МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

Специальность Информационные системы и технологии

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ НА ТЕМУ:**

**Исследование блочных шифров**

Ф.И.О.

Савельев Дмитрий Витальевич

Преподаватель

Берников Владислав Олегович

Минск 2021

Цель: изучение и приобретение практических навыков разработки и использования приложений для реализации блочных шифров (рассчитана на 4 часа аудиторных занятий).

Задачи:

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости блочных шифров.

2. Разработать приложение для реализации указанных преподавателем методов блочного зашифрования/расшифрования.

3. Выполнить анализ криптостойкости блочных шифров.

4. Оценить скорость зашифрования/расшифрования реализованных шифров.

5. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

# 1. Теоретические сведения

# 1.1. Краткая историческая информация и общая характеристика блочных шифров

В 1972 г. Национальное бюро стандартов США (ныне – Национальный институт стандартов и технологии, National Institute of Standarts & Technology – NIST) инициировало программу защиты каналов связи и компьютерных данных. Одна из целей – разработка единого стандарта криптографического шифрования. Основными критериями оценки алгоритма являлись следующие:

• алгоритм должен обеспечить высокий уровень защиты;

• алгоритм должен быть понятен и детально описан;

• криптостойкость алгоритма должна зависеть только от ключа;

• алгоритм должен допускать адаптацию к различным применениям;

• алгоритм должен быть разрешен для экспорта.

В качестве начального варианта нового алгоритма рассматривался Lucifer – разработка компании IBM начала семидесятых годов. В основе указанного алгоритма использовались два запатентованных в 1971 г. Хорстом Фейстелем (Horst Feistel) устройства, реализующие различные алгоритмы шифрования, позже получившие название шифр (сеть) Фейстеля (Feistel Cipher, Feistel Network). В первой версии проекта Lucifer сеть Фейстеля не использовалась.

После многочисленных согласований, специальных конференций, где рассматривались в основном вопросы криптостойкости алгоритма, подлежащего утверждению в качестве федерального стандарта, в ноябре 1976 г. был утвержден стандарт DES (Data Encryption Standard – стандарт шифрования данных). Предполагалось, что стандарт будет реализовываться только аппаратно.

В 1981 г. ANSI одобрил DES в качестве стандарта для публичного использования (стандарт ANSI Х3.92), назвав его алгоритмом шифрования данных (Data Encryption Algorithm – DEA).

В 1987 г. были разработаны алгоритмы FEAL и RC2. Сети Фейстеля получили широкое распространение в 1990-е гг. – в годы появления таких алгоритмов, как Blowfish (1993), TEA (1994), RC5 (1994), CAST-128 (1996), XTEA (1997), XXTEA (1998), RC6 (1998) и др. На основе сети Фейстеля в 1990 г. в СССР был принят в качестве ГОСТ 28147–89 стандарт шифрования.

Предполагалось, что DES будет сертифицироваться каждые 5 лет. Срок действия последнего сертификата на территории США истек практически к концу ХХ в. К тому времени DES был вскрыт «лобовой атакой».

В 1998 г. NIST объявил конкурс на новый стандарт, который завершился в 2001 г. принятием AES (Advanced Encryption Standard).

Все перечисленные стандарты и алгоритмы блочных шифров (БШ) строятся на основе подстановочных и перестановочных шифров, т. е. являются комбинационными. БШ относятся также к классу симметричных

Блочное зашифрование (расшифрование) предполагает разбиение исходного открытого (зашифрованного) текста на равные блоки, к которым применяется однотипная процедура зашифрования (расшифрования). Указанная однотипность характеризуется прежде всего тем, что процедура зашифрования (расшифрования) состоит из совокупности повторяющихся наборов преобразований, называемых раундами.

Основные требования к шифрам рассматриваемого класса можно сформулировать следующим образом:

• даже незначительное изменение исходного сообщения должно приводить к существенному изменению зашифрованного сообщения; • устойчивость к атакам по выбранному тексту;

• алгоритмы зашифрования/расшифрования должны быть реализуемыми на различных платформах;

• алгоритмы должны базироваться на простых операциях;

• алгоритмы должны быть простыми для написания кода, вероятность появления программных ошибок должна быть низкой;

• алгоритмы должны допускать их модификацию при переходе на иные требования по уровню криптостойкости.

# 1.2. Сеть Фейстеля

Само название конструкции Фейстеля (сеть) означает ее ячеистую топологию [28]. Формально одна ячейка сети соответствует одному раунду зашифрования или расшифрования сообщения.

При зашифровании сообщение разбивается на блоки одинаковой (фиксированной) длины (как правило – 64 или 128 битов).

Полученные блоки называются входными. В случае если длина входного блока меньше, чем выбранный размер, блок удлиняется установленным способом.

Каждый входной блок шифруемого сообщения изначально делится на два подблока одинакового размера: левый (L0) и правый (R0). Далее в каждом i-м раунде выполняются преобразования в соответствии с формальным представлением ячейки сети Фейстеля:



По какому-либо математическому правилу вычисляется раундовый ключ Ki. В приведенном выражение знак «+» соответствует поразрядному суммированию на основе «XOR». На рис. 1 приведено графическое отображение сети Фейстеля.

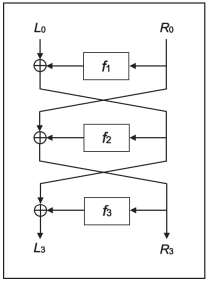


Рисунок 1 – Графическое отображение сети Фейстеля

Расшифрование происходит так же, как и зашифрование, с той лишь разницей, что раундовые ключи будут использоваться в обратном порядке по отношению к зашифрованию.

В своей статье [28] Х. Фейстель описывает два блока преобразований с использованием функции f (Ri – 1, Ki):

• блок подстановок (S-блок, англ. S-box);

• блок перестановок (P-блок, англ. P-box).

Блок подстановок состоит:

• из дешифратора, преобразующего n-разрядное двоичное число в одноразрядный сигнал по основанию 2n;

• внутреннего коммутатора;

• шифратора, преобразующего сигнал из одноразрядного 2n - ричного в n-разрядный двоичный.

Пример реализации трехразрядного S-блока показан на рис. 2.

Блок перестановок изменяет положение цифр, т. е. является линейным устройством. Этот блок может иметь очень большое количество входов-выходов, однако в силу линейности является слабым местом преобразования с точки зрения криптостойкости. На рис. 3 приведен пример реализации 8-разрядного Р-блока.

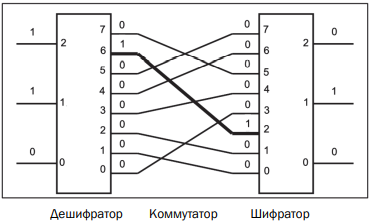


Рисунок 2 – Пример реализации трехразрядного S-блока

Изображение выглядит как мебель, комод

Автоматически созданное описание

Рисунок 3 – Пример реализации 8-разрядного P-блока

Термин «блок» в оригинальной статье используется вместо термина «функция» вследствие того, что речь идет о блочном шифре.

При большом размере блоков шифрования (128 битов и более) реализация такой конструкции Фейстеля на 32-разрядных архитектурах может вызвать затруднения, поэтому применяются модифицированные варианты этой конструкции. Обычно используются сети с четырьмя ветвями. На рис. 5.4 показаны наиболее распространенные модификации.

Такие модификации предусматривают использование не двух (L и R на рис. 1), а четырех ветвей. На рис. 4 показаны некоторые модификации.

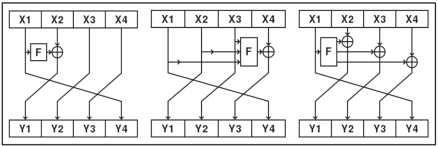


Рисунок 4 – Примеры модификаций базовой сети Фейстеля

В основе криптостойкости блочных шифров лежит идея К. Шеннона о представлении составного шифра таким образом, чтобы он обладал двумя важными свойствами: рассеиванием и перемешиванием. Рассеивание должно скрыть отношения между зашифрованным текстом и исходным текстом.

Рассеивание подразумевает, что каждый символ (или бит) в зашифрованном тексте зависит от одного или всех символов в исходном тексте. Другими словами, если единственный символ в исходном тексте изменен, несколько или все символы в зашифрованном тексте будут также изменены.

Идея относительно перемешивания заключается в том, что оно должно скрыть отношения между зашифрованным текстом и ключом.

# 1.3. Базовые операции сложения чисел в блочных шифрах

Как было указано выше, в основе сети Фейстеля лежит простейшая операция суммирования двух (А + В) n-разрядных чисел – XOR: А + В (mod n). Помимо этой операции некоторые алгоритмы (Blowfish, IDEA, ГОСТ и др.) предусматривают выполнение операций сложения чисел по модулю более высоких порядков: XOR: А + В (mod 2n). Понятно, что числа А и В также являются n-разрядными.

Реализация второй из указанных операций является более сложной. Вспомним основные ее особенности:

1) самое большое слагаемое меньше 2n. Например, при n = 3 самое большое слагаемое в двоичном виде – это 111 (или 7), а 2n = 8;

2) результатом сложения также должно быть n-разрядное число;

3) побитовое сложение предусматривает известную взаимосвязь между соседними символами (порядками);

4) в силу известных правил модулярной арифметики результат вычисления А + В (mod 2n) – это остаток от деления: (А + В) / 2n.

Таким образом, общие правила выполнения рассматриваемой операции формально можно представить следующим образом:

Изображение выглядит как текст, часы, датчик

Автоматически созданное описание

# 1.4. Некоторые блочные алгоритмы

# 1.4.1. Алгоритм DES

В силу причин, перечисленных в п. 1.1, данный алгоритм является, пожалуй, наиболее исследованным.

Алгоритм строится на основе сети Фейстеля.

Входной блок данных, состоящий из 64 битов, преобразуется в выходной блок идентичной длины. В алгоритме широко используются рассеивания (подстановки) и перестановки битов текста, о которых мы упоминали выше. Комбинация двух указанных методов преобразования образует фундаментальный строительный блок DES, называемый раундом или циклом. Один блок данных подвергается преобразованию (и при зашифровании, и при расшифровании) в течение 16 раундов.

После первоначальной перестановки и разделения 64-битного блока данных на правую (R0) и левую (L0) половины длиной по 32 бита выполняются 16 раундов одинаковых действий (см. рис. 5.5). Функционал этих действий подробно рассмотрен в п. 5.1 пособия. В табл. 5.1 показан принцип первоначальной перестановки разрядов (IP) входного 64-битного слова.

Изображение выглядит как стол

Автоматически созданное описание

Выполненная перестановка означает, например, что первый бит входного блока сообщения будет размещен на 40-й позиции (цифра «1» выделена жирным), а 58-й (выделено жирным с подчеркиванием) – на первой и т. д. Из беглого анализа выполненной перестановки легко понять принцип. Алгоритм перестановки разрабатывался для облегчения загрузки блока входного сообщения в специализированную микросхему. Вместе с тем эта операция придает некоторую «хаотичность» исходному сообщению, снижая возможность использования криптоанализа статистическими методами.

Левая и правая ветви каждого промежуточного значения обрабатываются как отдельные 32-битные значения, обозначенные Li и Ri.

Вначале правая часть блока Ri расширяется до 48 битов с использованием таблицы, которая определяет перестановку плюс расширение на 16 битов. Эта операция приводит размер правой половины в соответствие с размером ключа для выполнения операции XOR. Кроме того, за счет выполнения этой операции быстрее возрастает зависимость всех битов результата от битов исходных данных и ключа (это называется «лавинный эффект»).



Рисунок 5 – Общая схема алгоритма DES

После выполнения перестановки с расширением для полученного 48-битного значения выполняется операция XOR с 48-битным подключом Ki. Затем полученное 48-битное значение подается на вход блока подстановки S (от англ. substitution – подстановка), результатом которой является 32-битное значение. Подстановка выполняется в восьми блоках подстановки или восьми S-блоках (S-boxes). При выполнении этой операции 48 битов данных делятся на восемь 6-битных подблоков, каждый из которых по соответствующей таблице замен замещается четырьмя битами. Подстановка с помощью S-блоков является одним из важнейших этапов DES. Таблицы замен для этой операции специально спроектированы так, чтобы обеспечивать максимальную криптостойкость. В результате выполнения этого этапа получаются восемь 4-битных блоков, которые вновь объединяются в единое 32-битное значение.

Далее полученное 32-битное значение обрабатывается с помощью перестановки Р (от англ. permutation – перестановка), которая не зависит от используемого ключа. Целью перестановки является такое максимальное переупорядочивание битов, чтобы в следующем раунде шифрования каждый бит с большой вероятностью обрабатывался другим S-блоком.

И наконец, результат перестановки объединяется с помощью операции XOR с левой половиной первоначального 64-битного блока данных. Затем левая и правая половины меняются местами, и начинается следующий раунд (рис. 6).



Рисунок 6 – Схема реализации функции f на рис. 5

После выполнения 16-раундового зашифрования 64-битного блока данных осуществляется конечная перестановка (IP−1). Она является обратной к перестановке IP. Конечная перестановка определяется табл. 5.2

Изображение выглядит как стол

Автоматически созданное описание

Каждый 8-й бит исходного 64-битного ключа отбрасывается. Эти 8 битов, находящих в позициях 8, 16, 24, 32, 40, 48, 56, 64, изначально добавляются в исходный ключ таким образом, чтобы каждый байт содержал четное число единиц. Это используется для обнаружения ошибок при обмене и хранении ключей по известным алгоритмам избыточного кодирования. Один избыточный бит в ключе DES формируется, как видим, в соответствии с кодом простой четности. Этот код позволяет в кодовом слове (в нашем случае – в каждом байте ключа) обнаруживать ошибки, количество которых нечетно.

При расшифровании на вход алгоритма подается зашифрованный текст. Единственное отличие состоит в обратном порядке использования частичных ключей Ki. Ключ K16 используется в первом раунде, K1 – в последнем.

После последнего раунда процесса расшифрования две половины выхода меняются местами так, чтобы вход заключительной перестановки был составлен из подблоков R16 и L16. Выходом этой стадии является расшифрованный текст.

Слабые и полуслабые ключи. Из-за того, что первоначальный ключ изменяется при получении подключа для каждого раунда алгоритма, определенные первоначальные ключи являются слабыми. Вспомним, что первоначальное значение разделяется на две половины, каждая из которых сдвигается независимо. Если все биты каждой половины равны 0 или 1, то для всех раундов алгоритма используется один и тот же ключ. Это может произойти, если ключ состоит из одних 1, из одних 0, или если одна половина ключа состоит из одних 1, а другая – из одних 0.

Четыре слабых ключа показаны в шестнадцатеричном виде в табл. 5.3 (каждый восьмой бит – это бит четности).

Изображение выглядит как стол

Автоматически созданное описание

Кроме того, некоторые пары ключей при зашифровании переводят открытый текст в идентичный шифртекст. Иными словами, один из ключей пары может расшифровать сообщения, зашифрованные другим ключом пары. Это происходит из-за метода, используемого DES для генерации подключей: вместо 16 различных подключей эти ключи генерируют только два различных подключа. В алгоритме каждый из этих подключей используется восемь раз. Эти ключи, называемые полуслабыми, в шестнадцатеричном виде приведены в табл. 5.4.

Изображение выглядит как текст, кроссворд, шкафчик, табло

Автоматически созданное описание

Криптоанализ DES. Дифференциальный криптоанализ базируется на таблице неоднородных дифференциальных распределений S-блоков в блочном шифре. Криптоанализ шифртекстов на основе рассматриваемого стандарта «работает» с парами шифртекстов, открытые тексты которых имеют определенные разности. Метод анализирует эволюцию этих разностей в процессе прохождения открытых текстов раундов DES при шифровании одним и тем же ключом. Для DES термин «разность» определяется с помощью операции XOR. Затем, используя разности полученных шифртекстов, присваивают различные вероятности различным ключам. В процессе дальнейшего анализа следующих пар шифртекстов один из ключей станет наиболее вероятным. Это и есть правильный ключ.

Линейный криптоанализ: для того, чтобы найти линейное приближение для DES, нужно найти «хорошие» однораундовые линейные приближения и объединить их. Обратим внимание на S-блоки. У них 6 входных битов и 4 выходных. Входные биты можно объединить с помощью операции XOR 63 способами (26 – 1), а выходные биты – 15 способами. Теперь для каждого S-блока можно оценить вероятность того, что для случайно выбранного входа входная комбинация XOR равна некоторой выходной комбинации XOR, и т. д.

DES давно характеризуется низкой криптостойкостью: в январе 1999 г. закодированное посредством DES сообщение было взломано с помощью связанных через Интернет в единую сеть 100 тыс. персональных компьютеров за 24 часа. Данному алгоритму присуща проблема так называемых «слабых» и «частично слабых» ключей. Основное достоинство DES – относительно высокая скорость (из-за малой длины ключа); бесплатное распространение по всему миру; общедоступность и отсутствие необходимости лицензионных отчислений. Модификацией DES является 3DES, он создан У. Диффи, М. Хеллманом, У. Тачманном в 1978 г.

Формальная запись:

*Ci = f(Mj,(DES (K3, (DES(K2, (DES, K1)))))).*

Существуют несколько реализаций алгоритма 3DES. Вот некоторые из них:

• DES-EEE3: шифруется 3 раза с 3 разными ключами (операции шифрование-шифрование-шифрование);

• DES-EDE3: 3DES операции шифрование-расшифрование шифрование с разными ключами;

• DES-EEE2 и DES-EDE2: как и предыдущие, однако на первом и третьем шаге используется одинаковый ключ.

Расшифрование происходит, как и в простом DES, в обратном порядке по отношению к процедуре зашифрования.

3DES с тремя ключами реализован во многих Интернет-приложениях. Например, в PGP (Pretty Good Privacy) он позволяет выполнять операции шифрования и цифровой подписи сообщений, файлов и другой информации, представленной в электронном виде, например, на жестком диске; в S/mime применяется для обеспечения криптографической безопасности электронной почты.

3DES используется при управлении ключами в стандартах ANSI X9.17 (метод генерации 64-битных ключей) и ISO 8732 (управление ключами в банковском деле), а также в PEM (Privacy Enhanced Mail).

# 1.4.2. Стандарт AES

AES (Advanced Encryption Standard) – алгоритм шифрования, действующий в качестве государственного стандарта в США с 2001 г. В основу стандарта положен шифр Rijndael. Шифр Rijndael/AES (т. е. рекомендуемый стандартом) характеризуется размером блока 128 битов, длиной ключа 128, 192 или 256 битов и количеством раундов 10, 12 или 14 в зависимости от длины ключа.

Основу Rijndael составляют так называемые линейно-подстановочные преобразования. В алгоритме широко используются табличные вычисления, причем все необходимые таблицы задаются константно, т. е. не зависят ни от ключа, ни от данных.

# 1.4.3. Стандарт ГОСТ 28147-89

В Советском Союзе в качестве стандарта на блочные алгоритмы шифрования с закрытым ключом в 1989 г. был принят ГОСТ 28147–89 [31].

ГОСТ предусматривает три режима шифрования (простая замена, гаммирование, гаммирование с обратной связью) и один режим выработки имитовставки. Первый из режимов шифрования предназначен для шифрования ключевой информации и не может использоваться для шифрования других данных, для этого предусмотрены два других режима. Режим выработки имитовставки (криптографической контрольной комбинации) предназначен для имитозащиты шифруемых данных, т. е. для их защиты от случайных или преднамеренных несанкционированных изменений.

Шифр ГОСТ 28147–89 построен по тем же принципам, что и американский DES, однако по сравнению с DES первый более удобен для программной реализации. В ГОСТ 28147–89 применяется более длинный ключ – 256 битов, здесь используются 32 раунда шифрования. Таким образом, основные параметры алгоритма криптографического преобразования данных ГОСТ 28147–89: размер блока составляет 64 бита, размер ключа – 256 битов, количество раундов – 32.

Алгоритм представляет собой классическую сеть Фейстеля. Шифруемый блок данных разбивается на две одинаковые части, правую R и левую L. Правая часть складывается по модулю 232 с подключом раунда и посредством принятого алгоритма шифрует левую часть. Перед следующим раундом левая и правая части меняются местами. Такая структура позволяет использовать один и тот же алгоритм как для зашифрования, так и для расшифрования блока (рис. 7).

Таким образом, в алгоритме используются следующие операции:

• сложение слов по модулю 232: правый блок (Ri) складывается по модулю 232 с текущим подключом (Ki);

• циклический сдвиг слова влево на указанное число битов;

• побитовое сложение по модулю 2 (XOR);

• замена (подстановка в блоке S) по таблице.

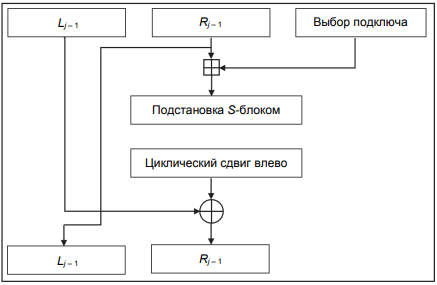


Рисунок 7 – Структура алгоритма реализации раунда в стандарте ГОСТ 28147-89 (знаком "+" в квадратной рамке обозначена операция сложения по модулю 232)

Таким образом, в алгоритме используются следующие операции:

• сложение слов по модулю 232: правый блок (Ri) складывается по модулю 232 с текущим подключом (Ki);

• циклический сдвиг слова влево на указанное число битов;

• побитовое сложение по модулю 2 (XOR);

• замена (подстановка в блоке S) по таблице.

На различных шагах алгоритмов ГОСТа данные, которыми они оперируют, интерпретируются и используются различным образом. В некоторых случаях элементы данных обрабатываются как массивы независимых битов, в других случаях – как целое число без знака, в третьих – как имеющий структуру сложный элемент, состоящий из нескольких более простых элементов.

Сначала правый блок складывается по модулю 232 с подключом. Полученное 32-битное сообщение делится на восемь 4-битных чисел. Каждое из этих 4-битных чисел преобразуется соответствующим S-блоком в другое 4-битное число. Поэтому любой S-блок определяется некоторой 16-битной перестановкой на множестве из 16 элементов: 0, 1, …, 15.

Исходный 256-битный ключ делится на восемь 32-битных подключей. Они используются в 32 тактах в следующем порядке: 1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8, 8, 7, 6, 5, 4, 3, 2, 1. При расшифровании порядок использования подключей меняется на противоположный.

Поскольку в ГОСТ 28147–89 используется 256-битный ключ, то объем ключевого пространства составляет 2256. Даже если предположить, что на взлом шифра брошены все силы вычислительного комплекса с возможностью перебора 1012 (это примерно равно 240) ключей в 1 секунду, то на полный перебор всех 2256 ключей потребуется 2216 секунд. Это время составляет более миллиарда лет.

К уже отмеченным отличиям между алгоритмами DES и ГОСТ можно добавить следующее. В основном раунде DES применяются нерегулярные перестановки исходного сообщения, в ГОСТ используется 11-битный циклический сдвиг влево. Последняя операция гораздо удобнее для программной реализации. Но перестановка DES увеличивает лавинный эффект. В ГОСТ изменение одного входного бита влияет на один 4-битовый блок при замене в одном раунде, который затем влияет на два 4-битовых блока следующего раунда, три блока следующего и т. д. В ГОСТ требуется 8 раундов прежде, чем изменение одного входного бита повлияет на каждый бит результата; в DES для этого нужно только 5 раундов.

# 1.4.4. Алгоритм Blowfish

Основой алгоритма является сеть Фейстеля с 16 раундами. Длина блока равна 64 битам, ключ может иметь любую длину в пределах 448 битов. Хотя перед началом любого зашифрования выполняется сложная фаза инициализации, само зашифрование данных выполняется достаточно быстро. Алгоритм предназначен в основном для приложений, в которых ключ меняется нечасто, к тому же существует фаза начального рукопожатия, во время которой происходит аутентификация сторон. Blowfish характеризуется более высокой скоростью обработки блоков, чем DES. Алгоритм состоит из двух частей: расширение ключа и зашифрование/расшифрование данных. Расширение ключа преобразует ключ длиной, по крайней мере, 448 битов в несколько массивов подключей общей длиной 4168 байтов. Каждый раунд состоит из перестановки, зависящей от ключа, и подстановки, зависящей от ключа и данных. Операциями являются XOR и сложение по модулю 232 (рис. 8).

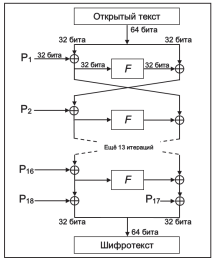


Рисунок 8 – Структура алгоритма Blowfish

Blowfish использует большое количество подключей. Эти ключи должны быть вычислены заранее, до начала любого зашифрования/расшифрования данных.

Элементы алгоритма:

• Р-массив, состоящий из восемнадцати 32-битных подключей: Р1, Р2, ..., Р18;

• S-блоки: каждый из четырех 32-битных S-блоков содержит 256 элементов;

• функция F, структура которой показана на рис. 5.9. Успешные атаки на рассмотренный алгоритм неизвестны.

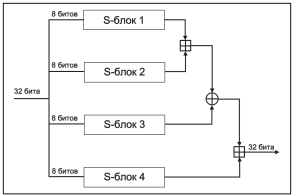


Рисунок 9 – Структура функции F алгоритма Blowfish

Следует также обратить внимание на еще одно обстоятельство: файл, который содержит зашифрованные данные, практически нельзя сжать. Это может служить одним из критериев поиска зашифрованных файлов, которые, как и обычные файлы, представляют собой набор случайных битов.

# 2. Практическое задание

1. Разработать авторское приложение в соответствии с целью лабораторной работы. При этом можно воспользоваться готовыми библиотеками либо программными кодами, реализующими некоторые блочные алгоритмы, из приложения.

Приложение должно реализовывать следующие операции:

• разделение входного потока данных на блоки требуемой длины с необходимым дополнением последнего блока;

• выполнение требуемых преобразований ключевой информации;

• выполнение операций зашифрования/расшифрования;

• оценка скорости выполнения операций зашифрования/расшифрования;

• пошаговый анализ лавинного эффекта с подсчетом количества изменяющихся символов по отношению к исходному слову.

Для выполнения алгоритма необходимо ввести ключ (7 символов, так как длина ключа должна быть 56 бит).

Изображение выглядит как текст, снимок экрана, оранжевый, закрыть

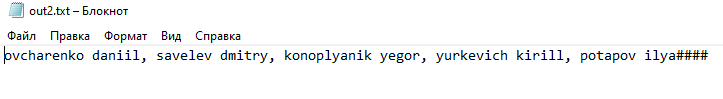
Автоматически созданное описание

После прохождения 16 раундов шифрования, мы получаем следующий результат:

Изображение выглядит как текст

Автоматически созданное описание

А после расшифрования:



Таким образом, я изучил блочные шифры, использующие в своей основе Сеть Фейстеля.