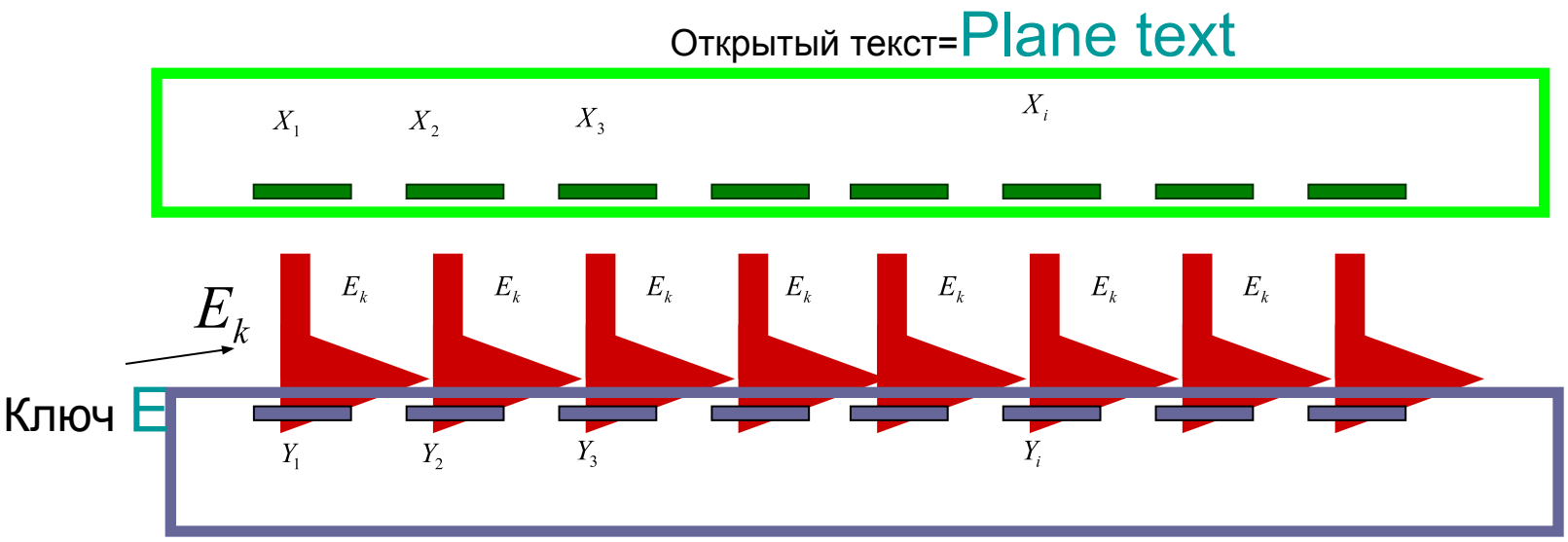
Симметричные блочные шифры



$$Y_i = E_k(X_i), i = 1,2... - \text{шифрование}$$

$$X_i = E_k^{-1}(Y_i), i = 1,2... - \text{дешифрование}$$

Блочными называются шифры, в которых логической единицей шифрования является некоторый блок открытого текста, после преобразования которого получается блок шифрованного текста такой же длины. Обычно используются блоки длиной 64 бита (128). Большинство сетевых приложений , в которых применяются схемы симметричного шифрования, используют именно блочные шифры.

Пусть блочный шифр оперирует с блоками длины n. Всего таких блоков 2**n. Обратимых преобразований блоков длины n в блоки такого же размера всего (2**n)!. Если n невелико, то такой шифр эквивалентен шифру подстановки на соответствующем алфавите и нестоек к частотному анализу. Чем больше n, тем менее эффективна статистическая атака. Выписать таблицу преобразования блоков длины 64 бита не представляется возможным.

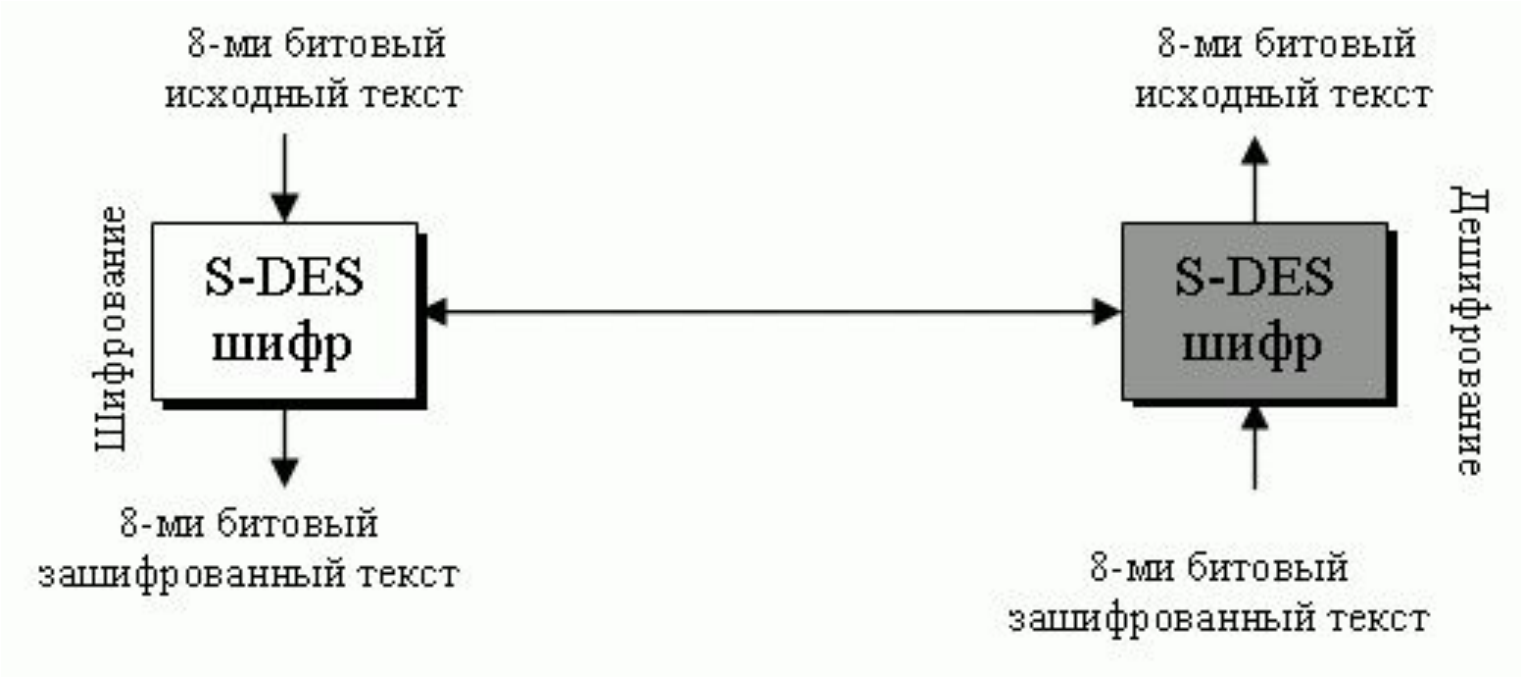
DES

Упрощенный DES (S-DES)

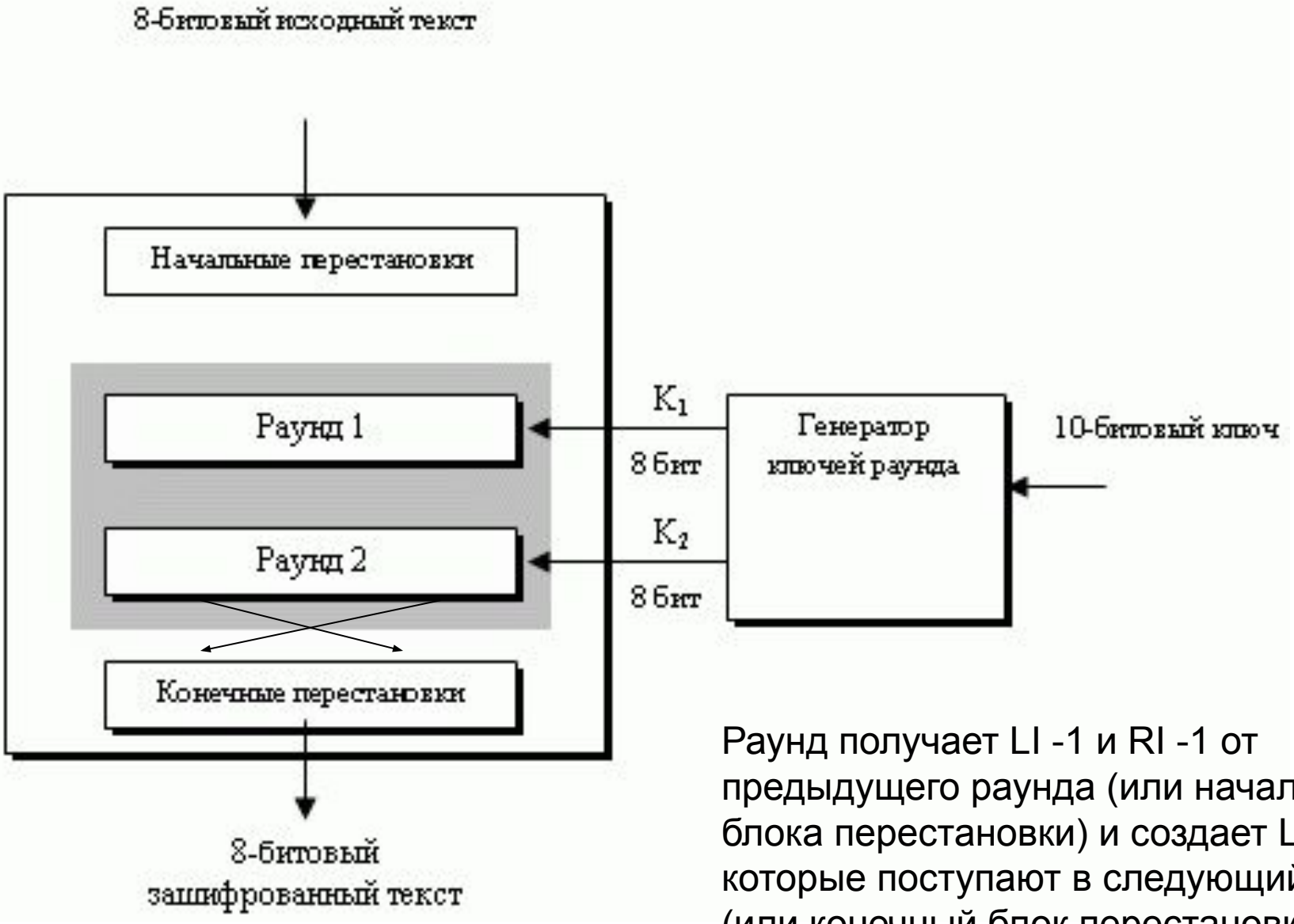
Разработан профессором Эдуардом Шаефером (Edward Schaefer) Университета Санта-Клары и является образовательным инструментом для помощи студентам при изучении структуры DES - для шифрования и дешифрования с использованием блочных шифров и ключей с небольшим количеством битов.

S-DES принимает исходный текст по 8 битов и создает зашифрованный текст по 8 битов. Один и тот же ключ шифра на 10 битов используется и для шифрования, и для дешифрования.

S-DES - блочный шифр, как это показано на рис.:



Процесс шифрования состоит из начальной перестановки IP, двух раундов Фейстеля и конечной перестановки IP-1. Каждый раунд использует



Раунд получает LI -1 и RI -1 от предыдущего раунда (или начального блока перестановки) и создает LI и RI, которые поступают в следующий раунд (или конечный блок перестановки).

Начальная и конечная перестановка

IP: [12345678]->[26314857].
IP-1:[12345678]->[41357286].

Пример:

Шифротекст:

1	2	3	4	5	6	7	8
1	1	0	0	1	1	0	0

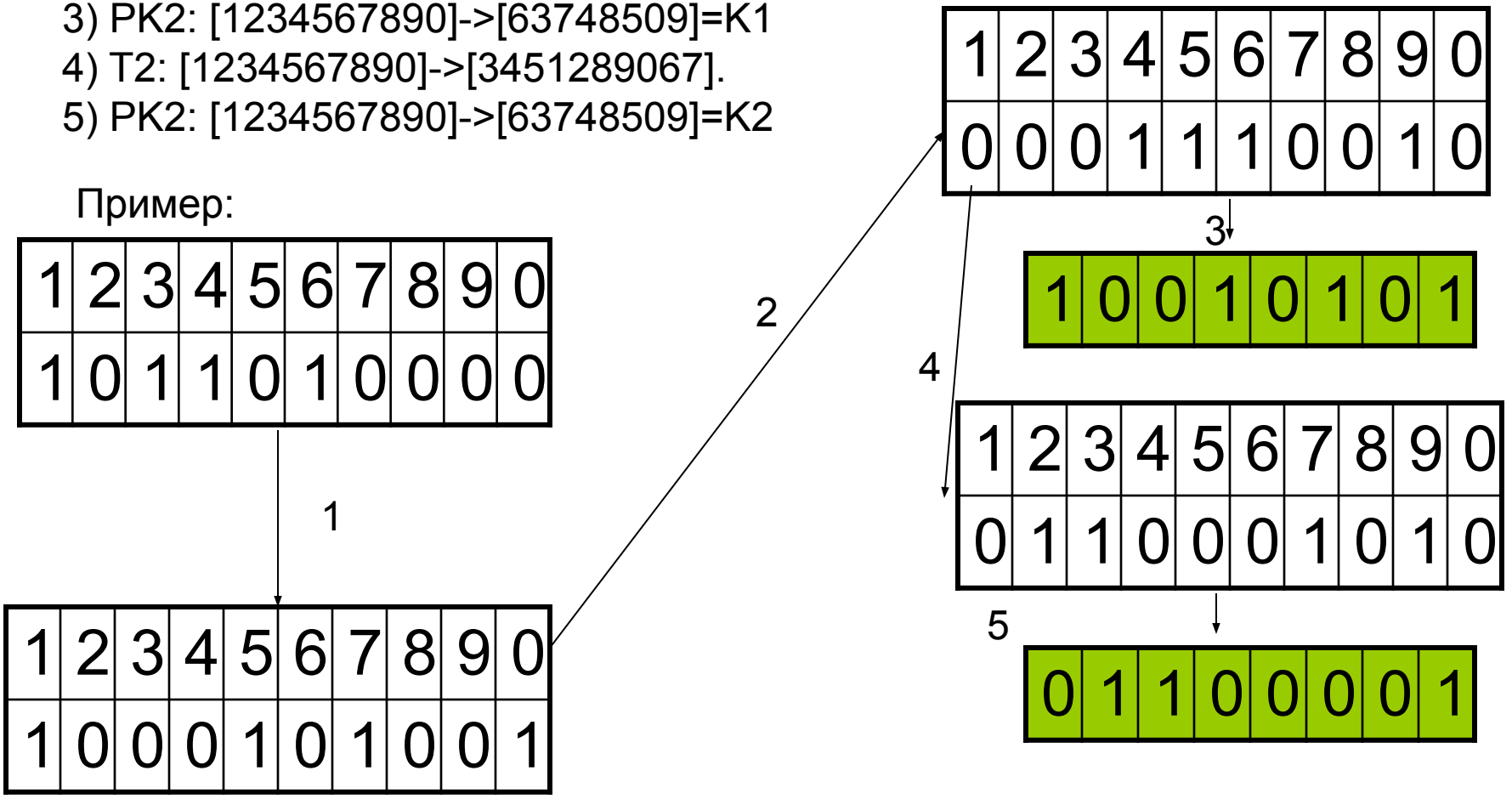
4	1	3	5	7	2	8	6
0	1	0	1	0	1	0	1

Генерация раундовых ключей

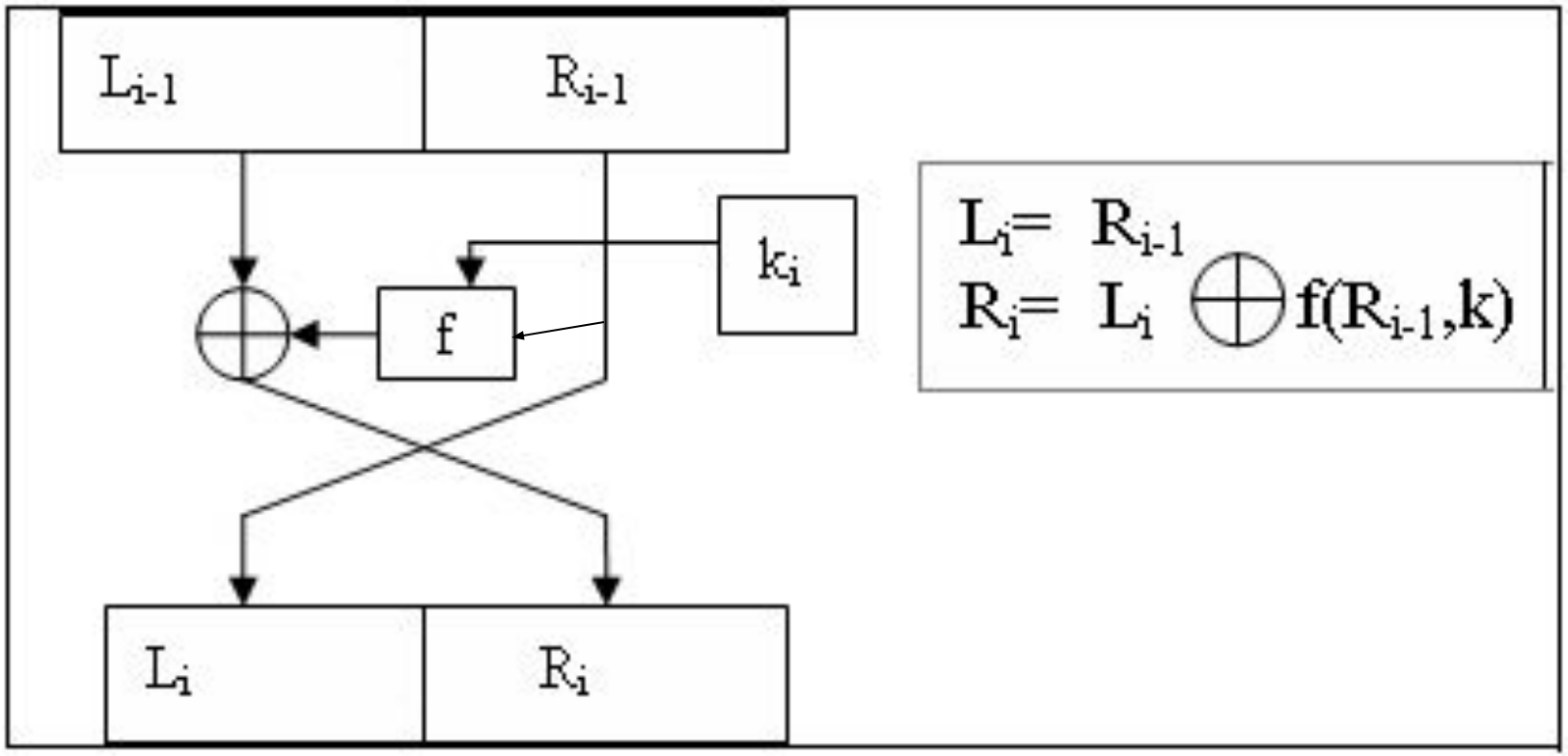
Генератор ключей раунда создает два ключа на 8 битов из ключа шифра на 10 битов.

1) PK1: [1234567890]->[3527401986].
2) T1: [1234567890]->[2345178906].
3) PK2: [1234567890]->[63748509]=K1
4) T2: [1234567890]->[3451289067].
5) PK2: [1234567890]->[63748509]=K2

Пример:



Структура раунда Сети Фейстеля:



Функция усложнения сети Фейстеля составлена из четырех действий:

1) P1-блока расширения,
2) XOR сложения с раундовым ключом,
3) группы S-блоков
4) P2-блока.

P1-блок расширения:

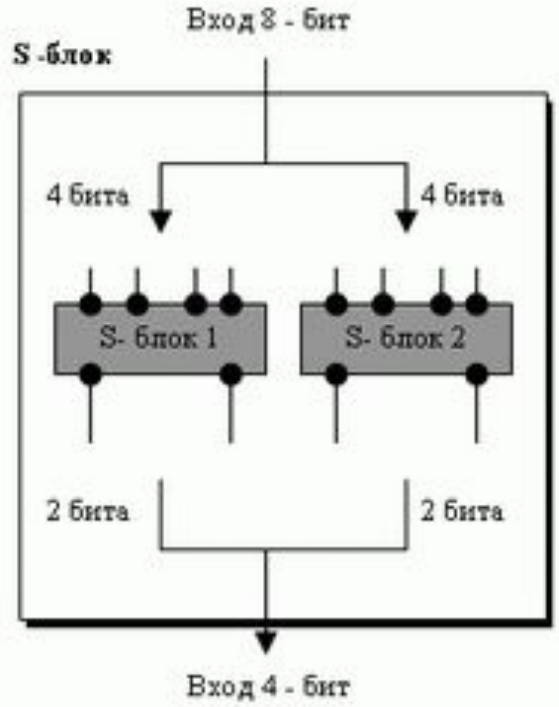
P1 :[1234]->[41232341].

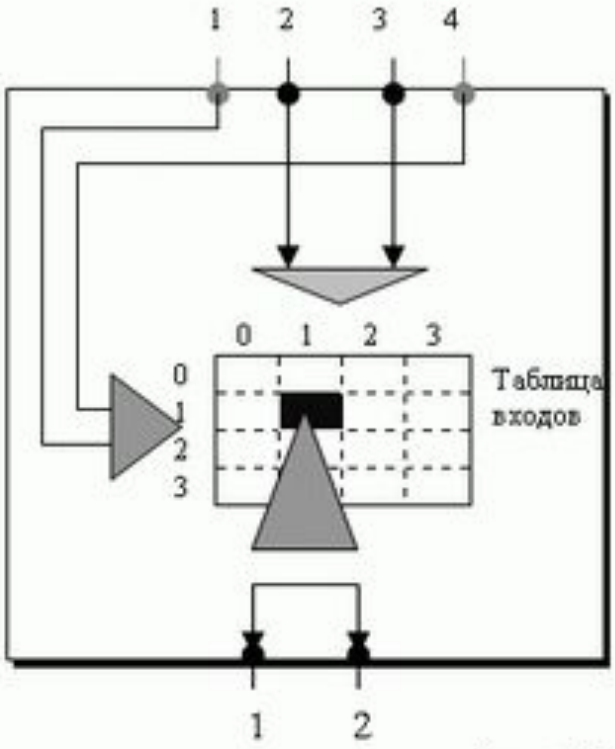
Пример.

1	2	3	4
1	1	0	0

4	1	2	3	2	3	4	1
0	1	1	0	1	0	0	1

S-блоки





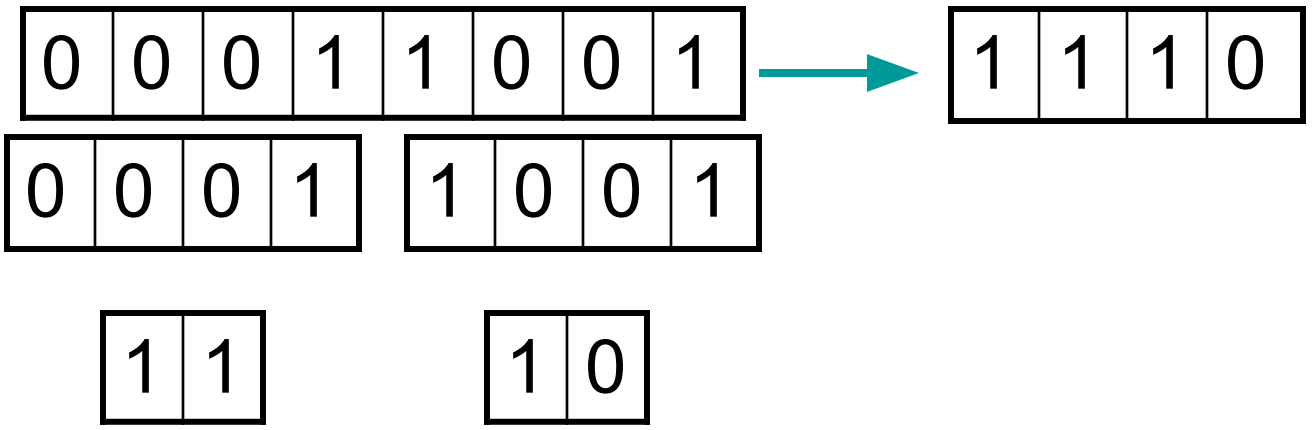
S1=[1 0 3 2;
3 2 1 0;
0 2 1 3;
3 1 0 2]

Таблица для S -блока 1

S2=[0 1 2 3;
2 1 0 3;
3 0 1 2;
2 1 0 3]

Таблица для S -блока 2

Пример:



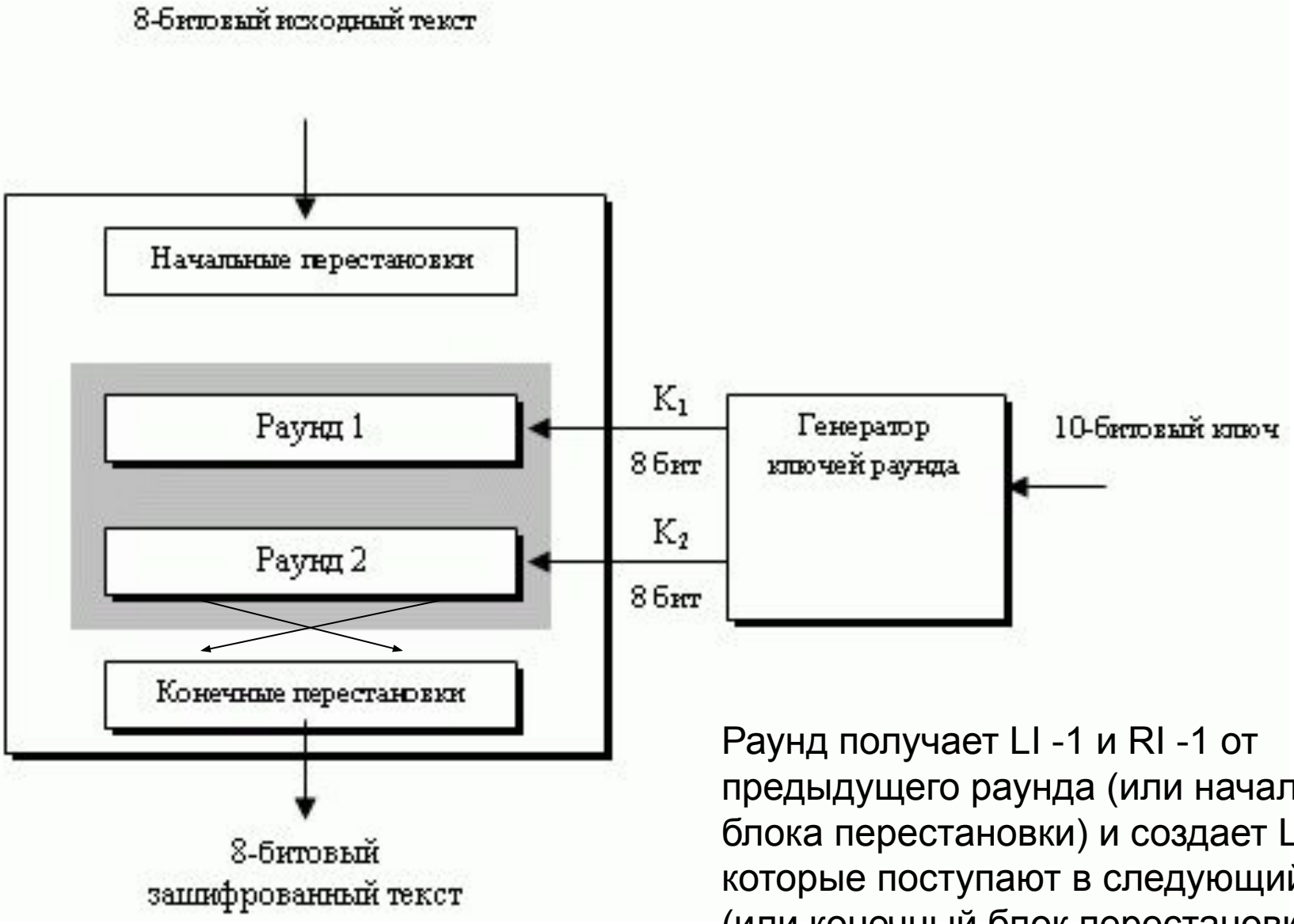
Перестановка P2

P2: [1234]->[2431].

1	2	3	4
1	1	0	0

2	4	3	1
1	0	0	1

Процесс шифрования состоит из начальной перестановки IP, двух раундов Фейстеля и конечной перестановки IP-1. Каждый раунд использует



Раунд получает LI -1 и RI -1 от предыдущего раунда (или начального блока перестановки) и создает LI и RI, которые поступают в следующий раунд (или конечный блок перестановки).

DES

История

В 1972 году было проведено исследование потребности правительства США в компьютерной безопасности. Американское «национальное бюро стандартов» (НБС) (ныне известное, как NIST — «национальный институт стандартов и технологий») определило необходимость в общеправительственном стандарте шифрования некритичной информации.

НБС проконсультировалось с АНБ (агентством национальной безопасности США) и 15 мая 1973 года объявило первый конкурс на создание шифра. Были сформулированы строгие требования к новому шифру. Фирма IBM представила на конкурсе разработанный её шифр, называемый «Люцифер» (Lucifer). Шифры ни одного из конкурсантов (включая «Люцифер») не обеспечивали выполнение всех требований. В течение 1973-1974 годов IBM доработала свой «Люцифер»: использовала в его основе алгоритм Хорста Фейстеля, созданный ранее. 27 августа 1974 года начался второй конкурс. На сей раз шифр «Люцифер» сочли приемлемым.

17 марта 1975 года предложенный алгоритм DES был издан в «Федеральном реестре». В 1976 году для обсуждения DES было проведено два открытых симпозиума. На симпозиумах жёсткой критике подверглись изменения, внесённые в алгоритм организацией АНБ. АНБ уменьшило первоначальную длину ключа и S-блоки (блоки подстановки), критерии проектирования которых не раскрывались. АНБ подозревалось в сознательном ослаблении алгоритма с той целью, чтобы АНБ могло легко просматривать зашифрованные сообщения. Сенат США проверил действия АНБ и в 1978 году опубликовал заявление, в котором сообщалось следующее:

в процессе разработки алгоритма представители АНБ убедили создателей DES в том, что уменьшенной длины ключа более чем достаточно для всех коммерческих приложений;

представители АНБ косвенно помогали в разработке S-перестановок;

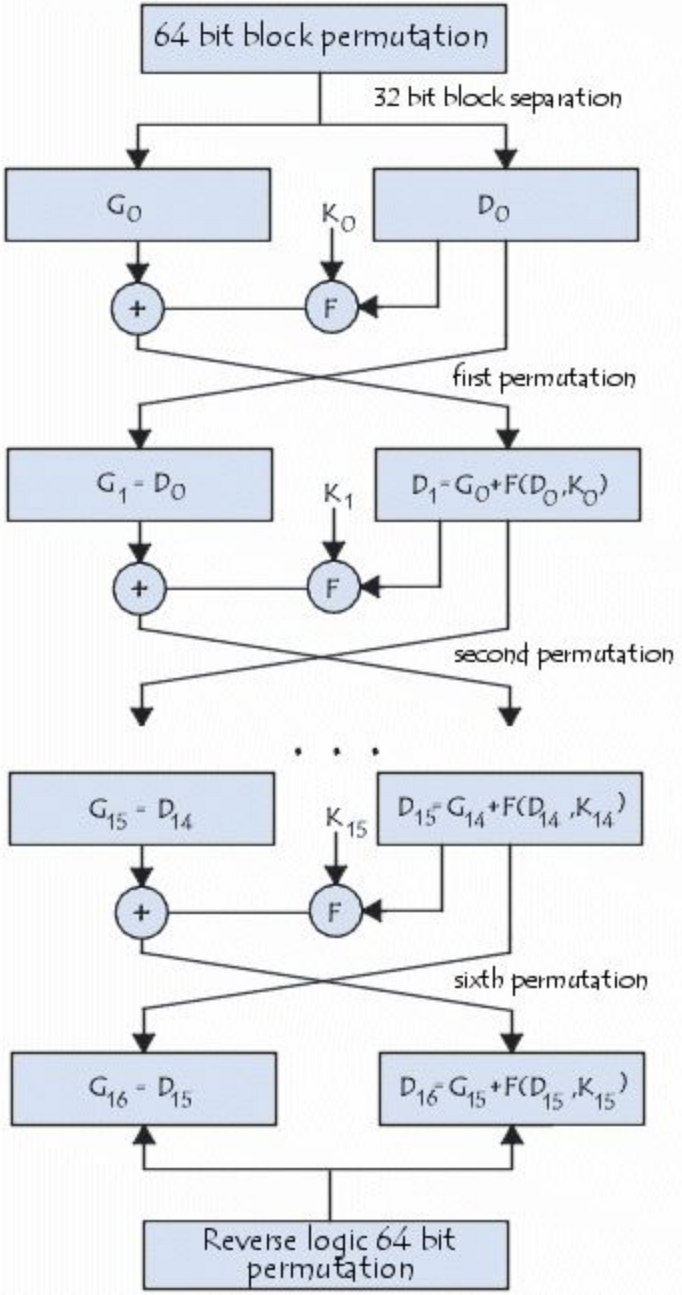
окончательная версия алгоритма была, по мнению проверяющих, лучшим алгоритмом шифрования, к тому же лишённым статистической или математической слабости;

представители АНБ никогда не вмешивались в разработку алгоритма DES.

В 1990 году Эли Бихам (Eli Biham) и Ади Шамир (Adi Shamir) провели независимые исследования по дифференциальному криптоанализу — основному методу взлома блочных алгоритмов симметричного шифрования. Эти исследования сняли часть подозрений в скрытой слабости S-перестановок. S-блоки алгоритма DES оказались намного более устойчивыми к атакам, чем если бы их выбрали случайно.

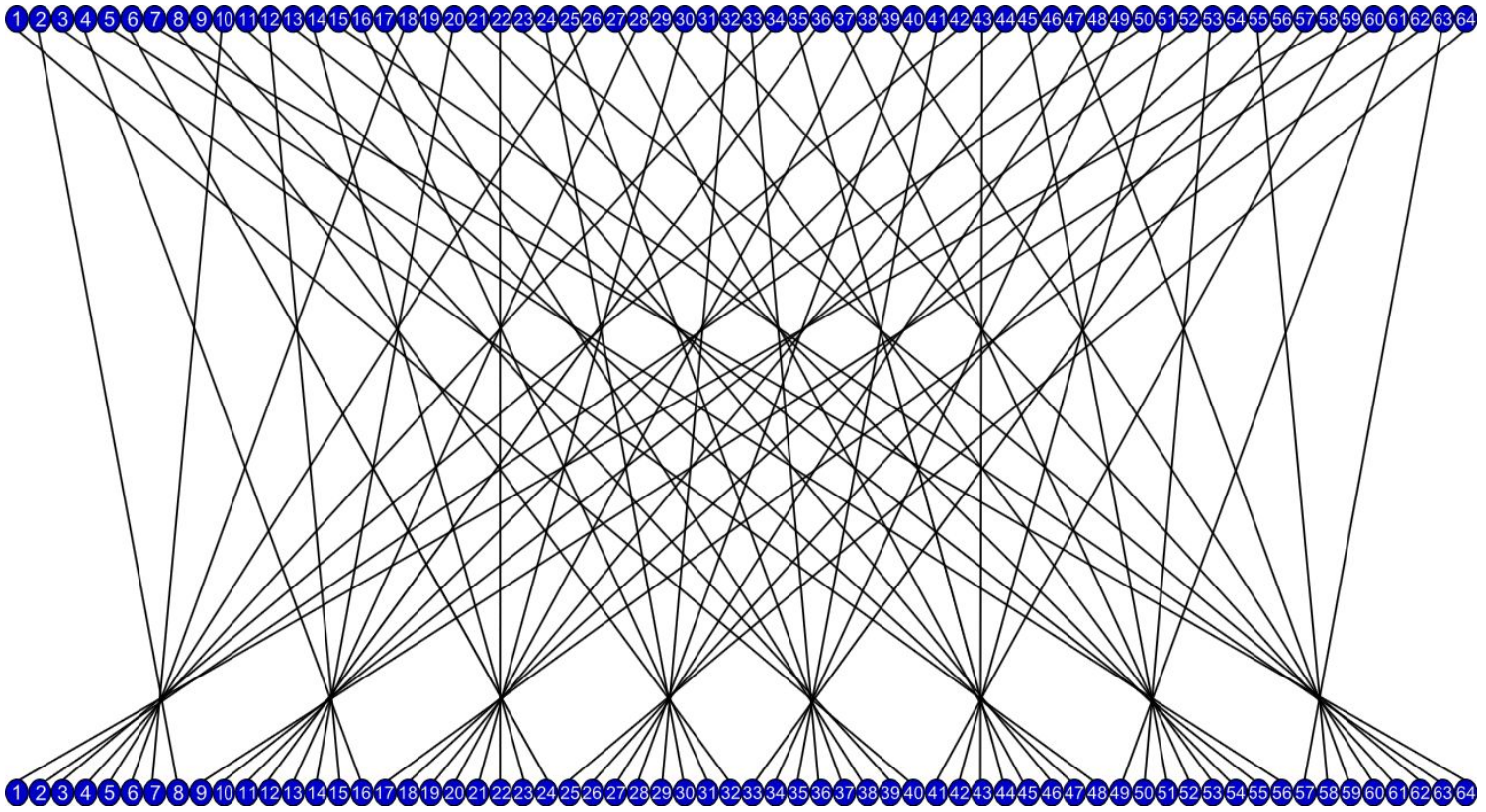
Алгоритм DES – шифр Фейстеля со следующими параметрами
1) 64 – битные блоки;
2) 56 битный ключ;
3) 16 раундов шифрования
4) Функция F –композиционная.
Схема шифрования.
Открытый текст -> Начальная перестановка IP -> Шифр Фейстеля -> IP-1 ->Шифрованный текст

В основе алгоритма лежит цепь Фейстеля, чья раундовая функция *F* обрабатывает 32-разрядные слова (половину блока) и использует в качестве параметра 48-разрядный подключ


64 bit block permutation
32 bit block separation
G_0
K_0
D_0
+
F
first permutation
$G_1 = D_0$
K_1
$D_1 = G_0 + F(D_0, K_0)$
+
F
second permutation
$G_{15} = D_{14}$
K_{15}
$D_{15} = G_{14} + F(D_{14}, K_{14})$
+
F
sixth permutation
$G_{16} = D_{15}$
$D_{16} = G_{15} + F(D_{15}, K_{15})$
Reverse logic 64 bit permutation

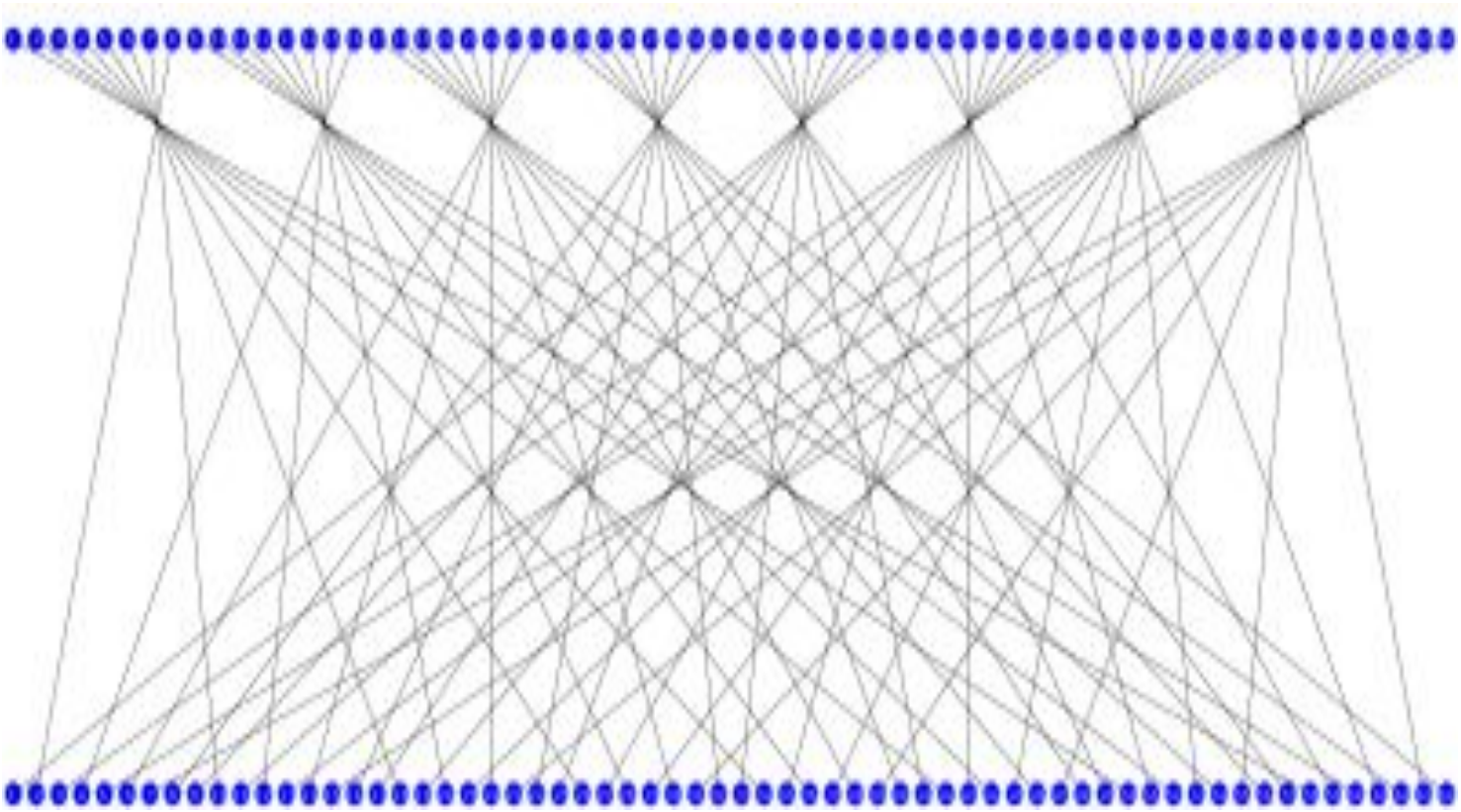
IP — Initial Permutation

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

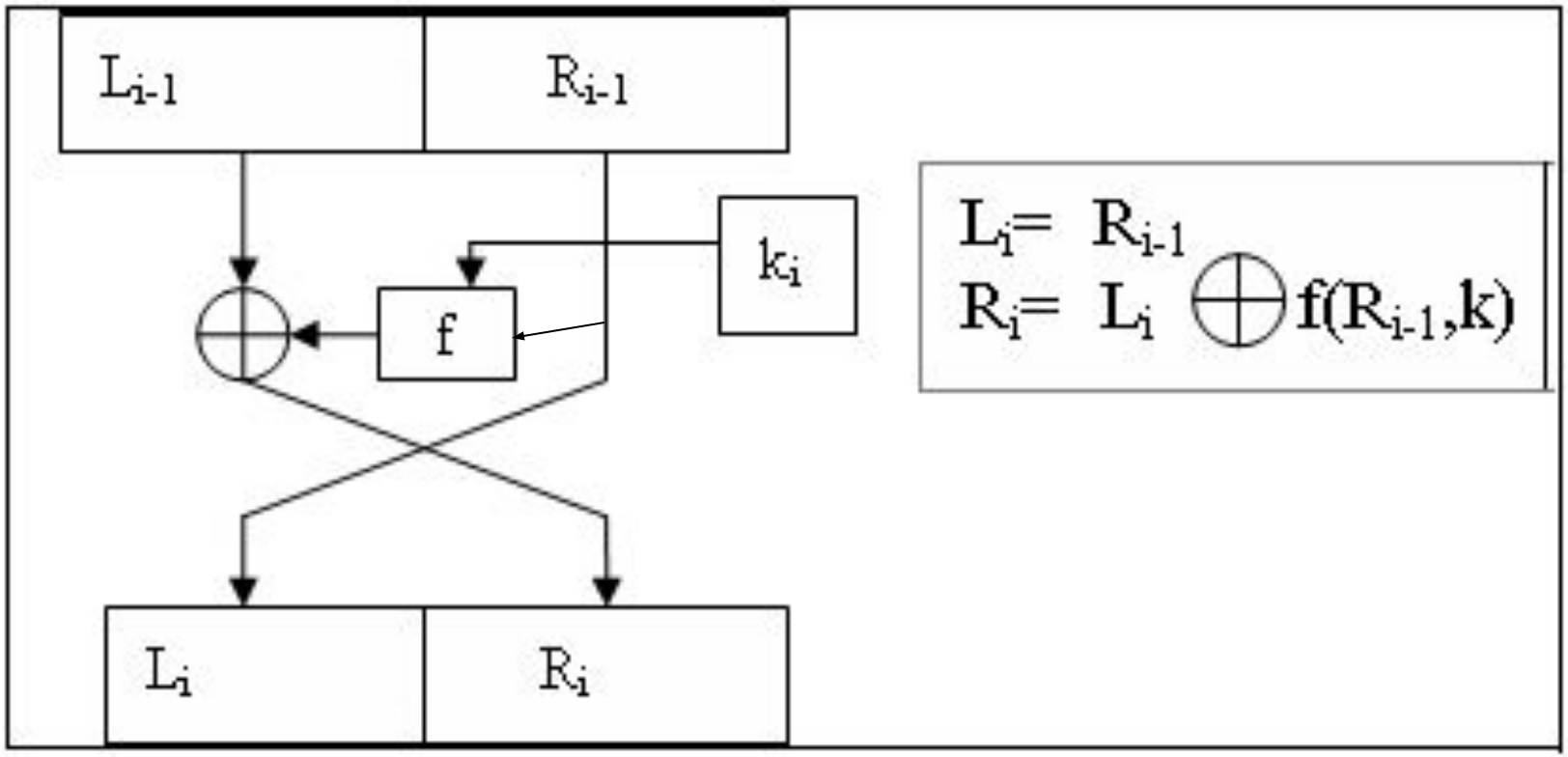


IP^{-1} — Final Permutation

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

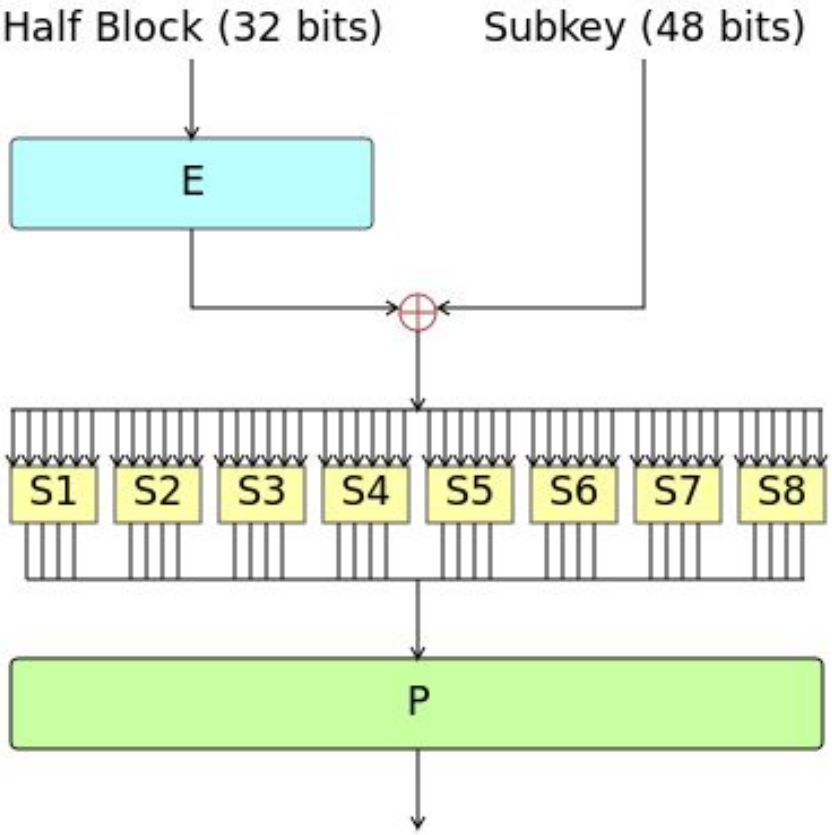


Структура раунда Сети Фейстеля:



Функция F

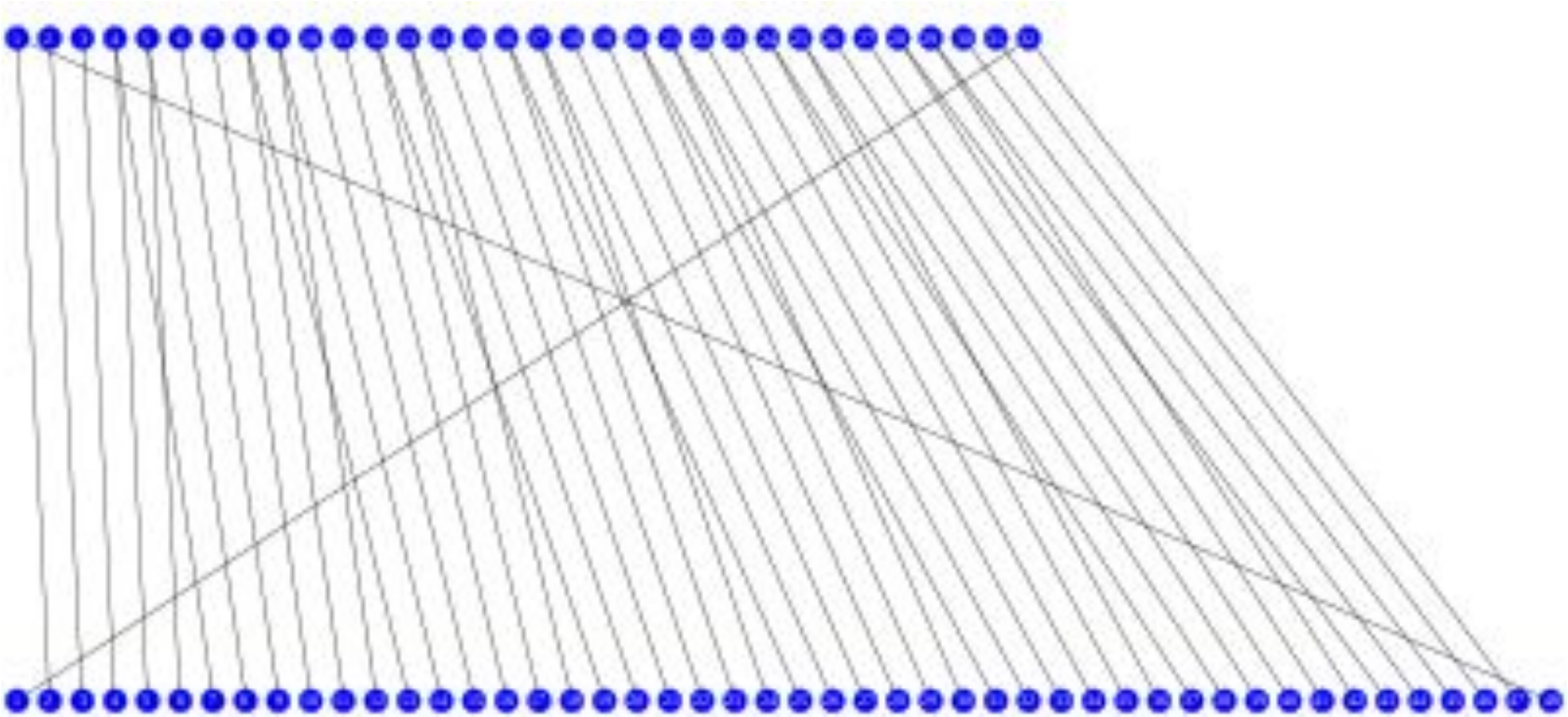
- Расширение E 32-битового вектора до 48-битового вектора.
- Сложение с 48-битовым ключом (XOR).
- Нелинейная замена с помощью s-блоков S1,S2,… S8 48-битового вектора на 32 битовый вектор.
- Перемешивание координат 32-битового вектора с помощью перестановки P



Перестановка с расширением E (Expansion function)

E					
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

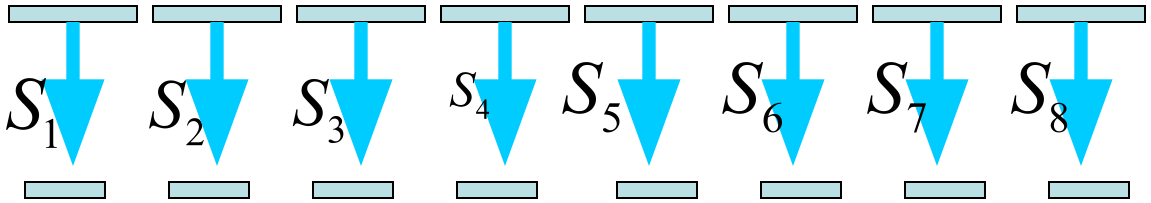
32 входные разряда с помощью расширяющей перестановки преобразуются в 48:



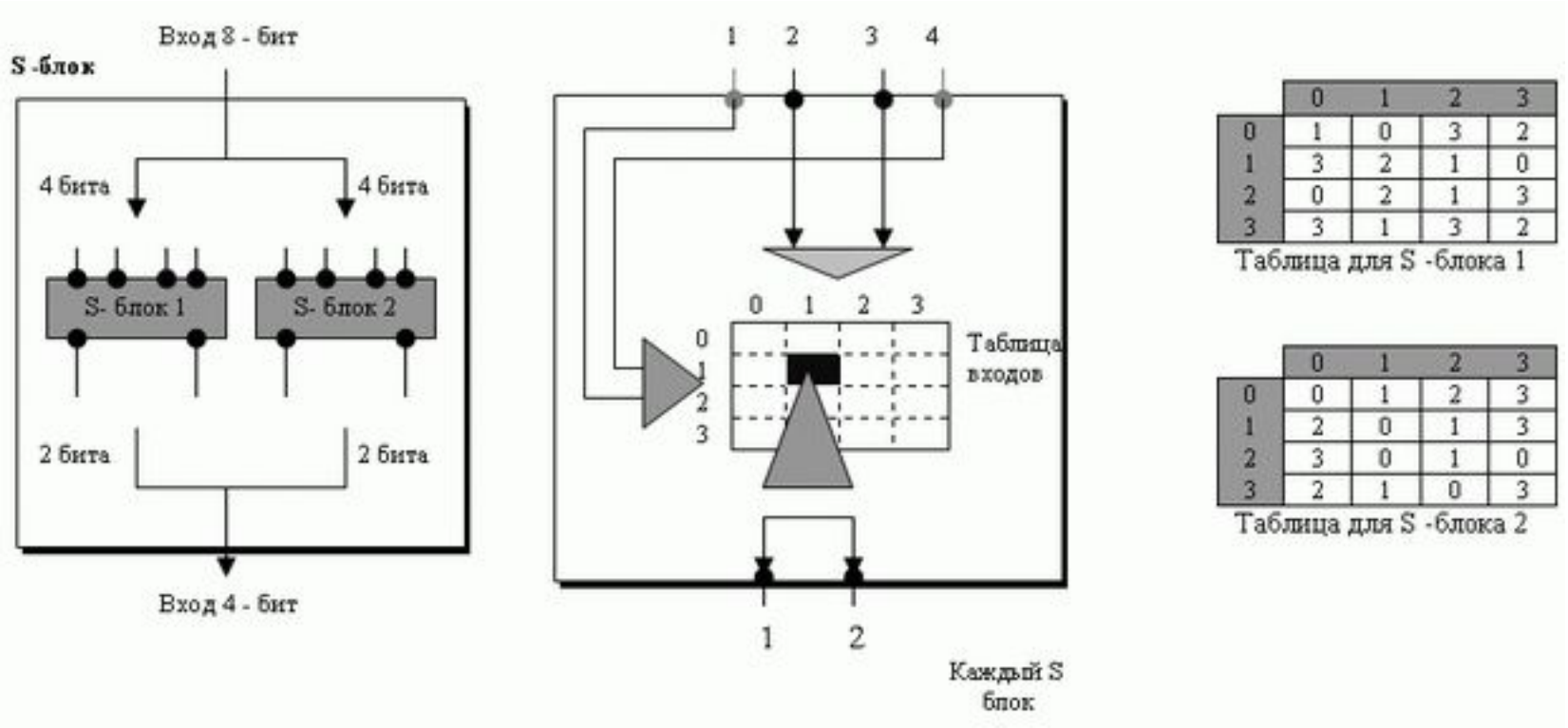
2. «Подмешивание» циклового ключа

Значение на выходе ключевого сумматора получается в результате побитового сложения сформированного 48-разрядного блока и текущего подключа.

3. Нелинейная замена S



Входной 48-разрядный блок делится на 8 групп по 6 разрядов.
В результате каждые 6 бит преобразуются в 4 бита,
а весь 48-разрядный код в 32-разрядный (для этого нужно 8 *S*-блоков).



	0	1	2	3
0	1	0	3	2
1	3	2	1	0
2	0	2	1	3
3	3	1	3	2

Таблица для S -блока 1

	0	1	2	3
0	0	1	2	3
1	2	0	1	3
2	3	0	1	0
3	2	1	0	3

Таблица для S -блока 2

Конструкция S-блока

S-блоки представляют собой таблицы, содержащие 4 строки и 16 столбцов.

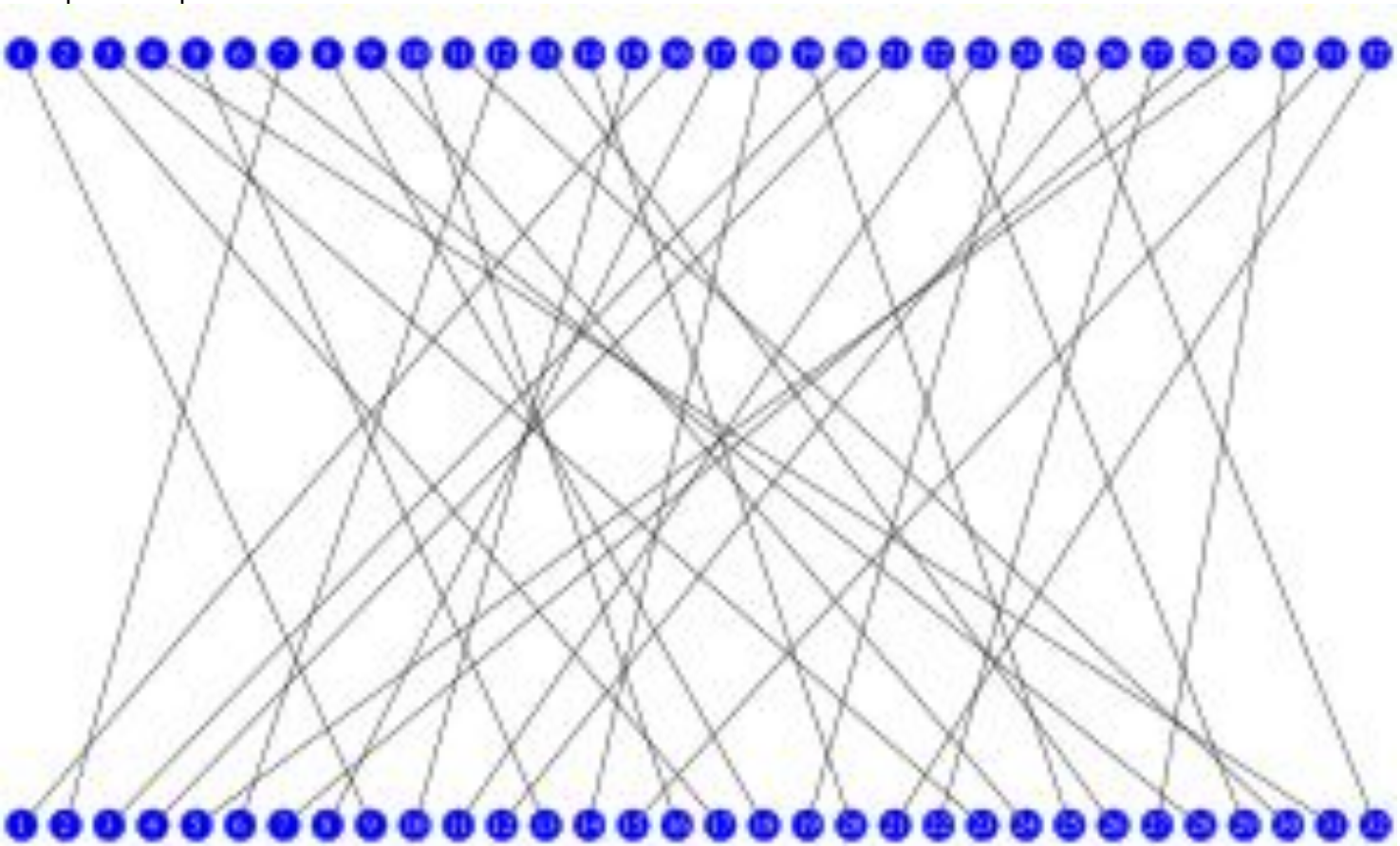
Первый и последний разряд в группе используется в качестве адреса строки, а средние 4 разряда — в качестве адреса столбца. В результате каждые 6 бит преобразуются в 4 бита

S-блок *S*1:

No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

P			
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

Перестановка P



Цикловые ключи.

1) Начальная перестановка с выбором PC1.

Используется 64-битный ключ.

Сначала к нему применяется функция перестановки с выбором PC1. Получается 56 битное значение.

PC1=

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**

2) Циклический сдвиг.

Полученное 56-битовое значение рассматривается как комбинация двух 28-битовых значений. В каждом раунде шифрования к правой и левой части ключа по отдельности применяются операции циклического сдвига влево на 1 или 2 бита в соответствии с таблицей сдвигов.

Таблица сдвигов

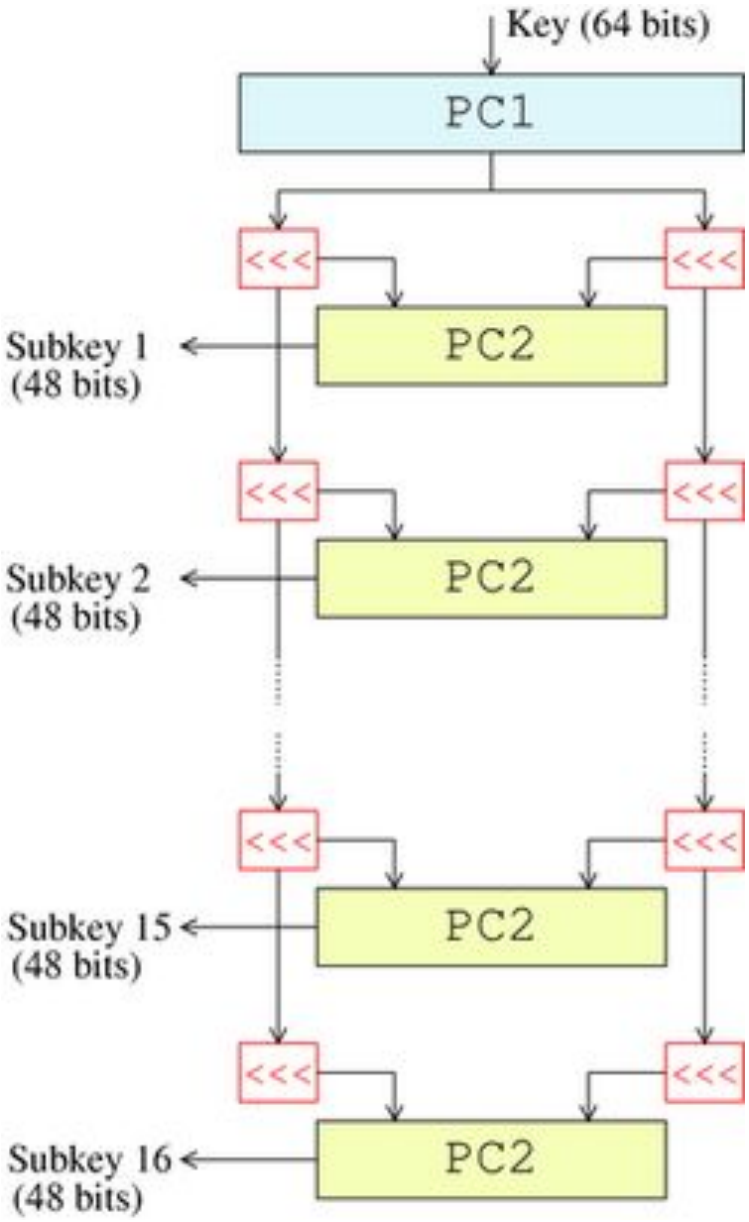
1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

3) Перестановка PC2

Затем с помощью другой перестановки PC2 из полученного результата для каждого из 16 раундов генерируется подключ Ki длины 48 битов. Функция перестановки одна и та же для всех раундов, но генерируемые подключи оказываются разными из-за того, что в результате циклического сдвига на ее вход поступают разные биты ключа.

PC2					
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

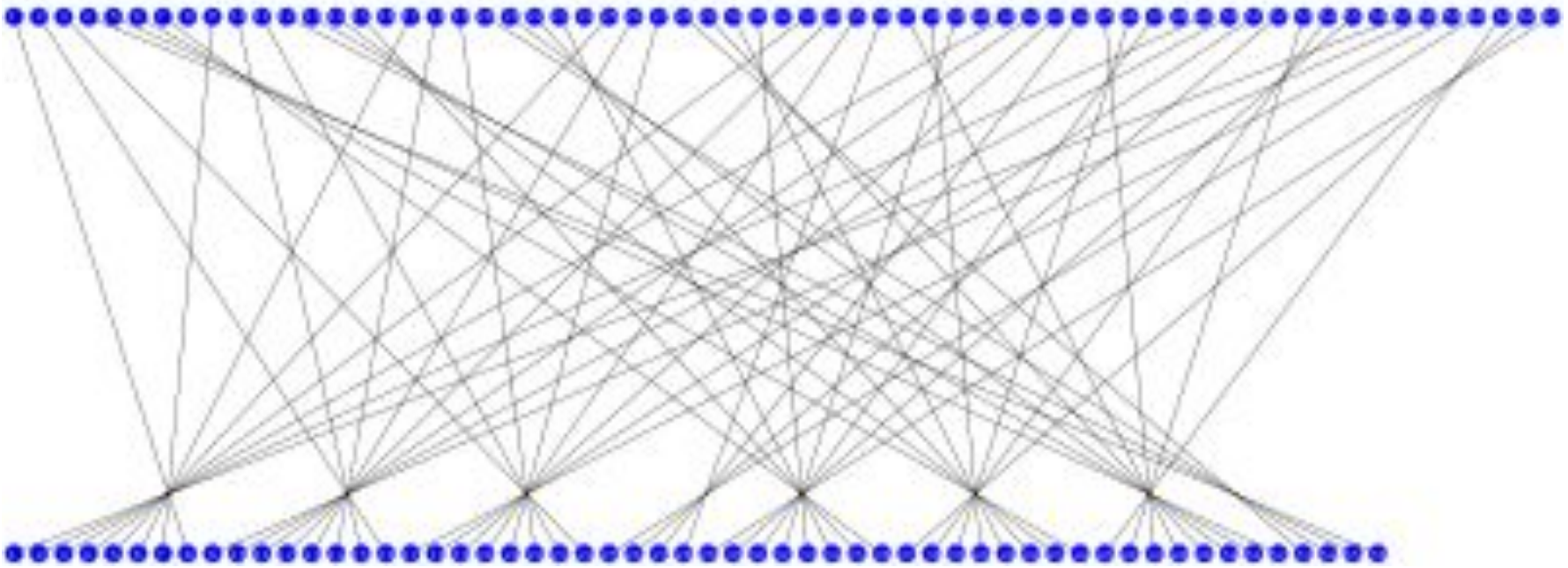
1-ый бит отображается в 14 позицию
2-ой бит отображается в 17 позицию

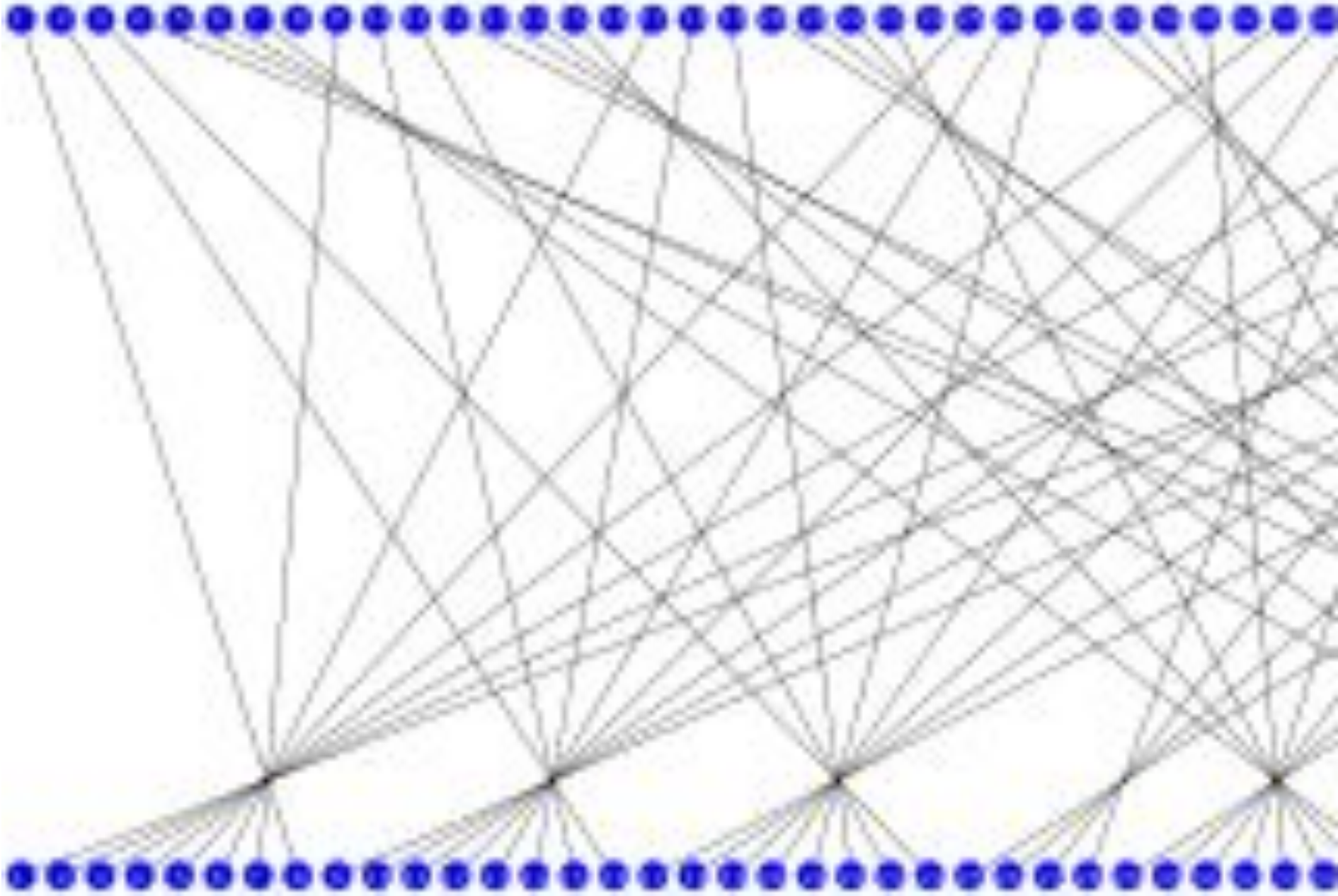


Цикловые ключи

Начальная перестановка с выбором PC1 :

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





2. **Циклический сдвиг.**

Полученное 56-битовое значение рассматривается как комбинация двух 28-битовых значений. В каждом раунде шифрования к правой и левой части ключа по отдельности применяются операции циклического сдвига влево на 1 или 2 бита в соответствии с таблицей сдвигов.

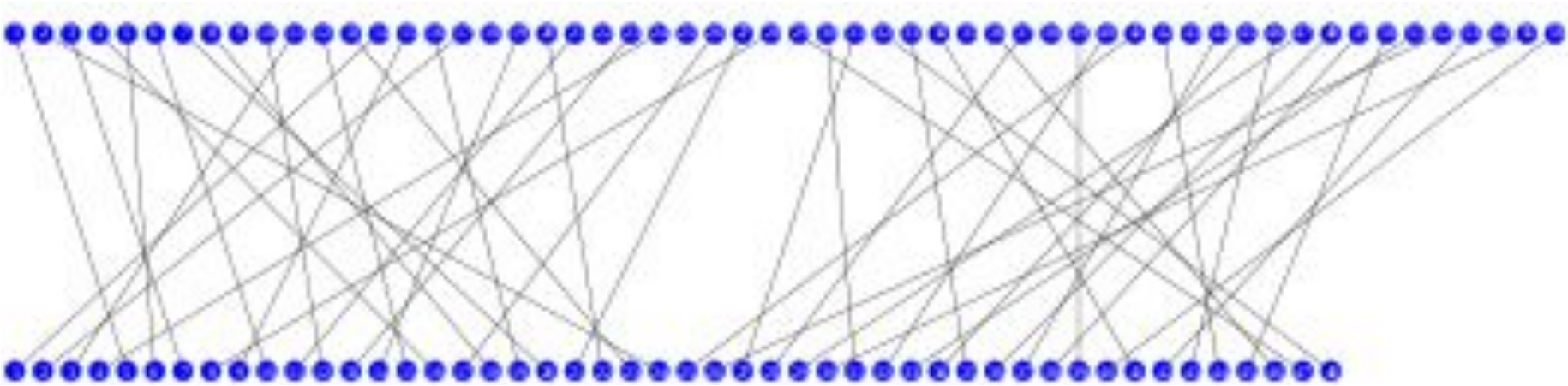
Таблица сдвигов

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

С помощью другой перестановки PC2 из полученного результата для каждого из 16 раундов генерируется подключ Ki длины 48 битов. Функция перестановки одна и та же для всех раундов, но генерируемые подключи оказываются разными из-за того, что в результате циклического сдвига на ее вход поступают разные биты ключа.

3. **Перестановка PC2**

PC2					
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



DES also has four so-called *weak keys*. Encryption (*E*) and decryption (*D*) under a weak key have the same effect (see involution):

or equivalently,

There are also six pairs of *semi-weak keys*. Encryption with one of the pair of semiweak keys, , operates identically to decryption with the other, :

or equivalently,

It is easy enough to avoid the weak and semiweak keys in an implementation, either by testing for them explicitly, or simply by choosing keys randomly; the odd

DES has also been proved not to be a group, or more precisely, the set (for all possible keys) under functional composition is not a group, nor "close" to bei

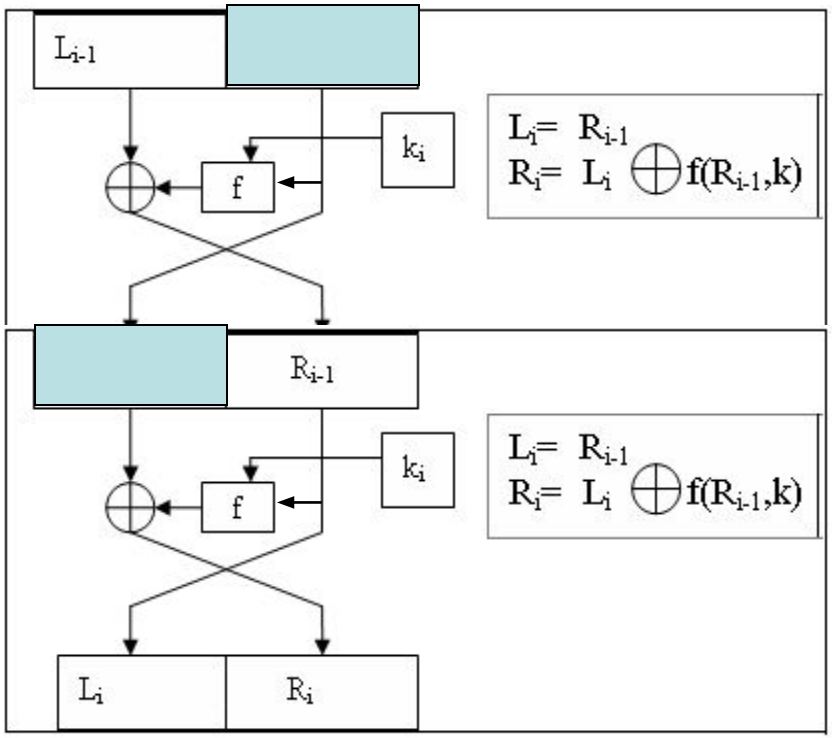
It is known that the maximum cryptographic security of DES is limited to about 64 bits, even when independently choosing all round subkeys instead of deriving

$$E_K(E_K(P)) = P = D_K.$$

$$E_{K_1}(E_{K_2}(P)) = P = D_{K_1}.$$

Прямое и обратное преобразование сети Фейстеля

Криптостойкость и диффузия



1	4
2	16
3	64
4	256
5	1024
6	4096
7	16384
8	65536

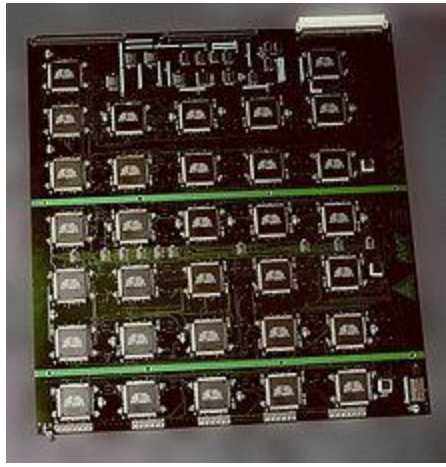
Нарастание диффузии от раунда к раунду.

Стойкость шифра часто измеряют в относительном числе раундов, которые поддаются взлому.

Brute force attack

For DES, questions were raised about the adequacy of its key size early on, even before it was adopted as a standard, and it was the small key size, rather than theoretical cryptanalysis, which dictated a need for a replacement algorithm.

In 1977, Diffie and Hellman proposed a machine costing an estimated US$20 million which could find a DES key in a single day. By 1993, Wiener had proposed a key-search machine costing US$1 million which would find a key within 7 hours. However, none of these early proposals were ever implemented—or, at least, no implementations were publicly acknowledged. The vulnerability of DES was practically demonstrated in the late 1990s. In 1997, RSA Security sponsored a series of contests, offering a $10,000 prize to the first team that broke a message encrypted with DES for the contest. That contest was won by the DESCHALL Project, led by Rocke Verser, Matt Curtin, and Justin Dolske, using idle cycles of thousands of computers across the Internet. The feasibility of cracking DES quickly was demonstrated in 1998 when a custom DES-cracker was built by the Electronic Frontier Foundation (EFF), a cyberspace civil rights group, at the cost of approximately US$250,000 (see EFF DES cracker). Their motivation was to show that DES was breakable in practice as well as in theory: "*There are many people who will not believe a truth until they can see it with their own eyes. Showing them a physical machine that can crack DES in a few days is the only way to convince some people that they really cannot trust their security to DES.*" The machine brute-forced a key in a little more than 2 days search. The next confirmed DES cracker was the COPACOBANA machine built in 2006 by teams of the Universities of Bochum and Kiel, both in Germany. One machine can be built for approximately $10,000.



1. Японский специалист Мицуру Мацуи (Mitsuru Matsui), изобретатель линейного криптоанализа, в 1993 году доказал возможность вычисления ключа шифрования DES методом линейного криптоанализа при наличии у атакующего 247 пар «открытый текст – зашифрованный текст2» (атака подробно описана, например, в [4] и [11]).
2. Криптологи из Израиля – изобретатели дифференциального криптоанализа3 Эли Бихам (Eli Biham) и Эди Шамир (Adi Shamir) – в 1991 году представили атаку, в которой ключ шифрования вычисляется методом дифференциального криптоанализа при наличии у атакующего возможности генерации 247 специально выбранных пар «открытый текст – зашифрованный текст4» [17].
3. В 1994 году Эли Бихам и Алекс Бирюков (Alex Biryukov) усилили известный с 1987 года метод вычисления ключа DES – метод Дэвиса (Davies), основанный на специфических свойствах таблиц замен DES [13]. Усиленный метод позволяет вычислить 6 бит ключа DES (остальные 50 бит – полным перебором возможных вариантов) при наличии 250 пар известных открытых текстов и шифртекстов или вычислить 24 бита ключа при наличии 252 пар.
В дальнейшем эти атаки были несколько усилены (например, атака линейным криптоанализом при наличии 243 пар вместо 247 [39]), появлялись также новые виды атак на DES (например, атака, позволяющая вычислить ключ высокоточным облучением аппаратного шифратора и последующим анализом ошибок шифрования [18]). Однако стоит сказать, что все эти атаки требуют наличия огромного количества пар «открытый текст – шифртекст», получение которых на практике является настолько трудоемкой операцией, что наиболее простой атакой на DES считается полный перебор возможных вариантов ключа шифрования [39].

Кроме того, практически сразу после появления DES были обнаружены следующие проблемы с ключами шифрования DES [4]:

1. 4 ключа из возможных 256 ключей алгоритма являются слабыми (т. е. не обеспечивают требуемой стойкости при зашифровании). Это ключи, в которых все биты какой-либо из половин расширяемого ключа (C или D на рис. 1) являются нулевыми или единичными. В этом случае все ключи раунда будут одинаковыми.
2. 6 пар ключей являются эквивалентными (т. е. информация, зашифрованная одним ключом из пары, расшифровывается другим ключом той же пары), например, пара ключей E0FEE0FEF1FEF1FE165 и FEE0FEE0FEE1FEE116. Процедура расширения такого ключа вместо 16 различных ключей раунда вырабатывает всего 2.
3. 48 ключей являются «возможно слабыми», их полный список приведен в [4]. Возможно слабые ключи при их расширении дают только 4 различных ключа раунда, каждый из которых используется при шифровании по 4 раза.
4. Существует свойство комплементарности ключей: если

Ek(M) = C,

где Ek(M) – зашифрование алгоритмом DES блока открытого текста M, а C – полученный в результате шифртекст, то

Ek'(M') = C',

где x' – побитовое дополнение к x (т. е. величина, полученная путем замены всех битовых нулей значения x на единицы, и наоборот).

DESX

Метод DESX создан Рональдом Ривестом и формально продемонстрирована Killian и Rogaway. Этод метод — усиленный вариант DES. DESX отличается от DES тем, что каждый бит входного открытого текста DESX логически суммируется по модулю 2 с 64 битами дополнительного ключа, а затем шифруется по алгоритму DES. Каждый бит результата также логически суммируется по модулю 2 с другими 64 битами ключа.

Таким образом, длина ключа увеличивается до

$56 + 2 \times 64 = 184$ бит.

Главной причиной использования DESX является простой в вычислительном смысле способ значительного повысить стойкость DES к атакам полного перебора ключа.

Скорость алгоритма DESX приблизительно равна скорости DES. DESX достаточно стоек, имеет аппаратную реализацию и широко используется.

Реализация DESX включена в криптографические библиотеки BSAFE компании RSA Security с конца 80х годов.

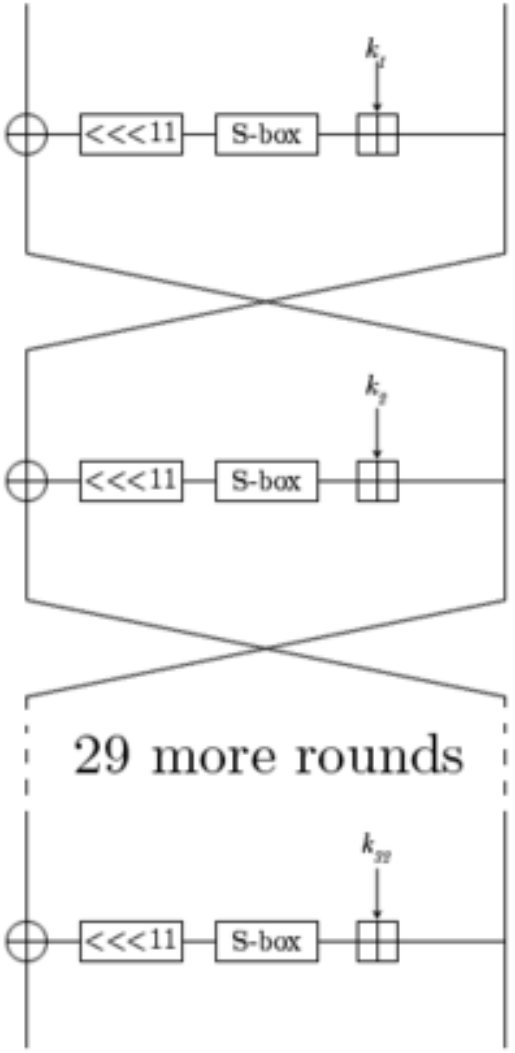
Отечественный стандарт шифрования ГОСТ 28147-89

«ГОСТ 28147-89 Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования»

ГОСТ 28147-89 - это стандарт, принятый в 1989 году в Советском Союзе и установивший алгоритм шифрования данных, составляющих гостайну. По свидетельству причастных к его реализациям и использованию людей, алгоритм был разработан в 70-е годы в 8-м Главном Управлении КГБ СССР, тогда он имел гриф Сов.Секретно. Затем гриф был понижен до Секретно, а когда в 89-м году алгоритм был проведен через Госстандарт и стал официальным государственным стандартом, гриф с него был снят. В начале 90-х годов он стал полностью открытым.

- ГОСТ 28147-89
- -Шифр Фейстеля.

- 1)размер шифруемого блока равен 64 битам;
- 2)размер ключа равен 256 битам;
- 3)число шагов цикла n = 32;
- 4) S –блоки не фиксированы.



- 4)получение цикловых ключей
- Ключ *К длины 256 битов* разбивается на блоки длиной 32 бит:
- $K = P1\ P2\ \ldots\ P8\ ,\ |Pi| = 32,\ i = 1,\ldots,8.$
- При шифровании используется ключевая последовательность:
- (К1,*К*2,...,*К*31,*К*32)=*(Р*1,*Р*2,...,*Р*8, *Р*1, *Р*2,...,*Р*8, *Р*1,*Р*2,...,*Р*8, *Р*8,*Р*7,...,*Р*1),

Функция усложнения F:

$$w_1 = r \oplus k \bmod 2^{32}$$

$$w_2 = f_s(w_1)$$

$$w_3 = T^{11}(w_2)$$

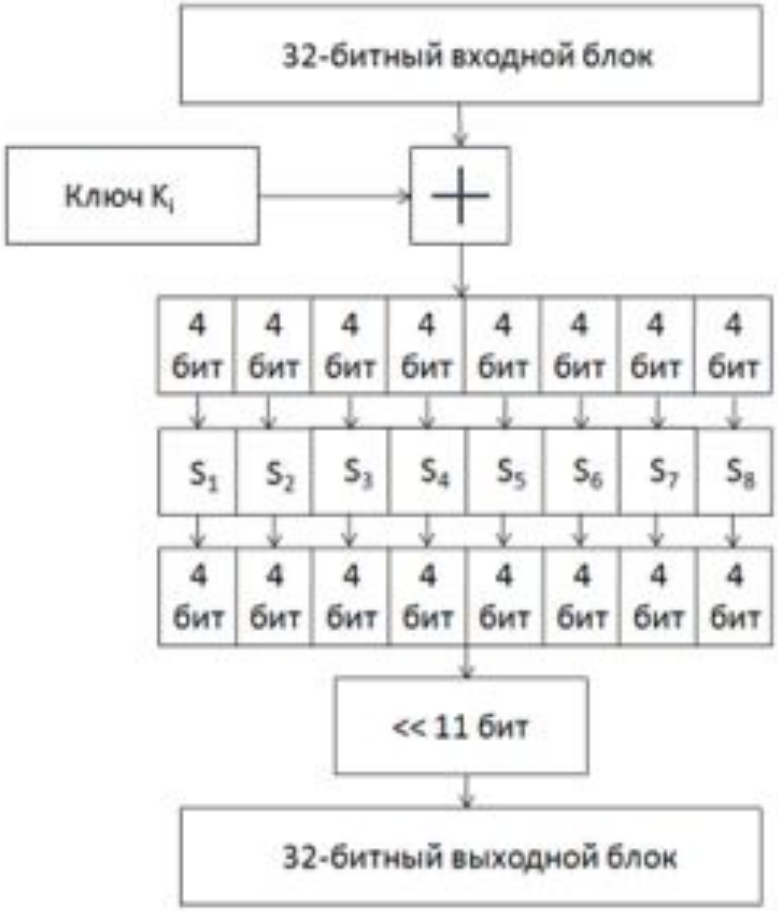


Таблица замен в ГОСТе - аналог S-блоков DES'а **- представляет собой таблицу (матрицу) размером 8x16, содержащую число от 0 до 15. В каждой строке каждое из 16-ти чисел должно встретиться ровно 1 раз. Таблица замен в ГОСТе одна и та же для всех раундов и, (В отличие от DES'а,) не зафиксирована в стандарте. От качества этой таблицы зависит качество шифра. При "сильной" таблице замен стойкость шифра не опускается ниже некоторого допустимого предела даже в случае ее разглашения. И наоборот, использование "слабой" таблицы может уменьшить стойкость шифра до недопустимо низкого предела. Никакой информации по качеству таблицы замен в открытой печати России не публиковалось, однако существование "слабых" таблиц не вызывает сомнения - примером может служить "тривиальная" таблица замен, по которой каждое значение заменяется на него самого. Это делает ненужным для компетентных органов России ограничивать длину ключа - можно просто поставить недостаточно "сильную" таблицу замен.**

В настоящее время известны *S-boxes*, которые используются в приложениях Центрального Банка РФ и считаются достаточно сильными. Напомню, что входом и выходом *S-box* являются 4-битные числа, поэтому каждый *S-box* может быть представлен в виде строки чисел от 0 до 15, расположенных в некотором порядке. Тогда порядковый номер числа будет являться входным значением *S-box*, а само число - выходным значением *S-box*.

0123456789ABCDEF
1 4A92D80E6B1C7F53
2 EB4C6DFA23810759
3 581DA342EFC7609B
4 7DA1089FE46CB253
5 6C715FD84A9E03B2
6 4BA0721D36859CFE
7 DB413F590AE7682C
8 1FD057A4923E6B8C

Данный набор S-блоков используется в криптографических приложениях ЦБ РФ.

Криптоанализ

В мае 2011 года известный криптоаналитик *Николя Куртуа* заявил об обнаружении серьезных уязвимостей в данном шифре.

Критика ГОСТа

Основные проблемы ГОСТа связаны с неполнотой стандарта в части генерации ключей и таблиц замен. Тривиально доказывается, что у ГОСТа существуют «слабые» ключи и таблицы замен, но в стандарте не описываются критерии выбора и отсева «слабых». Также стандарт не специфицирует алгоритм генерации таблицы замен (S-блоков). С одной стороны, это может являться дополнительной секретной информацией (помимо ключа), а с другой, поднимает ряд проблем:

нельзя определить криптостойкость алгоритма, не зная заранее таблицы замен;

реализации алгоритма от различных производителей могут использовать разные таблицы замен и могут быть несовместимы между собой;

возможность преднамеренного предоставления слабых таблиц замен лицензирующими органами РФ;

потенциальная возможность (отсутствие запрета в стандарте) использования таблиц замены, в которых узлы не являются перестановками, что может привести к чрезвычайному снижению стойкости шифра.

Сравнение ГОСТ и DES

П А Р А М Е Т Р	ГОСТ	DES
1. Размер блока шифрования	64 бита	64 бита
2. Длина ключа	256 бит	56 бит
3. Число раундов	32	16
4. Узлы замен (S-блоки)	не фиксированы	фиксированы
5. Длина ключа для одного раунда	32 бита	48 бит
6. Схема выработки раундового ключа	простая	сложная
7. Начальная и конечная перестановки битов	нет	есть

в DES сложная генерация подключей из ключей, а в ГОСТ простая.
в DES 56-битный ключ, а ГОСТ 256-битный + секретные S-блоки.
в DES 16 раундов, а в ГОСТ - 32 раунда.
в DES используются нерегулярные перестановки (P-блоки), а в ГОСТ 11-битный циклический сдвиг.

DES использует гораздо более сложную процедуру создания *подключей*, чем *ГОСТ 28147*. В *ГОСТ* эта процедура очень проста.
В DES применяется 56-битный ключ, а в *ГОСТ 28147* - 256-битный. При выборе сильных *S-boxes ГОСТ 28147* считается очень стойким.
У *S-boxes* DES 6-битные входы и 4-битные выходы, а у *S-boxes ГОСТ 28147* 4-битные входы и выходы. В обоих алгоритмах используется по восемь *S-boxes*, но размер *S-box ГОСТ 28147* существенно меньше размера *S-box* DES.
В DES применяются нерегулярные перестановки P, в *ГОСТ 28147* используется 11-битный циклический сдвиг влево. Перестановка DES увеличивает лавинный эффект. В *ГОСТ 28147* изменение одного входного бита влияет на один *S-box* одного *раунда*, который затем влияет на два *S-boxes* следующего *раунда*, три *S-boxes* следующего и т. д. В *ГОСТ 28147* требуется 8 *раундов* прежде, чем изменение одного входного бита повлияет на каждый бит результата; DES для этого нужно только 5 *раундов*.
В DES 16 *раундов*, в *ГОСТ 28147* - 32 *раунда*, что делает его более стойким к дифференциальному и линейному криптоанализу.

ГОСТ *не запантентован,* поэтому его может свободно использовать любое юридическое и физическое лицо, если, конечно, это не противоречит законодательству страны где находятся это лицо. Со стороны авторов ГОСТа претензий нет и быть не может, так как юридические права на алгоритм ни за кем не закреплены.

Слабые ключи DES и ГОСТ

- Слабыми ключами называется ключи *k* такие, что
- Ek Ek(x)=x
- , где *x* — 64-битный блок.

Алгоритм IDEA

International Data Encryption Algorithm

IDEA является одним из нескольких симметричных криптографических алгоритмов, которыми первоначально предполагалось заменить DES.
*Я*вляется блочным симметричным алгоритмом шифрования, разработанным Сюдзя Лай (Xuejia Lai) и Джеймсом Массей (James Massey) из швейцарского федерального института технологий. Первоначальная версия была опубликована в 1990 году.

Алгоритм IDEA

International Data Encryption Algorithm

IDEA является одним из нескольких симметричных криптографических алгоритмов, которыми первоначально предполагалось заменить DES.
*Я*вляется блочным симметричным алгоритмом шифрования, разработанным Сюдзя Лай (Xuejia Lai) и Джеймсом Массей (James Massey) из швейцарского федерального института технологий. Первоначальная версия была опубликована в 1990 году.
Структура шифра была разработана для легкого воплощения как программно, так и аппаратно.
Длина блока: длина блока должна быть достаточной, чтобы скрыть все статистические характеристики исходного сообщения. С другой стороны, сложность реализации криптографической функции возрастает экспоненциально в соответствии с размером блока. Использование блока размером в 64 бита в 90-е годы означало достаточную силу.
Длина ключа: длина ключа должна быть достаточно большой для того, чтобы предотвратить возможность простого перебора ключа. При длине ключа 128 бит *IDEA* считается достаточно безопасным.
Конфузия: зашифрованный текст должен зависеть от ключа сложным и запутанным способом.
Диффузия: каждый бит незашифрованного текста должен влиять на каждый бит зашифрованного текста. Распространение одного незашифрованного бита на большое количество зашифрованных бит скрывает статистическую структуру незашифрованного текста. Определить, как статистические характеристики зашифрованного текста зависят от статистических характеристик незашифрованного текста, должно быть непросто. *IDEA* с этой точки зрения является очень эффективным алгоритмом.

Частично основан на структуре FEAL

1)размер блока 64 бита;
2) размер ключа 128 битов;
3) число раундов 8:
4)выходное отображение, обеспечивающее обратимость;
5) входного отображения нет;

Безопасность IDEA основывается на использовании трех не совместимх типов арифметических операций над 16-битными словами.

В ходе алгоритма выполняются следующие операции:

1.XOR сложение (+)

2.сложение по модулю 2^{16} [+]

3.умножение по модулю $2^{16}+1$ (X) .

$$x \cdot (y + z) = x \cdot y + x \cdot z$$

$$x \otimes (y [+] z) \neq x \otimes y [+] x \otimes z$$
$$x [+] (y \oplus z) \neq x [+] y \oplus x [+] z$$
$$x \otimes (y \oplus z) \neq x \otimes y \oplus x \otimes z$$

Дистрибути́вность (от лат. *distributivus* — «распределительный») — свойство согласованности двух бинарных операций, определённых на одном и том же множестве.

Ключевое расписание

Ключ K 128 бит.
Получаем 52 раундовых ключа – 6 на каждый их 8 раундов, 4- на выходное отображение;
Ключ K разбивается на 8 подблоков длины 16 бит.
Обозначаем
K(1),K2(1),K3(1),K4(1),K5(1),K6(1),K1(2),K2(2).
Затем Начальный ключ K циклически сдвигается влево на 25 бит и снова разбивается на 8 подблоков:
K(2),K4(2),K5(2),K6(2),K1(3),K2(3),K3(3),K4(3).
Затем снова сдвигается на 25 битов и снова разбивается, и так до тех пор, пока не сгенерируются 52 подблока.
1 цикл - K1(1),K2(1),K3(1),K4(1),K5(1),K6(1),
2 цикл - K1(2),K2(2),K3(2),K4(2),K5(2),K6(2),
3 цикл - K1(3),K2(3),K3(3),K4(3),K5(3),K6(3),
8 цикл - K1(8),K2(8),K3(8),K4(8),K5(8),K6(8),
выходное отображение - K1(9),K2(9),K3(9),K4(9).

На каждом *раунде* используются только 96 бит *подключа*, множество бит ключа на каждой итерации не пересекаются, и не существует отношения простого сдвига между *подключами* разных *раундов*. Это происходит потому, что на каждом *раунде* используется только шесть *подключей*, в то время как при каждой ротации ключа получается восемь *подключей*.

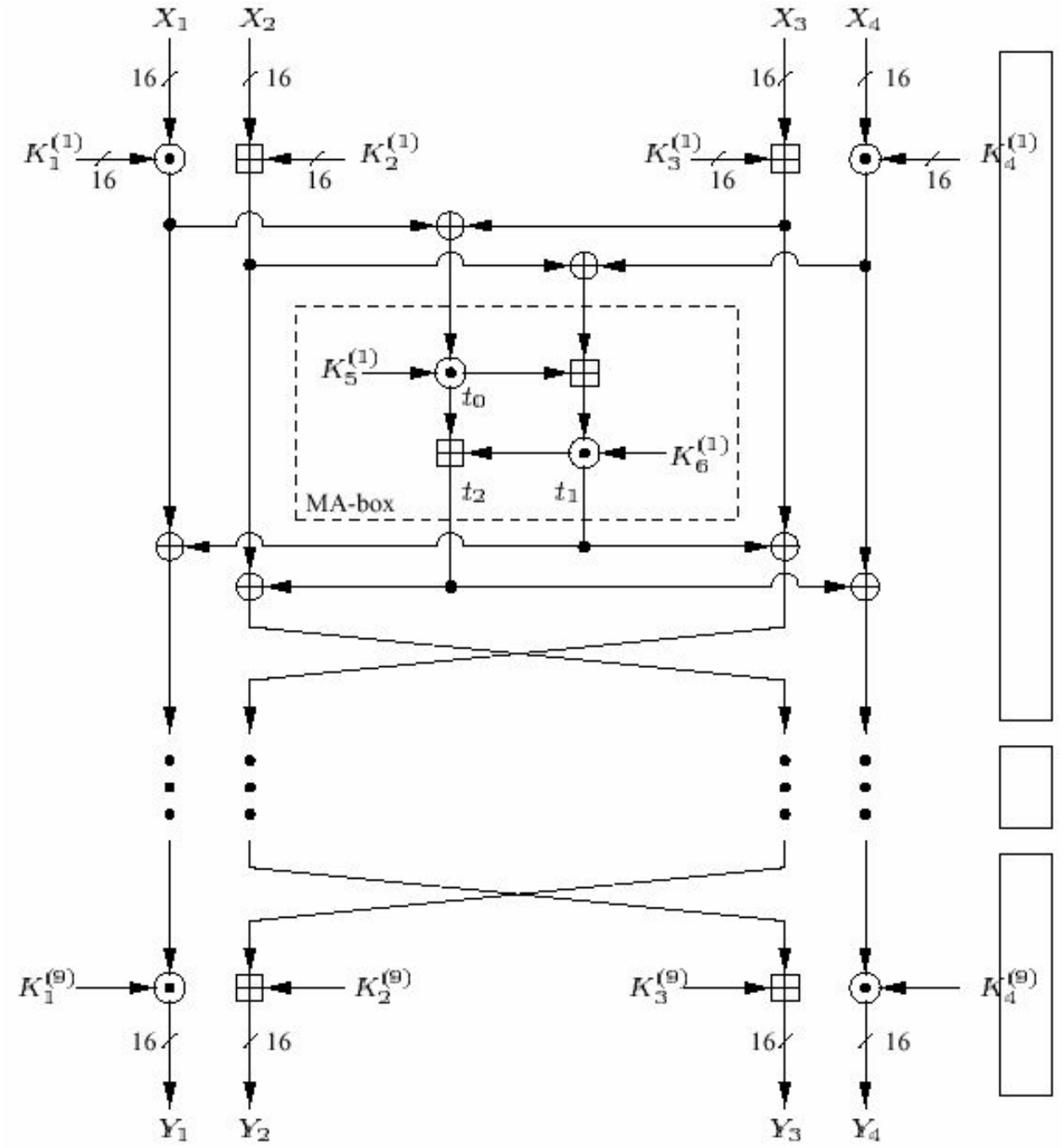


Схема алгоритма
Открытый текст x1(16 бит), x2(16 бит), x3(16 бит), x4(16 бит).
Шаг 1. c1=x1{*}k1
Шаг 2. c2=x2[+]k2
Шаг 3. c3=x3[+]k3
Шаг 4. c4=x4{*}k4
Шаг 5. c5=c1(+)c3
Шаг 6. c6=c2(+)c4
Шаг 7. c7=c5{*}k5
Шаг 8. c8=c6[+]c7
Шаг 9. c9=c8{*}k6
Шаг 10. c10=c9[+]c7
Шаг 11. c11=c1(+)c9
Шаг 12. c12=c3(+)c9
Шаг 13. c13=c2(+)c10
Шаг 14. c14=c4(+)c10
Входные блоки следующего цикла – c11,c12,c13,c14.

Выходное отображение

После 8 цикла реализуется выходное отображение
Шаг 1. y1=x1(9){*}k1(9)
Шаг 2. y2=x2(9)[+]k2(9)
Шаг 3. y3=x3(9)[+]k3(9)
Шаг 4. y4=x4(9){*}k4(9)
Шифрованным текстом является блок y1,y2,y3,y4.

Расшифровка

Метод вычисления, использующийся для расшифровки текста по существу такой же, как и при его шифровании. Единственное отличие состоит в том, что для расшифровки используются другие подключи. В процессе расшифровки подключи должны использоваться в обратном порядке. Первый и четвёртый подключи i-го раунда расшифровки получаются из первого и четвёртого подключа (10-i)-го раунда шифрования мультипликативной инверсией. Для 1-го и 9-го раундов второй и третий подключи расшифровки получаются из второго и третьего подключей 9-го и 1-го раундов шифрования аддитивной инверсией. Для раундов со 2-го по 8-й второй и третий подключи расшифровки получаются из третьего и второго подключей с 8-го по 2-й раундов шифрования аддитивной инверсией. Последние два подключа i-го раунда расшифровки равны последним двум подключам (9-i)-го раунда шифрования. Мультипликативная инверсия подключа K обозначается 1/K и . Так как —2^(16)+1 простое число, каждое целое не равное нулю K имеет уникальную мультипликативную инверсию по модулю . Аддитивная инверсия подключа K обозначается -K.

Аппаратная реализация

Аппаратная реализация имеет перед программной следующие преимущества:

существенное повышение скорости шифрования за счёт использования параллелизма при выполнении операций

меньшее энергопотребление

Развитие аппаратных реализаций IDEA

1998программная 23,53**Мб/сек** Limpaa

2000программная 44**Мб/сек** Limpaa

1992ASIC1992ASIC 1,5 мкм КМОП44**Мб/сек** Bonnenberg и др.

1994ASIC1994ASIC 1,2 мкм КМОП177**Мб/сек** Curiger, Zimmermann и др.

1995ASIC1995ASIC 0,8 мкм КМОП355**Мб/сек** Wolter и др.

1998ASIC1998ASIC 0,7 мкм КМОП424**Мб/сек** Salomao и др.

19984 x XC4020XL 528**Мб/сек**Mencer и др.

1999ASIC1999ASIC 0,25 мкм КМОП 720 **Мб/сек**Ascom

2000Xilinx Virtex XCV300-6 1166**Мб/сек** Leong и др.

2000ASIC2000ASIC 0,25 мкм КМОП 1013**Мб/сек**Goldstein и др.

В 2002 годуВ 2002 году была опубликована работа о реализации IDEA на ПЛИС все той же фирмы Xilinx семейства Virtex-E. Устройство XCV1000E-6BG560 при частоте 105,9 МГц достигает скорости шифрования 6,78 Гб/сек.

Брюс Шнайер отозвался об IDEA так: «Мне кажется, это самый лучший и надежный блочный алгоритм, опубликованный до настоящего времени».

Существуют успешные атаки, применимые к IDEA с меньшим числом раундов (полный IDEA имеет 8.5 раундов). Успешной считается атака, если вскрытие шифра с её помощью требует меньшего количества операций, чем при полном переборе ключей. Метод вскрытия Вилли Майера (*Willi Meier*) оказался эффективнее вскрытия полным перебором ключей только для IDEA с 2 раундами . Лучшая атака на 2007 год применима ко всем ключам и может взломать IDEA с 6-ю раундами

Сравнение с некоторыми блочными алгоритмами

лгоритм	Размер ключа, бит	Длина блока, бит	Число раундов	Скорость шифрования наIntel486SX/ 33МГц (Кбайт/с)
DES	56	64	16	35
IDEA	128	64	8	70
Blowfish	32-448	64	16	135
ГОСТ 28147-89	256	64	32	53

Преимущества и недостатки IDEA

Преимущества

- В программной реализации на Intel486SX по сравнению с DES IDEA в два раза быстрее, что является существенным повышением скорости,
- длина ключа у IDEA имеет размер 128 бит, против 56 бит у DES, что является хорошим улучшением против полного перебора ключей.
- Вероятность использования слабых ключей очень мала и составляет 2^(-64).
- IDEA быстрее алгоритма ГОСТ 28147-89 (в программной реализации на Intel486SX).
- Использование IDEA в параллельных режимах шифрования на процессорах Pentium III иPentium MMX позволяет получать высокие скорости. По сравнению с финалистами AES, 4-way IDEA лишь слегка медленнее, чем RC6 иRijndael на Pentium II, но быстрее, чем Twofish и MARS. На Pentium III 4-way IDEA даже быстрее RC6 и Rijndael.
- Преимуществом также является хорошая изученность и устойчивость к общеизвестным средствам криптоанализа.

Недостатки

- IDEA значительно медленнее, почти в два раза, чем Blowfish (в программной реализации на Intel486SX).
- Существенным недостатком является то, что IDEA запатентован, так как это препятствует его свободному распространению.
- IDEA не предусматривает увеличение длины ключа.
- Недостатком можно также считать тот факт, что не все работы по криптоанализу были опубликованы, то есть вполне возможно, что шифр взломан, или будет взломан в будущем.

Применение IDEA

Алгоритм IDEA является торговой маркой и запатентован в Австрии, Франции, Германии, Италии, Нидерландах, Испании, Швеции, Швейцарии, Англии (Европейский патент EP-B-0482154), Соединенных штатах Америки (U.S. Patent 5 214 703, выпущен 25 Мая, 1993) и в Японии (JP 3225440). Действие патентаАлгоритм IDEA является торговой маркой и запатентован в Австрии, Франции, Германии, Италии, Нидерландах, Испании, Швеции, Швейцарии, Англии (Европейский патент EP-B-0482154), Соединенных штатах Америки (U.S. Patent 5 214 703, выпущен 25 Мая, 1993) и в Японии (JP 3225440). Действие патента истекает в 2010—2011 годах. Сегодня лицензия принадлежит компании MediaCrypt и позволяет свободно использовать алгоритм в некоммерческих приложениях.

Типичные области применения IDEA:

шифрование аудиошифрование аудио и видеошифрование аудио и видео данных для кабельного телевиденияшифрование аудио и видео данных для кабельного телевидения, видеоконференцийшифрование аудио и видео данных для кабельного телевидения, видеоконференций, дистанционного обученияшифрование аудио и видео данных для кабельного телевидения, видеоконференций, дистанционного обучения, VoIP

защита коммерческой и финансовойзащита коммерческой и финансовой информации, отражающей конъюнктурные колебания

линии связилинии связи через модемлинии связи через модем, роутерлинии связи через модем, роутер или ATMлинии связи через модем, роутер или ATM линию, GSMлинии связи через модем, роутер или ATM линию, GSM технологию смарт-карты

Недостатки DES

Double DES

Наиболее логичным способом противодействия полному перебору ключа DES выглядит многократное зашифрование данных алгоритмом DES с различными ключами. Следующий алгоритм получил название Double DES – двойной DES:

C = Ek2/2(Ek1/2(M)),

где k1/2 и k2/2 – половины двойного ключа алгоритма Double DES, каждая из которых представляет собой обычный 56-битный ключ DES, а E – функция зашифрования блока данных обычным DES-ом.

Если бы при двойном шифровании DES выполнялось следующее свойство:

C = Ek2/2(Ek1/2(M)) = Ek(M),

для любых значений k1/2 и k2/2, то двойное шифрование не приводило бы к усилению против полного перебора ключа – всегда нашелся бы такой ключ k, однократное зашифрование которым было бы эквивалентно двукратному шифрованию на ключах k1/2 и k2/2, а для нахождения ключа k достаточно было бы перебрать 255 ключей. К счастью, DES не обладает таким свойством, что доказано в [20], поэтому Double DES действительно удваивает эффективный размер ключа – до 112 бит, а при современном развитии вычислительной техники полный перебор 112-битного ключа невозможен.

Атака «встреча посередине»

предложена Ральфом Мерклем (Ralph Merkle) и Мартином Хеллманом . С помощью этой атаки криптоаналитик может получить k1/2 и k2/2 при наличии всего двух пар открытого текста и шифртекста (M1 – C1 и M2 – C2) следующим образом:

Шаг 1. Выполняется зашифрование Ekx(M1) на всем ключевом пространстве с записью результатов в некоторую таблицу.

Шаг 2. Производится расшифрование Dky(C1) также на всем ключевом пространстве; результаты расшифрования сравниваются со всеми записями в таблице, сформированной на шаге 1.

Шаг 3. Если какой-либо результат, полученный на шаге 2, совпал с одним из результатов шага 1, то можно предположить, что нужный ключ найден, т. е. соответствующие совпадающему резальтату kx = k1/2, а ky = k2/2. Однако таких совпадений может быть много, их количество оценивается в [1] как 248.

Шаг 4. Для отсечения «ложных» ключей необходимо повторить предыдущие шаги с парой M2 – C2, сузив пространство перебора только до вариантов, приводящих к совпадениям (т. е. примерно 248). Вероятность наличия более чем одного совпадения после повторного перебора оценивается в [1] как 2–16.

Такая атака, выполняющая, фактически, перебор половинок двойного ключа, как со стороны открытого текста, так и со стороны шифртекста, требует примерно в 2 раза больше вычислений, чем перебор обычного ключа DES, однако, требует также много памяти для хранения промежуточных результатов. Тем не менее, атака является реально осуществимой на практике, поэтому алгоритм Double DES не используется. Используется Triple DES

Тройной DES

Достаточно медленно работает, так что некоторые приложения не могут работать по этим алгоритмом.

Так как текст, зашифрованный двойным DES оказывается не стойким при криптографической атаке то текст шифруется 3 раза DES. Таким образом длина ключа возрастает до 168-бит (56x3).

Типы тройного шифрования DES:

DES-EEE3: Шифруется 3 раза с 3 различными ключами.

DES-EDE3: 3 DES операции шифровка-расшифровка-шифровка с 3 различными ключами.

DES-EEE2 и DES-EDE2: Как и предыдущие, за исключением того, что первая и третья операции используют одинаковый ключ.

Тройной DES является достаточно популярной альтернативой *DES* и используется при управлении ключами в стандартах ANSI X9.17 и ISO 8732 и в PEM (Privacy Enhanced Mail).

Известных криптографических атак на *тройной DES* не существует. Цена подбора ключа в *тройном DES* равна 2^{112}.

AES

Недостатки DES

В 80-х годах в США был принят стандарт симметричного криптоалгоритма для внутреннего применения DES (Data Encryption Standard).

В настоящее время DES устарел по двум причинам :

1) Малая длина его ключа (56 бит).
2) Алгоритм ориентирован на аппаратную реализацию, то есть содержит операции, выполняемые на микропроцессорах за неприемлимо большое время (например, такие как перестановка бит внутри машинного слова по определенной схеме).

Требования, предъявленные к кандидатам на AES в 1998 году

NIST – National Institute of Standards & Technology объявил конкурс на новый стандарт симметричного криптоалгоритма

стандарт блочных шифров США с 2000 года

- алгоритм должен быть симметричным,
- алгоритм должен быть блочным шифром,
- алгоритм должен иметь длину блока 128 бит, и поддерживать три длины ключа : 128, 192 и 256 бит.

Дополнительные рекомендации

- использовать операции, легко реализуемые как аппаратно, так и программно,
- ориентироваться на 32-разрядные процессоры,
- не усложнять без необходимости структуру шифра

На первом этапе в оргкомитет соревнования поступило 15 заявок из совершенно разных уголков мира. В течение 2 лет специалисты комитета, исследуя самостоятельно, и изучая публикации других исследователей, выбрали 5 лучших представителей, прошедших в "финал" соревнования.
Полное описание всех 15 алгоритмов претендентов на AES, включая исследования по их криптостойкости представлены на сервере института NIST.

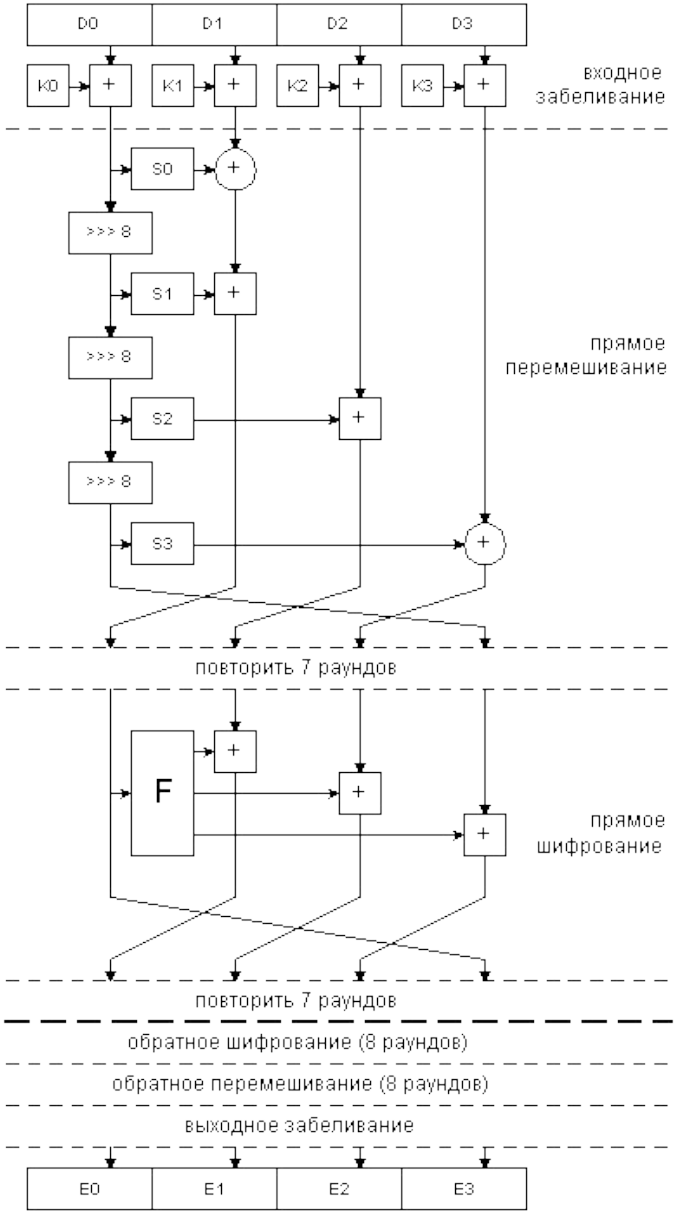
Алгоритм	**Автор**	**Страна**	**Быстродействие (asm, 200МГц)**
MARS	IBM	US	8 Мбайт/с
RC6	R.Rivest & Co	US	12 Мбайт/с
Rijndael	V.Rijmen & J.Daemen	BE	7 Мбайт/с
Serpent	*Universities*	IS, UK, NO	2 Мбайт/с
TwoFish	B.Schneier & Co	US	11 Мбайт/с

Все эти алгоритмы были признаны достаточно стойкими и успешно противостоящими всем широко известным методам криптоанализа.

Алгоритм	Автор	Страна	Быстродействие (asm, 200МГц)
MARS	IBM	US	8 Мбайт/с
RC6	R.Rivest & Co	US	12 Мбайт/с
Rijndael	V.Rijmen & J.Daemen	BE	7 Мбайт/с
Serpent	*Univ*		
TwoFish	B.Sc & Co		c

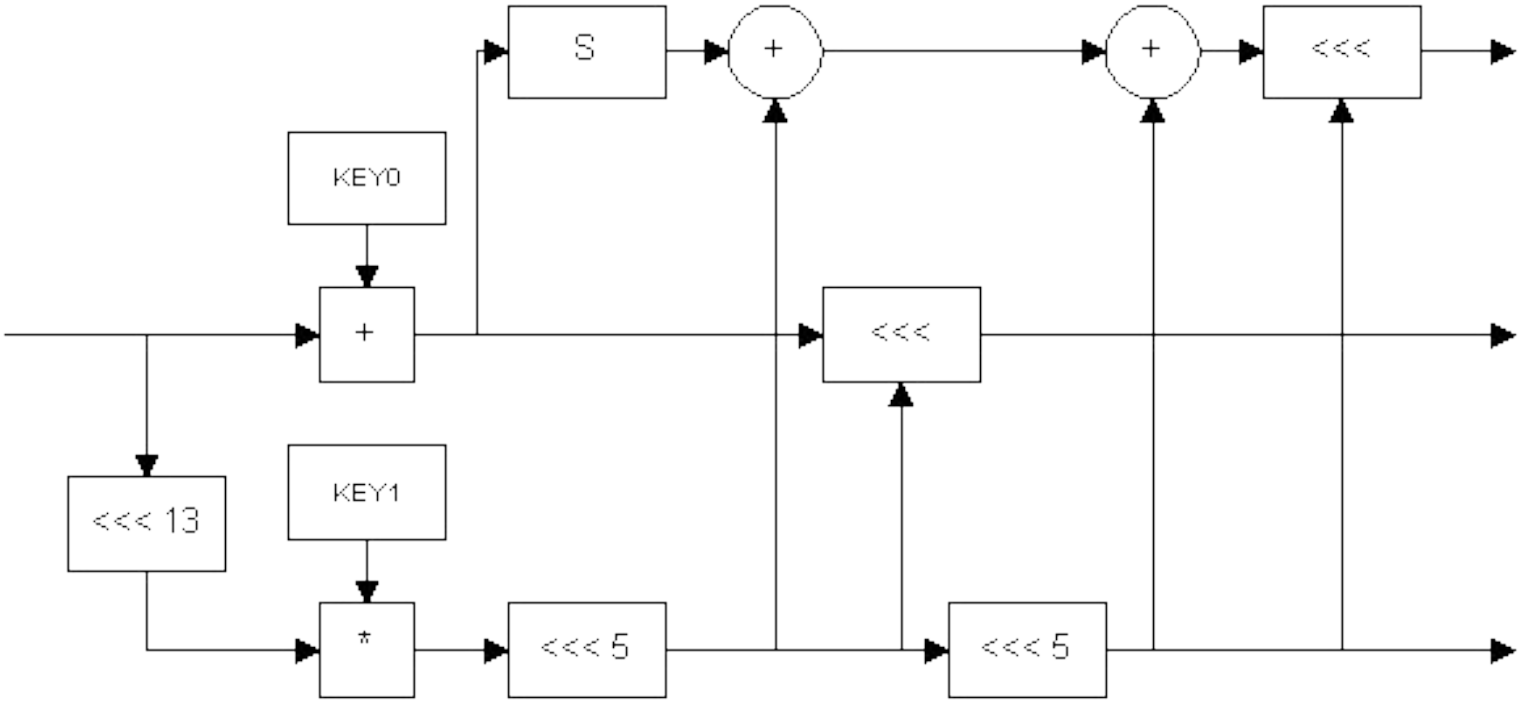
2 октября 2000 года NIST объявил о своем выборе – победителем конкурса стал бельгийский алгоритм RIJNDAEL. С этого момента с алгоритма-победителя сняты все патентные ограничения – его можно будет использовать в любой криптопрограмме без отчисления каких-либо средств создателю.

Финалист AES – шифр MARS



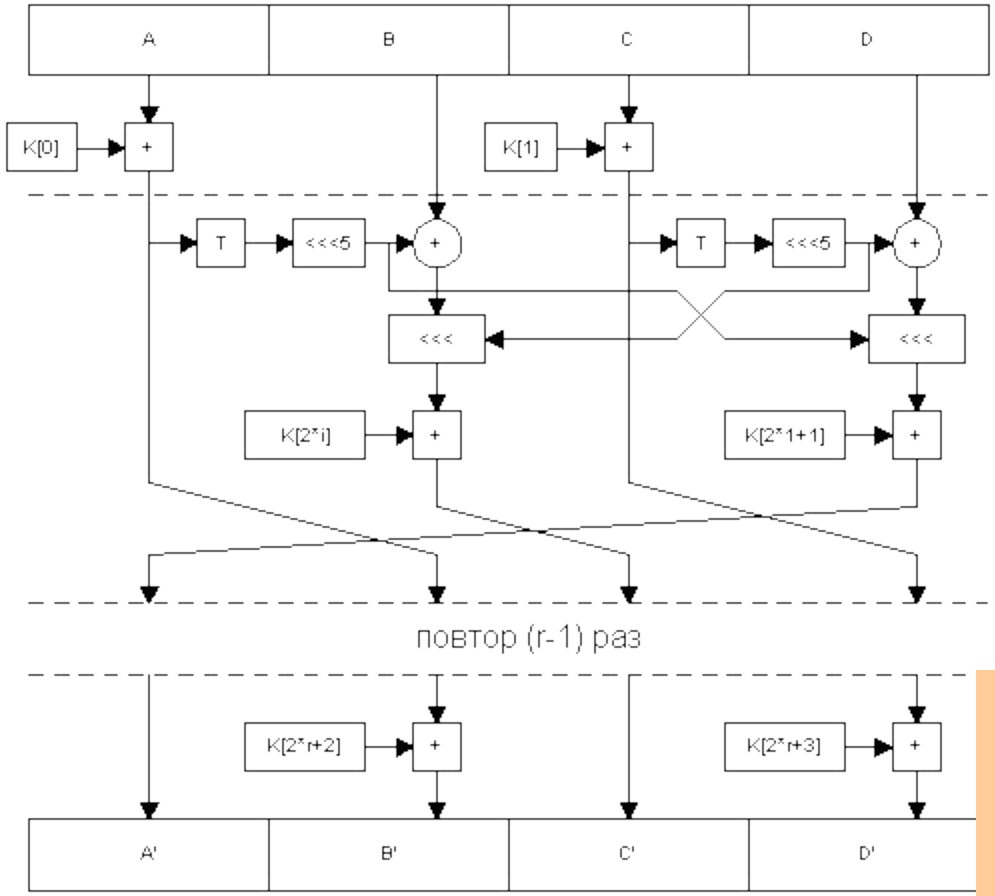
1 -входное забеливание : ко всем байтам исходного текста добавляются байты из материала ключа.
2 - прямое перемешивание, сеть Фейстеля (8 раундов) без добавления ключа.
3 - сеть Фейстеля треьего типа с 4 ветвями (8 раундов).
Затем повторяются те же операции, но в обратном порядке : сначала шифрование, перемешивание, забеливание. При этом во вторые варианты всех операций внесены некоторые изменения таким образом, чтобы криптоалгоритм в целом стал абсолютно симметричным. То есть, в алгоритме MARS для любого X выполняется выражение EnCrypt(EnCrypt(X))=X

Функция F:



В алгоритме MARS использованы практически все виды операций, применяемых в криптографических преобразованиях : сложение, "исключающее ИЛИ", сдвиг на фиксированное число бит, сдвиг на переменное число бит, умножение и табличные подстановки.

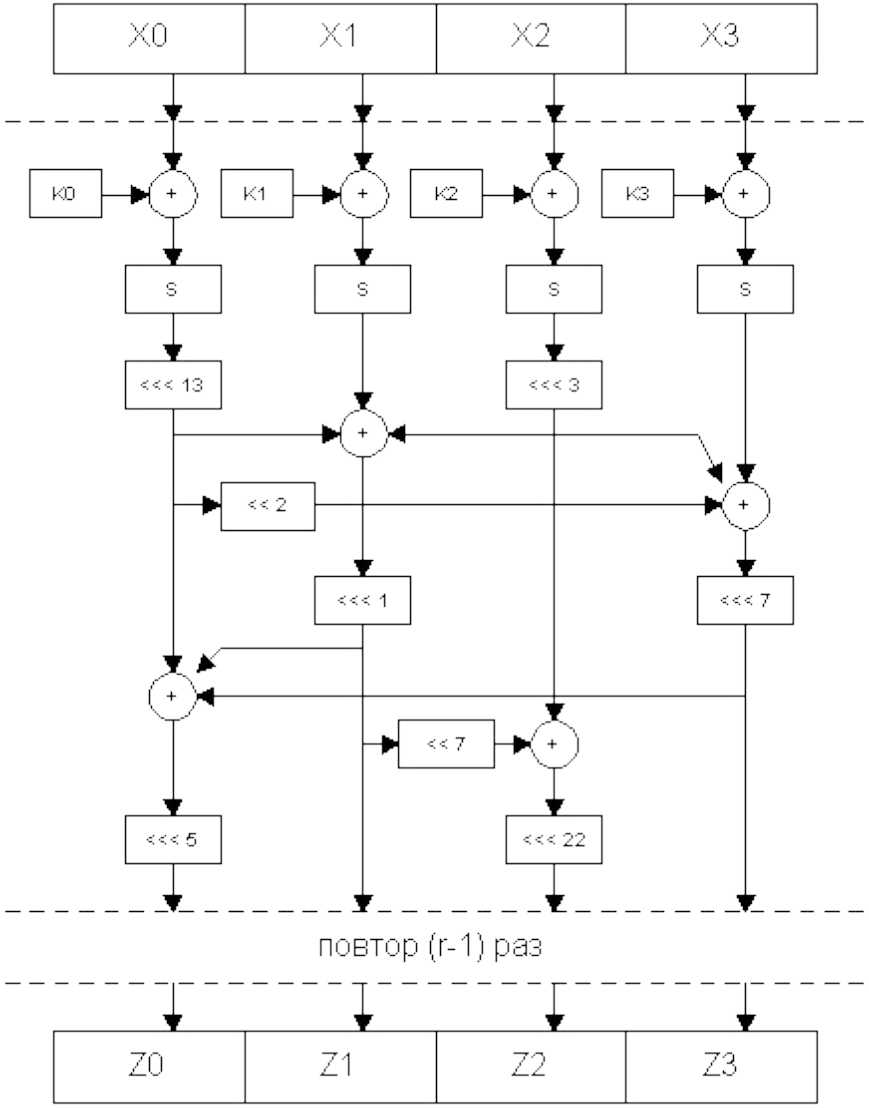
Финалист AES – шифр RC6



- продолжение RC5, разработанного Рональдом Ривестом. RC5 был незначительно изменен для того, чтобы соответствовать требованиям AES по длине ключа и размеру блока. При этом алгоритм стал еще быстрее, а его ядро, унаследованное от RC5, имеет солидный запас исследований, проведенных задолго до объявления конкурса AES.

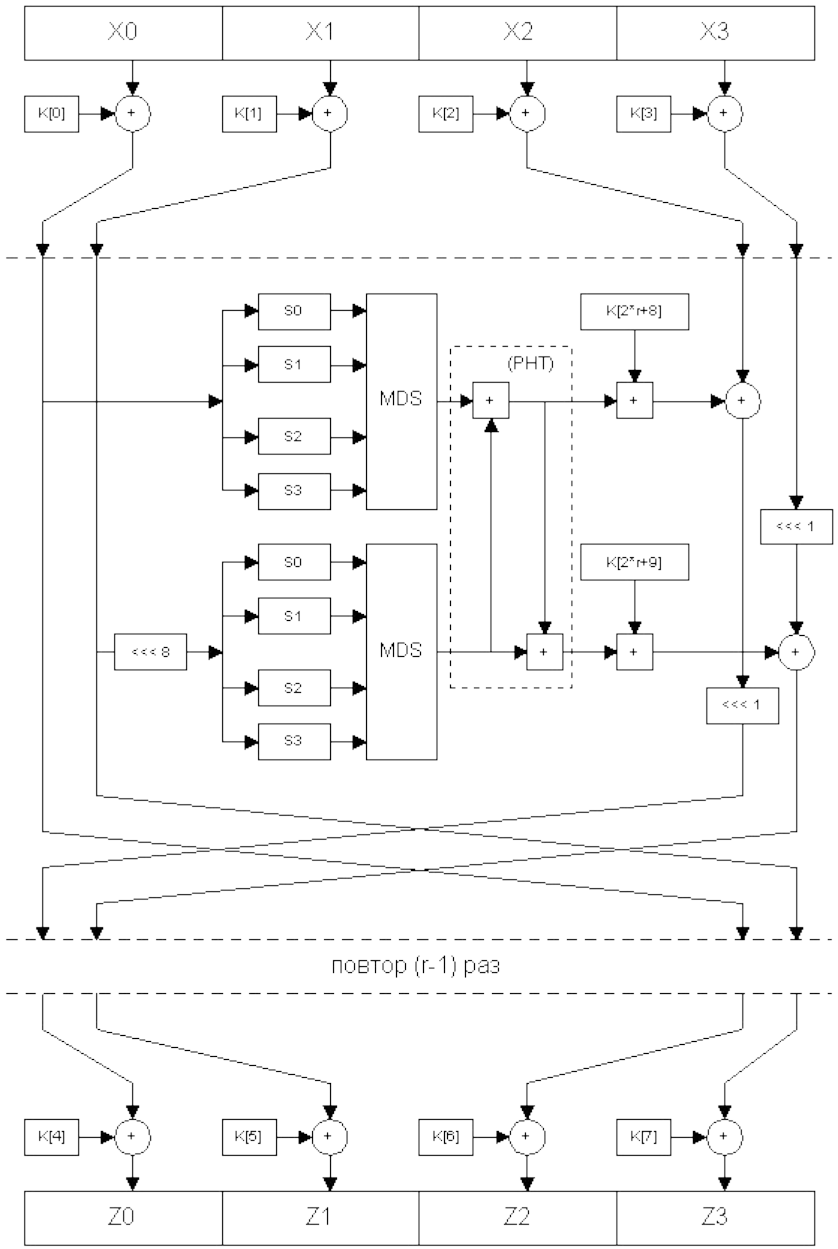
сеть Фейштеля с 4 ветвями. Разработчики рекомендуют при шифровании использовать 20 раундов сети, хотя в принципе их количество не регламентируется. При 20 повторах операции шифрования алгоритм имеет самую высокую скорость среди 5 финалистов AES.

Финалист AES – шифр Serpent



Алгоритм разработан группой ученых из нескольких исследовательских центров мира. Алгоритм представляет собой сеть Фейстеля для четырех ветвей.
Используются только исключающее "ИЛИ", табличные подстановки и битовые сдвиги. Алгоритм состоит из 32 раундов.

Финалист AES – шифр TwoFish



Алгоритм разработан комманией Counterpain Security Systems, возглавляемой Брюсом Шнайером. Предыдущая программная разработка этой фирмы, называвшаяся BlowFish, до сих пор является признанным криптостойким алгоритмом
Сеть Фейштеля.
Единственным нарицанием, поступившим в адрес TwoFish от независимых исследователей, является тот факт, что при расширении материала ключа в алгоритме используется сам же алгоритм. Двойное применение блочного шифра довольно сильно усложняет его анализ на предмет **наличия слабых ключей или недокументированных замаскированных связей между входными и выходными данными**.

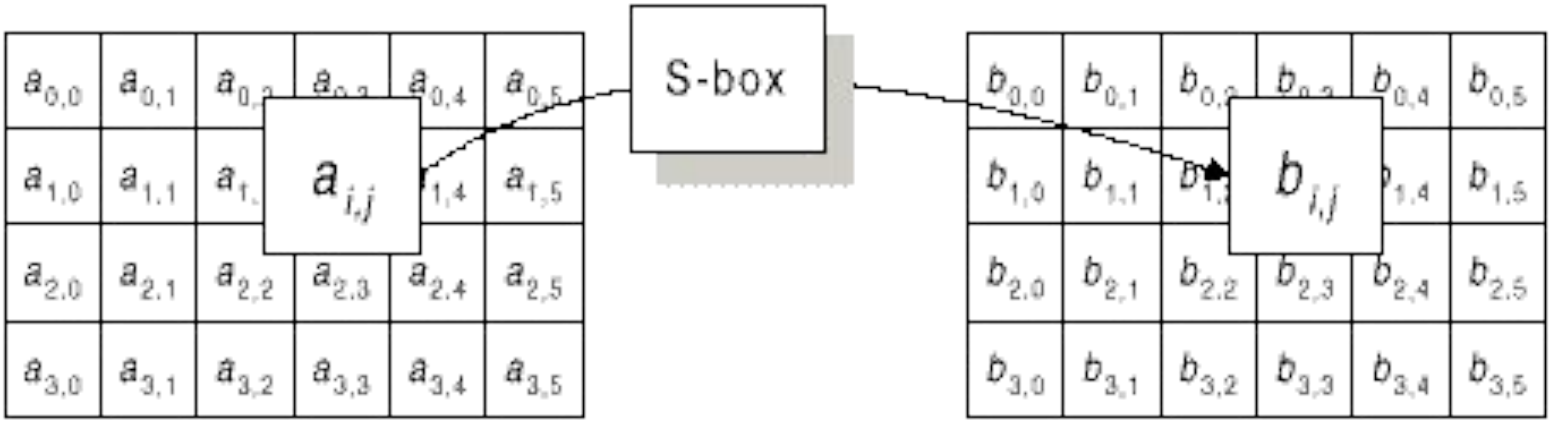
Победитель AES – шифр Rijndael

Не использует сеть Фейстеля для криптопреобразований. Алгоритм представляет каждый блок кодируемых данных в виде двумерного массива байт размером 4x4, 4x6 или 4x8 в зависимости от установленной длины блока. Далее на соответствующих этапах преобразования производятся либо над независимыми столбцами, либо над независимыми строками, либо вообще над отдельными байтами в таблице.

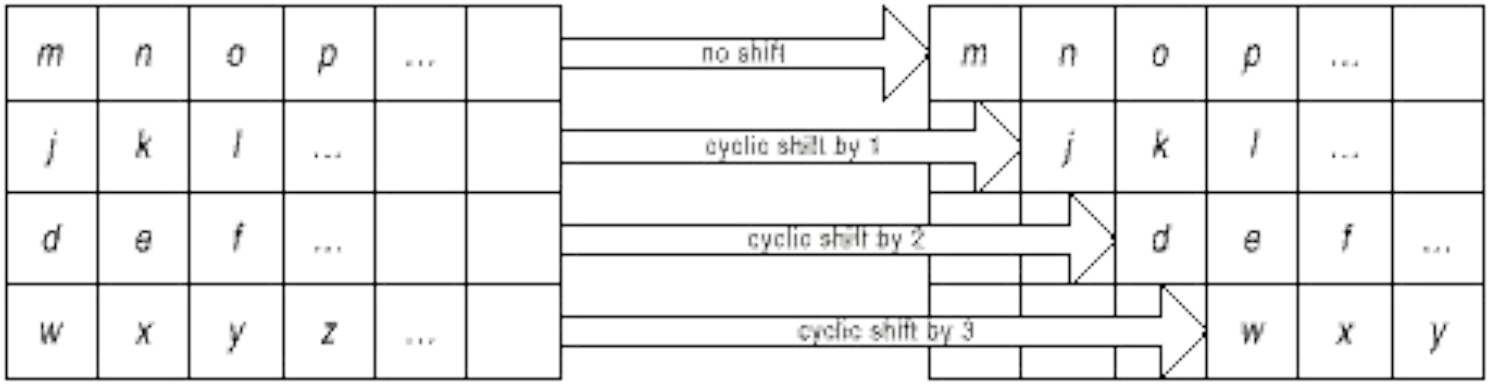
Все преобразования в шифре имеют строгое математическое обоснование. Сама структура и последовательность операций позволяют выполнять данный алгоритм эффективно как на 8-битных так и на 32-битных процессорах. В структуре алгоритма заложена возможность параллельного исполнения некоторых операций, что на многопроцессорных рабочих станциях может еще поднять скорость шифрования в 4 раза.

Алгоритм состоит из некоторого количества раундов (от 10 до 14 – это зависит от размера блока и длины ключа), в которых последовательно выполняются следующие операции :

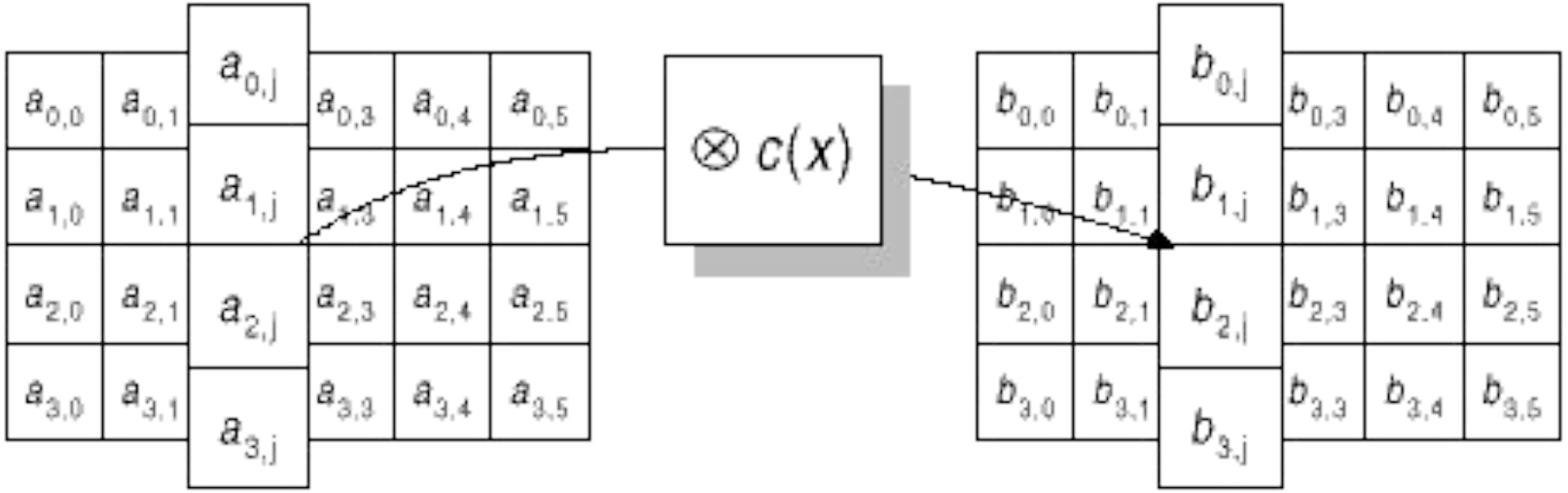
- 1. ByteSub – табличная подстановка 8x8 бит



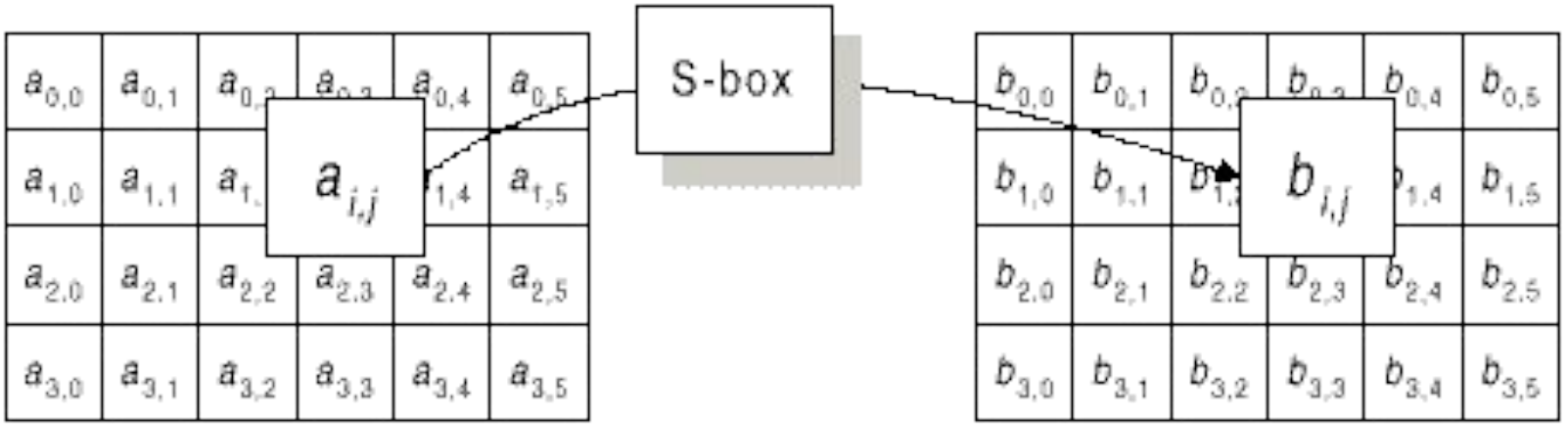
2. ShiftRow – сдвиг строк в двумерном массиве на различные смещения



3. MixColumn – перемешивание данных внутри столбца



AddRoundKey – добавление материала ключа операцией XOR



В последнем раунде операция перемешивания столбцов отсутствует, что делает всю последовательность операций симметричной.

Криптоанализ AES

На презентации алгоритма разработчики продемонстрировали атаку на 6 раундов алгоритма и заявили о необходимости 10-14 раундов шифрования (в зависимости от размера ключа). В процессе отбора кандидатов механизм атак был усовершенствован до 7 раундов для 128-битных ключей, 8 раундов для 196-битных ключей и 9 раундов для 256-битных ключей. В результате остается от 3 до 5 раундов на обеспечение безопасности. Однако даже возможное развитие данных атак на 100% шифра потребовало бы 2**120 шагов и 2**100 байт памяти. Подобную атаку в настоящее время невозможно осуществить на практике и ориентировочно в ближайшие 50 лет.

Криптоанализ Serpent

Наиболее консервативный из всех участников конкурса. Полностью ориентирован на обеспечение безопасности. Наилучшая из атак способна взломать 10 из 32 раундов. Главный недостаток – скорость (в 3 раза медленнее AES, примерно равная DES). Иначе он с большой вероятностью занял бы 1 место благодаря своей консервативности.

Криптоанализ Twofish

Почти такой же быстрый, как AES, но с большим запасом прочности. Наилучшая атака – 8 раундов из 16. Недостаток – длительность смены ключа шифрования.

Криптоанализ RC6

Успешная атака -17 из 20 раундов.

Криптоанализ MARS

Ошибка в коде программы привела к тому, что сгенерированная S-матрица не удовлетворяла условиям, выдвинутым самими разработчиками. В аргументах, приводимых авторами как доказательство устойчивости к линейному криптоанализу, обнаружились серьезные упущения.