# Сравнение стандарта шифрования РФ и нового стандарта шифрования США.

Андрей Винокуров (avin@chat.ru),
Эдуард Применко.

В настоящей статье выполнено сравнение двух стандартов шифрования – российского и нового американского с акцентом на технологичность и эффективность их реализаций. Сравнению криптографических характеристик обоих шифров здесь уделено несколько меньшее внимание ввиду высокой сложности вопроса – достаточно серьезное его раскрытие потребовало бы гораздо более объемной работы. Статья была опубликована в Журнале «Системы безопасности» издательства «Гротэк», №№1 и 2 за 2001 год, под длинным названием «Сравнение российского стандарта шифрования алгоритма ГОСТ 28147-89 и алгоритма Rijndael, выбранного в качестве нового стандарта шифрования США». По вине редакции журнала в нем вышла неокончательная версия статьи с массой опечаток, к тому же была искажена терминология: по какой-то причине литературные редакции журналов отказываются признавать существование терминов по ГОСТ «зашифрование» и «расшифрование». По сравнению с журнальным вариантом в настоящей версии статьи уточнена оценка характеристик быстродействия возможных реализаций шифров на платформе Intel Pentium и сделаны незначительные изменения в стилистике и обозначениях. При необходимости делайте ссылки на журнальный вариант статьи:

А.Винокуров, Э.Применко. Сравнение российского стандарта шифрования, алгоритма ГОСТ 28147-89, и алгоритма Rijndael, выбранного в качестве нового стандарта шифрования США., «Системы безопасности», М., изд. «Гротэк», 2001, №№1,2.

## С о д е р ж а н и е

## Введение.

Второго октября 2000 года департамент торговли США подвел итоги конкурса по выработке нового стандарта шифрования США. Победителем стал алгоритм «Rijndael» [1], разработанный бельгийскими криптографами. Конкурс был объявлен двумя годами раньше национальным институтом стандартов и технологии США, входящим на правах агентства в департамент торговли. Первоначально было выдвинуто 15 шифров-конкурсантов, после подведения промежуточных итогов осталось 5 финалистов, из которых и был выбран кандидат на утверждение в качестве стандарта. Новый шифр призван заменить алгоритм DES, являющийся стандартом шифрования США с 1977 года.

DES был разработан в исследовательской лаборатории фирмы IBM в первой половине 70-х годов и принадлежит к линии шифров, берущих свое начало с алгоритма «Люцифер», созданного там же несколькими годами раньше. Эта архитектура, получившая название «сеть Файстеля», занимала до сегодняшнего дня доминирующее положение в криптографии: к ней относится большинство современных шифров, включая отечественный стандарт алгоритм ГОСТ28147-89 [2,3]. В настоящей работе выполнен сравнительный анализ шифров ГОСТ28147-89 и «Rijndael», и на их примере сделано сравнение традиционного и более современного подходов к построению блочных шифров [4].

Прежде чем перейти непосредственно к сравнению шифров, необходимо сделать несколько замечаний об архитектурных решениях, характерных для шифров 70-х годов. В то время микроэлектроника находилась в самом начале своего пути к сегодняшнему потрясающему уровню развития, было налажено промышленное производство микросхем только малой и средней степени интеграции, и поэтому основной доминантой в проектировании шифров являлось обеспечение приемлемой стойкости при жестких требованиях к сложности реализации. В настоящее время возможности технической базы по реализации шифров возросли на несколько порядков и пропорционально увеличились возможности их криптоанализа. В связи с этим первоначальные требования к экономичности реализации шифров перестали быть доминирующим фактором, в то время как требования к их стойкости существенно возросли. Именно на фоне этих процессов и происходят современные изменения в подходах к построению блочных шифров с секретным ключом.

## Сравнение общих характеристик алгоритмов.

Сравнительные характеристики алгоритмов ГОСТ и Rijndael приведены в следующей ниже таблице 1.

**Таблица 1. Сравнительные характеристики алгоритмов ГОСТ28147-89 и Rijndael.**

| Показатель | ГОСТ28147-89 | Rijndael |
|---|---|---|
| Размер блока, бит | 64 | 128, 192, 256[1] |
| Размер ключа, бит | 256 | 128, 192, 256 |
| Архитектура | Однородная сбаланси-рованная сеть Файстеля | «Квадрат» (Square) |
| Число раундов | 32 | 10, 12, 14[2] |

[1] В качестве нового стандарта выбран вариант шифра с размером блока только 128 бит.

[2] Количество раундов зависит от размера ключа и шифруемого блока – из этих двух размеров берется максимальный, и если он равен 128 бит, то используются 10 раундов шифрования, если 192 бита, то 12 раундов, если 256 бит, то 14 раундов.

**Таблица 1. Сравнительные характеристики алгоритмов ГОСТ28147-89 и Rijndael.**

| Показатель | ГОСТ28147-89 | Rijndael |
|---|---|---|
| Часть блока, шифруемая за один раунд, бит | 32 (полблока) | 128, 192, 256 (полный блок) |
| Размер раундового ключевого элемента, бит | 32 (половина размера блока) | 128, 192, 256 (равен размеру блока) |
| Структура раунда | Простая | Более сложная |
| Используемые на раунде операции | Только аддитивные операции, подстановки и сдвиги | Широкое использование операций над конечными полями |
| Эквивалентность прямого и обратного преобразований. | С точностью до порядка следования ключевых элементов | С точностью до вектора ключевых элементов, узла замен и прочих констант алгоритма |

В отличие от ГОСТа, размер шифруемого блока и размер ключа в алгоритме Rijndael могут изменяться, что допускается использованной в нем архитектурой «квадрат». Данное свойство позволяет варьировать стойкость и быстродействие алгоритма в зависимости от внешних требований к реализации в некоторых пределах, –однако, не очень широких, – число раундов, а вместе с ним и быстродействие, в крайних случаях различаются в 1.4 раза.

## Сравнение общих архитектурных принципов.

Криптоалгоритм ГОСТ28147-89, как и большинство шифров «первого поколения», разрабатывавшихся в 70-е годы и в первой половине 80-х, базируется на архитектуре «сбалансированная сеть Файстеля» (balanced Feistel network) [5]. Основным принципом этой архитектуры является то, что весь процесс шифрования состоит из серии однотипных раундов. На каждом раунде шифруемый блок $\boldsymbol{T}$ делится на две части ($\boldsymbol{T}_0, \boldsymbol{T}_1$), одна из которых модифицируется путем побитового сложения по модулю 2 со значением, вырабатываемом из другой части и ключевого элемента раунда с помощью функции шифрования. Между раундами части блока меняются местами, и, таким образом, на следующем раунде текущий измененный блок станет неизменным и наоборот. Схема алгоритма шифрования по ГОСТ 28147-89 приведена на рисунке 1(а). Подобная архитектура позволяет легко получить обратимое криптографическое преобразование из сложной и, возможно необратимой, функции шифрования. Важной особенностью этого подхода является то, что за раунд шифруется ровно половина блока.

Шифр Rijndael имеет принципиально другую архитектуру, получившую название «квадрат» (Square) по имени первого выполненного в ней шифра,– он был разработан теми же специалистами несколькими годами раньше. Эта архитектура базируется на прямых преобразованиях шифруемого блока, который представляется в форме матрицы байтов. Зашифрование также состоит из серии однотипных шагов, раундов, однако на каждом раунде блок преобразуется как единое целое и не остается неизменных частей блока. Таким образом, за раунд шифруется полный блок, следовательно, для обеспечения сопоставимой сложности и нелинейности преобразования таких шагов требуется вдвое меньше по сравнению с сетью Файстеля. Каждый раунд заключается в побитовом сложении по модулю 2 текущего состояния шифруемого блока и ключевого элемента раунда, за которым следует сложное нелинейное преобразование блока, сконструированное из трех более простых преобразований, подробно рассмотренных в следующем разделе. Схема алгоритма Rijndael приведена ниже на рисунке 1(б).

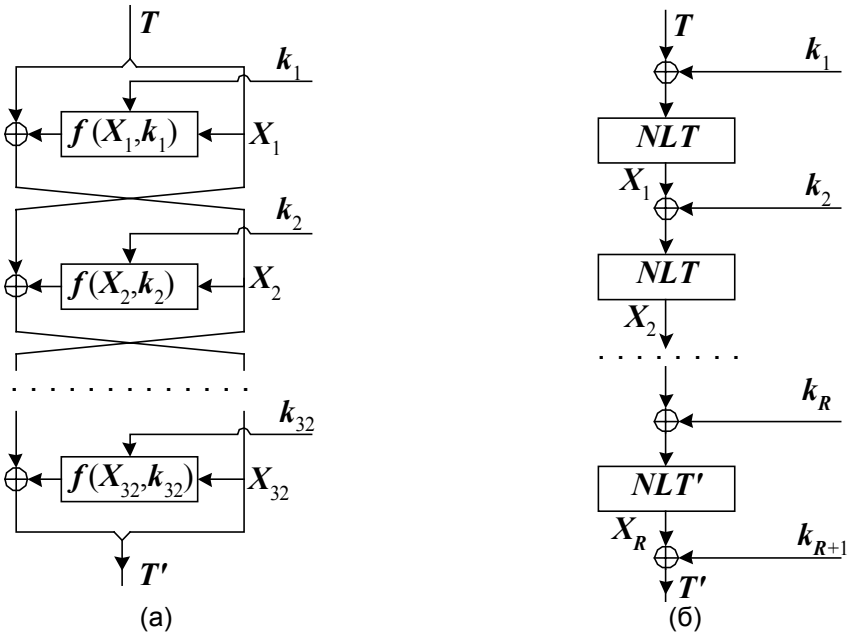


Рис. 1. Схема преобразования данных при шифровании по алгоритмам ГОСТ28147-89 (а) и Rijndael (б) соответственно, где:

$T, T'$ – исходный и зашифрованный блоки соответственно;

$k_i$ – ключевой элемент раунда;

$X_i$ – состояние процесса шифрования после $i$-того раунда;

$f(X,k)$ – функция шифрования алгоритма ГОСТ28147-89;

$NLT, NLT'$ – регулярное нелинейное преобразование и нелинейное преобразование последнего раунда алгоритма Rijndael соответственно;

$R$ – число раундов в алгоритме Rijndael (10, 12 или 14).

## Сравнение раундов шифрования.

В алгоритме ГОСТ28147-89 используется сравнительно несложная функция шифрования, состоящая из аддитивной операции комбинирования входного полублока с ключевым элементом раунда – сложения их по модулю $2^{32}$, подстановки, выполняемой независимо в восьми 4-битовых группах, и битовой перестановки – вращения на 11 бит в сторону старших разрядов. Схема раунда шифрования по ГОСТ изображена на рисунке 2(а).

В Rijndael шифруемый блок и его промежуточные состояния в ходе преобразования представляются в виде матрицы байтов $4 \times n$, где $n = 4, 6, 8$ в зависимости от размера блока. Функция нелинейного преобразования в алгоритме Rijndael состоит из трех следующих элементарных преобразований, выполняемых последовательно:

- байтовая подстановка – каждый байт преобразуемого блока заменяется новым значением, извлекаемым из общего для всех байтов матрицы вектора замены;
- побайтовый циклический сдвиг в строках матрицы: первая строка остается неизменной, вторая строка циклически сдвигается влево на один байт, третья и четвертая строка циклически сдвигаются влево соответственно на 2 и 3 байта для $n = 4$ или 6, и на 3 и 4 байта для $n = 8$;
- матричное умножение – полученная на предыдущем шаге матрица умножается слева на следующую матрицу–циркулянт размера $4 \times 4$:

$$M = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}.$$

При этом операции с элементами матриц (сложение и умножение) выполняются в конечном поле GF($2^8$), порождаемом неприводимым над GF(2) полиномом $\boldsymbol{m}(\boldsymbol{x}) = \boldsymbol{x}^8 + \boldsymbol{x}^4 + \boldsymbol{x}^3 + \boldsymbol{x} + 1$. В этом конечном поле сложение байтов выполняется как побитовое суммирование по модулю 2, а умножение – несколько более сложным способом. Схема раунда алгоритма Rijndael изображена на рисунке 2(б).

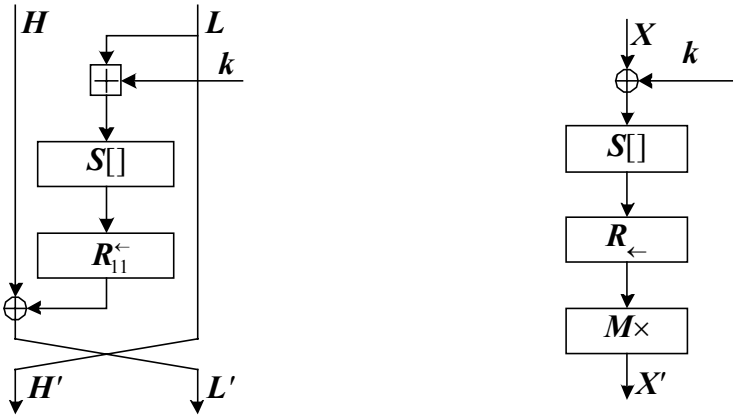


Рис. 2. Схема преобразования данных для одного раунда шифрования по алгоритмам ГОСТ28147-89 (а) и Rijndael (б) соответственно, где:

$\boldsymbol{X}(\boldsymbol{H},\boldsymbol{L})$, $\boldsymbol{X}'(\boldsymbol{H}',\boldsymbol{L}')$ – преобразуемый блок (или его старшая и младшая части соответственно) на входе и на выходе раунда;

$\boldsymbol{k}$ – ключевой элемент раунда;

$\boldsymbol{S}[]$ – функция подстановки, группами по 4 бита для ГОСТа и байтами для алгоритма Rijndael;

$\boldsymbol{R}_{11}^{\leftarrow}$ – операция циклического сдвига (вращения) 32-битового слова на 11 бит в сторону старших разрядов;

$\boldsymbol{R}_{\leftarrow}$ – операция построчного вращения матрицы алгоритма Rijndael;

$\boldsymbol{M}\times$ – умножение матрицы данных слева на матрицу $\boldsymbol{M}$ в алгоритме Rijndael.

Если в алгоритме ГОСТ28147-89 перестановку полублоков отнести к раунду шифрования, как это показано на рисунке 2(а), то можно заметить, что в обоих алгоритмах все раунды шифрования идентичны друг другу, за исключением последнего раунда, в котором отсутствует часть операций. Такой подход позволяет получить наиболее компактную реализацию, как аппаратную, так и программную. В последнем раунде ГОСТа отсутствует операция перестановки полублоков шифруемого блока, в последнем раунде алгоритма Rijndael – умножение слева на матрицу $\boldsymbol{M}$. В обоих обсуждаемых алгоритмах это сделано для того, чтобы обеспечить эквивалентность структуры прямого и обратного шифрующих преобразований, данный вопрос обсуждается в следующем разделе.

## Эквивалентность прямого и обратного преобразований.

В алгоритме ГОСТ28147-89 эквивалентность структуры прямого и обратного криптографического преобразования не обеспечивается специально, а является простым следствием использованного архитектурного решения. В любой однородной сбалансированной сети Файстеля оба эти преобразования алгоритмически идентичны и различаются

только порядком использования ключевых элементов: при расшифровании элементы используются в порядке, обратном тому, в котором они используются при зашифровании.

Шифр Rijndael построен на базе прямых преобразований. Как и для всех подобных алгоритмов, обратное преобразование строится из обращений шагов прямого преобразования, применяемых в обратном порядке. В силу сказанного обеспечить такую же степень идентичности прямого и обратного преобразования, которая достигается в сетях Файстеля, не представляется возможным. Однако специальными конструкторскими решениями достигается близкая степень соответствия: прямое и обратное преобразование получаются идентичными с точностью до используемых в преобразованиях констант. В следующей ниже таблице 2 представлены два последних раунда алгоритма Rijndael и их формальное обращение:

**Таблица 2. Два раунда алгоритма Rijndael и их формальное обращение.**

| **Прямое преобразование** | **Обратное преобразование** |
|---|---|
| $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}-1}$ | $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}+1}$ |
| $\boldsymbol{X} = \boldsymbol{S}(\boldsymbol{X})$ | $\boldsymbol{X} = \boldsymbol{R}_{\rightarrow}(\boldsymbol{X})$ |
| $\boldsymbol{X} = \boldsymbol{R}_{\leftarrow}(\boldsymbol{X})$ | $\boldsymbol{X} = \boldsymbol{S}^{-1}(\boldsymbol{X})$ |
| $\boldsymbol{X} = \boldsymbol{M} \times \boldsymbol{X}$ | $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}}$ |
| $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}}$ | $\boldsymbol{X} = \boldsymbol{M}^{-1} \times \boldsymbol{X}$ |
| $\boldsymbol{X} = \boldsymbol{S}(\boldsymbol{X})$ | $\boldsymbol{X} = \boldsymbol{R}_{\rightarrow}(\boldsymbol{X})$ |
| $\boldsymbol{X} = \boldsymbol{R}_{\leftarrow}(\boldsymbol{X})$ | $\boldsymbol{X} = \boldsymbol{S}^{-1}(\boldsymbol{X})$ |
| $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}+1}$ | $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}-1}$ |

Через $\boldsymbol{R}_{\rightarrow}$ обозначена обратная к $\boldsymbol{R}_{\leftarrow}$ операция построчного циклического сдвига матрицы. Как видно из таблицы 2, алгоритмическая структура прямого и обратного преобразований существенно различается. Однако путем тождественных преобразований можно добиться большего соответствия между ними. Прежде всего, стоит отметит, что операция побайтовой замены ($\boldsymbol{S}$) коммутативна с процедурой побайтового сдвига строк матрицы:

$$\boldsymbol{S}^{-1}(\boldsymbol{R}_{\rightarrow}(\boldsymbol{X})) = \boldsymbol{R}_{\rightarrow}(\boldsymbol{S}^{-1}(\boldsymbol{X})).$$

Кроме того, согласно правилам матричной алгебры по закону ассоциативности можно также поменять порядок побитового прибавления ключа по модулю два и умножения на матрицу:

$$\boldsymbol{M}^{-1} \times (\boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}}) = (\boldsymbol{M}^{-1} \times \boldsymbol{X}) \oplus (\boldsymbol{M}^{-1} \times \boldsymbol{k}_{\boldsymbol{R}}).$$

Применяем указанные изменения ко второму столбцу таблицы 2, получаем следующую последовательность операций при двух раундах обратного преобразования:

**Таблица 3. Два раунда алгоритма Rijndael и их обращение.**

| **Прямое преобразование** | **Обратное преобразование** |
|---|---|
| $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}-1}$ | $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}+1}$ |
| $\boldsymbol{X} = \boldsymbol{S}(\boldsymbol{X})$ | $\boldsymbol{X} = \boldsymbol{S}^{-1}(\boldsymbol{X})$ |
| $\boldsymbol{X} = \boldsymbol{R}_{\leftarrow}(\boldsymbol{X})$ | $\boldsymbol{X} = \boldsymbol{R}_{\rightarrow}(\boldsymbol{X})$ |
| $\boldsymbol{X} = \boldsymbol{M} \times \boldsymbol{X}$ | $\boldsymbol{X} = \boldsymbol{M}^{-1} \times \boldsymbol{X}$ |
| $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}}$ | $\boldsymbol{X} = \boldsymbol{X} \oplus (\boldsymbol{M}^{-1} \times \boldsymbol{k}_{\boldsymbol{R}})$ |
| $\boldsymbol{X} = \boldsymbol{S}(\boldsymbol{X})$ | $\boldsymbol{X} = \boldsymbol{S}^{-1}(\boldsymbol{X})$ |
| $\boldsymbol{X} = \boldsymbol{R}_{\leftarrow}(\boldsymbol{X})$ | $\boldsymbol{X} = \boldsymbol{R}_{\rightarrow}(\boldsymbol{X})$ |
| $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}+1}$ | $\boldsymbol{X} = \boldsymbol{X} \oplus \boldsymbol{k}_{\boldsymbol{R}-1}$ |

Из сопоставления столбцов таблицы 3 становится очевидно, что алгоритмическая структура прямого и обратного преобразований идентична. Результат легко обобщается на произвольное число раундов. Таким образом, в алгоритме Rijndael процедуры зашифрования и расшифрования алгоритмически идентичны и различаются только в следующих деталях:

- в обратном преобразовании используется вектор замен, обратный в операционном смысле вектору замен прямого преобразования;
- в обратном преобразовании число байтов, на которые сдвигается каждая строка матрицы данных в операции построчного байтового сдвига другое;
- в обратном преобразовании в шаге матричного умножения блок данных умножается слева на матрицу, обратную той, что используется при прямом преобразовании; эта обратная матрица равна:

$$\boldsymbol{M}^{-1} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix}$$

- в обратном преобразовании ключевые элементы используются в обратном порядке, и, кроме того, все элементы за исключением первого и последнего, должны быть умножены слева на матрицу $\boldsymbol{M}^{-1}$.

Таким образом, аналогично ГОСТу, в алгоритме Rijndael возможно совместить реализацию процедур за- и расшифрования как при аппаратной, так и при программной реализации.

## Выработка ключевых элементов.

В отечественном стандарте шифрования для выработки тридцати двух 32-битовых ключевых элементов из 256-битового ключа применен очень простой подход. Ключ интерпретируется как массив, состоящий из восьми ключевых элементов:

$$\boldsymbol{K} = (\boldsymbol{k}_1, \boldsymbol{k}_2, \boldsymbol{k}_3, \boldsymbol{k}_4, \boldsymbol{k}_5, \boldsymbol{k}_6, \boldsymbol{k}_7, \boldsymbol{k}_8).$$

Эти элементы используются на раундах шифрования – ключ «просматривается» три раза в прямом порядке и один раз в обратном, в итоге каждый ключевой элемент используется ровно четыре раза. В следующей ниже таблице 4 приведено соответствие номера раунда и используемого на раунде ключевого элемента:

**Таблица 4. Порядок использования ключевых элементов в цикле зашифрования ГОСТа.**

| № раунда | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ключ. элемент | $\boldsymbol{k}_1$ | $\boldsymbol{k}_2$ | $\boldsymbol{k}_3$ | $\boldsymbol{k}_4$ | $\boldsymbol{k}_5$ | $\boldsymbol{k}_6$ | $\boldsymbol{k}_7$ | $\boldsymbol{k}_8$ | $\boldsymbol{k}_1$ | $\boldsymbol{k}_2$ | $\boldsymbol{k}_3$ | $\boldsymbol{k}_4$ | $\boldsymbol{k}_5$ | $\boldsymbol{k}_6$ | $\boldsymbol{k}_7$ | $\boldsymbol{k}_8$ |
| № раунда | 17 | 18 | 18 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 |
| Ключ. элемент | $\boldsymbol{k}_1$ | $\boldsymbol{k}_2$ | $\boldsymbol{k}_3$ | $\boldsymbol{k}_4$ | $\boldsymbol{k}_5$ | $\boldsymbol{k}_6$ | $\boldsymbol{k}_7$ | $\boldsymbol{k}_8$ | $\boldsymbol{k}_8$ | $\boldsymbol{k}_7$ | $\boldsymbol{k}_6$ | $\boldsymbol{k}_5$ | $\boldsymbol{k}_4$ | $\boldsymbol{k}_3$ | $\boldsymbol{k}_2$ | $\boldsymbol{k}_1$ |

В шифре Rijndael используется чуть более сложная схема, учитывающая возможное различие в размерах блока алгоритма и ключа. Существуют два алгоритма генерации последовательности ключевых элементов – для ключа размером 128/192 бита и для ключа размером 256 бит, которые, впрочем, достаточно похожи и различаются довольно незначительно. Ключ и ключевая последовательность представляются в виде векторов 4-х байтовых слов, и начальный участок последовательности заполняется словами из ключа, – точно так же, как в ГОСТе. Последующие слова ключевой последовательности вырабатываются по рекуррентному соотношению группами, кратными размеру ключа.

Первое 4-байтовое слово такой группы вырабатывается с использованием достаточно сложного нелинейного преобразования, остальные – по простому линейному соотношению:

$$\boldsymbol{w_i} = \begin{cases} \boldsymbol{w_{i-N_K}} \oplus \boldsymbol{G}(\boldsymbol{w_{i-1}}), \text{ при } \boldsymbol{i} \bmod \boldsymbol{N_K} = 0 \\ \boldsymbol{w_{i-N_K}} \oplus \boldsymbol{w_{i-1}}, \text{ при } \boldsymbol{i} \bmod \boldsymbol{N_K} \neq 0 \end{cases},$$

где $\boldsymbol{N_k}$ – число 32-битовых слов в ключе (4 или 6, для восьми используется другая схема), $\boldsymbol{G}(\ldots)$ – нелинейное преобразование 32-биттовых слов – включает байтовый сдвиг, побайтовую подстановку по вектору замен и побитовое сложение по модулю 2 с вектором, зависящим от номера вырабатываемой группы элементов:

$$\boldsymbol{G}(\boldsymbol{x}) = \boldsymbol{S}(\boldsymbol{R}_8^{\leftarrow}(\boldsymbol{x})) \oplus \boldsymbol{P}(\boldsymbol{i}/\boldsymbol{N_k}),$$

где $\boldsymbol{S}$ – это описанная выше функция побайтовой замены, $\boldsymbol{R}_8^{\leftarrow}$ – операция циклического сдвига аргумента на восемь бит влево, а $\boldsymbol{P}(\boldsymbol{i}/\boldsymbol{N_k})$ – 4-байтовое слово, конструируемое особым образом и не зависящее от ключа. Использование преобразования $\boldsymbol{G}(\boldsymbol{x})$ вносит сложность и нелинейность в схему выработки ключевых элементов, затрудняя тем самым криптоанализ шифра.

Полученные из описанного выше потока 4-байтовые слова группируются в ключевые элементы необходимого размера, равного размеру шифруемого блока, и используются на раундах шифрования как описано в предыдущих разделах.

Как видно из изложенного выше, алгоритм выработки ключевой последовательности в шифре Rijndael является более сложным, чем в ГОСТе. Тем не менее, он остается достаточно простым и эффективным и не вносит сколь-нибудь существенного вклада в общий объем вычислительных затрат на шифрование – скорость шифрования с вычислением ключевых элементов «на лету» незначительно меньше скорости шифрования с использованием заранее подготовленных ключевых элементов.

## Выбор узлов замен и других констант.

Важнейшими константами ГОСТа являются долговременные ключевые элементы – узлы замен. Они не зафиксированы в стандарте, а поставляются специализированными организациями, снабжающими пользователей шифра ключевой информацией. В силу этого какие-либо сведения о критериях проектирования узлов замен отсутствуют. Из общих соображений можно заметить, что, скорее всего, узлы замен вырабатываются с использованием одной из методик конструирования узлов (например, с использованием так называемых бент-функций [5]), а затем оцениваются по нескольким критериям, и обладающие недостаточным качеством узлы бракуются. Среди критериев оценки, вероятно, присутствуют следующие:

- сложность и нелинейность булевых функций, описывающих узлы;
- дифференциальная характеристика узлов замен;
- линейная характеристика узлов замен;

В отличие от разработчиков ГОСТа, авторы шифра Rijndael не стали скрывать критерии проектирования вектора замен. При его конструировании помимо тривиальных требований обратимости и простоты описания были приняты во внимание следующие соображения:

- минимизация самой большой по величине характеристики корреляции между линейными комбинациями входных и выходных битов (определяет устойчивость к линейному криптоанализу);

- минимизация наибольшего нетривиального значения в таблице EXOR (определяет устойчивость к дифференциальному криптоанализу);
- сложность алгебраического выражения, описывающего узел, в $GF(2^8)$.

Операция байтовой замены в алгоритме Rijndael описывается следующим уравнением:

$$\boldsymbol{S}(\boldsymbol{X}) = (\boldsymbol{x}^7 + \boldsymbol{x}^6 + \boldsymbol{x}^2 + \boldsymbol{x}) + \boldsymbol{X}^{-1} \cdot (\boldsymbol{x}^7 + \boldsymbol{x}^6 + \boldsymbol{x}^5 + \boldsymbol{x}^4 + 1) \; mod \; (\boldsymbol{x}^8 + 1).$$

Это преобразование начинается с мультипликативной инверсии заменяемого байта в описанном выше конечном поле $GF(2^8)$, – значение 00 при этом меняется на самого себя, – затем результат подвергается аффинному преобразованию. Полиномы этого преобразования выбраны таким образом, чтобы у итогового отображения отсутствовали точки неподвижности ($\boldsymbol{S}(\boldsymbol{X}) = \boldsymbol{X}$) и «антинеподвижности» ($\boldsymbol{S}(\boldsymbol{X}) = \sim\boldsymbol{X}$). Здесь знаком «~» обозначена операция побитового инвертирования.

Авторы алгоритма Rijndael отметили, что если данный узел замен вызывает сомнения в его качестве и в отсутствии «потайного хода», то он может быть заменен на другой. Более того, структура шифра и количество раундов выбраны таким образом, чтобы даже в случае случайно выбранного вектора замен шифр был устойчив против дифференциального и линейного криптоанализа.

Из других важных констант алгоритма Rijndael необходимо отметить матрицу $\boldsymbol{M}$, на которую производится умножение в ходе раундового преобразования. Целью этого шага является диффузия изменения в одном байте на весь столбец матрицы. При выборе матрицы $\boldsymbol{M}$ помимо традиционных требований обратимости и простоты описания были приняты во внимание следующие желаемые характеристики преобразования:

- линейность в поле GF(2);
- достаточный уровень диффузии;
- скорость реализации на 8-битовых процессорах;

Данный шаг преобразования может также быть представлен как умножение столбцов преобразуемой матрицы данных, интерпретируемых как полиномы третьей степени с коэффициентами из поля $GF(2^8)$, на другой полином с коэффициентами из этого же поля, по модулю полинома $(\boldsymbol{x}^4+1)$. Согласно 3-му из перечисленных выше требований, коэффициенты множителя должны быть как можно меньше. Авторы шифра выбрали следующий полином:

$$\boldsymbol{c}(\boldsymbol{x}) = 03 \cdot \boldsymbol{x}^3 + 01 \cdot \boldsymbol{x}^2 + 01 \cdot \boldsymbol{x} + 02.$$

Та же процедура, записанная в матричной форме, эквивалентна умножению слева на матрицу $\boldsymbol{M}$ – циркулянт полинома $\boldsymbol{c}(\boldsymbol{x})$.

Таким образом, в отличие от ГОСТ 28147-89, раунд которого составлен из традиционных для криптоалгоритмов «первой волны» операций, – подстановок в малых битовых группах, битовых перестановок (сдвигов) и аддитивных операций комбинирования элементов данных, – при конструировании шифра Rijndael широко использован алгебраический подход. Это касается главным образом двух основных преобразований шифра – байтовой замены и операции перемешивания столбцов матрицы данных посредством ее умножения слева на матрицу $\boldsymbol{M}$.

## Диффузионные характеристики.

Рассмотрение характеристик стойкости алгоритмов начнем с исследования характеристик диффузии. Сначала рассмотрим распространение изменений в исходных данных на выходное значение функции шифрования алгоритма ГОСТ. Предположим, что во входном

32-битовом значении изменен один бит. Первая операция функции шифрования – это сложение по модулю $2^{32}$, т.е. с переносом из младших разрядов в старшие. Теоретически, изменение самого младшего бита операнда может привести к изменению всех битов суммы. Однако вероятность этого события чрезвычайно мала. Можно показать, что при условии равновероятного и независимого распределения битов операндов на множестве {0,1} вероятность события, что влияние одного бита операнда распространяется влево ровно на $n$ битов результата, равна приблизительно $2^{-n}$. Это означает, что если изменить значение одного бита операнда на противоположное, то помимо соответствующего ему бита результата, который инвертируется в любом случае, ровно $n$ битов результата, находящихся левее инвертированного, также поменяют значение на противоположное с указанной выше вероятностью. Исходя из этого можно констатировать, что при сложении двух чисел по модулю $2^{32}$ практическое значение имеет только влияние бита операнда на не более чем четыре старших бита результата.

Предположим теперь, что в аргументе функции шифрования изменил значение один из битов. Это приведет к тому, что после суммирования с ключевым элементом это изменение распространится на группу из 5 битов суммы. Следующей операцией в функции шифрования является замена в 4-битовых группах. Группа из пяти смежных битов суммы покрывается ровно двумя такими группами. Следовательно, после выполнения замены изменение распространится на 8 битов, входящих в эти две группы. Последующее вращение на 11 бит влево не изменяет числа затронутых битов, однако изменяет их положение в слове – эти 8 битов оказываются распределенными по трем смежным 4-битовым группам, как показано ниже на рисунке 3. На рисунке каждый прямоугольник соответствует 4-битовой группе, затронутые изменением биты выделены серым цветом.

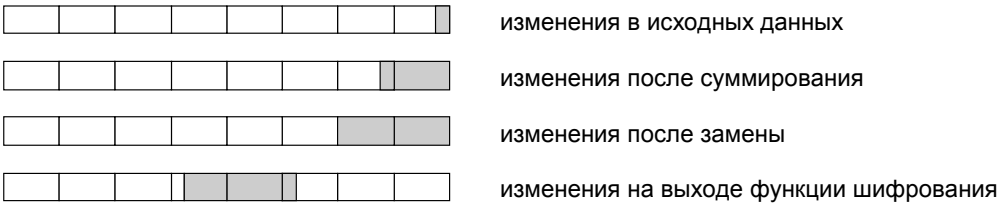


Рис. 3. Диффузия изменения в исходных данных через функцию шифрования алгоритма ГОСТ28147-89.

Теперь проследим изменение в исходных данных через несколько раундов шифрования в предположении, что инвертирован бит из младшей половины блока. Соответствующие диаграммы приведены ниже на рисунке 4.

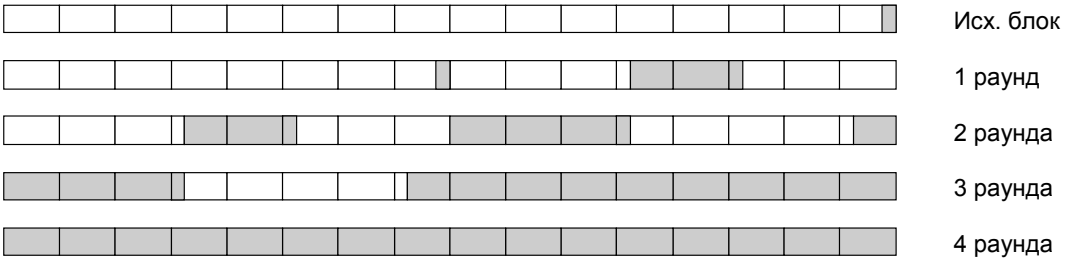


Рис. 4. Диффузия изменения в исходных данных через несколько раундов шифрования алгоритма ГОСТ28147-89.

Как видно из рисунка 4, изменения в одном бите распространяются на весь шифруемый блок после четырех раундов шифрования, если в исходном блоке изменен бит из младшей половины. Если же будет инвертирован бит из старшей половины, то это число

увеличится на 1 и станет равно 5. Таким образом, за 32 раунда шифрования, предусмотренные ГОСТом, данные успевают полностью перемешаться приблизительно $32:5 \approx 6$ раз.

Теперь рассмотрим диффузионные характеристики алгоритма Rijndael. Первая операция раунда шифрования этого алгоритма – побитовое суммирование с ключом по модулю 2 – не приводит к выходу изменения за пределы одного бита. Следующая операция – замена по таблице – распространяет изменение в одном бите на весь байт. Следующий за ней построчный байтовый сдвиг не изменяет ничего в этой картине. И, наконец, завершающая операция раунда – перемешивание байтов в столбцах матрицы – приводит к диффузии изменения на весь столбец. Таким образом, за один раунд шифрования изменение в одном бите входных данных окажет влияние на один столбец матрицы данных. На следующем раунде шифрования эти байты в ходе операции построчного байтового сдвига будут «разведены» по разным столбцам, и в результате последующей операции перемешивания байтов в столбцах исходное изменение распространится на 4 столбца. То есть на всю матрицу данных при 128-битовом блоке данных, на 2/3 матрицы при 192-битовом блоке и на ее половину при 256-битовом блоке данных. Диаграмма диффузии в алгоритме Rijndael приведена на рисунке 5.

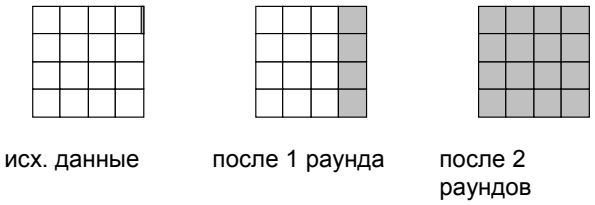


Рис. 5. Диффузия изменения в исходных данных через несколько раундов шифрования алгоритма Rijndael для 128-битового блока.

Таким образом, при шифровании 128-битовых блоков изменение в одном бите входных данных распространяется на весь блок ровно за два раунда, при большем размере блока – за три. В результате за 10-14 раундов алгоритма Rijndael данные успевают полностью перемешаться 5-7 раз.

Как видно из вышеизложенного, по показателям диффузии алгоритмы ГОСТ и Rijndael отличаются друг от друга не сильно.

## Показатели стойкости алгоритмов.

Рассмотрим устойчивость обоих алгоритмов к известным видам криптоанализа. Наиболее универсальными и эффективными для алгоритмов широкого класса являются дифференциальный и линейный виды криптоанализа.

Дать оценку устойчивости алгоритма ГОСТ28147-89 к конкретным видам криптоанализа невозможно без спецификации узлов замен, так как качество этого шифра существенным образом зависит от качества использованных узлов. Однако исследования близких по архитектуре шифров с заданными таблицами подстановок (DES) показали, что криптоанализ шифра с 16 раундами в принципе осуществим, однако требует очень большого числа исходных данных, а при 20-24 раундах становится теоретически бесполезным. ГОСТ предусматривает 32 раунда шифрования, и этого количества хватает с запасом, чтобы успешно противостоять указанным видам криптоанализа. В открытой печати отсутствуют сообщения об успешном вскрытии ГОСТа с какими-либо узлами замен, – как с тестовыми,

специфицированными в стандарте ГОСТ Р34.11-94, так и с теми, с которыми реализации ГОСТа поставлялись в коммерческие организации.

По оценкам разработчиков шифра Rijndael, уже на четырех раундах шифрования этот алгоритм приобретает достаточную устойчивость к указанным видам криптоанализа. Теоретической границей, за которой линейный и дифференциальный виды криптоанализа теряют смысл, является рубеж в 6-8 раундов в зависимости от размера блока. Согласно спецификации, в шифре предусмотрено 10-14 раундов. Следовательно, шифр Rijndael также устойчив к указанным видам криптоанализа с определенным запасом.

Таким образом, оба сравниваемых шифра обладают достаточной стойкостью к известным видам криптоанализа. В печати отсутствуют какие-либо сведения об успешных случаях вскрытия указанных шифров, а также описания процедур, которые теоретически позволили бы дешифровать сообщение с меньшими вычислительными затратами, чем полный перебор по всему ключевому пространству.

## Производительность и удобство реализации.

При оценке достижимой эффективности аппаратной реализации шифров главным критерием является количество и сложность элементарных операций, которые необходимо выполнить в цикле шифрования, а также возможность их параллельного выполнения. При оценке эффективности возможных программных реализаций главный интерес представляет реализация на 32-битовых платформах, так как 32-разрядные машины составляют в настоящее время большинство компьютерного парка человечества. Также представляет интерес реализация шифров на 8-битовых микроконтроллерах, являющихся основой технологии интеллектуальных карт. Подобные устройства могут использоваться в различных системах безналичных расчетов, становящихся все более популярными в мире, – число пользователей таких систем в последнее время растет весьма быстрыми темпами.

Российский стандарт шифрования ГОСТ 28147-89 удобен как для аппаратной, так и для программной реализации. При размере блока данных 64 бита, основная работа ведется с половинками этого блока – 32-битовыми словами, что позволяет эффективно реализовать российский стандарт шифрования на большинстве современных компьютеров.

При реализации на 32-битовых машинах наиболее трудоемкой операцией является замена. Предусмотренные ГОСТом подстановки в 4-битовых группах при программной реализации удобно попарно объединить и выполнять замену в 8-битовых группах, что существенно эффективнее. Надлежащая организация замены позволяет также избежать выполнения вращения слова на выходе функции шифрования, – если хранить узлы замены как массивы 4-байтовых слов, в которых уже выполнены необходимые сдвиги. Такая «раздутая» таблица замен потребует для своего хранения $4 \times 2^8 \times 4 = 2^{12}$ байт или 4K оперативной памяти. Одна замена реализуется за три однотактовые машинные команды: загрузка байта в индексный регистр, загрузка заменяющего значения в регистр, использование загруженного значения в операции побитового суммирования. В итоге перечисленные шаги оптимизации позволяют реализовать раунд шифрования по ГОСТ за 15 однотактовых машинных команд. С учетом возможности процессоров Intel Pentium по параллельному выполнению команд, раунд ГОСТа может быть реализован за 8 тактов работы процессора, а весь процесс шифрования – за $32 \times 8 = 256$ тактов. На процессоре Intel Pentium 200 это позволит достичь предела быстродействия шифрования примерно 6.0 Мбайт/с, в реальности эта величина будет меньше.

ГОСТ может быть также эффективно реализован на 8-битовых микроконтроллерах, поскольку составляющие его элементарные операции входят в систему команд большинства наиболее распространенных контроллеров. При этом суммирование по модулю $2^{32}$ придется

разделить на одну операцию сложения без переноса, и три операции сложения с переносом, выполняемые каскадно. Все остальные операции также легко могут быть представлены в терминах 8-байтовых операндов.

При аппаратной реализации ГОСТа один раунд предполагает последовательное выполнение трех операций над 32-битовыми аргументами: суммирование, выполняемая одновременно замена во всех восьми 4-битовых группах и побитовое суммирование по модулю 2. Циклический сдвиг не является отдельной операцией, т.к. обеспечивается простой коммутацией проводников. Таким образом, при аппаратной реализации цикл шифрования требует выполнения ста шести элементарных операций, и эта работа не может быть распараллелена.

Теперь рассмотрим особенности реализации алгоритма Rijndael. Этот алгоритм является байт-ориентированным, т.е. полностью может быть сформулирован в терминах операций с байтами. В алгоритме широко используются алгебраические операции в конечных полях, наиболее сложно реализуемой из которых является умножение в $GF(2^8)$. Непосредственное выполнение этих операций привело бы к крайне неэффективной реализации алгоритма. Однако байтовая структура шифра открывает широкие возможности по оптимизации программной реализации. Замена байта по таблице с последующим умножением на константу в конечном поле $GF(2^8)$ может быть представлена как одна замена по таблице. В прямом шифре используются три константы (01, 02, 03), и, следовательно, понадобятся три таких таблицы, в обратном – четыре (0E, 0D, 0B, 09). При надлежащей организации процесса шифрования построчный байтовый сдвиг матрицы данных можно не выполнять. При реализации на 32-битовых платформах возможно реализовать байтовую замену и умножение элемента матрицы данных на столбец матрицы $\boldsymbol{M}$ как одну замену 8 бит на 32 бита. Таким образом, преобразование одного 32-битового слова данных включает четыре байтовые замены, каждая из которых, как было отмечено выше, требует трех однотактовых машинных команд. В итоге часть раунда для одного 32-битового слова может быть реализована на процессорах Intel Pentium за 14 команд или за 7 тактов, что при 14 раундах шифрования позволяет на процессорах Intel Pentium 200 достичь теоретического предела быстродействия примерно 7.8 Мбайт/с вне зависимости от размера блока данных и ключа. Для меньшего числа раундов скорость пропорционально возрастет.

Указанная выше оптимизация потребует, однако, определенных расходов оперативной памяти. Для каждого столбца матрицы $\boldsymbol{M}$ строится свой вектор замены одного байта на 4-байтовое слово, получаем точно такую же по размеру, как и в случае ГОСТ, таблицу замен, ее размер равен $4 \times 2^8 \times 4 = 2^{12}$ байт или 4К. Далее, таблицы, используемые при зашифровании и расшифровании, различны, – это удваивает требования к оперативной памяти. Кроме того для выполнения последнего раунда расшифрования нужен отдельный узел замен, его размер равен 256 байт или 0.25К. В итоге получаем, что для 32-битовых программных реализаций шифра Rijndael необходимо 8.25 Кбайт оперативной памяти для хранения узлов замен. Для современных компьютеров на базе Intel Pentium под управлением ОС Windows 9x/NT/2000 это не выглядит чрезмерным требованием.

Байт-ориентированная архитектура алгоритма Rijndael позволяет чрезвычайно эффективно реализовать его на 8-битовых микроконтроллерах, используя только операции загрузки-выгрузки регистров, индексированного извлечения байта из памяти и побитового суммирования по модулю два. Также указанная особенность позволит выполнить эффективную программную реализацию алгоритма. Раунд шифрования требует выполнения 16 байтовых замен плюс четыре операции побитового исключающего или над 128-битовыми блоками, которые могут быть выполнены в три этапа. В итоге получаем 4 операции на раунд или 57 операций на 14-раундовый цикл шифрования с учетом «лишней» операции побитового прибавления ключа по модулю два – это примерно вдвое меньше, чем в ГОСТе.

Т.к. Rijndael обладает вдвое большим размером блока, это приводит к примерно четырехкратному преимуществу в скорости при условии аппаратной реализации на базе одной и той же технологии. Необходимо заметить, что указанная выше оценка является очень грубой.

Авторами статьи была проведена оценка практических характеристик быстродействия программных реализаций сравниваемых шифров на платформе Intel Pentium, для шифра Rijndael рассматривался вариант с 14 раундами. Для каждого из алгоритмов на языке Си был написан эквивалент функции зашифрования одного блока, в котором последовательность раундов была развернута в линейный код, – это позволяет достичь максимального быстродействия. В эквивалентах использовалась случайная ключевая информация и случайные узлы замен, однако на быстродействие реализации это никоим образом не влияет, так как скорость выполнения использованных команд процессора не зависит от их операндов. Функции, выполняющие зашифрование, вызывалась по 10 000 000 раз, и измерялось время их выполнения, которое затем пересчитывалось в показатель быстродействия. Для компиляции и построения исполняемого модуля использовался компилятор Intel C++ v4.5, так как он позволяет получить код с максимальным быстродействием. Были опробованы также компиляторы MS Visual C++ 6.0, Borland C++ v5.5 и GNU C++ v2.95.2, однако полученный с их использованием код был заметно медленнее. Код оптимизировался для процессоров Intel Pentium и Intel Pentium Pro/II/III. С помощью полученных тестовых задач было замерено быстродействие реализаций шифров на процессорах Intel Pentium 166 МГц и Intel Pentium III 500 МГц. Результаты измерений приведены в следующей ниже таблице 5.

**Таблица 5. Показатели быстродействия реализаций сравниваемых алгоритмов на языке Си.**

| | **ГОСТ 28147-89** | **Rijndael, 14 раундов** |
|---|---|---|
| **Pentium 166** | 2.04 Мбайт/с | 2.46 Мбайт/с |
| **Pentium III 500** | 8.30 Мбайт/с | 9.36 Мбайт/с |

Таким образом, рассмотренные алгоритмы обладают сопоставимыми характеристиками быстродействия при реализации на 32-битовых платформах. На 8-битовых платформах картина, вероятно, сходная. Что касается аппаратной реализации, то в отличие от ГОСТа, Rijnael позволяет достичь высокой степени параллелизма при выполнении шифрования, оперирует блоками меньшего размера и содержит меньшее число раундов, в силу чего его аппаратное воплощение может оказаться существенно более быстрым. Если судить по длине наибольшего пути в сетевом представлении обоих алгоритмов, его преимущество примерно четырехкратное.

## Выводы.

Проведенное выше сопоставление параметров алгоритмов шифрования ГОСТ 28147-89 и Rijndael показало, что несмотря на существенное различие в архитектурных принципах этих шифров, их основные рабочие параметры сопоставимы. Исключением является то, что, по всей вероятности, Rijnael будет иметь значительное преимущество в быстродействии перед ГОСТом при аппаратной реализации на базе одной и той же технологии. По ключевым для алгоритмов такого рода параметрам криптостойкости ни один из алгоритмов не обладает существенным преимуществом, также примерно одинаковы скорости оптимальной программной реализации для процессоров Intel Pentium, что можно экстраполировать на все современные 32-разрядные процессоры. Из сказанного можно сделать вывод, что отечественный стандарт шифрования соответствует требованиям, предъявляемым к современным

шифрам, и может оставаться стандартом еще достаточно долгое время. Очевидным шагом в его оптимизации может являться переход от замен в 4-битовых группах к байтовым заменам, что должно еще более повысить стойкость алгоритма к известным видам криптоанализа.

## Литература.


1. ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования.
2. А.Винокуров. «ГОСТ не прост, а очень прост». М., «Монитор», 1995, №1, с.60-73.
3. А.Винокуров. Алгоритм шифрования ГОСТ 28147-89, его использование и реализация для компьютеров платформы Intel x86. Работа на правах рукописи, доступна на веб-сайте http://www.enlight.ru/crypto.
4. RIJNDAEL description. Submission to NIST by Joan Daemen, Vincent Rijmen. http://csrc.nist.gov/encryption/aes/round1/docs.htm
5. А.Винокуров, Э.Применко. Новые подходы в построении блочных шифров с секретным ключом. Труды XXVI международной конференции по информационной технологии в образовании, бизнесе и т.д.. Гурзуф–Ялта, 20–30 мая 1999г.
6. А.Варфоломеев и др. Блочные криптосистемы. Основные свойства и методы анализа стойкости. М., МИФИ, 1998.