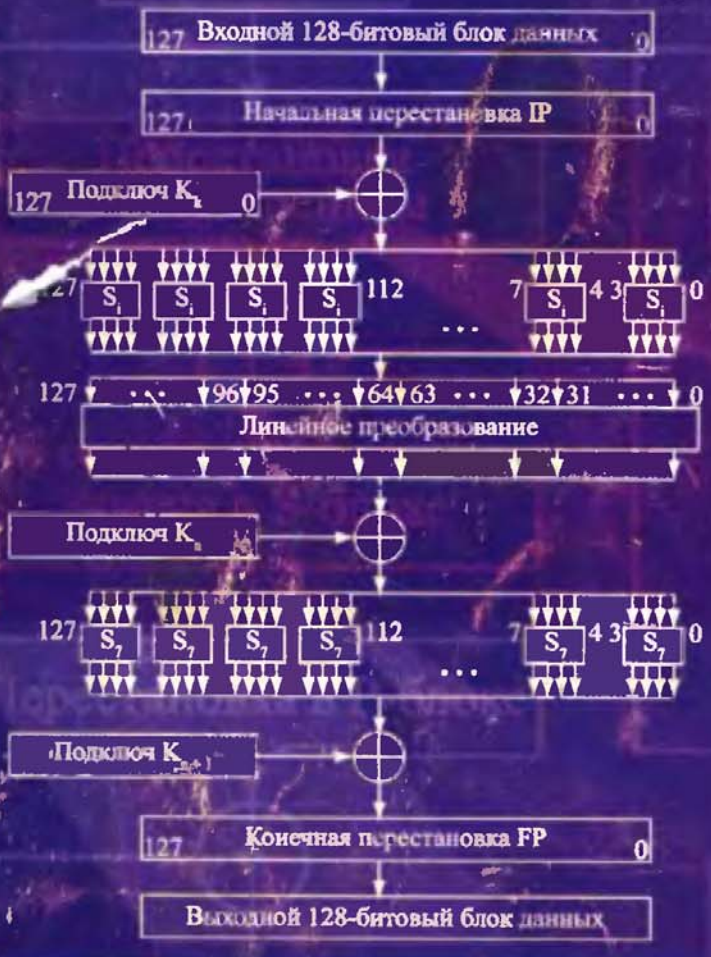
Л. К. Бабенко, Е. А. Ищукова

Современные алгоритмы блочного шифрования и методы их анализа



Гелиос РВ

Л. К. Бабенко, Е. А. Ищукова

СОВРЕМЕННЫЕ АЛГОРИТМЫ БЛОЧНОГО ШИФРОВАНИЯ И МЕТОДЫ ИХ АНАЛИЗА

Рекомендовано Учебно-методическим объединением вузов Российской Федерации в качестве учебного пособия для студентов высших учебных заведений, обучающихся по специальностям 090103 «Организация и технология защиты информации», 090104 «Комплексная защита объектов информатизации»

Москва
«Гелиос АРВ»
2006

УДК 004.056.55
ББК 32.973.2-018
Б12

Бабенко, Людмила Климентьевна.

Б12 Современные алгоритмы блочного шифрования и методы их анализа: учеб. пособие для студентов вузов, обучающихся по группе специальностей в обл. информ. безопасности / Л. К. Бабенко, Е. А. Ищукова. — М.: Гелиос АРВ, 2006. — 376 с., ил. ISBN 5-85438-149-4

I. Ищукова, Е. А.

Посвящено алгоритмам блочного шифрования: принципам их построения и анализа. Рассматриваются действующие стандарты, а также многие другие общеизвестные криптографические алгоритмы, в том числе и финалисты конкурса AES. Излагаются способы проведения атак на эти алгоритмы с помощью таких методов, как линейный и дифференциальный криптоанализ. Описан подход к применению нового метода криптоанализа — линейно-дифференциального. Представлены виды криптоанализа на основе слайдовой атаки. В приложениях приведены таблицы с результатами анализа наиболее известных алгоритмов шифрования. Большая часть учебного пособия посвящена практическим вопросам изучения атак: приведено пять лабораторных работ по описанным методам криптоанализа, которые представлены на сайте кафедры БИТ ТРТУ http://bit.tsure.ru, а также целый ряд задач для самостоятельного решения.

Для студентов, аспирантов и начинающих криптографов и криптоаналитиков.

ББК 32.973.2-018

ISBN 5-85438-149-4

Введение

С каждым годом компьютеры все прочнее и прочнее входят в наш мир. Современный человек уже не может представить свою жизнь без них. Как следствие, все острее встает вопрос защиты обрабатываемой информации. Если несколько десятилетий назад вопросы защиты информации интересовали в основном военные и государственные структуры, то сейчас каждый, владеющий компьютером, стремится обезопасить свои данные. Поэтому с каждым годом все больше и больше требуются квалифицированные специалисты, способные обеспечить сохранность информации при ее передаче по каналам связи.

Одним из самых эффективных средств защиты является шифрование информации. Наряду с ростом числа новых криптографических алгоритмов в последние годы стремительно развивается и криптоанализ, изучающий процессы воссоздания открытого текста или ключа или и того и другого из зашифрованного текста. Серьезный вклад в развитие криптоанализа положила работа Э. Бихама и А. Шамира [8], в которой они представили новый метод анализа алгоритма шифрования DES, названный дифференциальным анализом. Позднее появилась целая серия работ других авторов по усовершенствованию этого метода применительно к различным алгоритмам шифрования, а также были предложены другие методы, такие, как, например метод линейного криптоанализа, разработанный японским ученым М. Матсуи [6], метод слайдовой атаки, представленный А. Бирюковым и Д. Вагнером [21]. Ежегодно проводятся международные конференции, на которых авторы разных стран представляют как исследования уже существующих алгоритмов шифрования и методов их анализа, так и разработки новых алгоритмов и методов криптоанализа. К сожалению, большинство этой информации остается в стороне от российских ученых.

В нашей стране пока еще сравнительно мало учебников и пособий для подготовки будущих криптоаналитиков, которые бы в доступной форме описывали основы криптоанализа. Авторы данного учебного пособия считают своей целью дать подробное описание некоторых современных методов криптоанализа применительно к алгоритмам блочного шифрования, а также представить доступный практикум по закреплению изученного материала.

Однако невозможно начать изучение существующих методов криптоанализа, не ознакомившись с самим объектом анализа, не изучив детально основы его построения и работы. Именно поэтому первая часть нашего пособия посвящена изучению современных алгоритмов блочного шифрования. Здесь вы можете ознакомиться с основными принципами разработки алгоритмов блочного шифрования, найти достаточно подробное описание бывших и действующих стандартов шифрования, других известных криптографических алгоритмов, в том числе и финалистов конкурса AES.

Далее полезно перейти к изучению таких методов криптоанализа, как линейный, дифференциальный, линейно-дифференциальный, криптоанализ на основе использования слайдовой атаки.

Для закрепления изученного материала в конце каждой главы приводятся контрольные вопросы, а в конце пособия можно найти задачи, предназначенные для самостоятельного решения, которые помогут более детально понять изученный материал.

Помимо этого, на сайте кафедры БИТ Таганрогского государственного радиотехнического университета http://bit.tsure.ru представлены все описанные в книге лабораторные работы, выполнив которые, вы сможете убедиться в действенности представленных методов анализа.

Авторы желают удачи и надеются, что данное пособие поможет в освоении новой специальности! Все замечания и пожелания можно прислать на адрес blk@fib.tsure.ru.

Глава 1. ОБЗОР И КЛАССИФИКАЦИЯ СОВРЕМЕННЫХ МЕТОДОВ КРИПТОАНАЛИЗА

1.1. Основные типы криптоанализа

На сегодняшний день существует множество видов криптоанализа, каждый из которых зависит, в наибольшей степени, от имеющейся у криптоаналитика информации. Ниже приведены основные из них, а также указаны их отличительные особенности и свойства [2].

Криптоанализ на основе только известного шифртекста

Это наиболее мощная криптоаналитическая атака, так как для ее осуществления криптоаналитику требуется только пассивное прослушивание с целью получения шифртекстов. При этом имеются минимальные сведения об открытых текстах. В таких условиях один из возможных подходов криптоанализа заключается в простом переборе всех возможных вариантов ключей. Однако если пространство возможных ключей очень велико, этот подход становится нереальным. Поэтому криптоаналитику приходится больше полагаться на анализ самого шифрованного текста, что, как правило, означает выявление его различных статистических особенностей. Криптоаналитик должен иметь некоторые общие предположения о содержимом открытого текста. Например, он может знать, на каком языке был написан зашифрованный открытый текст или что это исполняемый файл определенной операционной системы, или что это исходный код программы на каком-либо языке программирования, и т. п. Алгоритмы шифрования, которые поддаются атаке такого типа, являются наглядным примером неправильного построения шифров и пригодны лишь для обучения будущих криптоаналитиков.

Криптоанализ на основе известного открытого текста

Этот вид криптоанализа основывается на том, что у атакующего есть какая-то часть подвергшихся зашифрованию данных. Целью является либо извлечение из этой части известных текстов секретного ключа, либо прочтение неизвестной части зашифрованных данных. Такой вид криптоанализа до сих пор широко используется, так как трудно запретить противнику предугадывать содержимое открытых текстов (раньше этот метод назывался «Методом возможных слов» (*probable word method*)) [41]. Так, например, в файле с открытым текстом может присутствовать стандартный заголовок или идентификатор. Владея подобной информацией, криптоаналитик может восстановить ключ шифрования на основе логических умозаключений и знания того, каким образом был преобразован известный открытый текст в шифртекст.

Криптоанализ на основе выбранного открытого текста

В этом случае предполагается, что у противника есть возможность зашифровывать выбранные им открытые тексты. На практике это может быть возможно в случае, когда к противнику в руки попал реализованный алгоритм шифрования с неизвестным секретным ключом или когда есть возможность послать выбранный открытый текст владельцу секретного ключа, а затем получить эти данные в зашифрованном виде при передаче их третьей стороне.

В общем случае, если криптоаналитик имеет возможность по своему усмотрению выбрать сообщение и шифровать его, то при правильном выборе сообщений для шифрования он может вполне оправданно надеяться разгадать ключ.

Криптоанализ на основе выбранного шифртекста

Этот вид криптоанализа аналогичен предыдущему за тем исключением, что требуется выбрать шифртексты для данной схемы дешифрования.

Естественно, что если алгоритм может быть взломан при анализе только шифрованного текста, то он является слабым. Поэтому обычно любой алгоритм шифрования разрабатывается так, чтобы он был устойчив к попыткам взлома с помощью анализа с известным открытым текстом, то есть так, чтобы при известном открытом тексте и соответствующем ему шифрованном тексте было достаточно трудно, а еще лучше — невозможно, определить секретный ключ шифрования.

Основной характеристикой криптографической стойкости шифра относительно того или иного метода анализа является трудоемкость E(r) этого метода. В качестве меры трудоемкости раскрытия шифров обычно используется количество элементарных операций, необходимых для дешифрования сообщения, или определения ключа. Под элементарной операцией понимают операцию, выполняемую на конкретной аппаратуре за один шаг ее работы. Трудоемкость дешифрования определяется объемом и характером информации, доступной криптоаналитику [4].

Алгоритм шифрования называется **безусловно защищенным**, или **абсолютно стойким**, в том случае, если шифртекст, полученный с помощью данного алгоритма шифрования, не содержит достаточной информации для однозначного восстановления соответствующего открытого текста, при этом объем шифрованного текста не играет никакой роли.

Это означает, что независимо от того, сколько времени потратит противник на расшифровку, ему не удастся расшифровать шифрованный текст просто потому, что в шифрованном тексте нет информации, требуемой для восстановления открытого текста [2]. Среди алгоритмов шифрования абсолютно стойких нет. Таким образом, максимум, чего может ожидать пользователь от того или иного алгоритма шифрования, это выполнения хотя бы одного из двух следующих критериев защищенности:

1. Стоимость взлома шифра превышает стоимость расшифрованной информации;
2. Время, которое требуется для того, чтобы взломать шифр, превышает время, в течение которого информация актуальна.

Алгоритм шифрования называется **защищенным по вычислениям**, если он соответствует обоим вышеуказанным требованиям [2].

Проблема заключается в том, что количественно оценить усилия, необходимые для криптоанализа шифрованного текста, созданного с помощью конкретного данного алгоритма шифрования, очень сложно.

Практически все формы криптоанализа для алгоритмов блочного шифрования используют тот факт, что некоторые характерные особенности структуры открытого текста могут сохраняться при шифровании и проявляться в соответствующих особенностях структур шифрованного текста.

1.2. Обзор основных универсальных методов криптоанализа

Универсальные методы криптоанализа — это такие методы, которые с определенными вариациями могут быть применимы ко многим шифрам. Однако не все универсальные методы всегда применимы. Так как некоторые из них значительно снижают стойкость шифра, то создатели стараются делать так, чтобы универсальные методы были к их шифрам неприменимы. Вместе с тем, все множества X, Y и K конечны. Поэтому потенциально можно перебрать все их элементы и найти все решения уравнения $T(x, k) = y$, где $x \in X$, $y \in Y$, $k \in K$. Следовательно, множество методов, применимых к данной шифрсистеме, непусто.

1.2.1. Метод полного перебора

Наиболее часто упоминаемым универсальным методом является метод полного перебора, заключающийся в последовательном применении всех ключей дешифрования к дешифруемому тексту, и проверке — получился ли при дешифровании на данном ключе открытый текст. Более подробное описание применения этого метода можно найти во многих источниках литературы, например [4, 9, 10].

1.2.2. Анализ на основе использования словарей

Это другой простой, но в тоже время важный вид анализа блочных алгоритмов шифрования. Если размер блока в шифре сравнительно маленький, то атакующий может собрать много блоков и анализировать частоту различий в блоках. Примером шифра, уязвимого с помощью такой атаки, является шифр с использо-

ванием простых перестановок, где каждая буква меняется на другую букву алфавита.

Другим типом с использованием словарей является запись зашифрований одного алгоритма шифрования на всех возможных ключах в словарь. После этого надо ждать, когда поступят данные, зашифрованные с помощью этого алгоритма на секретном ключе. Секретный ключ находится простым просмотром словаря.

1.2.3. Парадокс «Дней Рождений»

Это очень важный вероятностный парадокс с не перечислимым количеством применений в современной криптографии: от алгоритмов блочного шифрования до систем с открытым ключом.

Рассмотрим предпосылки возникновения парадокса «Дней Рождений». Ответьте на вопрос: как много учеников должно собраться в одном классе, чтобы как минимум двое из них имели день рождения в один и тот же день? С помощью простых вычислений можно выяснить, что если в классе будет находиться 23 ученика, то вероятность того, что у двух из них день рождения приходится на один и тот же день, будет больше, чем 1/2.

Итак, пронумеруем учеников в классе от 1 до k и обозначим день рождения i-го ученика как d_i (которое может принимать значение от 1 до n, а n = 365). Учитывая, что дни рождения не зависят друг от друга, получаем, что вероятность того, что ученики i и j имеют одинаковый день рождения, равна:

$$\sum_{d=1}^{n} P(d_i = d) * (d_j = d) = n * \frac{1}{n^2} = \frac{1}{n}.$$

Таким образом, вероятность того, что у i и j-го учеников день рождения в один и тот же день равна вероятности того, что один из них родился в определенный день года. Теперь найдем вероятность того, что как минимум двое из k учеников родились в один день. Вероятность того, что все k учеников имеют разные дни рождения равна:

$$P_{\text{разные дни рождения}} = 1 * (1 - \frac{1}{n}) * (1 - \frac{2}{n}) L (1 - \frac{k-1}{n}).$$

Ясно, что для двух человек — обозначим их как 1 и 2 — вероятность того, что они родились в один день, равна $\frac{1}{n}$. А значит, вероятность того, что у них разные дни рождения, равна $1 - \frac{1}{n}$. Так как известна вероятность того, что двое имеют разные дни рождения, рассматривая трех учеников, скажем 1, 2 и 3, можно определить вероятность того, что ученик 3 родился с учениками 1 и 2 в разные дни. Эта вероятность равна $1 - \frac{2}{n}$. Таким образом, увеличивая число учеников до k,

получим, что вероятность того, что ученик k не рожден в один день с учениками 1, 2, …, k – 1, равна $1 - \frac{k-1}{n}$.

В том случае, когда вероятность того, что все k учеников родились в разные дни, ниже 1/2, получается, что вероятность того, что как минимум двое из них родились в один день больше или равна 1/2. Пользуясь неравенством $(1 + n) \le e^n$, получаем:

$$P_{\text{разные дни рождения}} \le e^{-1/n} e^{-2/n} \dots e^{-(k-1)/n} = e^{-\frac{k(k-1)}{n}}.$$

Учитывая, что эта вероятность должна быть меньше 1/2, получаем:

$$k \ge \frac{1}{2} + \sqrt{\frac{1}{4} + 2n \log 2}.$$

Если n достаточно большое, то можно записать:

$$k \approx \sqrt{2n \log 2}.$$

Правомерен вопрос: как этот парадокс может быть применен в криптографии? Рассмотрим множество значений X таких, что $|X| = 2^n$, и случайную функцию F(x), определенную на элементах этого множества. Введем понятие коллизии функции F(x). Пара (x, y), где $x \in X$ и $y \in X$, называется коллизией функции F в том случае, если F(x) = F(y). Тогда возникает вопрос: сколько должно быть подмножеств A и B множества X, чтобы коллизия между ними встречалась с вероятностью больше, чем 0,5? Требуемое условие может быть выполнено тогда, когда $|A| \cdot |B| \approx |X|$, и, таким образом, в случае, если |A| = |B|, получается, что $|A| \approx 2^{n/2}$. Аналогичный результат можно получить, если A = B.

Вероятность того, что между подмножеством A и подмножеством B нет коллизии, может быть вычислена следующим образом:

$$\left(1 - \frac{|A|}{|B|}\right)^{|B|},$$

В случае, если $|A| << |X|$, можно считать, что вероятность равна:

$$e^{-\frac{|A| \cdot |B|}{|X|}}.$$

Теперь рассмотрим несколько практических применений парадокса «Дней Рождений» на практике. Предположим, что для атаки на алгоритм шифрования, оперирующий 64-битными блоками данных, противник должен получить две пары *открытый—закрытый текст*, которые отличаются только младшим значащим битом. Согласно парадоксу «Дней Рождений», только массив, состоящий примерно из 232 открытых текстов, будет содержать требуемые пары с высокой вероятностью. Для другого примера рассмотрим пример одного раунда 64-битно-

го шифра Фейстеля. Предположим, что в шифре используется произвольная функция F. Злоумышленник может захотеть узнать, как много открытых текстов ему нужно получить для того, чтобы на выходе встретилось два одинаковых шифртекста F-функции. Согласно парадоксу «Дней Рождений», можно определить, что в этом случае требуется только $O(2^{16})$ текстов.

1.2.4. Метод «Встреча посередине»

Метод «Встреча посередине» применим к алгоритмам шифрования, в которых используется два различных ключа K. Это может быть доятигнуто в том случае, если секретные подключи появляются с какой-то периодичночтью или, например если было произведено двойное зашифрование данных, то есть сначала данные зашифровали на одном ключе K_1, а затем полученный результат шифрования еще раз зашифровали на другом секретном ключе K_2.

Пусть нам известна пара *открытый—закрытый текст*, зашифрованная подобным образом. В этом случае, необходимо произвести зашифрование открытого текста на всех возможных значениях ключа K_1. Паралелльно с этим необходимо произвести дешифрование закрытого текста на всех возможных значениях ключа K_2. Та пара ключей (K_1, K_2), для которой результат шифрования открытого текста и результат дешифрования закрытого текста совпадут, и будет являться искомой.

В качестве примера рассмотрим алгоритм шифрования S_DES. Этот алгоритм как нельзя, кстати, нам подходит, потому как содержит всего два цикла шифрования. Таким образом, с помощью метода «Встреча посередине» мы сможем найти секретный ключ шифрования.

Итак, возьмем открытое восьмибитовое сообщение 00110110 и зашифруем его на секретном ключе 1100010001 (1000 в десятичной системе счисления). При этом опустим начальную и конечную перестановки для простоты анализа. Результатом шифрования будет значение 00111100. Естественно, нам известен секретный ключ. Однако мы попробуем его найти, основываясь только на известной паре *открытый—закрытый текст*. Итак, зашифровав сообщение 00110110 с помощью одного раунда алгоритма шифрования S_DES, на всех возможных значениях подключа (от 0 до 255) мы получим массив 1 соответствующих значений. Кроме того, произведем дешифрование значения 00111100 с помощью всех возможных значений подключа и получим 2-й массив аналогичных значений. Теперь каждое полученное значение массива 1 сравниваем с каждым значением массива 2 и в случае их совпадения заносим в массив 3 значение ключа K_1, с помощью которого было получено значение массива 1, и значение ключа K_2, с помощью которого было получено значение массива 2.

Так как алгоритм S_DES оперирует восьмибитовыми значениями, то совпавших пар окажется слишком много, для того чтобы сразу определить искомый секретный ключ. Для его определения обратимся к алгоритму выработки раундового

подключа из исходного ключа шифрования. Исходный секретный ключ подвергается перестановке, как это показано в таблице 1.1 (см. этап 2). На каждом этапе, кроме первого, в таблице в первой строке стоят числа, соответствующие исходным номерам битов секретного ключа, а во второй строке приведено их упорядочивание (для удобства работы).

Таблица 1.1

№ Этапа	Начальное состояние									
1	1	2	3	4	5	6	7	8	9	10
2	3	5	2	7	4	10	1	9	8	6
	1	2	3	4	5	6	7	8	9	10
3	5	2	7	4	3	1	9	8	6	10
	1	2	3	4	5	6	7	8	9	10
4	1	7	9	4	8	3	10	6	Первый подключ	
	1	2	3	4	5	6	7	8		
5	7	4	3	5	2	8	6	10	1	9
	1	2	3	4	5	6	7	8	9	10
6	8	3	6	5	10	2	9	1	Второй подключ	
	1	2	3	4	5	6	7	8		

После этого исходный преобразованный ключ разделяется на две части, каждая из которых циклически сдвигается влево на один бит. В таблице 1.1 это соответствует третьему этапу. Далее извлекается первый подключ (четвертый этап). Ключ, соответствующий третьему этапу, разбивается на две части, каждая из которых сдвигается влево на 2 позиции (пятый этап). И, наконец, на шестом этапе извлекается второй подключ. Теперь, если мы сопоставим четвертый и шестой этапы, мы увидим, что у первого и второго подключей совпадают шесть битов из восьми. Если нумеровать биты от 1 до 8 слева направо, то это будут следующие совпадающие пары:

— первый бит первого подключа и восьмой бит второго подключа;
— третий бит первого подключа и седьмой бит второго подключа;
— пятый бит первого подключа и первый бит второго подключа;
— шестой бит первого подключа и второй бит второго подключа;
— седьмой бит первого подключа и пятый бит второго подключа;
— восьмой бит первого подключа и третий бит второго подключа.

Зная это, мы можем удалить из массива 3 все значения, не удовлетворяющие определенному нами условию. У нас останется всего три пары подключей, одна из которых является истинной. Эти три пары подключей приведены в таблице 1.2.

Пара № 3 является истинной, то есть эти подключи соответствуют исходному секретному подключу 1100010001.

Таблица 1.2

Найденные пары подключей

№	Первый подключ K_1	Второй подключ K_2
1	00100100	01010010
2	01110100	01010010
3	11010100	01010101

1.2.5. Метод «Разделяй и побеждай»

Метод анализа «Разделяй и побеждай» основан на том, что множество ключей K допускает представление $K = K_1 \cdot K_2$. То есть каждый ключ $k \in K$ может быть записан в виде вектора $k = (k_1, k_2)$, где $k_1 \in K_1$ и $k_2 \in K_2$.

Также обязательным условием является существование некоторого критерия h, позволяющего при известных открытом и закрытом текстах проверять правильность первого подключа k_1 в ключе $k = (k_1, k_2)$.

Метод «Разделяй и побеждай» предлагает опробовать сначала элементы множества K_1, отбраковывая варианты k_1 по критерию h. Отсюда мы определим k_1 — первую компоненту вектора $k = (k_1, k_2)$. Затем, используя найденное значение k_1 и известный шифртекст, опробуем варианты k_2. Трудоемкость определения компоненты k_1 равна в среднем $\frac{|K_1|}{2}$ операций опробования, соответственно трудоемкость определения k_2 равна в среднем $\frac{|K_2|}{2}$ операций опробования. Тогда в среднем трудоемкость метода равна $\frac{|K_1|}{2}+\frac{|K_2|}{2}$ против $\frac{|K_1|*|K_2|}{2}$ при методе полного перебора [4].

Основная проблема при применении метода «Разделяй и побеждай» состоит в нахождении критерия h, который зависит от конкретного преобразования T или вообще может не существовать. Иногда критерий h может описываться вероятностно-статистической моделью.

1.3. Контрольные вопросы

1. Дайте определение криптоанализа.
2. Перечислите известные вам типы криптоанализа. При этом укажите их отличительные особенности и свойства.
3. Какой алгоритм шифрования называется безусловно защищенным?
4. Перечислите критерии защищенности алгоритма шифрования.

5. Какой алгоритм шифрования называется защищенным по вычислениям?
6. В каких целях применяются современные методы криптоанализа?
7. Что понимается под универсальными методами криптоанализа?
8. Какие универсальные методы криптоанализа вы знаете?
9. Чему равна средняя трудоемкость криптоаналитического метода полного перебора?
10. Что является коллизией функции F?
11. В чем заключается парадокс «Дней Рождений»?
12. Как парадокс «Дней Рождений» может быть применим в криптографии?
13. В чем заключается суть метода «Встречи посередине»?
14. Чему равна трудоемкость метода «Встречи посередине»? Почему?

Глава 2. СТРУКТУРА АЛГОРИТМОВ БЛОЧНОГО ШИФРОВАНИЯ

2.1. Определение алгоритма блочного шифрования

Алгоритм шифрования представляет собой множество обратимых преобразований формы сообщения с целью его защиты от несанкционированного прочтения. Исходное сообщение, которое подвергается преобразованию, называется открытым текстом, а результат, полученный с помощью применения алгоритма шифрования, называется шифртекстом. Переход от открытого текста к шифр-тексту называется зашифрованием, а обратный переход — дешифрованием [42]. Необходимым условием выполнения как прямого, так и обратного криптографического преобразования является наличие секретного ключа.

Блочным алгоритмом шифрования называется такой алгоритм шифрования, в котором как зашифрование, так и дешифрование выполняются над блоками фиксированной длины.

Процесс зашифрования обладает следующим свойством: различные блоки открытого текста отображаются в различные блоки шифртекста. При дешифровании соответствие сохраняется. Прямое преобразование можно рассматривать как перестановку на множестве сообщений с фиксированным размером блока. Совокупность данных, определяющих конкретное преобразование шифра из множества возможных, называется ключом.

Одним из основных приемов при разработке криптографических преобразований является многократная, состоящая из нескольких циклов, обработка одного блока открытого текста. Этот прием еще иногда называют итерированием. На каждом цикле данные подвергаются специальному преобразованию при участии вспомогательного ключа, полученного из заданного секретного ключа. Выбор числа циклов определяется требованиями криптостойкости и эффективности реализации блочного шифра. Как правило, чем больше циклов, тем выше криптостойкость и ниже эффективность реализации (больше задержка при шифровании/дешифровании) блочного шифра, и наоборот. Так, например, в случае алгоритма DES для того, чтобы все биты шифртекста зависели от всех битов ключа и всех битов открытого текста, необходимо пять циклов криптографического преобразования [3]. DES с шестнадцатью циклами обладает высокой криптостойкостью по отношению к ряду криптоаналитических атак.

Различают два вида построения блочных алгоритмов шифрования. Одни из них строятся на основе схемы Фейстеля. К числу таких алгоритмов относятся, например, бывший стандарт США — алгоритим DES и действующий стандарт России — ГОСТ 28147-89. Другие строятся по образу и подобию сети SPN (Substitution-Permutation Network). К числу таких алгоритмов принадлежит, на-

пример, новый стандарт США — AES. Ниже мы более подробно рассмотрим принципы построения алгоритмов блочного шифрования.

2.2. Схема Фейстеля

Схема Фейстеля, или сеть Фейстеля, представляет собой разновидность итерированного блочного шифра. При зашифровании блок открытого текста разделяется на две равные части — правую и левую. При этом длина исходного блока данных должна быть четной. В каждом цикле одна из частей подвергается преобразованию при помощи функции F и подключа K_i, полученного из исходного секретного ключа K. Результат операции суммируется по модулю 2 (операция XOR) с другой частью. Затем левая и правая части меняются местами. Общий вид одного цикла алгоритма шифрования, построенного по схеме Фейстеля, представлен на рис. 2.1. Преобразования в каждом цикле одинаковы, но на последнем не выполняется перестановка. Процедура дешифрования аналогична процедуре зашифрования за тем исключением, что подключи k_i выбираются в обратном порядке. Если же при зашифровании в последнем цикле была выполнена перестановка, то дешифрование следует начать с перестановки левой и правой части блока данных.

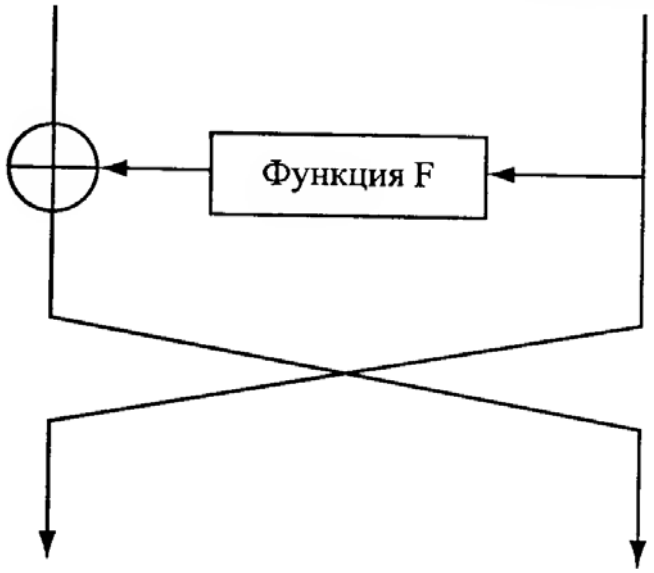


Рис. 2.1. ***Один раунд алгоритма шифрования, построенного по схеме Фейстеля***

Уже отмечалось, что к алгоритмам шифрования, построенным по схеме Фейстеля, относятся такие, как DES и ГОСТ 28147-89. Помимо вышеуказанных алгоритмов, существует целый ряд других, таких, как, например Lucifer, FEAL, Khufu, Khafre, LOKI, COST, Blowfish, также построенных по схеме Фейстеля. Блочный алгоритм шифрования, использующий описанную конструкцию, является обратимым и гарантирует возможность восстановления входных данных функции F в каждом цикле. Сама функция F необязательно должна быть обратимой. При зада-

нии произвольной функции F не потребуется реализовывать две различные процедуры — одну для шифрования, а другую для дешифрования. Структура схемы Фейстеля автоматически позаботится об этом [3].

Необходимо также отметить, что деление исходного шифра на две части может быть заменено делением на четыре, восемь частей и более. Такие алгоритмы шифрования называются производными от схемы Фейстеля. Некоторые современные шифры имеют такую структуру. Например, алгоритм шифрования CAST или закрытый до недавнего времени алгоритм шифрования SkipJack. На рис. 2.2 показаны пример некоторых модификаций схемы Фейстеля.

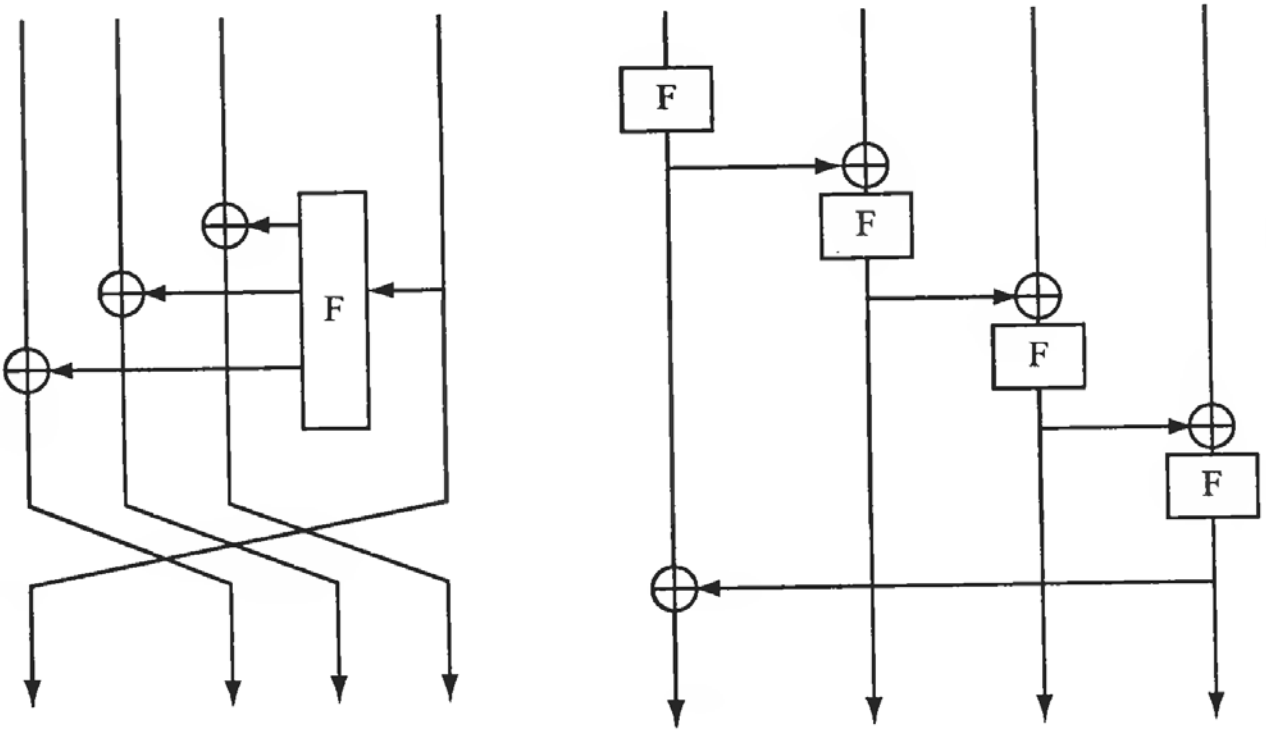


Рис. 2.2. ***Модификации схемы Фейстеля***

2.3. Принципы построения блочных шифров

Криптоаналитическая стойкость алгоритма шифрования, построенного по схеме Фейстеля зависит от трех основных параметров:

— числа раундов шифрования;

— вида функции F;

— алгоритма вычисления ключей.

Рассмотрим зависимость надежности произвольного алгоритма шифрования от каждого из этих параметров в отдельности.

2.3.1. Число раундов шифрования

Уже отмечалось, что, чем больше циклов в алгоритме шифрования, тем выше его криптостойкость и ниже эффективность реализации (больше задержка при шифровании/дешифровании), и наоборот. С ростом числа циклов возрастает трудоемкость криптоанализа алгоритма шифрования даже при относительно слабой функции F, что и объясняет возрастание криптостойкости. В общем случае число раундов должно выбираться так, чтобы все известные методы криптоанализа требовали больше усилий, чем анализ с помощью перебора всех ключей. Этот критерий, без сомнения, использовался при разработке алгоритма шифрования DES [2].

С помощью приведенного критерия сравнительно легко можно для любого алгоритма шифрования оценить уровень его надежности и сравнить с другими алгоритмами. Можно сказать, что надежность любого удовлетворяющего данному критерию алгоритма можно оценивать просто по длине ключа.

2.3.2. Требования, предъявляемые к функции преобразования F

Здесь нам кажется разумным привести критерии, обозначенные В. Столингсом в его работе [2]. Функция F в алгоритмах шифрования, построенных по схеме Фейстеля, обеспечивает необходимый уровень конфузии. Термин ***конфузия*** был введен в шифрование Клодом Шенноном и означает степень сложности статистической взаимосвязи между шифрованным текстом и ключом. Обеспечение необходимого уровня конфузии предполагает невозможность «дешифровывания» подстановки, задаваемой данной функцией. Для этого функция F должна быть нелинейной. Чем выше степень нелинейности F, тем сложнее будет шифр для криптоанализа любого типа. Степень нелинейности можно характеризовать несколькими способами. Говоря в общем, чем труднее аппроксимировать функцию F с помощью системы линейных уравнений, тем выше степень нелинейности этой функции.

Кроме того, при построении функции F должны приниматься во внимание и другие критерии. Например, полученный в результате алгоритм должен обладать заметным ***лавинным эффектом***. Суть лавинного эффекта заключается в том, что изменение одного бита входных данных влечет изменение множества битов на выходе. Аналогичный более сильный критерий, называемый ***строгим критерием лавинного эффекта*** (SAC — strict avalanche criterion), требует, чтобы для любых i и j при инвертировании входного бита i S-блока замены (блока, в которм происходит замена одного значения на другое) любой выходной бит j изменялся с вероятностью 1/2. Хотя приведенный здесь строгий критерий лавинного эффекта формулируется для S-блоков, подобный критерий можно сформулировать и для функции F в целом. В такой формулировке критерий может использоваться при разработке алгоритмов, не имеющих в своей структуре S-блоков.

Другим критерием является ***критерий независимости битов*** (BIC — bit independence criterion), согласно которому для любых значений i, j и k при инвертировании входного бита i S-блока выходные биты j и k должны меняться независимо. Строгий критерий лавинного эффекта и критерий независимости битов призваны гарантировать алгоритму необходимый уровень конфузии.

2.3.3. Структура S-блоков

S-блоки, как правило, входят в состав фукнкции F и имеют большое значение для стойкости алгоритма шифрования. На данный момент изучение свойств S-блоков и способов их усовершенствования является одной из главнейших задач в области симметричных алгоритмов блочного шифрования. По сути требуется, чтобы любые изменения входных данных S-блока выливались в случайные на вид изменения выходных данных. Зависимость выходных значений от входных должна быть нелинейной и плохо аппроксимироваться линейными функциями (именно это свойство используется при применении линейного криптоанализа).

Очевидно, что одной из важных характеристик S-блока является его размер. S-блок размером n×m предполагает n-битовые входные значения и m-битовые выходные. Вообще говоря, чем больше S-блоки, тем выше устойчивость алгоритма по отношению к методам линейного и дифференциального криптоанализа. С другой стороны, чем больше n, тем больше (в экспоненциальной зависимости) задающая преобразование таблица. Поэтому, исходя из практических соображений, значение n, как правило, выбирают в диапазоне от 8 до 10. Кроме того, чем больше S-блок, тем сложнее его проектировать [2].

При проектировании достаточно больших блоков замены, к которым можно отнести и матрицы размера 8×32, возникает вопрос о выборе метода, с помощью которого подбираются элементы S-блока, обеспечивающие соответствие строгому критерию лавинного эффекта критерию независимости битов, а также ***критерию гарантированного лавинного эффекта*** (GA — guaranteed avalanche) порядка γ, когда при изменении одного бита входных данных меняются как минимум γ выходных битов. Найберг предложил использовать следующие подходы при разработке S-блоков [2].

1. **Случайный выбор**. Элементы S-блоков выбираются с помощью генератора или специальных таблиц псевдослучайных чисел. В случае небольшого размера (6×4) такой способ может привести к созданию S-блоков с нежелательными характеристиками, но для больших блоков (8×32) он должен быть вполне приемлемым.
2. **Случайный выбор с проверкой**. Элементы S-блока выбираются случайным образом, но после этого полученные результаты должны проверяться на соответствие различным критериям, описанным выше, с отсеиванием тех матриц, которые не выдержали такой проверки.

3. **Выбор вручную**. Элементы S-блока выбираются практически вручную с использованием элементарных математических преобразований. Для больших S-блоков использование данного подхода сопряжено с немалыми трудностями.
4. **Математический подход**. Элементы S-блока генерируются на основе тех или иных математических принципов. Такой подход обеспечивает S-блоки, гарантирующие заданный уровень надежности по отношению к методам линейного и дифференциального криптоанализа, а также хорошие показатели ***диффузии*** (то есть рассеивания статистических особенностей открытого текста по широкому диапазону статистических характеристик шифрованного текста).

2.3.4. Алгоритм вычисления ключа

Практически в каждом алгоритме блочного шифрования из основного секретного ключа происходит вычисление раундовых подключей по определенной схеме. Поэтому исследование, анализ и разработка алгоритмов вычисления подключей также является важной задачей при построении алгоритмов блочного шифрования. В общем случае желательно, чтобы алгоритм выбора подключей в максимальной степени усложнял как задачу определения этих подключей при криптоанализе, так и задачу восстановления главного ключа по значениям подключей. Однако до настоящего времени публикаций, в которых были бы представлены общие подходы к решению этой задачи, нет.

2.4. Контрольные вопросы

1. Дайте определение алгоритма блочного шифрования.
2. Как влияет количество циклов, используемых в алгоритме, на результат шифрования?
3. Опишите принцип построения схемы Фейстеля.
4. Перечислите несколько алгоритмов шифрования, построенных по схеме Фейстеля.
5. Какие параметры влияют на криптоаналитическую стойкость алгоритма, построенного по схеме Фейстеля?
6. Что в криптографии понимается под термином *конфузия*?
7. Что в криптографии понимается под термином *диффузия*?
8. Какие критерии должны учитываться при разработке F-функции?
9. Дайте определение строгого критерия лавинного эффекта.
10. Сформулируйте критерий независимости битов.
11. Перечислите и дайте краткое описание способов построения S-блоков, предложенных Найбергом.

Глава 3. СТАНДАРТЫ БЛОЧНОГО ШИФРОВАНИЯ

3.1. Федеральный стандарт США — DES

Не смотря на то, что с мая 2002 г. в США вступил в силу новый стандарт шифрования данных AES, предыдущий стандарт шифрования данных DES (Data Encryption Standard), который ANSI называет Алгоритмом шифрования данных DEA (Data Encryption Algorithm), а ISO — DEA-1, остается одним из самых известных алгоритмов. Можно даже сказать, что в настоящий момент алгоритм шифрования DES используется в основном для обучения специалистов по защите информации, являясь открытым и в то же время стойким алгоритмом шифрования. За годы своего существования он выдержал натиск различных атак и для многих применений все еще является криптостойким.

DES представляет собой блочный шифр, на вход которого поступают 64-битовые блоки исходных данных и на выходе которого появляются 64-битовые блоки шифрованных данных.

DES является симметричным алгоритмом: для шифрования и дешифрования используются одинаковые алгоритмы и ключ (за исключением небольших отличий в использовании ключа). Длина секретного ключа равна 56 битам. (Ключ обычно представляется 64-битным числом, но каждый восьмой бит используется для проверки четности и игнорируется. Биты четности являются наименьшими значащими битами байтов ключа.)

Криптостойкость полностью определяется ключом. Фундаментальным строительным блоком DES является комбинация подстановок и перестановок. DES состоит из 16 циклов (см. рис. 3.1). В общем виде цикл преобразования представлен на рис. 3.2.

Если L_i и R_i — левая и правая половины, полученные в результате i-й итерации, K_i — 48-битный ключ для цикла i, а F — функция, выполняющая все подстановки, перестановки и XOR с ключом, то один цикл преобразования можно представить как

$$(L_i, R_i) = (R_{i-1}, L_{i-1} \oplus F(R_{i-1}, K_i)).$$

Учитывая подстановку $F_i(\cdot)$ и перестановку $T(\cdot)$, цикл преобразования можно представить так, как это сделано на рис. 3.3. Из него видно, что каждый цикл DES представляет собой композиционный шифр с двумя последовательными преобразованиями — подстановкой $F_i(\cdot)$ и перестановкой $T(\cdot)$ (за исключением последнего, шестнадцатого цикла, где перестановка опускается).

Таким образом, алгоритм шифрования DES построен по схеме Фейстеля и сконструирован так, чтобы выполнялось полезное свойство: для шифрования и

дешифрования используется один и тот же алгоритм. Единственное отличие состоит в том, что ключи должны использоваться в обратном порядке. То есть если при шифровании использовались ключи K_1, K_2, K_3,.., K_{16}, то ключами дешифрования будут K_{16}, K_{15}, K_{14},.., K_1. Алгоритм использует только стандартную арифметику 64-битовых чисел и логические операции, поэтому легко реализуется на аппаратном уровне.

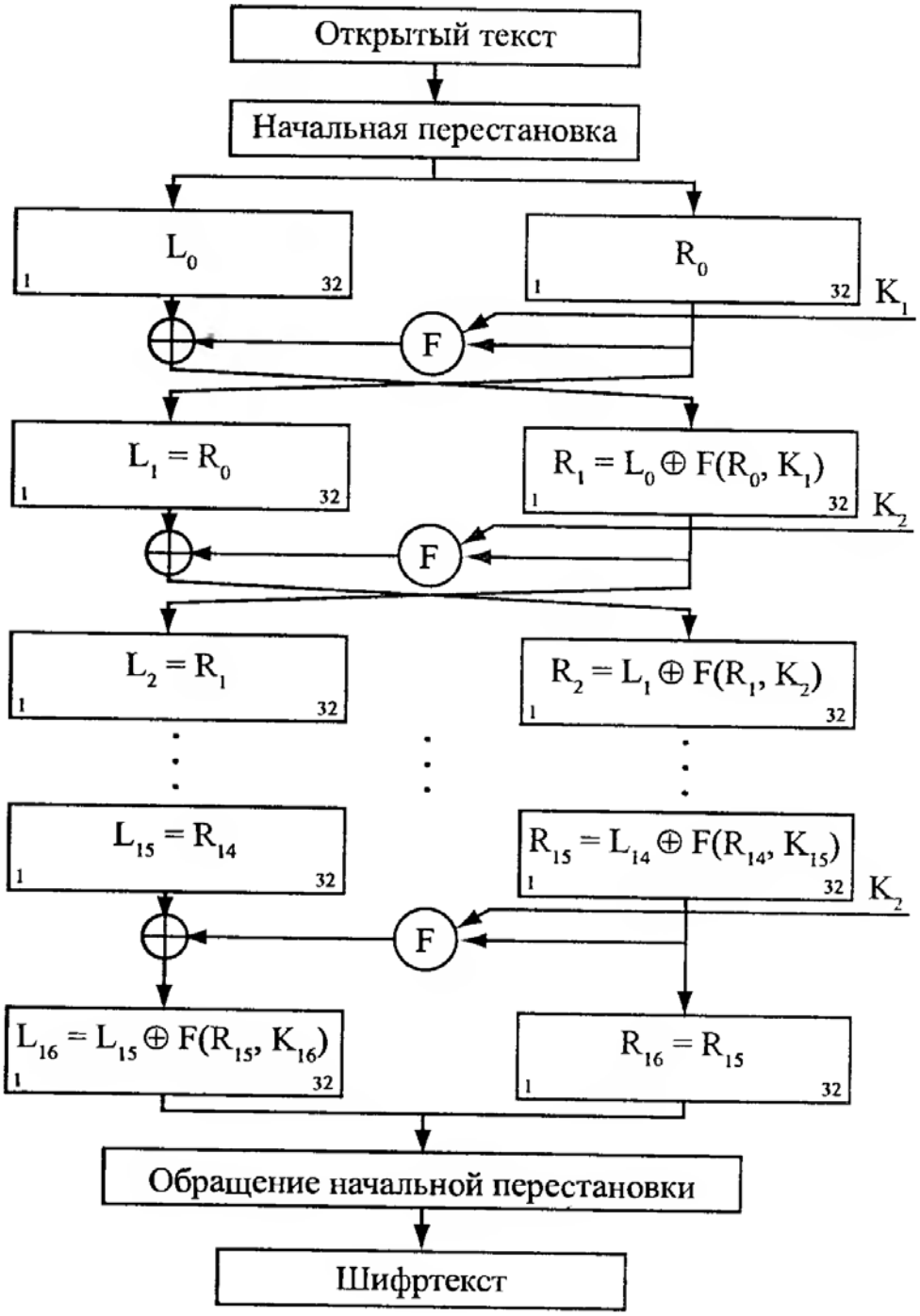


Рис. 3.1. ***Схема DES-преобразования***

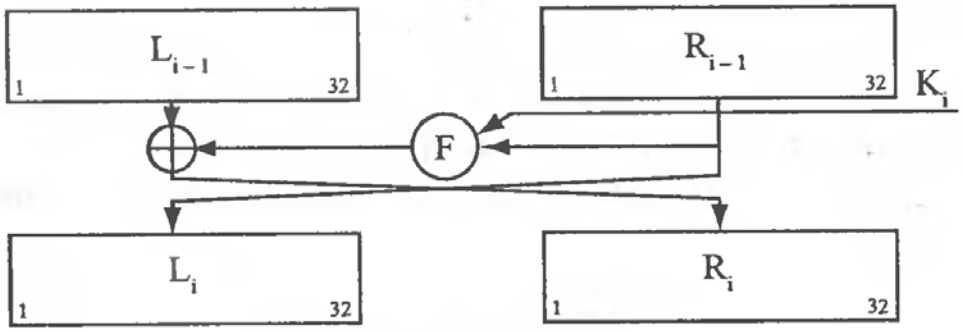


Рис. 3.2. ***Общий вид цикла DES-преобразования***

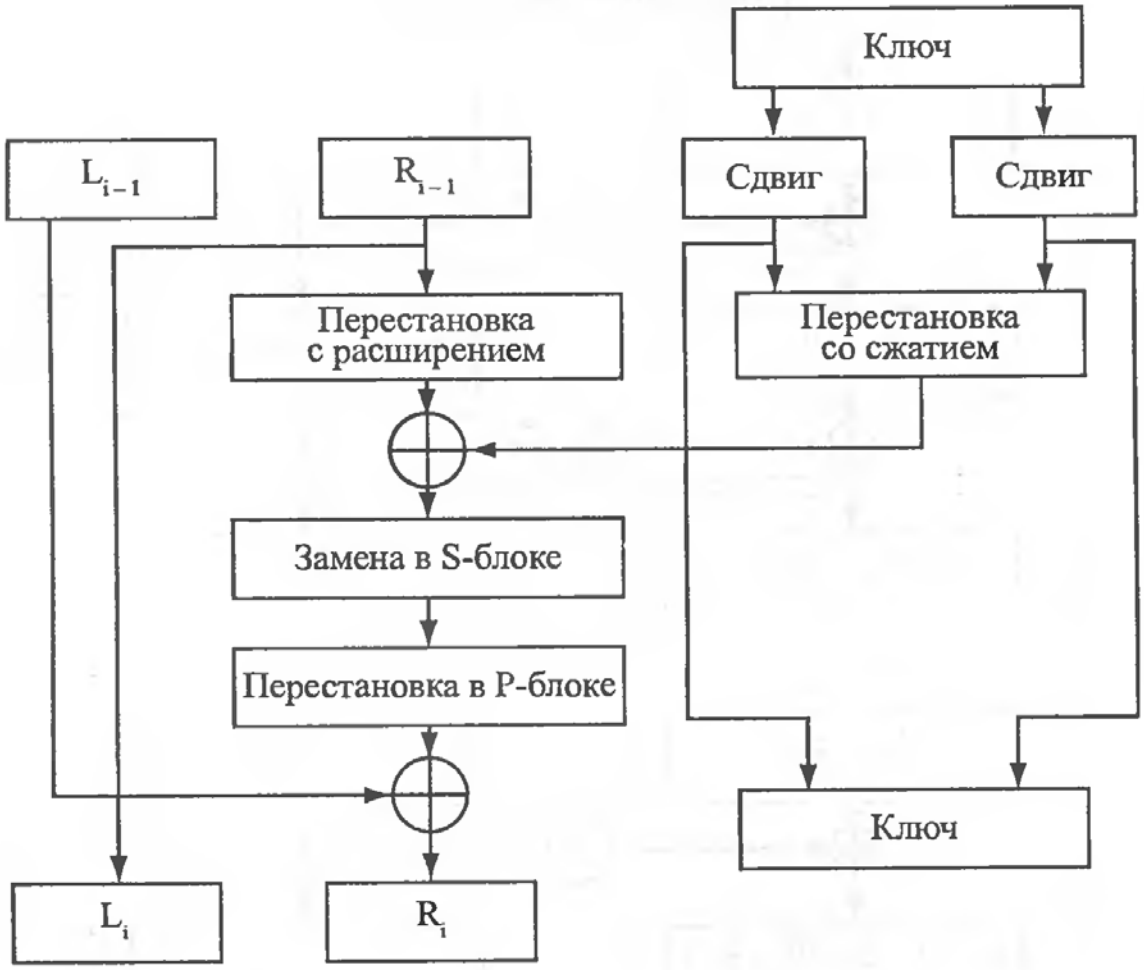


Рис. 3.3. ***Один цикл DES-преобразования***

DES работает с 64-битовыми блоками открытого текста. После первоначальной перестановки блок разбивается на правую и левую половины длиной по 32 бита. Затем выполняется 16 преобразований (функция F), в которых данные объединяются с ключом. После шестнадцатого цикла правая и левая половины объединяются и алгоритм завершается заключительной перестановкой (обратной по отношению к первоначальной). На каждом цикле (см. рис. 3.3) биты ключа сдвигаются, и затем из 56 битов ключа выбираются 48 битов. Правая половина

данных увеличивается до 48 битов с помощью перестановки с расширением, объединяется посредством XOR с 48 битами смещенного и переставленного ключа, проходит через 8 S-блоков, образуя 32 новых бита, и переставляется снова. Эти четыре операции выполняются функцией F.

Затем результат функции F объединяется с левой половиной с помощью другого XOR. В итоге этих действий появляется новая правая половина, а старая правая — становится новой левой половиной. Эти действия повторяются 16 раз, образуя 16 циклов DES.

Начальная перестановка выполняется еще до первого цикла, при этом входной блок переставляется так, как показано в таблице 3.1.

Таблица 3.1

DES — 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

Эту и все последующие таблицы данной главы следует читать слева направо и сверху вниз. Например, начальная перестановка перемещает бит 58 в позицию 1, бит 50 — в позицию 2, бит 42 — в позицию 3 и т. д.

Начальная перестановка и соответствующая заключительная перестановка не влияют на криптостойкость DES.

Процедура преобразования ключа сводится к следующим действиям. Сначала 64-битовый ключ DES уменьшается до 56-битового ключа отбрасыванием каждого восьмого бита, как показано в таблице 3.2.

Таблица 3.2

DES — преобразование 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

После извлечения 56-битного ключа для каждого из 16 циклов генерируется новый 48-битный подключ. Эти подключи K_i определяются следующим образом. Сначала 56-битный ключ делится на две 28-битные половины. Затем половины циклически сдвигаются влево на один или два бита в зависимости от номера цикла. Этот сдвиг показан в таблице 3.3.

Таблица 3.3

DES — сдвиг ключа в зависимости от номера цикла

Номер цикла	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Сдвиг (бит)	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

После сдвига выбирается 48 бит из 56. Так как при этом не только выбирается подмножество битов, но и изменяется их порядок, эта операция получила название **перестановка со сжатием**. Ее результатом является набор из 48 бит. Перестановка со сжатием определена в таблице 3.4.

Таблица 3.4

DES — перестановка со сжатием

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

Например, бит сдвинутого ключа в позиции 33 перемещается в позицию 35 результата, а 18-й бит сдвинутого ключа отбрасывается.

Из-за сдвига для каждого подключа используются различные подмножества бит ключа. Каждый бит используется приблизительно в четырнадцати из шестнадцати подключей, при этом не все биты используются одинаковое число раз.

Операция перестановки с расширением правой половины данных, R_i, от 32 до 48 битов и изменением их порядка, называется **перестановкой с расширением**. У этой операции две задачи: привести размер правой половины в соответствие с ключом для операции XOR и получить более длинный результат, который можно будет сжать в ходе операции подстановки.

Однако криптографический смысл данной операции состоит в том, что за счет влияния одного бита на две подстановки быстрее возрастает зависимость битов результата от битов исходных данных. Данная зависимость называется лавинным эффектом. DES спроектирован так, чтобы как можно быстрее добиться зависимости каждого бита шифртекста от каждого бита открытого текста и каждого бита ключа. Перестановка с расширением приведена в таблице. 3.5. Иногда она называется E-блоком (от английского слова *expansion*). Для каждого 4-битного входного блока первый и четвертый биты представляют собой два бита выходного блока. В таблице 3.5 показано соответствие позиций результата и исходных данных. Например, бит входного блока в позиции 3 переместится в позицию 4 выходного блока, а бит входного блока в позиции 21 — в позиции 30 и 32 выходного блока. Хотя выходной блок больше входного, каждый входной блок генерирует уникальный выходной блок.

Таблица 3.5

DES — перестановка с расширением

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

После объединения сжатого блока с расширенным блоком с помощью операции XOR над 48-битным результатом выполняется операция подстановки. Подстановка производится в восьми блоках, называемых S-блоками (от английского слова *substitution*). У каждого S-блока есть свой 6-битный вход и 4-битный выход, всего используется восемь различных S-блоков. (Для восьми S-блоков DES потребуется 256 байтов памяти.) 48 битов делятся на восемь 6-битовых подблоков. Каждый отдельный подблок обрабатывается отдельным S-блоком: первый подблок — первым S-блоком, второй — вторым S-блоком и т. д. (см. рис. 3.4).

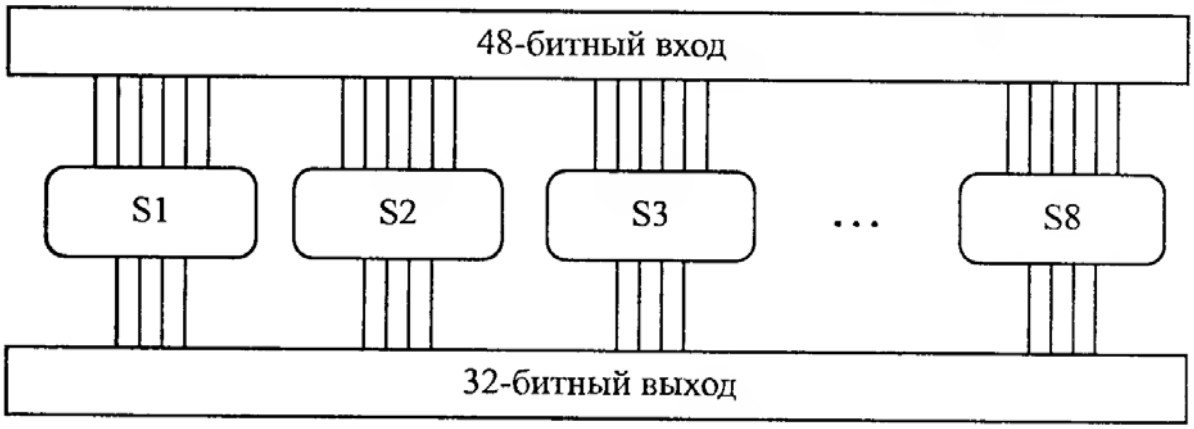


Рис. 3.4. ***DES-подстановка при помощи S-блоков***

Каждый S-блок представляет собой таблицу из четырех строк и шестнадцати столбцов. Каждый элемент в блоке является 4-битным столбцом. По шести входным битам S-блока определяется, под какими номерами столбцов и строк следует искать выходное значение. Все восемь S-блоков показаны в таблицах 3.6–3.13.

Таблица 3.6

DES — S-блок 1

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
01	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
10	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
11	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Таблица 3.7

DES — S-блок 2

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
01	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
10	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
11	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

Таблица 3.8

DES — S-блок 3

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
01	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
10	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
11	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

Таблица 3.9

DES — S-блок 4

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
01	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
11	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

Таблица 3.10

DES — S-блок 5

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
01	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
10	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

Таблица 3.11

DES — S-блок 6

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
01	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
10	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
11	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

Таблица 3.12

DES — S-блок 7

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
01	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
10	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
11	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

Таблица 3.13

DES — S-блок 8

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
01	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
10	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
11	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Входные биты особым образом определяют элемент S-блока. Рассмотрим 6-битовый вход S-блока: b_1, b_2, b_3, b_4, b_5 и b_6. Биты b_1 и b_6 объединяются, образуя 2-битное число от 0 до 3, соответствующее строке таблицы. Средние четыре бита, с b_2 по b_5, объединяются, образуя 4-битное число от 0 до 15, соответствующее столбцу таблицы. Необходимо учитывать, что строки и столбцы нумеруются с нуля, а не с единицы. Например, пусть на вход шестого S-блока (то есть биты функции XOR с 31 по 36) попадает 110011. Первый и последний биты, объединяясь, образуют 11, что соответствует строке три шестого S-блока. Средние четыре бита образуют 1001, что соответствует столбцу девять того же S-блока. Элемент шестого S-блока, находящийся на пересечении строки три и столбца девять, — это 14. В результате этой подстановки получается восемь 4-битных блоков, которые вновь объединяются в единый 32-битный блок. Этот блок поступает на вход следующего этапа — перестановки с помощью P-блока.

32-битный выход подстановки с помощью S-блоков перетасовывается в соответствии с P-блоком. Эта перестановка перемещает каждый входной бит в другую позицию, ни один бит не используется дважды, и ни один бит не игнорируется. Этот процесс называется **прямой перестановкой**, или просто **перестановкой**. Позиции, в которые перемещаются биты, показаны в таблице 3.14. Например, двадцать первый бит перемещается в позицию четыре, а четвертый бит — в позицию тридцать один.

Таблица 3.14

DES — перестановка с помощью P-блока

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

Наконец, результат перестановки с помощью P-блока объединяется посредством XOR с левой половиной первоначального 64-битового блока. Затем левая и правая половины меняются местами, и начинается следующий цикл криптографического преобразования.

Заключительная перестановка является обратной по отношению к первоначальной и описана в таблице 3.15.

Таблица 3.15

DES — 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

Отметим, что левая и правая части не меняются местами после завершения последнего цикла DES. Вместо этого объединенный блок $R_{16}L_{16}$ используется как вход заключительной перестановки. Перестановка половин с последующим циклическим сдвигом привела бы к точно такому же результату. Это сделано для того, чтобы алгоритм шифрования можно было использовать как для шифрования, так и для дешифрования.

Алгоритм, который создает ключ для каждого цикла, также цикличен. Ключ сдвигается направо, а число позиций сдвига равно 0, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1.

3.1.1. Упрощенный DES

Упрощенный DES — это алгоритм шифрования, имеющий, скорее, учебное, чем практическое значение. По свойствам и структуре он подобен DES. Данный алгоритм был разработан профессором Эдвардом Шейфером из университета Санта-Клара. Для удобства дальнейшей работы будем называть данный алгоритм S-DES [2].

В процессе зашифрования с помощью алгоритма S-DES 8-битовый блок открытого текста преобразуется в 8-битовый блок шифртекста. В каждом раунде используется свой 8-битовый секретный подключ, который может быть получен из исходного 10-битового ключа. Алгоритм дешифрования S-DES аналогичен алгоритму шифрования за тем исключением, что секретные подключи используются в обратном порядке.

Алгоритм шифрования сосотит из начальной перестановки IP, двух раундов шифрования и конечной перестановки IP^{-1}. Каждый раунд в свою очередь состоит из перестановок, подстановок и сложения данных с секретным ключом. Между раундами происходит перестановка SW, в процессе которой две половинки последовательности данных меняются местами.

Как уже упоминалось выше, функция F_K использует в качестве исходных данных не только шифруемый текст, но и 8-битовый подключ. В качестве секретных подключей можно использовать два 8-битовых ключа, сгенерированных независимо друг от друга, можно использовать один и тот же 8-битовый подключ, для чего просто следует ввести его дважды. Наконец, можно прибегнуть к комбинированному решению, когда требуется 10-битовый ключ, из которого генерируются два 8-битовых.

3.1.2. Вычисление ключей S-DES

Как уже говорилось, одним из вариантов иолучения 8-битового подключа является его извлечение из исходного секретного 10-битового ключа. На рис. 3.5 показана схема создания двух раундовых подключей из исходного секретного ключа.

В начале исходный 10-битовый подключ подвергается перестановке с помощью таблицы 3.16. Перестановка происходит аналогично тому, как это происходило в случае с алгоритмом шифрования DES, то есть третий бит должен переместиться в первую позицию, пятый — во вторую и т. д.

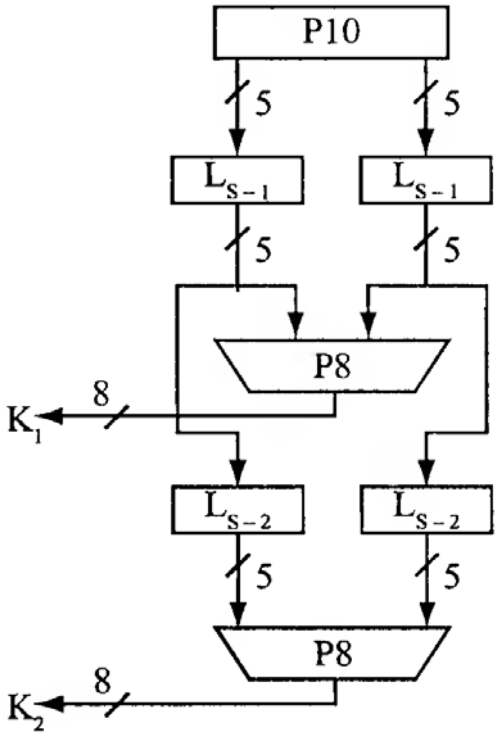


Рис. 3.5. ***Вычисление ключей S-DES***

Таблица 3.16

Начальная перестановка битов ключа в алгоритме шифрования S-DES

P10									
3	5	2	7	4	10	1	9	8	6

Так, например, в соответствии с данной таблицей ключ (1010101010) будет преобразован к виду (1101001100). После применения операции перестановки данные разделяются на две равные части по пять бит каждая, и отдельно для каждой из частей выполняется циклический сдвиг влево на одну позицию (L_{S-1}), который еще называют вращением. В нашем случае в результате будет получена последовательность (10101 11000).

После этого к полученным данным применяется перестановка P8, аналогичная перестановке со сжатием в алгоритме шифрования DES. В результате применения этой перестановки из 10-битового ключа сначала выбираются, а затем переставляются 8 битов, согласно таблице 3.17.

Таблица 3.17

Перестановка для извлечения раундового подключа

P8							
6	3	7	4	8	5	10	9

В результате этой операции получается первый подключ (K_1). В нашем примере он будет иметь вид (11100100).

После этого необходимо вернуться к двум 5-битовым половинкам данных, полученным в результате применения функций L_{S-1}, и применить к каждой из этих половинок операцию циклического сдвига влево на две позиции (L_{S-2}). В нашем конкретном случае значение (10101 11000) будет преобразовано к виду (10110 00011). Наконец, применив к полученной в результате последовательности перестановку P8, получим второй подключ K_2. Для нашего примера результатом будет (01010011).

3.1.3. Шифрование S-DES

Общий вид алгоритма шифрования S-DES представлен на рис. 3.6. Рассмотрим каждую из используемых в данном алгоритме шифрования операций отдельно.

Начальная и завершающая перестановки IP и IP^{-1}

На вход алгоритма поступает 8-битовый блок открытого текста, к которому применяется начальная перестановка IP, заданная с помощью таблицы 3.18. Перестановка выполняется аналогично тому, как это делается в алгоритме шифрования DES, то есть согласно таблице 3.18 второй бит исходного текста переместится в первую позицию, шестой — во вторую и т. д.

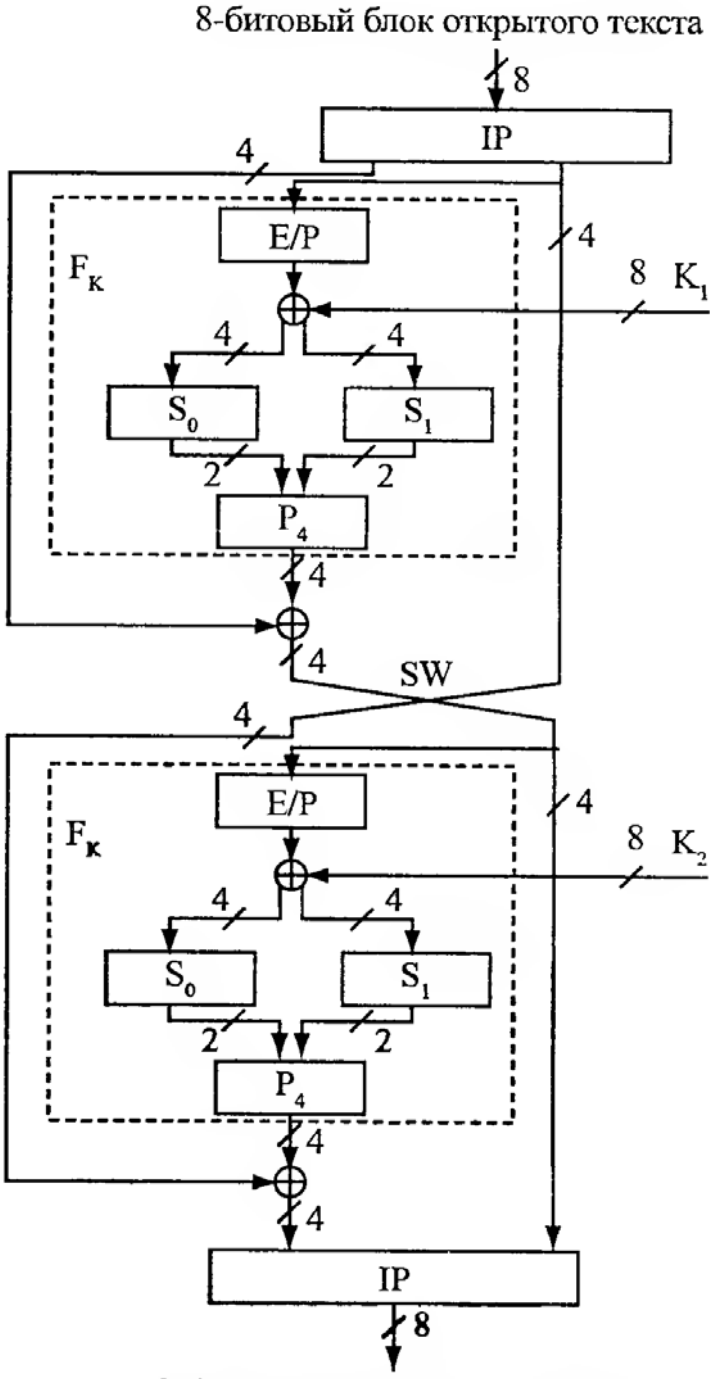


Рис. 3.6. ***Алгоритм шифрования S-DES***

Таблица 3.18

Начальная перестановка IP

IP							
2	6	3	1	4	8	5	7

На завершающей стадии алгоритма выполняется обратная перестановка IP^{-1}, заданная с помощью таблицы 3.19, то есть если при перестановке IP второй бит переместился в первую позицию, то после применения перестановки IP^{-1} первый бит должен будет переместиться во вторую позицию.

Таблица 3.19

Завершающая перестановка

IP^{-1}							
4	1	3	5	7	2	8	6

Говоря иными словами, вторая из приведенных выше перестановок действительно является обратной по отношению к первой, то есть $IP^{-1}(IP(X)) = X$.

Функция F_k

Самым сложным компонентом S-DES является функция F_k, представляющая собой один раунд алгоритма. На вход функции F алгоритма шифрования S-DES поступает 4-битовое сообщение (n_1, n_2, n_3, n_4), представляющее собой правую часть шифруемого текста. Первой операцией является операция перестановки с расширением, выполняемая с помощью таблицы 3.20, в результате которой четырехбитовое сообщение преобразуется к восьмибитовому путем повторения битов исходного текста.

Таблица 3.20

Таблица перестановки с расширением

E/P							
4	1	2	3	2	3	4	1

Таким образом, исходное четырехбитовое сообщение (n_1, n_2, n_3, n_4) после применения перестановки с расширением преобразуется к виду (n_4, n_1, n_2, n_3, n_3, n_3, n_4, n_1). К полученному сообщению с помощью операции сложения по модулю два (XOR) добавляется первый 8-битовый секретный подключ $K_1 = (k_{11}, k_{12}, k_{13}, k_{14}, k_{15}, k_{16}, k_{17}, k_{18})$, в результате чего получается ($n_4 \oplus k_{11}$, $n_1 \oplus k_{12}$, $n_2 \oplus k_{13}$, $n_3 \oplus k_{14}$, $n_2 \oplus k_{15}$, $n_3 \oplus k_{16}$, $n_4 \oplus k_{17}$, $n_1 \oplus k_{18}$).

Первые четыре бита полученного сообщения поступают на вход блока замены S_0, на выходе которого получается 2-битовая последовательность, а оставшиеся четыре бита — на вход блока замены S_1, на выходе которого получается другая 2-битовая последовательность. Блоки S_0 и S_1 можно определить так, как показано в таблицах 3.21 и 3.22, соответственно.

Эти S-блоки (матрицы кодирования) работают следующим образом. Первый и четвертый биты входной последовательности рассматриваются как 2-битовые числа, определяющие строку, а второй и третий — как числа, определяющие столбец S-блока. Элементы, находящиеся на пересечении соответствующих строки и столбца, задают 2-битовые входные значения. Например, если первые четыре бита последовательности равны 1010, то выходные два бита задаются значением, которое находится на пересечении строки (10) 2 и столбца (01) 1 блока S_0. Точно так же происходит определение номеров строки и столбца для блока замены S_1.

Таблица 3.21

Блок S_0

S_0	0	1	2	3
0	1	0	3	2
1	3	2	1	0
2	0	2	1	3
3	3	1	3	1

Таблица 3.22

Блок S_1

S_1	0	1	2	3
0	1	1	2	3
1	2	0	1	3
2	3	0	1	0
3	2	1	0	3

В заключение четыре бита, полученные на выходах блоков S_0 и S_1, преобразуются с помощью перестановки P4 так, как показано в таблице 3.23.

Таблица 3.23

Таблица перестановки

P4			
2	4	3	1

Результат применения перестановки P4 и является результатом функции F.

Функция-переключатель SW

Алгоритм шифрования S-DES построен по схеме Фейстеля. Поэтомумежду раундами щифрования используется функция SW, которая меняет местами первые и последние четырс бита последовательности, чтобы при следующем вызове функции F_K последняя работала уже с другой четверкой битов. При втором вызове функции F_K операции E/P S_0, S_1 и P4 остаются теми же, что и при первом, но вместо первого подключа K_1 используется второй подключ K_2.

3.2. Стандарт России — ГОСТ 28147-89

Алгоритм шифрования ГОСТ 28147-89 был создан в недрах спецслужб Советского Союза и первоначально на нем стоял гриф «Совершенно секретно», затем «Для служебного пользования» и в конце концов шифр был опубликован.

ГОСТ 28147-89 (рис. 3.7) представляет собой блочный алгоритм шифрования с 256-битным ключом и 32 циклами преобразования, оперирующий 64-битными блоками.

Для шифрования открытый текст разбивается на две равные части N1 и N2 по 32 бита каждая (см. рис. 3.7). В i-м цикле используется подключ K_i. Функция F реализована следующим образом. Сначала правая половина N2 и подключ K_i складываются по модулю 2^{32}. Результат разбивается на восемь 4-битовых последовательностей, каждая из которых поступает на вход своего S-блока. ГОСТ использует восемь различных S-блоков: первые 4 бита попадают в первый S-блок, вторые четыре — во второй и т. д. Каждый S-блок представляет собой перестановку чисел от 0 до 15. Все восемь S-блоков различны, они фактически являются дополнительным ключевым материалом. Выходы всех восьми S-блоков объединяются в 32-битное слово, затем слово циклически сдвигается влево на 11 битов. Наконец, результат объединяется с помощью операции XOR с левой половиной N1, и получается новая правая половина, а правая половина становится новой левой половиной.

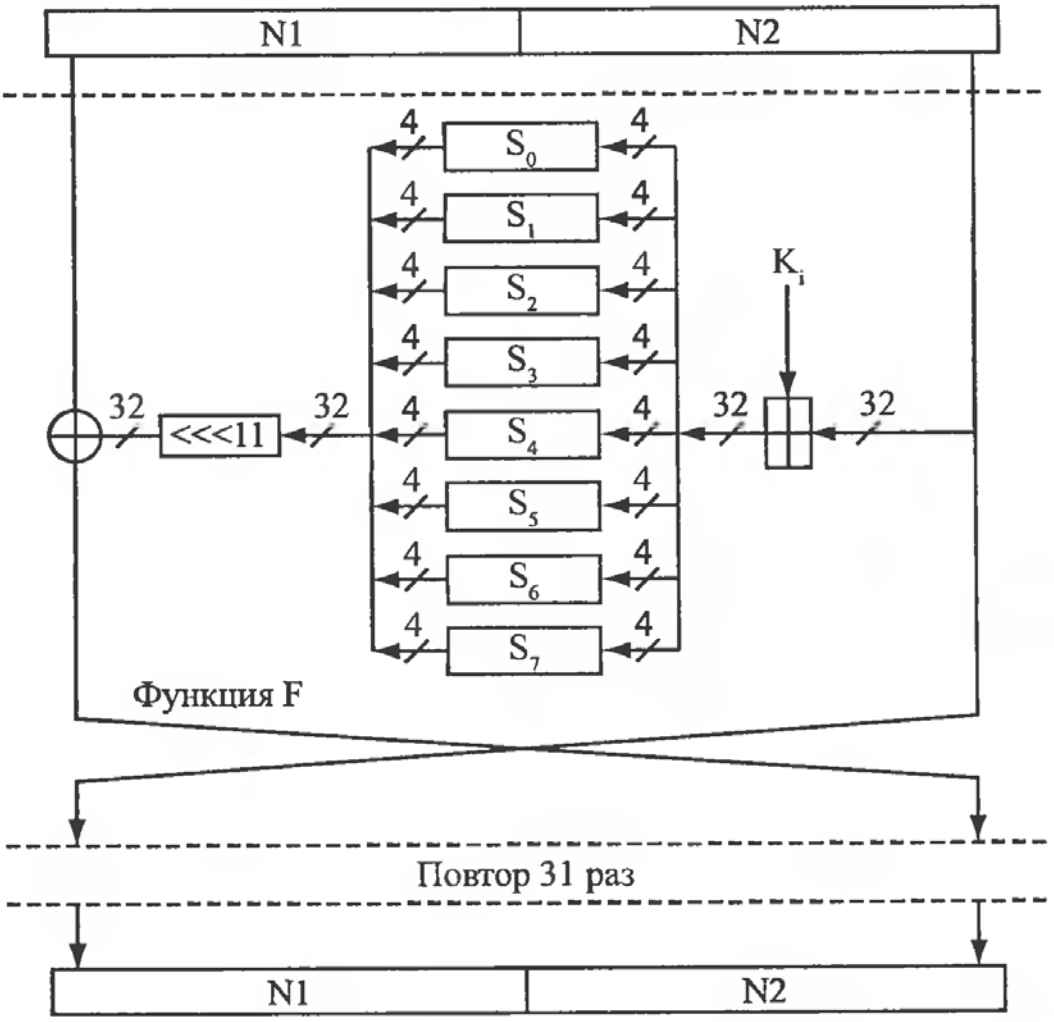


Рис. 3.7. ***Алгоритм шифрования ГОСТ 28147-89***

Расширение материала ключа производится по схеме K[i] = (K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0).

Очевидно, что решающую роль в криптостойкости алгоритма играют S-блоки замены. Поэтому они должны выбираться с учетом несложных условий, обеспечивающих стойкость к потенциально возможным атакам для того, чтобы алгоритм имел запас прочности. Это связано и с применением 256-битного ключа и с двух-трехкратным запасом по числу раундов. Алгоритм слабо подвержен распараллеливанию, а вот в плане многоплатформенности он имеет достаточно «удобный» дизайн.

3.3. Новый алгоритм Rijndael — победитель конкурса AES

3.3.1. История проведения конкурса AES

В связи с бурным развитием криптоанализа в последние десятилетия появились алгоритмы шифрования, превосходящие по своим параметрам стандарт шифрования DES. Поэтому в 1997 г. правительство США объявило на базе Института стандартизации NIST (The National Institute of Standards and Technology) открытый конкурс на новый стандарт блочного шифра США. Победитель конкурса получал статус нового стандарта шифрования AES (Advanced Encryption Standard) и рекомендовался к повсеместному использованию на территории США. К новому криптостандарту были заявлены следующие требования [13, 43]:

— криптоалгоритм должен быть открыто опубликован;
— криптоалгоритм должен быть симметричным блочным шифром, поддерживающим 128-, 192- и 256-битные ключи;
— криптоалгоритм должен быть предназначен как для аппаратной, так и для программной реализации;
— криптоалгоритм должен быть доступен для открытого использования в любых продуктах, а значит, не может быть запатентован, в противном случае, патентные права должны быть аннулированы;
— криптоалгоритм подвергается изучению по следующим параметрам: стойкости, стоимости, гибкости, реализуемости в смарткартах.

Всего к рассмотрению на конкурсе к середине 1998 г. поступило 15 предложений из научно-исследовательских центров всего мира. Все алгоритмы с комментариями авторов были опубликованы в печатном и электронном виде. После этого в течение года NIST принимал письма и электронные послания, в которых каждый мог высказаться за или против того или иного алгоритма шифрования (естественно, обосновав при этом свое мнение). В общей сложности в обсуждении приняли участие свыше 50 организаций. Кроме этого, независимые исследования

по просьбе NIST проводили несколько научных центров США и группа специалистов самого Института стандартизации.

По проблемам безопасности все алгоритмы-претенденты исследовались на предмет: общих известных криптоаналитических атак; криптоатак, специфичных для примененных в алгоритме примитивов; наличия нестойких ключей; минимального критического количества раундов, достаточного для стойкости к известным криптоатакам; отсутствия корреляций между входными данными или ключом и временем шифрования/дешифрования или потребляемой на этом процессе мощностью; четкости и ясности дизайна (дающих некоторую гарантию от закладки разработчиками недокументированных возможностей); использования в алгоритме частей опубликованных ранее криптоалгоритмов (и их стойкости или, наоборот, известных уязвимостей).

В плане практической реализации комитетом уделялось особое внимание следующему: оптимизации по скорости исполнения под множество современных архитектур и, в первую очередь, конечно, под 32-разрядные системы с достаточно большими ресурсами оперативной памяти; возможности оптимизации по размеру кода для чрезвычайно ограниченных в ресурсах систем; возможности распараллеливания вычислений в «разумных» пределах; учету различий и технических тонкостей существующих архитектур (как, например различному порядку хранения байт в многобайтовых числах); сравнению быстродействия прямых и обратных операций, то есть шифрования и дешифрования временным параметрам процедуры расширения ключей; изменению скорости работы алгоритма в случае применения 128-, 192- и 256-битных ключей, соответственно.

15 сентября 1999 г. NIST объявил о начале второго этапа конкурса. В течение полугода институтом принимались дополнительные исследования всех заинтересованных сторон, посвященные уже исключительно 5 финалистам (MARS, Serpent, Twofish, RC6 и Rijndael). Необходимо заметить, что во второй этап конкурса были допущены только те алгоритмы, в криптографической стойкости которых не было никаких сомнений. Поэтому все эти разработки, несомненно, являются важными в своей области.

Комплексный анализ всех достоинств и недостатков финалистов конкурса AES привел к объявлению новым стандартом блочного шифрования алгоритма Rijndael. Авторами алгоритма являются Йон Даемен (Joan Daeman) и Винсент Рюмен (Vincent Rijmen), начальные буквы фамилий которых и образуют название алгоритма. С алгоритма Rijndael по условиям конкурса были сняты все патентные ограничения — его использование в различных продуктах на территории США теперь контролируется только государственными нормативными актами этой страны.

В феврале 2001 г. на сайте http://crsc.nist.gov/encryption/aes/ была опубликована предварительная версия Федерального Стандарта Обработки Информации (Federal Information Processing Standard — FIPS). В течение 90-дневного периода

открытого обсуждения предварительная версия FIPS пересматривалась с учетом комментариев, после чего начался процесс исправлений и утверждения. Наконец 26 ноября 2001 г. была опубликована окончательная версия стандарта FIPS-197, описывающего новый стандарт шифрования AES. Согласно этому документу стандарт вступил в силу с 26 мая 2002 г.

3.3.2. Описание криптоалгоритма

Как уже говорилось ранее, алгоритм шифрования Rijndael был разработан двумя специалистами по криптографии из Бельгии. По сути шифр представляет собой разновидность модифицированной сети SPN (Substitution-Permutation Network) и является итерационным блочным шифром.

Сам алгоритм шифрования имеет переменную длину блоков и различные длины ключей. Длина ключа и длина блока могут быть равны независимо друг от друга 128, 192 или 256 битам. В стандарте AES определена длина блока данных, равная 128 битам.

Промежуточные результаты преобразований, выполняемых в рамках криптоалгоритма, называют **состояниями** (State). Состояние можно представить в виде прямоугольного массива байтов (рис. 3.8).

При размере блока, равном 128 битам, этот 16-байтовый массив (рис. 3.9, 3.10) имеет 4 строки и 4 столбца (каждая строка и каждый столбец в этом случае могут рассматриваться как 32-разрядные слова). Входные данные для шифра обозначаются как байты состояния в порядке S_{00}, S_{10}, S_{20}, S_{30}, s_{01}, s_{11}, s_{21}, s_{31}, ….

После завершения действия шифра выходные данные получаются из байтов состояния в том же порядке. В общем случае число столбцов N_b равно длине блока, деленной на 32.

На рис. 3.8 представлен пример 128-разрядного блока данных в виде массива State, где a_i — байт блока данных, а каждый столбец — одно 32-разрядное слово.

a_0	a_4	a_8	a_{12}
a_1	a_5	a_9	a_{13}
a_2	a_6	a_{10}	a_{14}
a_3	a_7	a_{11}	a_{15}

Рис. 3.8. ***Пример представления 128-разрядного блока данных***

S_{00}	S_{01}	S_{02}	S_{03}
S_{10}	S_{11}	S_{12}	S_{13}
S_{20}	S_{21}	S_{22}	S_{23}
S_{30}	S_{31}	S_{32}	S_{33}

Рис. 3.9. ***Формат представления блока входных данных***

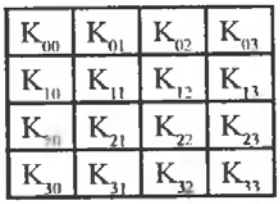


Рис. 3.10. ***Формат данных ключа шифрования***

Ключ шифрования также представлен в виде прямоугольного массива с четырьмя строками (рис. 3.10). Число столбцов N_k этого массива равно длине ключа, деленной на 32. В стандарте определены ключи всех трех размеров — 128 бит, 192 бита и 256 бит, то есть соответственно 4, 6 и 8 32-разрядных слова (или столбца — в табличной форме представления). В некоторых случаях ключ шифрования рассматривается как линейный массив 4-байтовых слов. Слова состоят из 4 байтов, которые находятся в одном столбце (при представлении в виде прямоугольного массива).

Число раундов N_r в алгоритме Rijndael зависит от значений N_b — длина блока и N_k — длина ключа, как показано в таблице 3.25.

Таблица 3.25

Число раундов Nr как функция от длины ключа Nk и длины блока Nb

N_r	$N_b = 4$	$N_b = 6$	$N_b = 8$
$N_k = 4$	10	12	14
$N_k = 6$	12	12	14
$N_k = 8$	14	14	14

3.3.2.1. Раундовое преобразование

Раунд состоит из четырех различных преобразований:

— замена байтов SubBytes() — побайтовой подстановки в S-блоках с фиксированной таблицей замен размерностью 8×256;

— сдвига строк ShiftRows() — побайтового сдвига строк массива State на различное количество байт;

— перемешивание столбцов MixColumns() — умножение столбцов состояния, рассматриваемых как многочлены над $GF(2^8)$, на многочлен третьей степени g(x) по модулю $x^4 + 1$;

— сложение с раундовым ключом AddRoundKey() — поразрядного XOR с текущим фрагментом развернутого ключа.

Алгоритм шифрования Rijndael оперирует байтами, которые рассматриваются как элементы конечного поля $GF(2^8)$. Элементами поля $GF(2^8)$ являются многочлены степени не более семи, которые могут быть заданы строкой своих коэффициентов.

Операция сложения над элементами поля представляет собой поразрядное сложение по модулю 2. Умножение в поле $GF(2^8)$ представляет собой операцию умножения со взятием результата по модулю неприводимого многочлена восьмой степени. Для алгоритма AES авторами был выбран многочлен $\varphi(x) = x^8 \oplus x^4 \oplus \oplus x^3 \oplus x \oplus 1$.

Преобразование SubBytes() представляет собой нелинейную замену байтов, выполняемую независимо с каждым байтом состояния. Таблицы замены S-блока являются инвертируемыми и построены из композиции следующих двух преобразований входного байта:

1) получение обратного элемента относительно умножения в поле $GF(2^8)$, нулевой элемент {00} переходит сам в себя;

2) применение преобразования над GF(2), определенного следующим образом:

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

Суть преобразования может быть описана уравнением $b_i' = b_i \oplus b_{(i+4)\bmod 8} \oplus \oplus b_{(i+5)\bmod 8} \oplus b_{(i+6)\bmod 8} \oplus b_{(i+7)\bmod 8} \oplus c_i$, где $c_0 = c_1 = c_5 = c_6 = 1$, $c_2 = c_3 = c_4 = c_7 = 0$, b_i и b_i' — соответственно исходное и преобразованное значение i-го бита, $i = \overline{0,7}$.

Применение описанного S-блока ко всем байтам состояния обозначается как SubBytes(State). Рис. 3.11 иллюстрирует применение преобразования SubBytes() к состоянию. На этом и последующих рисунках, описывающих раундовые операции алгоритма шифрования Rijndael, нижние два индекса состояния S обозначают его местоположение в таблице состояний, а верхние два индекса обозначают номер раунда, в котором происходит преобразование, и порядковый номер операции в данном раунде, соответственно. Так, например, запись $S_{31}^{r,1}$ обозначает блок данных, находящийся на пересечении четвертой строки и второго столбца раунда r после применения первой раундовой операции SubBytes(), а запись $S_{31}^{r,0}$ обозначает входной блок данных раунда r, находящийся в той же позиции, что и первый рассмотренный блок данных.

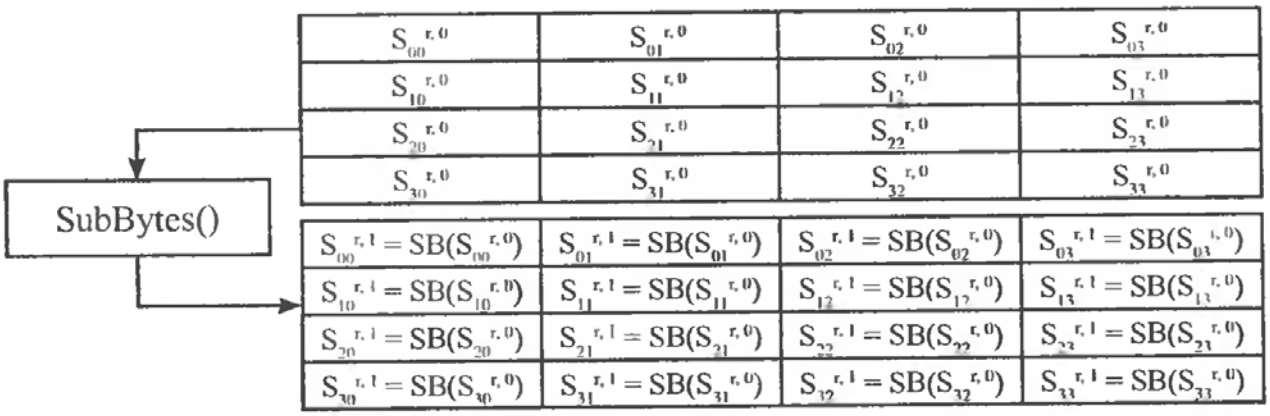


Рис. 3.11. ***Применение преобразования SubBytes()***

Логика работы S-блока при преобразовании байта {xy} отражена в таблице 3.26.

Таблица 3.26

Таблица замен S-блока

X	Y															
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**a**	**b**	**c**	**d**	**e**	**f**
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b_2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b_3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b_1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b_0	54	bb	16

Рассмотрим более подробно, как преобразуется байт {xy} в байт {x′y′}. Возьмем, к примеру, байт {23}. Если записать этот байт в виде многочлена, то получится $x^5 \oplus x \oplus 1$. Первым делом необходимо получить обратный ему многочлен относительно умножения в поле $GF(2^8)$. Как уже отмечалось ранее, операция умножения в поле $GF(2^8)$ является операцией умножения многочленов со взятием

результата по модулю неприводимого многочлена $\varphi(x)$ восьмой степени с использованием операции XOR (сложения по модулю 2) при приведении подобных членов. Так как авторами алгоритма шифрования Rijndael был выбран неприводимый многочлен $\varphi(x) = x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1$, то нам необходимо найти такой многочлен A, который бы при умножении на многочлен $x^5 \oplus x \oplus 1$ по модулю $\varphi(x) = x^8 \oplus x^4 \oplus \oplus x^3 \oplus x \oplus 1$ давал единицу, то есть

$$(x^5 \oplus x \oplus 1)\cdot A = 1 \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1). \quad (3.1)$$

Поделив неприводимый многочлен $\varphi(x)$ на многочлен $x^5 \oplus x \oplus 1$, легко можно определить, что целой частью от деления будет значение x^3, а в остатке останется многочлен $x \oplus 1$, то есть

$$(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \oplus x^3\cdot(x^5 \oplus x \oplus 1) = x \oplus 1. \quad (3.2)$$

В свою очередь, поделив многочлен $x^5 \oplus x \oplus 1$ на многочлен $x \oplus 1$, можно определить, что многочлен $(x \oplus 1)$ входит в многочлен $(x^5 \oplus x \oplus 1)$ $(x^4 \oplus x^3 \oplus x^2 \oplus x)$ раз и при этом в остатке остается единица. А значит,

$$(x^5 \oplus x \oplus 1) \oplus ((x^4 \oplus x^3 \oplus x^2 \oplus x) \cdot (x \oplus 1)) = 1. \quad (3.3)$$

Подставив (3.2) в (3.3), получим:

$$(x^5 \oplus x \oplus 1) \oplus ((x^4 \oplus x^3 \oplus x^2 \oplus x) \cdot ((x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \oplus x^3 \cdot (x^5 \oplus x \oplus 1))) = 1;$$

$$(x^5 \oplus x \oplus 1) \oplus ((x^4 \oplus x^3 \oplus x^2 \oplus x) \cdot (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)) \oplus ((x^4 \oplus x^3 \oplus x^2 \oplus x) \cdot x^3 \cdot (x^5 \oplus x \oplus 1)) = 1;$$

$$((x^4 \oplus x^3 \oplus x^2 \oplus x) \cdot (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)) \oplus ((x^7 \oplus x^6 \oplus x^5 \oplus x^4 \oplus 1) \cdot (x^5 \oplus x \oplus 1)) = 1$$

или

$$(x^7 \oplus x^6 \oplus x^5 \oplus x^4 \oplus 1) \cdot (x^5 \oplus x \oplus 1) = 1 \oplus (x^4 \oplus x^3 \oplus x^2 \oplus x) \cdot (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1).$$

А значит, искомым многочленом A будет являться многочлен $x^7 \oplus x^6 \oplus x^5 \oplus x^4 \oplus 1$ или значение {f1} в шестнадцатеричном виде, или значение 11110001 в двоичном.

Вторым этапом является преобразование следующего вида:

$b_0' = b_0 \oplus b_4 \oplus b_5 \oplus b_6 \oplus b_7 \oplus 1 = 1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 = 0$
$b_1' = b_1 \oplus b_5 \oplus b_6 \oplus b_7 \oplus b_0 \oplus 1 = 0 \oplus 1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 = 1$
$b_2' = b_2 \oplus b_6 \oplus b_7 \oplus b_0 \oplus b_1 \oplus 0 = 0 \oplus 1 \oplus 1 \oplus 1 \oplus 0 \oplus 0 = 1$
$b_3' = b_3 \oplus b_7 \oplus b_0 \oplus b_1 \oplus b_2 \oplus 0 = 0 \oplus 1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 = 0$
$b_4' = b_4 \oplus b_0 \oplus b_1 \oplus b_2 \oplus b_3 \oplus 0 = 1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 \oplus 0 = 0$
$b_5' = b_5 \oplus b_1 \oplus b_2 \oplus b_3 \oplus b_4 \oplus 1 = 1 \oplus 0 \oplus 0 \oplus 0 \oplus 1 \oplus 1 = 1$
$b_6' = b_6 \oplus b_2 \oplus b_3 \oplus b_4 \oplus b_5 \oplus 1 = 1 \oplus 0 \oplus 0 \oplus 1 \oplus 1 \oplus 1 = 0$
$b_7' = b_7 \oplus b_3 \oplus b_4 \oplus b_5 \oplus b_6 \oplus 0 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 1 \oplus 0 = 0$

Результатом проведенного преобразования является значение 00100110 или {26}. По таблице 3.26 можно убедиться, что на пересечении третьей строки и четвертого столбца действительно находится значение {26}.

Преобразование сдвига строк (ShiftRows) выглядит следующим образом: последние 3 строки состояния циклически сдвигаются влево на различное число байтов. Строка 1 сдвигается на C_1 байт, строка 2 — на C_2 байт, и строка 3 — на C_3 байт. Значение сдвигов C_1, C_2 и C_3 в Rijndael зависят от длины блока N_b. Их величины приведены в таблице 3.27.

Таблица 3.27

Величина сдвига для разной длины блоков

N_b	C_1	C_2	C_3
4	1	2	3
6	1	2	3
8	1	3	4

В стандарте AES, где определен единственный размер блока, равный 128 битам, $C_1 = 1$, $C_2 = 2$, $C_3 = 3$.

Операция сдвига последних трех строк состояния обозначена как ShiftRows(State). Рис. 3.12 показывает влияние преобразования на состояние.

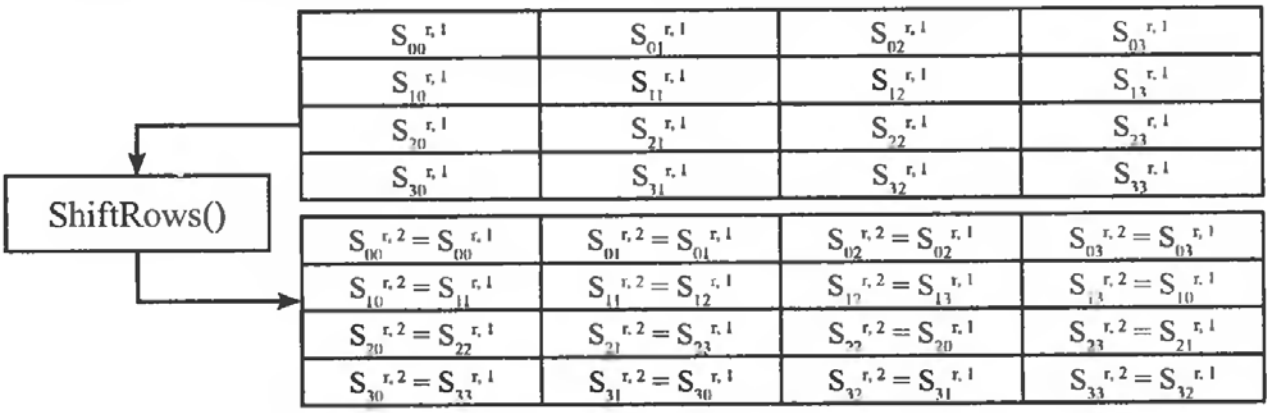


Рис. 3.12. ***Применение преобразования ShiftRows()***

Преобразование перемешивания столбцов (MixColumns) — это такое преобразование, при котором столбцы состояния рассматриваются как многочлены над $GF(2^8)$ и умножаются по модулю $x^4 + 1$ на многочлен $g(x)$, выглядящий следующим образом: $g(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$.

Это может быть представлено в матричном виде следующим образом:

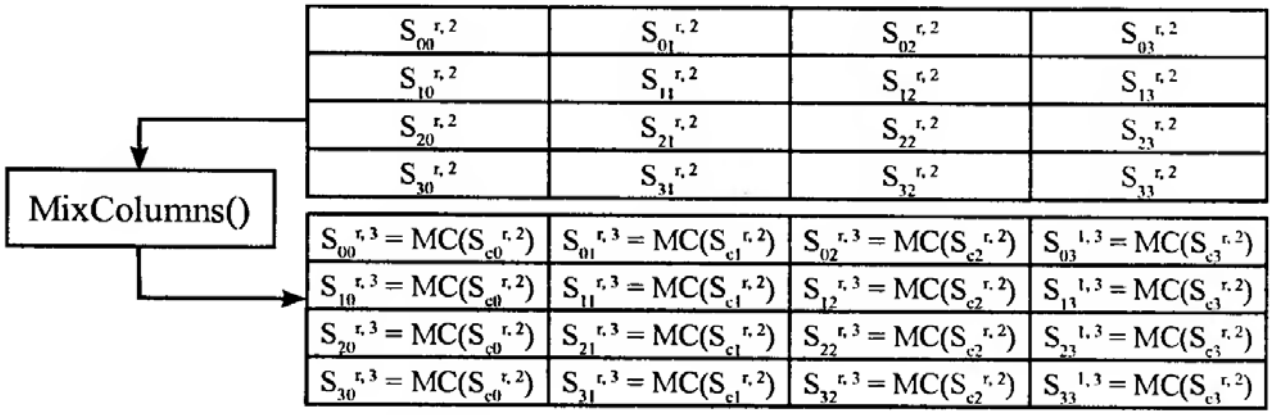
$$\begin{bmatrix} S'_{0c} \\ S'_{1c} \\ S'_{2c} \\ S'_{3c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0c} \\ S_{1c} \\ S_{2c} \\ S_{3c} \end{bmatrix}, \quad 0 \le c \le 3,$$

где c — номер столбца массива State.

В результате такого умножения байты столбца S_{0c}, S_{1c}, S_{2c}, S_{3c} заменяются соответственно на байты:

$$S'_{0c} = (\{02\} \cdot S_{0c}) \oplus (\{03\} \cdot S_{1c}) \oplus S_{2c} \oplus S_{3c},$$
$$S'_{1c} = S_{0c} \oplus (\{02\} \cdot S_{1c}) \oplus (\{03\} \cdot S_{2c}) \oplus S_{c3},$$
$$S'_{2c} = S_{0c} \oplus S_{1c} \oplus (\{02\} \cdot S_{2c}) \oplus (\{03\} \cdot S_{3c}),$$
$$S'_{3c} = (\{03\} \cdot S_{0c}) \oplus S_{1c} \oplus S_{2c} \oplus (\{02\} \cdot S_{3c}).$$

Применение этой операции ко всем четырем столбцам состояния обозначено как MixColumns(). Рис. 3.13 демонстрирует применение преобразования MixColumns() к столбцу состояния.



$S_{00}^{r,2}$	$S_{01}^{r,2}$	$S_{02}^{r,2}$	$S_{03}^{r,2}$
$S_{10}^{r,2}$	$S_{11}^{r,2}$	$S_{12}^{r,2}$	$S_{13}^{r,2}$
$S_{20}^{r,2}$	$S_{21}^{r,2}$	$S_{22}^{r,2}$	$S_{23}^{r,2}$
$S_{30}^{r,2}$	$S_{31}^{r,2}$	$S_{32}^{r,2}$	$S_{33}^{r,2}$

$S_{00}^{r,3} = MC(S_{c0}^{r,2})$	$S_{01}^{r,3} = MC(S_{c1}^{r,2})$	$S_{02}^{r,3} = MC(S_{c2}^{r,2})$	$S_{03}^{1,3} = MC(S_{c3}^{r,2})$
$S_{10}^{r,3} = MC(S_{c0}^{r,2})$	$S_{11}^{r,3} = MC(S_{c1}^{r,2})$	$S_{12}^{r,3} = MC(S_{c2}^{r,2})$	$S_{13}^{1,3} = MC(S_{c3}^{r,2})$
$S_{20}^{r,3} = MC(S_{c0}^{r,2})$	$S_{21}^{r,3} = MC(S_{c1}^{r,2})$	$S_{22}^{r,3} = MC(S_{c2}^{r,2})$	$S_{23}^{1,3} = MC(S_{c3}^{r,2})$
$S_{30}^{r,3} = MC(S_{c0}^{r,2})$	$S_{31}^{r,3} = MC(S_{c1}^{r,2})$	$S_{32}^{r,3} = MC(S_{c2}^{r,2})$	$S_{33}^{1,3} = MC(S_{c3}^{r,2})$

Рис. 3.13. ***Применение преобразования MixColumns()***

Рассмотрим операцию MixColumns() более подробно. Пусть нам необходимо преобразовать первый столбец:

$S_{00} = \{35\}$
$S_{10} = \{c4\}$
$S_{20} = \{5b\}$
$S_{30} = \{e2\}$

Первым делом необходимо представить байты S_{00}, S_{10}, S_{20} и S_{30} в виде многочленов в поле GF(2^8). Многочлен $(x^5 \oplus x^4 \oplus x^2 \oplus 1)$ будет эквивалентен байту $S_{00} = \{35\}$, многочлен $(x^7 \oplus x^6 \oplus x^2)$ — байту $S_{10} = \{c4\}$, многочлен $(x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$ — байту $S_{20} = \{5b\}$, и, наконец, многочлен $(x^7 \oplus x^6 \oplus x^5 \oplus x)$ — байту $S_{30} = \{e2\}$.

Для нахождения значения S_{00}' необходимо выполнить следующие действия.

1. Умножить многочлен $(x^5 \oplus x^4 \oplus x^2 \oplus 1)$ на байт {02}, или на x по модулю $(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$((x^5 \oplus x^4 \oplus x^2 \oplus 1) \cdot x) \mod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^6 \oplus x^5 \oplus x^3 \oplus x) \mod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = x^6 \oplus x^5 \oplus x^3 \oplus x$.

2. Умножить многочлен $(x^7 \oplus x^6 \oplus x^2)$ на байт {03}, или на многочлен $(x \oplus 1)$ по модулю $(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$((x^7 \oplus x^6 \oplus x^2) \cdot (x \oplus 1)) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^8 \oplus x^7 \oplus x^3 \oplus x^7 \oplus x^6 \oplus x^2)$
$\bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^8 \oplus x^6 \oplus x^3 \oplus x^2) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) =$
$= ((x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \oplus (x^6 \oplus x^4 \oplus x^2 \oplus x \oplus 1)) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) =$
$= x^6 \oplus x^4 \oplus x^2 \oplus x \oplus 1.$

3. Сложить по модулю 2 многочлены, полученные в пунктах 1 и 2, и многочлены $(x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$ и $(x^7 \oplus x^6 \oplus x^5 \oplus x)$:

$(x^6 \oplus x^5 \oplus x^3 \oplus x) \oplus (x^6 \oplus x^4 \oplus x^2 \oplus x \oplus 1) \oplus (x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \oplus (x^7 \oplus$
$\oplus x^6 \oplus x^5 \oplus x) = x^7 \oplus x^2$.

Искомым значением S_{00}' будет являться многочлен $x^7 \oplus x^2$, или в байтовом представлении значение {84}.

Теперь найдем значение S_{10}'. Для этого

1. Умножим многочлен $(x^7 \oplus x^6 \oplus x^2)$ на байт {02}, или на x по модулю $(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$((x^7 \oplus x^6 \oplus x^2) \cdot x) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = ((x^8 \oplus x^7 \oplus x^3)) \bmod (x^8 \oplus x^4 \oplus$
$\oplus x^3 \oplus x \oplus 1) = ((x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \oplus (x^7 \oplus x^4 \oplus x \oplus 1)) \bmod (x^8 \oplus x^4 \oplus$
$\oplus x^3 \oplus x \oplus 1) = x^7 \oplus x^4 \oplus x \oplus 1$;

2. Умножим многочлен $(x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$ на байт {03}, или на многочлен $(x \oplus 1)$ по модулю $(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$((x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \cdot (x \oplus 1)) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^7 \oplus x^5 \oplus x^4 \oplus$
$\oplus x^2 \oplus x \oplus x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = x^7 \oplus x^6 \oplus$
$\oplus x^5 \oplus x^3 \oplus x^2 \oplus 1$;

3. Сложим по модулю 2 многочлены, полученные в пунктах 1 и 2, и многочлены $(x^5 \oplus x^4 \oplus x^2 \oplus 1)$ и $(x^7 \oplus x^6 \oplus x^5 \oplus x)$:

$(x^7 \oplus x^4 \oplus x \oplus 1) \oplus (x^7 \oplus x^6 \oplus x^5 \oplus x^3 \oplus x^2 \oplus 1) \oplus (x^5 \oplus x^4 \oplus x^2 \oplus 1) \oplus (x^7 \oplus$
$\oplus x^6 \oplus x^5 \oplus x) = x^7 \oplus x^5 \oplus x^3 \oplus 1$.

Таким образом, искомым значением S_{10}' будет являться многочлен $x^7 \oplus x^5 \oplus x^3 \oplus 1$, или в байтовом представлении значение {a9}.

Для нахождения значения S_{20}' необходимо:

1. Умножить многочлен $(x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$ на байт {02}, или на x по модулю $(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$((x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \cdot x) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^7 \oplus x^5 \oplus x^4 \oplus x^2 \oplus x)$
$\bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = x^7 \oplus x^5 \oplus x^4 \oplus x^2 \oplus x$;

2. Умножить многочлен $(x^7 \oplus x^6 \oplus x^5 \oplus x)$ на байт {03}, или на многочлен $(x \oplus 1)$ по модулю $(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$((x^7 \oplus x^6 \oplus x^5 \oplus x) \cdot (x \oplus 1)) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^8 \oplus x^7 \oplus x^6 \oplus x^2 \oplus$
$\oplus x^7 \oplus x^6 \oplus x^5 \oplus x) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^8 \oplus x^5 \oplus x^2 \oplus x) \bmod$
$(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = ((x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \oplus (x^5 \oplus x^4 \oplus x^3 \oplus x^2 \oplus 1))$
$\bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = x^5 \oplus x^4 \oplus x^3 \oplus x^2 \oplus 1$;

3. Сложить по модулю 2 многочлены, полученные в пунктах 1 и 2, и многочлены $(x^5 \oplus x^4 \oplus x^2 \oplus 1)$ и $(x^7 \oplus x^6 \oplus x^2)$:

$(x^7 \oplus x^5 \oplus x^4 \oplus x^2 \oplus x) \oplus (x^5 \oplus x^4 \oplus x^3 \oplus x^2 \oplus 1) \oplus (x^5 \oplus x^4 \oplus x^2 \oplus 1) \oplus (x^7 \oplus$ $\oplus x^6 \oplus x^2) = x^6 \oplus x^5 \oplus x^4 \oplus x^3 \oplus x$.

Таким образом, мы нашли значение S_{20}', которым является многочлен $x^6 \oplus x^5 \oplus x^4 \oplus x^3 \oplus x$, или в байтовом представлении значение {7a}.

Оставшееся значение S_{30}' находим следующим образом:

1. Умножим многочлен $(x^7 \oplus x^6 \oplus x^5 \oplus x)$ на байт {02}, или на x по модулю $(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$((x^7 \oplus x^6 \oplus x^5 \oplus x) \cdot x) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^8 \oplus x^7 \oplus x^6 \oplus x^2) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = ((x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) \oplus (x^7 \oplus x^6 \oplus x^4 \oplus x^3 \oplus \oplus x^2 \oplus x \oplus 1)) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = x^7 \oplus x^6 \oplus x^4 \oplus x^3 \oplus x^2 \oplus x \oplus 1$;

2. Умножим многочлен $(x^5 \oplus x^4 \oplus x^2 \oplus 1)$ на байт {03}, или на многочлен $(x \oplus 1)$ по модулю $(x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$((x^5 \oplus x^4 \oplus x^2 \oplus 1) \cdot (x \oplus 1)) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^6 \oplus x^5 \oplus x^3 \oplus x \oplus \oplus x^5 \oplus x^4 \oplus x^2 \oplus 1) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = (x^6 \oplus x^4 \oplus x^3 \oplus x^2 \oplus x \oplus \oplus 1) \bmod (x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = x^6 \oplus x^4 \oplus x^3 \oplus x^2 \oplus x \oplus 1$;

3. Сложим по модулю 2 многочлены, полученные в пунктах 1 и 2, и многочлены $(x^7 \oplus x^6 \oplus x^2)$ и $(x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1)$:

$(x^7 \oplus x^6 \oplus x^4 \oplus x^3 \oplus x^2 \oplus x \oplus 1) \oplus (x^6 \oplus x^4 \oplus x^3 \oplus x^2 \oplus x \oplus 1) \oplus (x^7 \oplus x^6 \oplus x^2) \oplus$ $\oplus (x^6 \oplus x^4 \oplus x^3 \oplus x \oplus 1) = x^4 \oplus x^3 \oplus x^2 \oplus x \oplus 1$.

Итак, последним искомым значением S_{30}' является многочлен $x^4 \oplus x^3 \oplus x^2 \oplus$ $\oplus x \oplus 1$, или в байтовом представлении значение {1f}.

В операции добавления подключа (AddRoundKey) подключ добавляется к данным с помощью операции побитового сложения по модулю два (операция XOR). На рис. 3.15 показана операция сложения данных с раундовым подключом.

На рис. 3.14 наглядно показано преобразование исходного столбца S_{i0} в столбец S_{i0}' с помощью операции MixColumns().

Как видно из рис. 3.15, каждый блок подключа K, помимо двух нижних индексов, определяющих его местонахождения, имеет еще один индекс, определяющий номер используемого подключа. Так как первой операцией при зашифровании с помощью алгоритма Rijndael является операция сложения данных с подключом, то в i-м раунде будет использоваться (i + 1)-подключ.

Подключи вырабатываются из исходного секретного ключа шифрования с помощью алгоритма выработки ключей (Key Shedule). Длина подключа (в 32-разрядных словах) равна длине блока N_b.

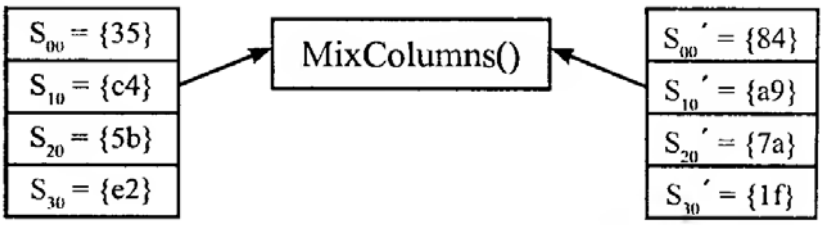


Рис. 3.14. ***Преобразование столбца с помощью операции MixColumns()***

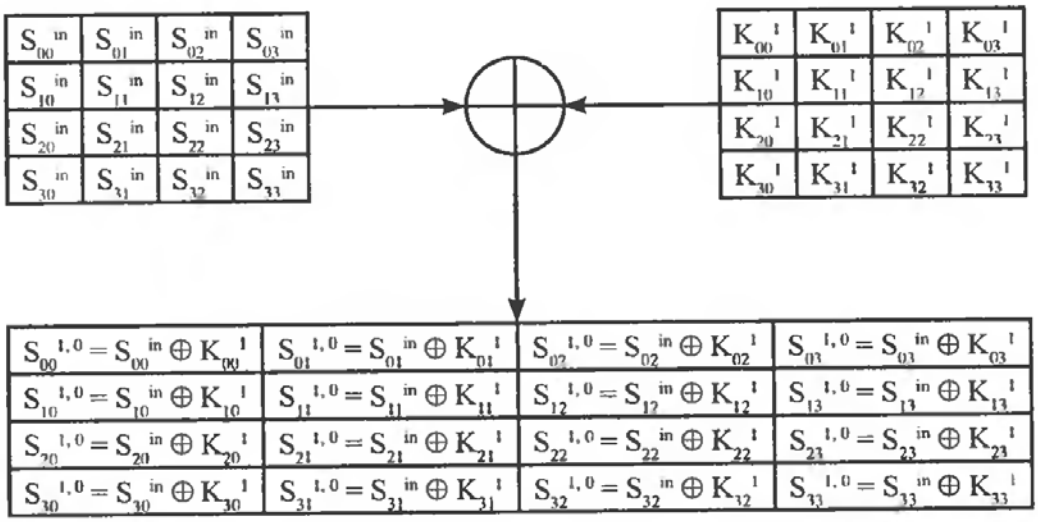


Рис. 3.15. ***Операция сложения данных с раундовым ключом***

3.3.2.2. Алгоритм выработки ключей

Подключи, используемые в каждом раунде шифрования, получаются из исходного секретного ключа шифрования с помощью алгоритма выработки подключей. Он содержит два компонента: расширение ключа (Key Expansion) и выбор подключа (Round Key Selection). Основополагающие принципы алгоритма выглядят следующим образом:

— чтобы определить общее число битов подключей, необходимо длину блока умножить на число раундов и добавить 1 (например, для длины блока 128 бит и 10 раундов требуется 1408 бит подключей);

— ключ шифрования расширяется в расширенный ключ (Expanded Key);

— подключи берутся из расширенного ключа следующим образом: первый подключ содержит первые N_b слов, второй — следующие N_b слов и т. д.

Расширенный ключ (Key Expansion) в Rijndael представляет собой линейный массив w[i] из $N_b(N_r + 1)$ 4-байтовых слов, где $0 < i < N_b(N_r + 1)$.

Первые N_k слов содержат ключ шифрования. Все остальные слова определяются рекурсивно из слов с меньшими индексами. Алгоритм выработки подключей зависит от величины N_k.

Первые N_k слов заполняются ключом шифрования (рис. 3.16). Каждое последующее слово w[i] получается посредством сложения по модулю два предыдущего слова w[i – 1] и слова на N_k позиций ранее, то есть $w[i - N_k]$:

$$w[i] = w[i - 1] \oplus w[i - N_k].$$

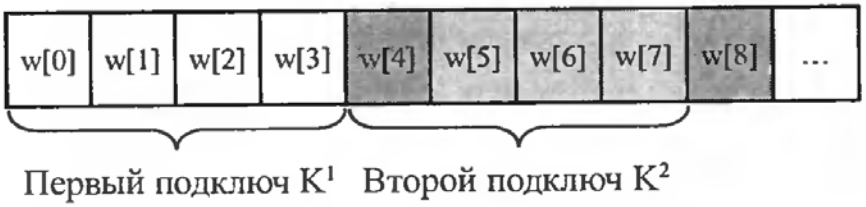


Рис. 3.16. ***Массив раундовых подключей***

Для слов, позиция которых кратна N_k, перед операцией сложения по модулю два применяется преобразование к w[i – 1], а затем еще прибавляется раундовая константа R_{const}. Преобразование реализуется с помощью двух дополнительных функций: RotWord(), осуществляющей побайтовый сдвиг 32-разрядного слова по формуле $\{a_0, a_1, a_2, a_3\} \rightarrow \{a_1, a_2, a_3, a_0\}$, и SubWord(), осуществляющей побайтовую замену с использованием S-блока функции SubBytes(). Значение $R_{const}[j]$ равно 2^{j-1}. Значение w[i] в этом случае равно:

$$w[i] = \text{SubWord}(\text{RotWord}(w[i-1])) \oplus R_{const}[i/N_k] \oplus w[i-N_k].$$

Раундовый ключ i получается из слов массива раундового ключа от $W[N_b i]$ и до $W[N_b(i+1)]$.

3.3.2.3. Функция зашифрования

Шифр Rijndael состоит из следующих основных операций:

— начального добавления раундового ключа;

— N_{r-1} раундов;

— заключительного раунда, в котором отсутствует операция MixColumns().

На вход алгоритма подаются блоки данных State. В ходе преобразований содержимое блока изменяется и на выходе образуется шифртекст, организованный опять же в виде блоков State, как показано на рис. 3.17, где $N_b = 4$, in_m и out_m — байты соответственно входного и выходного блоков, $m = \overline{0,15}$, s_{ij} — байт, находящийся на пересечении i-й строки и j-го столбца массива State, $i = \overline{0,3}$, $j = \overline{0,3}$.

Перед началом первого раунда происходит суммирование по модулю 2 с первым подключом шифрования, затем выполняется преобразование массива байтов State в течение 10, 12 или 14 раундов в зависимости от длины ключа. Последний раунд несколько отличается от предыдущих тем, что не задействует функцию перемешивания байт в столбцах MixColumns().

На рис. 3.17 показан общий вид алгоритма шифрования Rijndael. Блоки входных данных вместо двух верхних индексов имеют индекс in, а блоки выходных данных — индекс out.

3.3.2.4. Функция обратного дешифрования

Если вместо SubBytes(), ShiftRows(), MixColumns() и AddRoundKey() в обратной последовательности выполнить инверсные им преобразования, то можно построить алгоритм дешифрования данных. При этом порядок использования раундовых ключей является обратным по отношению к тому, который используется при зашифровании.

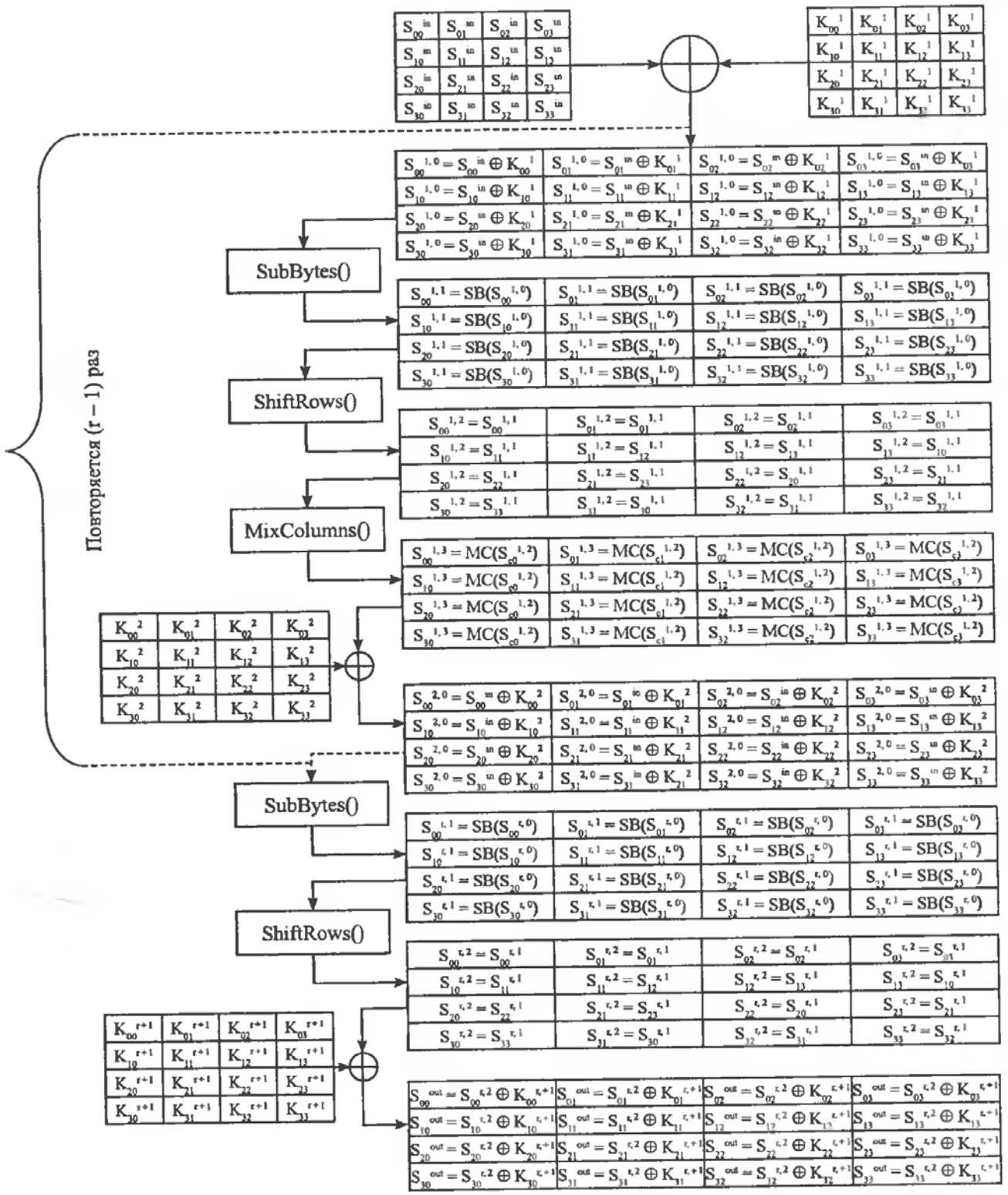


Рис. 3.17. ***Алгоритм шифрования Rijndael***

Функция AddRoundKey() обратна сама себе, учитывая свойства используемой в ней операции XOR.

Логика работы инверсного S-блока InvSubBytes при преобразовании байта {xy} отражена в таблице 3.28 и заключается в следующем. Если при прямом

преобразовании байт {23} заменялся на байт {26} (как это было рассмотрено нами выше), то при обратном преобразовании байт {26} будет заменяться на байт {23}.

Таблица 3.28

Таблица замен инверсного S-блока

x	y															
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b_2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b_3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	c2	f9	37	e8	1c	75	df	6e
a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c	1f	dd	a8	33	88	07	c7	31	b_1	12	10	59	27	80	ec	5f
d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e	a0	e0	3b	4d	ae	2a	f5	b_0	c8	eb	bb	3c	83	53	99	61
f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

В преобразовании InvShiftRows последние 3 строки состояния сдвигаются вправо на различное число байтов. Строка 1 сдвигается на C_1 байт, строка 2 — на C_2 байт, и строка 3 — на C_3 байт. Значение сдвигов C_1, C_2 и C_3 зависят от длины блока N_b. Их величины приведены в таблице 3.27.

В преобразовании InvMixColumns столбцы состояния рассматриваются как многочлен над GF(2^8) и умножаются по модулю $x^4 + 1$ на многочлен $g^{-1}(x)$, выглядящий следующим образом:

$$g^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}.$$

Многочлен $g^{-1}(x)$ является мультипликативным обратным многочлена $g(x)$, используемого в операции прямого преобразования MixColumns(), то есть для многочленов $g(x)$ и $g^{-1}(x)$ выполняется соотношение:

$$g(x) \cdot g^{-1}(x) = \{01\}$$

Умножение столбцов состояния на многочлен $g^{-1}(x)$ по модулю $x^4 + 1$ может быть представлено в матричном виде следующим образом:

$$\begin{bmatrix} S'_{0c} \\ S'_{1c} \\ S'_{2c} \\ S'_{3c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 9d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0c} \\ S_{1c} \\ S_{2c} \\ S_{3c} \end{bmatrix}.$$

В результате на выходе получаются следующие байты:

$$S'_{0c} = (\{0e\} \cdot S_{0c}) \oplus (\{0b\} \cdot S_{1c}) \oplus (\{0d\} \cdot S_{2c}) \oplus (\{09\} \cdot S_{3c}),$$
$$S'_{1c} = (\{09\} \cdot S_{0c}) \oplus (\{0e\} \cdot S_{1c}) \oplus (\{0b\} \cdot S_{2c}) \oplus (\{0d\} \cdot S_{3c}),$$
$$S'_{2c} = (\{0d\} \cdot S_{0c}) \oplus (\{09\} \cdot S_{1c}) \oplus (\{0e\} \cdot S_{2c}) \oplus (\{0b\} \cdot S_{3c}),$$
$$S'_{3c} = (\{0b\} \cdot S_{0c}) \oplus (\{0d\} \cdot S_{1c}) \oplus (\{09\} \cdot S_{2c}) \oplus (\{0e\} \cdot S_{3c}).$$

3.3.2.5. Функция прямого дешифрования

Алгоритм обратного дешифрования, описанный выше, имеет порядок приложений операций-функций, обратный порядку операций в алгоритме прямого дешифрования, но использует те же параметры (развернутый ключ). Однако некоторые свойства алгоритма шифрования Rijndael позволяют применить для дешифрования тот же порядок приложения функций (обратных, используемых для зашифрования) за счет изменения некоторых параметров, а именно — развернутого ключа.

Два следующих свойства позволяют применить алгоритм прямого дешифрования.

Порядок приложения функций SubBytes() и ShiftRows() не играют роли. То же самое верно и для операций InvSubBytes() и InvShiftRows(). Это происходит потому, что функция SubBytes() и InvSubBytes() работают с байтами, а операции ShiftRows() и InvShiftRows() сдвигают целые байты, не затрагивая их значения.

Операция MixColumns() является линейной относительно входных данных, что означает: InvMixColumns(State XOR RoundKey) = InvMixColumns(State) XOR InvMixColumns(RoundKey).

Эти свойства функций алгоритма шифрования позволяют изменить порядок применения функций InvSubBytes() и InvShiftRows(). Функции AddRoundKey() и InvMixColumns() также могут быть применены в обратном порядке, но при условии, что столбцы (32-битные слова) развернутого ключа дешифрования предварительно пропущены через функцию InvMixColumns().

В таблице 3.29 приведена процедура зашифрования, а также два эквивалентных варианта процедуры дешифрования при использовании двухраундового варианта Rijndael. Первый вариант функции дешифрования представляет собой обычную инверсию функции зашифрования. Второй вариант функции зашифрования получен из первого после изменения порядка следования операций в трех парах преобразований:

InvShiftRows – InvSubBytes(дважды) и AddRoundKey – InvMixColumns.

Таблица 3.29

Последовательность преобразований в двухраундовом варианте Rijndael

Функция зашифрования двухраундового варианта Rijndael	Функция обратного дешифрования двухраундового варианта Rijndael	Эквивалентная функция прямого дешифрования двухраундового варианта Rijndael
AddRoundKey	AddRoundKey	AddRoundKey
SubBytes	InvShiftRows	InvSubBytes
ShiftRows	InvSubBytes	InvShiftRows
MixColumns	AddRoundKey	InvMixColumns
AddRoundKey	InvMixColumns	AddRoundKey
SubBytes	InvShiftRows	InvSubBytes
ShiftRows	InvSubBytes	InvShiftRows
AddRoundKey	AddRoundKey	AddRoundKey

Очевидно, что результат преобразования при переходе от исходной к обратной последовательности выполнения операций в указанных парах не изменится.

Видно, что процедура зашифрования и второй вариант процедуры дешифрования совпадают с точностью до порядка использования раундовых ключей (при выполнении операции AddRoundKey), таблиц замен (при выполнении операции SubBytes) и матриц преобразования (при выполнении операции MixColumns и InvMixColumns). Данный результат легко обобщить и на любое другое число раундов.

3.4. Алгоритм шифрования S_AES

3.4.1. Математическое обоснование алгоритма

Алгоритм шифрования Simple AES, или S_AES, разработан в учебных целях и предназначен для обучения будущих криптографов и криптоаналитиков. В основу алгоритма S_AES заложены те же основные принципы, что и в новом стандарте шифрования США AES. Данный алгоритм оперирует полубайтами, которые рассматриваются как элементы поля $GF(2^4)$.

Элементами поля $GF(2^4)$ являются многочлены степени не более трех, которые могут быть заданы строкой своих коэффициентов. Если представить байт в виде:

$$\{a^3, a^2, a^1, a^0\}, \text{ где } a^i \in \{0, 1\}, \text{ i меняется от 0 до 3.}$$

Например, полубайту {0101} (или {5} в шестнадцатеричной системе счисления) соответствует многочлен $x^2 \oplus 1$.

Операция сложения над элементами поля представляет собой поразрядное сложение по модулю 2. Умножение в поле $GF(2^4)$ представляет собой операцию умножения со взятием результата по модулю неприводимого многочлена четвертой степени. Для алгоритма S_AES мы взяли многочлен $\varphi(x) = x^4 \oplus x \oplus 1$.

Раундовые преобразования в S_AES оперируют байтами. Байту может быть поставлен в соответствие многочлен a(x) с коэффициентами из $GF(2^4)$ степени не более единицы:

$$a(x) = a_1 x \oplus a_0.$$

При сложении двух многочленов с коэффициентами из $GF(2^4)$ получаем:

$$a(x) \oplus b(x) = (a_1 \oplus b_1)x \oplus (a_0 \oplus b_0).$$

Если же мы хотим перемножить два многочлена $a(x) = a_1 x \oplus a_0$ и $b(x) = b_1 x \oplus b_0$, то в результате получится многочлен:

$$c(x) = a(x) \otimes b(x) = (a_1 x \oplus a_0) \otimes (b_1 x \oplus b_0) = a_1 b_1 x^2 \oplus a_0 b_1 x \oplus b_0 a_1 x \oplus a_0 b_0 = a_1 b_1 x^2 \oplus$$
$$\oplus (a_0 b_1 \oplus b_0 a_1)x \oplus a_0 b_0.$$

Для того чтобы результат умножения мог быть представлен байтом, необходимо взять результат по модулю многочлена степени не более 2. Нами был взят многочлен $x^2 \oplus 1$. Таким образом, результатом c(x) умножения $\otimes$ двух многочленов по модулю $x^2 \oplus 1$:

$$c(x) = a(x) \otimes b(x)$$

будет многочлен

$$c(x) = c_1 x \oplus c_0,$$

где

$$c_0 = a_0 b_0 \oplus a_1 b_1,$$

$$c_1 = a_1 b_0 \oplus a_0 b_1.$$

Итак, если записать в матричной форме, то:

$$\begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = \begin{bmatrix} a_0 & a_1 \\ a_1 & a_0 \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \end{bmatrix}.$$

Пусть $c(x) = c_1 x \oplus c_0$. Умножению на x многочлена c(x) с коэффициентами из $GF(2^4)$ по модулю $x^2 \oplus 1$ соответствует циклический сдвиг полубайтов в пределах байта в сторону старшего полубайта, так как

$$x \otimes c(x) = c \otimes (c_1 x \oplus c_0) = c_1 x^2 \oplus c_0 x = c_1(x^2 \oplus 1) \oplus c_0 x \oplus c_1 = c_0 x \oplus c_1.$$

3.4.2. Описание алгоритма S_AES

Алгоритм S_AES оперирует 16-битовыми входными данными и данными секретного ключа такой же длины. Входные данные алгоритма S_AES обозначают-

ся как полубайты состояния S_{00}, S_{10}, S_{01} и S_{11} и представляют собой массив из двух строк и двух столбцов:

S_{00}	S01
S_{10}	S_{11}

Ключ шифрования состоит из четырех полубайтов K_{00}, K_{10}, K_{01} и K_{11} и также представляется в виде массива из двух строк и двух столбцов:

K_{00}	K_{01}
K_{10}	K_{11}

Алгоритм S_AES состоит из двух раундов. Перед началом первого раунда происходит сложение исходных данных с первым подключом K_1.

Всего в алгоритме S_AES используются три подключа. Есть три варианта их использования. В первом случае можно использовать в качестве раундового подключа заранее определенный секретный ключ K, то есть в этом случае в каждом раунде будет использоваться один и тот же подключ K. Во втором случае можно заранее определить три секретных подключа и использовать их при шифровании. И, наконец, третьим вариантом, пожалуй, самым сложным, но в то же время и самым правильным, является извлечение раундового подключа из исходного секретного ключа K.

Первый раунд состоит из четырех преобразований, первое из которых заключается в замене полубайтов с помощью S-блока. Второе преобразование заключается в сдвиге строк на один полубайт, третье представляет собой перемешивание столбцов с помощью умножения столбцов данных на многочлен первой степени по модулю $x^2 \oplus 1$. И, наконец, последнее четвертое преобразование представляет собой поразрядное сложение данных с раундовым подключом по модулю 2.

Второй раунд алгоритма S_AES содержит те же преобразования, что и первый раунд за исключением операции перемешивания столбцов. В последнем втором раунде алгоритма S_AES она отсутствует так же, как и в стандарте шифрования AES.

Рассмотрим каждую из вышеперечисленных операций более подробно.

3.4.2.1. Замена полубайтов Sub Half-Bytes*()

С помощью операции **Sub Half-Bytes*()** осуществляется нелинейная замена полубайтов данных. Операция **Sub Half-Bytes*()** применяется к каждому полубайту данных независимо от остальных данных. Таблица замены S-блока является инвертируемой и построена из следующих двух преобразований входного полубайта:

1) получение обратного элемента относительно умножения (то есть мультипликативного обратного) в поле $GF(2^4)$;

2) применение преобразования над GF(2^4), определенного следующим образом:

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Иными словами, вышеописанное преобразование можно представить в виде уравнения:

$$b_i' = b_i \oplus b_{(i+2) \bmod 4} \oplus b_{(i+3) \bmod 4} \oplus c_i,$$

где $c_0 = c_3 = 1$ и $c_1 = c_2 = 0$, а b_i и b_i' — соответственно исходное и преобразованное значение i-го бита.

На рис. 3.18 показано применение преобразования **Sub Half-Bytes*()** к шифруемым данным. Сам S-блок замены приведен в таблице 3.30.

Таблица 3.30

x	Y			
	00	**01**	**10**	**11**
00	9	e	5	l
01	8	b	d	a
10	6	7	f	3
11	c	4	0	2

При рассмотрении полубайта {xy}, x обозначает два старших бита полубайта, а y — два младших. Так, например, полубайт {d} (или {1101} в двоичной системе счисления) будет заменен на полубайт {4}, стоящий на пересечении третьей строки и второго столбца таблицы 3.30.

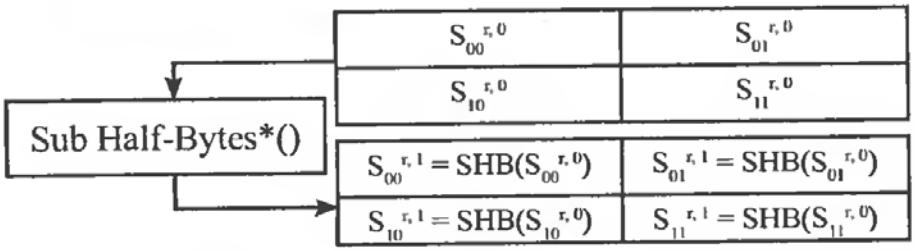


Рис. 3.18. ***Применение операции Sub Half-Bytes*()***

На рис. 3.18 и последующих рисунках, описывающих раундовые операции алгоритма шифрования S_AES, нижние два индекса состояния S обозначают его местоположение в таблице состояний, а верхние два индекса обозначают номер раунда, в котором происходит преобразование, и порядковый номер операции в данном раунде, соответственно. Так, например, запись $S_{10}^{r,1}$ обозначает блок

данных, находящийся на пересечении второй строки и первого столбца раунда r после применения первой раундовой операции **Sub Half-Bytes*()**, а запись $S_{10}^{r,0}$ обозначает входной блок данных раунда r, находящийся в той же позиции, что и предыдущий блок данных.

Рассмотрим более детально, как происходит преобразование с помощью операции **Sub Half-Bytes*()**. Пусть нам надо преобразовать полубайт {a}. Первым делом, как уже отмечалось выше, необходимо найти мультипликативный обратный ему элемент в поле $GF(2^4)$. Полубайт {a} можно представить в виде многочлена $(x^3 \oplus x)$, а значит, нам необходимо найти такой многочлен A, что:

$$(x^3 \oplus x) \cdot A = 1 \bmod (x^4 \oplus x \oplus 1).$$

Поделив неприводимый многочлен $(x^4 \oplus x \oplus 1)$ на многочлен $(x^3 \oplus x)$, легко можно определить, что целой частью от деления будет значение x, а в остатке останется многочлен $(x^2 \oplus x \oplus 1)$, то есть

$$(x^4 \oplus x \oplus 1) \oplus x \cdot (x^3 \oplus x) = x^2 \oplus x \oplus 1. \quad (3.4)$$

В свою очередь, поделив многочлен $(x^3 \oplus x)$ на многочлен $(x^2 \oplus x \oplus 1)$, можно определить, что многочлен $(x^2 \oplus x \oplus 1)$ входит в многочлен $(x^3 \oplus x)$ $(x \oplus 1)$ раз, и при этом в остатке остается многочлен $(x \oplus 1)$. А значит:

$$(x^3 \oplus x) \oplus ((x \oplus 1) \cdot (x^2 \oplus x \oplus 1)) = x \oplus 1. \quad (3.5)$$

Подставив (3.4) в (3.5), получим:

$(x^3 \oplus x) \oplus ((x \oplus 1) \cdot ((x^4 \oplus x \oplus 1) \oplus x \cdot (x^3 \oplus x))) = x \oplus 1;$

$(x^3 \oplus x) \oplus ((x \oplus 1) \cdot (x^4 \oplus x \oplus 1)) \oplus ((x \oplus 1) \cdot x \cdot (x^3 \oplus x)) = x \oplus 1;$

$$((x \oplus 1) \cdot (x^4 \oplus x \oplus 1)) \oplus ((x^2 \oplus x \oplus 1) \cdot (x^3 \oplus x)) = x \oplus 1. \quad (3.6)$$

Поделив многочлен $(x^2 \oplus x \oplus 1)$ на многочлен $(x \oplus 1)$, можно определить, что многочлен $(x \oplus 1)$ входит в многочлен $(x^2 \oplus x \oplus 1)$ (x) раз, и при этом в остатке остается единица, то есть:

$$(x^2 \oplus x \oplus 1) \oplus ((x \oplus 1) \cdot x) = 1. \quad (3.7)$$

Подставив (3.4) и (3.6) в (3.7), получим:

$((x^4 \oplus x \oplus 1) \oplus x \cdot (x^3 \oplus x)) \oplus ((((x \oplus 1) \cdot (x^4 \oplus x \oplus 1)) \oplus ((x^2 \oplus x \oplus 1) \cdot (x^3 \oplus \oplus x))) \cdot x) = 1;$

$(x^4 \oplus x \oplus 1) \oplus (x \cdot (x^3 \oplus x)) \oplus ((x^2 \oplus x) \cdot (x^4 \oplus x \oplus 1)) \oplus ((x^3 \oplus x^2 \oplus x) \cdot (x^3 \oplus \oplus x)) = 1;$

$((x^4 \oplus x \oplus 1) \cdot (x^2 \oplus x \oplus 1)) \oplus ((x^3 \oplus x^2) \cdot (x^3 \oplus x)) = 1,$

или

$$(x^3 \oplus x^2) \cdot (x^3 \oplus x) = 1 \oplus ((x^4 \oplus x \oplus 1) \cdot (x^2 \oplus x \oplus 1)).$$

А значит, искомым многочленом A будет являться многочлен $(x^3 \oplus x^2)$, или значение {c} в шестнадцатеричном виде, или значение {1100} в двоичном.

Вторым этапом является преобразование следующего вида:

$$b_0' = b_0 \oplus b_2 \oplus b_3 \oplus 1 = 0 \oplus 1 \oplus 1 \oplus 1 = 1;$$
$$b_1' = b_1 \oplus b_3 \oplus b_0 \oplus 0 = 0 \oplus 1 \oplus 0 \oplus 0 = 1;$$

$$b_2' = b_2 \oplus b_0 \oplus b_1 \oplus 0 = 1 \oplus 0 \oplus 0 \oplus 0 = 1;$$
$$b_3' = b_3 \oplus b_1 \oplus b_2 \oplus 1 = 1 \oplus 0 \oplus 1 \oplus 1 = 1.$$

Результатом проведенного преобразования является значение {1111}, или {f}. По таблице 3.10 можно убедиться, что на пересечении третьей строки и третьего столбца действительно находится значение {f}.

3.4.2.2. Операция сдвига строк Shift Rows*()

Эта операция осуществляется аналогично одноименной операции в алгоритме шифрования AES, то есть первая строка массива данных остается без изменений, а вторая — циклически сдвигается влево на один полубайт. На рис. 3.19 показано применение операции **Shift Rows*()** к шифруемым данным.

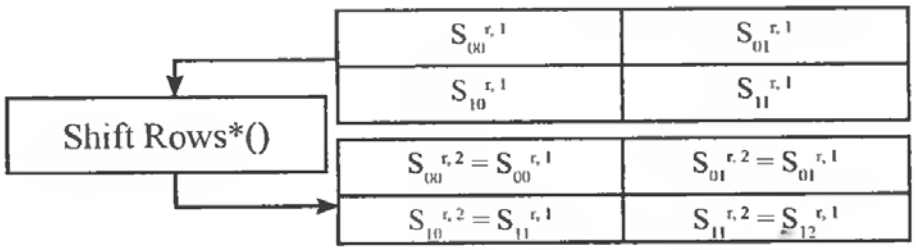


Рис. 3.19. ***Преобразование данных с помощью операции Shift Rows*()***

3.4.2.3. Операция перемешивания столбцов Mix Columns*()

При выполнении этой операции столбцы данных рассматриваются как многочлены над $GF(2^4)$ и умножаются по модулю $x^2 \oplus 1$ на многочлен g(x), выглядящий следующим образом:

$$g(x) = \{2\}x \oplus \{3\}.$$

Это может быть представлено в матричном виде следующим образом:

$$\begin{bmatrix} S_{0c}' \\ S_{1c}' \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} S_{0c} \\ S_{1c} \end{bmatrix}, \quad 0 \le c \le 1,$$

где c — номер столбца массива данных. В результате такого умножения полубайты столбцов S_{0c} и S_{1c} заменяются соответственно на полубайты:

$$S_{0c}' = (\{3\} \cdot S_{0c}) \oplus (\{2\} \cdot S_{1c}),$$
$$S_{1c}' = (\{2\} \cdot S_{0c}) \oplus (\{3\} \cdot S_{1c}).$$

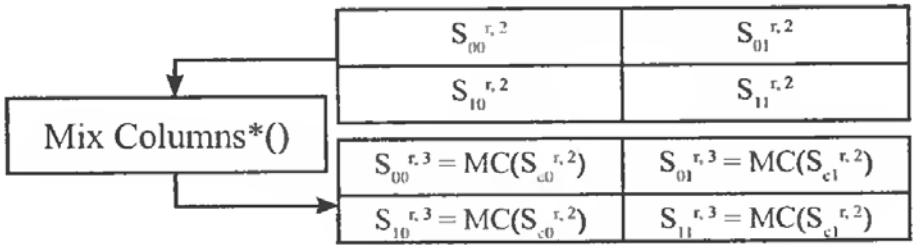


Рис. 3.20. ***Применение операции* Mix Columns*() *к преобразуемым данным***

На рис. 3.20 показано применение операции **Mix Columns*()** к данным, преобразуемым в ходе зашифрования.

Рассмотрим операцию **Mix Columns*()** более подробно. Пусть нам надо преобразовать столбец:

$S_{00} = \{c\}$
$S_{10} = \{9\}$

Первым делом необходимо представить полубайты S_{00} и S_{10} в виде многочленов в поле $GF(2^4)$. Многочлен $(x^3 \oplus x^2)$ будет эквивалентен байту $S_{00} = \{c\}$, а многочлен $(x^3 \oplus 1)$ — байту $S_{10} = \{9\}$.

Для нахождения значения S_{00}' необходимо выполнить следующие действия.

1. Умножить многочлен $(x^3 \oplus x^2)$ на полубайт $\{3\}$, или на $(x \oplus 1)$ по модулю $(x^4 \oplus x \oplus 1)$:

$$((x^3 \oplus x^2) \cdot (x \oplus 1)) \bmod (x^4 \oplus x \oplus 1) = (x^4 \oplus x^3 \oplus x^2) \bmod (x^4 \oplus x \oplus 1) = (x^4 \oplus x^2) \bmod (x^4 \oplus x \oplus 1) = x^2 \oplus x \oplus 1.$$

2. Умножить многочлен $(x^3 \oplus 1)$ на полубайт $\{2\}$, или на x по модулю $(x^4 \oplus x \oplus 1)$:

$$((x^3 \oplus 1) \cdot x) \bmod (x^4 \oplus x \oplus 1) = (x^4 \oplus x) \bmod (x^4 \oplus x \oplus 1) = 1.$$

3. Сложить по модулю 2 многочлены, полученные в пунктах 1 и 2:

$$(x^2 \oplus x \oplus 1) \oplus 1 = x^2 \oplus x.$$

Искомым значением S_{00}' будет являться многочлен $x^2 \oplus x$, или полубайт $\{6\}$.

Теперь найдем значение S_{10}'. Для этого

1. Умножим многочлен $(x^3 \oplus x^2)$ на полубайт $\{2\}$, или на x по модулю $(x^4 \oplus x \oplus 1)$:

$$((x^3 \oplus x^2) \cdot x) \bmod (x^4 \oplus x \oplus 1) = (x^4 \oplus x^3) \bmod (x^4 \oplus x \oplus 1) = x^3 \oplus x \oplus 1;$$

2. Умножим многочлен $(x^3 \oplus 1)$ на полубайт $\{3\}$, или на $(x \oplus 1)$ по модулю $(x^4 \oplus x \oplus 1)$:

$$((x^3 \oplus 1) \cdot (x \oplus 1)) \bmod (x^4 \oplus x \oplus 1) = (x^4 \oplus x \oplus x^3 \oplus 1) \bmod (x^4 \oplus x \oplus 1) = x^3;$$

3. Сложим по модулю 2 многочлены, полученные в пунктах 1 и 2:

$$(x^3 \oplus x \oplus 1) \oplus x^3 = x \oplus 1.$$

Таким образом, искомым значением S_{10}' будет являться многочлен $x \oplus 1$, или полубайт $\{3\}$.

На рис. 3.21 наглядно показано преобразование исходного столбца S_{00} в столбец S_{10}' с помощью операции **Mix Columns*()**.

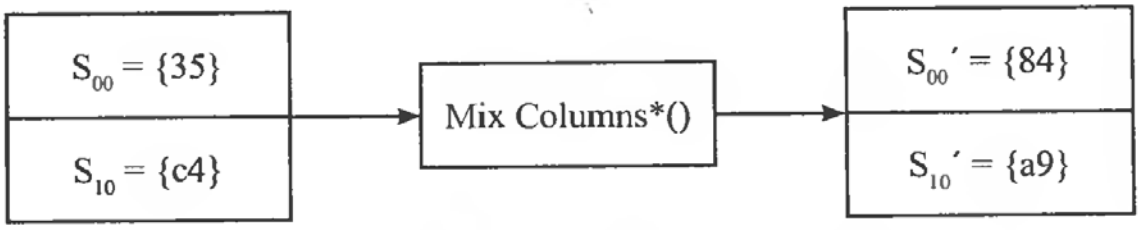


Рис. 3.21. ***Преобразование столбца с помощью операции*** **Mix Columns*()**

В операции добавления подключа **(Add Round Key*)** подключ добавляется к данным с помощью операции побитового сложения по модулю два (операция XOR). На рис. 3.22 показана операция сложения данных с раундовым подключом.

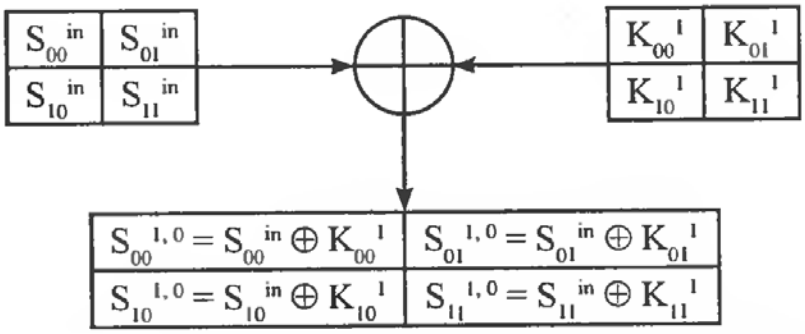


Рис. 3.22. ***Операция AddRoundKey*() сложения данных с раундовым подключом***

Обратите внимание, что на рис. 3.22 исходные данные, складываемые с ключом, имеют верхний индекс in, который означает, что эти данные являются исходными данными, подлежащими зашифрованию с помощью алгоритма шифрования S_AES. Верхний индекс при обозначении полубайтов ключа определяет номер используемого подключа.

На рис. 3.23 показан полный алгоритм S_AES. При дешифровании данных вместо операции Sub Half-Bytes*() выполняется инверсное ей преобразование. Неизменным остается преобразование Shift Rows*(). Дело в том, что массив данных в алгоритме шифрования S_AES содержит всего две строки и два столбца. При зашифровании с помощью операции Shift Rows*() происходит циклический сдвиг влево на один полубайт. А значит, при дешифровании необходимо будет провести циклический сдвиг вправо также на один полубайт. Но так как в строке содержится всего два элемента, то, в принципе, все равно в какую сторону их циклически сдвигать. Поэтому при дешифровании смело можно пользоваться операцией Shift Rows*(). Кроме того, все операции используются в обратном порядке, также как и раундовые подключи, то есть сначала K_3, потом K_2 и последний — K_1. На рис. 3.24 приведен алгоритм дешифрования данных с помощью алгоритма S_AES.

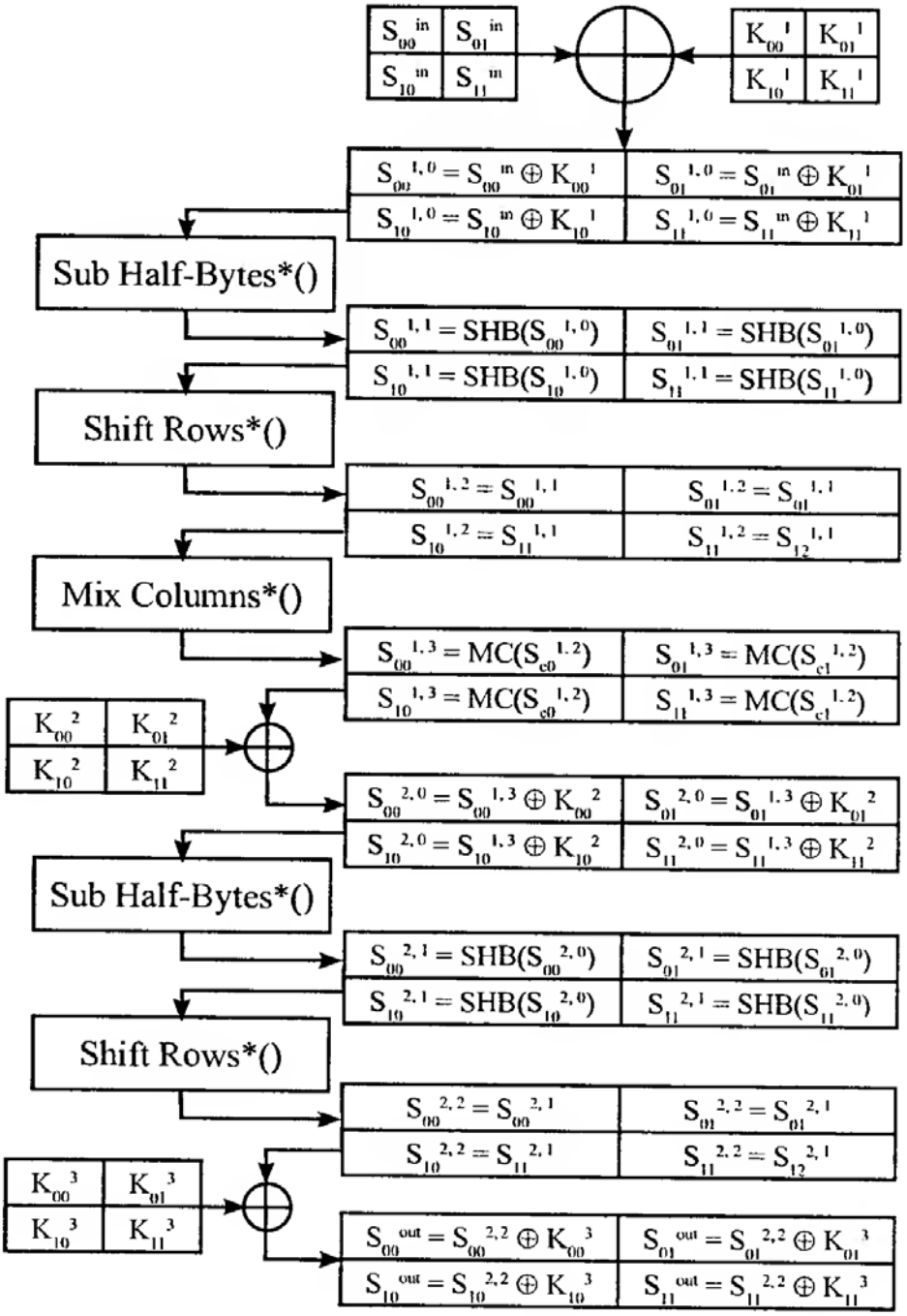


Рис. 3.23. ***Алгоритм шифрования S_AES***

Таблица 3.31

x	Y			
	00	**01**	**10**	**11**
00	e	3	f	B
01	d	2	8	9
10	4	0	7	5
11	c	6	1	a

Преобразование с помощью операции **ISub Half-Bytes*()** осуществляет замену полубайтов так, как указано в таблице 3.31. Логика работы данной операции заключается в следующем. Если при прямом преобразовании полубайт {9} заменялся на полубайт {7}, то при обратном преобразовании полубайт {7} будет заменяться на полубайт {9}.

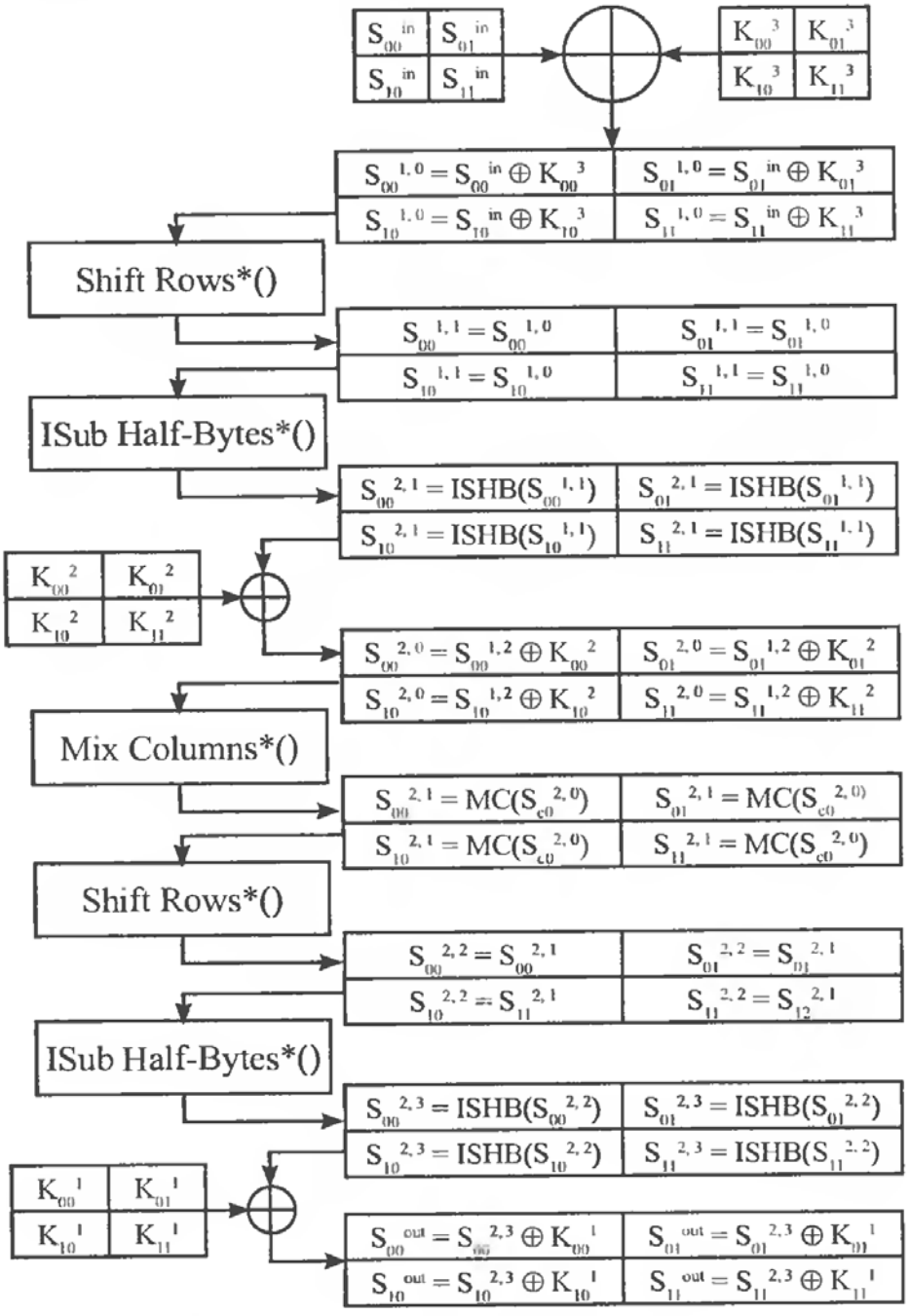


Рис. 3.24. ***Алгоритм дешифрования S_AES***

Неизменным остается и преобразование Mix Columns*(). При дешифровании столбцы состояния рассматриваются как многочлен над $GF(2^4)$ и должны умножаться по модулю $x^2 \oplus 1$ на многочлен $g^{-1}(x)$. Многочлен $g^{-1}(x)$ является мульти-

пликативным обратным многочлена g(x), используемого в операции прямого преобразования Mix Columns*(), то есть для многочленов g(x) и $g^{-1}(x)$ выполняется соотношение:

$$g(x) \cdot g^{-1}(x) = \{01\}.$$

Это соотношение будет выполняться только в том случае, если

$$g^{-1}(x) = g(x) = \{2\}x \oplus \{3\},$$

поэтому мы можем использовать при дешифровании то же преобразование Mix Columns*(), что использовали и при шифровании.

3.4.3. Алгоритм выработки подключей

Раундовые ключи вырабатываются из исходного 16-битового секретного ключа по следующей схеме. Исходный 16-битовый ключ, как уже говорилось ранее, можно представить в виде четырех полубайтов K_{00}, K_{10}, K_{01}, K_{11}. Эти четыре полубайта формируют первый раундовый подключ: K_{00}^1, K_{10}^1, K_{01}^1, K_{11}^1. Второй и третий раундовые подключи можно получить по следующим формулам:

$$K_{00}^r = \text{Sub Half-Bytes*}(K_{11}^{r-1}) \oplus K_{00}^{r-1};$$
$$K_{10}^r = \text{Sub Half-Bytes*}(K_{01}^{r-1}) \oplus K_{10}^{r-1} \oplus 2^{r-2}; \quad (3.8)$$
$$K_{01}^r = K_{00}^r \oplus K_{01}^{r-1};$$
$$K_{11}^r = K_{10}^r \oplus K_{11}^{r-1},$$

где верхний индекс r — номер вырабатываемого подключа, а преобразование Sub Half-Bytes*() осуществляется по таблице 3.30.

3.4.4. Пример преобразования данных с помощью алгоритма S_AES

Пусть дан входной блок данных: {7, e, 3, b}, и дан секретный ключ K: {3, e, f, a}. Запишем эти данные в виде двумерных массивов:

Массив исходных данных

7	3
e	b

Ключ

3	f
e	a

Для начала необходимо получить раундовые подключи. Первым раундовым подключом будет являться сам секретный ключ, то есть

$$K_{00}1 = 3; \qquad K_{01}1 = f;$$
$$K_{10}1 = e; \qquad K_{11}1 = a.$$

Воспользовавшись формулами (3.8), найдем:

$K_{00}^2 = \text{Sub Half-Bytes*}(K_{11}^1) \oplus K_{00}^1 = \text{Sub Half-Bytes*}(a) \oplus 3 = 1 \oplus 3 = c;$

K_{10}^{2} = Sub Half-Bytes*(K_{01}^{1}) ⊕ K_{10}^{1} ⊕ 20= Sub Half-Bytes*(f) ⊕ e ⊕ 1 = 2 ⊕ e ⊕ ⊕ 1 = d;

$K_{01}^{2} = K_{00}^{2} \oplus K_{01}^{1} = c \oplus f = 3$;

$K_{11}^{2} = K_{10}^{2} \oplus K_{11}^{1} = d \oplus a = 7$;

K_{00}^{3} = Sub Half-Bytes*(K_{11}^{2}) ⊕ K_{00}^{2} = Sub Half-Bytes*(7) ⊕ c = a ⊕ c = 6;

K_{10}^{3} = Sub Half-Bytes*(K_{01}^{2}) ⊕ K_{10}^{2} ⊕ 21= Sub Half-Bytes*(3) ⊕ d ⊕ 2 = 1 ⊕ ⊕ d ⊕ 2 = e;

$K_{01}^{3} = K_{00}^{3} \oplus K_{01}^{2} = 6 \oplus 3 = 5$;

$K_{11}^{2} = K_{10}^{3} \oplus K_{11}^{2} = e \oplus 7 = 9$.

Таким образом, получили три раундовых подключа:

Первый подключ K_1

3	f
e	a

Второй подключ K_2

c	3
d	7

Третий подключ K_3

6	5
e	9

При зашифровании с помощью алгоритма S_AES начальной операцией является сложение исходных данных с первым раундовым подключом. В результате получим

7 ⊕ 3 = 4	3 ⊕ f = c
e ⊕ e = 0	b ⊕ a = 1

Затем произведем замену полубайтов с помощью таблицы 3.10 и получим

8	c
9	e

Теперь сдвинем вторую строку на один полубайт влево

8	c
e	9

И произведем перемешивание столбцов с помощью операции Mix Columns*(). Для начала представим табличные данные в виде многочленов. Итак, $S_{00} = \{8\} = x^3$, $S_{10} = \{e\} = x^3 \oplus x^2 \oplus x$, $S_{01} = \{c\} = x^3 \oplus x^2$, $S_{11} = \{9\} = x^3 \oplus 1$.

Теперь можно приступить к нахождению новых элементов массива:

$S_{00}' = (\{3\} \cdot S_{00}) \oplus (\{2\} \cdot S_{10}) = ((x \oplus 1) \cdot x^3) \oplus (x \cdot (x^3 \oplus x^2 \oplus x)) = (x^4 \oplus x^3) \oplus (x^4 \oplus \oplus x^3 \oplus x^2) = x^3 \oplus x \oplus 1 \oplus x^3 \oplus x^2 \oplus x \oplus 1 = x^2 = \{4\}$;

$S_{10}' = (\{2\} \cdot S_{00}) \oplus (\{3\} \cdot S_{10}) = (x \cdot x^3) \oplus ((x \oplus 1) \cdot (x^3 \oplus x^2 \oplus x)) = (x^4) \oplus (x^4 \oplus x^3 \oplus \oplus x^2 \oplus x^3 \oplus x^2 \oplus x) = (x^4) \oplus (x^4 \oplus x) = x \oplus 1 \oplus 1 = x = \{2\}$;

$S_{01}' = (\{3\} \cdot S_{01}) \oplus (\{2\} \cdot S_{11}) = ((x \oplus 1) \cdot (x^3 \oplus x^2)) \oplus (x \cdot (x^3 \oplus 1)) = (x^4 \oplus x^3 \oplus x^3 \oplus \oplus x^2) \oplus (x^4 \oplus x) = (x^4 \oplus x^2) \oplus (x^4 \oplus x) = x^2 \oplus x \oplus 1 \oplus 1 = x^2 \oplus x = \{6\}$;

$S_{11}' = (\{2\} \cdot S_{01}) \oplus (\{3\} \cdot S_{11}) = (x \cdot (x^3 \oplus x^2)) \oplus ((x \oplus 1) \cdot (x^3 \oplus 1)) = (x^4 \oplus x^3) \oplus$
$\oplus (x^4 \oplus x \oplus x^3 \oplus 1) = x^3 \oplus x \oplus 1 \oplus x^3 = x \oplus 1 = \{3\}.$

Получили следующий массив преобразованных данных:

4	6
2	3

Следующим шагом является сложение полученных данных со вторым раундовым подключом, то есть

$4 \oplus c = 8$	$6 \oplus 3 = 5$
$2 \oplus d = f$	$3 \oplus 7 = 4$

Теперь еще раз произведем замену полубайтов с помощью таблицы 3.10 и получим:

6	b
2	2

Теперь сдвинем вторую строку на один полубайт влево:

6	b
8	2

Заключительной операцией является сложение с третьим раундовым подключом:

$6 \oplus 6 = 0$	$b \oplus 5 = e$
$8 \oplus e = 6$	$2 \oplus 9 = b$

Итак, в результате шифрования данных с помощью алгоритма шифрования S_AES мы получили в результате значение {0, 6, e, b}.

3.5. Криптоанализ упрощенного варианта алгоритма шифрования Rijndael

Как известно, алгоритм шифрования Rijndael является алгоритмом блочного шифрования. При этом входной блок данных и секретный ключ этого алгоритма может иметь длину 128, 192 или 256 бит независимо друг от друга. Мы будем рассматривать тот вариант, когда входной блок данных и секретный ключ имеют одинаковую длину, равную 128 битам.

Ррассмотрим два вида анализа алгоритма шифрования Rijndael: анализ на основе невозможных дифференциалов и анализ с помощью новой атаки типа «Квадрат». Эти виды анализа основаны на следующем свойстве преобразования MixColumn: если два входных значения преобразования имеют различие только

в одном байте, то соответствующие им выходные значения будут иметь различия во всех четырех байтах. Мы можем использовать это свойство для анализа четырех раундов алгоритма шифрования Rijndael. Если два исходных текста, подвергающихся зашифрованию, имеют различие только в одном байте перед первым преобразованием MixColumn, то после преобразования MixColumn данные будут иметь различия во всем столбце. А в следующем раунде, благодаря преобразованию ShiftRow, перед преобразованием MixColumn данные будут иметь различия в одном байте каждого столбца. Поэтому после второго преобразования MixColumn данные будут иметь различия во всех 16 байтах. Отсюда вытекают следующие интересные свойства.

1. Рассмотрим 256 входных сообщений, имеющих одинаковые значения во всех байтах, кроме одного. Соответственно этот единственный байт может принимать одно из 256 возможных значений. Мы выяснили, что перед преобразованием MixColumn в третьем раунде входные значения будут иметь различия во всех 16 байтах, то есть каждый байт может принимать 256 возможных значений. Так как мы рассматриваем все 256 возможных входных значений, то перед преобразованием MixColumn третьего раунда сумма по модулю два одного и того же байта всех 256 рассматриваемых текстов будет равна нулю. Это связано с тем, что для каждого из рассматриваемых текстов на входе преобразования MixColumn будет находиться единственно возможное значение, и оно не повторится ни для одного из остальных рассматриваемых текстов. А значит, получается, что выходные байты образуют сумму: $0 \oplus 1 \oplus 2 \oplus \ldots \oplus 255 = 0$. Так как преобразование MixColumn является линейным, то данное свойство будет применимо и к выходным байтам преобразования.

2. Свойство невозможных дифференциалов заключается в следующем. Рассмотрим два открытых текста, которые имеют различие в одном байте. Если соответствующие им шифртексты после четвертого раунда шифрования имеют одинаковые значения в первом, шестом, одиннадцатом и шестнадцатом байтах, то перед преобразованием ShiftRow в четвертом раунде соответствующие данные имели одинаковые значения в первом, пятом, девятом и тринадцатом байтах (то есть в первом столбце). Следовательно, после преобразования MixColumn в третьем раунде шифрования данные имели одинаковые значения в первом столбце. А значит, и перед преобразованием MixColumn данные также имели одинаковые значения в первом столбце. Однако в соответствии со всем, что было сказано ранее, на этом этапе данные должны иметь различия во всех байтах. Поэтому при зашифровании таких пар открытых текстов не могли быть получены соответствующие шифртексты. И именно на этом свойстве основан анализ пяти раундов алгоритма шифрования Rijndael, описанный ниже.

3.5.1. Анализ алгоритма с помощью атаки типа «Квадрат»

Рассмотрим анализ четырехраундового алгоритма шифрования Rijndael. Возьмем 256 входных сообщений, имеющих одинаковые значения во всех байтах, кроме одного. Так как операция SubBytes является простой операцией замены одного значения на другое, то после этой операции первого раунда данные все еще будут иметь различия в том единственном байте.

Пусть у нас есть два входа преобразования MixColumn (a, b, c, d) и (a′, b, c, d), тогда соответствующие им выходные значения будут иметь разность ($02_x \cdot (a - a')$, $01_x \cdot (a - a')$, $01_x \cdot (a - a')$, $03_x \cdot (a - a')$), где все операции производятся в поле $GF(2^8)$. Следовательно, выходные значения преобразования MixColumn будут иметь отличия во всех четырех байтах. А значит, в нашем случае все четыре байта столбца могут принять любое значение из 256 возможных. Как мы уже выяснили ранее, после преобразования третьего раунда MixColumn выходные данные при сложении друг с другом по модулю 2 в результате дадут ноль. А значит, и перед операцией SubBytes четвертого раунда шифрования это свойство сохранится.

Таким образом, анализ алгоритма шифрования Rijndael может быть проведен с помощью выполнения следующих действий:

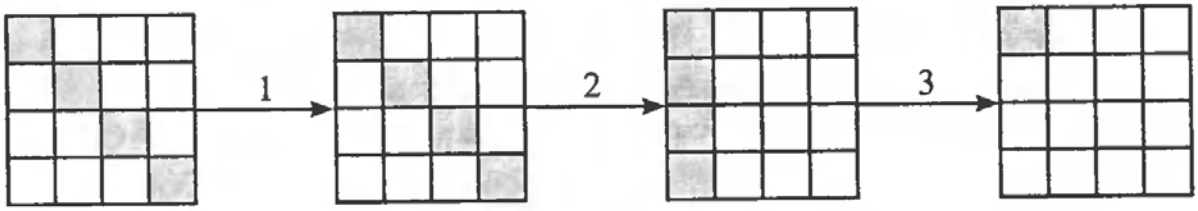
1. Необходимо выбрать 256 открытых текстов, которые будут иметь различия только в первом байте;
2. Получить соответствующие им шифртексты, зашифровав их с помощью четырех раундов шифрования Rijndael на секретном ключе;
3. Предположить значение (то есть взять одно из возможных) первого байта подключа четвертого раунда шифрования;
4. Используя 256 известных шифртекстов, получить 256 значений первого байта на входе преобразования SubBytes четвертого раунда;
5. Если сумма полученных 256 значений по модулю два равна нулю, то предположенное значение байта подключа верно;
6. Используя этот же принцип, найти остальные байты подключа четвертого раунда шифрования;
7. Зная значение подключа четвертого раунда, извлечь значение исходного секретного ключа.

3.5.2. Анализ пяти раундов алгоритма шифрования Rijndael с помощью *невозможных* дифференциалов

Рассмотрим алгоритм шифрования Rijndael, состоящий из 4 циклов. Если пара открытых текстов имеет различие только лишь в одном байте, то соответствующие им шифртексты не могут иметь одинаковые значения в следующих комбинациях байтов: (1, 6, 11, 16), (2, 7, 12, 13), (3, 8, 9, 14) или (4, 5, 10, 15). Это происходит вследствие того, что перед преобразованием MixColumn в первом раунде различие имеется только в одном байте, после преобразования — во всем столбце,

а после преобразования MixColumn второго раунда — во всех 16 байтах. С другой стороны, если шифртексты имеют одинаковые значения в одной из запрещенных комбинаций байтов, то после преобразования MixColumn третьего раунда данные имеют одинаковые значения в одном из столбцов, а это значит, что и до преобразования MixColumn третьего раунда данные также имеют одинаковые значения в этом столбце. Следовательно, после преобразования MixColumn второго раунда в данных осталось четыре байта, равных между собой. Это противоречит тому, что мы выяснили ранее: после преобразования MixColumn данные должны иметь различия во всех байтах. Заметим, что вероятность возникновения такого противоречия при рассмотрении случайной пары текстов равна 2^{-30} [13].

Таким образом, анализ пяти раундов алгоритма шифрования Rijndael основан на исключении неправильных подключей первого раунда, при использовании которых в последних четырех раундах появляется значение невозможной разности. Нам необходимо рассмотреть такие пары текстов, которые на входе второго раунда шифрования будут иметь различие только в первом байте. Это возможно в том случае, когда на вход алгоритма шифрования поступает пара текстов, имеющая различие в первом, шестом, одиннадцатом и шестнадцатом байтах (см. рис. 3.25). На рис. 3.25 серым цветом обозначены байты, в которых тексты не имеют сходства.



1 — операция сложения с раундовым ключом AddRoundKey();
2 — операция циклического сдвига ShiftRows();
3 — операция преобразования столбцов MixColumns().

Рис. 3.25. ***Анализ алгоритма шифрования Rijndael***

Зная это, можно провести анализ данного алгоритма шифрования по следующей схеме:

1. Выбрать 2^{63} пар открытых текстов, таких, что разность в первом, шестом, одиннадцатом и шестнадцатом байтах у них не будет равна нулю;
2. Зашифровать их на одном и том же секретном ключе;
3. Выбрать те пары текстов, у которых разность шифртекстов имеет ненулевые значения в одной из невозможных комбинаций байтов. Таких пар будет примерно $2^{63} \cdot 2^{-30} \approx 2^{33}$;
4. Для каждой такой пары открытых текстов (P, P*) выполнить следующие действия:
   а) предположить значение первого, шестого, одиннадцатого и шестнадцатого байтов первого подключа K^0;

b) вычислить, чему будут равны первый, пятый, девятый и тринадцатый байты выхода первого раунда для шифртекстов P и P*, зашифровав их с помощью выбранных байтов подключа K^0;
c) если разность первых вычисленных байтов отлична от нуля, а разность остальных вычисленных байтов равна нулю, то байты подключа K^0 выбраны неверно;
d) продолжать выполнять действия пунктов a, b и c до тех пор, пока не будет найдено верное значение байтов первого подключа.

Остальные байты первого подключа могут быть найдены с использованием аналогичной схемы, однако рассматривать в этом случае надо не первый столбец второго раунда, а все остальные.

3.6. Европейский криптопроект NESSIE

Первого января 2000 г. стартовал 40-месячный европейский проект NESSIE (New European Schemes for Signature, Integrity and Encryption).

В отличие от проекта AES, криптопроект NESSIE ориентирован не только на рассмотрение блочных алгоритмов шифрования, но также и поточных алгоритмов, алгоритмов цифровой подписи, хэш-сумм и асимметричных шифров. В плане представления наиболее интересно то, что за проектом не стоит ни одно правительственное учреждение. Устроители конкурса основной своей задачей считают не только мощное криптографическое исследование претендентов, но и знакомство с ними всех заинтересованных сторон.

Для подачи заявок на участие в проекте организаторы отвели лето 2000 г. На момент первой конференции (ноябрь того же года) в NESSIE было зарегистрировано 39 алгоритмов. Среди них было 26 симметричных алгоритмов шифрования [15, 16]:

a) 17 алгоритмов блочного шифрования, что не удивительно, так как в начале проведения конкурса AES внимание к алгоритмам блочного шифрования значительно возросло. Их можно разделить следующим образом:
   — шесть 64-битовых алгоритмов блочного шифрования: CS-Cipher, Hierocrypt-L1, IDEA, Khazad, $MYSTY_1$ и Nimbus;
   — семь 128-битовых алгоритмов блочного шифрования: Anubis, Camellia, Grand Cru, Hierocrypt-3, Noekeon, Q и CS2000 (ни один из этих семи алгоритмов не участвовал в конкурсе AES);
   — один 160-битовый алгоритм блочного шифрования Shacal;
   — три блочных алгоритма шифрования с различной длиной блока данных: NUSH (64, 128 или 256 бит), RC6 (начиная со 128 бит), SAFER + + (64 и 128 бит);
   — шесть поточных алгоритмов шифрования: BMGL, Levithan, LILI-128, SNOW, SOBER-t16 и SOBER-t32;

— два MAC-алгоритма: Two-Track-MKAC и UMAC;
— одна хэш-функция, устойчивая к столкновениям: Whirpool;

b) 13 ассиметричных алгоритмов шифрования:
— пять ассиметричных алгоритмов шифрования: ACE Encrypt, ECIES, EPOC, PSEC и RSA-OAEP (алгоритмы EPOC и PSEC имеют три различных варианта);
— семь алгоритмов цифровой подписи: ACE Sign, ECDSA, ESIGN, FLASH, QUARTZ, RSA-PSS и SFLASH;
— одна схема идентификации: GPS.

Примечательно то, что в конкурсе приняли участие авторы практически со всех концов света. Преобладающее число разработок было представлено авторами Европы (в частности, авторами Франции, Бельгии, Швеции и Швейцарии), также большое число работ было представлено из Северной Америки (Канада и США) и Азии, где практически все работы были из Японии. Также были работы Южной Америки и Австралии. Конечно, нельзя не упомянуть, что в классе блочных шифров представлена и Россия (алгоритмом NUSH — детищем компании «ЛАНкрипто»).

После изучения всех претендентов были отобраны алгоритмы, перешедшие во второй тур конкурса, начавшийся 24 сентября 2001 г. Разработчикам алгоритмов было разрешено внести небольшие изменения в их алгоритмы. При этом изменения должны были улучшить алгоритм шифрования и не повлиять на результаты проведенных анализов стойкости алгоритма. Ниже приведены алгоритмы, прошедшие во второй тур конкурса. В алгоритмы, помеченные знаком *, авторами были внесены изменения. Среди алгоритмов блочного шифрования ко второму туру отбора были допущены:

— IDEA;
— Khazad*;
— $MISTY_1$;
— SAFER + + 64, SAFER + + 128;
— Camellia;
— RC6;
— Shacal.

Среди поточных алгоритмов шифрования:

— SOBER-t16, SOBER-t32;
— SNOW*;
— BMGL*.

Однако весной 2002 г. стало ясно, что ни один из этих поточных алгоритмов шифрования не отвечает требованиям NESSIE. Алгоритмы SOBER-t16, SOBER-t32 и SNOW* имеют недостаточную стойкость, а алгоритм шифрования BMGL* достаточно медленный (более, чем в 10 раз медленнее AES).

Во второй тур также перешли представленные MAC-алгоритмы, хэш-функция Whirlpool* и схема идентификации GPS*. Среди систем с открытым ключом во второй тур попали:

— ACE-KEM*;
— EPOC-2*;
— PSEC-KEM*;
— ECIES*;
— RSA-OAEP*.

И, наконец, среди алгоритмов цифровой подписи, во второй тур прошли:

— ECDSA;
— ESIGN*;
— RSA-PSS;
— SFLASH*;
— QUARTZ*.

Как и в конкурсе AES, в криптопроекте NESSIE в обсуждении алгоритмов, представленных во втором туре, могли принять все желающие. Все отзывы о представленных алгоритмах принимались организаторами проекта до середины декабря 2002 г. После этого были объявлены победители криптопроекта NESSIE. Ими стали:

— Camellia;
— Khazad;
— $MISTRY_1$;
— IDEA;
— SAFER + + 64;
— SAFER + + 128;
— Shacal;
— RC6.

И, наконец, в 2003 г. были объявлены победители. Ими стали три блочных алгоритма шифрования: Camellia, $MISTRY_1$ и Shacal [NESSIE].

3.7. Контрольные вопросы

1. Дайте краткое описание процесса шифрования данных с помощью алгоритма шифрования DES.
2. Опишите процедуру извлечения подключа для алгоритма шифрования DES.
3. По какому принципу работают S-блоки замены в алгоритме шифрования DES?
4. Влияют ли на криптографическую стойкость начальная IP и конечная IP^{-1} перестановки?

5. В чем заключается криптографический смысл операции перестановки с расширением в алгоритме шифрования DES?
6. Какие различия имеют процесс зашифрования и процесс дешифрования для алгоритма шифрования DES?
7. Опишите принцип работы алгоритма шифрования S-DES.
8. Опишите принцип работы алгоритма шифрования ГОСТ 28147-89.
9. Перечислите пять финалистов конкурса AES.
10. По какому принципу происходит замена байтов SubBytes() в алгоритме шифрования Rijndael.
11. Опишите процесс зашифрования данных с помощью алгоритма шифрования Rijndael.
12. Перечислите основные особенности алгоритма шифрования Rijndael.
13. Что вы знаете о криптопроекте NESSIE?

Глава 4. ДРУГИЕ ИЗВЕСТНЫЕ АЛГОРИТМЫ БЛОЧНОГО ШИФРОВАНИЯ

4.1. RC5

Криптоалгоритм RC5 был разработан Ривестом в 1994 г. по заказу компании RSA Data Security, Inc. Это блочный шифр с переменной длиной блока данных (32, 64 и 128 бит), ключа и числом циклов криптографического преобразования, которое варьируется от 1 до 255. Длина ключа может меняться от 1 до 2048 бит. Возможность параметризации позволяет гибко настраивать криптоалгоритм с учетом конкретных требований по криптостойкости и эффективности реализации. Номинальным выбором параметров считается 32-битный блок данных, 12 раундов и 16-байтовый ключ. Эта версия алгоритма известна как RC5-32/12/16.

Каждый раунд алгоритма шифрования состоит из двух полураундов. Пример полураунда алгоритма шифрования RC5 приведен на рис. 4.1. В номинальной версии RC5 содержится 24 таких полураундов. Каждый полураунд состоит из трех основных операций: целочисленного суммирования по модулю 2^w, суммирования по модулю 2 (операция XOR) и циклического сдвига на $\log_2 w$ битов влево, где w — размерность одной половины блока данных.

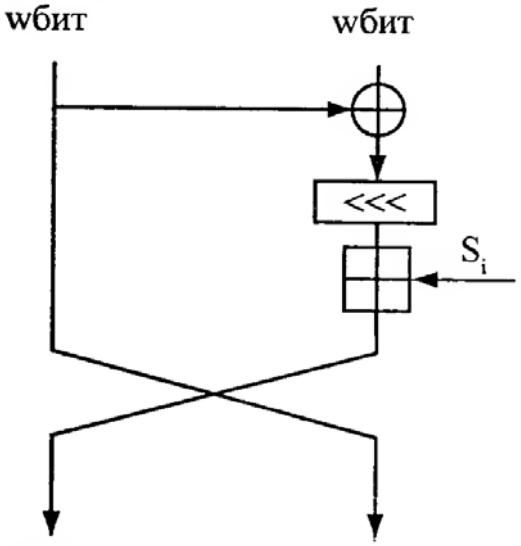


Рис. 4.1. ***Полураунд алгоритма шифрования RC5***

Криптоалгоритм RC5 состоит из трех основных процедур: расширения ключа, шифрования и дешифрования. В процедуре расширения ключа заданный секретный ключ подвергается специальному преобразованию — заполнению ключевой таблицы, причем размер таблицы зависит от числа циклов криптографического преобразования. Ключевая таблица используется затем для шифрования и дешифрования. Подробное описание заполнения ключевой таблицы для алгоритма

шифрования RC5 можно найти в [2]. Процедура шифрования начинается с разделения блока на две равные части. После этого левая часть текста складывается с первым подключом, а правая половина данных — со вторым. Затем новообразованные правая и левая части меняются местами. Далее следует выполнение полураундов. В каждом полураунде левая и правая части складываются по модулю два. После этого $\log_2\omega$ (где $2*\omega$ — число битов, поступающих на вход алгоритма шифрования) младших значащих битов левой половины текста указывают число битов, на которое необходимо сдвинуть полученную правую часть. Далее происходит целочисленное сложение по модулю правой части сообщения с подключом Si, и правая и левая части меняются местами. Исключение составляет последний полураунд, в котором не происходит обмена частей текста. Дешифрование выполняется в обратном порядке с использованием обратных операций: целочисленного вычитания по модулю 2^w, суммирования по модулю 2 (операция XOR) и циклического сдвига на $\log_2 w$ битов влево, где w — размерность одной половины блока данных.

Безусловное преимущество криптоалгоритма заключается в простоте реализации. Непредсказуемость результата операции циклического сдвига, зависящей от конкретных видов данных при шифровании, обеспечивает необходимый уровень криптостойкости. Исследования криптостойкости RC5 [3] показали, что вариант криптоалгоритма с разрядностью блока 64 бита и двенадцатью (и более) циклами преобразования гарантирует достаточную криптостойкость по отношению к дифференциальному и линейному криптоанализу.

4.2. IDEA

Первый вариант шифра IDEA, предложенный Ксуеджа Лай (Xuejia Lai) и Джеймсом Мэсси (James Massey), появился в 1990 г. Он назывался PES (Proposed Encryption Standard — Предполагаемый стандарт шифрования). В следующем году, после демонстрации Бихамом и Шамиром возможностей дифференциального криптоанализа, авторы усовершенствовали свой шифр так, чтобы он мог противостоять такой атаке, и назвали новый алгоритм IPES (Improved Proposed Encryption Standard — Улучшенный предполагаемый стандарт шифрования). В 1992 г. название IPES было изменено на IDEA (International Data Encryption Algorithm — Международный алгоритм шифрования данных). Криптоалгоритм IDEA представляет собой вариант 64-битного итеративного блочного шифра со 128-битным ключом и восемью циклами криптографического преобразования. IDEA не соответствует схеме Фейстеля. Процедура дешифрования выполняется аналогично процедуре шифрования.

Как и другие алгоритмы блочного шифрования, IDEA использует рассеивание и перемешивание. В основе конструкции алгоритма лежит «смещение опера-

ций различных алгебраических групп». Смешиваются три алгебраические группы, которые нетрудно реализовать аппаратными или программными средствами:

1. Операция XOR;
2. Сложение по модулю 2^{16};
3. Умножение по модулю $(2^{16} + 1)$. (Эту операцию можно рассматривать как S-блок алгоритма IDEA.)

Все эти операции (в алгоритме используются только они: перестановки на битовом уровне не применяются) работают с 16-битовыми подблоками.

Схема алгоритма IDEA приведена на рис. 4.2, где 64-битовый блок данных делится на четыре 16-битовых подблока X_1, X_2, X_3 и X_4. Эти подблоки используются как входы первого раунда алгоритма. Всего в алгоритме восемь раундов. В каждом из них четыре подблока подвергаются операциям XOR, сложениям и умножениям друг с другом и шестью 16-битовыми подключами $Z_i^{(r)}$, где i меняется от 1 до 6 и r меняется от 1 до 9. Между раундами второй и третий подблоки меняются местами. Наконец в выходном преобразовании четыре подблока объединяются с четырьмя подключами.

В каждом раунде события происходят в следующем порядке:

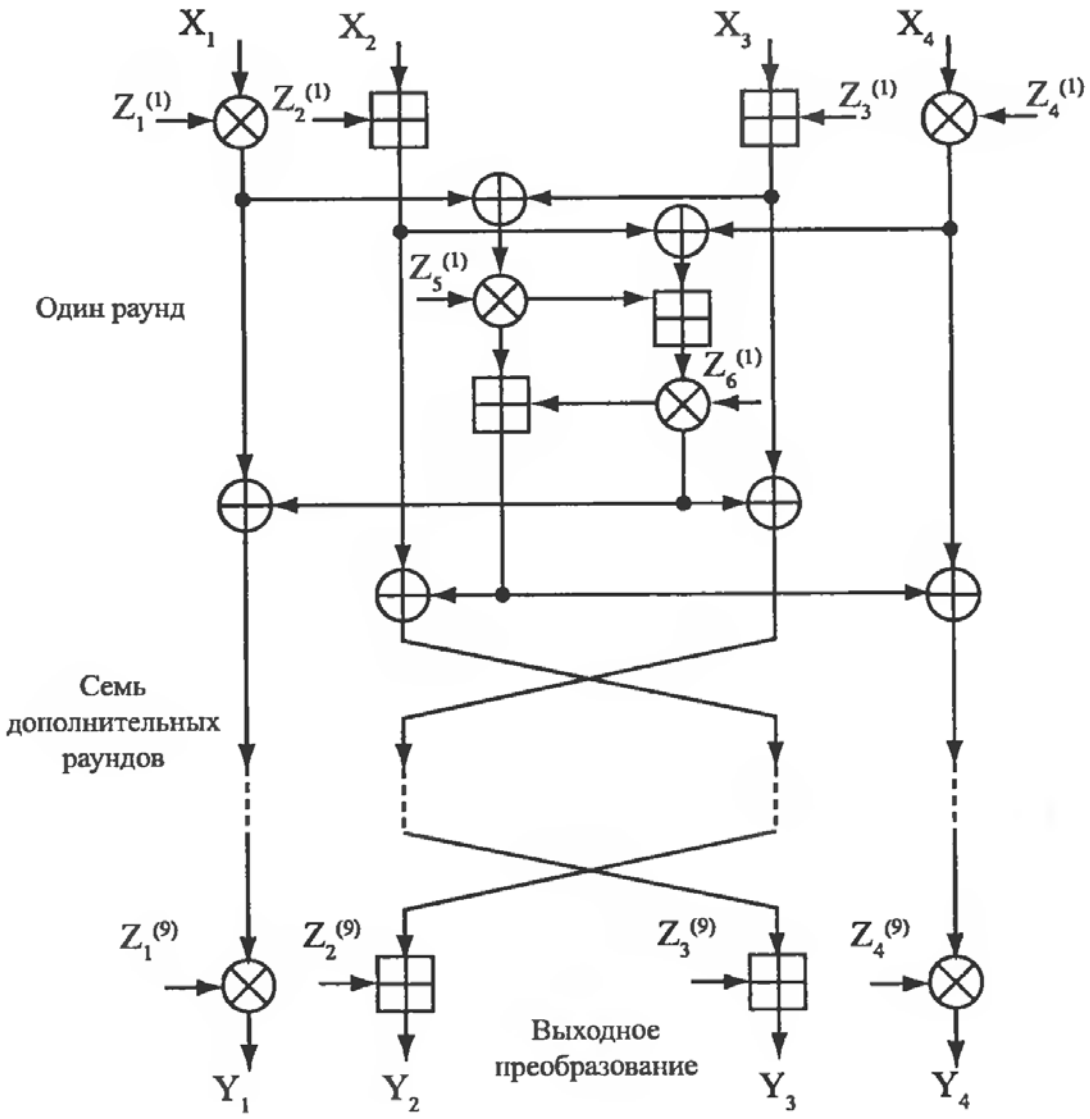
1. Перемножаются X_1 и первый подключ;
2. Складываются X_2 и второй подключ;
3. Складываются X_3 и третий подключ;
4. Перемножаются X_4 и четвертый подключ;
5. Выполняется операция XOR над результатами пунктов 1 и 3;
6. Выполняется операция XOR над результатами пунктов 2 и 4;
7. Перемножаются результаты пункта 5 и пятый подключ;
8. Складываются результаты пунктов 6 и 7;
9. Перемножаются результаты пункта 8 и шестой подключ;
10. Складываются результаты пунктов 7 и 9;
11. Выполняется операция XOR над результатами пунктов 1 и 9;
12. Выполняется операция XOR над результатами пунктов 3 и 9;
13. Выполняется операция XOR над результатами пунктов 2 и 10;
14. Выполняется операция XOR над результатами пунктов 4 и 10.

На выходе раунда создаются четыре подблока — результаты действий 11 — 14. Поменяйте местами два внутренних подблока в любом (но не последнем) раунде, и вы получите исходные данные для следующего раунда.

После восьмого раунда выполняется заключительное преобразование:

1. Перемножаются X_1 и первый подключ $Z_1^{(9)}$;
2. Складываются X_2 и второй подключ $Z_2^{(9)}$;
3. Складываются X_3 и третий подключ $Z_3^{(9)}$;
4. Перемножаются X_4 и четвертый подключ $Z_4^{(9)}$.

Наконец четыре подблока снова соединяются, образуя шифртекст Y_1, Y_2, Y_3 и Y_4.



X_1 — 16-битовый подблок открытого текста;

Y_1 — 16-битовый подблок шифртекста;

$Z_1^{(9)}$ — 16-битовый подблок ключа;

⊕ — побитовая операция исключающее ИЛИ (XOR) над 16-битовыми подблоками;

⊞ — сложение по модулю 2^{16} 16-разрядных целых чисел;

⊗ — умножение по модулю $(2^{16} + 1)$ 16-разрядных целых чисел, причем нулевой подблок соответствует 2^{16}.

Рис. 4.2. ***Алгоритм шифрования IDEA***

Создание подключей столь же просто. В алгоритме используются 52 подключа — шесть в каждом из восьми раундов и еще четыре в заключительном преобразовании. Сначала 128-битовый ключ разделяется на восемь 16-битовых подключей. Это первые восемь подключей алгоритма (шесть для первого раунда и

два — для второго). Затем ключ циклически сдвигается налево на 25 битов и снова делится на восемь подключей. Первые четыре подключа используются в раунде 2, а оставшиеся четыре — в раунде 3. Ключ циклически сдвигается налево на 25 битов для получения следующих восьми подключей и т. д. до завершения алгоритма.

Дешифрование выполняется аналогично зашифрованию за исключением того, что подключи инвертируются и немного изменяются. Подключи, используемые при дешифровании, представляют собой обратные значения ключей шифрования по отношению к операциям либо сложения, либо умножения. Эти вычисления могут занять некоторое время, но их нужно выполнить всего один раз для каждого ключа дешифрования. Подключи зашифрования и соответствующие им подключи дешифрования представлены в таблице 4.1.

Таблица 4.1

Подключи зашифрования и дешифрования алгоритма шифрования IDEA

Раунд	Подключи зашифрования	Подключи дешифрования
1	$Z_1^{(1)} Z_2^{(1)} Z_3^{(1)} Z_4^{(1)} Z_5^{(1)} Z_6^{(1)}$	$Z_1^{(9)-1} - Z_2^{(9)} - Z_3^{(9)} Z_4^{(9)-1} Z_5^{(8)} Z_6^{(8)}$
2	$Z_1^{(2)} Z_2^{(2)} Z_3^{(2)} Z_4^{(2)} Z_5^{(2)} Z_6^{(2)}$	$Z_1^{(8)-1} - Z_3^{(8)} - Z_2^{(8)} Z_4^{(8)-1} Z_5^{(7)} Z_6^{(7)}$
3	$Z_1^{(3)} Z_2^{(3)} Z_3^{(3)} Z_4^{(3)} Z_5^{(3)} Z_6^{(3)}$	$Z_1^{(7)-1} - Z_3^{(7)} - Z_2^{(7)} Z_4^{(7)-1} Z_5^{(6)} Z_6^{(6)}$
4	$Z_1^{(4)} Z_2^{(4)} Z_3^{(4)} Z_4^{(4)} Z_5^{(4)} Z_6^{(4)}$	$Z_1^{(6)-1} - Z_3^{(6)} - Z_2^{(6)} Z_4^{(6)-1} Z_5^{(5)} Z_6^{(5)}$
5	$Z_1^{(5)} Z_2^{(5)} Z_3^{(5)} Z_4^{(5)} Z_5^{(5)} Z_6^{(5)}$	$Z_1^{(5)-1} - Z_3^{(5)} - Z_2^{(5)} Z_4^{(5)-1} Z_5^{(4)} Z_6^{(4)}$
6	$Z_1^{(6)} Z_2^{(6)} Z_3^{(6)} Z_4^{(6)} Z_5^{(6)} Z_6^{(6)}$	$Z_1^{(4)} {}_{-1} - Z_3^{(4)} - Z_2^{(4)} Z_4^{(4)} {}_{-1} Z_5^{(3)} Z_6^{(3)}$
7	$Z_1^{(7)} Z_2^{(7)} Z_3^{(7)} Z_4^{(7)} Z_5^{(7)} Z_6^{(7)}$	$Z_1^{(3)-1} - Z_3^{(3)} - Z_2^{(3)} Z_4^{(3)-1} Z_5^{(2)} Z_6^{(2)}$
8	$Z_1^{(8)} Z_2^{(8)} Z_3^{(8)} Z_4^{(8)} Z_5^{(8)} Z_6^{(8)}$	$Z_1^{(2)-1} - Z_3^{(2)} - Z_2^{(2)} Z_4^{(2)-1} Z_5^{(1)} Z_6^{(1)}$
Заключительное преобразование	$Z_1^{(9)} Z_2^{(9)} Z_3^{(9)} Z_4^{(9)}$	$Z_1^{(1)-1} - Z_3^{(1)} - Z_2^{(1)} Z_4^{(1)-1}$

Структура криптоалгоритма допускает простую программную и аппаратную реализацию. Секретность преобразования обеспечивается за счет трех различных типов арифметических операций над 16-битными словами. Эффективность программной реализации IDEA не уступает DES.

4.3. SAFER

Криптоалгоритм SAFER (Secure And Fast Encription Routine) представляет собой блочный шифр, разработанный по заказу корпорации Cylink. Длина секретного ключа в одной из версий составляет 64 бита. Криптоалгоритм ориентирован на байтовую обработку блоков по 64 бита. Имеет переменное число циклов криптографического преобразования (от 6 до 10). В отличие от большинства блочных шифров имеет различные процедуры шифрования и дешифрования. Первая версия SAFER с длиной ключа 64 бита известна под названием SAFER K-64

(Secure And Fast Encription Routine with a Key of 64-bits). Вторая версия — SAFER K-128 — имеет 128-битный ключ.

Блок открытого текста разделяется на восемь байтовых подблоков: B_1, B_2, … …, B_7, B_8. Затем подблоки обрабатываются в r раундах алгоритма. Наконец подблоки подвергаются заключительному преобразованию. В каждом раунде используются два подключа: K_{2i-1} и K_{2i}.

На рис. 4.3 показан один раунд алгоритма шифрования SAFER K-64. Сначала над подблоками выполняется либо операция XOR, либо сложение с байтами подключа K_{2r-1}. Затем восемь подблоков подвергаются одному из двух нелинейных преобразований:

$y = 45^x \bmod 257$ (если x = 128, то y = 0) (на рис. 4.3 обозначена как $45^{(\cdot)}$);

$y = \log_{45} x$ (если x = 0, то y = 128) (на рис. 4.3 обозначена как $\log_{45}$).

Эти операции выполняются в конечном поле GF(257), причем 45 — примитивный корень этого поля. При возведении числа 45 в степень 128 образуется число, которое нацело делится на 257, поэтому $y = 45^{128} \bmod 257 = 0$. Получается, что при умножении чисел по модулю, необходимо 45 возвести в 128 степень, чтобы получить ноль. Именно поэтому $y = \log_{45} 0 = 128$. В практических применениях SAFER K-64 эти операции эффективнее реализовать с помощью таблицы подстановок вместо постоянных вычислений новых результатов.

Далее подблоки либо подвергаются операции XOR, либо складываются с байтами подключа K_{2r}. Результат этой операции проходит через три уровня линейных операций, предназначенных для усиления лавинного эффекта.

Каждая операция называется псевдоадамаровым преобразованием (Pseudo-Hadamard Transform — PHT). Если на вход PHT подать a_1 и a_2, то выходом будут:

$$b_1 = (2a_1 + a_2) \bmod 256;$$

$$b_2 = (a_1 + a_2) \bmod 256.$$

После r раундов выполняется заключительное преобразование. Оно совпадает с первым этапом каждого раунда. Над B_1, B_4, B_5 и B_8 выполняется операция XOR с соответствующими байтами последнего подключа, а B_2, B_3, B_6 и B_7 складываются с соответствующими байтами последнего подключа. В результате получается шифртекст.

Дешифрование представляет собой обратный процесс: сначала выполняется заключительное преобразование (с вычитанием вместо сложения), затем r раундов, в которых операции выполняются в обратном порядке. Обратное преобразование PHT (Inverse PHT — IPHT) представляет собой:

$$a_1 = (b_1 - b_2) \bmod 256;$$

$$a_2 = (-b_1 + 2b_2) \bmod 256.$$

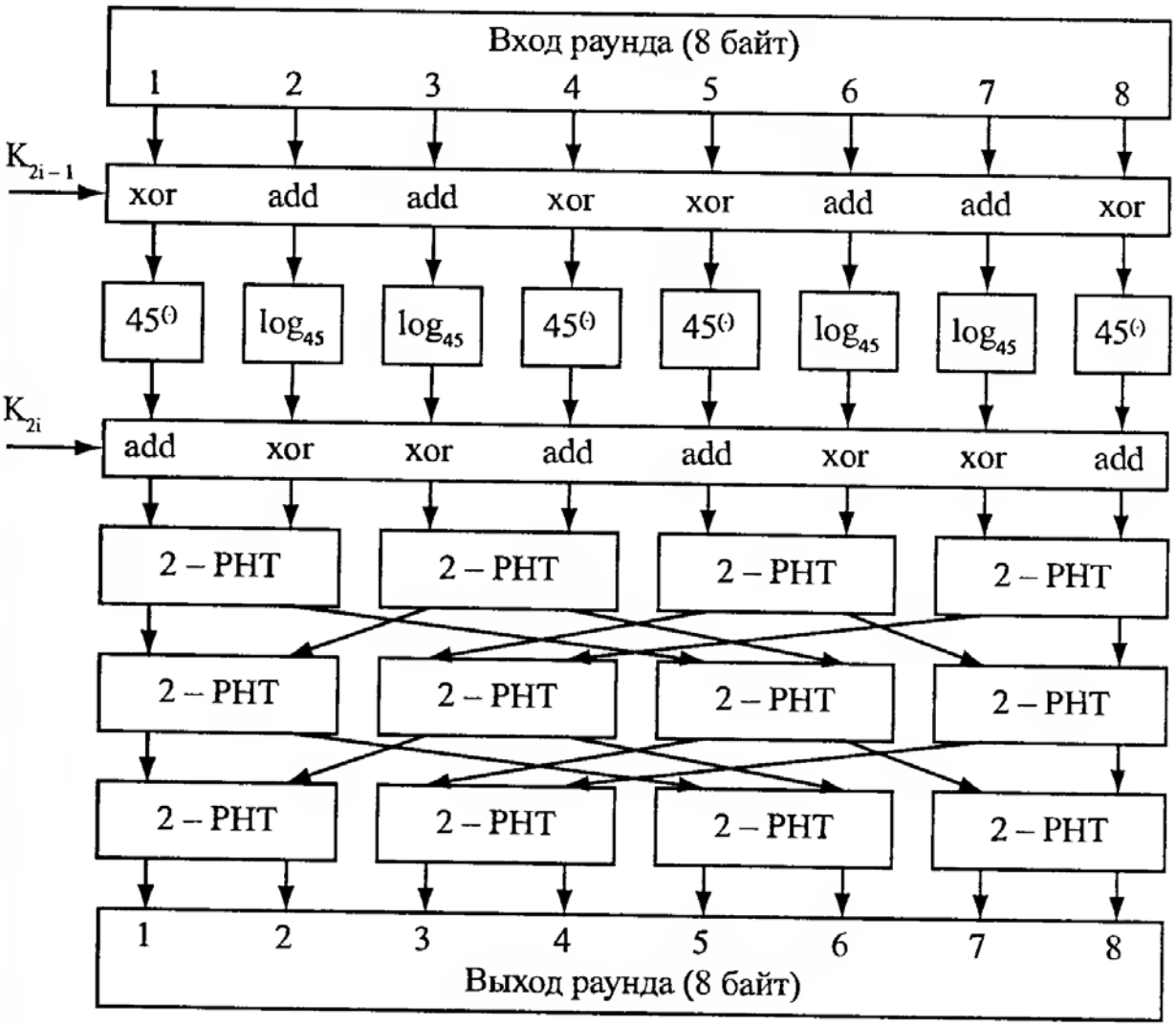


Рис. 4.3. ***Алгоритм шифрования SAFER***

Генерировать подключи совсем нетрудно. Первый подключ K_1 — это просто ключ пользователя. Последующие ключи генерируются следующей процедурой:

$$K_{i+1} = (K_i <<< 3i) + c_i,$$

где «<<<» обозначает циклический сдвиг влево. Сдвиг выполняется побайтово, причем c_i — константа раунда. Если c_{ij} — это j-й байт константы i-го раунда, то для расчета всех констант раундов можно использовать следующую формулу:

$$c_{ij} = 45^{45^{\wedge((9i+j) \bmod 256) \bmod 257}} \bmod 257.$$

Обычно эти значения хранятся в таблице.

Результаты криптоанализа продемонстрировали криптостойкость SAFER K-64 по отношению к дифференциальному и линейному криптоанализу при соблюдении одного условия — число циклов преобразования должно быть больше шести [3]. Модификация SAFER SK-40 имеет 40-битный ключ и пять циклов преобразования и требует большего объема вычислений при дифференциальном и

линейном криптоанализе по сравнению с силовой атакой (исчерпывающим перебором на множестве ключей) [3].

4.4. LOKI91

Алгоритм LOKI был разработан в Австралии и впервые представлен в 1990 г. в качестве возможной замены DES. В нем используются 64-битовый блок и 64-битовый ключ. Обнаруженные в исходном варианте алгоритма шифрования слабости были устранены разработчиками в 1991 г. Таким образом появился новый алгоритм шифрования LOKI91.

Механизм алгоритма LOKI91 подобен DES (см. рис. 4.4). Блок данных разделяется на левую и правую половины и проходит 16 раундов, что весьма напоминает DES. В каждом раунде правая половина сначала подвергается операции XOR с частью ключа, а затем расширяющей перестановке E согласно таблице 4.2.

Таблица 4.2

Перестановка с расширением для алгоритма шифрования LOKI91

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

48-битовый выход разделяется на четыре 12-битовых блока. В каждом блоке выполняется такая подстановка с использованием S-блока: берется каждый 12-битовый вход, 2 старших и 2 младших бита используются для образования номера r, а восемь внутренних битов образуют номер c. Выход S-блока имеет следующее значение:

$$S(r,c) = (c + ((r*17) \oplus 0xff)\ \&0xff)^{31} \bmod P_r.$$

Значения P_r приведены в таблице 4.3.

Таблица 4.3

Значения P_r

r	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
P_r	375	379	391	395	397	415	419	425	433	445	451	463	471	477	487	499

Затем четыре 8-битовых результата снова объединяются. Образуя 32-битовое число, которое подвергается операции перестановки, описанной в таблице 4.4. Наконец для получения новой левой половины выполняется операция XOR правой половины с прежней левой половиной, а левая половина становится новой правой половиной. После 16 раундов для получения окончательного шифртекста снова выполняется операция XOR над блоком и ключом.

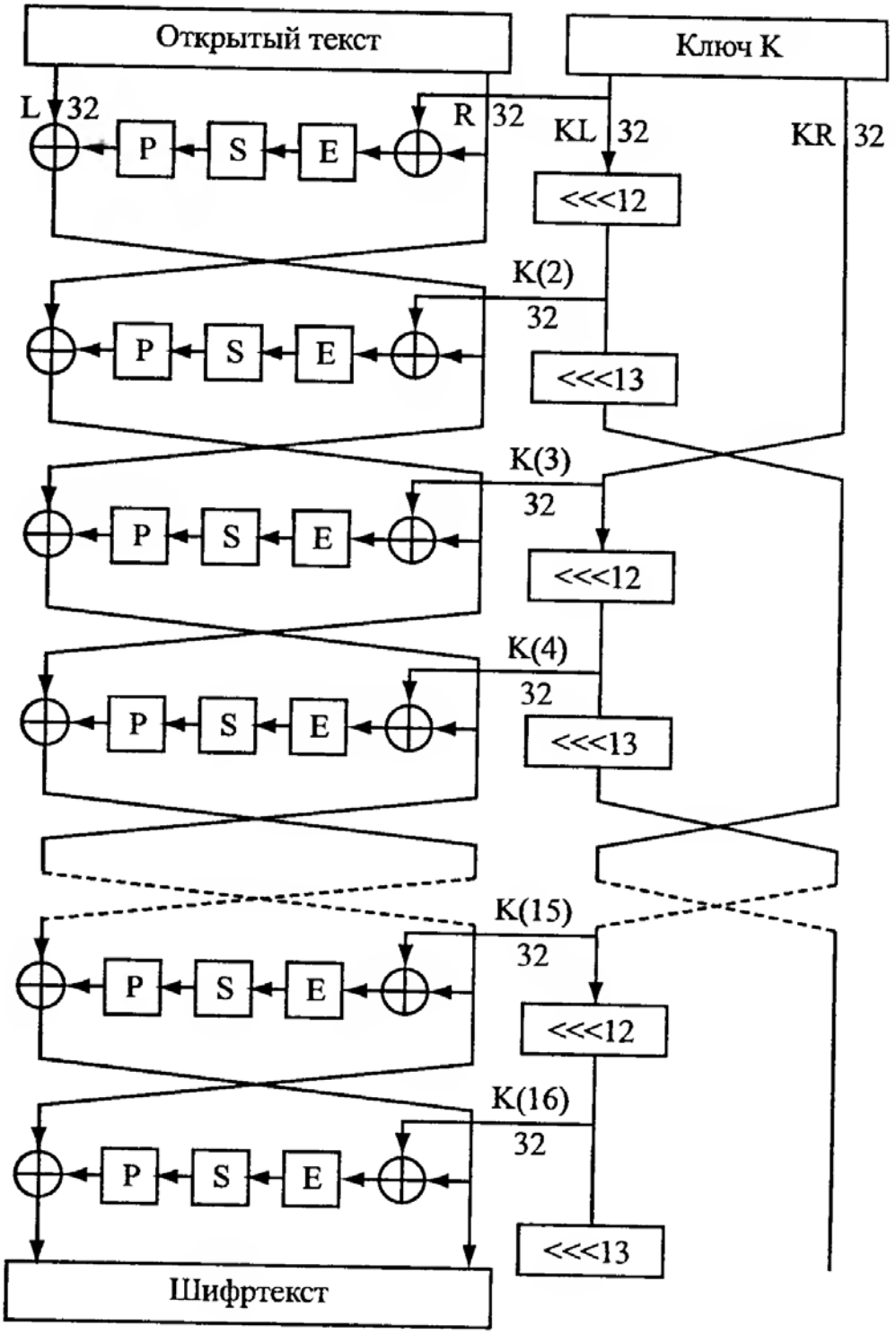


Рис. 4.4. ***Алгоритм шифрования LOKI91***

Таблица 4.4

Перестановка с помощью P-блока

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

Подключи генерируются из ключа достаточно прямолинейно: 64-битовый ключ разбивается на левую и правую половины. На каждом раунде подключом служит левая половина. Далее она циклически сдвигается влево на 12 или 13 битов, затем после каждых двух раундов левая и правая половина меняются местами. Как и в DES, для зашифрования и дешифрования используется один и тот же алгоритм с некоторыми изменениями в использовании подключей.

4.5. FEAL

Криптоалгоритм FEAL был разработан как альтернатива DES. Оригинальный криптоалгоритм (FEAL-4) был рассчитан на программную реализацию и имел четыре цикла преобразования при обработке 64-битного блока и 64-битный секретный ключ.

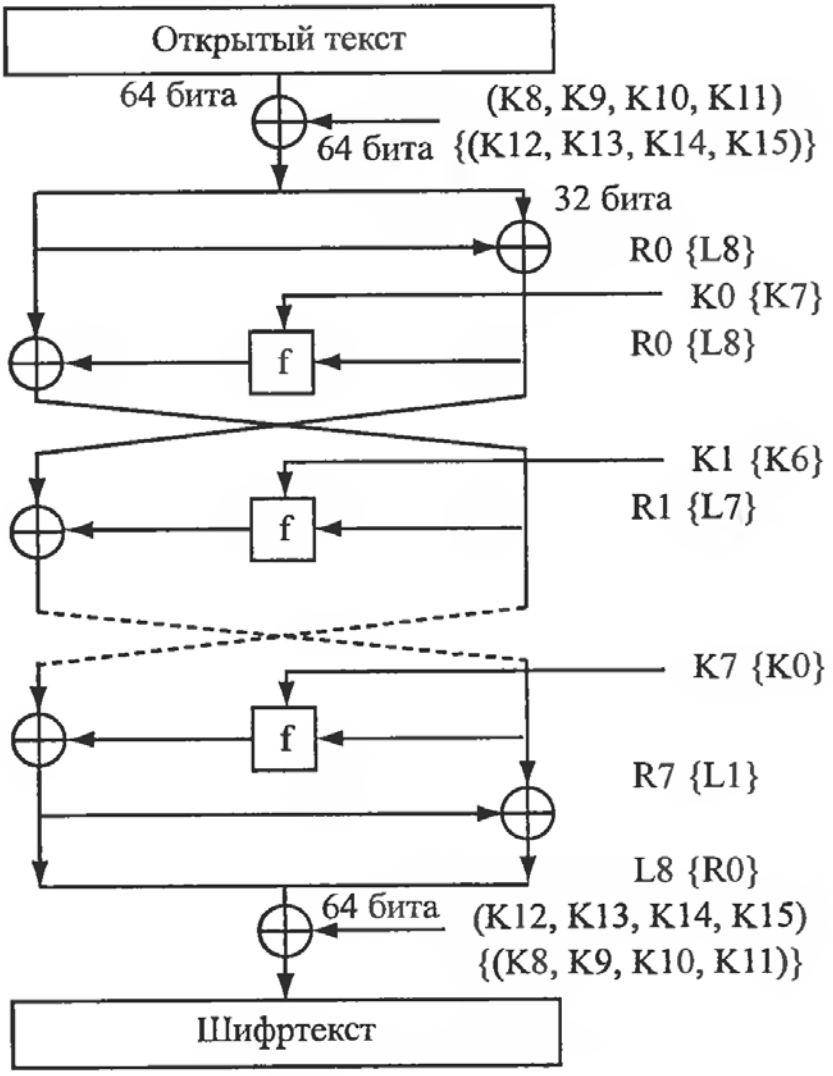


Рис. 4.5. ***Один раунд алгоритма шифрования FEAL***

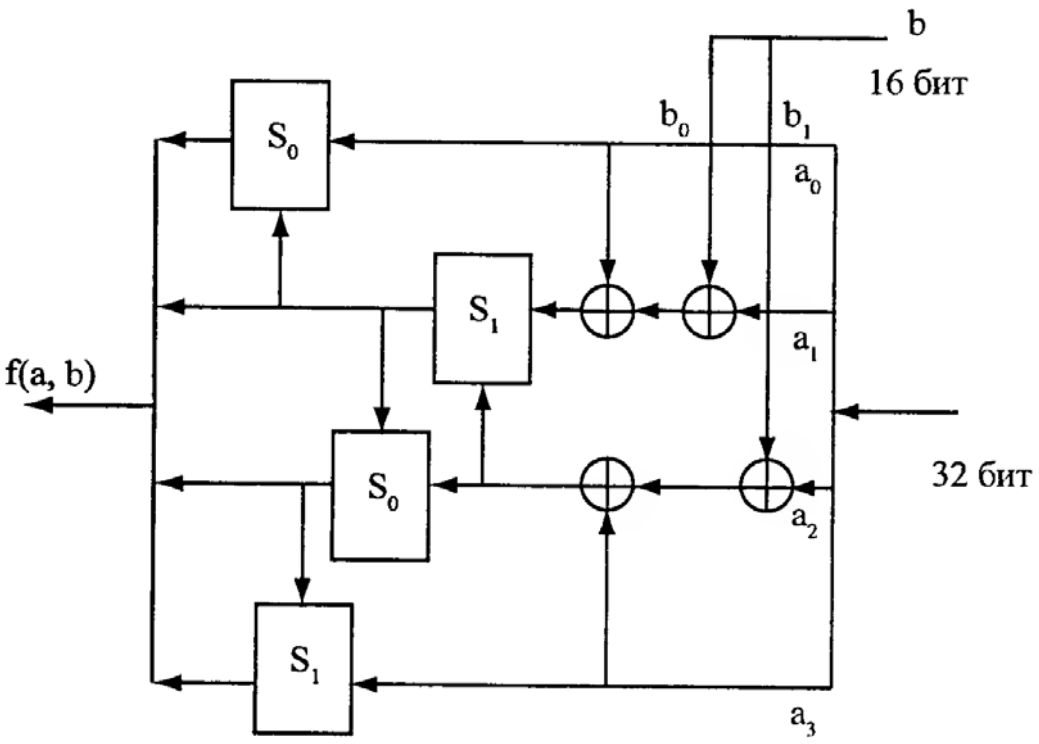


Рис. 4.6. ***Функция f алгоритма шифрования FEAL***

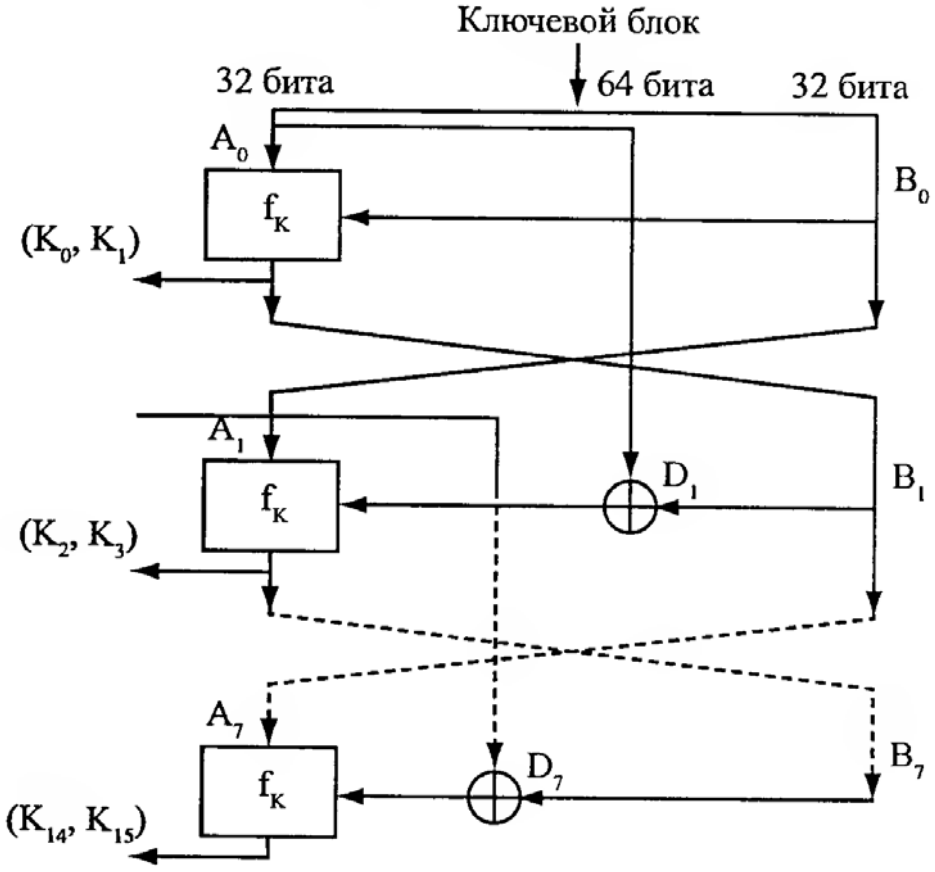
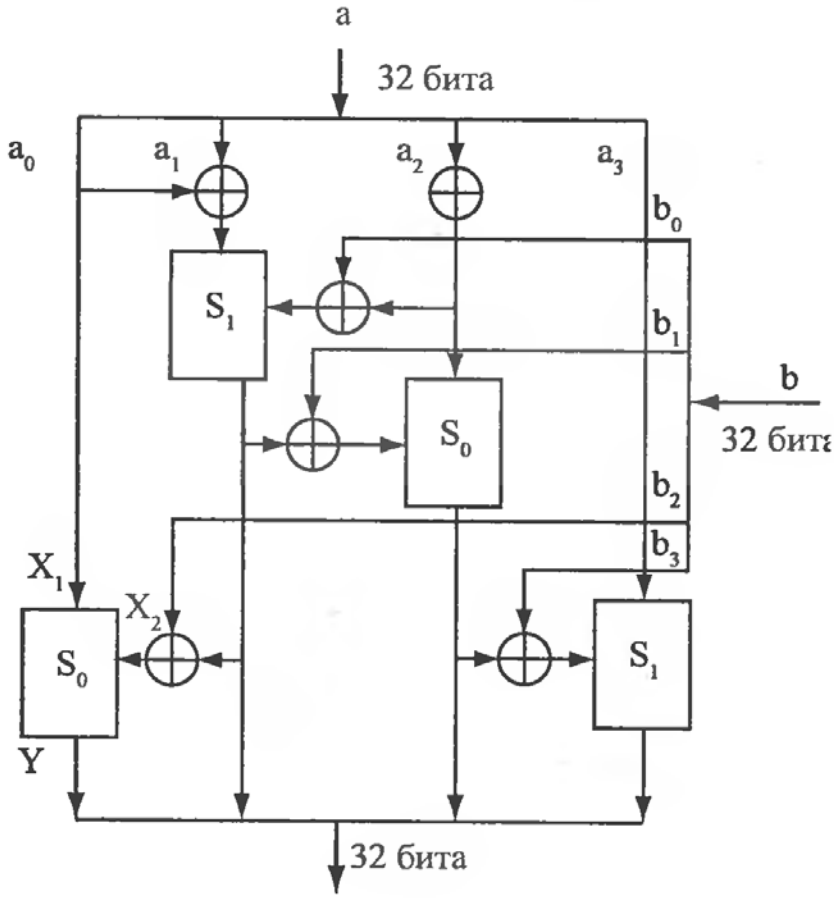


Рис. 4.7. ***Схема генерации ключа***

На рис. 4.5 представлена схема одного раунда алгоритма FEAL. В качестве входа процесса зашифрования используется 64-битовый блок открытого текста. Сначала блок данных подвергается операции XOR с 64 битами ключа. Затем блок данных расщепляется на левую и правую половины. Сложение левой и правой половин операцией XOR создает новую правую половину. Левая половина и новая правая половина проходят через n раундов алгоритма (первоначально их было четыре). В каждом раунде функция f объединяет правую половину с шестнадцатью битами ключа, а операция XOR — с левой половиной, создавая новую правую половину. Исходная (на начало раунда) правая половина становится новой левой половиной. После n раундов (после n-го раунда левая и правая половины не переставляются) левая половина снова объединяется операцией XOR с правой половиной, образуя новую правую половину, затем левая и правая половины соединяются вместе в 64-битовое целое. Блок дннх объединяется операцией XOR с другими 64 битами ключа, и алгоритм завершается.



a_i, b_i: 8 битов;
$Y = S_0(X_1, X_2)$ = циклический сдвиг влево на 2 бита $((X_1 + X_2) \bmod 256)$
$Y = S_1(X_1, X_2)$ = циклический сдвиг влево на 2 бита $((X_1 + X_2 + 1) \bmod 256)$

Рис. 4.8. ***Функция f_k алгоритма шифрования FEAL***

Схема функции f приведена на рис. 4.6. Функция f принимает 32 бита данных и 16 некоторых бит ключа и перемешивает их друг с другом. Сначала блок данных разбивается на 8-битовые подблоки, которые затем объединяются операцией XOR и заменяют друг друга. Функции S_0 S_1 определяются следующим образом:

$S_0(a,b)$ = циклический сдвиг влево на 2 бита ((a + b) mod 256);

$S_1(a,b)$ = циклический сдвиг влево на 2 бита ((a + b + 1) mod 256).

Тот же алгоритм можно использовать для дешифрования. Единственное отличие заключается в обратном порядке использования частей ключа.

На рис. 4.7 представлена схема генерации ключа. Сначала 64-битовый ключ расщепляется на две половины, которые, как показано на схеме, обрабатываются операцией XOR и функцией f_k. На рис. 4.8 показана схема функции f_k. Два 32-битовых входа разбиваются на 8-битовые блоки, объединяемые и заменяемые в соответствии со схемой. S_0 и S_1 определяются, как показано на рис. 4.8. Затем 16-битовые блоки ключа используются в алгоритме зашифрования/дешифрования.

4.6. Blowfish

Криптоалгоритм Blowfish является алгоритмом шифрования, построенным по схеме Фейстеля, и рассчитан на обработку 64-битных входных блоков, 16 раундов и может использовать до 448 битов секретного ключа. S-блоки, используемые в алгоритме шифрования, зависят от битов ключа, и поэтому не известны злоумышленнику. Каждый цикл криптографического преобразования состоит из серии перестановок, которые зависят от ключа и подстановок, которые, в свою очередь, зависят от ключа и входных данных. Процедура криптографического преобразования построена на операциях целочисленного суммирования 32-разрядных чисел и суммирования по модулю 2 (операция XOR). Ключ используется для построения нескольких вспомогательных ключевых таблиц. Один раунд криптографического преобразования алгоритма Blowfish приведен на рис. 4.9.

Исходный блок 64-битных данных разделяется на две половины по 32 бита каждая. После этого к левой половине по модулю 2 добавляется раундовый подключ P_i. Результат сложения подвергается следующему преобразованию. 32-битный блок данных разделяется на 4 блока по 8 бит, и каждый из них заменяется с помощью соответствующего S-блока. Результат замены блоков S_1 и S_2 складываются друг с другом по модулю 2^{32}. Полученная сумма складывается по модулю два с результатом преобразования блока замены S_3. И, наконец, полученное значение складываются по модулю 2^{32} с результатом преобразования блока замены S_4. Полученное в результате преобразования значение добавляется к правой части преобразуемых данных. После этого левая часть сообщения, сложенная с ключом P_i, и новообразованная правая часть меняются местами.

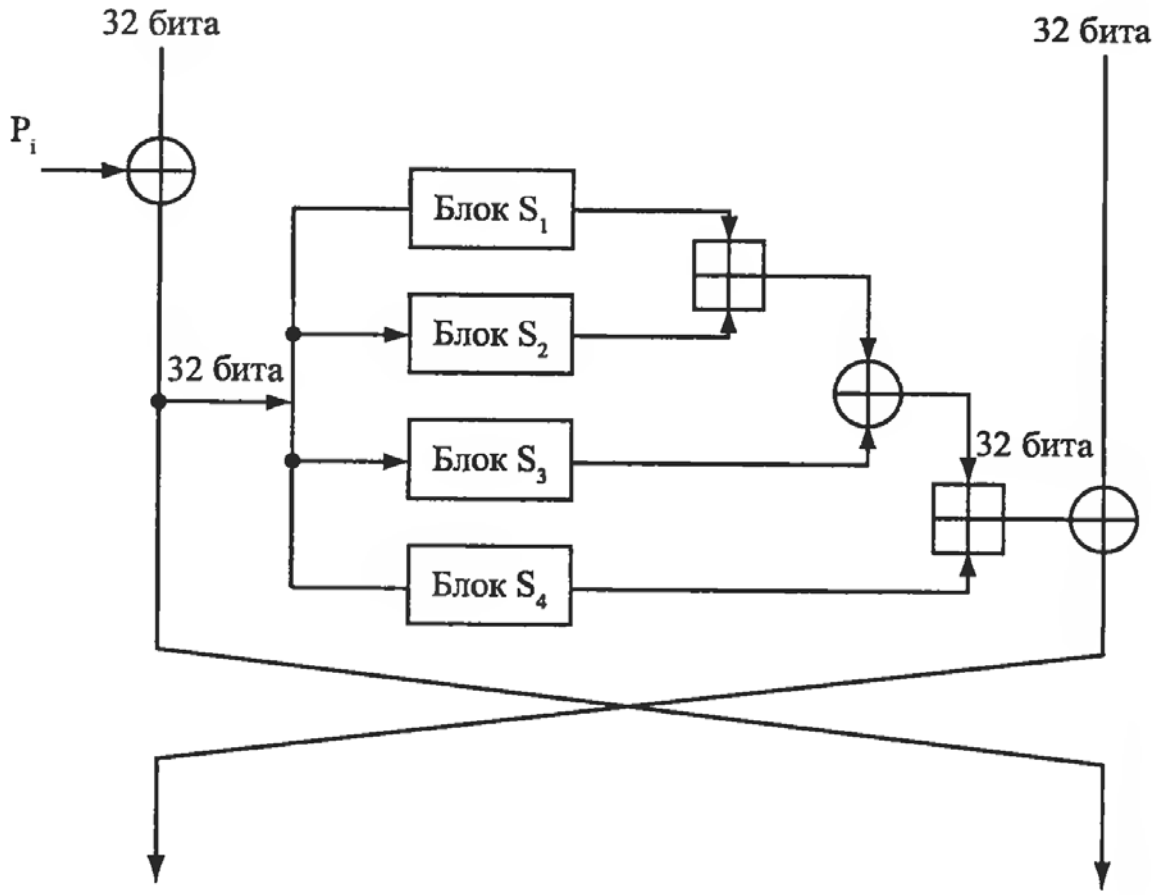


Рис. 4.9. ***Один раунд алгоритма шифрования Blowfish***

Криптоалгоритм разработан специально для 32-битных компьютеров. По эффективности программная реализация Blowfish значительно превосходит аналогичную реализацию DES. Известен ряд успешных атак на Blowfish с тремя циклами преобразования [3].

4.6.1. Вычисление подключей и S-блоков

Как уже упоминалось выше, алгоритм Blowfish позволяет использовать ключ, длина которого меняется в диапазоне от 32 до 448 битов, то есть может составлять от 1 до 14 32-битовых слов. На основе этого ключа генерируется 18 32-битовых подключей и четыре S-блока размером 8×32, в общей сложности содержащие 1024 32-битовых элемента. Всего получается 1042 32-битовых значения, или 4168 байтов.

Ключи хранятся в массиве:

$$K_1, K_2, \ldots, K_j, \text{ где } 1 \le j \le 14.$$

Подключи хранятся в P-массиве:

$$P_1, P_2, \ldots, P_{18}.$$

Имеется четыре S-блока, каждый из которых содержит 256 32-битовых элементов:

$$S_{1,0}, S_{1,1}, \ldots, S_{1,256};$$
$$S_{2.0}, S_{2,1}, \ldots, S_{2,256};$$
$$S_{3.0}, S_{3,1}, \ldots, S_{3,256};$$
$$S_{4,0}, S_{4,1}, \ldots, S_{4.256}.$$

P-массив и S-блоки генерируются в следующей последовательности.

1. Сначала инициализируется P-массив, а затем четыре S-блока, для чего используются биты дробной части числа π — первые 32 бита дробной части числа π присваиваются P_1, следующие 32 бита присваиваются P_2 и т. д. Например, выразив все числа в шестнадцатеричной форме, получим:

$$P_1 = 243F6A88,$$
$$P_2 = 85A308D3,$$
$$\ldots$$
$$S_{4,254} = 578FDFE3,$$
$$S_{4,255} = 3AC372E6.$$

2. Выполняется операция XOR (побитовое исключающее «ИЛИ») P-массива и K-массива (при необходимости слова из K-массива используются повторно). Например, при максимальной длине ключа (14 32-битовых слов) будем иметь $P_1 = P_1 \oplus K_1$, $P_2 = P_2 \oplus K_2$, …, $P_{14} = P_{14} \oplus K_{14}$, $P_{15} = P_{15} \oplus K_1$, …, $P_{18} = P_{18} \oplus K_4$.
3. Используя текущее значение P-массивов и S-блоков, шифруется 64-битовый блок, состоящий из одних нулей, и результат шифрования замещает значения P_1 и P_2.
4. На основе текущих значений P-массивов и S-блоков шифруется значение, полученное на выходе шага 3, и результат шифрования замещает значения P_3 и P_4.
5. Процесс продолжается до тех пор, пока не будут обновлены все элементы P-массива по порядку, а затем — все элементы S-блоков, используя на каждом шаге выходные значения постоянно меняющегося алгоритма Blowfish.

Процесс обновления можно представить в следующем виде:

$$\begin{aligned} P_1, P_2 &= E_{P,S}[0], \\ P_3, P_4 &= E_{P,S}[P_1 \| P_2], \\ &\ldots \\ P_{17}, P_{18} &= E_{P,S}[P_{15} \| P_{16}], \\ S_{1,0}, S_{1,1} &= E_{P,S}[P_{17} \| P_{18}], \\ &\ldots \\ S_{4,254}, S_{4,255}, \end{aligned}$$

где $E_{P,S}[Y]$ обозначает шифрованный текст, получаемый шифрованием Y по алгоритму Blowfish с текущими значениями массивов P и блоков S.

4.7. Khufu

Криптоалгоритм Khufu был разработан Р. Мерклем (R. Merkl) и представляет собой 64-битный шифр с 512-битным ключом и переменным числом циклов криптографического преобразования. Криптостойкость алгоритма обеспечивается за счет использования S-блоков (8×32 бит), зависящих от битов ключа.

Алгоритм Khufu представляет собой схему Фейстеля, так что входной блок разделяется на две 32-битные половины L и R. Каждый раунд состоит из следующих простых шагов:

1. Используется младший значащий байт L в качестве входа в S-блок;
2. Выход S-блока складывается по модулю 2 с правой частью R;
3. Переставляются байты левой половины согласно схеме перестановки;
4. Правая R и левая L части сообщения меняются местами.

Значение S-блока меняется каждые 8 раундов во избежание атаки, основанной на предположении единственного входа в S-блок. На рис. 4.10 приведен алгоритм шифрования Khufu.

Автор криптоалгоритма отметил, что Khufu с 8 циклами менее криптостоек при атаке на основе выборочного открытого текста, чем вариант с 16, 24 или 32 циклами. Известен результат успешной атаки на Khufu с 16 циклами. Для осуществления атаки понадобилось 2^{31} образцов выборочного открытого текста. Результат, однако, не удалось распространить на большее число циклов преобразования. Силовая атака на Khufu безнадежна: 2^{512} попыток дешифрования — такой объем перебора не реализуем ни при каких условиях [3].

4.8. Khafre

Khafre — еще один криптоалгоритм, предложенный Мерклем. По конструкции этот алгоритм похож на Khufu (см. рис. 4.10). Отличие заключается лишь в том, что S-блоки известны и представляют собой данные, складываемые с дополнительным 64-битным подключом каждые восемь раундов.

При реализации Khafre менее эффективен, чем Khufu за счет использования 64- и 128-битных ключей и большего числа циклов преобразования. Атака на основе метода дифференциального криптоанализа позволила раскрыть ключ Khafre с 16 циклами после часа работы на персональном компьютере [3].

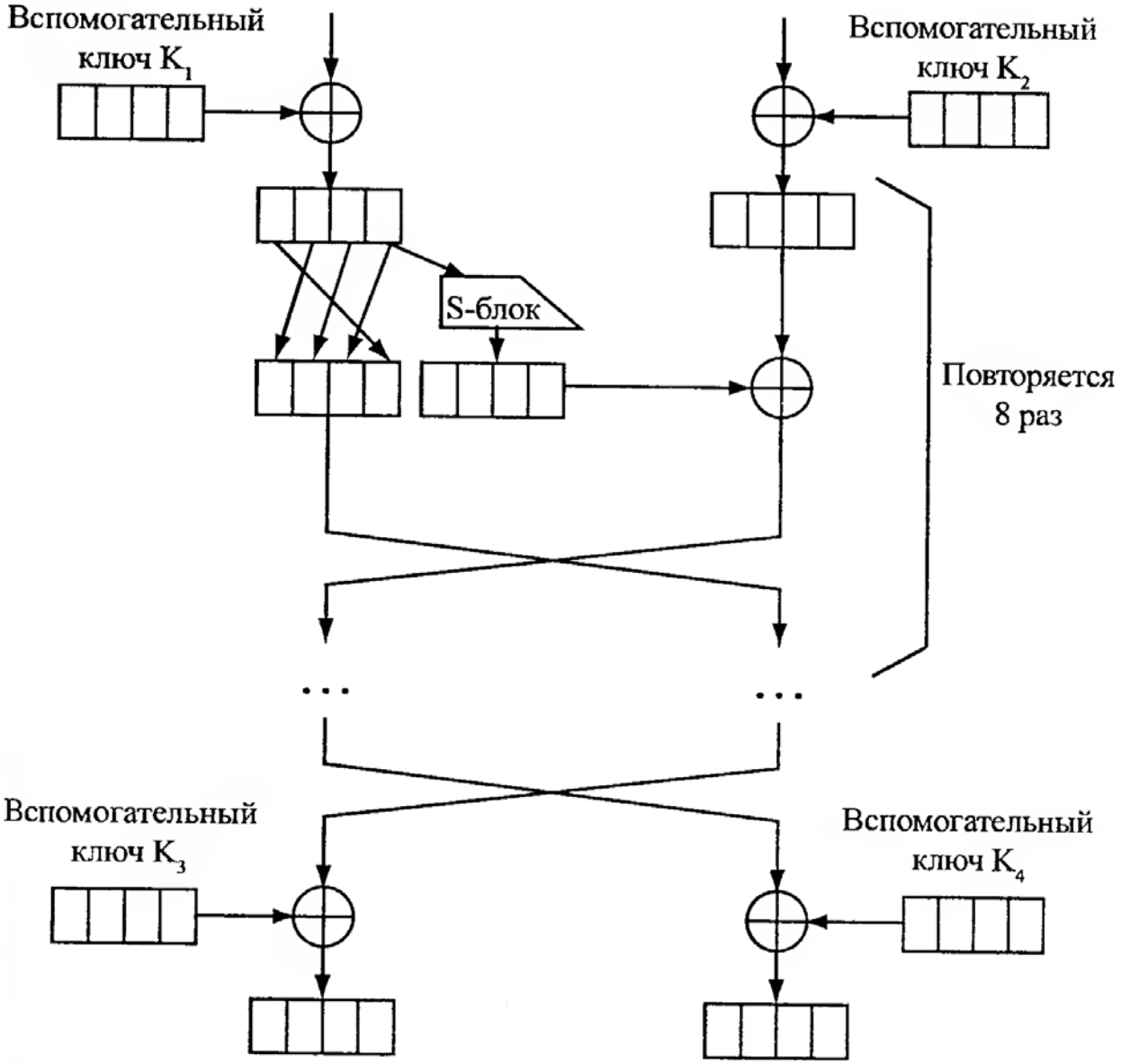


Рис. 4.10. *Алгоритмы шифрования Khufu и Khafre*

4.9. MARS

Алгоритм шифрования MARS является блочным алгоритмом шифрования. Входные данные алгоритма представляют собой 4 32-битных слова (то есть всего 128 бит). Общий вид алгоритма приведен на рис. 4.11. Алгоритм состоит из так называемого «криптографического ядра» («cryptographic core»), новаторски упакованного в «защитную скорлупу» специальной структуры, призванной защитить стойкость алгоритма от всех нынешних и будущих атак. Можно сказать, что алгоритм шифрования MARS состоит из трех основных этапов.

Первый этап обеспечивает быстрое преобразование и сложение с ключом для снижения возможности проведения криптографических атак на основе выбран-

ных открытых текстов. Он состоит из сложения ключа с входными данными и последующими восемью раундами, основанными на использовании S-блоков.

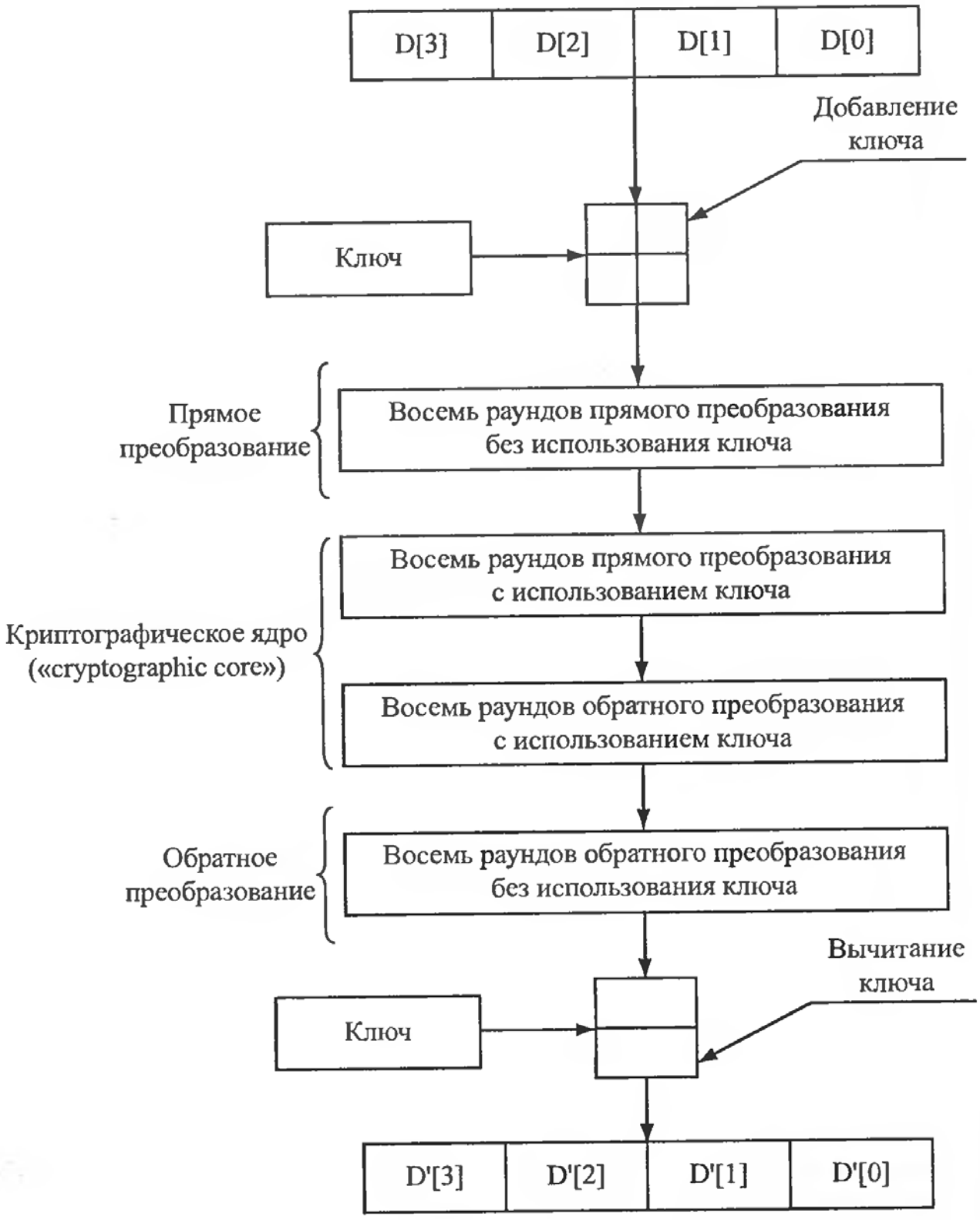


Рис. 4.11. ***Высокоуровневая структура алгоритма шифрования MARS***

Второй этап («криптографическое ядро») состоит из 16 раундов преобразования по схеме Фейстеля с участием ключа. Для того чтобы шифрование и дешифрование имели одинаковую стойкость, первые 8 раундов (из шестнадцати) осуществляют прямое преобразование, последние 8 раундов — обратное.

Последний этап, как и первый, обеспечивает быстрое преобразование и сложение с ключом, но на этот раз для снижения возможности проведения криптографических атак на основе выбранных шифртекстов. Этот этап является обратным по отношению к первому и состоит из восьми раундов, основанных на использовании S-блоков и последующего сложения с ключом.

Ниже приведено более подробное описание алгоритма шифрования MARS. При этом использованы следующие обозначения:

D[] — массив четырех 32-битовх слов. Изначально D обозначает открытый текст, поступающий на вход алгоритма шифрования. В результате шифрования он содержит биты шифртекста (D'[]);

K[] — массив, содержащий расширенный ключ, состоящий из 40 32-битовых слов;

S[] — S-блок, состоящий из 512 32-битовых слов.

4.9.1. Первый этап: прямое преобразование

Как уже говорилось раньше, на этом этапе сначала происходит сложение битов ключа с битами входного сообщения, а после осуществляется 8 раундов преобразования. Алгоритм этапа прямого преобразования изображен на рис. 4.12. При этом использованы следующие обозначения:

⊕ — сложение данных по модулю два;

\+ — целочисленное сложение данных по модулю;

8>>> — циклический сдвиг данных вправо на 8 бит.

В каждом раунде используется одно слово данных (называемое ключевым словом (source word)) для изменения остальных трех слов (называемых целевым словами (target word)). Четыре байта ключевого слова рассматриваются как входы двух S-блоков S_0 и S_1, каждый из которых состоит из 256 32-битовых слов, после чего происходит либо целочисленное сложение данных, либо сложение по модулю 2 соответствующих выходов S-блоков с остальными тремя словами данных.

Если обозначить четыре байта ключевого слова как b_0, b_1, b_2 и b_3 (где b_0 — самый младший байт, а b_3 — самый старший), то байты b_0 и b_2 будут являться входами в блок S_0, а b_1 и b_3 — входами в блок S_1. Первоначально происходит сложение по модулю 2 выхода $S_0[b_0]$ с первым намеченным словом, после чего к полученному результату добавляется выход блока $S_1[b_1]$. Также происходит добавление выхода $S_0[b_2]$ ко второму целевому слову и сложение по модулю 2 выхода $S_1[b_3]$ с третьим целевым словом. В результате этих действий ключевое слово оказывается циклически сдвинутым на 24 бита вправо.

В следующем раунде ключевым словом становится первое целевое слово. Соответственно, бывшее вторым целевое слово становится первым целевым, а слово, бывшее ключевым, становится третьим целевым.

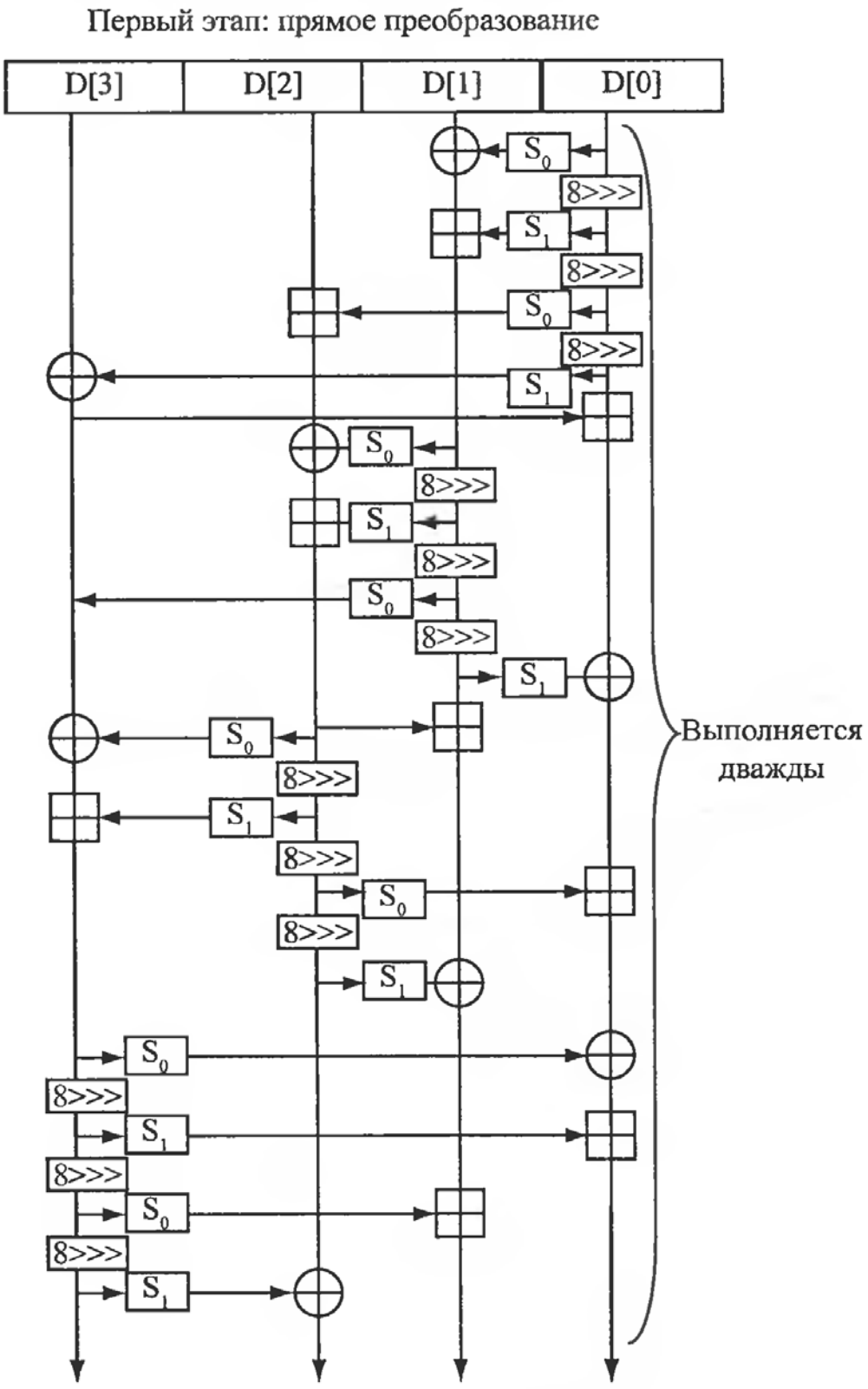


Рис. 4.12. ***Первый этап: прямое преобразование***

Вдобавок, после первого и пятого раундов происходит добавление третьего целевого слова к текущему ключевому слову, а после второго и шестого — добавление первого целевого слова к текущему ключевому слову. Это сделано для устранения некоторых простых атак, которые могут быть направлены на первый этап шифрования. Подробный обзор таких атак приведен в [20].

4.9.2. Второй этап: «криптографическое ядро»

«Криптографическое ядро» алгоритма шифрования MARS представляет собой сеть Фейстеля и состоит из 16 раундов. В каждом раунде используется функция E, зависящая от ключа, которая базируется на необычной комбинации умножения, управляемого циклического сдвига и S-блока. Структура используемой сети Фейстеля изображена на рис. 4.13, а сама функция E показана на рис. 4.14.

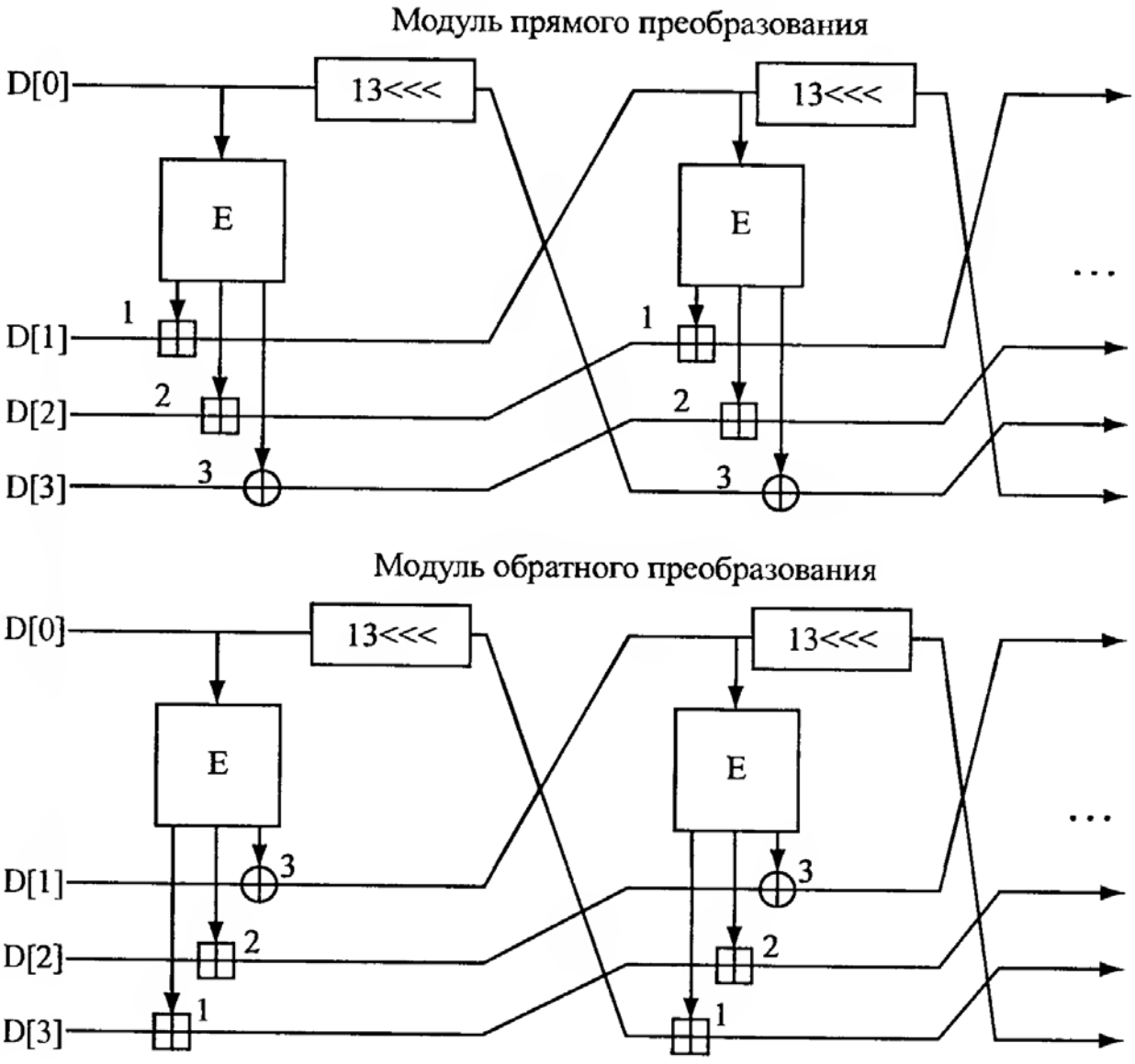


Рис. 4.13. ***Сеть Фейстеля для основного преобразования ключа***

При этом на рисунках использованы следующие обозначения:

⊕ — сложение данных по модулю два;

\+ — целочисленное сложение данных по модулю;

* — целочисленное умножение данных по модулю;

5<<< — циклический сдвиг данных влево на 5 бит;

13<<< — циклический сдвиг данных влево на 13 бит;

<<< — зависимый циклический сдвиг данных влево (аналогичен сдвигу в алгоритме шифрования RC5).

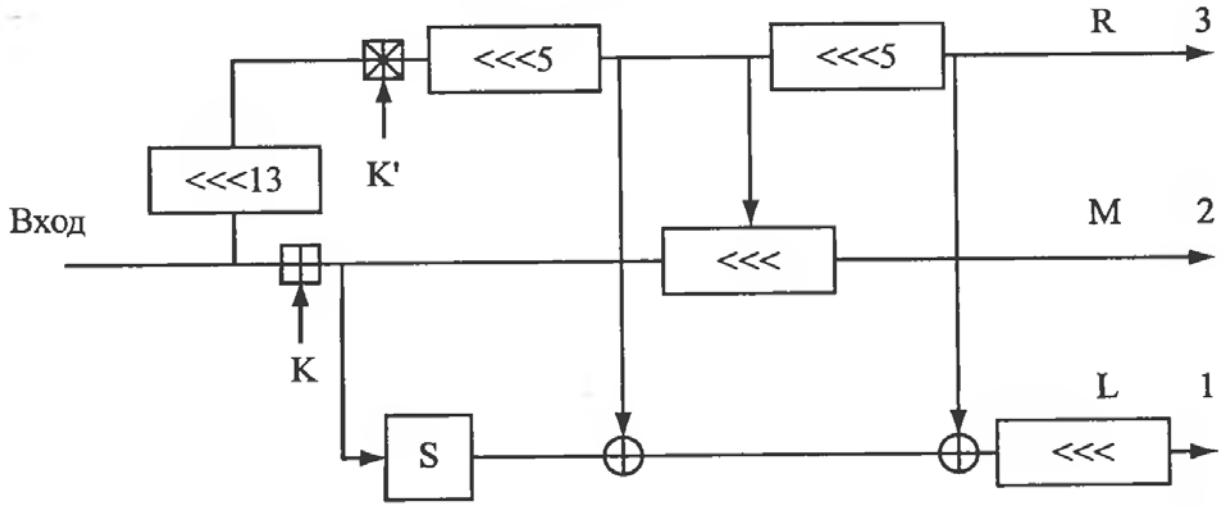


Рис. 4.14. ***Схема функции E***

В каждом раунде используется одно 32-битное слово данных в качестве входов в функцию E. Три выходных слова функции E добавляются или складываются по модулю 2 с остальными словами данных. Вдобавок, ключевое слово в функции E циклически сдвигается на 13 позиций влево.

Для того чтобы алгоритм шифрования имел одинаковую стойкость к атакам как на основе шифртекстов, так и на основе открытых текстов, три выхода функции E в последних восьми раундах используются обратно тому, как они были использованы в первых восьми раундах.

Функция E использует два дополнительных слова ключа для выработки трех выходных слов. В этой функции используются три временных переменных, обозначенных ниже как L, M и R (то есть левая, средняя и правая).

Изначально в R заносится ключевое слово, сдвинутое циклически влево на 13 позиций, в M заносится сумма ключевого слова с первым ключом. После этого младшие 9 битов переменной M используются как вход в S-блок (который получается с помощью объединения блока S_0 и S_1, используемых в предыдущем этапе), а в переменную L заносится соответствующее значение выхода S-блока.

После этого второй ключ умножается со значением переменной R, и полученное значение сдвигается циклически влево на 5 позиций (так что 5 старших зна-

чащих битов после сдвига становятся соответственно пятью младшими битами). После этого значения переменных R и L складываются по модулю 2. Пять младших битов переменной R определяют число позиций (от 0 до 31), на которое необходимо циклически сдвинуть влево значение переменной M. После этого значение переменной R еще раз сдвигается циклически влево на 5 бит, и полученное значение складывается по модулю два со значением переменной L. В очередной раз пять младших битов переменной R определяют число позиций (от 0 до 31), на которое необходимо циклически сдвинуть влево на этот раз значение переменной L. Первым выходным словом функции E будет являться значение переменной L, вторым — переменной M, и третьим — переменной R.

4.9.3. Последний этап

Последний этап аналогичен первому за тем исключением, что данные преобразовываются в обратном порядке. Как и на первом этапе, здесь есть одно ключевое слово и три целевых. Напомним, что четыре байта ключевого слова обозначаются как b_0, b_1, b_2 и b_3. При этом b_0 и b_2 используются в качестве входов в S-блок S_1, а байты b_1 и b_3 — в качестве входов в S_0. Выход $S_1[b_0]$ складывается по модулю два с первым целевым словом, выход $S_0[b_3]$ вычитается из второго целевого слова, а выход $S_1[b_2]$ — из третьего. В итоге ключевое слово сдвигается влево на 24 позиции.

К следующему раунду мы меняем четыре слова местами, то есть текущее первое целевое слово становится ключевым словом, текущее второе целевое становится первым, третье — становится вторым, а текущее ключевое слово становится третьим целевым.

Также перед четвертым и восьмым раундами происходит вычитание первого целевого слова из ключевого, а перед третьим и седьмым раундами — вычитание третьего целевого слова из ключевого. Алгоритм последнего этапа преобразования приведен на рис. 4.15.

4.9.4. Псевдокод алгоритма шифрования MARS

В этом разделе мы приводим псевдокод алгоритма шифрования MARS. В этом псевдокоде все операции применимы к 32-битовым словам, которые рассматриваются как целые беззнаковые числа. В каждом слове биты пронумерованы от 0 до 31, где 0 бит — самый младший значащий, а 31 бит — самый старший значащий.

Операция $a \oplus b$ обозначает побитовое сложение по модулю два двух слов a и b. Операции $a \wedge b$ и $a \vee b$ обозначают побитовую операцию «И» и «ИЛИ», соответственно. Операция $a + b$ обозначает сложение по модулю 2^{32}, операция $a - b$ — вычитание по модулю 2^{32}, а операция $a \times b$ — умножение по модулю 2^{32}.

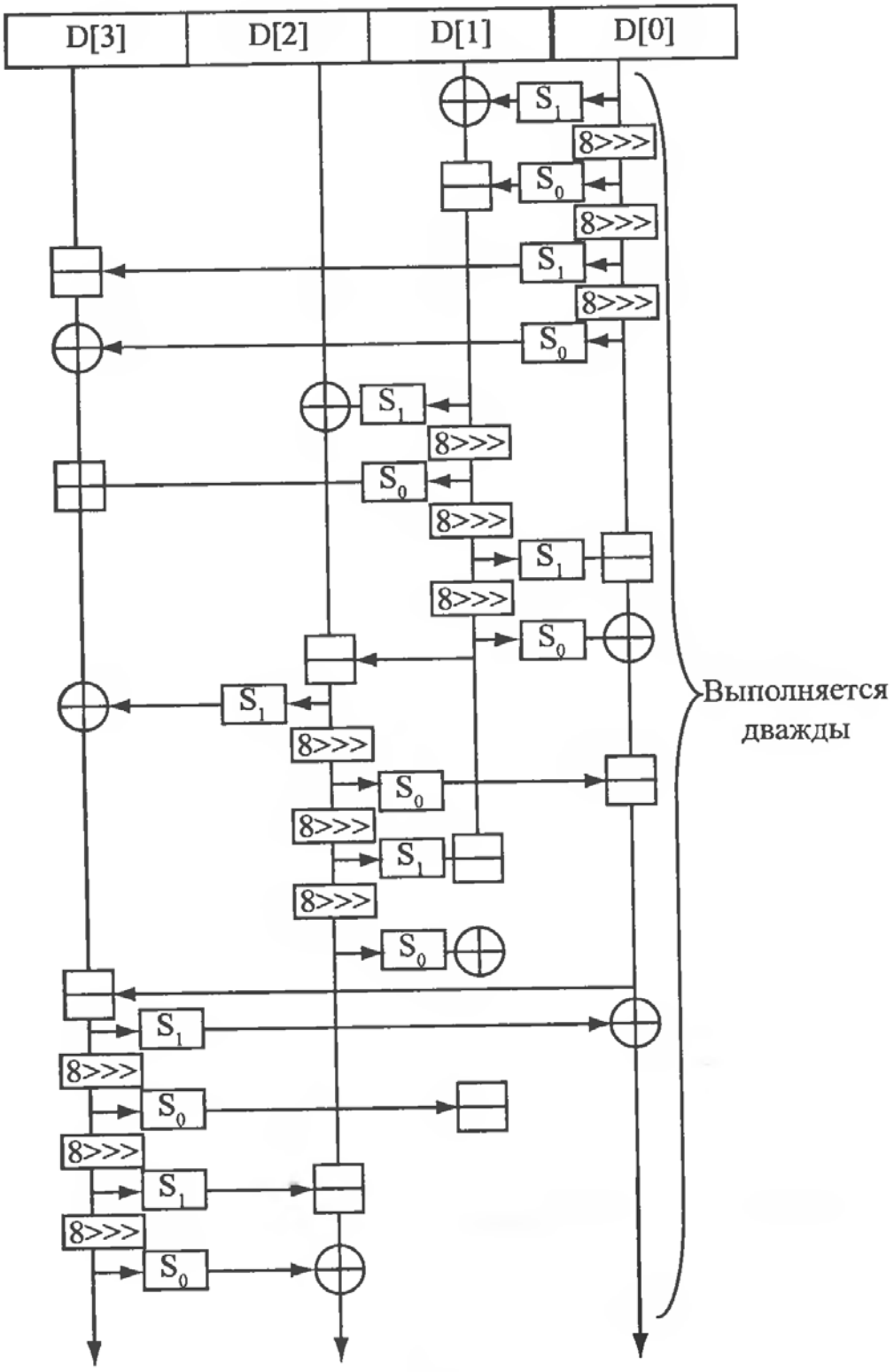


Рис. 4.15. ***Алгоритм последнего этапа преобразования***

Также операции (a<<<b) и (a>>>b), соответственно, обозначают циклический сдвиг 32-битового слова a на b позиций влево и вправо. При циклическом сдвиге влево на b позиций, бит из позиции i перемещается в позицию i + b, взятую по модулю 2^{32} (например, самый младший бит переместится из позиции 0 в позицию

b). Аналогично, циклический сдвиг вправо на b позиций означает, что бит с позиции i переместится в позицию i – b, взятую по модулю 32.

Функция E (входы in, Key1, Key2)

```
//Используются три промежуточных переменных L, M и R
M = in + key1
R = (in<<<13) * key2
I = младшие 9 бит M
L = S[I]
R = R<<<5
r = младшие 5 бит R
M = M<<<r
L = L ⊕ R
R = R<<<5
L = L ⊕ R
r = младшие 5 бит R
L = L<<<r
Выходы (L, M, R)
```

Шифрование MARS (входы D[], K[])

```
Первый этап: прямое преобразование
//Начальное сложение данных с подключом
for I = 0 to 3 do
    D[I] = D[I] + K[I]
    //Выполняется 8 раундов прямого преобразования
    for I = 0 to 7 do//Используем D[0] для изменения D[1], D[2], D[3]
    //Четыре замены в S-блоках
    D[1] = D[1] ⊕ S0[младший байт D[0]]
    D[1] = D[1] + S1[второй байт D[0]]
    D[2] = D[2] + S0[третий байт D[0]]
    D[3] = D[3] ⊕ S1[старший байт D[0]]
    //Сдвиг ключевого слова вправо
    D[0] = D[0]>>>24
    If I = 0 or 4 then
        D[0] = D[0] + D[3] //добавляем D[3] к ключевому слову
    If I = 1 or 5 then
        D[0] = D[0] + D[1] //добавляем D[1] к ключевому слову
    //сдвигаем D[] на одно слово вправо для следующего раунда
    (D[3], D[2], D[1], D[0]) ← (D[0], D[3], D[2], D[1])
end-for
Второй этап: «Криптографическое ядро»
//Выполняется 16 раундов преобразования
```

```
for I = 0 to 15 do
    (out1, out2, out3) = E(D[0], K[2i + 4], K[2i + 5])
    D[0] = D[0]<<<13
    D[2] = D[2] + out2
        If I<8 then          //Первые восемь раундов
            D[1] = D[1] + out1
            D[3] = D[3] ⊕ out3
        Else                 //Последние восемь раундов
            D[3] = D[3] + out1
            D[1] = D[1] ⊕ out3
        End-if
    //Сдвигаем D[] на одно слово вправо для следующего раунда
    (D[3], D[2], D[1], D[0]) ← (D[0], D[3], D[2], D[1])
end-for
Третий этап: обратное преобразование
//Выполняется восемь раундов обратного преобразования
for I = 0 to 7 do
    if I = 2 or 6 then
        D[0] = D[0] — D[1] //Вычитаем D[3] из ключевого слова
    If I = 3 or 7 then
        D[0] = D[0] — D[1] //Вычитаем D[1] из ключевого слова
    //Четыре замены в S-блоках
    D[1] = D[1] ⊕ S1[младший байт D[0]]
    D[2] = D[2] — S0[второй байт D[0]]
    D[3] = D[3] + S1[третий байт D[0]]
    D[3] = D[3] ⊕ 01[старший байт D[0]]
    //Сдвиг ключевого слова влево
    D[0] = D[0]<<<24
    //Сдвигается D[] на одно слово вправо для следующего раунда
    (D[3], D[2], D[1], D[0]) ← (D[0], D[3], D[2], D[1])
end-for
//Вычитаем подключ из полученных данных
for I = 0 to 3 do
    D[I] = D[I] — K[36 + I]
```

4.9.5. Дешифрование

Процесс дешифрования обратен процессу зашифрования. Псевдокод для дешифрования аналогичен коду зашифрования.

4.9.6. Процедура извлечения подключа

С помощью процедуры извлечения подключа мы получаем массив ключей k[], состоящий из n 32-битовых слов (где n обозначает любой номер от 4 до 14) из массива K[], состоящего из 40 слов. Процедура извлечения ключа гарантирует, что слова ключа, которые используются для умножения в процедуре зашифрования, имеют следующие свойства:

I. Два младших бита в слове ключа, которое используется при умножении, установлены в 1;

II. Ни одно из слов ключа не содержит десять 0 или 1, идущих подряд.

Итак, процедура извлечения ключа состоит из следующих пунктов:

1. Сначала исходный ключ копируется во вспомогательный массив T[], состоящий из 15 слов. При этом сначала заносится n слов ключа, а остальные ячейки заполняются нулями.

$$T[0\ldots n-1] = k[0\ldots n-1],\ T[n] = n,\ T[n+1\ldots 14] = 0;$$

2. После этого нижеописанный процесс повторяется четыре раза, где каждая итерация вычисляет следующие десять слов извлекаемого подключа.

Массив T[] преобразуется с помощью использования следующей линейной формулы:

i меняется от 1 до 14,

$$T[i] = T[i] \oplus ((T[i-7 \bmod 15] \oplus T[i-2 \bmod 15])<<<3) \oplus (4i+j),$$

где j обозначает номер текущей итерации (0 для первой итерации, 1 для второй и т. д.).

Далее перемешивается массив T[] с использованием модифицированной сети Фейстеля, то есть четыре раза повторяется следующая операция:

$$T[i] = (T[i] + S[\text{младшие 9 бит } T[i-1 \bmod 15]])<<<9,$$

где i меняется от 0 до 14.

После этого берется десять слов массива T[] и записываются в массив подключа в следующем порядке:

$$K[10j+i] = T[4i \bmod 15],$$

где i меняется от 0 до 9;

3. В заключение изменяются шестнадцать слов, которые используются в алгоритме шифрования для умножения (это слова K[5], K[7], ..., K[35]) таким образом, чтобы им были присущи свойства I и II. Заметим, что вероятность того, что случайно выбранному слову не присуще второе свойство, равна примерно 1/41. Изменение слов осуществляется следующим образом:

а) два младшие бита K[i] определяем как j = K[i] ^ 3 (имеется ввиду операция AND), после чего считаем, что слово с двумя этими битами установлено в 1 и w = K[i] $\vee$ 3 (операция OR);

б) строится маска M из битов w, которые принадлежат последовательности из десяти 0 или 1, то есть M_i будет равно 1 в том и только том случае, если w_i принадлежит последовательности из десяти 0 или 1. После этого сбрасываются в ноль все единицы в конечных точках последовательностей 0 и 1 в w, а также два младших бита и старший бит маски M. Иначе говоря, i-й бит маски M установится в 0, если $i < 2$, $i = 31$ или i-й бит последовательности w отличается от $(i + 1)$-го или $(i - 1)$-го бита.

Например, предположим, что у нас есть $w = 0^3 1^{13} 0^{12} 1011$ (где за 0^i, 1^i обозначена последовательность из i нулей и единиц, соответственно). В этом случае мы сперва определяем маску M как $M = 0^3 1^{25} 0^4$, а после этого сбрасываем единицу в позициях 4, 15, 16 и 28 и получаем $M = 0^4 1^{11} 001^{10} 0^5$.

в) далее используется фиксированная таблица B, состоящая из 4 слов, где 4 входа в B выбраны таким образом, что они не содержат последовательностей из 7 нулей и 10 единиц. Обычно используется таблица B[] = {0xa4a8d57b, 0x5b5d193b, 0xc8a8309b, 0x73f9a978}. Мы используем два записанных бита j (в пункте а)) для выбора входа из B и используем младшие 5 битов K[i – 1] для преобразования этого входа следующим образом: p = B[j]<<<(младшие пять бит K[i – 1]).

В заключение складываются по модулю два исходное значение w с полученным значением p с наложенной на него маской M, и результат записывается в K[i]:

$$K[i] = w \oplus (p \wedge M).$$

Ниже приведен псевдокод для процедуры извлечения расширенного ключа.

Расширение ключа (входы: k[], n; выход: K[])

```
//n    — количество слов в буфере ключей k[], (1 ≤ n ≤ 14)
// K[] — массив расширенных ключей, состоящий из 40 слов
//T[]  — временный массив, состоящий из 15 слов
//B[]  — фиксированная таблица из четырех слов
//Инициализация B[]
B[] = {0xa4a8d57b, 0x5b5d193b, 0xc8a8309b, 0x73f9a978}
//Инициализация T[] с помощью данных ключа
T[0...n – 1] = k[0...n – 1], T[n] = n, T[n + 1...14] = 0
For j = 0 to 3 do
   For I = 0 to 14 do
      T[I] = T[I] ⊕ ((T[I – 7 mod 15] ⊕ T[I – 2 mod 15]) <<<3) ⊕ (4i + j)
Повтор 4 раза
   For I = 0 to 14 do
      T[i] = (T[i] + S[младшие 9 бит T[i – 1 mod 15]]) <<<9)
Конец повтора
   For I = 0 to 9 do
      K[10I + j] = T[4i mod 15]
```

```
    For I = 5,7,…,35 do
        j = младшие два бита K[I]
        w = K[I] с двумя младшими битами, установленными в 1
// Генерируем маску M
    M_I = 1 iff w_I принадлежит последовательности из десяти 0 или 1 в w
        а также 2 ≤ I ≤ 30 и w_{I-1} = w_I = w_{I+1}
    r = младшие 5 бит K[i – 1]
    p = B[j]<<<r
// Изменяем K[i] с помощью p под воздействием маски M
    K[i] = w ⊕ (p ∧ M)
Конец цикла for.
```

4.10. Serpent

Шифр «Змей» (Serpent) последний из совместных проектов англичанина Росса Андерсона и израильтянина Эли Бихама, имеющих в своем активе уже целый зверинец криптоалгоритмов: Bear, Lion, Tiger (медведь, лев, тигр). Для разработки алгоритма шифрования Serpent авторы специально пригласили датчанина Ларса Кнудсена (работающего в университете Бергена, Норвегия), знаменитого своими крайне успешными криптоаналитическими работами по вскрытию блочных шифров, чтобы он выявил все возможные слабые места в стойкости нового алгоритма. Поскольку и сам Бихам — соавтор метода дифференциального анализа, лежащего в основе современных методов вскрытия блочных шифров, то в результате такого сотрудничества появился чрезвычайно стойкий алгоритм. «Ультраконсервативный по запасу прочности», как охарактеризовали его в NIST, так как, создав шифр, противостоящий всем известным на сегодня атакам, разработчики затем удвоили количество его итераций. Как следствие этого, по своей скорости Serpent оказывается медленнее, чем четыре остальных финалиста конкурса AES (MARS, TwoFish, RC6 и Rijndael). Тем не менее для повышения производительности разработчиками предложен специальный метод.

Алгоритм Serpent хорошо ложится в аппаратное исполнение и в ограниченные по памяти устройства. Реализующая SP-сеть имеет простую конструкцию, а консервативный выбор операций существенно упрощают анализ.

Алгоритм шифрования Serpent (рис. 4.16) представляет собой 32-храундовую сеть SP. Алгоритм преобразует 128 битовый вход открытых данных в 128 битовый выход шифрованных данных при участии 33 128-битных подключей.

Serpent включает в свой состав следующие основные действия:

— начальную перестановку IP;
— 32 раунда Rk, где k меняется от 0 до 31 (n = 31), криптографических преобразований, каждый из которых состоит из операции сложения с ключом,

прохождения через S-блоки и линейного преобразования. В последнем раунде линейное преобразование заменено операцией сложения с ключом;

— конечную перестановку FP.

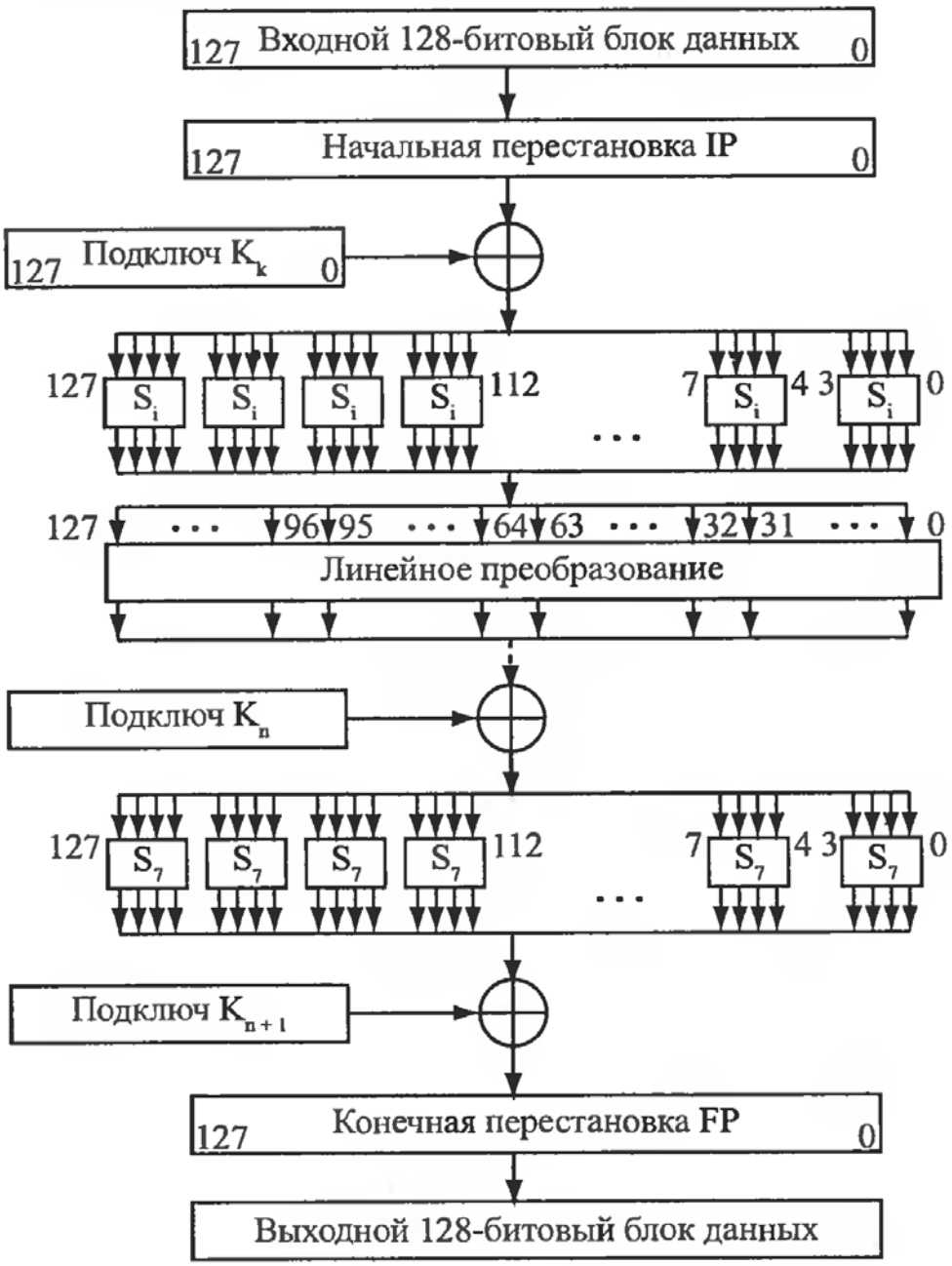


Рис. 4.16. ***Алгоритм шифрования Serpent***

Начальная (таблица 4.5) и конечная (таблица 4.6) перестановки не имеют какого-либо криптографического значения. Они предназначены для упрощения оптимизации использования шифра и для улучшения его вычислительной эффективности. Замена в этих таблицах осуществляется аналогично замене, производимой в алгоритме шифрования DES, то есть число 32 на пересечении первой строки и второго столбца таблицы 4.5 означает, что 32 бит входного сообщения окажется

на втором месте. Аналогично алгоритму шифрования DES конечная перестановка является обратной по отношению к начальной.

Таблица 4.5

Начальная перестановка IP

0	32	64	96	1	33	65	97	2	34	66	98	3	35	67	99
4	36	68	100	5	37	69	101	6	38	70	102	7	39	71	103
8	40	72	104	9	41	73	105	10	42	74	106	11	43	75	107
12	44	76	108	13	45	77	109	14	46	78	110	15	47	79	111
16	48	80	112	17	49	81	113	18	50	82	114	19	51	83	115
20	52	84	116	21	53	85	117	22	54	86	118	23	55	87	119
24	56	88	120	25	57	89	121	26	58	90	122	27	59	91	123
28	60	92	124	29	61	93	125	30	62	94	126	31	63	95	127

Таблица 4.6

Конечная перестановка FP

0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
64	68	72	76	80	84	88	92	96	100	104	104	112	116	120	124
1	5	9	13	17	21	25	29	33	37	41	45	49	53	57	61
65	69	73	77	81	85	89	93	97	101	105	109	113	117	121	125
2	6	10	14	18	22	26	30	34	38	42	46	50	54	58	62
66	70	74	78	82	86	90	94	98	102	106	110	114	118	122	126
3	7	11	15	19	23	27	31	35	39	43	47	51	55	59	63
67	71	75	79	83	87	91	95	99	103	107	111	115	119	123	127

В каждом раунде R используется один повторяющийся S-блок. Так, например, R_0 использует блок S_0, 32 копии которого располагаются параллельно. Таким образом, на вход первой копии S_0 поступят первые четыре бита исходного сообщения, сложенного по модулю два с ключом. На рис. 4.16 в каждом раунде шифрования S-блок имеет индекс i, который определяется следующим образом:

$$i := (k - 1) \bmod 8,$$

где k — текущее значение цикла (переменная n обозначает общее количество циклов в алгоритме, то есть 32).

После этого к полученному промежуточному вектору применяется линейное преобразование, результат которого поступает на вход следующего раунда. Каждый из восьми имеющихся S-блоков используется четыре раза, то есть после применения блока S_7 следующим используемым блоком будет блок S_0.

S-блоки алгоритма шифрования Serpent представляют собой 4-битовую замену, обладающую следующими свойствами [14]:

— каждая разностная характеристика имеет вероятность не больше чем 1/4, и разность одного входного бита никогда не приведет к разности одного выходного бита;
— каждая линейная характеристика имеет вероятность в пределах $1/2 \pm 1/4$, и линейные отношения между одним отдельным битом входа и одним отдельным битом выхода имеют вероятность в диапазоне $1/2 \pm 1/8$;
— должно достигаться максимальное нелинейное распределение выходных битов.

S-блоки замены, используемые при зашифровании, и инверсные им блоки, используемые при дешифровании, приведены соответственно в таблице 4.7 и таблице 4.8.

Таблица 4.7

S-блоки замены, используемые при шифровании

S_0	3	8	15	1	10	6	5	11	14	13	4	2	7	0	9	12
S_1	15	12	2	7	9	0	5	10	1	11	14	8	6	13	3	4
S_2	8	6	7	9	3	12	10	15	13	1	14	4	0	11	5	2
S_3	0	15	11	8	12	9	6	3	13	1	2	4	10	7	5	14
S_4	1	15	8	3	12	0	11	6	2	5	4	10	9	14	7	13
S_5	15	5	2	11	4	10	9	12	0	3	14	8	13	6	7	1
S_6	7	2	12	5	8	4	6	11	14	9	1	15	13	3	10	0
S_7	1	13	15	0	14	8	2	11	7	4	12	10	9	3	5	6

Таблица 4.8

Инверсные S-блоки, используемые при дешифровании

$InvS_0$	13	3	11	0	10	6	5	12	1	14	4	7	15	9	8	2
$InvS_1$	5	8	2	14	15	6	12	3	11	4	7	9	1	13	10	0
$InvS_2$	12	9	15	4	11	14	1	2	0	3	6	13	5	8	10	7
$InvS_3$	0	9	10	7	11	14	6	13	3	5	12	2	4	8	15	1
$InvS_4$	5	0	8	3	10	9	7	14	2	12	11	6	4	15	13	1
$InvS_5$	8	15	2	9	4	1	13	14	11	6	5	3	7	12	10	0
$InvS_6$	15	10	1	13	5	3	6	0	4	9	14	7	2	12	8	11
$InvS_7$	3	0	6	13	9	14	15	8	5	12	11	7	10	1	4	2

Выходы, поступающие из S-блоков, подвергаются линейному преобразованию. Линейное преобразование состоит из трех основных операций: прямой перестановки IP (приведенной в таблице 4.5), самого линейного преобразования и конечной перестановки FP (приведенной в таблице 4.6). Таким образом, в рамках линейного преобразования выходы S-блоков подвергаются IP перестановке, после чего объединяются в четыре группы по 32 бита: X_0, X_1, X_2, X_3, где X_0 —

самая младшая группа (то есть содержит биты с 0 по 31), а X_3 — самая старшая (содержит биты с 96 по 127). И к ним применяется линейное преобразование (см. рис. 4.17). В него входят операции сдвига, циклического сдвига и сложения по модулю 2 по следующей схеме:

$$X_0, X_1, X_2, X_3 := S_i(B_i \oplus K_i);$$
$$X_0 := X_0 <<< 13;$$
$$X_2 := X_2 <<< 3;$$
$$X_1 := X_1 \oplus X_0 \oplus X_2;$$
$$X_3 := X_3 \oplus X_2 \oplus (X_0 << 3);$$
$$X_1 := X_1 <<< 1;$$
$$X_3 := X_3 <<< 7;$$
$$X_0 := X_0 \oplus X_1 \oplus X_3;$$
$$X_2 := X_2 \oplus X_3 \oplus (X_1 << 7);$$
$$X_0 := X_0 <<< 5;$$
$$X_2 := X_2 <<< 22;$$
$$B_{i+1} := X_0, X_1, X_2, X_3;$$

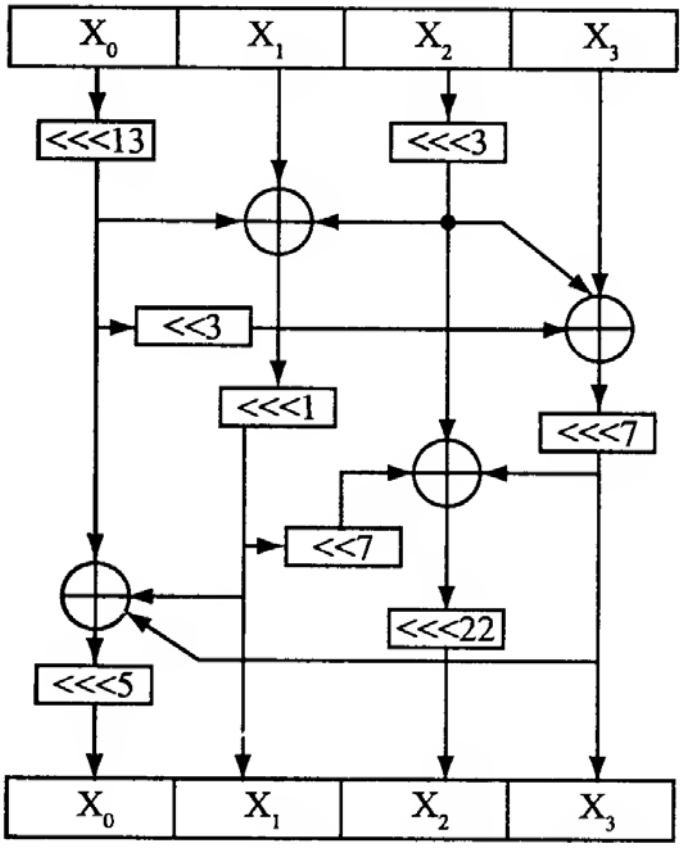


Рис. 4.17. ***Линейное преобразование, используемое в алгоритме шифрования Serpent***

В вышеприведенном алгоритме два знака <, стоящие подряд перед числом обозначают сдвиг влево на количество разрядов, равное этому числу, а три анало-

гичных знака — циклический сдвиг влево. B_i — это блок данных, поступающий на вход очередного раунда алгоритма шифрования, а B_{i+1} — соответствующий ему блок выходных данных. K_i — это очередной секретный ключ шифрования. После этого полученные значения X_0, X_1, X_2, X_3 подвергаются конечной перестановке FP.

Для определения 33 128-битных ключей требуется 256-битный ключ пользователя. Он расширяется по следующей схеме:

$$w_i := (w_{i-8} \oplus w_{i-5} \oplus w_{i-3} \oplus w_{i-1} \oplus \theta \oplus i) <<< 11,$$

где $w_{i-1} \ldots w_{i-8}$ — 8 32-битных слов исходного 256-битного ключа;

$w_0 \ldots w_{131}$ — расширенные промежуточные ключи;

θ — дробная часть от соотношения $\frac{(\sqrt{5}+1)}{2}$, или 0x9e3779b9 в шестнадцатеричной системе счисления.

После этого раундовые ключи вычисляются из промежуточных подключей с использованием S-блоков по следующему принципу:

$$\{k_0, k_1, k_2, k_3\} := S_3(w_0, w_1, w_2, w_3)$$
$$\{k_4, k_5, k_6, k_7\} := S_3(w_4, w_5, w_6, w_7)$$
$$\{k_8, k_9, k_{10}, k_{11}\} := S_3(w_8, w_9, w_{10}, w_{11})$$
$$\{k_{12}, k_{13}, k_{14}, k_{15}\} := S_3(w_{12}, w_{13}, w_{14}, w_{15})$$
$$\{k_{16}, k_{17}, k_{18}, k_{19}\} := S_3(w_{16}, w_{17}, w_{18}, w_{19})$$
$$\vdots$$
$$\{k_{124}, k_{125}, k_{126}, k_{127}\} := S_3(w_{124}, w_{125}, w_{126}, w_{127})$$
$$\{k_{128}, k_{129}, k_{130}, k_{131}\} := S_3(w_{128}, w_{129}, w_{130}, w_{131})$$

В итоге 32-битовые значения k_j перенумеровываются как 128-битовые подключи K_i (для $i \in \{0, \ldots, r\}$) следующим образом:

$$K_i := \{k_{4i}, k_{4i+1}, k_{4i+2}, k_{4i+3}\}.$$

Дешифрование отличается от шифрования тем, что инверсные S-блоки должны использоваться в обратном порядке точно так же, как инверсное линейное преобразование и обратный порядок подключей.

4.11. RC6

Алгоритм является продолжением криптоалгоритма RC5. 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 стойкости согласно проведенным исследованиям задолго до объявления конкурса AES [25].

Алгоритм является сетью Фейстеля с 4 ветвями смешанного типа: в нем два четных блока (A и C) используются для одновременного изменения содержимого двух нечетных блоков (B и D). Затем производится обычный для сети Фейсте-

ля сдвиг на одно машинное слово, что меняет четные и нечетные блоки местами. Сам алгоритм предельно прост и изображен на рис. 4.18. 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 [43].

Преобразование T(x) очень просто: $T(X) = (X*(X + 1)) \bmod 2^N$. Оно используется в качестве нелинейного преобразования с хорошими показателями перемешивания битового значения входной величины.

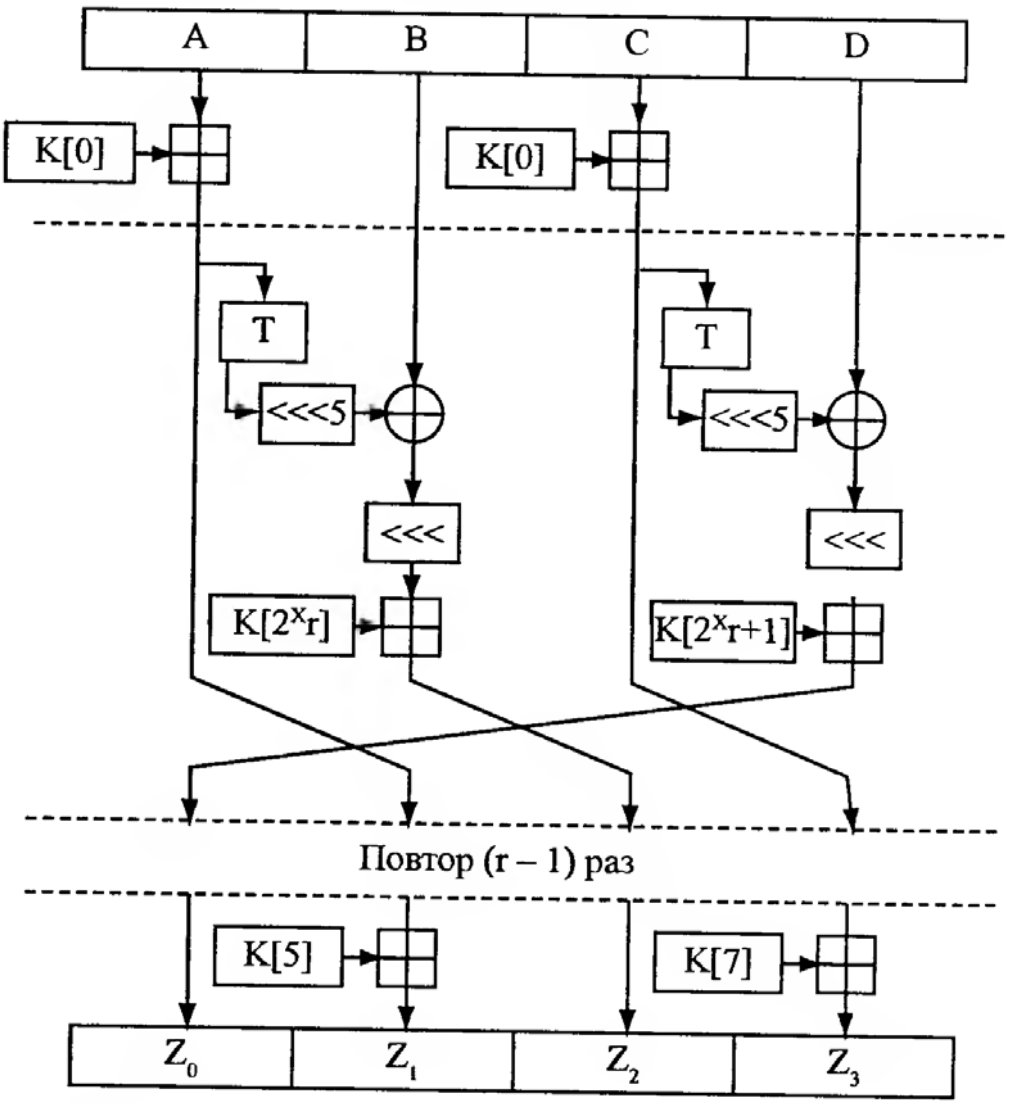


Рис. 4.18. ***Алгоритм шифрования RC6***

4.12. TwoFish

Алгоритм разработан компанией Counterpain Security Systems, возглавляемой Брюсом Шнайером (Bruce Schneier). Предыдущая программная разработка этой

фирмы, называвшаяся BlowFish, является признанным криптостойким алгоритмом.

В алгоритме TwoFish разработчики оставили некоторые удачные решения из проекта-предшественника, кроме этого произвели тщательные исследования по перемешиванию данных в сети Фейстеля. Алгоритм представляет собой сеть Фейстеля смешанного типа: первая и вторая ветвь (X_0 и X_1) на нечетных раундах производят модификацию третьей и четвертой (X_2 и X_3), на четных раундах ситуация меняется на противоположную (см. рис. 4.19).

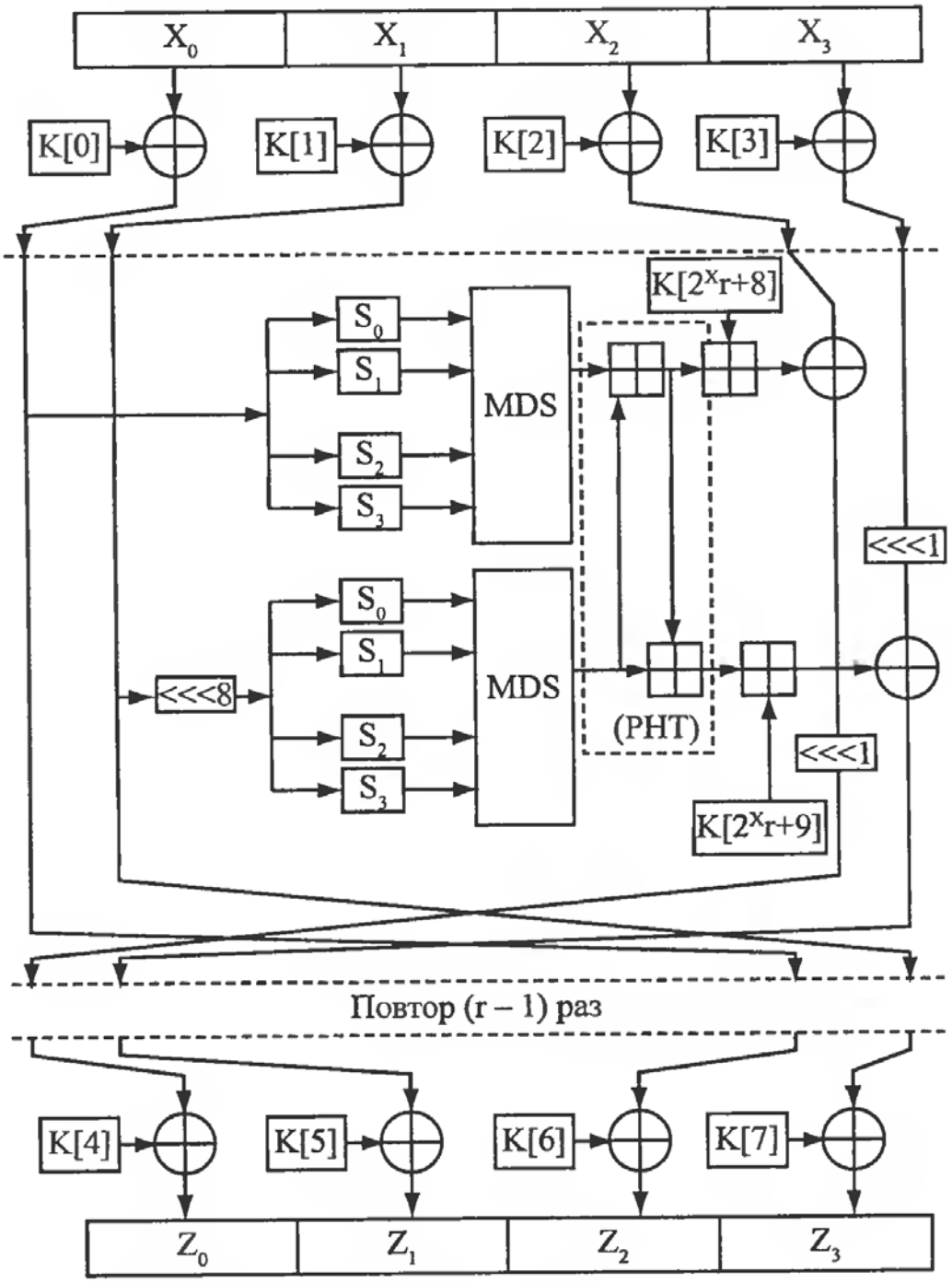


Рис. 4.19. ***Алгоритм шифрования TwoFish***

В алгоритме используется обратимое арифметическое сложение первого потока со вторым, а затем второго с первым.

Единственным нарицанием, поступившим в адрес TwoFish от независимых исследователей, является тот факт, что при расширении битов 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 [44].

4.13. Lucifer

Алгоритм шифрования Lucifer представляет собой шифровальную сеть на основе замен и перестановок, его основные блоки напоминают блоки алгоритма шифрования DES. В отличие от алгоритма DES половины блоков данных между раундами не меняются местами. У алгоритма Lucifer 16 раундов, 128-битовые блоки данных и более простая, чем в DES схема извлечения подключей. Кроме того, в алгоритме шифрования Lucifer отсутствуют начальная и конечная перестановки. Функция F преобразует 64 бита правой половины данных, при этом используется 64-битовый подключ и восемь меняющихся контрольных битов (см. рис. 4.20). В алгоритме Lucifer в функции F используется всего два S-блока замены S_0 и S_1, меняющих 4-битовые входы на 4-битовые выходы. Вход S-блоков представляет собой перемешанный выход S-блоков предыдущего раунда, входом S-блоков первого раунда является открытый текст. Для выбора используемого S-блока из двух возможных используется бит ключа. Выход S-блоков складывается по модулю два с битами ключа, и эта операция называется «ключевым барьером». Последним этапом функции F является перестановка выходных битов. В [27] финальная перестановка представлена в виде двух этапов. На первом этапе каждый байт подвергается фиксированной перестановке, а на втором — биты перемешиваются между байтами (каждый бит входит в разный байт в ту позицию, в которой он находился в исходном байте). Второй этап еще называют диффузией. На рис. 4.20 два эти этапа объединены в один блок P.

На рис. 4.20 пары смежных S-блоков представлены как единый объединенный блок, названный блоком T (Transposition block — преобразованный блок). T-блоки являются функциями, преобразующими девять входных битов (восемь битов данных и один меняющийся контрольный бит) в восемь выходных. Если значение XY является входным байтом данных в T-блок, где X является старшей половиной байта, а Y — соответственно младшей, тогда, если контрольный бит равен нулю, то выходом T-блока будет результат перестановок $S_0[X]$ и $S_1[Y]$. Если же контрольный бит равен единице, то результатом работы T-блока будут значение $S_0[Y]$ и $S_1[X]$.

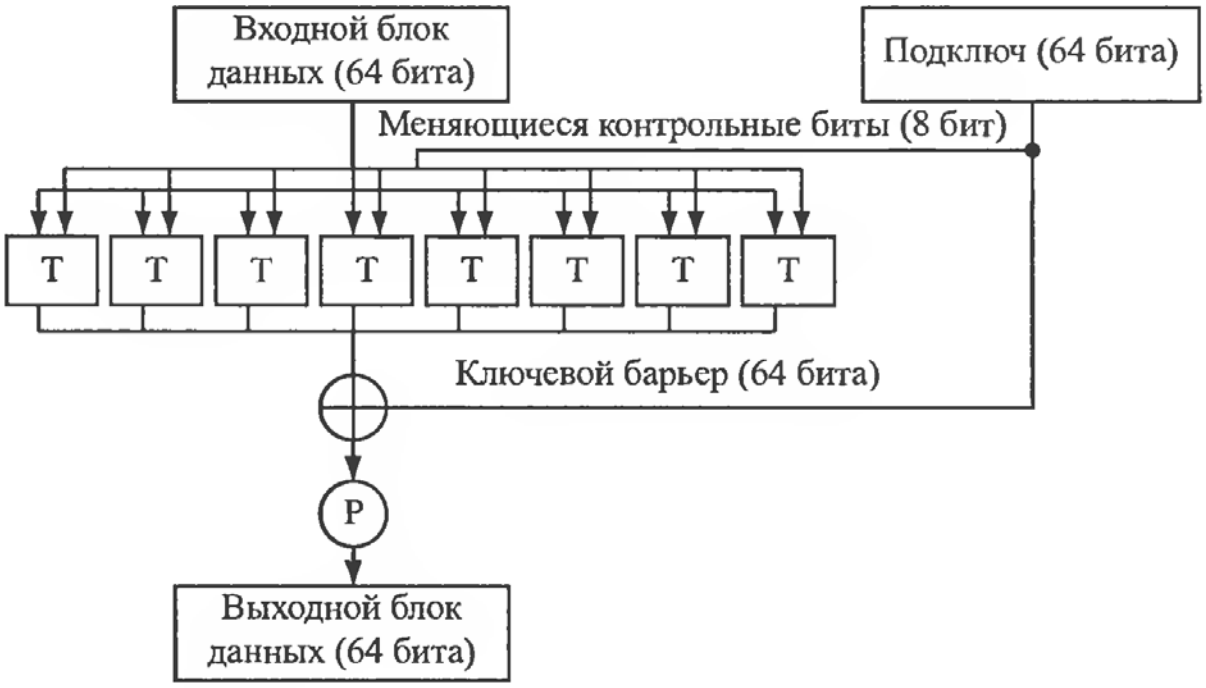


Рис. 4.20. ***Функция F алгоритма шифрования Lucifer***

Алгоритм выработки подключа в алгоритме Lucifer гораздо легче, чем в алгоритме шифрования DES. Ключ помещается в 128-битовый сдвиговый регистр. Каждый раунд в качестве подключа берутся 64 самых левых бита, а в качестве контрольных битов — 8 самых правых. После каждого раунда шифрования происходит циклический сдвиг регистра влево на 56 позиций.

4.14. Контрольные вопросы

1. Дайте описание алгоритма шифрования RC-32/12/16.
2. Охарактеризуйте один раунд криптографического преобразования алгоритма шифрования IDEA.
3. Чем отличается процесс зашифрования от процесса дешифрования в алгоритме шифрования SAFER?
4. Охарактеризуйте работу функции f в алгоритме шифрования FEAL.
5. Охарактеризуйте процесс извлечения раундового подключа из ключа шифрования в алгоритме шифрования FEAL.
6. Как генерируются S-блоки замены в алгоритме шифрования Blowfish?
7. В чем состоит отличие алгоритмов шифрования Khufu и Khafre?
8. Из каких основных этапов состоит алгоритм шифрования MARS? Дайте их краткое описание.
9. Охарактеризуйте процесс извлечения раундового подключа из ключа шифрования в алгоритме шифрования MARS.
10. Из каких основных этапов состоит алгоритм шифрования Serpent?
11. Что вы знаете о таких алгоритмах шифрования, как алгоритм шифрования RC6 и алгоритм шифрования TwoFish?

Глава 5. ЛИНЕЙНЫЙ КРИПТОАНАЛИЗ БЛОЧНЫХ АЛГОРИТМОВ ШИФРОВАНИЯ

5.1. Общие сведения о линейном криптоанализе

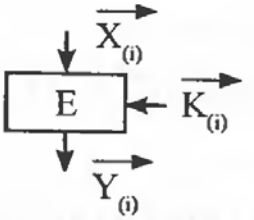
Линейный анализ базируется на знании криптоаналитиком открытого и зашифрованного текста при использовании блочных схем шифрования, таких как DES. Одним из первых, кто вплотную занялся данным видом криптоанализа, был М. Матсуи [6, 17]. В связи с тем, что алгоритм шифрования DES является открытым, то есть заранее известны все его таблицы перестановок и замен, то Матсуи и взял его для криптоанализа. Как видно из таблицы 5.1, линейный криптоанализ Матсуи алгоритма DES требует значительного числа открытых текстов.

Таблица 5.1

Количество циклов алгоритма DES	Количество необходимых известных открытых текстов для нахождения ключа
8	2^{21}
12	2^{33}
16	2^{47}

Линейный криптоанализ основывается на том, что существует возможность заменить нелинейную функцию ее линейным аналогом. Так как все зашифровываемые тексты представлены в двоичном виде, то будем рассматривать случайную величину S — {0,1}, для которой P(S = 1) = p, соответственно P(S = 0) = 1 – p, а $\Delta(S) = |1 - 2p|$.

Рассмотрим схему произвольного блочного шифра в i-м цикле:



$$Y_{(i)} = E(X_{(i)}, K_{(i)}). \qquad (5.1)$$

Здесь E — функция шифрования, $X_{(i)}$ — блок открытого текста в i-м цикле, $Y_{(i)}$ — блок шифртекста, $K_{(i)}$ — подключ, используемый в i-м цикле. $Y_{(i)}, X_{(i)} \in V_n$, $K_{(i)} \in V_m$, где V — двоичный вектор, n — размер блока, m — размер подключа.

Обозначим через $(X, \alpha) = X_1\alpha_1 \oplus \ldots \oplus X_n\alpha_n = X_{i1} \oplus \ldots \oplus X_{ik} = X[i_1, \ldots, i_k]$ — скалярное произведение двух двоичных векторов X и α, где $(\alpha_{i1}, \ldots, \alpha_{ik})$ — единичные координаты вектора α, а по сути дела сложение по модулю 2 битов вектора X, соответствующих ненулевым битам вектора α.

Линейным статистическим аналогом нелинейной функции (5.1) называется случайная величина

$$S_{(i)} = (Y_{(i)}, \alpha_{(i)}) \oplus (X_{(i)}, \beta_{(i)}) \oplus (K_{(i)}, \chi(i)), \qquad (5.2)$$

если вероятность $P(S_{(i)} = 1) = p \neq 1/2$ для случайно выбранного открытого текста $X_{(i)}$ [4].

Отклонение $\Delta(S_{(i)}) = |1 - 2p|$ определяет эффективность линейного статистического аналога (5.2).

Для применения линейного криптоанализа необходимо решить следующие задачи:

1. Нахождение эффективного статистического линейного аналога и вычисление его вероятности;
2. Определение ключа (или некоторых битов ключа) с использованием эффективного линейного статистического аналога.

5.2. Линейный криптоанализ алгоритма шифрования DES

5.2.1. Криптоанализ одного раунда алгоритма шифрования DES

Наиболее эффективным является рассмотрение блока F алгоритма DES, а именно находящихся в нем S-блоков, так как именно в этом случае можно явно проследить зависимость выходных битов от входных в их всевозможных комбинациях. Один цикл DES преобразования изображен на рис. 3.3.

Нелинейная функция, реализующая a-й S-блок алгоритма DES, может быть записана в виде:

$$Y = F_a(X \oplus K),\ a = 1, \ldots, 8,\ Y \in V_4,\ X, K \in V_6. \qquad (5.3)$$

Линейным статистическим аналогом каждого из всевозможных уравнений (5.3) может являться уравнение вида:

$$(\overline{Y}, J) = (\overline{X}', I), \qquad (5.4)$$

где $\overline{X}' = \overline{X} \oplus \overline{K}'$; может принимать значения от 1 до 15; I может принимать значения от 1 до 63.

Обозначим через $S_a(i, j)$, ($a = 1, \ldots, 8$; $i = 1, \ldots, 63$; $j = 1, \ldots, 15$), число ненулевых $X \in V_6$ для a-го S-блока DES таких, что для i и j выполняется равенство (5.4). В этом случае значение $S_a(i, j)$ будет определять количество совпадений суммы по модулю 2 некоторых битов входных данных с суммой по модулю 2 некоторых битов выходных данных. Так как сумма по модулю 2 двух одинаковых бит (1 и 1 или 0 и 0) в результате дает 0, то $S_a(i, j)$ и будет по сути дела указывать, сколько раз из возможных 64 комбинаций выполняется формула (5.2) с результатом 0. Так как всего 64 возможных комбинации, то при $S_a(i, j) = 32$ будем иметь $P = (S = 0) = 32/64 = 1/2$, а $P(S = 1) = 1 - 1/2 = 1/2$. Поэтому, когда $S_a \neq 32$ можно сказать, что есть статистическая связь между входными и выходными битами a-го S-блока.

При этом чем больше будет отклонение значения вероятности для каждой пары (i, j), тем более эффективным будет линейный аналог.

Пусть

$$S^*_a(i^*, j^*): |S^*_a(i^*, j^*) - 32| = \max|S_a(i, j) - 32|; 1 \le i \le 63; 1 \le j \le 15. \quad (5.5)$$

Тогда уравнение

$$(X, i^*) \oplus (Y, j^*) = (K, i^*) \quad (5.6)$$

является эффективным линейным статистическим аналогом для a-го S-блока в классе всех линейных статистических аналогов вида (5.4) с вероятностью:

$$P_a = S^*_a(i^*, j^*)/64, a = 1, \ldots, 8.$$

Анализ всех восьми S-блоков показал, что наибольшее отклонение от 1/2 находится в S_5 блоке. Таблицы, полученные в результате анализа, приведены в Приложении 1 настоящей книги. Чтобы все вышесказанное стало более понятным, проанализируем S_5 блок алгоритма DES.

Шестибитовым входам в S_5 блок однозначно соответствуют четырехбитовые выходы. Их значения можно определить с помощью известной таблицы замены, которая представляет собой таблицу из четырех строк и шестнадцати столбцов. По шести входным битам S-блока определяется, под каким номером столбцов и строк следует искать выходное значение (см. таблицу 3.10).

Однако для удобства проводимого анализа целесообразней сопоставить значения входов и выходов не в виде двухмерной матрицы, а в виде таблицы 1, приведенной в Приложении 1.

В ходе анализа мы прослеживаем всевозможные комбинации двоичных векторов i и j. Каждую пару векторов мы используем в качестве маски, которую накладываем на всевозможные пары вход—выход блока замены. Эти маски указывают нам биты входа и выхода соответственно, которые необходимо сложить по модулю два, а затем сравнить полученные результаты. Результат анализа приведен в Приложении 1 в таблице 3. В этой таблице значение $S_5(i, j)$, имеющее наибольшее отклонение от 1/2, помечено * . Следовательно, $S^*_5(I^*, j^*) = 12$, где i = (1,1,1,1), а j = (0,1,0,0,0,0). Это говорит о том, что из всех возможных входных значений (от 0 до 63) и соответствующих им выходных значений 12 раз встречается совпадение второго входного бита и суммы по модулю 2 всех четырех выходных битов. Из таблицы 1, приведенной в Приложении 1, следует, что данными 12 парами являются следующие:

1. 000010 — 1100 ($0 = 1 \oplus 1 \oplus 0 \oplus 0$);
2. 000111 — 1100 ($0 = 1 \oplus 1 \oplus 0 \oplus 0$);
3. 001010 — 1010 ($0 = 1 \oplus 0 \oplus 1 \oplus 0$);
4. 001110 — 0110 ($0 = 0 \oplus 1 \oplus 1 \oplus 0$);
5. 010000 — 1000 ($1 = 1 \oplus 0 \oplus 0 \oplus 0$);

6. 011000 — 1101 ($1 = 1 \oplus 1 \oplus 0 \oplus 1$);
7. 011100 — 1110 ($1 = 1 \oplus 1 \oplus 1 \oplus 0$);
8. 011101 — 1000 ($1 = 1 \oplus 0 \oplus 0 \oplus 0$);
9. 100101 — 1100 ($0 = 1 \oplus 1 \oplus 0 \oplus 0$);
10. 101000 — 1010 ($0 = 1 \oplus 0 \oplus 1 \oplus 0$);
11. 111011 — 0100 ($1 = 0 \oplus 1 \oplus 0 \oplus 0$);
12. 111110 — 1110 ($1 = 1 \oplus 1 \oplus 1 \oplus 0$).

Отсюда эффективным линейным статистическим аналогом S_5 блока является уравнение:

$$X_2 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = K_2, \qquad (5.7)$$

и это уравнение выполняется с вероятностью $p = 3/16$.

Уравнения для эффективных линейных статистических аналогов всех S-блоков приведены в таблице 5.2. Здесь $X = (x_1, \ldots, x_6)$, $Y = (y_1, \ldots, y_4)$, $K = (k_1, \ldots, k_6)$ — входные, выходные и ключевые вектора, соответственно.

Таблица 5.2

№ S-блока	Эффективное линейное уравнение	P	$\Delta = \lvert 1 - 2p \rvert$
1	$X_2 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = K_2$	7/32	9/16
2	$X_1 \oplus X_5 \oplus Y_1 \oplus Y_3 \oplus Y_4 = K_1 \oplus K_5$	1/4	1/2
3	$X_1 \oplus X_5 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = K_1 \oplus K_5$	1/4	1/2
4	$X_1 \oplus X_5 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = K_1 \oplus K_5$ $X_1 \oplus X_3 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = K_1 \oplus K_3$ $X_1 \oplus X_3 \oplus X_5 \oplus X_6 \oplus Y_2 \oplus Y_3 = K_1 \oplus K_3 \oplus K_5 \oplus K_6$ $X_1 \oplus X_3 \oplus X_5 \oplus X_6 \oplus Y_1 \oplus Y_4 = K_1 \oplus K_3 \oplus K_5 \oplus K_6$	1/4 1/4 1/4 1/4	1/2 1/2 1/2 1/2
5	$X_2 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = K_2$	3/16	5/8
6	$X_2 \oplus Y_2 \oplus Y_3 \oplus Y_4 = K_2$ $X_1 \oplus X_5 \oplus Y_1 \oplus Y_3 \oplus Y_4 = K_1 \oplus K_5$	9/32	7/16
7	$X_1 \oplus X_2 \oplus X_3 \oplus X_5 \oplus X_6 \oplus Y_2 = K_1 \oplus K_2 \oplus K_3 \oplus K_5 \oplus K_6$	7/32	9/16
8	$X_2 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = K_2$ $X_1 \oplus X_5 \oplus Y_1 \oplus Y_2 \oplus Y_3 = K_1 \oplus K_5$	1/4	1/2

Так как в рассматриваемом нами S_5 блоке в уравнении участвует второй бит, то легко можно посчитать, что этот бит на самом деле будет 26 битом всего входного сообщения X. Зная, что при входе в F-блок, входной вектор претерпевает изменения с помощью перестановки с расширением, то, воспользовавшись приведенной ниже таблицей 5.3, легко можно определить, что на 26 позиции будет находиться на самом деле 17 входной бит.

Таблица 5.3

DES-перестановка с расширением

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

Точно так же выходной вектор, прежде чем выйти из F-блока претерпевает перестановку с помощью P-блока. Поэтому выходящие из S_5 блока 17, 18, 19 и 20 биты, соответственно, окажутся на 8, 14, 25 и 3 позиции выходного вектора (таблица 5.4).

Таблица 5.4

DES-перестановка с помощью P-блока

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

Тогда полученное нами эффективное линейное статистическое уравнение примет вид:

$$X_{17} \oplus Y_3 \oplus Y_8 \oplus Y_{14} \oplus Y_{25} = K_{26}, \text{ с } \Delta = 5/8. \qquad (5.8)$$

5.2.2. Нахождение статистических аналогов для трех циклов алгоритма DES

Матсуи не описывал алгоритма получения эффективных линейных аналогов для алгоритма DES, состоящего из r циклов, но из конечных результатов можно предположить, что основная идея состоит в следующем: использовать связанные тройные суммы с наибольшим Δ [4].

Прежде чем пояснить вышесказанное на примере, рассмотрим несколько базовых понятий. Чтобы в дальнейшем пользоваться этими важными соотношениями и понятиями приведем их так, как изложено в [4].

Лемма 5.1. Если $S_{(1)}, \ldots, S_{(n)}$ — независимые бинарные случайные величины, тогда

$$\Delta(S_{(1)} \oplus \ldots \oplus S_{(n)}) = \prod \Delta(S_{(i)}), \qquad (5.9)$$

где i =1, …, n.

Покажем, как это будет выглядеть для двух величин. Так как $S_{(1)}$ и $S_{(2)}$ независимы, и при этом $P(S_{(1)} = 1) = p_1$, $P(S_{(2)} = 1) = p_2$, $P(S_{(1)} = 0) = 1 - p_1$ и $P(S_{(2)} = 0) = 1 - p_2$.

Тогда:

$$P(S_{(1)} \oplus S_{(2)} = 1) = P(S_{(1)} = 1)P(S_{(2)} = 0) + P(S_{(1)} = 0)P(S_{(2)} = 1) =$$
$$= p_2(1 - p_1) + p_1(1 - p_2) = p_2 - p_1p_2 + p_1 - p_1p_2 = p_2 + p_1 - 2p_1p_2.$$
$$\Delta(S_{(1)} \oplus S_{(2)}) = 1 - 2p(S_{(1)} \oplus S_{(2)} = 1) = 1 - 2(p_2 + p_1 - 2p_1p_2) =$$
$$= 1 - 2p_2 - 2p_1 + 4p_1p_2 = (1 - 2p_1)(1 - 2p_2) = \Delta(S_{(1)})\,\Delta(S_{(2)}).$$

Тройной суммой $S_{(i)}$ для i-го раунда называют случайную величину вида

$$S_{(i)} = f_i(X_{(i)}) \oplus g_i(Y_{(i)}) \oplus h_i(K_{(i)}), \qquad (5.10)$$

где f_i, g_i, h_i — равновероятностные булевы функции [4].

Последовательные тройные суммы $S_{(1+i)}$ и $S_{(i)}$ называются связанными, если $f_{i+1} = g_i$ [4].

Из этого следует, что

$$S_{(i)} \oplus S_{(1+i)} = f_i(X_{(i)}) \oplus g_i(Y_{(i)}) \oplus h_i(K_{(i)}) \oplus f_{i+1}(X_{(i+1)})) \oplus g_{i+1}(Y_{(i+1)}) \oplus b_{i+1}(K_{(i+1)}),$$

где $Y_{(i)} = X_{(i+1)}$.

То есть, говоря другими словами, в последовательных тройных суммах выход предыдущего раунда является входом следующего.

Если $S_{(1)}, \ldots, S_{(n)}$ — связанные тройные суммы, то тогда

$$S_{1\ldots n} = S_{(1)} \oplus \ldots \oplus S_{(n)} = f_1(X_{(1)}) \oplus g_n(Y_{(n)}) \oplus \sum_{i=1}^{n} h_i(K_{(i)}), \qquad (5.11)$$

и $S_{1\ldots n}$ называется n-раундовой тройной суммой.

Подставив выражение (5.2) в (5.11), получим

$$S_{1\ldots n} = (X_{(1)}, \alpha) \oplus (Y_{(n)}, \beta) \oplus \sum_{i=1}^{n} (K_{(i)}, \gamma_{(i)}), \qquad (5.12)$$

которое выполняется с вероятностью, равной $\Delta(S_{(1)} \oplus \ldots \oplus S_{(n)})$ и вычисляемой по лемме 5.1.

В работе [4] определено, что **эффективным линейным статистическим аналогом** называется линейный статистический аналог (5.12) из заданного множества с наибольшим Δ.

Поясним все вышесказанное на примере. Для этого рассмотрим схему 3-раундового DES (см. рис. 5.1).

Согласно (5.8), эффективными линейными статистическими аналогами первой (1) и третьей (3) итераций являются уравнения:

$$X(1)_{17} \oplus Y(1)_3 \oplus Y(1)_8 \oplus Y(1)_{14} \oplus Y(1)_{25} = K(1)_{26},$$
$$X(3)_{17} \oplus Y(3)_3 \oplus Y(3)_8 \oplus Y(3)_{14} \oplus Y(3)_{25} = K(3)_{26},$$

каждое из которых выполняется с вероятностью 3/16. Тогда Δ_1, Δ_2 для этих уравнений определяются как $|1 - 2p| = |1 - 2*\frac{3}{16}| = |1 - 3/8| = 5/8$.

Учитывая, что $X(1) = P_R$, $X(3) = C_L$, $Y(1) = P_L \oplus X(2)$, $Y(3) = C_R \oplus X(2)$, после сложения этих уравнений получим

$$(P_R \oplus C_L)_{17} \oplus (P_L \oplus C_R)_3 \oplus (P_L \oplus C_R)_8 \oplus (P_L \oplus C_R)_{14} \oplus (P_L \oplus C_R)_{25} =$$
$$= (K(1) \oplus K(3))_{26}. \quad (5.13)$$

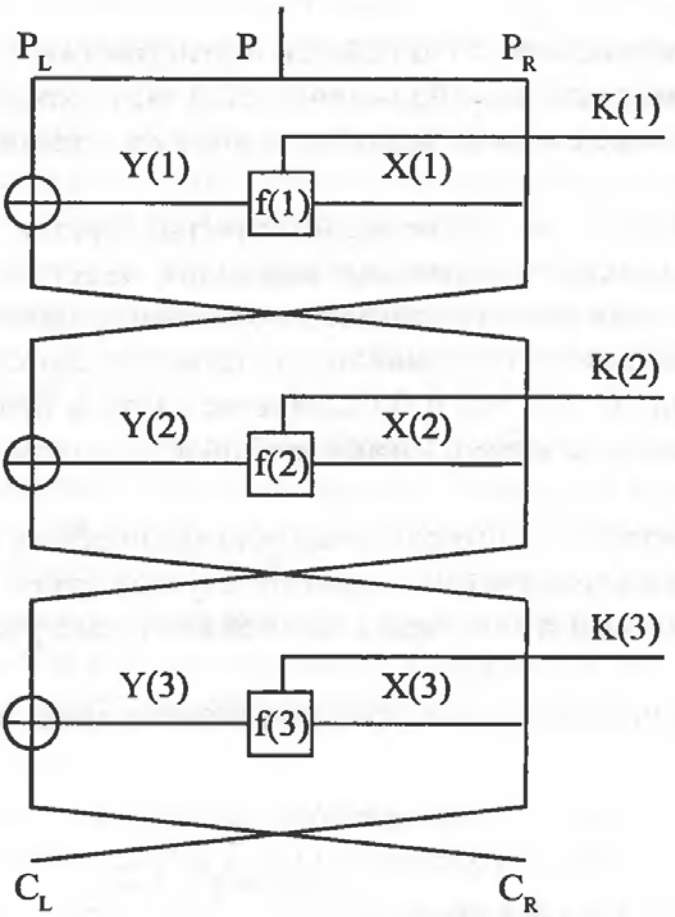


Рис. 5.1. ***Схема трех раундов алгоритма шифрования DES***

Для вычисления общей вероятности мы не можем просто перемножить вероятности двух линейных аналогов, потому что очень маленькая вероятность может давать нам большое отклонение, и наоборот. Поэтому вначале, пользуясь леммой 5.1, мы находим значение отклонения $\Delta(S_{(1)} \oplus S_{(3)})$, а затем из него находим значение вероятности.

Итак, по лемме 5.1 $\Delta(S_{(1)} \oplus S_{(3)}) = \Delta_1 * \Delta_2 = (5/8)*(5/8) = 25/64$, и это уравнение выполняется с вероятностью $p = (1 - \Delta)/2 = 39/128$.

Найти биты ключа $K(1) \oplus K(3)$ можно, решая уравнение (5.13) с использованием следующего алгоритма.

Алгоритм. Пусть N — число всех открытых текстов, и T — число открытых текстов, для которых левая часть линейного статистического аналога равна 0. Рассмотрим два случая.

1. Если $T > N/2$, то в этом случае число открытых текстов, для которых левая часть аналога равна нулю, больше половины, то есть в большинстве случаев в левой части аналога появляется значение, равное нулю, то

а) если вероятность этого линейного статистического аналога $p > 1/2$ — это говорит о том, что в большинстве случаев правая и левая части аналога равны, а значит, левая часть аналога, содержащая биты ключа, равна 0;

б) если вероятность этого линейного статистического аналога $p < 1/2$ — это говорит о том, что в большинстве случаев правая и левая части аналога не равны, а значит, левая часть аналога, содержащая биты ключа, равна 1.

2. Если $T < N/2$, то в этом случае число открытых текстов, для которых левая часть аналога равна нулю, меньше половины, то есть в большинстве случаев в левой части аналога появляется значение, равное единице, то

   а) если вероятность этого линейного статистического аналога $p > 1/2$ — это говорит о том, что в большинстве случаев правая и левая части аналога равны, а значит, левая часть аналога, содержащая биты ключа, равна 1;

   б) если вероятность этого линейного статистического аналога $p < 1/2$ — это говорит о том, что в большинстве случаев правая и левая части аналога не равны, а значит, левая часть аналога, содержащая биты ключа, равна 0.

Пользуясь вышеприведенным алгоритмом, можем определить для уравнения (5.13) следующее

если $T > N/2$, то

$$K(1) \oplus K(3) = \begin{cases} 0, \text{ если } p > 1/2; \\ 1, \text{ если } p < 1/2; \end{cases}$$

если $T < N/2$, то

$$K(1) \oplus K(3) = \begin{cases} 1, \text{ если } p > 1/2; \\ 0, \text{ если } p < 1/2. \end{cases}$$

Успех алгоритма возрастает с ростом N и $\Delta = |1 - 2p|$.

5.2.3. Нахождение статистических аналогов для 5, 7, 8, 12, 14 и 16 циклов алгоритма DES

Теперь, когда нам известны основные принципы построения линейных статистических аналогов для одного и трех раундов алгоритмов шифрования, можно перейти к решению более сложных задач. Это необходимо сделать для того, чтобы наглядно продемонстрировать применение такого мощного метода криптоанализа, как линейный к одному из серьезнейших алгоритмов шифрования. Как извес-

тно, алгоритм шифрования DES состоит из 16 раундов. Чтобы получить итоговый линейный статистический аналог для полного алгоритма шифрования DES, мы будем находить линейные аналоги для 5, 7, 8, 12, и 14 циклов. Это необходимо, чтобы можно было как можно лучше понять принцип нахождения линейных статистических аналогов для алгоритмов шифрования, имеющих различное число раундов. При нахождении линейного статистического аналога для трех раундов алгоритма шифрования DES мы использовали полученные характеристики для блока замены S_5. При рассмотрении алгоритма, содержащего более трех раундов шифрования, мы скорее всего не сможем ограничиться использованием только самого наилучшего приближения. Понадобятся дополнительные приближения, которые также могут использоваться с наилучшим приближением (с ранее определенным уравнением (5.8)).

Естественно, возможны многочисленные комбинации различных уравнений, имеющие различные значения вероятностей. Однако мы будем рассматривать ту комбинацию уравнений, которая была предложена основоположником линейного криптоанализа М. Матсуи [6, 17]. По его утверждению, использование этих уравнений приводит к нахождению наиболее эффективных статистических аналогов для любого количества раундов алгоритма шифрования DES. Для удобства дальнейшей работы эти уравнения сведены в таблицу 5.5.

Таблица 5.5

Комбинация уравнений для нахождения эффективных статистических аналогов для алгоритма шифрования DES

№	Уравнение	№ S-блока	Единичные векторы i, j	p	Δ
1	X[17] ⊕ Y[3,8,14,25] = K[26]	5	01000, 1111	12/64	5/8
2	X[1,2,4,5] ⊕ Y[17] = K[2,3,5,6]	1	011011, 0100	22/64	20/64
3	X[3] ⊕ Y[17] = K[4]	1	000100, 0100	30/64	4/64
4	X[17] ⊕ Y[8,14,25] = K[26]	5	010000, 1110	42/64	20/64
5	X[16,20] ⊕ Y[8,14,25] = K[25,29]	5	100010, 1110	16/64	1/2

Как видно из таблицы 5.5, будут использованы значения блоков замены S_5 и S_1. В Приложении 1 приведены результаты анализа этих блоков. Далее станет ясна необходимость именно такого выбора уравнений.

Следует сразу оговориться, что в работах М. Матсуи [6, 17] все данные изложены достаточно лаконично и не содержат развернутых объяснений. При этом еще и биты шифруемых данных пронумерованы не так, как мы привыкли, то есть не слева направо, а наоборот. Поэтому авторы считают необходимым представить развернутые объяснения по получению каждого из линейных аналогов, так как данное пособие призвано научить основам криптоанализа, а это не возможно без тщательного изучения азов.

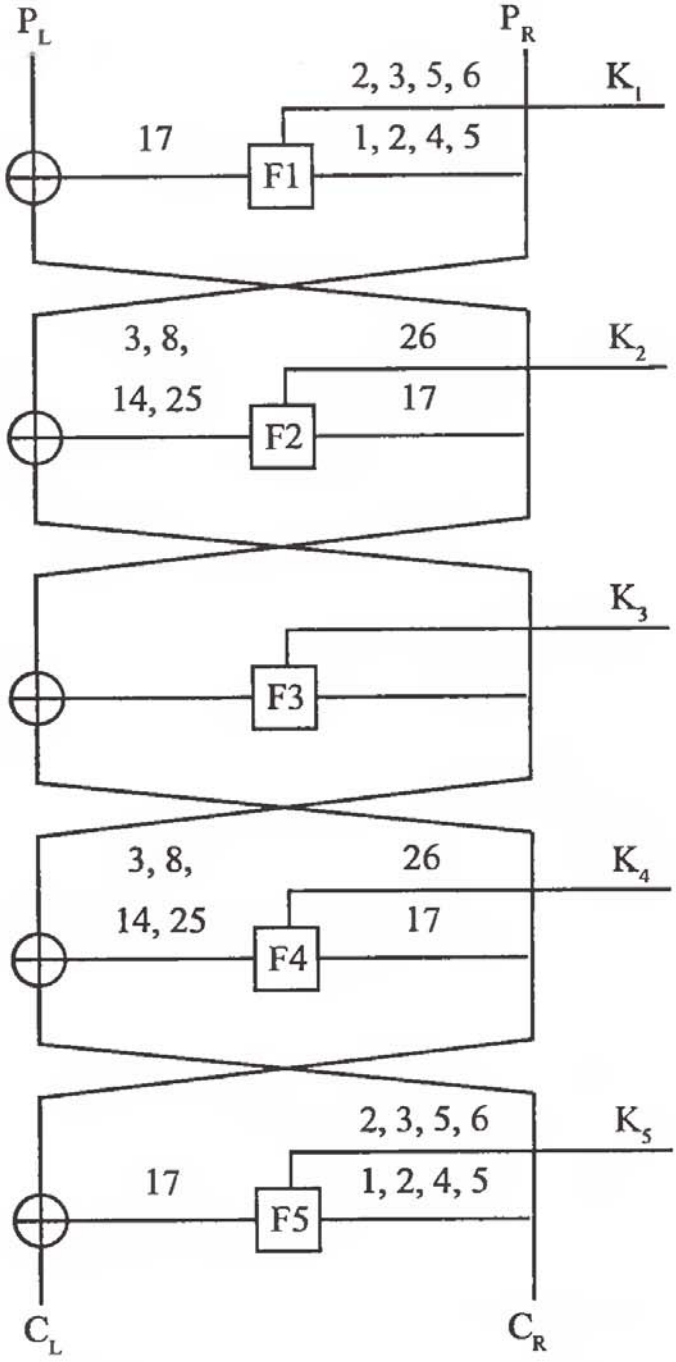


Рис. 5.2. ***Схема пяти раундов алгоритма шифрования DES***

На рис. 5.2 представлена схема пяти раундов алгоритма шифрования DES с указанием тех битов входных и выходных сообщений, которые будут участвовать в нахождении эффективного статистического аналога. Как видно из рис. 5.2, при нахождении эффективного статистического аналога будут задействованы 1, 2, 4 и 5 раунды алгоритма шифрования. При этом для составления общего линейного аналога мы будем использовать уравнение 2 из таблицы 5.5 для 1 и 5 раундов, уравнение 1 из таблицы 5.5 для 2 и 4 раундов. Если обозначить вход F-функции в i-м раунде как X_i, а выход F-функции в i-м раунде как Y_i, то, сложив между собой

в левой части входы и выходы первого, второго, четвертого и пятого раундов, а в правой части — ключи этих раундов, получим следующее уравнение:

$$X_1 \oplus Y_1 \oplus X_2 \oplus Y_2 \oplus X_4 \oplus Y_4 \oplus X_5 \oplus Y_5 = K_1 \oplus K_2 \oplus K_4 \oplus K_5.$$

Согласно рис. 5.2, для составления общего линейного аналога мы будем использовать уравнение 2 из таблицы 5.5 для первого и пятого раундов и уравнение 1 из таблицы 5.5 для второго и четвертого раундов.

Так как все выходы F-функции можно выразить через соответствующие входы F-функций следующим образом:

$$Y_1 = P_L \oplus X_2;$$
$$Y_2 = P_R \oplus X_3;$$
$$Y_4 = X_3 \oplus C_R;$$
$$Y_5 = X_4 \oplus C_L,$$

то, с учетом используемых битов, обозначенных на рис. 5.2, получим:

$$P_L[17] \oplus P_R[1,2,4,5] \oplus X_2[17] \oplus P_R[3,8,14,25] \oplus X_3[3,8,14,25] \oplus X_2[17] \oplus X_3[3,8,14,25] \oplus C_R[3,8,14,25] \oplus X_4[17] \oplus X_4[17] \oplus C_L[17] \oplus C_R[1,2,4,5] = K_1[2,3,5,6] \oplus K_2[26] \oplus K_4[26] \oplus K_5[2,3,5,6].$$

Так как сумма по модулю два двух одинаковых битов равна 0, то, произведя сокращение одинаковых слагаемых, получим:

$$P_L[17] \oplus P_R[1,2,4,5] \oplus P_R[3,8,14,25] \oplus C_R[3,8,14,25] \oplus C_L[17] \oplus C_R[1,2,4,5] = K_1[2,3,5,6] \oplus K_2[26] \oplus K_4[26] \oplus K_5[2,3,5,6]. \quad (5.14)$$

Для вычисления общей вероятности мы не можем просто перемножить вероятности двух линейных аналогов, потому что очень маленькая вероятность может давать нам большое отклонение, и наоборот. Поэтому вначале, пользуясь леммой 5.1, мы находим значение отклонения $\Delta(S_{(1)} \oplus S_{(2)} \oplus S_{(4)} \oplus S_{(5)})$, а затем из него находим значение вероятности.

Уравнение (5.14) является эффективным линейным аналогом для алгоритма шифрования DES, состоящего из 5-ти раундов, и по лемме 5.1 $\Delta(S_{(1)} \oplus S_{(2)} \oplus S_{(4)} \oplus \oplus S_{(5)}) = \Delta_1 * \Delta_2 * \Delta_4 * \Delta_5 = (10/32) * (5/8) * (5/8) * (10/32) = 0{,}03815$, а значит, это уравнение выполняется с вероятностью $p = (1 - \Delta)/2 = 0{,}480925$.

Мы знаем, что отклонение определяет эффективность линейного статистического аналога. Так, чем больше отклонение для найденного аналога, тем меньше текстов надо проанализировать для правильного определения ключа, а значит, тем выше будет скорость проведения анализа.

Как видно из рис. 5.2, при нахождении линейного статистического аналога 2, 2, 3 и 4 раунды такие же, как при нахождении уравнения для трех раундов алгоритма DES, а 1 и 5 раунд используют такие приближения, чтобы на выходе их функций появлялся тот же самый бит, который учитывается во входах F-функций 2 и 4 раундов.

После того как линейный статистический аналог найден, можно приступить к нахождению битов ключа. Для этого необходимо воспользоваться алгоритмом, приведенном в пункте 5.2.2.

Теперь усложним задачу и рассмотрим алгоритм шифрования DES, содержащий 7 раундов. Схема этого алгоритма приведена на рис. 5.3.

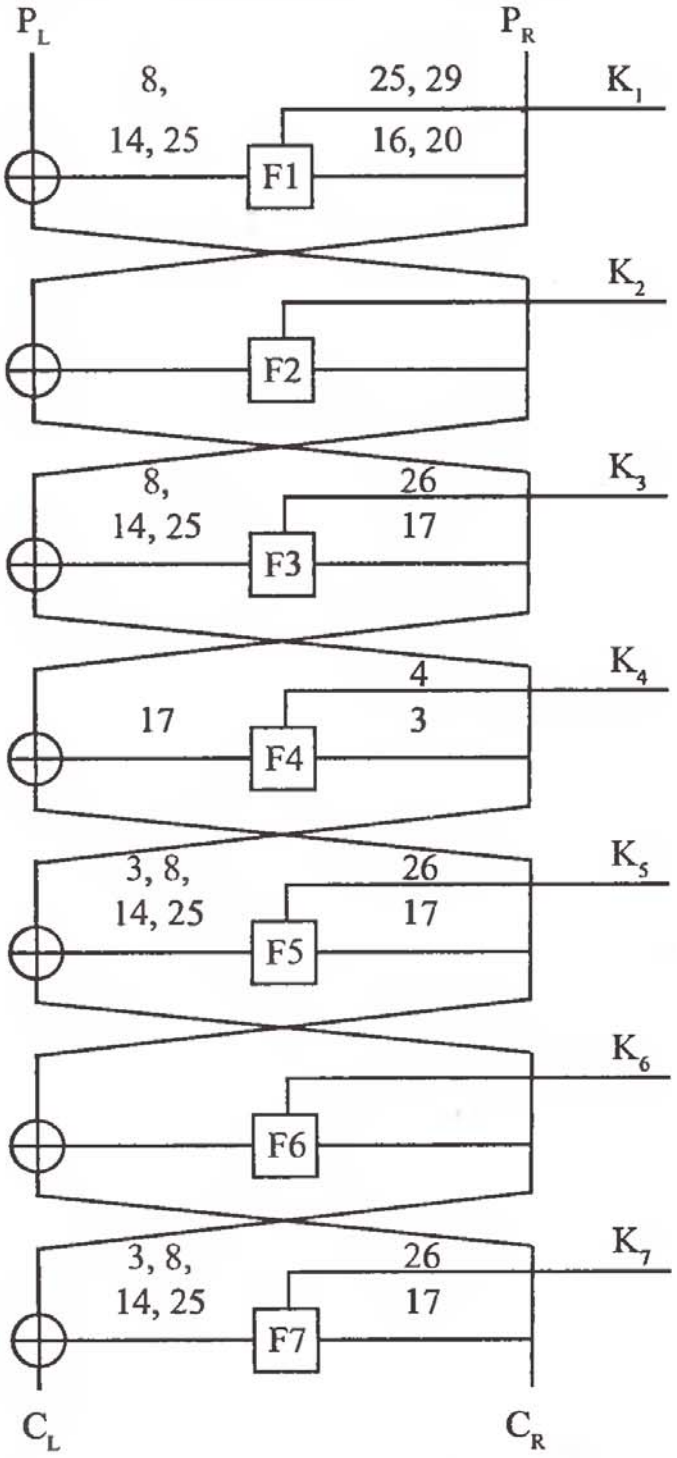


Рис. 5.3. ***Схема 7-раундового алгоритма шифрования DES***

Как видно из рис. 5.3, при нахождении эффективного статистического аналога будут задействованы 1, 3, 4, 5 и 7 раунды алгоритма шифрования. При этом для составления общего линейного аналога мы будем использовать уравнение 1 из таблицы 5.5 для 1, 5 и 7 раундов, уравнение 3 из таблицы 5.5 для 4 раунда и уравнение 5 из таблицы 5.5 для 1 раунда.

Аналогично предыдущему разу обозначим вход F-функции в i -м раунде как x_i, а выход F-функции в i -м раунде как y_i, и получим следующее уравнение:

$$X_1 \oplus Y_1 \oplus X_3 \oplus Y_3 \oplus X_4 \oplus Y_4 \oplus X_5 \oplus Y_5 \oplus X_7 \oplus Y_7 = K_1 \oplus K_3 \oplus K_4 \oplus K_5 \oplus K_7.$$

Так как все выходы F-функции можно выразить через соответствующие входы F-функций следующим образом:

$$Y_1 = P_L \oplus X_2;$$
$$Y_3 = X_2 \oplus X_4;$$
$$Y_4 = X_3 \oplus X_5;$$
$$Y_5 = X_4 \oplus X_6;$$
$$Y_7 = X_6 \oplus C_L,$$

то, с учетом используемых битов, обозначенных на рис. 5.3, получим:

$$P_L[8,14,25] \oplus P_R[16,20] \oplus X_2[8,14,25] \oplus X_3[17] \oplus X_2[8,14,25] \oplus X_4[3,8,14,25] \oplus$$
$$\oplus X_4[3] \oplus X_3[17] \oplus X_5[17] \oplus X_5[17] \oplus X_4[3,8,14,25] \oplus X_6[3,8,14,25] \oplus C_R[17] \oplus$$
$$\oplus X_6[3,8,14,25] \oplus C_L[3,8,14,25] = K_1[22,29] \oplus K_3[26] \oplus K_4[4] \oplus K_5[26] \oplus K_7[26].$$

Так как сумма по модулю два двух одинаковых битов равна 0, то, произведя сокращение одинаковых слагаемых, получим уравнение (5.15). При этом обратите внимание, что в разных раундах используется разное количество битов X_4, но, так как один и тот же бит X_4 каждый раз используется дважды, то в конечном уравнении X_4 отсутствует:

$$P_L[8,14,25] \oplus P_R[16,20] \oplus C_R[17] \oplus C_L[3,8,14,25] =$$
$$= K_1[22,29] \oplus K_3[26] \oplus K_4[4] \oplus K_5[26] \oplus K_7[26]. \quad (5.15)$$

Для вычисления общей вероятности мы не можем просто перемножить вероятности двух линейных аналогов, потому что очень маленькая вероятность может давать нам большое отклонение, и наоборот. Поэтому вначале, пользуясь леммой 5.1, мы находим значение отклонения $\Delta(S_{(1)} \oplus S_{(3)} \oplus S_{(4)} \oplus S_{(5)} \oplus S_{(7)})$, а затем из него находим значение вероятности.

Уравнение (5.15) является эффективным линейным аналогом для алгоритма шифрования DES, состоящего из 7-ми раундов, и по лемме 5.1 $\Delta(S_{(1)} \oplus S_{(3)} \oplus \oplus S_{(4)} \oplus S_{(5)} \oplus S_{(7)}) = \Delta_1 * \Delta_3 * \Delta_4 * \Delta_5 * \Delta_7 = (1/2)*(20/64)*(4/64)*(5/8)*(5/8) = 0{,}00381$, а значит, это уравнение выполняется с вероятностью $p = (1 - \Delta)/2 = 0{,}498095$.

Как видно из рис. 5.3, при нахождении линейного статистического аналога используемые входные и выходные биты функций F Матсуи подобрал таким образом, что при упрощении уравнения остаются биты известных нам открытого сообщения P и зашифрованного сообщения C.

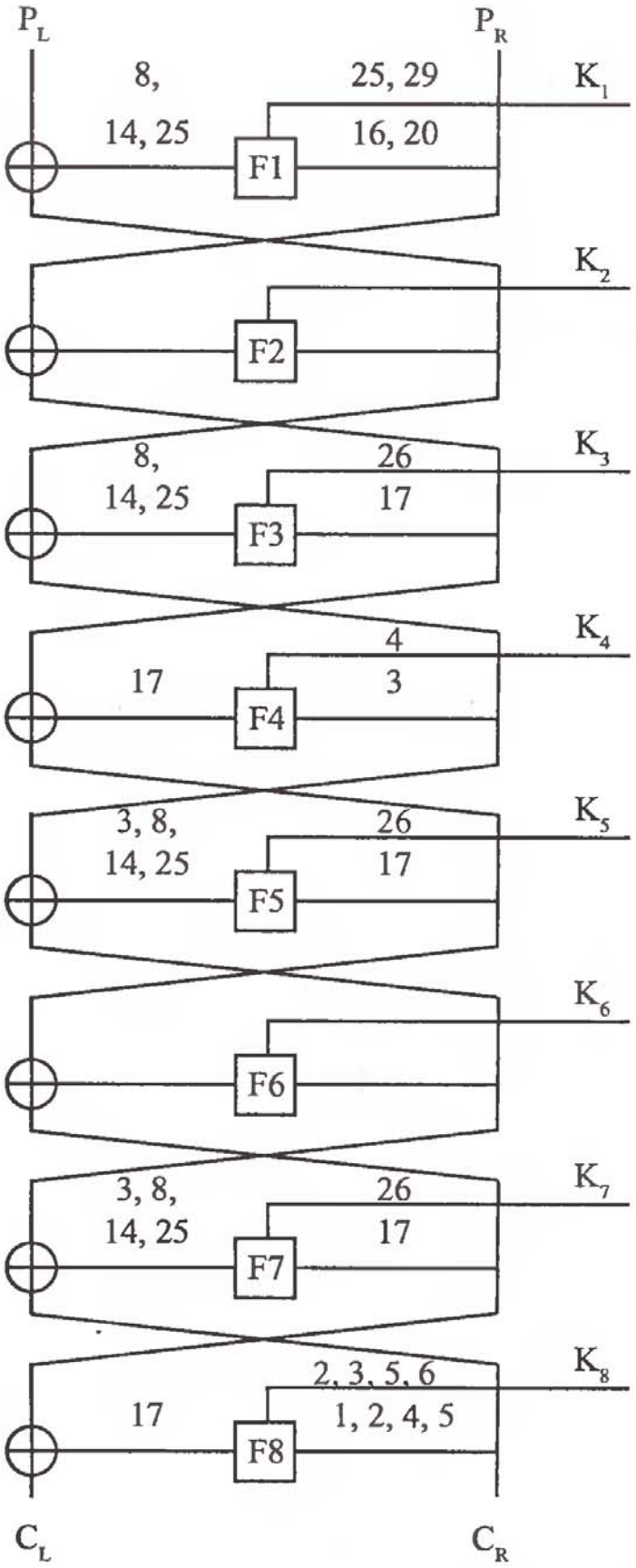


Рис. 5.4. ***Схема 8-раундового алгоритма шифрования DES***

Вывод эффективного уравнения для 8-ми раундового алгоритма шифрования будет отличаться от вывода предыдущего уравнения лишь тем, что при этом будут учитываться биты восьмого раунда алгоритма. При этом будет использовано уравнение 2 из табл. 5.5. На рис. 5.4 приведена структура 8-раундового алгоритма DES. Ниже представлен вывод эффективного линейного аналога.

$$P_L[8,14,25] \oplus P_R[16,20] \oplus X_2[8,14,25] \oplus X_3[17] \oplus X_2[8,14,25] \oplus X_4[3,8,14,25] \oplus$$
$$\oplus X_4[3] \oplus X_3[17] \oplus X_5[17] \oplus X_5[17] \oplus X_4[3,8,14,25] \oplus X_6[3,8,14,25] \oplus X_7[17] \oplus$$
$$\oplus X_6[3,8,14,25] \oplus C_R[3,8,14,25] \oplus C_R[1,2,4,5] \oplus X_7[17] \oplus C_L[17] =$$
$$= K_1[22,29] \oplus K_3[26] \oplus K_4[4] \oplus K_5[26] \oplus K_7[26] \oplus K_8[2,3,5,6].$$

После сокращения одинаковых слагаемых получим:

$$P_L[8,14,25] \oplus P_R[16,20] \oplus C_R[3,8,14,25] \oplus C_R[1,2,4,5] \oplus C_L[17] =$$
$$= K_1[22,29] \oplus K_3[26] \oplus K_4[4] \oplus K_5[26] \oplus K_7[26] \oplus K_8[2,3,5,6]. \quad (5.16)$$

Для вычисления общей вероятности мы не можем просто перемножить вероятности двух линейных аналогов, потому что очень маленькая вероятность может давать нам большое отклонение, и наоборот. Поэтому вначале, пользуясь леммой 5.1, мы находим значение отклонения $\Delta(S_{(1)} \oplus S_{(3)} \oplus S_{(4)} \oplus S_{(5)} \oplus S_{(7)} \oplus S_{(8)})$, а затем из него находим значение вероятности.

Уравнение (5.16) является эффективным линейным аналогом для алгоритма шифрования DES, состоящего из 8-ми раундов, и по лемме 5.1 $\Delta(S_{(1)} \oplus \oplus S_{(3)} \oplus S_{(4)} \oplus S_{(5)} \oplus S_{(7)} \oplus S_{(8)}) = \Delta_1 * \Delta_3 * \Delta_4 * \Delta_5 * \Delta_7 * \Delta_8 = (1/2) * (20/64) * (4/64) * * (5/8) * (5/8) * (10/32) = 0{,}001192$, а значит, это уравнение выполняется с вероятностью $p = (1 - \Delta)/2 = 0{,}499404$.

За счет того, что при выводе уравнения (5.16) используются два последних цикла алгоритма шифрования, в уравнении участвуют биты выходного сообщения, используемые в обоих этих раундах.

При рассмотрении 12-раундового алгоритма шифрования DES, приведенного на рис. 5.5, будем использовать первый и два последних раунда шифрования, а так же две пары по три раунда: третий, четвертый и пятый раунды и седьмой, восьмой и девятый раунды. Согласно рис. 5.5, для составления общего линейного аналога мы будем использовать уравнение 1 из таблицы 5.5 для 1, 3, 9 и 11 раундов, уравнение 2 из таблицы 5.5 для 12 раунда, уравнение 3 из таблицы 5.5 для 4 и 8 раундов и уравнение 4 из таблицы 5.5 для 5 и 7 раундов.

Эффективное линейное уравнение будет иметь следующий вид:

$$P_R[17] \oplus P_L[3,8,14,25] \oplus X_2[3,8,14,25] \oplus X_3[17] \oplus X_2[3,8,14,25] \oplus X_4[3,8,14,25] \oplus$$
$$\oplus X_4[3] \oplus X_3[17] \oplus X_5[17] \oplus X_5[17] \oplus X_4[8,14,25] \oplus X_6[8,14,25] \oplus X_7[17] \oplus$$
$$\oplus X_8[8,14,25] \oplus X_6[8,14,25] \oplus X_8[3] \oplus X_7[17] \oplus X_9[17] \oplus X_9[17] \oplus X_8[3,8,14,25] \oplus$$
$$\oplus X_{10}[3,8,14,25] \oplus X_{11}[17] \oplus X_{10}[3,8,14,25] \oplus C_R[3,8,14,25] \oplus C_R[1,2,4,5] \oplus$$
$$\oplus X_{11}[17] \oplus C_L[17] = K_1[26] \oplus K_3[26] \oplus K_4[4] \oplus K_5[26] \oplus K_7[26] \oplus K_8[4] \oplus$$
$$\oplus K_9[26] \oplus K_{11}[26] \oplus K_{12}[2,3,5,6].$$

После сокращения одинаковых слагаемых, получим:

$$P_R[17] \oplus P_L[3,8,14,25] \oplus C_R[3,8,14,25] \oplus C_R[1,2,4,5] \oplus C_L[17] = K_1[26] \oplus$$
$$\oplus K_3[26] \oplus K_4[4] \oplus K_5[26] \oplus K_7[26] \oplus K_8[4] \oplus K_9[26] \oplus K_{11}[26] \oplus$$
$$\oplus K_{12}[2,3,5,6]. \qquad (5.17)$$

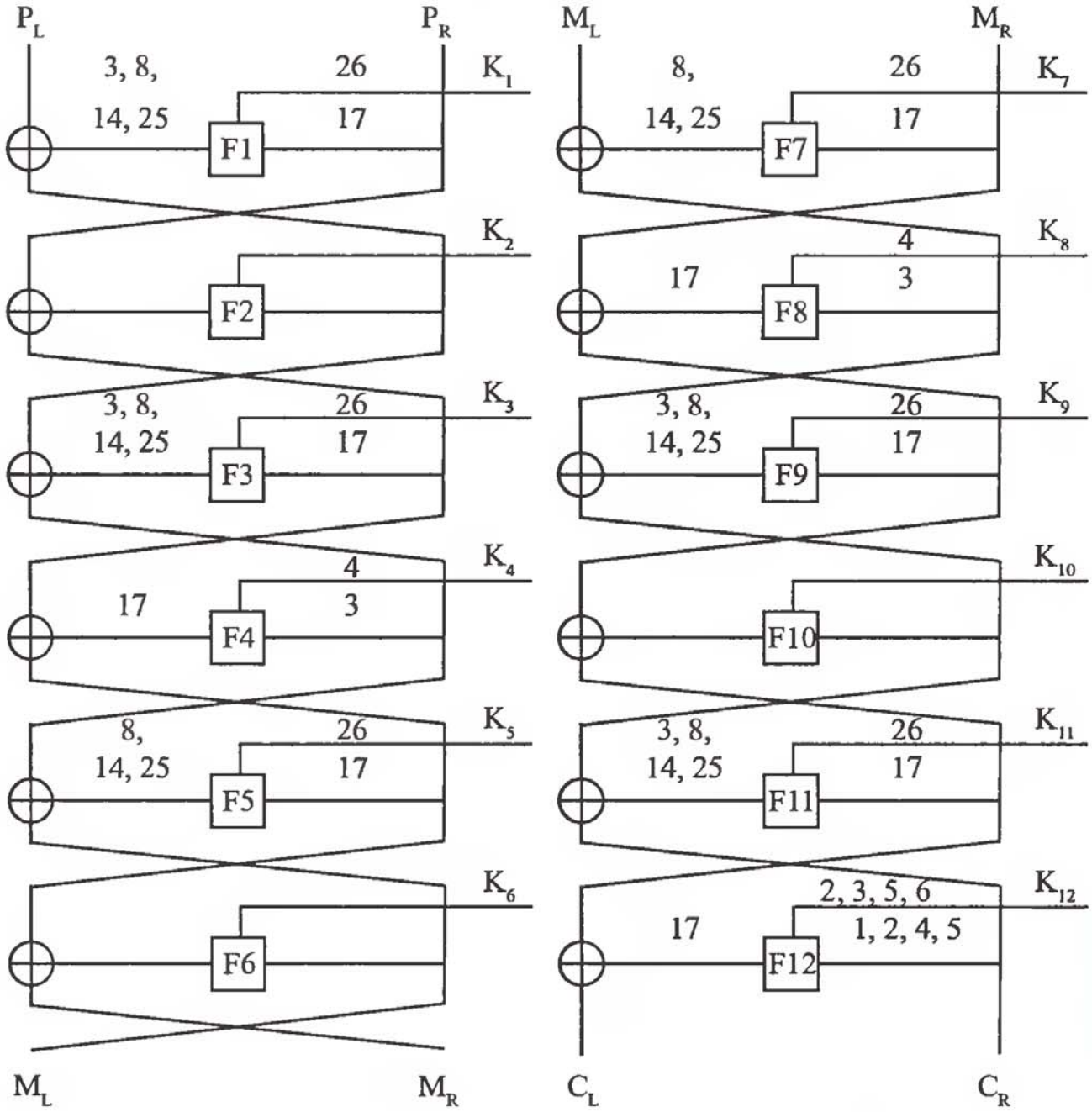


Рис. 5.5. ***Схема 12-раундового алгоритма шифрования DES***

Для вычисления общей вероятности мы не можем просто перемножить вероятности двух линейных аналогов, потому что очень маленькая вероятность может давать нам большое отклонение, и наоборот. Поэтому вначале, пользуясь леммой 5.1, мы находим значение отклонения $\Delta(S_{(1)} \oplus S_{(3)} \oplus S_{(4)} \oplus S_{(5)} \oplus S_{(7)} \oplus \oplus S_{(8)} \oplus S_{(9)} \oplus S_{(11)} \oplus S_{(12)})$, а затем из него находим значение вероятности.

Уравнение (5.17) является эффективным линейным аналогом для алгоритма шифрования DES, состоящего из 12-ти раундов, и по лемме 5.1 $\Delta(S_{(1)} \oplus S_{(3)} \oplus S_{(4)} \oplus \oplus S_{(5)} \oplus S_{(7)} \oplus S_{(8)} \oplus S_{(9)} \oplus S_{(11)} \oplus S_{(12)}) = \Delta_1 * \Delta_3 * \Delta_4 * \Delta_5 * \Delta_7 * \Delta_8 * \Delta_9 * \Delta_{11} * \Delta_{12} = (5/8)4*(4/64)2*(20/64)2*(10/32) = 0{,}00001818$, а значит, это уравнение выполняется с вероятностью $p = (1 - \Delta)/2 = 0{,}49999091$.

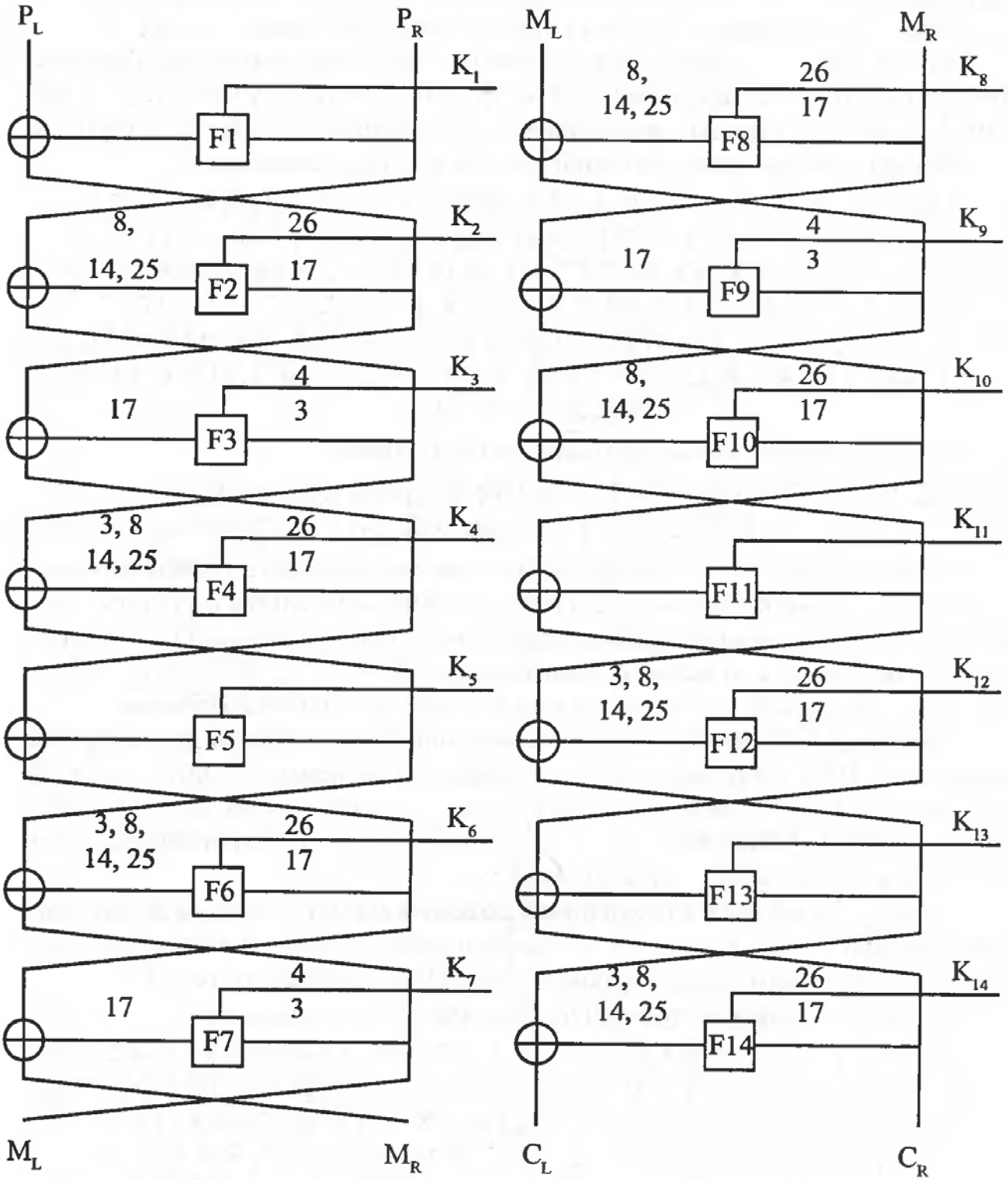


Рис. 5.6. ***Схема 14-раундового алгоритма шифрования DES***

Аналогично уравнению (5.16) в уравнении (5.17) участвуют биты выходного сообщения, используемые в двух последних раундах.

В случае с 14-раундовым алгоритмом DES мы не будем использовать первый и предпоследний раунды алгоритма шифрования. В этом случае в итоговом уравнении не будут присутствовать биты правой части исходного сообщения.

Схема 14-раундового алгоритма шифрования DES приведена на рис. 5.6.

Согласно рис. 5.6, для составления общего линейного аналога мы будем использовать уравнение 1 из таблицы 5.5 для 6, 12 и 14 раундов, уравнение 3 из таблицы 5.5 для 3, 7 и 9 раундов и уравнение 4 из таблицы 5.5 для 2, 8 и 10 раундов.

Эффективное линейное уравнение будет иметь следующий вид:

$$P_R[8,14,25] \oplus X_2[17] \oplus X_3[8,14,25] \oplus X_3[3] \oplus X_2[17] \oplus X_4[17] \oplus X_4[17] \oplus$$
$$\oplus X_3[3,8,14,25] \oplus X_5[3,8,14,25] \oplus X_6[17] \oplus X_5[3,8,14,25] \oplus X_7[3,8,14,25] \oplus$$
$$\oplus X_7[3] \oplus X_6[17] \oplus X_8[17] \oplus X_8[17] \oplus X_7[8,14,25] \oplus X_9[8,14,25] \oplus X_{10}[17] \oplus$$
$$\oplus X_9[8,14,25] \oplus X_{11}[8,14,25] \oplus X_{11}[3] \oplus X_{10}[17] \oplus X_{12}[17] \oplus X_{12}[17] \oplus$$
$$\oplus X_{11}[3,8,14,25] \oplus X_{13}[3,8,14,25] \oplus C_R[17] \oplus X_{13}[3,8,14,25] \oplus CL[3,8,14,25] =$$
$$= K_2[26] \oplus K_3[4] \oplus K_4[26] \oplus K_6[26] \oplus K_7[4] \oplus K_8[26] \oplus K_{10}[26] \oplus K_{11}[4] \oplus$$
$$\oplus K_{12}[26] \oplus K_{14}[26].$$

После сокращения одинаковых слагаемых, получим:

$$P_R[8,14,25] \oplus C_R[17] \oplus C_L[3,8,14,25] = K_2[26] \oplus K_3[4] \oplus K_4[26] \oplus K_6[26] \oplus$$
$$\oplus K_7[4] \oplus K_8[26] \oplus K_{10}[26] \oplus K_{11}[4] \oplus K_{12}[26] \oplus K_{14}[26]. \quad (5.18)$$

Для вычисления общей вероятности мы не можем просто перемножить вероятности двух линейных аналогов, потому что очень маленькая вероятность может давать нам большое отклонение, и наоборот. Поэтому вначале, пользуясь леммой 5.1, мы находим значение отклонения $\Delta(S_{(2)} \oplus S_{(3)} \oplus S_{(4)} \oplus S_{(6)} \oplus S_{(7)} \oplus S_{(8)} \oplus \oplus S_{(9)} \oplus S_{(10)} \oplus S_{(12)} \oplus S_{(14)})$, а затем из него находим значение вероятности.

Уравнение (5.18) является эффективным линейным аналогом для алгоритма шифрования DES, состоящего из 14-ти раундов, и по лемме 5.1 $\Delta(S\ S_{(2)} \oplus S_{(3)} \oplus \oplus S_{(4)} \oplus S_{(6)} \oplus S_{(7)} \oplus S_{(8)} \oplus S_{(9)} \oplus S_{(10)} \oplus S_{(12)} \oplus S_{(14)})) = \Delta_2*\Delta_3*\Delta_4*\Delta_6*\Delta_7*\Delta_8*\Delta_9*\Delta_{10}* *\Delta_{12}*\Delta_{14} = (20/64)_3*(4/64)_3*(5/8)_4 = 1{,}1368*10^{-6}$, а значит, это уравнение выполняется с вероятностью $p = (1 - \Delta)/2 = 0{,}4999994316$.

И, наконец, в случае с 16-раундовым алгоритмом DES, мы будем действовать также, как при выводе уравнения для 12-раундового алгоритма DES. Схема полного 16-ти раундового алгоритма шифрования DES приведена на рис. 5.7.

Эффективное линейное уравнение будет иметь следующий вид:

$$P_R[16,20] \oplus P_L[8,14,25] \oplus X_2[8,14,25] \oplus X_3[17] \oplus X_2[8,14,25] \oplus X_4[8,14,25] \oplus$$
$$\oplus X_4[3] \oplus X_3[17] \oplus X_5[17] \oplus X_5[17] \oplus X_4[3,8,14,25] \oplus X_6[3,8,14,25] \oplus X_7[17] \oplus$$
$$\oplus X_6[3,8,14,25] \oplus X_8[3,8,14,25] \oplus X_8[3] \oplus X_7[17] \oplus X_9[17] \oplus X_9[17] \oplus$$
$$\oplus X_8[8,14,25] \oplus X_{10}[8,14,25] \oplus X_{11}[17] \oplus X_7[17] \oplus X_9[17] \oplus X_9[17] \oplus$$
$$\oplus X_8[8,14,25] \oplus X_{10}[8,14,25] \oplus X_{11}[17] \oplus X_{10}[8,14,25] \oplus X_{12}[8,14,25] \oplus X_{12}[3] \oplus$$
$$\oplus X_{11}[17] \oplus X_{13}[17] \oplus X_{13}[17] \oplus X_{12}[3,8,14,25] \oplus X_{14}[3,8,14,25] \oplus X_{15}[17] \oplus$$

$$\oplus X_{14}[3,8,14,25] \oplus C_R[3,8,14,25] \oplus C_R[1,2,4,5] \oplus X_{15}[17] \oplus C_L[17] =$$
$$= K_1[29,25] \oplus K_3[26] \oplus K_4[4] \oplus K_5[26] \oplus K_7[26] \oplus K_8[4] \oplus K_9[26] \oplus K_{11}[26] \oplus$$
$$\oplus K_{12}[4] \oplus K_{13}[26] \oplus K_{15}[26] \oplus K_{16}[2,3,5,6].$$

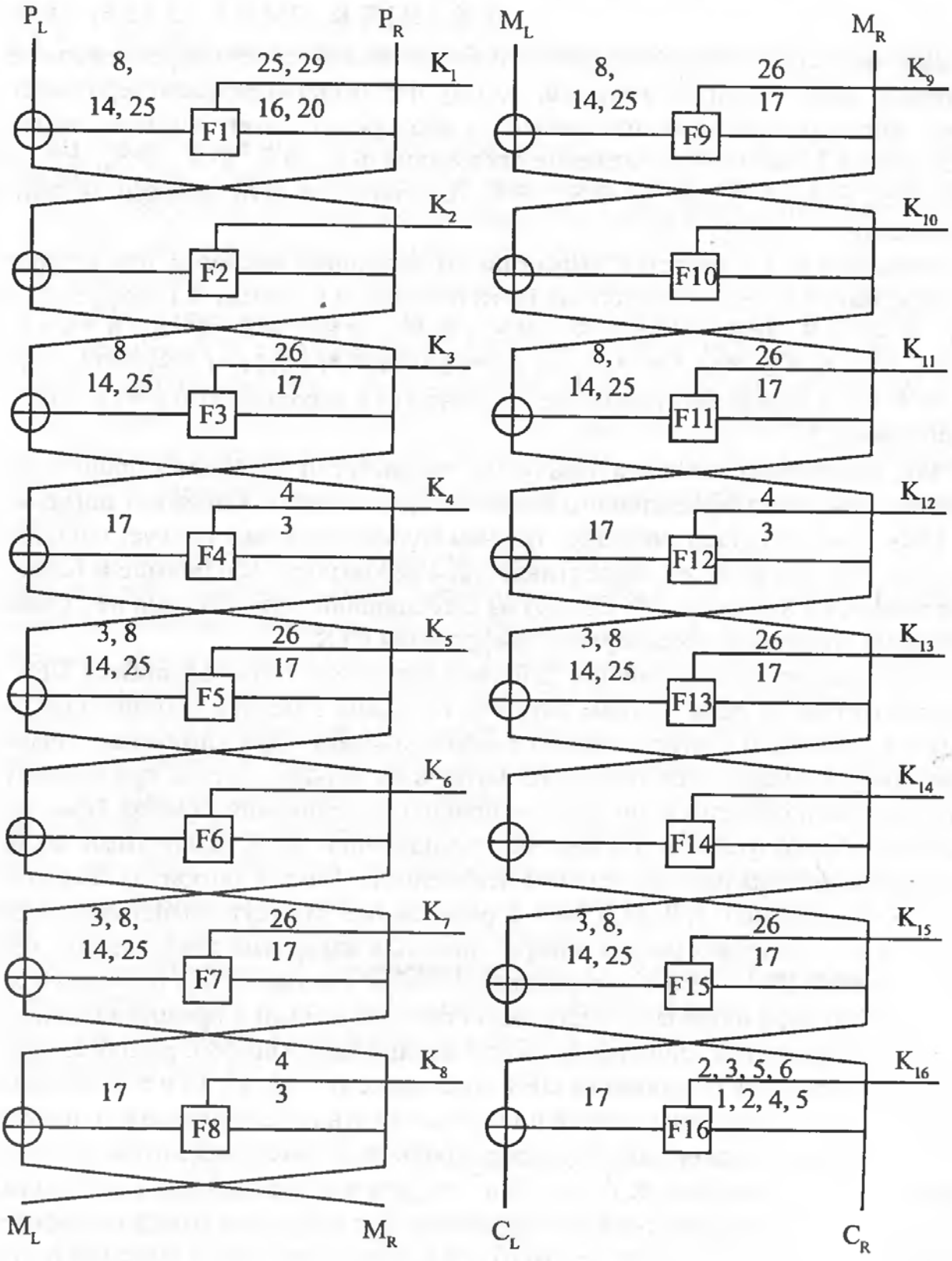


Рис. 5.7. ***Схема 16-раундового алгоритма шифрования DES***

После сокращения одинаковых слагаемых, получим:

$$P_R[16,20] \oplus P_L[8,14,25] \oplus C_R[3,8,14,25] \oplus C_R[1,2,4,5] \oplus C_L[17] = K_1[29,25] \oplus$$
$$\oplus K_3[26] \oplus K_4[4] \oplus K_5[26] \oplus K_7[26] \oplus K_8[4] \oplus K_9[26] \oplus K_{11}[26] \oplus K_{12}[4] \oplus$$
$$\oplus K_{13}[26] \oplus K_{15}[26] \oplus K_{16}[2,3,5,6]. \quad (5.19)$$

Для вычисления общей вероятности мы не можем просто перемножить вероятности двух линейных аналогов, потому что очень маленькая вероятность может давать нам большое отклонение, и наоборот. Поэтому вначале, пользуясь леммой 5.1, мы находим значение отклонения $\Delta(S_{(1)} \oplus S_{(3)} \oplus S_{(4)} \oplus S_{(5)} \oplus S_{(7)} \oplus \oplus S_{(8)} \oplus S_{(9)} \oplus S_{(11)} \oplus S_{(12)} \oplus S_{(13)} \oplus S_{(15)} \oplus S_{(16)})$, а затем из него находим значение вероятности.

Уравнение (5.19) является эффективным линейным аналогом для алгоритма шифрования DES, состоящего из 16-ти раундов, и по лемме 5.1 $\Delta(S_{(1)} \oplus S_{(3)} \oplus \oplus S_{(4)} \oplus S_{(5)} \oplus S_{(7)} \oplus S_{(8)} \oplus S_{(9)} \oplus S_{(11)} \oplus S_{(12)} \oplus S_{(13)} \oplus S_{(15)} \oplus S_{(16)}) = \Delta_1 * \Delta_3 * \Delta_4 * \Delta_5 * * \Delta_7 * \Delta_8 * \Delta_9 * \Delta_{11} * \Delta_{12} * \Delta_{13} * \Delta_{15} * \Delta_{16} = (1/2 * (20/64)^3) * (4/64)_3 * (5/8)_4 * (10/32) = = 1{,}776 * 10^{-7}$, а значит, это уравнение выполняется с вероятностью $p = (1 - \Delta)/2 = = 0{,}4999999112$.

Так, постепенно усложняя задачу, мы рассмотрели детальный процесс построения линейного эффективного уравнения для полного алгоритма шифрования DES. При этом стало очевидно, что, чем больше раундов участвует в построении аналога, тем ниже его эффективность, а вероятность все больше и больше приближается к значению 0.5. Однако на сегодняшний день это одно из лучших найденных уравнений для алгоритма шифрования DES.

Если выполнить аналогичные действия для алгоритма шифрования DES с нерассмотренными нами числами раундов, то можно заметить, что принцип построения линейного статистического аналога одинаков для алгоритма шифрования при увеличении количества его раундов на четыре, то есть при нахождении аналога используется один и тот же принцип объединения раундов. При этом уравнения, используемые в том или ином раунде, могут быть различными, но порядок использования раундов остается неизменным. Так для алгоритма шифрования DES, содержащего 5, 9, 13, 17 и т. д. раундов, всегда будут задействованы два первых и два последних раунда шифрования. Для алгоритма шифрования DES, содержащего 6, 10, 14, 18 и т. д. раундов, всегда будет задействован последний раунд шифрования, и никогда не будут задействованы первый и предпоследний раунды. А объединение в тройки будет всегда начинаться со второго раунда шифрования. Для алгоритма шифрования DES, содержащего 7, 11, 15, 19 и т. д. раундов, всегда будут задействованы первый и последний раунды шифрования, и никогда не будут задействованы второй и предпоследний. И, наконец, для алгоритма шифрования DES, содержащего 8, 12, 16, 20 и т. д. раундов, всегда будут задействованы первый и два последних раунда шифрования. Все остальные раунды шифрования независимо от общего количества раундов в алгоритме будут объединяться в тройки, разделенные между собой неиспользуемыми раундами шифрования.

5.3. Линейный криптоанализ шифровальной сети на основе подстановок и перемешивания

5.3.1. Шифровальная сеть на основе подстановок и перемешивания

Шифр, который мы собираемся представить к рассмотрению, — это шифровальная сеть на основе подстановок и перемешивания (Substitution—Permutation Network (SPN)) [18]. Рассмотрим алгоритм шифрования, показанный на рис. 5.8. На вход данного алгоритма поступает 16-битовый блок входного текста. Этот блок обрабатывается с помощью повторения четырех циклов (раундов), состоящих из простейших операций замены и перестановки битов (т. е. перестановки битовых позиций) и сложения с ключом. Используемые в алгоритме простейшие операции аналогичны тем, которые используются в алгоритме шифрования DES, а также многих других современных шифрах, включая шифр Rijndael. И, хотя мы рассматриваем простую структуру, анализ атак на такого рода шифры представляет ценное понимание в защите больших, часто используемых алгоритмов шифрования.

В нашем алгоритме мы разделяем исходный 16-битовый блок данных на четыре подблока. Каждый подблок формируется как вход в 4 × 4 S-блок (замена четырех входных битов на четыре выходных бита), который может быть легко представлен в виде таблицы, состоящей из 16 4-битовых величин, пронумерованных целыми числами, изображающими 4 входных бита. Наиболее важное свойство S-блоков — это нелинейное преобразование, то есть выходные биты не могут быть представлены в виде линейной функции, зависящей от входных битов.

Мы будем использовать аналогичное нелинейное преобразование для всех S-блоков. В алгоритме шифрования DES все S-блоки цикла различны, однако все циклы используют один и тот же набор S-блоков. Атаки линейного и дифференциального криптоанализа применяются равно как для одинаково построенных S-блоков, так и для S-блоков, имеющих различное строение. Строение S-блока, выбранное для нашего примера, приведено в таблице 5.6 и представляет собой первую строку первого S-блока алгоритма шифрования DES. В данной таблице наиболее значимый бит шестнадцатеричных значений является самым левым битом выхода S-блока на рис. 5.8.

Таблица 5.6

S-блок в шестнадцатеричной системе счисления

Вход	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Выход	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

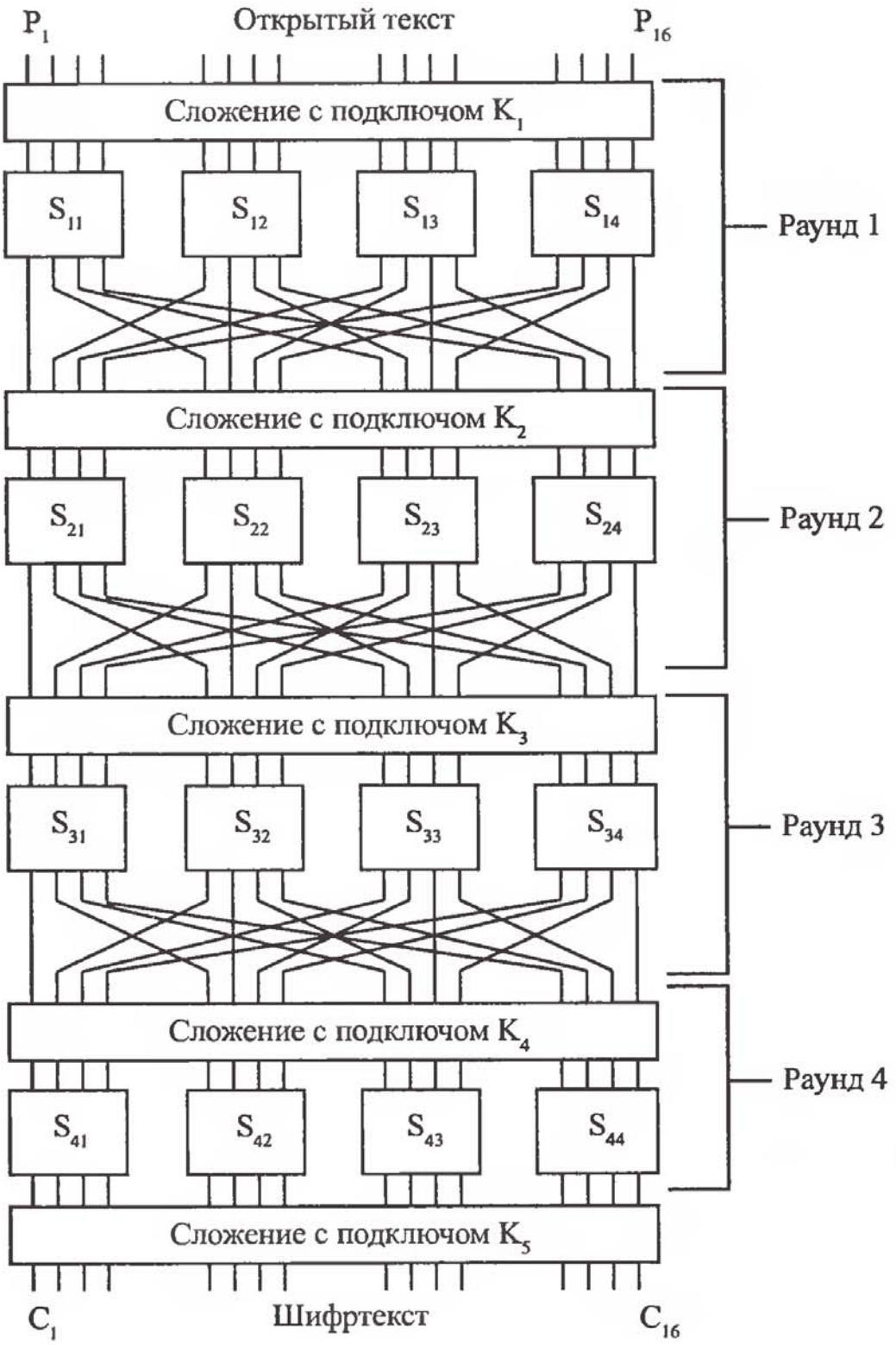


Рис. 5.8. ***Шифровальная сеть на основе подстановок и перемешивания***

Перестановка каждого цикла представляет собой простую перестановку битов сообщения, или изменение позиций битов. Перестановка, показанная на рис. 5.8 для удобства приведена в таблице 5.7 (где номера обозначают позиции битов в блоке: первый бит является крайним левым, а последний — крайним пра-

вым) и может быть просто описана следующим образом: выход i S-блока j соединен со входом j S-блока i. В последнем цикле перестановка отсутствует.

Таблица 5.7

Перестановка битов

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

Для сложения с ключом используется простая операция побитового сложения по модулю 2 (XOR) между битами подключа и битами блока данных, поступающего на вход данного цикла. Кроме того, происходит сложение битов подключа с выходными битами последнего раунда. Это делается для того, чтобы затруднить анализ алгоритма дешифрования. Обычно в алгоритме шифрования подключ для цикла шифрования извлекается из главного ключа. В рассматриваемом алгоритме шифрования предположим, что все биты подключа генерируются независимо и не связаны друг с другом.

При дешифровании данные необходимо пропустить через алгоритм шифрования в обратном порядке. Таким образом, алгоритм дешифрования имеет ту же форму, что и изображенный на рис. 5.8 [18]. Однако схема использования S-блоков в алгоритме дешифрования отличается от схемы использования S-блоков при зашифровании: входы становятся выходами, а выходы — входами. Это означает, что все S-блоки должны быть симметричными, то есть один к одному повторять схему с теми же номерами входных и выходных битов. Чтобы шифр был правильно дешифрован, необходимо использовать все подключи в обратном порядке, и биты подключа должны быть переставлены в соответствии с таблицей перестановки. Отсутствие перестановки после последнего цикла обеспечивает для алгоритма дешифрования ту же структуру, что и для алгоритма шифрования.

5.3.2. Принцип накопления

Прежде чем мы рассмотрим построение линейных выражений для шифра, приведенного в качестве примера в данном пособии, нам необходимо ознакомиться с некоторыми базовыми понятиями. Предположим, что вероятности распределены следующим образом:

$$P(X_1 = i) = p_1, \quad \text{если } i = 0;$$

$$P(X_1 = i) = 1 - p_1, \quad \text{если } i = 1;$$

и

$$P(X_2 = i) = p_2, \quad \text{если } i = 0;$$

$$P(X_2 = i) = 1 - p_2, \quad \text{если } i = 1.$$

Если две случайных величины независимы, то:

$$P(X_1 = i, X_2 = j) = p_1 p_2, \quad \text{если } i = 0, j = 0;$$
$$P(X_1 = i, X_2 = j) = p_1(1 - p_2), \quad \text{если } i = 0, j = 1;$$
$$P(X_1 = i, X_2 = j) = (1 - p_1) p_2, \quad \text{если } i = 1, j = 0;$$
$$P(X_1 = i, X_2 = j) = (1 - p_1)(1 - p_2), \quad \text{если } i = 1, j = 1.$$

Тогда мы можем их использовать следующим образом:

$$P(X_1 \oplus X_2 = 0) = P(X_1 = X_2) = \\ = P(X_1 = 0, X_2 = 0) + P(X_1 = 1, X_2 = 1) = p_1p_2 + (1 - p_1)(1 - p_2). \quad (5.20)$$

С другой стороны, мы знаем, что отклонение Δ_1 для вероятности p_1 равно

$$\Delta_1 = |1 - 2p_1|, \quad (5.21)$$

и отклонение Δ_2 для вероятности p_2 равно

$$\Delta_2 = |1 - 2p_2|. \quad (5.22)$$

При этом $0 \le \Delta_1, \Delta_2 \le 1$. Отсюда следует, что

$$2p_1 = 1 + \Delta_1,$$
$$2p_2 = 1 + \Delta_2,$$

а значит,

$$p_1 = 1/2 + 2^{-1}\Delta_1; \quad (5.23)$$
$$p_2 = 1/2 + 2^{-1}\Delta_2. \quad (5.24)$$

Подставив (5.23) и (5.24) в (5.20), получим:

$$P(X_1 \oplus X_2 = 0) = p_1p_2 + (1 - p_1)(1 - p_2) = \\ = (1/2 + 2^{-1}\Delta_1)(1/2 + 2^{-1}\Delta_2) + (1 - (1/2 + 2^{-1}\Delta_1))(1 - (1/2 + 2^{-1}\Delta_2)) = \\ = 1/4 + 2^{-2}\Delta_1 + 2^{-2}\Delta_2 + 2^{-4}\Delta_1\Delta_2 + (1/2 - 2^{-1}\Delta_1)(1/2 - 2^{-1}\Delta_2) = \\ = 1/4 + 2^{-2}\Delta_1 + 2^{-2}\Delta_2 + 2^{-4}\Delta_1\Delta_2 + 1/4 - 2^{-2}\Delta_1 - 2^{-2}\Delta_2 + 2^{-4}\Delta_1\Delta_2 = \\ = 1/2 + 2^{-2}\Delta_1\Delta_2. \quad (5.25)$$

Теперь подставим выражения (5.21) и (5.22) в (5.25). Получим:

$$P(X_1 \oplus X_2 = 0) = 1/2 + 2^{-2}\Delta_1\Delta_2 = 1/2 + 2^{-2}(|1 - 2p_1|)(|1 - 2p_2|) = \\ = 1/2 + 2^{-2}(2p^1 - 1)(2p_2 - 1) = 1/2 + 2(p_1 - 1/2)(p_2 - 1/2).$$

Это может быть распространено больше, чем на 2 случайные двоичные величины от X_1 до X_n, при этом вероятность того, что $X_1 \oplus \ldots \oplus X_n = 0$, может быть определена с помощью Накопительной леммы [6, 18], которая предполагает, что все n случайных двоичных величин независимы.

Накопительная лемма

Для n независимых, случайных двоичных величин $X_1, X_2, \ldots, X_n$

$$P(X_1 \oplus \ldots \oplus X_n = 0) = 1/2 + 2_{n-1}((p_1 - 1/2)*(p_2 - 1/2)*\ldots*(p_n - 1/2)).$$

Отметим, что если $p_i = 0$ или 1 для всех i, то $P(X_1 \oplus \ldots \oplus X_n = 0) = 0$ или 1. Если только одна вероятность $p_i = 1/2$, то $P(X_1 \oplus \ldots \oplus X_n = 0) = 1/2$.

В построении линейного приближения для шифра величина X_i будет фактически представлять собой линейное приближение S-блоков.

5.3.3. Анализ компонентов шифра

Рассмотрим S-блок, изображенный на рис. 5.9, с входным вектором $X = [X_1, X_2, X_3, X_4]$ и соответствующим ему выходным вектором $Y = [Y_1, Y_2, Y_3, Y_4]$. Все линейные приближения могут быть протестированы для определения их эффективности путем вычисления вероятности отклонения для каждого. В результате анализа таблицы замены 5.7, мы получаем таблицу 5.8, аналогичную той, что мы получали при анализе алгоритма шифрования DES.

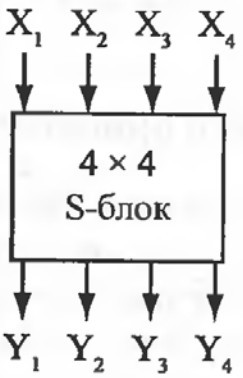


Рис. 5.9. ***Строение S-блока***

Таблица 5.8

Таблица линейных приближений

Выходные суммы / Входные суммы	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	6	6	0	0	6	14	10	10	0	0	10	10	0	0
2	0	0	6	6	0	0	6	6	0	0	10	10	0	0	2	10
3	0	0	0	0	0	0	0	0	10	2	6	6	10	10	6	6
4	0	10	0	6	6	4	6	0	0	6	0	10	10	4	10	0
5	0	6	6	0	6	0	12	10	6	0	4	10	0	6	6	0
6	0	10	6	12	10	0	0	10	0	6	10	12	6	0	0	6
7	0	6	0	10	10	4	10	0	6	0	10	0	12	10	0	10
8	0	0	0	0	0	0	0	0	6	10	10	6	10	6	6	2
9	0	0	6	6	0	0	6	6	4	0	6	10	0	12	10	6
A	0	12	6	10	4	0	10	6	10	10	0	0	10	10	0	0
B	0	12	0	4	12	0	12	0	0	0	0	0	0	0	0	0
C	0	6	12	6	6	0	10	0	10	0	10	12	0	10	0	6
D	0	10	10	0	6	12	0	10	4	6	10	0	10	0	0	10
E	0	10	10	0	6	4	0	10	6	0	0	6	4	10	6	0
F	0	6	4	6	6	0	10	0	0	6	12	6	6	0	10	0

Шестнадцатиричная величина, представляющая собой сумму, определяет переменные, участвующие в сложении. Мы знаем, что для линейной комбинации входных переменных можно записать $a_1 \cdot X_1 \oplus a_2 \cdot X_2 \oplus a_3 \cdot X_3 \oplus a_4 \cdot X_4$, где $a_i \in \{0,1\}$ и знак « · » обозначает двоичную операцию AND, шестнадцатиричная переменная представляет собой двоичную величину $a_1a_2a_3a_4$, где a_1 — старший бит. Аналогично для линейной комбинации выходных битов будем иметь $b_1 \cdot Y_1 \oplus \oplus b_2 \cdot Y_2 \oplus b_3 \cdot Y_3 \oplus b_4 \cdot Y_4$, где $b_i \in \{0,1\}$, и шестнадцатиричное значение представляется двоичным вектором $b_1b_2b_3b_4$. Из таблицы 5.8 следует, что вероятность линейного равенства $a_1 \cdot X_1 \oplus a_2 \cdot X_2 \oplus a_3 \cdot X_3 \oplus a_4 \cdot X_4$, где $a_i \in \{0,1\}$, $X_3 \oplus X_4 = Y_1 \oplus Y_4$ (то есть входная величина — 3, а выходная — 9), равна 2/16 = 1/8, и отклонение равно $\Delta = |1 - 2p| = |1 - 1/4| = 3/4$.

5.3.4. Построение линейного приближения для полного шифра

Одиночные линейные приближения должны быть собраны для всех S-блоков алгоритма шифрования SPN. У нас есть данные для нахождения линейного приближения для всего алгоритма шифрования. Оно может быть найдено с помощью объединения соответствующих линейных приближений S-блоков. Построение линейного приближения, включающего биты исходного открытого текста и выходные биты последнего цикла S-блоков, позволяет осуществлять атаку на шифр, получая биты подключа последнего цикла. Покажем это на примере.

Будем строить линейное приближение таким образом, чтобы вероятность приближения каждого из S-блоков замены, участвующих в построении общего приближения, отличалась от 1/2, и при этом, по возможности, линейное приближение должно иметь как можно большее отклонение.

Рассмотрим приближение, включающее блоки S_{12}, S_{22}, S_{32} и S_{42}, как это показано на рис. 5.10. Можно заметить, что это приближение в действительности включает выражения первых трех циклов и затрагивает четвертый цикл не полностью. В пункте 5.3.5 мы увидим, насколько это полезно при извлечении битов подключа, используемого после последнего цикла.

Мы используем следующие приближения S-блоков (см. таблицу 5.8):

S_{12}: $X_1 \oplus X_3 \oplus X_4 = Y_2$ с вероятностью $p_1 = 12/16 = 3/4$ и отклонением 1/2;

S_{22}: $X_2 = Y_2 \oplus Y_4$ с вероятностью $p_2 = 4/16 = 1/4$ и отклонением 1/2;

S_{32}: $X_2 = Y_2 \oplus Y_4$ с вероятностью $p_3 = 4/16 = 1/4$ и отклонением 1/2;

S_{34}: $X_2 = Y_2 \oplus Y_4$ с вероятностью $p_4 = 4/16 = 1/4$ и отклонением 1/2.

Следует отметить, что индексы в вышеуказанных приближениях взяты в рамках одного обособленного блока, то есть, например, для блока S_{22} значение X_2 указывает на второй бит данных, входящих в этот блок, хотя на самом деле этот бит будет являться шестым битом общего блока данных.

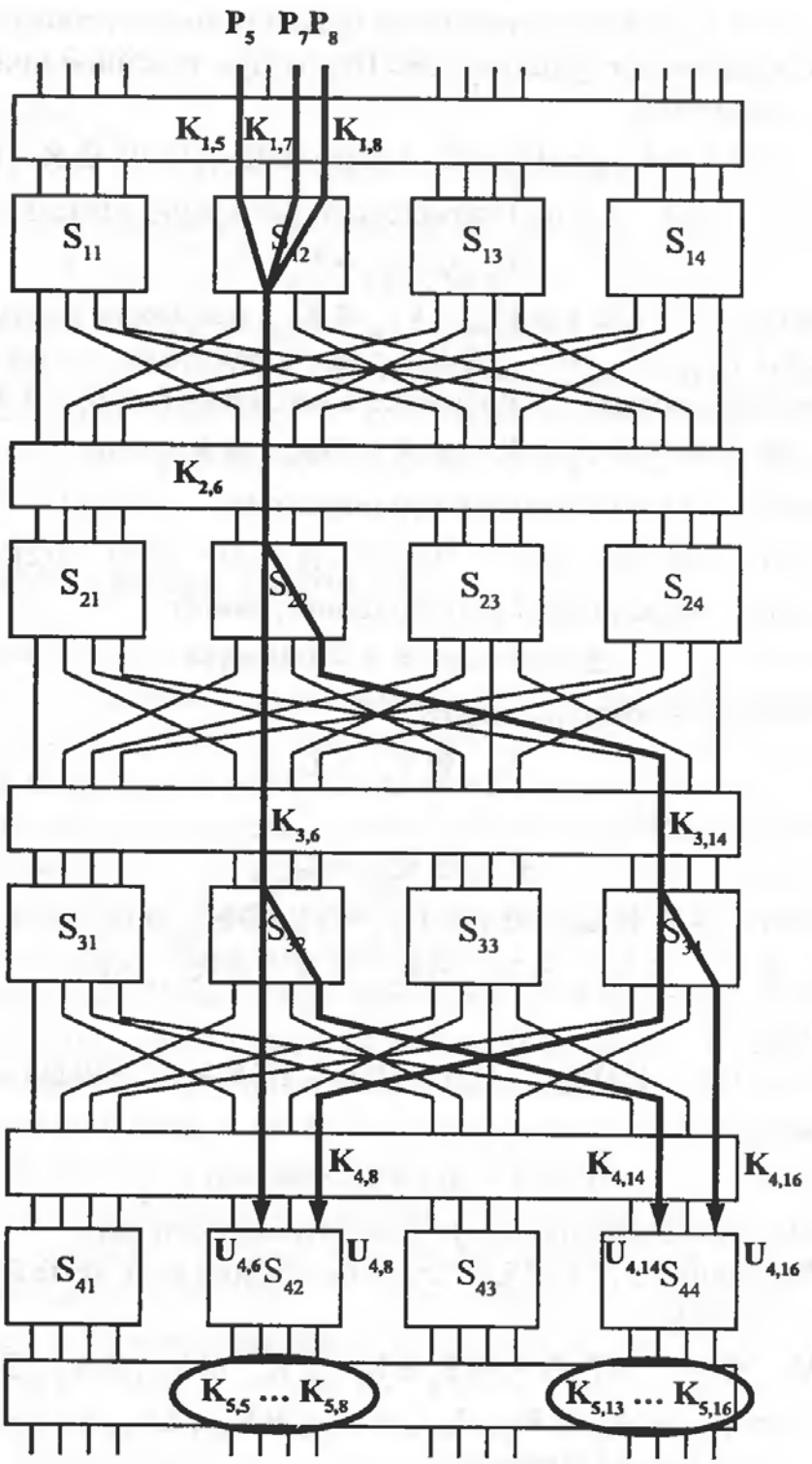


Рис. 5.10. ***Пример линейного приближения***

Пусть $U_i(V_i)$ обозначает блок входных (выходных) данных цикла, содержащий 16 бит, для i-го S-блока, и $U_{ij}(V_{ij})$ обозначает j-й бит блока $U_i(V_i)$ (где биты нумеруются от 1 до 16 слева направо). Также пусть K_i обозначает подключ, который складывается по модулю 2 с входным блоком i-го цикла за исключением подключа K_5, который складывается по модулю 2 с выходом четвертого цикла.

Итак, $U_1 = P \oplus K_1$, где P обозначает 16-битный блок исходного открытого текста и знак $\oplus$ обозначает операцию XOR. Используя линейное приближение первого цикла, мы получаем:

$$V_{1,6} = U_{1,5} \oplus U_{1,7} \oplus U_{1,8} = (P_5 \oplus K_{1,5}) \oplus (P_7 \oplus K_{1,7}) \oplus (P_8 \oplus K_{1,8}) \quad (5.26)$$

с вероятностью $p_1 = 3/4$. Для приближения второго раунда имеем:

$$V_{2,6} \oplus V_{2,8} = U_{2,6}$$

с вероятностью $p_2 = 1/4$. Так как $U_{2,6} = V_{1,6} \oplus K_{2,6}$, мы можем получить приближение для формулы $V_{2,6} \oplus V_{2,8} = V_{1,6} \oplus K_{2,6}$ с вероятностью $p_2 = 1/4$ и, объединив это с выражением (5.26), которое выполняется с вероятностью $p_1 = 3/4$, получим:

$$V_{2,6} \oplus V_{2,8} \oplus P_5 \oplus P_7 \oplus P_8 \oplus K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} = 0. \quad (5.27)$$

Выражение (5.27) выполняется с вероятностью

$$p_5 = 1/2 + 2^{n-1}((p_1 - 1/2)*(p_2 - 1/2)*\ldots*(p_n - 1/2)) = 1/2 + 2(3/4 - 1/2)(1/4 - 1/2) = 3/8,$$

исходя из Накопительной леммы и отклонения, равного:

$$\Delta_5 = |1 - 2p_5| = |1 - 3/4| = 1/4.$$

Для третьего цикла мы определяем:

$$V_{3,6} \oplus V_{3,8} = U_{3,6}$$

с вероятностью p3 = 1/4 и

$$V_{3,14} \oplus V_{3,16} = U_{3,14}$$

с вероятностью $p_4 = 1/4$. Итак, так как $U_{3,6} = V_{2,6} \oplus K_{3,6}$ и $U_{3,14} = V_{2,8} \oplus K_{3,14}$, то

$$V_{3,6} \oplus V_{3,8} \oplus V_{3,14} \oplus V_{3,16} \oplus V_{2,6} \oplus K_{3,6} \oplus V_{2,8} \oplus K_{3,14} = 0 \quad (5.28)$$

с вероятностью

$$p_6 = 1/2 + 2^{n-1}((p_1 - 1/2)*(p_2 - 1/2)*\ldots*(p_n - 1/2)) = 1/2 + 2(1/4 - 1/2)^2 = 5/8$$

и отклонением

$$\Delta_6 = |1 - 2p_6| = |1 - 5/4| = 3/4.$$

При этом мы снова использовали Накопительную лемму.

Теперь объединим (5.27) и (5.28) для соединения всех четырех приближений S-блоков. Мы получим:

$$V_{3,6} \oplus V_{3,8} \oplus V_{3,14} \oplus V_{3,16} \oplus P_5 \oplus P_7 \oplus P_8 \oplus K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14} = 0.$$

Заметим, что $U_{4,6} = V_{3,6} \oplus K_{4,6}$, $U_{4,8} = V_{3,14} \oplus K_{4,8}$, $U_{4,14} = V_{3,8} \oplus K_{4,14}$ и $U_{4,16} = V_{3,16} \oplus K_{4,16}$, тогда можно записать:

$$U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \oplus P_5 \oplus P_7 \oplus P_8 \oplus \Sigma_K = 0,$$

где

$$\Sigma_K = K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14} \oplus K_{4,6} \oplus K_{4,8} \oplus K_{4,14} \oplus K_{4,16}$$

и Σ_K принимает значение 0 или 1 в зависимости от ключа шифра. Используя Накопительную лемму, определим, что приведенное выше выражения выполняется с вероятностью

$$p_7 = 1/2 + 2^{n-1}((p_1 - 1/2)*(p_2 - 1/2)*\ldots$$
$$\ldots*(p_n - 1/2)) = 1/2 + 2^3(3/4 - 1/2)(1/4 - 1/2)^3 = 15/32$$

и имеет отклонение

$$\Delta_7 = |1 - 2p_7| = |1 - 15/16| = 1/16.$$

Теперь, так как значение Σ_K определено, мы заметим, что

$$U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \oplus P_5 \oplus P_7 \oplus P_8 = 0 \qquad (5.29)$$

должно выполняться с вероятностью 15/32 или (1 – 15/32) в зависимости от того, равна сумма Σ_K нулю или единице, соответственно. Другими словами, теперь мы имеем линейное приближение первых трех циклов с величиной отклонения 1/16. Теперь рассмотрим, как это отклонение может быть использовано для нахождения некоторых битов ключа.

5.3.5. Нахождение битов ключа

Так как линейное приближение R–1 цикла найдено для алгоритма шифрования, содержащего R циклов, с подходящей довольно большой вероятностью отклонения, становится возможным произвести атаку с помощью нахождения битов последнего подключа. В нашем случае возможно извлечь биты из подключа K_5 с помощью линейного приближения 3 цикла. Назовем найденные биты последнего подключа искомым частичным подключом. Очевидно, что биты искомого частичного подключа связаны с S-блоком последнего раунда, влияющим на биты данных, используемые в линейном приближении.

Последующий процесс включает частичное дешифрование последнего цикла. Очевидно, что для всех возможных значений искомого частичного подключа соответствующие биты шифртекста складываются по модулю 2 с битами искомого частичного подключа, и результат пропускается в обратном порядке через соответствующие S-блоки. Это делается для всех известных пар *открытый—закрытый текст*, и определяется количество совпадений для каждого значения искомого частичного подключа. Количество для данного искомого частичного подключа счетчик увеличивает на 1, когда линейное выражение истинно для битов в S-блоках последнего раунда (определяемых частичной дешифрацией) и битов исходного открытого текста. Значение искомого частичного подключа, счетчик которого имеет самое большое отличие от половины пар *открытый—закрытый текст*, определяется как правильное значение искомого частичного подключа. Этот механизм работает за счет того, что предположение правильного значения искомого частичного подключа будет результатом линейного приближения, выполняющегося с вероятностью, наиболее отличающейся от 1/2. (Независимо от того, больше 1/2 эта вероятность или меньше.) Неправильный подключ получается в результате относительно случайного предположения битов, входящих в

S-блок последнего раунда, а следовательно, линейное выражение будет выполняться с вероятностью, близкой к 1/2.

Теперь рассмотрим вышесказанное в контексте нашего примера. Линейное выражение (5.19) использует входы S-блоков S_{42} и S_{44} последнего цикла. Для каждой пары *открытый—закрытый текст* мы попробуем все 256 значений для искомого частичного подключа $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$. Для каждого значения частичного подключа мы будем увеличивать значение счетчика на единицу, когда равенство (5.19) будет истинно. Значение $[U_{4,5}...U_{4,8}, U_{4,13}...U_{4,16}]$ мы определяем, пропуская данные в обратном порядке через искомый частичный подключ и S-блоки S_{42} и S_{44}. Счетчик, значение которого больше всего отличается от половины пар *открытый—закрытый текст*, укажет нам правильное значение частичного ключа. Положительное отклонение или отрицательное — будет зависеть от значений битов подключа, включенных в Σ_K. Когда сумма Σ_K равна нулю, линейное приближение (5.29) будет оцениваться с вероятностью меньше 1/2, а когда равна единице — с вероятностью, большей 1/2.

Таблица 5.9

Экспериментальные результаты для атаки с помощью линейного криптоанализа

Частичный подключ $[K_5,5...K_5,8, K_5,13...K_5,16]$	|отклонение|	Частичный подключ $[K_5,5...K_5,8, K_5,13...K_5,16]$	|отклонение|
1 C	0.0031	2 A	0.0044
1 D	0.0078	2 B	0.0186
1 E	0.0071	2 C	0.0094
1 F	0.0170	2 E	0.0053
2 0	0.0025	2 F	0.0062
2 1	0.0220	3 0	0.0133
2 2	0.0211	3 1	0.0027
2 3	0.0064	3 2	0.0050
2 4	**0.0336**	3 3	0.0075
2 5	0.0106	3 4	0.0162
2 6	0.0096	3 5	0.0218
2 7	0.0074	3 6	0.0052
2 8	0.0224	3 7	0.0056
2 9	0.0054	3 8	0.0048

В [18] приводятся результаты атаки на данный алгоритм шифрования при помощи генерации 10 000 пар *открытый—закрытый текст*. Следующий криптоаналитический процесс описан для значений частичных подключей $[K_{5,5}...$ $...K_{5,8}] = [0010]$ (2 в шестнадцатеричной системе счисления) и $[K_{5,13}...$ $K_{5,16}] = [0100]$ (4 в шестнадцатеричной системе счисления). Как и следовало ожи-

дать, счетчик, значение которого наибольшим образом отличается от 5000, соответствует значению искомого частичного подключа $[2,4]_{hex}$ и доказывает, что с помощью данной атаки мы успешно определили биты подключа. Это иллюстрирует таблица 5.9, где приведены значения частичных сумм Σ_K данных, полученных из соответствующих счетчиков подключей. (Полный объем данных включает 256 записей для каждого из значений искомого частичного подключа.) Значения в таблице показывают величину отклонения, определяемую по формуле:

$$|\text{отклонение}| = |\text{значение счетчика} - 5000| / 10000,$$

где значение счетчика — это количество соответствующих значений для данного частичного подключа.

Как видно из таблицы (5.9), наибольшее отклонение соответствует значению частичного подключа $[K_{5,5}\ldots K_{5,8}, K_{5,13}\ldots K_{5,16}] = [2,4]$, и эти значения действительно соответствуют истинным битам подключа.

Экспериментально определенное значение отклонения, равное 0,0336, очень близко к ожидаемому значению 1/32 = 0,03125.

5.4. Линейный криптоанализ алгоритма шифрования RC5

5.4.1. Основные понятия

Основа линейного криптоанализа состоит в использовании линейного приближения, зависящего от битов открытого текста P, битов шифртекста C и битов используемого секретного ключа K. Такое линейное приближение может быть представлено в виде следующего выражения, выполняющегося с некоторой вероятностью:

$$\alpha \cdot P \oplus \beta \cdot K = \gamma \cdot C, \qquad (5.30)$$

где α, β и γ — двоичные вектора,

$\chi \cdot X = \chi_0 X_0 \oplus \chi_1 X_1 \oplus \ldots \oplus \chi_n X_n$ (n — размерность двоичного вектора χ).

Вместо P и C могут быть использованы промежуточные значения, которые можно вычислить из P и C с помощью предполагаемого значения ключа. Если линейное приближение выполняется с вероятностью $p = \frac{1}{2} + \delta$, где $\delta \neq 0$, линейная атака может быть осуществлена с использованием $m \cdot |\delta|^{-2}$ открытых текстов. При этом значение m зависит от анализируемого алгоритма шифрования. Величина δ определяет отклонение, а $|\delta| = \varepsilon$ называется **смещением** [25].

Так как ключ K является фиксированным, то выражение (5.30) может быть преобразовано к выражению (5.31) без изменения значения смещения:

$$\alpha \cdot P = \gamma \cdot C. \qquad (5.31)$$

Мы отметили, что для известных значений $\tilde{P}$ и $\tilde{C}$ выражение (5.31) ориентировано в сторону отклонения, если $\alpha \cdot P \oplus \gamma \cdot C = b$ и $\alpha \cdot \tilde{P} \oplus \gamma \cdot \tilde{C} = b$ имеют положительные отклонения [25].

Линейное приближение для полного алгоритма шифрования может быть получено с помощью объединения линейных приближений между промежуточными значениями. Если вероятности этих приближений независимы, то значение отклонения может быть вычислено с использованием Накопительной леммы (см. п. 5.3.2).

5.4.2. Основная идея анализа алгоритма шифрования RC5

В этом разделе мы опишем основную идею анализа алгоритма шифрования RC5 (описание алгоритма приведено в п. 4.1). Заметим, что при анализе данного алгоритма можно преследовать цель нахождения как исходного секретного ключа, так и расширенной таблицы ключей S.

Общая идея анализа заключается в сведении проблемы вычисления расширенной таблицы ключей S к проблеме вычисления значения $R_{n-1}[b]$ для некоторого $0 \le b \le \omega - 1$. (Заметим, что $R_{n-1}[b]$ — это один бит правой части входного значения последнего полураунда. И при этом, зная шифртекст, нельзя определить его значение.) В самом общем виде это может быть достигнуто с помощью следующих двух шагов:

1. Сведение проблемы вычисления расширенной таблицы ключей S к вычислению последнего подключа S_n. Это возможно в связи с повторяющейся структурой алгоритма шифрования;
2. Сведение проблемы вычисления последнего подключа S_n к вычислению значения $L_{n-1}[b]$. Это возможно из-за структуры последнего полураунда.

Таким образом, необходимо сосредоточить свое внимание на анализе последнего полураунда. Рассмотрим два равенства, выполняемых в последнем полураунде:

$$Rn = L_{n-1};$$
$$L_n = ((L_{n-1} \oplus R_{n-1}) <<< L_{n-1}) + S_n,$$

где операция <<< определяет циклический сдвиг данных в зависимости от последних $\log_2 w$ битов значения L_{n-1}.

В последнем равенстве присутствуют четыре переменные. Две из них: L_{n-1} и L_n нам известны, так как нам известно значение шифртекста. Таким образом, если мы сможем узнать значение R_{n-1}, то мы легко сможем узнать и значение последнего подключа S_n. Известно, что сдвиг суммы $(L_{n-1} \oplus R_{n-1})$ зависит от последних ω бит значения L_{n-1}, то есть сдвиг производится на $(L_{n-1} \bmod \omega)$ позиций.

Рассмотрим вначале случай, когда $L_{n-1}[b] \bmod \omega = 0$. В этом случае b-й бит суммы $(L_{n-1} \oplus R_{n-1})$ после сдвига окажется в нулевой позиции, а значит,

$$L_n[0] = (L_{n-1}[b] \oplus R_{n-1}[b]) \oplus S_n[0].$$

Так как значения $L_n[0]$ и $L_{n-1}[b]$ нам известны, то, если определить значение бита $R_{n-1}[b]$, можно вычислить значение младшего значащего бита последнего подключа $S_n[0]$.

В общем случае может оказаться, что $L_{n-1}[b] \bmod \omega = s$. Это немного сложнее предыдущего случая в связи с тем, что велика возможность возникновения переноса при сложении данных с ключом S_n. Пусть

$$Y = (L_{n-1} \oplus R_{n-1}) <<< L_{n-1},$$

а значит,

$$L_n = Y + S_n.$$

Обозначим через carry(s) значение переноса от сложения $Y[s-1\ldots0] + S_n[s-1\ldots0]$.

Тогда получается, что:

$$L_n[s] = Y[s] \oplus S_n[s] \oplus carry(s) = (L_{n-1}[b] \oplus R_{n-1}[b]) \oplus S_n[s] \oplus carry(s).$$

Если нам известно значение $S_n[s-1\ldots0]$, то, зная шифртекст (L_n, R_n), можно вычислить значение переноса carry(s) по следующей схеме:

$$\text{если } S_n[s-1\ldots0] \le R_n[s-1\ldots0], \text{ то } carry(s) = 0,$$

иначе carry(s) = 1.

Таким образом, нахождение секретного подключа сводится к нахождению значения R_{n-1}.

5.4.3. Линейные приближения для алгоритма RC5

Как уже говорилось, суть линейного криптоанализа заключается в сведении нелинейных криптографических преобразований к их линейным аналогам, выполняющимся с определенной вероятностью. Если принять во внимание все вышесказанное, то получается, что нам необходимо найти такой линейный статистический аналог для алгоритма шифрования RC5, чтобы из него можно было определить значение R_{n-1}, с помощью которого становится возможным нахождение таблицы секретных ключей S.

Преобразование одного полураунда алгоритма шифрования RC5 можно представить в виде трех последовательных простых операций (см. п. 4.1 настоящей книги):

1. Операция сложения по модулю два правой и левой частей текста:

$$X = L_{i-1} \oplus R_{i-1};$$

2. Операция зависимого сдвига:

$$Y = X <<< L_{i-1};$$

3. Операция целочисленного сложения данных с секретным ключом:

$$L_i = Y + S_i.$$

Рассмотрим каждую из этих операций в отдельности. Первая операция будет иметь отклонение в том случае, если мы будем рассматривать одни и те же биты для значений X, L_{i-1} и R_{i-1}, то есть линейный аналог вида:

$$X[i] = L_{i-1}[i] \oplus R_{i-1}[i] \qquad (5.32)$$

будет выполняться с вероятностью p = 1. В случае, если будут рассматриваться разные биты значений X, L_{i-1} и R_{i-1}, вероятность будет равна 1/2, а значит, отклонение для такого аналога будет равно 0, что говорит о его неэффективности.

Операция зависимого сдвига может быть рассмотрена с двух разных позиций, в зависимости от того рассматривать ли значение L_{i-1} целиком или лишь некоторые его биты. В первом случае получается, что операцию зависимого сдвига можно заменить линейным аналогом вида:

$$Y[i] = X[i],$$

который выполняется с вероятностью

$$p = \frac{1}{2} + \frac{1}{2\omega}.$$

Вероятность в данном случае вычисляется с учетом двух факторов. В случае, если последние ω бит значения $L_{i-1}[i]$ будут равны нулю, все биты значения X останутся равными значению Y (то есть никакого сдвига не произойдет). Это возможно в одном случае из ω, то есть с вероятностью 1/ω. В остальных (ω – 1) случаях биты значений X и Y останутся равными с вероятностью 1/2. А значит, вероятность того, что Y[i] = X[i] для операции зависимого сдвига равна:

$$p = \frac{1}{\omega} * 1 + \frac{\omega - 1}{\omega} * \frac{1}{2} = \frac{1}{\omega} + \frac{1}{2} - \frac{1}{2\omega} = \frac{1}{2} + \frac{1}{2\omega}.$$

Если же рассматривать лишь некоторые биты значения $L_{i-1}[i]$, то можно построить много различных приближений. Одним из них будет приближение вида:

$$Y[0] = X[0] \oplus L_{i-1}[0]. \qquad (5.33)$$

Вероятность выполнения которого, также как и в предыдущем случае, равна $p = \frac{1}{2} + \frac{1}{2\omega}$ (в случае, если последние ω бит равны нулю (1 случай из ω), то X останется равным Y и при добавлении $L_{i-1}[0] = 0$ ничего не изменит, а в остальных (ω – 1) случаях их ω вероятность равенства будет равна 1/2).

Для операции целочисленного сложения данных с ключом существует единственный линейный статистический аналог, вероятность выполнения которого равна p = 1:

$$\begin{aligned} Li[0] &= Y[0] \oplus Si[0], \text{ если } i \le 2r, \\ Ri[0] &= Y[0] \oplus Si[0], \text{ если } i > 2r, \end{aligned} \qquad (5.34)$$

где 2r — общее число полураундов.

Все остальные аналоги не являются эффективными и зависят, в первую очередь, от значения секретного ключа S. Так, например, вероятность выполнения линейного аналога $L_i[k] = Y[k] \oplus S_i[k]$ может меняться от 1/2 до 1/4 для $k \geq 1$, то есть среднее значение вероятности равно $p = 3/8$.

Перед тем как приступить к выполнению первого полураунда алгоритма шифрования RC5, происходит целочисленное сложение правой и левой частей исходного сообщения с секретными ключами S_0 и S_1. Таким образом, получается:

$$L_1[0] = R_0[0] \oplus S_1[0]; \tag{5.35}$$

$$R_1[0] = L_0[0] \oplus S_0[0]. \tag{5.36}$$

Объединив вместе выражения (5.33) и (5.34), а также выражение (5.32) для случая, когда $i = 0$, получим:

$$L_i[0] = R_{i-1}[0] \oplus S_i[0]. \tag{5.37}$$

Так как вероятность выполнения равенств (5.32) и (5.34) равна 1, а вероятность выполнения выражения (5.33) равна $p = \frac{1}{2} + \frac{1}{2\omega}$, то и для равенства (5.37) вероятность также будет равна $p = \frac{1}{2} + \frac{1}{2\omega}$.

В связи с тем, что в конце каждого полураунда части шифруемых данных меняются местами, то можно сказать, что $L_{i-1}[0] = R_i[0]$ с вероятностью $p = 1$. И только в последнем полураунде, когда не происходит обмена частями, получается, что $L_{2r}[0] = R_{2r+1}[0]$, где $2r$ — общее число полураундов.

Таким образом, для алгоритма шифрования RC5, состоящего из r раундов (2r полураундов), можно получить следующий линейный аналог.

Учитывая, что $R_{2r}[0] = L_{2r-1}[0]$, получаем, что

$$R_0[0] \oplus R_{2r}[0] = S_1[0] \oplus S_3[0] \oplus \ldots \oplus S_{2r-1}[0]. \tag{5.38}$$

Обозначим через T правую часть равенства (5.38). Тогда:

$$R_0[0] \oplus R_{2r}[0] = T. \tag{5.39}$$

После того как мы определили линейный статистический аналог, вычисление таблицы подключей будет производиться в три этапа.

I этап

На первом этапе мы определим значение $S_{2r+1}[0]$. Для этого мы будем рассматривать такие пары *открытый—закрытый текст*, в которых $L_{2r+1} \bmod w = 0$. В этом случае получается, что в последнем полураунде отсутствует циклический сдвиг. Поэтому можно выделить два эффективных линейных аналога:

$$L_{2r}[0] = L_{2r+1}[0] \oplus R_{2r+1}[0], \tag{5.40}$$

если $S_{2r+1}[0] = 0$ (то есть переноса не может быть);

$$L_{2r}[0] = L_{2r+1}[0] \oplus R_{2r+1}[1] \oplus R_{2r+1}[0], \qquad (5.41)$$

если $S_{2r+1}[0] = 1$ (то есть может возникнуть перенос, и он должен быть учтен).

Алгоритм

Пусть N — общее число текстов, для которых $L_{2r+1} \bmod w = 0$, U_0 — число текстов, для которых $R_0[0] \oplus (L_{2r+1}[0] \oplus R_{2r+1}[0]) = 0$, а U_1 — число текстов для которых $R_0[0] \oplus (L_{2r+1}[0] \oplus R_{2r+1}[1] \oplus R_{2r+1}[0]) = 0$. Тогда

если $|U_0 - N/2| \geq |U_1 - N/2|$, то $S_{2r+1}[0] = 0$, иначе $S_{2r+1}[0] = 1$.

II этап

На втором этапе определяем, чему равно значение T. Так как мы определили, что равенство (5.40) выполняется с вероятностью p = 1, то используем найденное значение $S_{2r+1}[0]$.

Алгоритм

Пусть N — общее число текстов, для которых $L_{2r+1} \bmod \omega = 0$, а U — число текстов, для которых $R_0[0] \oplus (L_{2r+1}[0] \oplus R_{2r+1}[1] \oplus R_{2r+1}[0]) = 0$. Тогда

если $U \geq N/2$, то $T = 0$, иначе $T = 1$.

III этап

Будем последовательно вычислять значения $S_{2r+1}[k]$ $(1 < k < \omega - 1)$ с помощью найденного значения T и ранее вычисленных значений $S_{2r+1}[t]$ $(k - 1 < t < 0)$.

Пусть $Y = R_{2r+1} — S_{2r+1}$ (то есть Y — это состояние данных до сложения с секретным подключом). Выше было введено обозначение carry(k), равное переносу от сложения последних k бит значения Y с последними k битами значения S_{2r+1} $(Y[k-1\ldots0] + S_{2r+1}[k-1\ldots0])$. Если нам известно значение $S_n[k-1\ldots0]$, то мы, зная шифртекст (L_{2r+1}, R_{2r+1}), можем вычислить значение переноса carry(k) по следующей схеме:

если $S_{2r+1}[k-1\ldots0] \leq R_{2r+1}[k-1\ldots0]$, то carry(k) = 0, иначе carry(k) = 1.

Для пары *открытый—закрытый текст*, для которой выполняется условие $L_{2r+1} \bmod \omega = k$, с вероятностью p = 1 выполняется линейное выражение:

$$L_{2r}[0] = L_{2r+1}[0] \oplus R_{2r+1}[k] \oplus S_{2r+1}[k] \oplus \text{carry}(k). \qquad (5.42)$$

Алгоритм

Пусть N — общее число текстов, для которых $L_{2r+1} \bmod \omega = k$, а U — число текстов, для которых $(R_0[0] \oplus T) \oplus (L_{2r+1}[0] \oplus R_{2r+1}[k] \oplus \text{carry}(k)) = 0$. Тогда

если $U \geq N/2$, то $S_{2r+1}[k] = 0$, иначе $S_{2r+1}[k] = 1$.

5.5. Контрольные вопросы

1. Дайте определение линейного статистического аналога функции шифрования.

2. На чем основывается принцип линейного криптоанализа?
3. Что определяет эффективность линейного статистического аналога?
4. Какое уравнение является эффективным линейным статистическим аналогом одного раунда алгоритма шифрования DES?
5. Дайте определение тройной связанной суммы.
6. Выведите эффективный линейный статистический аналог для трех раундов алгоритма шифрования DES.
7. По какому алгоритму производится извлечение битов ключа с помощью известного статистического аналога?
8. Какой принцип используется при построении линейных аналогов для n раундов алгоритма шифрования DES, где n>1?
9. Какой принцип используется при линейном криптоанализе алгоритма шифрования SPN?

Глава 6. ДИФФЕРЕНЦИАЛЬНЫЙ КРИПТОАНАЛИЗ БЛОЧНЫХ АЛГОРИТМОВ ШИФРОВАНИЯ

6.1. Общие сведения о дифференциальном криптоанализе

Понятие дифференциального криптоанализа впервые было введено в 1990 г. Эли Бихамом и Ади Шамиром. Используя этот метод, Бихам и Шамир нашли способ атаки алгоритма DES с использованием подобранного открытого текста, который оказался эффективнее атаки «в лоб».

Ядро метода составляет атака на основе выборочного открытого текста. Анализ заключается в изучении процесса изменения несходства для пары открытых текстов, имеющих определенные исходные различия, в процессе прохождения через циклы шифрования с одним и тем же ключом. Не существует каких-либо ограничений на выбор открытых текстов. Достаточно различия в некоторых позициях.

Исходя из различий в получившихся шифртекстах, ключам присваиваются различные вероятности. Истинный ключ определяется в процессе дальнейшего анализа пар шифртекстов как наиболее вероятный ключ среди множества претендентов.

Пусть некоторый блочный шифратор с длиной блока N строится по схеме, приведенной на рис. 6.1.

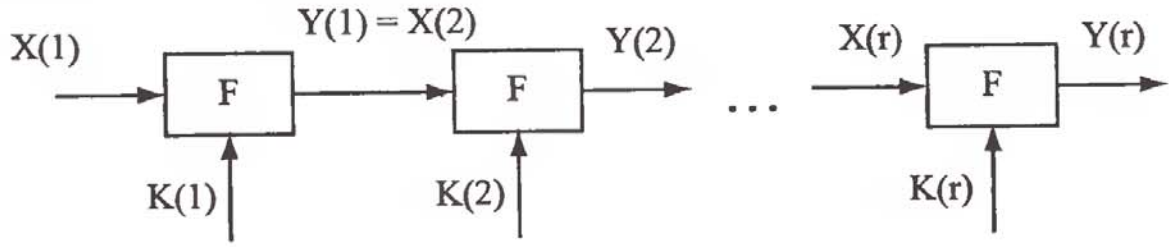


Рис. 6.1. ***Схема блочного шифратора***

где K = (K(1), K(2), …, K(r)) получается по некоторой схеме из K_0 или независимо и равновероятно выбирается для каждого цикла. Пусть X(1) и X′(1) — пара открытых текстов. Рассмотрим следующие разности:

$$\Delta X(1) = X(1) — X'(1);$$

$$\Delta Y(i) = Y(i) — Y'(i).$$

Задача дифференциального криптоанализа заключается в том, чтобы найти такие ΔX(1), что при случайном равновероятном выборе X(1), K(1), K(2), … …, K(r – 1) с вероятностью более $1/(2^N)$ появится ΔY(r – 1).

Пара (α, β) возможных значений вектора (ΔX(1), ΔY(i)) называется дифференциалом I-го цикла [4].

Тогда дифференциальный анализ описывается моделью, представленной на рис. 6.2.

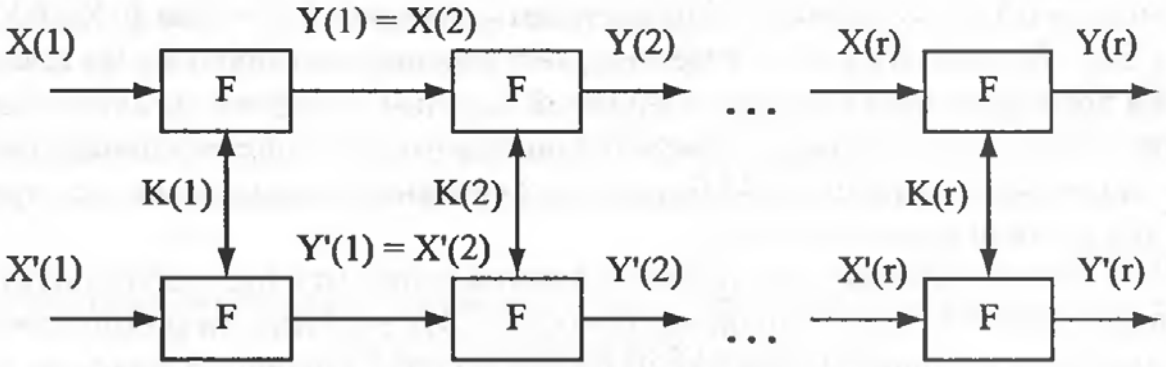


Рис. 6.2. ***Модель дифференциального криптоанализа***

Подключ последнего цикла шифрования можно найти, используя следующий алгоритм:

1. Выбираем дифференциал (r – 1)-го цикла (α, β), для которого вероятность $P(\Delta X(1) = \alpha, \Delta Y(r-1) = \beta)$ большая;
2. Случайно выбираем X(1) и подбираем X′(1) так, чтобы $\Delta X(1) = \alpha$. Пусть известны Y(r) и Y′(r);
3. Делаем предположение, что $\Delta Y(r-1) = \beta$, и, зная Y(r) и Y′(r), находим K(r);
4. Повторяем пп. 2 и 3, пока один подключ не начнет появляться чаще других.

Несмотря на кажущуюся простоту приведенного алгоритма, существует ряд проблем его реализации. Так, например, при атаке на алгоритм шифрования DES для хранения вероятностей 2^{48} возможных ключей необходимо использовать счетчики, и к тому же для анализа понадобится слишком много данных, зашифрованных на одном и том же значении секретного ключа.

6.2. Дифференциальный криптоанализ алгоритма шифрования DES

6.2.1. Дифференциальный криптоанализ одного раунда алгоритма шифрования DES

Рассмотрим один раунд криптографического преобразования DES (см. рис. 6.3).

Пусть задана пара входов X и X′, с несходством ΔX. Выходы Y и Y′ известны, следовательно, известно и несходство ΔY. Известны перестановка с расширением E и P-блок, следовательно, известны ΔA и ΔC. Значения на выходе сум-

матора по модулю 2 неизвестны, однако их несходство ΔB известно и равно ΔA. Доказано [3], что для любого заданного ΔA не все значения ΔC равновероятны. Комбинация ΔA и ΔC позволяет предположить значения битов для $E(X) \oplus K_i$ и $E(X') \oplus K_i$. То что E(X) и E(X′) известны, дает нам информацию о K_i. На каждом цикле в преобразовании участвует 48-битный подключ исходного 56-битного секретного ключа. Таким образом, раскрытие циклового ключа шестнадцатого раунда K_{16} позволяет восстановить 48 бит ключа. Остальные восемь можно восстановить при помощи силовой атаки.

Несходство различных пар открытых текстов приводит к несходству получаемых шифртекстов с определенной вероятностью. Эти вероятности можно определить, построив таблицы для каждого из блоков замены. Таблицы строятся по следующему принципу: по вертикали располагаются все возможные комбинации ΔA, по горизонтали — все возможные комбинации ΔC, а на пересечении — число соответствий данного ΔC данному ΔA.

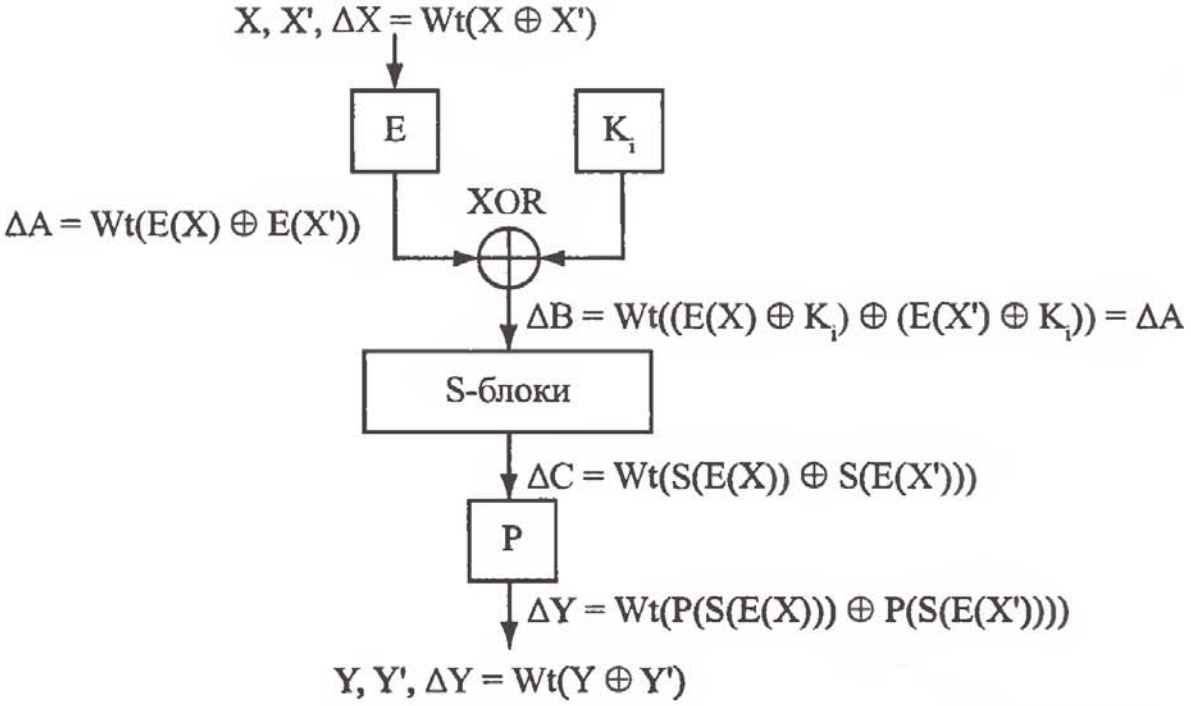


Рис. 6.3. ***Один цикл DES-преобразования для входных блоков X и X′***

Число наибольших совпадений указывает нам пару ΔA и ΔC, с помощью которой можно определить секретный ключ. Пара открытых текстов, соответствующих данным ΔA и ΔC называется **правильной парой**, а пара открытых текстов, не соответствующих данным ΔA и ΔC — **неправильной парой**. Правильная пара подскажет правильный ключ цикла, а неправильная пара — случайный. Чтобы найти правильный ключ, необходимо просто собрать достаточное число предположений. Один из подключей будет встречаться чаще, чем все остальные. Фактически, правильный подключ появляется из всех возможных случайных подключей.

В таблице 6.1 приводится обзор наилучших результатов успешного дифференциального анализа DES с различным количеством циклов [1]. Первый столбец содержит количество циклов. Два следующих столбца содержат нижнюю оценку числа выборочных или заданных (известных) открытых текстов, необходимых для осуществления атаки. Четвертый столбец содержит количество действительно проанализированных открытых текстов. В последнем столбце приведена оценка трудоемкости операций атаки после обнаружения требуемой пары.

Таблица 6.1

Обзор наилучших результатов успешного дифференциального анализа DES с различным количеством циклов

Количество циклов	Выборочные открытые тексты	Известные открытые тексты	Проанализированные открытые тексты	Трудоемкость атаки
8	2^{14}	2^{38}	4	29
9	2^{24}	2^{44}	2	2^{32}
10	2^{24}	2^{43}	2^{14}	2^{15}
11	2^{31}	2^{47}	2	32
12	2^{31}	2^{47}	2^{21}	21
13	2^{39}	2^{52}	2	2^{32}
14	2^{39}	2^{51}	2^{29}	2^{29}
15	2^{47}	2^{56}	27	2^{37}
16	2^{47}	2^{55}	2^{36}	2^{37}

Итак, анализ начинается с изучения блоков замены, то есть S-блоков, и построения таблиц зависимости значений ΔC от значений ΔA. В алгоритме шифрования DES каждому из 64 входов в S-блок строго определен соответствующий выход. Таким образом, у нас может быть 64 возможных значения ΔA, каждое из которых может быть получено 64 возможными способами. Например, $\Delta A = 000001 = 000111 \oplus 000110$ или $\Delta A = 000001 = 000001 \oplus 000000$ и т. д. Если мы будем рассматривать блок замены S_1, то увидим, что входу 000111 соответствует выход 0100, а входу 000110 — выход 0001 (см. в Приложении 1 таблицу 1). Таким образом, значению $\Delta A = 000001$, определенному в нашем примере первым, соответствует $\Delta C = 0100 \oplus 0001 = 0101$. Если мы проанализируем второе значение ΔA нашего примера, то увидим, что для него $\Delta C = 0000 \oplus 1110$. Таким образом, анализируя входы и выходы S-блоков, мы получаем таблицы (см. в Приложении 1 таблицы 10–18), позволяющие нам определить правильные пары (ΔA, ΔC), а также соответствующие им вероятности. Значение вероятности получается делением количества соответствий данного значения ΔC на возможное количество значений (то есть для алгоритма шифрования DES на 64).

Обратите внимание, что в таблице 10, приведенной в Приложении 1, отсутствует строчка, соответствующая значению $\Delta A = 000000$. Это связано с тем, что $\Delta A = 0$ может быть получено в том случае, если по модулю два складываются два одинаковых числа. А так как одинаковые входы S-блока естественно имеют одинаковые выходы, то сумма по модулю два одинаковых выходов всегда будет равна 0. Таким образом, 64 значениям $\Delta A = 000000$ всегда соответствуют 64 значения $\Delta C = 0000$. И пара (000000, 0000) имеет вероятность $p = 1$.

Анализируя дальше полученные таблицы, можно увидеть некоторые закономерности в их построении, которые могут помочь в дальнейшем:

1. Сумма всех значений одной горизонтальной линии, то есть количества различных значений ΔC, соответствующих одному и тому же значению ΔA, всегда равна 2^n, где n — размерность вектора, входящего в S-блок, то есть для DES — 2^6;
2. Для алгоритма DES не встречается ни одной пары (ΔA, ΔC), вероятность которой превышала бы 1/4;
3. Количество соответствий данного значения ΔC данному значению ΔA всегда четное. Это связано с тем, что $\Delta A = X \oplus X' = X' \oplus X$, то есть одно и то же значение ΔA получается дважды одной и той же парой входов, меняется только порядок поступления значений на вход.

Теперь, когда мы знаем основные свойства таблицы, мы можем рассмотреть несколько базовых приемов, на основе которых и строится дифференциальный криптоанализ. Прежде всего, необходимо отметить, что, как правило, при рассмотрении дифференциального криптоанализа начальная и конечная перестановки, которые присущи некоторым алгоритмам блочного шифрования, не рассматриваются. Это связано с тем, что они практически не влияют на криптографическую стойкость, а если мы будем учитывать еще и их, то просто легко запутаемся. На рис. 6.4 показан один раунд шифрования. При этом правая часть используемой разности P_R равна 0. Как мы уже отметили раньше, в этом случае разность на выходе F-блока всегда будет равна 0, а значит, на выходе одного раунда шифрования мы получим ту же самую разность, что и на входе.

Как уже отмечалось выше, нам необходимо выбрать правильные пары (ΔA, ΔC). Для алгоритма шифрования DES таких пар несколько. Все они имеют вероятность 1/4. Рассмотрим одну из них, полученную путем анализа блока замены S_2 (таблицы анализа всех блоков замены алгоритма шифрования DES приведены в Приложении 1). Это пара $(001000, 1010)_{bin} = (8, 10)_{hex}$. Если мы будем использовать эту характеристику для второго блока замены, а для остальных блоков замены — нулевые характеристики, то в результате получим разность, показанную на рис. 6.5. При этом обратите внимание, что входная разность P_R определена таким образом, чтобы, пройдя перестановку с расширением, на входе второго блока замены S_2 оказалось значение $\Delta A = 001000$.

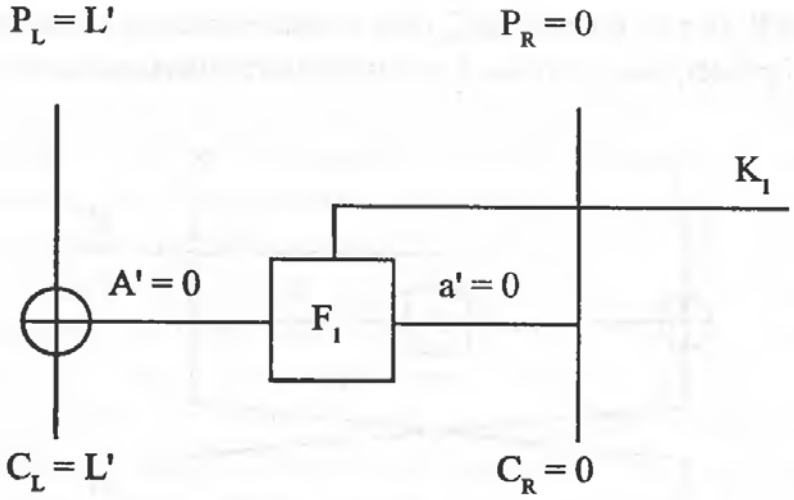


Рис. 6.4. ***Однораундовая характеристика***

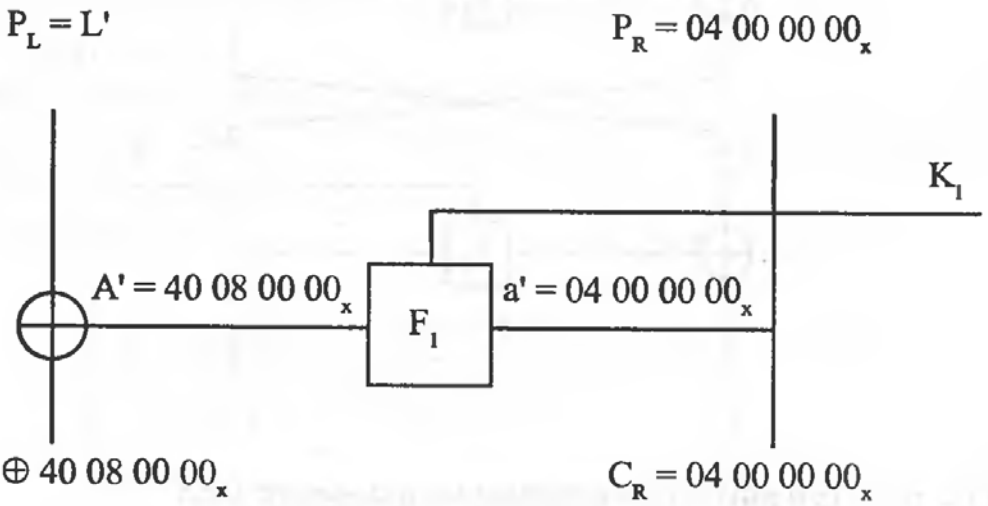


Рис. 6.5. ***Однораундовая характеристика алгоритма шифрования DES***

Ниже мы покажем, как характеристики одного раунда будут объединяться друг с другом при построении многораундовых характеристик.

6.2.2. Дифференциальный криптоанализ трех раундов алгоритма шифрования DES

Теперь перейдем к рассмотрению более сложного вопроса — анализу алгоритма, состоящего более чем из одного раунда. На рис. 6.6 показана общая схема алгоритма шифрования DES, состоящего из трех раундов, на основании которой проведем наш анализ. Пусть правая часть разности открытых текстов P_R равна 0 ($a = 0$). В этом случае выход F-блока первого раунда тоже будет равен нулю ($A = 0$). Тогда на вход второго раунда поступит левая часть P_L исходной разности ($b = P_L$). Левая часть выходной разности C_L получается сложением по модулю два выхода C F-функции третьего раунда и входа второго раунда, который, как мы уже определили раньше, равен левой части исходной разности. Таким образом, полу-

чается, что $C_L = P_L \oplus C$. То что P_L и C_L известны, позволяет нам определить значение C. Таким образом, мы легко можем определить биты подключа третьего раунда шифрования.

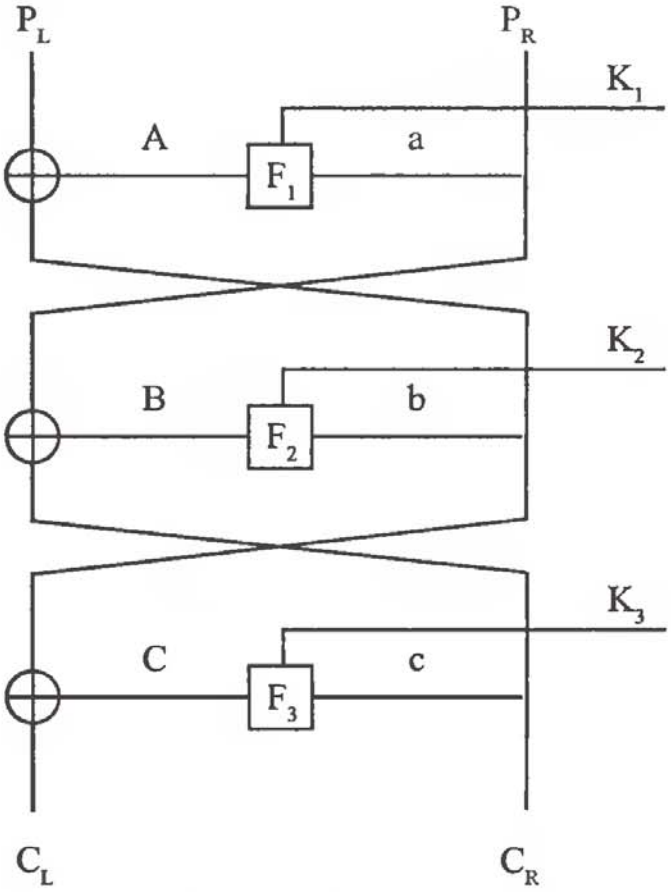


Рис. 6.6. ***Три раунда алгоритма шифрования DES***

Существует еще один прием, позволяющий анализировать трехраундовый алгоритм шифрования. Входную разность необходимо подобрать таким образом, чтобы выход F-функции первого раунда шифрования совпал с левой частью входной разности. В этом случае на вход второго раунда шифрования поступит разность, равная 0. И как следствие, на выходе F-функции второго раунда шифрования всегда будет разность, равная 0. Таким образом, на вход третьего раунда поступит та же самая разность, что и на вход первого раунда, то есть правая часть исходной входной разности. В этом случае правильный подбор пар *открытый—закрытый текст* обеспечит правильное нахождение битов ключа.

Бихам и Шамир предложили использовать входную разность, равную 60 00 00 00x. Эта разность после прохождения таблицы перестановки с расширением обеспечит поступление нулевых разностей во все блоки замены, кроме первого. На вход первого блока замены S_1 поступит разность $\Delta A = 001100$. С вероятностью $p = 14/64$ ей будет соответствовать разность $\Delta C = 0110$ (это видно из таблицы 10, приведенной в Приложении 1 — на пересечении 001100 строки и 0110 столбца стоит значение 14. Это говорит о том, что при входной разности

$\Delta A = 001100$ выходная разность $\Delta C = 0110$ появляется в 14 из 64 случаев). С учетом того, что на выходе остальных блоков замены будут находиться разности, равные 0, а также, учитывая таблицу перестановки, выходная разность раунда будет равна 00 80 82 00x (см. рис. 6.7). Как видно, Бихам и Шамир предложили использовать пару, не имеющую максимально возможную вероятность. Видимо, это связано с тем, что при рассмотрении правильных пар необходимо также учитывать, сколько S-блоков будут участвовать в анализе. Если бы мы взяли к рассмотрению правильную пару для блока замены S_1 (110100, 0010), то вынуждены были бы рассматривать еще и последний блок замены, так как, согласно таблице перестановки с расширением, 32-й бит исходного сообщения является первым и пятым входным битом соответственно первого и восьмого блоков замены. При этом вероятность p определялась бы как произведение вероятностей для пары первого блока замены и вероятности для пары восьмого блока замены, в результате чего итоговое значение вероятности оказалось гораздо меньше, чем для пары, рассмотренной Бихамом и Шамиром.

Некоторые особенности применения дифференциального криптоанализа будут рассмотрены также в главе 7 при изучении метода линейно-дифференциального криптоанализа.

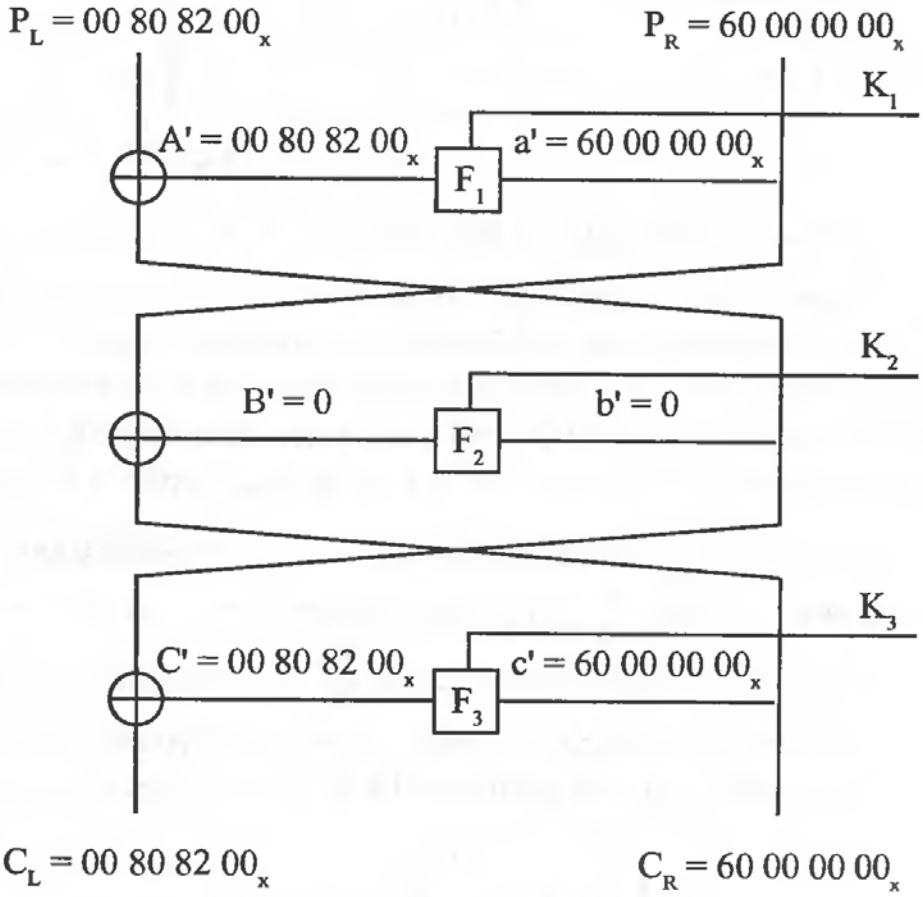


Рис. 6.7. ***Трехраундовая характеристика для алгоритма шифрования DES***

6.2.5. Дифференциальный криптоанализ полного алгоритма шифрования DES

Для проведения атаки на полный 16-раундовый алгоритм шифрования Бихам и Шамир предложили использовать в одном раунде такую разность, которая бы при прохождении F-функции давала на выходе нулевую разность. В этом случае можно было бы строить двухраундовые характеристики, как это показано на рис. 6.8.

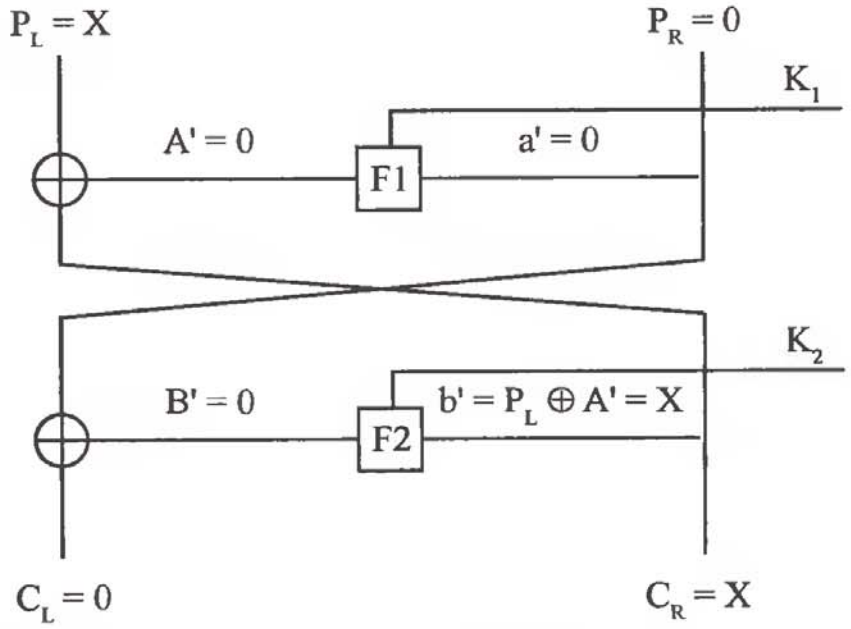


Рис. 6.8. ***Пример двухраундовой характеристики***

Один из возможных вариантов такой входной разности — 19 60 00 00x. В этом случае после прохождения таблицы перестановки с расширением на вход первого S-блока попадет значение разности 000011, на вход второго S-блока — 110010, на вход третьего — 101100, а на вход всех остальных блоков замены — нули. Вероятность того, что в этом случае на выходе первого S-блока появится нулевая разность, равна $\frac{14}{64}$; вероятность того, что на выходе второго S-блока будет нулевая разность, равна $\frac{8}{64}$, и наконец, вероятность того, что на выходе третьего S-блока будет нулевая разность, равна $\frac{10}{64}$ (см. таблицы 10, 11 и 12 в Приложении 1). Объединяя все вышесказанное, получаем, что вероятность того, что при поступлении на вход раунда разности 19 60 00 00, на выходе появится нулевая разность, равна:

$$P = \frac{14}{64} * \frac{8}{64} * \frac{10}{64} = \frac{35}{8192} = \frac{1}{234} \approx \frac{1}{2^{7.87}}.$$

При рассмотрении полного алгоритма шифрования DES Бихам и Шамир предложили использовать двухраундовую характеристику 6 раз (со 2-го по 13 раунды). Вероятность шести двухраундовых характеристик будет равна:

$$P = \left(\frac{1}{2^{7.87}}\right)^6 = \left(\frac{1}{2^{47.22}}\right) = 2^{-47.22}.$$

В этом случае на вход 14-го раунда поступит нулевая разность (см. рис. 6.9), и с вероятностью p = 1 даст на выходе тоже нулевую разность. Поэтому четырнадцатый цикл мы не учитываем. Рассматриваемая нами пара входной и выходной разности имеет максимальную вероятность из всех возможных пар *входная—выходная разность* для полного алгоритма DES. Поэтому мы можем ее использовать для нахождения битов ключа.

Так как нам известна выходная разность 16-го раунда, и входная разность 14-го раунда равна нулю, то на выходе F-функции 15-го раунда появится разность, равная правой части выходной разности 16-го раунда. Разность на входе 15-го раунда нам известна, а значит, мы легко можем определить разность на выходе F-функции 16-го раунда шифрования. Таким образом, использование двухраундовой характеристики со 2 по 13 раунд и точное знание выходной разности 16-раунда шифрования позволяют нам исключить три последних раунда шифрования из учета общей вероятности характеристики. Последние три раунда будут выполняться всегда с вероятностью p = 1.

Осталось разобраться с первым раундом шифрования. Здесь Бихамом и Шамиром были предприняты некоторые математические приемы и вычисления, которые позволили также исключить первый раунд из вычисления общей вероятности характеристики. Мы не будем подробно останавливаться на этих приемах. Однако общее направление размышлений все-таки обозначим.

Во-первых, необходимо учитывать тот факт, что на выходе первого раунда ненулевые биты могут присутствовать в разности только лишь с первого по двенадцатый. (3 выхода S-блоков по 4 бита.) Таким образом, у нас может быть всего 2^{12} выходных разностей, при этом каждая 12-битовая разность может быть получена одним способом из 2^{12}, то есть всего возможно $(2^{12})^2 = 2^{24}$ комбинаций входных пар, и вероятность появления нужной разности равна $p = \frac{1}{2^{12}}$.

Также при выборе правильных пар необходимо учитывать, что на выходе F-функции 16-го раунда последние пять S-блоков дают нулевую разность. Таким образом, если у нас есть 2^{24} возможных пар текстов, и при этом 2^{20} из них могут иметь ненулевые последние 20 бит (выходы 5 последних S-блоков по 4 бита), то у нас остается примерно $2^4 = 16$ возможных пар текстов.

Если внимательно изучить таблицы 10, 11, 12, приведенные в Приложении 1, то можно увидеть, что в некоторых случаях для данной входной разности некоторые выходные разности никогда не могут появиться.

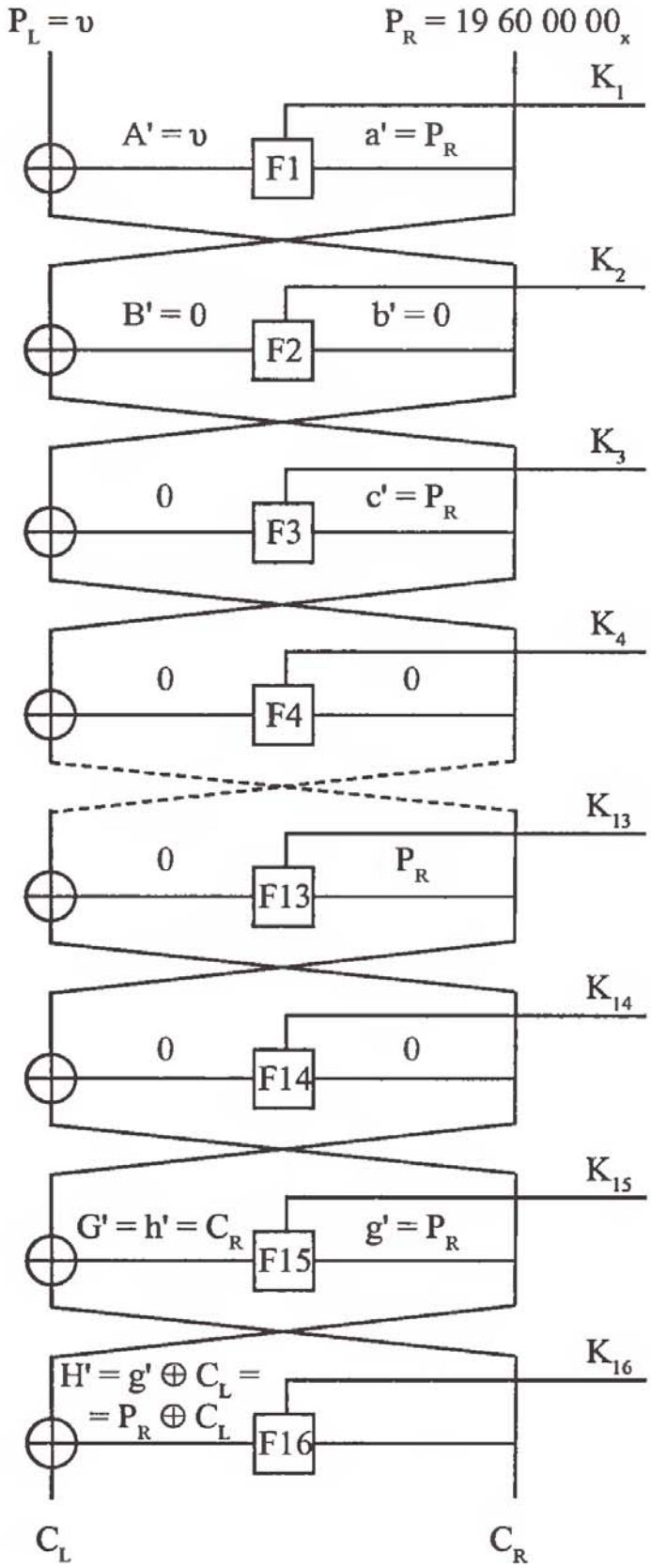


Рис. 6.9. ***Применение дифференциального криптоанализа к 16 раундам алгоритма шифрования DES***

Учитывая все вышесказанное, Бихам и Шамир показали, что возможно максимально сузить круг поиска правильных пар за счет анализа первого раунда шифрования. При этом правильное значение ключа можно будет определить, как только будут найдены несколько правильных пар. Более подробную информацию об атаке на полный алгоритм шифрования DES можно найти в [7, 8].

6.3. Дифференциальный криптоанализ шифровальной сети на основе подстановок и перемешивания

Шифровальная сеть на основе подстановок и перемешивания была рассмотрена при изучении линейного криптоанализа. На основе этой же сети мы рассмотрим применение дифференциального криптоанализа. Для начала мы определим пару разностей для S-блока. Рассмотрим 4 × 4 S-блок, изображенный на рис. 6.10, имеющий входной $X = [X_1\ X_2\ X_3\ X_4]$ и выходной $Y = [Y_1\ Y_2\ Y_3\ Y_4]$ векторы. Все пары разностей S-блока, $(\Delta X, \Delta Y)$, могут быть протестированы, и вероятность того, что ΔY обусловлена ΔX, может быть получена путем рассмотрения входных пар (X', X''), таких что $X' \oplus X'' = \Delta X$. Так как вид пар не имеет значения, то для 4 × 4 S-блока нам необходимо рассмотреть только 16 значений для X' и затем значение ΔX, составленное с помощью такого X'', что $X'' = \Delta X \oplus X'$.

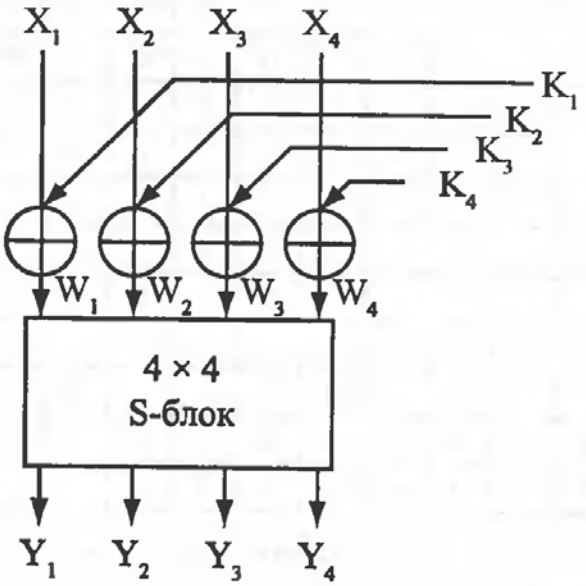


Рис. 6.10. ***S-блок***

Рассматривая S-блок нашего алгоритма шифрования, приведенный в таблице 5.5, мы можем получить результирующие значения ΔY для каждой входной пары $(X', X'' = \Delta X \oplus X')$. Мы можем занести все значения для S-блока в таблицу 6.2 распределения дифференциалов, в которой ряды представляют собой значения ΔX (в шестнадцатеричной системе счисления), а столбцы — значения ΔY (также в шестнадцатеричной системе счисления). Каждый элемент таблицы пред-

ставляет собой количество случаев соответствия значения выходной разности ΔY данному входному значению разности ΔX. Заметим, что кроме особенного случая, когда ΔY = 0 и ΔX = 0, самое большое значение в таблице 8 соответствует ΔX = B и ΔY = 2. Следовательно, вероятность того, что ΔY = 2 в случайной паре выходных значений, которым соответствует ΔX = B, равна 8/16. Самое малое значение в таблице равно 0 и соответствует многим разностным парам. В этом случае вероятность того, что значение ΔY соответствует данному значению ΔX равна 0.

Таблица 6.2

Таблица распределения разностей

Выходные разностн / Входные разности	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Прежде чем приступить к обсуждению построения дифференциальных пар S-блока для получения дифференциальных характеристик и определения правильной пары для использования в атаке, обсудим влияние ключа на дифференциал S-блока. Рассмотрим рис. 6.10. Входом в S-блок без учета ключа является X, а выходом — Y. Однако в алгоритме шифрования мы должны рассматривать ключи, складываемые с входами каждого из S-блоков. В этом случае, если мы обозначить входы в S-блок, сложенные с ключом, как $W = [W_1\ W_2\ W_3\ W_4]$, то можно рассматривать входную разность в S-блок как

$$\Delta W = [W_1' \oplus W_1''\ W_2' \oplus W_2''\ \ldots\ W_n' \oplus W_n''],$$

где $W' = [W_1' W_2' \ldots W_n']$ и $W'' = [W_1'' W_2'' \ldots W_n'']$ обозначают два входных значения.

Так как биты ключа одинаковы как для W', так и для W'', то

$$\Delta W_i = W_i' \oplus W_i'' = (X_i' \oplus K_i) \oplus (X_i'' \oplus K_i) = X_i' \oplus X_i'' = \Delta X_i,$$

Итак, биты ключа не влияют на значение входной разности, а значит, их можно игнорировать. Другими словами, таблицы будут одинаковы как с учетом использования ключа, так и без него.

6.3.1. Нахождение разностной характеристики

Как только информация о разностях для S-блока алгоритма шифрования SPN собрана, мы можем приступить к нахождению полезных дифференциальных характеристик для полного алгоритма шифрования. Если построить такую дифференциальную характеристику, которая будет состоять из входной разности алгоритма шифрования, преобразующейся при прохождении раундовых блоков замены (S-блоков) в разность на входе последнего раунда алгоритма шифрования, то станет возможным провести анализ и найти некоторые биты секретного подключа, используемого в последнем раунде. Мы продемонстрируем построение дифференциальных характеристик на примере.

Рассмотрим дифференциальную характеристику, включающую S_{12}, S_{23}, S_{32} и S_{33}. Как и в случае линейного криптоанализа, полезно проиллюстрировать дифференциальную характеристику для данного алгоритма шифрования с использованием рис. 6.11. С помощью выделенных линий на этом рисунке показаны пути преобразования ненулевых битов входной разности ΔX при прохождении через S-блоки. Для извлечения битов последнего подключа в последнем разделе необходимо определить дифференциальную характеристику для первых трех раундов шифрования.

Будем использовать следующие пары разностей для S-блоков:

S_{12}: $\Delta X = B \rightarrow \Delta Y = 2$ с вероятностью 8/16,
S_{23}: $\Delta X = 4 \rightarrow \Delta Y = 6$ с вероятностью 6/16,
S_{32}: $\Delta X = 2 \rightarrow \Delta Y = 5$ с вероятностью 6/16,
S_{33}: $\Delta X = 2 \rightarrow \Delta Y = 5$ с вероятностью 6/16.

Все остальные блоки будут иметь нулевые входные разности и соответствующие им нулевые выходные разности.

Входная разность всего алгоритм шифрования аналогична входной разности первого цикла и равна

$$\Delta X = \Delta U_1 = [0000\ 1011\ 0000\ 0000],$$

где U_i — вход i-го цикла и V_i — выход i-го цикла S-блоков. Следовательно, ΔU_i и ΔV_i обозначают соответствующие разности. В результате:

$$\Delta V_1 = [0000\ 0010\ 0000\ 0000].$$

Рассматривая разностную пару для S_{12}, приведенную выше и следующую в первом раунде перестановку, получаем:

$$\Delta U_2 = [0000\ 0000\ 0100\ 0000]$$

с вероятностью 8/16 = 1/2 для входной разности ΔX.

ΔX = [0000 1011 0000 0000]

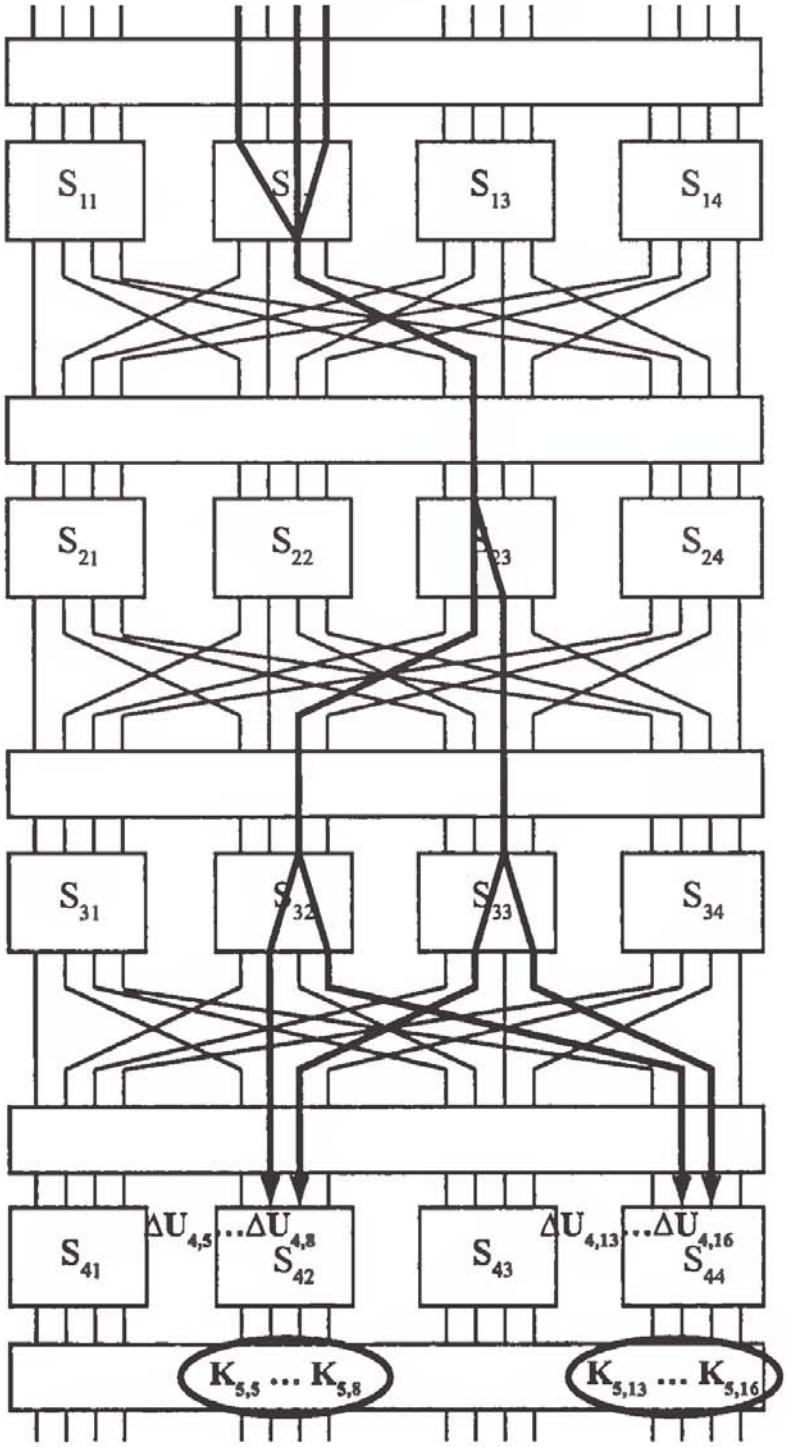


Рис. 6.11. ***Пример дифференциальной характеристики***

Теперь дифференциал второго цикла использует разностную пару для S_{23}, в результате имеем:

$$\Delta V_2 = [0000\ 0000\ 0110\ 0000],$$

и перестановка второго цикла дает:

$$\Delta U_3 = [0000\ 0010\ 0010\ 0000]$$

с вероятностью 6/16 для ΔU_2 и вероятностью $8/16 \times 6/16 = 3/16$ для ΔX. При определении вероятности для разности исходных открытых текстов ΔX мы предположили, что дифференциал первого цикла не зависит от дифференциала второго цикла, а значит, можно использовать и определять вероятность как произведение вероятностей.

Следовательно, мы можем использовать разности для S_{32} и S_{33} блоков третьего цикла, и с учетом перестановки третьего цикла записать как

$$\Delta V_3 = [0000\ 0101\ 0101\ 0000]$$

и

$$\Delta U_4 = [0000\ 0110\ 0000\ 0110] \qquad (6.1)$$

с вероятностью $(6/16)^2$ для ΔU_3 и, значит, с вероятностью $8/16 \times 6/16 \times (6/16)^2 = 27/1024$ для разности исходных открытых текстов ΔX, где, опять же, можем предположить независимость между разностными парами S-блоков во всех циклах и воспользоваться этим.

В процессе криптоанализа существуют многие пары открытых текстов (около 10000, а если надо, то и больше), для которых $\Delta X = [0000\ 1011\ 0000\ 0000]$ будут зашифрованы. С большой вероятностью, 27/1024, эта разностная характеристика будет соответствовать действительности, как будет показано в п. 6.3.2.

6.3.2. Извлечение битов ключа

Так как разностная характеристика R – 1 цикла найдена для алгоритма шифрования, содержащего R циклов, с подходящей довольно большой вероятностью, становится возможным подвергнуть алгоритм шифрования криптоанализу с помощью нахождения битов последнего подключа. В нашем случае возможно извлечь биты из подключа K_5. Последующий процесс включает частичное дешифрование последнего цикла алгоритма шифрования и тестирование последнего цикла для выявления, правильная ли это пара. Мы назовем биты подключа, складываемые в последнем цикле шифрования с выходами S-блоков, на которые влияют ненулевые разности выходного дифференциала, искомым частичным подключом. Частичная дешифрация последнего цикла шифрования будет включать (для всех S-блоков последнего цикла, на которые влияют ненулевые разности выходного дифференциала) сложение по модулю 2 шифртекста с битами искомого частичного подключа и прохождение данных в обратном порядке через S-блоки, где будут проанализированы всевозможные значения искомого частичного подключа.

Алгоритм нахождения битов подключа может выглядеть следующим образом. Частичная дешифрация выполняется для каждой пары шифртекстов, соответствующих парам исходных открытых текстов, используемых для нахождения выходной разности ΔX для всех возможных значений искомого частичного подключа. Для каждого значения искомого частичного подключа необходимо выделить счетчик. Значение счетчика увеличивается на 1 тогда, когда разность частично дешифрованных текстов совпадет со входной разностью последнего цикла шифрования. Значение искомого частичного подключа, счетчик которого имеет наибольшее значение, предположительно определяет правильное значение битов подключа. Это получается за счет предположения о том, что правильное значение частичного подключа будет получено в разности последнего цикла так часто, как это соответствует статистической характеристике (например, появление правильных пар). Когда появляется неправильная пара, даже при частичной дешифрации с правильным подключом, счетчик правильного подключа, вероятно, не будет увеличен.

Рассматривая атаку на примере нашего алгоритма шифрования, видим, что разностная характеристика влияет на входы S-блоков S_{42} и S_{44} в последнем цикле шифрования. Для каждой пары шифртекстов мы будем пробовать все 256 возможных значений для $[K_{5,5}\ldots K_{5,8}, K_{5,13}\ldots K_{5,16}]$. Для каждого значения искомого частичного подключа мы будем увеличивать значение счетчика на 1 в том случае, когда входная разность последнего цикла, определяемая с помощью частичной дешифрации, соответствует (6.1), где мы определяем значение $[\Delta U_{4,5}\ldots \Delta U_{4,8}, \Delta U_{4,13}\ldots \Delta U_{4,16}]$, пропуская данные в обратном порядке через искомый частичный подключ и S-блоки S_{24} и S_{44}. Для каждого значения частичного подключа значение счетчика определяет количество совпадений разностей, составляющих правильные пары, которые предположительно определяют правильный частичный подключ. Счетчик, имеющий наибольшее значение, предположительно определяет правильное значение частичного подключа, так как связан статистически с правильными парами, имеющими наибольшую вероятность возникновения.

Заметим, что нет необходимости для частичной дешифрации каждой пары шифртекстов. Так как входная разность последнего цикла затрагивает блоки S_{42} и S_{44}, где рассматриваются характеристики, и разности битов шифртекстов, соответствующих блокам S_{41} и S_{43}, должны быть равны нулю, то можно отбросить множество неправильных пар шифртекстов, для которых нули не появляются в разностях шифртекстов соответствующих подблоков. Эти пары шифртекстов не могут соответствовать правильной паре, и нет необходимости тестировать $[\Delta U_{4,5}\ldots \Delta U_{4,8}, \Delta U_{4,13}\ldots \Delta U_{4,16}]$.

Мы провели численный эксперимент, сымитировав атаку на наш основной алгоритм, с использованием случайным образом сгенерированных подключей. При этом были сгенерированы 5000 пар *открытый—закрытый текст* (то есть всего 10000 зашифрованных открытых текстов, удовлетворяющих

ΔX = [0000 1011 0000 0000]). Правильным значением искомого частичного подключа было значение $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}] = [0010, 0100] = [2,4]_{hex}$. Как и ожидалось, счетчик, имеющий наибольшее значение, указал на значение искомого частичного подключа $[2,4]_{hex}$, доказывая работоспособность предложенного алгоритма по определению битов подключа. Значения таблицы 6.3 показывают значения вероятностей появления правильных пар для рассматриваемого искомого частичного подключа, вычисленные по следующей формуле:

Вероятность = значение счетчика / 5000,

где значение счетчика — это значение счетчика, соответствующее значению данного частичного подключа. Полный набор данных включает 256 входных данных для каждого значения искомого частичного подключа.

Таблица 6.3

Экспериментальные результаты атаки на основе дифференциального криптоанализа

Частичный подключ $[K_5,5...K_5,8, K_5,13...K_5,16]$	вероятность	Частичный подключ $[K_5,5...K_5,8, K_5,13...K_5,16]$	вероятность
1 C	0.0000	2 A	0.0032
1 D	0.0000	2 B	0.0022
1 E	0.0000	2 C	0.0000
1 F	0.0000	2 E	0.0000
2 0	0.0000	2 F	0.0000
2 1	0.0136	3 0	0.0000
2 2	0.0068	3 1	0.0004
2 3	0.0068	3 2	0.0000
2 4	**0.0244**	3 3	0.0004
2 5	0.0000	3 4	0.0004
2 6	0.0068	3 5	0.0000
2 7	0.0068	3 6	0.0004
2 8	0.0030	3 7	0.0000
2 9	0.0024	3 8	0.0008

6.4. Дифференциальный криптоанализ других алгоритмов блочного шифрования

6.4.1. Дифференциальный криптоанализ алгоритма шифрования LOKI

Алгоритм шифрования LOKI был описан в п. 4.4. Здесь мы рассмотрим некоторые его особенности, которые могут помочь при более детальном анализе алгоритма.

Пары (ΔA, ΔC) в таблице распределения вероятностей для блока замены S имеют значения, которые значительно ниже любой вероятности для пар алгоритма шифрования DES (в среднем значение вероятности равно 1/256, максимальное значение вероятности — 1/64). Однако существует возможность использовать такую входную разность функции F, чтобы были задействованы всего два S-блока, и при этом на выходе получалась нулевая разность. В DES аналогичный результат достигается за счет использования как минимум трех блоков замены. Бихам и Шамир нашли двухраундовую характеристику для алгоритма шифрования LOKI, показанную на рис. 6.12 (под характеристикой подразумевается пара входной и выходной разностей, имеющая определенную вероятность). Вероятность этой характеристики равна $p = \frac{32}{2^{12}} * \frac{14}{2^{12}} * \frac{18}{2^{12}} * 1 * \frac{18}{2^{12}} * \frac{14}{2^{12}} * \frac{32}{2^{12}} * 1 = \frac{118}{2^{20}} \approx 2^{-13.12}$ (вероятность рассчитана с учетом того, что два соседних S-блока имеют четыре общих входных бита, иначе вероятность была бы намного меньше).

С учетом полученной характеристики в [38] показана возможность провести атаку 9-раундового алгоритма LOKI с вероятностью $2^{-52.5}$ и 11-раундового алгоритма LOKI, с вероятностью примерно равной $2^{-65.5}$.

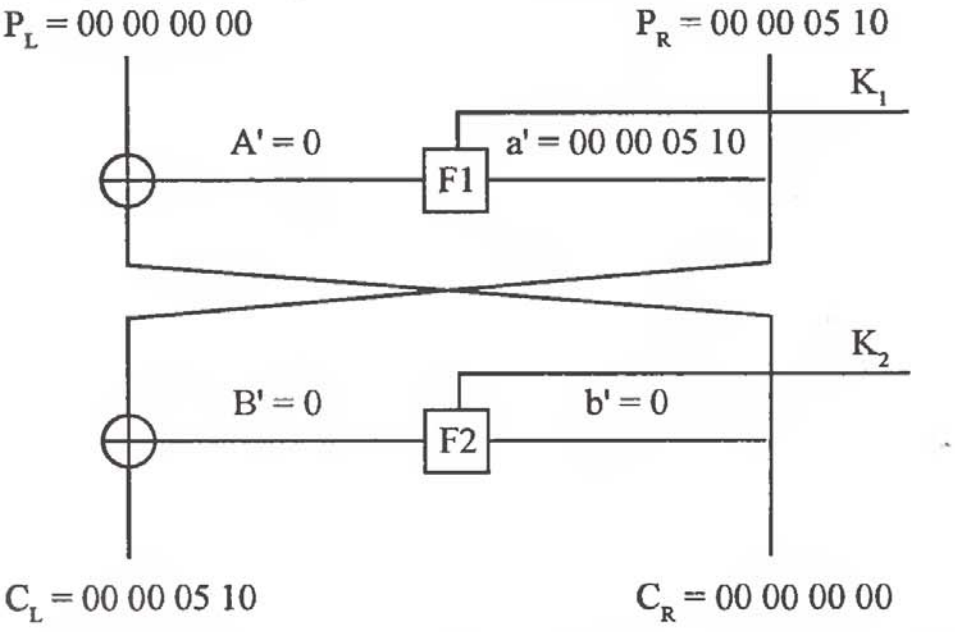


Рис. 6.12. ***Двухраундовая характеристика алгоритма шифрования LOKI***

Существует еще одна возможность проведения дифференциального криптоанализа для 8 раундов шифрования, как показано на рис. 6.13. Здесь используются биты, которые после прохождения перестановки с расширением, встречаются только один раз. При этом выходные разности отличаются только одним битом. Вероятность данной характеристики равна приблизительно 2^{-46}. Используя эту характеристику, возможно взломать алгоритм шифрования LOKI с одиннадцатью циклами при наличие меньше, чем 264 выбранных или известных открытых текстов.

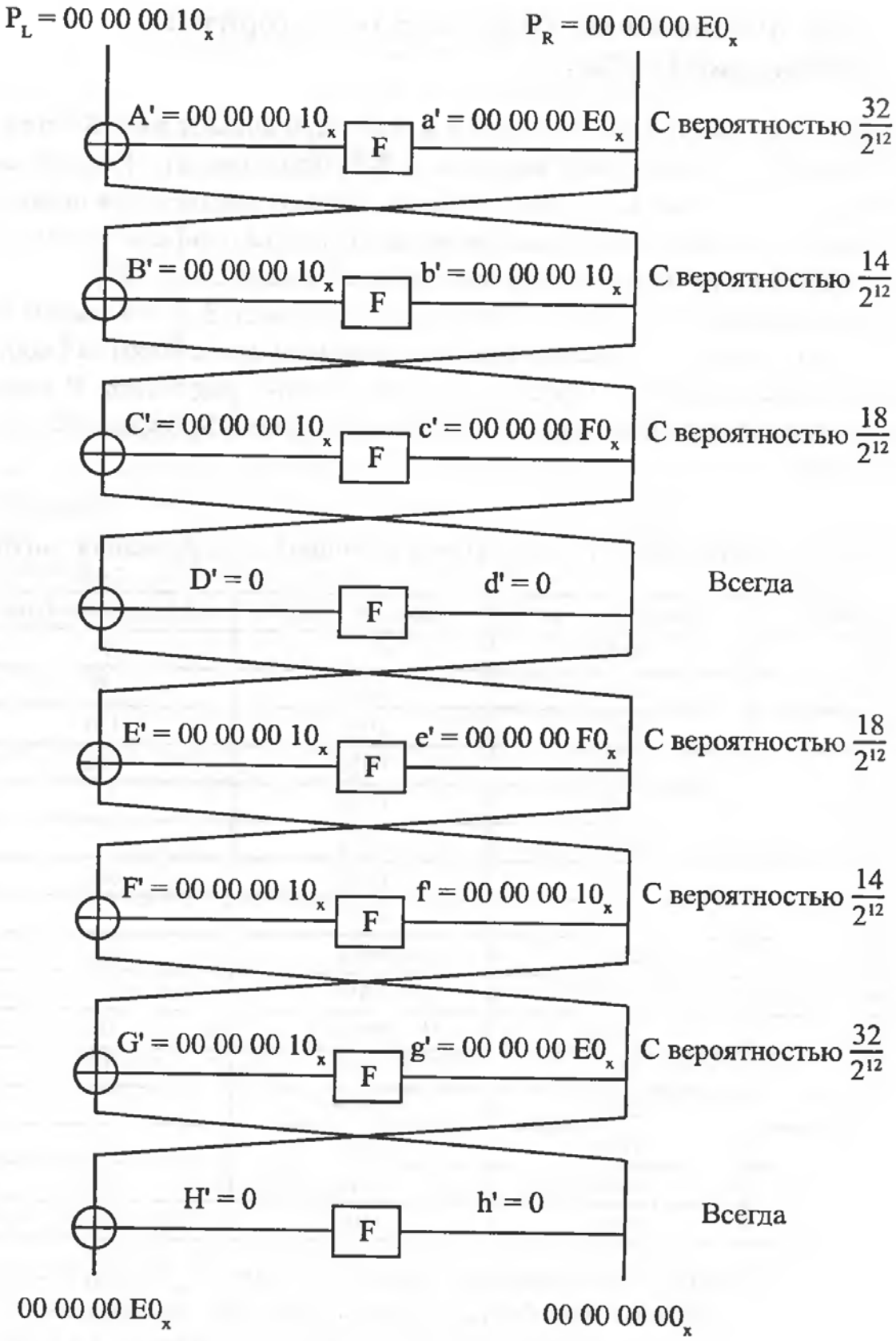


Рис. 6.13. ***Применение дифференциального криптоанализа к 8 раундам алгоритма шифрования LOKI***

6.4.2. Дифференциальный криптоанализ алгоритма шифрования Lucifer

Алгоритм шифрования Lucifer описан в п. 4.20. При анализе вход S-блока и соответствующие ему выходы двух возможных S-блоков известны. Каждый выходной бит двух этих блоков может как совпадать, так и отличаться. Как правило, на выходе разных S-блоков при одинаковом входе совпадают один или два бита. В некоторых случаях встречаются совпадения в трех выходных битах [27].

Для примера возьмем третью и четвертую строку первого S-блока замены алгоритма DES (см. таблицу 3.6) в качестве S_0 и S_1 блоков замены алгоритма Lucifer, соответственно. Другие выборы S-блоков дают аналогичные результаты. В таблице 6.4 показаны выходные биты, одинаковые как для первого S_0 блока, так и для второго S_1 блока.

Таблица 6.4

Совпадающие выходные биты блоков замены алгоритма шифрования Lucifer

Вход S-блока	Выход S_0 блока	Выход S_1 блока	Совпадающие биты
0000	0100	1111	.1..
0001	0001	1100	..0.
0010	1110	1000	1..0
0011	1000	0010	.0.0
0100	1101	0100	.10.
0101	0110	1001	....
0110	0010	0001	00..
0111	1011	0111	..11
1000	1111	0101	.1.1
1001	1100	1011	1...
1010	1001	0011	.0.1
1011	0111	1110	.11.
1100	0011	1010	.01.
1101	1010	0000	.0.0
1110	0101	0110	01..
1111	0000	1101	..0.

Из таблицы 6.4 видно, что одиннадцать значений входов в S-блок приведут к различию на выходе S-блоков в двух битах, четыре входа — в одном бите, и только один вход приведет к полному различию битов на выходе. Таблица 6.5 описывает входы S-блока, которые имеют одинаковый бит на выходе любого из двух блоков замены.

Таблица 6.5

Входы S-блока, имеющие одинаковый бит на выходе любого из двух блоков замены

Вход S-блока	Совпадающий выходной бит
.000	.1..
0.00	.1..
001.	...0
.110	0...
10.0	...1
110.	.0..

Пользуясь таблицами 6.4 и 6.5, мы можем подобрать множество открытых текстов так, чтобы некоторые биты во внутренних циклах были установлены в фиксированные значения, независимо от выбора ключа. Этим свойством можно воспользоваться для сокращения области поиска наиболее вероятных разностей. Применив дифференциальный криптоанализ к реализации алгоритма шифрования Lucifer с 32-битовыми блоками данных и 8 раундами, Бихам и Шамир показали, что его можно взломать с помощью 40 подобранных открытых текстов за 2^{29} операций [27]. Этот же метод позволяет вскрыть Lucifer с 128-битовыми блоками и 8 раундами с помощью 60 подобранных открытых текстов за 2^{53} шагов [27]. Другая дифференциальная атака позволяет вскрыть 18-раундовый, 128-битный Lucifer с помощью 24 подобранных открытых текстов за 2^{21} операций [27].

6.4.3. Дифференциальный криптоанализ алгоритма шифрования RC5

Описание алгоритма шифрования RC5 приведено в п. 4.1. Рассмотрим некоторые особенности алгоритма, а также основные операции, которые могут помочь в применении метода дифференциального криптоанализа к его вскрытию.

Операция сдвига, которая по существу является базовой в алгоритме RC5 и с учетом сложения по модулю 2 (XOR) обладает следующим свойством:

$$(A<<<s) \oplus (B<<<s) = (A \oplus B)<<<s, \qquad (6.2)$$

где A и B — два разных сообщения; r — число раундов, на которое необходимо произвести сдвиг.

Операция целочисленного сложения очень близка к линейным операциям. Так, разность в одном бите A и B остается неизменной с вероятностью 1/2 после операции их целочисленного сложения с одним и тем же числом S_i:

$$(A + S) \oplus (B + S), \qquad (6.3)$$

а с вероятностью 1/2 становится разностью, в которой различны два соседних бита (6.3). В случае, если разность, полученная в результате сложения по модулю

два, имеет различие в младшем значащем бите, то она всегда остается неизменной, то есть с вероятностью, равной 1.

Поясним сказанное на примере. Для этого рассмотрим 3-битовые значения A и B. Пусть A и B имеют разность во втором бите, то есть $A \oplus B = 010$. Трехбитовая разность может быть получена одним из следующих восьми вариантов:

	A	⊕	B	=	010
1.	000	⊕	010	=	010
2.	001	⊕	011	=	010
3.	010	⊕	000	=	010
4.	011	⊕	001	=	010
5.	100	⊕	110	=	010
6.	101	⊕	111	=	010
7.	110	⊕	100	=	010
8.	111	⊕	101	=	010

Теперь для каждой пары текстов (A, B) мы должны сложить каждое из значений A и B с ключом S и рассмотреть полученное значение $(A + S) \oplus (B + S)$. При этом, так как ключ является тоже трехбитовым, то он может принимать одно из восьми следующих значений:

$S_1 = 000$;	$S_5 = 100$;
$S_2 = 001$;	$S_6 = 101$;
$S_3 = 010$;	$S_7 = 110$;
$S_4 = 011$;	$S_8 = 111$;

Итак, для первого варианта получения разности $A \oplus B = 000 \oplus 010 = 010$ имеем:

$A = 000$; $B = 010$;

$A \oplus S_1 = 000 \oplus 000 = 000$; $B \oplus S_1 = 010 \oplus 000 = 010$;

$(A + S_1) \oplus (B + S_1) = 000 \oplus 010 = \mathbf{010}$;

$A \oplus S_2 = 000 \oplus 001 = 001$; $B \oplus S_2 = 010 \oplus 001 = 011$;

$(A + S_2) \oplus (B + S_2) = 001 \oplus 011 = \mathbf{010}$;

$A \oplus S_3 = 000 \oplus 010 = 010$; $B \oplus S_3 = 010 \oplus 010 = 100$;

$(A + S_3) \oplus (B + S_3) = 010 \oplus 100 = \mathbf{110}$;

$A \oplus S_4 = 000 \oplus 011 = 011$; $B \oplus S_4 = 010 \oplus 011 = 101$;

$(A + S_4) \oplus (B + S_4) = 011 \oplus 101 = \mathbf{110}$;

$A \oplus S_5 = 000 \oplus 100 = 100$; $B \oplus S_5 = 010 \oplus 100 = 110$;

$(A + S_5) \oplus (B + S_5) = 100 \oplus 110 = \mathbf{010}$;

$A \oplus S_6 = 000 \oplus 101 = 101$; $B \oplus S_6 = 010 \oplus 101 = 111$;

$(A + S_6) \oplus (B + S_6) = 101 \oplus 111 = \mathbf{010}$;

$A \oplus S_7 = 000 \oplus 110 = 110$; $B \oplus S_7 = 010 \oplus 110 = 000$;

$(A + S_7) \oplus (B + S_7) = 110 \oplus 000 = \mathbf{110}$;

$A \oplus S_8 = 000 \oplus 111 = 111$; $B \oplus S_8 = 010 \oplus 111 = 001$;

$(A + S_8) \oplus (B + S_8) = 111 \oplus 001 = \mathbf{110}$.

Как видно, в четырех случаях из восьми разность после сложения с ключом останется неизменной:

$$A \oplus B = (A + S) \oplus (B + S) = 010.$$

А в четырех случаях из восьми после сложения с ключом становятся различны два соседних бита, то есть вместо значения 010 появляется значение 110.

Анализ оставшихся семи вариантов получения разности $A \oplus B = 010$ даст аналогичный результат (в этом вы можете убедиться самостоятельно).

Как уже не раз отмечалось ранее, при рассмотрении метода дифференциального криптоанализа рассматриваются пары входов и выходов, имеющие различия битов в некоторых позициях. В случае DES-подобных алгоритмов шифрования правильные входные пары текстов возможно определить с помощью анализа блоков замены (например S-блоков в алгоритме DES). А как быть, если алгоритм не содержит блоков замены? В этом случае необходимо провести тщательный анализ всех криптографических элементов системы и определить, как та или иная операция влияет на видоизменение разности, поступающей на вход алгоритма шифрования, в процессе ее последовательного прохождения через раунды шифрования.

Рассмотрим три полураунда алгоритма шифрования RC5, показанные на рис. 6.14.

Под характеристикой будем понимать пару входной и выходной разностей, имеющую определенную вероятность. Проведем расчеты для входного блока данных, равного 2ω битам (на рис. 6.14 $\omega = 32$).

Обозначим через e_i вектор разности, состоящий из ω бит, в котором во всех позициях, кроме i-й, находятся 0 (здесь вводится новое обозначение разности e_i, в отличие от привычного обозначения Δ, для удобства оперирования разностями, имеющими единицу в единственной позиции), то есть, если оперировать шестнадцатиразрядными данными, то на входе нашего алгоритма шифрования находятся две разности (e_{32} = 80 00 00 00, e_{32} = 80 00 00 00), а соответствующие им выходные пары разностей при этом — (e_{21} = 00 10 00 00, e_{21} = 00 10 00 00).

В каждом из трех полураундов на рис. 6.14 мы можем воспользоваться уравнением (6.2). Так как разность последних $\log_2\omega$ бит входных векторов равна нулю, то слагаемые, из которых получена разность, подвергающаяся циклическому сдвигу, будут сдвинуты на одно и то же число позиций. Тогда, согласно уравнению (6.2), получаем, что правая разность e_{32}, сложившись с левой разностью e_{32} по модулю два, даст на выходе нулевую разность, которая всегда останется нулевой независимо от того, на сколько позиций мы ее циклически сдвинем, то есть для первого полураунда вероятность появления на его выходе разности (e_0 = 00 00 00 00, e_{32} = 80 00 00 00) равна p = 1.

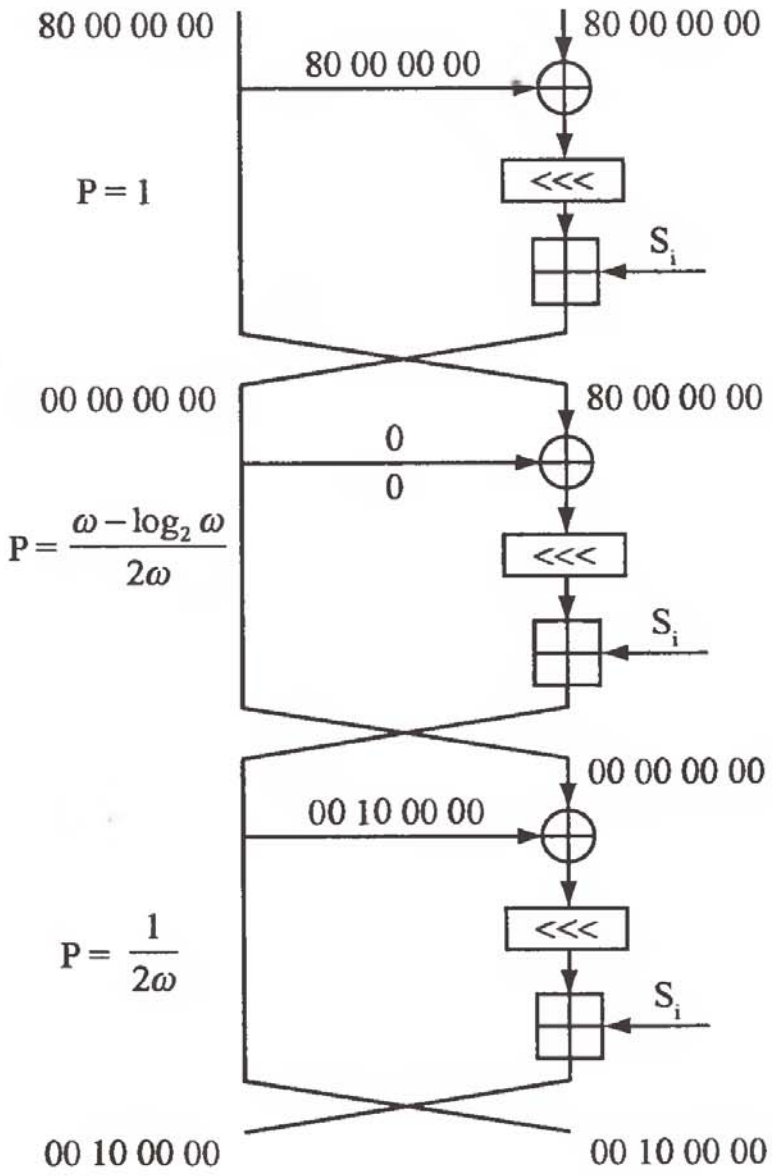


Рис. 6.14. ***Характеристика трех полураундов алгоритма шифрования RC5***

Так как нам известны выходные разности, то мы знаем, чему будет равна левая часть разности, после прохождения второго полураунда. Правая половина этой разности нам известна, так как она является левой частью разности на выходе первого полураунда. Кроме того, нам известно, что целочисленное сложение оставляет неизменной разность в одной позиции с вероятностью 1/2, а вероятность того, что позиция бита, в котором есть отличие, не попадет в биты, влияющие на циклический сдвиг, равна $\frac{\omega - \log_2 \omega}{\omega}$. Значит, вероятность выполнения второго полураунда с получением выходной разности (e_{21} = 00 10 00 00, e_0 = 00 00 00 00) равна $p = \frac{\omega - \log_2 \omega}{\omega} * \frac{1}{2} = \frac{\omega - \log_2 \omega}{2\omega}$.

С учетом известных выходных разностей видно, что в третьем полураунде не происходит циклического сдвига. Это может произойти в одном случае из ω (когда все сдвиговые разряды будут равны нулю), то есть с вероятностью $\frac{1}{\omega}$. Учитывая, как и в предыдущем случае, что целочисленное сложение оставляет неизменной разность в одной позиции с вероятностью 1/2, получаем, что вероятность выполнения третьего полураунда равна $p = \frac{1}{2} * \frac{1}{\omega} = \frac{1}{2\omega}$.

Таким образом, получаем, что вероятность для характеристики из трех полураундов алгоритма шифрования RC5 равна:

$$p = 1 * \frac{\omega - \log_2 \omega}{2\omega} * \frac{1}{2\omega} = \frac{\omega - \log_2 \omega}{4\omega^2}.$$

С помощью вероятности мы определяем, сколько пар текстов в среднем надо проанализировать для нахождения одной правильной пары. Так, если вероятность равна p, то в среднем потребуется проанализировать 1/p пар текстов.

Мы рассмотрели один частный случай и определили вероятность его выполнения. Бартон Калиски (Burton S.Kaliski) и Лиза Йин (Lisa Yin) в своей работе [26] выделили пять основных характеристик для одного полураунда алгоритма шифрования RC5. С их помощью можно успешно анализировать до шести раундов алгоритма, то есть 12 полураундов. Если обозначить через (L_{r-1}, R_{r-1}) разность на входе полураунда r, а через (L_r, R_r) соответствующую ей разность на выходе, то эти пять основных характеристик будут выглядеть так, как показано в таблице 6.6.

Таблица 6.6

Разностные характеристики для одного полураунда алгоритма шифрования RC5

Номер хар-ки	(Lr – 1, Rr – 1)	(Lr, Rr)	Условие	Вероятность
1	(ei, 0)	(ei, ei)	$i \geq \log 2\omega$	$p \geq (1/\omega) \cdot (1/2)$
2	(ei, ei)	(0, ei)	$i \geq \log 2\omega$	$p = 1$
3	(0, ei)	(et, 0)	$i, t \geq \log 2\omega$	$p \geq (1/\omega) \cdot (1/2)$
4	(ei, 0)	(et, ei)	$i, t \geq \log 2\omega, i \neq t$	$p \geq (1/\omega) \cdot (1/2)$
5	(ei, ei)	(eu ⊕ ev, et)	$i, t \geq \log 2\omega, i \neq t, u > v,$ $t - i = \pm (u-v) \bmod \omega$	$p \geq (1/\omega) \cdot (1/2) \cdot (1/2)$

Необходимо отметить, что для третьей, четвертой и пятой характеристик таблицы 6.6 существует много различных вариантов выходной характеристики. Так, для третьей характеристики существует $(\omega - \log_2\omega)$ вариантов значения параметра t, для четвертой характеристики это значение на единицу меньше, то есть равно $(\omega - \log_2\omega - 1)$. По мнению Б. Калиски и Л. Йин, очень полезным может оказать-

ся объединение первых трех характеристик при анализе трех полураундов (см. рис. 6.15). Из рис. 6.15 видно, что данное сочетание характеристик имеет вероятность:

$$p = \frac{1}{2} * \frac{1}{\omega} * 1 * \frac{1}{2} * \frac{1}{\omega} = \frac{1}{4\omega^2}.$$

Более того, теперь становится возможным объединить две характеристики для трех полураундов и получить одну характеристику для шести полураундов с вероятностью:

$$p = \frac{1}{4\omega^2} * \frac{1}{4\omega^2} = \frac{1}{16\omega^4}.$$

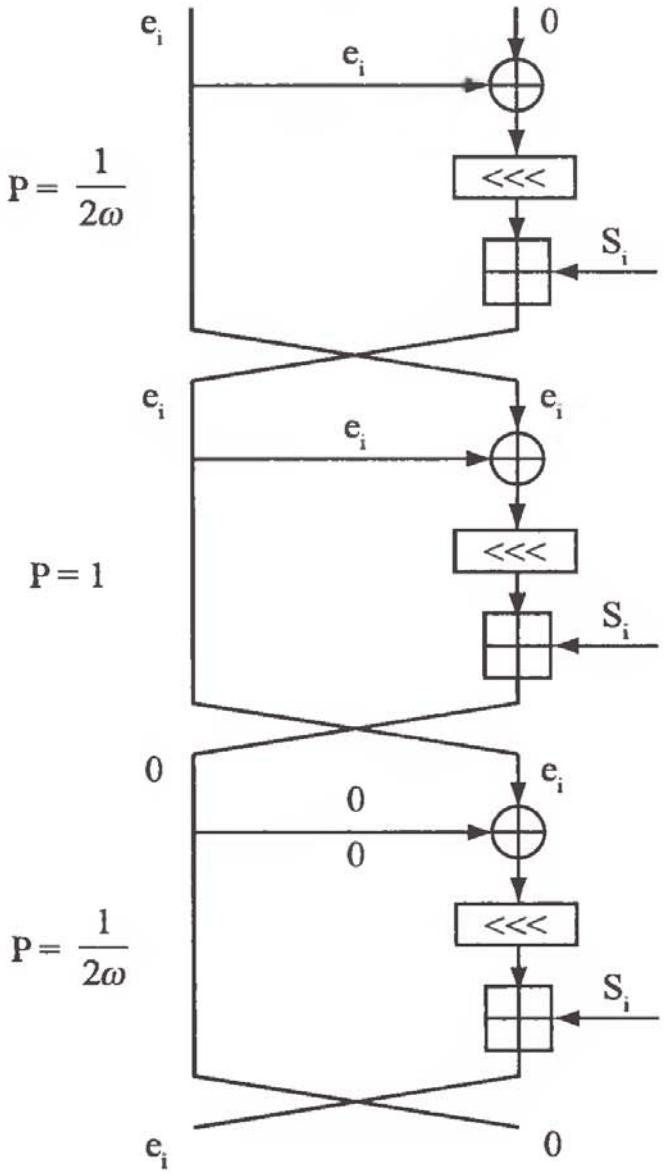


Рис. 6.15. ***Характеристика трех полураундов алгоритма шифрования RC5***

Важной задачей является выявление характеристик, имеющих наибольшую вероятность выполнения. Естественно, что при длине входного блока, равной 2ω, разностей, имеющих отличие в одном бите, при этом не влияющем на циклический сдвиг, будет $2^{2(\omega - \log_2 \omega)}$ вариантов. Поэтому, в отличие от дифференциального криптоанализа алгоритма шифрования DES, в котором мы могли использовать только одну наиболее вероятностную характеристику, здесь имеется достаточно широкий диапазон подбора правильных пар текстов, что позволяет подойти к анализу алгоритма с разных сторон (то есть если анализ с использованием одной характеристики не даст желаемого результата, то всегда есть возможность использовать другую характеристику).

Дифференциальная атака требует 2^{24} подобранных открытых текстов для 5 раундов, 245 — для 10 раундов, 2^{53} — для 12 раундов и 2^{68} — для 15 раундов [26]. Конечно же, существует только 2^{64} возможных открытых текстов, поэтому такая атака пока не применима к алгоритму с 15 и более раундами.

6.4.4. Дифференциальный криптоанализ алгоритма шифрования Serpent

Рассмотрим некоторые особенности дифференциального криптоанализа алгоритма шифрования Serpent. Для этого в начале введем несколько понятий, которыми в дальнейшем будем оперировать.

Под активным S-блоком будем понимать S-блок, на вход которого поступает пара значений, образующая ненулевую разность. Также будем считать, что разность на входе и выходе раунда R_k содержит в себе n частей (n = 32), и при этом каждая часть поступает на вход своего S-блока.

Для удобства работы в Приложении 2 в таблице 18 приведена зависимость выходных битов линейного преобразования от соответствующих входных битов преобразования для алгоритма шифрования Serpent. Так, например, нулевой выходной бит линейного преобразования (то есть, по сути дела, самый младший значащий бит) будет образован с помощью сложения по модулю два 16, 52, 56, 70, 83, 94 и 105 битов, находящихся на входе этого преобразования.

Как мы уже знаем, дифференциальный анализ начинается с анализа блоков замены, используемых в алгоритме и построения таблиц зависимостей значений ΔC от значения ΔA. Эти таблицы для алгоритма шифрования Serpent приведены в Приложении 2 с таблицы 19 по таблицу 26.

Рассмотрим основные свойства дифференциального криптоанализа рассматриваемого нами алгоритма.

1. Если входная разность раунда R_k затрагивает только один активный S-блок, то в следующем раунде R_{k+1} входная разность будет затрагивать по меньшей мере два активных S-блока. Так получается за счет использования линейного преобра-

зования (см. в Приложении 2 таблицу 18). Рассмотрим несколько примеров. Если в нулевом раунде (то есть в раунде, в котором используются S_0 блоки замены) на вход i-го ($0 \le i < 25$) S_0 блока (напомним, что нумерация блоков начинается с нуля, то есть i-й блок фактически будет являться (i + 1)) поступает значение разности, равное 0111, и при этом на выходе появляется значение разности, равное 0010 (по таблице 19, приведенной в Приложении 2, можно найти, что вероятность такого события равна $\frac{2}{16} = \frac{1}{8} = \frac{1}{2^3}$), то можно наблюдать следующую картину. Единица в выходной разности i-го S_0 блока будет являться (i*4 + 1) битом. В этом случае он повлияет на ((i + 6)*4), ((i + 1)*4 + 1), (((i + 30) mod 32)*4 + 2) выходные биты линейного преобразования, которые в свою очередь затронут три S_1 блока замены следующего раунда. Если же входная ненулевая разность будет затрагивать i-й S_0 блок, где $25 < i \le 31$, то единичный бит выходной разности этого блока повлияет на (((i + 6) mod 32)*4) и (((i + 1) mod 32)*4 + 1) выходные биты линейного преобразования. А это означает, что в следующем раунде активными будут два блока ((i + 6) mod 32) и ((i + 1) mod 32). К примеру, если речь пойдет о седьмом S_0 блоке, то единичный бит в разности на выходе этого блока будет являться 7*4 + 1 = 29 битом, который окажет влияние на ((7 + 6)*4) = 52 , ((7 + 1)*4 + 1) = 33 и (((7 + + 30) mod 32)*4 + 2) = 22 биты. В этом можно убедиться с помощью таблицы 18, приведенной в Приложении 2. Из нее видно, что:

52 бит = 7 ⊕ 18 ⊕ **29** ⊕ 104 ⊕ 108 ⊕ 122;

33 бит = 18 ⊕ **29** ⊕ 104;

22 бит = 18 ⊕ 22 ⊕ **29** ⊕ 35 ⊕ 50 ⊕ 96 ⊕ 104.

Если же речь пойдет, например, о 30-м S_0 блоке, то единичный бит в разности на выходе этого блока будет являться 30*4 + 1 = 121 битом, который окажет влияние на (((30 + 6) mod 32)*4) = 16 и (((30 + 1) mod 32)*4 + 1) = 125 биты. В этом можно убедиться с помощью таблицы 18, приведенной в Приложении 2. Из нее видно, что:

16 бит = 32 ⊕ 68 ⊕ 72 ⊕ 86 ⊕ 99 ⊕ 110 ⊕ **121**;

125 бит = 68 ⊕ 110 ⊕ **121**.

Если обозначить через S_n^{pos} блок замены, где нижний индекс n обозначает его номер (то есть $0 \le n < 8$), а верхний индекс pos — его порядковый номер (то есть $0 \le pos < 32$), то все вышесказанное можно представить так, как это сделано в таблице 6.7. При этом в таблице 6.7 значения разностей на входах и выходах блоков замены приведены для удобства в шестнадцатеричной системе счисления. Также в таблицу добавлена еще одна, не рассмотренная нами, одноraундовая характеристика для $S_0^i = 13$.

Таблица 6.7

Однораундовые характеристики для дифференциального криптоанализа алгоритма шифрования Serpent

Разность на входе S_n^{pos} блока замены	Разность на выходе S_n^{pos} блока замены	Вероятность	Измененные биты на входе линейного преобразования	Измененные биты на выходе линейного преобразования
$S_0^i = 7$, $0 \le i < 25$	$S_0^i = 2$	$p = \frac{1}{2^3}$	(i*4) + 1	((i + 6)*4), ((i + 1)*4 + 1), (((i + 30) mod 32)*4 + 2)
$S_0^i = 7$, $25 < i \le 31$	$S_0^i = 2$	$p = \frac{1}{2^3}$	(i*4) + 1	(((i + 6) mod 32)*4), (((i + 1) mod 32)*4 + 1)
$S_0^i = 13$	$S_0^i = 8$	$p = \frac{1}{2^2}$	(i*4) + 3	(((i + 7) mod 32)*4 + 3), (((i + 12) mod 32)*4), (((i + 29) mod 32)*4 + 2)

2. Еще одним свойством рассматриваемого алгоритма является то, что если на вход раунда R_k поступает разность, затрагивающая два активных S-блока, то в следующем раунде R_{k+1} входная разность может затрагивать только один активный S-блок. Так, если в нулевом раунде на вход (i + 10)-го S_0 блока поступает значение разности, равное 1010, а на вход (i + 7)-го S_0 блока поступает значение разности, равное 1100, и при этом на выходе (i + 10)-го блока появляется значение разности, равное 1010, а на выходе (i + 7)-го блока — 0100 (по таблице 19, приведенной в Приложении 2, можно найти, что вероятность такого события равна $\frac{4}{16} * \frac{4}{16} = \frac{1}{4} * \frac{1}{4} = \frac{1}{2^4}$), то можно наблюдать следующую картину. В выходной разности (i + 7)-го блока замены присутствует только одна единица, которая является (((i + 7) mod 32)*4 + 2) битом. В выходной разности (i + 10)-го блока замены присутствует две единицы, которые соответственно являются (((i + + 10) mod 32)*4x + 1) и (((i + 10) mod 32) · 4 + 3) битами. Эти три бита повлияют на изменение (i · 4 + 2) выходного бита линейного преобразования. К примеру, если взять значение i равным 3, то мы будем рассматривать десятый и тринадцатый S_0 блоки замены. В этом случае единичный бит в разности на выходе десятого блока будет являться (((3 + 7) mod 32)*4 + 2) = 42 битом, а единичные выходы тринадцатого блока замены будут являться (((3 + 10) mod 32)*4 + 1) = 53 и (((3 + 10) mod 32)*4 + 3) = 55 битами, соответственно. При этом 42 бит окажет влияние на изменение (3*4 + 2) = 14 бита. Все три бита будут участвовать в формировании и других выходных битов линейного преобразования. Однако во всех остальных случаях они всегда будут присутствовать парно, что не окажет никакого влияния на изменение выходных битов преобразования.

В этом можно убедиться с помощью таблицы 18, приведенной в Приложении 2. Из нее видно, что:

$$14 \text{ бит} = 10 \oplus 14 \oplus 21 \oplus 27 \oplus \mathbf{42} \oplus 88 \oplus 96.$$

Биты 42, 53 и 55 будут участвовать также в формировании 42, 46, 57, 83 и 100 выходных битов линейного преобразования:

42 бит = 38 ⊕ **42** ⊕ 49 ⊕ **55** ⊕ 70 ⊕ 116 ⊕ 124;

46 бит = 0 ⊕ **42** ⊕ 46 ⊕ **53** ⊕ 59 ⊕ 74 ⊕ 120;

57 бит = 0 ⊕ **42** ⊕ **53**;

83 бит = **42** ⊕ **55** ⊕ 116;

100 бит = 24 ⊕ 28 ⊕ **42** ⊕ **55** ⊕ 66 ⊕ 77 ⊕ 116,

однако не окажут на их изменение никакого влияния.

В таблице 6.8 приведены примеры одноraундовых характеристик, аналогичные рассмотренным выше. В таблице 6.8, также как и в таблице 6.7, значения разностей на входах и выходах блоков замены приведены для удобства в шестнадцатеричной системе счисления.

Таблица 6.8

Одноraундовые характеристики для дифференциального криптоанализа алгоритма шифрования Serpent

Разность на входе S_n^{pos} блока замены	Разность на выходе S_n^{pos} блока замены	Вероятность	Измененные биты на входе линейного преобразования	Измененные биты на выходе линейного преобразования
$S_0^i + 10 = 10$, $S_0^i + 7 = 12$	$S_0^i + 10 = 10$, $S_0^i + 7 = 4$	$p = \frac{1}{2^4}$	(((i + 7) mod 32)*4 + 2) (((i + 10) mod 32)*4 + 1) (((i + 10) mod 32)*4 + 3)	(i*4) + 2
$S_0^i + 10 = 9$, $S_0^i + 7 = 14$	$S_0^i + 10 = 10$, $S_0^i + 7 = 4$	$p = \frac{1}{2^5}$	(((i + 7) mod 32)*4 + 2) (((i + 10) mod 32)*4 + 1) (((i + 10) mod 32)*4 + 3)	(i*4) + 2

3. Для любой разности, поступающей на вход любого блока замены (за исключением разности, равной нулю), выходная разность появится с вероятностью, равной нулю или $\frac{1}{2^2}$, или $\frac{1}{2^4}$. В этом легко убедиться, проанализировав таблицы с 19 по 26, приведенные в Приложении 2.

Рассмотрим один из вариантов анализа пяти раундов алгоритма Serpent (при этом будем прослеживать процесс изменения разностей со второго по шестой раунды). Во втором раунде для замены битов используется S_1 блок. Подадим на нулевой S_1 блок значение разности, равное 1110, при этом с вероятностью $p = \frac{1}{2^2}$ (см. таблицу 20 в Приложении 2) на выходе появится значение разности, равное 1000. Таким образом, получаем первый единичный бит в позиции 3. На четвер-

тый S_1 блок подадим значение разности, равное 1010. В этом случае с вероятностью $p = \frac{1}{2^2}$ на выходе появится значение разности, равное 0100. А значит, получаем второй единичный бит в позиции 18. На вход шестого S_1 блока замены подадим значение разности, равное 1101. При этом с вероятностью $p = \frac{1}{2^2}$ на выходе появится значение разности, равное 0010, а это уже третий единичный бит в позиции 25. Аналогичным образом получаем единичные биты в позициях 29 и 31, которые образуются при изменении входной разности 0010 седьмого и шестого S_1 блока на разность 1010 с вероятностью $p = \frac{1}{2^2}$. Единичный бит в позиции 73 образуется при преобразовании входной разности 1100 с помощью восемнадцатого S_1 блока замены в выходную разность 0001 с вероятностью $p = \frac{1}{2^2}$. И, наконец, единичный бит в позиции 125 образуется при преобразовании входной разности 1101 с помощью тридцать первого S_1 блока замены в выходную разность 0010 с вероятностью $p = \frac{1}{2^2}$.

После этого биты подвергаются линейному преобразованию. Из таблицы 18 в Приложении 2 видно, что 3, 25, 18, 31 и 72 входные биты преобразования окажут влияние на изменение 16, 18, 29 и 31 выходных битов:

16 бит = 32 ⊕ 68 ⊕ **72** ⊕ 86 ⊕ 99 ⊕ 110 ⊕ 121;

18 бит = 14 ⊕ **18** ⊕ **25** ⊕ **31** ⊕ 46 ⊕ 92 ⊕ 100;

29 бит = 14 ⊕ **25** ⊕ 100;

31 бит = **3** ⊕ 118.

Биты 3, 18, 25, 72 и 125 будут участвовать также в формировании 1, 20, 48 и 118 выходных битов линейного преобразования:

1 бит = **72** ⊕ 114 ⊕ **125**;

20 бит = 36 ⊕ **72** ⊕ 76 ⊕ 90 ⊕ 103 ⊕ 114 ⊕ **125**;

48 бит = **3** ⊕ 14 ⊕ **25** ⊕ 100 ⊕ 104 ⊕ 118;

118 бит = **3** ⊕ **18** ⊕ **72** ⊕ 114 ⊕ 118 ⊕ **125**,

однако не окажут на их изменение никакого влияния.

Тогда, можно сказать, что входная разность

$$D0\ 00\ 00\ 00\ 00\ 00\ 0C\ 00\ 00\ 00\ 00\ 00\ 2D\ 0A\ 00\ 0E_x$$

преобразуется в выходную разность, равную

$$00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ A0\ 05\ 00\ 00_x,$$

с вероятностью, равной $p = \frac{1}{2^2} * \frac{1}{2^2} * \frac{1}{2^2} * \frac{1}{2^3} * \frac{1}{2^2} * \frac{1}{2^2} = \frac{1}{2^{13}}$.

Это значит, что на входе третьего раунда шифрования ненулевые разности будут поступать только на четвертый и седьмой блоки замены. В третьем раунде замена битов производится с помощью S_2 блока замены. Таким образом, получаем, что на вход четвертого S_2 блока замены поступит значение разности, равное 0101, а на вход седьмого — 1010. При этом на выходе четвертого блока замены с вероятностью $p = \frac{1}{2^3}$ (см. таблицу 21 в Приложении 2) появится значение разности, равное 0100, что даст нам единичный бит в позиции 18, а на выходе седьмого блока — разность 1010 с вероятностью $p = \frac{1}{2^2}$, что даст нам единичные биты в позициях 29 и 31.

Из таблицы 18 в Приложении 2 видно, что 18 входной бит линейного преобразования окажет влияние на изменение 118 выходного бита:

118 бит = 3 ⊕ **18** ⊕ 72 ⊕ 114 ⊕ 118 ⊕ 125.

Биты 18, 29 и 31 будут участвовать также в формировании 18, 22, 33, 52, 59 и 76 выходных битов линейного преобразования, однако не окажут на их изменение никакого влияния (убедиться в этом можно с помощью таблицы 18, приведенной в Приложении 2).

Тогда, можно сказать, что входная разность третьего раунда шифрования

$$00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ A0\ 05\ 00\ 00_x$$

преобразуется в выходную разность, равную

$$00\ 40\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00_x,$$

с вероятностью, равной $p = \frac{1}{2^3} * \frac{1}{2^2} = \frac{1}{2^5}$.

Поэтому на входе четвертого раунда шифрования поступит только одна ненулевая разность, а именно на 29 блок замены. В четвертом раунде замена битов производится с помощью S_3 блока замены. Таким образом, получаем, что на вход 29 S_3 блока замены поступит значение разности, равное 0100. С вероятностью $p = \frac{1}{2^3}$ (см. таблицу 22 в Приложении 2) на выходе этого блока появится значение разности, равное 1010, что даст нам единичные биты в позициях 119 и 117.

Из таблицы 18 в Приложении 2 видно, что 117 и 119 входные биты линейного преобразования окажут влияние на изменение 12, 19, 36, 106 и 121 выходных битов:

12 бит = 28 ⊕ 64 ⊕ 68 ⊕ 82 ⊕ 95 ⊕ 106 ⊕ **117**;

19 бит = 52 ⊕ 106 ⊕ **119**;

36 бит = 2 ⊕ 13 ⊕ 52 ⊕ 88 ⊕ 92 ⊕ 106 ⊕ **119**;

106 бит = 6 ⊕ 52 ⊕ 106 ⊕ **119**;

121 бит = 64 ⊕ 106 ⊕ **117**.

Таблица 6.9

Построение пятираундовой характеристики для алгоритма шифрования Serpent

№ раунда, R	Входы S_R^{ind} блока замены	Выходы S_R^{ind} блока замены	Вероятность	Измененные биты на входе линейного преобразования	Измененные биты на выходе линейного преобразования
2	$S_1^0 = 14$ $S_1^4 = 10$ $S_1^6 = 13$ $S_1^7 = 2$ $S_1^{18} = 12$ $S_1^{31} = 13$	$S_1^0 = 8$ $S_1^4 = 4$ $S_1^6 = 2$ $S_1^7 = 10$ $S_1^{18} = 1$ $S_1^{31} = 2$	$P = \frac{1}{2^{13}}$	3, 18, 25, 29, 31, 73, 125	16, 18, 29, 31
3	$S_2^4 = 5$ $S_2^7 = 10$	$S_2^4 = 4$ $S_2^7 = 10$	$P = \frac{1}{2^5}$	18, 33, 35	118
4	$S_3^{29} = 4$	$S_3^{29} = 10$	$P = \frac{1}{2^3}$	117, 119	12, 19, 36, 106, 121
5	$S_4^3 = 1$ $S_4^4 = 8$ $S_4^9 = 1$ $S_4^{26} = 4$ $S_4^{30} = 2$	$S_4^3 = 14$ $S_4^4 = 12$ $S_4^9 = 7$ $S_4^{26} = 3$ $S_4^{30} = 6$	$P = \frac{1}{2^{10}}$	13, 14, 15, 18, 19, 36, 37, 38, 104, 105, 121, 122	0, 2, 3, 9, 10, 14, 16, 17, 20, 28, 29, 35, 36, 38, 41, 42, 43, 47, 52, 53, 55, 59, 64, 71, 76, 79, 82, 88, 90, 93, 94, 96, 108, 109, 112, 114, 118, 122, 125, 126
6	$S_5^0 = 13$ $S_5 4 = 6$ $S_5^3 = 4$ $S_5^4 = 3$ $S_5^5 = 1$ $S_5^7 = 3$ $S_5^8 = 8$ $S_5^9 = 5$ $S_5^{10} = 14$ $S_5^{11} = 8$ $S_5^{13} = 11$ $S_5^{14} = 8$ $S_5^{16} = 1$ $S_5^{17} = 8$ $S_5^{19} = 9$ $S_5^{20} = 4$ $S_5^{22} = 5$ $S_5^{23} = 6$ $S_5^{24} = 1$ $S_5^{27} = 3$ $S_5^{28} = 5$ $S_5^{29} = 4$ $S_5^{30} = 4$ $S_5^{31} = 6$	$S_5^0 = 2$ $S_5^4 = 6$ $S_5^3 = 5$ $S_5^4 = 8$ $S_5^5 = 6$ $S_5^7 = 8$ $S_5^8 = 12$ $S_5^9 = 14$ $S_5^{10} = 15$ $S_5^{11} = 12$ $S_5^{13} = 1$ $S_5^{14} = 12$ $S_5^{16} = 6$ $S_5^{17} = 12$ $S_5^{19} = 5$ $S_5^{20} =$ $S_5^{22} = 14$ $S_5^{23} = 6$ $S_5^{24} = 6$ $S_5^{27} = 8$ $S_5^{28} = 14$ $S_5^{29} = 9$ $S_5^{30} = 9$ $S_5^{31} = 6$	$P = \frac{1}{2^{49}}$		

Аналогичным образом, в пятом раунде пройдет преобразование, которое приведет к изменению 0, 2, 3, 9, 10, 14, 16, 17, 20, 28, 29, 35, 36, 38, 41, 42, 43, 47, 52, 53, 55, 59, 64, 71, 76, 79, 82, 88, 90, 93, 94, 96, 108, 109, 112, 114, 118, 122, 125 и 126 битов с вероятностью $p = \frac{1}{2^{10}}$.

В шестом раунде, так как он у нас является конечным, линейное преобразование отсутствует, поэтому выполняется замена входных разностей на выходные так, как показано в таблице 6.9.

Из таблицы 3 видно, что вероятность выполнения определенной нами пятираундовой характеристики равна $p = \frac{1}{2^{13}} * \frac{1}{2^{5}} * \frac{1}{2^{3}} * \frac{1}{2^{10}} * \frac{1}{2^{49}} = \frac{1}{2^{80}}$. Еще несколько многораундовых характеристик для алгоритма шифрования Serpent приведены в Приложении 2 в таблицах 35–38. Пользуясь построенными характеристиками, можно проводить анализ упрощенного алгоритма Serpent, то есть алгоритма, содержащего меньше раундов, чем обусловлено разработчиками алгоритма, аналогично тому, как это описано для алгоритма шифрования DES. Вначале надо подобрать правильные пары *открытый—закрытый текст* по построенным характеристикам, а затем анализировать каждую из найденных пар с целью накопления статистики по наиболее часто встречающимся значениям подключей.

6.5. Контрольные вопросы

1. Что является дифференциалом i-го цикла?
2. Опишите алгоритм нахождения подключа, используемого в последнем раунде шифрования.
3. Какая пара текстов в дифференциальном криптоанализе называется правильной?
4. Опишите принцип дифференциального криптоанализа одного раунда алгоритма шифрования DES.
5. Какими свойствами обладают таблицы, получаемые в процессе проведения дифференциального криптоанализа?
6. Какие наиболее распространенные одноraундовые характеристики алгоритма шифрования DES вы знаете?
7. За счет чего становится возможным проведение дифференциального криптоанализа трехраундовых алгоритмов блочного шифрования, построенных по схеме Фейстеля?
8. Какой принцип используется при проведении дифференциального криптоанализа полного 16-раундового алгоритма шифрования DES?
9. Почему два последние цикла 16-раундового алгоритма шифрования DES не влияют на вероятность успеха проведения дифференциального криптоанализа этого алгоритма?
10. Почему при дифференциальном криптоанализе алгоритма шифрования SPN используются характеристики, имеющие не самые высокие вероятности?

Глава 7. ЛИНЕЙНО-ДИФФЕРЕНЦИАЛЬНЫЙ КРИПТОАНАЛИЗ

Эта глава посвящена новой атаке, называемой **линейно-дифференциальным криптоанализом**. Данный вид криптоанализа основан на выборе открытого текста и главным образом предназначен для циклических алгоритмов шифрования, таких как, например DES. Используя этот новый метод атаки, С. Лангфорд и М. Хеллман сумели определить 10 битов ключа алгоритма шифрования DES с вероятностью успеха 80% при использовании 512 открытых текстов [19]. При этом вероятность успеха возрастает до 95% с увеличением количества открытых текстов до 768 [19].

Циклические алгоритмы шифрования основаны на n-кратном повторении функции шифрования. При этом каждый раунд является функцией, зависящей от выхода предыдущего раунда и битов ключа. Таким алгоритмом является DES и в силу своей известности является предметом большинства исследований по изучению криптостойкости циклических блочных алгоритмов шифрования.

Существуют три наиболее эффективные атаки, применимые к алгоритму DES: метод полного перебора, дифференциальный криптоанализ Бихама и Шамира и линейный криптоанализ Матсуи. В то время как метод полного перебора все еще остается самой эффективной атакой на полный 16-раундовый алгоритм шифрования DES, исследования ученых сосредоточены на аналитических атаках, появившихся позднее, в надежде, что эти исследования позволят сделать новые атаки более практичными. Так, например, метод линейного криптоанализа гораздо быстрее метода полного перебора, но требует невозможное количество открытых текстов (2^{43}), зашифрованных на одном ключе [6, 19]. Напротив, метод полного перебора нуждается только в одном открытом тексте.

Метод дифференциально-линейного криптоанализа основан на комбинировании методов линейного и дифференциального криптоанализа, описанных выше в главах 5 и 6.

7.1. Основные принципы построения линейно-дифференциального криптоанализа

Рассмотрим основные принципы, на которых строится линейно-дифференциальный криптоанализ. Мы можем опустить начальную перестановку IP и конечную перестановку IP^{-1}, так как они не влияют на криптографическую стойкость зашифрованных сообщений. Рассмотрим дифференциальный криптоанализ трех первых раундов алгоритма шифрования DES. Нас интересуют ситуации, когда правые части входных сообщений будут иметь одинаковые значения, за исключением второго или третьего, или и второго, и третьего битов вместе. Это обуслов-

лено тем, что в этих случаях будет затронута только разность, поступающая на вход S_1 блока, а на вход всех остальных блоков будут поступать нулевые разности. В этом случае левая часть разности будет состоять из всех нулей, за исключением второй и третьей позиции, то есть тех позиций, в которых имеется различие двух входных сообщений. Мы рассмотрим вариант, когда различие есть и во втором, и в третьем битах. В правой части входные сообщения не будут иметь различий, и поэтому их разность будет равна нулю (см. рис. 7.1).

На рис. 7.1 разности, поступающие на вход функции F первого и второго раунда обозначены соответственно как a′ и b′, а разность на выходе функции F первого раунда шифрования — как A′.

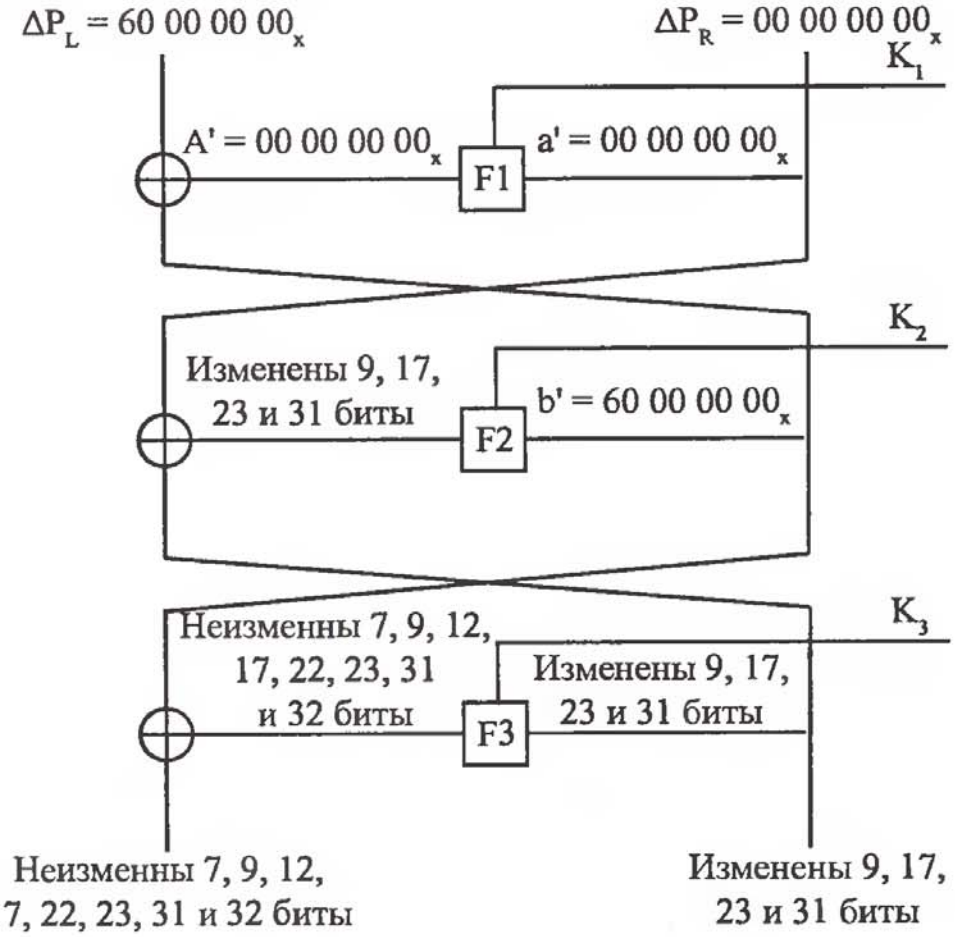


Рис. 7.1. ***Трехраундовая характеристика алгоритма шифрования DES***

Входная нулевая разность всегда даст на выходе нулевую разность, поэтому на вход F-функции второго раунда шифрования поступит левая часть входной разности. После того как входные сообщения F-функции подвергнутся перестановке с расширением по таблице 3.5, мы получим, что на вход S_1 блока поступит разность, равная 001100 (первый ноль разности на входе S_1 блока является 32 битом входной разности, 2, 3, 4, 5 и 6 биты разности на входе S_1 блока являются 1, 2, 3, 4 и 5 битами входной разности, соответственно). На вход остальных S-блоков пос-

тупят разности, равные нулю. Таким образом, на выходе всех S-блоков, за исключением S_1 блока, появятся нулевые разности. Какая разность появится на выходе S_1 блока мы не знаем, но мы знаем, что выходы S_1 блока после перестановки (см. таблицу 3.14) окажутся в 9, 17, 23 и 31 позициях. Здесь также может быть полезна информация о том, что мы можем предположить с некоторой вероятностью появление той или иной разности на выходе S_1 блока при поступлении на его вход конкретно определенной разности.

Итак, на выходе F-функции второго раунда алгоритма шифрования DES появится разность, которая возможно имеет различие в 9, 17, 23 и 31 позициях. Эта разность не изменится после сложения по модулю два с нулевой разностью, поступившей с выхода предыдущего раунда, и будет являться входной разностью F-функции третьего раунда шифрования.

После того как входная разность F-функции третьего раунда шифрования подвергнется перестановке с расширением, возможно измененные 9, 17, 23 и 31 биты окажутся во входных разностях всех S-блоков замены, кроме S_1 и S_7 блоков (9 бит будет присутствовать во входных разностях S_2 и S_3 блоков, 17 бит — S_4 и S_5 блоков, 23 — S_5 блока, 31 — S_8 блока). А значит, мы точно знаем, что на выходе первого и седьмого блоков замены появятся нулевые разности. При этом выходы остальных блоков замены нам точно не известны. Как и в предыдущем раунде, выходы S_1 блока после перестановки (см. таблицу 3.14) окажутся в 9, 17, 23 и 31 позициях, а выходы блока S_7 — в 32, 12, 22 и 7 позициях. После перестановки выходная разность F-функции третьего раунда шифрования будет иметь неизвестные нам значения в первых четырех битах (так как мы уже выяснили, что неизменными останутся только 7, 9, 12, 17, 22, 23, 31 и 32 биты). Поэтому ее сложение по модулю два с выходом предыдущего раунда $b' = 60\ 00\ 00\ 00_x$ не изменит текущего положения дел, так как в b' в 7, 9, 12, 17, 22, 23, 31 и 32 битах стоят нули.

Итак, после трех раундов шифрования левая часть выходной разности **всегда** будет иметь неизменные значения в 7, 9, 12, 17, 22, 23, 31 и 32 позициях (то есть в нашем случае, согласно входной разности, это будут нули). А правая часть выходной разности, возможно, будет иметь измененное значение в 9, 17, 23 и 31 позициях (то есть все остальные биты правой части выходной разности, согласно входной разности, **всегда** будут иметь нулевые значения).

Если бы мы рассматривали алгоритм шифрования DES, состоящий из 5 раундов, то, зная выходную разность, легко могли бы определить входы S-блоков последнего пятого раунда шифрования и выходные разности первого и седьмого блоков замены F-функции последнего раунда шифрования. Это позволило бы нам без труда определить 12 битов ключа, пользуясь только методом дифференциального криптоанализа (то есть шесть битов ключа, изменяющих вход S_1 блока замены последнего раунда шифрования, и шесть битов, изменяющих вход S_7 блока замены того же раунда).

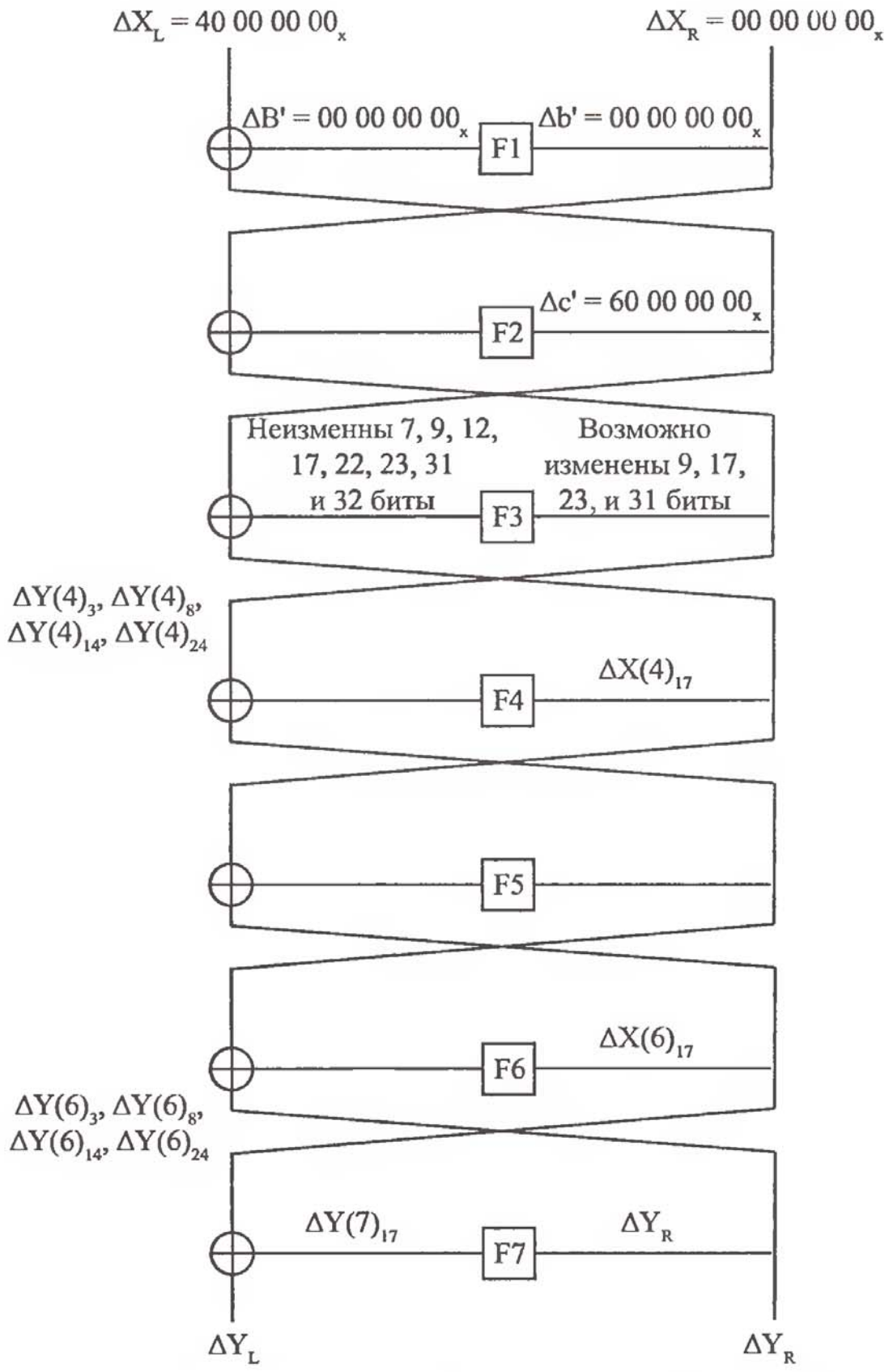


Рис. 7.2. ***Применение линейно-дифференциального криптоанализа к 7 раундам алгоритма шифрования DES***

Теперь перейдем к рассмотрению алгоритма шифрования DES, состоящего из 7 раундов (см. рис. 7.2). К первым трем раундам применим дифференциальный криптоанализ, как было описано выше. В следующих трех раундах (то есть с 4 по 6) будем оперировать понятиями линейного криптоанализа.

В п. 5.2.2 нами был получен эффективный линейный аналог для трех раундов алгоритма шифрования DES (см. уравнение (5.13)). Мы рассматриваем два входных сообщения. Обозначим их как X и X′, имеющих исходную, заранее определенную нами, разность. Для первого сообщения X эффективным линейным статистическим аналогом 4–6 циклов будет являться уравнение:

$$X(4)_{17} \oplus Y(4)_3 \oplus Y(4)_8 \oplus Y(4)_{14} \oplus Y(4)_{25} \oplus X(6)_{17} \oplus Y(6)_3 \oplus Y(6)_8 \oplus Y(6)_{14} \oplus \oplus Y(6)_{25} = K(4)_{26} \oplus K(6)_{26}. \quad (7.1)$$

В уравнении (7.1) в скобках указан номер цикла, а индексы обозначают соответствующий номер бита.

Аналогичным образом для второго сообщения X′ эффективным линейным статистическим аналогом 4–6 циклов будет являться уравнение:

$$X'(4)_{17} \oplus Y'(4)_3 \oplus Y'(4)_8 \oplus Y'(4)_{14} \oplus Y'(4)_{25} \oplus X'(6)_{17} \oplus Y'(6)_3 \oplus Y'(6)_8 \oplus \oplus Y'(6)_{14} \oplus Y'(6)_{25} = K(4)_26 \oplus K(6)_{26}. \quad (7.2)$$

Так как и сообщение X, и сообщение X′ зашифрованы на одном и том же ключе K, то правые части в уравнениях (7.1) и (7.2) совпадают. Поэтому, если мы сложим по модулю два эти уравнения, то получим уравнение (7.3), в котором левая часть состоит из суммы по модулю два разностей соответствующих битов, а правая часть уравнения равна нулю:

$$\Delta X(4)_{17} \oplus \Delta Y(4)_3 \oplus \Delta Y(4)_8 \oplus \Delta Y(4)_{14} \oplus \Delta Y(4)_{25} \oplus \Delta X(6)_{17} \oplus \Delta Y(6)_3 \oplus \oplus \Delta Y(6)_8 \oplus \Delta Y(6)_{14} \oplus Y\,\Delta(6)_{25} = 0. \quad (7.3)$$

Ранее с помощью дифференциального криптоанализа трех первых циклов алгоритма шифрования мы определили, что левая часть разности останется неизменной в 17 позиции. Так как левая часть выходной разности третьего раунда является правой частью входной разности четвертого раунда шифрования, то мы точно знаем значение $\Delta X(4)_{17}$. Мы также определили, что правая часть выходной разности третьего раунда шифрования останется неизменной во всех позициях, кроме 6, 17, 23 и 31 позиций. Так как правая часть выходной разности третьего раунда шифрования является левой частью входной разности четвертого раунда шифрования, то нам точно известны значения $\Delta Y(4)_3$, $\Delta Y(4)_8$, $\Delta Y(4)_{14}$ и $\Delta Y(4)_{25}$.

Обязательным условием атаки такого рода является знание криптоаналитиком шифртекстов, соответствующих исходным открытым текстам. Поэтому нам заранее известен шифртекст Y, соответствующий сообщению X, и шифртекст Y′, соответствующий сообщению X′. А значит, нам известна и их разность ΔY. Так как левая часть выходной разности шестого раунда является правой частью входной разности седьмого раунда шифрования, а также является правой частью раз-

ности шифртекстов ΔY, то нам точно известны значения $\Delta Y(6)_3$, $\Delta Y(6)_8$, $\Delta Y(6)_{14}$ и $\Delta Y(6)_{25}$.

Получается, что уравнение (7.3) является уравнением с одной неизвестной $\Delta X(6)_{17}$, которая может быть легко вычислена. Необходимо помнить, что уравнение (7.3) выполняется с вероятностью, равной произведению вероятностей для уравнения (7.1) и уравнения (7.2):

$$p = \frac{39}{128} * \frac{39}{128} \approx 0{,}0928 .$$

Теперь перейдем к рассмотрению последнего седьмого раунда шифрования. Сложив найденное значение $\Delta X(6)_{17}$ по модулю 2 с 17 битом левой части выходной разности, мы получим значение 17-го бита разности выхода F-функции 7 раунда шифрования. Если воспользоваться таблицей перестановки, то можно определить, что семнадцатый выходной бит F-функции является вторым выходным битом S_1 блока замены. Так как нам известна правая часть разности шифртекста, то нам известна разность, поступающая на вход F-функции седьмого раунда шифрования. А значит, мы точно можем определить разность, поступающую на вход S_1 блока замены.

7.2. Применение линейно-дифференциального криптоанализа к алгоритму DES

В 10 главе приведен ряд практических задач по линейно-дифференциальному криптоанализу, в которых надо определить входную разность функции F третьего раунда шифрования пятираундового алгоритма шифрования, зная значения всех таблиц, используемых в этом алгоритме. Решение этих задач поможет более детально разобраться в данном методе анализа и получить практические навыки анализа.

Зная разность на входе S-блока и разность второго бита выхода S-блока, можно отсеять половину выходных разностей S_1 блока, которая не соответствует разности второго выходного бита. Например, если нам точно известно, что второй бит выходной разности S_1 блока будет равен единице, то для анализа мы оставляем следующие 8 выходных значений (из возможных шестнадцати): 0100, 0101, 0110, 0111, 1100, 1101, 1110, 1111. Однако этой информации сравнительно мало для точного извлечения ключа. Поэтому С. Лангфорд и М. Хеллман предложили вариант атаки на алгоритм шифрования DES, состоящий из восьми раундов (см. рис. 7.3) [19]. К уже рассмотренным семи раундами они добавили дополнительный первый раунд. В этом случае, чтобы сохранить правильное поступление разностей на вход второго раунда шифрования, правые части входных сообщений должны иметь различие во втором или третьем бите или и в том и в другом одновременно.

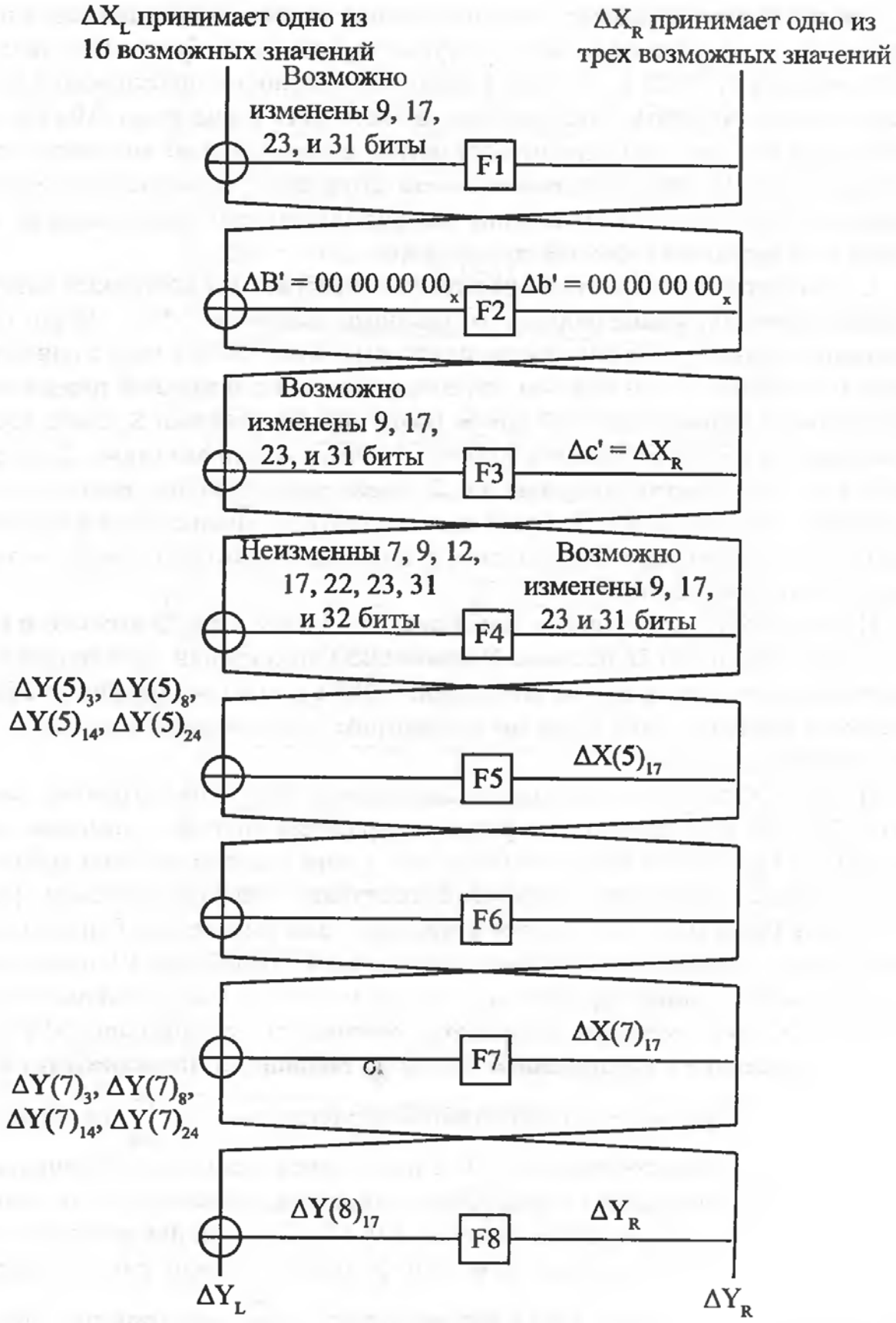


Рис. 7.3. ***Применение линейно-дифференциального криптоанализа к 8 раундам алгоритма шифрования DES***

Как мы выяснили раньше, при поступлении на вход F-функции такой разности, разность, получаемая на входе F-функции, возможно, будет иметь измененные значения в 9, 17, 23 и 31 битах. Правая часть разности, поступающей на вход второго раунда алгоритма шифрования, должна быть равна нулю. Мы не знаем разность на выходе F-функции первого раунда шифрования, но мы знаем, что существует всего 16 вариантов этой разности. (При более внимательном изучении таблиц, построенных для проведения дифференциального криптоанализа, количество этих вариантов окажется еще меньше.)

С. Лангфорд и М. Хеллман предложили перебрать всевозможные варианты входных значений, удовлетворяющих заданным разностям 3*16 = 48 (то есть 3 возможных правых разности, каждая из которых может быть в паре с одной из 16 левых разностей). Таким образом, перебирая для каждой входной пары возможные варианты битов подключей для S_1 блока первого раунда и S_1 блока восьмого раунда, они смогли определить значение 10 битов исходного ключа. Дело в том, что два из шести битов подключа для S_1 блока первого раунда присутствуют в шести битах подключа для S_1 блока восьмого раунда. Знание этого факта также оказало свое влияние на отбор подключей. Конечно, десять битов ключа — это немного. Но это уже кое-что.

Изучив работы Лангфорда и Хеллмана, Эли Бихам, Орр Дункелман и Натан Келлер (Eli Biham, Orr Dunkelman, Nathan Keller) предложили свой подход к анализу восьми раундов алгоритма шифрования DES с помощью линейно-дифференциального криптоанализа. Ниже мы рассмотрим предложенный ими способ анализа данного алгоритма.

На рис. 7.4 показан еще один способ анализа 8 раундов алгоритма шифрования DES. На вход данного алгоритма шифрования поступает значение разности $(\Delta X_L, \Delta X_R)$ = (00 80 82 00, 60 00 00 00). Таким образом, на вход функции F первого раунда алгоритма шифрования поступает значение разности, равное 60 00 00 00. Ранее мы выяснили, что в этом случае на выходе всех блоков замены, кроме блока 1 появится нулевое значение разности. По таблице 10, приведенной в Приложении 1, можно определить, что если на вход S_1 блока замены поступит значение 001100 (а поступит именно это значение, так как значение 60 00 00 00 после перестановки с расширением, согласно таблице 3.5, преобразуется к значению C0 00 00 00 00 00), то с наибольшей вероятностью $p = \frac{14}{64}$ на выходе появится значение разности, равное 1110. Так как перед выходом из F функции выходы S-блоков подвергаются перестановке с помощью таблицы 3.14, то единицы разности, вышедшей из S_1 блока окажутся в 9, 17 и 23 позициях разности на выходе F функции. Таким образом, на выходе функции F первого раунда шифрования алгоритма шифрования DES с вероятностью $p = \frac{14}{64}$ появится значение разности $\Delta A' = 00\ 80\ 82\ 00$.

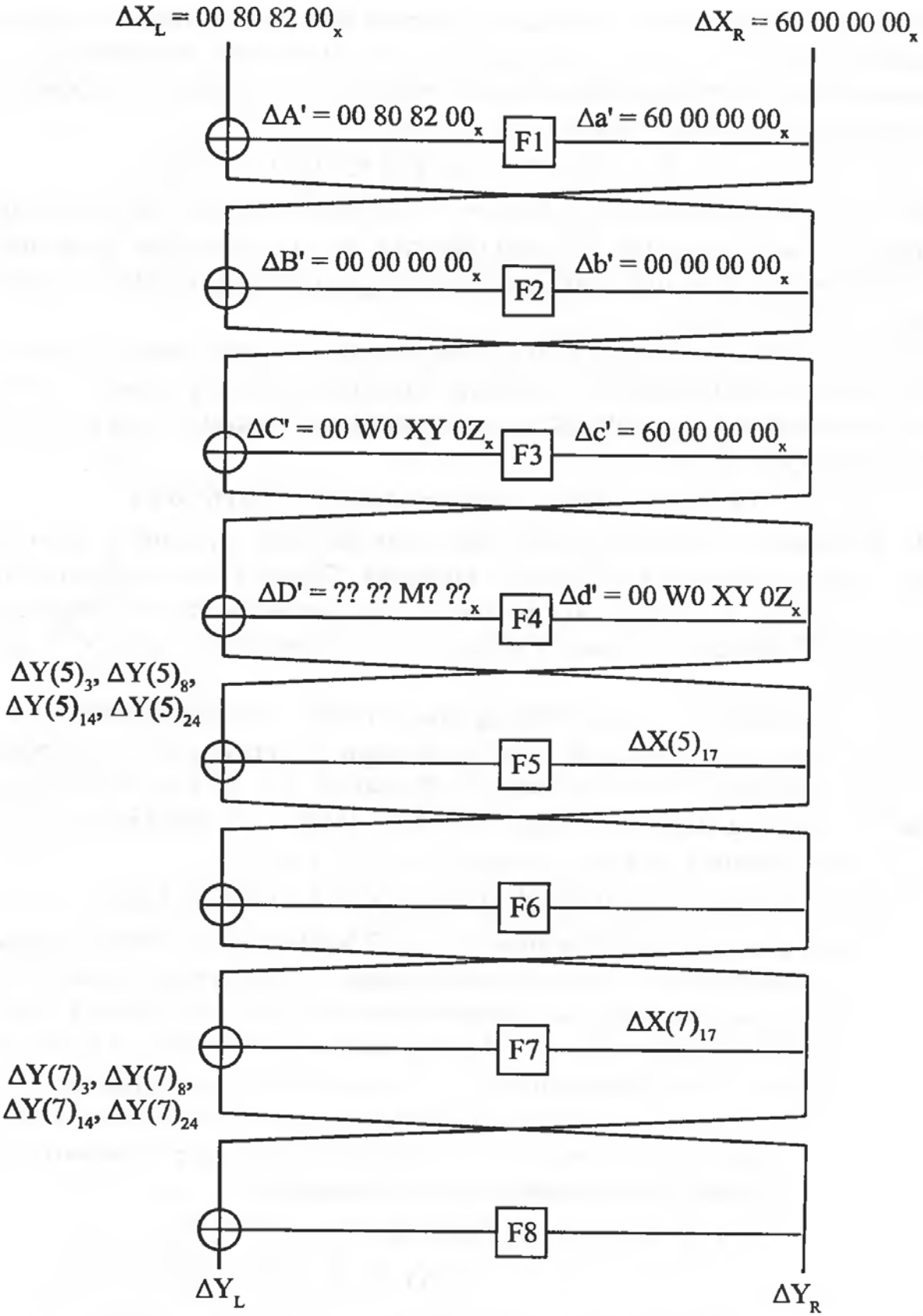


Рис. 7.4. ***Еще один способ применения линейно-дифференциального криптоанализа к 8 раундам алгоритма шифрования DES***

В связи с тем, что на вход функции F второго раунда шифрования поступает разность, равная сумме по модулю два левой части входной разности и выхода функции F первого раунда шифрования, то на вход второго раунда шифрования поступит значение разности, равное: .

$$\Delta b' = \Delta X_L \oplus \Delta A' = 00\ 80\ 82\ 00 \oplus 00\ 80\ 82\ 00 = 00\ 00\ 00\ 00.$$

Мы знаем, что если на вход функции F поступает значение разности, равное нулю, то и на выходе этой функции появится значение разности, также равное нулю. Поэтому на выходе функции F второго раунда шифрования находится значение $\Delta B' = 00\ 00\ 00\ 00$.

На вход функции F третьего раунда шифрования поступает разность, равная сумме по модулю два правой части входной разности и выхода функции F второго раунда шифрования. А значит, на вход третьего раунда шифрования поступит значение разности, равное:

$$\Delta c' = \Delta XR \oplus \Delta B' = 60\ 00\ 00\ 00 \oplus 00\ 00\ 00\ 00 = 60\ 00\ 00\ 00.$$

Мы уже знаем, что в этом случае выходная разность функции F возможно будет иметь единицы в 9, 17, 23 и 31 позициях. Поэтому можно сказать, что на выходе функции F третьего раунда шифрования появится значение разности $\Delta C' = 00\ W0\ XY\ 0Z$, где W и X могут принимать значения 0 или 8, а Y и Z — значения 0 или 2.

На вход функции F четвертого раунда шифрования поступает разность, равная сумме по модулю два входной разности функции F второго раунда шифрования и выхода функции F третьего раунда шифрования. Так как входная разность функции F второго раунда шифрования равна $\Delta b' = 00\ 00\ 00\ 00$, то на вход четвертого раунда шифрования поступит значение разности, равное:

$$\Delta d' = \Delta b' \oplus \Delta C' = 00\ 00\ 00\ 00 \oplus 00\ W0\ XY\ 0Z = 00\ W0\ XY\ 0Z.$$

Ранее мы с вами видели, что в этом случае 17 бит выходной разности функции F всегда останется неизменным, поэтому можно сказать, что на выходе функции F четвертого раунда шифрования появится значение $\Delta D' = ??\ ??\ M?\ ??$, где ? означает любое возможное значение, а M может принимать значения от 0 до 7, то есть 17 бит всегда остается равным нулю.

Уравнение для линейного анализа последних трех раундов шифрования будет аналогично уравнению (7.3) за тем исключением, что в нем будут участвовать значения не четвертого и шестого раундов, а пятого и седьмого:

$$\Delta X(5)_{17} \oplus \Delta Y(5)_3 \oplus \Delta Y(5)_8 \oplus \Delta Y(5)_{14} \oplus \Delta Y(5)_{25} \oplus \Delta X(7)_{17} \oplus \Delta Y(7)_3 \oplus$$

$$\oplus \Delta Y(7)_8 \oplus \Delta Y(7)_{14} \oplus Y\,\Delta(7)_{25} = 0. \quad (7.4)$$

Теперь можно проводить анализ, опираясь на следующий алгоритм.

1. Выбрать $N = 2^{13.81}$ пар открытых текстов, разность которых равна $(\Delta X_L, \Delta X_R) = (00\ 80\ 82\ 00, 60\ 00\ 00\ 00)$.

2. Получить соответствующие этим открытым текстам шифртексты, зашифровав их на одном и том же секретном ключе.
3. Инициализировать массив, состоящий из 64 счетчиков и обнулить их.
4. Для каждой пары шифртекстов выполнить следующие действия:
   — определить 64 возможных значения 6-битового подключа K_1, который складывается с данными, поступающими на вход первого блока замены S_1;
   — для каждого возможного значения подключа вычислить пару выходов S_1 блока восьмого раунда шифрования;
   — второй бит выхода S_1 блока будет являться 17-м выходным битом функции F. Поэтому далее для нахождения значения $\Delta X(8)_{17}$ необходимо сложить по модулю два вторые биты выходов S_1 блока;
   — зная значения ΔX_R (а значит, и значения $\Delta Y(7)_3$, $\Delta Y(7)_8$, $\Delta Y(7)_{14}$, $\Delta Y(7)_{24}$, $\Delta Y(5)_3$, $\Delta Y(5)_8$, $\Delta Y(5)_{14}$, $\Delta Y(5)_{24}$ и $\Delta X(5)_{17}$), вычислить значение $\Delta X(7)_{17}$;
   — если сумма $\Delta X(7)_{17} \oplus \Delta X(8)_{17}$ равна семнадцатому биту разности ΔY_L, то увеличить на единицу значение счетчика, соответствующего этому подключу.
5. Счетчик, имеющий наибольшее значение, укажет правильное значение подключа K_1.
6. Остальные биты ключа могут быть найдены с помощью дополнительного анализа.

Авторы идеи утверждают, что с помощью данного алгоритма при наличии $N = 213{,}81$ пар открытых текстов эта атака будет иметь успех в 77,27% случаев и больше.

В изучении метода линейно-дифференциального криптоанализа ведутся разработки многими учеными. Однако применение этого метода к различным алгоритмам шифрования достаточно сложно и требует детального изучения алгоритма и его свойств. Но нет ничего невозможного, и мы надеемся, что в скором будущем этот метод криптоанализа получит более широкое применение.

7.3. Контрольные вопросы

1. Почему в линейно-дифференциальном криптоанализе рассматриваются пары текстов, у которых имеется различие во втором или третьем, или и втором, и третьем битах правой части?
2. Каким образом строятся уравнения при проведении линейно-дифференциального криптоанализа?
3. За счет чего становится возможным нахождение значения $\Delta X(6)_{17}$ в уравнении (7.3)?

Глава 8. КРИПТОАНАЛИЗ С ПОМОЩЬЮ СЛАЙДОВОЙ АТАКИ

С ростом скорости современных компьютеров скоростные алгоритмы шифрования используют все больше и больше раундов, признавая все существующие криптоаналитические технологии бесполезными. Это главным образом происходит из-за того, что такие популярные методы, как линейный и дифференциальный криптоанализ, являются статистическими атаками, превосходными при статистических непостоянствах. Однако, когда алгоритм шифрования имеет большое количество раундов, каждый добавленный раунд требует экспоненциального роста усилий атакующего.

Стремление к большому числу раундов можно наглядно увидеть, рассмотрев претендентов на конкурс AES. Несмотря на то, что одним из основных критериев для претендентов была скорость, некоторые представленные кандидаты (при этом не самые медленные) имели действительно большое число раундов: RC6(20), MARS(32), SERPENT(32), CAST(48). Это является следствием того, что после некоторого большого числа раундов даже относительно слабый шифр становится стойким. Например, у алгоритма шифрования DES уже взлом шестнадцати раундов представляет трудную задачу, не говоря о 32 и 48 раундах (двойном и тройном алгоритме DES). Таким образом, для криптоаналитика становится естественным поиск новых методов анализа, не зависящих от числа раундов в алгоритме шифрования.

В этой главе мы рассмотрим новый метод криптоанализа, не зависящий от числа раундов в алгоритме шифрования, который называется «слайдовой атакой», или «скользящей атакой» (Slide Attacks), предложенный в 1999 г. эмигрантом из России Алексом Бирюковым и известным американским криптографом Дэвидом Вагнером [21, 37]. Этот метод применим ко всем алгоритмам блочного шифрования.

В то время как два других метода криптоанализа, такие как линейный и дифференциальный, концентрируются главным образом на распространенных свойствах техники шифрования, слайдовая атака использует степень самоподобия, что является принципиальным отличием. Под самоподобием понимается использование одной и той же криптографической F-функции, зависящей от одного и того же подключа, в каждом раунде шифрования. В зависимости от структуры алгоритма шифрования слайдовая атака может использовать как слабость процедуры формирования подключей, так и более общие структурные свойства шифра. Самый простой вид этой атаки обычно легко пресечь, избавившись от самоподобия в алгоритме шифрования. Более сложные варианты этой атаки (см. пп. 8.3 и 8.4) имеют более сложный анализ, и против них гораздо труднее защититься.

Самый простой вариант слайдовой атаки рассчитан на анализ алгоритмов шифрования, состоящих из r раундов, каждый из которых содержит в себе F-фун-

кцию, зависящую от одного и того же значения ключа K. Такой тип алгоритмов шифрования называется **гомогенным**. К гомогенным также относятся алгоритмы, в которых функция формирования подключа периодична, то есть один и тот же подключ извлекается через равное количество раундов. Говоря математическим языком, $F_i = F_j$ для всех $i \equiv j \bmod p$, где p является периодом. В самом простом случае $p = 1$, и в каждом раунде используется один и тот же подключ.

При изучении слайдовых атак с целью упрощения мы будем вначале рассматривать алгоритмы шифрования, в которых сложение шифруемых данных с подключом происходит непосредственно перед F-функцией. Так, если мы рассматриваем алгоритм шифрования, построенный по схеме Фейстеля, на вход которого поступает n-битное сообщение, то длина подключа будет составлять n/2 битов. Анализ более сложных алгоритмов будет рассмотрен позже.

8.1. Обычная слайдовая атака

На рис. 8.1 показан процесс зашифрования n-битового открытого текста X_0, в результате которого получается шифр-текст X_r. Здесь X_j обозначает промежуточное значение данных после j-го раунда зашифрования, так что

$$X_j = F_j(X_{j-1}, k_j),$$

где $j = 1, 2, 3, \ldots r$. В дальнейшем мы иногда будем опускать значение k в обозначении F-функций и будем писать F(x), или $F_i(x)$, вместо F(x, k), или $F_i(x, k)$.

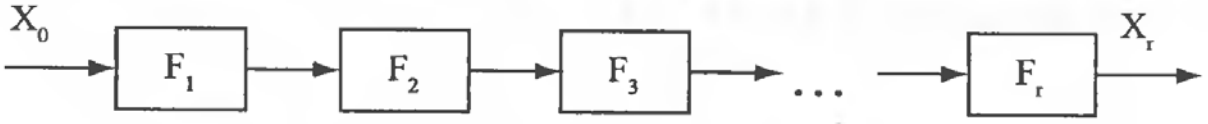


Рис. 8.1. ***Схема обычного блочного алгоритма шифрования***

Функция F называется **слабой** в том случае, если при известных двух равенствах $F(x_1,k) = y_1$ и $F(x_2,k) = y_2$ ключ k легко определяется. На рис. 8.2 показано, как может быть применена слайдовая атака к алгоритмам шифрования такого типа.

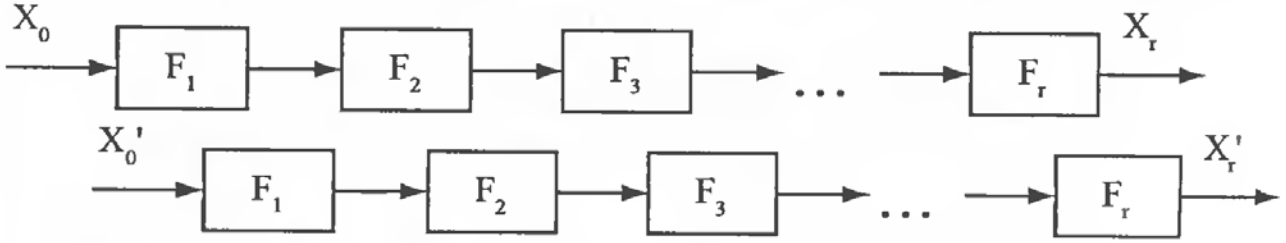


Рис. 8.2. ***Схема обычной слайдовой атаки***

Идея заключается в том, что можно сопоставить один процесс зашифрования с другим таким образом, что один из процессов будет «отставать» от другого на один раунд.

Пусть X_0 и X_0' обозначают исходные открытые тексты, $X_j = F_j(X_{j-1})$ и $X_j' = F_j(X_{j-1}')$, где j = 1, 2, 3,.., r. Идея заключается в том, что если мы имеем такую пару значений, что $X_1 = X_0'$, то мы также будем иметь соответствующую им пару значений, такую что $X_r = X_{r-1}'$. Предположим, что $X_j = X_{j-1}'$, тогда мы можем сказать, что $X_{j+1} = F(X_j) = F(X_{j-1}') = X_j'$. Пара открытых текстов и соответствующих им шифртекстов (P,C), (P′,C′) называется **слайдовой парой** в том случае, если F(P) = P′ и F(C) = C′.

Слайдовая атака проходит следующим образом. Мы получаем $2^{n/2}$ пар *открытый—закрытый текст* (P_i, C_i) и ищем среди них слайдовые пары. Согласно парадоксу «Дней Рождений», мы ожидаем найти хотя бы одну пару индексов i, i′, такую, что $F(P_i) = P_i'$ и $F(C_i) = C_i'$ одновременно выполняются для некоторого ключа. После того как слайдовая пара найдена, мы можем найти некоторые биты подключа. В том случае, если раундовая функция слабая, мы можем найти весь подключ данного раунда. Для нахождения оставшихся битов секретного ключа необходимо определить следующую слайдовую пару, и с ее помощью провести анализ. Таким образом, достаточно найти всего несколько слайдовых пар для полного определения битов секретного ключа, что и является задачей, стоящей перед криптоаналитиком.

8.2. Объемы требуемой для анализа информации при различных типах атак

8.2.1. Атака на основе известного открытого текста

В случае, когда речь идет об алгоритмах шифрования, построенных по схеме Фейстеля, раундовая функция F((l, r), k) = ((l ⊕ f(r), r), k) модифицирует только половину входного сообщения. Таким образом, условие F(x) = x′ можно легко проверить с помощью сравнения левой части сообщения x и правой части сообщения x′. Это условие позволяет нам снизить сложность атаки на основе известных открытых текстов до $2^{n/2}$ известных текстов. У нас есть n-битовое условие нахождения потенциальной слайдовой пары: если (P_i, C_i) образует слайдовую пару вместе с (P_i', C_i'), то тогда $F(P_i) = P_j'$ и $F(C_i) = C_j'$. Для нахождения слайдовой пары для алгоритмов шифрования, построенных по схеме Фейстеля, необходимо известные тексты (P_i, C_i) занести в таблицу, после чего для каждого j найти такой текст, чтобы правые половины P_i и C_j' были равны левым половинам P_j' и C_i.

Если не все биты подключа будут найдены с помощью определенной слайдовой пары, то можно будет использовать другие слайдовые пары для определения оставшихся битов.

8.2.2. Атака на основе выбранного открытого текста

Для алгоритмов шифрования, построенным по схеме Фейстеля, сложность анализа может быть снижена до $2^{n/4}$ текстов в том случае, если существует возможность использовать выбранные открытые тексты. Для этого необходимо выбрать произвольным образом n/2 битовое значение x. После этого надо подобрать массив из $2^{n/4}$ открытых текстов $P_i = (x, y_i)$, которые будут различаться только случайно выбранной правой частью, и массив из $2^{n/4}$ текстов $P_j' = (y_j', x)$, которые будут различаться только случайно выбранной левой частью. Таким образом, у нас появится $2^{n/2}$ пар открытых текстов, и мы надеемся найти среди них хотя бы одну правильную пару.

8.3. Улучшенная слайдовая атака

8.3.1. Слайдовая атака с использованием дополнений (Complementation slide)

В этом разделе будет продемонстрировано несколько способов расширения простой слайдовой атаки для применения ее к большему классу алгоритмов шифрования. Сначала рассмотрим метод, позволяющий анализировать алгоритм шифрования, построенный по схеме Фейстеля, с двухраундовым самоподобием (то есть когда в алгоритме используются два постоянно чередующихся подключа K_0 и K_1, как показано на рис. 8.3).

При применении обычной слайдовой атаки к алгоритму с двухраундовым самоподобием логично было бы сопоставлять два процесса зашифрования с «запаздыванием» на два раунда, однако в этом случае атака является неэффективной.

Существует возможность сопоставить процессы зашифрования с «запаздыванием» на один раунд, если ввести разность $\Delta = K_0 \oplus K_1$ между двумя процессами зашифрования в каждом раунде. В этом случае мы переходим от двухраундового самоподобия к одноraундовому самоподобию. Но вместе с этим, мы получаем разность между раундами зашифрования в слайдовых парах.

Для проведения атаки необходимо выбрать такую слайдовую пару, чтобы разность открытых текстов компенсировала разность подключей. Для этого необходимо найти открытые тексты, имеющие слайдовую разность (Δ, Δ). Очевидно, что пара открытых текстов P и P′ имеет слайдовую разность Δ в том случае, если $F(P) \oplus P' = \Delta$. Такая слайдовая разность будет переходить из раунда в раунд с вероятностью $p = 1$ и таким образом приведет к такой же разности шифртекстов. На рис. 8.3 наглядно показано применение этого метода криптоанализа.

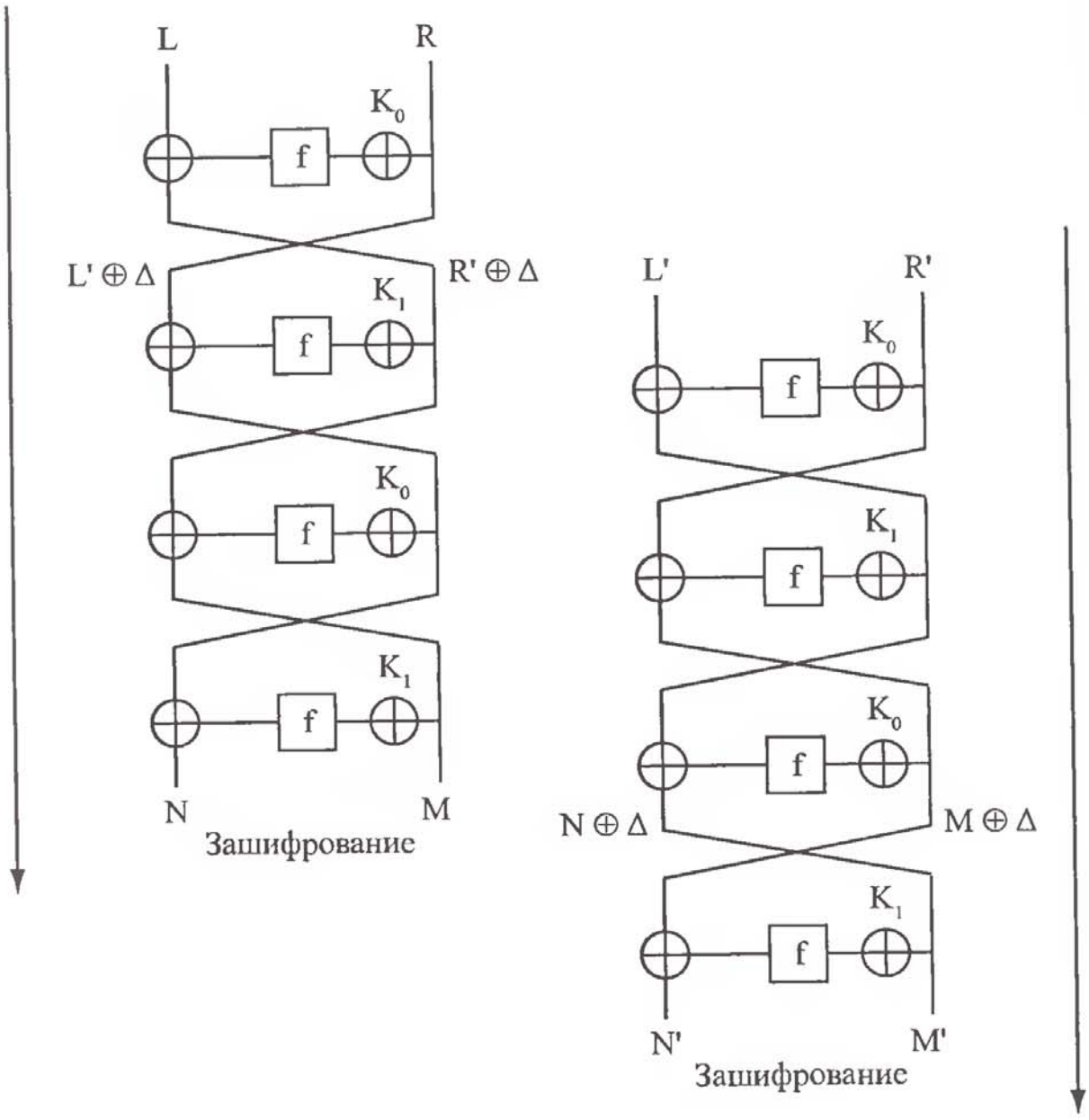


Рис. 8.3. ***Слайдовая атака с использованием дополнений***

Как и в предыдущих случаях, слайдовая пара может быть выделена из массива, состоящего из $2^{n/2}$ открытых текстов. Если мы обозначим открытый текст как P = (L, R) и шифртекст как C = (N, M), то получим следующие равенства:

$$(\mathbf{L}', R') = (\mathbf{R}, L \oplus f(K0 \oplus R)) \oplus (\Delta, \Delta); \tag{8.1}$$

$$(N', \mathbf{M}') = (M \oplus f(K_1 \oplus N \oplus \Delta), N) \oplus (\Delta, \Delta). \tag{8.2}$$

Из приведенных уравнений видно, что $L' = R \oplus \Delta$ и $M' = N \oplus \Delta$. Таким образом, получается, что $L' \oplus M' = R \oplus \Delta \oplus N \oplus \Delta = R \oplus N$, что и является n/2-битовым условием определения слайдовой пары. Найденная слайдовая пара даст нам значения L, R, M, N, L′, R′, M′ и N′. Мы знаем, что $R = L \oplus \Delta$ (см. рис. 8.3), а значит, можем определить значение Δ: $\Delta = R \oplus L$. Сложив по модулю два значения

$L \oplus \Delta \oplus R'$, получим значение на выходе функции F первого раунда шифрования первой слайдовой пары. Так как нам известны преобразования функции F, и они не зависят от ключа, то мы можем предположить, какое значение могло поступать на вход функции F первого раунда шифрования первой слайдовой пары. А так как мы знаем, что на вход этой функции поступает сумма значений $R \oplus K_0$, то это позволит нам предположить значение K_0. Аналогичные действия необходимо провести с последним раундом шифрования второй слайдовой пары. Там на выходе функции F окажется сумма $M \oplus \Delta \oplus N'$. На вход же этой функции поступает значение $M' \oplus K_1$. А значит, мы можем предположить возможные значения K_1. Из предположенных ранее значений K_0 и K_1 верными будут те, сумма которых по модулю два даст значение Δ. Таким образом, главной задачей является нахождение правильной слайдовой пары, последующий же анализ не представляет особых трудностей и протекает довольно легко.

8.3.2. Слайдовая атака с петлей (Sliding with a Twist)

В этом разделе рассмотрим еще один метод слайдовой атаки для алгоритмов шифрования, построенных по схеме Фейстеля, с двухраундовым самоподобием.

Если не принимать во внимание начальную и конечную перестановки, часто используемые в алгоритмах блочного шифрования, то можно сказать, что для алгоритма шифрования, построенного по схеме Фейстеля, процесс дешифрования с использованием подключей K_0, K_1 будет аналогичен процессу зашифрования с использованием подключей K_1, K_0. Также можно сказать, что процесс зашифрования с помощью алгоритмов, построенных по схеме Фейстеля, с использованием подключей K_0, K_1 очень близок к процессу зашифрования тем же самым алгоритмом с использованием подключей K_1, K_0: просто один процесс «запаздывает» относительно другого на один раунд. Поэтому мы можем сопоставить процессы зашифрования и дешифрования таким образом, что один из них будет «запаздывать» на один раунд. В этом случае слайдовые пары будут иметь частичное совпадение во всех раундах шифрования, за исключением первого раунда зашифрования и последнего раунда дешифрования. На рис. 8.4 показано, как необходимо сопоставить два этих процесса.

Криптоанализ начинается с создания массива из $2^{n/2}$ открытых текстов для того, чтобы мы могли найти, по крайней мере, одну слайдовую пару. Для искомой слайдовой пары должны выполняться равенства:

$$(N', M') = (R, L \oplus f(K_0 \oplus R)), \tag{8.3}$$

$$(L', R') = (M \oplus f(K_0 \oplus N), N). \tag{8.4}$$

Уравнения (8.3) и (8.4) дают нам n-битовое условие отбора слайдовой пары (то есть $N' = R$ и $R' = N$). Найдя слайдовую пару, мы легко сможем определить биты подключа K_0 тем же способом, который был описан в предыдущем разделе. Подключ K_1 может быть получен путем слайдовой атаки с «запаздыванием»

на два раунда шифрования. При этом для анализа можно использовать ранее созданный массив текстов.

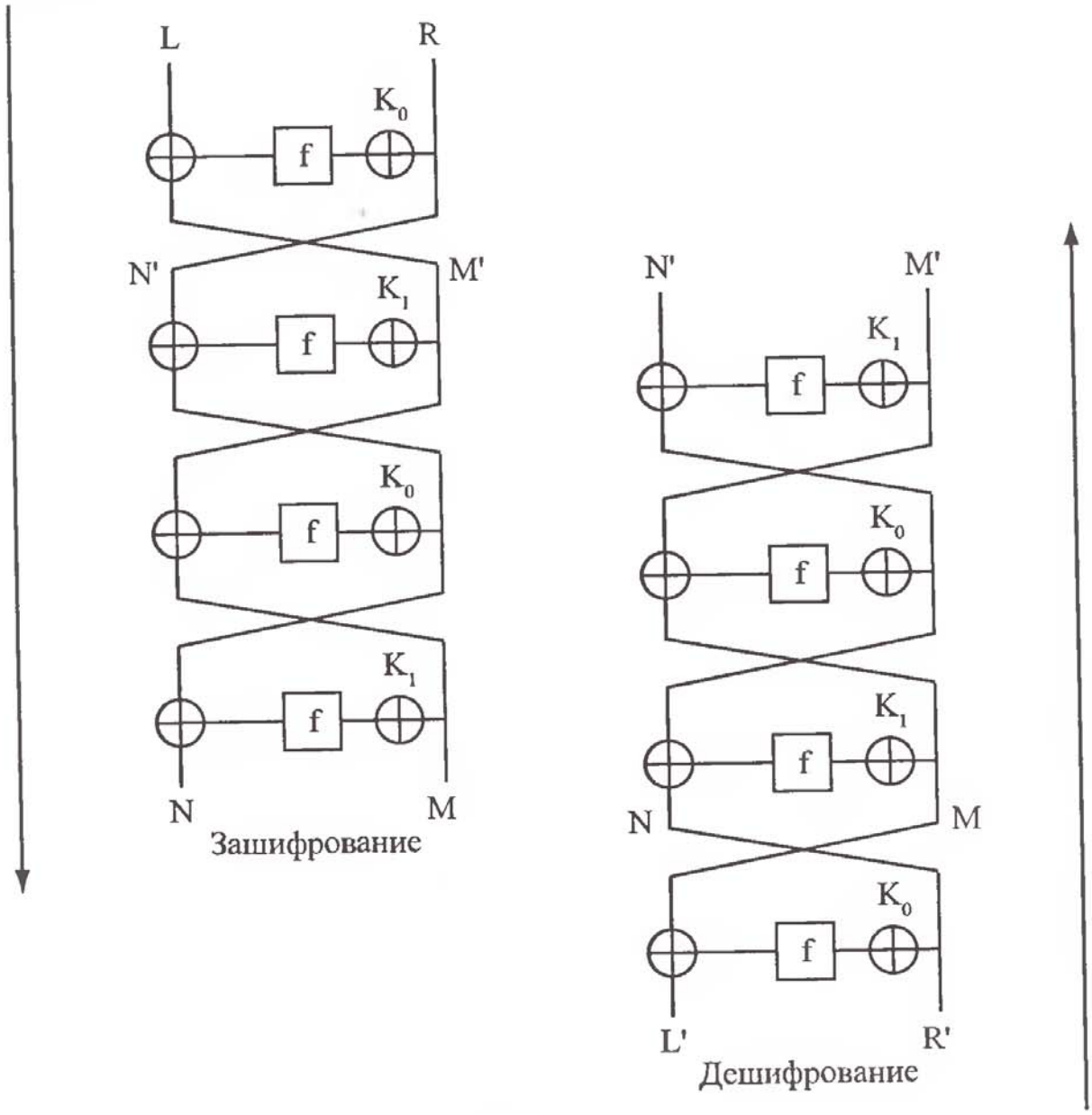


Рис. 8.4. ***Слайдовая атака с петлей***

Существует также возможность проводить атаку на основе выборочных пар *открытый—закрытый текст*. В этом случае число требуемых текстов снижается до $2^{n/4}$. При этом необходимо сгенерировать массив из $2^{n/4}$ открытых текстов (L_i, R), имеющих различие только в левой половине текста, и получить соответствующие им значения шифртекстов. Так же необходимо сгенерировать массив из $2^{n/4}$ шифртекстов текстов (M_j', N'), где N' = R и где имеются различия только в левой половине текста, а также получить соответствующие им значения открытых текстов. Имея такое количество открытых текстов и шифртекстов, мы можем найти

как минимум одну слайдовую пару. Дальнейший анализ будет аналогичен предыдущему.

8.4. Применение методов слайдовой атаки к алгоритмам шифрования с четырехраундовым самоподобием

На практике часто приходится пользоваться ни каким-либо одним методом анализа, а комбинацией методов, позволяющей достичь цели с минимальными затратами. В этом пункте мы рассмотрим анализ алгоритма шифрования, построенного по схеме Фейстеля, с четырехраундовым самоподобием (4K-Feistel шифр), основанный на комбинации методов слайдовой атаки с использованием дополнений и слайдовой атаки с петлей.

Конечно, существует вариант, когда можно просто сопоставить процессы зашифрования с «запаздыванием» на два раунда следующим образом:

$$K_0\ K_1\ K_2\ K_3\ K_0\ K_1 \ldots$$
$$K_0\ K_1\ K_2\ K_3\ K_0\ K_1 \ldots$$

А после этого применить слайдовую атаку с использованием дополнений ($K_1 \oplus K_3$, $K_0 \oplus K_2$). Однако для проведения такого анализа требуется никак не меньше $2^{n/2}$ текстов, да и сам процесс анализа довольно трудоемок [37].

Лучшего результата можно достигнуть, применив слайдовую атаку с петлей. Этот метод атаки применим в том случае, если одновременно объединить его со слайдовой атакой с использованием дополнений следующим образом:

$$K_0 K_1\ K_2\ K_3\ K_0\ K_1\ K_2\ K_3\ K_0 \ldots$$
$$K_3\ K_2\ K_1\ K_0\ K_3\ K_2\ K_1\ K_0\ K_3 \ldots$$

Здесь верхний ряд соответствует процессу зашифрования, а нижний — процессу дешифрования (или процессу зашифрования с последовательным использованием подключей K_3, K_2, K_1 и K_0).

Теперь можно заметить, что появилась возможность применения слайдовой атаки с использованием дополнений, при получении текстов, имеющих слайдовую разность $(0, K_1 \oplus K_3)$. Как видно из рис. 8.5, такой выбор разностей между слайдовыми парами обусловлен тем, что во втором раунде шифрования первой слайдовой пары, а также в предпоследнем раунде шифрования второй слайдовой пары используется подключ K_1, в то время, как в первом раунде шифрования второй слайдовой пары и в последнем раунде первой слайдовой пары, используется подключ K_3. После этого становится возможным проведение анализа по схеме, приведенной в п. 8.3.1. При анализе, чтобы не запутаться, обращайтесь к рис. 8.5.

Такой подход к анализу алгоритма позволит определить значение подключа K_0, а также значение Δ, позволяющее предположить значения подключей K_1 и K_3. В любом случае выбранный криптоаналитиком метод для анализа требует от него творческого подхода и фундаментальных знаний.

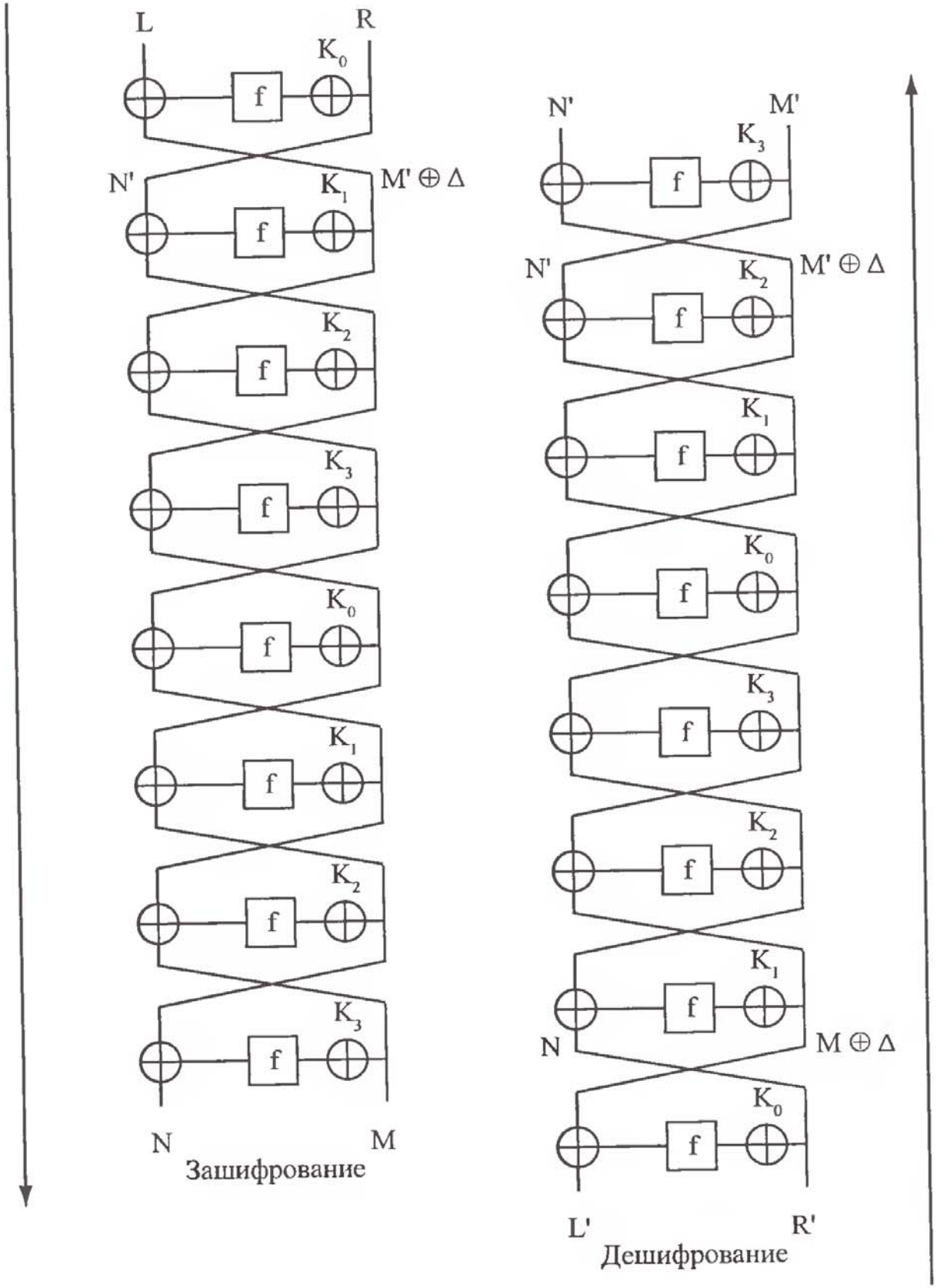


Рис. 8.5. ***Комбинирование методов слайдовой атаки***

8.5. Применение обычной слайдовой атаки к алгоритму шифрования S-DES

Для того чтобы убедиться на практике в действенности метода слайдовой атаки, рассмотрим ее применение к алгоритму шифрования S-DES. Алгоритм S-DES был разработан в учебных целях и имеет длину открытого блока 8 бит (более подробное описание алгоритма приведено в п. 3.1.1). Так как алгоритм S-DES является блочным алгоритмом шифрования, построенном по схеме Фейстеля, то мы вполне можем применить к нему анализ на основе выборочного открытого текста. Для простоты работы будем использовать один и тот же фиксированный 8-битовый ключ K (то есть будем опускать процедуру получения 8-битового подключа из исходного 10-битового). Но при этом сложение шифруемых данных с ключом будет происходить не перед F-функцией, а непосредственно в ней, как это и предусмотрено алгоритмом. Также мы опустим начальную и конечную перестановки, так как они не влияют на криптографическую стойкость алгоритма, и будем использовать не 2 цикла криптографического преобразования, а 20.

Сначала выберем произвольным образом 4-битовое значение x. Пусть x = 1101. После этого подберем массив из $2^{n/4} = 2^{8/4} = 2^2 = 4$ открытых текстов $P_i = (x = X\_L, y_i = X\_R)$, которые будут различаться только случайно выбранной правой частью, и массив из $2^{n/4} = 2^{8/4} = 2^2 = 4$ открытых текстов $P_j' = (y_j' = X\_L', x = X\_R')$, которые будут различаться только случайно выбранной левой частью. После этого зашифруем выбранные тексты на случайно выбранном 8-битовом ключе $K = (K_1, K_2)$, состоящем из двух 4-битовых частей (каждая из которых влияет на вход S-блока). Результаты зашифрования массивов $P_i = (x, y_i)$ и $P_j' = (y_j', x)$ приведены соответственно в таблицах 8.1 и 8.2.

Таблица 8.1

Результаты зашифрования массива $P_i = (x, y_i)$ на секретном ключе K

№	X_L	X_R	Y_L	Y_R
1	1101	0110	0101	1100
2	1101	0001	1101	1110
3	1101	0010	1001	1010
4	1101	1000	0000	0111

Итак, у нас есть 16 пар открытых текстов. Проанализировав результаты шифрования, приведенные в таблицах 8.1 и 8.2, определяем, что нашему условию определения слайдовых пар удовлетворяют две пары текстов.

Так как при анализе найденных пар нам придется работать с таблицами замены, используемыми в алгоритме шифрования S-DES, то для удобства работы мы сопоставили входы и выходы блоков так, как показано в таблице 8.3.

Таблица 8.2

Результаты зашифрования массива $P_j' = (y_j', x)$ на секретном ключе K

№	X_L′	X_R′	Y_L′	Y_R′
1	1010	1101	1010	1101
2	0010	1101	0101	1111
3	1110	1101	1100	1000
4	1100	1101	0000	0011

Таблица 8.3

Соответствие входов и выходов S-блоков для алгоритма шифрования S-DES

Вход в S-блок	Выход S_0 блока	Выход S_1 блока
0000	01	01
0001	11	10
0010	00	01
0011	10	00
0100	11	10
0101	01	01
0110	10	11
0111	00	11
1000	00	11
1001	11	10
1010	10	00
1011	01	01
1100	01	01
1101	11	00
1110	11	00
1111	01	11

Первую пару текстов составляют текст № 1 в таблице 8.1 и текст № 3 в таблице 8.2. Проанализируем найденную слайдовую пару. Для этого рассмотрим первые два раунда шифрования, показанные на рис. 8.6.

То, что нам известны значения X_R′ и X_L, дает нам информацию о значении входа F_1' функции. Так как значения X_R и X_L′ тоже известны, то легко можно определить значение выхода F-функции, которое будет равно 1000.

Так как перед выходом из F-функции данные подвергаются перестановке, согласно таблице 3.23, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 0100, то есть 01 является выходом S_0 блока, а 00 — выходом S_1 блока.

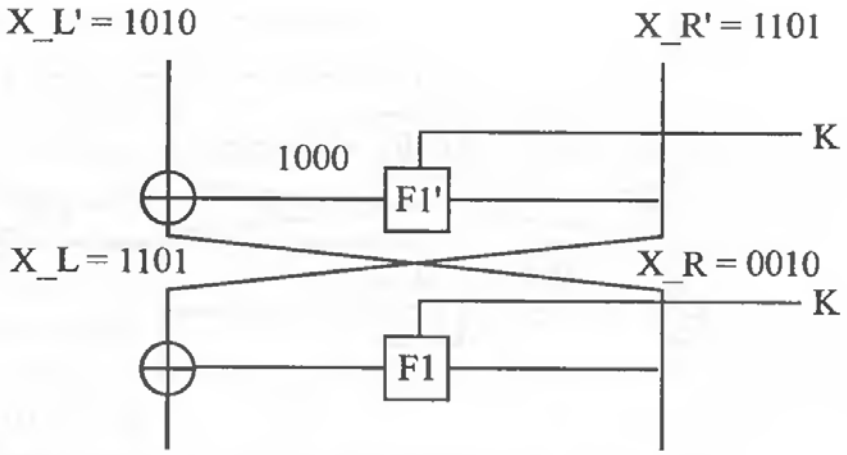


Рис. 8.6. ***Анализ первых раундов первой слайдовой пары***

Входное сообщение F-функции подвергается перестановке с расширением, согласно таблице 3.20. А значит, вход F-функции 1101 преобразуется к значению 11101011, которое и будет сложено с ключом $K = (K_1, K_2)$, то есть получается, что вход S_0 блока $1110 \oplus K_1$ даст на выходе значение 01, а вход S_1 блока $1011 \oplus K_2$ — значение 00.

Воспользовавшись таблицей 8.3, можно определить, что значение 01 появляется на выходе S_0 блока в случае, если на его вход поступило одно из следующих значений: 0000, 0101, 1011, 1100 или 1111. Таким образом, добавив к каждому из возможных значений входа значение 1110, получим возможные значения K_1. Это будут значения 1110, 1011, 0101, 0010 или 0001.

Аналогичным образом определяем, что значение 00 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих значений: 0011, 1010, 1101 или 1110. А значит, добавив к каждому из возможных значений входа значение 1011, получим возможные значения K_2. Это будут значения 1000, 0001, 0110 или 0101.

Теперь рассмотрим последние два раунда шифрования для этой же слайдовой пары, показанные на рис. 8.7.

То, что нам известны значения Y_L′ и Y_R, дает нам информацию о значении входа функции F_{20}′. Так как значения Y_R′ Y_L тоже известны, то легко можно определить значение выхода этой же F-функции, которое будет равно 0100.

Так как перед выходом из F-функции данные подвергаются перестановке, согласно таблице 3.23, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 0001, то есть 00 будет являться выходом S_0 блока, а 01 — выходом S_1 блока.

Входное сообщение F-функции подвергается перестановке с расширением, согласно таблице 3.20. А значит, вход F-функции 1010 преобразуется к значению 01010101, которое и будет сложено с ключом $K = (K_1, K_2)$, то есть получается, что вход S_0 блока $0101 \oplus K_1$ даст на выходе значение 00, а вход S_1 блока $0101 \oplus K_2$ — значение 01.

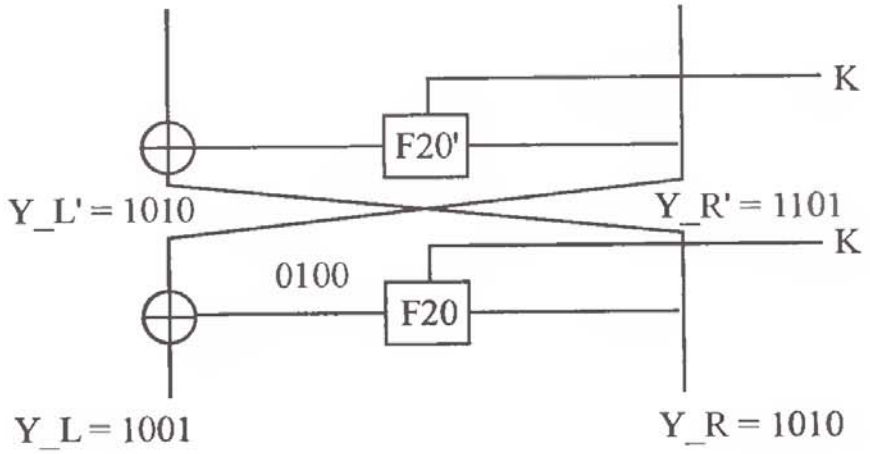


Рис. 8.7. ***Анализ последних раундов первой слайдовой пары***

Воспользовавшись таблицей 8.3, можно определить, что значение 00 появляется на выходе S_0 блока в случае, если на его вход поступило одно из следующих значений: 0010, 0111 или 1000. Таким образом, добавив к каждому из возможных значений входа значение 0101, получим возможные значения K_1. Это будут значения 0111, 0010 или 1000.

Аналогичным образом определяем, что значение 01 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих значений: 0000, 0010, 0101, 1011 или 1100. А значит, добавив к каждому из возможных значений входа значение 0101, получим возможные значения K_2. Это будут значения 0101, 0111, 0000, 1110 или 1001.

Так как во всех раундах шифрования использовался один и тот же ключ, то для первого и последнего раундов должны совпадать значения K_1 и K_2. Сопоставив все возможные значения K_1 и K_2, можно увидеть, что есть только одно значение $K_1 = 0010$, использование которого возможно как в первом, так в последнем раундах, и только одно значение $K_2 = 0101$, использование которого также возможно как в первом, так и в последнем раундах. Таким образом, нами найден искомый ключ K = 00100101. Убедиться в его правильности можно, зашифровав на нем сгенерированные нами открытые тексты и сравнив результаты шифрования.

Вторая слайдовая пара состоит из текста № 3 в таблице 8.1 и текста № 1 в таблице 8.2. Анализ этой пары текстов, приводит к тому же результату. Вы можете самостоятельно провести его и убедиться в этом.

8.5. Контрольные вопросы

1. Какие условия должны выполняться для того, чтобы стало возможным проведение обычной слайдовой атаки?
2. В каком случае функция F называется слабой?

3. Опишите алгоритм проведения криптоанализа с помощью обычной слайдовой атаки.
4. Какая пара текстов называется «слайдовой парой»?
5. Какой критерий отбора слайдовых пар используется при анализе алгоритмов шифрования, построенных по схеме Фейстеля?
6. Опишите алгоритм проведения криптоанализа шифра, построенного по схеме Фейстеля, с помощью слайдовой атаки на основе выбранного открытого текста.
7. В чем заключается смысл слайдовой атаки с использованием дополнений?
8. В чем заключается смысл слайдовой атаки с петлей?
9. Что понимается под самоподобием алгоритма шифрования?
10. Как слайдовая атака может быть применена к алгоритму шифрования с четырехраундовым самоподобием?

Глава 9. ЛАБОРАТОРНО-ПРАКТИЧЕСКИЕ РАБОТЫ

Данный курс лабораторно-практических работ предназначен для изучения современных методов криптоанализа блочных шифров, таких как линейный, дифференциальный, линейно-дифференциальный криптоанализ и слайдовая атака. Прежде чем приступить к выполнению лабораторных работ, необходимо ознакомиться с алгоритмами шифрования, на примере которых будет рассматриваться применение вышеперечисленных методов криптоанализа. В первых двух лабораторных работах будет использоваться специально разработанный Учебный Алгоритм Шифрования, вторые две лабораторные работы посвящены анализу алгоритма, построенного на основе сети SPN, и, наконец, последняя лабораторная работа посвящена анализу алгоритма шифрования S-DES. Ниже приведено подробное рассмотрение всех трех алгоритмов. На сайте кафедры БИТ Таганрогского государственного радиотехнического университета http://bit.tsure.ru представлены все, описанные в этом разделе, лабораторные работы.

9.1. Учебный Алгоритм Шифрования, предназначенный для проведения лабораторных работ по изучению методов линейного и дифференциального криптоанализа применительно к алгоритмам блочного шифрования, построенным по схеме Фейстеля

Общая схема Учебного Алгоритма Шифрования приведена на рис. 9.1. Входной блок данных представляет собой последовательность из 16 бит. Первоначально входной блок данных разделяется на две части, после чего правая часть подвергается преобразованию с помощью функции F. Выход функции F складывается по модулю 2 с левой частью входных данных. Выходное шифрованное сообщение состоит из двух 8-битовых частей (то есть всего 16 бит), правая часть которого представляет собой правую часть входного сообщения, а левая — результат сложения по модулю два левой части входного сообщения и выхода функции F.

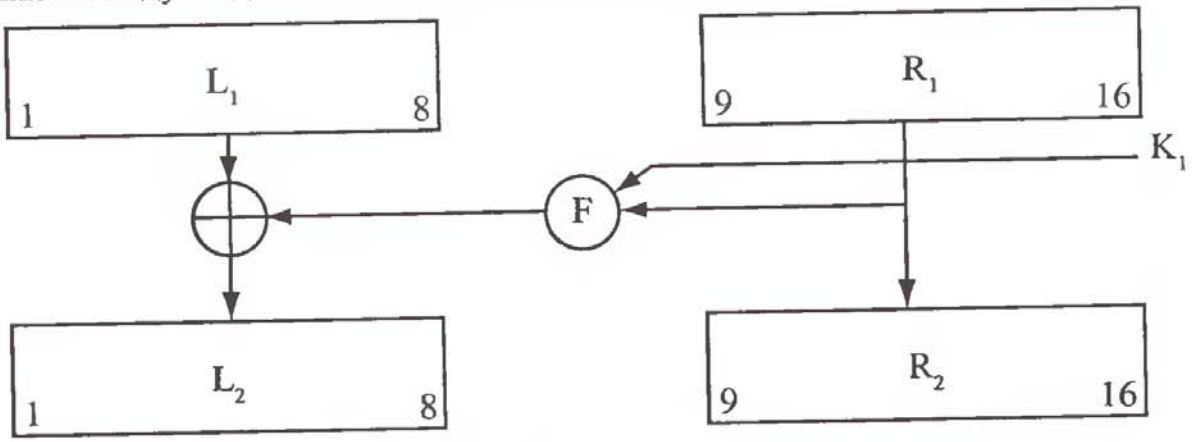


Рис. 9.1. *Общая схема Учебного Алгоритма Шифрования*

Рассмотрим подробнее работу функции F (см. рис. 9.2). Входное 8-битовое сообщение проходит через блок перестановки с расширением так, что на выходе получается 12 бит. Данные таблицы перестановки с расширением приведены в таблице 9.1. Таблицу 9.1 следует читать слева направо, то есть 3 бит станет первым, 4 бит станет вторым, 1 — третьим и т. д. После этого 12-битовый ключ и преобразованное входное сообщение складываются по модулю 2. Результат сложения разбивается на три группы по четыре бита, каждая из которых проходит соответственно Блок1, Блок2 и Блок3. При этом из первых двух блоков выходит по три бита, а из третьего блока — два.

Замена в первых двух блоках производится по приведенным таблицам замены (соответственно, таблицы 9.2 и 9.3) по следующему принципу. Пусть на вход блока поступает четыре бита a_1, a_2, a_3, a_4. При этом первый бит определяет номер строки (если $a_1 = 0$, то это соответствует первой строке, а если $a_1 = 1$ — второй), а остальные три — номер столбца (000 = первому столбцу, 111 = восьмому столбцу).

Несколько иначе дело обстоит с третьим блоком. Из него в отличие от первых двух выходит не три бита, а два. Замена осуществляется, согласно таблице 9.4. При этом, если a_1, a_2, a_3, a_4 — входные биты Блока3, то биты a_1 и a_4 определяют номер сроки, а биты a_2 и a_3 — номер столбца (аналогично S-блокам алгоритма DES).

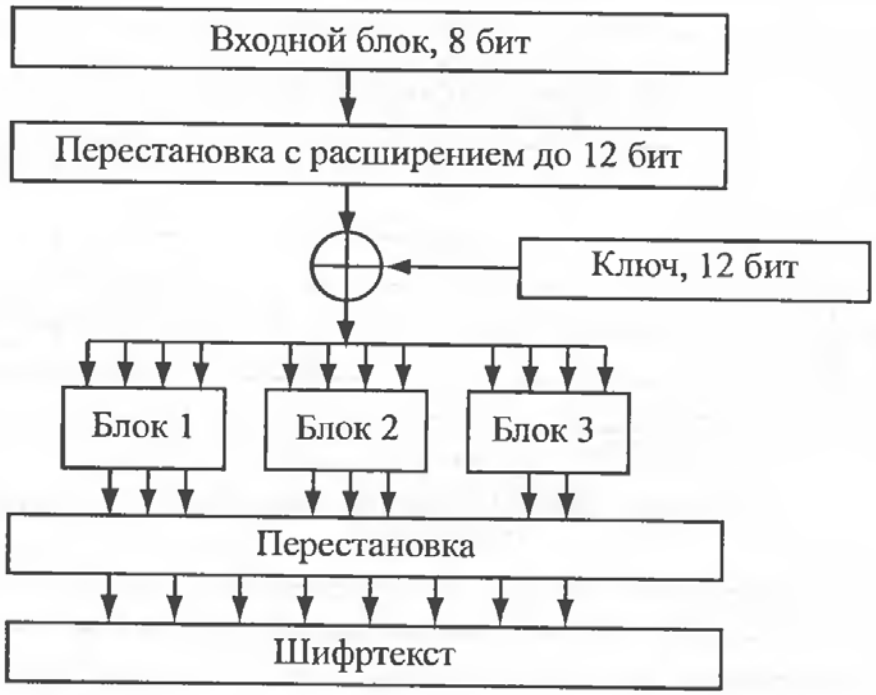


Рис. 9.2. ***Функция F Учебного Алгоритма Шифрования***

После этого восьмибитовое сообщение претерпевает перестановку, согласно таблице 9.5, и на выходе получается готовый выход функции F.

Таблица 9.1

Таблица перестановки с расширением

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

Таблица 9.2

Таблица замены в блоке 1

a2a3a4 / a1	000	001	010	011	100	101	110	111
0	4	6	1	3	5	7	2	5
1	5	7	2	4	6	1	3	6

Таблица 9.3

Таблица замены в блоке 2

a2a3a4 / a1	000	001	010	011	100	101	110	111
0	3	5	7	2	4	6	1	7
1	4	6	1	3	5	7	2	1

Таблица 9.4

Таблица замены в блоке 3

a2a3 / a1 a4	00	01	10	11
00	1	3	2	1
01	2	1	3	2
10	3	2	1	3
00	1	3	2	1

Таблица 9.5

Таблица перестановки

8	7	3	2	5	4	1	6

Необходимо заметить, что таблицы, используемые в алгоритме шифрования, задаются каждому варианту отдельно. Неизменной из всех таблиц является таблица перестановки (таблица 9.5). Таблица перестановки с расширением в первой лабораторной работе для всех вариантов одинаковая, а для второй лабораторной работы задается каждому варианту индивидуально.

9.2. Алгоритм Шифрования, предназначенный для проведения лабораторных работ по изучению методов линейного и дифференциального криптоанализа применительно к алгоритмам шифрования, построенным по принципу сети SPN

Шифр, который мы будем рассматривать, является разновидностью шифровальной сети, построенной на основе подстановок и перемешивания (Substitution — Permutation Network (SPN)). Алгоритм этого шифра приведен на рис. 9.3. Данный шифр является блочным алгоритмом шифрования и оперирует 9-битовыми входными и выходными блоками данных. Входной блок данных обрабатывается с помощью повторения трех циклов, включающих простейшие операции. Каждый цикл состоит из замены и перестановки битов, а также сложения с ключом. Используемые в алгоритме простейшие операции аналогичны тем, которые используются в алгоритме шифрования DES, а также во многих других современных шифрах, включая Rijndael.

Исходный 9-битовый блок данных разделяется на три подблока. Каждый подблок является входом S-блока, в котором происходит замена трех входных битов на три выходных бита. Такой S-блок может быть легко представлен в виде таблицы, состоящей из 8 3-битовых величин, пронумерованных целыми числами, изображающими 3 входных бита. Таблица 9.6 наглядно иллюстрирует такое представление S-блока. В нашем алгоритме шифрования мы будем использовать один и тот же S-блок, приведенный в таблице 9.6, во всех раундах шифрования.

Таблица 9.6

S-блок

Вход	0	1	2	3	4	5	6	7
Выход	7	0	6	5	2	1	3	4

Перестановка каждого цикла представляет собой простую перестановку битов сообщения, или изменение позиций битов. Перестановка, показанная на рис. 9.3, для удобства приведена в таблице 9.7 (где номера обозначают позиции битов в блоке: первый бит является крайним левым, а последний — крайним правым) и может быть просто описана следующим образом: выход i S-блока j соединен со входом j S-блока i. Обратите внимание, что в последнем цикле можно обойтись без перестановки, поэтому в нашем алгоритме шифрования она отсутствует.

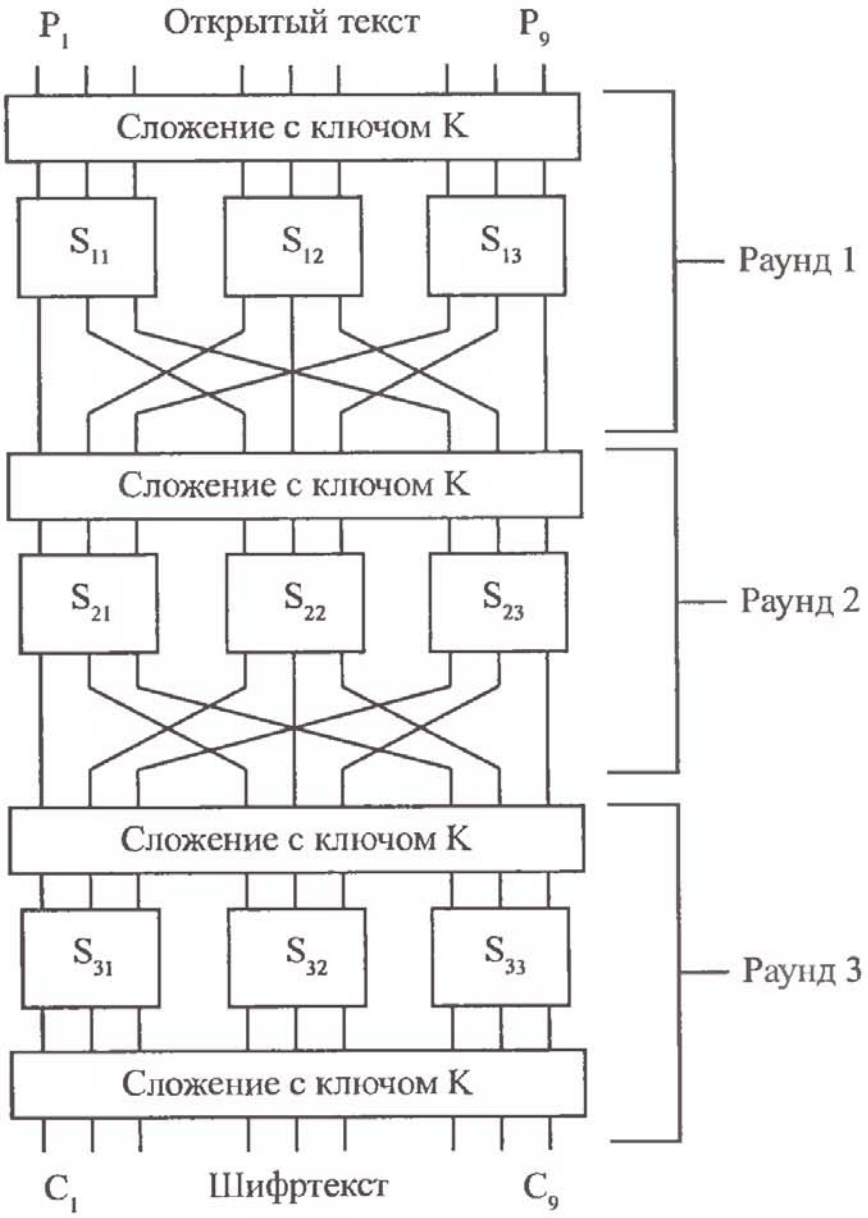


Рис. 9.3. ***Алгоритм шифрования, построенный на основе сети SPN***

Таблица 9.7

Перестановка битов

Вход	1	2	3	4	5	6	7	8	9
Выход	1	4	7	2	5	8	3	6	3

Для достижения сложения с ключом используется простая операция побитового сложения по модулю 2 (XOR) между битами подключа и битами блока данных, поступающего на вход данного цикла. Более того, происходит сложение битов подключа с выходными битами последнего раунда. Обычно в алгоритме шифрования подключ для цикла шифрования извлекается из главного ключа. Од-

нако в этом Учебном Алгоритме Шифрования мы предположим, что в каждом раунде используется один и тот же подключ K.

При дешифровании данные необходимо пропустить через алгоритм шифрования в обратном порядке. Таким образом, алгоритм дешифрования имеет ту же форму, что и алгоритм шифрования и изображен на рис. 9.3. Однако схема использования S-блоков в алгоритме дешифрования отличается от схемы использования S-блоков при зашифровании (то есть входы становятся выходами, а выходы становятся входами). Это подразумевает, что в алгоритме шифрования SPN, разрешенном для дешифрования, все S-блоки должны быть симметричными, то есть один к одному повторять схему с теми же номерами входных и выходных битов. Заметим, что отсутствие перестановки после последнего цикла обеспечивает для алгоритма дешифрования ту же структуру, что и для алгоритма шифрования.

9.3. Описание лабораторных работ

Ниже приведены лабораторные работы, предназначенные для изучения современных методов криптоанализа. В каждой лабораторной работе приведен пример ее выполнения с подробными объяснениями. Первые две работы посвящены изучению методов линейного и дифференциального криптоанализа алгоритмов блочного шифрования, построенных по схеме Фейстеля. Третья и четвертая работы предназначены для изучения тех же методов анализа применительно к алгоритмам шифрования, построенным по принципу сети SPN. И, наконец, последняя работа предназначена для изучения нового метода анализа, названного слайдовой атакой.

ЛАБОРАТОРНАЯ РАБОТА № 1

Изучение метода линейного криптоанализа применительно к алгоритмам шифрования, построенным по схеме Фейстеля

Цель работы — изучение методов линейного криптоанализа, применение изученных методов к алгоритму блочного шифрования, построенному по схеме Фейстеля, на практике.

Пример выполнения лабораторной работы

Рассмотрим Учебный Алгоритм Шифрования, описанный выше. В качестве таблиц замены и перестановок будем использовать таблицы с 9.1 по 9.5. Криптоанализ данного алгоритма начнется с анализа Блоков замены и нахождения наиболее эффективного линейного уравнения. Результаты анализа Блока1, Блока2 и Блока3 приведены соответственно в таблицах 1, 2 и 3.

Таблица 1

Анализ Блока 1

i	Значения j						
S(i, j)	1	2	3	4	5	6	7
1	9	9	6	10	9	5	8
2	7	9	8	4	9	11	8
3	9	11	8	4	7	5	12
4	9	9	10	8	7	7	6
5	7	11	6	8	9	9	6
6	9	7	8	6	9	7	10
7	7	9	12	10	7	5	6
8	7	9	8	8	9	7	8
9	9	11	8	8	7	9	4
10	11	7	10	6	7	7	8
11	5	9	10	10	9	5	8
12	9	7	8	10	1	11	10
13	7	9	12	6	7	9	6
14	13	9	6	8	7	7	6
15	11	3	10	8	9	9	6

Таблица 2

Анализ Блока 2

i	Значения j						
S(i,j)	1	2	3	4	5	6	7
1	8	11	9	9	7	6	6
2	10	9	9	3	7	6	12
3	8	9	11	9	7	4	8
4	8	7	9	9	7	6	10
5	6	7	11	7	7	12	6
6	8	9	7	9	7	8	8
7	10	5	9	11	7	6	8
8	8	7	9	7	9	8	8
9	6	7	11	9	5	10	8
10	8	9	7	11	5	6	10
11	10	5	9	9	9	8	6
12	6	7	7	9	13	6	8
13	8	3	9	7	9	8	12
14	2	9	9	9	9	8	10
15	8	9	3	11	5	10	10

Таблица 3

Анализ Блока 3

I	Значения j		
S(i,j)	1	2	3
1	7	8	9
2	9	8	7
3	7	10	7
4	7	8	9
5	9	6	9
6	7	10	7
7	5	8	11
8	9	8	7
9	7	10	7
10	9	6	9
11	11	8	5
12	7	10	7
13	5	8	11
14	11	8	5
15	5	14	5

Итак, согласно таблице 1, мы можем составить первые три наиболее эффективных линейных уравнения. Сразу следует отметить, что одно из уравнений будет более эффективным, однако его недостаточно для нахождения битов ключа, поэтому мы берем еще уравнение, наиболее приближенное по своей эффективности к первому. В таблице 1 определены три пары (i, j) — (12,5), (14,1) и (15,2). Так как в Блок1, согласно таблице перестановки с расширением, входят биты X_3, X_4, X_1, X_2, а выходные биты после перестановки оказываются на местах Y_7, Y_4, Y_3, то, учитывая, что мы работаем с правой 8-битовой частью исходного 16-битового сообщения, а также сложение по модулю два левой части исходного сообщения с результатом функции F, получаем следующие уравнения:

$$X_{11} \oplus X_{12} \oplus Y_7 \oplus Y_3 \oplus X_7 \oplus X_3 = K_1 \oplus K_2,$$

которое выполняется с вероятностью p = 1/16, а соответственно, $\Delta = |1 - 2p| = 7/8$;

(Так как выходные биты функции F нам не известны, то мы можем их получить путем сложения по модулю два соответствующих битов открытого и шифрованного текста. Поэтому в левой части уравнения для каждого значения бита шифрованного текста присутствует соответствующее ему значение бита открытого текста.)

$$X_{11} \oplus X_{12} \oplus X_9 \oplus Y_3 \oplus X_3 = K_1 \oplus K_2 \oplus K_3,$$

которое выполняется с вероятностью p = 13/16, а соответственно, $\Delta = |1 - 2p| = 5/8$;

$$X_{11} \oplus X_{12} \oplus X_9 \oplus X_{10} \oplus Y_4 \oplus X_4 = K_1 \oplus K_2 \oplus K_3 \oplus K_4,$$

которое выполняется с вероятностью p = 3/16, а соответственно, $\Delta = |1 - 2p| = 5/8$.

Аналогичным образом составляем уравнения для двух остальных блоков. Результаты анализа сведены в таблицу 4.

Таблица 4

№ блока	Эффективное линейное уравнение	P	$\Delta = \lvert 1 - 2p \rvert$
1	$X_{11} \oplus X_{12} \oplus Y_7 \oplus Y_3 \oplus X_7 \oplus X_3 = K_1 \oplus K_2$ (1) $X_{11} \oplus X_{12} \oplus X_9 \oplus Y_3 \oplus X_3 = K_1 \oplus K_2 \oplus K_3$ (2) $X_{11} \oplus X_{12} \oplus X_9 \oplus X_{10} \oplus Y_3 \oplus X_3 = {}_1 \oplus K_2 \oplus K_3 \oplus K_4$ (3)	1/16 13/16 3/16	7/8 (1) 5/8 (2) 5/8 (3)
2	$X_{13} \oplus Y_6 \oplus X_6 = K_7$ (4) $X_{14} \oplus X_{16} \oplus Y_6 \oplus Y_8 \oplus X_6 \oplus X_8 = K_5 \oplus K_6$ (5) $X_{14} \oplus X_{16} \oplus X_{15} \oplus Y_5 \oplus X_5 = K_5 \oplus K_6 \oplus K_8$ (6) $X_{14} \oplus X_{16} \oplus X_{13} \oplus Y_8 \oplus X_8 = K_6 \oplus K_5 \oplus K_7$ (7) $X_{14} \oplus X_{16} \oplus X_{13} \oplus X_{15} \oplus Y_5 \oplus Y_8 \oplus X_5 \oplus X_8 = K_5 \oplus K_6 \oplus K_7 \oplus K_8$ (8)	3/16 13/16 3/16 1/8 3/16	5/8 (4) 5/8 (5) 5/8 (6) 3/4 (7) 5/8 (8)
3	$X_{11} \oplus X_{16} \oplus X_{10} \oplus X_{12} \oplus Y_2 \oplus X_2 = K_9 \oplus K_{10} \oplus K_{11} \oplus K_{12}$ (9)	7/8	3/4 (9)

Анализ последнего блока дает слишком мало информации для нахождения битов ключа. А вот анализ первых двух блоков предоставляет возможность определить первые биты ключа. Для проведения лабораторной работы была разрабо-

тана программа, позволяющая генерировать и зашифровывать открытые тексты на секретном ключе варианта, а также проводить анализ полученных результатов. С помощью данной программы сгенерируем и зашифруем необходимое количество текстов, например 30, на секретном ключе варианта (в данном примере в качестве секретного ключа взято значение K = 101010101010) и приведем результат зашифрования в таблице 5.

Таблица 5

Открытый блок	**Зашифрованный блок**
0000000000001001	1101010100001001
0000000000010010	1111100100010010
0000000000011011	1011110100011011
0000000000100100	1110110100100100
0000000000101101	1010010000101101
0000000000110110	1001100000110110
0000000000111111	0101110000111111
0000000001001000	1100011001001000
0000000001010001	0101101001010001
0000000001011010	1001111001011010
0000000001100011	0111000101100011
0000000001101100	1011100101101100
0000000001110101	1110001101110101
0000000001111110	0110011101111110
0000000010000111	1110111110000111
0000000010010000	1101001110010000
0000000010011001	1001011110011001
0000000010100010	1100101110100010
0000000010101011	1000111110101011
0000000010110100	1010111110110100
0000000010111101	0110011010111101
0000000011000110	1111101011000110
0000000011001111	1011111011001111
0000000011011000	1010010011011000
0000000011100001	0101101011100001
0000000011101010	1001111011101010
0000000011110011	1111001111110011
0000000011111100	0111101111111100
0000000100000101	1101000000000101
0000000100001110	0101010000001110

Используя уравнения, полученные из блока 1, с помощью программы находим, что из N = 100 открытых текстов число открытых текстов T, для которых левая часть уравнения (1) равна нулю, равно 93, T = 93. Так как в этом случае вероятность p = 1/16, то по алгоритму, описанному в п. 5.2.2, находим, что

$$K_1 \oplus K_2 = 1; \quad (10)$$

для уравнения (2)

$$T = 80, \text{ а } p = 13/16, \text{ тогда } K_1 \oplus K_2 \oplus K_3 = 0; \quad (11)$$

для уравнения (3)

$$T = 17, \text{ а } p = 3/16, \text{ тогда } K_1 \oplus K_2 \oplus K_3 \oplus K_4 = 0. \quad (12)$$

Зная это из уравнений (11) и (12), находим, что $K_4 = 0$. Из уравнений (11) и (10) находим, что $K_3 = 1$. Так как $K_1 \oplus K_2 = 1$, то возможны два варианта: либо $K_1 = 0$ и $K_2 = 1$; либо $K_1 = 1$ и $K_2 = 0$.

Аналогичным образом находим биты ключа для второго блока:

для уравнения (4)

$$T = 81, \text{ а } p = 3/16, \text{ тогда } K_7 = 1; \quad (13)$$

для уравнения (5)

$$T = 18, \text{ а } p = 13/16, \text{ тогда } K_5 \oplus K_6 = 1; \quad (14)$$

для уравнения (6)

$$T = 80, \text{ а } p = 3/16, \text{ тогда } K_5 \oplus K_6 \oplus K_8 = 1; \quad (15)$$

для уравнения (7)

$$T = 13, \text{ а } p = 1/8, \text{ тогда } K_5 \oplus K_6 \oplus K_7 = 0; \quad (16)$$

для уравнения (8)

$$T = 18, \text{ а } p = 3/16, \text{ тогда } K_5 \oplus K_6 \oplus K_7 \oplus K_8 = 0. \quad (17)$$

Из уравнений (14) и (15) находим $K_8 = 0$. Так как $K_5 \oplus K_6 = 1$, то возможны два варианта: либо $K_5 = 0$ и $K_6 = 1$; либо $K_5 = 1$ и $K_6 = 0$.

Итак, после анализа двух таблиц имеем четыре возможных варианта первых восьми бит ключа (знаком x обозначены биты, которые было невозможно определить):

$K_1 = 01100110xxxx;$
$K_2 = 10100110xxxx;$
$K_3 = 01101010xxxx;$
$K_4 = 10101010xxxx.$

С помощью программы для лабораторной работы определяем, что четвертая из возможных комбинаций является истинной. Остальные четыре бита ключа можно найти методом грубого перебора.

Подготовка к работе

Номер варианта индивидуального задания назначается преподавателем. Согласно варианту, при домашней подготовке необходимо выполнить следующее:

1. Для каждой из заданных таблиц замены провести статистический анализ и построить соответствующие таблицы;
2. Из полученных таблиц необходимо выбрать элементы, имеющие наибольшее отклонение Δ, а также наиболее приближенные к ним;
3. Построить эффективные линейные уравнения и вычислить их вероятности. **ВНИМАНИЕ!** При составлении линейных уравнений необходимо учитывать особенности алгоритма шифрования, а именно сложение по модулю два выхода F-блока с левой частью исходного текста (см. составление уравнения (1)).

Методические указания по выполнению лабораторной работы

При выполнении лабораторной работы используется программа Crypto1.exe. С помощью данной программы для каждого варианта производится зашифрование указанного числа открытых текстов на определенном ключе. После этого пользователь имеет возможность сохранить либо напечатать полученные пары *открытый—закрытый текст*.

Анализ полученных текстов может быть проведен для каждого из полученных эффективных уравнений после того, как будут указаны биты (согласно данному уравнению), участвующие в анализе (при этом биты X_1 и Y_1 обозначают самый старший бит, а биты X_{16} и Y_{16} — соответственно, самый младший бит), а также вероятность, с которой данное уравнение выполняется. Результатом анализа будет являться заполнение таблицы результата, состоящей из четырех полей. Первое поле, обозначенное буквой N, отображает количество текстов, участвующих в анализе; второе поле, обозначенное буквой P, отображает введенную пользователем вероятность; третье поле, обозначенное буквой T, показывает число открытых текстов, для которых левая часть эффективного уравнения равна 0, и последнее четвертое поле содержит результат анализа, а именно то значение, которому соответствует правая часть данного эффективного уравнения.

Правильность анализа может быть проверена введением всех найденных битов ключа в Форму проверки. Те биты, которые из-за нехватки данных определить невозможно, должны быть помечены как x. Если все биты, которые возможно было определить, найдены верно, то программа выдаст соответствующее подтверждающее сообщение, которое необходимо предъявить преподавателю для допуска к защите работы.

Рекомендуемый порядок работы

1. Вызвать программу Crypto1.exe.
2. Ввести ФИО студента, номер группы и номер варианта.
3. Ввести количество известных текстов (от 1000 до 5000).
4. Произвести их зашифрование.
5. Провести анализ для каждого из полученных эффективных линейных уравнений.
6. На основании проведенного анализа вычислить биты ключа.
7. Проверить правильность анализа.
8. Повторить вышеописанные действия еще два раза. Первый раз, указав количество текстов от 100 до 1000, а второй раз — от 1 до 10. Сделать выводы о том, как влияет количество пар анализируемых текстов на получаемые результаты.

Содержание отчета

Отчет по лабораторной работе должен содержать:

1. Титульный лист с указанием варианта задания;
2. Цель работы;
3. Таблицы статистического анализа и построенные на их основании эффективные линейные уравнения и их вероятности;
4. Не менее десяти пар *открытый—закрытый текст*, полученных для данного варианта, чтобы была возможность наглядно продемонстрировать использование линейных уравнений для конкретных пар текстов;
5. Результаты анализа для каждого из уравнений (полученные с помощью программы);
6. Найденные биты ключа;
7. Выводы по пункту 8 рекомендованного порядка работы;
8. Выводы по проделанной работе.

Варианты индивидуальных заданий

Вариант № 1

S_1:

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

S_2:

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

S_3:

00	3	1	2	3
01	1	2	3	1
10	2	3	1	2
11	3	1	2	3

Вход	S_1	S_2	S_3
0000	001	111	11
0001	101	011	01
0010	011	110	01
0011	010	010	10
0100	100	110	10
0101	111	010	11
0110	011	101	11
0111	011	001	01
1000	111	101	10
1001	100	001	11
1010	001	100	11
1011	110	101	01
1100	101	100	01
1101	010	111	10
1110	101	011	10
1111	110	011	11

Примечание. Для блока S_1 взять пары (i, j), для которых отклонение Δ больше или равно 1/2. Для блока S_2 взять пары (i, j), для которых отклонение Δ больше или равно 5/8. Для блока S_3 взять пары (i, j), для которых отклонение Δ больше или равно 1/4.

Вариант № 2

S_1:

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

S_2:

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

S_3:

00	2	3	3	1
01	1	2	1	2
10	3	1	2	1
11	1	3	3	2

Вход	S_1	S_2	S_3
0000	010	100	10
0001	001	001	01
0010	011	100	11
0011	110	111	10
0100	110	101	11
0101	011	110	01
0110	100	010	01
0111	100	001	10
1000	111	010	11
1001	101	110	01
1010	100	100	01
1011	111	101	11
1100	001	011	10
1101	110	111	11
1110	010	110	01
1111	101	011	10

Примечание. Для блока S_1 взять пары (i, j), для которых отклонение Δ больше или равно 5/8. Для блока S_2 взять пары (i, j), для которых отклонение Δ больше или равно 1/2. Для блока S_3 взять пары (i, j), для которых отклонение Δ больше или равно 5/8.

Вариант № 3

S_1:

0	5	6	2	5	7	1	6	2
1	6	2	3	7	6	1	7	1

S_2:

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

S_3:

00	1	2	2	3
01	3	1	3	2
10	1	2	1	2
11	2	3	3	1

Вход	S_1	S_2	S_3
0000	101	001	01
0001	110	110	11
0010	010	110	10
0011	101	001	01
0100	111	101	10
0101	001	010	11
0110	110	010	11
0111	010	111	10
1000	110	010	01
1001	010	001	10
1010	011	011	10
1011	111	100	11
1100	110	111	01
1101	001	011	11
1110	111	101	10
1111	001	101	01

Примечание. Для всех блоков замены взять пары (i, j), для которых отклонение Δ больше или равно 1/2.

Вариант № 4

S_1:

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

S_2:

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

S_3:

00	2	1	2	3
01	3	2	3	1
10	1	2	3	2
11	2	3	1	3

Вход	S_1	S_2	S_3
0000	011	001	10
0001	100	001	11
0010	001	010	01
0011	001	011	10
0100	010	010	10
0101	011	111	11
0110	110	100	11
0111	110	110	01
1000	110	011	01
1001	111	111	10
1010	010	011	10
1011	100	110	11
1100	001	100	11
1101	101	101	01
1110	111	101	10
1111	101	001	11

Примечание. Для блока S_1 взять пары (i, j), для которых отклонение Δ больше или равно 5/8. Для блоков S_2 и S_3 взять пары (i, j), для которых отклонение Δ больше или равно 1/2.

Вариант № 5

S_1:

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

S_2:

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

S_3:

00	1	2	3	3
01	3	1	3	2
10	2	1	2	3
11	1	3	1	2

Вход	S_1	S_2	S_3
0000	101	010	01
0001	010	111	11
0010	101	010	10
0011	011	111	01
0100	001	110	11
0101	110	001	11
0110	111	011	11
0111	001	001	10
1000	111	100	10
1001	100	001	01
1010	100	100	01
1011	010	101	11
1100	011	101	10
1101	101	011	01
1110	110	110	11
1111	011	110	10

Примечание. Для всех блоков замены взять пары (i, j), для которых отклонение Δ больше или равно 1/2.

Вариант № 6

S_1:

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

S_2:

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

S_3:

00	3	2	1	3
01	2	1	3	2
10	1	3	2	1
11	3	2	1	3

Вход	S_1	S_2	S_3
0000	110	110	11
0001	010	101	10
0010	111	011	10
0011	100	101	01
0100	001	111	01
0101	001	001	11
0110	010	010	11
0111	011	010	10
1000	101	101	01
1001	001	001	11
1010	010	110	11
1011	101	100	10
1100	011	110	10
1101	100	011	01
1110	001	100	01
1111	110	111	11

Примечание. Для блока S_1 взять пары (i, j), для которых отклонение Δ больше или равно 5/8. Для блока S_2 взять пары (i, j), для которых отклонение Δ больше или равно 1/2. Для блока S_3 взять пары (i, j), для которых отклонение Δ больше или равно 1/4.

Вариант № 7

S_1:

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

S_2:

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

S_3:

00	3	1	2	1
01	3	2	2	1
10	2	3	1	3
11	1	3	2	1

Вход	S_1	S_2	S_3
0000	001	100	11
0001	101	101	11
0010	001	010	01
0011	100	101	10
0100	010	111	10
0101	110	110	10
0110	110	001	01
0111	011	011	01
1000	100	001	10
1001	101	010	01
1010	101	111	11
1011	111	100	11
1100	010	111	01
1101	010	011	10
1110	011	011	11
1111	111	110	01

Примечание. Для всех блоков замены взять пары (i, j), для которых отклонение Δ больше или равно 1/2.

Вариант № 8

S_1:

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

S_2:

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

S_3:

00	1	1	1	2
01	1	2	2	1
10	3	2	2	3
11	3	3	3	1

Вход	S_1	S_2	S_3
0000	110	110	01
0001	011	010	01
0010	001	011	01
0011	111	010	10
0100	001	110	01
0101	100	001	10
0110	111	011	10
0111	011	100	01
1000	011	111	11
1001	010	101	11
1010	101	100	10
1011	100	101	11
1100	110	010	10
1101	111	001	11
1110	010	111	11
1111	101	101	01

Примечание. Для всех блоков замены взять пары (i, j), для которых отклонение Δ больше или равно 1/2.

Вариант № 9

S_1:

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

S_2:

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

S_3:

00	1	3	1	3
01	3	2	2	2
10	2	2	1	3
11	3	1	3	1

Вход	S_1	S_2	S_3
0000	011	001	01
0001	101	110	11
0010	110	111	11
0011	100	100	10
0100	101	100	01
0101	001	010	10
0110	010	110	11
0111	101	010	10
1000	111	011	10
1001	110	100	11
1010	010	011	10
1011	011	001	01
1100	111	101	01
1101	100	111	11
1110	001	011	11
1111	011	101	01

Примечание. Для всех блоков замены взять пары (i, j), для которых отклонение Δ больше или равно 1/2.

Вариант № 10

S_1:

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

S_2:

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

S_3:

00	1	1	2	1
01	1	3	1	2
10	3	2	3	2
11	2	2	3	3

Вход	S_1	S_2	S_3
0000	111	111	01
0001	111	001	01
0010	001	111	01
0011	110	001	11
0100	110	011	10
0101	010	110	01
0110	101	011	01
0111	011	010	10
1000	001	100	11
1001	100	010	10
1010	011	110	10
1011	111	011	10
1100	100	100	11
1101	010	001	11
1110	011	101	10
1111	101	101	11

Примечание. Для всех блоков замены взять пары (i, j), для которых отклонение Δ больше или равно 1/2.

Контрольные вопросы

1. Что такое криптоанализ?
2. Опишите принцип работы алгоритма шифрования S-DES.
3. Что такое линейный статистический аналог нелинейной функции? Ответ поясните.
4. Что определяет эффективность линейного статистического аналога?
5. Что называется n-раундовой тройной суммой?
6. Определите статистический аналог для трех циклов алгоритма шифрования DES.
7. Приведите алгоритм нахождения битов ключа при найденных эффективных линейных уравнениях.
8. По какому принципу строятся таблицы статистического анализа при проведении линейного криптоанализа?
9. Как определяется отклонение Δ?

ЛАБОРАТОРНАЯ РАБОТА № 2

Изучение метода дифференциального криптоанализа применительно к алгоритмам шифрования, построенным по схеме Фейстеля

Цель работы — изучение методов дифференциального криптоанализа, применение изученных методов к алгоритму блочного шифрования, построенного по схеме Фейстеля, на практике.

Пример выполнения лабораторной работы

В данной лабораторной работе рассматривается Учебный алгоритм шифрования, описанный в п. 9.1. Значения таблиц замены и перестановок приведены в таблицах с 9.1 по 9.5. Для удобства работы с данными создадим таблицу 1, отражающую зависимость выходов блоков замены от входных данных.

Таблица 1

Зависимость входов и выходов блоков замены Учебного Алгоритма Шифрования

Вход в блок перестановки	Выход Блока 1	Выход Блока 2	Выход Блока 3
0000	100	011	01
0001	110	101	10
0010	001	111	11
0011	011	010	01
0100	101	100	10
0101	111	110	11
0110	010	001	01
0111	101	111	10
1000	101	100	11
1001	111	110	01
1010	010	001	10
1011	100	011	11
1100	110	101	01
1101	001	111	10
1110	011	010	11
1111	110	001	01

После того как подготовительная работа проведена, можно приступить к анализу блоков замены и построению таблиц зависимостей ΔA от ΔC (см. п. 6.2.1).

Строятся данные таблицы следующим образом. Так как на вход каждого блока замены подается по четыре бита, то и размер их суммы по модулю 2 не будет превышать четырех бит. Таким образом, диапазон изменения ΔA лежит в пределах 0000–1111. Но, так как пара анализируемых текстов должна различаться хотя бы одним битом, то мы не можем использовать ΔA = 0000. Поэтому диапазон изменения ΔA составляет 15 значений от 0001 до 1111. Каждое из значений ΔA может быть получено шестнадцатью возможными комбинациями входных данных блоков замены. Так, например ΔA = 0001 может быть получено следующими возможными комбинациями:

1. 0000 ⊕ 0001;
2. 0001 ⊕ 0000;
3. 0010 ⊕ 0011;
4. 0011 ⊕ 0010;
5. 0100 ⊕ 0101;
6. 0101 ⊕ 0100;
7. 0110 ⊕ 0111;
8. 0111 ⊕ 0110;
9. 1000 ⊕ 1001;
10. 1001 ⊕ 1000;
11. 1010 ⊕ 1011;
12. 1011 ⊕ 1010;
13. 1100 ⊕ 1101;
14. 1101 ⊕ 1100;
15. 1110 ⊕ 1111;
16. 1111 ⊕ 1110.

При этом сумма выходов по модулю 2, полученных после прохождения любой пары данных входов через конкретный блок замены, не всегда совпадет с суммой выходов того же блока замены по модулю 2 другой пары. Поясним сказанное на примере. Рассмотрим пару входов 0011 ⊕ 0010. Значение 0011 при прохождении через Блок1 даст нам 011, а 0010 — 001. Сумма этих выходов по модулю 2 будет равна ΔC = 011 ⊕ 001 = 010.

Рассмотрим другую пару входов 0110 ⊕ 0111. При прохождении через Блок1 значение 0110 даст нам 010, а 0111 — 101. Таким образом, значение ΔC = 010 ⊕ 101 = 111.

Из данного примера наглядно видно, что одному и тому же значению ΔA могут соответствовать различные ΔC. Анализ блоков замены приведен в таблицах 2, 3 и 4, соответственно. Как видно из таблицы 2, в случае, когда ΔA = 0001, 8 раз появляется ΔC = 010, 2 раза ΔC = 101 и ΔC = 110 и 4 раза ΔC = 111.

Таблица 2

Зависимость ΔC от ΔA в Блоке 1

ΔC / ΔA	000	001	010	011	100	101	110	111
0001	0	0	8	0	0	2	2	4
0010	0	0	2	2	0	6	0	6
0011	4	2	2	0	0	4	0	4
0100	0	6	2	4	0	0	4	0
0101	0	4	0	4	6	0	0	2
0110	0	2	0	2	6	2	4	0
0111	0	2	2	4	4	0	4	0
1000	0	6	0	6	0	0	2	2
1001	0	4	0	4	4	2	2	0
1010	0	2	2	0	8	0	4	0
1011	2	0	0	6	4	0	4	0
1100	6	2	4	0	0	2	0	2
1101	4	0	4	0	0	2	2	4
1110	2	0	4	2	0	0	0	8
1111	4	2	2	0	0	8	0	0

Таблица 3

Зависимость ΔC от ΔA в Блоке 2

ΔC / ΔA	000	001	010	011	100	101	110	111
0001	0	0	8	2	0	2	4	0
0010	0	2	0	0	2	6	2	4
0011	0	2	2	2	2	2	0	6
0100	0	4	2	4	0	2	2	2
0101	4	4	0	6	0	2	0	0
0110	0	0	4	2	6	0	2	2
0111	0	2	0	0	6	2	6	0
1000	0	6	0	4	0	0	4	2
1001	2	2	0	6	2	4	0	0
1010	2	6	2	2	4	0	4	2
1011	2	2	2	0	4	2	4	0
1100	6	0	2	2	2	2	2	0
1101	4	0	6	0	2	0	4	0
1110	0	4	2	0	2	8	0	0
1111	2	0	2	2	0	0	0	10

Таблица 4

Зависимость ΔC от ΔA в Блоке 3

ΔC / ΔA	00	01	10	11
0001	0	4	6	6
0010	0	4	6	6
0011	8	4	2	2
0100	0	4	6	6
0101	8	4	2	2
0110	8	4	2	2
0111	4	0	6	6
1000	0	4	6	6
1001	8	4	2	2
1010	8	4	2	2
1011	4	0	6	6
1100	8	4	2	2
1101	4	0	6	6
1110	4	0	6	6
1111	6	10	0	0

После того как проведен анализ и построены таблицы, можно приступить к выявлению наилучшего значения ΔA и соответствующего ему ΔC, то есть пары (ΔA, ΔC). Из таблиц 3 и 4 сразу видно, что для Блока2 оптимальна пара (1111,111), а для Блока3 — (1111, 01). Таким образом, сразу можно сказать, что однозначно определены последние 8 бит ΔA (того ΔA, которое получается суммированием по модулю 2 E(X) и $E(X_1)$), то есть ΔA = xxxx11111111, а значит? однозначно определены последние 5 бит соответствующего ΔC (ΔC = xxx11101). Немного иначе обстоит дело с таблицей 2. В ней можно выделить несколько равновероятных пар (ΔA, ΔC). Это — (0001,010), (1010,100), (1110,111), (1111,101). Соответственно возникает 4 варианта 12-битного ΔA. При этом следует учитывать еще один факт: ΔA равно сумме по модулю 2 переставленных и расширенных входных битов. Поэтому, согласно таблице перестановки с расширением, используемой в данном алгоритме шифрования (таблица 9.1), в ΔA соответственно должны быть равны следующие пары бит: 1 и 9, 6 и 10, 4 и 11, 2 и 12. Этому условию удовлетворяет единственное значение ΔA = 111111111111, которому соответствует ΔC = 10111101. Именно с ним и будет вестись дальнейшая работа. Сведения по определению наилучшего 12-битного ΔA приведены в таблице 5.

Таблица 5

Выбор наилучшего 12-битного ΔA

ΔA	ΔX	ΔC
000111111111	—	01011101
101011111111	—	10011101
111011111111	—	11111101
111111111111	xxxxxxxx11111111	10111101

Теперь, зная наилучшее значение ΔA, можно приступить к нахождению ключа. Для этого нам понадобятся несколько пар открытых текстов (X, X_1), таких, что $\Delta A = E(X) \oplus E(X_1) = 111111111111$, а $\Delta C = S(E(X)) \oplus S(E(X_1)) = 10111101$. Для того чтобы из зашифрованного сообщения X выделить S(E(X)) необходимо к последним восьми битам шифрованного сообщения добавить первые восемь бит открытого текста, а затем учесть последнюю перестановку. Для удобства работы, полученные с помощью программы, разработанной для проведения данной лабораторной работы, открытые X, X_1 и шифрованные Y, Y_1 тексты и относящиеся к ним данные E(X), $E(X_1)$, S(E(X)), $S(E(X_1))$ занесены в таблицу 6.

Таблица 6

Пары открытых текстов (X,X_1), соответствующие наилучшим (ΔA, ΔC)

№	X	E(X)	S(E(X))	Y
1	0000000000001111	000011110100	01011011	1101110000001111
2	0000000000011110	010010110001	01110111	1111010100011110
3	0000000001101001	100101101110	01110110	0111010101101001
№	X_1	$E(X_1)$	$S(E(X_1))$	Y_1
1	0000000011110000	111100001011	11100110	0111001111110000
2	0000000011100001	101101001110	11001010	0101101011100001
3	0000000010010110	011010010001	11001011	1101101010010110

Рассмотрим приведенные пары открытых текстов. Необходимо также учитывать, что результат F-функции складывается по модулю 2 с левой частью исходного сообщения. Но в связи с тем, что в нашем случае левые восемь бит исходных сообщений равны нулю, то можно сделать вывод, что левые восемь бит шифрованного сообщения и есть выход F-функции. Так как на вход блоков замены поступают значения E(X), то для всех X и X_1 имеем следующую картину.

Для Блока 1

1. Пара (0000000000001111, 0000000011110000).

Так как на вход блока поступает значение E(X) ⊕ Ki, то имеем следующее:

0000 ⊕ K_1 даст на выходе 010;

1111 ⊕ K_1 даст на выходе 111.

Из таблицы 1 определяем, что на выходе Блока 1 значение 010 получается в том случае, когда на его вход подается одно из значений 0110 или 1010, а значение 111 — при входных 1001 или 0101. Исходя из этого имеем следующие возможные варианты:

$0000 \oplus K_1 = 0110$; $K_1 = 0110$;
$0000 \oplus K_1 = 1010$; $K_1 = 1010$;
$1111 \oplus K_1 = 1001$; $K_1 = 0110$;
$1111 \oplus K_1 = 0101$; $K_1 = 1010$.

2. Пара (0000000000011110, 0000000011100001).

Так как на вход блока поступает значение $E(X) \oplus K_1$, то имеем следующее:

$0100 \oplus K_1$ даст на выходе 011;
$1011 \oplus K_1$ даст на выходе 110.

Из таблицы 1 определяем, что на выходе Блока 1 значение 011 получается в том случае, когда на его вход подается одно из значений 0011 или 1110, а значение 110 — при входных 0001 или 1100, или 1111. Исходя из этого имеем следующие возможные варианты:

$0100 \oplus K_1 = 0011$; $K_1 = 0111$;
$0100 \oplus K_1 = 1110$; $K_1 = 1010$;
$1011 \oplus K_1 = 0001$; $K_1 = 1010$;
$1011 \oplus K_1 = 1100$; $K_1 = 0111$;
$1011 \oplus K_1 = 1111$; $K_1 = 0100$.

3. Пара (0000000001101001, 0000000010010110).

Аналогично предыдущим пунктам определяем:

$1001 \oplus K_1$ даст на выходе 011;
$0110 \oplus K_1$ даст на выходе 110.

Из таблицы 1 определяем, что на выходе Блока 1 значение 011 получается в том случае, когда на его вход подается одно из значений 0011 или 1110, а значение 110 — при входных 0001, или 1100. Исходя из этого имеем следующие возможные варианты:

$1001 \oplus K_1 = 0011$; $K_1 = 1010$;
$1001 \oplus K_1 = 1110$; $K_1 = 0111$;
$0110 \oplus K_1 = 0001$; $K_1 = 0111$;
$0110 \oplus K_1 = 1100$; $K_1 = 1010$.

Вывод: даже проанализировав две пары открытых текстов, мы видим, что один из подключей, а именно $K_1 = 1010$, встречается чаще остальных. Таким образом, можно предположить, что это и есть первый подключ.

Для Блока 2

1. Пара (0000000000001111, 0000000011110000).

Так как на вход блока поступает значение $E(X) \oplus K_1$, то имеем следующее:

$1111 \oplus K_2$ даст на выходе 110;
$0000 \oplus K_2$ даст на выходе 001.

Из таблицы 1 определяем, что на выходе Блока 2 значение 110 получается в том случае, когда на его вход подается одно из значений 0101 или 1001, а значение 001 — при входных 0110 или 1010, или 1111. Исходя из этого имеем следующие возможные варианты:

$1111 \oplus K_2 = 0101$; $K_2 = 1010$;
$1111 \oplus K_2 = 1001$; $K_2 = 0110$;
$0000 \oplus K_2 = 0110$; $K_2 = 0110$;
$0000 \oplus K_2 = 1010$; $K_2 = 1010$;
$0000 \oplus K_2 = 1111$; $K_2 = 1111$.

2. Пара (0000000000011110, 0000000011100001).

$1011 \oplus K_2$ даст на выходе 101;

$0100 \oplus K_2$ даст на выходе 010.

Из таблицы 1 определяем, что на выходе Блока 2 значение 101 получается в том случае, когда на его вход подается одно из значений 0001 или 1100, а значение 010 — при входных 0011, или 1110. Исходя из этого имеем следующие возможные варианты:

$1011 \oplus K_2 = 0001$; $K_2 = 1010$;
$1011 \oplus K_2 = 1100$; $K_2 = 0111$;
$0100 \oplus K_2 = 0011$; $K_2 = 0111$;
$0100 \oplus K_2 = 1110$; $K_2 = 1010$.

3. Пара (0000000001101001, 0000000010010110).

Аналогично предыдущим пунктам определяем:

$0110 \oplus K_2$ даст на выходе 101;

$1001 \oplus K_2$ даст на выходе 010.

Из таблицы 1 определяем, что на выходе Блока 2 значение 101 получается в том случае, когда на его вход подается одно из значений 0001 или 1100, а значение 010 — при входных 0011, или 1110. Исходя из этого имеем следующие возможные варианты:

$0110 \oplus K_2 = 0001$; $K_2 = 0111$;
$0110 \oplus K_2 = 1100$; $K_2 = 1010$;
$1001 \oplus K_2 = 0011$; $K_2 = 1010$;
$1001 \oplus K_2 = 1110$; $K_2 = 0111$.

Вывод: проанализировав три пары открытых текстов, мы видим, что один из подключей, а именно $K_2 = 1010$, встречается чаще остальных. Таким образом, можно предположить, что это и есть второй подключ.

Для Блока 3

1. Пара (0000000000001111, 0000000011110000).

Так как на вход блока поступает значение $E(X) \oplus K_1$, то имеем следующее:

$0100 \oplus K_3$ даст на выходе 11;

$1011 \oplus K_3$ даст на выходе 10.

Из таблицы 1 определяем, что на выходе Блока 3 значение 11 получается в том случае, когда на его вход подается одно из 5 значений: 0010 или 0101, или 1000, или 1011, или 1110, а значение 10 — при входных 0001 или 0100, или 0111, или 1010, или 1101. Исходя из этого имеем следующие возможные варианты:

$0100 \oplus K_3 = 0010$; $K_3 = 0110$;
$0100 \oplus K_3 = 0101$; $K_3 = 0001$;
$0100 \oplus K_3 = 1000$; $K_3 = 1100$;
$0100 \oplus K_3 = 1011$; $K_3 = 1111$;
$0100 \oplus K_3 = 1110$; $K_3 = 1010$;
$1011 \oplus K_3 = 0001$; $K_3 = 1010$;
$1011 \oplus K_3 = 0100$; $K_3 = 1111$;
$1011 \oplus K_3 = 0111$; $K_3 = 1100$;
$1011 \oplus K_3 = 1010$; $K_3 = 0001$;
$1011 \oplus K_3 = 1101$; $K_3 = 0110$.

2. Пара (0000000000011110, 0000000011100001).

$0001 \oplus K_3$ даст на выходе 11;

$1110 \oplus K_3$ даст на выходе 10.

Из таблицы 1 определяем, что на выходе Блока 3 значение 11 получается в том случае, когда на его вход подается одно из 5 значений: 0010 или 0101, или 1000, или 1011, или 1110, а значение 10 — при входных 0001 или 0100, или 0111, или 1010, или 1101. Исходя из этого имеем следующие возможные варианты:

$0001 \oplus K_3 = 0010$; $K_3 = 0011$;
$0001 \oplus K_3 = 0101$; $K_3 = 0100$;
$0001 \oplus K_3 = 1000$; $K_3 = 1001$;
$0001 \oplus K_3 = 1011$; $K_3 = 1010$;
$0001 \oplus K_3 = 1110$; $K_3 = 1111$;
$1110 \oplus K_3 = 0001$; $K_3 = 1111$;
$1110 \oplus K_3 = 0100$; $K_3 = 1010$;
$1110 \oplus K_3 = 0111$; $K_3 = 1001$;
$1110 \oplus K_3 = 1010$; $K_3 = 0100$;
$1110 \oplus K_3 = 1101$; $K_3 = 0011$.

3. Пара (0000000001101001, 0000000010010110).

$1110 \oplus K_3$ даст на выходе 10;

$0001 \oplus K_3$ даст на выходе 11.

Из таблицы 1 определяем, что на выходе Блока 3 значение 10 получается в том случае, когда на его вход подается одно из 5 значений: 0001 или 0100, или 0111, или 1010, или 1101. А значение 11, если на вход блока поступило значение 0010 или 0101, или 1000, или 1011, или 1110. Исходя из этого имеем следующие возможные варианты:

$1110 \oplus K_з = 0001$; $K_з = 1111$;
$1110 \oplus K_з = 0100$; $K_з = 1010$;
$1110 \oplus K_з = 0111$; $K_з = 1001$;
$1110 \oplus K_з = 1010$; $K_з = 0100$;
$1110 \oplus K_з = 1101$; $K_з = 0011$;
$0001 \oplus K_з = 0010$; $K_з = 0011$;
$0001 \oplus K_з = 0101$; $K_з = 0100$;
$0001 \oplus K_з = 1000$; $K_з = 1001$;
$0001 \oplus K_з = 1011$; $K_з = 1010$;
$0001 \oplus K_з = 1110$; $K_з = 1111$.

Вывод: проанализировав три пары открытых текстов, мы видим, что два подключа, а именно $K_з = 1010$ и $K_з = 1111$, встречаются чаще остальных. Таким образом, можно предположить, что один из этих подключей и есть третий подключ.

Объединив результаты анализа, получим два варианта искомого ключа K: K = 101010101010 и K = 101010101111. С помощью программы, разработанной для данной лабораторной работы, определяем, что ключ K = 101010101010 и есть секретный ключ анализируемого варианта.

Подготовка к работе

Номер варианта индивидуального задания назначается преподавателем. Согласно варианту, при домашней подготовке необходимо выполнить следующее:

1. Для каждой из заданных таблиц замены провести статистический анализ и построить соответствующие таблицы зависимостей значений ΔC от ΔA;
2. Из полученных таблиц необходимо выбрать наиболее вероятную пару значений, и на основании данных всех трех таблиц построить пару (ΔA, ΔC);
3. Если для одной таблицы получилось несколько равновероятных пар значений, то выбираем ту из них, которая в совокупности с остальными будет удовлетворять равенству битов в таблице перестановки с расширением.

Методические указания по выполнению лабораторной работы

При выполнении данной лабораторной работы используется программа Crypto2.exe. С помощью этой программы для каждого варианта производится зашифрование указанного числа открытых текстов на определенном ключе. Также для каждого из этих открытых текстов программа подбирает парный текст, согласно указанному значению ΔA, который в свою очередь зашифровывается на том же ключе. Если получившиеся пары выходов удовлетворяют указанному значению ΔC, то данные пары текстов выводятся на экран. После этого пользователь име-

ет возможность сохранить либо напечатать полученные результаты для удобства дальнейшей работы.

В таблицах с результатами шифрования указаны значения ΔX и ΔY, которые соответствуют введенным значениям ΔA и ΔC.

После того как биты ключа будут найдены, их правильность может быть проверена введением всех найденных битов в Форму проверки. Те биты, которые определить невозможно, должны быть помечены как x. Если все биты, которые возможно было определить, найдены верно, то программа выдаст соответствующее сообщение, которое необходимо предъявить преподавателю для допуска к защите работы.

Рекомендуемый порядок работы

1. Вызвать программу Crypto2.exe.
2. Ввести Ф.И.О. студента, номер группы и номер варианта.
3. Ввести количество известных текстов (от 1 до 5000).
4. Произвести зашифрование.
5. Исходя из полученных результатов определить биты ключа.
6. Проверить правильность найденного ключа (если получилось несколько возможных вариантов ключа, то осуществлять проверку до тех пор, пока не будет найден правильный ключ).
7. Повторить вышеописанные действия для какой-либо пары (ΔA, ΔC), не обладающей максимальной вероятностью, и попробовать определить биты ключа. Сделать выводы о влиянии вероятности пар (ΔA, ΔC) на получаемый результат анализа.

Содержание отчета

1. Титульный лист с указанием варианта задания.
2. Цель работы.
3. Таблицы статистического анализа с выделенными значениями пар, имеющих наибольшую вероятность и построенная на их основании пара (ΔA, ΔC).
4. Не менее пяти пар шифрованных данных, полученных для данного варианта, чтобы была возможность проследить ход действий при проведении дифференциального анализа.
5. Анализ как минимум трех пар текстов с указанием возможных значений подключей.
6. Найденные биты ключа.
7. Выводы по п. 7 рекомендованного порядка работы.

Варианты индивидуальных заданий

Таблицы замены соответствуют тем, что указаны для соответствующего варианта в предыдущей лабораторной работе.

Вариант № 1

Таблица перестановки с расширением

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

Примечание. Для таблицы замены S_1 взять наименьшее значение ΔA, удовлетворяющее таблице перестановки с расширением.

Вариант № 2

Таблица перестановки с расширением

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

Вариант № 3

Таблица перестановки с расширением

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

Вариант № 4

Таблица перестановки с расширением

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

Вариант № 5

Таблица перестановки с расширением

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

Вариант № 6

Таблица перестановки с расширением

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

Вариант № 7

Таблица перестановки с расширением

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

Вариант № 8

Таблица перестановки с расширением

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

Вариант № 9

Таблица перестановки с расширением

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

Примечание. Для таблицы S_2 взять наибольшее значение ΔA.

Вариант № 10

Таблица перестановки с расширением

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

Контрольные вопросы

1. Какие типы криптоанализа вы знаете?
2. Опишите алгоритм получения подключей для алгоритма шифрования S-DES.
3. Опишите принцип, на котором основан метод дифференциального криптоанализа.
4. По какому принципу строятся таблицы статистического анализа при проведении дифференциального криптоанализа?
5. Какая пара текстов называется правильной?
6. Какая пара текстов называется неправильной?
7. Какое обязательное условие должно выполняться при выборе пар открытых текстов?
8. Почему при рассмотрении одного раунда шифрования нельзя использовать ΔA, равное нулю?

ЛАБОРАТОРНАЯ РАБОТА № 3

Изучение метода дифференциального криптоанализа применительно к многораундовым алгоритмам шифрования, построенным на основе сети SPN

Цель работы — изучить применение метода дифференциального криптоанализа к многораундовым алгоритмам блочного шифрования, построенным по схеме сети SPN.

Пример выполнения лабораторной работы

Нам задан такой алгоритм шифрования, построенный на основе сети SPN, какой изображен на рис. 9.3. Пусть у него S-блок замены работает согласно таблице 1.

Таблица 1

S-блок замены для рассматриваемого алгоритма шифрования, построенного на основе сети SPN

Вход	0	1	2	3	4	5	6	7
Выход	7	0	6	5	2	1	3	4

Для удобства работы представим таблицу 1 в двоичном виде так, как показано в таблице 2.

Таблица 2

Двоичное представление блока замены

Вход	Выход
000	111
001	000
010	110
011	101
100	010
101	001
110	011
111	100

При выборе пар анализируемых текстов необходимо руководствоваться следующими соображениями. Количество анализируемых разностей должно быть

достаточным для нахождения всех битов используемого секретного ключа. При этом каждая из анализируемых разностей должна при своем прохождении через раунды шифрования затрагивать как можно меньше блоков замены (большое количество затрагиваемых блоков возможно лишь в том случае, когда вероятность прохождения разности через блок равна 1) и иметь как можно большую вероятность получения выходной разности. Если анализ наиболее вероятных разностей не позволяет выявить все биты используемого секретного ключа, то в этом случае возможно рассмотрение менее вероятных разностей; при этом необходимо проанализировать большее количество пар *открытый—закрытый текст*.

Для упрощения лабораторной работы каждому варианту задаются исходные разности, опираясь на которые, необходимо проанализировать данные с помощью метода дифференциального анализа и попытаться найти биты используемого ключа. Для рассматриваемого нами варианта исходными данными будут являться данные, представленные в таблице 3.

Таблица 3

Исходные данные варианта

i	ΔA_i
1	110 000 000
2	000 000 111
3	001 001 000

Так как на вход данного алгоритма шифрования поступает 9-битовый блок входных данных, то можно сказать, что у нас есть 29 = 512 вариантов входной разности ΔA. Каждая входная разность при прохождении через три раунда шифрования даст на выходе определенную выходную разность ΔC с некоторой вероятностью. Для определения ключа, используемого в алгоритме шифрования, нам достаточно будет проанализировать в среднем четыре входные разности ΔA (каждому варианту будут указаны входные разности, с помощью которых необходимо провести анализ).

Итак, первым делом нам необходимо проанализировать S-блок замены, используемый в нашем алгоритме шифрования и построить таблицу зависимостей выходной разности S-блока ΔC от входной разности того же блока ΔA.

Строится данная таблица следующим образом. Так как на вход блока замены подается три бита, то и сумма по модулю 2 двух входов не будет превышать трех бит. Таким образом, диапазон изменения ΔA лежит в пределах 000–111. Каждое из значений ΔA может быть получено 8 возможными комбинациями входных данных блоков замены. Так, например, ΔA = 001 может быть получено следующими возможными комбинациями:

1. $000 \oplus 001$;
2. $001 \oplus 000$;
3. $010 \oplus 011$;
4. $011 \oplus 010$;
5. $100 \oplus 101$;
6. $101 \oplus 100$;
7. $110 \oplus 111$;
8. $111 \oplus 110$.

При этом сумма выходов по модулю 2, полученных после прохождения любой пары данных входов через блок замены, не всегда совпадет с суммой выходов того же блока замены по модулю 2 другой пары. Поясним сказанное на примере. Рассмотрим пару входов $011 \oplus 010$. Значение 011 при прохождении через блок замены даст нам 101, а 010 — 110 (см. таблицу 2). Сумма этих выходов по модулю 2 будет равна $\Delta C = 101 \oplus 110 = 011$.

А теперь рассмотрим другую пару входов $110 \oplus 111$. При прохождении через блок замены вход 110 даст нам на выходе значение 011, а входное значение 111 — выходное значение 100. Таким образом, $\Delta C = 011 \oplus 100 = 111$.

Из данного примера наглядно видно, что одному и тому же значению ΔA могут соответствовать различные ΔC. Результаты анализа нашего блока замены приведены в таблице 4.

Таблица 4

Анализ блока замены

ΔC / ΔA	000	001	010	011	100	101	110	111
000	8	0	0	0	0	0	0	0
001	0	0	0	4	0	0	0	4
010	0	4	0	0	0	4	0	0
011	0	0	4	0	0	0	4	0
100	0	4	0	0	0	4	0	0
101	0	0	4	0	0	0	4	0
110	0	0	0	0	8	0	0	0
111	0	0	0	4	0	0	0	4

Рассмотрим первое значение входной разности $\Delta A_1 = 110\ 000\ 000$. Согласно входному значению разности ΔA_1, на вход блока S_{11} поступит значсние разности 110, а на вход блоков S_{12} и S_{13} — значение разности 000. Мы знаем, что входная разность, равная нулю, всегда дает нулевую выходную разность. Поэтому на выходе блоков S_{12} и S_{13} будут находиться нулевые выходные разности. Согласно таблице 4, при поступлении на вход блока замены разности, равной 110, на выходе всегда будет разность, равная 100. Таким образом, после первого раунда преобразования (с учетом таблицы перестановки), входная разность второго раунда шифрования будет равна 100 000 000. А значит, на вход блока замены S_{21} поступит входная разность 100, а на вход блоков замены S_{22} и S_{23} — входная разность 000.

Согласно таблице 4, входная разность 100 с равной вероятностью может дать на выходе значение разности, равное 001 или 101. Выходные разности блоков замены S_{22} и S_{23} будут равны 0. Таким образом, с учетом таблицы перестановки на вход третьего раунда шифрования может поступить разность, равная 000 000 100, либо разность, равная 100 000 100 (см. рис. 1).

Теперь с помощью программы, разработанной для проведения данной лабораторной работы, сформируем пять пар текстов, удовлетворяющих заданному значению входной разности $\Delta A_1 = 110\ 000\ 000$, и зашифруем их на секретном ключе данного варианта. Результаты шифрования приведены в таблице 5.

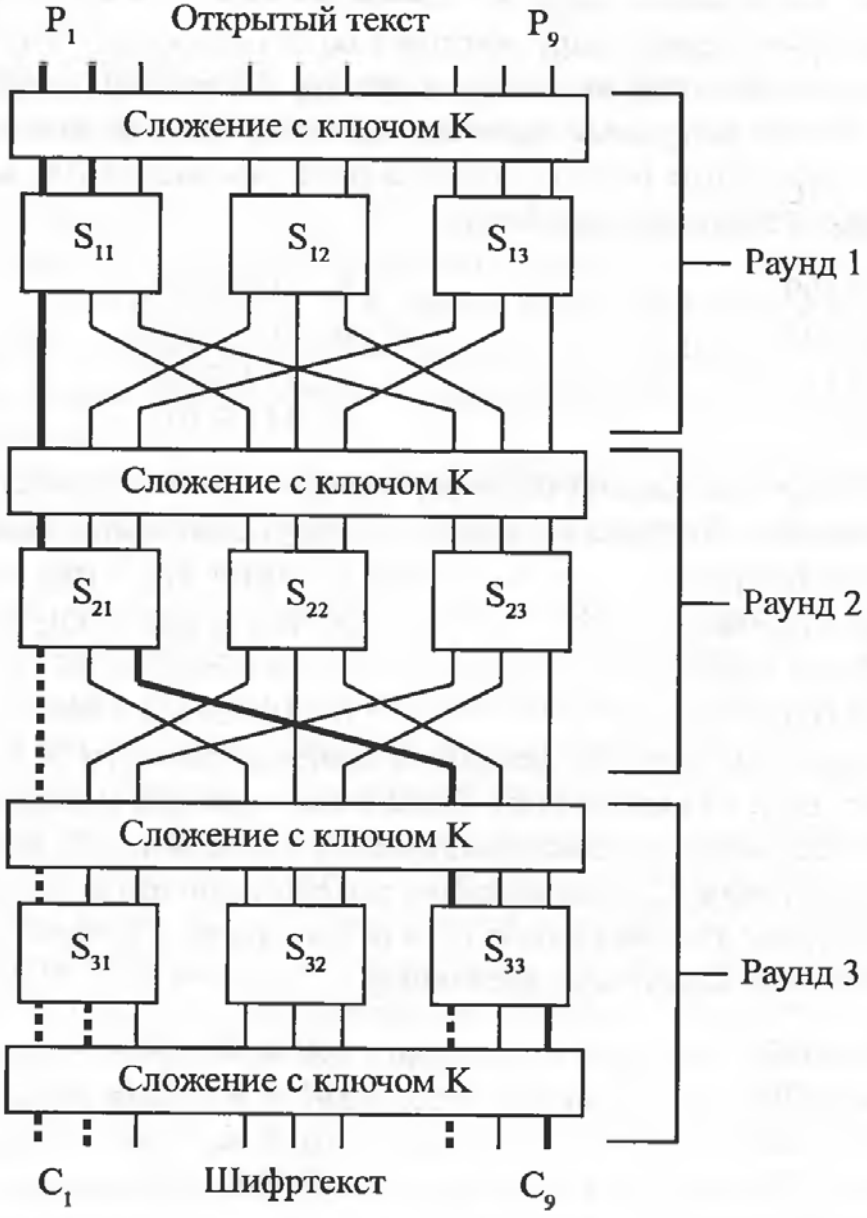


Рис. 1. ***Дифференциальный криптоанализ шифровальной сети на основе подстановок и перемешивания***

Таблица 5

Результаты шифрования

№	X	Y	X′	Y′
1	100111010	111011110	010111010	110011011
2	011101110	011000100	101101110	010000101
3	000101111	101111111	110101111	101111010
4	101111111	000111000	011111111	000111001
5	100110100	111010101	010110100	111010100

Проанализируем первую пару текстов. Соответствующие этой входной паре выходные тексты образуют выходную разность $\Delta C = 111011110 \oplus 110011011 = 001000101$. Значит, ненулевые значения разностей будут на выходах блоков S_{31} и S_{33}. На вход этих блоков подается входная разность, равная 100, которая может быть образована 8 разными способами:

1. $000 \oplus 100$;
2. $001 \oplus 101$;
3. $010 \oplus 110$;
4. $011 \oplus 111$;
5. $100 \oplus 000$;
6. $101 \oplus 001$;
7. $110 \oplus 010$;
8. $111 \oplus 011$.

Соответственно для каждой такой пары входов можно получить соответствующую пару выходов. Согласно таблице 2, это будут следующие пары выходов:

1. $111 \oplus 010 = 101$;
2. $000 \oplus 001 = 001$;
3. $110 \oplus 011 = 101$;
4. $101 \oplus 100 = 001$;
5. $010 \oplus 111 = 101$;
6. $001 \oplus 000 = 001$;
7. $011 \oplus 110 = 101$;
8. $100 \oplus 101 = 001$.

Так как на выходе блока S_{31} находится значение разности 001, то оно могло быть получено, если на выходе этого блока были пары под номерами 2, 4, 6 и 8. В связи с тем, что выход S_{31} блока складывается с подключом K_1 (под подключом понимается часть ключа K, складываемая с выходом данного S-блока, то есть подлюч K_1 — это первые три бита ключа K), в результате чего получается известный шифртекст, получаем следующие уравнения:

$000 \oplus K_1 = 111$;
$001 \oplus K_1 = 110$;
$001 \oplus K_1 = 111$;
$000 \oplus K_1 = 110$;
$100 \oplus K_1 = 111$;
$101 \oplus K_1 = 110$;
$101 \oplus K_1 = 111$;
$100 \oplus K_1 = 110$.

Правые части полученных уравнений представляют собой выходы блока S_{31}, сложенные по модулю два с первым подключом, и получены исходя из известных шифртекстов.

Таким образом, подключ K_1 может принимать одно из следующих значений: 111 или 110 или 011 или 010.

На выходе блока S_{33} находится значение разности 101, а оно могло быть получено, если выходными значениями этого блока были пары под номерами 1, 3, 5 и 7. В связи с тем, что выход S_{33} блока складывается с подключом, в результате чего получается известный шифртекст, получаем следующие уравнения:

$111 \oplus K_3 = 110;$
$010 \oplus K_3 = 011;$
$010 \oplus K_3 = 110;$
$111 \oplus K_3 = 011;$
$011 \oplus K_3 = 110;$
$110 \oplus K_3 = 011;$
$011 \oplus K_3 = 110;$
$110 \oplus K_3 = 011.$

Правые части полученных уравнений представляют собой выходы блока S_{33}, сложенные по модулю два с третьим подключом, и получены исходя из известных шифртекстов.

Таким образом, подключ K_3 может принимать одно из следующих значений: 001 или 100, или 101, или 000. Оставшиеся пары текстов можно не анализировать, так как на входе блоков S_{31} и S_{33} всегда будет одна и та же разность, совпадающая с рассмотренной нами разностью $\Delta A_1 = 110\ 000\ 000$, а значит, мы будем получать одинаковые четыре варианта возможных значений подключа, уже полученные нами ранее.

Теперь рассмотрим второе значение входной разности $\Delta A_2 = 000\ 000\ 111$ (см. рис. 2). Согласно входному значению разности ΔA_2, на вход блока S_{13} поступит значение разности 111, а на вход блоков S_{11} и S_{12} — значение разности 000. Мы знаем, что входная разность, равная нулю, всегда дает нулевую выходную разность. Поэтому на выходе блоков S_{11} и S_{12} будут находиться нулевые выходные разности. Согласно таблице 4, при поступлении на вход блока замены разности, равной 111, на выходе может появиться либо разность, равная 011, либо разность, равная 111. Таким образом, после первого раунда преобразования (с учетом таблицы перестановки), входная разность второго раунда шифрования будет равна либо 000 001 001, либо 001 001 001. А это значит, что на вход блоков замены S_{22}, S_{23} и, возможно, блока S_{21} поступит входная разность 001. Согласно таблице 4, входная разность 001 с равной вероятностью может дать на выходе значение разности, равное 011 или 111. Таким образом, с учетом таблицы перестановки (таблица 9.7), на вход третьего раунда шифрования поступит входная разность ххх х11 х11, где знаком х обозначено неизвестное значение входа. Так как на вход блока S_{31} поступает полностью неизвестное значение входной разности ххх, то этот блок будет невозможно подвергнуть анализу. На вход блоков S_{32} и S_{33} поступают частично известные значения входных разностей х11, то есть входная разность блоков может принимать значения 011 или 111. Так как двум этим входным разностям на выходе соответствуют отличные друг от друга выходные разности, то становится возможным подвергнуть оба этих блока анализу.

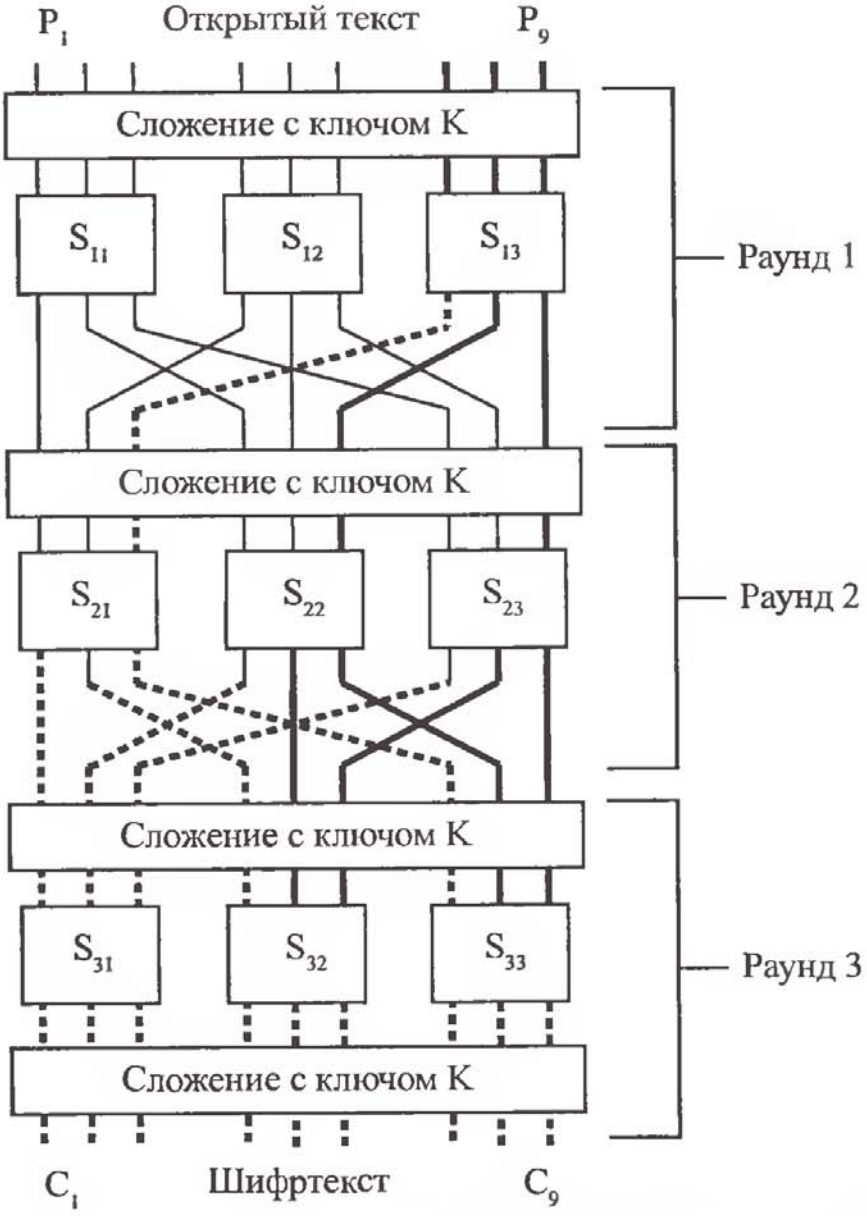


Рис. 2. ***Дифференциальный криптоанализ шифровальной сети на основе подстановок и перемешивания***

Таблица 6

Результаты шифрования

№	X	Y	X'	Y'
1	110011110	110000000	110011001	101110010
2	111111011	000001001	111111100	010010010
3	000111010	000011001	000111101	000100010
4	011100000	001101110	011100111	001111100
5	000000010	000011010	000000101	011100001

Продолжим анализ и сформируем с помощью программы, разработанной для проведения данной лабораторной работы, пять пар текстов, удовлетворяющих заданному значению входной разности $\Delta A_2 = 000\ 000\ 111$, после чего зашифруем их на секретном ключе данного варианта. Результаты шифрования приведены в таблице 6.

Проанализируем первую пару текстов. Соответствующие этой входной паре выходные тексты образуют выходную разность $\Delta C = 110000000 \oplus 1011100101 = 011110010$. А значит, мы будем рассматривать как блок S_{32}, так и блок S_{33}. На выходе блока S_{32} находится разность, равная 110. Используя таблицу 4 определяем, что этой выходной разности могут соответствовать значения входных разностей, равные 011 или 101. Так как ранее мы определили, что на вход блока S_{32} может поступить входная разность равная либо 011, либо 111, то делаем вывод, что входной разностью этого блока в данном случае является значение 011. Входное значение разности 011 может быть образовано 8-ю разными способами:

1. $000 \oplus 011$;
2. $001 \oplus 010$;
3. $010 \oplus 001$;
4. $011 \oplus 000$;
5. $100 \oplus 111$;
6. $101 \oplus 110$;
7. $110 \oplus 101$;
8. $111 \oplus 100$.

Соответственно для каждой такой пары входов можно получить соответствующую пару выходов. Согласно таблице 2, это будут следующие пары выходов:

1. $111 \oplus 101 = 010$;
2. $000 \oplus 110 = 110$;
3. $110 \oplus 000 = 010$;
4. $101 \oplus 111 = 110$;
5. $010 \oplus 100 = 110$;
6. $001 \oplus 011 = 010$;
7. $011 \oplus 001 = 010$;
8. $100 \oplus 010 = 110$.

Так как на выходе блока S_{32} находится значение разности 110, то оно могло быть получено, если на выходе этого блока были пары под номерами 2, 4, 5 и 8. В связи с тем, что выход S_{32} блока складывается с подключом, в результате чего получается известный шифртекст, получаем следующие уравнения:

$000 \oplus K_2 = 000$;
$110 \oplus K_2 = 110$;
$110 \oplus K_2 = 000$;
$000 \oplus K_2 = 110$;
$100 \oplus K_2 = 000$;
$010 \oplus K_2 = 110$;
$010 \oplus K_2 = 000$;
$100 \oplus K_2 = 110$.

Правые части полученных уравнений представляют собой выходы блока S_{32}, сложенные по модулю два со вторым подключом, они получены исходя из известных шифртекстов.

Таким образом подключ K_2 может принимать одно из следующих значений: 000 или 110, или 100, или 010.

На выходе блока S_{33} находится разность, равная 010. Используя таблицу 4, определяем, что этой выходной разности могут соответствовать значения входных

разностей, равные 011 или 101. Так как ранее мы определили, что на вход блока S_{33} может поступить входная разность, равная либо 011, либо 111, то делаем вывод, что входной разностью этого блока в данном случае является значение 011. Выше мы определили 8 способов, которыми может быть образовано входное значение разности 011 и соответствующие каждому из этих способов пары выходных значений.

Так как на выходе блока S_{33} находится значение разности 010, то оно могло быть получено, если на выходе этого блока были пары под номерами 1, 3, 6 и 7. В связи с тем, что выход S_{33} блока складывается с подключом, в результате чего получается известный шифртекст, получаем следующие уравнения:

$111 \oplus K_3 = 000;$ $001 \oplus K_3 = 000;$
$101 \oplus K_3 = 010;$ $011 \oplus K_3 = 010;$
$101 \oplus K_3 = 000;$ $011 \oplus K_3 = 000;$
$111 \oplus K_3 = 010;$ $001 \oplus K_3 = 010.$

Правые части полученных уравнений представляют собой выходы блока S_{33}, сложенные по модулю два с третьим подключом, они получены исходя из известных шифртекстов.

Таким образом, подключ K_3 может принимать одно из следующих значений: 111 или 101, или 001, или 011.

Ранее нами были определены еще четыре возможных варианта значений подключа K_3: 001, 100,101 или 000.

Как видно, из всех этих значений совпадают только два возможных подключа, а именно: 101 и 001, а значит, один из этих подключей и является истинным.

Теперь проанализируем вторую пару текстов. Соответствующие этой входной паре выходные тексты образуют выходную разность $\Delta C = 000001001 \oplus \oplus 010010010 = 010011011$. А значит, мы будем рассматривать как блок S_{32}, так и блок S_{33}. На выходе блока обоих этих блоков находится разность, равная 011. Используя таблицу 4, определяем, что этой выходной разности могут соответствовать значения входных разностей, равные 001 или 111. Так как ранее мы определили, что на вход блоков S_{32} и S_{33} может поступить входная разность, равная либо 011, либо 111, то делаем вывод, что входной разностью этих блоков в данном случае будет являться значение 111. Входное значение разности 111 может быть образовано 8 разными способами:

1. $000 \oplus 111;$
2. $001 \oplus 110;$
3. $010 \oplus 101;$
4. $011 \oplus 100;$
5. $100 \oplus 011;$
6. $101 \oplus 010;$
7. $110 \oplus 001;$
8. $111 \oplus 000.$

Соответственно для каждой такой пары входов можно получить соответствующую пару выходов. Согласно таблице 2, это будут следующие пары выходов:

1. $111 \oplus 100 = 011$;
2. $000 \oplus 011 = 011$;
3. $110 \oplus 001 = 111$;
4. $101 \oplus 010 = 111$;
5. $010 \oplus 101 = 111$;
6. $001 \oplus 110 = 111$;
7. $011 \oplus 000 = 011$;
8. $100 \oplus 111 = 011$.

Так как на выходе блоков S_{32} и S_{33} находится значение разности 011, то оно могло быть получено, если на выходе этого блока были пары под номерами 1, 2, 7 и 8. В связи с тем, что выходы блоков S_{32} и S_{33} складываются с подключами, в результате чего получается известный шифртекст, а также, учитывая, что последние три бита рассматриваемых шифртекстов равны предпоследним трем битам тех же текстов, получаем следующие уравнения:

$111 \oplus K_2 = 111 \oplus K_3 = 001$;
$100 \oplus K_2 = 100 \oplus K_3 = 010$;
$100 \oplus K_2 = 100 \oplus K_3 = 001$;
$111 \oplus K_2 = 111 \oplus K_3 = 010$;
$011 \oplus K_2 = 011 \oplus K_3 = 001$;
$000 \oplus K_2 = 000 \oplus K_3 = 010$;
$000 \oplus K_2 = 000 \oplus K_3 = 001$;
$011 \oplus K_2 = 011 \oplus K_3 = 010$.

Правые части полученных уравнений представляют собой выходы блока замены, сложенные по модулю два со вторым подключом, они получены исходя из известных шифртекстов.

Таким образом, подключ K_2 также, как и подключ K_3, может принимать одно из следующих значений: 110 или 101, или 010, или 001.

Ранее нами были определены еще четыре возможных варианта значений подключа K_2: 000, 110, 100 или 010.

Как видно, из всех этих значений совпадают только два возможных подключа, а именно: 110 и 010, а значит, один из этих подключей и является истинным.

Также нами ранее были определены два значения подключа K_3: 101 и 001. И мы еще раз убеждаемся, что одно из этих значений и есть искомый подключ K_3.

Оставшиеся пары текстов можно не анализировать, так как на входе блоков S_{32} и S_{33} всегда будут входные разности, равные либо 011, либо 111, а значит, мы будем получать одинаковые возможные значения подключа.

Завершим анализ рассмотрением третьего значения входной разности $\Delta A_3 = 001\ 001\ 000$ (см. рис. 3). Согласно входному значению разности ΔA_3, на вход блоков S_{11} и S_{12} поступит значение разности 001, а на вход блока S_{13} — значение разности 000. Мы знаем, что входная разность, равная нулю, всегда дает нулевую выходную разность. Поэтому на выходе блока S_{13} будет находиться нулевая выходная разность. Согласно таблице 4, при поступлении на вход блока замены разности, равной 001, на выходе может появиться либо разность, равная 011, либо разность, равная 111. Таким образом, после первого раунда преобразования (с учетом таблицы перестановки), входная разность второго раунда шифрования будет равна xx0 110 110, где знаком x обозначены неизвестные биты входной разно-

сти. Пользуясь таблицей 4 определяем, что входная разность, равная 110, в любом случае даст на выходе разность, равную 100. Если же на входе будет находиться значение разности, равное 010 или 100, то на выходе появится значение разности, равное 001 или 101. Таким образом, с учетом таблицы перестановки, на вход третьего раунда шифрования поступит разность, равная x11 000 x0x. Таким образом, на вход блока S_{31} может поступить входная разность, равная 011 или 111. В первом случае на выходе должна будет появиться разность, равная 010 или 110, а во втором случае — 011 или 111.

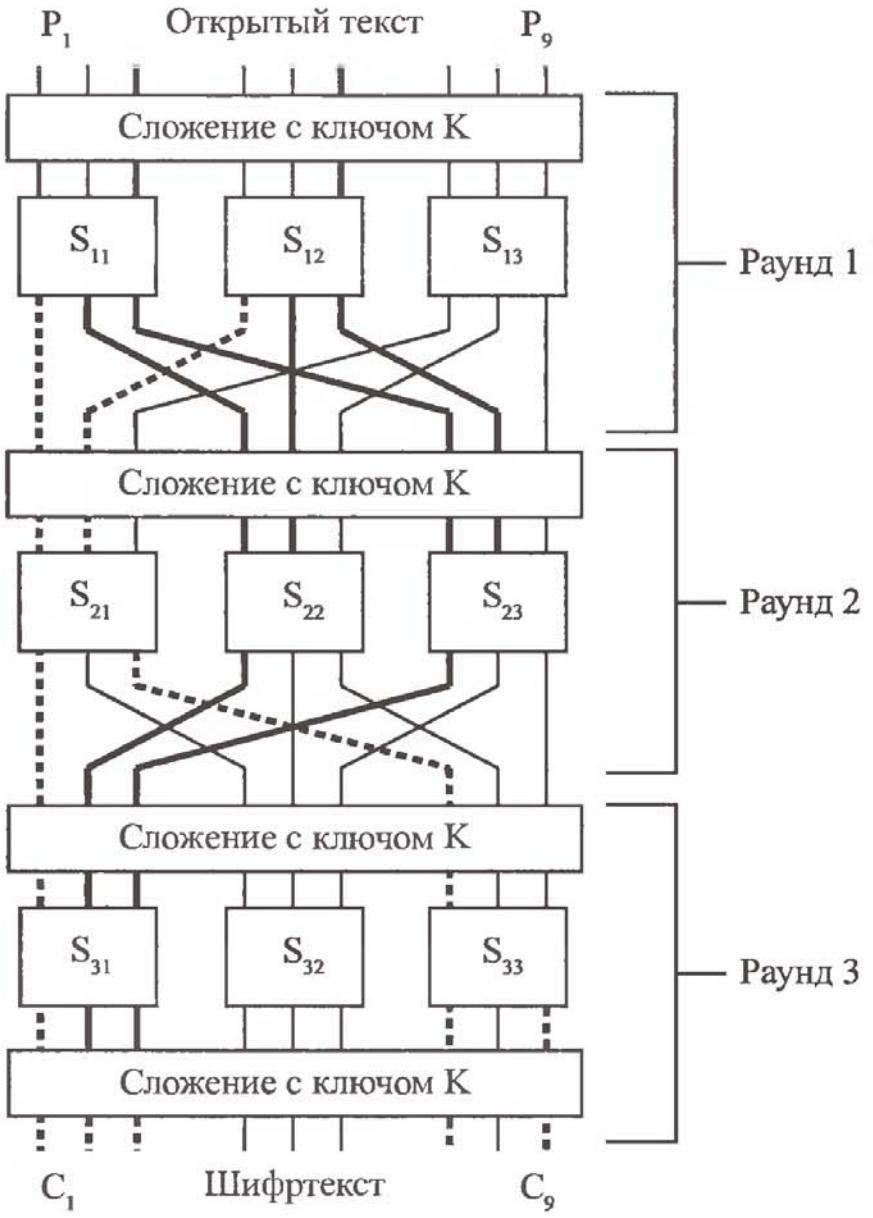


Рис. 3. ***Дифференциальный криптоанализ шифровальной сети на основе подстановок и перемешивания***

Сформируем с помощью программы, разработанной для проведения данной лабораторной работы, пять пар текстов, удовлетворяющих заданному значению

входной разности $\Delta A_3 = 001\ 001\ 000$, после чего зашифруем их на секретном ключе данного варианта. Результаты шифрования приведены в таблице 7.

Таблица 7

Результаты шифрования

№	X	Y	X′	Y′
1	111010111	011111111	110011111	000111111
2	011001001	101110001	010000001	011110001
3	011101110	011000100	010100110	101000101
4	110101000	110101000	111100000	001101000
5	110100111	001111001	111101111	110111001

Проанализируем первую пару текстов. Соответствующие этой входной паре выходные тексты образуют выходную разность

$$\Delta C = 011111111 \oplus 000111111 = 011111111.$$

На выходе блока S_{31} находится разность, равная 011. Используя таблицу 4, определяем, что этой выходной разности могут соответствовать значения входных разностей, равные 001 или 111. Так как ранее мы определили, что на вход блока S_{31} может поступить входная разность, равная либо 011, либо 111, то делаем вывод, что входной разностью этого блока в данном случае является значение 111. Входное значение разности 111 может быть образовано 8 разными способами:

1. $000 \oplus 111$;
2. $001 \oplus 110$;
3. $010 \oplus 101$;
4. $011 \oplus 100$;
5. $100 \oplus 011$;
6. $101 \oplus 010$;
7. $110 \oplus 001$;
8. $111 \oplus 000$.

Соответственно для каждой такой пары входов можно получить соответствующую пару выходов. Согласно таблице 2, это будут следующие пары выходов:

1. $111 \oplus 100 = 011$;
2. $000 \oplus 011 = 011$;
3. $110 \oplus 001 = 111$;
4. $101 \oplus 010 = 111$;
5. $010 \oplus 101 = 111$;
6. $001 \oplus 110 = 111$;
7. $011 \oplus 000 = 011$;
8. $100 \oplus 111 = 011$.

Так как на выходе блока S_{31} находится значение разности 0111, то оно могло быть получено, если на выходе этого блока были пары под номерами 1, 2, 7 и 8. В связи с тем, что выход S_{31} блока складывается с подключом, в результате чего получается известный шифртекст, получаем следующие уравнения:

$111 \oplus K_1 = 011$;
$100 \oplus K_1 = 000$;
$100 \oplus K_1 = 011$;
$111 \oplus K_1 = 000$;
$011 \oplus K_1 = 011$;
$000 \oplus K_1 = 000$;
$000 \oplus K_1 = 011$;
$110 \oplus K_1 = 000$.

Правые части полученных уравнений представляют собой выходы блока S_{31}, сложенные по модулю два первым подключом, они получены исходя из известных шифртекстов.

Таким образом, подключ K_1 может принимать одно из следующих значений: 100 или 111, или 000, или 011.

Ранее нами были определены еще четыре возможных варианта значений подключа K_1: 111, 110, 011 или 010.

Как видно, из всех этих значений совпадают только два возможных подключа, а именно: 111 и 011, а значит, один из этих подключей и является истинным.

Итак, для каждого из подключей K_1, K_2 и K_3 мы получили два возможных значения. Для K_1 — это значения 111 и 011, для K_2 — 110 и 010, для K_3 — 101 и 001. Таким образом, у нас есть восемь возможных значений ключа из всех 512 возможных комбинаций:

1. 111110101;
2. 111010101;
3. 111110001;
4. 111010001;
5. 011110101;
6. 011110001;
7. 011010101;
8. 011010001.

С помощью программы, разработанной для этой лабораторной работы, можно определить, что ключ под номером 4, K = 111010001, является истинным.

Подготовка к работе

Номер варианта индивидуального задания назначается преподавателем. Согласно варианту, при домашней подготовке необходимо выполнить следующее:

1. Для таблицы замены провести анализ и построить соответствующую таблицу зависимости значений ΔC от значений ΔA;
2. С помощью заданных входных значений ΔA провести предварительный анализ алгоритма шифрования с целью определения предполагаемой выходной разности.

Методические указания по выполнению лабораторной работы

При выполнении лабораторной работы используется программа Crypto3.exe. С помощью данной программы для каждого варианта производится зашифрование указанного числа открытых текстов на секретном ключе варианта. Также для каждого из этих открытых текстов подбирается парный текст, согласно указанному значению ΔA, который в свою очередь зашифровывается на том же ключе. После этого полученные пары текстов выводятся на экран.

После того как биты ключа будут найдены, их правильность может быть проверена введением всех найденных битов в Форму проверки. Если биты ключа найдены верно, то программа выдаст соответствующее сообщение, которое необходимо предъявить преподавателю для допуска к защите работы.

Рекомендуемый порядок работы

1. Вызвать программу Crypto3.exe.
2. Ввести ФИО студента, номер группы и номер варианта.
3. Ввести количество известных открытых текстов (от 1 до 512).
4. Для каждого заданного значения ΔA произвести зашифрование данных и проанализировать полученные результаты.
5. Определить возможные значения подключей и ключа.
6. Проверить правильность найденного ключа (если получилось несколько возможных вариантов ключа, то осуществлять проверку до тех пор, пока не будет найден правильный ключ).

Содержание отчета

1. Титульный лист с указанием варианта задания.
2. Цель работы.
3. Таблица зависимости значений ΔC от ΔA для S-блока данного варианта.
4. Не менее пяти пар шифрованных данных, полученных для каждого из заданных значений ΔA для данного варианта.
5. Анализ текстов с указанием возможных значений подключей.
6. Найденные биты ключа.
7. Выводы по проделанной работе.

Варианты индивидуальных заданий

Вариант № 1

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	5	2	1	4	7	0	3	6

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	100 000 000
2	000 011 000
3	000 000 011

Вариант № 2

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	4	1	2	6	5	0	3	7

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	100 000 000
2	000 100 000
3	000 011 000

Вариант № 3

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	6	2	1	4	3	7	5	0

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	000 101 000
2	000 011 000

Вариант № 4

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	7	5	3	2	0	1	4	6

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	000 000 110
2	110 000 000
3	000 001 000

Вариант № 5

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	6	3	4	2	5	0	7	1

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	000 100 000
2	000 000 100
3	000 000 001

Вариант № 6

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	3	7	2	5	1	6	0	4

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	000 000 110
2	000 000 010

Вариант № 7

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	6	0	5	1	7	3	4	2

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	000 110 000
2	000 000 110
3	000 101 000
4	000 000 101

Вариант № 8

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	5	3	4	6	0	2	1	7

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	000 000 110
2	000 110 110

Вариант № 9

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	5	6	2	7	0	3	1	4

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	101 000 000
2	000 000 010

Вариант № 10

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	7	3	1	5	6	2	4	0

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	001 000 000
2	000 001 000
3	010 010 010

Вариант № 11

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	0	2	6	4	1	7	3	5

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	011 000 000
2	000 011 000
3	100 000 000
4	000 000 100

Вариант № 12

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	1	7	6	5	4	2	3	0

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	100 000 000
2	000 000 100
3	001 000 000
4	000 010 000

Вариант № 13

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	4	5	2	0	7	6	1	3

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	100 000 000
2	000 100 000
3	000 001 000

Вариант № 14

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	6	7	2	3	0	4	1	5

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	011 000 000
2	000 011 000
3	001 000 001

Вариант № 15

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	7	6	4	2	0	1	3	5

Значения входных разностей, которые необходимо проанализировать:

i	ΔA_i
1	100 000 000
2	001 000 000

Контрольные вопросы

1. Опишите принцип работы алгоритма, построенного по схеме SPN.
2. Почему в алгоритме, построенном по схеме SPN, начальной и конечной операцией является операция сложения с ключом?

ЛАБОРАТОРНАЯ РАБОТА № 4

Изучение метода линейного криптоанализа применительно к многораундовым алгоритмам шифрования, построенным на основе сети SPN

Цель работы — изучить приемы применения метода линейного криптоанализа к многораундовым алгоритмам блочного шифрования, построенным по схеме сети SPN.

Пример выполнения лабораторной работы

Задан такой алгоритм шифрования, построенный на основе сети SPN, который изображен на рис. 9.3. Пусть у него S-блок замены работает согласно таблице 1.

Таблица 1

S-блок замены для рассматриваемого алгоритма шифрования, построенного на основе сети SPN

Вход	0	1	2	3	4	5	6	7
Выход	7	1	4	0	6	2	5	3

Для удобства работы представим таблицу 1 в двоичном виде так, как показано в таблице 2.

Таблица 2

Двоичное представление блока замены

Вход	Выход
000	111
001	001
010	100
011	000
100	110
101	010
110	101
111	011

Для каждого варианта заданы исходные данные, опираясь на которые, необходимо построить линейные статистические аналоги и попытаться найти биты используемого ключа. Применение линейного анализа к алгоритмам шифрования,

построенным по принципу сети SPN, описано в п. 5.3 настоящей книги. Для нахождения наиболее эффективных аналогов необходимо перебрать всевозможные комбинации. В лабораторной работе для уменьшения трудоемкости поиска дается информация (подсказка), которую необходимо использовать при нахождении эффективных аналогов. Это, прежде всего, указание, какие блоки замены необходимо рассмотреть при анализе алгоритма. Следующее — какие биты открытого текста, или входные биты, и какие биты шифрованного текста, или выходные биты, должны присутствовать в получаемых линейных аналогах. Для рассматриваемого варианта, исходными будут являться данные, приведенные в таблице 3. Согласно этой таблице, нам необходимо построить такой аналог, который будет зависеть от 9 бита открытого текста, и при этом будут задействованы блоки замены S_{13}, S_{21} и S_{31} (сцепление блоков). То есть для блока S_{13} мы должны подобрать такой выход, который затронет только вход блока S_{21}. А для блока S_{21} — такой, который затронет только вход блока S_{31}.

Таблица 3

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_9	$(S_{13}) \rightarrow (S_{21}) \rightarrow (S_{31})$	Всевозможные выходные комбинации блока S_{31}, кроме выхода, равного 111
2	X_7, X_8, X_9	$(S_{13}) \rightarrow (S_{21}, S_{23}) \rightarrow (S_{31})$	Выходы блока S_{31}, равные 010 и 110
3	X_5	$(S_{12}) \rightarrow (S_{22}) \rightarrow (S_{32})$	Выходы блока S_{32}, равные 011 и 110
4	X_1, X_4	$(S_{11}, S_{12}) \rightarrow (S_{22}) \rightarrow (S_{33})$	Выход блока S_{33}, равный 011
5	X_2	$(S_{11}) \rightarrow (S_{22}) \rightarrow (S_{32})$	Выход блока S_{32}, равный 010
6	X_8	$(S_{13}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{31})$	Выход блока S_{31}, равный 001

Выполнение лабораторной работы начинается с анализа блока замены, используемого в данном алгоритме шифрования. Трехбитовым входам в S-блок однозначно соответствуют трехбитовые выходы. Их значения можно определить с помощью таблицы замены 1.

В ходе анализа мы прослеживаем всевозможные комбинации двоичных векторов i и j. Каждую пару векторов мы используем в качестве маски, которую накладываем на всевозможные пары *вход—выход* блока замены. Эти маски указывают нам на биты входа и выхода, соответственно, которые необходимо сложить по модулю два, а затем сравнить полученные результаты. Подобные действия проделываются и при выполнении лабораторной работы № 1. Более подробное описание можно найти в п. 5.2.1. Результат анализа приведен в таблице 4. Из таблицы 4 нетрудно увидеть, что максимальное отклонение имеют пары векторов (i, j): (1, 4), (6, 1), (7, 5). Заметим, что в ходе анализа мы будем пользоваться не только этими

парами векторов. Это связано с тем, что эффективный выход одного блока замены не всегда будет являться эффективным входом для следующего блока замены. Поясним сказанное на примере. Для первого блока замены S_{11} используем первую пару векторов (1, 4), то есть выходом блока S_{11} будет значение 100 (4 в двоичном виде). Пройдя через сложение с ключом, единица окажется на входе блока S_{21} (см. рис. 1). Таким образом, нам придется рассматривать блок замены с входным значением 100. А данный вход не имеет выходов, которые бы имели максимальное отклонение. Это значит, что придется использовать пару векторов, имеющую отклонение меньше максимального.

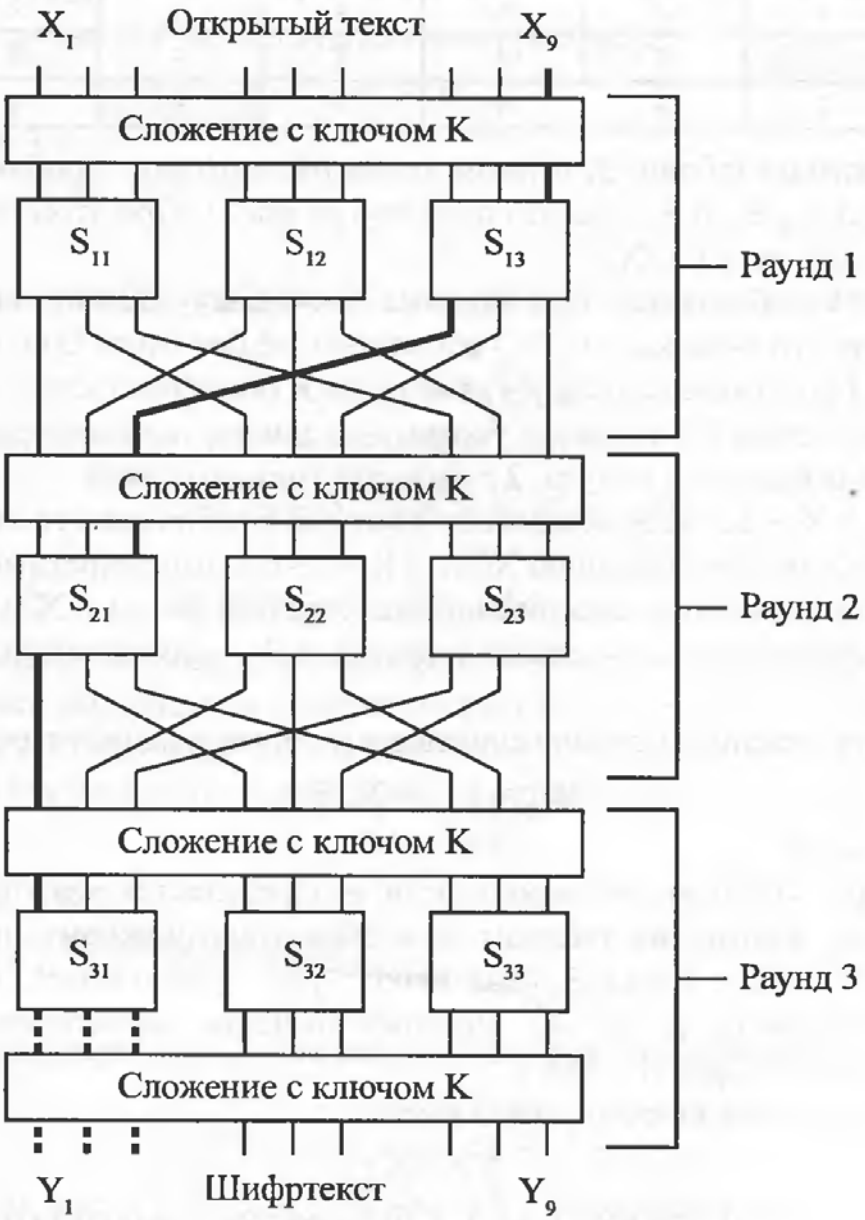


Рис. 1. ***Линейный криптоанализ шифровальной сети на основе подстановок и перемешивания***

Таблица 4

Результаты анализа S-блока для проведения линейного криптоанализа

j / i	1	2	3	4	5	6	7
1	4	4	4	0	4	4	4
2	4	2	2	4	4	6	2
3	4	2	6	4	4	6	6
4	4	6	6	4	4	6	2
5	4	2	6	4	4	2	2
6	0	4	4	4	4	4	4
7	4	4	4	4	8	4	4

Итак, согласно таблице 3, первым делом рассмотрим приближение, включающее блоки S_{13}, S_{21} и S_{31}, как это показано на рис. 1. При этом входным битом должно являться значение X_9.

Пусть $U_i(V_i)$ обозначает блок входных (выходных) данных цикла, содержащий 9 бит, для i-го S-блока, и $U_{ij}(V_{ij})$ обозначает j-й бит блока $U_i(V_i)$ (где биты нумеруются от 1 до 9 слева направо). Также пусть K обозначает ключ, который складывается по модулю 2 с входным блоком i-го цикла, за исключением ключа K, который складывается по модулю 2 с выходом третьего цикла.

Итак, $U_1 = X \oplus K$, где X обозначает 9-битный блок исходного открытого текста, и знак $\oplus$ обозначает операцию XOR, а K — 9-битный секретный ключ. В следующих далее уравнениях нижний индекс, стоящий рядом с X и K обозначает порядковый номер бита (1 — самый старший, а 9 — самый младший, соответственно).

Используя линейное приближение первого цикла, мы получаем:

$$V_{1,7} = U_{1,9} = X_9 \oplus K_9 \qquad (1)$$

с вероятностью 0.

Напомним, что значение вероятности p определяется как отношение числа совпадений, взятого из таблицы 4, к общему возможному числу совпадений (см. п. 5.1). Для блока S_{13} мы используем приближение, соответствующе паре векторов (i, j) = (1, 4). Поэтому значение вероятности $p_1 = 0/8 = 0$. Отклонение $\Delta = |1 - 2p| = |1 - 0| = 1$.

Для приближения второго раунда имеем:

$$V_{2,1} = U_{2,3}$$

с вероятностью $p_2 = 0$. Так как $U_{2,3} = V_{1,7} \oplus K_3$, мы можем получить приближение для формулы $V_{2,1} = V_{1,7} \oplus K_3$ с вероятностью 0 и, объединив это с выражением (1), которое выполняется с вероятностью 0, получим:

$$V_{2,1} \oplus X_9 \oplus K_9 \oplus K_3 = 0. \qquad (2)$$

Выражение (2) выполняется с вероятностью $p = 1/2 + 2^{n-1}(p_1 - 1/2)(p^2 - 1/2)\ldots$ $\ldots(p_n - 1/2) = 1/2 + 2^1(0 - 1/2)(0 - 1/2) = 1$ (и отклонением 1) по Накопительной лемме.

Согласно таблице 3, мы должны рассмотреть всевозможные выходные комбинации блока S_{31}, кроме выхода, равного 111. Мы знаем, что входом в блок S_{31} будет являться значение 100 в двоичном виде, или 4 в десятичном. Пользуясь таблицей 4, определяем, что при таком входном значении блока замены возможно 4 варианта выходных значений, а именно: 010, 011, 110 и 111. Так как по условию, мы не должны рассматривать выходное значение, равное 111, то остается рассмотреть три случая.

В первом случае для третьего цикла мы определяем:

$$V_{3,2} = U_{3,1}$$

с вероятностью 3/4.

Так как $U_{3,1} = V_{2,1} \oplus K_1$, то

$$V_{3,2} \oplus V_{2,1} \oplus K_1 = 0. \qquad (3)$$

Теперь объединим (3) и (2) для соединения всех трех приближений S-блоков. Мы получим:

$$V_{3,2} \oplus X_9 \oplus K_9 \oplus K_3 \oplus K_1 = 0.$$

Заметим, что $V_{3,2} = K_2 \oplus Y_2$, тогда можно записать:

$$K_2 \oplus Y_2 \oplus X_9 \oplus K_9 \oplus K_3 \oplus K_1 = 0,$$

или

$$Y_2 \oplus X_9 = K_2 \oplus K_9 \oplus K_3 \oplus K_1.$$

Используя Накопительную лемму (см. п. 5.3.2), определим, что приведенное выше выражение выполняется с вероятностью $p = 1/2 + 2^{n-1}(p_1 - 1/2)(p_2 - 1/2)\ldots$ $\ldots(p_n - 1/2) = 1/2 + 2^2(0 - 1/2)^2(3/4 - 1/2) = 3/4$.

Во втором случае для третьего цикла мы определяем:

$$V_{3,2} \oplus V_{3,3} = U_{3,1}$$

с вероятностью 3/4.

Так как $U_{3,1} = V_{2,1} \oplus K_1$, то

$$V_{3,2} \oplus V_{3,3} \oplus V_{2,1} \oplus K_1 = 0. \qquad (4)$$

Теперь объединим (4) и (2) для соединения всех трех приближений S-блоков. Мы получим:

$$V_{3,2} \oplus V_{3,3} \oplus X_9 \oplus K_9 \oplus K_3 \oplus K_1 = 0.$$

Заметим, что $V_{3,2} = K_2 \oplus Y_2$ и $V_{3,3} = K_3 \oplus Y_3$, тогда можно записать:

$$K_2 \oplus Y_2 \oplus K_3 \oplus Y_3 \oplus X_9 \oplus K_9 \oplus K_3 \oplus K_1 = 0,$$

или

$$Y_2 \oplus Y_3 \oplus X_9 = K_2 \oplus K_9 \oplus K_3 \oplus K_3 \oplus K_1 = K_2 \oplus K_9 \oplus K_1.$$

Используя Накопительную лемму, определим, что приведенное выше выражение выполняется с вероятностью $p = 1/2 + 2^{n-1}(p_1 - 1/2)(p_2 - 1/2)\ldots$ $\ldots(p_n - 1/2) = 1/2 + 2^2(0 - 1/2)^2(3/4 - 1/2) = 3/4$.

И, наконец, в третьем случае для третьего цикла мы определяем:

$$V_{3,1} \oplus V_{3,2} = U_{3,1}$$

с вероятностью 3/4.

Так как $U_{3,1} = V_{2,1} \oplus K_1$, то

$$V_{3,1} \oplus V_{3,2} \oplus V_{2,1} \oplus K_1 = 0. \quad (5)$$

Теперь объединим (5) и (2) для соединения всех трех приближений S-блоков. Мы получим:

$$V_{3,1} \oplus V_{3,2} \oplus X_9 \oplus K_9 \oplus K_3 \oplus K_1 = 0.$$

Заметим, что $V_{3,1} = K_1 \oplus Y_1$ и $V_{3,2} = K_2 \oplus Y_2$, тогда можно записать:

$$K_1 \oplus Y_1 \oplus K_2 \oplus Y_2 \oplus X_9 \oplus K_9 \oplus K_3 \oplus K_1 = 0,$$

или

$$Y_1 \oplus Y_2 \oplus X_9 = K_2 \oplus K_9 \oplus K_3 \oplus K_1 \oplus K_1 = K_2 \oplus K_9 \oplus K_3.$$

Используя Накопительную лемму, определим, что приведенное выше выражение выполняется с вероятностью $p = 1/2 + 2^{n-1}(p_1 - 1/2)(p^2 - 1/2)\ldots(p_n - 1/2) =$ $= 1/2 + 2^2(0 - 1/2)^2(3/4 - 1/2) = 3/4$.

Далее, согласно таблице 3, мы должны рассмотреть приближение, включающее блоки S_{13}, S_{21}, S_{23}, и S_{31}, как это показано на рис. 2. При этом входными битами должны являться значения X_7, X_8 и X_9.

Итак, $U_1 = X \oplus K$, где X обозначает 9-битный блок исходного открытого текста, и знак $\oplus$ обозначает операцию XOR. Используя линейное приближение первого цикла, мы получаем:

$$V_{1,7} \oplus V_{1,9} = U_{1,7} \oplus U_{1,8} \oplus U_{1,9} = X_7 \oplus K_7 \oplus X_8 \oplus K_8 \oplus X_9 \oplus K_9 \quad (6)$$

с вероятностью 1. Для приближения второго раунда имеем:

$$V_{2,1} = U_{2,3}$$

с вероятностью 0 и

$$V_{2,7} = U_{2,9}$$

с вероятностью 0.

Так как $U_{2,3} = V_{1,7} \oplus K_3$, а $U_{2,9} = V_{1,9} \oplus K_9$, то мы можем получить приближения для формулы $V_{2,1} = V_{1,7} \oplus K_3$ с вероятностью 0 и для формулы $V_{2,7} = V_{1,9} \oplus K_9$ с вероятностью 0. Объединив это с выражением (6), которое выполняется с вероятностью 1, получим:

$$V_{2,1} \oplus K_3 \oplus V_{2,7} \oplus K_9 \oplus X_7 \oplus K_7 \oplus X_8 \oplus K_8 \oplus X_9 \oplus K_9 = 0. \quad (7)$$

Теперь можно рассмотреть два случая. В первом случае для третьего цикла мы определяем:

$$V_{3,2} = U_{3,1} \oplus U_{3,3}$$

с вероятностью 1/4.

Так как $U_{3,1} = V_{2,1} \oplus K_1$ и $U_{3,3} = V_{2,7} \oplus K_3$, то

$$V_{3,2} \oplus V_{2,1} \oplus K_1 \oplus V_{2,7} \oplus K_3 = 0. \quad (8)$$

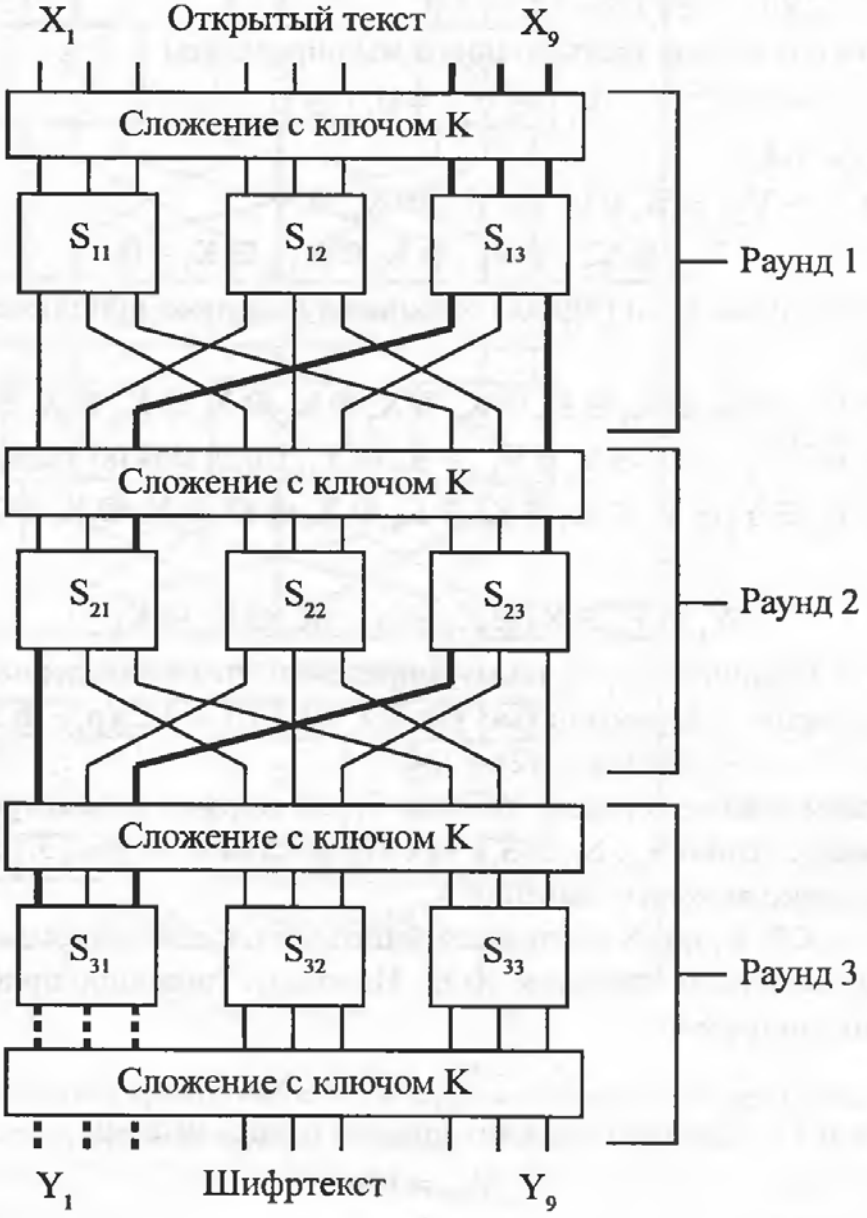


Рис. 2. ***Линейный криптоанализ шифровальной сети на основе подстановок и перемешивания***

Теперь объединим (7) и (8) для соединения всех трех приближений S-блоков. Мы получим:

$$V_{3,2} \oplus K_1 \oplus K_3 \oplus K_3 \oplus K_9 \oplus X_7 \oplus K_7 \oplus X_8 \oplus K_8 \oplus X_9 \oplus K_9 = 0.$$

Заметим, что $V_{3,2} = K_2 \oplus Y_2$, тогда можно записать:

$$K_2 \oplus Y_2 \oplus K_1 \oplus K_3 \oplus K_3 \oplus K_9 \oplus X_7 \oplus K_7 \oplus X_8 \oplus K_8 \oplus X_9 \oplus K_9 = 0,$$

или

$$Y_2 \oplus X_7 \oplus X_8 \oplus X_9 = K_1 \oplus K_7 \oplus K_8 \oplus K_2.$$

Используя Накопительную лемму, определим, что приведенное выше выражение выполняется с вероятностью $p = 1/2 + 2^{n-1}(p_1 - 1/2)(p_2 - 1/2)\ldots(p_n - 1/2) = = 1/2 + 2^3(1 - 1/2)(0 - 1/2)(1/4 - 1/2) = 1/4$.

Во втором случае для третьего цикла мы определяем:

$$V_{3,1} \oplus V_{3,2} = U_{3,1} \oplus U_{3,3}$$

с вероятностью 1/4.

Так как $U_{3,1} = V_{2,1} \oplus K_1$ и $U_{3,3} = V_{2,7} \oplus K_3$, то

$$V_{3,1} \oplus V_{3,2} \oplus V_{2,1} \oplus K_1 \oplus V_{2,7} \oplus K_3 = 0. \quad (9)$$

Теперь объединим (7) и (9) для соединения всех трех приближений S-блоков. Мы получим:

$$V_{3,1} \oplus V_{3,2} \oplus K_1 \oplus K_3 \oplus K_3 \oplus K_9 \oplus X_7 \oplus K_7 \oplus X_8 \oplus K_8 \oplus X_9 \oplus K_9 = 0.$$

Заметим, что $V_{3,1} = K_1 \oplus Y_1$ и $V_{3,2} = K_2 \oplus Y_2$, тогда можно записать:

$$K_1 \oplus Y_1 \oplus K_2 \oplus Y_2 \oplus K_1 \oplus K_3 \oplus K_3 \oplus K_9 \oplus X_7 \oplus K_7 \oplus X_8 \oplus K_8 \oplus X_9 \oplus K_9 = 0,$$

или

$$Y_1 \oplus Y_2 \oplus X_7 \oplus X_8 \oplus X_9 = K_7 \oplus K_8 \oplus K_2.$$

Используя Накопительную лемму, определим, что приведенное выше выражение выполняется с вероятностью $p = 1/2 + 2^{n-1}(p_1 - 1/2)(p_2 - 1/2)\ldots(p_n - 1/2) = = 1/2 + 2^3(1 - 1/2)(0 - 1/2)^2(1/4 - 1/2) = 1/4$.

Следующим шагом, согласно таблице 3, мы должны рассмотреть приближение, включающее блоки S_{12}, S_{22} и S_{32}, как это показано на рис. 3. При этом входным битом должно являться значение X_5.

Итак, $U_1 = X \oplus K$, где X обозначает 9-битный блок исходного открытого текста, и знак $\oplus$ обозначает операцию XOR. Используя линейное приближение первого цикла, мы получаем:

$$V_{1,5} = U_{1,5} = X_5 \oplus K_5 \quad (10)$$

с вероятностью 1/4. Для приближения второго раунда имеем:

$$V_{2,5} = U_{2,5}$$

с вероятностью 1/4.

Так как $U_{2,5} = V_{1,5} \oplus K_5$, то мы можем получить приближения для формулы $V_{2,5} = V_{1,5} \oplus K_5$. Объединив это с выражением (10), которое выполняется с вероятностью 1/4, получим:

$$V_{2,5} \oplus K_5 \oplus X_5 \oplus K_5 = V_{2,5} \oplus X_5 = 0. \quad (11)$$

Теперь можно рассмотреть два случая. В первом случае для третьего цикла мы определяем:

$$V_{3,5} \oplus V_{3,6} = U_{3,5}$$

с вероятностью 1/4.

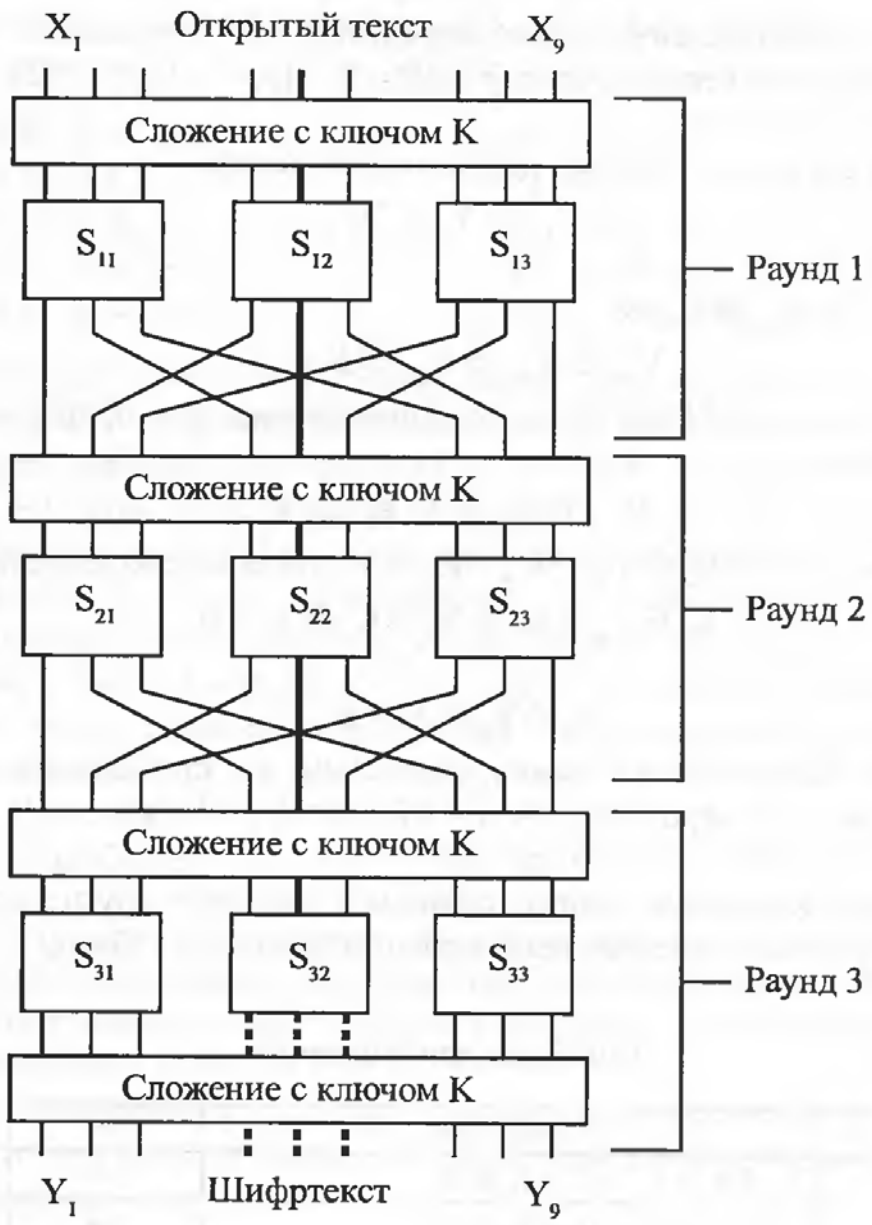


Рис. 3. ***Линейный криптоанализ шифровальной сети на основе подстановок и перемешивания***

Так как $U_{3,5} = V_{2,5} \oplus K_5$, то

$$V_{3,5} \oplus V_{3,6} \oplus V_{2,5} \oplus K_5 = 0. \qquad (12)$$

Теперь объединим (11) и (12) для соединения всех трех приближений S-блоков. Мы получим:

$$V_{3,5} \oplus V_{3,6} \oplus K_5 \oplus X_5 = 0.$$

Заметим, что $V_{3,5} = K_5 \oplus Y_5$ и $V_{3,6} = K_6 \oplus Y_6$, тогда можно записать:

$$K_5 \oplus Y_5 \oplus K_6 \oplus Y_6 \oplus K_5 \oplus X_5 = 0,$$

или

$$Y_5 \oplus Y_6 \oplus X_5 = K_6.$$

Используя Накопительную лемму, определим, что приведенное выше выражение выполняется с вероятностью $p = 1/2 + 2^{n-1}(p_1 - 1/2)(p_2 - 1/2)\ldots(p_n - 1/2) = 1/2 + 2^2(1/4 - 1/2)^3 = 7/16$.

Во втором случае для третьего цикла мы определяем:

$$V_{3,4} \oplus V_{3,5} = U_{3,5}$$

с вероятностью 3/4.

Так как $U_{3,5} = V_{2,5} \oplus K_5$, то

$$V_{3,4} \oplus V_{3,5} \oplus V_{2,5} \oplus K_5 = 0. \quad (13)$$

Теперь объединим (11) и (13) для соединения всех трех приближений S-блоков. Мы получим:

$$V_{3,4} \oplus V_{3,5} \oplus K_5 \oplus X_5 = 0.$$

Заметим, что $V_{3,4} = K_4 \oplus Y_4$ и $V_{3,5} = K_5 \oplus Y_5$, тогда можно записать:

$$K_4 \oplus Y_4 \oplus K_5 \oplus Y_5 \oplus K_5 \oplus X_5 = 0,$$

или

$$Y_4 \oplus Y_5 \oplus X_5 = K_4.$$

Используя Накопительную лемму, определим, что приведенное выше выражение выполняется с вероятностью $p = 1/2 + 2^{n-1}(p_1 - 1/2)(p_2 - 1/2)\ldots(p_n - 1/2) = 1/2 + 2^2(1/4 - 1/2)^2(3/4 - 1/2) = 9/16$.

Аналогично, используя данные таблицы 3, строятся другие приближения. Приближения рассматриваемого нами варианта сведены в таблицу 5.

Таблица 5

Линейные приближения

№	Уравнение	p
1	$Y_2 \oplus X_9 = K_2 \oplus K_9 \oplus K_3 \oplus K_1$.	3/4
	$Y_2 \oplus Y_3 \oplus X_9 = K_2 \oplus K_9 \oplus K_1$.	3/4
	$Y_1 \oplus Y_2 \oplus X_9 = K_2 \oplus K_9 \oplus K_3$.	3/4
2	$Y_2 \oplus X_7 \oplus X_8 \oplus X_9 = K_1 \oplus K_7 \oplus K_8 \oplus K_2$.	1/4
	$Y_1 \oplus Y_2 \oplus X_7 \oplus X_8 \oplus X_9 = K_7 \oplus K_8 \oplus K_2$.	1/4
3	$Y_5 \oplus Y_6 \oplus X_5 = K_6$	7/16
	$Y_4 \oplus Y_5 \oplus X_5 = K_4$	9/16
4	$Y_8 \oplus Y_9 \oplus X_1 \oplus X_4 = K_1 \oplus K_5 \oplus K_9$	5/8
5	$Y_5 \oplus X_2 = K_2 \oplus K_4$	9/16
6	$Y_3 \oplus X_8 = K_8 \oplus K_6 \oplus K_2 \oplus K_1$	1/4

Программа, разработанная для проведения данной лабораторной работы, производит зашифрование текстов на секретном ключе варианта. Далее, введя в программу, разработанную для проведения лабораторной работы, построенные ли-

нейные аналоги и вероятности, с которыми они выполняются, можно получить, чему будут равны правые части построенных уравнений. Алгоритм определения битов ключа приведен в п. 5.2.2. Так, например, в нашем случае получаем, что:

1) $K_2 \oplus K_9 \oplus K_3 \oplus K_1 = 0$;
2) $K_2 \oplus K_9 \oplus K_1 = 1$;
3) $K_2 \oplus K_9 \oplus K_3 = 0$;
4) $K_1 \oplus K_7 \oplus K_8 \oplus K_2 = 1$;
5) $K_7 \oplus K_8 \oplus K_2 = 1$;
6) $K_6 = 0$;
7) $K_4 = 0$;
8) $K_1 \oplus K_5 \oplus K_9 = 1$;
9) $K_2 \oplus K_4 = 0$;
10) $K_8 \oplus K_6 \oplus K_2 \oplus K_1 = 0$.

Сопоставив уравнения (1) и (2), сразу можно сделать вывод, что $K_3 = 1$. Аналогичным образом, сопоставив уравнения (4) и (5), находим, что $K_1 = 0$. Так как $K_4 = 0$, то из уравнения (9) получаем, что $K_2 = 0$. Зная K_1 и K_2, из уравнения (2) находим, что $K_9 = 1$. Зная K_1, K_2 и K_6, из уравнения (10) находим, что $K_8 = 0$. Подставив значения $K_1 = 0$ и $K_9 = 1$ в уравнение (8), получаем, что $K_5 = 0$. И, наконец, подставив значения $K_8 = 0$ и $K_2 = 0$ в уравнение (5), получаем, что $K_7 = 1$.

Таким образом, мы определили секретный ключ, с помощью которого производилось зашифрование. Он равен K = 001000101. Проверить правильность найденного ключа можно с помощью программы, предназначенной для выполнения лабораторной работы.

Подготовка к работе

Номер варианта индивидуального задания назначается преподавателем. Согласно варианту, при домашней подготовке необходимо выполнить следующее:

1. Для блока замены провести анализ и построить соответствующую таблицу (аналогично таблице 4);
2. С помощью заданных исходных данных провести предварительный анализ алгоритма шифрования с целью построения линейных статистических аналогов.

Методические указания по выполнению лабораторной работы

При выполнении лабораторной работы используется программа Crypto4.exe. С помощью данной программы для каждого варианта производится зашифрование открытых текстов на секретном ключе варианта. После этого пользователь имеет возможность сохранить либо напечатать полученные пары *открытый—закрытый текст*.

Анализ полученных текстов может быть проведен для каждого из полученных эффективных уравнений после того, как будут указаны биты (согласно данному уравнению), участвующие в анализе (при этом биты X_1 и Y_1 обозначают самый старший бит, а биты X_9 и Y_9 — соответственно, самый младший бит), а также вероятность, с которой данное уравнение выполняется. Результатом анализа бу-

дет являться заполнение таблицы результата, состоящей из четырех полей. Первое поле, обозначенное буквой N, отображает количество текстов, участвующих в анализе; второе поле, обозначенное буквой P, по сути дела отображает введенную пользователем вероятность; третье поле, обозначенное буквой T, показывает число открытых текстов, для которых левая часть эффективного уравнения равна 0; и последнее четвертое поле содержит результат анализа, а именно: то значение, которому соответствует правая часть данного эффективного уравнения.

Правильность анализа может быть проверена введением всех найденных битов в Форму проверки. В случае, когда в таблицах, построенных в ходе проведения линейного анализа, равномерно распределены вероятности, определить биты ключа достаточно сложно. В этом случае допускается биты, которые определить достаточно сложно из-за недостаточности данных, пометить в Форме проверки как «x». Если все биты, которые было возможно определить, найдены верно, то программа выдаст соответствующее сообщение, которое необходимо предъявить преподавателю для допуска к защите работы.

Рекомендуемый порядок работы

1. Вызвать программу Crypto4.exe.
2. Ввести Ф.И.О. студента, номер группы и номер варианта.
3. Произвести зашифрование данных.
4. Для каждого статистического аналога провести анализ и определить, чему равна его правая часть.
5. Согласно полученным данным, определить значение секретного ключа.
6. Проверить правильность найденного ключа (если получилось несколько возможных вариантов ключа, то осуществлять проверку до тех пор, пока не будет найден правильный ключ).

Содержание отчета

1. Титульный лист с указанием варианта задания.
2. Цель работы.
3. Таблица с результатами анализа блока замены.
4. Найденные статистические аналоги в соответствии с исходными данными варианта.
5. Результаты анализа с использованием найденных статистических аналогов.
6. Найденные биты ключа.
7. Выводы по проделанной работе.

Варианты индивидуальных заданий

Вариант № 1

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	4	0	7	3	5	1	2	6

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_2	$(S_{11}) \rightarrow (S_{22}) \rightarrow (S_{32}, S_{33})$	
2	X_1, X_2	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{32}, S_{33})$	1) Выход блока S_{32} — 100, выход блока S_{33} — 100 2) Выход блока S_{32} — 100, выход блока S_{33} — 101
3	X_1	$(S_{11}) \rightarrow (S_{22}, S_{23}) \rightarrow (S_{32}, S_{33})$	1) Выход блока S_{32} — 100, выход блока S_{33} — 100 2) Выход блока S_{32} — 110, выход блока S_{33} — 100 3) Выход блока S_{32} — 111, выход блока S_{33} — 100
4	X_9	$(S_{13} \rightarrow (S_{21}) \rightarrow (S_{31})$	
5	X_1, X_2, X_4, X_5	$(S_{11}, S_{12}) \rightarrow (S_{23}) \rightarrow (S_{33})$	Выход блока S_{33} — 100
6	X_3	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{32}, S_{33})$	

Вариант № 2

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	4	7	5	1	2	3	6	0

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_3	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{33})$	Всевозможные выходные комбинации блока S_{33}
2	X_3	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{31})$	Выходы блока S_{31} — 100 и 110
3	X_1	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{33})$	Выход блока S_{33} — 111
4	X_3	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{32})$	Выходы блока S_{32} — 110 и 011
5	X_1	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{32})$	Выход блока S_{32} — 010

Вариант № 3

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	4	2	6	0	1	3	7	5

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_1	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{33})$	Всевозможные выходные комбинации блока S_{33}, кроме выхода 100
2	X_1, X_2, X_3	$(S_{11}) \rightarrow (S_{22}, S_{23}) \rightarrow (S_{33})$	
3	$X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9$	$(S_{11}, S_{12}, S_{13}) \rightarrow (S_{22}, S_{23}) \rightarrow (S_{32}, S_{33})$	
4	X_5, X_6, X_8, X_9	$(S_{12}, S_{13}) \rightarrow (S_{22}) \rightarrow (S_{32})$	Всевозможные выходные комбинации блока S_{32}, кроме выхода 111
5	X_5, X_6	$(S_{12}) \rightarrow (S_{22}) \rightarrow (S_{31})$	Выход блока S_{31} — 100

Вариант № 4

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	6	3	4	1	0	7	2	5

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_3	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{31}, S_{33})$	
2	X_1	$(S_{11}) \rightarrow (S_{21}, S_{23}) \rightarrow (S_{31}, S_{33})$	
3	X_2, X_8	$(S_{11}, S_{13}) \rightarrow (S_{22}) \rightarrow (S_{31})$	Выход блока S_{31} — 010
4	X_8	$(S_{13}) \rightarrow (S_{22}) \rightarrow (S_{33})$	Выход блока S_{33} — 010
5	X_6	$(S_{12}) \rightarrow (S_{23}) \rightarrow (S_{32})$	
6	X_5	$(S_{12}) \rightarrow (S_{22}) \rightarrow (S_{32})$	Выходы блока S_{32} — 011 и 110
7	X_4	$(S_{12}) \rightarrow (S_{21}, S_{23}) \rightarrow (S_{33})$	
8	X_1, X_3	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{31}, S_{33})$	
9	X_7, X_9	$(S_{13}) \rightarrow (S_{21}) \rightarrow (S_{33})$	
10	X_2, X_8	$(S_{11}, S_{13}) \rightarrow (S_{22}) \rightarrow (S_{31})$	Выход блока S_{31} — 011

Вариант № 5

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	5	0	1	4	7	2	6	3

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_1	$(S_{11}) \rightarrow (S_{22}) \rightarrow (S_{32})$	
2	X_4	$(S_{12}) \rightarrow (S_{22}) \rightarrow (S_{31}, S_{33})$	
3	X_5	$(S_{12}) \rightarrow (S_{21}, S_{23}) \rightarrow (S_{32}, S_{31})$	1) Выход блока S_{31} — 110, выход блока S_{32} — 110 2) Выход блока S_{31} — 011, выход блока S_{32} — 011 3) Выход блока S_{31} — 011, выход блока S_{32} — 001 4) Выход блока S_{31} — 001, выход блока S_{32} — 011 5) Выход блока S_{31} — 100, выход блока S_{32} — 100
4	X_3	$(S_{11}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{32})$	
5	X_8, X_9	$(S_{13}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{31})$	
6	X_7, X_8, X_9	$(S_{13}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{31})$	
7	X_6, X_9	$(S_{12}, S_{13}) \rightarrow (S_{21}) \rightarrow (S_{31})$	

Примечание. В результате анализа должно получиться два возможных значения ключа. Проверить правильность одного из них необходимо с помощью программы, предназначенной для выполнения данной лабораторной работы.

Вариант № 6

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	3	2	7	6	1	4	0	5

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_1	$(S_{11}) \rightarrow (S_{22}) \rightarrow (S_{32})$	Всевозможные выходные комбинации блока S_{32}, кроме 111
2	X_4	$(S_{12}) \rightarrow (S_{22}) \rightarrow (S_{31})$	Всевозможные выходные комбинации блока S_{31}, кроме 111
3	X_7	$(S_{13}) \rightarrow (S_{22}) \rightarrow (S_{33})$	Выходы блока S_{33} — 100 и 101
4	X_7	$(S_{13}) \rightarrow (S_{22}) \rightarrow (S_{31})$	Выход блока S_{31} — 100

Вариант № 7

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	0	3	6	5	7	2	4	1

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_3	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{31}, S_{32}, S_{33})$	1) Выход блока S_{31} — 011, выход блока S_{32} — 001, выход блока S_{33} — 001 2) Выход блока S_{31} — 100, выход блока S_{32} — 001, выход блока S_{33} — 001 3) Выход блока S_{31} — 100 выход блока S_{32} — 011, выход блока S_{33} — 001 4) Выход блока S_{31} — 001, выход блока S_{32} — 001, выход блока S_{33} — 011 5) Выход блока S_{31} — 001, выход блока S_{32} — 100, выход блока S_{33} — 001
2	X_9	$(S_{13}) \rightarrow (S_{21}) \rightarrow (S_{31})$	
3	X_6	$(S_{12}) \rightarrow (S_{23}) \rightarrow (S_{32})$	Выход блока S_{32} 001
4	X_6	$(S_{12}) \rightarrow (S_{23}) \rightarrow (S_{32})$	Выход блока S_{32} 011
5	X_2	$(S_{11}) \rightarrow (S_{22}) \rightarrow (S_{31}, S_{32}, S_{33})$	Выход блока S_{31} — 100, выход блока S_{32} — 010 Выход блока S_{33} — 010
6	X_6	$(S_{12}) \rightarrow (S_{21}) \rightarrow (S_{32})$	

Вариант № 8

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	6	7	4	3	2	5	1	0

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X^1X^2	$(S_{11}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{31})$	
2	X^4X^5	$(S_{12}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{32})$	
3	X^7X^8	$(S_{13}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{33})$	
4	X_3	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{32})$	Всевозможные выходные значения блока S_{32}, кроме значения 111
5	X_1	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{31})$	Выход блока S_{31} — 010
6	X_9	$(S_{13}) \rightarrow (S_{23}) \rightarrow (S_{33})$	Выход блока S_{33} — 011

Вариант № 9

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	5	1	4	0	2	6	7	3

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_4	$(S_{12}) \rightarrow (S_{22}) \rightarrow (S_{32}, S_{33})$	
2	X_1	$(S_{11}) \rightarrow (S_{22}) \rightarrow (S_{32})$	
3	X^1X^2	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{32})$	Всевозможные выходные значения блока S_{32}, кроме значения 111
4	X_5	$(S_{12}) \rightarrow (S_{22}, S_{23}) \rightarrow (S_{32}, S_{33})$	1) Выход блока S_{32} — 100, выход блока S_{33} — 100 2) Выход блока S_{32} — 101, выход блока S_{33} — 100 3) Выход блока S_{32} — 110, выход блока S_{33} — 110
5	X^7X^8	$(S_{13}) \rightarrow (S_{23}) \rightarrow (S_{31})$	Выход блока S_{31} — 100

Вариант № 10

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	7	6	3	2	0	1	5	4

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_2	$(S_{11}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{32})$	
2	X_5	$(S_{12}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{31}, S_{32})$	
3	X_3, X_6	$(S_{11}, S_{12}) \rightarrow (S_{23}) \rightarrow (S_{31})$	Всевозможные выходные значения блока S_{31}, кроме значения 011
4	X_3	$(S_{11}) \rightarrow (S_{23}) \rightarrow (S_{32})$	Выход блока S_{32} — 011
5	X_9	$(S_{13}) \rightarrow (S_{23}) \rightarrow (S_{33})$	Выходы блока S_{33} — 011 и 101

Вариант № 11

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	6	0	5	2	4	1	3	7

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_8, X_9	$(S_{13}) \rightarrow (S_{23}) \rightarrow (S_{31})$	Всевозможные выходные значения блока S_{31}, кроме значения 111
2	X_5	$(S_{12}) \rightarrow (S_{23}) \rightarrow (S_{32})$	Всевозможные выходные значения блока S_{32}, кроме значения 101
3	X_5	$(S_{12}) \rightarrow (S_{23}) \rightarrow (S_{33})$	Выходы блока S_{33} — 100 и 110

Вариант № 12

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	5	2	0	3	6	1	4	7

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_9	$(S_{13}) \rightarrow (S_{23}) \rightarrow (S_{31})$	Выходы блока S_{31} — 010 и 100
2	X_9	$(S_{13}) \rightarrow (S_{23}) \rightarrow (S_{32})$	Выходы блока S_{32} — 010 и 111
3	X_9	$(S_{13}) \rightarrow (S_{23}) \rightarrow (S_{33})$	Выходы блока S_{33} — 010 и 100
4	X_5	$(S_{12}) \rightarrow (S_{22}, S_{23}) \rightarrow (S_{32}, S_{33})$	Выход блока S_{32} — 001, выход блока S_{33} — 001
5	X_2	$(S_{11}) \rightarrow (S_{22}, S_{23}) \rightarrow (S_{31})$	Выход блока S_{31} — 001
6	X_1	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{31})$	Выход блока S_{31} — 101

Вариант № 13

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	1	3	0	5	6	4	7	2

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_9	$(S_{13}) \rightarrow (S_{21}, S_{22}) \rightarrow (S_{31}, S_{32})$	1) Выход блока S_{31} — 111, выход блока S_{32} — 111 2) Выход блока S_{31} — 100, выход блока S_{32} — 100 3) Выход блока S_{31} — 010, выход блока S_{32} — 111
2	X_5, X_6	$(S_{12}) \rightarrow (S_{22}, S_{23}) \rightarrow (S_{31}, S_{33})$	Выход блока S_{31} — 011, выход блока S_{33} — 011
3	X_8, X_9	$(S_{13}) \rightarrow (S_{22}, S_{23}) \rightarrow (S_{31}, S_{32})$	Выход блока S_{31} — 011, выход блока S_{32} — 011
4	X_4	$(S_{12}) \rightarrow (S_{21}) \rightarrow (S_{31}, S_{33})$	1) Выход блока S_{31} — 111, выход блока S_{33} — 100 2) Выход блока S_{31} — 100, выход блока S_{33} — 100
5	X_1	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{32})$	Выход блока S_{32} - 010
6	X_7	$(S_{13}) \rightarrow (S_{21}) \rightarrow (S_{31}, S_{32})$	Выход блока S_{31} — 100, выход блока S_{32} — 100
7	X_1	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{33})$	Выходы блока S_{33} — 001 и 010

Вариант № 14

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	3	1	4	5	2	0	7	6

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_4	$(S_{12}) \rightarrow (S_{23}) \rightarrow (S_{31})$	Выходы блока S_{31} — 010 и 011
2	X_5	$(S_{12}) \rightarrow (S_{21}) \rightarrow (S_{31})$	Всевозможные выходные комбинации блока S_{31}, кроме значения 111
3	X_8	$(S_{13}) \rightarrow (S_{21}) \rightarrow (S_{32})$	Выходы блока S_{32} — 001 и 010
4	X_2	$(S_{11}) \rightarrow (S_{21}) \rightarrow (S_{33})$	Всевозможные выходные комбинации блока S_{33}, кроме значения 001

Вариант № 15

Блок замены:

Вход	0	1	2	3	4	5	6	7
Выход	5	7	4	3	1	6	0	2

Исходные данные варианта

№	Входные биты	Сцепление блоков	Выходные биты
1	X_9	$(S_{13}) \rightarrow (S_{22}) \rightarrow (S_{32})$	Всевозможные выходы блока S_{32}, кроме значения 100
2	X_8, X_7	$(S_{13}) \rightarrow (S_{21}, S_{23}) \rightarrow (S_{32})$	Выход блока S_{32} — 001
3	X_7, X_8, X_9	$(S_{13}) \rightarrow (S_{21}, S_{22}, S_{23}) \rightarrow (S_{32})$	Выход блока S_{32} — 111
4	X_3	$(S_{11}) \rightarrow (S_{22}) \rightarrow (S_{31})$	Всевозможные выходы блока S_{32}, кроме значения 110
5	X_6	$(S_{12}) \rightarrow (S_{23}) \rightarrow (S_{31})$	Выход блока S_{31} — 100
6	X_6	$(S_{12}) \rightarrow (S_{22}) \rightarrow (S_{33})$	Выход блока S_{33} — 110 и 001

Контрольные вопросы

1. Что такое линейный криптоанализ?
2. Опишите принцип работы алгоритма шифрования Rijndael.
3. Что такое линейный статистический аналог нелинейной функции? Ответ поясните.
4. Как строятся эффективные линейные статистические аналоги для алгоритмов шифрования, построенных по принципу сети SPN?
5. Приведите алгоритм нахождения битов ключа при найденных эффективных линейных уравнениях.
6. По какому принципу строятся таблицы статистического анализа при проведении линейного криптоанализа?
7. Как определяется вероятность и отклонение Δ для конечного уравнения, зависящего только от битов входного и выходного сообщений, а также от битов секретного ключа?

ЛАБОРАТОРНАЯ РАБОТА № 5

Изучение метода слайдовой атаки на примере алгоритмов шифрования, построенных по схеме Фейстеля

Цель работы — изучить применение метода слайдовой атаки к многораундовым алгоритмам блочного шифрования, построенным по схеме Фейстеля.

Пример выполнения лабораторной работы

Исходные данные варианта, который мы рассмотрим, приведены в таблицах 1–5, они представляют собой таблицы перестановки с расширением, простой перестановки и таблицы замены, используемые в анализируемом алгоритме шифрования. Задана также таблица с маской, позволяющей определить слайдовые пары для проведения успешной атаки.

Таблица 1

Таблица перестановки с расширением

3	1	4	3	2	1	4	2

Таблица 2

Таблица перестановки

4	2	3	1

Таблица 3

S_1-блок

	00	**01**	**10**	**11**
00	0	2	1	1
01	1	3	0	2
10	0	3	2	3
11	2	1	3	0

Таблица 4

S_2-блок

	00	**01**	**10**	**11**
00	0	1	3	2
01	3	2	0	1
10	1	0	1	3
11	3	2	0	2

Таблица 5

Маска

1	0	1	0

Так как алгоритм S-DES (см. п. 3.1.1) является блочным алгоритмом шифрования, построенным по схеме Фейстеля, то мы вполне можем применить к нему слайдовую атаку. Для простоты работы будем использовать один и тот же фиксированный 8-битовый ключ K (то есть будем опускать процедуру извлечения 8-битового подключа из исходного 10-битового). Мы опустим начальную и конечную перестановки, так как они не влияют на криптографическую стойкость алгоритма. И будем использовать не 2 цикла криптографического преобразования, а 37, так как данный вид криптоаналитической атаки не зависит от количества циклов, используемых в алгоритме.

С помощью программы, разработанной для проведения данной лабораторной работы, мы зашифруем тексты на секретном ключе данного варианта. После этого, введя в программу заданную нам маску, сможем отобрать пары текстов, подходящие под определение слайдовой пары. Условия выбора слайдовых пар приведены в п. 8.2. Маска вводится для сужения диапазона возможных слайдовых пар и облегчения работы. Это будут такие пары текстов, для которых правые четыре бита первого открытого текста будут равны левым четырем битам второго открытого текста и будут равны заданной маске, и при этом левые четыре бита первого шифртекста будут равны правым четырем битам второго шифртекста. Для рассматриваемого нами варианта, это будут пять пар текстов, приведенные в таблице 6.

Таблица 6

Слайдовые пары

№	X′	Y′	X	Y
1	10001010	10111000	10101000	10111011
2	10011010	10011010	10101001	11011001
3	10111010	10111010	10101000	10111011
4	11111010	10011111	10101001	11011001
5	11111010	10011111	10101111	10001001

Так как при анализе найденных пар нам придется работать с таблицами замены, используемыми в алгоритме шифрования, то для удобства работы мы сопоставили входы и выходы блоков замены так, как показано в таблице 7.

Таблица 7

Соответствие входов и выходов S-блоков для анализируемого алгоритма шифрования

Вход в S-блок	Выход S_1 блока	Выход S_2 блока
0000	00	00
0001	01	11
0010	10	01
0011	11	10
0100	01	11
0101	00	00
0110	01	10
0111	10	01
1000	00	01
1001	10	11
1010	11	00
1011	01	10
1100	10	01
1101	11	00
1110	11	11
1111	00	10

Рассмотрим первую пару текстов. Для этого рассмотрим первые два раунда шифрования, показанные на рис. 1.

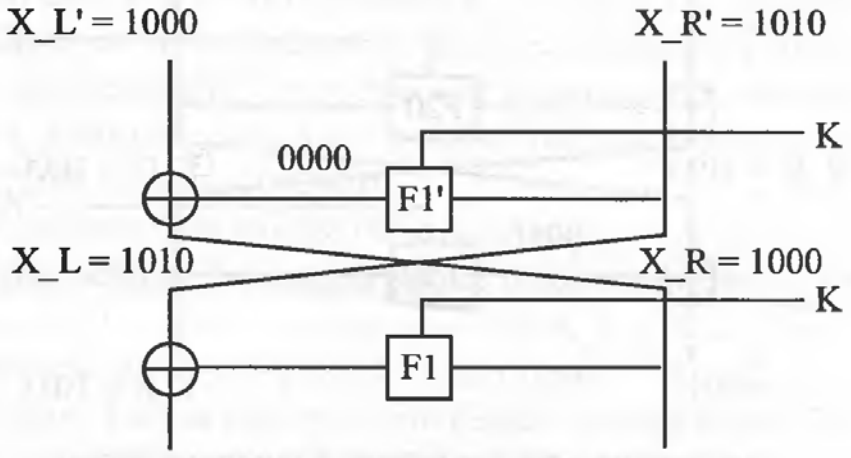


Рис. 1. ***Анализ первых раундов первой слайдовой пары***

То, что известны значения правой части одного открытого текста X_R′ и значение левой части другого открытого текста X_L, дает нам информацию о значении входа F_1' функции. Так как значения X_R и X_L′ тоже известны, то легко можно определить значение выхода функции F_1, которое будет равно 0000.

Так как перед выходом из функции F_1' данные подвергаются перестановке, согласно таблице 2, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 0000, то есть 00 является выходом S_1 блока, а 00 — выходом S_2 блока.

Входное сообщение функции F_1' подвергается перестановке с расширением, согласно таблице 1. А значит, вход 1010 преобразуется к значению 11010100, которое и будет сложено с ключом $K = (K_1, K_2)$. Так как в процессе шифрования текст разделяется на две части, каждая из которых складывается с частью ключа, а затем поступает на вход соответствующего S-блока, то для удобства мы представляем 8-битный секретный ключ шифрования K в виде двух 4-битных подключей K_1 и K_2, что и отражает запись $K = (K_1, K_2)$, то есть получается, что вход S_1 блока $1101 \oplus K_1$ даст на выходе значение 00, а вход S_2 блока $0100 \oplus K_2$ — значение 00.

Воспользовавшись таблицей 7, можно определить, что значение 00 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих значений: 0000, 0101, 1000 или 1111. Таким образом, добавив к каждому из возможных значений входа значение 1101, получим возможные значения K_1. Это будут значения 1101, 1000, 0101 или 0010.

Аналогичным образом определяем, что значение 00 появляется на выходе S_2 блока в случае, если на его вход поступило одно из следующих значений: 0000, 0101, 1010 или 1101. А значит, добавив к каждому из возможных значений входа значение 0100, получим возможные значения K_2. Это будут значения 0100, 0001, 1110 или 1001.

Теперь рассмотрим последние два раунда шифрования для этой же слайдовой пары, показанные на рис. 2.

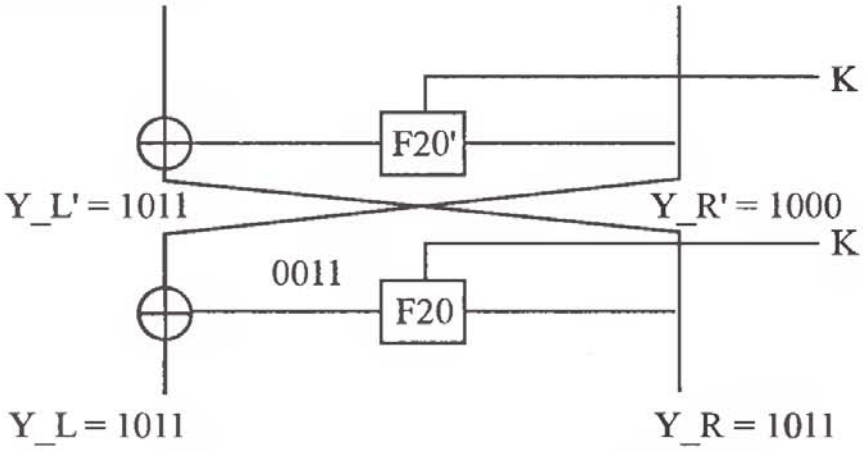


Рис. 2. ***Анализ последних раундов первой слайдовой пары***

То, что нам известны значения левой части одного закрытого текста Y_L′ и значение правой части другого закрытого текста Y_R, дает нам информацию о значении входа функции F_{20}'. Так как значения Y_R′ Y_L тоже известны, то легко можно определить значение выхода этой же F-функции, которое будет равно 0011.

Так как перед выходом из F-функции данные подвергаются перестановке, согласно таблице 2, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 1010, то есть 10 будет являться выходом S_1 блока, а 10 — выходом S_2 блока.

Входное сообщение F-функции подвергается перестановке с расширением, согласно таблице 1. А значит, вход F-функции 1011 преобразуется к значению 11110110, которое и будет сложено с ключом $K = (K_1, K_2)$, то есть получается, что вход S_1 блока $1111 \oplus K_1$ даст на выходе значение 10, а вход S_2 блока $0110 \oplus K_2$ — значение 10.

Воспользовавшись таблицей 7, можно определить, что значение 10 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих значений: 0010, 0111, 1001 или 1100. Таким образом, добавив к каждому из возможных значений входа значение 1111, получим возможные значения K_1. Это будут значения 1101, 1000, 0110 или 0011.

Аналогичным образом определяем, что значение 10 появляется на выходе S_2 блока в случае, если на его вход поступило одно из следующих значений: 0011, 0110, 1011 или 1111. А значит, добавив к каждому из возможных значений входа значение 0110, получим возможные значения K_2. Это будут значения 0101, 0000, 1101 или 1001.

Так как во всех раундах шифрования использовался один и тот же ключ, то значение K_1 первого раунда должно совпадать со значением K_1 последнего раунда, точно также как значение K_2 первого раунда должно совпадать со значением K_2 последнего раунда. Сопоставив, всевозможные значения K_1 и K_2, можно увидеть, что есть только два значения $K_1 = 1101$ и $K_1 = 1000$, использование которых возможно как в первом, так в последнем раундах, и только одно значение $K_2 = 1001$, использование которого также возможно как в первом, так и в последнем раундах. Таким образом, нами найдено два возможных значения искомого ключа K = 11011001 или K = 10001001.

Теперь проанализируем вторую пару текстов.

Для этого рассмотрим первые два раунда шифрования, показанные на рис. 3.

Исходя из того, что нам известны значения X_R и X_L′, легко можно определить, что значение выхода F-функции будет равно 0000, как и для первой рассматриваемой пары. Так как вход функции F определяется значением заданной изначально маски, то получается, что он будет совпадать со входом функции F для рассмотренной выше слайдовой пары. Таким образом, подключи K_1 и K_2 будут принимать те же значения, что и для первой рассмотренной пары текстов, то есть возможными значениями подключа K_1 будут являться значения 1101, 1000, 0101 или 0010, а подключа K_2 — значения 0100, 0001, 1110 или 1001.

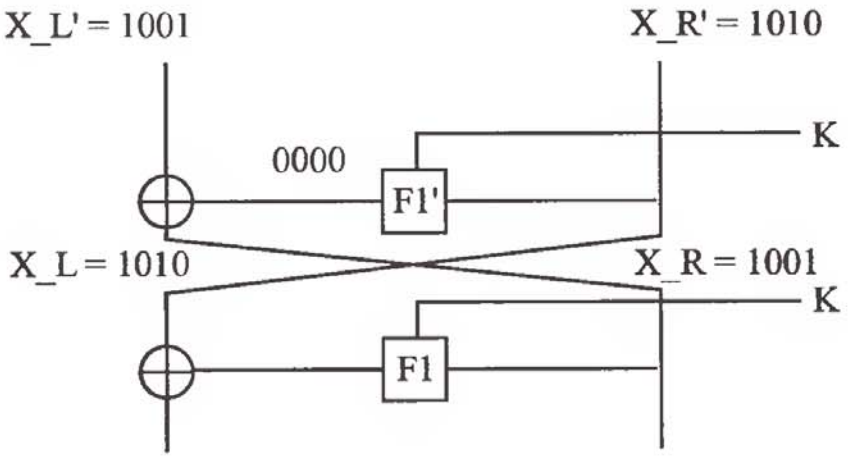


Рис. 3. ***Анализ первых раундов второй слайдовой пары***

Теперь рассмотрим последние два раунда шифрования для этой же слайдовой пары, показанные на рис. 4.

То, что нам известны значения Y_L′ и Y_R, дает нам информацию о значении входа функции F_{20}'. Так как значения Y_R′ Y_L тоже известны, то легко можно определить значение выхода этой же F-функции, которое будет равно 0011.

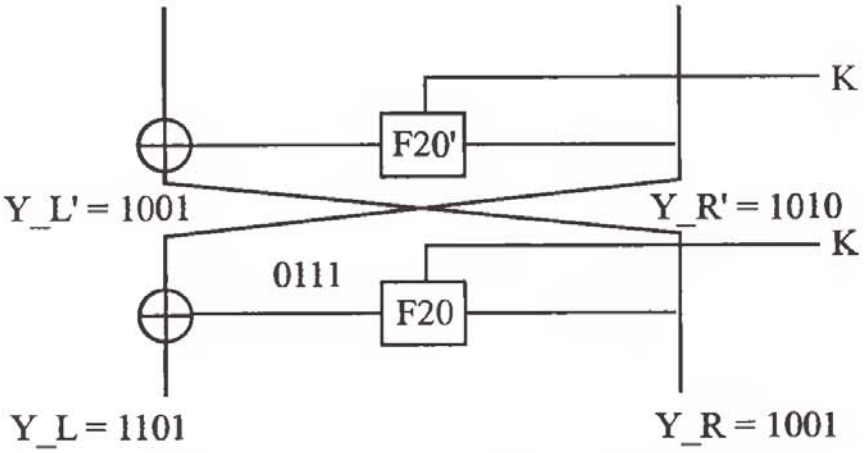


Рис. 4. ***Анализ последних раундов второй слайдовой пары***

Так как перед выходом из F-функции данные подвергаются перестановке, согласно таблице 2, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 1110, то есть 11 будет являться выходом S_1 блока, а 10 — выходом S_2 блока.

Входное сообщение F-функции подвергается перестановке с расширением, согласно таблице 1. А значит, вход F-функции 1001 преобразуется к значению 01100110, которое и будет сложено с ключом $K = (K_1, K_2)$, то есть получается, что вход S_1 блока $0110 \oplus K_1$ даст на выходе значение 11, а вход S_2 блока $0110 \oplus K_2$ — значение 10.

Воспользовавшись таблицей 7, можно определить, что значение 11 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих

значений: 0011, 1010, 1101 или 1110. Таким образом, добавив к каждому из возможных значений входа значение 0110, получим возможные значения K_1. Это будут значения 0101, 1100, 1011 или 1000.

Аналогичным образом определяем, что значение 10 появляется на выходе S_2 блока в случае, если на его вход поступило одно из следующих значений 0011, 0110, 1011 или 1111. А значит, добавив к каждому из возможных значений входа значение 0110, получим возможные значения K_2. Это будут значения 0101, 0000, 1101 или 1001.

Так как во всех раундах шифрования использовался один и тот же ключ, то значение K_1 первого раунда должно совпадать со значением K_1 последнего раунда, точно так же, как значение K_2 первого раунда должно совпадать со значением K_2 последнего раунда. Сопоставив всевозможные значения K_1 и K_2, можно увидеть, что есть только два значения $K_1 = 1000$ и $K_1 = 0101$, использование которых возможно как в первом, так в последнем раундах, и только одно значение $K_2 = 1001$, использование которого также возможно как первом, так и в последнем раундах. Таким образом, нами найдено два возможных значения искомого ключа K = 10001001 или K = 01011001.

Теперь проанализируем третью пару текстов.

Для этого рассмотрим первые два раунда шифрования, показанные на рис. 5.

То, что нам известны значения X_R′ и X_L, дает нам информацию о значении входа F_1' функции. Так как значения X_R и X_L′ тоже известны, то легко можно определить значение выхода F-функции, которое будет равно 0011.

Так как перед выходом из F-функции данные подвергаются перестановке, согласно таблице 2, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 1010, то есть 10 является выходом S_1 блока, а 10 — выходом S_2 блока.

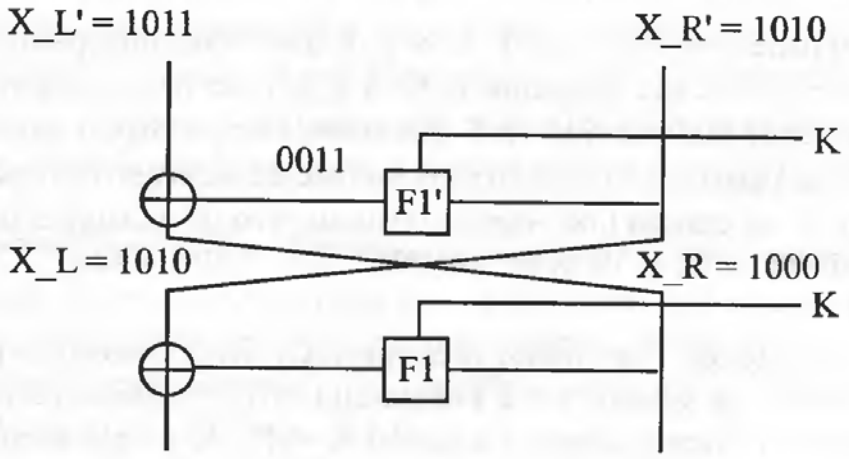


Рис. 5. ***Анализ первых раундов третьей слайдовой пары***

Входное сообщение F-функции подвергается перестановке с расширением, согласно таблице 1. А значит, вход F-функции 1010 преобразуется к значению 11010100, которое и будет сложено с ключом $K = (K_1, K_2)$, то есть получается, что

вход S_1 блока 1101 $\oplus K_1$ даст на выходе значение 10, и вход S_2 блока 0100 $\oplus K_2$ даст то же значение 10.

Воспользовавшись таблицей 7, можно определить, что значение 10 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих значений 0010, 0111, 1001 или 1100. Таким образом, добавив к каждому из возможных значений входа значение 1101, получим возможные значения K_1. Это будут значения 1111, 1010, 0100 или 0001.

Аналогичным образом определяем, что значение 10 появляется на выходе S_2 блока в случае, если на его вход поступило одно из следующих значений 0011, 0110, 1011 или 1111. А значит, добавив к каждому из возможных значений входа значение 0100, получим возможные значения K_2. Это будут значения 0111, 0010, 1111 или 1011.

Теперь рассмотрим последние два раунда шифрования для этой же слайдовой пары, показанные на рис. 6.

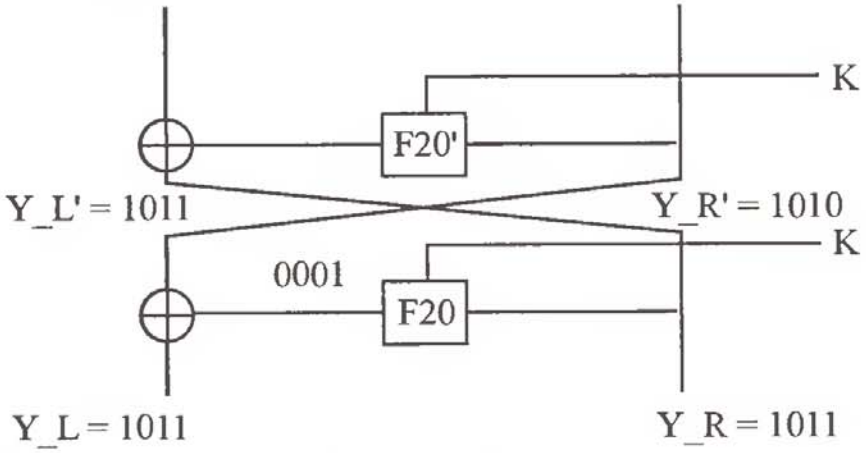


Рис. 6. ***Анализ последних раундов третьей слайдовой пары***

То, что нам известны значения Y_L′ и Y_R, дает нам информацию о значении входа функции F_{20}'. Так как значения Y_R′ и Y_L тоже известны, то легко можно определить значение выхода этой же F-функции, которое будет равно 0011.

Так как перед выходом из F-функции данные подвергаются перестановке, согласно таблице 2, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 1000, то есть 10 будет являться выходом S_1 блока, а 00 — выходом S_2 блока.

Входное сообщение F-функции подвергается перестановке с расширением, согласно таблице 1. А значит, вход F-функции 1011 преобразуется к значению 11110110, которое и будет сложено с ключом $K = (K_1, K_2)$, то есть получается, что вход S_1 блока 1111 $\oplus K_1$ даст на выходе значение 10, а вход S_2 блока 0110 $\oplus K_2$ — значение 00.

Воспользовавшись таблицей 7, можно определить, что значение 10 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих значений 0010, 0111, 1001 или 1100. Таким образом, добавив к каждому из воз-

можных значений входа значение 1111, получим возможные значения K_1. Это будут значения 1101, 1000, 0110 или 0011.

Сравнив возможные значения подключа K_1 для первого и последнего цикла, мы не найдем ни одного совпадения. Из чего можно сделать вывод, что данная пара не является слайдовой, и нет смысла ее анализировать дальше.

Перейдем к анализу четвертой пары текстов.

Для этого рассмотрим первые два раунда шифрования, показанные на рис. 7.

То, что нам известны значения X_R′ и X_L, дает нам информацию о значении входа F_1' функции. Так как значения X_R и X_L′ тоже известны, то легко можно определить значение выхода F-функции, которое будет равно 0110.

Так как перед выходом из F-функции данные подвергаются перестановке, согласно таблице 2, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 0101, то есть 01 является выходом S_1 блока, а 01 — выходом S_2 блока.

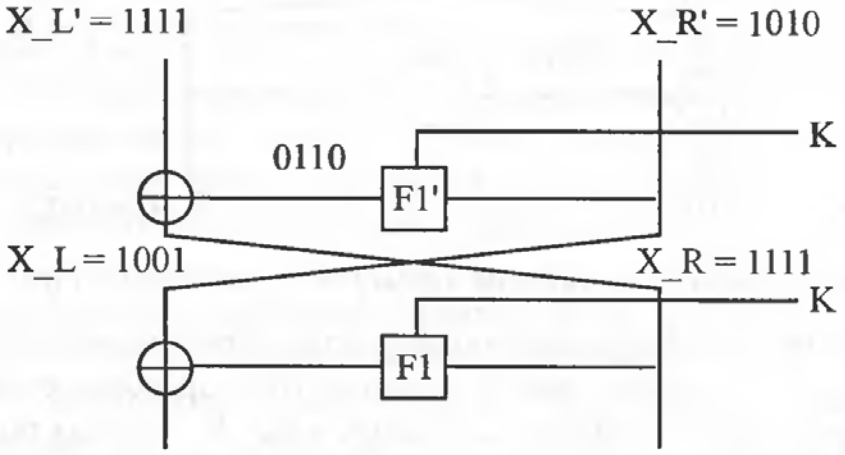


Рис. 7. ***Анализ первых раундов четвертой слайдовой пары***

Входное сообщение F-функции подвергается перестановке с расширением, согласно таблице 1. А значит, вход F-функции 1010 преобразуется к значению 11010100, которое и будет сложено с ключом $K = (K_1, K_2)$, то есть получается, что вход S_1 блока 1101 $\oplus$ K_1 даст на выходе значение 01, а вход S_2 блока 0100 $\oplus$ K_2 — значение 01.

Воспользовавшись таблицей 7, можно определить, что значение 01 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих значений 0001, 0100, 0110 или 1011. Таким образом, добавив к каждому из возможных значений входа значение 1101; получим возможные значения K_1. Это будут значения 1100, 1001, 1011 или 0110.

Аналогичным образом определяем, что значение 01 появляется на выходе S_2 блока в случае, если на его вход поступило одно из следующих значений 0110, 0011, 1100 или 1000. А значит, добавив к каждому из возможных значений входа значение 0100, получим возможные значения K_2. Это будут значения 0110, 0011, 1100 или 1000.

Теперь рассмотрим последние два раунда шифрования для этой же слайдовой пары, показанные на рис. 8.

То, что нам известны значения Y_L′ и Y_R, дает нам информацию о значении входа функции F_{20}'. Так как значения Y_R′ и Y_L тоже известны, то легко можно определить значение выхода этой же F-функции, которое будет равно 0010.

Так как перед выходом из F-функции данные подвергаются перестановке, согласно таблице 2, то, сделав шаг назад, находим, что на выходе S-блоков появляется значение 0010, то есть 00 будет являться выходом S_1 блока, а 10 — выходом S_2 блока.

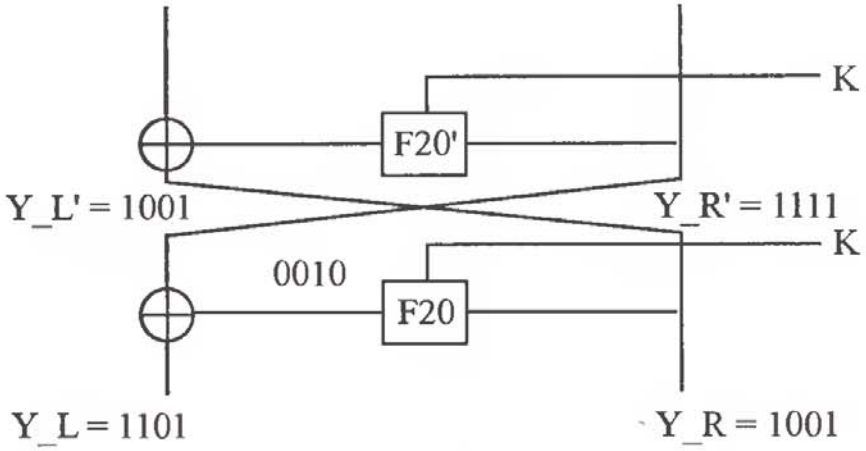


Рис. 8. ***Анализ последних раундов четвертой слайдовой пары***

Входное сообщение F-функции подвергается перестановке с расширением, согласно таблице 1. А значит, вход F-функции 1001 преобразуется к значению 01100110, которое и будет сложено с ключом $K = (K_1, K_2)$, то есть получается, что вход S_1 блока $0110 \oplus K_1$ даст на выходе значение 00, а вход S_2 блока $0110 \oplus K_2$ — значение 10.

Воспользовавшись таблицей 7, можно определить, что значение 00 появляется на выходе S_1 блока в случае, если на его вход поступило одно из следующих значений 0000, 0101, 1000 или 1111. Таким образом, добавив к каждому из возможных значений входа значение 0110, получим возможные значения K_1. Это будут значения 0110, 0011, 1110 или 1001.

Аналогичным образом определяем, что значение 10 появляется на выходе S_2 блока в случае, если на его вход поступило одно из следующих значений: 0011, 0110, 1011 или 1111. А значит, добавив к каждому из возможных значений входа значение 0110, получим возможные значения K_2. Это будут значения 0101, 0000, 1101 или 1001.

Так как во всех раундах шифрования использовался один и тот же ключ, то значение K_1 первого раунда должно совпадать со значением K_1 последнего раунда, точно так же, как значение K_2 первого раунда должно совпадать со значением K_2 последнего раунда. Сопоставив, всевозможные значения K_1 и K_2, можно увидеть, что есть только два значения $K_1 = 0110$ и $K_1 = 1001$, использование которых воз-

можно как в первом, так в последнем раундах, и ни одного значения K_2, использование которого также возможно было бы как в первом, так и в последнем раундах. Таким образом, можно сделать вывод, что данная пара не является слайдовой и не дает нам ни одного возможного значения секретного ключа.

Анализ пятой пары текстов даст нам точно такой же результат, как и анализ второй пары текстов (можете убедиться в этом самостоятельно), то есть для пятой пары текстов возможными значениями секретного ключа будут K = 10001110 и K = 01011001.

Таким образом, проанализировав все пять пар текстов и сопоставив полученные возможные значения секретного ключа, можно сделать вывод, что ключ K = 10001001, встречающийся чаще всего, является искомым секретным ключом.

Методические указания по выполнению лабораторной работы

Номер варианта индивидуального задания назначается преподавателем. При выполнении лабораторной работы используется программа Crypto5.exe. С помощью данной программы для каждого варианта производится зашифрование открытых текстов на секретном ключе варианта. После этого пользователь имеет возможность сохранить либо напечатать полученные пары *открытый—закрытый текст*.

Далее, введя, согласно заданному варианту, маску для отбора слайдовых пар и нажав кнопку «Анализ», пользователь получает возможные слайдовые пары, которые ему будет необходимо проанализировать. Проведя анализ, необходимо выбрать из всех возможных значений ключей тот, который встречается чаще всего.

Правильность проведенного анализа может быть проверена введением всех найденных битов в Форму проверки. Если все биты найдены верно, то программа выдаст соответствующее сообщение, которое необходимо предъявить преподавателю для допуска к защите работы.

Рекомендуемый порядок работы

1. Вызвать программу Crypto5.exe.
2. Ввести ФИО студента, номер группы и номер варианта.
3. Произвести зашифрование данных.
4. На закладке «Анализ» ввести маску, согласно индивидуальному заданию варианта, и, нажав кнопку «Анализ», получить возможные слайдовые пары.
5. Согласно полученным данным, определить значение секретного ключа.
6. Проверить правильность найденного ключа (если получилось несколько возможных вариантов ключа, то осуществлять проверку до тех пор, пока не будет найден правильный ключ).

Содержание отчета

1. Титульный лист с указанием варианта задания.
2. Цель работы.
3. Полученные возможные слайдовые пары текстов
4. Анализ полученных пар текстов
5. Найденные биты ключа.
6. Выводы по проделанной работе.

Варианты индивидуальных заданий

Вариант № 1

Таблица перестановки с расширением:

4	1	2	3	2	3	4	1

Таблица перестановки:

2	4	3	1

S_1 блок:

	00	**01**	**10**	**11**
00	1	0	3	2
01	3	2	1	0
10	0	2	1	3
11	3	1	3	1

S_2 блок:

	00	**01**	**10**	**11**
00	1	1	2	3
01	2	0	1	3
10	3	0	1	0
11	2	1	0	3

Маска:

0	1	0	0

Вариант № 2

Таблица перестановки с расширением:

3	2	1	4	4	1	2	3

Таблица перестановки:

1	2	4	3

S_1 блок:

	00	**01**	**10**	**11**
00	3	1	1	0
01	2	3	0	1
10	1	2	3	1
11	0	1	2	3

S_2 блок:

	00	**01**	**10**	**11**
00	2	0	1	3
01	0	2	3	1
10	1	3	2	0
11	3	1	0	2

Маска:

1	0	0	1

Вариант № 3

Таблица перестановки с расширением:

3	4	2	1	1	4	2	3

Таблица перестановки:

1	3	4	2

S_1 блок:

	00	**01**	**10**	**11**
00	1	0	3	2
01	3	1	2	0
10	0	2	1	3
11	2	3	0	1

S_2 блок:

	00	**01**	**10**	**11**
00	2	3	0	1
01	3	2	1	3
10	0	1	2	0
11	1	3	0	2

Маска:

1	1	0	0

Вариант № 4

Таблица перестановки с расширением:

3	2	4	1	4	1	3	2

Таблица перестановки:

1	4	3	2

S_1 блок:

	00	01	10	11
00	1	1	2	0
01	3	2	3	0
10	0	0	1	3
11	3	2	1	2

S_2 блок:

	00	01	10	11
00	2	2	0	3
01	1	1	0	0
10	3	3	2	2
11	0	3	1	1

Маска:

1	1	1	1

Вариант № 5

Таблица перестановки с расширением:

2	1	4	3	3	4	1	2

Таблица перестановки:

2	1	4	3

S_1 блок:

	00	01	10	11
00	3	0	2	0
01	1	2	1	1
10	3	2	1	3
11	3	2	0	0

S_2 блок:

	00	01	10	11
00	2	0	2	3
01	3	1	2	1
10	1	0	0	0
11	3	2	3	1

Маска:

0	1	0	1

Вариант № 6

Таблица перестановки с расширением:

3	1	4	2	3	1	4	2

Таблица перестановки:

2	3	4	1

S_1 блок:

	00	**01**	**10**	**11**
00	2	3	1	3
01	1	0	0	2
10	3	2	3	1
11	1	0	2	0

S_2 блок:

	00	**01**	**10**	**11**
00	0	1	2	0
01	1	2	1	2
10	3	1	2	3
11	0	3	3	0

Маска:

1	0	1	0

Вариант № 7

Таблица перестановки с расширением:

3	3	1	1	2	2	4	4

Таблица перестановки:

3	2	4	1

S_1 блок:

	00	**01**	**10**	**11**
00	3	0	1	2
01	1	0	3	0
10	2	2	3	1
11	3	1	0	2

S_2 блок:

	00	**01**	**10**	**11**
00	1	3	0	1
01	0	2	3	2
10	2	1	2	0
11	3	0	1	3

Маска:

1	0	0	0

Вариант № 8

Таблица перестановки с расширением:

4	1	3	4	2	1	3	2

Таблица перестановки:

4	2	1	3

S_1 блок:

	00	**01**	**10**	**11**
00	1	2	1	0
01	0	3	2	3
10	1	2	0	2
11	3	0	3	1

S_2 блок:

	00	**01**	**10**	**11**
00	1	2	3	0
01	0	1	3	3
10	2	3	0	1
11	1	0	2	2

Маска:

1	1	0	0

Вариант № 9

Таблица перестановки с расширением:

2	4	1	3	4	1	3	2

Таблица перестановки:

3	4	1	2

S_1 блок:

	00	**01**	**10**	**11**
00	0	2	3	1
01	2	0	1	2
10	3	0	1	3
11	1	3	2	0

S_2 блок:

	00	**01**	**10**	**11**
00	2	3	0	1
01	3	1	2	0
10	2	0	3	1
11	1	3	2	0

Маска:

1	0	1	0

Вариант № 10

Таблица перестановки с расширением:

3	2	4	1	4	3	1	2

Таблица перестановки:

3	1	4	2

S_1 блок:

	00	**01**	**10**	**11**
00	3	2	3	0
01	0	3	2	2
10	1	2	1	3
11	0	1	0	1

S_2 блок:

	00	**01**	**10**	**11**
00	0	3	1	3
01	1	2	3	1
10	3	1	2	0
11	0	2	0	2

Маска:

0	1	0	0

Вариант № 11

Таблица перестановки с расширением:

2	4	1	3	4	2	3	1

Таблица перестановки:

4	1	3	2

S_1 блок:

	00	**01**	**10**	**11**
00	3	0	1	1
01	0	2	0	3
10	1	1	3	2
11	0	3	2	2

S_2 блок:

	00	**01**	**10**	**11**
00	1	2	1	3
01	3	0	2	0
10	0	3	1	2
11	2	1	3	0

Маска:

0	1	1	0

Вариант № 12

Таблица перестановки с расширением:

2	3	1	2	4	1	3	4

Таблица перестановки:

4	3	1	2

S_1 блок:

	00	**01**	**10**	**11**
00	0	1	0	1
01	1	1	2	2
10	0	2	2	3
11	3	0	3	3

S_2 блок:

	00	01	10	11
00	1	2	0	0
01	0	3	2	1
10	2	3	0	2
11	3	1	1	3

Маска:

1	0	1	1

Вариант № 13

Таблица перестановки с расширением:

1	4	3	4	2	1	3	2

Таблица перестановки:

3	4	2	1

S_1 блок:

	00	01	10	11
00	0	1	2	1
01	2	3	0	3
10	2	1	2	1
11	3	0	3	0

S_2 блок:

	00	01	10	11
00	0	2	0	1
01	0	1	3	2
10	3	2	3	1
11	0	2	3	1

Маска:

1	0	1	1

Вариант № 14

Таблица перестановки с расширением:

4	3	2	1	4	3	2	1

Таблица перестановки:

2	3	1	4

S_1 блок:

	00	01	10	11
00	2	3	1	3
01	3	2	0	1
10	0	1	3	2
11	1	0	2	0

S_2 блок:

	00	01	10	11
00	1	1	2	3
01	0	2	1	3
10	2	0	3	1
11	0	2	3	0

Маска:

0	0	0	0

Вариант № 15

Таблица перестановки с расширением:

2	3	1	4	1	3	4	2

Таблица перестановки:

2	1	3	4

S_1 блок:

	00	01	10	11
00	3	2	3	2
01	1	3	2	1
10	3	2	3	2
11	1	0	0	1

S_2 блок:

	00	01	10	11
00	2	1	0	1
01	0	2	1	0
10	2	1	3	1
11	0	3	2	3

Маска:

1	1	1	0

Контрольные вопросы

1. На чем основан принцип применения слайдовой атаки?
2. Что такое слайдовая пара?
3. В каком случае функция F является слабой функцией?
4. В чем особенность применения метода слайдовой атаки к алгоритмам шифрования, построенным по схеме Фейстеля?
5. Опишите метод слайдовой атаки с использованием дополнений.
6. Опишите метод слайдовой атаки с петлей.
7. Как можно применить метод слайдовой атаки к алгоритмам шифрования с четырехраундовым самоподобием?

Глава 10. ЗАДАЧИ ПО ЛИНЕЙНО-ДИФФЕРЕНЦИАЛЬНОМУ КРИПТОАНАЛИЗУ

Нижеприведенные задачи предназначены для приобретения навыков по использованию основных принципов линейно-дифференциального криптоанализа. Мы будем рассматривать пятираундовый алгоритм шифрования, построенный по образу алгоритма S_DES. Отличаться он будет только тем, что для каждого варианта задания будут иметься свои таблицы замен и перестановок.

Рассмотрим, к примеру, такой алгоритм шифрования, у которого замены и перестановки выполняются согласно таблицам 10.1–10.4.

Таблица 10.1

Перестановка с расширением E/P

3	4	2	1	1	4	2	3

Таблица 10.2

Блок замены S_1

	00	**01**	**10**	**11**
00	1	0	3	2
01	3	1	2	0
10	0	2	1	3
11	2	3	0	1

Таблица 10.3

Блок замены S_2

	00	**01**	**10**	**11**
00	2	3	0	1
01	3	2	1	3
10	0	1	2	0
11	1	3	0	2

Таблица 10.4

Перестановка P

1	3	4	2

Требуется определить входную разность функции F третьего раунда шифрования пятираундового алгоритма шифрования.

Так как нам известен алгоритм шифрования, то мы можем проследить, какие выходные разности получаются при том или ином значении входной разности. Пусть нам известно, что на вход алгоритма шифрования поступает разность $\Delta_{вх} = 11110000$, при этом на выходе алгоритма шифрования может появиться одна из разностей $\Delta_{вых1} = 00000110$, $\Delta_{вых2} = 01100110$, $\Delta_{вых3} = 11110110$. Требуется найти возможное значение разности, поступающей на вход третьего раунда шифрования. Зная эту разность, станет возможным нахождение битов секретного ключа с помощью технологий, описанных в главе 7.

Анализ начнем с того, что рассмотрим блоки замены S_1 и S_2 и построим таблицы 10.5–10.8 с результатами анализа. Так как на вход каждого из блоков замены поступает по четыре бита, а на выход всего два, то диапазон изменения входных значений будет лежать в пределах от 0 до 15, а выходных — от 0 до 3.

Так как на вход алгоритма шифрования поступает разность, равная $\Delta_{вх1} = 11110000$, то на вход функции F первого раунда поступит разность, равная нулю. Мы знаем, что если на вход поступает нулевая разность, то на выходе будет тоже нулевая разность. Поэтому можем определить, что на вход второго раунда шифрования поступит входная разность $\Delta = 00001111$. А значит, на вход функции F второго раунда шифрования поступит входная разность, равная 1111. Воспользовавшись таблицей перестановки с расширением, можем определить, что на вход S_1 блока замены поступит разность, равная 1111, и такая же разность поступит на вход S_2 блока замены. С помощью таблицы 10.7 мы можем определить, что с вероятностью $p = 1$ на выходе первого блока замены S_1 будет находиться выходная разность, равная 00. С помощью таблицы 10.8 можем определить, что на выходе второго блока замены S_2 будет находиться с вероятностью $p = 1/2$ разность, равная 00, и с такой же вероятностью разность, равная 11. Таким образом, мы не можем однозначно определить выходную разность второго блока замены, однако мы с уверенностью можем сказать, что сумма по модулю два битов выходной разности второго блока замены будет всегда равна нулю: $1 \oplus 1 = 0 \oplus 0 = 0$. После перестановки P, произведенной по таблице 10.4, выходные биты разности второго блока замены окажутся на втором и третьем местах (см. рис. 10.1).

Таблица 10.5

Результаты статистического линейного анализа блока S_1

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

Таблица 10.6

Результаты статистического линейного анализа блока S_2

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

Таблица 10.7

Результаты зависимости значений ΔC от ΔA для блока S_1

ΔC / ΔA	0	1	2	3
0	16	0	0	0
1	0	8	8	0
2	0	8	8	0
3	8	0	0	8
4	0	8	8	0
5	8	0	0	8
6	0	0	0	16
7	0	8	8	0
8	0	8	8	0
9	0	0	0	16
10	8	0	0	8
11	0	8	8	0
12	8	0	0	8
13	0	8	8	0
14	0	8	8	0
15	16	0	0	0

Таблица 10.8

Результаты зависимости значений ΔC от ΔA для блока S_2

ΔC / ΔA	0	1	2	3
0	16	0	0	0
1	0	8	8	0
2	0	8	8	0
3	12	0	0	4
4	0	8	8	0
5	4	0	0	12
6	4	0	0	12
7	0	8	8	0
8	0	8	8	0
9	4	0	0	12
10	4	0	0	12
11	0	8	8	0
12	12	0	0	4
13	0	8	8	0
14	0	8	8	0
15	8	0	0	8

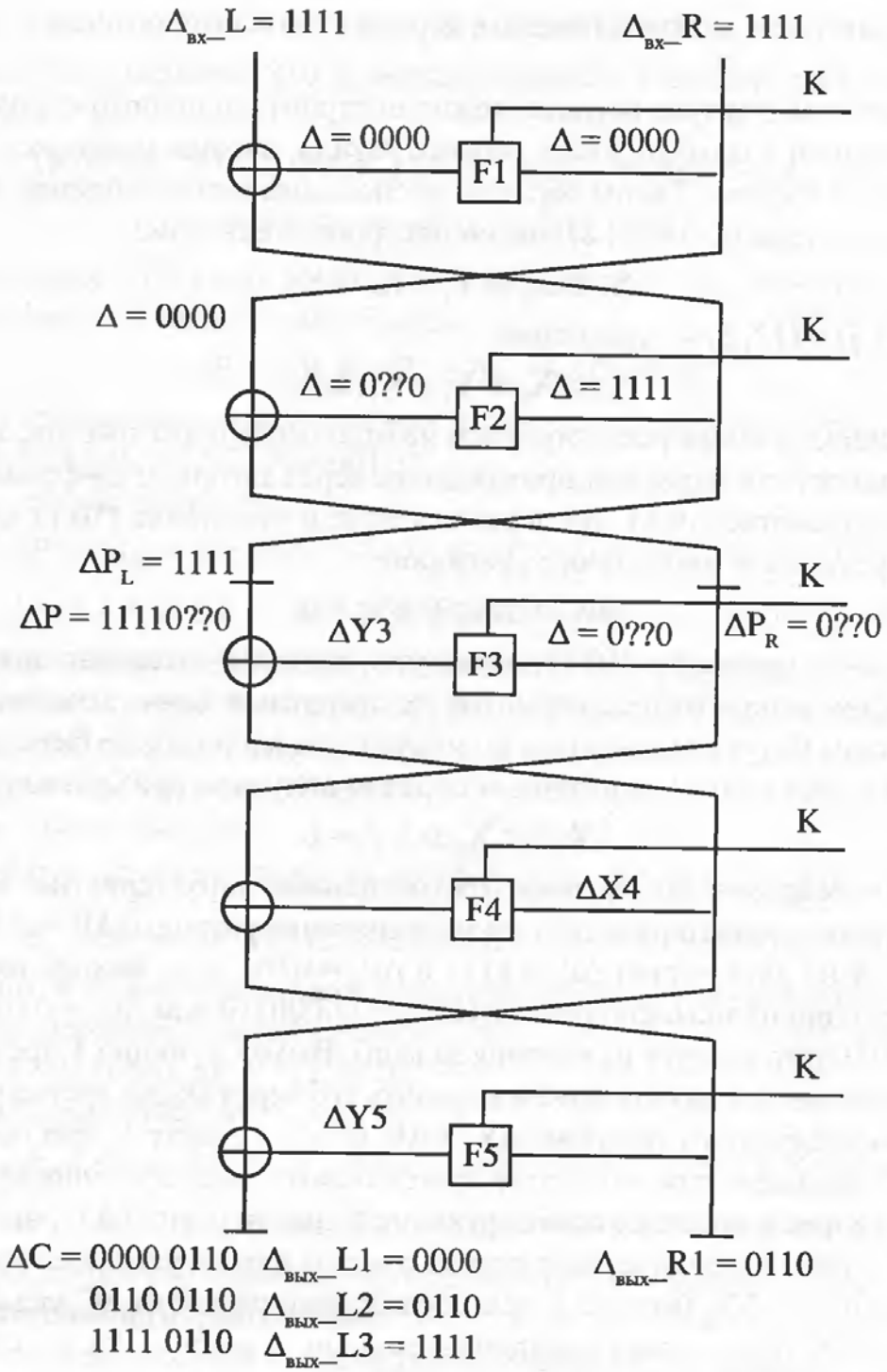


Рис. 10.1. ***Схема анализируемого алгоритма шифрования***

Теперь обратимся к таблице 10.5. С ее помощью мы можем составить линейные аналоги. Если принять, что на вход функции F поступают биты входных данных с пятого по восьмой, то есть X_5, X_6, X_7 и X_8, то, согласно таблице перестановки с расширением (таблица 10.1), на вход первого блока замены поступают X_7, X_8, X_6 и X_5. Выходные биты первого блока замены после перестановки P (по табли-

це 10.4) оказываются, соответственно, в первой и четвертой позициях, то есть это значения Y_1 и Y_4.

Мы знаем, что для трех раундов можно построить линейный статистический аналог, зависящий только от входа первого раунда, выхода последнего раунда и битов секретного ключа. Таким образом, воспользовавшись таблицей 10.5, определяем, что для пары (i, j) = (3, 2) можно построить уравнение:

$$X_7 \oplus X_8 \oplus Y_1 = K_3 \oplus K_4, \quad (10.1)$$

а для пары (i, j) = (12, 2) — уравнение:

$$X_6 \oplus X_5 \oplus Y_1 = K_1 \oplus K_2. \quad (10.2)$$

Мы в нашем примере рассматриваем не отдельные пары текстов, а разности, которые образуют эти пары при прохождении через алгоритм шифрования. Если объединить уравнение (10.1) для первого текста и уравнение (10.1) для второго текста, то в результате мы получим уравнение:

$$\Delta X_7 \oplus \Delta X_8 \oplus \Delta Y_1 = 0. \quad (10.3)$$

Правая часть уравнения (10.3) равна нулю, так как по условию для шифрования пар текстов используется один и тот же секретный ключ, а значит, в правой части уравнения будут складываться по модулю два одни и те же биты ключа. Что в результате и даст ноль. Аналогичным образом получаем для уравнения (10.2):

$$\Delta X_6 \oplus \Delta X_5 \oplus \Delta Y_1 = 0. \quad (10.4)$$

Теперь перейдем к построению трехраундовых характеристик. Будем считать, что на вход третьего раунда поступает значение разности $\Delta P = 11110??0$, которое состоит из двух частей $\Delta P_L = 1111$ и $\Delta P_R = 0??0$, а на выходе пятого раунда находится одно из значений разности $\Delta C_1 = 00000110$ или $\Delta C_2 = 01100110$, или $\Delta C_3 = 11110110$ (это следует из условия задачи). Выход функции F третьего раунда ΔY_3 нам неизвестен, но мы можем выразить его через входы третьего и четвертого раундов следующим образом: $\Delta Y_3 = \Delta P_L \oplus \Delta X_4$ (индекс L при обозначении разности ΔP указывает нам, что используется только левая половина входной разности). Также нам неизвестен выход функции F пятого раунда ΔY_5, но мы можем выразить его через вход четвертого раунда и выход пятого раунда следующим образом: $\Delta Y_5 = \Delta C_L \oplus \Delta X_4$ (индекс L при обозначении разности ΔC указывает нам, что используется только левая половина выходной разности). Зная все это, составим с помощью уравнения (10.3) трехраундовую характеристику, при этом будем учитывать, что входная разность обозначена как ΔP, а выходная разность — как ΔC:

$$\Delta P_7 \oplus \Delta P_8 \oplus (\Delta P_1 \oplus \Delta X_{41}) \oplus \Delta C_7 \oplus \Delta C_8 \oplus (\Delta C_1 \oplus \Delta X_{41}) = 0;$$

$$\Delta P_7 \oplus \Delta P_8 \oplus \Delta P_1 \oplus \Delta C_7 \oplus \Delta C_8 \oplus \Delta C_1 = 0. \quad (10.5)$$

В уравнении (10.5) есть всего одно неизвестное — это значение ΔP_7. Мы можем определить его следующим образом:

$$\Delta P_7 = \Delta P_8 \oplus \Delta P_1 \oplus \Delta C_7 \oplus \Delta C_8 \oplus \Delta C_1. \quad (10.6)$$

С помощью уравнения (10.4) можем составить еще одну трехраундовую характеристику:

$$\Delta P_6 \oplus \Delta P_5 \oplus (\Delta P_1 \oplus \Delta X_{41}) \oplus \Delta C_6 \oplus \Delta C_5 \oplus (\Delta C_1 \oplus \Delta X_{41}) = 0;$$

$$\Delta P_6 \oplus \Delta P_5 \oplus \Delta P_1 \oplus \Delta C_6 \oplus \Delta C_5 \oplus \Delta C_1 = 0. \quad (10.7)$$

В уравнении (10.7) есть всего одно неизвестное — это значение ΔP_6. Мы можем определить его следующим образом:

$$\Delta P_6 = \Delta P_5 \oplus \Delta P_1 \oplus \Delta C_6 \oplus \Delta C_5 \oplus \Delta C_1. \quad (10.8)$$

Теперь, опираясь на известные нам данные, можем рассмотреть три случая:

1) $\Delta P = 11110??0$, $\Delta C = 00000110$.

В этом случае получаем:

$$\Delta P_7 = \Delta P_8 \oplus \Delta P_1 \oplus \Delta C_7 \oplus \Delta C_8 \oplus \Delta C_1 = 0 \oplus 1 \oplus 1 \oplus 0 \oplus 0 = 0;$$

$$\Delta P_6 = \Delta P_5 \oplus \Delta P_1 \oplus \Delta C_6 \oplus \Delta C_5 \oplus \Delta C_1 = 0 \oplus 1 \oplus 1 \oplus 0 \oplus 0 = 0.$$

Заметьте, что сумма $\Delta P_7 \oplus \Delta P_6 = 0$, как и было определено нами раньше. Таким образом, мы нашли, что на вход функции F третьего раунда шифрования поступит значение разности, равное 0000.

2) $\Delta P = 11110??0$, $\Delta C = 01100110$.

В этом случае получаем:

$$\Delta P_7 = \Delta P_8 \oplus \Delta P_1 \oplus \Delta C_7 \oplus \Delta C_8 \oplus \Delta C_1 = 0 \oplus 1 \oplus 1 \oplus 0 \oplus 0 = 0;$$

$$\Delta P_6 = \Delta P_5 \oplus \Delta P_1 \oplus \Delta C_6 \oplus \Delta C_5 \oplus \Delta C_1 = 0 \oplus 1 \oplus 1 \oplus 0 \oplus 0 = 0.$$

А значит, в этом случае на вход функции F третьего раунда шифрования также поступит значение разности, равное 0000.

3) $\Delta P = 11110??0$, $\Delta C = 11110110$.

В этом случае получаем:

$$\Delta P_7 = \Delta P_8 \oplus \Delta P_1 \oplus \Delta C_7 \oplus \Delta C_8 \oplus \Delta C_1 = 0 \oplus 1 \oplus 1 \oplus 0 \oplus 1 = 1;$$

$$\Delta P_6 = \Delta P_5 \oplus \Delta P_1 \oplus \Delta C_6 \oplus \Delta C_5 \oplus \Delta C_1 = 0 \oplus 1 \oplus 1 \oplus 0 \oplus 1 = 1.$$

А значит, в этом случае на вход функции F третьего раунда шифрования поступит значение разности, равное 0110.

Итак, поставленная перед нами задача решена. Ниже приводятся задачи для самостоятельного решения. Далее, как уже говорилось раньше, можно найти биты секретного ключа с помощью технологий, описанных в главе 7.

Варианты для самостоятельного решения

Определить входную разность функции F третьего раунда шифрования пятираундового алгоритма шифрования для представленных вариантов.

Вариант № 1

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

3	2	1	4	4	1	2	3

Блок замены S_1

	00	**01**	**10**	**11**
00	3	1	1	0
01	2	3	0	1
10	1	2	3	1
11	0	1	2	3

Блок замены S_2

	00	**01**	**10**	**11**
00	2	0	1	3
01	0	2	3	1
10	1	3	2	0
11	3	1	2	0

Перестановка P

1	2	4	3

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 01100000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 00011000$, $\Delta_{вых2} = 01100000$, $\Delta_{вых3} = 11011000$.

Вариант № 2

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

3	2	4	1	4	1	3	2

Блок замены S_1

	00	**01**	**10**	**11**
00	1	1	2	0
01	3	2	3	0
10	0	0	1	3
11	3	2	1	2

Блок замены S_2

	00	**01**	**10**	**11**
00	2	2	0	3
01	1	1	0	0
10	3	3	2	2
11	0	3	1	1

Перестановка P

1	4	3	2

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 00110000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 01010011$, $\Delta_{вых2} = 01001110$, $\Delta_{вых3} = 01111011$.

Вариант № 3

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

2	1	4	3	3	4	1	2

Блок замены S_1

	00	**01**	**10**	**11**
00	3	0	2	0
01	1	2	1	1
10	3	2	1	3
11	3	2	0	0

Блок замены S_2

	00	**01**	**10**	**11**
00	2	0	2	3
01	3	1	2	1
10	1	0	0	0
11	3	2	3	1

Перестановка P

2	1	4	3

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 10110000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 00100111$, $\Delta_{вых2} = 10011100$, $\Delta_{вых3} = 10101100$.

Вариант № 4

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

3	1	4	2	3	1	4	2

Блок замены S_1

	00	**01**	**10**	**11**
00	2	3	1	3
01	1	0	0	2
10	3	2	3	1
11	1	0	2	0

Блок замены S_2

	00	**01**	**10**	**11**
00	0	1	2	0
01	1	2	1	2
10	3	1	2	3
11	0	3	3	0

Перестановка P

2	3	4	1

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 00110000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 01001101, \Delta_{вых2} = 11101001, \Delta_{вых3} = 10100011$.

Вариант № 5

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

3	3	1	1	2	2	4	4

Блок замены S_1

	00	**01**	**10**	**11**
00	3	0	1	2
01	1	0	3	0
10	2	2	3	1
11	3	1	0	2

Блок замены S_2

	00	**01**	**10**	**11**
00	1	3	0	1
01	0	2	3	2
10	2	1	2	0
11	3	0	1	3

Перестановка P

3	2	4	1

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 01010000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 00010000$, $\Delta_{вых2} = 01001000$.

Вариант № 6

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

2	4	1	3	4	1	3	2

Блок замены S_1

	00	**01**	**10**	**11**
00	0	2	3	1
01	2	0	1	2
10	3	0	1	3
11	1	3	2	0

Блок замены S_2

	00	**01**	**10**	**11**
00	2	3	0	1
01	3	1	2	0
10	2	0	3	1
11	1	3	2	0

Перестановка P

3	4	1	2

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 01010000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 01100111$, $\Delta_{вых2} = 10111011$, $\Delta_{вых3} = 11001010$.

Вариант № 7

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

2	4	1	3	4	2	3	1

Блок замены S_1

	00	**01**	**10**	**11**
00	3	0	1	1
01	0	2	0	3
10	1	1	3	2
11	0	3	2	2

Блок замены S_2

	00	**01**	**10**	**11**
00	1	2	1	3
01	3	0	2	0
10	0	3	1	2
11	2	1	3	0

Перестановка P

4	1	3	2

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 01010000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 01010000$, $\Delta_{вых2} = 00100100$, $\Delta_{вых3} = 00101011$.

Вариант № 8

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

1	4	3	4	2	1	3	2

Блок замены S_1

	00	**01**	**10**	**11**
00	0	1	2	1
01	2	3	0	3
10	2	1	2	1
11	3	0	3	0

Блок замены S_2

	00	**01**	**10**	**11**
00	0	2	0	1
01	0	1	3	2
10	3	2	3	1
11	0	2	3	1

Перестановка P

3	4	2	1

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 00010000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 11100111$, $\Delta_{вых2} = 01000101$, $\Delta_{вых3} = 10011000$.

Вариант № 9

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

2	3	1	4	1	3	4	2

Блок замены S_1

	00	**01**	**10**	**11**
00	3	2	3	2
01	1	3	2	1
10	3	2	3	2
11	1	0	0	1

Блок замены S_2

	00	01	10	11
00	2	1	0	1
01	0	2	1	0
10	2	1	3	1
11	0	3	2	3

Перестановка P

2	1	3	4

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 10100000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 00011010$, $\Delta_{вых2} = 00000110$, $\Delta_{вых3} = 11101001$.

Вариант № 10

Шифрование выполняется с помощью следующих таблиц.

Перестановка с расширением E/P

4	1	2	3	2	3	4	1

Блок замены S_1

	00	01	10	11
00	1	0	3	2
01	3	2	1	0
10	0	2	1	3
11	3	1	3	1

Блок замены S_2

	00	01	10	11
00	1	1	2	3
01	2	0	1	3
10	3	0	1	0
11	2	1	0	3

Перестановка P

2	4	3	1

На вход алгоритма шифрования поступает значение разности $\Delta_{вх} = 11100000$, а на выходе алгоритма шифрования может появиться одна из трех разностей: $\Delta_{вых1} = 10000100$, $\Delta_{вых2} = 00100110$, $\Delta_{вых3} = 11110001$.

ЗАДАЧИ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Если одна машина опробует один ключ за 10^{-6} с, то сколько потребуется машин, чтобы найти ключ алгоритма шифрования DES методом полного перебора за 24 часа на R-машинной системе?
2. Покажите, что дешифрование DES является операцией, представляющей собой обращение операции шифрования DES.
3. Заполните таблицу недостающими данными для алгоритма шифрования S-DES:

Исходный ключ шифрования	Ключ после начальной перестановки	Ключ после первого сдвига	Первый подключ	Ключ после второго сдвига	Второй подключ
1000100010					
0111001111					
1011110000					
1111101000					
0010110011					

4. Заполните таблицу недостающими данными для алгоритма шифрования S-DES:

Ключ шифрования K	Исходное сообщение X	Зашифрованное сообщение Y
1000100010	10010010	
	10001011	
0111001111	01101011	
	01111110	
1011110000	11110000	
	01101001	
0010110011	10001101	
	10100101	

5. Заполните таблицу соответствующими выходами S-блоков для алгоритма шифрования DES:

Вход	7	15	37	48	59
№ S-блока					
S_3					
S_6					
S_8					

6. Заполните таблицу S-блока алгоритма шифрования Rijndael (процедура SubBytes()), используя только второе преобразование над GF(2):

X	y															
	0	1	2	3	4	5	6	7	8	9	a	·b	c	d	e	F
0	63	7c	5d	42	1f	00	21	3e	9b	84	a5	ba	e7	f8	d9	c6
1	92	8d	?	?	?	?	?	?	?	?	?	?	?	?	?	37
2	80	?	be	?	?	?	?	?	?	?	?	?	?	?	?	25
3	71	?	?	50	?	?	?	?	?	?	?	?	?	?	?	d4
4	a4	?	?	?	d8	?	?	?	?	?	?	?	?	?	?	01
5	55	?	?	?	?	36	?	?	?	?	?	?	?	?	?	f0
6	47	?	?	?	?	?	05	?	?	?	?	?	?	?	?	e2
7	b6	?	?	?	?	?	?	eb	?	?	?	?	?	?	?	13
8	fc	?	?	?	?	?	?	?	14	?	?	?	?	?	?	49
9	1d	?	?	?	?	?	?	?	?	fa	?	?	?	?	?	b8
A	0f	?	?	?	?	?	?	?	?	?	c9	?	?	?	?	aa
B	fe	?	?	?	?	?	?	?	?	?	?	27	?	?	?	5b
C	2b	?	?	?	?	?	?	?	?	?	?	?	af	?	?	8e
d	da	?	?	?	?	?	?	?	?	?	?	?	?	41	?	7f
e	c8	?	?	?	?	?	?	?	?	?	?	?	?	?	72	6d
f	39	26	04	18	45	5a	7b	64	c1	de	ff	e0	bd	a2	83	9c

7. Для алгоритма шифрования DES постройте линейные уравнения для следующих пар (i, j); определите вероятности выполнения этих уравнений и их эффективность:

(i, j)	Блок S_2			Блок S_4			Блок S_6		
	Уравнение	p	Δ	Уравнение	p	Δ	Уравнение	p	Δ
(7,15)									
(29,4)									
(47,12)									
(5,13)									

8. Постройте эффективные линейные уравнения для алгоритма шифрования DES, состоящего из n раундов:

n	Уравнение	p	Δ	Подсказка
6				1-й раунд не задействован, использованы уравнения 4, 3 и 1 из табл. 5.5
9				Первый и девятый раунды задействованы, использованы уравнения 1, 3 и 4 из табл. 5.5
10				1-й раунд не задействован, использованы уравнения 4, 3 и1 из табл. 5.5
11				То же, что и для n=10, но в первом раунде использовано уравнение 1 из табл. 5.5
13				Задействованы 1, 2, 12 и 13 раунды, использованы уравнения 1, 2, 3 и 4 из табл. 5.5
15				Задействованы 1 и 15 раунды, не задействованы раунды 2 и 14, использованы уравнения 5, 4, 3 и 1 из табл. 5.5

9. Определите недостающие на рис. 1 и рис. 2 данные для проведения дифференциального криптоанализа одного раунда алгоритма шифрования DES.

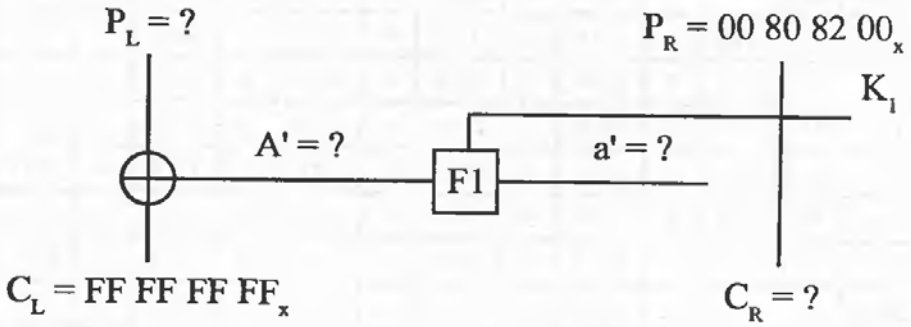


Рис. 1

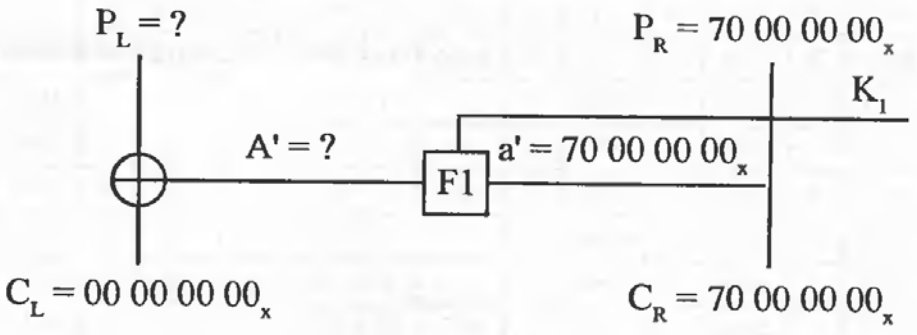


Рис. 2

10. Определите всроятность процесса, изображенного на рис. 3.

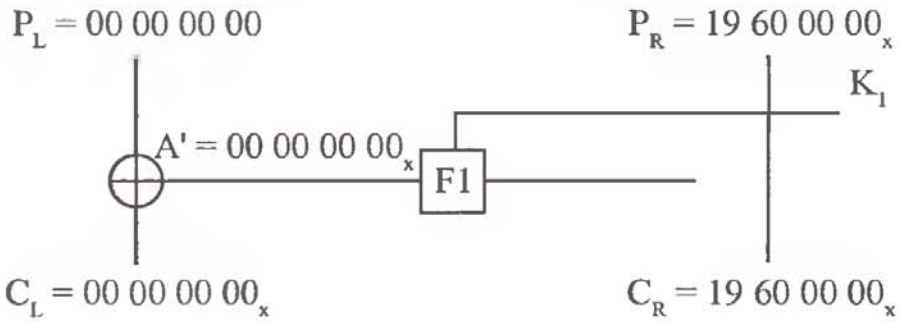


Рис. 3

11. Рассмотрите анализ алгоритма шифрования с четырехраундовым самоподобием, где будут использованы подключи K_0, K_1, K_2,K_3, с помощью слайдовой атаки.

Подсказка. Сопоставьте процессы зашифрования с «запаздыванием» на 2 раунда.

12. Покажите, что в алгоритме шифрования DES первые 24 бита любого подключа выбираются из одного 28-битного подмножества, а другие — из непересекающегося с ним 28-битового подмножества исходного ключа шифрования.

ПРИЛОЖЕНИЕ 1

Таблицы линейного и дифференциального криптоанализа алгоритма шифрования DES

Таблица 1

Соответствие входов и выходов алгоритма шифрования DES для S-блоков

Вход блока S		Выход блока S_1		Выход блока S_2		Выход блока S_3		Выход блока S_4		Выход блока S_5		Выход блока S_6		Выход блока S_7		Выход блока S_8	
Десятичное значение	Двоичное значение	Двоичное значение	Десятичное значение	Двоичное значение	Десятичное значение	Двоичное значение	Десятичное значение	Двоичное значение	Десятичное значение	Двоичное значение	Десятичное значение	Двоичное значение	Десятичное значение	Двоичное значение	Десятичное значение	Двоичное значение	Десятичное значение
0	000000	1110	14	1111	15	1010	10	0111	7	0010	2	1100	12	0100	4	1101	13
1	000001	0000	0	0011	3	1101	13	1101	13	1110	14	1010	10	1101	13	0001	1
2	000010	0011	3	0001	1	0000	0	1101	13	1100	12	0001	1	1011	11	0010	2
3	000011	1111	15	1101	13	0111	7	1000	8	1011	11	1111	15	0000	0	1111	15
4	000100	1101	13	1000	8	1001	9	1110	14	0100	4	1010	10	0010	2	1000	8
5	000101	0111	7	0100	4	0000	0	1011	11	0010	2	0100	4	1011	11	1101	13
6	000110	0001	1	1110	14	1110	14	0011	3	0001	1	1111	15	1110	14	0100	4
7	000111	0100	4	0111	7	1001	9	0101	5	1100	12	0010	2	0111	7	1000	8
8	001000	0010	2	0110	6	0110	6	0000	0	0111	7	1001	9	1111	15	0110	6
9	001001	1110	14	1111	15	0011	3	0110	6	0100	4	0111	7	0100	4	1010	10
10	001010	1111	15	1011	11	0011	3	0110	6	1010	10	0010	2	0000	0	1111	15
11	001011	0010	2	0010	2	0100	4	1111	15	0111	7	1100	12	1001	9	0011	3
12	001100	1011	11	0011	3	1111	15	1001	9	1011	11	0110	6	1000	8	1011	11
13	001101	1101	13	1000	8	0110	6	0000	0	1101	13	1001	9	0001	1	0111	7
14	001110	1000	8	0100	4	0101	5	1010	10	0110	6	1000	8	1101	13	0001	1
15	001111	0001	1	1110	14	1010	10	0011	3	0001	1	0101	5	1010	10	0100	4
16	010000	0011	3	1001	9	0001	1	0001	1	1000	8	0000	0	0011	3	1010	10
17	010001	1010	10	1100	12	0010	2	0100	4	0101	5	0110	6	1110	14	1100	12
18	010010	1010	10	0111	7	1101	13	0010	2	0101	5	1101	13	1100	12	1001	9
19	010011	0110	6	0000	0	1000	8	0111	7	0000	0	0001	1	0011	3	0101	5
20	010100	0110	6	0010	2	1100	12	1000	8	0011	3	0011	3	1001	9	0011	3
21	010101	1100	12	0001	1	0101	5	0010	2	1111	15	1101	13	0101	5	0110	6
22	010110	1100	12	1101	13	0111	7	0101	5	1111	15	0100	4	0111	7	1110	14
23	010111	1011	11	1010	10	1110	14	1100	12	1010	10	1110	14	1100	12	1011	11

24	011000	0101	5	1100	12	1011	11	1011	11	1101	13	1110	14	0101	5	0101	5
25	011001	1001	9	0110	6	1100	12	0001	1	0011	3	0000	0	0010	2	0000	0
26	011010	1001	9	0000	0	0100	4	1100	12	0000	0	0111	7	1010	10	0000	0
27	011011	0101	5	1001	9	1011	11	1010	10	1001	9	1011	11	1111	15	1110	14
28	011100	0000	0	0101	5	0010	2	0100	4	1110	14	0101	5	0110	6	1100	12
29	011101	0011	3	1011	11	1111	15	1110	14	1000	8	0011	3	1000	8	1001	9
30	011110	0111	7	1010	10	1000	8	1111	15	1001	9	1011	11	0001	1	0111	7
31	011111	1000	8	0101	5	0001	1	1001	9	0110	6	1000	8	0110	6	0010	2
32	100000	0100	4	0000	0	1101	13	1010	10	0100	4	1001	9	0001	1	0111	7
33	100001	1111	15	1101	13	0001	1	0011	3	1011	11	0100	4	0110	6	0010	2
34	100010	0001	1	1110	14	0110	6	0110	6	0010	2	1110	14	0100	4	1011	11
35	100011	1100	12	1000	8	1010	10	1111	15	1000	8	0011	3	1011	11	0001	1
36	100100	1110	14	0111	7	0100	4	1001	9	0001	1	1111	15	1011	11	0100	4
37	100101	1000	8	1010	10	1101	13	0000	0	1100	12	0010	2	1101	13	1110	14
38	100110	1000	8	1011	11	1001	9	0000	0	1011	11	0101	5	1101	13	0001	1
39	100111	0010	2	0001	1	0000	0	0110	6	0111	7	1100	12	1000	8	0111	7
40	101000	1101	13	1010	10	1000	8	1100	12	1010	10	0010	2	1100	12	1001	9
41	101001	0100	4	0011	3	0110	6	1010	10	0001	1	1001	9	0001	1	0100	4
42	101010	0110	6	0100	4	1111	15	1011	11	1101	13	1000	8	0011	3	1100	12
43	101011	1001	9	1111	15	1001	9	0001	1	1110	14	0101	5	0100	4	1010	10
44	101100	0010	2	1101	13	0011	3	0111	7	0111	7	1100	12	0111	7	1110	14
45	101101	0001	1	0100	4	1000	8	1101	13	0010	2	1111	15	1010	10	1000	8
46	101110	1011	11	0001	1	0000	0	1101	13	1000	8	0011	3	1110	14	0010	2
47	101111	0111	7	0010	2	0111	7	1000	8	1101	13	1010	10	0111	7	1101	13
48	110000	1111	15	0101	5	1011	11	1111	15	1111	15	0111	7	1010	10	0000	0
49	110001	0101	5	1011	11	0100	4	1001	9	0110	6	1011	11	1001	9	1111	15
50	110010	1100	12	1000	8	0001	1	0001	1	1001	9	0000	0	1111	15	0110	6
51	110011	1011	11	0110	6	1111	15	0100	4	1111	15	1110	14	0101	5	1100	12
52	110100	1001	9	1100	12	0010	2	0011	5	1100	12	0100	4	0110	6	1010	10
53	110101	0011	3	0111	7	1110	14	0101	5	0000	0	0001	1	0000	0	1001	9
54	110110	0111	7	0110	6	1100	12	1110	14	0101	5	1010	10	1000	8	1101	13
55	110111	1110	14	1100	12	0011	3	1011	11	1001	9	0111	7	1111	15	0000	0
56	111000	0011	3	1001	9	0101	5	0101	5	0110	6	0001	1	0000	0	1111	15
57	111001	1010	10	0000	0	1011	11	1100	12	1010	10	0110	6	1110	14	0011	3
58	111010	1010	10	0011	3	1010	10	0010	2	0011	3	1101	13	0101	5	0011	3
59	111011	0000	0	0101	5	0101	5	0111	7	0100	4	0000	0	0010	2	0101	5
60	111100	0101	5	0010	2	1110	14	1000	8	0000	0	1011	11	1001	9	0101	5
61	111101	0110	6	1110	14	0010	2	0010	2	0101	5	1000	8	0011	3	0110	6
62	111110	0000	0	1111	15	0111	7	0100	4	1110	14	0110	6	0010	2	1000	8
63	111111	1101	13	1001	9	1100	12	1110	14	0011	3	1101	13	1100	12	1011	11

Таблица 2

Анализ блока S_1 алгоритма шифрования DES для применения метода линейного криптоанализа

I	Значения j														
S(i,j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	30	30	28	30	32	28	38	34	32	32	38	36	30	26	36
3	30	30	28	30	32	28	38	34	40	32	30	36	38	26	28
4	34	30	28	30	32	28	26	30	36	40	34	32	30	26	44
5	30	30	32	30	28	28	30	34	28	28	34	36	22	30	28
6	32	32	36	32	36	32	32	32	28	36	36	32	32	28	24
7	28	32	40	32	32	32	36	36	28	24	28	36	32	32	32
8	36	30	38	26	26	32	28	28	28	34	30	34	30	32	32
9	32	38	36	30	26	36	28	32	28	30	38	34	26	32	28
10	30	32	34	32	38	40	34	30	32	30	36	30	32	30	36
11	34	24	30	28	22	36	34	26	40	34	36	30	28	30	32
12	30	32	38	32	34	32	34	34	32	38	28	34	28	38	32
13	38	32	38	36	30	28	30	34	32	38	36	30	40	26	28
14	32	30	30	34	34	32	32	36	36	38	30	34	34	28	36
15	32	30	38	30	30	36	28	28	28	30	30	30	30	32	32
16	34	34	32	30	32	36	26	32	28	24	28	38	28	28	14
17	34	30	28	34	28	28	42	28	34	34	28	30	28	32	26
18	36	32	32	28	36	32	36	26	34	34	38	34	38	38	22
19	36	28	28	32	32	24	20	30	30	26	38	34	38	34	34
20	36	32	36	24	28	36	32	34	38	30	34	38	34	30	34
21	32	36	28	28	36	36	28	42	34	34	34	26	34	38	30
22	38	34	32	34	28	32	34	36	34	34	32	30	32	32	34
23	34	38	24	38	36	32	30	20	30	30	32	26	32	32	30
24	34	40	34	32	38	36	34	36	30	36	38	32	30	28	34
25	30	36	26	32	26	32	34	36	26	40	38	32	34	32	26
26	32	26	34	30	30	36	36	30	30	32	32	28	36	34	34
27	36	38	34	22	34	24	36	30	26	36	32	36	32	30	34
28	28	34	34	34	26	32	28	30	30	36	32	32	36	34	34
29	36	30	30	34	26	28	32	34	34	28	32	20	32	26	26
30	34	32	30	36	30	32	30	32	38	28	30	32	30	32	34
31	34	28	34	28	30	36	34	36	26	36	30	28	34	32	34
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	34	30	32	34	32	32	38	30	32	28	38	36	42	42	32

35	34	30	32	34	32	32	38	38	32	36	22	28	26	34	32
36	34	26	24	34	36	36	34	34	32	32	34	32	38	22	32
37	30	34	36	34	32	28	30	30	32	36	34	28	38	26	32
38	36	36	28	40	24	36	32	32	24	32	28	32	36	32	32
39	32	28	24	24	36	28	28	36	24	28	28	36	36	28	32
40	36	30	30	30	30	28	32	36	36	34	38	30	26	44	36
41	32	30	26	34	30	24	40	32	28	30	30	38	30	28	32
42	34	32	30	32	34	32	30	34	24	26	28	34	32	34	28
43	32	32	34	28	34	36	38	30	32	22	36	34	28	26	32
44	38	28	34	40	34	36	38	30	28	30	44	30	24	30	32
45	30	28	34	28	30	32	34	30	28	30	36	26	36	34	28
46	28	34	38	38	26	24	36	28	32	34	38	38	34	36	32
47	28	34	30	34	22	44	32	28	32	34	30	42	38	32	36
48	30	30	32	30	36	32	34	32	34	38	36	38	32	32	30
49	30	34	36	34	32	32	26	28	30	30	28	30	32	28	34
50	28	28	28	32	32	32	36	30	30	26	30	26	38	34	34
51	28	32	32	36	28	32	28	26	34	34	30	34	30	30	30
52	40	24	40	36	36	32	32	30	30	34	26	38	38	30	30
53	36	28	32	24	28	32	28	30	34	30	34	34	30	30	34
54	38	34	24	34	28	40	34	36	26	34	32	38	32	32	34
55	34	22	32	38	36	40	30	36	38	30	22	34	32	32	30
56	22	36	34	28	30	36	30	36	34	32	38	28	38	28	30
57	34	32	26	28	34	32	30	28	38	28	30	36	34	40	30
58	34	32	26	28	34	32	30	28	38	28	30	36	34	40	30
59	36	34	30	30	42	36	32	18	30	36	32	32	28	30	34
60	32	42	30	26	30	32	40	26	38	32	24	28	36	30	34
61	40	30	34	26	30	36	36	30	26	32	32	32	32	30	34
62	34	32	30	24	34	36	34	32	30	36	30	28	34	36	30
63	18	20	26	32	34	32	30	28	26	44	30	32	30	36	30

Таблица 3

Анализ блока S_2 алгоритма шифрования DES для применения метода линейного криптоанализа

I	Значения j														
S(i,j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	32	32	36	32	32	28	32	32	32	28	32	32	32	32	36
3	32	36	24	32	40	32	28	32	32	24	28	32	40	28	40
4	30	34	36	34	32	36	38	32	38	30	32	34	32	32	42

5	34	34	32	34	28	36	26	32	26	38	36	26	36	32	30
6	30	30	28	30	28	32	30	28	34	34	32	34	32	36	42
7	34	34	28	30	32	36	30	28	38	38	32	26	28	32	34
8	32	34	34	30	34	36	32	30	26	28	32	32	32	42	26
9	32	30	30	30	34	40	36	30	34	32	28	32	40	22	30
10	28	34	34	34	34	28	24	34	34	36	32	32	36	26	34
11	36	34	42	34	34	36	32	34	34	36	32	32	28	34	34
12	30	36	30	32	34	32	38	30	32	34	32	34	32	26	28
13	26	32	30	32	38	36	22	38	28	38	32	34	28	30	36
14	26	32	34	32	30	24	38	30	36	34	32	34	36	30	32
15	30	32	30	32	34	32	42	38	40	34	36	34	32	30	36
16	32	28	36	32	32	32	32	32	32	36	36	36	20	28	20
17	32	32	32	32	24	36	36	32	32	40	32	28	36	40	32
18	32	32	36	32	40	32	36	32	32	28	40	28	20	32	44
19	32	24	36	32	40	40	36	32	32	28	32	36	28	40	28
20	30	26	28	30	36	28	30	28	34	34	28	38	28	32	34
21	42	30	28	30	32	32	30	36	38	38	28	30	32	36	34
22	30	26	32	34	32	36	34	40	30	34	32	38	36	32	30
23	42	34	36	34	36	28	30	32	34	34	28	30	32	32	34
24	28	34	30	34	34	36	36	30	30	36	36	32	36	30	34
25	36	26	38	34	34	36	28	30	30	36	28	40	36	30	34
26	40	30	34	30	34	32	32	34	38	32	32	32	32	34	30
27	24	42	38	30	34	36	28	34	30	28	28	24	32	30	26
28	34	40	30	32	30	32	34	34	32	26	36	42	28	34	32
29	30	32	34	32	26	32	30	34	36	34	32	42	32	34	36
30	30	32	38	32	34	36	30	34	36	30	32	34	32	34	32
31	34	28	38	32	30	24	30	18	32	34	32	34	36	30	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	32	32	36	32	24	36	32	32	32	28	16	32	24	24	36
35	32	28	32	32	32	32	36	32	32	32	36	32	32	28	32
36	30	30	32	30	28	36	22	32	38	26	28	38	36	40	34
37	26	30	36	30	32	36	34	32	34	34	24	30	32	40	30
38	38	26	32	18	32	32	30	28	26	30	36	30	28	28	34
39	34	38	32	34	28	28	30	28	38	26	36	38	32	32	26
40	28	34	30	34	34	32	24	34	34	32	40	40	28	26	30
41	28	30	26	34	34	36	28	26	34	28	28	32	28	30	26
42	40	34	30	38	26	32	32	30	34	32	32	32	32	26	30
43	32	26	30	38	42	32	32	22	42	32	32	24	32	34	30
44	26	32	38	32	30	28	30	34	24	34	36	30	32	34	40

45	30	28	30	32	26	32	22	34	36	30	36	22	28	30	32
46	30	28	34	40	34	36	30	26	28	26	36	30	36	30	36
47	26	36	30	24	30	36	34	26	32	42	32	38	32	30	32
48	32	28	36	28	28	36	36	32	32	28	28	32	32	24	32
49	32	32	32	36	28	32	32	32	32	32	24	32	24	28	36
50	32	32	36	36	36	36	32	32	32	36	32	32	32	28	32
51	32	32	28	28	28	28	32	32	32	28	32	32	32	36	32
52	22	30	24	38	36	24	34	36	34	38	24	30	28	36	30
53	26	34	32	30	32	36	34	28	30	26	32	30	32	32	30
54	30	30	36	26	32	40	30	32	38	38	36	30	36	28	34
55	34	30	24	18	36	32	34	40	34	30	32	30	32	28	30
56	32	34	34	26	30	28	32	34	30	32	28	28	28	34	34
57	40	42	26	34	38	36	32	26	22	40	28	36	28	34	34
58	28	30	38	30	30	24	36	30	30	36	32	28	32	30	30
59	28	34	34	22	38	28	32	22	30	32	28	36	32	34	34
60	30	28	30	36	30	32	30	30	32	34	32	34	24	30	32
61	34	36	42	28	42	24	26	38	36	34	28	34	36	30	28
62	34	20	30	36	34	28	34	38	20	30	28	34	36	30	32
63	30	24	30	28	30	32	36	30	32	34	36	34	32	34	32

Таблица 4

Анализ блока S_3 алгоритма шифрования DES для применения метода линейного криптоанализа

I	Значения j														
S(i,j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	34	30	28	32	30	26	36	32	38	30	24	36	30	30	28
3	30	30	32	36	30	30	28	28	30	34	32	28	34	30	32
4	28	32	32	34	30	26	34	32	36	36	28	26	22	38	30
5	32	32	28	34	34	34	38	32	32	36	32	34	26	38	34
6	30	34	32	34	28	32	34	32	26	30	32	30	36	40	30
7	30	34	32	30	40	36	38	28	30	34	28	22	36	32	30
8	32	38	26	32	32	34	30	30	34	32	40	38	34	36	28
9	32	34	30	32	24	30	26	30	34	28	28	30	34	40	32
10	30	40	30	32	34	24	26	34	32	38	36	30	32	34	28
11	34	28	38	36	34	32	30	30	32	30	32	30	28	30	36
12	32	30	30	34	42	36	28	34	30	32	28	36	24	38	26
13	28	34	30	34	30	32	44	34	34	36	36	36	36	34	34
14	30	28	30	34	28	30	28	30	36	34	36	36	30	24	30

15	30	24	34	30	32	30	28	26	24	34	36	36	30	36	34
16	32	34	30	34	34	40	36	34	38	28	44	28	24	34	42
17	32	30	34	30	30	32	36	30	34	36	28	36	32	22	38
18	30	36	34	30	32	34	36	30	28	26	32	32	18	36	30
19	34	32	34	30	36	30	20	30	32	30	40	24	30	32	30
20	36	26	30	28	32	34	30	34	26	24	32	38	30	32	32
21	40	30	38	32	40	34	34	30	34	24	28	30	34	36	32
22	34	32	22	32	38	32	34	30	32	34	32	34	32	30	32
23	42	36	26	32	30	28	30	30	32	30	36	34	32	34	28
24	32	36	28	34	34	30	30	28	36	36	36	30	38	38	30
25	40	28	28	30	30	34	34	32	32	32	40	22	30	22	30
26	26	34	32	30	36	28	30	28	34	34	36	38	32	32	30
27	30	34	28	30	32	28	34	36	30	34	32	30	36	32	34
28	32	28	32	28	36	24	36	32	24	36	32	28	36	40	36
29	36	36	36	32	36	28	28	36	40	32	32	36	32	32	40
30	26	38	32	32	30	34	32	24	30	30	28	36	30	30	32
31	26	22	32	32	38	26	32	32	30	30	36	36	30	30	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	30	30	32	32	34	34	24	32	34	30	28	36	34	38	48
35	34	30	28	36	26	38	24	28	34	34	28	28	30	38	28
36	36	32	40	34	30	26	34	36	32	24	40	38	34	34	26
37	32	32	28	34	26	34	30	36	36	40	36	30	30	34	38
38	34	26	20	34	32	32	30	28	26	34	32	34	28	28	30
39	34	26	36	30	28	36	34	40	30	38	28	26	28	36	30
40	32	34	30	32	40	38	34	30	26	28	36	30	42	32	32
41	32	30	34	32	32	34	30	30	26	40	40	38	26	36	36
42	34	36	30	40	30	28	34	34	28	34	28	30	28	30	36
43	30	40	30	28	22	36	30	30	20	26	32	30	32	26	36
44	32	34	26	34	42	32	32	30	34	32	28	32	28	30	34
45	36	38	34	34	38	28	24	30	30	36	28	32	32	26	34
46	34	24	30	26	32	34	28	34	36	34	32	32	30	32	26
47	34	36	34	38	36	34	28	46	24	34	32	32	30	28	30
48	32	26	30	30	30	28	32	34	30	36	36	32	36	30	38
49	32	30	26	34	34	36	32	30	26	36	28	32	36	34	34
50	34	28	30	34	32	38	36	30	40	34	36	28	34	32	30
51	30	32	30	26	36	34	28	30	36	30	28	28	30	28	38
52	28	34	38	32	36	30	34	38	30	36	36	30	30	32	32
53	40	30	30	44	28	30	30	34	30	28	32	30	34	36	40
54	30	32	34	28	30	20	34	26	32	30	32	26	28	30	36

55	38	28	30	20	30	32	30	42	32	34	28	34	36	34	32
56	32	32	40	38	30	38	30	28	36	32	32	34	34	30	30
57	40	32	32	42	34	26	34	32	32	36	28	34	34	30	30
58	38	30	32	34	36	28	34	28	30	30	36	26	32	32	34
59	18	22	28	42	32	28	30	36	34	22	32	26	36	32	30
60	32	32	36	24	32	32	28	28	36	36	32	28	36	36	32
61	28	32	40	36	32	36	28	32	28	40	32	28	32	28	28
62	38	34	32	28	30	26	36	28	30	30	32	28	26	34	32
63	38	26	40	36	30	34	36	20	30	38	32	36	34	34	32

Таблица 5

Анализ блока S_4 алгоритма шифрования DES для применения метода линейного криптоанализа

i	**Значения j**														
S(i,j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	32	36	32	36	32	32	32	32	32	32	36	32	36	32	32
3	28	32	40	32	32	40	28	28	24	32	32	24	32	28	32
4	30	30	32	30	32	32	22	34	32	32	26	32	26	42	32
5	34	30	28	34	32	28	38	34	36	32	22	36	42	38	32
6	30	34	32	30	28	28	34	30	28	36	34	32	30	34	40
7	30	30	36	30	44	32	30	34	32	44	34	36	34	34	32
8	28	32	32	32	32	32	36	36	32	32	32	32	32	28	32
9	32	28	24	36	32	40	32	32	24	32	36	40	28	32	32
10	28	28	32	36	32	32	28	28	32	32	36	32	28	28	32
11	36	28	32	28	32	32	28	28	32	32	28	32	28	36	32
12	30	34	32	30	36	36	26	30	36	28	42	32	22	26	24
13	30	30	28	30	36	32	22	34	32	36	26	28	26	42	32
14	30	30	32	30	32	32	30	34	32	32	34	32	34	34	32
15	34	30	36	34	32	36	30	34	28	32	30	28	34	30	32
16	30	30	32	34	36	36	42	30	36	28	38	32	26	42	40
17	34	30	28	30	28	32	26	30	32	28	42	28	42	38	42
18	34	30	32	30	32	32	26	30	32	32	42	32	42	38	32
19	34	34	28	30	32	36	22	34	28	32	26	36	38	22	32
20	36	28	32	28	32	32	36	28	32	32	28	32	28	28	32
21	36	36	40	28	32	32	36	36	32	32	36	24	28	36	32
22	40	28	32	36	40	24	32	40	24	24	36	32	28	32	32
23	36	40	40	40	32	32	28	28	32	32	32	40	32	36	32
24	34	30	32	30	32	32	34	30	32	32	34	32	34	30	32

25	34	34	36	30	32	28	30	34	36	32	34	28	30	30	32
26	30	30	32	34	28	28	34	30	28	36	30	32	34	34	24
27	34	30	20	30	36	32	34	30	32	36	34	20	34	30	32
28	28	32	32	32	40	24	36	28	24	24	32	32	32	36	32
29	32	28	24	28	32	32	32	32	32	32	36	24	36	32	32
30	24	32	32	32	32	32	32	40	32	32	32	32	32	32	32
31	32	24	40	40	32	32	32	32	32	32	32	24	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	36	32	32	32	24	40	36	36	40	40	32	32	32	36	16
35	32	36	32	36	32	32	32	32	32	32	36	32	36	32	32
36	34	34	32	30	36	36	38	34	36	28	34	32	30	38	24
37	30	34	36	34	28	32	30	34	32	28	38	36	38	34	32
38	22	34	32	34	32	32	30	42	32	32	30	32	30	34	32
39	30	22	36	42	32	36	34	30	28	32	30	28	34	34	32
40	32	36	32	28	40	24	32	32	24	24	28	32	36	32	16
41	28	32	32	32	32	32	36	36	32	32	32	32	32	28	32
42	36	28	32	28	32	32	28	28	32	32	28	32	28	36	32
43	36	36	32	28	32	48	36	36	16	32	28	32	36	36	32
44	30	34	32	34	32	32	30	34	32	32	38	32	38	34	32
45	30	30	28	34	32	28	26	30	36	32	30	36	34	26	32
46	34	42	32	22	28	28	30	34	28	36	34	32	30	30	40
47	22	34	28	34	20	32	30	42	32	20	30	28	30	34	32
48	30	30	32	30	32	32	38	34	32	32	34	32	34	26	32
49	34	30	28	34	32	36	30	34	28	32	38	36	26	30	32
50	30	34	32	30	36	36	34	30	36	28	26	32	38	34	40
51	30	30	36	30	36	32	38	34	32	36	34	36	34	26	32
52	32	40	32	24	40	40	32	32	40	24	32	32	32	32	32
53	24	32	32	32	40	32	32	40	32	40	32	32	32	32	32
54	32	28	32	28	32	32	32	32	32	32	36	32	36	32	32
55	36	32	40	32	32	32	28	36	32	32	32	24	32	28	32
56	22	34	32	30	28	28	34	22	28	36	34	32	30	34	24
57	30	22	44	22	28	32	30	34	32	28	34	44	34	34	32
58	30	22	32	22	32	32	30	34	32	32	34	32	34	34	32
59	42	30	36	34	32	28	30	42	36	32	30	28	34	30	32
60	36	40	32	40	32	32	28	28	32	32	32	32	32	36	32
61	34	36	40	28	32	32	32	24	32	32	28	24	36	32	32
62	28	28	32	36	40	40	28	28	40	24	28	32	36	28	32
63	36	28	32	28	24	32	36	28	32	24	28	32	28	28	32

Таблица 6

Анализ блока S_5 алгоритма шифрования DES для применения метода линейного криптоанализа

i	Значения j														
S(i,j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	36	30	34	30	34	28	32	36	32	34	30	34	30	32	28
3	32	30	38	30	30	36	28	32	32	30	38	30	30	36	28
4	34	30	32	32	34	30	32	32	34	34	36	28	30	30	32
5	34	34	28	32	42	26	28	32	34	22	32	36	30	34	36
6	30	28	26	30	28	34	32	32	30	32	30	26	24	34	32
7	34	32	34	30	40	38	32	28	38	32	26	30	32	26	28
8	32	34	38	32	32	30	26	30	34	36	20	34	38	28	36
9	28	38	30	32	28	26	26	38	30	32	28	34	26	24	28
10	36	32	32	30	26	34	34	34	34	30	34	36	28	28	32
11	36	36	36	38	34	30	30	30	30	30	34	32	24	28	32
12	34	32	30	32	34	36	42	30	36	30	24	30	36	26	28
13	38	32	34	32	30	36	22	30	32	30	36	30	40	26	32
14	30	30	32	30	36	32	34	30	32	36	34	28	38	30	28
15	30	30	40	38	36	32	34	34	36	40	30	40	26	34	32
16	34	30	32	32	30	26	24	32	30	30	28	32	34	42	12
17	34	30	32	36	34	30	28	36	34	34	32	24	26	34	36
18	30	32	30	34	28	30	24	36	38	36	38	30	36	26	32
19	26	32	34	30	36	34	32	36	26	36	34	26	36	30	32
20	35	28	32	32	32	32	32	28	28	36	36	32	36	28	32
21	35	32	28	28	36	24	24	32	32	28	36	40	36	32	36
22	32	38	38	34	30	36	32	36	32	38	34	34	34	32	32
23	36	26	30	38	30	28	36	36	28	26	34	30	34	32	36
24	38	32	34	36	22	28	34	34	32	30	32	34	36	30	28
25	34	36	26	32	30	36	30	38	40	38	36	42	32	34	28
26	34	34	24	30	36	32	34	30	32	36	34	32	30	30	32
27	34	38	28	26	32	32	34	38	40	32	30	28	26	30	32
28	32	30	34	36	32	26	34	30	38	28	32	34	30	32	32
29	36	30	38	24	32	30	34	42	30	24	24	34	34	32	36
30	28	24	32	30	30	30	34	30	34	30	38	36	36	36	32
31	28	40	24	34	26	26	30	30	34	30	30	24	32	32	28
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	28	30	34	30	34	28	40	28	32	26	38	34	30	16	20

35	32	30	30	38	30	28	36	32	32	30	30	30	38	36	28
36	30	38	36	32	38	30	36	36	26	30	36	32	46	34	32
37	38	34	32	32	38	34	32	28	26	34	24	32	30	38	28
38	34	36	30	30	32	34	28	36	30	28	30	38	32	30	32
39	22	32	30	38	36	38	28	32	38	20	34	34	32	38	28
40	36	30	30	32	36	26	34	34	26	36	32	38	30	28	32
41	32	34	38	32	32	38	34	34	30	24	32	30	26	32	32
42	32	28	24	38	38	38	26	38	34	30	30	24	36	28	36
43	40	32	36	38	30	26	38	34	38	30	38	28	32	36	36
44	34	36	26	32	26	32	38	30	28	34	28	30	36	38	32
45	30	28	30	32	30	24	34	30	32	26	24	30	32	30	36
46	38	34	28	38	36	36	30	22	24	32	30	36	30	34	32
47	38	26	28	38	28	36	30	34	36	36	26	32	34	30	28
48	34	30	32	28	26	30	28	36	34	34	32	32	34	34	36
49	34	30	32	32	30	34	32	32	30	30	28	32	34	34	36
50	38	32	30	30	40	34	36	32	42	32	34	30	36	34	32
51	26	32	42	34	32	30	28	32	38	32	22	34	36	30	32
52	32	20	36	28	32	36	24	28	32	28	32	28	28	32	32
53	24	32	32	40	28	36	32	32	28	28	32	36	36	28	36
54	36	30	26	30	30	40	32	36	28	30	30	38	34	28	32
55	24	26	26	26	38	32	36	44	32	34	30	34	34	36	28
56	34	36	26	32	30	36	30	26	36	26	32	38	36	30	32
57	30	40	34	28	38	28	26	30	28	34	36	30	32	34	32
58	38	22	32	34	36	32	30	38	28	32	34	36	30	30	28
59	30	26	28	22	32	24	30	22	36	36	30	32	34	30	36
60	24	26	30	32	28	34	34	26	34	36	32	42	30	36	36
61	36	34	34	36	36	30	34	30	42	32	32	34	34	36	32
62	28	35	28	34	34	30	34	34	30	30	30	36	28	32	36
63	28	28	28	46	38	26	30	34	30	38	30	32	32	28	32

Таблица 7

Анализ блока S_6 алгоритма шифрования DES для применения метода линейного криптоанализа

I	**Значения j**														
S(i,j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	32	32	32	34	26	34	26	34	38	30	34	38	32	32	36
3	32	32	32	30	30	30	30	30	26	34	30	36	40	32	36
4	32	34	30	34	30	40	32	34	34	32	36	36	40	38	30

5	32	38	26	34	38	36	28	30	30	24	36	40	36	30	22
6	36	30	30	32	32	34	30	32	32	30	34	24	36	34	34
7	28	34	34	36	28	34	30	32	32	34	30	36	32	26	34
8	34	30	32	28	30	30	24	34	32	40	38	38	28	32	30
9	30	38	36	32	38	34	32	30	32	36	38	30	32	32	42
10	34	34	36	34	36	24	34	32	34	30	32	26	28	36	30
11	30	26	40	34	32	40	30	32	30	22	36	34	32	28	34
12	30	32	34	34	28	34	40	28	30	32	30	30	28	34	36
13	34	36	34	30	28	34	36	36	34	28	30	34	44	42	32
14	34	32	38	28	34	28	30	30	32	30	28	26	40	34	36
15	30	36	30	28	38	32	30	34	32	30	32	30	32	34	32
16	34	32	30	32	34	28	18	28	30	28	26	32	34	20	42
17	34	32	30	36	30	40	38	28	30	28	26	36	30	32	30
18	34	32	30	30	32	34	24	26	40	30	40	28	34	36	30
19	34	32	30	30	32	26	32	30	36	26	28	32	30	40	42
20	30	34	32	34	28	28	30	30	28	28	34	28	30	26	28
21	38	30	28	38	24	28	30	34	40	36	26	28	38	30	32
22	26	38	32	36	34	26	32	32	30	30	36	32	34	30	32
23	26	34	36	28	34	30	36	32	38	26	32	32	34	26	36
24	32	30	34	32	32	30	34	30	34	36	28	34	30	32	32
25	20	30	22	24	28	38	30	26	26	40	28	30	38	36	32
26	40	34	30	34	34	28	32	32	24	38	34	30	30	28	32
27	28	34	34	38	34	32	32	24	36	30	30	34	30	32	32
28	32	32	32	30	26	34	30	32	28	36	40	34	26	34	34
29	36	36	32	30	22	30	30	32	24	24	32	34	30	30	26
30	36	40	28	28	28	32	32	30	34	30	26	38	30	34	34
31	32	28	36	32	28	32	36	34	34	30	30	30	34	30	34
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	36	32	28	30	34	30	42	34	26	30	46	32	32	36	44
35	36	32	28	34	30	34	38	30	30	34	26	32	32	28	36
36	32	30	34	30	34	32	24	30	30	32	28	36	32	42	34
37	32	34	30	30	26	28	36	34	34	32	36	32	36	26	34
38	40	34	38	32	36	38	38	28	32	38	30	36	36	26	30
39	32	38	42	28	24	30	30	20	40	34	34	32	32	34	30
40	34	34	28	32	34	30	32	30	36	32	30	38	20	36	34
41	30	34	40	36	26	26	32	34	28	28	30	38	32	36	30
42	38	30	36	34	32	28	34	28	26	30	28	30	36	36	34
43	34	30	32	42	36	28	30	36	30	38	32	38	40	36	30
44	30	40	26	34	36	34	32	36	30	32	22	30	28	34	36

45	34	36	34	30	36	26	36	28	34	36	30	34	28	34	24
46	38	24	26	32	26	32	30	38	36	30	32	38	32	30	36
47	34	36	42	24	38	36	30	42	36	38	28	34	32	30	32
48	34	32	30	32	34	36	26	32	34	32	30	28	30	40	30
49	34	32	30	36	30	32	30	32	34	32	30	32	26	36	34
50	30	32	34	34	24	30	32	30	24	34	32	20	30	36	26
51	30	32	34	26	32	30	32	34	36	30	36	32	34	32	30
52	30	30	36	30	32	28	30	30	28	32	30	40	34	34	36
53	22	42	32	34	28	28	30	42	32	32	30	32	34	30	32
54	22	26	32	28	38	30	32	32	26	26	28	32	30	34	32
55	38	38	36	28	30	42	28	24	26	30	32	32	30	30	36
56	32	26	38	36	36	30	26	30	26	40	32	30	34	32	32
57	36	34	34	28	32	30	30	34	26	28	32	34	34	36	32
58	36	38	30	26	30	24	32	32	28	34	34	38	34	32	32
59	40	30	26	22	38	28	32	32	32	26	30	26	34	28	24
60	32	32	32	30	34	34	38	28	32	32	28	30	38	30	30
61	36	28	40	30	30	38	30	44	28	28	28	30	26	34	30
62	32	24	32	40	36	28	32	26	34	26	34	38	34	34	30
63	28	20	32	20	28	28	36	30	34	34	30	38	30	30	30

Таблица 8

Анализ блока S_7 алгоритма шифрования DES для применения метода линейного криптоанализа

i	**Значения j**														
S(i,j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	32	34	26	36	28	34	34	34	34	32	32	30	30	32	32
3	32	30	38	32	32	34	26	34	34	28	28	26	34	32	40
4	32	34	34	32	28	26	30	36	36	42	34	36	32	26	38
5	32	34	42	32	36	26	30	32	24	38	30	32	28	38	42
6	36	32	28	24	32	36	28	30	26	30	34	38	30	34	26
7	28	36	32	36	36	28	36	34	30	38	34	38	30	30	30
8	28	32	36	30	30	30	38	28	36	32	32	30	34	34	30
9	36	28	32	34	26	30	22	32	32	32	32	30	26	30	26
10	32	30	22	34	30	28	32	34	38	32	36	32	32	34	34
11	32	30	30	42	30	28	32	38	26	36	32	36	28	30	30
12	32	30	30	34	26	32	32	32	28	30	34	30	26	36	32
13	32	34	34	30	30	32	24	32	36	34	30	34	30	28	24
14	32	32	32	34	34	38	30	30	26	26	38	28	24	28	32

15	32	32	32	34	34	30	38	38	26	34	38	28	32	36	32
16	30	34	36	32	30	30	32	30	36	36	38	30	28	40	18
17	32	30	36	28	30	30	36	30	32	32	30	42	36	24	30
18	30	32	34	28	34	28	22	32	30	32	18	36	26	36	30
19	34	32	38	36	38	36	26	32	26	32	30	28	38	28	26
20	34	40	38	32	22	36	30	38	40	30	28	30	28	34	36
21	30	28	30	36	30	36	26	34	32	30	32	30	32	30	28
22	30	34	28	24	34	34	32	32	30	30	32	32	26	30	36
23	34	34	24	24	22	34	28	20	38	34	32	36	34	34	36
24	34	30	28	34	32	28	38	34	40	32	26	32	34	34	24
25	30	38	24	34	28	28	26	38	32	44	34	28	34	30	32
26	30	32	34	30	32	30	36	32	34	28	34	34	32	30	32
27	34	36	34	34	32	38	32	36	34	36	30	34	36	34	32
28	34	32	30	30	40	34	32	34	32	38	32	28	34	36	30
29	30	40	34	30	28	34	36	34	28	30	36	20	30	28	26
30	34	38	28	30	32	36	34	40	30	30	32	34	32	32	34
31	30	34	36	34	32	36	30	32	34	34	32	38	32	32	30
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	32	34	34	28	36	26	34	34	34	32	24	30	30	16	24
35	32	30	30	32	32	34	34	34	34	28	36	34	26	24	40
36	32	30	38	32	36	30	34	28	36	30	38	36	32	30	34
37	32	30	30	32	28	30	34	32	32	34	26	24	36	34	30
38	28	28	32	24	32	24	32	38	34	34	38	30	30	30	30
39	20	32	36	28	28	40	32	34	30	34	30	30	30	26	34
40	32	28	36	30	26	26	30	36	24	28	40	38	46	30	30
41	32	32	32	34	30	34	30	32	36	28	32	30	38	26	34
42	28	34	30	34	34	32	32	26	34	36	36	32	28	30	34
43	36	42	38	34	34	32	24	38	38	24	40	36	32	34	30
44	36	38	34	34	30	32	44	32	32	30	30	38	22	36	28
45	28	34	38	30	26	24	36	32	32	26	26	26	34	36	36
46	36	32	36	26	30	38	34	30	30	34	34	36	36	36	28
47	44	24	36	34	30	30	34	38	38	34	34	28	28	28	36
48	34	34	32	36	38	34	32	30	40	36	34	34	36	28	34
49	38	30	32	32	38	34	36	30	36	32	26	30	28	28	30
50	34	32	38	32	26	32	30	32	34	32	38	32	26	32	30
51	38	32	26	32	38	32	26	32	30	32	34	32	30	32	34
52	30	36	30	28	38	36	34	30	36	34	36	26	36	34	32
53	26	24	38	32	30	36	30	34	36	42	32	34	32	38	32
54	34	30	36	28	34	26	28	24	34	34	40	28	26	30	32

55	38	30	32	36	30	34	32	36	34	30	32	32	34	34	32
56	34	26	32	30	36	36	30	34	32	36	30	28	30	34	32
57	38	26	28	30	24	28	34	30	24	32	30	32	30	30	32
58	38	36	30	34	36	22	28	32	34	24	30	30	36	38	32
59	34	32	30	14	36	30	32	44	34	32	34	30	32	34	32
60	26	40	30	34	36	22	28	26	32	30	32	32	30	32	34
61	30	24	34	34	32	30	32	26	36	30	36	24	34	32	30
62	26	38	28	34	28	32	30	32	30	30	32	30	36	28	30
63	46	42	36	30	28	32	34	24	26	42	32	26	36	28	34

Таблица 9

Анализ блока S_8 алгоритма шифрования DES для применения метода линейного криптоанализа

I	**Значения j**														
S(i,j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	34	32	30	30	32	34	32	30	32	30	36	28	30	32	38
3	30	32	34	30	36	26	36	30	28	38	32	28	34	32	34
4	30	30	32	32	34	30	28	34	32	36	30	38	32	32	46
5	30	30	32	32	34	38	36	30	44	24	34	26	36	36	34
6	32	34	34	34	38	32	28	36	36	34	34	34	30	36	24
7	28	34	22	26	34	40	32	32	28	30	34	30	30	32	32
8	32	34	30	32	32	34	30	30	34	32	32	34	30	36	28
9	36	30	30	36	32	26	34	34	34	32	44	26	26	32	28
10	34	30	32	34	36	28	30	32	30	30	36	38	28	32	30
11	34	34	44	38	40	36	30	36	26	30	36	30	28	36	34
12	30	32	30	32	34	32	26	32	34	28	22	32	38	36	26
13	34	28	30	36	34	32	38	32	30	32	38	28	38	36	30
14	32	32	36	30	34	34	34	30	26	34	34	36	36	36	32
15	32	36	32	26	30	26	34	30	34	38	38	40	32	36	32
16	32	32	32	32	32	32	24	32	32	32	32	32	24	32	16
17	32	32	32	32	24	32	32	32	40	32	24	32	40	32	32
18	34	32	30	30	32	34	40	30	32	30	36	28	38	32	22
19	30	32	34	30	28	42	36	30	36	22	40	28	26	32	34
20	30	26	28	36	30	30	28	30	36	28	30	30	32	36	34
21	30	34	36	20	38	30	36	26	24	24	34	34	36	32	30
22	32	38	22	30	26	32	28	40	32	26	34	26	30	32	36
23	28	30	34	38	30	32	32	36	32	30	34	38	30	36	36
24	28	34	34	32	36	34	34	38	30	32	28	34	34	28	24

25	32	38	42	28	36	26	30	34	30	32	32	26	30	32	32
26	30	38	28	26	32	28	34	32	34	30	32	38	32	32	34
27	30	34	32	22	36	36	26	28	30	30	24	30	32	28	30
28	26	28	30	36	34	32	38	36	34	36	34	40	34	32	26
29	30	24	30	32	34	24	34	28	30	32	34	36	34	32	30
30	28	28	36	34	34	34	30	34	26	26	30	28	32	32	32
31	28	32	32	22	30	34	30	26	34	38	34	32	28	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	30	36	30	34	32	26	36	30	28	26	28	32	30	48	34
35	34	28	34	34	28	26	40	30	32	26	24	32	26	24	30
36	34	34	24	32	38	34	36	30	32	36	34	34	32	32	34
37	34	34	24	32	38	26	28	34	36	32	30	30	28	28	30
38	40	34	26	30	26	28	32	32	24	38	30	26	30	36	32
39	28	26	30	22	30	28	36	36	32	34	38	30	30	32	32
40	32	34	30	22	30	28	36	36	32	34	38	32	30	32	32
41	36	30	30	36	32	34	26	30	30	28	40	38	38	20	32
42	30	34	32	30	28	36	34	36	30	30	32	38	24	36	30
43	22	30	28	34	40	36	34	24	34	38	32	30	32	32	34
44	26	36	30	32	30	28	18	32	30	32	38	32	34	32	34
45	30	32	30	36	30	28	30	24	34	28	30	36	26	40	30
46	32	32	36	34	22	30	30	22	34	34	34	32	32	32	28
47	40	28	32	30	42	30	30	30	42	38	30	28	28	32	36
48	28	32	36	32	36	32	36	32	36	32	28	32	20	32	36
49	36	32	28	32	36	32	36	32	36	32	28	32	28	32	28
50	26	28	26	34	36	34	32	30	32	34	32	32	34	40	30
51	38	36	38	34	32	34	36	30	36	34	28	32	38	32	34
52	30	22	32	28	30	34	32	34	32	28	30	42	28	28	34
53	38	30	32	28	30	34	32	38	36	32	34	38	32	32	30
54	36	38	34	34	26	36	36	28	32	38	34	34	26	32	32
55	32	38	38	26	30	28	32	32	40	26	34	38	34	36	32
56	40	42	30	32	40	34	30	26	38	28	36	38	34	32	32
57	36	30	30	28	32	26	34	38	30	28	32	30	38	36	32
58	22	42	32	38	28	36	34	28	30	38	32	30	32	28	30
59	38	30	28	34	32	36	34	24	26	30	32	38	32	32	34
60	34	32	34	28	26	36	34	36	34	40	30	32	34	36	30
61	30	44	26	40	34	28	38	36	30	28	30	36	34	28	34
62	24	36	32	30	34	30	38	42	38	34	34	32	32	28	32
63	24	32	36	34	30	22	30	26	38	30	38	28	36	28	32

Таблица 10

Анализ блока S_1 алгоритма шифрования DES для применения метода дифференциального криптоанализа

	0000	1000	0100	1100	0010	1010	0110	1110	0001	1001	0101	1011	1100	1101	1110	1111
000001	0	0	0	6	0	2	4	4	0	10	12	4	10	6	2	4
000010	0	0	0	8	0	4	4	4	0	6	8	6	12	6	4	2
000011	14	4	2	2	10	6	4	2	6	4	4	0	2	2	2	0
000100	0	0	0	6	0	10	10	6	0	4	6	4	2	8	6	2
000101	4	8	6	2	2	4	4	2	0	4	4	0	12	2	4	6
000110	0	4	2	4	8	2	6	2	8	4	4	2	4	2	0	12
000111	2	4	10	4	0	4	8	4	2	4	8	2	2	2	4	4
001000	0	0	0	12	0	8	8	4	0	6	2	8	8	2	2	4
001001	10	2	4	0	2	4	6	0	2	2	8	0	10	0	2	12
001010	0	8	6	2	2	8	6	0	6	4	6	0	4	0	2	10
001011	2	4	0	10	2	2	4	0	2	6	2	6	6	4	2	12
001100	0	0	0	8	0	6	6	0	0	6	6	4	6	6	14	2
001101	6	6	4	8	4	8	2	6	0	6	4	6	0	2	0	2
001110	0	4	8	8	6	6	4	0	6	6	4	0	0	4	0	8
001111	2	0	2	4	4	6	4	2	4	8	2	2	2	6	8	8
010000	0	0	0	0	0	0	2	14	0	6	6	12	4	6	8	6
010001	6	8	2	4	6	4	8	6	4	0	6	6	0	4	0	0
010010	0	8	4	2	6	6	4	6	6	4	2	6	6	0	4	0
010011	2	4	4	6	2	0	4	6	2	0	6	8	4	6	4	6
010100	0	8	8	0	10	0	4	2	8	2	2	4	4	8	4	0
010101	0	4	6	4	2	2	4	10	6	2	0	10	0	4	6	4
010110	0	8	10	8	0	2	2	6	10	2	0	2	0	6	2	6
010111	4	4	6	0	10	6	0	2	4	4	4	6	6	6	2	0
011000	0	6	6	0	8	4	2	2	2	4	6	8	6	6	2	2
011001	2	6	2	4	0	8	4	6	10	4	0	4	2	8	4	0
011010	0	6	4	0	4	6	6	6	6	2	2	0	4	4	6	8
011011	4	4	2	4	10	6	6	4	6	2	2	4	2	2	4	2
011100	0	10	10	6	6	0	0	12	6	4	0	0	2	4	4	0
011101	4	2	4	0	8	0	0	2	10	0	2	6	6	6	14	0
011110	0	2	6	0	14	2	0	0	6	4	10	8	2	2	6	2
011111	2	4	10	6	2	2	2	8	6	8	0	0	0	4	6	4
100000	0	0	0	10	0	12	8	2	0	6	4	4	4	2	0	12
100001	0	4	2	4	4	8	10	0	4	4	10	0	4	0	2	8
100010	10	4	6	2	2	8	2	2	2	2	6	0	4	0	4	10

100011	0	4	4	8	0	2	6	0	6	6	2	10	2	4	0	10
100100	12	0	0	2	2	2	2	0	14	14	2	0	2	6	2	4
100101	6	4	4	12	4	4	4	10	2	2	2	0	4	2	2	2
100110	0	0	4	10	10	10	2	4	0	4	6	4	4	4	2	0
100111	10	4	2	0	2	4	2	0	4	8	0	4	8	8	4	4
101000	12	2	2	8	2	6	12	0	0	2	6	0	4	0	6	2
101001	4	2	2	10	0	2	4	0	0	14	10	2	4	6	0	4
101010	4	2	4	6	0	2	8	2	2	14	2	6	2	6	2	2
101011	12	2	2	2	4	6	6	2	0	2	6	2	6	0	8	4
101100	4	2	2	4	0	2	10	4	2	2	4	8	8	4	2	6
101101	6	2	6	2	8	4	4	4	2	4	6	0	8	2	0	6
101110	6	6	2	2	0	2	4	6	4	0	6	2	12	2	6	4
101111	2	2	2	2	2	6	8	8	2	4	4	6	8	2	4	2
110000	0	4	6	0	12	6	2	2	8	2	4	4	6	2	2	4
110001	4	8	2	10	2	2	2	2	6	0	0	2	2	4	10	8
110010	4	2	6	4	4	2	2	4	6	6	4	8	2	2	8	0
110011	4	4	6	2	10	8	4	2	4	0	2	2	4	6	2	4
110100	0	8	16	6	2	0	0	12	6	0	0	0	0	8	0	6
110101	2	2	4	0	8	0	0	0	14	4	6	8	0	2	14	0
110110	2	6	2	2	8	0	2	2	4	2	6	8	6	4	10	0
110111	2	2	12	4	2	4	4	10	4	4	2	6	0	2	2	4
111000	0	6	2	2	2	0	2	2	4	6	4	4	4	6	10	10
111001	6	2	2	4	12	6	4	8	4	0	2	4	2	4	4	0
111010	6	4	6	4	6	8	0	6	2	2	6	2	2	6	4	0
111011	2	6	4	0	0	2	4	6	4	6	8	6	4	4	6	2
111100	0	10	4	0	12	0	4	2	6	0	4	12	4	4	2	0
111101	0	8	6	2	2	6	0	8	4	4	0	4	0	12	4	4
111110	4	8	2	2	2	4	4	14	4	2	0	2	0	8	4	4
111111	4	8	4	2	4	0	2	4	4	2	4	8	8	6	2	2

Таблица 11

Анализ блока S_2 алгоритма шифрования DES для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
000001	0	0	0	4	0	2	6	4	0	14	8	6	8	4	6	2
000010	0	0	0	2	0	4	6	4	0	0	4	6	10	10	12	6
000011	4	8	4	8	4	6	4	2	4	2	2	4	6	2	0	4
000100	0	0	0	0	0	6	0	14	0	6	10	4	10	6	4	4

000101	2	0	4	8	2	4	6	6	2	0	8	4	2	4	10	2
000110	0	12	6	4	6	4	6	2	2	10	2	8	2	0	0	0
000111	4	6	6	4	2	4	4	2	6	4	2	4	4	6	0	6
001000	0	0	0	4	0	4	0	8	0	10	16	6	6	0	6	4
001001	14	2	4	10	2	8	2	6	2	4	0	0	2	2	2	4
001010	0	6	6	2	10	4	10	2	6	2	2	4	2	2	4	2
001011	6	2	2	0	2	4	6	2	10	2	0	6	6	4	4	8
001100	0	0	0	4	0	14	0	10	0	6	2	4	4	8	6	6
001101	6	2	6	2	10	2	0	4	0	10	4	2	8	2	2	4
001110	0	6	12	8	0	4	2	0	8	2	4	4	6	2	0	6
001111	0	8	2	0	6	6	8	2	4	4	4	6	8	0	4	2
010000	0	0	0	8	0	4	10	2	0	2	8	10	0	10	6	4
010001	6	6	4	6	4	0	6	4	8	2	10	2	2	4	0	0
010010	0	6	2	6	2	4	12	4	6	4	0	4	4	6	2	2
010011	4	0	4	0	8	6	6	0	0	2	0	6	4	8	2	14
010100	0	6	6	4	10	1	2	12	6	2	2	2	4	4	2	2
010101	6	8	2	0	8	2	0	2	2	2	2	2	2	14	10	2
010110	0	8	6	4	2	2	4	2	6	4	6	2	6	0	6	6
010111	6	4	8	6	4	4	0	4	6	2	4	4	4	2	4	2
011000	0	6	4	6	10	4	0	2	4	8	0	0	4	8	2	6
011001	2	4	6	4	4	2	4	2	6	4	6	8	0	6	4	2
011010	0	6	8	4	2	4	2	2	8	2	2	6	2	4	4	8
011011	0	6	4	4	0	12	6	4	2	2	2	4	4	2	10	2
011100	0	4	6	6	12	0	4	0	10	2	6	2	0	0	10	2
011101	0	6	2	2	6	0	4	16	4	4	2	0	0	4	6	8
011110	0	4	8	2	10	6	6	0	8	4	0	2	4	4	0	6
011111	4	2	6	6	2	2	2	4	8	6	10	6	4	0	0	2
100000	0	0	0	2	0	12	10	4	0	0	0	2	14	2	8	10
100001	0	4	6	8	2	10	4	2	2	6	4	2	6	2	0	6
100010	4	12	8	4	2	2	0	0	2	8	8	6	0	6	0	2
100011	8	2	0	2	8	4	2	6	4	8	2	2	6	4	2	4
100100	10	4	0	0	0	4	0	2	6	8	6	10	8	0	2	4
100101	6	0	12	2	8	6	10	0	0	8	2	6	0	0	2	2
100110	2	2	4	4	2	2	10	14	2	0	4	2	2	4	6	4
100111	6	0	0	2	6	4	2	4	4	4	8	4	8	0	6	6
101000	8	0	8	2	4	12	2	0	2	6	2	0	6	2	0	10
101001	0	2	4	10	2	8	6	4	0	10	0	2	10	0	2	4
101010	4	0	4	8	6	2	4	4	6	6	2	6	2	2	4	4
101011	2	2	6	4	0	2	2	6	2	8	8	4	4	4	8	2
101100	10	6	8	6	0	6	4	4	4	2	4	4	0	0	2	4

101101	2	2	2	4	0	0	0	2	8	4	4	6	10	2	14	4
101110	2	4	0	2	10	4	2	0	2	2	6	2	8	8	10	2
101111	12	4	6	8	2	6	2	8	0	4	0	2	0	8	2	0
110000	0	4	0	2	4	4	8	6	10	6	2	12	0	0	0	6
110001	0	10	2	0	6	2	10	2	6	0	2	0	6	6	4	8
110010	8	4	6	0	6	4	4	8	4	6	8	0	2	2	2	0
110011	2	2	6	10	2	0	0	6	4	4	12	8	4	2	2	0
110100	0	12	6	4	6	0	4	4	4	0	4	6	4	2	4	4
110101	0	12	4	6	2	4	4	0	10	0	0	8	0	8	0	6
110110	8	2	4	0	4	0	4	2	0	8	4	2	6	16	2	2
110111	6	2	2	2	6	6	4	8	2	2	6	2	2	2	4	6
111000	0	8	8	10	6	2	2	0	4	0	4	2	4	0	4	10
111001	0	2	0	0	8	0	10	4	10	0	8	4	4	4	4	6
111010	4	0	2	8	4	2	2	2	4	8	2	0	4	10	10	2
111011	16	4	4	2	8	2	2	6	4	4	4	2	0	2	2	2
111100	0	2	6	2	8	4	6	0	10	2	2	4	4	10	4	0
111101	0	16	10	2	4	2	4	2	8	0	0	8	0	6	2	0
111110	4	4	0	10	2	4	2	14	4	2	6	6	0	0	6	0
111111	4	0	0	2	0	8	2	4	0	2	4	4	4	14	10	6

Таблица 12

Анализ блока S_3 алгоритма шифрования DES для применения метода дифференциального криптоанализа

	0000	1000	0100	1100	0010	1010	0110	1110	0001	1001	0101	1101	0011	1011	0111	1111
000001	0	0	0	2	0	4	2	12	0	14	0	4	8	2	6	10
000010	0	0	0	2	0	2	0	8	0	4	12	10	4	6	8	8
000011	8	6	10	4	8	6	0	6	4	4	0	0	0	4	2	2
000100	0	0	0	4	0	2	4	2	0	12	8	4	6	8	10	4
000101	6	2	4	8	6	10	6	2	2	8	2	0	2	0	4	2
000110	0	10	6	6	10	0	4	12	2	4	0	0	6	4	0	0
000111	2	0	0	4	4	4	4	2	10	4	4	8	4	4	4	6
001000	0	0	0	10	0	4	4	6	0	6	6	6	6	0	8	8
001001	10	2	0	2	10	4	6	2	0	6	0	4	6	2	4	6
001010	0	10	6	0	14	6	4	0	4	6	6	0	4	0	2	2
001011	2	6	2	10	2	2	4	0	4	2	6	0	2	8	14	0
001100	0	0	0	8	0	12	12	4	0	8	0	4	2	10	2	2
001101	8	2	8	0	0	4	2	0	2	8	14	2	6	2	4	2
001110	0	4	4	2	4	2	4	4	10	4	4	4	4	4	2	8

001111	4	6	4	6	2	2	4	8	6	2	6	2	0	6	2	4
010000	0	0	0	4	0	12	4	8	0	4	2	6	2	14	0	8
010001	8	2	2	6	4	0	2	0	8	4	12	2	10	0	2	2
010010	0	2	8	2	4	8	0	8	8	0	2	2	4	2	14	0
010011	4	4	12	0	2	2	2	10	2	2	2	2	4	4	4	8
010100	0	6	4	4	6	4	6	2	8	6	6	2	2	0	0	8
010101	4	8	2	8	2	4	8	0	4	2	2	2	2	6	8	2
010110	0	6	10	2	8	4	2	0	2	2	2	8	4	6	4	4
010111	0	6	6	0	6	2	4	4	6	2	2	10	6	8	2	0
011000	0	8	4	6	6	0	6	2	4	0	4	2	10	0	6	6
011001	4	2	4	8	4	2	10	2	2	2	6	8	2	6	0	2
011010	0	8	6	4	4	0	6	4	4	8	0	10	2	2	2	4
011011	4	10	2	0	2	4	2	4	8	2	2	8	4	2	8	2
011100	0	6	8	8	4	2	8	0	12	0	10	0	4	0	2	0
011101	0	2	0	6	2	8	4	6	2	0	4	2	4	10	0	14
011110	0	4	8	2	4	6	0	4	10	0	2	6	4	8	4	2
011111	0	6	8	0	10	6	4	6	4	2	2	10	4	0	0	2
100000	0	0	0	0	0	4	4	8	0	2	2	4	10	16	12	2
100001	10	8	8	0	8	4	2	4	0	6	6	6	0	0	2	0
100010	12	6	4	4	2	4	10	2	0	4	4	2	4	4	0	2
100011	2	2	0	6	0	2	4	0	4	12	4	2	6	4	8	8
100100	4	8	2	12	6	4	2	10	2	2	2	4	2	0	4	0
100101	6	0	2	0	8	2	0	2	8	8	2	2	4	4	10	6
100110	6	2	0	4	4	0	4	0	4	2	14	0	8	10	0	6
100111	0	2	4	16	8	6	6	6	0	2	4	4	0	2	2	2
101000	6	2	10	0	6	4	0	4	4	2	4	8	2	2	8	2
101001	0	2	8	4	0	4	0	6	4	10	4	8	4	4	4	2
101010	2	6	0	4	2	4	4	6	4	8	4	4	4	2	4	6
101011	10	2	6	6	4	4	8	0	4	2	2	0	2	4	4	6
101100	10	4	6	2	4	2	2	2	4	10	4	4	0	2	6	2
101101	4	2	4	4	4	2	4	16	2	0	0	4	4	2	6	6
101110	4	0	2	10	0	6	10	4	2	6	6	2	2	0	2	8
101111	8	2	0	0	4	4	4	2	6	4	6	2	4	8	4	6
110000	0	10	8	6	2	0	4	2	10	4	4	6	2	0	6	0
110001	2	6	2	0	4	2	8	8	2	2	2	0	2	12	6	6
110010	2	0	4	8	2	8	4	4	8	4	2	8	6	2	0	2
110011	4	4	6	8	6	6	0	2	2	2	6	4	12	0	0	2
110100	0	6	2	2	16	2	2	2	12	2	4	0	4	2	0	8
110101	4	6	0	10	8	0	2	2	6	0	0	6	2	10	2	6
110110	4	4	4	4	0	6	6	4	4	4	4	4	0	6	2	8

110111	4	8	2	4	2	2	6	0	2	4	8	4	10	0	6	2
111000	0	8	12	0	2	2	6	6	2	10	2	2	0	8	0	4
111001	2	6	4	0	6	4	6	4	8	0	4	4	2	4	8	2
111010	6	0	2	2	4	6	4	4	4	2	2	6	12	2	6	2
111011	2	2	6	0	0	10	4	8	4	2	4	8	4	4	0	6
111100	0	2	4	2	12	2	0	6	2	0	2	8	4	6	4	10
111101	4	6	8	6	2	2	2	2	10	2	6	6	2	4	2	0
111110	8	6	4	4	2	10	2	0	2	2	4	2	4	2	10	2
111111	2	6	4	0	0	10	8	2	2	8	6	4	6	2	0	4

Таблица 13

Анализ блока S_4 алгоритма шифрования DES для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
000001	0	0	0	0	0	16	16	0	0	16	16	0	0	0	0	0
000010	0	0	0	8	0	4	4	8	0	4	4	8	8	8	8	0
000011	8	6	2	0	2	4	8	2	6	0	4	6	0	6	2	8
000100	0	0	0	8	0	0	12	4	0	12	0	4	8	4	4	8
000101	4	2	2	8	2	12	0	2	2	0	12	2	8	2	2	4
000110	0	8	8	4	8	8	0	0	8	0	8	0	4	0	0	8
000111	4	2	6	4	6	0	16	6	2	0	0	2	4	2	6	4
001000	0	0	0	4	0	8	4	8	0	4	8	8	4	8	8	0
001001	8	4	4	4	4	0	8	4	4	0	0	4	4	4	4	8
001010	0	6	6	0	6	4	4	6	6	4	4	6	0	6	6	0
001011	0	12	0	8	0	0	0	0	12	0	0	12	8	12	0	0
001100	0	0	0	4	0	8	4	8	0	4	8	8	4	8	8	0
001101	8	4	4	4	4	0	0	4	4	8	0	4	4	4	4	8
001110	0	6	6	4	6	0	4	6	6	4	0	6	4	6	6	0
001111	0	6	6	4	6	4	0	6	6	0	4	6	4	6	6	0
010000	0	0	0	0	0	8	12	4	0	12	8	4	0	4	4	8
010001	4	2	2	16	2	4	0	2	2	0	4	2	16	2	2	4
010010	0	0	0	8	0	4	4	8	0	4	4	8	8	8	8	0
010011	8	2	6	0	6	4	0	6	2	8	4	2	0	2	6	8
010100	0	8	8	0	8	0	8	0	8	8	0	0	0	0	0	16
010101	8	4	4	0	4	8	0	4	4	0	8	4	0	4	4	8
010110	0	8	8	4	8	8	0	0	8	0	8	0	4	0	0	8
010111	4	6	2	4	2	0	0	2	6	16	0	6	4	6	2	4
011000	0	8	8	8	8	4	0	0	8	0	4	0	8	0	0	8

011001	4	4	4	0	4	4	16	4	4	0	4	4	0	4	4	4
011010	0	6	6	4	6	0	4	6	6	4	0	6	4	6	6	0
011011	0	6	6	4	6	4	0	6	6	0	4	6	4	6	6	0
011100	0	8	8	8	8	4	0	0	8	0	4	0	8	0	0	8
011101	4	4	4	0	4	4	0	4	4	16	4	4	0	4	4	4
011110	0	6	6	0	6	4	4	6	6	4	4	6	0	6	6	0
011111	0	0	12	8	12	0	0	12	0	0	0	0	8	0	12	0
100000	0	0	0	8	0	0	0	12	0	0	0	12	8	12	12	0
100001	0	4	8	0	8	4	8	8	4	0	4	4	0	4	8	0
100010	8	2	2	0	2	4	8	6	2	8	4	6	0	6	6	0
100011	4	6	2	8	2	4	0	2	6	0	4	6	8	6	2	4
100100	0	6	6	4	6	4	0	6	6	0	4	6	4	6	6	0
100101	0	8	4	4	4	0	0	4	8	8	0	8	4	8	4	0
100110	0	6	6	0	6	4	8	2	6	8	4	2	0	2	2	8
100111	4	6	2	8	2	4	0	2	6	0	4	6	8	6	2	4
101000	16	4	4	0	4	4	4	4	4	4	4	4	0	4	4	0
101001	0	6	2	8	2	4	0	2	6	8	4	6	8	6	2	0
101010	0	2	2	16	2	4	4	2	2	4	4	2	16	2	2	0
101011	8	0	4	0	4	8	16	4	0	0	8	0	0	0	4	8
101100	8	4	4	4	4	0	8	4	4	8	0	4	4	4	4	0
101101	4	2	6	4	6	8	0	6	2	0	8	2	4	2	6	4
101110	16	0	0	0	0	16	0	0	0	0	16	0	0	0	0	16
101111	16	0	0	0	0	0	16	0	0	16	0	0	0	0	0	16
110000	0	6	6	4	6	4	0	6	6	0	4	6	4	6	6	0
110001	0	8	4	4	4	0	0	4	8	8	0	8	4	8	4	0
110010	16	6	6	4	6	0	4	2	6	4	0	2	4	2	2	0
110011	0	2	6	4	6	8	8	6	2	0	8	2	4	2	6	0
110100	0	12	12	8	12	0	0	0	12	0	0	0	8	0	0	0
110101	0	4	8	0	8	4	8	8	4	0	4	4	0	4	8	0
110110	0	2	2	4	2	0	4	6	2	4	0	6	4	6	6	16
110111	0	2	6	4	6	8	8	6	2	0	8	2	4	2	6	0
111000	0	4	4	0	4	4	4	4	4	4	4	4	0	4	4	16
111001	0	6	2	8	2	4	0	2	6	8	4	6	8	6	2	0
111010	0	4	4	0	4	8	8	4	4	8	8	4	0	4	4	0
111011	16	4	4	0	4	0	0	4	4	0	0	4	0	4	4	16
111100	0	4	4	4	4	0	8	4	4	8	0	4	4	4	4	8
111101	4	2	6	4	6	8	0	6	2	0	8	2	4	2	6	4
111110	0	2	2	8	2	12	4	2	2	4	12	2	8	2	2	0
111111	8	4	0	8	0	0	0	0	4	16	0	4	8	4	0	8

Таблица 14

Анализ блока S_5 алгоритма шифрования DES для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
000001	0	0	0	4	0	10	8	6	0	4	2	2	12	10	2	4
000010	0	0	0	4	0	10	6	4	0	6	4	2	4	8	10	6
000011	8	2	4	6	4	4	2	2	6	8	6	4	4	0	2	2
000100	0	0	0	8	0	4	10	6	0	6	6	4	8	6	0	6
000101	12	2	0	4	0	4	8	2	4	0	16	2	0	2	0	8
000110	0	8	4	6	4	6	2	2	4	4	6	0	6	0	2	10
000111	2	0	4	8	4	2	6	6	2	8	6	2	2	0	6	6
001000	0	0	0	2	0	8	10	4	0	4	10	4	8	4	4	6
001001	8	6	0	4	0	6	6	2	2	10	2	8	6	2	0	2
001010	0	6	8	6	0	8	0	0	8	10	4	2	8	0	0	4
001011	4	2	2	4	8	10	6	4	2	6	2	2	6	2	2	2
001100	0	0	0	10	0	2	10	2	0	6	10	6	6	6	2	4
001101	10	4	2	2	0	6	16	0	0	2	10	2	2	4	0	4
001110	0	6	4	8	4	6	10	2	4	4	4	2	4	0	2	4
001111	4	4	0	8	0	2	0	2	8	2	4	2	8	4	4	12
010000	0	0	0	0	0	4	4	12	0	2	8	10	4	6	12	2
010001	6	6	10	10	4	0	2	6	2	4	0	6	2	4	2	0
010010	0	2	4	2	10	4	0	10	8	6	0	6	0	6	6	0
010011	0	0	6	2	8	0	0	4	4	6	2	8	2	8	10	4
010100	0	12	2	6	4	0	4	4	8	4	4	4	6	2	4	0
010101	4	8	0	2	8	0	2	4	2	2	4	2	4	8	8	6
010110	0	6	10	2	14	0	2	2	4	4	0	6	0	4	6	4
010111	0	6	8	4	8	4	0	2	8	4	0	2	2	8	6	2
011000	0	10	8	0	6	4	0	4	4	4	6	4	4	4	0	6
011001	0	4	6	2	4	4	2	6	4	2	2	4	12	2	10	0
011010	0	2	16	2	12	2	0	6	4	0	0	4	0	4	4	8
011011	2	8	12	0	0	2	2	6	8	4	0	6	0	0	8	6
011100	0	10	2	6	6	6	6	4	8	2	0	4	4	4	2	0
011101	4	6	2	0	8	2	4	6	6	0	8	6	2	4	2	4
011110	0	2	6	2	4	0	0	2	12	2	2	6	2	10	10	4
011111	0	6	8	4	8	8	0	6	6	2	0	6	0	6	2	2
100000	0	0	0	8	0	8	2	6	0	4	4	4	6	6	8	8
100001	0	0	0	6	6	2	6	4	6	10	14	4	0	0	4	2
100010	14	4	0	10	0	2	12	2	2	2	10	2	0	0	2	2

100011	2	0	0	4	2	2	10	4	0	8	8	2	6	8	0	8
100100	6	2	8	4	4	4	6	2	2	6	6	2	6	2	2	2
100101	6	0	0	8	2	8	2	6	6	4	2	2	4	2	6	6
100110	12	0	0	4	0	4	4	4	0	8	4	0	12	8	0	4
100111	12	2	0	2	0	12	2	2	4	4	8	4	8	2	2	0
101000	2	8	4	6	2	4	6	0	6	6	4	0	2	2	2	10
101001	6	4	6	8	8	4	6	2	0	0	2	2	10	0	2	4
101010	4	4	0	2	2	4	6	2	0	0	6	4	10	4	4	12
101011	4	6	2	6	0	0	12	2	0	4	12	2	6	4	0	4
101100	8	6	2	6	4	8	6	0	4	4	0	2	6	0	6	2
101101	4	4	0	4	0	6	4	2	4	12	0	4	4	6	4	6
101110	6	0	2	4	0	6	6	4	2	10	6	10	6	2	0	0
101111	10	4	0	2	2	6	10	2	0	2	2	4	6	2	2	10
110000	0	4	8	4	6	4	0	6	10	4	2	4	2	6	4	0
110001	0	6	6	4	10	2	0	0	4	4	0	0	4	6	12	6
110010	4	6	0	2	6	4	6	0	6	0	4	6	4	10	6	0
110011	8	10	0	14	8	0	0	8	2	0	2	4	0	4	4	0
110100	0	4	4	2	14	4	0	8	6	8	2	2	0	4	6	0
110101	0	4	16	0	8	4	0	4	4	4	0	8	0	4	4	4
110110	4	4	4	6	2	2	2	12	2	4	4	8	2	4	4	0
110111	4	2	2	2	4	2	0	8	2	2	2	12	6	2	8	6
111000	0	4	8	4	12	0	0	8	10	2	0	0	0	4	2	10
111001	0	8	12	0	2	2	2	2	12	4	0	8	0	4	4	4
111010	0	14	4	0	4	6	0	0	6	2	10	8	0	0	4	6
111011	0	2	2	2	4	4	8	6	8	2	2	2	6	14	2	0
111100	0	0	10	2	6	0	0	2	6	2	2	10	2	4	10	8
111101	0	6	12	2	4	8	0	8	8	2	2	0	2	2	4	4
111110	4	4	10	0	2	4	8	8	2	2	0	2	6	8	4	0
111111	8	6	6	0	4	2	2	4	4	2	8	6	2	4	6	0

Таблица 15

Анализ блока S_6 алгоритма шифрования DES для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
000001	0	0	0	6	0	2	6	2	0	4	2	4	6	16	14	2
000010	0	0	0	2	0	10	6	10	0	2	4	8	6	6	8	2
000011	0	8	0	8	0	6	4	6	4	4	4	12	2	4	2	0
000100	0	0	0	8	0	0	8	0	0	6	8	10	2	4	10	8

000101	10	2	4	4	4	8	8	4	2	2	0	4	0	8	0	4
000110	0	8	4	4	8	4	2	2	12	0	2	6	6	2	2	2
000111	6	6	4	0	2	10	2	2	2	2	6	6	8	0	6	2
001000	0	0	0	6	0	2	16	4	0	2	6	2	4	12	6	4
001001	10	4	2	6	0	2	6	2	4	0	8	6	4	4	2	4
001010	0	14	4	4	0	2	2	2	10	4	4	4	6	4	2	2
001011	4	6	2	0	2	2	12	8	2	2	2	6	8	2	0	6
001100	0	0	0	12	0	10	4	6	0	8	4	4	2	12	2	0
001101	12	0	2	10	6	4	4	2	4	2	6	0	2	6	0	4
001110	0	6	4	0	4	4	10	8	6	2	4	6	2	0	6	2
001111	2	2	2	2	6	2	6	2	10	4	8	2	6	4	4	2
010000	0	0	0	8	0	8	0	12	0	4	2	6	8	4	6	6
010001	6	2	6	4	6	2	6	4	6	6	4	2	4	0	6	0
010010	0	8	4	2	0	4	2	0	4	10	6	2	8	6	4	4
010011	6	6	12	0	12	2	0	6	6	2	0	4	0	2	4	2
010100	0	4	6	2	8	6	0	2	6	10	4	0	2	4	6	4
010101	2	2	6	6	4	4	2	6	2	6	8	4	4	0	4	4
010110	0	4	14	6	8	4	2	6	2	0	2	0	4	2	0	10
010111	2	6	8	0	0	2	0	2	2	6	0	8	8	2	12	6
011000	0	4	6	6	8	4	2	2	6	4	6	4	2	4	2	4
011001	2	6	0	2	4	4	4	6	4	8	6	4	2	2	6	4
011010	0	6	6	0	8	2	4	6	4	2	4	6	2	0	4	10
011011	0	4	10	2	4	4	2	6	6	6	2	2	6	6	2	2
011100	0	0	8	2	12	2	6	2	8	6	6	2	4	0	4	2
011101	2	4	0	6	8	6	0	2	6	8	6	0	2	4	0	10
011110	0	10	8	2	8	2	0	2	6	4	2	4	6	4	2	4
011111	0	6	6	8	6	4	2	4	4	2	2	0	2	4	2	12
100000	0	0	0	0	0	6	6	4	0	4	8	8	4	6	10	8
100001	2	8	6	8	4	4	6	6	8	4	0	4	0	2	2	0
100010	16	2	4	6	2	4	2	0	6	4	8	2	0	2	2	4
100011	0	4	0	4	4	6	10	4	2	2	6	2	4	6	6	4
100100	10	8	0	6	12	6	10	4	8	0	0	0	0	0	0	0
100101	0	2	4	2	0	4	4	0	4	0	10	10	4	10	6	4
100110	2	2	0	12	2	2	6	2	4	4	8	0	6	6	8	0
100111	8	4	0	8	2	4	2	4	0	6	2	4	4	8	2	6
101000	6	8	4	6	0	4	2	2	4	8	2	6	4	2	2	4
101001	2	4	4	0	8	8	6	8	6	4	0	4	4	4	2	0
101010	6	0	0	6	6	4	6	8	2	4	0	2	2	4	6	8
101011	12	0	4	0	0	4	2	2	2	6	10	6	10	2	4	0
101100	4	2	6	0	0	6	8	6	4	2	2	8	4	6	4	2

101101	6	2	2	6	6	4	4	2	6	2	4	8	4	2	4	2
101110	4	6	2	4	2	4	4	2	4	2	4	6	4	10	4	2
101111	10	0	4	8	0	6	6	2	0	4	4	2	6	2	2	8
110000	0	12	8	2	0	6	0	0	6	6	0	2	8	2	6	6
110001	2	6	10	4	2	2	2	4	6	0	2	6	0	2	4	12
110010	4	2	2	8	10	8	8	6	0	2	2	4	4	2	2	0
110011	4	2	2	2	6	0	4	0	10	6	6	4	0	4	8	6
110100	0	4	4	2	6	4	0	4	6	2	6	4	2	8	0	12
110101	6	12	4	2	4	2	2	4	8	2	2	0	6	4	4	2
110110	0	2	2	4	4	4	4	0	2	10	12	4	0	10	4	2
110111	10	2	2	6	14	2	2	6	2	0	4	6	2	0	4	2
111000	0	4	14	0	8	2	0	4	4	4	2	0	8	2	4	8
111001	2	4	8	0	6	2	0	6	2	6	4	2	8	6	2	6
111010	8	4	0	4	6	2	0	4	6	8	6	0	6	0	4	6
111011	0	4	6	6	2	2	2	14	0	12	0	4	2	2	8	0
111100	0	6	16	0	2	2	2	8	4	2	0	12	6	2	2	0
111101	0	6	2	2	2	6	8	2	4	2	6	2	6	2	4	10
111110	4	2	2	4	4	0	6	10	4	2	4	6	6	2	6	2
111111	0	4	6	6	4	8	4	0	4	8	4	0	4	8	2	2

Таблица 16

Анализ блока S_7 алгоритма шифрования DES для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
000001	0	0	0	2	0	4	4	14	0	12	4	6	2	6	6	4
000010	0	0	0	0	0	12	2	2	0	4	0	4	8	12	6	14
000011	8	2	12	2	6	8	6	0	6	4	4	2	2	0	0	2
000100	0	0	0	8	0	4	4	8	0	8	8	12	2	6	2	2
000101	6	0	0	2	8	0	8	4	0	2	6	0	10	6	6	6
000110	0	2	12	0	8	4	8	2	4	4	4	2	6	0	6	2
000111	4	6	4	12	0	4	2	0	0	14	2	6	4	0	0	6
001000	0	0	0	8	0	0	6	10	0	4	12	4	6	6	0	8
001001	10	8	4	8	6	2	2	0	2	6	8	2	0	6	0	0
001010	0	10	6	2	12	2	4	0	4	4	6	4	4	0	0	6
001011	0	2	2	2	4	8	6	4	4	0	4	2	6	4	2	14
001100	0	0	0	4	0	4	8	4	0	2	6	0	14	12	8	2
001101	6	6	2	4	2	6	4	6	6	4	8	8	0	2	0	0
001110	0	12	10	10	0	2	4	2	8	6	4	2	0	0	2	2

001111	2	0	0	0	6	8	8	0	6	2	4	6	8	0	6	8
010000	0	0	0	4	0	2	8	6	0	6	4	10	8	4	8	4
010001	6	10	10	4	4	2	0	4	4	0	2	8	4	2	2	2
010010	0	0	8	8	2	8	2	8	6	4	2	8	0	0	8	0
010011	4	4	2	2	8	6	0	2	2	2	0	4	6	8	14	0
010100	0	8	6	2	8	8	2	6	4	2	0	2	8	6	0	2
010101	4	4	8	2	4	0	4	10	8	2	4	4	4	2	0	4
010110	0	6	10	2	2	2	2	4	10	8	2	2	0	4	10	1
010111	8	2	4	2	6	4	0	6	4	4	2	2	0	4	8	8
011000	0	16	2	2	6	0	6	0	6	2	8	0	6	0	2	8
011001	0	8	0	2	4	4	10	4	8	0	6	4	2	6	2	4
011010	0	2	4	8	12	4	0	6	4	4	0	2	0	6	4	8
011011	0	6	2	6	4	2	4	4	6	4	8	4	2	0	10	2
011100	0	8	4	4	2	6	6	6	6	4	6	8	0	2	0	2
011101	4	4	4	0	0	2	4	2	4	2	2	4	10	10	8	4
011110	0	0	2	2	12	6	2	0	12	2	2	4	2	6	8	4
011111	2	2	10	14	2	4	2	4	4	6	0	2	4	8	0	0
100000	0	0	0	14	0	8	4	2	0	4	2	8	2	6	0	14
100001	4	2	6	2	12	2	4	0	6	4	10	2	4	2	2	2
100010	10	6	0	2	4	4	10	0	4	0	12	2	8	0	0	2
100011	0	6	2	2	2	4	6	10	0	4	8	2	2	6	0	10
100100	4	2	0	6	8	2	6	0	8	2	2	0	8	2	12	2
100101	2	0	2	16	2	4	6	4	6	8	2	4	0	6	0	2
100110	6	10	0	10	0	6	4	4	2	2	4	6	2	4	2	2
100111	4	0	2	0	2	2	14	0	4	6	6	2	12	2	4	4
101000	14	4	6	4	4	6	2	0	6	6	2	2	4	0	2	2
101001	2	2	0	2	0	8	4	2	4	6	4	4	6	4	12	4
101010	2	4	0	0	0	2	8	12	0	8	2	4	8	4	4	6
101011	16	6	2	4	6	10	2	2	2	2	2	2	4	2	2	0
101100	2	6	6	8	2	2	0	6	0	8	4	2	2	6	8	2
101101	6	2	4	2	8	8	2	8	2	4	4	0	2	0	8	4
101110	2	4	8	0	2	2	2	4	0	2	8	4	14	6	0	6
101111	2	2	2	8	0	2	2	6	4	6	8	8	6	2	0	6
110000	0	6	8	2	8	4	4	0	10	4	4	6	0	0	2	6
110001	0	8	4	0	6	2	2	6	6	0	0	2	6	4	8	10
110010	2	4	0	0	6	4	10	6	6	4	6	2	4	6	2	2
110011	0	16	6	8	2	0	2	2	4	2	8	4	0	4	6	0
110100	0	4	14	8	2	2	2	4	16	2	2	2	0	2	0	4
110101	0	6	0	0	10	8	2	2	6	0	0	8	6	4	4	8
110110	2	0	2	2	4	6	4	4	2	2	4	2	4	16	10	0

110111	6	6	6	8	4	2	4	4	4	0	6	8	2	4	0	0
111000	0	2	2	2	8	8	0	2	2	2	0	6	6	4	10	10
111001	4	4	16	8	0	6	4	2	4	4	2	6	0	2	2	0
111010	16	6	4	0	2	0	2	6	0	4	8	10	0	0	4	2
111011	2	0	0	2	0	4	4	4	2	6	2	6	6	12	12	2
111100	0	0	8	0	12	8	2	6	6	4	0	2	2	4	6	4
111101	2	4	12	2	2	2	0	4	6	10	2	6	4	2	0	6
111110	4	6	6	6	2	0	4	8	2	10	4	6	0	4	2	0
111111	14	0	0	0	8	0	6	8	4	2	0	0	4	8	4	6

Таблица 17

Анализ блока S_8 алгоритма шифрования DES для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
000001	0	0	0	6	0	16	10	0	0	0	6	0	14	6	2	4
000010	0	0	0	8	0	10	4	2	0	10	2	4	8	8	6	2
000011	6	0	2	8	2	6	4	0	6	6	6	2	2	0	8	6
000100	0	0	0	2	0	4	6	12	0	6	8	4	10	4	8	0
000101	4	10	6	0	0	2	6	0	4	10	4	6	8	2	0	2
000110	0	0	10	4	6	4	4	8	2	6	4	2	4	2	2	6
000111	6	2	8	2	8	10	6	6	4	2	0	4	0	0	0	6
001000	0	0	0	4	0	6	4	2	0	8	6	10	8	2	2	12
001001	8	4	0	6	0	4	4	6	2	4	6	2	12	2	0	4
001010	0	0	16	4	6	6	4	0	4	6	4	2	2	0	0	10
001011	2	8	0	6	2	6	0	4	4	10	0	2	10	2	6	2
001100	0	0	0	2	0	10	10	6	0	6	6	6	2	6	10	0
001101	6	0	4	10	2	0	8	6	2	2	6	10	2	2	2	2
001110	0	0	6	8	4	8	0	2	10	6	2	4	6	2	4	2
001111	8	0	4	2	2	4	2	2	2	6	4	6	0	2	14	6
010000	0	0	0	4	0	0	8	12	0	0	8	8	2	10	6	6
010001	0	6	4	6	2	2	6	6	4	6	4	6	0	4	4	4
010010	0	4	0	8	6	2	8	4	2	4	4	6	2	4	10	0
010011	4	2	2	6	8	6	2	2	14	2	2	4	2	2	2	4
010100	0	16	4	2	6	0	2	6	4	0	4	6	4	6	4	0
010101	0	10	6	0	6	0	2	8	2	2	0	8	2	6	6	6
010110	0	12	6	4	6	0	0	0	8	6	6	2	2	6	4	2
010111	0	6	8	0	6	2	4	6	6	0	2	6	4	4	2	8
011000	0	12	2	2	8	0	8	0	10	4	4	2	4	2	0	6

011001	6	4	8	0	8	0	4	2	0	0	12	2	4	6	2	6
011010	0	4	6	2	8	8	0	4	8	0	0	0	6	2	0	16
011011	2	4	8	10	2	4	2	8	2	4	8	2	0	2	4	2
011100	0	12	6	4	6	4	2	2	6	0	4	4	2	10	2	0
011101	8	6	0	0	10	0	0	8	10	4	2	2	2	8	4	0
011110	0	4	8	6	8	2	4	4	10	2	2	4	2	0	6	2
011111	4	2	4	2	6	2	4	0	2	6	2	2	2	16	8	2
100000	0	0	0	16	0	4	0	0	0	14	6	4	2	0	4	14
100001	0	0	2	10	2	8	10	0	0	6	6	0	10	2	2	6
100010	8	0	6	0	6	4	10	2	0	6	8	0	4	4	2	4
100011	4	8	0	6	0	4	8	6	2	2	10	4	8	0	0	2
100100	4	0	4	8	4	6	2	4	8	6	2	0	0	4	4	8
100101	0	4	6	8	2	8	8	0	4	2	4	4	2	2	6	4
100110	2	6	0	6	4	4	4	6	6	0	4	4	10	4	2	2
100111	6	6	0	0	2	2	6	2	4	4	6	10	2	6	2	6
101000	10	2	6	2	4	12	12	0	2	2	4	0	0	0	2	6
101001	4	0	0	14	2	10	4	2	8	6	4	0	4	2	2	2
101010	8	8	0	2	0	2	4	0	2	6	8	14	2	8	0	0
101011	2	2	0	0	4	2	10	4	6	2	4	0	6	4	8	10
101100	2	6	6	2	4	6	2	0	2	6	4	0	6	4	10	4
101101	8	0	4	4	6	2	0	0	6	8	2	4	6	4	4	6
101110	6	2	2	4	2	2	6	12	4	0	4	2	8	8	0	2
101111	8	12	4	6	6	4	2	2	2	2	4	2	2	4	0	4
110000	0	4	6	2	10	2	2	2	4	8	0	0	8	4	6	6
110001	4	6	8	0	4	6	0	4	4	6	10	2	2	4	4	0
110010	6	6	6	2	4	6	0	2	0	6	8	2	2	6	6	2
110011	6	6	4	2	4	0	0	10	2	2	0	6	8	4	0	10
110100	0	2	12	4	10	4	0	4	12	0	2	4	2	2	2	4
110101	6	4	4	0	10	0	0	4	10	0	0	4	2	8	8	4
110110	4	6	2	2	2	2	6	8	6	4	2	6	0	4	10	0
110111	2	2	8	2	4	4	4	2	6	2	0	10	6	10	2	0
111000	0	4	8	4	2	6	6	2	4	2	2	4	6	4	4	6
111001	4	4	4	8	0	6	0	6	4	8	2	2	2	4	8	2
111010	8	8	0	4	2	0	10	4	0	0	0	4	8	6	8	2
111011	8	2	6	4	4	4	4	0	6	4	4	6	4	4	4	0
111100	0	6	6	6	6	0	0	8	8	2	4	8	4	2	4	0
111101	2	2	8	0	10	0	2	12	0	4	0	8	0	2	6	8
111110	6	4	0	0	4	4	0	10	6	2	6	12	2	4	0	4
111111	0	6	6	0	4	4	6	10	0	6	8	2	0	4	8	0

ПРИЛОЖЕНИЕ 2

Таблицы линейного и дифференциального криптоанализа алгоритма шифрования Serpent

Таблица 18

Зависимость выходных битов линейного преобразования от соответствующих входных битов преобразования для алгоритма шифрования Serpent

Позиция выходного бита после линейного преобразования	Биты на входе линейного преобразования, сумма по модулю два которых образуют выходной бит линейного преобразования	Биты на входе линейного преобразования, сумма по модулю два которых образуют выходной бит инверсного линейного преобразования
0	16,52,56,70,83,94,105	53,55,72
1	72,114,125	1,5,20,90
2	2,9,15,30,76,84,126	15,102
3	36,90,103	3,31,90
4	20,56,60,74,87,98,109	57,59,76
5	1,76,118	5,9,24,94
6	2,6,13,19,34,80,88	19,106
7	40,94,107	7,35,94
8	24,60,64,78,91,103,113	61,63,80
9	5,80,122	9,13,28,98
10	6,10,17,23,38,84,92	23,110
11	44,98,111	11,39,98
12	28,64,68,82,95,106,117	65,67,84
13	9,84,126	13,17,32,102
14	10,14,21,27,42,88,96	27,114
15	48,102,115	1,3,15,20,43,120
16	32,68,72,86,99,110,121	69,71,88
17	2,13,88	17,21,36,106
18	14,18,25,31,46,92,100	1,31,118
19	52,106,119	5,7,19,24,47,106
20	36,72,76,90,103,114,125	73,75,92
21	6,17,92	21,25,40,110
22	18,22,29,35,50,96,104	5,35,122
23	56,110,123	9,11,23,28,51,110
24	1,40,76,80,94,107,118	77,79,96
25	10,21,96	25,29,44,114
26	22,26,33,39,54,100,108	9,39,126
27	60,114,127	13,15,27,32,55,114

28	5,44,80,84,98,111,122	81,83,100
29	14,25,100	1,29,33,48,118
30	26,30,37,43,58,104,112	2,13,43
31	3,118	1,17,19,31,36,59,118
32	9,48,84,88,102,115,126	85,87,104
33	18,29,104	5,33,37,52,122
34	30,34,41,47,62,108,116	6,17,47
35	7,122	5,21,23,35,40,63,122
36	2,13,52,88,92,106,119	89,91,108
37	22,33,108	9,37,41,56,126
38	34,38,45,51,66,112,120	10,21,51
39	11,126	9,25,27,39,44,67,126
40	6,17,56,92,96,110,123	93,95,112
41	26,37,112	2,13,41,45,60
42	38,42,49,55,70,116,124	14,25,55
43	2,15,76	2,13,29,31,43,48,71
44	10,21,60,96,100,114,127	97,99,116
45	30,41,116	6,17,45,49,64
46	0,42,46,53,59,74,120	18,29,59
47	6,19,80	6,17,33,35,47,52,75
48	3,14,25,100,104,118	101,103,120
49	34,45,120	10,21,49,53,68
50	4,46,50,57,63,78,124	22,33,63
51	10,23,84	10,21,37,39,51,56,79
52	7,18,29,104,108,122	105,107,124
53	38,49,124	14,25,53,57,72
54	0,8,50,54,61,67,82	26,37,67
55	14,27,88	14,25,41,43,55,60,83
56	11,22,33,108,112,126	0,109,111
57	0,42,53	18,29,57,61,76
58	4,12,54,58,65,71,86	30,41,71
59	18,31,92	18,29,45,47,59,64,87
60	2,15,26,37,76,112,116	4,113,115
61	4,46,57	22,33,61,65,80
62	8,16,58,62,69,75,90	34,45,75
63	22,35,96	22,33,49,51,63,68,91
64	6,19,30,41,80,116,120	8,117,119
65	8,50,61	26,37,65,69,84
66	12,20,62,66,73,79,94	38,49,79
67	26,39,100	26,37,53,55,67,72,95

68	10,23,34,45,84,120,124	12,121,123
69	12,54,65	30,41,69,73,88
70	16,24,66,70,77,83,98	42,53,83
71	30,43,104	30,41,57,59,71,76,99
72	0,14,27,38,49,88,124	16,125,127
73	16,58,69	34,45,73,77,92
74	20,28,70,74,81,87,102	46,57,87
75	34,47,108	34,45,61,63,75,80,103
76	0,4,18,31,42,53,92	1,3,20
77	20,62,73	38,49,77,81,96
78	24,32,74,78,85,91,106	50,61,91
79	38,51,112	38,49,65,67,79,84,107
80	4,8,22,35,46,57,96	5,7,24
81	24,66,77	42,53,81,85,100
82	28,36,78,82,89,95,110	54,65,95
83	42,55,116	42,53,69,71,83,88,111
84	8,12,26,39,50,61,100	9,11,28
85	28,70,81	46,57,85,89,104
86	32,40,82,86,93,99,114	58,69,99
87	46,59,120	46,57,73,75,87,92,115
88	12,16,30,43,54,65,104	13,15,32
89	32,74,85	50,61,89,93,108
90	36,90,103,118	62,73,103
91	50,63,124	50,61,77,79,91,96,119
92	16,20,34,47,58,69,108	17,19,36
93	36,78,89	54,65,93,97,112
94	40,94,107,122	66,77,107
95	0,54,67	54,65,81,83,95,100,123
96	20,24,38,51,62,73,112	21,23,40
97	40,82,93	58,69,97,101,116
98	44,98,111,126	70,81,111
99	4,58,71	58,69,85,87,99,104,127
100	24,28,42,55,66,77,116	25,27,44
101	44,86,97	62,73,101,105,120
102	2,48,102,115	74,85,115
103	8,62,75	3,62,73,89,91,103,108
104	28,32,46,59,70,81,120	29,31,48
105	48,90,101	66,77,105,109,124
106	6,52,106,119	78,89,119
107	12,66,79	7,66,77,93,95,107,112

108	32,36,50,63,74,85,124	33,35,52
109	52,94,105	0,70,81,109,113
110	10,56,110,123	82,93,123
111	16,70,83	11,70,81,97,99,111,116
112	0,36,40,54,67,78,89	37,39,56
113	56,98,109	4,74,85,113,117
114	14,60,114,127	86,97,127
115	20,74,87	15,74,85,101,103,115,120
116	4,40,44,58,71,82,93	41,43,60
117	60,102,113	8,78,89,117,121
118	3,18,72,114,118,125	3,90
119	24,78,91	19,78,89,105,107,119,124
120	8,44,48,62,75,86,97	45,47,64
121	64,106,117	12,82,93,121,125
122	1,7,22,76,118,122	7,94
123	28,82,95	0,23,82,93,109,111,123
124	12,48,52,66,79,90,101	49,51,68
125	68,110,121	1,16,86,97,125
126	5,11,26,80,122,126	11,98
127	32,86,99	4,27,86,97,113,115,127

Таблица 19

Анализ блока S_0 алгоритма шифрования Serpent для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	0	0	0	2	0	2	2	2	0	0	0	2	2	0	4	0
0010	0	0	0	0	0	0	0	0	0	2	2	0	4	2	2	4
0011	0	2	2	2	0	0	0	2	0	4	0	2	2	0	0	0
0100	0	0	0	0	0	0	0	0	0	4	4	0	0	4	4	0
0101	0	0	2	0	4	2	0	0	2	0	2	2	0	0	2	0
0110	0	2	2	4	0	2	2	4	0	0	0	0	0	0	0	0
0111	0	0	2	0	4	2	0	0	2	2	0	2	0	2	0	0
1000	0	0	0	2	0	2	2	2	0	0	0	2	2	4	0	0
1001	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2	0
1010	0	2	2	2	0	0	0	2	0	0	4	2	2	0	0	0
1011	0	2	2	0	0	2	2	0	0	0	0	0	4	0	0	4
1100	0	2	0	0	4	0	2	0	2	2	0	2	0	2	0	0
1101	0	0	0	4	0	0	0	4	4	0	0	0	0	0	0	4
1110	0	2	0	0	4	0	2	0	2	0	2	2	0	0	2	0
1111	0	2	2	0	0	2	2	0	4	0	0	0	0	0	0	4

Таблица 20

Анализ блока S_1 алгоритма шифрования Serpent для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	0	0	0	2	0	2	2	2	0	2	2	2	0	0	0	2
0010	0	0	0	2	0	2	0	0	0	2	2	2	2	2	0	2
0011	0	0	2	2	0	4	0	0	2	2	0	0	0	0	4	0
0100	0	0	0	0	0	0	4	4	0	0	0	0	4	4	0	0
0101	0	0	2	0	0	2	0	0	2	0	2	2	2	2	0	2
0110	0	0	2	0	0	2	2	2	2	0	2	2	0	0	0	2
0111	0	0	2	2	0	4	0	0	2	2	0	0	0	0	4	0
1000	0	0	0	0	0	0	2	2	0	0	0	0	2	2	4	4
1001	0	2	0	0	4	0	2	0	0	4	2	0	0	2	0	0
1010	0	2	0	4	4	0	0	2	0	0	2	0	2	0	0	0
1011	0	0	2	2	0	0	2	2	2	2	0	0	2	2	0	0
1100	0	4	2	2	0	0	0	0	2	2	0	4	0	0	0	0
1101	0	2	4	0	4	0	2	0	0	0	2	0	0	2	0	0
1110	0	2	0	0	4	0	0	2	4	0	2	0	2	0	0	0
1111	0	4	0	0	0	0	0	0	0	0	0	4	0	0	4	4

Таблица 21

Анализ блока S_2 алгоритма шифрования Serpent для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	0	0	0	0	0	2	0	2	0	0	2	2	2	0	4	2
0010	0	0	0	4	0	4	0	0	0	4	0	0	0	0	0	4
0011	0	4	2	0	0	0	2	0	0	2	0	0	2	0	2	2
0100	0	0	0	0	0	0	4	0	0	0	4	4	0	4	0	0
0101	0	4	0	2	2	2	2	0	2	0	0	0	2	0	0	0
0110	0	0	2	2	2	2	0	0	2	2	0	0	0	0	2	2
0111	0	0	0	0	4	2	0	2	0	0	2	2	2	0	0	2
1000	0	0	0	2	0	2	0	4	0	2	0	0	0	4	0	2
1001	0	0	0	2	0	0	0	2	4	2	2	2	2	0	0	0
1010	0	0	2	0	2	0	4	0	2	0	4	0	0	0	2	0
1011	0	4	0	0	2	0	2	0	2	2	0	0	2	0	0	2
1100	0	0	2	0	2	0	0	0	2	0	0	4	0	4	2	0
1101	0	4	2	2	0	2	2	0	0	0	0	0	2	0	2	0
1110	0	0	2	0	2	0	0	4	2	0	0	0	0	4	2	0
1111	0	0	4	2	0	0	0	2	0	2	2	2	2	0	0	0

Таблица 22

Анализ блока S_3 алгоритма шифрования Serpent для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	0	0	0	2	0	4	2	0	0	0	0	2	2	2	0	2
0010	0	0	0	0	0	2	0	2	0	2	4	2	0	0	0	4
0011	0	0	2	2	4	0	0	0	2	2	0	0	0	0	0	4
0100	0	0	0	0	0	0	4	4	0	0	2	2	2	2	0	0
0101	0	2	0	2	0	0	0	0	2	2	2	2	2	0	2	0
0110	0	2	0	2	0	0	2	2	4	0	0	0	2	0	0	2
0111	0	0	2	4	4	2	0	0	0	2	0	0	0	0	2	0
1000	0	0	0	2	0	0	2	0	0	2	0	0	2	4	4	0
1001	0	2	2	2	0	0	2	0	2	0	2	2	0	0	0	2
1010	0	0	2	0	2	0	2	2	0	4	0	2	2	0	0	0
1011	0	2	2	0	2	2	0	0	0	2	2	0	2	2	0	0
1100	0	2	0	0	2	0	2	2	4	0	2	0	0	0	2	0
1101	0	2	2	0	2	4	0	2	0	0	0	0	0	4	0	0
1110	0	4	2	0	0	2	0	0	0	0	0	2	0	2	2	2
1111	0	0	2	0	0	0	0	2	2	0	2	2	2	0	4	0

Таблица 23

Анализ блока S_4 алгоритма шифрования Serpent для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	0	0	0	0	0	0	0	4	0	0	2	2	2	2	4	0
0010	0	0	0	2	0	0	4	2	0	2	0	0	2	0	2	2
0011	0	2	2	0	2	0	0	2	2	2	2	2	0	0	0	0
0100	0	0	0	4	0	2	0	2	0	0	0	4	0	2	0	2
0101	0	2	0	2	0	0	0	0	2	2	0	0	4	2	2	0
0110	0	0	0	2	4	2	0	0	2	2	2	0	0	2	0	0
0111	0	0	2	2	2	0	0	2	2	0	2	0	0	0	0	4
1000	0	0	0	2	0	2	0	0	0	2	2	2	4	0	2	0
1001	0	2	4	0	2	0	2	2	0	2	0	0	0	2	0	0
1010	0	0	2	0	0	4	2	0	2	0	2	2	0	2	0	0
1011	0	4	0	0	0	2	0	2	0	0	0	4	0	2	0	2
1100	0	2	0	0	0	2	0	0	2	0	0	0	2	0	4	4
1101	0	2	4	0	2	2	2	0	0	2	0	0	0	0	0	2
1110	0	2	2	2	0	0	2	0	2	2	2	0	0	2	0	0
1111	0	0	0	0	4	0	4	0	2	0	2	0	2	0	2	0

Таблица 24

Анализ блока S_5 алгоритма шифрования Serpent для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	0	0	0	2	0	2	4	0	0	2	2	2	0	0	2	0
0010	0	0	0	0	0	0	2	2	0	0	2	2	0	4	4	0
0011	0	2	0	2	2	0	0	2	4	0	0	0	2	2	0	0
0100	0	0	0	0	0	2	0	2	0	4	0	4	0	2	0	2
0101	0	2	2	0	0	2	2	0	0	0	0	0	0	0	4	4
0110	0	2	2	2	0	0	4	2	0	2	0	0	0	0	2	0
0111	0	2	0	2	2	2	0	0	4	0	0	0	2	0	0	2
1000	0	0	0	2	0	0	2	0	0	2	0	0	4	2	2	2
1001	0	0	2	0	0	4	0	2	2	0	2	2	2	0	0	0
1010	0	2	2	2	2	0	0	0	2	0	2	2	0	2	0	0
1011	0	4	0	0	0	2	0	2	0	0	0	4	0	2	0	2
1100	0	0	2	2	4	2	0	2	0	2	2	0	0	0	0	0
1101	0	2	2	2	0	0	0	2	2	2	2	0	2	0	0	0
1110	0	0	2	0	2	0	0	0	2	2	2	0	0	2	0	4
11110	0	0	2	0	4	0	2	0	0	0	2	0	4	0	2	0

Таблица 25

Анализ блока S_6 алгоритма шифрования Serpent для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	0	0	0	0	0	2	0	2	0	2	2	0	2	2	4	0
0010	0	0	0	2	0	0	2	4	0	0	0	2	0	0	2	4
0011	0	2	4	2	0	0	0	0	2	2	0	0	0	2	2	0
0100	0	0	0	2	0	0	2	0	0	0	4	2	0	0	2	4
0101	0	2	0	4	2	2	0	2	0	0	2	0	0	2	0	0
0110	0	4	0	0	4	0	0	0	0	4	0	0	4	0	0	0
0111	0	0	4	2	2	0	0	0	2	0	0	0	2	2	2	0
1000	0	0	0	0	0	2	0	2	0	2	2	4	2	2	0	0
1001	0	2	0	2	2	0	2	0	0	2	0	2	2	0	2	0
1010	0	0	4	0	2	0	2	0	2	0	0	2	2	2	0	0
1011	0	0	0	2	0	4	2	0	4	0	0	2	0	0	2	0
1100	0	2	0	0	2	2	4	2	0	0	2	0	0	2	0	0
1101	0	2	0	0	2	0	0	0	0	2	4	0	2	0	0	4
1110	0	2	4	0	0	0	2	0	2	2	0	2	0	2	0	0
1111	0	0	0	0	0	4	0	4	4	0	0	0	0	0	0	4

Таблица 26

Анализ блока S_7 алгоритма шифрования Serpent для применения метода дифференциального криптоанализа

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	0	0	0	4	0	0	4	0	0	2	2	0	2	0	0	2
0010	0	0	0	2	0	2	0	0	0	0	0	2	4	2	4	0
0011	0	2	2	0	0	2	2	0	2	0	2	0	0	2	0	2
0100	0	0	0	0	0	2	0	2	0	2	0	2	2	2	2	2
0101	0	0	2	2	4	0	0	0	0	2	2	0	0	2	0	2
0110	0	2	4	2	0	2	2	0	2	2	0	0	0	0	0	0
0111	0	4	0	2	0	0	0	2	0	0	2	0	0	0	2	4
1000	0	0	0	2	0	0	2	4	0	2	2	2	0	2	0	0
1001	0	2	0	0	2	4	0	0	0	0	2	0	2	2	2	0
1010	0	0	0	0	2	0	0	2	4	0	0	4	0	2	2	0
1011	0	4	2	0	0	0	0	2	2	0	0	4	0	2	0	0
1100	0	2	0	0	0	0	2	0	2	2	2	0	2	0	4	0
1101	0	0	2	0	2	2	0	2	2	2	2	0	0	0	0	2
1110	0	0	4	2	2	2	2	0	0	0	0	2	0	0	0	2
1111	0	0	0	0	4	0	2	2	2	2	0	0	4	0	0	0

Таблица 27

Анализ блока S_0 алгоритма шифрования Serpent для применения метода линейного криптоанализа

	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	6	6	8	6	8	8	6	8	6	10	12	6	8	4	10
0010	10	6	8	8	10	10	12	8	6	6	12	8	6	10	8
0011	8	4	8	10	6	6	6	8	4	8	8	10	10	10	6
0100	8	8	8	8	8	8	8	8	8	8	8	12	4	4	4
0101	10	6	12	6	4	8	10	8	10	10	8	10	8	8	10
0110	6	10	8	8	6	6	12	4	6	6	8	8	10	6	8
0111	8	8	4	10	6	10	10	12	8	8	8	10	10	6	10
1000	6	6	8	10	8	8	10	8	6	10	4	6	4	8	10
1001	8	8	8	8	12	4	8	8	8	8	8	12	8	8	12
1010	12	8	8	10	6	6	6	8	8	4	8	6	6	6	10
1011	6	10	8	12	6	6	8	8	10	10	12	8	6	10	8
1100	6	10	12	10	8	12	6	8	6	6	8	10	8	8	10
1101	4	4	8	8	8	8	8	8	12	4	8	8	8	8	8
1110	8	8	12	10	10	6	10	12	8	8	8	6	10	8	8
1111	6	10	8	4	6	6	8	12	6	6	8	8	6	10	8

Таблица 28

Анализ блока S_1 алгоритма шифрования Serpent для применения метода линейного криптоанализа

	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	6	6	4	8	6	10	8	10	8	8	6	10	8	4	10
0010	6	10	8	8	6	10	8	6	12	8	6	10	8	12	10
0011	8	8	8	8	8	12	4	8	8	8	8	4	4	8	8
0100	8	6	10	8	8	10	6	6	6	8	4	10	10	8	4
0101	10	8	6	8	10	8	6	8	10	12	10	12	6	8	6
0110	6	8	10	8	6	4	6	8	6	12	6	8	6	8	10
0111	4	10	10	8	12	10	10	10	6	8	8	10	6	8	8
1000	8	8	8	8	12	8	4	8	8	8	8	8	12	8	12
1001	6	10	8	8	10	6	8	6	12	8	6	6	8	4	6
1010	10	10	12	8	6	10	8	6	8	8	10	10	8	4	10
1011	4	4	8	8	8	8	8	4	8	8	12	8	8	8	8
1100	8	6	10	4	8	6	6	10	10	4	8	10	6	8	8
1101	10	4	10	12	10	8	10	8	10	8	6	8	6	8	10
1110	10	8	6	4	10	8	10	4	6	8	6	8	6	8	10
1111	8	6	10	4	8	10	10	10	10	12	8	6	10	8	8

Таблица 29

Анализ блока S_2 алгоритма шифрования Serpent для применения метода линейного криптоанализа

	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	8	8	8	8	12	8	12	8	4	8	12	8	8	8	8
0010	8	10	10	10	6	8	12	8	8	6	6	10	6	12	8
0011	8	10	10	10	10	8	8	8	12	6	10	10	6	4	0
0100	8	10	6	6	6	12	8	8	8	6	10	6	6	8	4
0101	8	6	10	6	10	8	8	4	8	6	6	10	10	8	4
0110	8	8	4	8	12	8	8	8	8	8	4	8	4	8	8
0111	8	12	8	8	8	4	8	4	8	8	8	4	8	8	8
1000	8	6	10	8	8	10	6	6	6	4	8	6	6	8	12
1001	8	6	10	12	8	6	6	10	6	8	8	6	6	8	4
1010	12	8	8	6	6	6	10	10	6	6	6	8	8	4	8
1011	4	8	8	10	6	10	10	6	6	10	6	8	8	4	8
1100	8	8	8	10	10	10	10	10	10	6	6	4	12	8	8
1101	8	12	12	6	10	10	6	10	6	10	6	8	8	8	8
1110	12	6	10	8	8	10	10	6	10	12	8	6	6	8	8
1111	12	10	6	12	8	10	6	6	6	8	8	10	10	8	8

Таблица 30

Анализ блока S_3 алгоритма шифрования Serpent для применения метода линейного криптоанализа

	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	10	8	6	8	10	4	10	8	10	8	6	8	10	12	10
0010	6	10	8	8	10	10	4	6	8	8	10	10	8	12	10
0011	12	10	10	8	8	6	10	6	6	8	12	10	6	8	8
0100	8	10	10	10	10	8	8	8	12	10	6	10	6	8	4
0101	6	6	8	10	8	8	10	8	10	6	12	10	12	8	6
0110	6	8	6	6	8	10	12	10	8	10	8	12	6	8	10
0111	8	12	8	6	10	10	10	10	10	6	10	4	8	8	8
1000	8	8	12	10	6	10	10	6	10	6	6	8	8	8	12
1001	10	8	10	6	4	10	8	10	8	10	8	8	10	12	6
1010	10	10	8	6	8	8	6	8	10	10	8	10	12	4	10
1011	8	10	10	10	10	8	8	12	4	6	6	10	10	8	8
1100	8	6	10	8	12	10	10	6	6	12	8	6	10	8	8
1101	6	10	8	12	6	6	8	10	8	12	10	6	8	8	10
1110	10	4	10	8	10	8	6	12	10	8	10	8	6	8	10
1111	12	8	4	12	8	12	8	8	8	8	8	8	8	8	8

Таблица 31

Анализ блока S_4 алгоритма шифрования Serpent для применения метода линейного криптоанализа

	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	8	10	10	10	6	8	4	8	8	6	6	6	10	8	4
0010	8	10	6	8	8	10	6	8	8	10	6	12	12	6	10
0011	8	8	4	10	6	6	6	8	8	4	8	10	6	10	10
0100	8	8	8	10	10	6	6	10	6	10	6	8	4	4	8
0101	12	10	6	8	8	10	10	10	10	8	8	10	6	8	4
0110	4	6	6	6	10	8	8	10	10	8	4	8	8	10	6
0111	8	4	8	12	8	8	8	10	6	10	10	10	10	10	6
1000	8	8	12	10	6	10	10	8	8	8	4	10	6	10	10
1001	8	10	6	8	8	6	10	4	4	10	6	8	8	10	6
1010	8	10	10	6	10	8	4	8	8	10	10	10	6	12	8
1011	8	8	8	4	4	8	8	12	4	8	8	8	8	8	8
1100	4	8	8	8	8	12	8	6	6	6	10	10	6	6	6
1101	8	10	6	10	10	12	8	10	6	8	8	4	8	10	10
1110	8	6	6	8	4	10	6	6	10	12	8	6	6	8	8
1111	4	12	8	10	6	6	10	10	10	10	10	8	8	8	8

Таблица 32

Анализ блока S_5 алгоритма шифрования Serpent для применения метода линейного криптоанализа

	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	8	8	8	6	10	10	6	8	12	8	12	10	10	6	6
0010	8	8	8	6	10	6	10	10	10	10	10	8	4	8	12
0011	8	12	12	8	8	8	8	6	10	10	6	10	6	10	6
0100	8	6	10	10	10	8	12	8	8	10	6	10	10	4	8
0101	12	6	6	8	8	10	6	8	8	10	6	12	8	10	10
0110	4	10	10	8	8	10	6	10	6	8	8	10	10	8	12
0111	8	6	10	10	10	12	8	6	10	8	8	4	8	10	10
1000	8	8	8	8	8	4	4	6	10	10	6	6	10	6	10
1001	8	8	8	10	6	10	6	10	10	6	6	8	4	4	8
1010	8	8	8	6	10	10	6	8	4	12	8	6	6	6	6
1011	8	4	12	4	4	8	8	8	8	8	8	8	8	8	8
1100	4	6	6	6	10	8	8	10	10	8	4	8	8	10	6
1101	8	6	10	8	12	6	6	6	6	4	8	10	6	8	8
1110	8	10	6	4	8	10	10	4	8	6	6	8	8	6	10
1111	4	6	6	10	6	8	8	4	8	10	10	10	6	8	8

Таблица 33

Анализ блока S_6 алгоритма шифрования Serpent для применения метода линейного криптоанализа

	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	10	8	6	6	8	10	8	6	4	6	8	8	6	12	6
0010	8	8	8	8	8	8	8	8	8	4	12	8	8	4	4
0011	6	8	10	6	12	10	12	6	8	10	8	8	10	8	6
0100	6	8	6	6	8	10	4	8	6	8	6	10	12	6	8
0101	8	8	12	8	4	8	8	10	6	10	10	10	10	10	6
0110	10	4	6	10	8	10	8	8	10	8	10	6	12	10	8
0111	8	4	8	4	8	8	8	10	6	10	10	6	6	6	10
1000	10	8	10	6	8	6	8	10	8	6	4	4	10	8	6
1001	8	8	12	8	8	12	8	8	8	4	8	8	8	8	12
1010	10	8	10	10	12	10	4	10	8	10	8	8	6	8	6
1011	4	8	8	12	8	8	8	8	4	8	8	4	8	8	8
1100	8	8	8	8	12	4	8	10	6	6	10	10	10	10	10
1101	6	8	10	6	8	6	4	4	10	8	10	6	8	10	8
1110	4	4	8	8	8	8	8	10	10	6	6	10	6	10	6
1111	6	12	6	6	8	10	8	12	10	8	10	6	8	10	8

Таблица 34

Анализ блока S_7 алгоритма шифрования Serpent для применения метода линейного криптоанализа

	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0001	6	8	6	6	8	10	4	8	6	8	6	10	12	6	8
0010	6	10	8	8	6	10	8	8	6	6	12	8	6	6	4
0011	8	6	6	10	6	12	8	8	8	10	10	6	10	12	8
0100	8	10	10	6	10	8	4	8	8	10	10	10	6	12	8
0101	6	6	12	8	6	10	8	8	6	6	4	8	6	10	8
0110	6	8	6	6	8	6	8	12	10	8	6	6	8	10	4
0111	8	8	8	12	12	8	8	12	4	8	8	8	8	8	8
1000	8	8	8	10	10	10	10	6	10	10	6	12	8	8	4
1001	10	4	6	8	6	8	6	10	8	10	8	10	4	6	8
1010	10	10	4	6	8	8	10	6	4	8	8	8	8	8	8
1011	8	10	10	8	4	6	10	10	6	12	8	10	10	8	8
1100	4	10	6	8	8	10	10	10	10	8	8	10	6	10	12
1101	6	10	8	10	8	8	6	6	8	12	6	4	6	6	4
1110	6	4	6	12	6	4	6	6	8	6	8	10	8	10	8
1111	12	12	8	10	6	10	6	10	10	6	6	8	8	8	8

Таблица 35

Построение пятираундовой характеристики для алгоритма шифрования Serpent

№ раунда, R	Входы S_R^{ind} блока замены	Выходы S_R^{ind} блока замены	Вероятность	Измененные биты на входе линейного преобразования	Измененные биты на выходе линейного преобразования
2	$S_1^0 = 14$ $S_1^4 = 10$ $S_1^6 = 13$ $S_1^7 = 2$ $S_1^{18} = 12$ $S_1^{31} = 13$	$S_1^0 = 8$ $S_1^4 = 4$ $S_1^6 = 2$ $S_1^7 = 10$ $S_1^{18} = 1$ $S_1^{31} = 2$	$P = \frac{1}{2^{13}}$	3, 18, 25, 29, 31, 73, 125	16, 18, 29, 31
3	$S_2^4 = 5$ $S_2^7 = 10$	$S_2^4 = 4$ $S_2^7 = 10$	$P = \frac{1}{2^5}$	18, 33, 35	118
4	$S_3^{29} = 4$	$S_3^{29} = 10$	$P = \frac{1}{2^5}$	117, 119	12, 19, 36, 106, 121
5	$S_4^3 = 1$ $S_4^4 = 8$ $S_4^9 = 1$ $S_4^{26} = 4$ $S_4^{30} = 2$	$S_{43} = 14$ $S_{44} = 12$ $S_{49} = 10$ $S_{426} = 3$ $S_{430} = 3$	$P = \frac{1}{2^{13}}$	13, 14, 15, 18, 19, 37, 39, 104, 105, 120, 121	0, 2, 16, 17, 18, 26, 36, 38, 41, 43, 46, 47, 52, 53, 58, 59, 66, 67, 68, 69, 71, 76, 87, 104, 107, 109, 118, 124, 125

6	$S_5^0 = 5$ $S_5^4 = 7$ $S_5^6 = 4$ $S_5^9 = 5$ $S_5^{10} = 10$ $S_5^{11} = 12$ $S_5^{13} = 3$ $S_5^{14} = 12$ $S_5^{16} = 12$ $S_5^{17} = 11$ $S_5^{19} = 1$ $S_5^{21} = 8$ $S_5^{26} = 9$ $S_5^{27} = 2$ $S_5^{29} = 4$ $S_5^{31} = 3$	$S_5^0 = 14$ $S_5^4 = 8$ $S_5^6 = 11$ $S_5^9 = 14$ $S_5^{10} = 1$ $S_5^{11} = 4$ $S_5^{13} = 8$ $S_5^{14} = 4$ $S_5^{16} = 4$ $S_5^{17} = 1$ $S_5^{19} = 6$ $S_5^{21} = 12$ $S_5^{26} = 5$ $S_5^{27} = 13$ $S_5^{29} = 9$ $S_5^{31} = 8$	$P = \frac{1}{2^{33}}$		

Таблица 36

Построение шестираундовой характеристики для алгоритма шифрования Serpent

№ раунда, R	Входы S_R^{ind} блока замены	Выходы S_R^{ind} блока замены	Вероятность	Измененные биты на входе линейного преобразования	Измененные биты на выходе линейного преобразования
5	$S_4^0 = 1$ $S_4^1 = 6$ $S_4^2 = 2$ $S_4^4 = 10$ $S_4^7 = 4$ $S_4^8 = 1$ $S_4^{11} = 5$ $S_4^{13} = 5$ $S_4^{16} = 1$ $S_4^{17} = 1$ $S_4^{21} = 1$ $S_4^{23} = 5$ $S_4^{24} = 4$ $S_4^{26} = 4$ $S_4^{27} = 6$ $S_4^{30} = 4$	$S_4^0 = 10$ $S_4^1 = 9$ $S_4^2 = 6$ $S_4^4 = 5$ $S_4^7 = 3$ $S_4^8 = 13$ $S_4^{11} = 12$ $S_4^{13} = 12$ $S_4^{16} = 10$ $S_4^{17} = 14$ $S_4^{21} = 7$ $S_4^{23} = 12$ $S_4^{24} = 3$ $S_4^{26} = 11$ $S_4^{27} = 4$ $S_4^{30} = 3$	$P = \frac{1}{2^{36}}$	1, 3, 4, 7, 9, 10, 16, 18, 28, 29, 32, 34, 35, 46, 47, 54, 55, 65, 67, 69, 70, 71, 84, 85, 86, 94, 95, 96, 97, 104, 105, 107, 110, 120, 121	5, 6, 22, 23, 24, 26, 28, 30, 31, 33, 35, 58, 59, 64, 66, 71, 82, 83, 104
6	$S_5^1 = 6$ $S_5^5 = 12$ $S_5^6 = 5$ $S_5^7 = 13$ $S_5^8 = 10$ $S_5^{14} = 12$ $S_5^{16} = 5$ $S_5^{17} = 8$ $S_5^{20} = 12$ $S_5^{26} = 1$	$S_5^1 = 1$ $S_5^5 = 4$ $S_5^6 = 2$ $S_5^7 = 10$ $S_5^8 = 10$ $S_5^{14} = 2$ $S_5^{16} = 2$ $S_5^{17} = 12$ $S_5^{20} = 2$ $S_5^{26} = 3$	$P = \frac{1}{2^{28}}$	4, 22, 25, 29, 31, 33, 35, 57, 65, 70, 71, 81, 104, 105	29, 30, 42, 58, 59, 69, 70, 71, 109, 111, 122

7	$S_6^7 = 6$ $S_6^{10} = 4$ $S_6^{14} = 12$ $S_6^{17} = 14$ $S_6^{27} = 10$ $S_6^{30} = 4$	$S_6^7 = 4$ $S_6^{10} = 10$ $S_6^{14} = 1$ $S_6^{17} = 1$ $S_6^{27} = 2$ $S_6^{30} = 10$	$P = \frac{1}{2^{14}}$	30, 41, 43, 56, 68, 109, 121, 123	0, 2, 12
8	$S_7^0 = 5$ $S_7^3 = 1$	$S_7^0 = 4$ $S_7^3 = 10$	$P = \frac{1}{2^5}$	2, 13, 15	102
9	$S_0^{25} = 4$	$S_0^{25} = 10$	$P = \frac{1}{2^2}$	101, 103	3, 20, 90, 105, 124
10	$S_1^0 = 8$ $S_1^5 = 1$ $S_1^{22} = 4$ $S_1^{26} = 2$ $S_1^{31} = 1$	$S_1^0 = 14$ $S_1^5 = 3$ $S_1^{22} = 6$ $S_1^{26} = 3$ $S_1^{31} = 3$	$P = \frac{1}{2^{13}}$		

Таблица 37

Построение пятираундовой характеристики для алгоритма шифрования Serpent

№ раунда, R	Входы S_R^{ind} блока замены	Выходы S_R^{ind} блока замены	Вероятность	Измененные биты на входе линейного преобразования	Измененные биты на выходе линейного преобразования
6	$S_5^1 = 11$ $S_5^5 = 12$ $S_5^6 = 6$ $S_5^7 = 1$ $S_5^8 = 1$ $S_5^{14} = 6$ $S_5^{16} = 6$ $S_5^{17} = 8$ $S_5^{20} = 6$ $S_5^{26} = 1$	$S_5^1 = 1$ $S_5^5 = 4$ $S_5^6 = 2$ $S_5^7 = 10$ $S_5^8 = 10$ $S_5^{14} = 2$ $S_5^{16} = 2$ $S_5^{17} = 12$ $S_5^{20} = 2$ $S_5^{26} = 3$	$P = \frac{1}{2^{27}}$	4, 22, 25, 29, 31, 33, 35, 57, 65, 70, 71, 81, 105, 105	29, 30, 42, 58, 59, 69, 70, 71, 109, 111, 122
7	$S_6^7 = 6$ $S_6^{10} = 4$ $S_6^{14} = 12$ $S_6^{17} = 14$ $S_6^{27} = 10$ $S_6^{30} = 4$	$S_6^7 = 4$ $S_6^{10} = 10$ $S_6^{14} = 1$ $S_6^{17} = 1$ $S_6^{27} = 2$ $S_6^{30} = 10$	$P = \frac{1}{2^{14}}$	30, 41, 43, 56, 68, 109, 121, 123	0, 2, 12
8	$S_7^0 = 5$ $S_7^3 = 1$	$S_7^0 = 4$ $S_7^3 = 10$	$P = \frac{1}{2^5}$	2, 13, 15	102
9	$S_0^{25} = 4$	$S_0^{25} = 10$	$P = \frac{1}{2^2}$	101, 103	3, 20, 90, 105, 124

10	$S_1^0 = 8$ $S_1 5 = 1$ $S_1^{22} = 4$ $S_1^{26} = 2$ $S_1^{31} = 1$	$S_1^0 = 14$ $S_1^5 = 3$ $S_1^{22} = 6$ $S_1^{26} = 3$ $S_1^{31} = 3$	$P = \frac{1}{2^{13}}$		

Таблица 38

Построение шестираундовой характеристики для алгоритма шифрования Serpent

№ раунда, R	Входы S_R^{ind} блока замены	Выходы S_R^{ind} блока замены	Вероятность	Измененные биты на входе линейного преобразования	Измененные биты на выходе линейного преобразования
2	$S_1^0 = 14$ $S_1^2 = 3$ $S_1^5 = 12$ $S_1^9 = 14$ $S_1^{13} = 9$ $S_1^{16} = 13$ $S_1^{18} = 12$ $S_1^{21} = 1$ $S_1^{22} = 13$ $S_1^{26} = 4$ $S_1^{29} = 8$ $S_1^{31} = 13$	$S_1^0 = 8$ $S_1^2 = 5$ $S_1^5 = 11$ $S_1^9 = 8$ $S_1^{13} = 4$ $S_1^{16} = 2$ $S_1^{18} = 11$ $S_1^{21} = 10$ $S_1^{22} = 2$ $S_1^{26} = 12$ $S_1^{29} = 14$ $S_1^{31} = 2$	$P = \frac{1}{2^{25}}$	3, 8, 10, 20, 21, 23, 39, 54, 65, 72, 73, 75, 85, 87, 89, 94, 95, 117, 118, 119, 125	5, 7, 16, 65, 67, 80, 82, 89, 90, 92, 93, 94, 95, 108, 110, 122
3	$S_2^1 = 10$ $S_2^4 = 1$ $S_2^{16} = 10$ $S_2^{20} = 5$ $S_2^{22} = 6$ $S_2^{23} = 15$ $S_2^{27} = 5$ $S_2^{30} = 4$	$S_2^1 = 4$ $S_2^4 = 10$ $S_2^{16} = 8$ $S_2^{20} = 4$ $S_2^{22} = 2$ $S_2^{23} = 10$ $S_2^{27} = 4$ $S_2^{30} = 10$	$P = \frac{1}{2^{23}}$	6, 17, 19, 67, 82, 89, 93, 95, 110, 121, 123	93, 95, 106
4	$S_3^{23} = 10$ $S_3^{26} = 4$	$S_3^{23} = 4$ $S_3^{26} = 10$	$P = \frac{1}{2^6}$	94, 105, 107	66
5	$S_4^{16} = 4$	$S_4^{16} = 3$	$P = \frac{1}{2^2}$	64, 65	8, 12, 58, 69, 88, 121
6	$S_5^2 = 1$ $S_5^3 = 1$ $S_5^{14} = 4$ $S_5^{17} = 2$ $S_5^{22} = 1$ $S_5^{30} = 2$	$S_5^2 = 10$ $S_5^3 = 6$ $S_5^{14} = 5$ $S_5^{17} = 10$ $S_5^{22} = 3$ $S_5^{30} = 10$	$P = \frac{1}{2^{17}}$	9, 11, 13, 14, 56, 58, 69, 71, 88, 89, 121, 123	0, 2, 4, 13, 16, 18, 29, 30, 39, 48, 56, 82, 93, 112, 113, 114, 125, 126

7	$S_6^0 = 5$ $S_6^1 = 1$ $S_6^3 = 2$ $S_6^4 = 5$ $S_6^7 = 6$ $S_6^9 = 8$ $S_6^{12} = 1$ $S_6^{14} = 1$ $S_6^{20} = 4$ $S_6^{23} = 2$ $S_6^{28} = 7$ $S_6^{31} = 6$	$S_6^0 = 3$ $S_6^1 = 14$ $S_6^3 = 7$ $S_6^4 = 3$ $S_6^7 = 1$ $S_6^9 = 11$ $S_6^{12} = 14$ $S_6^{14} = 14$ $S_6^{20} = 10$ $S_6^{23} = 7$ $S_6^{28} = 2$ $S_6^{31} = 1$	$P = \frac{1}{2^{24}}$		

ЛИТЕРАТУРА

[1] *Шнайер Б.* Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. — М.: Триумф, 2002.

[2] *Столлингс В.* Криптография и защита сетей: принципы и практика. 2-е изд. / Пер. с англ. — М.: Вильямс, 2001.

[3] *Чмора А. Л.* Современная прикладная криптография. 2-е изд. — М.: Гелиос АРВ, 2002.

[4] *Грушо А. А., Тимонина Е. Е., Применко Э. А.* Анализ и синтез криптоалгоритмов. — Йошкар-Ола: Изд-во МФ МОСУ, 2000.

[5] *Зензин О. С., Иванов М. А.* Стандарт криптографической защиты — AES. Конечные поля. — М.: КУДИЦ-ОБРАЗ, 2002.

[6] *Matsui M.* Linear Cryptanalysis Method for DES Cipher, Advances in Cryptology // EUROCRYPT'93, Springer-Verlag, 1998, p. 386.

[7] *Biham E., Shamir A.* Differential Cryptanalysis of the Full 16-round DES // Crypto'92, Springer-Verlag, 1998, p. 487.

[8] *Biham E., Shamir A.* Differential Cryptanalysis of DES-like Cryptosystems, Extended Abstract // Crypto'90, Springer-Verlag, 1998, p. 2.

[9] http://zeus.sai.msu.ru:7000/internet/infsecure — *Беляев А. В.* Методы и средства защиты информации.

[10] http://cins.ict.nsc.ru/citonod/My/crypto — *Ростовцев А. Г., Михайлова Н. В.* Методы криптоанализа классических шифров.

[11] http://kiwibyrd.chat.ru/aes/aes2.htm — *Киви Б.* О процессе принятия AES.

[12] http://www.bre.ru/security/12050.html — Основные тенденции развития открытой криптографии.

[13] http://csrc.nist.gov/encryption/aes — *Daemen J., Rijmen V.* AES Proposial: Rijndael.

[14] http://www1.cs.columbia.edu/~dcook/candexam/Y_17_serpent.pdf — *Anderson R., Biham E., Knudsen L.* Serpent: A Proposal for the Advanced Encryption Standard.

[15] http://electronica.finestreet.ru — *Петренко С., Беляев А., Панасенко С.* Европейский криптопроект NESSIE.

[16] http://planeta.terra.com.br/informatica/paulobarreto — *Preneel B.* The NESSIE Project: Towards New Cryptographic Algorithms.

[17] *Matsui M.* The First Experimental Cryptanalysis of the Data Encryption Standard // Crypto'94, Springer-Verlag, 1998, p. 1.

[18] *Heys H. M.* A Tutorial on Linear and Differential Cryptanalysis.

[19] *Langford S. K., Hellman M. E.* Differential-linear cryptanalysis // Crypto'94, Springer-Verlag, 1998, p. 17.

[20] http://www.research.ibm.com/security/mars.html — *Safford D., Zunic N. et al.* MARS — a Candidate cipher for AES.

[21] http://www.cs.berkeley.edu/~daw/papers/advslide-cS$_{00}$.ps, — *Birukov A., Wagner D.* Slide Attacks // Eurocrypt'00, p. 589.

[22] citeseer.nj.nec.com/572086.html — *Biham E., Dunkelman O., Keller N.* Enhancing Differential-Linear Cryptanalysis.

[23] citeseer.nj.nec.com/knudsen96improved.html — *Knudsen L., Meier W.* Improved Attacks on RC5.

[24] citeseer.nj.nec.com/kaliski98security.html — *Kaliski B. S., Jr., Yin Y. L.* On the Security of the RC5 Encryption Algorithm.

[25] citeseer.nj.nec.com/borst99linear.html — *Borst J., Preneel B., Vandewalle J.* Linear Cryptanalysis of RC5 and RC6.

[26] *Kaliski B. S., Jr., Yin Y. L.* On Differential and Linear Cryptanalysis of the RC5 Encryption Algorithm // Crypto'95, Springer-Verlag, 1998, p. 171.

[27] *Ben-Aroya I., Biham E.* Differential Cryptanalysis of Lucifer // Crypto'93, Springer-Verlag, 1998, p. 187.

[28] citeseer.nj.nec.com/biham01linear.html — *Biham E., Dunkelman O., Keller N.* Linear Cryptanalysis of Reduced Round Serpent.

[29] *Birukov A., Biham E., Shamir A.* Cryptanalysis of Skipjack Reduced to 31 Rounds using Impossible Differentials.

[30] www.schneier.com/paper-serpent-aes.pdf — *Kohno T., Kelsey J., Schneier B.* Preliminary Cryptanalysis of Reduced-Round Serpent.

[31] http://www.cs.hku.hk/research/techreps/document/TR-2000-04.pdf — *Wang X. Y., Hui L. C. K. et al.* The Differential Cryptanalysis of an AES Finalist — Serpent.

[32] *Gilbert H., Chauvaud P.* A Chosen Plaintext Attack of the 16-round Khufu Cryptosystem // Eurocrypt'94, p. 359–368.

[33] http://www.schneier.com/paper-rijndael.pdf — *Ferguson N., Kelsey J. et al.* Improved Cryptanalysis of Rijndael.

[34] http://caislab.icu.ac.kr/paper/2001/kkj/icisc2001.ps — *Cheon J., Kim M. et al.* Improved Impossible Differential Cryptanalysis of Rijndael and Crypton // Proceedings of the 4th International Conference Seoul on Information Security and Cryptology, p. 39–49, 2001.

[35] http://www.cs.colorado.edu/~jrblack/class/csci7000/s05/project/sasas-birsham.pdf — *Birukov A., Shamir A.* Structural Cryptanalysis of SASAS.

[36] http://csrc.nist.gov/CryptoToolkit/aes/round2/conf3/papers/35-ebiham.pdf — *Biham E., Keller N.* Cryptanalysis of Reduced Variants of Rijndael.

[37] http://www.csberkeley.edu~daw/papers/advslide-ec00.ps — *Birukov A., Wagner D.* Advanced Slide Attacks.

[38] *Biham E., Shamir A.* Differential Cryptanalysis of Snefru, Khafre, REDOC-II, LOKI and Lucifer // Extended Abstract, Crypto'91, Springer-Verlag, 1998, p. 156.

[39] *Shimizu A., Miyaguchi S.* Fast Encipherment Algorithm FEAL // Eurocrypt '87, Springer-Verlag, 1998, p. 267.

[40] *Biham E., Shamir A.* Differential Cryptanalysis of Feal and N-Hash // Crypto ´87, Springer-Verlag, 1998, p. 1.

[41] *Birukov A.* Thesis.

[42] *Бабенко Л. К.* Введение в специальность. Организация и технология защиты информации. — Таганрог, 1999.

[43] *Зензин О. С., Иванов М. А.* Стандарт криптографичекой защиты AES. Конечные поля. — М.: КУДИЦ-ОБРАЗ, 2002.

[44] http://www.secinf.net/uplarticle/4/rc6.pdf — *Riverst R.* The RC6 Block Cipher.

ОГЛАВЛЕНИЕ

Научное издание

Людмила Климентьевна Бабенко
Евгения Александровна Ищукова
Современные алгоритмы блочного шифрования и методы их анализа

Заведующая редакцией *Т. А. Денисова*
Корректор *Е. Н. Клитина*
Компьютерная верстка *С. Н. Авилкина*

Издательство «Гелиос АРВ»
Издательская лицензия ЛР № 066255
107140, г. Москва, Верхняя Красносельская ул., 16.
Тел./факс: (495) 264-44-39, e-mail: info@gelios-arv.ru
Адрес в Internet: http://www.gelios-arv.ru

Формат 70×100/16. Бумага офсет 65 гр., 23,5 п. л.
Печать офсетная. Тираж 1500 экз.
Заказ № 16
Отпечатано с готовых диапозитивов в типографии ГП «Облиздат».
248640, г Калуга, пл. Старый торг, 5.

Учебное пособие

Посвящено алгоритмам блочного шифрования: принципам их построения и анализа. Рассматриваются действующие стандарты, а также многие другие общеизвестные криптографические алгоритмы, в том числе и финалисты конкурса AES. Излагаются способы проведения атак на эти алгоритмы с помощью таких методов, как линейный и дифференциальный криптоанализ. Описан подход к применению нового метода криптоанализа — линейно-дифференциального. Представлены виды криптоанализа на основе слайдовой атаки. В приложениях приведены таблицы с результатами анализа наиболее известных алгоритмов шифрования. Большая часть учебного пособия посвящена практическим вопросам изучения атак: приведено пять лабораторных работ по описанным методам криптоанализа, которые представлены на сайте кафедры БИТ ТРТУ http://bit.tsure.ru, а также целый ряд задач для самостоятельного решения.

Гелиос РВ