G , P , q являются открытыми и могут быть общими для всех пользователей сети.

Отправитель выбирает случайное целое число X , $1 < X < q$. Число X является секретным ключом отправителя для формирования электронной цифровой подписи.

Затем отправитель вычисляет значение

$$Y = G^X \bmod P.$$

Число Y является открытым ключом для проверки подписи отправителя. Число Y передается всем получателям документов.

Этот алгоритм также предусматривает использование односторонней функции хэширования $h(-)$. В стандарте DSS определен алгоритм безопасного хэширования SHA (Secure Hash Algorithm).

Для того чтобы подписать документ M , отправитель хэширует его в целое хэш-значение m :

$$m = h(M), 1 < m < q,$$

затем генерирует случайное целое число K , $1 < K < q$, и вычисляет число r

$$r = (G^K \bmod P) \bmod q.$$

Затем отправитель вычисляет с помощью секретного ключа X целое число s :

$$S = [(m + r \cdot X)/k] \bmod q.$$

Пара чисел r и s образует цифровую подпись $S=(r,s)$ документа M .

Таким образом, подписанное сообщение представляет собой тройку чисел $[M, r, s]$.

Получатель подписанного сообщения $[M, r, s]$ проверяет выполнение условий

$$0 < r < q, 0 < s < q$$

и отвергает подпись, если хотя бы одно из этих условий не выполнено.

Затем получатель вычисляет вспомогательное число $w = (1/s) \bmod q$, хэш-значение $m=h(M)$ и числа

$$u_1 = (m \cdot w) \bmod q,$$

$$u_2 = (r \cdot w) \bmod q.$$

Далее с помощью открытого ключа Y получатель вычисляет значение

$$v = ((G^{u_1} \cdot Y^{u_2}) \bmod P) \bmod q$$

и проверяет выполнение условия $v = r$. Если это условие выполняется, то подпись $S = (r,s)$ под документом M признается получателем подлинной.

Существует вариант оптимизации, упрощающий вычисление подписи и позволяющий на практике проверить работу алгоритма. Все используемые параметры – такие же, но

для вычисления подписи генерируется два случайных числа k и d , меньшие q . Процедура вычисления подписи выглядит следующим образом:

$$\begin{aligned}r &= (G^k \bmod P) \bmod q, \\S &= [(m + r \cdot X) \cdot d] \bmod q, \\t &= (k \cdot d) \bmod q.\end{aligned}$$

Проверка подписи:

$$\begin{aligned}w &= t/s \bmod q \\u_1 &= (m \cdot w) \bmod q, \\u_2 &= (r \cdot w) \bmod q.\end{aligned}$$

Если $r = v = ((G^{u_1} \cdot Y^{u_2}) \bmod P) \bmod q$, то подпись правильна.

Последнее равенство будет выполняться тогда, и только тогда, когда подпись $S = (r, s)$ под документом M получена с помощью именно того секретного ключа X , из которого был получен открытый ключ Y . Таким образом, можно надежно удостовериться, что отправитель сообщения владеет именно данным секретным ключом X (не раскрывая при этом значения ключа X) и что отправитель подписал именно данный документ M .

По сравнению с алгоритмом цифровой подписи Эль Гамала алгоритм DSA имеет следующие основные преимущества:

1. При любом допустимом уровне стойкости, т.е. при любой паре чисел G и P (от 512 до 1024 бит), числа q , X , g , s имеют длину по 160 бит, сокращая длину подписи до 320 бит.
2. Большинство операций с числами K , g , s , X при вычислении подписи выполняется по модулю числа q длиной 160 бит, что сокращает время вычисления подписи.
3. При проверке подписи большинство операций с числами U_1 , U_2 , v , w также выполняется по модулю числа q длиной 160 бит, что сокращает объем памяти и время вычислений.

Недостатком алгоритма DSA является необходимость и при подписывании и при проверке подписи выполнять сложные операции деления по модулю q :

$$S = [(m + r \cdot X)/k] \bmod q, w = 1/s \bmod q.$$

Выполнение алгоритма DSA можно ускорить с помощью предварительных вычислений. Заметим, что значение g не зависит от сообщения M и его хэш-значения m . Можно заранее создать строку случайных значений K и затем для каждого из этих значений вычислить значения g . Можно также заранее вычислить обратные значения K^{-1} для каждого из значений K . Затем, при поступлении сообщения M , можно вычислить значение s для данных значений g и K^{-1} . Эти предварительные вычисления значительно ускорят работу алгоритма DSA.

1.3.4 Отечественный стандарт цифровой подписи (ГОСТ Р 34. 10-94).

Отечественный стандарт цифровой подписи обозначается как ГОСТ Р 34. 10-94. Алгоритм цифровой подписи, определяемый этим стандартом, концептуально близок к алгоритму DSA. В нем используются следующие параметры:

- p – большое простое число длиной от 509 до 512 либо от 1020 до 1024;
- q – простой сомножитель числа $(p - 1)$, имеющий длину 254...256 бит;
- a – любое число, меньшее $(p - 1)$, причем такое, что $a^q \bmod p = 1$;
- x – некоторое число, меньшее q ;
- $y = a^x \bmod p$.

Кроме того, этот алгоритм использует однонаправленную хеш-функцию $H(x)$. Стандарт ГОСТ Р 34. 11-94 определяет хеш-функцию, основанную на использовании стандартного симметричного алгоритма ГОСТ 28147-89.

Первые три параметра p , q и a являются открытыми и могут быть общими для всех пользователей сети. Число x является секретным ключом. Число y является открытым ключом.

Чтобы подписать некоторое сообщение m , а затем проверить подпись, выполняются следующие шаги.

1. пользователь A генерирует случайное число k , причем $k < q$.
2. пользователь A вычисляет значения

$$r = (a^k \bmod p) \bmod q,$$

$$s = (x \cdot r + k(H(m))) \bmod q.$$

Если $H(m) \bmod q = 0$, то значение $H(m) \bmod q$ принимают равным единице. Если $r = 0$, то выбирают другое значение k и начинают снова. Цифровая подпись представляет собой два числа:

$$r \bmod 2^{256} \text{ и } s \bmod 2^{256},$$

которые отправляются пользователю В.

3. Пользователь В проверяет полученную подпись, вычисляя

$$v = H(m)^{q-2} \bmod q,$$

$$z_1 = (s \cdot v) \bmod q,$$

$$z_2 = ((q - r) \cdot v) \bmod q,$$

$$u = ((a^{z_1} \cdot y^{z_2}) \bmod p) \bmod q.$$

Если $u = r$, то подпись считается верной.

Различие между этим алгоритмом и алгоритмом DSA заключается в том, что в DSA

$$s = (k^{-1}(x \cdot r + (H(m)))) \bmod q,$$

что приводит к другому уравнению верификации.

В отечественном стандарте ЭЦП параметр q имеет длину 256 бит. Западных криптографов вполне устраивает q длиной примерно 160 бит. Этот стандарт вступил в действие с начала 1995 г.

1.4 Техническая надежность использования криптосистем

Уровень криптостойкости системы зависит от комплекса факторов.

Можно выделить следующие основные группы причин ненадежности криптографических систем:

- Применение нестойких алгоритмов
- Ошибки в реализации криптоалгоритмов
- Неправильное применение криптоалгоритмов
- Человеческий фактор.

Для обеспечения надежной работы системы все эти задачи должны решаться в комплексе. Технические аспекты проблемы обусловлены неправильным применением криптоалгоритмов и ошибками в их реализации

На сегодняшний день известен и апробирован ряд криптоалгоритмов (как с симметричными, так и несимметричными ключами), криптостойкость которых либо доказана математически, либо основана на необходимости решения математически сложной задачи (факторизации, дискретного логарифмирования и т. п.). К наиболее известным из них относятся DES, RSA, ГОСТ. Они могут быть вскрыты либо полным перебором ключей, либо решением указанной задачи. Однако, ошибки в реализации этих алгоритмов могут привести к снижению криптостойкости системы. К числу наиболее распространенных ошибок можно отнести следующие:

- 1) применение недостаточно криптостойких ключей;
- 2) программные ошибки и “люки”;
- 3) низкое качество датчиков случайных чисел (ДСЧ).

Снижение криптостойкости ключей может быть вызвано как недостаточной длиной ключей, так и применением ключей, не удовлетворяющих требованиям криптоустойчивости. Известно множество таких ключей для стандарта DES.

К снижению надежности криптосистемы могут привести программные ошибки и “люки” – потайные варианты входа в криптосистему, которые могут быть оставлены ее разработчиками как для устранения аварийных ситуаций, так и с другими целями.

2. Домашняя подготовка

1. Изучить симметричные криптоалгоритмы.
2. Зашифровать произвольную фразу длиной 20-25 символов с помощью симметричных криптоалгоритмов.
3. Расшифровать зашифрованный в п.2 криптотекст.

4. Изучить несимметричные криптоалгоритмы.
5. Найти открытые и закрытые ключи для алгоритма RSA, зашифровать и расшифровать с их помощью произвольный текст.

3. Лабораторное задание

1. Лабораторное исследование выполняется с помощью программы Lab_Crypt, которая позволяет исследовать симметричные (замены, перестановки, Вижинера, составные) и несимметричные (RSA в варианте цифровой подписи) криптоалгоритмы. Программа Lab_Crypt.exe является управляющей оболочкой для модулей симметричных (Crypt_Simm.exe) и несимметричных (RSA.exe) криптоалгоритмов. Все эти модули должны находиться в одной папке. Помимо них там же должны находиться вспомогательные файлы Crypt.hlp, RSA.hlp, а также word.txt и keyword.txt.

2. Загрузить программу Lab_Crypt.exe в память ЭВМ, изучить ее интерфейс и правила проведения исследований.

3. Исследовать симметричные криптоалгоритмы в режиме демонстрации, а затем перейти в режим контроля и проверить правильность шифрования контрольных текстов, использованных при домашней подготовке, а также задаваемых преподавателем или коллегой по работе.

4. Исследовать алгоритм RSA в варианте цифровой подписи. Ввести в окно "Текст документа" текст сообщения, задать константы P и Q для формирования ключей и подписать документ. Исследовать, как изменяется цифровая подпись документа в зависимости от изменения констант P и Q при неизменном исходном тексте. Затем провести исследование зависимости цифровой подписи от текста документа при неизменных P и Q .

5. Передать подписанный документ получателю и проверить правильность подписи. Исправить открытые ключи E или N и снова проверить подпись. Восстановите ключи. Исправить принятый документ и проверить подпись. Проведите аналогичные эксперименты с исходным документом после его подписания но до передачи получателю.

6. Оформить отчет по работе и сделать выводы по работе.

4 Содержание отчета по работе

Отчет по работе должен содержать:

- а) результаты выполнения домашнего задания: исходные тексты и криптотексты для каждого алгоритма шифрования;
- б) результаты выполнения лабораторного задания: сопоставление теоретических и экспериментальных результатов;
- в) анализ результатов работы и выводы по ней.

5 Контрольные вопросы

1. Каким образом классифицируются криптоалгоритмы? От чего зависит стойкость криптоалгоритма?
2. Какие симметричные криптоалгоритмы вы знаете? Поясните принципы их работы на примерах.
3. Какие несимметричные криптоалгоритмы вы знаете? Поясните принципы их работы. Каким требованиям должны удовлетворять односторонние функции?
4. Приведите пример расчета открытых и закрытых ключей для алгоритма RSA.
5. Проиллюстрируйте процедуру шифрования и расшифрования информации для алгоритма RSA.
6. Математические основы алгоритма RSA. Какие требования предъявляются к константам P и Q ?
7. Какие проблемы возникают при технической реализации алгоритма RSA? Возможна ли его аппаратная реализация?
8. Поясните принцип работы систем цифровой подписи на основе алгоритма DSA.
9. Каким образом вычисляются ключи в алгоритме DSA?

10. От чего зависит криптостойкость несимметричных криптоалгоритмов?

Литература

1. Шнайер Б. Прикладная криптография.
2. Питерсон У. Коды, исправляющие ошибки. М.: Мир, 1977.
3. Ю. В. Романцев, П. А. Тимофеев, В. Ф. Шаньгин: Защита информации в компьютерных системах и сетях./ Под. ред д. т. н. проф. В. Ф. Шаньгина.: М. «Радио и связь», 1999.
4. Журнал «Вопросы защиты информации», № 4.: М. – 2001.
5. ГОСТ Р 34.11-94
6. ГОСТ Р 34.10-94
7. ГОСТ 28147-89