Раздел IV. Методы и средства криптографии и стеганографии

УДК 681.03.245

Л.К. Бабенко, Е.А. Ищукова

АНАЛИЗ АЛГОРИТМА ГОСТ 28147-89: ПОИСК СЛАБЫХ БЛОКОВ*

Рассмотрено влияние S-блоков замены на устойчивость алгоритма шифрования ГОСТ 28147-89 (далее по тексту ГОСТ) к методу линейного криптоанализа. Представлен детальный, программно ориентированный универсальный алгоритм поиска слабых блоков замены по отношению к методу линейного криптоанализа. Показана возможность построения эффективных линейных статистических аналогов для упрощенной версии алгоритма ГОСТ, содержащего слабые S-блоки. Данное исследование направлено на предотвращение использования слабых блоков замены для тех алгоритмов блочного шифрования, в которых данные элементы не являются фиксированными. Работа разработанного алгоритма поиска слабых блоков была опробована на примере анализа блоков замены для алгоритма шифрования ГОСТ 28147-89. Применение разработанного алгоритма позволяет без труда обнаружить большое число ослабленных блоков замены, использование которых может значительно ослабить стойкость используемого алгоритма шифрования. Использование данного алгоритма может быть полезно для тех, кто пользуется данным шифром, но не владеет навыками криптоанализа.

Симметричные алгоритмы шифрования; анализ стойкости; сеть Фейстеля; ГОСТ 28147-89; раундовые ключи шифрования; блок замены; линейный криптоанализ.

L.K. Babenko, E.A. Ischukova

ANALYSIS OF ALGORITHM GOST 28147-89: RESEARCH OF WEAK S-BOXES

This work is devoted to finding the influence of S-Boxes to resistance of GOST 28147-89 algorithm (GOST) against linear cryptanalysis. The universal algorithm for searching particular layouts of S-Boxes, which are vulnerable to linear cryptanalysis is presented. The possibility of building of efficient linear statistical analogs for simplified GOST with weak S-Boxes has been shown. This research is aimed to ensuring that certain arbitrary S-Box layouts are not weak when they are not fixed. Applicability of the presented method was tested by analyzing S-Boxes used in GOST. Application of the designed method made it possible to discover a number of weak S-Boxes, which make the overall cryptographic strength of GOST much lower.

GOST; S-Box; secret key; linear cryptanalysis; probability.

Метод линейного криптоанализа, впервые предложенный М. Матсуи для анализа алгоритма DES [1], базируется на составлении линейных аналогов, которые с некоторой вероятностью описывают работу криптоалгоритма. После появления работы [1] большинство существовавших на тот момент алгоритмов шифрования были подвергнуты анализу с использованием данного метода. Исследования показали, что метод линейного криптоанализа является универсальным, т.е. может

* Работа выполнена при поддержке грантов РФФИ №12-07-33007_мол_а_вед, № 12-07-00037-а.

быть применен к анализу большинства известных симметричных криптосистем. Именно поэтому вновь создаваемые алгоритмы шифрования в первую очередь тестируются на устойчивость к данному виду анализа.

Наше исследование направлено на изучение стойкости к методу линейного криптоанализа алгоритма ГОСТ, определенного в качестве государственного стандарта в Российской Федерации. До сих пор в открытой печати имеется сравнительно мало информации о возможных уязвимостях данного шифра. Отличительной чертой алгоритма ГОСТ является использование в его структуре нефиксированных блоков замены. Предполагается, что при любом заполнении S-блоков тридцати двух раундов шифрования будет достаточно для того, чтобы противостоять таким мощным методам анализа, как линейный и дифференциальный криптоанализ. Долгое время считалось, что если оставлять S-блоки в секрете, то их можно рассматривать как дополнительный ключевой материал [2]. Однако в работе [3] предложен метод, применение которого позволяет достаточно просто восстановить значения S-блоков, используемых для шифрования данных. В настоящей работе предлагается рассмотреть влияние блоков замены на устойчивость алгоритма ГОСТ к методу линейного криптоанализа. Для этого предлагается разработанный нами универсальный алгоритм поиска блоков, использование которых может значительно ослабить стойкость алгоритма ГОСТ. Исследование преследует две цели. Во-первых, необходимо получить инструмент для быстрого определения полного списка слабых блоков по отношению в линейному криптоанализу для исследования их влияния на стойкость ГОСТ. Во-вторых, при использовании нашего алгоритма можно легко получить полный список блоков, не рекомендованных к использованию для алгоритма ГОСТ, что может быть полезно для тех, кто пользуется данным шифром, но не владеет навыками криптоанализа.

С использованием полученных нами слабых блоков, будет показано как на практике можно получить эффективные уравнения для применения метода линейного криптоанализа.

Метод линейного криптоанализа впервые предложен в начале 90-х годов XX века японским ученым М. Матсуи (Matsui). В работе [1] М. Матсуи показал, как можно осуществить атаку на алгоритм шифрования DES, сократив сложность анализа до 2^{47}. Существенным недостатком метода стала необходимость иметь в наличии большой объем данных, зашифрованных на одном и том же секретном ключе, что делало метод малопригодным для практического применения к вскрытию шифра. Однако, если предположить, что к аналитику в руки попал шифрованный текст, содержащий важные сведения, а также некий черный ящик (устройство или программа), который позволяет выполнить любое число текстов, зашифрованных с помощью известного алгоритма шифрования на секретном ключе, не раскрывая при этом самого ключа, то применение метода линейного криптоанализа становится вполне реальным. Многие алгоритмы шифрования, известные на момент опубликования работы [1], в последствии были проверены на устойчивость к этому методу и не все из них оказались достаточно стойкими и, как следствие, потребовали доработки.

Любой алгоритм шифрования в самом общем виде можно представить как некоторую функцию E, зависящую от входного сообщения X, секретного ключа K и возвращающую шифрованное сообщение Y:

$$Y = E(X, K). \quad (1)$$

Зная само преобразование E и входное сообщение X, нельзя однозначно сказать каким будет выходное сообщение Y. В данном случае нелинейность функции зависит от внутренних механизмов преобразования E и секретного ключа K.М. Матсуи показал, что существует возможность представить функцию шифрования в виде системы

уравнений, которые выполняются с некоторой вероятностью **p**. При этом для успешного проведения анализа вероятность уравнений **p** должна быть как можно дальше удалена от значения 0,5.

Так как уравнения, получаемые в ходе анализа криптоалгоритма, являются вероятностными, то их называют линейными статистическими аналогами. Линейным статистическим аналогом нелинейной функции шифрования (1) называется величина Q, равная сумме по модулю два скалярных произведений входного вектора X, выходного вектора Y и вектора секретного ключа K соответственно с двоичными векторами α, β и γ, имеющими хотя бы одну координату равную единице:

$$Q = (X, \alpha) \oplus (Y, \beta) \oplus (K, \gamma) \quad (2)$$

в том случае, если вероятность того, что Q=0 отлична от 0,5 (P(Q=0)≠0,5).

В отличие от дифференциального криптоанализа, в котором большое значение вероятности гарантировало успех атаки, в линейном криптоанализе успех анализа может быть обеспечен как уравнениями с очень большой вероятностью, так и уравнениями с очень маленькой вероятностью. Для того, чтобы понять, какое из возможных уравнений лучше всего использовать для анализа используют отклонения. Отклонением линейного статистического аналога называют величину

$$\eta = |1 - 2p|, \quad (3)$$

где p – вероятность, с которой выполняется линейный аналог.

Отклонение определяет эффективность линейного статистического аналога. Чем отклонение больше, тем выше вероятность успешного проведения анализа. Фактически отклонение показывает насколько вероятность статистического аналога отдалена от значения p = 0,5.

Для успешного применения метода линейного криптоанализа необходимо решить следующие задачи. Найти максимально эффективные (или близкие к ним) статистические линейные аналоги. При нахождении аналогов обратить внимание на то, что в них должно быть задействовано как можно больше битов искомого секретного ключа K. Получить статистические данные: необходимый объем пар текстов (открытый – закрытый текст), зашифрованных с помощью анализируемого алгоритма на одном и том же секретном ключе. Определить ключ (или некоторые биты ключа) путем анализа статистических данных с помощью линейных аналогов.

Первый шаг анализа заключается в нахождении эффективных статистических аналогов. Для алгоритмов шифрования, в которых все блоки заранее известны, этот шаг можно выполнить единожды, основываясь на анализе линейных свойств всех криптографических элементов шифра. Для алгоритма ГОСТ такой подход неприемлем, в связи с использованием нефиксированных S-блоков. То есть данный этап анализа каждый раз при смене используемых S-блоков необходимо повторять сначала. В результате анализа должна быть получена система уравнений, выполняющихся с некоторыми вероятностями. Левая часть уравнений должна содержать в себе сумму битов входного и выходного сообщения, правая часть уравнения – биты секретного ключа. Если первый шаг анализа является чисто теоретическим и полностью зависит от структуры алгоритма, то второй шаг – является исключительно практической частью, которая заключается в анализе известных пар открытый-закрытый текст с помощью полученной ранее системы статистических аналогов. Для этого используется следующий алгоритм.

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.

На сегодняшний день нет достаточно подробных исследований стойкости алгоритма шифрования ГОСТ к методу линенйого криптоанализа. Однако в книге Б. Шнайера [4] говорится о том, что за счет большого числа раундов шифрования линейный криптоанализ практически не применим к полнораундовому альгоритму ГОСТ.

Описание алгоритма ГОСТ. Алгоритм шифрования ГОСТ 28147-89 является государственным стандартом Российской Федерации. Его использование обязательно для шифрования данных в государственных организациях РФ. Алгоритм ГОСТ является симметричным блочным шифром, построенным по схеме Фейстеля (Feistel). На вход алгоритма поступает 64-битовый блок данных, который под воздействием 256-битового ключа преобразуется в 64-битовый блок шифрованных данных. В каждом раунде правая часть шифруемого сообщения поступает на вход функции F, где преобразуется с использованием трех криптографических операций: сложения данных с раундовым подключом по модулю 2^{32}, замена данных с использованием S-блоков, циклический сдвиг влево на 11 позиций. Выход функции F складывается по модулю 2 с левой частью шифруемого сообщения, после чего правая и левая части меняются местами. Алгоритм содержит 32 раунда, в последнем раунде шифрования правая и левая части местами не меняются. Структура алгоритма ГОСТ приведена на рис. 1.

В алгоритме шифрования ГОСТ используется 8 S-блоков, которые преобразуют 4 бита на входе в S-блок в 4 бита на выходе. В отличие от большинства алгоритмов шифрования ГОСТ не имеет фиксированных блоков замены и может использовать любые варианты блоков.

Секретный ключ шифрования содержит 256 битов и представляется в виде последовательности из восьми 32-битовых слов: К1, К2, К3, К4, К5, К6, К7, К8. В каждом раунде шифрования в качестве раундового подключа используется одно из этих 32-битовых слов. При определения раундового подключа руководствуются следующим принципом: с 1 по 24 раунды используются последовательно К1, К2, К3, К4, К5, К6, К7, К8, К1, К2 и т.д. С 25 по 32 раунды: К8, К7, К6, К5, К4, К3, К2, К1.

Таким образом, получается, что в первом и последнем раундах используется один и тот же раундовый подключ К1.

Несмотря на то, что алгоритм ГОСТ является «старичком» в криптографии и насчитывает более 20 лет, в настоящее время можно найти сравнительно мало литературы, посвященной вопросам анализа данного алгоритма шифрования. Отчасти это связано с тем, что первое время алгоритм был засекречен и стал доступен широкой общественности только после 1994 г. Кроме того, до появления работы [3] считалось, что S-блоки могут служить дополнительным ключом, что существенно затрудняло проведение анализа.

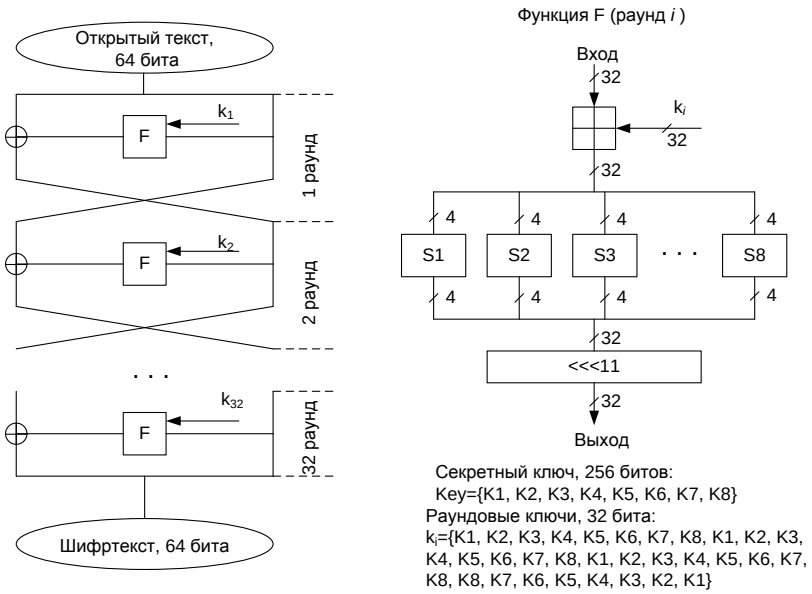

Рис. 1. Алгоритм шифрования ГОСТ

Существенную сложность в проведение анализа вносит тот факт, что S-блоки замены не являются фиксированным элементом и могут иметь любое заполнение. Это накладывает ограничение на применение методов анализа, основанных на изучении их свойств, таких как, например, методы линейного, дифференциального и алгебраического анализа. Каждый раз при использовании новых таблиц замены анализ необходимо будет начинать с самого начала, в отличие, например от алгоритма DES, в котором таблицы замены были строго определены стандартом.

Очень часто изучение слабых свойств алгоритма шифрования начинают с исследования его упрощенных моделей [5]. Для алгоритма ГОСТ помимо использования нефиксированных S-блоков выделяют еще два момента, которые затрудняют проведение анализа алгоритма: это использование операции целочисленного сложения по модулю 2^{32}, которая используется для сложения данных с секретным ключом, и использование раундовых подключей в обратном порядке для последних 8 раундов шифрования. В [5] приводятся результаты анализа упрощенной версии алгоритма ГОСТ. Для алгоритма GOST-H изменен порядок следования раундовых подключей в последних 8 раундах, то есть они также используются последовательно с К1 по К8. Показано, что модифицированный алгоритм гораздо слабее исходного и обладает рядом слабых ключей. Вариант упрощенной версии алгоритма ГОСТ, в которой операция целочисленного сложения заменена на операцию сложения по модулю два и количество раундов сокращено до 20, рассмотрен известными специалистами в области криптографии А. Бирюковм и Д. Вагнером [7].

Алгоритм поиска слабых блоков. Алгоритм ГОСТ содержит 8 S-блоков. При этом сами блоки не являются фиксированными. То есть теоретически считается, что могут быть использованы блоки замены, сформированные случайным образом. Криптографическую стойкость алгоритму должно обеспечить достаточно большое число раундов шифрования (32 раунда). Долгое время считалось, что если держать S-блоки в секрете, то можно рассматривать их как дополнительный ключевой материал. Однако в работе [5] было показано, что S-блоки, используемые в алгоритме шифрования, можно достаточно просто восстановить.

В связи с этим разумно рассмотреть слабые S-блоки для алгоритма ГОСТ и оценить степень сложности атаки на основе линейного криптоанализа при их использовании.

Итак, какие же блоки необходимо считать слабыми и как их получить? Для того, чтобы ответить на этот вопрос, необходимо вспомнить о двух вещах. Во-первых, итоговая вероятность аналога будет получена, исходя из значения вероятностей для каждого S-блока, вовлеченного в процесс построения аналога. Так как в алгоритма ГОСТ используется 32 раунда шифрования, то для получения максимальной вероятности аналога желательно производить его построение так, чтобы в каждом раунде было задействовано минимальное количество S-блоков, в идеале это должен быть всего один S-блок. Во-вторых, необходимо вспомнить, что в качестве перемешивания битов в каждом раунде используется операция циклического сдвига влево на 11 позиций. Таким образом получается, что биты на выходе одного блока поступают входы двух блоков в следующем раунде. Кроме того, необходимо отметить, что слабые блоки могут быть организованы таким образом, что будут позволять конструировать линейные аналоги, вероятность которых равна нулю или единице. Таким образом, можно предположить, что слабыми блоками будут являться те блоки, которые будут позволять строить линейные аналоги для значений входов и выходов, содержащих минимальное количество единиц, то есть для значений 1, 2, 4, 8. Также слабыми будут являться те блоки, которые для любого входного значения позволят построить аналог, вероятность которого будет равна нулю или единице.

Определив критерии для отбора слабых S-блоков, мы задумались о том, каким образом нам лучше всего попробовать получить эти самые блоки. Так как на вход S-блока алгоритма ГОСТ поступает 4 бита, которые заменяются также на 4 бита, то всего возможно 16! различных комбинаций таких блоков, что соответствует примерно $2^{44,2}$. Перебор такого количества блоков весьма трудоемок и длителен, даже при использовании распределенных многопроцессорных вычислений. Вариант случайной генерации блока и его последующей оценки нами также был отвергнут, как не позволяющий получить полный набор необходимых нам блоков за приемлемое время. Вместо этого нами был разработан новый универсальный алгоритм поиска слабых блоков по отношению к линейному анализу. Рассмотрим его более детально.

Для начала вспомним, что для алгоритма ГОСТ на вход каждого блока замены поступает часть преобразованного входного сообщения X, сложенная по модулю два с частью секретного ключа. Таким образом, получается, что биты сообщения X и биты ключа K неразрывно связаны, а значит к ним всегда должен быть применен один и тот же вектор, например вектор α. В связи с этим мы можем преобразовать выражение (1) для линейного статистического аналога к виду:

$$Q = \left(X \oplus K, \alpha\right) \oplus \left(Y, \beta\right). \quad (4)$$

По определенным ранее условиям, нам необходимо, чтобы при составлении аналога было задействовано как можно меньше блоков. Поэтому, мы предлагаем рассматривать такой вариант, когда при составлении аналога будет задействован всего один бит входного сообщения X (и соответствующий ему бит ключа K. Далее мы будем опускать значение битов ключа K, полагая что оно неразрывно связано со значением используемого бита значения X) и всего один бит сообщения Y. И дальше рассматривать все возможные комбинации Xi $\oplus$Yj для i=1,..4 и для j=1,..,4. Так как нам необходимо рассмотреть все возможные комбинации блоков замен, а мы заранее не знаем какая пара значений вход-выход позволит нам получить искомый результат, то для каждого входа мы составляем таблицу размерностью 16x24. Всего у нас будет 16 таких таблиц, по одной таблице для каждого входа. В каждую таблицу заносим биты одного входа, все возможные биты выхода, а также все рассматриваемые комбинации Xi $\oplus$ Yj для значений i=1,..4; j=1,..,4. В табл. 1 приведен пример построения такой таблицы для первого рассматриваемого входа X = 0.

Таблица 1

Пример построения таблицы для входа X=0

X1	X2	X3	X4	Y1	Y2	Y3	Y4	X1 ⊕ Y1	X1 ⊕ Y2	X1 ⊕ Y3	X1 ⊕ Y4	X2 ⊕ Y1	X2 ⊕ Y2	X2 ⊕ Y3	X2 ⊕ Y4	X3 ⊕ Y1	X3 ⊕ Y2	X3 ⊕ Y3	X3 ⊕ Y4	X4 ⊕ Y1	X4 ⊕ Y2	X4 ⊕ Y3	X4 ⊕ Y4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
0	0	0	0	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1
0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
0	0	0	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
0	0	0	0	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1
0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
0	0	0	0	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1
0	0	0	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

После того как составлено 16 таблиц, можно приступить к их обработке. Мы будем оперировать следующими значениями.

N_{min} – значение, которое соответствует искомому минимальному количеству линейных аналогов, выполняемых с вероятностью 0 или 1.

$Comb_i$ – очередное возможные сочетание N_{min} элементов из 16 возможных (взятое из ячеек в столбцах под номерами 9 – 24). Всего таких рассматриваемых сочетаний будет

$$C = \frac{16!}{N_{min}!(16 - N_{min})!}$$

Qvalue – значение, которое отражает чему должно быть равно значение Q для каждой из позиции в сочетании $Comb_i$

Для того, чтобы стало немного понятнее, рассмотрим как будут соотноситься эти три значения между собой на примере табл. 1. Пусть $N_{min} = 5$; Qvalue = 12. Например, для первой строки табл. 1, соответствующей паре (X,Y)=(0000, 0000) ни одно из сочетаний ячеек не даст значения Qvalue = 01100. То же самое справедливо и для последней строки табл. 1, соответствующей паре (X,Y)=(1111, 1111). Для всех остальных таблиц возможно получить сочетания ячеек в строке такое, которое будет соответствовать значению Qvalue = 01100. На рис. 2 для примера представлен один из вариантов (но не единственно возможный!!!) комбинаций ячеек для строк 2 и 8.

Если во всех других таблицах сочетание данных ячеек таблицы совпадет со значением Qvalue, и при этом не произойдет перекрытие значений выходов, то значения X-Y соответствующей строки в каждой из 16 таблиц будут отражать работу искомого блока замены. Таким образом, для нахождения всех возможных вариантов заполнения блоков замены, отвечающих условиям слабого блока, необходимо для каждого значения Qvalue перебрать все возможные комбинации ячеек Comb.

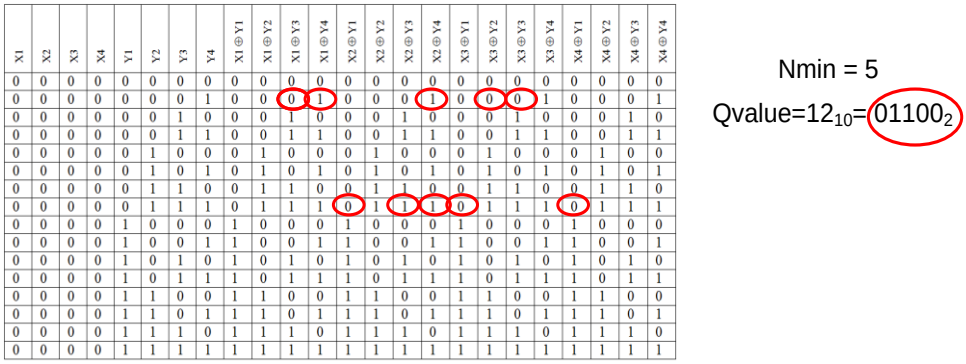

Рис. 2. Анализ таблиц

В общем виде вышеописанный алгоритм сводится к выполнению следующих шагов:

1. Инициализация значения N_{min};
2. Qvalue = 0;
3. i=0;
3. Определение очередного значения $Comb_i$;
4. Определение строк в каждой из 16 таблиц, для которых ячейки комбинации строк совпадают со значением Qvalue;
5. Если в каждой таблице есть строка, для которой $Comb_i$ = Qvalue и при этом нет перекрытия значений выходов (то есть номера выбранных строк в каждой из 16 анализируемых таблиц разные), то искомая таблица найдена;
6. i=i+1;
7. если ($i \le C$), то переход к пункту 3;
8. Qvalue=QValue+1;
9. если ($Qvalue < 2^{Nmin}$), то переход к пункту 3.

При желании данный алгоритм можно расширить и включить для проведения анализа в таблицу не только комбинации аналогов, для которых используется всего один бит входа и один бит выхода, но и другие комбинации из двух и более битов входа и выхода. Это повлечет за собой увеличение размера таблицы, изменение параметров анализа, и как следствие увеличение времени проведения анализа.

Результаты экспериментов. Алгоритм, представленный выше, был реализован и опробован на практике. Полный анализ выполняется в течение нескольких минут (все исследования проводились на процессоре Intel Celeron M CPU 530 1.73 GHz, RAM 1007Mb). В ходе эксперимента варьировался критерий слабого блока по минимальному количеству экстремумов, которое необходимо найти. Результаты экспериментов представлены в табл. 2.

Таблица 2

Результаты экспериментов

Номер эксперимента	Минимальное количество экстремумов	Количество найденных слабых блоков
1	2	1288
2	3	768
3	4	384
4	5	0

Как видно из табл. 2, при увеличении числа искомых экстремумов количество найденных блоков уменьшается. Однако при этом уменьшается и стойкость обнаруживаемых блоков замены.

Для более наглядного примера, рассмотрим одну из таблиц замены (табл. 3), полученную в результате использования предложенного нами алгоритма анализа.

Таблица 3

Слабый блок замены, определенный в результате работы предложенного алгоритма

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

Блок замены в табл. 3 имеет таблицу анализа, отражающую возможность построения линейных аналогов для линейного криптоанализа, предствленную в табл. 4.

Таблица 4

Таблица анализа полученного S-блока

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
1	8	8	8	8	8	8	8	0	8	8	8	8	8	8	8
2	8	8	8	0	8	8	8	8	8	8	8	8	8	8	8
3	8	8	8	8	8	8	8	8	8	8	8	16	8	8	8
4	8	0	8	8	8	8	8	8	8	8	8	8	8	8	8
5	8	8	8	8	8	8	8	8	8	16	8	8	8	8	8
6	8	8	8	8	8	16	8	8	8	8	8	8	8	8	8
7	8	8	8	8	8	8	8	8	8	8	8	8	8	0	8
8	0	8	8	8	8	8	8	8	8	8	8	8	8	8	8
9	8	8	8	8	8	8	8	8	16	8	8	8	8	8	8
10	8	8	8	8	16	8	8	8	8	8	8	8	8	8	8
11	8	8	8	8	8	8	8	8	8	8	8	8	0	8	8
12	8	8	16	8	8	8	8	8	8	8	8	8	8	8	8
13	8	8	8	8	8	8	8	8	8	8	0	8	8	8	8
14	8	8	8	8	8	8	0	8	8	8	8	8	8	8	8
15	8	8	8	8	8	8	8	8	8	8	8	8	8	8	16

Из табл. 4 можно видеть, что в найденной таблице экстремумов гораздо больше минимального числа. Это связано с тем, что рассматриваются только те входы и выходы, которые содержат в своем составе одну единицу, то есть это значения 1, 2, 4, 8. Использование такого блока анализа крайне не желательно, так как будет заметно ослаблять свойства используемого алгоритма шифрования.

Выводы. В результате исследования получен универсальный инструмент для быстрого определения полного списка слабых блоков по отношению в линейному криптоанализу, который может быть, например, использован при разработке новых алгоритмов шифрования. При использовании данного алгоритма можно легко получить полный список блоков, не рекомендованных к использованию для алгоритма ГОСТ, что может быть полезно для тех, кто пользуется данным шифром, но не владеет навыками криптоанализа.

Дальнейшее исследование в данной области будет направлено на решение проблемы быстрого построения линейных аналога при использовании различных наборов S-блоков, на комплексную оценку устойчивости алгоритма шифрования ГОСТ и других блочных шифров, малоизученных по отношению к линейному криптоанализу.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Matsui M.* Linear Cryptanalysis Method for DES Cipher, Advances in Cryptology – EUROCRYPT'93, Springer-Verlag, 1998. – 386 p.
2. *Popov V., Kurepkin I., Leontiev S.* Additional Cryptographic Algorithms for Use with GOST 28147-89, GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 Algorithms. – January 2006. - http://www.ietf.org/rfc/rfc4357.
3. *Saarien M.-J.* A Chosen Key Attack Against the Secret S-boxes of GOST // http://www.m.-js.com – Helsinki University of Technology, Finland.
4. *Schneier B.* Applied Cryptography, Protocols, Algorithms and Source Code in C (Second Edition). John Wiley and Sons, Inc. 1996.
5. *Oreku G.S., Li J., Pazynyuk T., Mtenzi F.J.* Modified S-box to Archive Accelerated GOST // http://paper.ijcsns.org, International Journal of Computer Science and Network Security. – June 2007. – Vol. 7, № 6.
6. *Biham E., Shamir A.* Differential Cryptanalysis of DES-like Cryptosystems, Extended Abstract, Crypto'90, Springer-Velgar, 1998. – P. 2.
7. *Birukov A., Wagner D.* Advanced Slide Attacks // http://citeseer.ist.psu.edu.
8. *Babenko L.K., Ishchukova E.A., Maro E.A.* Theory and Practice of Cryptography Solutions for Secure Information Sysmems. GOST Encryption Algorithm and Approaches to its Analysis. IGI Global book series Advances in Information Security, Privacy, and Ethics (AISPE) Book Series, USA, 2013. – P. 34-62.
9. *Babenko L.K., Ishchukova E.A., Maro E.A.* Research about Strength of GOST 28147-89 Encryption Algorithm. – Proceedings of the 5th international conference on Security of information and networks (SIN 2012). – ACM, New York, NY, USA, 2012. – P. 138-142.
10. *Babenko L.K., Ishchukova E.A.* Differential Analysis of GOST Encryption Algorithm. – Proceedings of the 3rd International Conference of Security of Information and Networks (SIN 2010). – ACM, New York, NY, USA, 2010. – P. 149-157.

Статью рекомендовал к опубликованию д.т.н., профессор Я.Е. Ромм.

Бабенко Людмила Климентьевна – Южный федеральный университет; e-mail: blk@fib.tsure.ru; 347928, г. Таганрог, ул. Чехова, 2, корпус "И"; тел.: 88634312018; кафедра безопасности информационных технологий; профессор.

Ищукова Евгения Александровна – e-mail: jekky82@mail.ru; тел.: 88634371905; кафедра безопасности информационных технологий; доцент.

Babenko Lyudmila Klimentevna – Southern Federal University; e-mail: blk@fib.tsure.ru; Block "I", 2, Chehov street, Taganrog, 347928, Russia; phone: +78634312018; the department of security of information technologies; professor.

Ischukova Evgeniya Aleksandrovna – e-mail: jekky82@mail.ru; phone: +78634371905; the department of security of information technologies; associate professor.

УДК 681.03.245

Л.К. Бабенко, Е.А. Ищукова

ИСПОЛЬЗОВАНИЕ СЛАБЫХ БЛОКОВ ЗАМЕНЫ ДЛЯ ЛИНЕЙНОГО КРИПТОАНАЛИЗА БЛОЧНЫХ ШИФРОВ*

Работа является продолжением исследований влияния используемых слабых S-блоков на возможность проведения атаки с помощью метода линейного криптоанализа для алгоритма шифрования ГОСТ 28147-89. Ранее авторами статьи разработан универсальный алгоритм поиска блоков замены, ослабленных по отношению к методу линейного криптоа-

* Работа выполнена при поддержке грантов РФФИ №12-07-31120_мол_а, №12-07-33007_мол_а_вед, № 12-07-00037-а.