Шифрование

Основные механизмы шифрования

- Алгоритмы симметричного шифрования - алгоритмы шифрования, в которых для шифрования и дешифрования используется один и тот же ключ или ключ дешифрования легко может быть получен из ключа шифрования.

- Алгоритмы асимметричного шифрования - алгоритмы шифрования, в которых для шифрования и дешифрования используются два разных ключа, называемые открытым и закрытым ключами, причем, зная один из ключей, вычислить другой невозможно.

- Хэш-функции - функции, входным значением которых является сообщение произвольной длины, а выходным значением - сообщение фиксированной длины. Хэш-функции обладают рядом свойств, которые позволяют с высокой долей вероятности определять изменение входного сообщения.

Алгоритмы симметричного шифрования

В процессе шифрования используется определенный алгоритм шифрования.

На вход подаются исходное незашифрованное сообщение (plaintext) и ключ. Выходом алгоритма является зашифрованное сообщение (ciphertext).

Ключ представляет значение, не зависящее от шифруемого сообщения. Изменение ключа должно приводить к изменению зашифрованного сообщения.

Безопасность, обеспечиваемая криптографией

Безопасность, обеспечиваемая традиционной криптографией, зависит от нескольких факторов.

- Криптографический алгоритм должен быть достаточно сильным, чтобы передаваемое зашифрованное сообщение невозможно было расшифровать без ключа, используя только различные статистические закономерности зашифрованного сообщения или какие-либо другие способы его анализа.

- Безопасность передаваемого сообщения должна зависеть от секретности ключа, но не от секретности алгоритма. В алгоритме должна быть скрыта взаимосвязь между незашифрованным и зашифрованным сообщениями. При этом производители могут создавать дешевые аппаратные чипы и свободно распространяемые программы, реализующие данный алгоритм шифрования.

- Алгоритм должен быть таким, чтобы нельзя было узнать ключ, даже зная достаточно много пар (зашифрованное сообщение, незашифрованное сообщение), полученных при шифровании с использованием данного ключа.

Стойкость алгоритма шифрования

Клод Шеннон ввел понятия диффузии и конфузии для описания стойкости алгоритма шифрования.

- Диффузия - это рассеяние статистических особенностей незашифрованного текста в широком диапазоне статистических особенностей зашифрованного текста.
  (любой элемент зашифрованного текста зависит от многих элементов незашифрованного текста).

- Конфузия - это уничтожение статистической взаимосвязи между зашифрованным текстом и ключом.

Если X - это исходное сообщение и K - криптографический ключ, то зашифрованный передаваемый текст можно записать в виде

$Y = E_K[X]$

Получатель, используя тот же ключ, расшифровывает сообщение

$X = D_K[Y]$

Противник, не имея доступа к K и X , должен попытаться узнать X, K или и то, и другое.

Блочное шифрование

Блок текста рассматривается как неотрицательное целое число, либо как несколько независимых неотрицательных целых чисел. Длина блока всегда равна целой степени двойки.

При этом для шифрования могут использоваться следующие типы операций:

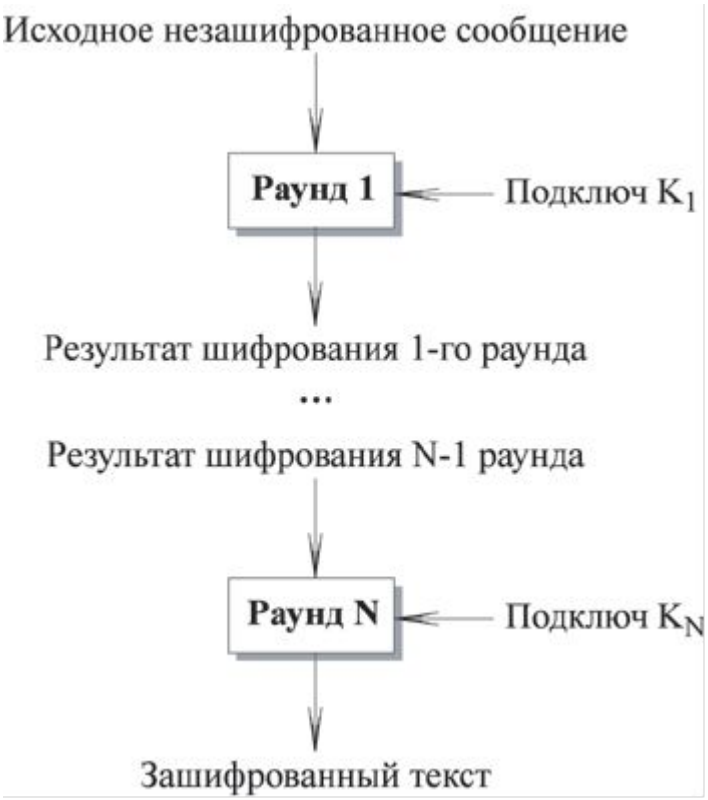
- Табличная подстановка, при которой группа битов отображается в другую группу битов.
- Перемещение, с помощью которого биты сообщения переупорядочиваются.
- Операция сложения по модулю 2.
- Операция сложения по модулю 232 или по модулю 216.
- Циклический сдвиг на некоторое число битов.

Раунды блочного шифрования

Операции над блоками циклически повторяются в алгоритме, образуя «раунды».

Входом каждого раунда является выход предыдущего раунда и ключ, который получен по определенному алгоритму из ключа шифрования K.

Ключ раунда называется подключом.



Требования к алгоритму шифрования

Области применения

- Шифрование данных. Алгоритм должен быть эффективен при шифровании файлов данных или большого потока данных.
- Создание случайных чисел. Алгоритм должен быть эффективен при создании определенного количества случайных битов.
- Хэширование. Алгоритм должен преобразовываться в одностороннюю хэш-функцию.

Платформы

- Алгоритм должен эффективно реализовываться на специализированной аппаратуре, предназначенной для выполнения шифрования/дешифрования.
- Большие процессоры. Алгоритм должен допускать эффективную программную реализацию на 32-битных процессорах.
- Процессоры среднего размера. Алгоритм должен работать на микроконтроллерах и других процессорах среднего размера.
- Малые процессоры. Должна существовать возможность реализации алгоритма на смарт-картах, пусть даже с учетом жестких ограничений на используемую память.

Дополнительные требования

- Алгоритм должен быть простым для написания кода, чтобы минимизировать вероятность программных ошибок.
- Алгоритм должен иметь плоское пространство ключей и допускать любую случайную строку битов нужной длины в качестве возможного ключа. Наличие слабых ключей нежелательно.
- Алгоритм должен легко модифицироваться для различных уровней безопасности и удовлетворять как минимальным, так и максимальным требованиям.
- Все операции с данными должны осуществляться над блоками, кратными байту или 32-битному слову.

Блочный шифр на основе сетей Фейстеля

- В 1971 году Хорст Фейстель (Horst Feistel) запатентовал два устройства, реализовавшие различные алгоритмы шифрования, названные затем общим названием «Люцифер» (Lucifer). Одно из устройств использовало конструкцию, впоследствии названную «сетью Фейстеля» .

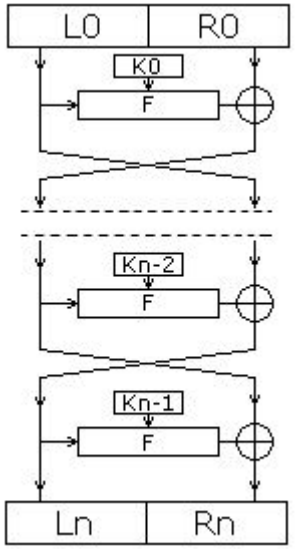
- Проект «Люцифер» был экспериментальным, но стал базисом для алгоритма Data Encryption Standard (DES).

- DES стал стандартом в США на шифрование данных, и до последнего времени широко использовался в криптографических системах.

- Новый блочный шифр был утверждён 26 мая 2002 года под названием Advanced Encryption Standard и вместо сети Фейстеля использует SP-сеть.

Сеть Фейстеля

Входной блок делится на несколько подблоков равной длины, называемых ветвями. В случае, если блок имеет длину 64 бита, используются две ветви по 32 бита каждая. Каждая ветвь обрабатывается независимо от другой, после чего осуществляется циклический сдвиг всех ветвей влево. Такое преобразование выполняется несколько раундов.



Функция F называется образующей.

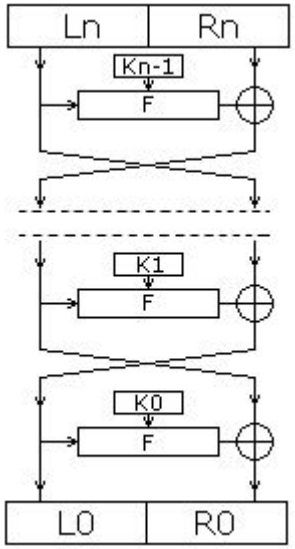
Каждый раунд состоит из вычисления функции F для одной ветви и побитового выполнения операции XOR результата F с другой ветвью. После этого ветви меняются местами. Считается, что оптимальное число раундов - от 8 до 32. Важно то, что увеличение количества раундов значительно увеличивает криптостойкость алгоритма. Возможно, эта особенность и повлияла на столь активное распространение сети Фейштеля, так как для большей криптостойкости достаточно просто увеличить количество раундов, не изменяя сам алгоритм. В последнее время количество раундов не фиксируется, а лишь указываются допустимые пределы.

Обратимость сети Фейстеля

Сеть Фейстеля является обратимой даже в том случае, если функция F не является таковой, так как для дешифрования не требуется вычислять F-1.

Для дешифрования используется тот же алгоритм, но на вход подается зашифрованный текст, и ключи используются в обратном порядке.

В настоящее время все чаще используются различные разновидности сети Фейстеля для 128-битного блока с четырьмя ветвями. Увеличение количества ветвей, а не размерности каждой ветви связано с тем, что наиболее популярными до сих пор остаются процессоры с 32-разрядными словами, следовательно, оперировать 32-разрядными словами эффективнее, чем с 64-разрядными.



Основной характеристикой алгоритма, построенного на основе сети Фейштеля, является функция F.

Различные варианты касаются также начального и конечного преобразований. Подобные преобразования, называемые забеливанием (whitening), осуществляются для того, чтобы выполнить начальную рандомизацию входного текста.

Алгоритм DES

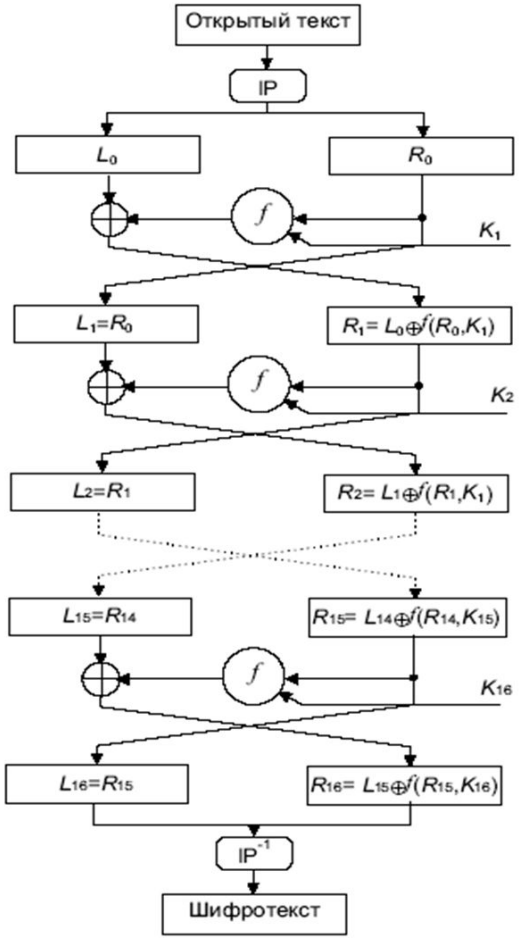
- Самым распространенным и наиболее известным алгоритмом симметричного шифрования является DES (Data Encryption Standard). Алгоритм был разработан в 1977 году, в 1980 году был принят NIST (National Institute of Standards and Technology США) в качестве стандарта (FIPS PUB 46).
- DES является классической сетью Фейштеля с двумя ветвями.
- Данные шифруются 64-битными блоками, используя 56-битный ключ. Алгоритм преобразует за несколько раундов 64-битный вход в 64-битный выход.

Процесс шифрования состоит из четырех этапов.

- выполняется начальная перестановка (IP) 64-битного исходного текста (забеливание), во время которой биты переупорядочиваются в соответствии со стандартной таблицей.
- Выполняется 16 раундов одной и той же функции, которая использует операции сдвига и подстановки.
- левая и правая половины выхода последней (16-й) итерации меняются местами.
- выполняется перестановка IP-1 результата, полученного на третьем этапе. Перестановка IP-1 инверсна начальной перестановке.

Первоначально ключ подается на вход функции перестановки. Затем для каждого из 16 раундов подключ Ki является комбинацией левого циклического сдвига и перестановки. Функция перестановки одна и та же для каждого раунда, но подключи Ki для каждого раунда получаются разные вследствие повторяющегося сдвига битов ключа.

Алгоритм


Открытый текст
IP
L_0
R_0
f
K_1
$L_1=R_0$
$R_1= L_0 \oplus f(R_0,K_1)$
K_2
$L_2=R_1$
$R_2= L_1 \oplus f(R_1,K_1)$
$L_{15}=R_{14}$
$R_{15}= L_{14} \oplus f(R_{14},K_{15})$
K_{16}
$L_{16}=R_{15}$
$R_{16}= L_{15} \oplus f(R_{15},K_{16})$
IP^{-1}
Шифротекст

Начальная перестановка

Начальная перестановка и ее инверсия определяются стандартной таблицей.
Если M - это произвольные 64 бита,
то X = IP (M) - переставленные 64 бита.

Если применить обратную функцию перестановки
$Y = IP^{-1}(X) = IP^{-1}(IP(M))$,
то получится первоначальная последовательность битов.

Пусть из файла считан очередной 8-байтовый блок T, который преобразуется с помощью матрицы начальной перестановки IP (табл.1) следующим образом: бит 58 блока T становится битом 1, бит 50 - битом 2 и т.д., что даст в результате: T(0) = IP(T).

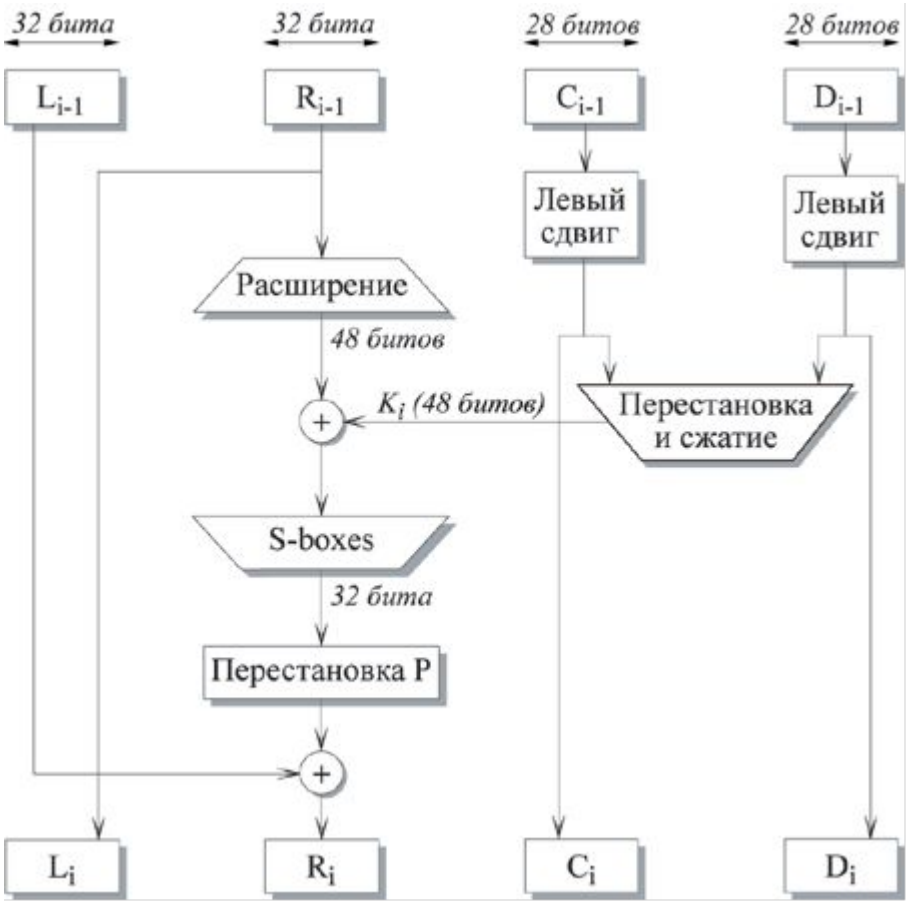
Полученная последовательность битов T(0) разделяется на две последовательности по 32 бита каждая: L(0) - левые или старшие биты, R(0) - правые или младшие биты.

Начальная перестановка

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

Последовательность преобразований отдельного раунда

Теперь рассмотрим последовательность преобразований, используемую в каждом раунде.



Описание раунда

64-битный входной блок проходит через 16 раундов, при этом на каждой итерации получается промежуточное 64-битное значение.

Левая и правая части каждого промежуточного значения трактуются как отдельные 32-битные значения, обозначенные L и R. Каждую итерацию можно описать следующим образом:

- Li = Ri-1
- Ri = Li-1+ F(Ri-1, Ki), где + обозначает операцию XOR.

Таким образом, выход левой половины Li равен входу правой половины Ri-1. Выход правой половины Ri является результатом применения операции XOR к Li-1 и функции F, зависящей от Ri-1 и Ki.

Рассмотрим функцию F более подробно

Ri, подается на вход функции F, имеет длину 32 бита.

- В начале Ri расширяется до 48 битов, используя таблицу, которая определяет перестановку плюс расширение на 16 битов.

- Расширение происходит следующим образом. 32 бита разбиваются на группы по 4 бита и затем расширяются до 6 битов, присоединяя крайние биты из двух соседних групп.

- Например, если часть входного сообщения

. . . efgh ijkl mnop . . .то в результате расширения получается сообщение

. . . defghi hijklm lmnopq . . .

Расширяющая перестановка

32	1	2	3	4	5	4	5	6	7	8	9
8	9	10	11	12	13	12	13	14	15	16	17
16	17	18	19	20	21	20	21	22	23	24	25
24	25	26	27	28	29	28	29	30	31	32	1

Функция F

- Для полученного 48-битного значения выполняется операция XOR с 48-битным подключом Ki.
- Затем полученное 48-битное значение подается на вход функции подстановки, результатом которой является 32-битное значение.

  Подстановка состоит из восьми S-boxes , каждый из которых на входе получает 6 бит, а на выходе создает 4 бита. Эти преобразования определяются специальными таблицами.

  Первый и последний биты входного значения S-box определяют номер строки в таблице, средние 4 бита определяют номер столбца. Пересечение строки и столбца определяет 4-битный выход.

  Например, если входом является 011011, то номер строки равен 01 (строка 1) и номер столбца равен 1101 (столбец 13). Значение в строке 1 и столбце 13 равно 5, т.е. выходом является 0101.
- Далее полученное 32-битное значение обрабатывается с помощью перестановки P, целью которой является максимальное переупорядочивание битов, чтобы в следующем раунде шифрования с большой вероятностью каждый бит обрабатывался другим S-box.

S-блоки

S1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Для получения результата функции шифрования надо переставить биты этой последовательности. Для этого применяется функция перестановки P (табл.5). Во входной последовательности биты перестанавливаются так, чтобы бит 16 стал битом 1, а бит 7 - битом 2 и т.д.

16 7 20 21 29 12 28 17 1 15 23 26 5 18 31 10
2 8 24 14 32 27 3 9 19 13 30 6 22 11 4
25

Создание подключей

Ключ для отдельного *раунда* K_i состоит из 48 битов.

Ключи K_i получаются по следующему алгоритму.

На каждой итерации используется новое значение ключа K(i), которое вычисляется из начального ключа K.

K представляет собой 64-битовый блок с восемью битами контроля по четности, расположенными в позициях 8,16,24,32,40,48,56,64.

Для удаления контрольных битов и перестановки остальных используется функция G первоначальной подготовки ключа

57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

Полученный 56-битный ключ разделяется на две 28-битные части, обозначаемые как C0 и D0 соответственно.

На каждом раунде Ci и Di независимо циклически сдвигаются влево на 1 или 2 бита, в зависимости от номера раунда.

Число битов сдвига ключа в зависимости от раунда

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

1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1

Полученные значения являются входом следующего раунда. Они также представляют собой вход в Permuted Choice 2 (PC-2), который создает 48-битное выходное значение, являющееся входом функции F(Ri-1, Ki).

Сжимающая перестановка

14 17 11 24 1 5 3 28 15 6 21 10

23 19 12 4 26 8 16 7 27 20 13 2

41 52 31 37 47 55 30 40 51 45 33 48

44 49 39 56 34 53 46 42 50 36 29 32

Заключительная перестановка

- На 16-й итерации получают последовательности R(16) и L(16) (без перестановки), которые конкатенируют в 64-битовую последовательность R(16)L(16).

- Затем позиции битов этой последовательности переставляют в соответствии с матрицей IP-1.

- Заключительная перестановка

40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

Криптоанализ

Процесс, при котором предпринимается попытка узнать X и K называется криптоанализом. Одной из возможных атак на алгоритм шифрования является лобовая атака, т.е. простой перебор всех возможных ключей. При длине ключа n бит количество возможных ключей равно 2n. Таким образом, чем длиннее ключ, тем более стойким считается алгоритм для лобовой атаки.

Существуют различные типы атак, основанные на том, что противнику известно определенное количество пар незашифрованное сообщение - зашифрованное сообщение. При анализе зашифрованного текста противник часто применяет статистические методы анализа текста.

Криптоаналитик может иметь возможность перехвата одного или нескольких незашифрованных сообщений вместе с их зашифрованным видом. Или криптоаналитик может знать основной формат или основные характеристики сообщения.

Криптографическая схема абсолютно безопасна, если зашифрованное сообщение не содержит никакой информации об исходном сообщении.

Криптографическая схема вычислительно безопасна, если:

- Цена расшифровки сообщения больше цены самого сообщения.
- Время, необходимое для расшифровки сообщения, больше срока жизни сообщения.

Дифференциальный криптоанализ

Предполагается, что известно достаточно большое количество пар (незашифрованнный текст, зашифрованный текст).

Понятие дифференциального криптоанализа было введено Эли Бихамом (Biham) и Ади Шамиром (Shamir) в 1990 году.

Конечная задача дифференциального криптоанализа - используя свойства алгоритма, в основном свойства табличной подстановки, определить подключ раунда.

Конкретный способ дифференциального криптоанализа зависит от рассматриваемого алгоритма шифрования.

Дешифрование

- Процесс дешифрования аналогичен процессу шифрования. На входе алгоритма используется зашифрованный текст, но ключи K_i используются в обратной последовательности. K_{16} используется на первом *раунде,* K_1 используется на последнем *раунде*. Пусть выходом i-ого *раунда* шифрования будет $L_i||R_i$. Тогда соответствующий вход (16-i)-ого *раунда* дешифрования будет $R_i||L_i$.
- После последнего *раунда* процесса расшифрования две половины выхода меняются местами так, чтобы вход заключительной перестановки IP^{-1} был $R_{16}||L_{16}$. Выходом этой стадии является незашифрованный текст.

Режимы работы алгоритма DES

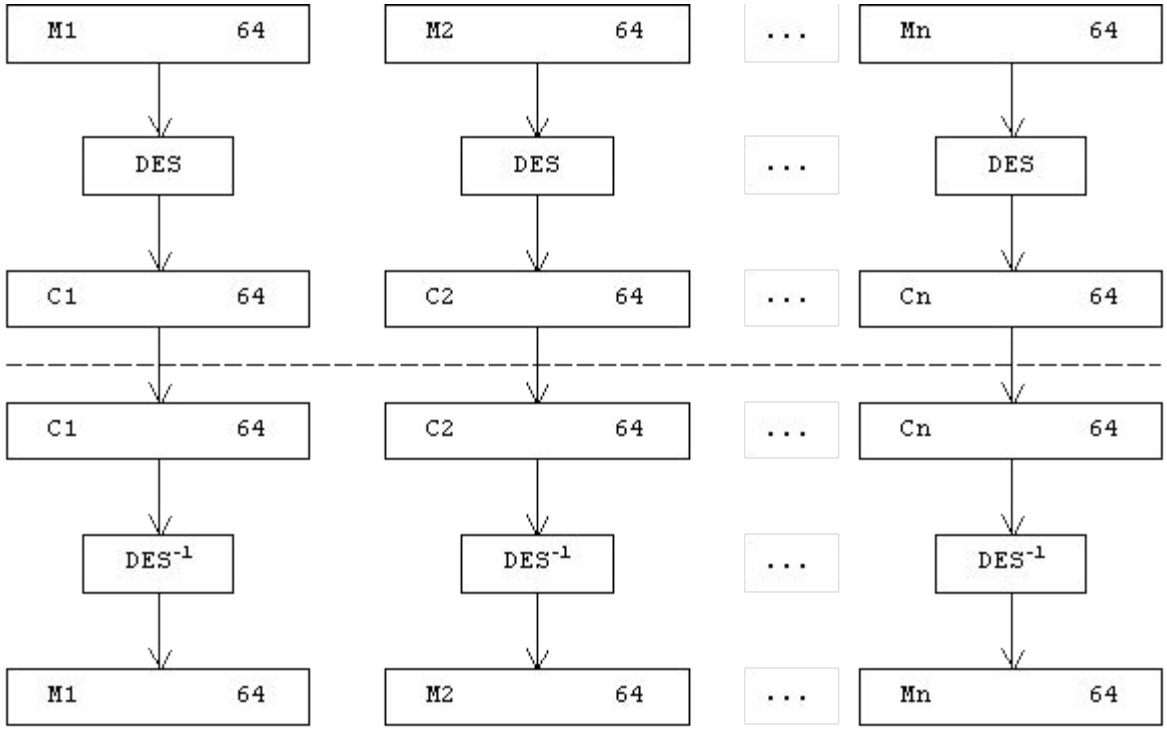
Для наиболее полного удовлетворения всем требованиям, предъявляемым к коммерческим системам шифрования, реализованы несколько режимов работы алгоритма DES.

Наиболее широкое распространение получили режимы:

- электронный шифроблокнот (Electronic Codebook) - ECB;
- цепочка цифровых блоков (Cipher Block Chaining) - CBC;
- цифровая обратная связь (Cipher Feedback) - CFB;
- внешняя обратная связь (Output Feedback) - OFB.

DES-ECB

- В этом режиме исходный файл M разбивается на 64-битовые блоки (по 8 байтов): M = M(1)M(2)...M(n). Каждый из этих блоков кодируется независимо с использованием одного и того же ключа шифрования.
- Основное достоинство этого алгоритма - простота реализации. Недостаток - относительно слабая устойчивость против квалифицированных криптоаналитиков.



DES-CBC

В этом режиме исходный файл M также, как и в режиме ECB, разбивается на 64-битовые блоки: M = M(1)M(2)...M(n).

- Первый блок M(1) складывается по модулю 2 с 64-битовым начальным вектором IV, который меняется ежедневно и держится в секрете.
- Полученная сумма затем шифруется с использованием ключа DES, известного и отправителю, и получателю информации.
- Полученный 64-битовый блок шифртекста C(1) складывается по модулю 2 со вторым блоком исходного текста, результат шифруется и получается второй 64-битовый блок шифртекста C(2) и т.д.
- Процедура повторяется до тех пор, пока не будут обработаны все блоки исходного текста .

Таким образом для всех i = 1...n блок шифртекста C(i) определяется следующим образом:

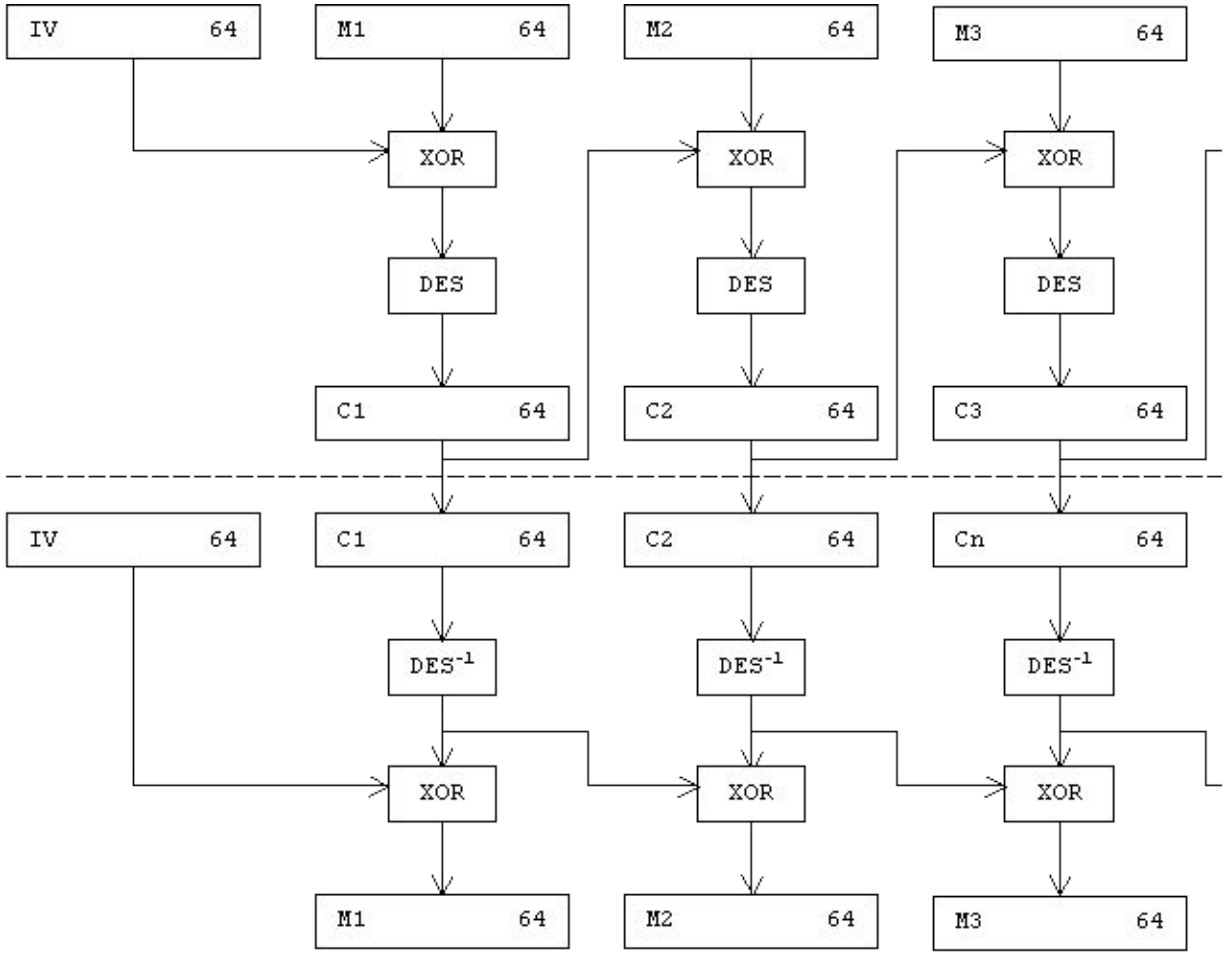
C(i) = DES(M(i) xor C(i-1)),

C(0) = IV - начальное значение шифра, равное начальному вектору.

Расшифрование выполняется следующим образом:

M(i) = C(i-1) xor DES^{-1}(C(i)),

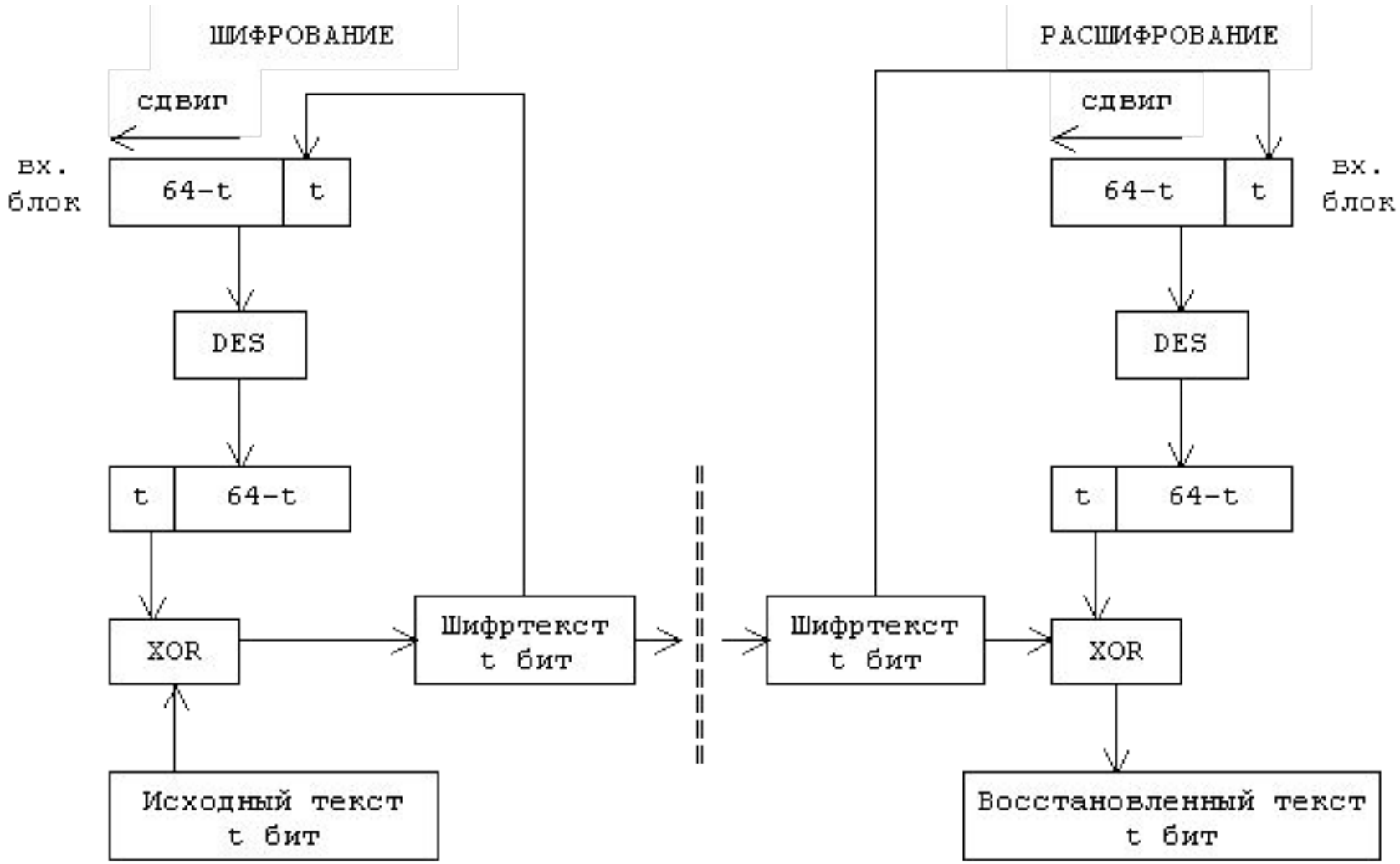
C(0) = IV - начальное значение шифра, равное начальному вектору.


IV 64
M1 64
M2 64
M3 64
XOR
XOR
XOR
DES
DES
DES
C1 64
C2 64
C3 64
IV 64
C1 64
C2 64
Cn 64
DES-1
DES-1
DES-1
XOR
XOR
XOR
M1 64
M2 64
M3 64

DES-CFB

В этом режиме размер блока может отличаться от 64. Исходный файл M считывается последовательными t-битовыми блоками (t <= 64): M = M(1)M(2)...M(n) (остаток дописывается нулями или пробелами).

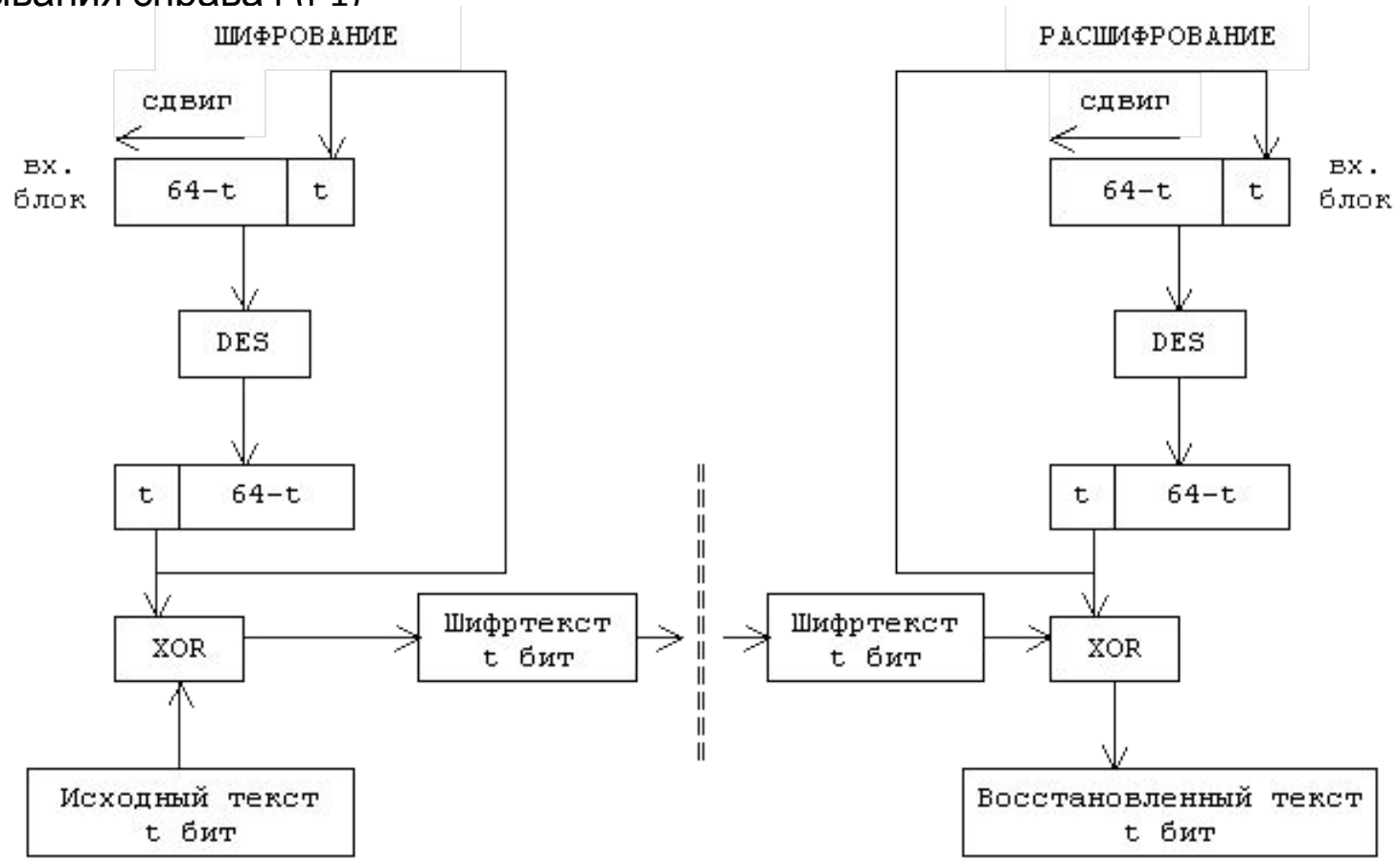
- 64-битовый сдвиговый регистр (входной блок) вначале содержит вектор инициализации IV, выравненный по правому краю. Для каждого сеанса шифрования используется новый IV.
- Для всех i = 1...n блок шифртекста C(i) определяется следующим образом:
- C(i) = M(i) xor P(i-1) ,
  где P(i-1) - старшие t битов операции DES(C(i-1)), причем C(0)=IV.Обновление сдвигового регистра осуществляется путем удаления его старших t битов и дописывания справа C(i).
- Восстановление зашифрованных данных также не представляет труда: P(i-1) и C(i) вычисляются аналогичным образом и
- M(i) = C(i) xor P(i-1) .


ШИФРОВАНИЕ
сдвиг
вх. блок
64-t
t
DES
t
64-t
XOR
Шифртекст t бит
Исходный текст t бит
РАСШИФРОВАНИЕ
сдвиг
64-t
t
вх. блок
DES
t
64-t
Шифртекст t бит
XOR
Восстановленный текст t бит

DES-OFB

Режим OFB очень похож на режим CFB.

Отличие от режима CFB состоит только в методе обновления сдвигового регистра. В данном случае это осуществляется путем удаления его старших t битов и дописывания справа P(i-1)



Криптоанализ сети Фейштеля

Если в основе алгоритма лежит сеть Фейштеля, то можно считать, что:

1. Блок m состоит из двух половин - m0 и m1.
2. Рассматриваются отличия, которые происходят в каждой половине при шифровании (операция XOR).
3. Выбирается пара незашифрованных текстов с фиксированным отличием. Затем анализируются отличия, получившиеся после шифрования одним раундом алгоритма, и определяются вероятности различных ключей.
4. Если для многих пар входных значений, имеющих одно и то же отличие X, при использовании одного и того же подключа одинаковыми (Y) оказываются и отличия соответствующих выходных значений, то можно говорить, что X влечет Y с определенной вероятностью.
5. Если эта вероятность близка к единице, то можно считать, что подключ раунда найден с данной вероятностью. Так как раунды алгоритма независимы, вероятности определения подключа каждого раунда следует перемножать.

Результаты дифференциального криптоанализа используются как при разработке конкретных подстановочных таблиц (S-box), так и при определении оптимального числа раундов.

Линейный криптоанализ

Линейный криптоанализ использует линейные приближения преобразований, выполняемых алгоритмом шифрования. Данный метод позволяет найти ключ, имея достаточно большое число пар (незашифрованный текст, зашифрованный текст).

Рассмотрим основные принципы, на которых базируется линейный криптоанализ.

Обозначим

- P[1], … , P[n] - незашифрованный блок сообщения.
- C[1], … , C[n] - зашифрованный блок сообщения.
- K[1], … , K[m] - ключ.
- A[i, j, …, k] = A[i]+ A[j] … A[k]

Линейный криптоанализ

Целью линейного криптоанализа является поиск линейного уравнения вида

$P[\alpha 1, \alpha 2, \ldots, \alpha a] + C[\beta 1, \beta 2, \ldots, \beta b] = K[\gamma 1, \ldots, \gamma c]$

Выполняющееся с вероятностью $p \neq 0.5$. αi, βi и γi - фиксированные позиции в блоках сообщения и ключе.

Чем больше p отклоняется от 0.5, тем более подходящим считается уравнение.

Если выполнить операцию XOR над некоторыми битами незашифрованного сообщения и над некоторыми битами зашифрованного сообщения, получится бит, представляющий собой XOR некоторых битов ключа. Это называется линейным приближением, которое может быть верным с вероятностью p.

Уравнения составляются следующим образом.

Вычисляются значения левой части для большого числа пар соответствующих фрагментов незашифрованного и зашифрованного блоков. Если результат оказывается равен нулю более чем в половине случаев, то полагают, что $K[\gamma 1, \ldots, \gamma c] = 0$. Если в большинстве случаев получается 1, полагают, что $K[\gamma 1, \ldots, \gamma c] = 1$. Таким образом получают систему уравнений, решением которой является ключ.

Как и в случае дифференциального криптоанализа, результаты линейного криптоанализа должны учитываться при разработке алгоритмов симметричного шифрования.