КОМП'ЮТЕРНІ ЗАСОБИ, МЕРЕЖІ ТА СИСТЕМИ

A.O. Sevruk

DEVELOPING METHODS TO CONSTRUCTION STREAM CIPHER

The modern requirements to the cryptography increase demands to speed of processing the information. In view of this the stream algorithms of the encryption is rather actual. This article deals with cryptographic primitives, that are used in the stream cipher, and also describes the example of building algorithm that are based on the stream cipher methods.

Современные требования к криптографии накладывают жесткие рамки к скорости обработки информации. В данном ключе актуальными становятся потоковые алгоритмы шифрования. В работе описаны криптографические примитивы, используемые в потоковых шифрах, а также приводится пример построения алгоритма, основанного на методах потокового шифрования.

УДК 004.056.5:004.272.44(045)

А.О. СЕВРУК

РАЗРАБОТКА МЕТОДОВ ПОСТРОЕНИЯ ПОТОКОВЫХ ШИФРОВ

На современном этапе существования криптографии самые распространенные системы – симметричная криптография и криптография с открытым ключом. В свою очередь, симметричная криптография делится на блочные и потоковые шифры. Каждый вид шифров, будь то потоковый алгоритм шифрования или алгоритм с открытым ключом, имеет свои преимущества и недостатки. Так, основным преимуществом потоковых шифров перед другими типами алгоритмов являются: высокое быстродействие – шифрование осуществляется практически в реальном масштабе времени сразу при поступлении очередного элемента входной последовательности, эффективная прграммная, аппаратная реализация, высокая криптостойкость и ряд других преимуществ.

Одной из основных компонент для построения потоковых шифров (как при аппаратной, так и программной реализации) является LFSR – (Linear Feedback Shift Register) сдвиговый регистр с линейной обратной связью [1]. Этот компонент выполняет две группы преобразований: функции сдвига и обратной связи.

Функция сдвига является тривиальной. Функция обратной связи может представлять из себя вид:

$$\Phi(x) = x_n \oplus x_{n-1} \oplus \ldots \oplus x_3 \oplus 1.$$

Традиционно операция применяемая при поточном шифровании данных – логическая операция XOR. В методе потокового шифрования эта операция используется как для шифрования данных, так и для их расшифровки, так как XOR – обратимая операция.

Если разрядность регистра LFSR обозначим m, тогда период повторения элементов ключевой последовательности будет равен

$$P = 2^m - 1.$$

Известно, что, если длина ключа больше или равна размеру шифруемой информации и применяемый при процедуре шифрования ключ уникальный, то такой шифр теоретически невозможно взломать. Такой способ шифрования получил название «метод одноразовых блокнотов» [2]. Несмотря на то, что с помощью LFSR можно добиться большого периода повтора ключевой последовательности, сам LFSR не является криптостойким и возможно вычисление начального заполнения LFSR и его функции обратной связи по выходной информации регистра.

Улучшить криптоустойчивость системы можно за счет многократного использования LFSR [3] коммутируя вариации LFSR через мультиплексор или другую схему подключения, а именно: генератор Геффа; генератор «Stop-and-Go»; самопрореживающийся генератор.

Генератор Геффа (рис. 1): в этом генераторе используется три LFSR, объединенные нелинейным образом. Если a_1, a_2 и a_3 – выходы трех LFSR, выход генератора Геффа можно описать как $b = (a_1 \,\&\, a_2) \oplus ((!a_1) \,\&\, a_3)$.

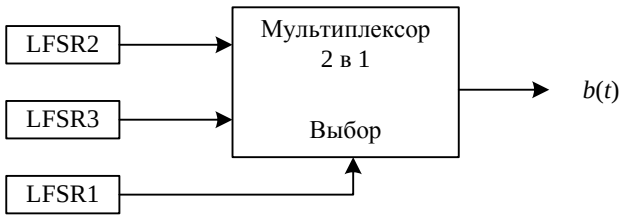

РИС. 1. Генератор Геффа

Если длины LFSR равны n_1, n_2 и n_3, соответственно, то линейная сложность генератора равна

$$L = (n_1 + 1) * n_2 + n_1 * n_3 .$$

Период генератора равен наименьшему общему делителю периодов трех LFSR. При условии, что степени трех примитивных многочленов обратной связи взаимно просты, период этого генератора будет равен произведению периодов трех LFSR.

Криптографическая система, использующая генератор Геффа не обладает достаточной криптоустойчивостью, выход генератора равен выходу LFSR-2 в 75 % времени. С такой корреляционной зависимостью генератор потока ключей может быть легко взломан.

Генератор «Stop-and-Go» использует выход одного LFSR для управления тактовой частотой другого LFSR (рис. 2). Тактовый вход LFSR-2 управляется выходом LFSR-1, так, что LFSR-2 может изменять свое состояние в момент времени t только, если выход LFSR-1 в момент времени (t-1) был равен 1.

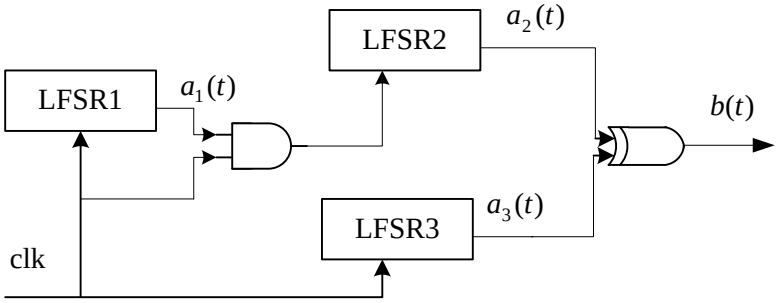

РИС. 2. Генератор «Stop-and-Go»

Что касается линейной сложности данного генератора, то для общего случая в литературных источниках отсутствуют работы на данную тему, однако установлено, что он восприимчив к корреляционному методу вскрытия.

Самопрореживающие (Self-Decimated) генераторы это генераторы, которые управляют собственной тактовой частотой (рис. 3). В генераторе, если выход LFSR равен 0, LFSR тактируется *d* раз. Если выход LFSR равен 1, LFSR тактируется *k* раз, генератор не безопасен, хотя был предложен ряд модификаций, которые могут исправить встречающиеся проблемы.

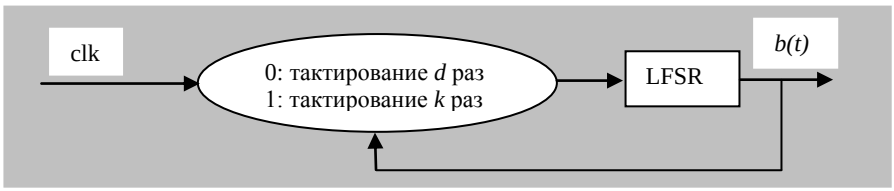

РИС. 3. Самопрореживающийся генератор

Чаще всего, алгоритмы потокового шифрования взламывают либо с помощью статистических характеристик битов ключевой последовательности с законом распределения отличной от идеальной (частота появления одного из значений генератора намного превышает частоту появления других значений), либо при повторном шифровании – потоком ключей, который использовался ранее для шифрования другого сообщения [4].

Для полноты изложения материала приведем пример выделения ключевой последовательности из зашифрованной информации. Допустим, у нас имеется два сообщения p_1 и p_2 . Для шифрования этих сообщений используем одинаковую последовательность бит x, т. е. получаем:

$$y_1 = p_1 \oplus x; \quad y_2 = p_2 \oplus x.$$

Наложим два сообщения друг на друга с помощью операции xor:

$$(p_1 \oplus x) \oplus (p_2 \oplus x) = p_1 \oplus p_2. \quad (1)$$

Выражение (1) показывает, что в итоге этого преобразования мы избавимся от ключевой последовательности x. Это облегчит выделение из (p_1 хог p_2) с помощью специальных методов, включающих статистические исследования, подбор букв и т. д., отдельно p_1 и p_2. Отсюда следует, что

- гамма, производимая шифром, должна обладать очень хорошими стохастическими характеристиками;
- период гаммы шифра должен превышать длину наибольшего из сообщений, планируемых к пересылке на одном и том же ключе.

Заметим, что при использовании потоковых шифров также возникает проблема достоверности данных. Поэтому с потоковым шифрованием или внутри алгоритма шифрования необходимо предусматривать системы подтверждения целостности сообщения.

Средством для подтверждения целостности данных может стать хэш-функция вида $h(p)$. Хэш-функцией в криптографии принято называть необратимое преобразование данных.

Для построения хэш-функций обычно используются как специальные алгоритмы хеширования (MD5, SHA-1) [5], так и на основе криптографии с открытым ключом, блочных и потоковых шифров.

Проанализировав достоинства и недостатки существующих подходов к построению криптосистем, целесообразным является вопрос проектирования криптопроцессора, основанного на смешанном подходе организации алгоритма. Для работы данного устройства необходимо три регистра типа LFSR. Выбор разрядности регистров необходимо производить так, чтоб их длины были взаимно простыми друг к другу. В данном примере длины были выбраны так: 43, 47 и 37 бит. Таким образом, длинна ключа для начального заполнения сдвиговых регистров будет равняться 127 битам. Период повторения наибольшего из регистров типа LFSR–2^{47}–1. От результата его работы будет зависеть, с каким из регистров будет применена операция XOR. Бит, не использующийся в данном такте работы, динамически заполняет и изменяет две таблицы ключевой информации разрядностью по 256 бит каждая, использующиеся для придания нелинейности алгоритма и служит защитой от корреляционной атаки на алгоритм. Для начала работы необходимо проинициализировать регистры ключевой информации, для чего необходимо сделать 512 циклов, после этого на вход устройства можно побитово подавать данные. Зашифрованные данные могут выдаваться побайтно каждые 256 циклов или побитно, но с начальной задержкой в 256 циклов.

Приведем пошагово работу алгоритма для шифрования данных:

- инициализируем ключом регистры LFSR;
- заполняем регистры ключевой информации;
- подаем на вход данные;
- вычисляем ключ для непосредственного шифрования одного бита данных и одновременно с каждым битом ключа по очереди динамически изменяем ре-

гистры ключевой информации;

- заполняем регистр данных уже зашифрованной информацией;
- при заполненном регистре еще раз зашифровываем его регистрами ключей и получаем результирующую информацию.

Для расшифровки сообщения выполняем следующие шаги:

- инициализируем ключом регистры LFSR;
- заполняем регистры ключевой информации и динамически их изменяем еще 256 тактов;
- подаем на вход данные;
- получаем один регистр ключевой информации вместо двух с помощью операции XOR;
- с помощью операции XOR получаем регистр данных, зашифрованный только ключом без регистра ключевой информации;
- с помощью ключевой последовательности расшифровываем сообщение.

Структурная схема метода показана на рис. 4.

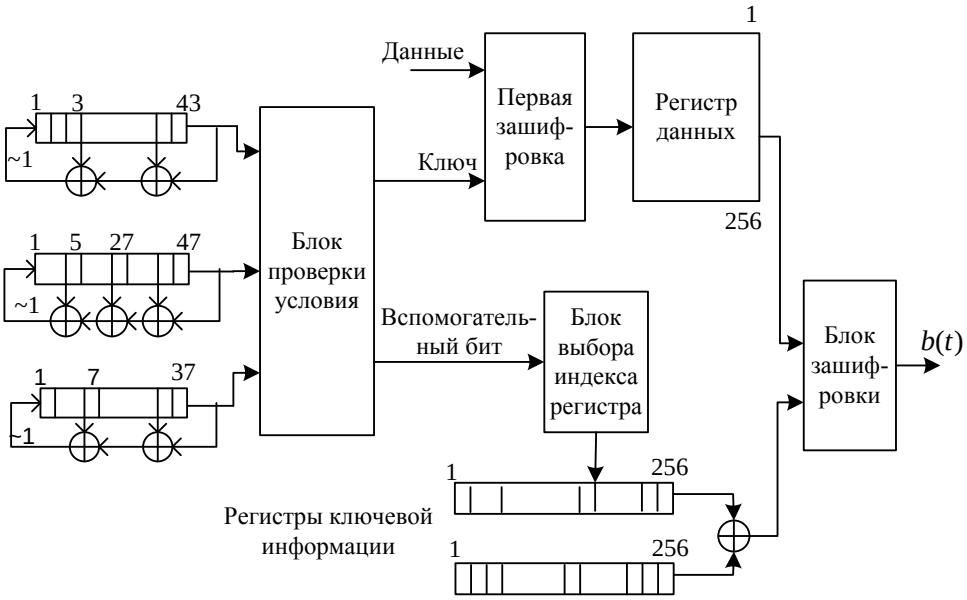

РИС. 4

Рассмотрим по отдельности некоторые блоки алгоритма. Основное перемешивание и распределение бит лежит на сдвиговых регистрах с линейной обратной связью.

Функции обратной связи в LFSR описываются такими многочленами:

- для первого LFSR: $\Phi(x) = x_{43} \oplus x_{42} \oplus x_3 \oplus 1$;
- для второго LFSR: $\Phi(x) = x_{47} \oplus x_{46} \oplus x_{27} \oplus x_5 \oplus 1$;

- для третьего LFSR: $\Phi(x) = x_{37} \oplus x_{36} \oplus x_7 \oplus 1$.

Блок проверки условия работает по такому принципу: если выход LFSR2 равен 1, то ключ образуется путем складывания с помощью операции XOR LFSR2 и LFSR3, а выход LFSR1 поступает в блок выбора индекса регистров (регистров ключевой информации). Если выход LFSR2 равен 0, то ключ образуется из выхода регистров LFSR2 и LFSR1, а выход LFSR3 поступает в блок выбора индекса регистров.

После того, как был сформирован очередной бит гаммы, он используется для шифрования одного бита информации и уже зашифрованная информация помещается на временное хранение в специальный регистр размером 256 бит. Для усиления защиты шифрованная информация подвергается повторному шифрованию с помощью регистров ключевой информации и только после этого получаем выходной результат.

Перед повторным шифрованием данных регистры ключевой информации преобразуются в одну, путем накладывания одного регистра на другой с помощью операции XOR и в результате получаем перемешанные биты регистров разрядностью в 256 бит. С помощью полученного регистра и производится шифрование регистра данных. После этого зашифрованная информация подается на выход алгоритма.

Для исследования криптоустойчивости шифров целесообразно тестировать их на предмет устойчивости к распространенным криптоатакам. На сегодняшний день существует достаточное количество схем криптоанализа алгоритмов. Самыми распространенными являются линейный и дифференциальный криптоанализ.

Работы по линейному криптоанализу впервые опубликовал M. Matsui в 1993 году и классифицируется как атака по известному открытому тексту. Его основой является попытка выделить в блоке исходного текста X, ключе K и блоке шифротекста Y такие биты, чтобы их сумма по модулю 2 на большом объеме зашифрованной информации имела неравномерное распределение. То есть прослеживалось отклонение Z от закономерных вероятностей p(0) = 0.5; p(1) = 0.5 . Защитой от линейного криптоанализа является активное внедрение в криптоалгоритм нелинейных криптопримитивов а также тщательная проверка последовательности выполняемых операций внутри блока и между блоками на предмет «просачивания» линейных зависимостей.

Дифференциальный криптоанализ впервые был опубликован Edi Biham и Adi Shamir в 1991 году. Данный метод требует наличия у криптоаналитика возможности зашифровать на искомом ключе любое свое сообщение. Алгоритм требует поиска такого изменения входного блока данных, чтобы после шифрования вероятность изменения выходных данных имела закон распределения, отличный от равномерного. Если подобное изменение найдено, то путем шифрования и последующего анализа специальным образом подбираемых значений криптоаналитик может в итоге построить систему из достаточного количества линейных уравнений, описывающих материал ключа. Алгоритм подбора шиф-

руемых значений аналитиком очень сильно варьируется в зависимости от атакуемого шифра. Цель, которую ставит перед собой криптоаналитик в исследуемых криптоалгоритмах, является предугадыванием или получением какой-либо информации о промежуточных значениях данных внутри блоков шифрования.

1. *Xilinx.* Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators. XAPP 052 July 7,1996 (Version 1.1)
http://www.xilinx.com/bvdocs/appnotes/xapp052.pdf
2. *Конеев И., Беляев А.* Информационная безопасность предприятия. – СПб.: БХВ-Петербург, 2003. – 740 с.
3. *Шнайдер Б.* Прикладная криптография. Протоколы, алгоритмы и исходные тексты на языке Си. 2-е издание. – М.: Триумф, 2003. – 816 с.
4. *Gustavson F.G.* Analysis of the Berlekamp-Massey Linear Feedback Shift Register Synthesis Algorithm. - http://www.research.ibm.com/journal/rd/203/ibmrd2003C.pdf
5. *Файстель Х.* Криптография и компьютерная безопасность
http://www.enlight.ru/crypto/articles/feistel/feist_i.htm

Получено 02.03.2006