МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования

«БЕЛОРУССКИЙ

ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Направление специальности Информационные системы и технологии .

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №5**

**ПО ДИСЦИПЛИНЕ**

**«ЗАЩИТА ИНФОРМАЦИИ**

**И НАДЕЖНОСТЬ ИНФОРМАЦИОННЫХ СИСТЕМ»**

Выполнил: студент 3 курса 1 группы

Кашперко Василиса Сергеевна

Проверил: ассистент

Нистюк Ольга Александровна

Минск 2023

**Исследование блочных шифров**

**Цель:** изучение и приобретение практических навыков разработки и использования приложений для реализации блочных шифров.

**Теоретическая часть**

В симметричных системах Отправитель и Получатель используют один и тот же ключ К.

К блочным шифрам относятся: DES, FEAL, RC2, Blowfish, TEA, RC5. CAST-128, XTEA, XXTEA, RC6, AES.

**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 шифрование (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.

**Сеть Фейстеля**

Название конструкции Фейстеля (**сеть**) означает ее ячеистую топологию.

Формально **одна ячейка сети соответствует одному раунду** зашифрования или расшифрования сообщения.

При зашифровании сообщение разбивается на блоки одинаковой (фиксированной) длинны (как правило – 64 или128 бит).

Каждый входной блок шифруемого сообщения изначально делится на два подблока одинакового размера: левый (*L0*) и правый (*R0*).

Два блока преобразований:

– блок подстановок (S - блок)

Дешифратор преобразует n-разрядное двоичное число в одноразрядный сигнал по основанию 2^n

Шифратор преобразует сигнал из одноразрядного 2^n-ричного в n-разрядный двоичный

– блок перестановок (P - блок)

**Симметричная криптография. Алгоритм DES**

Базовые методы – подстановка и перестановка данных: 1 под. + 1 перест. – раунд (цикл)

Алгоритм состоит из 16 раундов, т.е. один блок данных длиной 64 бита обрабатывается 16 раз, в каждом из которых используется новый ключ: в каждый раз биты ключа сдвигаются, затем из 56 бит выбирается 48 бит.

Блок перестановок изменяет положение цифр т.е. является линейным устройством.

**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.

**Практическая часть**

В ходе лабораторной работы было разработано приложение, реализующее шифрование и расшифрование сообщения, используя алгоритм шифрования DES.

Сущность алгоритма заключается в следующем:

Каждый входной блок шифруемого сообщения изначально делится на два подблока одинакового размера: левый (*L0*) и правый (*R0*).

В каждом *i-*ом раунде выполняются преобразования:

*,* где *i –* номер раунда*; i = 1…N*

*N* – количество раундов:

*F* – некоторая функция*;*

– ключ *i-1*-го раунда (раундовый ключ)

Начальная и конечная перестановки не имеют никакого значения для криптографии DES. Обе перестановки – без ключей и переопределены.

Начальные и конечные перестановки – прямые *P-*блоки, которые инверсны друг другу.

Начальные и конечные перестановки – прямые *P-*блоки, которые инверсны друг другу.

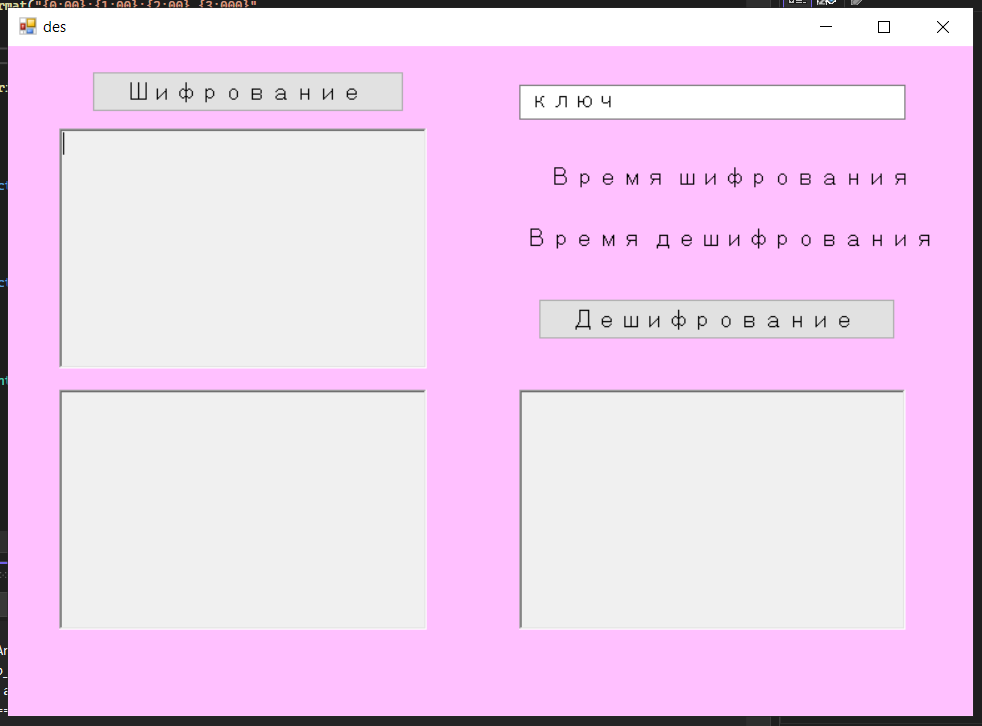


Рисунок 1 – Интерфейс приложения

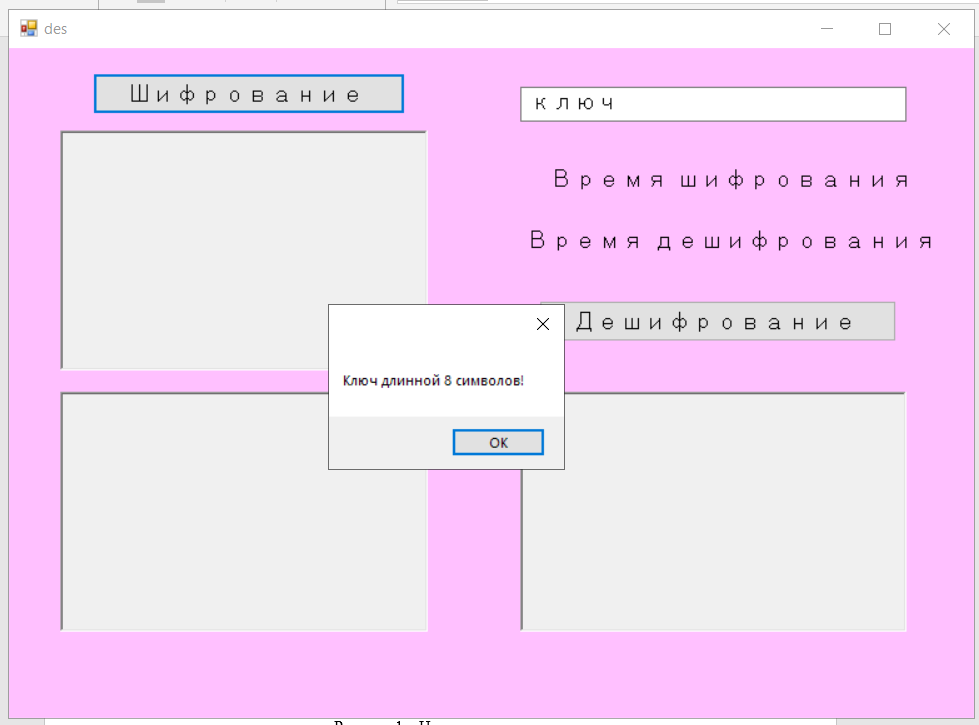


Рисунок 2 – Ключ должен быть длиной 8 символов

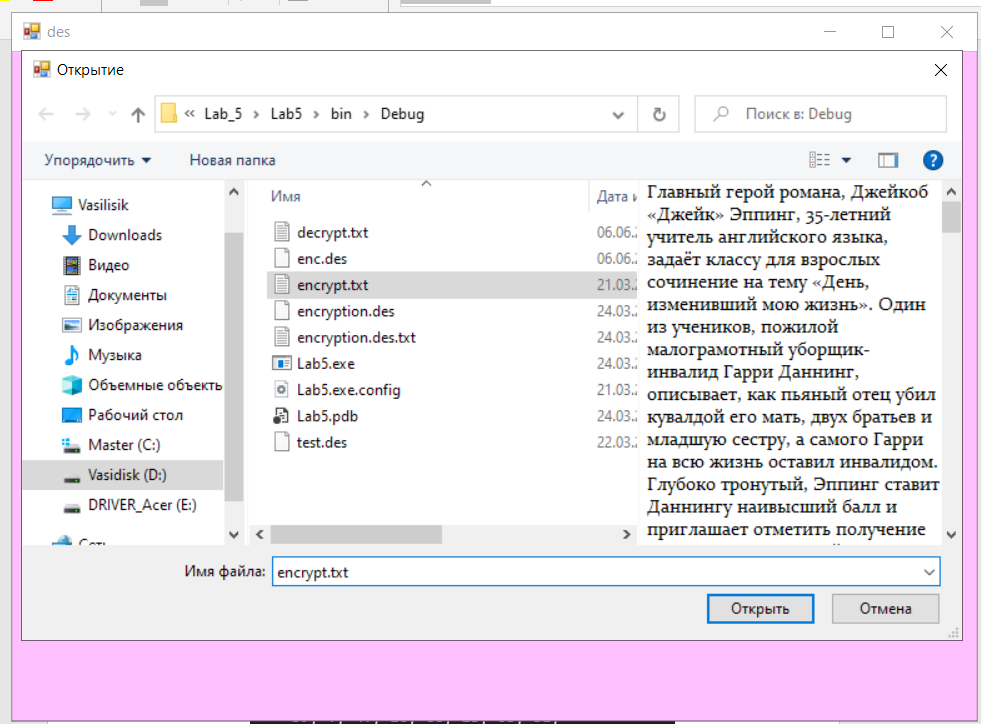


Рисунок 3 – Выбираем текстовый файл для зашифрования

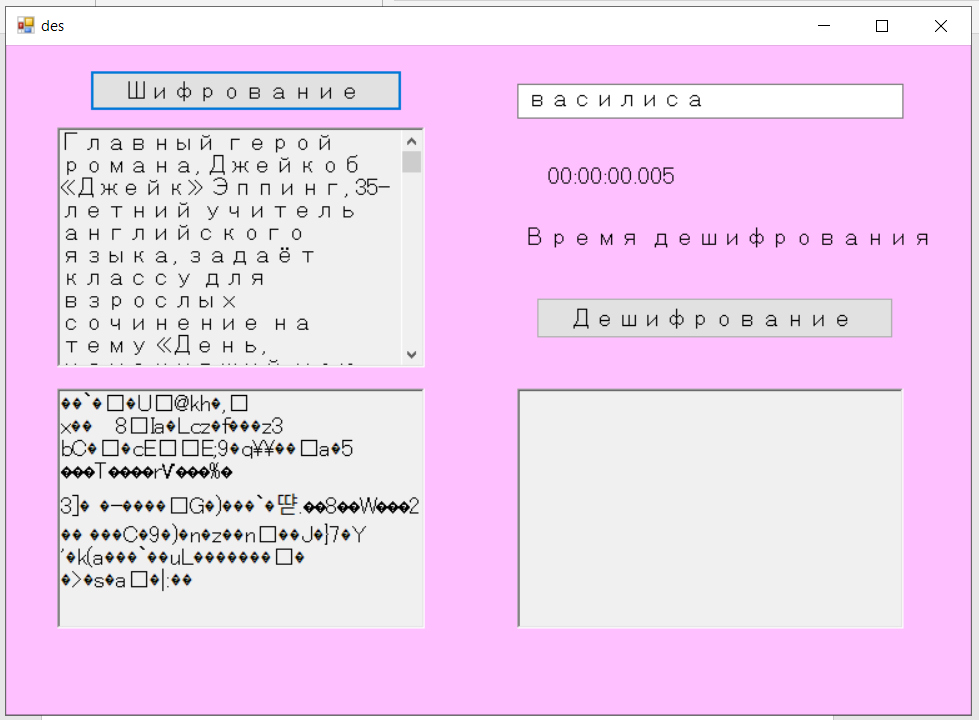


Рисунок 4 – Результат зашифрования

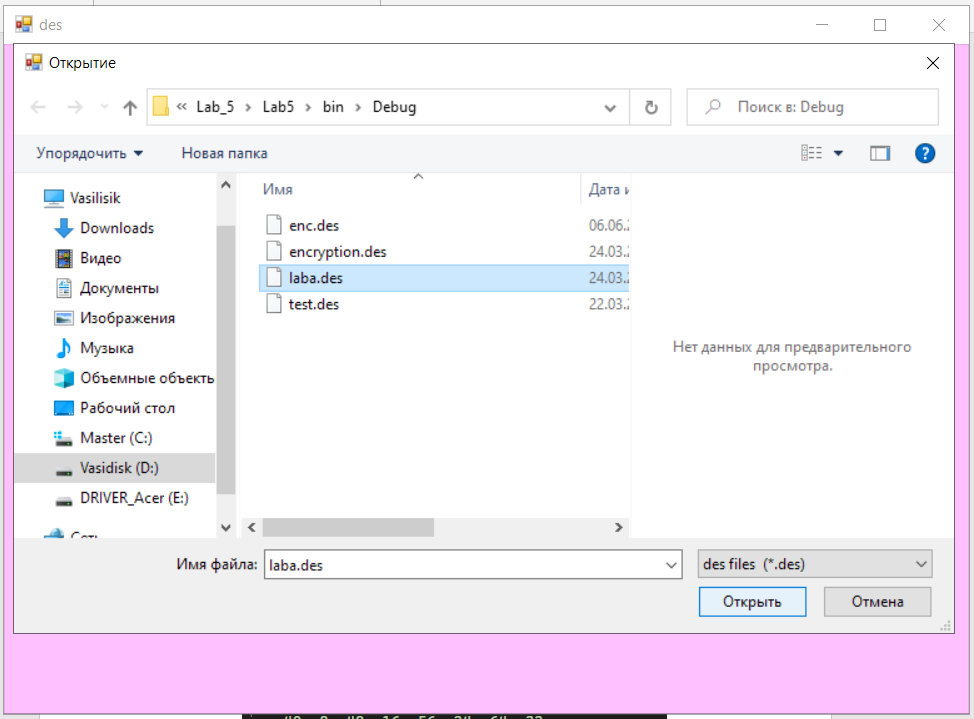


Рисунок 5 – Открываем .des файл

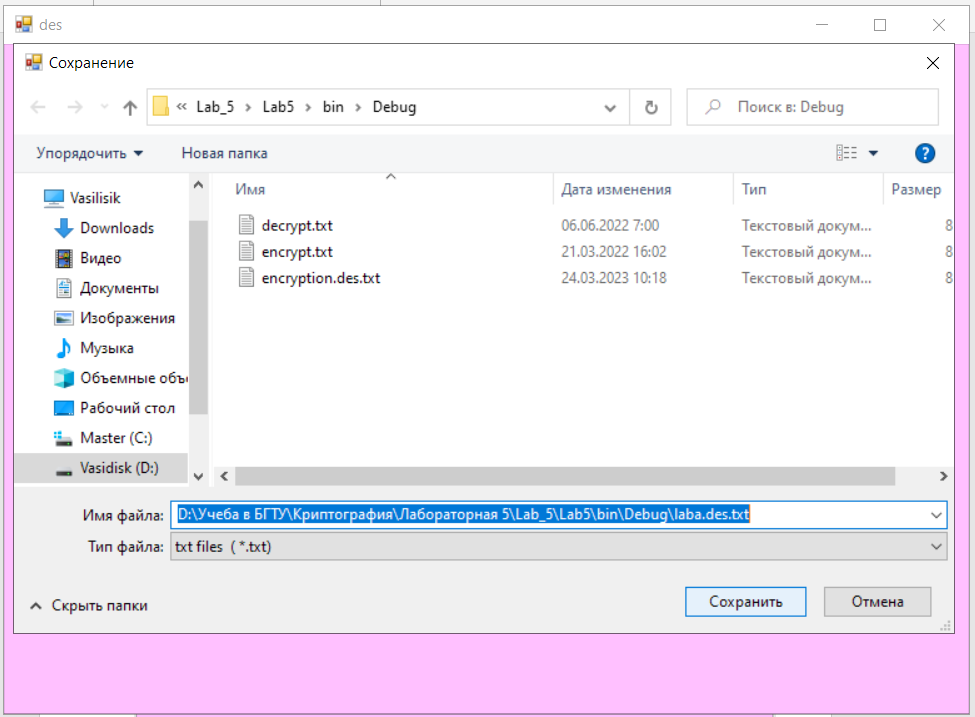


Рисунок 6 – Выбираем место сохранения нового файла с расшифрованным сообщением

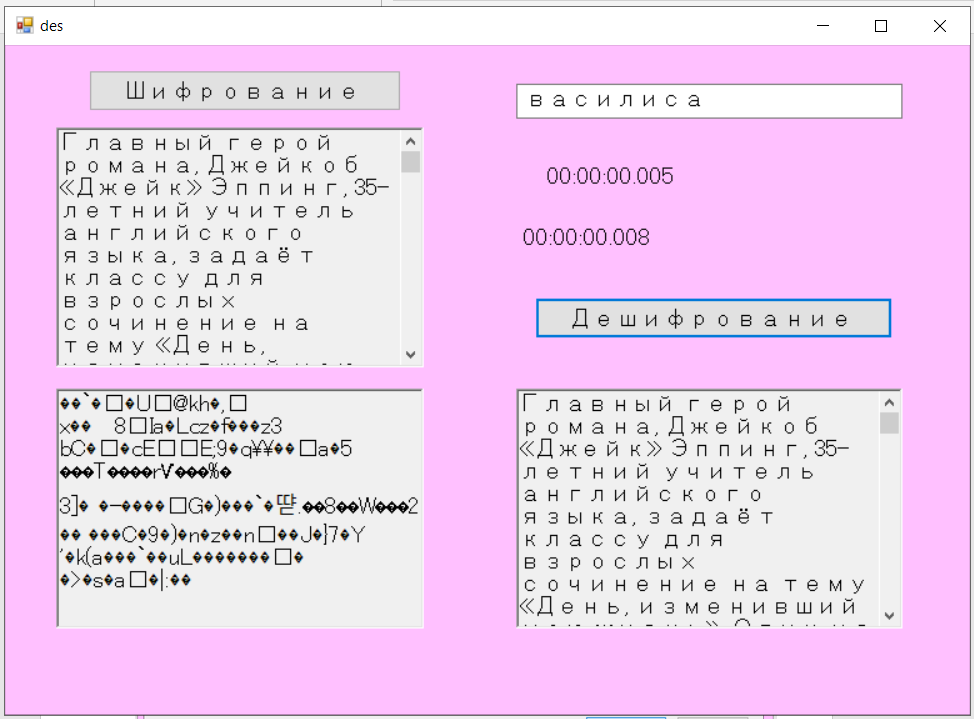


Рисунок 7 – Итоговый результат работы приложения

По условию лабораторной работы, приложение должно реализовывать следующие операции:

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

• выполнение требуемых преобразований ключевой информации;

В качестве ключа благодаря приложению мы можем использовать любой ключ. Далее на рисунке 8 мы посмотрим, как преобразуется ключ в коде и как вообще выглядит функция зашифрования сообщения (Рис. 8).

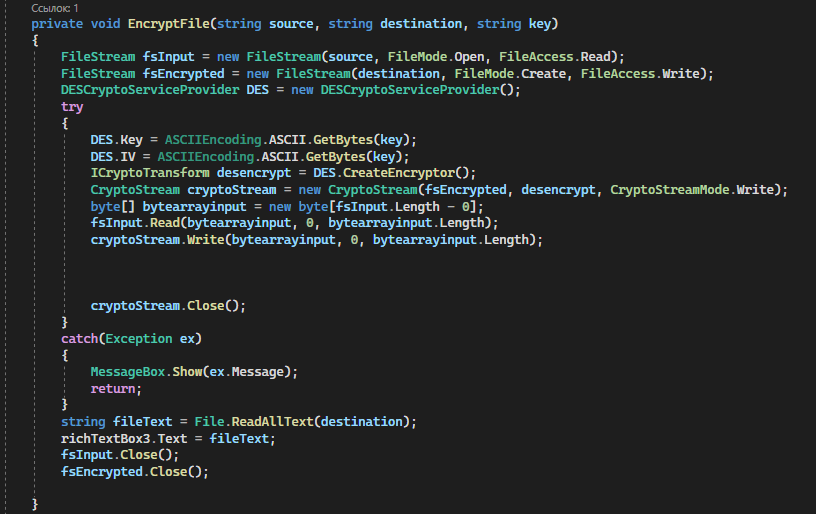


Рисунок 8 – Функция преобразования ключа

• выполнение операций зашифрования/расшифрования;

• оценка скорости выполнения операций зашифрования/расшифрования;

Мы увидели на рисунке 7, что зашифрование выполняется быстрее, чем расшифрование.

После выполнения анализа, можно сделать вывод, что на выполнение шифрования уходит больше времени, нежели на выполнения операции дешифрования.

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

Эффект лавины означает, что изменение значения малого количества битов во входном тексте или ключе ведет к «лавинному» изменению значений выходных битов шифротекста. Другими словами, это зависимость всех (или хотя бы половины) выходных битов от каждого входного бита.

В результате видим, что несмотря на то, что символы отличаются лишь на единицу, а все последующие символы совпадают, сообщение после шифрования в первом и втором случаях абсолютно разные, что является доказательством того, что этот эффект является желательным свойством криптографических алгоритмов, так как он делает невозможным прогнозирование выхода алгоритма для незначительно измененных входных данных.

Для анализа влияния слабых ключей и полуслабых ключей на конечный результат зашифрования можно провести следующие шаги:

1. Выбрать блочный шифр, который будет использоваться для анализа.
2. Сгенерировать все возможные ключи для выбранного блочного шифра.
3. Разделить сгенерированные ключи на категории: сильные ключи, слабые ключи и полуслабые ключи.
4. Зашифровать тестовый набор данных при помощи каждого ключа и записать результаты.
5. Сравнить результаты зашифрования, полученные при помощи сильных ключей, слабых ключей и полуслабых ключей, для определения различий в конечном результате зашифрования.
6. Проанализировать лавинный эффект для каждого ключа, чтобы определить, как изменения в ключе влияют на изменения в конечном результате зашифрования.

Результаты анализа позволят определить, как слабые ключи и полуслабые ключи влияют на конечный результат зашифрования и на лавинный эффект. Это позволит определить, насколько важно использовать сильные ключи при шифровании данных и какие меры можно принять для улучшения безопасности данных при использовании блочных шифров.

**Вывод:** блочные шифры широко применяются для защиты конфиденциальных данных, таких как пароли, кредитные карты, медицинские записи и другие. Разработка и использование приложений для реализации блочных шифров требует глубоких знаний в области криптографии, математических основ криптографии и программирования. Для эффективной реализации и использования блочных шифров необходимо учитывать такие факторы, как скорость шифрования, сложность алгоритма, длина ключа и стойкость шифра.

**Контрольные вопросы**

*1. Какие простейшие операции применяются в блочных алгоритмах шифрования?*

Блочные алгоритмы шифрования работают с блоками данных фиксированного размера. Они применяют различные операции для обработки блоков данных, чтобы зашифровать или расшифровать их.

Простейшие операции, которые применяются в блочных алгоритмах шифрования, включают в себя:

Побитовые операции: XOR, AND, OR, NOT - используются для комбинирования данных с ключом шифрования.

Перестановки - перемещение битов в блоке данных.

Замены - замена блоков данных определенными значениями с использованием таблиц замены (S-блоки).

Циклические сдвиги - смещение битов в блоке данных вправо или влево на фиксированное количество разрядов.

Обратимые линейные преобразования - преобразование блоков данных в линейное пространство и их изменение с помощью матриц.

Эти операции могут комбинироваться в различных комбинациях для создания сложных блочных алгоритмов шифрования. Каждый алгоритм имеет свои особенности и преимущества, и выбор подходящего алгоритма зависит от конкретных требований безопасности и производительности.

*2. В чем отличие блочных алгоритмов шифрования от потоковых?*

Блочные и потоковые алгоритмы шифрования – это два различных типа алгоритмов, которые используются для защиты конфиденциальности данных в криптографии.

Основное отличие между блочными и потоковыми алгоритмами шифрования заключается в том, как они обрабатывают данные.

Блочные алгоритмы шифрования обрабатывают данные блоками фиксированного размера (например, 64 или 128 бит), при этом каждый блок обрабатывается отдельно. Блочные алгоритмы работают с целыми блоками данных, которые могут быть большими или маленькими, в зависимости от конкретного алгоритма. Они могут использоваться для защиты как сообщений целиком, так и их частей.

Потоковые алгоритмы шифрования обрабатывают данные непрерывно, байт за байтом. Они не имеют фиксированного размера блока, а вместо этого шифруют поток битов. Потоковые алгоритмы работают с данными, поступающими на вход по мере их появления.

Еще одно отличие между блочными и потоковыми алгоритмами заключается в том, как они обрабатывают ошибки передачи данных. Потоковые алгоритмы шифрования способны корректировать ошибки в режиме реального времени, в то время как блочные алгоритмы могут обрабатывать только целые блоки данных и могут потребовать дополнительных операций для обработки ошибок.

Выбор между блочными и потоковыми алгоритмами зависит от конкретного применения и требований к производительности и безопасности.

*3. Что понимается под «раундом» алгоритма шифрования?*

Раундом в криптографии понимается один шаг алгоритма шифрования, в котором происходит обработка данных для получения промежуточного результата. Большинство блочных алгоритмов шифрования, таких как AES, DES, Blowfish и другие, состоят из нескольких раундов, каждый из которых выполняет некоторую последовательность операций над блоком данных, чтобы изменить его таким образом, чтобы стало трудно восстановить исходную информацию.

Обычно в каждом раунде алгоритма шифрования используется определенный набор операций, таких как замена байтов, перестановка байтов, смешивание блоков данных и др. Каждый раунд добавляет сложность к процессу шифрования, увеличивая стойкость алгоритма к атакам.

Количество раундов в алгоритме шифрования может быть разным в зависимости от конкретного алгоритма и его режима работы. Обычно количество раундов зависит от требуемого уровня стойкости алгоритма к атакам, а также от скорости работы алгоритма. Большинство современных блочных алгоритмов шифрования имеют от 10 до 16 раундов.

*4. Охарактеризовать и привести формальное описание сети Фейстеля.*

Сеть Фейстеля – это тип блочных алгоритмов шифрования, который был предложен Хорстом Фейстелем в 1973 году. Она используется в таких известных алгоритмах, как DES, Blowfish, и Twofish.

Основная идея сети Фейстеля заключается в том, что исходный блок данных разбивается на две половины, и затем происходит ряд раундов, каждый из которых преобразует правую половину блока на основе левой половины и некоторого ключа. После всех раундов происходит обмен местами левой и правой половин блока и повторение процедуры до тех пор, пока не будет достигнуто нужное количество раундов или пока не будет достигнут конечный результат.

Формально, сеть Фейстеля может быть описана следующим образом:

Разделение блока данных на две половины: *L0* и *R0*.

Повторение *N* раундов:

a. Вычисление функции *f(Ri, Ki)*, где *Ri* - правая половина блока, *Ki* - ключ для *i*-го раунда, *f* - некоторая нелинейная функция, которая обычно является перестановкой битов.

b. Вычисление *Li+1 = Ri*

c. Вычисление *Ri+1 = Li XOR f(Ri, Ki)*

Обмен местами левой и правой половин блока: *Ln+1 = Rn* и *Rn+1 = Ln*

Ключи для каждого раунда могут быть получены из основного ключа шифрования с помощью алгоритма расширения ключа, который преобразует основной ключ в набор ключей, каждый из которых используется для шифрования соответствующего раунда.

Преимуществом сети Фейстеля является её простота, скорость работы и высокий уровень безопасности, который обеспечивается использованием нелинейной функции *f*.

*5. Какие стандартные операции используются в блочных алгоритмах шифрования?*

Блочные алгоритмы шифрования используют ряд стандартных операций, которые могут варьироваться в зависимости от конкретного алгоритма, но в общем виде можно выделить следующие:

1. XOR - операция исключающего ИЛИ, которая применяется для комбинирования данных с ключом шифрования.
2. Перестановки - операции, которые изменяют порядок битов в блоке данных или ключе шифрования. Эти операции могут быть линейными или нелинейными, в зависимости от конкретного алгоритма.
3. Замены - операции, которые заменяют каждый байт данных или блок данных на другой байт или блок данных, используя таблицу замен (S-блоки).
4. Сдвиги - операции, которые сдвигают биты влево или вправо на определенное количество позиций. Это позволяет перемещать данные в блоке и изменять их порядок.
5. Циклические сдвиги - операции, которые сдвигают биты блока данных или ключа шифрования влево или вправо с переносом на другой конец блока. Это позволяет создавать сложные перестановки данных.

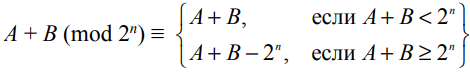
Эти операции могут комбинироваться в различных сочетаниях для создания более сложных алгоритмов шифрования. Кроме того, многие блочные алгоритмы шифрования также используют сеть Фейстеля, которая является комбинацией перестановок, замен и XOR операций, и позволяет обеспечить высокий уровень безопасности при небольшом количестве применяемых операций.

*6. В чем состоит особенность сложения чисел по модулю 2^n?*

Особенность сложения чисел по модулю 2^n заключается в том, что результатом операции сложения является остаток от деления суммы чисел на 2^n. То есть если мы складываем два числа a и b по модулю 2^n, то результат будет равен (a + b) mod 2^n.

Это означает, что при сложении двух чисел по модулю 2^n мы ограничиваем результат до n битов. Если сумма чисел превышает 2^n, то результатом будет остаток от деления этой суммы на 2^n. Таким образом, при сложении по модулю 2^n мы получаем результат в виде n-битного числа, которое можно хранить в регистре фиксированной длины.

Сложение по модулю 2^n широко используется в криптографических алгоритмах, таких как блочные шифры и хэш-функции, где требуется обработка больших объемов данных с ограниченным количеством битов. В таких алгоритмах результаты операций сложения по модулю 2^n могут быть дальше обработаны с помощью других криптографических операций, таких как логические операции и замены, для достижения определенного уровня безопасности.



*7. Сложить по модулю 102 пары чисел: 55 и 14; 76 и 24; 99 и 99.*

55 + 14 = 69. Поскольку 69 < 102, результатом будет остаток от деления 69 на 102: 69 mod 102 = 69.

76 + 24 = 100. Результатом будет 100 mod 102 = 100.

99 + 99 = 198. Результатом будет 198 mod 102 = 96.

Таким образом, результатом сложения по модулю 102 этих трех пар чисел являются числа 69, 100 и 96.

*8. Сложить по модулю 2^8: двоичные числа 10101100 и 11001010; 01111111 и 01101101; шестнадцатеричные числа 0В5 и 37.*

Для сложения двоичных чисел по модулю 2^8 мы должны использовать только последние 8 битов каждого числа и затем сложить эти биты по модулю 2^8.

1) В данном случае, последние 8 битов каждого числа равны:

10101100 -> 0101100

11001010 -> 1001010

Суммируя эти два числа, мы получаем:

0101100

* 1001010 =

1110110

Результат сложения по модулю 2^8 двоичных чисел 10101100 и 11001010 равен 1110110, или в десятичной системе счисления - 118.

2) В данном случае, последние 8 битов каждого числа равны:

01111111 -> 1111111

01101101 -> 1101101

Суммируя эти два числа, мы получаем:

1111111

* 1101101 =

11001000

Результат сложения по модулю $2^8$ двоичных чисел 01111111 и 01101101 равен 11001000, или в десятичной системе счисления - 200.

3) Переведем оба числа в двоичную систему счисления:

0В5 = 0000 1011 37 = 0011 0111

Теперь сложим их по модулю 2^8, то есть оставим только младшие 8 бит результата:

0000 1011

0011 0111=

0100 0010

Таким образом, сумма чисел 0В5 и 37 по модулю 2^8 равна 0100 0010, что в шестнадцатеричной системе счисления соответствует числу 42. Ответ: 42.

*9. Дать пояснение принципам реализации «лавинного» эффекта.*

"Лавинный" эффект в криптографии является явлением, при котором изменение даже одного бита входного сообщения или ключа приводит к значительному изменению выходного значения хеш-функции или шифра. Этот эффект является желательным свойством криптографических алгоритмов, так как он делает невозможным прогнозирование выхода алгоритма для незначительно измененных входных данных.

Основными принципами реализации "лавинного" эффекта в криптографических алгоритмах являются:

1. Равномерное распределение: для достижения "лавинного" эффекта, каждый бит выходного значения должен зависеть от каждого бита входного значения или ключа. Это достигается за счет использования перестановок и замен, которые распределяют информацию равномерно по всем битам выходного значения.
2. Криптографическая стойкость: чтобы предотвратить возможность прогнозирования выхода алгоритма, необходимо использовать криптографически стойкие функции для создания "лавинного" эффекта. Это означает, что любая возможная зависимость между входом и выходом алгоритма должна быть нелинейной и сложной для выявления.
3. Итеративность: для достижения более сильного "лавинного" эффекта, криптографические алгоритмы могут быть построены как итеративные процессы. В этом случае, каждый раунд алгоритма принимает на вход результаты предыдущего раунда и выполняет операции, которые усиливают "лавинный" эффект.
4. Случайность: Использование случайных значений в качестве ключей и входных данных может усилить "лавинный" эффект. Это связано с тем, что случайные данные могут создавать случайные зависимости между входом и выходом алгоритма, делая его еще более сложным для анализа.

*10. Выбрать два произвольных блочных алгоритма. В чем состоят отличия между ними?*

Два произвольных блочных алгоритма, которые можно рассмотреть, это AES (Advanced Encryption Standard) и DES (Data Encryption Standard).

1. AES:

AES является одним из самых популярных блочных алгоритмов, используемых в настоящее время. Он использует блок размером 128 бит и ключи длиной 128, 192 или 256 бит. AES является симметричным алгоритмом, то есть для шифрования и расшифровки данных используется один и тот же ключ.

Одной из главных особенностей AES является его высокая степень безопасности, обусловленная сильной нелинейностью и необратимостью функций замены и перестановки. Это делает атаки на алгоритм AES, основанные на анализе статистики блоков, неэффективными.

1. DES:

DES является одним из старейших блочных шифров, используемых до сих пор, несмотря на то, что его ключ длиной 56 бит уже давно считается небезопасным. DES использует блок размером 64 бит и симметричный ключ.

Одной из особенностей DES является использование множества преобразований блока, включая перестановки, замены и циклические сдвиги. Это делает DES относительно медленным алгоритмом, но при этом он остается достаточно безопасным при использовании достаточно длинных ключей.

Основное отличие между AES и DES заключается в их безопасности и скорости работы. AES является более безопасным и быстрым алгоритмом, по сравнению с DES. Он также использует ключи более длинной длины и имеет более эффективную структуру. DES является устаревшим алгоритмом, который стал подвержен атакам со стороны хакеров, и его использование уже не рекомендуется для защиты конфиденциальных данных.

*11. Представить графически и пояснить функционал одного раунда блочного алгоритма DES (АES, ГОСТ 28147–89, Blowfish).*

Один раунд DES включает в себя следующие шаги:

1. Начальная перестановка (IP): биты блока входных данных переставляются в соответствии с определенной таблицей перестановки.
2. Разделение блока на левую и правую части: входной блок размером 64 бита разбивается на две части по 32 бита каждая.
3. Функция расширения (E): правая часть блока размером 32 бита расширяется до 48 бит с помощью определенной таблицы расширения.
4. Ключевая функция (K): 48-битный ключ, полученный из основного ключа DES в соответствии с определенным алгоритмом, складывается по модулю 2 с расширенной правой частью блока.
5. Функция подстановки (S): результат функции K разбивается на 8 блоков по 6 бит, каждый из которых заменяется на 4-битное значение, выбираемое из соответствующей таблицы замены.
6. Функция перестановки (P): результат функции S подвергается перестановке с помощью определенной таблицы перестановки.
7. XOR с левой частью: результат функции P складывается по модулю 2 с левой частью блока.
8. Обмен правой и левой частями блока: левая часть блока становится новой правой частью, а правая - новой левой.

Таким образом, каждый раунд DES осуществляет перестановку и преобразование данных в блоке с использованием ключей. Общее количество раундов DES зависит от режима работы алгоритма и может составлять 16 или 32 раунда.

Рисунок 11 – Один раунд DES

*12. Сколько можно реализовать (теоретически) разновидностей алгоритма 3DES?*

Теоретически, можно реализовать несколько разновидностей алгоритма 3DES, в зависимости от использования различных ключей и режимов работы.

3DES (Triple DES) – это блочный шифр, который использует блоки размером 64 бита и работает с тремя ключами, каждый из которых имеет длину 56 бит. Существуют два режима работы 3DES: EDE (Encrypt-Decrypt-Encrypt) и EEE (Encrypt-Encrypt-Encrypt).

Количество возможных вариантов 3DES можно оценить, учитывая следующее:

* Существует три ключа, каждый из которых имеет длину 56 бит. Это означает, что можно создать 2^56 различных ключей для каждой из трех позиций.
* Существует два режима работы: EDE и EEE.
* Кроме того, можно изменить порядок ключей в режиме EDE, что дает еще один вариант.

Таким образом, общее число возможных вариантов 3DES можно оценить как: 2^56 \* 2^56 \* 2^56 \* 2 \* 2 = 2^168 \* 4.

Такое большое число означает, что количество возможных вариантов 3DES настолько велико, что практически невозможно перебрать их все.

*13. Какие факторы влияют на стойкость блочного алгоритма шифрования?*

Стойкость блочного алгоритма шифрования зависит от нескольких факторов:

1. Длина ключа. Длина ключа должна быть достаточно большой, чтобы сделать перебор всех возможных ключей практически невозможным. Чем больше длина ключа, тем выше стойкость алгоритма.
2. Размер блока. Размер блока должен быть достаточно большим, чтобы затруднить вычисление статистических свойств шифротекста, основанных на анализе частоты появления отдельных символов или битов.
3. Количество раундов. Количество раундов должно быть достаточным, чтобы обеспечить хорошую диффузию и конфузию. Диффузия означает, что изменение одного бита в открытом тексте должно привести к изменению большого числа битов в шифротексте. Конфузия означает, что каждый бит шифротекста должен зависеть от максимального числа битов ключа и открытого текста.
4. Режим шифрования. Режим шифрования должен обеспечивать безопасность от возможных атак, связанных с дублированием блоков, а также от атак, связанных с известным открытым текстом или шифротекстом.
5. Криптографические методы. Важно использовать сильные криптографические методы, такие как подстановочно-перестановочные сети (SP-сети), хэш-функции и др., которые обеспечивают хорошую диффузию и конфузию.
6. Разнообразие ключей. Важно использовать разные ключи для разных блоков, чтобы уменьшить вероятность обнаружения структуры в шифротексте.
7. Реализация алгоритма. Важно использовать правильную реализацию алгоритма, чтобы избежать возможных уязвимостей, связанных с неправильным использованием алгоритма или ошибками в его реализации.

*14. В чем состоит сущность дифференциального криптоанализа?*

Сущность дифференциального криптоанализа заключается в анализе различий (дифференциалов) между парами открытого текста и соответствующих им шифротекстов, полученных с использованием одного и того же ключа.

Дифференциальный криптоанализ заключается в поиске таких пар открытого текста, для которых существует большая вероятность получения определенного различия в соответствующих им шифротекстах. Затем на основе этой информации атакующий может использовать статистические методы для вычисления ключа шифрования.

При проведении дифференциального криптоанализа используется понятие дифференциальной характеристики, которая описывает вероятность получения определенного различия в шифротекстах для заданных пар открытого текста. Хороший блочный шифр должен иметь диффузионные и конфузионные свойства, которые затрудняют проведение дифференциального криптоанализа.

Дифференциальный криптоанализ является одним из наиболее эффективных методов взлома блочных шифров, и поэтому он широко используется для оценки стойкости блочных шифров и разработки новых криптографических алгоритмов.

*15. В чем состоит сущность линейного криптоанализа?*

Сущность линейного криптоанализа заключается в анализе линейных зависимостей между битами ключа, открытым текстом и соответствующим шифротекстом. Линейный криптоанализ ищет линейную функцию (или комбинацию функций), которая наилучшим образом аппроксимирует зависимость между входными и выходными данными алгоритма шифрования.

Для проведения линейного криптоанализа атакующий использует линейные приближения, которые описывают зависимость между битами ключа и битами шифротекста с некоторой вероятностью. Далее, атакующий строит линейное уравнение, которое приближает эту зависимость, и на основе этого уравнения вычисляет биты ключа.

Линейный криптоанализ особенно эффективен для блочных шифров с небольшой длиной ключа, где линейные приближения могут описывать значительную часть зависимостей между ключом и шифротекстом. Однако, для более современных блочных шифров, которые имеют более длинный ключ, линейный криптоанализ обычно не является таким эффективным методом взлома, как дифференциальный криптоанализ.

Линейный криптоанализ также используется для оценки стойкости блочных шифров и разработки новых криптографических алгоритмов.

*16. Какие ключевые комбинации относятся к слабым (к полуслабым) и почему?*

Слабые ключевые комбинации в блочных шифрах являются такими, что они позволяют провести атаки на шифр с помощью известных методов криптоанализа с меньшей вычислительной сложностью, чем при использовании случайно сгенерированных ключей.

К полуслабым ключевым комбинациям относятся такие ключи, при использовании которых шифр может быть взломан в некоторых специальных случаях, но общие методы взлома не существуют. Примерами таких блочных шифров могут служить DES или 3DES, если они используются с определенными ключевыми значениями.

Некоторые примеры слабых ключей:

* В DES существует 8 ключей, которые называются слабыми, потому что они порождают относительно простые перестановки в начальном и конечном раундах, что делает их уязвимыми для известных атак.
* В 3DES существуют ключи, которые являются слабыми, если все три подключа используются одинаково. В таком случае шифрование сводится к обычному DES со слабым ключом.

В целом, при использовании криптографических алгоритмов важно избегать использования слабых ключей, поскольку это может привести к уязвимостям в системе безопасности. Лучше всего использовать случайно сгенерированные ключи или ключи, полученные с помощью безопасных криптографических протоколов для обмена ключами, таких как Diffie-Hellman или RSA.

*17. Где применяются блочные криптоалгоритмы?*

Блочные криптоалгоритмы широко применяются для защиты информации во многих областях, включая:

1. Компьютерная безопасность: блочные криптоалгоритмы используются для шифрования конфиденциальной информации в компьютерных сетях, интернет-передаче данных, защиты файлов, дисков и т.д.
2. Финансы: блочные криптоалгоритмы используются для защиты банковских транзакций и финансовых операций, включая электронные платежи, кредитные карты, банковские переводы и т.д.
3. Телекоммуникации: блочные криптоалгоритмы используются для защиты конфиденциальности телефонных звонков и сообщений, а также в сотовых сетях.
4. Государственная безопасность: блочные криптоалгоритмы широко используются для защиты государственной тайны и другой конфиденциальной информации.
5. Медицина: блочные криптоалгоритмы могут использоваться для защиты конфиденциальных медицинских данных пациентов, таких как медицинские записи, результаты анализов, истории болезни и т.д.
6. Промышленность: блочные криптоалгоритмы используются для защиты конфиденциальной промышленной информации, такой как патенты, проекты, технологии, изобретения и т.д.
7. Армия и военная безопасность: блочные криптоалгоритмы широко используются в военных целях для защиты конфиденциальных данных, таких как планы операций, геополитические данные, военные сообщения и т.д.
8. Личная безопасность: блочные криптоалгоритмы могут использоваться для защиты личных данных, таких как пароли, пин-коды, личная информация и т.д.

Таким образом, блочные криптоалгоритмы играют важную роль в обеспечении безопасности в различных сферах и областях.