УДК 681.391.1

ОСОБЕННОСТИ ИСПОЛЬЗОВАНИЯ БУЛЕВЫХ ФУНКЦИЙ ДЛЯ ОРГАНИЗАЦИИ КРИПТОГРАФИЧЕСКИХ ПРЕОБРАЗОВАНИЙ ПОТОКОВОЙ ИНФОРМАЦИИ

А. Б. Бубликов,
ассистент
И. Л. Ерош,
д-р техн. наук, профессор
М. Б. Сергеев,
д-р техн. наук, профессор
Санкт-Петербургский государственный университет аэрокосмического приборостроения (ГУАП)

Рассмотрены характеристики булевых преобразований при использовании для защиты потоковой информации. Приведены сравнительные результаты исследования программной и аппаратной реализации, отмечены преимущества и особенности рассматриваемых преобразований.

Characteristics of the boolean transformations for protection stream information are considered. Comparative results of research of program and hardware realization are presented. Advantages and features of considered transformations are emphasized.

Введение

Целью данной работы является исследование возможности использования алгоритмов криптографических преобразований на основе булевых функций в реальных системах передачи потоковой информации. Долгое время стандартом в этой области являлся алгоритм RC4, однако слабая (на сегодняшний день) стойкость этого алгоритма к атакам и достаточно ресурсоемкая аппаратная реализация (алгоритм RC4 изначально разрабатывался с оптимизацией для использования программных реализаций на базе персональных компьютеров) заставляют искать новые алгоритмы для таких применений. Поэтому в качестве объекта исследования было выбрано семейство алгоритмов криптографических преобразований на основе булевых функций.

Основные принципы построения алгоритмов таких криптографических преобразований предложены в работе [1] на примере систем распределенного контроля и управления. Обсуждая вопрос простоты преобразований и возможных применений, автор ориентировался на результаты теоретических расчетов. Однако возникшая практическая задача защиты потоковой видеоинформации [2] с интенсивностью передачи от 2 Мбит/с и выше привела к необходимости оценки реальных параметров программной и, учитывая особенности таких систем, аппаратной реализаций.

Оценка программной реализации

Для оценки эффективности предложенного алгоритма основным критерием было выбрано время преобразования при условии достаточной стойкости полученного шифротекста к прямым атакам, т. е. перебору всех возможных вариантов для получения исходного сообщения. В качестве алгоритма, используемого для сравнения, был выбран RC4 [3] как удовлетворяющий описанным выше требованиям [4].

В исследуемом алгоритме использовались функции, построенные с помощью метода минимизации слабо определенных функций, описанных в работе [1]. Для упрощения процесса оценки функция строилась для аргументов размером 4 бита, доопределенных до размера 5 бит таким образом, чтобы все элементы набора не были связаны между собой сдвигом [1].

■ ***Таблица 1.*** *Исходные данные*

Входной массив информации	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
Выходной массив информации	03	09	0E	0B	0F	04	07	0C	0A	02	06	00	0D	05	08	01

Выходной массив информации задавался случайным образом и приведен в табл. 1.

Построенные методом минимизации [5] функции кодирования $b1_{enc}[i]$ и декодирования $b1_{dec}[i]$ для данных значений имеют вид

```
b1enc[i]=(b0[7]&b0[9]&b0[10]&b0[11])|(!b0[4]&b0[6]&b0[7]
&b0[8]&!b0[9]&!b0[10])|(!b0[6]&b0[7]&b0[9]
&!b0[10]&!b0[11])|(!b0[5]&b0[6]&!b0[7]
&!b0[8]&b0[9])|(!b0[4]&!b0[5]&b0[7]
&b0[8]&!b0[9]&!b0[10])|(b0[4]&b0[5]&!b0[7]
&b0[8])|(b0[6]&b0[7]&!b0[8]&b0[9]&!b0[10])
|(!b0[6]&!b0[9]&b0[10]&!b0[11])|(!b0[7]&b0[9]
&!b0[10]&b0[11])|(!b0[4]&!b0[5]&b0[6]&!b0[7]&b0[8]
&!b0[10])|(!b0[7]&!b0[8]&b0[9]&b0[11])
|(b0[5]&!b0[6]&!b0[7]&b0[8]&!b0[9])|(b0[6]&b0[7]
&!b0[8]&!b0[9]&b0[10])|(!b0[6]&!b0[8]&b0[10]&!b0[11])
|(b0[4]&!b0[6]&!b0[7]&b0[8])|(b0[7]&!b0[9]&!b0[10]&b0[11])
|(!b0[4]&!b0[5]&!b0[6]&!b0[7]&!b0[8]&!b0[9]&!b0[10]);
```

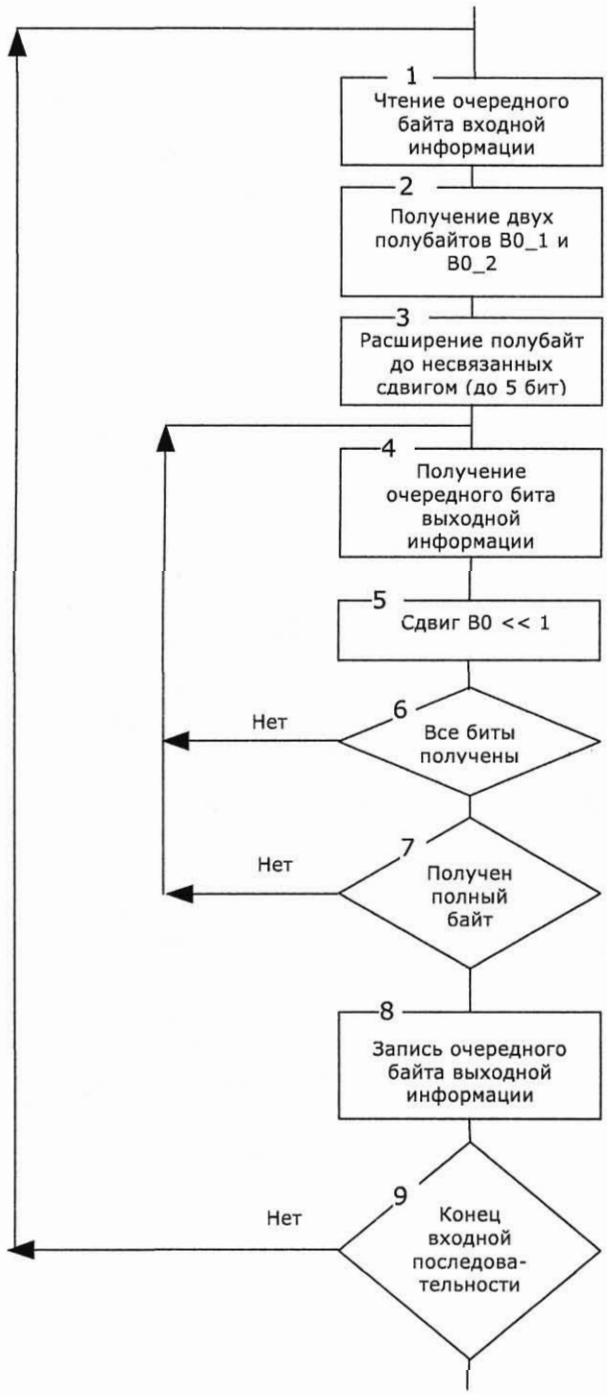
```
b1dec[i]=(b0[5]&!b0[6]&!b0[8]&b0[9])|(b0[6]&!b0[7]&!b0[8]
&!b0[9]&b0[10])|(b0[7]&!b0[8]&!b0[9]&b0[11])|(!b0[4]
&!b0[6]&b0[7]&b0[8]&!b0[11])|(!b0[5]&!b0[6]&b0[8]
&b0[9]&!b0[11])|(b0[4]&!b0[5]&!b0[7]&b0[8])|(b0[5]&b0[7]
&b0[9])|(!b0[6]&b0[7]&!b0[8]&b0[9]&!b0[10])|(b0[4]
&!b0[5]&b0[6]&b0[8])|(!b0[6]&!b0[7]&!b0[8]&b0[10]&!
b0[11])|(!b0[4]&!b0[5]&!b0[6]&!b0[7]&!b0[9]&!
b0[10])|(!b0[4]&b0[5]&!b0[7]&b0[8]&!b0[9])|(b0[7]
&b0[8]&!b0[9]&b0[10])|(!b0[7]&!b0[8]&b0[9]
&b0[11])|(b0[6]&b0[7]&!b0[8]&b0[9]&b0[10]),
```

где b0 – регистр, содержащий расширенный до 12 бит полубайт входной информации; b1 – регистр, содержащий полубайт выходной информации; b0[*n*] – *n*-й бит регистра b0; ! – булева операция «НЕ»; & – булева операция «И»; | – булева операция «ИЛИ».

Рассматриваемый алгоритм является итерационным и поэтому для получения 5 бит выходной информации необходимо преобразовать функцией 5 бит входной информации пять раз, на каждом новом шаге сдвигая ее влево на одну позицию. Как видно из самой функции, очередное значение бита выходного полубайта рассчитывается с использованием 11 бит входного слова.

Укрупненная структурная схема алгоритма преобразования представлена на рис. 1. В данном алгоритме на первом шаге считывается очередной байт информации, который на втором шаге разбивается на два полубайта. Каждый из этих двух полубайтов расширяется до 5 бит с целью получения чисел, не связанных сдвигом, путем установки пятого бита в единицу. В этот момент времени два полубайта хранятся в переменных размером 12 бит, в которых биты с шестого по двенадцатый равны нулю. Каждая из переменных преобразуется в итерационном цикле четыре раза для получения четырех бит выходной информации, которые затем объединяются в байт и записываются в выходную последовательность.

Преобразования по рассматриваемому алгоритму были реализованы на языке C++ с использованием контейнерного класса std::bitset стандартной библиотеки STL. Для компиляции программы были использованы компиляторы Borland C++ (с оптимизацией по скорости и без) и Microsoft C++ .NET с оптимизацией по скорости.



■ ***Рис. 1.*** *Укрупненная схема алгоритма реализации криптопреобразования на основе булевых функций*

Тестирование полученных реализаций проводилось при шифровании файла объемом 451 710 байт без учета дисковых операций и второстепенных вычислений. Учитывалось суммарное время работы основного итерационного цикла алгоритма (см. рис. 1).

■ ***Таблица 2.*** *Сравнение результатов*

Алгоритм	Булевы преобразования			RC4
	Borland C++, без оптимизации по скорости	Borland C++, с оптимизацией по скорости	Microcoft C++ .NET, с оптимизацией по скорости	Borland C++, с оптимизацией по скорости
Время преобразования файла размером 451 710 байт, с	41,93	8,90	0,88	0,03

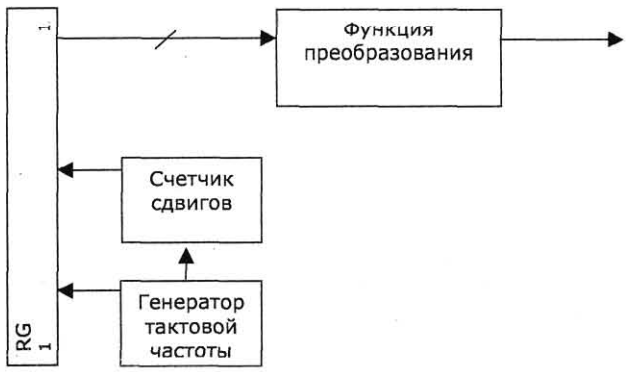
Сравнительные результаты времен преобразования указанного файла предлагаемым алгоритмом и RC4 приведены в табл. 2.

Как видно из результатов тестирования, даже лучшая реализация на основе булевых преобразований уступает по скорости (при прочих равных условиях) реализации алгоритма RC4, скорость которого принята за достаточную для потокового шифрования [3]. Основная причина – сложность программной реализации методов работы с векторами бит и операций над отдельными элементами таких векторов в языках программирования высокого уровня.

Оценка аппаратной реализации

Для защиты потоковой информации с использованием булевых преобразований в информационно-управляющих системах на основе дистанционно-управляемых автономных модулей [3] с интенсивными обменами необходимо использовать простую аппаратную реализацию, встраиваемую в сетевой канал таких модулей. Анализ сложности преобразования и обеспечения непрерывности информационного тракта показывает, что аппаратно возможна реализация на программируемой логике, например фирмы «ALTERA».

Аппаратная реализация алгоритма представлена на рис. 2.



■ ***Рис. 2.*** *Схема аппаратной реализации*

Логика функционирования

Каждый такт в сдвиговом регистре *RG* происходит сдвиг исходной информации вправо на один бит на входах логики преобразования, реализующей функцию преобразования. После установления новых значений по истечении времени срабатывания на выходе логики преобразования появляется очередной бит кодированной информации. Таким образом, при выборе метода кодирования по 4 бита для преобразования байта информации потребуется два полных цикла работы схемы.

Период тактовой частоты выбирается из соображений гарантированного срабатывания схемы программируемой логики, реализующей преобразования. Таким образом, для определения периода тактовой частоты и, как следствие, временного интервала кодирования порции потоковой информации необходимо определить время срабатывания самого длинного пути в логике преобразования. Такой путь представлен на рис. 3.

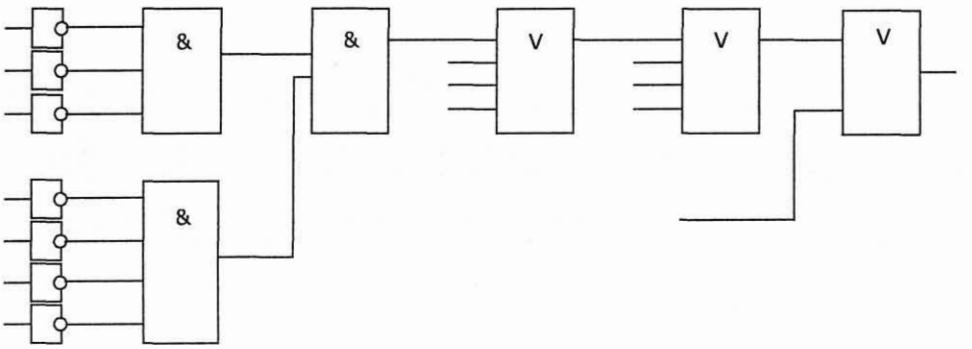
Полученный для наших исходных данных путь полностью умещается в одном логическом блоке ALTERA. Соответственно, период тактовой частоты для данной схемы будет равен времени срабатывания одного логического блока, что соответствует, в зависимости от типа выбранного для реализации блока, задержке от 4,5 до 15 нс. Полученная скорость кодирования составит от 10 до 30 Мбайт/с. Следует помнить, что данный результат получен для функции кодирования полубайта. Данное условие было введено для наглядности анализа и не подходит для реальных условий эксплуатации в связи с небольшой стойкостью к атакам по шифротексту. Даже булева функция кодирования байта гораздо менее тривиальна, ее аппаратная реализация имеет гораздо более длинные пути прохождения сигнала.

Для аппаратной реализации стандартного алгоритма RC4 ($n = 8$) требуется один буфер 256 байт только для чтения, для хранения ключа, один буфер 256 байт с произвольным доступом для хранения рабочего блока и два 8-битных регистра. Для кодирования одного блока информации в 256 байт по алгоритму RC4 требуется 1286 циклов работы [6]. Аппаратная реализация алгоритма RC4 на семействе микросхем ALTERA FLEX8000 способна выдавать 11 000 блоков в секунду, что составит около 3 Мбайт/с. Такая достаточно низкая скорость для потокового алгоритма связана с тем, что RC4 разрабатывался с условием оптимизации скорости программных реализаций алгоритма. Для сравнения компьютер на базе процессора Intel Pentium 100 способен выдавать около 22 000 блоков в секунду [6].

Заключение

Предложенный в работе алгоритм открывает новое направление в защите информации, поскольку использует булевы функции как функции криптографического преобразования данных.

Хотя программные реализации показали несколько худшие по сравнению с выбранным эталонным алгоритмом RC4 результаты, необходимо отметить несколько несомненных достоинств булевых преобра-



■ ***Рис. 3.*** *Самый длинный путь в блоке логики преобразования при аппаратной реализации*

зований, привлекательных при аппаратной реализации:

гибкость по отношению к размеру элементарного блока – метод булевых функций можно использовать с любым произвольным элементарным блоком, вплоть до двух бит;

криптостойкость и скорость преобразования информации булевыми функциями находятся в линейной зависимости от размера элементарного блока и легко поддаются прогнозированию;

отсутствие зависимости от предыдущей информации, позволяющее избежать необходимости в синхронизации при обрыве связи в информационном канале;

отсутствие буферных схем для временного хранения информации;

отсутствие ключа в обычном смысле – в качестве ключа в алгоритме используется булева функция, что несколько увеличивает криптостойкость;

значительно более простая аппаратная реализация по сравнению с известными алгоритмами.

Литература

1. **Ерош И. Л.** Защита информационных потоков в системах распределенного контроля и управления // Информационно-управляющие системы. – 2002. – № 1. – С. 40–46.
2. **Бубликов А. Б.** Организация защищенного канала передачи видеопотока в информационно-управляющих IP-системах // VI научная сессия аспирантов ГУАП: Сб. докл.: В 2-х ч. – Ч. I. Технические науки. – СПб.: Изд-во ГУАП, 2003. – С. 211 – 212.
3. **Шнайер Б.** Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. – М.: Триумф, 2003. – 816 с.
4. **Астапкович А. М., Востриков А. А., Сергеев М. Б., Чудиновский Ю. Г.** Информационно-управляющие системы на основе Интернет // Информационно-управляющие системы, 2002. – № 1. – С. 12–18.
5. **Ерош И. Л.** Дискретная математика. Булева алгебра, комбинационные схемы, преобразования двоичных последовательностей. Учебн. пособ. – СПб.: Изд-во ГУАП, 2001. – 96 с.
6. **Goldberg I., Wagner. D.** Architectural considerations for cryptanalytic hardware – ISAAC Group U. C. Berkeley, (**http://www.cs.berkeley.edu/~iang/isaac/hardware**)