УДК 004.432

СИММЕТРИЧНЫЙ БЛОЧНЫЙ RSB-32 КРИПТОГРАФИЧЕСКИЙ АЛГОРИТМ С ДИНАМИЧЕСКИМ УПРАВЛЕНИЕМ ПАРАМЕТРАМИ ШИФРОВАНИЯ

Белецкий А.Я., Белецкий А.А.
Национальный авиационный университет, Киев

Рассмотрен итерационный блочный криптоалгоритм с переменным размером общего ключа шифрования, кратным 32-м битам. Шифрование текста выполняется за s *шагов* $(s \geq 1)$. *В качестве криптографических примитивов используются: стохастическая прокрутка блока, двунаправленное скользящее кодирование, нелинейная подстановка и стохастическая перестановка элементов блока. Криптопреобразование каждого 256-битного блока находится под управлением индивидуального блочного ключа, изменяющегося в зависимости от шифруемых данных и секретного раундового ключа.*

Ключевые слова: *криптографический алгоритм, примитив, параметры шифрования*

Введение и постановка задачи

Современные методы защиты информации в компьютерных сетях (шифрование) представляют собой математические преобразования, в которых сообщения рассматриваются как числа или алгебраические элементы в некотором пространстве [1,2]. С позиции теории сигналов и процессов зашифрование исходного текста (коррелированного, избыточного, сжимаемого) состоит в его отбеливании, т.е. обращении в некоррелированную последовательность символов (элементов) практически несжимаемой шифрограммы с плотностью распределения вероятностей элементов выходного алфавита максимально близкой к равномерной.

Несмотря на многообразие существующих блочных криптографических систем, все еще сохраняет актуальность разработка новых более гибких алгоритмов шифрования. В данной статье предлагается симметричный блочный криптоалгоритм, названный ***RSB-32*** шифром. Аббревиатура ***RSB*** происходит от ключевых слов ***R****ound,* ***S****tep,* ***B****lok* – подчеркивая тем самым, что основными для криптоалгоритма являются *раундовые* преобразования (***R***), разбитые на определенное число *шагов* (***S***), а процесс преобразования осуществляется над *блоками* открытого или закрытого текстов (***B***). Отличительная особенность ***RSB*** алгоритма состоит в том, что в нем используется оригинальный криптографический примитив типа *скользящего кодирования* [3], который обеспечивает не только глубокое перемешивание открытого текста, но и участвует в формировании *блочных* раундовых ключей для очередных шифруемых блоков. Тем самым все преобразования, выполняемые криптоалгоритмом, становятся зависимыми не только от секретного ключа, но и от шифруемых данных, т.е. относятся к классу *управляемых криптопреобразований* [4, 5].

До настоящего времени управляемые криптографические примитивы еще не получили сколько-нибудь заметного применения в шифраторах. Мы можем лишь отметить такие шифры, как ***MARS*** или ***RC6*** [1], вошедших в состав финалистов международного конкурса по разработке нового стандарта криптографической защиты ***AES*** (Advanced Encryption Standard) [6]. В этих шифрах используется операция циклического сдвига блоков на число разрядов, изменяющихся в зависимости от секретного ключа и шифруемых данных. Теоретические исследования показали [7, 8], что применение оператора стохастической прокрутки блока, зависящего от преобразуемых данных, является эффективным средством противодействия важнейшим типам атак, к которым относятся линейный и дифференциальный анализ, что считается достойным качеством криптосистем.

Общая характеристика алгоритма

Предлагаемый ***RSB*** алгоритм шифрования закладывает основу создания принципиально новой технологии симметричной блочной криптографической защиты информации, не имеющей аналогов в мировой практике. Реализация алгоритма позволяет существенно повысить криптостойкость систем шифрования по сравнению с уже существующими продуктами и в то же время сохраняет высокую скорость криптопреобразования.

Достижение первого заявляемого качества (криптостойкости) базируется на таких предпосылках. В сложившейся мировой практике построения симметричных блочных криптографических алгоритмов в пределах раунда все блоки шифруемого текста подвергаются одинаковым преобразованиям. С одной стороны это создает возможность параллельной обработки информации, что обеспечивает повышение скорости шифрования. Вместе с тем, такая технология шифрования облегчает работу криптоаналитиков. В самом деле, если в открытом тексте присутствуют одинаковые блоки, то одинаковыми будут также эти блоки и после зашифрования, что приводит к снижению криптостойкости алгоритма.

Отмеченный недостаток классических блочных шифраторов устраняется в ***RSB*** алгоритме за счет применения двунаправленного скользящего кодирования, посредством которого каждый шифруемый блок текста становится управляемым своим индивидуальным *блочным раундовым ключом*, зависящим не только от *базового* раундового ключа, но и всего текста, предшествующего преобразуемому блоку. Тем самым интуитивно становится понятным, что ***RSB*** технология значительно усложняет работу криптоаналитика (что эквивалентно повышению криптостойкости шифра), поскольку опыт, приобретенный на этапе взлома одной шифрограммы, может оказаться практически бесполезным для взлома другой шифрограммы (за счет различия исходных текстов).

Обобщенная структурная схема ***RSB*** алгоритма представлена на рис. 1.

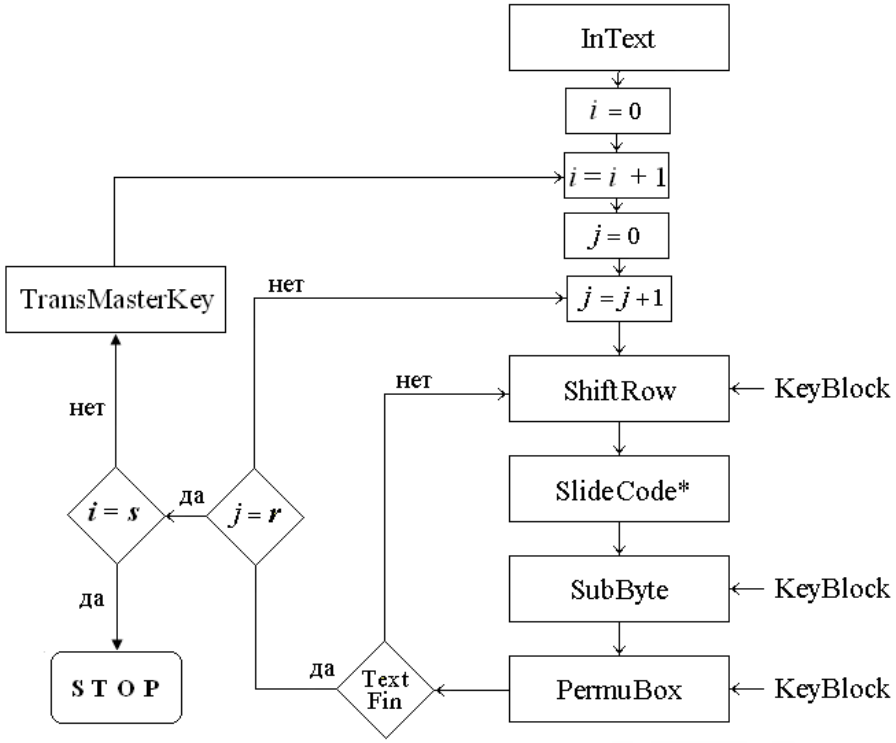

*Рисунок 1. Обобщенная структурная схема **RSB**-криптоалгоритма.*

Данная схема отображает процесс преобразования текстов как для алгоритма зашифрования, так и расшифрования (естественно – с учетом выполнения требований обратимости преобразований). Поэтому в дальнейшем мы ограничимся в основном пояснением организации процесса зашифрования открытого текста.

Структурная схема алгоритма, показанная на рис.1, содержит два вложенных цикла. Внешним циклом (параметр ***i***) задается шаг шифрования (от одного до ***s***), а внутренним (параметр ***j***) осуществляется ***r***-раундовое шифрование (под которым понимается как зашифрование, так и расшифрование) текста. Каждый раунд зашифрования предполагает выполнение следующих четырех криптографических примитивов над блоками открытого текста:

- - стохастическая прокрутка блока (ShiftRow);
- - скользящее кодирование (SlideCode);
- - нелинейная подстановка байтов блока (SubByte);
- - стохастическая перестановка 16-битных слов блока (PermuBox).

Перед началом процедуры зашифрования входной открытый текст (InText) разбивается на блоки, размером в *N* бит. Если последний блок оказался меньше выбранного размера, то он дополняется (пробелами) до полного блока. Назовем такой текст *расширенным файлом*. Объем расширенного файла в ходе зашифрования не меняется, поэтому объем шифротекста всегда будет кратным размеру блока.

Основные параметры ***RSB*** алгоритма таковы:

- Размер блока: *N* = 256 бит;
- Длина раундового ключа – 32 бита;
- Длина общего (шагового) ключа: ***r*** * 32, ***r*** = 1,2,…;
- Число шагов шифрования: ***s*** = 1,2,…;
- Общее число раундов шифрования: ***r*** * ***s***;
- Размер элементов скользящего кодирования – 32 бита;
- Размер элементов нелинейной замены: 8 бит.

Развернутая структурная схема ***RSB*** алгоритма в режиме зашифрования приведена на рис. 2, в котором использованы такие обозначения:

- ***RC*** (***R****ound* ***C****ode*) – операции зашифрования текста раундовым ключом (подключом общего ключа);
- $\boldsymbol{RK}_{ij}$ (***R****ound* ***K****ey*) – *j*-й раундовый ключ на *i*-м шаге зашифрования.

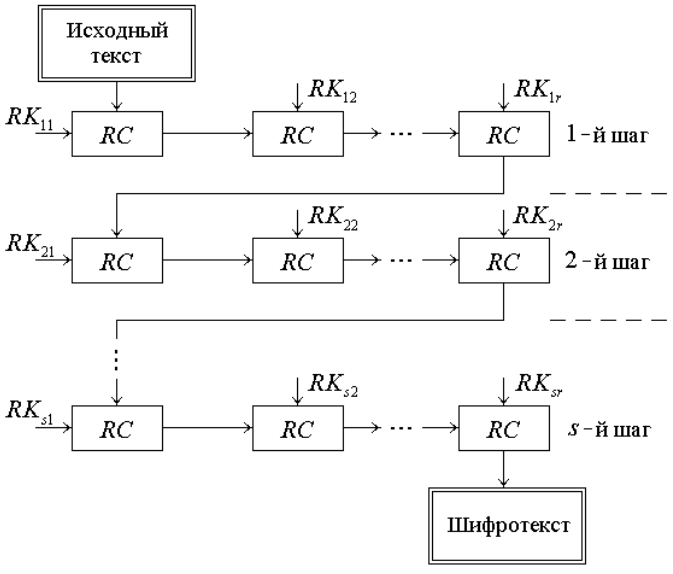

*Рисунок 2. Структурная схема **RSB** алгоритма в режиме зашифрования*

Таким образом, ***RSB*** алгоритм (как и большинство современных итерационных симметричных блочных шифров) состоит из большого количества повторяющихся преобразований – раундов. Как следует из структурной схемы криптоалгоритма (рис. 2), сначала производятся последовательные преобразования всех блоков расширенного файла раундовым ключом $\boldsymbol{RK}_1$, затем ключом $\boldsymbol{RK}_2$ и, наконец, ключом $\boldsymbol{RK}_r$. На этом заканчивается

обработка текста на первом шаге зашифрования. При условии, что число шагов шифрования ***s*** больше единицы, происходит частичное обновление (модификация – TransMasterKey на рис.1) общего ключа зашифрования (***C**ommon **K**ey*). Одновременно с модификацией общего ключа модифицируются и базовые раундовые ключи. Это осуществляется за счет циклического (кругового) сдвига на семь разрядов влево общего (шагового) ключа шифрования ***СК*** (рис. 3). Далее описанная выше процедура преобразования повторяется на очередном шаге зашифрования.

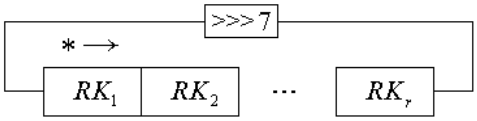

Рисунок 3. Алгоритм модификации шагового ключа зашифрования

Естественно, что на этапе расшифрования последовательность *базовых* раундовых ключей RK должна быть инверсной по отношению к последовательности раундовых ключей зашифрования. Обобщенные структурные схемы ***RSB*** алгоритма в режиме расшифрования приведены на рис. 4.

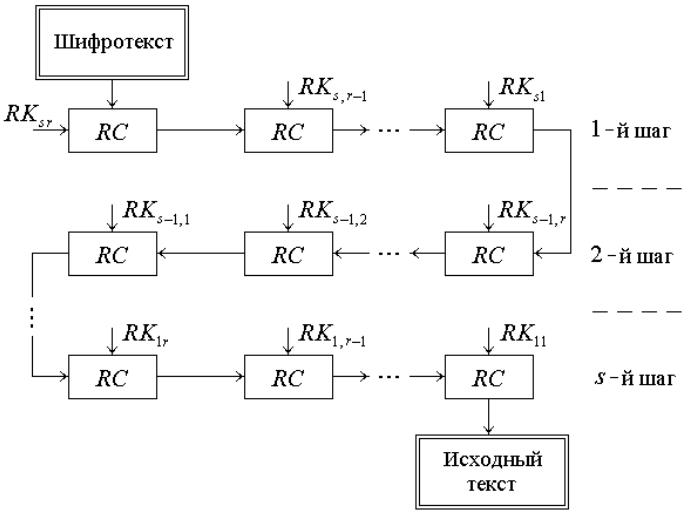

*Рисунок 4. Структурная схема **RSB** алгоритма в режиме расшифрования*

Процесс расшифрования начинается на первом шаге под управлением сначала базового раундового ключа $\boldsymbol{RK_{s,r}}$, затем $\boldsymbol{RK_{s,r-1}}$ и, наконец, $\boldsymbol{RK_{s,1}}$. Переход к очередному шагу расшифрования сопровождается модификацией базовых раундовых ключей. Такая модификация достигается за счет циклического сдвига на семь разрядов, но теперь уже вправо, общего ключа ***СК*** (рис.5), после чего процедура преобразования повторяется на очередном шаге.

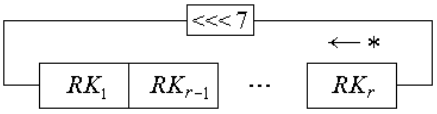

Рисунок 5. Алгоритм модификации шагового ключа расшифрования

Символом * отмечен стартовый (базовый) раундовый ключ RK на этапе зашифрования (рис.3) и расшифрования (рис.5), а стрелка на рисунках указывает, в каком направлении выбираются базовые раундовые ключи на очередном шаге шифрования.

Естественно, что на этапе расшифрования операции криптографических преобразований, входящие во внутренний цикл на рис. 1, должны выполняться в обратном порядке.

Управление примитивами осуществляется содержимым раундовых ключей ***RK*** (как базовых, так и блочных), структура которых приведена на рис.6.

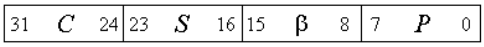

Рисунок 6. Структурная схема блочного раундового ключа

32-разрядный раундовый ключ ***RK*** состоит из четырех восьмиразрядных секторов (байтов), посредством которых осуществляются такие функциональные преобразования блоков шифруемого текста:

C – циклический сдвиг;

S – нелинейная замена байтов;

β – параметризация аддитивной компоненты в операторе замены;

P – перестановка 16-битных слов в пределах шифруемого блока.

Как будет показано в дальнейшем, раундовые ключи, под управлением которых осуществляется шифрование блоков, меняются каждый раз при переходе к очередному преобразуемому блоку. Такая модификация блочных раундовых ключей достигается операцией скользящего кодирования текста, содержащегося в предыдущих блоках. В силу отмеченной особенности 32-разрядные компоненты общего ключа шифрования (рис. 3 и 5) выше названы *базовыми раундовыми ключами*, а результат их преобразования примитивом скользящего кодирования будем называть *блочными раундовыми ключами.* Для первого блока шифруемого текста блочный раундовый ключ совпадает с базовым ключом.

***RSB* криптографические примитивы**

Далее приводится более подробное описание ***RSB*** криптографических примитивов.

Стохастическая круговая прокрутка блока. Посредством данной операции осуществляется циклический сдвиг (круговая прокрутка) шифруемого блока на случайное нечетное число, которое задается семиразрядным двоичным кодом ***RS***. Шесть старших разрядов этого кода считываются из сектора ***C*** блочного раундового ключа (разряды 31–24 на рис.6), а в младший разряд формируемой кодовой комбинации принудительно записывается единица. Тем самым код, которым определяется порядок циклического сдвига блока, будет содержать нечетное число в интервале от 1 до127.

Скользящее кодирование 32-разрядных элементов блока. Операция скользящего кодирования выполняет в криптоалгоритме двойную роль. Во-первых, она обеспечивает достаточно глубокое *перемешивание* преобразуемого текста, цель которого состоит в том, чтобы сделать как можно более сложной зависимость между ключом и шифрограммой. И, во-вторых, с помощью такой операции осуществляется модификация блочных раундовых ключей, под управлением которых выполняются функциональные преобразования блоков текста, начиная со второго. В результате выполняемой модификации блочный раундовый ключ i-го блока ($i \geq 2$) становится зависимым как от исходного базового раундового ключа $\boldsymbol{RK}_j$, под управлением которого осуществляются преобразования первого блока текста, так и от шифруемых данных всех предыдущих (i-1)-х блоков.

В ***RSB*** шифраторе реализованы два типа скользящего кодирования: лево- и правостороннее, причем *левостороннее скользящее кодирование* применяется на нечетных раундах шифрования, а *правостороннее* – на четных раундах. Структурная схема алгоритма прямого левостороннего скользящего кодирования на этапе зашифрования первого блока приведена на рис.7.

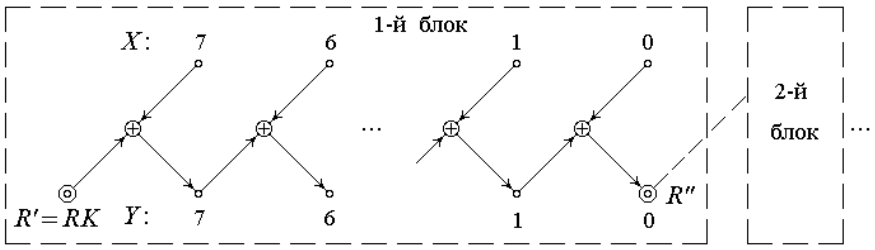

Рисунок 7. Структурная схема алгоритма прямого левостороннего скользящего кодирования на этапе зашифрования

Пусть x_i $(i=\overline{0,7})$ – 32-разрядные элементы (слова) шифруемого блока, а y_i – соответствующие элементы блока после выполнения операции левостороннего скользящего кодирования. Согласно рис. 7 имеем

$$\begin{aligned} y_7 &= R' \oplus x_7 ; \\ y_6 &= y_7 \oplus x_6 ; \\ &\cdots\cdots\cdots\cdots \\ y_1 &= y_2 \oplus x_1 ; \\ y_0 &= y_1 \oplus x_0 , \end{aligned} \qquad (1)$$

где $\oplus$ есть оператор поразрядного сложения по mod 2, а R' – 32-разрядный исходный (базовый) раундовый ключ, принимающий значение RK_j на j - м раунде зашифрования (рис.1).

Как следует из алгоритма прямого левостороннего скользящего кодирования (рис.7) блочным раундовым ключом для второго блока на этапе зашифрования является ключ

$$R'' = y_0 = R' \oplus \bigcup_{i=0}^{7} x_i ,$$

где символом $\bigcup$ обозначена поразрядная сумма по mod2 всех восьми 32-разрядных элементов первого блока (в предположении, что размер блока N составляет 256 бит).

Таким образом, для произвольного k - го блока, $k>1$, блочным раундовым ключом $R^{(k)}$ служит ключ, образованный поразрядным суммированием по mod2 базового раундового ключа R' и всех 32-разрядных элементов (слов) шифруемого текста, предшествующего k - му блоку.

Скользящее кодирование является линейным обратимым преобразованием. Если система уравнений (1) отвечает прямому левостороннему скользящему кодированию шифруемого текста, то обратному преобразованию, применяемому на этапе расшифрования, соответствует система:

$$\begin{aligned} x_7 &= R' \oplus y_7 ; \\ x_6 &= y_7 \oplus y_6 ; \\ &\cdots\cdots\cdots\cdots \\ x_1 &= y_2 \oplus y_1 ; \\ x_0 &= y_1 \oplus y_0 . \end{aligned} \qquad (2)$$

Структурная схема алгоритма обратного левостороннего скользящего кодирования, отвечающая системе преобразований (2), показана на рис.8.

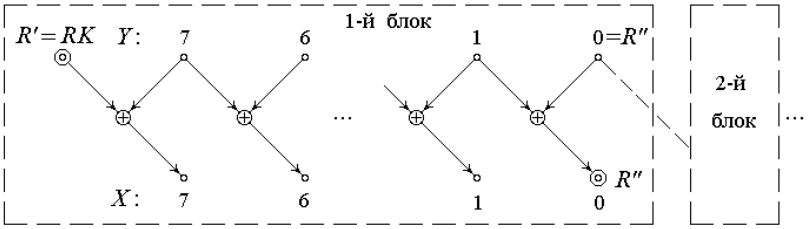

Рисунок 8. Структурная схема алгоритма обратного левостороннего скользящего кодирования на этапе расшифрования

Правостороннее скользящее кодирование, выполняемое на четных раундах шифрования, развивается по направлению справа налево, начиная с младшего (правого) 32-разрядного элемента последнего блока расширенного файла к старшему элементу первого блока. Структурная схема прямого правостороннего скользящего кодирования показана на рис.9.

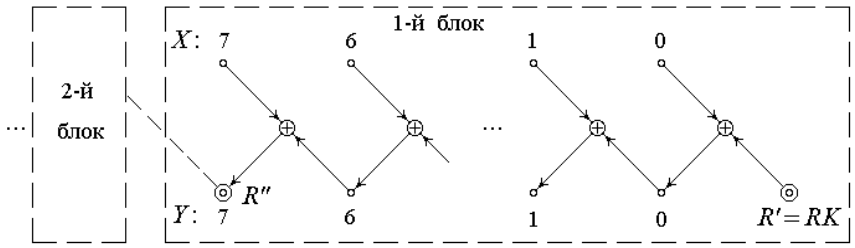

Рисунок 9. Структурная схема алгоритма прямого правостороннего скользящего кодирования на этапе зашифрования

Правостороннему скользящему кодированию (рис.9) отвечает система уравнений

$$\begin{aligned} &y_0 = R' \oplus x_0 ; \\ &y_1 = y_0 \oplus x_1 ; \\ &\text{.....} \\ &y_6 = y_5 \oplus x_6 ; \\ &y_7 = y_6 \oplus x_7 = R'' . \end{aligned} \qquad (3)$$

Согласно рис. 9 старший преобразованный 32-разрядный элемент y_7 первого блока используется в качестве блочного раундового ключа R'' для второго блока. Правостороннее обратное скользящее кодирование выполняется по схеме, приведенной на рис.10.

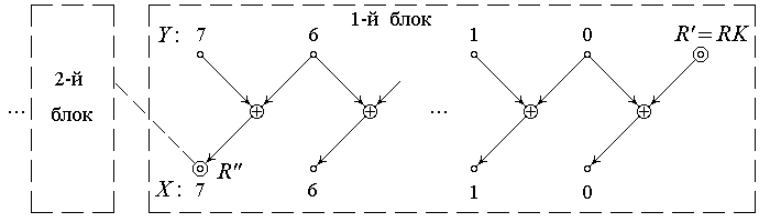

Рисунок 10. Структурная схема алгоритма обратного правостороннего скользящего кодирования на этапе расшифрования

Обратному правостороннему скользящему кодированию соответствует система уравнений

$$
\begin{aligned}
x_0 &= R' \oplus y_0 ; \\
x_1 &= y_0 \oplus y_1 ; \\
&\ldots\ldots\ldots\ldots\ldots \\
x_6 &= y_5 \oplus y_6 ; \\
x_7 &= y_6 \oplus y_7 .
\end{aligned} \qquad (4)
$$

Примитив типа левостороннего скользящего кодирования в общих чертах напоминает схему шифрования в режиме *сцепления блоков шифротекста* – ***CBC*** (***C****iphertext* ***B****lock* ***C****haining*) [1]. Отметим следующие особенности ***RSB*** скользящего кодирования по сравнению с ***CBC*** режимом. Во-первых, в схеме шифрования ***CBC*** осуществляются преобразования над блоками шифротекста, тогда как в ***RSB*** шифре скользящему кодированию подвергаются 32-разрядные элементы блоков на каждом раунде шифрования, причем базовые раундовые ключи в ***RSB*** шифре исполняют роль синхропосылки (вектора инициализации) ***CBC*** режима. Во-вторых, в ***RSB*** алгоритме скользящим кодированием достигается не только глубокое перемешивание шифротекста (как и в ***CBC*** режиме), что повышает криптостойкость шифра, но и происходит существенная модификация блочных раундовых ключей, управляющих преобразованиями последующих блоков текста. В результате таких преобразований блочные раундовые ключи, начиная со второго, становятся зависимыми не только от базового раундового ключа (компоненты общего ключа шифрования), но также и от шифруемых данных всех предыдущих блоков текста. И, наконец, в-третьих, кроме левостороннего кодирования в ***RSB*** алгоритме применяется также правостороннее скользящее кодирование (для тех же целей, что и левостороннее). Последняя отмеченная особенность ***RSB*** шифра является уникальной, не встречающейся ни в одном из известных блочных криптографических алгоритмов.

Обратим внимание на то, что предлагаемые схемы скользящего кодирования (рис.7-10) и отвечающие им системы уравнений (1)–(4) являются ничем иным, как *преобразованиями Грея* «наоборот». Это означает, что прямое скользящее кодирование есть не что иное, как соответствующее (по направлению) *обратное преобразование Грея*, тогда как обратное скользящее кодирование представляет собой *прямое* преобразование Грея [9].

Нелинейная замена байтов. В современных блочных криптосистемах операции нелинейной замены (подстановки) выполняются над двоичными кодами небольшой разрядности. Так, в популярном в недалеком прошлом криптоалгоритме ***DES*** размер элемента замены составлял четыре бита, а в российском шифре ***ГОСТ*** 28147-89, как и в американском криптостандарте ***AES***, элементом замены является восьмиразрядная бинарная комбинация – байт. В ***RSB*** криптосистеме в качестве элемента замены также выбран байт.

Схема построения S - боксов, посредством которых реализуются операции нелинейной замены байтов в ***RSB*** алгоритме, наследует основные черты S - боксов, принятых в ***AES*** шифре. S - преобразование ***RSB*** алгоритма (как об этом уже было сказано выше) представляет собой нелинейную замену байтов, выполняемую независимо с каждым байтом обрабатываемого блока. Таблицы замены S - боксов инвертируемы и построены из композиции двух таких преобразований:

1) получения обратного элемента относительно умножения в расширенном поле Галуа $GF(2^8)$, нулевой элемент '00' переходит сам в себя;

2) применения преобразования над $GF(2)$, определяемого соотношением

$$\boldsymbol{y} = (\boldsymbol{x} \oplus \boldsymbol{S})^{-1} \otimes \boldsymbol{M} \oplus \boldsymbol{\beta} . \qquad (5)$$

В аффинном преобразовании (5) использованы переменные: $\boldsymbol{S}$ и $\boldsymbol{\beta}$ – байты раундового ключа, структурная схема которого представлена на рис. 6; $\boldsymbol{x}$ – байт шифруемого блока; $\boldsymbol{a}^{-1}$ – восьмиразрядный элемент (байт), мультипликативно обратный

байту $\boldsymbol{a}$ над неприводимым двоичным полиномом $\varphi(x)$, в качестве которого, как и в ***AES***, выбран полином восьмого порядка $\varphi(x) = x^8 + x^4 + x^3 + x + 1$; $\boldsymbol{M}$ – инволютивная матрица преобразования, заданная соотношением

$$\boldsymbol{M} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Впрочем, в качестве $\boldsymbol{M}$ можно было выбрать любую другую невырожденную двоичную *инволютивную* матрицу восьмого порядка. Инволютивной называется матрица, обратная которой совпадает с исходной. Мультипликативный обратный элемент $\boldsymbol{a}^{-1}$ над полем $GF(2^8)$ как раз и доставляет преобразованию (5) нелинейные свойства.

Стохастическая перестановка слов блока. С помощью этого криптографического примитива осуществляется стохастическая перестановка (перемешивание) двоичных слов (16-битных кодовых комбинаций) в пределах шифруемого блока. Следовательно, в 256-битном блоке содержится 16 слов.

Операция стохастической перестановки слов блока реализуется следующим образом. Пусть $\boldsymbol{x}$ означает четырехразрядный двоичный номер слова шифруемого блока. Слово, расположенное в ячейке блока, двоичный номер которой равен $\boldsymbol{x}$, перемещается в ячейку под номером $\boldsymbol{y}$, причем

$$\boldsymbol{y} = (\boldsymbol{x} \bullet \boldsymbol{M}_p)_2, \tag{6}$$

где $(a)_2$ означает приведение результатов матричного произведения в правой части (6) к остатку по mod 2, а M_p – матрица перестановки, в качестве которой выбирается одна из 16-ти инволютивных двоичных матриц четвертого порядка. Адрес А матрицы M_p содержится в секторе P блочного раундового ключа RK (рис. 6) и образуется по правилу:

$$P = p_1 \,\|\, p_2 \Rightarrow \boldsymbol{A} = \oplus \begin{matrix} p_1 \\ p_2 \end{matrix}, \tag{7}$$

где p_1 и p_2 - полубайты сектора P.

Реализация *RSB* алгоритмов

Приведем далее краткое описание функционирования ***RSB*** криптоалгоритма сначала на этапе зашифрования, а затем на этапе расшифрования.

Зашифрование открытого текста. Пусть $RK_1, RK_2, ..., RK_r$ – исходные 32-битные компоненты (базовые раундовые ключи) общего ключа CK на первом шаге зашифрования. Обозначим через $R' = RK_1$ раундовый ключ, с помощью которого осуществляется управление криптопреобразованиями первого (расположенного слева) 256-разрядного блока открытого текста (InText). Обозначим также через C', S', β' и P' байты раундового ключа R' (рис.6). Шесть старших разрядов сектора C' ключа R' переписываются в старшие разряды семиразрядного регистра $\boldsymbol{RS}$, причем в младшем разряде регистра постоянно удерживается единица. Таким образом, регистр $\boldsymbol{RS}$ будет содержать случайное (за счет случайности секретного ключа RK_1) нечетное число $C \in [1, 127]$. Первый блок открытого текста подвергается круговой прокрутке на C разрядов влево. Затем восемь 32-разрядных элементов первого блока преобразуется по схеме, приведенной на рис.7, при этом элементы

$x_7, x_6, ..., x_0$ блока замещаются элементами $y_7, y_6, ..., y_0$ (см. ф-лу (1)). 32-битный элемент y_0 запоминается в качестве блочного раундового ключа RK'', под управлением которого будут выполняться криптопреобразования второго блока открытого текста. После завершения процедуры левостороннего скользящего кодирования байты первого блока подвергаются сначала нелинейной замене (5), в ходе которой используется содержимое секторов S' и β' раундового ключа R' (рис.6), а затем стохастической перестановке (6), при этом матрица перестановки $\boldsymbol{M}_p$ извлекается из памяти по адресу, который содержится в секторе P' ключа R' (см. преобразование (7)). На этом завершаются преобразования первого блока открытого текста.

Аналогичным образом осуществляются зашифрование второго блока открытого текста, но уже под управлением блочного раундового ключа R'', сформированного на этапе скользящего кодирования первого блока. Операция скользящего кодирования второго блока текста приводит к формированию раундового ключа RK''' для третьего блока и т.д. После окончания обработки последнего блока открытого текста (TextFin) процесс зашифрования (теперь уже частично зашифрованного текста) продолжается по вышеописанной схеме под управлением базовых раундовых ключей $RK_2, RK_3, ..., RK_r$. На этом завершается первый шаг зашифрования.

Перед началом второго шага зашифрования общий ключ CK модифицируется, что реализуется циклическим сдвигом CK на семь разрядов влево (рис.3). В результате такого сдвига модифицируются также базовые раундовые ключи $RK_1, RK_2, ..., RK_r$. Процесс зашифрования входного текста на втором и последующих шагах подобен зашифрованию на первом шаге.

Расшифрование криптограммы. Естественно, что перед началом расшифрования криптограммы необходимо восстановить то состояние общего ключа CK, в котором он находился на последнем этапе зашифрования. С этой целью исходный ключ CK должен быть подвергнут циклическому сдвигу на $7\boldsymbol{s}$ разрядов влево. Кроме того, следует иметь ввиду, что если на этапе зашифрования базовые раундовые ключи «отрабатывают» в натуральной последовательности (рис. 2 и 3), т.е. сначала управление зашифрованием передается базовому ключу RK_1, затем RK_2 и, наконец, RK_r, то при расшифровании порядок использования раундовых ключей должен быть обратным. Это означает, что на каждом шаге расшифрования базовые раундовые ключи используются в инверсной последовательности, а именно: $RK_r, RK_{r-1}, ..., RK_1$ (рис. 4 и 5).

Номер блока, с которого начинается процесс расшифрования, зависит от того, четным или нечетным являлось число раундов $\boldsymbol{r}$ в одном шаге зашифрования. Предположим, что $\boldsymbol{r}$ – нечетное число. В таком случае расшифрование начинается с первого (расположенного слева) блока шифротекста и развивается далее по направлению слева направо. Если же $\boldsymbol{r}$ – четное число, то процесс расшифрования развивается по направлению справа налево, начиная с последнего (правого) блока шифрограммы. Последней криптооперацией, которой был подвергнут первый блок (а это левый блок шифрограммы при нечетном $\boldsymbol{r}$ и правый, если $\boldsymbol{r}$ – четное число), являлась операция стохастической перестановки байтов. Поэтому, прежде всего, необходимо восстановить первоначальное расположение байтов первого блока. Эта операция осуществляется преобразованием

$$\boldsymbol{x} = \boldsymbol{y} \cdot \overline{\boldsymbol{M}}_p \,, \tag{8}$$

являющимся обратным преобразованию (6), и реализована в ***RSB*** шифре табличным способом.

После завершения операции обратной перестановки байтов (8) выполняется операция восстановления исходного состояния $\boldsymbol{x}$ байтов, подвергнутых нелинейной замене (5). Как следует из соотношения (5), сначала необходимо устранить компоненту β, в результате чего получим

$$(\boldsymbol{x} \oplus \boldsymbol{S})^{-1} \otimes \boldsymbol{M}_s = \boldsymbol{y} \oplus \beta \ , \qquad (9)$$

а затем из равенства (9) извлечь $Z = \boldsymbol{x} \oplus \boldsymbol{S}$, что также производится табличным способом.

И, наконец, разрешая Z относительно $\boldsymbol{x}$, получим

$$\boldsymbol{x} = Z \oplus \boldsymbol{S} \ .$$

В результате выполнения этих двух операций (обратной перестановки и восстановления исходных значений байтов) приходим к состоянию первого блока шифротекста, в котором он находился после скользящего кодирования под управлением раундового ключа R', равного RK_r, а точнее – RK_{rs}. Поэтому необходимо запомнить значение операнда y_0, являющегося блочным раундовым ключом R'', под управлением которого проводилась операция зашифрования второго блока шифротекста на r - м раунде криптопреобразования. Этот же ключ R'' должен быть использован также и для расшифрования второго блока. После того, как вычислен (и сохранен) ключ R'', можно провести операции обратного левостороннего скользящего кодирования первого блока (рис.8) и обратной стохастической прокрутки (но теперь уже вправо) на число разрядов, определяемого семью старшими разрядами сектора C ключа R' (с учетом единицы в младшем разряде байта, которым задается нечетный порядок обратного циклического сдвига).

Аналогичным образом выполняется обратное преобразование второго блока, третьего и т.д. Затем точно также преобразования блоков шифротекста выполняются под управлением базовых раундовых ключей $RK_{r-1}, RK_{r-2}, ..., RK_1$. После завершения процесса преобразований шифрограммы на первом шаге расшифрования общий ключ подвергается циклическому сдвигу на семь разрядов вправо, тем самым восстанавливаются базовые раундовые ключи, отвечающие второму шагу расшифрования.

Статистический анализ *RSB* алгоритма

Один из возможных способов оценки стойкости криптографических алгоритмов состоит в оценке статистической безопасности шифратора. Считается, что шифр является статистически безопасным, если последовательность, которую он генерирует (или образуемая в результате зашифрования открытого текста шифрограмма), обладает свойствами, не отличающимися от случайных последовательностей. Такие последовательности называются «псевдослучайными». Для оценки того, насколько близко криптоалгоритмы аппроксимируют генераторы «случайных» последовательностей, используются статистические тесты. Предложенный Американским Национальным Институтом стандартов пакет ***NIST STS*** [10] для тестирования генераторов случайных или псевдослучайных чисел служит одним из подходов к реализации задачи оценки статистической безопасности криптографических алгоритмов. Использование данного пакета позволяет с высокой долей вероятности выносить решение относительно того, насколько последовательность, генерируемая исследуемым шифратором, является статистически безопасной. Пакет ***NIST STS*** (версия 18.1) содержит в себе 15 статистических тестов. Однако фактически вычисляется 188 статистических параметров, предназначенных для определения соответствующей глобальной или локальной оценки.

Пакет позволяет представить результаты статистических испытаний криптоалгоритмов в виде статистических портретов шифрограмм, формируемых тестируемым шифратором. На рис.11 приведен пример одного из портретов шифрограммы, образованной в результате зашифрования алгоритмом ***RSB*** русскоязычного текстового файла (словарь Даля) объемом 16.5 Мбайт.

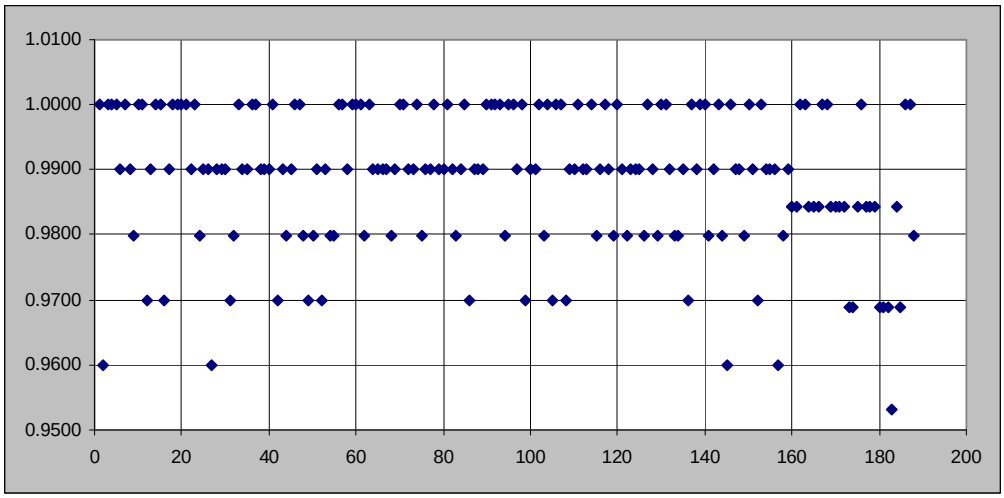

Рисунок 11. Пример статистического портрета шифрограммы, образованной криптоалгоритмом ***RSB***

На оси абсцисс портрета отложены номера статистических характеристик $(i = \overline{1,188})$ псевдослучайных последовательностей, а на оси ординат – их относительные значения γ_i. Пороговым значением параметра γ_i является значение, не меньшее 0,96. Таким образом, согласно рис. 11, лишь один из 188 статистических параметров анализируемой шифрограммы не укладывается в расчетные пределы допустимых границ, что вполне допустимо для последовательностей, характеризуемых как псевдослучайные.

Выводы:

1. ***RSB*** алгоритм допускает динамичное управление в широком диапазоне такими параметрами шифрования, как размер общего секретного ключа и число шагов (а, следовательно, и раундов) криптографических преобразований.

2. Криптографические преобразования в каждом блоке осуществляются под управлением индивидуальных локальных раундовых ключей, зависящих не только от значения секретного базового раундового ключа, но и всего текста, предшествующего преобразуемому блоку.

3. Основные выполняемые в ***RSB*** шифре криптографические преобразования (циклический сдвиг блока, скользящее кодирование 32-битных элементов, нелинейная подстановка байтов и перестановка слов в блоках) относятся к классу стохастических управляемых операций шифрования.

4. Стохастичность операций ***RSB*** шифрования обеспечивается не только выбором случайных базовых раундовых ключей, но и многократным домешиванием в локальные раундовые ключи криптографически преобразуемых (в силу чего приобретающих стохастические свойства) 32–битных элементов шифруемого текста.

5. ***RSB*** алгоритм допускает аппаратную реализацию на платформах с 32-х разрядными шинами, причем для повышения быстродействия возможно распараллеливание операций нелинейных подстановок байтов и перестановок слов в блоках шифрования.

6. Как показали результаты статистических испытаний ***RSB*** шифра пакетом ***NIST STS***, эффективность ***RSB*** алгоритма оказалась на уровне не ниже, а для отдельных параметров шифрования – превышающем эффективность широко используемых стандартов криптографической защиты, таких как ***DES***, ***ÃÎÑÒ***, ***AES*** и др.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Шнайер Б. Прикладная криптография / Б. Шнайер. – М.: «ТРИУМФ», 2003. – 816 с.
2. Венбо Мао. Современная криптография: теория и практика / Венбо Мао. – М.: «Вильямс», 2005. – 768 с.
3. Белецкий А.Я. Преобразования Грея / Белецкий А.Я., Белецкий А.А., Белецкий Е.А. Т. 2. Прикладные аспекты. – К.: Кн. Изд-во НАУ, 2007. – 644 с.
4. Белецкий А.Я. Симметричный блочный криптоалгоритм / Белецкий А.Я., Белецкий А.А. // Захист інформації № 2 (29), 2006. – С. 42-51.
5. Белецкий А.Я. / Белецкий А.Я., Белецкий А.А., Кузнецов А.А. Семейство симметричных блочних криптографических алгоритмов с динамически управляемыми параметрами шифрования // Електроніка та системи управління. – К..: НАУ, 2007. № 1 (11). Белецкий А.Я., Белецкий А.А., Кузнецов А.А. – С. 5-16.
6. Зензин О.С. Стандарт криптографической защиты – AES. Конечные поля / Зензин О.С., Иванов М.А. / Под редакцией М.А. Иванова. – М.: КУДИЦ-ОБРАЗ, 2002. – 176 с.
7. Молдаван Н.А. Криптография: от примитивов к синтезу алгоритмов. / Молдаван Н.А., Молдаван А.А., Еремеев М.А. –СПб.: БХВ-Петербург, 2004. – 448 с.
8. Молдаван Н.А. Криптография: скоростные шифры. / Молдаван Н.А., Молдаван А.А.,Гуц Н.Д., Изотов Б.В. – СПб.: БХВ-Петербург, 2002. – 496 с.
9. Белецкий А.Я. Преобразования Грея / Белецкий А.Я., Белецкий А.А., Белецкий Е.А. Т. 1. Основы теории. – К.: Кн. Изд-во НАУ, 2007. – 412с.
10. Random Number Generation and Testing. http: www.csrc.nist.gov/rng/