**Тема:**

**«Шифрование и дешифрование данных при помощи симметричных криптографических алгоритмов»**

Алгоритмы шифрования и дешифрования данных широко применяются в компьютерной технике в системах сокрытия конфиденциальной и коммерческой информации от злонамеренного использования сторонними лицами. Главным принципом в них является условие, что передатчик и приемник заранее знают алгоритм шифрования, а также ключ к сообщению, без которых информация представляет собой всего лишь набор символов, не имеющих смысла.

Классическим примером таких алгоритмов являются симметричные криптографические алгоритмы, перечисленные ниже:

*Подстановки:*

* Подстановка Цезаря
* Квадрат Полибия.
* "Тюремный шифр"
* Шифр Плэйфера
* Двойной квадрат

### Метод переименования

### Метод псевдослучайной инверсии

* Шифром Гронсфельда
* Система шифрования Вижинера

### Шифр Бофора

* Шифр с автоключом.
* Шифр машины Энигма

*Гаммирование:*

* Шифр гаммирования по Лемеру
* Конгруэнтные датчики ПСЧ для гаммирования
* Целочисленные датчики (ряд Фибоначчи) для гаммирования
* Датчики М-последовательностей для гаммирования

### Метод псевдослучайного гаммирования

### Метод цепочечного гаммирования

*Перестановки:*

### Простая перестановка

### Одиночная перестановка по ключу

* Двойная перестановка
* Двойная перестановка столбцов и строк
* Перестановка “Магический квадрат”

### Метод "спутанной шины"

* Многомерная перестановка
* Шифр взбивания.

Рассмотрим примеры некоторых из них ниже.

### **Простая перестановка**

Простая перестановка без ключа - один из самых простых методов шифрования. Сообщение записывается в таблицу по столбцам. После того, как открытый текст записан колонками, для образования шифровки он считывается по строкам. Для использования этого шифра отправителю и получателю нужно договориться об общем ключе в виде размера таблицы. Объединение букв в группы не входит в ключ шифра и используется лишь для удобства записи несмыслового текста.

### **Одиночная перестановка по ключу**

Более практический метод шифрования, называемый одиночной перестановкой по ключу очень похож на предыдущий. Он отличается лишь тем, что колонки таблицы переставляются по ключевому слову, фразе или набору чисел длиной в строку таблицы.

**Двойная перестановка**

Для дополнительной скрытности можно повторно шифровать сообщение, которое уже было зашифровано. Этот способ известен под названием двойная перестановка. Для этого размер второй таблицы подбирают так, чтобы длины ее строк и столбцов были другие, чем в первой таблице. Лучше всего, если они будут взаимно простыми. Кроме того, в первой таблице можно переставлять столбцы, а во второй строки. Наконец, можно заполнять таблицу зигзагом, змейкой, по спирали или каким-то другим способом. Такие способы заполнения таблицы если и не усиливают стойкость шифра, то делают процесс шифрования гораздо более занимательным.

##### Двойная перестановка столбцов и строк

Кроме одиночных перестановок использовались еще двойные перестановки столбцов и строк таблицы с сообщением. При этом перестановки определялись отдельно для столбцов и отдельно для строк. В таблицу вписывался текст и переставлялись столбцы, а потом строки. При расшифровке порядок перестановок был обратный.

**Перестановка “Магический квадрат”**

Магическими квадратами называются квадратные таблицы со вписанными в их клетки последовательными натуральными числами от 1, которые дают в сумме по каждому столбцу, каждой строке и каждой диагонали одно и то же число. Подобные квадраты широко применялись для вписывания шифруемого текста по приведенной в них нумерации. Если потом выписать содержимое таблицы по строкам, то получалась шифровка перестановкой букв. На первый взгляд кажется, будто магических квадратов очень мало. Тем не менее, их число очень быстро возрастает с увеличением размера квадрата. Так, существует лишь один магический квадрат размером 3 х 3, если не принимать во внимание его повороты. Магических квадратов 4 х 4 насчитывается уже 880, а число магических квадратов размером 5 х 5 около 250000. Поэтому магические квадраты больших размеров могли быть хорошей основой для надежной системы шифрования того времени, потому что ручной перебор всех вариантов ключа для этого шифра был немыслим.

В квадрат размером 4 на 4 вписывались числа от 1 до 16. Его магия состояла в том, что сумма чисел по строкам, столбцам и полным диагоналям равнялась одному и тому же числу - 34. Впервые эти квадраты появились в Китае, где им и была приписана некоторая "магическая сипа".

|  |  |  |  |
| --- | --- | --- | --- |
| 16 | 3 | 2 | 13 |
| 5 | 10 | 11 | 8 |
| 9 | 6 | 7 | 12 |
| 4 | 15 | 14 | 1 |

Шифрование по магическому квадрату производилось следующим образом. Например, требуется зашифровать фразу: "Приезжаю сегодня". Буквы этой фразы вписываются последовательно в квадрат согласно записанным в них числам, а в пустые клетки ставится точка.

|  |  |  |  |
| --- | --- | --- | --- |
| 16. | 3 И | 2 Р | 13 Д |
| 5 З | 10 Е | 11 Г | 8 Ю |
| 9 С | 6 Ж | 7 А | 12 О |
| 4 Е | 15 Я | 14 Н | 1 П |

После этого шифрованный текст записывается в строку:

БИРДЗЕГЮСЖАОЕЯНП

При расшифровывании текст вписывается в квадрат, и открытый текст читается в последовательности чисел "магического квадрата"

Программа должна генерировать «магические квадраты» и по ключу выбирать необходимый. Размер квадрата больше чем 3х3.

### **Метод "спутанной шины"**

Ключом является трехбайтовая последовательность. Этот ключ разбивается на 8 зон по 3 бита, каждая зона хранит адрес этого бита в новом байте (см. рис)

 Байт 1 Байт 2 Байт 3



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Бит1 Бит2 Бит3 Бит4 Бит5 Бит6 Бит7 Бит8

Для удобства реализации и для увеличения быстродействия метода можно использовать массив масок вместо сдвигов байт.

**Многомерная перестановка**

В 1991 г. В.М.Кузьмч предложил схему перестановки, основанной на кубике Рубика. Согласно этой схеме открытый текст записывается в ячейки граней куба по строкам. После осуществления заданного числа заданных поворотов слоев куба считывание шифртекста осуществляется по столбикам. Сложность расшифрования в этом случае определяется количеством ячеек на гранях куба и сложностью выполненных поворотов слоев. Перестановка, основанная на кубике Рубика, получила название объемной (многомерной) перестановки.

В 1992-1994 г.г. идея применения объемной перестановки для шифрования открытого текста получила дальнейшее развитие. Усовершенствованная схема перестановок по принципу кубика Рубика, в которой наряду с открытым текстом перестановке подвергаются и функциональные элементы самого алгоритма шифрования, легла в основу секретной системы "Рубикон". В качестве прообразов пространственных многомерных структур, на основании объемных преобразований которых осуществляются перестановки, в системе "Рубикон" используются трехмерные куб и тэтраэдр. Другой особенностью системы "Рубикон" является генерация уникальной версии алгоритма и ключа криптографических преобразований на основании некоторого секретного параметра (пароля). Это обеспечивает как дополнительные трудности для криптоанализа перехваченных сообщений нарушителем (неопределенность примененного алгоритма), так и возможность априорного задания требуемой криптостойкости. Криптостойкость данной системы определяется длиной ключа, криптостойкостью отдельных функциональных элементов алгоритма криптографических преобразований, а также количеством таких преобразований.

Шифр взбивания

Результат шифрования можно ощутимо улучшить, если вместо перестановки использовать линейное преобразование: s=L\*t

где L - невырожденная матрица случайного линейного преобразования бит, или, что то же самое, детерминант L не равен нулю. И хотя расшифровывание в этом случае придется осуществлять решением систем линейных уравнений, но каждый бит шифровки начинает уже зависеть от каждого бита текста. Шифры на основе этого преобразования называют скрамблерами или взбивалками. К сожалению, доля невырожденных матриц с увеличением их размера стремительно убывает. Детерминант матрицы L, как и ее элементы, может быть равен либо 0, либо 1. Если det(L)=0, то матрица вырождена.

для матрицы, составленной из квадратных подматриц А, В, С и D, имеем:

¦А В¦

¦ ¦ =det(A)det(B)-det(C)-det(D)

¦C D¦

Для того, чтобы матрица L была невырожденной, случайной и при расшифровании не нужно было производить много вычислений, американскими криптографами был предложен алгоритм, легший в основу стандартного криптографического преобразования DES. Суть его одного шага можно описать следующей схемой. Входной блок данных делится пополам на левую L' и правую R' части. После этого формируется выходной массив так, что его левая часть L" представлена правой частью R' входного, а правая R" формируется как сумма L' и R' операцией XOR. Далее, выходной массив шифруется перестановкой с заменой. Можно убедиться, что все проведенные операции могут быть обращены и расшифровывание осуществляется за число операций, линейно зависящее от размера блока. В то же самое время, после нескольких таких взбиваний можно считать, что каждый бит выходного блока шифровки может зависеть от каждого бита сообщения. С увеличением числа взбиваний порча единственного бита в шифровке делает нечитаемой половину текста, что обусловлено побайтовой перестановкой. Если бы перестановка была побитовая, то весь текст от ошибки в единственном бите перестал бы читаться.

'------программа шифрования взбиванием

DEFINT I-N: DEFSTR S

RANDOMIZE 231

CLS: LOCATE 1, 1

Lot = 5

s$ = ""

FOR i=1 TO 64:s$=s$+CHR$(65+25\*RND):NEXT

PRINT s$; " - text": sav = s$

s$ = ""

FOR i=1 TO 192: s$=s$+CHR$(255\*RND): NEXT

'---------------------шифрование

FOR i = 0 TO Lot

sc=MID$(ss,1+I\*32,32)

l=2^i:sl="": r=""

FOR j = 1 TO 32

kg=ASC(MID$(sc, j, 1))

kl=ASC(MID$(s$, j, 1))

kr=ASC(MID$(s$, j+32,1))

sl = sl+ CHR$(kl XOR kr)

sr = sr+ CHR$(kr XOR kg)

NEXT

s$=sr+RIGHT$(sl,l)+LEFT$(sl,32-l)

 NEXT

'----------------------порча бита

ss=CHR$(ASC(s$) XOR 4)+RIGHT$(s$,63)

'-----------------печать шифровки

FOR i =1 TO 64

k = ASC(MID$(s$, i, 1))

DEF SEG=47114: POKE 2\*i-2, k: DEF SEG

NEXT

LOCATE 2, 65: PRINT " - code"

'---------------расшифровывание

FOR i = Lot TO 0 STEP -1

sc=MID$(s$, 1+i\*32, 32): l=2^i

s$=RIGHT$ (s$ ,32- l)+MID$ (s$, 33,l)+LEFT$ (s$, 32)

sl = "": sr = ""

 FOR j = 1 TO 32

 kg = ASC(MID$(sc, j, 1))

 kl = ASC(MID$(ss, j, 1))

 kr = ASC(MID$(ss, j+32, 1))

 sl = sl+ CHR$(kl XOR kr XOR kg)

 sr = sr+ CHR$(kr XOR kg)

 NEXT

ss = sl+sr

NEXT

FOR i =1 TO 64

k = ASC(MID$(s$, i, 1))

DEF SEG=47124: POKE 2\*i-2,k: DEF SEG

NEXT

LOCATE 3, 65: PRINT " - text"

n = 0

FOR i =1 TO 64

 IF MID$ (s$, i, 1) =MID$(sav, i,1) THEN

 LOCATE 4, i: PRINT "+";: n = n+I

 ELSE

 LOCATE 4, i: PRINT "-";

 END IF

NEXT

LOCATE 6, 1: PRINT 64 - n; "errors"

END

**Шифр Цезаря**

Подстановка Цезаря является самым простым вариантом подстановки. Она относится к группе *моноалфавитных подстановок*.

При моноалфавитной замене каждой букве алфавита открытого текста ставится в соответствие одна буква шифртекста из этого же алфавита.

*Определение*. Подмножество Cm={C*k*: 0Ј*k*<m} симметрической группы SYM(Zm), содержащее *m* подстановок C*k*: j®(j+*k*) (mod *m*), 0Ј*k* < *m*, называется подстановкой Цезаря.

Подстановки приведены в Табл. 1. Стрелка (а) означает, что буква исходного текста (слева) шифруется при помощи C3 в букву шифрованного текста (справа).

Определение. Системой Цезаря называется моноалфавитная подстановка, преобразующая n-грамму исходного текста (x0, x1 ,..,xn-1) в n‑грамму шифрованного текста (y0 ,y1 ,...,yn-1) в соответствии с правилом

yi=C*k*(xi), 0Јi<n.

Например, ВЫШЛИТЕ\_НОВЫЕ\_УКАЗАНИЯ посредством подстановки C3 преобразуется в еюыолхиврсеюивцнгкгрлб.

*Таблица 1*.

|  |  |  |  |
| --- | --- | --- | --- |
| Ааг | Йам | Тах | Ыаю |
| Бад | Кан | Уац | Ьая |
| Вае | Лао | Фач | Эа\_ |
| Гаж | Мап | Хаш | Юаа |
| Даз | Нар | Цащ | Яаб |
| Еаи | Оас | Чаъ | \_ав |
| Жай | Пат | Шаы |  |
| Зак | Рау | Щаь |  |
| Иал | Саф | Ъаэ |  |

Основным недостатком рассмотренного метода является то, что статистические свойства открытого текста (частоты повторения букв) сохраняются в шифртексте.

При своей несложности система легко уязвима. Если злоумышленник имеет

1. шифрованный и соответствующий исходный текст или
2. шифрованный текст выбранного злоумышленником исходного текста, то определение ключа и дешифрование исходного текста тривиальны.

Квадрат Полибия

Применительно к современному латинскому алфавиту из 26 букв шифрование по этому квадрату заключалось в следующем. В квадрат размером 5x6 клеток выписываются все буквы алфавита, при этом буквы I,J не различаются (J отождествляется с буквой I);

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| A | A | B | C | D | E |
| D | F | G | H | I | K |
| C | L | M | N | O | P |
| D | Q | R | S | T | U |
| E | V | W | X | Y | Z |

Шифруемая буква заменялась на координаты квадрата, в котором она записана. Так, B заменялась на AB, F на BA, R на DB и т.д. При расшифровании каждая такая пара определяла соответствующую букву сообщения. Ключом такого шифра являлось расположение букв в таблице к примеру 5x5. Начальное расположение букв должно определяться ключом.

**"Тюремный шифр"**

В несколько измененном виде шифр Полибия получил своеобразное название "тюремный шифр". Для его использования нужно только знать естественный порядок расположения букв алфавита (как в указанном выше примере для английского языка). Стороны квадрата обозначаются не буквами (ABCDE), а числами (12345). Число 3, например, передается путем тройного стука. При передаче буквы сначала "отстукивается число, соответствующее строке, в которой находится буква, а затем номер соответствующего столбца. Например, буква "F" передается двойным стуком (вторая строка) и затем одинарным (первый столбец).

Отметим, что при произвольном расположении букв в квадрате возникает одно затруднение: либо нужно помнить отправителю и получателю сообщения заданный произвольный порядок следования букв в таблице (ключ шифра), что вообще говоря затруднительно, либо иметь при себе запись этих букв. Во втором случае появляется опасность ознакомления с ключом посторонних лиц. Поэтому в ряде случаев ключ составляется следующим образом. Берется некоторое "ключевое слово", которое легко запомнить, например, "CRYPTOLOGY", удаляют из него повторы букв (получают "CRYPTOLOG") и записывают его в начальных клетках квадрата. В оставшиеся клетки записываются остальные буквы алфавита в естественном порядке.

A B C D E A C R Y P T B O L G A B C D E F H I E U V W X Z

В таком шифре ключом является указанное "ключевое слово" ("пароль").

Шифр Плэйфера (Биграммы)

Более эффективны обобщения подстановки Цезаря - шифр Хилла и шифр Плэйфера. Они основаны на подстановке не отдельных символов, а 2-грамм (шифр Плэйфера) или n-грамм (*n*-граммой называется последовательность из *n* символов алфавита.) (шифр Хилла). При более высокой криптостойкости они значительно сложнее для реализации и требуют достаточно большого количества ключевой информации.

Наиболее известный шифр биграммами называется Playfair. Он применялся Великобританией в Первую мировую войну. Опишем его на примере той же самой таблицы. Открытый текст разбивался на пары букв (биграммы) и текст шифровки строился из него по следующим двум очень простым правилам.

1. Если обе буквы биграммы исходного текста принадлежали одной колонке таблицы, то буквами шифра считались буквы, которые лежали под ними. Так биграмма УН давала текст шифровки ВЧ. Если буква открытого текста находилась в нижнем ряду, то для шифра бралась соответствующая буква из верхнего ряда и биграмма ОЯ давала шифр ШБ. (Биграмма из одной буквы или пары одинаковых букв тоже подчинялась этому правилу и текст ЕЕ давал шифр ИИ).

2. Если обе буквы биграммы исходного текста принадлежали одной строке таблицы, то буквами шифра считались буквы, которые лежали справа от них. Так биграмма ИВ давала текст шифровки КГ. Если буква открытого текста находилась в правой колонке, то для шифра бралась соответствующая буква из левой колонки и биграмма ОМ давала шифр ДН.

Если обе буквы биграммы открытого текста лежали в разных рядах и колонках, то вместо них брались такие две буквы, чтобы вся четверка их представляла прямоугольник. При этом последовательность букв в шифре была зеркальной исходной паре.

# Двойной квадрат

Шифровка биграммами, которую называют двойной квадрат, открыл новый этап в криптографии. Двойной квадрат использует сразу две таблицы, расположенные по горизонтали, а шифрование идет биграммами, как в шифре Playfair. Эти, казалось бы и не столь уж значительные изменения привели к появлению на свет новой криптографической системы ручного шифрования. Имеются две таблицы со случайно расположенными в них алфавитами. Для шифрования сообщение разбивают на биграммы. Первая буква биграммы находится в левой таблице, а вторая в правой. Затем, мысленно в таблице строится прямоугольник так, чтобы буквы биграммы лежали в его противоположных вершинах. Другие две вершины этого прямоугольника дают буквы шифровки.

Метод переименования

Суть метода - создается таблица из 256 элементов с неодинаковыми значениями (по случайному закону). Каждый байт заменяется элементом этой таблицы с индексом равным байту. После этого таблица вписывается в тело файла, а адрес таблицы в файле является ключом. Таким образом длина файла увеличивается на 256 байт. Декодирование производится в обратном порядке.

Шифр Гронсфельда

Шифры сложной замены называют многоалфавитными, так как для шифрования каждого символа исходного сообщения применяется свой шифр простой замены. Шифр Гронсфельда тоже многоалфавитный шифр - в нем 10 вариантов замены.

Состоит в модификации шифра Цезаря числовым ключом. Для этого под сообщением пишут ключ. Если ключ короче сообщения, то его повторяют циклически. Шифровку получают будто в шифре Цезаря, но отсчитывая необязательно только третью букву по алфавиту, а ту, которая сдвинута на соответствующую цифру ключа. Шифр Гронсфелвда имеет массу модификаций, претендующих на его улучшение, от курьезных, вроде записи текста шифровки буквами другого алфавита, до нешуточных, как двойное шифрование разными ключами.

Кроме этих шифров, зачастую использовался шифр простой замены, заключающийся в замене каждой буквы сообщения на соответствующую ей букву шифра. Такой шифр, популярный среди школьников, является простым кодом и вскрытие его возможно при длине шифровки всего в 20-30 букв, а при длинах текста свыше 100 символов представляет собой очень простую, но весьма увлекательную задачу.

АБВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ

А АБВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ

Б \_АБВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ

В Я\_АБВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮ

Г ЮЯ\_АБВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЪЭ

 .......

Я ВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ\_АБ

\_ БВГДЕЖЗИКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ\_А

Каждая строка в этой таблице соответствует одному шифру замены вроде шифра Юлия Цезаря для алфавита, дополненного пробелом. При шифровании сообщения его выписывают в строку, а под ним ключ. Если ключ оказался короче сообщения, то его циклически повторяют. Шифровку получают, находя символ в колонке таблицы по букве текста и строке, соответствующей букве ключа. Этот очень распространенный вид шифра сохранился до наших дней.

В компьютере такая операция соответствует сложению кодов ASCII символов сообщения и ключа по некоторому модулю. Кажется, что если таблица будет более сложной, чем циклическое смещение строк, то шифр станет надежнее. Это действительно так, если ее менять почаще, например, от слова к слову. Но составление таких таблиц, представляющих собой латинские квадраты, где любая буква встречается в строке или столбце один раз, трудоемко и его стоит делать лишь на ЭВМ. Для ручного же многоалфавитного шифра полагаются лишь на длину и сложность ключа, используя приведенную таблицу, которую можно не держать в тайне, а это упрощает шифрование и расшифровывание.

Системы шифрования Вижинера

Начнем с конечной последовательности ключа

*k* = (*k*0 ,*k*1 ,...,*k*n),

которая называется *ключом пользователя*, и продлим ее до бесконечной последовательности, повторяя цепочку. Таким образом, получим *рабочий ключ*

*k* = (*k*0 ,*k*1 ,...,*k*n), *k*j = *k*(jmod *r*, 0 Ј j < Ґ .

Например, при r = Ґ и ключе пользователя 15 8 2 10 11 4 18 рабочий ключ будет периодической последовательностью:

15 8 2 10 11 4 18 15 8 2 10 11 4 18 15 8 2 10 11 4 18 ...

Определение. Подстановка Вижинера VIGk определяется как

VI*Gk* : (x0, *x*1, ..., *x*n-1) ® (y0, *y*1, ..., *y*n-1) = (x0+*k*, *x*1+*k*,. .., *x*n-1+*k*).

Таким образом:

1) исходный текст x делится на r фрагментов

 *x*i = (xi , *x*i+*r* , ..., *x*i+*r*(n-1)), 0 Ј i < *r*;

2) i-й фрагмент исходного текста xi шифруется при помощи подстановки Цезаря Ck :

(xi , *x*i+*r* , ..., *x*i+*r*(n-1)) ® (yi , *y*i+*r* , ..., *y*i+*r*(n-1)),

Вариант системы подстановок Вижинера при m=2 называется системой Вернама (1917г).

В то время ключ k=(k0 ,k1 ,...,kк-1) записывался на бумажной ленте. Каждая буква исходного текста в алфавите, расширенном некоторыми дополнительными знаками, сначала переводилась с использованием кода Бодо в пятибитовый символ. К исходному тексту Бодо добавлялся ключ (по модулю 2). Старинный телетайп фирмы AT&T со считывающим устройством Вернама и оборудованием для шифрования, использовался корпусом связи армии США.

Очень распространена плохая с точки зрения секретности практика использовать слово или фразу в качестве ключа для того, чтобы k=(k0 ,k1 ,...,kк-1) было легко запомнить. В ИС для обеспечения безопасности информации это недопустимо. Для получения ключей должны использоваться программные или аппаратные средства случайной генерации ключей.

*Пример. Преобразование текста с помощью подстановки Вижинера (r=4)*

Исходный текст (ИТ1):

НЕ\_СЛЕДУЕТ\_ВЫБИРАТЬ\_НЕСЛУЧАЙНЫЙ\_КЛЮЧ

Ключ: КЛЮЧ

Разобьем исходный текст на блоки по 4 символа:

НЕ\_С ЛЕДУ ЕТ\_В ЫБИР АТЬ\_ НЕСЛ УЧАЙ НЫЙ\_ КЛЮЧ

и наложим на них ключ (используя таблицу Вижинера):

H+К=Ч, Е+Л=Р и т.д.

Получаем зашифрованный (ЗТ1) текст:

ЧРЭЗ ХРБЙ ПЭЭЩ ДМЕЖ КЭЩЦ ЧРОБ ЭБЮ\_ ЧЕЖЦ ФЦЫН

Можно выдвинуть и обобщенную систему Вижинера. ЕЕ можно сформулировать не только при помощи подстановки Цезаря.

Пусть x - подмножество симметрической группы SYM(Zm).

Определение. r-многоалфавитный ключ шифрования есть r-набор p = (p0, p1, ..., pr-1) с элементами в x.

Обобщенная система Вижинера преобразует исходный текст (x0, x1 ,..., xn-1) в шифрованный текст (y0 ,y1 ,...,yn-1) при помощи ключа p = (p0, p1, ..., pr-1) по правилу VIGk : (x0 ,x1 ,...,xn-1) ® (y0 ,y1 ,...,yn-1) = (p0(х0), p1(х1), ..., pn-1(xn-1)), где используется условие pi = pi mod r .

Следует признать, что и многоалфавитные подстановки в принципе доступны криптоаналитическому исследованию. Криптостойкость многоалфавитных систем резко убывает с уменьшением длины ключа.

Тем не менее такая система как шифр Вижинера допускает несложную аппаратную или программную реализацию и при достаточно большой длине ключа может быть использован в современных ИС.

Шифры Бофора

Используют формулы:

Yi=C*K*i(xi)=(*K*i-Xi) (mod *m*) или

Yi=C*K*i(xi)=(Xi-Ki) (mod *m*) i=0...n-1

Гомофоническая замена одному символу открытого текста ставит в соответствие несколько символов шифртекста. Этот метод применяется для искажения статистических свойств шифртекста.

Таким образом, при гомофонической замене каждая буква открытого текста заменяется по очереди цифрами соответствующего столбца.

Шифр с автоключом

При рассмотрении этих видов шифров становится очевидным, что чем больше длина ключа (например, в шифре Виженера), тем лучше шифр. Существенного улучшения свойств шифртекста можно достигнуть при использовании шифров с автоключом.

Шифр, в котором сам открытый текст или получающаяся криптограмма используются в качестве "ключа", называется шифром с автоключом. Шифрование в этом случае начинается с ключа, называемого первичным, и продолжается с помощью открытого текста или криптограммы, смещенной на длину первичного ключа.

Шифр машины Энигма

При моделировании шифра машины Энигма на ЭВМ можно достичь хорошей устойчивости при сравнительной простоте программы. Напомним, что эта машина представляла собой ряд вращающихся на одной оси барабанов с электрическими контактами, обеспечивающих множество вариантов простой замены, определяемой текущим положением барабанов. В ранних моделях было 5 барабанов, которые перед началом работы устанавливались по кодовому слову, а в ходе шифрования они поворачивались при кодировании очередного символа как в счетчике электроэнергии. Таким образом, получался ключ заведомо более длинный, чем текст сообщения. Слабость шифра:

1. Пять барабанов могли обеспечить лишь около ста миллионов ключей, что позволяло их за день перебрать на ЭВМ. Если использовать не 25 латинских символов, а 256 кодов ASCII и увеличить число барабанов, то сложность раскалывания шифра существенно возрастет.
2. Набор барабанов был ограничен и менялся ред- ко, что вызвало охоту англичан за их экземплярами в подводных лодках. ЭВМ может для каждой шифровки использовать индивидуальные барабаны, генерируемые по ключу, а это опять-таки резко усложняет вскрытие шифра.
3. Наконец, можно сделать движение барабанов хаотичным по случайной последовательности, тоже вырабатываемой по ключу.

Число ключей такого шифра. Пусть длина периода программного генератора случайных чисел равна 2\*\*24. Восемь барабанов, генерируемые с помощью этого генератора, дадут вместе 2\*\*192 вариантов ключа, а если учесть еще варианты псевдослучайной последовательности, управляющей движением барабанов, то получится внушительная цифра в 2\*\*216 вариантов ключа. Таким образом, довольно просто получить устойчивый шифр даже при использовании программного генератора случайных чисел с периодом малой для криптографии длины.

'----------имитация Энигмы

DEFINT I-N: DEFSTR S

CLS : RANDOM12E 231

DIM s(4, 32) AS STRING \* 1

ns = 4

ss = "ААААААААААААААААААААААААААААА'

PRINT ss

'-----------ШИФРОВАНИЕ

x = RND(-231)

FOR i=0 TO ns

 FOR j=0 TO 32:set(i,j) = CHR$(j):NEXT

 FOR j=0 TO 32:SWAP s(i,j),s(i,32\*RND):

NEXT

NEXT

s=""

FOR i = 1 TO LEN(ss) 'шифрование символа

k=ASC (MID$ (ss ,i ,1)): IF k>32 THEN k=k-128

FOR j = 0 TO ns:k=ASC(set(j, k)):NEXT

IF k < 32 THEN k = k+ 128

PRINT CHR$ (k); : s = s + CHR$ (k)

k = ns \* RND 'поворот колес

FOR j=0 TO 31:SWAP s(k,j),s(k,j+1):NEXT

 FOR j=0 TO 32

 s(k,j)=CHR$((ASC(set(k, j)) + 32) MOD 33)

 NEXT

NEXT

PRINT

'----------РАСШИФРОВЫВАНИЕ

x = RND(-231)

FOR i=0 TO ns

FOR j=0 TO 32:s(i,j)=CHR$(j):NEXT

FOR j=0 TO 32:SWAP s(i,j),s(i,32\*RND):NEXT

 FOR j=0 TO 32

 IF ASC (set (i, j)) < 64 THEN

 m =j:n=ASC(s(i, j))

DO

k=ASC(s(i,n)):s(i,n)=CHR$(m OR 64)

m=n: n=k

LOOP UNTIL m = j

 END IF

NEXT j

 FOR j=0 TO 32

 s(i,j)=CHR$(ASC(s(i,j)) AND 63)

 NEXT

NEXT i

ss = s

FOR i = 1 TO LEN(ss)

k=ASC(MID$(ss,i ,1)): IF k>32 THEN k=k-128

 FOR j=ns TO 0 STEP -1

 k=ASC(s(j,k))

 NEXT

 IF k<32 THEN k=k+128

 PRINT CHR$ (k) ;

 k = ns \* RND 'поворот колес

 FOR j = 0 TO 31: SWAP s(k,j),s(k,j+1):NEXT

 FOR j = 0 TO 32

 s(k,j)=CHR$((ASC(s(k,j))+32) MOD 33)

 NEXT

 NEXT i

 END

для упрощения текста программы ключ задается лишь из 3 бит числом 231 и используются лишь 5 барабанов.

Гаммирование

Суть метода состоит в том, что ключ к декодированию байта вычисляется на основе предыдущего байта. При этом сам ключ может модифицироваться от байта к байту. Алгоритм кодирования следующий :

1) вводим ключ;

2) производим с байтом файла одно из действий из множества {+,-,} (с игнорированием переноса);

3) полученный байт является ключом к следующему байту файла ;

4) пока не дошли до конца файла, повторяем п.3.

Декодирование производится по аналогичному алгоритму.

Из простейших процедур, имитирующих случайные числа, наиболее употребляем так называемый конгруэнтный способ, приписываемый Д.Х. Лемеру:

G(n+1)=KGn+C MOD M

В нем каждое последующее псевдослучайное число G(n+1) получается из предыдущего Gn умножением его на К, сложением с С и взятием остатка от деления на М. Весь фокус здесь в том, чтобы подобрать хорошие значения К, С и М. Например, выбрав закон генерации последовательности G(N+1) = Ent (sqr(2)\*Gn) на IBM PC при формате представления чисел с плавающей запятой IEEE в 4 байта, получим псевдослучайные ряды, обязательно заканчивающиеся циклами с периодами длиной всего лишь 1225 и 147 в зависимости от начального заполнения. Разумнее вычисления вести в целых числах. Для них было установлено, что при С=0 и М=2\*\*N наибольший период М/4 достигается при K=3+8\*i или K=5+8\*i и нечетном начальном числе. При достаточно больших К ряд производит впечатление случайного.

**Конгруэнтные датчики ПСЧ для гаммирования**

Одним из хороших конгруэнтных генераторов является линейный конгруэнтный датчик ПСЧ. Он вырабатывает последовательности псевдослучайных чисел T(i), описываемые соотношением

T(i+1) = (A\*T(i)+C)mod *m*,

где А и С - константы, Т(0) - исходная величина, выбранная в качестве порождающего числа. Очевидно, что эти три величины и образуют ключ.

Такой датчик ПСЧ генерирует псевдослучайные числа с определенным периодом повторения, зависящим от выбранных значений А и С. Значение *m* обычно устанавливается равным 2n , где *n* - длина машинного слова в битах. Датчик имеет максимальный период М до того, как генерируемая последовательность начнет повторяться. По причине, отмеченной ранее, необходимо выбирать числа А и С такие, чтобы период М был максимальным. Как показано Д. Кнутом, линейный конгруэнтный датчик ПСЧ имеет максимальную длину М тогда и только тогда, когда С - нечетное, и Аmod 4 = 1.

Целочисленные последовательности

Интересный класс генераторов случайных чисел неоднократно предлагался многими специалистами целочисленной арифметике, в частности Джорджем Марсалиа и Арифом Зейманом. Генераторы этого типа основаны на использовании последовательностей Фибоначчи. Классический пример такой последовательности {0, 1, 1, 2, 3, 5, 8, 13, 21, 34...}. За исключением первых двух ее членов, каждый последующий член равен сумме двух предшествующих. Если брать только последнюю цифру каждого числа в последовательности, то получится последовательность чисел {0, 1, 1, 2, 5, 8, 3, 1, 4, 5, 9, 4...} Если эта последовательность применяется для начального заполнения массива большой длины, то, используя этот массив, можно создать генератор случайных чисел Фибоначчи с запаздыванием, где складываются не соседние, а удаленные числа. Марсалиа и Зейман предложили ввести в схему Фибоначчи "бит переноса", который может иметь начальное значение 0 или 1. Построенный на этой основе генератор "сложения с переносом" приобретает интересные свойства, на их основании можно создавать последовательности, период которых значительно больше, чем у применяемых в настоящее время конгруэнтных генераторов.

Датчики М-последовательностей

М-последовательности также популярны, благодаря относительной легкости их реализации.

М-последовательности представляют собой линейные рекуррентные последовательности максимального периода, формируемые *k*-разрядными генераторами на основе регистров сдвига. На каждом такте поступивший бит сдвигает k предыдущих и к нему добавляется их сумма по модулю 2. Вытесняемый бит добавляется к гамме.

Строго это можно представить в виде следующих отношений:

r1:=r0 r2:=r1 ... rk-1:=rk-2

r0:=a0 r1 Е a1 r2 Е ... Е ak-2 rk-1

Гi:=rk-

Здесь r0 r1 ... rk-1 - k однобитных регистров, a0 a1 ... ak-1 - коэффициенты неприводимого двоичного полинома степени k-1. Гi - i-е значение выходной гаммы.

*Период* М-последовательности исходя из ее свойств равен 2*k*-1.

Другим важным свойством М-последовательности является *объем ансамбля*, т.е. количество различных М-последовательностей для заданного *k*. Эта характеристика приведена в таблице:

|  |  |
| --- | --- |
| *k*  | Объем ансамбля  |
| 5  | 6  |
| 6  | 8  |
| 7  | 18  |
| 8  | 16  |
| 9  | 48  |
| 10  | 60  |
| 16  | 2048  |

### **Метод псевдослучайного гаммирования**

Этот метод похож на метод гаммирования, но к нему добавляется еще модификация ключа. В данном случае ключ модифицируется по случайному закону по формуле

где С mod 4 = 1,

N+1 - максимальное псевдослучайное число этого ряда,

 - ключ для предыдущей итерации;

Ki- ключ для новой итерации.

### **Метод цепочечного гаммирования**

Метод аналогичен методу гаммирования, но ключ изменяется по-другому. Суть метода: имеется последовательность ключей (например, 16). Каждый ключ содержит в себе 12 разрядов ключевого числа для кодирования и 4 разряда для указания следующего ключа в цепочке. После использования ключа берем новый по адресу, указанному в текущем ключе.

**Список литературы**

1. Гатчин Ю.А., Коробейников А.Г. Основы криптографических алгоритмов. Учебное пособие. - СПб.: СПбГИТМО(ТУ), 2002. - 29 с.
2. Главная редакция физико-математической литературы, 1974. 324с.
3. Иванов М.А. Криптографические методы защиты информации в компьютерных системах и сетях. – М.: КУДИЦ-ОБРАЗ, 2001, - 368 с.
4. Кон П. Универсальная алгебра. - М.: Мир. - 1968. 351 с
5. Коробейников А. Г. Математические основы криптографии. Учебное пособие. СПб: СПб ГИТМО (ТУ), 2002. 41 с
6. Левин Максим. Криптография. Руководство пользователя. - М.: Познавательная книга плюс, 2001, - 320 с.
7. Левин М. Криптография. Руководство пользователя. - М.: Познавательная книга плюс, 2001, - 320 с.
8. Молдовян А.А., Молдовян Н.А., Советов Б.Я. Криптография. – СПб.: Издательство "Лань", 2001, - 224 с.
9. Панасенко С.П. Алгоритмы шифрования. Специальный справочник BHV-Санкт-Петербург, 2009. – 576с.
10. Смирнов В.И. Курс высшей математики, том III, часть I – М:. Наука,
11. Чмора А.Л. Современная прикладная криптография. 2-е изд. – М.: Гелиос, АРВ, 2002. – 256 с. ил.