Трехмерные алгоритмы стохастического преобразования данных, ориентированные на реализацию с использованием гибридных суперкомпьютерных технологий

Three-dimensional data stochastic transformation algorithms for hybrid supercomputer implementation

М. А. Иванов, Н. П. Васильев, И. В. Чугунков

M. A. Ivanov, N. P. Vasilyev, I. V. Chugunkov
National Research Nuclear University MEPhI

Рассматриваются трехмерные стохастические алгоритмы преобразования данных, ориентированные на решение задач защиты информации. Особенностью предлагаемых решений является высокая степень параллелизма.

3*D* algorithms for data stochastic transformation oriented to the information security problems are considered. The most important feature of these algorithms is high degree of parallelism.

Ключевые слова: Генератор псевдослучайных чисел, архитектура «Квадрат», архитектура «Куб», Rijndael, *S*-блок, гибридные вычислительные системы, CPU, GPU.

Keywords: Pseudo-random number generator, Hybrid computing systems, Rijndael, *S*-box, CPU, GPU, Square architecture, Cube architecture.

Введение

Важнейшим элементом любой системы защиты информации (СЗИ) являются генераторы псевдослучайных чисел (ГПСЧ). Можно выделить следующие основные функции ГПСЧ в СЗИ:

- генерация ключевой информации и паролей пользователей;
- формирование гаммы при поточном шифровании данных;
- формирование случайных запросов при аутентификации субъектов информационного взаимодействия;
- внесение неопределенности в работу средств и объектов защиты и др.

Непредсказуемые ГПСЧ являются основой стохастических методов, с использованием которых решаются такие задачи, как обеспечение секретности и конфиденциальности информации, подтверждение подлинности субъектов информационного взаимодействия, контроль хода выполнения программ, обеспечение целостности объектов (сообщений, массивов данных) информационного взаимодействия. Стохастическими являются все протоколы защищенного взаимодействия удаленных абонентов. К генераторам, ориентированным на решение задач защиты информации, предъявляются наиболее жесткие требования по непредсказуемости, статистической безопасности и периоду формируемых последовательностей.

1. Постановка задачи

Основной проблемой при проектировании ГПСЧ является трудно разрешимое противоречие между качеством формируемых псевдослучайных последовательностей, с одной стороны, и эффективностью программной и аппаратной реализации, определяющей быстродействие генераторов, с другой стороны. Наиболее рациональным решением с точки зрения этих критериев следует признать ГПСЧ, нелинейные функции обратной связи или выхода которых суть функции итеративного блочного шифрования.

На рис. 1 показана общая структурная схема ГПСЧ. На практике используются две ее разновидности − схема OFB (Output Feedback − обратная связь по выходу), когда качество генератора определяется нелинейной функцией обратной связи, и двухступенчатая схема CTR (Counter), когда первая ступень (счетчик) обеспечивает максимальный период формируемой последовательности, а качество генератора определяется нелинейной функцией выхода второй ступени.

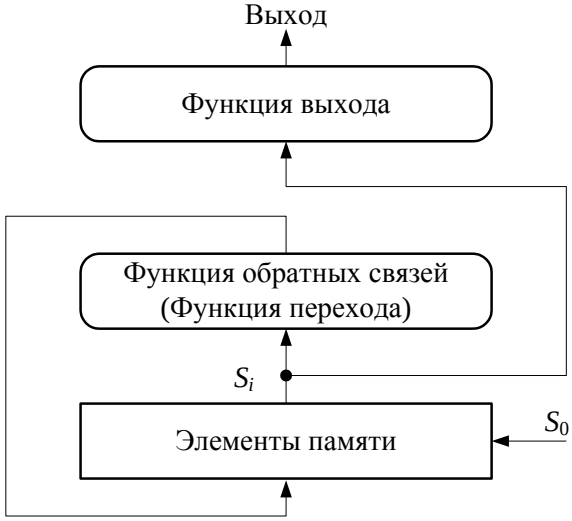

Рис. 1. Структура ГПСЧ. S_0 – начальное состояние генератора, S_i – i-е состояние элементов памяти генератора.

Одной из перспективных современных архитектур для построения качественных ГПСЧ является архитектура «Квадрат», предложенная авторами криптоалгоритмов Square и Rijndael; при этом последний в 2001 г. победил в конкурсе на принятие нового Американского стандарта криптозащиты AES (Advanced Encryption Standard) [1, 2]. Наиболее ярко достоинства архитектуры «Квадрат» проявляются в версии стандарта AES-128. В этом случае все входные и выходные блоки данных, все промежуточные результаты преобразований, все раундовые ключи представляются в виде квадратного массива байтов 4×4.

Основными достоинствами стохастических алгоритмов с архитектурой «Квадрат» являются:

- простая и понятная логика работы, удобная для анализа и обоснования стойкости;
- простое обоснование наличия свойств рассеивания и перемешивания информации при использовании двухраундовой конструкции, что в свою очередь облегчает обоснование требуемого числа раундов преобразования;
- высокий параллелизм на уровне инструкций;

– байт-ориентированная структура.

В состав раунда AES-128 входят преобразования замены байтов (*SubBytes*), циклического сдвига строк (*ShiftRows*), перемешивания столбцов (*MixColumns*) и сложения с раундовым ключом (*AddRoundKey*). При выполнении преобразования перемешивания байты столбца рассматриваются как коэффициенты многочлена $A(x)$ степени 3 над полем $GF(2^8)$. Суть операции перемешивания столбца − умножение по модулю $x^4 + 1$ многочлена $A(x)$ на многочлен $'03'x^3 + x^2 + x + '02'$.

Целью работы является дальнейшее развитие этой архитектуры.

2. Направления совершенствования архитектуры «Квадрат»

Можно выделить следующие пути совершенствования архитектуры «Квадрат»:

– замена раундовой операции сдвига строк *ShiftRows* на операцию перемешивания строк *MixRows* позволит обеспечить полное рассеивание и перемешивание информации за один раунд;
– совместное использование архитектур «Петля Фейстеля» и «Квадрат», например, использование в качестве раундовой функции «Петли Фейстеля» операций сложения с раундовым ключом, замены байтов, перемешивания строк и перемешивания столбцов;
– переход от архитектуры «Квадрат» к архитектуре «Куб» [2, 3].

3. Генератор псевдослучайных чисел DOZEN

Рассмотрим нелинейный алгоритм ГПСЧ с архитектурой «Куб», названный авторами DOZEN (dozen − англ. дюжина; название обусловлено тем, что алгоритм имеет двенадцать основных шагов, как будет раскрыто далее). Основные принципы, лежащие в основе работы данного ГПСЧ:

– представление входных и выходных блоков данных, всех промежуточных результатов преобразований и раундовых ключей (RoundKeys) K_0, K_1, K_2, K_3 в виде кубического массива байтов $4 \times 4 \times 4$ (рис. 2);
– определение понятия слоя (Layer) − квадратного массива байтов 4×4 (рис. 3);
– представление i-го раундового ключа в виде четырех подключей (RoundSubKeys) K_{i0}, K_{i1}, K_{i2}, K_{i3}, $i = 1, 2, 3$, каждый из которых суть квадратный массив байтов 4×4;
– трехмерное преобразование блока данных по слоям вдоль осей *x*, *y*, *z*;
– включение в состав операции преобразования слоя (T_Layer) четырех шагов – замены байтов (*SubBytes*), перемешивания строк (*MixRows*), перемешивания столбцов (*MixColumns*), сложения (XOR) с раундовым подключом (*AddRoundSubKey*);
– использование при выполнении преобразований *MixRow* и *MixColumn* операции, использующейся в криптоалгоритме Rijndael при реализации преобразования *MixColumn*.

Последовательность преобразования блока данных размером 512 бит ($4 \times 4 \times 4 \times 8$), имеющего структуру, показанную на рис. 2, где $a_{x,y,z}$, $x = 0, 1, 2, 3$, $y = 0, 1, 2, 3$, $z = 0, 1, 2, 3$, – байты:

1) разбиение блока данных на слои (Layers) L_{x0}, L_{x1}, L_{x2}, L_{x3} вдоль оси *x* (рис. 3);

2) первый раунд: стохастическое преобразование слоев (*T_Layer*) L_{x0}, L_{x1}, L_{x2}, L_{x3} путем выполнения для каждого слоя L_{xk} шестнадцати (по числу байтов) операций *SubByte*, четырех (по числу строк) операций *MixRow*, четырех (по числу столбцов) операций *MixColumn* и операции *AddRoundSubKey*;
3) разбиение блока данных на слои L_{y0}, L_{y1}, L_{y2}, L_{y3} вдоль оси *y* (рис. 3);
4) второй раунд: стохастическое преобразование слоев (*T_Layer*) L_{y0}, L_{y1}, L_{y2}, L_{y3} путем выполнения для каждого слоя L_{yk} шестнадцати (по числу байтов) операций *SubByte*, четырех (по числу строк) операций *MixRow*, четырех (по числу столбцов) операций *MixColumn* и операции *AddRoundKey*;
5) разбиение блока данных на слои L_{z0}, L_{z1}, L_{z2}, L_{z3} вдоль оси *z* (рис. 3);
6) третий раунд: стохастическое преобразование слоев (*T_Layer*) L_{z0}, L_{z1}, L_{z2}, L_{z3} путем выполнения для каждого слоя L_{zk} шестнадцати (по числу байтов) операций *SubByte*, четырех (по числу строк) операций *MixRow*, четырех (по числу столбцов) операций *MixColumn* и операции *AddRoundKey*.

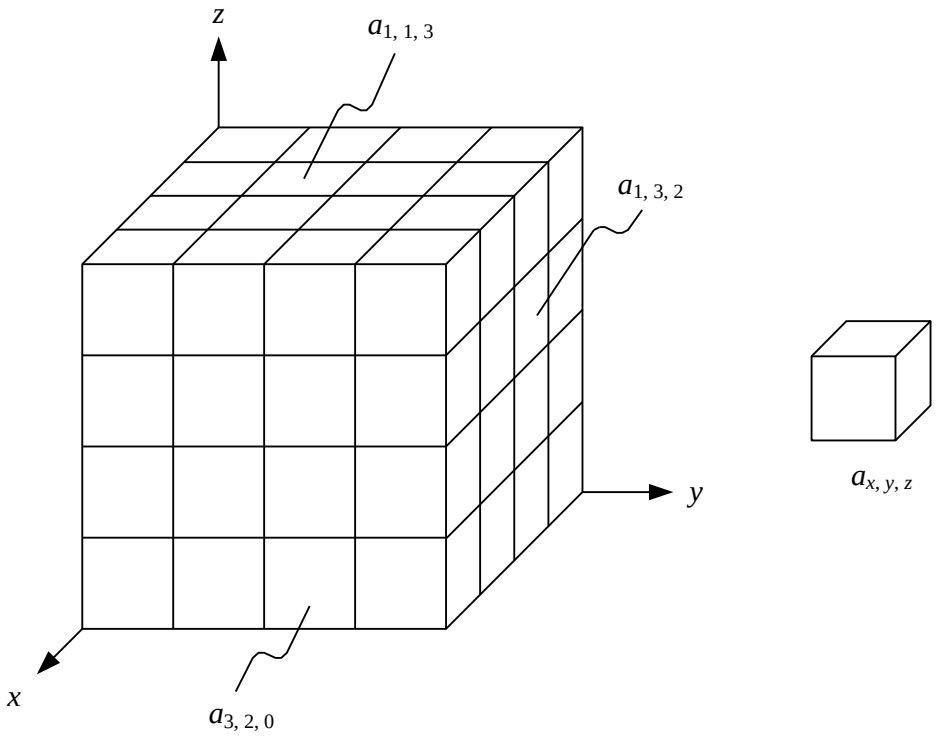

Рис. 2. ГПСЧ DOZEN: блок данных и отдельный байт блока данных.

Окончательно, получаем следующую последовательность 3D-преобразований нелинейной функции ГПСЧ (рис. 4,*а*):

Шаг 0. Сложение с раундовым ключом K_0 (*AddRoundKey* K_0).

Первый раунд:

Шаг 1. Преобразование слоя L_{x0} (*T_Layer* L_{x0}).

Шаг 2. Преобразование слоя L_{x1} (*T_Layer* L_{x1}).

Шаг 3. Преобразование слоя L_{x2} (*T_Layer* L_{x2}).

Шаг 4. Преобразование слоя L_{x3} (*T_Layer* L_{x3}).

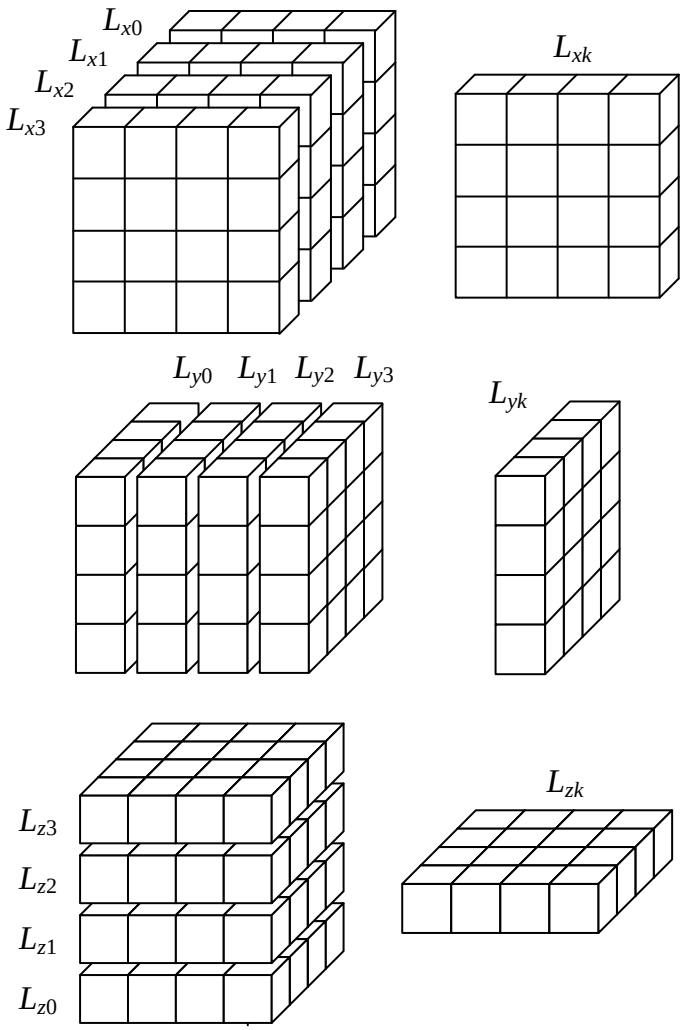

Рис. 3. ГПСЧ DOZEN: разделение на слои вдоль осей *x*, *y*, z и отдельные слои L_{xk}, L_{yk}, L_{zk}.

Второй раунд:
Шаг 5. Преобразование слоя L_{y0} ($T\_Layer\ L_{y0}$).
Шаг 6. Преобразование слоя L_{y1} ($T\_Layer\ L_{y1}$).
Шаг 7. Преобразование слоя L_{y2} ($T\_Layer\ L_{y2}$).
Шаг 8. Преобразование слоя L_{y3} ($T\_Layer\ L_{y3}$).
Третий раунд:
Шаг 9. Преобразование слоя L_{z0} ($T\_Layer\ L_{z0}$).
Шаг 10. Преобразование слоя L_{z1} ($T\_Layer\ L_{z1}$).
Шаг 11. Преобразование слоя L_{z2} ($T\_Layer\ L_{z2}$).
Шаг 12. Преобразование слоя L_{z3} ($T\_Layer\ L_{z3}$).

На рис. 4,*б* показан пример построения ГПСЧ по схеме CTR, т.е. пример использования преобразования DOZEN в качестве функции выхода ГПСЧ.

4. Реализация ГПСЧ DOZEN с помощью гибридных вычислительных технологий

В последние годы все большую популярность завоевывают гибридные вычислительные системы, сочетающие удобство классических вычислений на цен-

тральных процессорах (CPU) с массово-параллельными вычислениями на графических процессорах (GPU) [4, 5]. Особенностью GPU является наличие большого числа (десятки и сотни) вычислительных ядер, работающих параллельно. В задачах, допускающих распараллеливание обработки потока исходных данных, выигрыш в производительности для системы CPU/GPU составляет до нескольких десятков раз по сравнению с классической CPU-системой. Многие из наиболее производительных современных суперкомпьютеров также имеют гибридную архитектуру CPU/GPU.

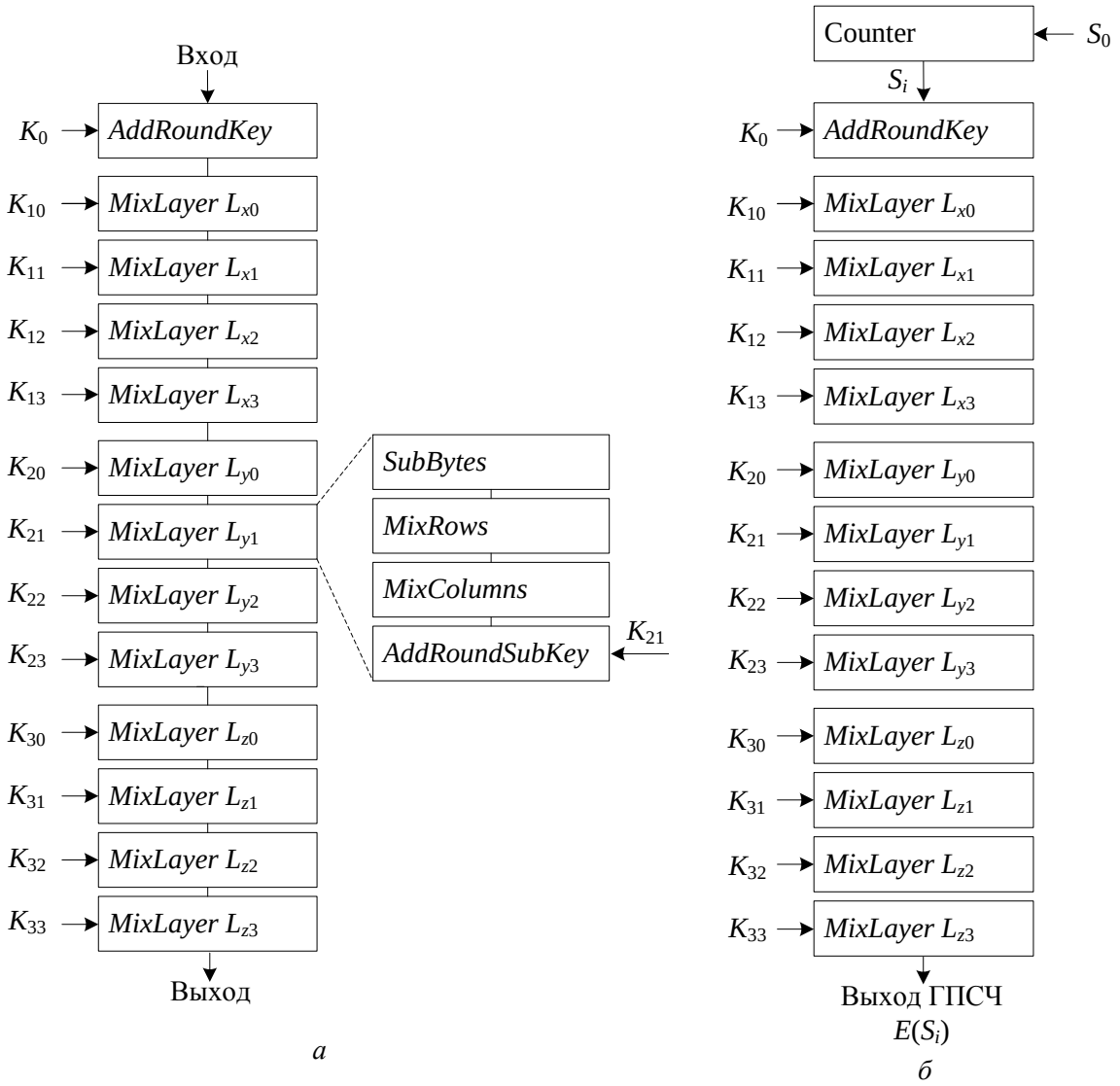

Рис. 4. Трехмерное преобразование DOZEN: *а* – последовательность 3*D*-преобразований; *б* – пример построения ГПСЧ по схеме CTR.

В гибридных системах CPU решает задачи управления выполнением программы в целом и проведения не очень «тяжелых» вычислений; наиболее критичные по производительности участки программы оформляются в виде специальных функций-ядер (kernel), которые запускаются на GPU. Современные производители графических процессоров, в частности, компания NVIDIA, предоставляют разработчикам программ для систем CPU/GPU мощные инструментальные средства. Полезной и приятной особенностью таких средств является

то, что они являются бесплатными. В качестве примеров можно указать CUDA Toolkit [6] и Parallel NSight [7], которые интегрируются с современными популярными системами разработки ПО, такими как Microsoft Visual Studio и NetBeans.

Для программной реализации предложенного алгоритма ГПСЧ DOZEN наиболее целесообразной представляется технология CUDA (Compute Unified Device Architecture – вычислительная унифицированная архитектура устройств) от компании NVIDIA [4, 5]. Минимальной вычислительной единицей в CUDA является нить (thread). По сути, нить есть набор конкретных действий над элементом данных, нити группируются в пучки (warp); все нити одного пучка физически параллельно выполняются на потоковом мультипроцессоре; из потоковых мультипроцессоров состоит графический процессор. Очень важной особенностью CUDA является то, что при программировании нити образуют трехмерные структуры, именуемые блоками (block). Блоки, в свою очередь, группируются в еще более крупную многомерную структуру, именуемую сеткой (grid). Другими словами, сетка есть совокупность всех нитей, выполняющих параллельную обработку данных, и вместе с тем представляющая собой гибкую многомерную иерархическую структуру. Таким образом, CUDA-программист может оперировать с одно-, двух- или трехмерными структурами для параллельной обработки исходных данных, в том числе, комбинируя размерности этих структур.

Очевидно, что в пределах каждого раунда работы ГПСЧ DOZEN все слои могут быть обработаны параллельно, а применение CUDA позволит существенно упростить процесс разработки ПО на основе алгоритма DOZEN.

5. Трехмерный блок замены

Важнейшим элементом криптографических примитивов являются блоки замены (*S*-блоки). Именно от качества используемых *S*-блоков зависят непредсказуемость формируемых псевдослучайных последовательностей и криптостойкость алгоритмов хеширования, блочного и поточного шифрования. Рассмотрим два новых способа построения *S*-блоков, соответствующие устройства могут быть названы 2*D* и 3*D* *S*-блоками соответственно.

Алгоритм функционирования 2D S-блока. Представим входные и выходные блоки данных, а также все промежуточные результаты преобразований в виде квадратного массива битов размерностью $N \times N$, где N – разрядность используемых узлов замены. Таким образом, объем ключевой информации, однозначно определяющей логику работы каждого узла замены, равен $N \times 2^N$.

Последовательность выполнения операции $A = SubSquare[A]$ (или, кратко, $A = S_{sq}[A]$), замены квадратного массива битов A размерностью $N \times N$, имеет следующий вид (рис. 5):

1) разбиение входного массива A на N строк R_i длины N, $i = 0, 1, \ldots, (N - 1)$;
2) замена строк *SubRows*, т.е. преобразование каждого i-го N-разрядного двоичного набора R_i с использованием соответствующего узла замены S_i: $R_i = S_i[R_i]$;
3) разбиение получившегося массива $A = SubRows[A]$ на N столбцов C_i длины N
4) замена столбцов *SubColumns*, т.е. преобразование каждого i-го N-разрядного двоичного набора C_i с использованием соответствующего узла замены S_{i+N}: $C_i = S_{i+N}[C_i]$;
5) результатом замены является $A = SubColumns[A]$.

В частном случае, когда используется только одна таблица замен, т.е. $S_i = S$, получаем следующий алгоритм:

1) разбиение входного массива A на N строк R_i длины N;

2) замена строк *SubRows*, т.е. преобразование каждого i-го N-разрядного двоичного набора R_i:

$$R_i = S[R_i],\ i = 0, 1, ..., (N - 1);$$

3) разбиение получившегося массива $A = SubRows[A]$ на N столбцов C_i длины N;
4) замена столбцов *SubColumns*, т.е. преобразование каждого i-го N-разрядного двоичного набора C_i: $C_i = S[C_i]$;
5) результатом замены является $A = SubColumns[A]$.

Алгоритм функционирования 3D S-блока. Представим входные и выходные блоки данных, а также все промежуточные результаты преобразований в виде кубического массива битов размерностью $N \times N \times N$, где N – разрядность используемых узлов замены. Таким образом, объем ключевой информации, однозначно определяющей логику работы каждого узла замены, равен $N \times 2^N$.

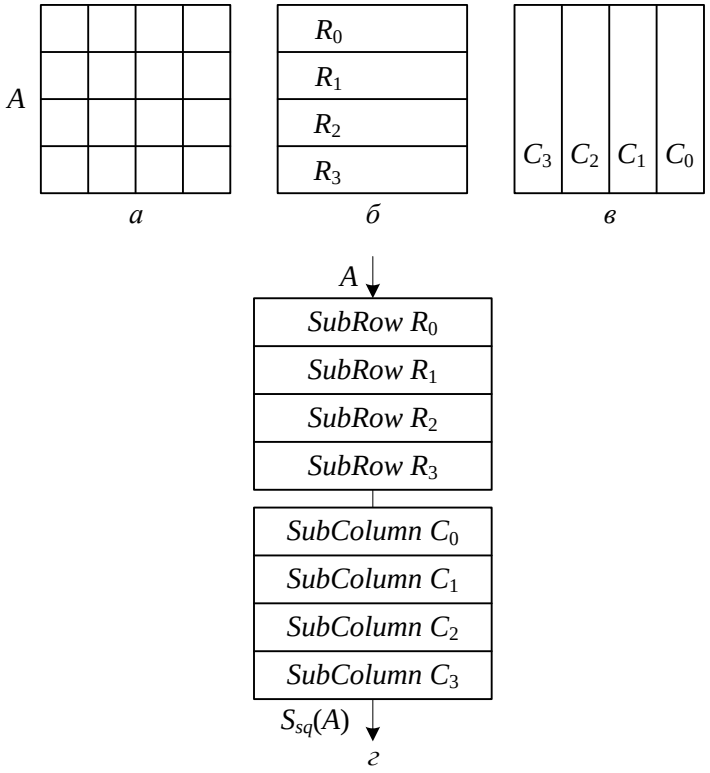

Рис. 5. 2*D S*-блок при $N = 4$: *а* – блок данных; *б* – деление блока данных по строкам; *в* – деление блока данных по столбцам; *г* - последовательность шагов преобразования.

Последовательность выполнения операции замены кубического массива битов $A = SubCube[A]$ (или кратко, $A = S_{cu}[A]$) размерностью $N \times N \times N$ имеет следующий вид:

1) разбиение входного массива A на N слоев L_{xi} размерностью $N \times N$ вдоль оси x, $i = 0, 1, \ldots, (N - 1)$;
2) замена слоев вдоль оси x *SubLayersX*, т.е. выполнение преобразования $L_{xi} = SubSquare[L_{xi}]$ каждого i-го слоя L_{xi} с использованием соответствующих узлов замены;
3) разбиение получившегося массива $A = SubLayersX[A]$ на N слоев L_{yi} размерностью $N \times N$ вдоль оси y;

4) замена слоев вдоль оси y $SubLayersY$, т.е. выполнение преобразования $L_{yi} = SubSquare[L_{yi}]$ каждого i-го слоя L_{yi} с использованием соответствующих узлов замены;
5) разбиение получившегося массива $A = SubLayersY[A]$ на N слоев L_{zi} размерностью $N \times N$ вдоль оси z;
6) замена слоев вдоль оси z $SubLayersZ$, т.е. выполнение преобразования $L_{zi} = SubSquare[L_{zi}]$ каждого i-го слоя L_{zi} с использованием соответствующих узлов замены;
7) результатом замены является $A = SubLayersZ\ [A]$.

Наиболее очевидное назначение предлагаемых алгоритмов – преобразование N^2- и N^3-разрядных блоков данных с использованием таблицы замен размерности $N \times 2^N$.

Выводы

Предложен трехмерный алгоритм стохастического преобразования данных DOZEN. Суть предлагаемого решения – использование архитектуры «Куб» при построении нелинейных функций обратной связи (режим OFB) или выхода (режим Счетчика) ГПСЧ. Описаны операции преобразования слоев блока данных. В состав результирующего 3D-преобразования входят нулевой шаг сложения с раундовым ключом и 12 шагов 2D-преобразований слоев блока данных: по четыре на каждую из координат x, y и z. Особенностями предложенного преобразования является байт-ориентированная структура и высокая степень параллелизма на уровне инструкций, что позволит достичь высокой производительности работы алгоритма для гибридных вычислительных систем CPU/GPU на основе технологии CUDA.

Та же идея использована для построения 2D и 3D блоков замены. Суть предлагаемых решений – стохастическое преобразование N^2- и N^3-разрядных блоков данных с использованием таблиц замен размерности $N \times 2^N$.

Наиболее очевидные области использования рассмотренных алгоритмов, помимо генерации непредсказуемых псевдослучайных последовательностей, это построение криптографических примитивов хеширования, блочного и поточного шифрования.

Список литературы

1. *Daemen J., Rijmen V.* AES Proposal: Rijndael. [Электронный ресурс] : URL http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf.

2. *Иванов М.А., Ковалев А.В., Чугунков И.В. и др.* Стохастические методы защиты информации в компьютерных системах и сетях. М.: КУДИЦ-ПРЕСС, 2009.

3. *Иванов М.А., Васильев Н.П., Чугунков И.В. и др.* Трехмерный генератор псевдослучайных чисел, ориентированный на реализацию в гибридных вычислительных системах // Вестник НИЯУ МИФИ, 2012, № 2.

4. *Боресков А.В., Харламов А.А.* Основы работы с технологией CUDA. М.: ДМК Пресс, 2011.

5. CUDA Zone. [Электронный ресурс] : URL http://developer.nvidia.com/ category/zone/cuda-zone

6. CUDA Toolkit. [Электронный ресурс] : URL http://developer.nvidia.com/ cuda-toolkit

7. NVIDIA Parallel Nsight. [Электронный ресурс] : URL http://developer. nvidia.com/nvidia-parallel-nsight

Сведения об авторах

Иванов Михаил Александрович
Ivanov Michael Aleksandrovich
Национальный исследовательский ядерный университет «МИФИ»
(НИЯУ МИФИ)
National Research Nuclear University MEPhI (NRNU MEPhI)
Заведующий кафедрой Компьютерных систем и технологий
Chief of the Computer Systems and Technologies department
1978 г. Московский инженерно-физический институт
1978, MEPhI
Д.т.н., профессор
Post Doctoral Research Fellow, Professor
Более 100 печатных работ, в том числе 6 монографий
More than 100 publications including 6 books
Криптография, генераторы псевдослучайных чисел
Cryptography, Pseudorandom number generators
MAIvanov@mephi.ru, 8-926-558-60-99

Васильев Николай Петрович
Vasilyev Nikolay Petrovich
НИЯУ МИФИ
NRNU MEPhI
Доцент
Associate Professor
1995 г. Московский инженерно-физический институт
1995, MEPhI
К.т.н., доцент
PhD
Более 60 печатных работ
More than 60 publications
Облачные вычисления, гибридные суперкомпьютерные технологии
Cloud computing, Hybrid supercomputer technologies
NPVasilyev@mephi.ru, 8-905-752-80-37

Чугунков Илья Владимирович
Chugunkov Ilya Vladimirovich
НИЯУ МИФИ
NRNU MEPhI
Доцент
1999 г. Московский инженерно-физический институт
1999, MEPhI
К.т.н., доцент
PhD
Более 20 печатных работ, в том числе 2 монографии
More than 20 publications including 2 books
Генераторы псевдослучайных чисел (ГПСЧ), статистическое тестирование
ГПСЧ
Pseudorandom number generators (PRNG), statistical testing of PRNG
IVChugunkov@mephi.ru, 8-910-469-41-71