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

В основе сети Фейстеля лежит простейшая операция суммирования 2-х (А + В) n-разрядных чисел – XOR: А + В (mod n).

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

Потоковые шифры - это шифры, при которых каждый бит информации шифруется с помощью гаммирования.

Гаммирование - это наложение на открытые данные гаммы шифра по определенному правилу. Для расшифрования та же гамма накладывается на зашифрованный текст.

В потоковых шифрах шифрование происходит следующим образом. Генератор (за основу берут генератор псевдослучайных чисел) выдает последовательность битов (гамму). Она накладывается на текст, который мы хотим зашифровать с помощью операции XOR. Получаем зашифрованную информацию. Чтобы расшифровать данные следует наложить гамму на зашифрованный текст.

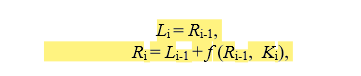
Чаще других используются блочные шифры. Информация, которую хотят зашифровать делится на блоки определенной длины, и шифруется поблочно.

1. **Что понимается под "раундом" алгоритма шифрования?**

Совокупности повторяющихся наборов преобразований

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

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 бит). Полученные блоки называются входными. В случае, если длина входного блока меньше, чем выбранный размер, то 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-ом раунде выполняются преобразования в соответствии с формальным представлением ячейки 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.

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

Некоторые алгоритмы (Blowfish, IDEA, ГОСТ и др.) предусматривают выполнение операций сложения чисел по модулю более высоких порядков: XOR: А + В (mod 2n)

1. **\*В чем состоит особенность сложения чисел по модулю 2n?**

В особенностях сложения чисел по модулю 2n можно выделить 4 пункта:

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

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

102=100

1. А=55 В=14;

A+B = 69;

69(mod 100) = 69.

Для получения нужного результата – вычисления – следует взять младшие 2 разряда (n=2) суммы: 69.

1. А=76 В=24;

A+B = 100;

100(mod 100) = 0.

Для получения нужного результата – вычисления – следует взять младшие 2 разряда (n=2) суммы: 00

1. А=99 В=99;

A+B = 198;

198(mod 100) = 98.

Для получения нужного результата – вычисления – следует взять младшие 2 разряда (n=2) суммы: 98.

1. **Сложить по модулю 28: двоичные числа 10101100 и 11001010; 01111111 и 01101101; шестнадцатеричные числа 0В5 и 37.**

А) A= 10101100, B = 11001010

A+B = 101110110

Для получения нужного результата – вычисления – следует взять младшие 8 разрядов (n=8) суммы: 01110110

Б) A= 01111111, B= 01101101

A+B= 01111111 + 01101101 = 011101100

Для получения нужного результата – вычисления – следует взять младшие 8 разрядов (n=8) суммы: 1110110

В) A=0В5, B=37

Переведем числа из шестнадцатеричной системы счисления в двоичную:

A= 10110101, B= 110111

A+B = 10110101+ 110111= 011101100

Для получения нужного результата – вычисления – следует взять младшие 8 разрядов (n=8) суммы: 11101100

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

В алгоритме DES . Вначале правая часть блока Ri расширяется до 48 битов, используя таблицу, 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 «лавинным эффектом»).

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

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 бит, здесь используются 32 раунда шифрования

Таким образом, основные параметры алгоритма криптографического преобразования данных ГОСТ 28147-89: размер блока составляет 64 бита, размер ключа – 256 бит, количество раундов – 32.

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

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

Модификацией DES является 3DES. Создан У. Диффи, М. Хеллманом, У. Тачманном в 1978 г. Формальная запись: 

Cуществуют 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.

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

В основе криптостойкости блочных шифров лежит идея К. Шеннона в представлении составного шифра таким образом, чтобы от обладал двумя важными свойствами: рассеянием и перемешиванием. Рассеивание должно скрыть отношения между зашифрованным текстом и исходным текстом. Рассеивание подразумевает, что каждый символ (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.

**14. В чем состоит сущность дифференциального криптоанализа?**

Дифференциальный криптоанализ базируется на таблице неоднородных дифференциальных распределений S-блоков в блочном шифре.

**15. В чем состоит сущность линейного криптоанализа?**

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.

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

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.

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*

**17. Где применяются блочные криптоалгоритмы?**

3DES с тремя ключами реализован во многих Интернет-приложениях. Например, в PGP (Pretty Good Privacy) – позволяет выполнять операции шифрования и цифровой подписи сообщений, файлов и другой информации, представленной в электронном виде, например, на жёстком диске ); в S/mime 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).

Стандарт шифрования ГОСТ 28147—89 предназначен для аппаратной и программной реализации, удовлетворяет криптографическим требованиям и не накладывает ограничений на степень секретности защищаемой информации.

**DES**

Исходный текст — блок 64 бит.

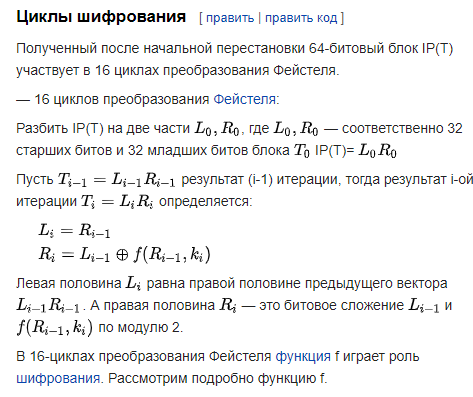
Процесс шифрования состоит из начальной перестановки, 16 циклов шифрования и конечной перестановки.

**Начальная перестановка**

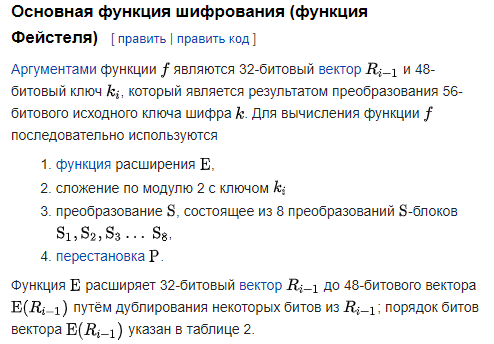
Исходный текст  T{\displaystyle T} (блок 64 бит) преобразуется c помощью начальной перестановки IP(T) {\displaystyle \mathrm {IP} } которая определяется таблицей 1:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Таблица 1. Начальная**[**перестановка**](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BA%D0%B0)**IP** | | | | | | | | | | | | | | | |
| 58 | 50 | 42 | 34 | 26 | 18 | 10 | 2 | 60 | 52 | 44 | 36 | 28 | 20 | 12 | 4 |
| 62 | 54 | 46 | 38 | 30 | 22 | 14 | 6 | 64 | 56 | 48 | 40 | 32 | 24 | 16 | 8 |
| 57 | 49 | 41 | 33 | 25 | 17 | 9 | 1 | 59 | 51 | 43 | 35 | 27 | 19 | 11 | 3 |
| 61 | 53 | 45 | 37 | 29 | 21 | 13 | 5 | 63 | 55 | 47 | 39 | 31 | 23 | 15 | 7 |

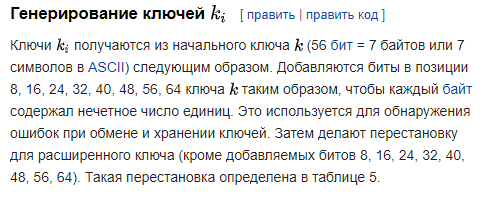
По таблице первые 3 бита результирующего блока {\displaystyle \mathrm {IP} (T)}IP(T)  после начальной перестановки {\displaystyle \mathrm {IP} }IP являются битами 58, 50, 42 входного блока {\displaystyle T}T, а его 3 последние бита являются битами 23, 15, 7 входного блока.



Cети Фейстеля - Информация разбивается на блоки одинаковой (фиксированной) длины. Полученные блоки называются *входными*, так как поступают на вход алгоритма. В случае, если длина входного блока меньше, чем размер, который выбранный алгоритм шифрования способен зашифровать единовременно (размер блока), то блок удлиняется каким-либо способом. Как правило длина блока является [степенью](https://ru.wikipedia.org/wiki/%D0%92%D0%BE%D0%B7%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B2_%D1%81%D1%82%D0%B5%D0%BF%D0%B5%D0%BD%D1%8C) двойки, например, составляет 64 бита или 128 бит.

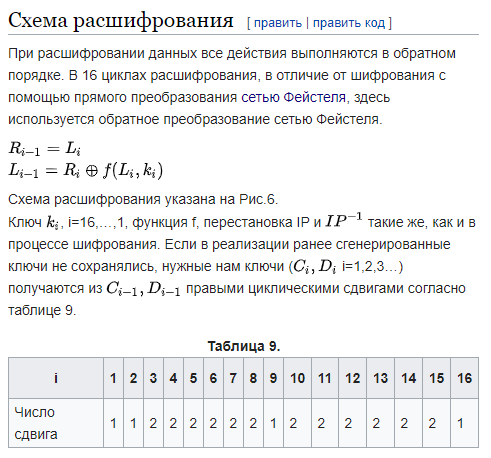


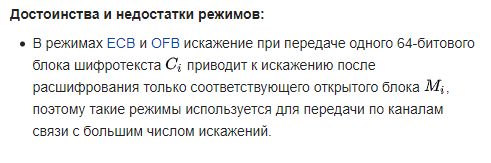












**AES**

Этот алгоритм преобразует один 128-битный блок в другой, используя секретный ключ который нужен для такого преобразования. Для расшифровки полученного 128-битного блока используют второе преобразование с тем же секретным ключом. Размер блока всегда равен 128 бит. Размер ключа также имеет фиксированный размер. Чтобы зашифровать произвольный текст любым паролем можно поступить так:

* получить хеш от пароля
* преобразовать хеш в ключ по правилам описанным в стандарте AES
* разбить текст на блоки по 128 бит
* зашифровать каждый блок функцией cipher

AES – симметричный итеративный блоковый алгоритм; AES – не шифр Фейстеля, базируется на принципах новой сети подстановок-перестановок. Имеет новую архитектуру SQUARE (КВАДРАТ), для которой характерно:

1) представление шифруемого блока в виде двумерного байтового массива;

2) шифрование за один раунд всего блока данных (байт-ориентированная структура);

3) выполнение криптографических преобразований, как над отдельными байтами массива, так и над его строками и столбцами. Это обеспечивает диффузию данных одновременно в двух направлениях - по строкам и по столбцам. Архитектура SQUARE присуща, кроме шифра AES(RIJNDAEL), шифрам SQUARE (его название и дало имя всей архитектуре), CRYPTON (один из кандидатов на AES). Второе место в конкурсе AES занял другой SP-шифр, SERPENT. По-видимому, SP-сети и, в частности, архитектура SQUARE, в ближайшем будущем станут безраздельно доминировать. Общие характеристики AES

• AES зашифровывает и расшифровывает 128-битовые блоки данных.

• AES позволяет использовать три различных ключа длиной 128, 192 или 256 бит (в зависимости от длины ключа версии шифра обозначают AES-128, AES-192 или AES-256).

• От размера ключа зависит число раундов шифрования: длина 128 бит – 10 раундов; длина 192 бита – 12 раундов; длина 256 бит – 14 раундов.

• Все раунды, кроме последнего, идентичны.

**TwoFish**

В криптографии, Twofish является симметричным ключом блочного шифра с размером блока 128 бит и ключевыми размерами до 256 бит. Это был один из пяти финалистов конкурса Advanced Encryption Standard, но он не был выбран для стандартизации. Twofish связано с более ранним блочным шифром Blowfish.

Отличительные особенности Twofish являются использованием предварительно вычисленным ключевыми зависящих от S-блоков, и относительно сложного ключа графика. Одна половина из п-битного ключа используется в качестве фактического ключа шифрования, а другая половина ключа н-битном используется для изменения алгоритма шифрования (ключ-зависимые S-боксы). Twofish заимствует некоторые элементы из других конструкций; например, псевдо-преобразование Адамара (PHT) от SAFER семейства шифров. Twofish имеет структуру Фейстеля, как DES.

Еще в 2000 году на большинстве программных платформ Twofish был немного медленнее, чем Rijndael (выбранного алгоритма Advanced Encryption Standard) для 128-битных ключей, но несколько быстрее 256-битных ключей. Но после того, как Rijndael был выбран в качестве Advanced Encryption Standard, Twofish стал намного медленнее, чем Rijndael на процессорах, которые поддерживают набор инструкций AES.

Twofish был разработан Брюс Шнайер, Джон Келси , Дуг Уайтинг , Дэвид Вагнер , Крис Холл , и Niels Ferguson ; «Расширенный Twofish команда» , которые собрались , чтобы выполнить дальнейший криптоанализ Twofish и других участников конкурса AES включена Стефан Лакс , Тадаёсите Коы и Майк Stay .

Twofish шифр не запатентован и эталонная реализация была помещена в общественном достоянии. В результате алгоритм Twofish свободен для любого использования без каких - либо ограничений. Это один из нескольких шифров, включенных в OpenPGP стандарт ( RFC 4880 ). Тем не менее, Twofish видел менее широкое применение, чем Blowfish, который был доступен дольше.

**Serpent**

Serpent (с латыни — «змея») — симметричный блочный алгоритм шифрования.

Разработан Россом Андерсоном, Эли Бихамом и Ларсом Кнудсеном.

Алгоритм являлся одним из финалистов 2-го этапа конкурса AES. Как и другие алгоритмы, участвовавшие в конкурсе AES, Serpent имеет размер блока 128 бит и возможные длины ключа 128, 192 или 256 бит. Алгоритм представляет собой 32-раундовую SP-сеть, работающую с блоком из четырёх 32-битных слов. Serpent был разработан так, что все операции могут быть выполнены параллельно, используя 32 1-битных «потока».

При разработке Serpent использовался более консервативный подход к безопасности, нежели у других финалистов AES, проектировщики шифра считали, что 16 раундов достаточно, чтобы противостоять известным видам криптоанализа, но увеличили число раундов до 32, чтобы алгоритм мог лучше противостоять ещё неизвестным методам криптоанализа.

Став финалистом конкурса AES, алгоритм Serpent в результате голосования занял 2 место.

Шифр Serpent не запатентован и является общественным достоянием.

Алгоритм Serpent представляет собой SP-сеть, в которой весь блок данных длиной 128 бит на каждом раунде разбивается на 4 слова длиной по 32 бита. Все значения, использующиеся при шифровании, представляются битовыми потоками. Индексы бит пробегают значения от 0 до 31 для 32-битных слов, от 0 до 127 для 128-битных блоков, от 0 до 255 для 256-битных ключей и так далее. Для внутренних вычислений все биты величин представлены в прямом порядке (little-endian).

Serpent шифрует открытый текст P длиной 128 бит в шифротекст C длиной так же 128 бит за 32 раунда с помощью 33 подключей K0,…K32 длиной 128 бит. Длина используемого ключа может принимать различные значения, но для конкретики зафиксируем их длину в 128, 192 или 256 бит. Короткие ключи длиной менее 256 бит дополняются до полной длины в 256 бит.

Шифрование состоит из следующих основных шагов:

* начальная перестановка;
* 32 раунда, каждый из которых состоит из операции смешивания с 128-битным ключом (побитовое логическое исключающее «или»), табличная замена (S-box) и линейное преобразование. В последнем раунде линейное преобразование заменяется дополнительным наложением ключа;
* конечная перестановка;

Начальная и конечная перестановки не имеют какой-либо криптографической значимости. Они используются для упрощения оптимизированной реализации алгоритма и повышения вычислительной эффективности.

Расшифрование отличается от шифрования только тем, что должны быть использованы инверсные (обратные) таблицы замен, а также обратные линейные преобразования, но только для раундовой функции R. Ключи подаются в обратном порядке. Битовые операции раундовой функции расшифрования (в т. ч. замены и перестановки) должны проходить в порядке обратном порядку битовых операций раундовой функции шифрования.

При разработке и анализе алгоритма Serpent не было выявлено каких-либо уязвимостей в полной 32-раундовой версии. Но при выборе победителя конкурса AES это было справедливо и для остальных алгоритмов-финалистов.

По мнению создателей Serpent, алгоритм может быть взломан, только если будет создана новая мощная математическая теория.

Стоит отметить, что XSL-атака, если будет доказана эффективность её проведения, ослабит криптостойкость Serpent.

**DES**

Рассказать сначала:

У нас есть исходный текст, который размером 64 бита, сначала делаем начальную перестановку, она делается с помощью таблицы и значение битов в этой таблицы меняется местами.

Вопросы от Берника.

Изначально сколько бит ключ, как он генерируется и как он в дальнейшем преобразовывается?

У нас есть начальный ключ размером 56 бит или 7 байтов. Добавляются биты к ключу таким образом, чтобы каждый байт содержал нечетное число единиц, это делается для того чтобы обнаружить ошибки. Для того чтобы их обнаружить мы можем использовать код с одной проверкой на четность. Мы проверяем так чтобы символ, который мы выбираем, его сумма по модулю 2 со всеми информационными символами равнялась нулю, если она не равна нулю получается он нечетный.

**Берник: так хорошо у нас 56 бит, как мы преобразуем дальше?**

Далее мы расширяем ключ (добавляем ему биты) и для этого расширенного ключа мы делаем перестановку, у нас есть таблица, в которой мы делим на два блока наш ключ, 28 бит каждый, один это старшие биты, а второй это младшие биты. И потом мы делаем циклические сдвиги также с помощью таблицы.

В таблице сдвигов у нас число сдвигов от 1 до 16 и эти сдвиги состоят из 48 бит, выбранных из битов, которые были 56 бит. (То есть мы какие-то биты удаляем).

**Берник: как у нас генерируется ключ имеется ввиду, как определить слабый или полу это слабый ключ?**

Для каждого раунда алгоритма определенные первоначальные ключи они могут является слабыми или полу слабыми. Так как первоначальное значение оно разбивается на две половины каждый из которых сдвигается независимо, если все биты каждой половины равны 0 или 1, то получается для всех раундов алгоритма используется один и тот же ключ. Это может получится если ключ состоит из одних 0-ей или 1-ц или если одна половина ключа состоит из одних единиц, а другая из одних нулей. Также некоторые пары ключей при зашифровании они переводят текст в такой же текст – это называется слабыми ключами.

**Берник: как работает cеть Фейстеля как данные заходят, какой блок данных?**

В сетях Фейстеля информация разбивается на блоки одинаковой длинны, полученные блоки называются входными, так как поступают на вход алгоритма, в случае если длинна входного блока меньше чем размер, который выбрал алгоритм, на который можно разбить, то блок удлиняется, каким-либо способом. Как правило для блока берется размер этой степени к двойке, то есть 64 бита или 128.

В Алгоритме DES функция Фейстеля, для функции аргументами являются 32битный вектор и 48битовый ключ, который мы сгенерировали. Для вычисления этой функции последовательно у нас используется функция расширения, потом сложения по модулю 2 с ключом, потом преобразование, состоящее из 8-ми преобразований из блоков.

Берник: каким образом у нас что расширяется или сжимается. Откройте схему.

**Открываем схему**

У нас поступает исходный текст на вход 64 бита, происходит сначала начальная перестановка, также генерируются ключи, разделяется блок на left и right по 32 бита. f – это функция Фейстеля, к – это ключ. Для того чтобы сгенерировать ключ фейстеля у нас используется ключ и наши блоки данных. То есть сначала у нас функция расширяется. Берник: как? С помощью таблицы также функции расширения. Берник: как таблица работает, как расширение происходит?

Допустим у нас 32битовый вектор расширяется до 48 битного. Путём дублирования некоторых битов это происходит. Далее у нас происходит сложение по модулю 2 с ключом. Потом преобразование, состоящее из 8 преобразований из блоков. Далее конечная перестановка.

**Как 48 бит с помощью s боксов преобразовать?**

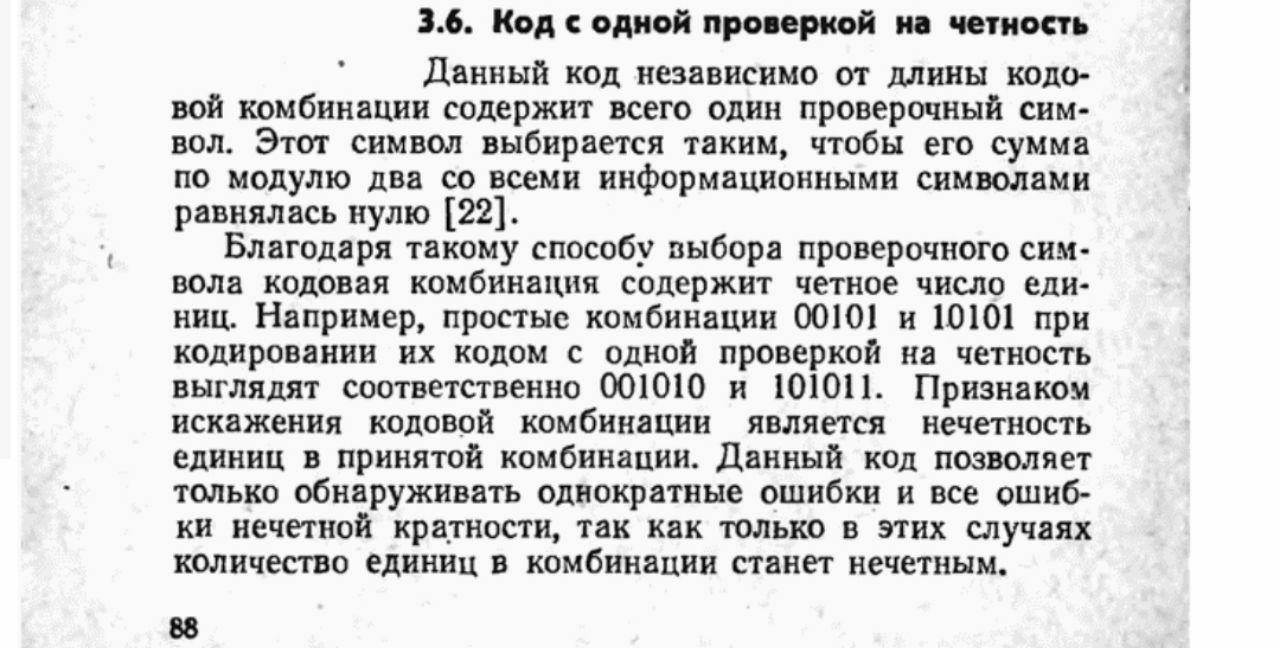
Преобразование происходит с помощью s и преобразование происходит с помощью таблицы, эта таблица от 1 до 8.

**Как мы 48битовую последовательность разбиваем на s боксы?**

Как генерируются ключи? Как понять, что есть ошибка?

DES:

Схема шифрования алгоритма DES

****