УДК 004

АЛГОРИТМИЧЕСКАЯ РЕАЛИЗАЦИЯ ГЕНЕРАТОРА ПСЕВДОСЛУЧАЙНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

Морозов В.В., студент гр. ИВМ-24, 2 курс
Научный руководитель: Гусаров А.В., к.т.н., доцент
Рыбинский государственный авиационный технический университет
имени П.А. Соловьева
г. Рыбинск

В контексте информационной безопасности распределённых информационно-управляющих систем критической задачей для её обеспечения является разработка эффективных способов защиты каналов связи, аутентификации узлов и целостности данных. В таком случае безопасность распределённой среды напрямую зависит от качества используемых криптографических примитивов. Особую роль здесь играют генераторы псевдослучайных последовательностей (ГПСП). Эти ГПСП как раз применяются для формирования различных данных. Например, ключевой информации, векторов инициализации и сеансовых ключей.

Более того в условиях перехода на отечественное программное и аппаратное обеспечение возрастает потребность в верифицируемых алгоритмах ГПСП. Использование зарубежных аналогов несёт риски наличия недекларированных возможностей. В то же время это в принципе может быть запрещено при использовании в критических инфраструктурах, как например, в государственном секторе. Поэтому приоритет отдается решениям на базе отечественных стандартов ГОСТ. Однако стоит отметить, что простая реализация стандартов требует тщательной проработки алгоритмической базы для обеспечения производительности и качества разрабатываемых генераторов в распределённой среде.

В статье рассматривается процесс проектирования алгоритмической базы ГПСП. Основное внимание уделено структуре алгоритма, включая развёртывание ключа и раундовые преобразования. Приведённые алгоритмические решения обеспечивают соответствие требованиям регуляторов РФ и гарантируют криптостойкость формируемой гаммы.

Основой проектируемой системы выступает алгоритм блочного шифрования ГОСТ Р 34.12–2018 («Кузнечик») [1]. Данный стандарт определяет симметричный блочный шифр с длиной блока 128 бит и ключа 256 бит. Структура алгоритма итерационная и состоит из 10 раундов, каждый из которых включает нелинейные и линейные преобразования, обеспечивающие лавинный эффект.

Для построения ГПСП необходимо преобразовать блочный шифр в потоковый. Согласно ГОСТ Р 34.13–2018 [2], наиболее подходящим для этих

задач является режим гаммирования (режим счетчика, *CTR*). Эффективность применения шифра «Кузнечик» в качестве основы детерминированного генератора подтверждена в работе [3], где продемонстрирована криптостойкость выходных последовательностей при соблюдении стандартов *NIST*. Преимущества режима *CTR* заключаются в следующем:

- отсутствие дополнения. Генерация последовательности произвольной длины с точностью до бита без выравнивания по размеру блока;
- параллелизм. Возможность независимого вычисления блоков гаммы, так как значения счётчика известны заранее;
- детерминированность. При одинаковых входных параметрах (ключ, *IV*) генерируется идентичная последовательность, что важно для синхронизации.

Принцип работы ГПСП в режиме *CTR* заключается в зашифровании последовательности значений счётчика базовым алгоритмом. Полученный шифртекст используется в качестве гаммы. Начальное значение счётчика формируется конкатенацией вектора инициализации (*IV*) и нулевых битов, последующие значения получаются инкрементом в кольце вычетов по модулю 2^n.

Общая логика работы генератора представлена на рисунке 1.

Рисунок 1 – Блок-схема алгоритма генерации псевдослучайной последовательности

Криптографическое ядро системы реализует функцию зашифрования блока данных. Алгоритм принимает на вход 128-битный блок (значение счётчика) и 256-битный мастер-ключ. В результате работы формируется 128-битный блок гаммы. Схема процесса зашифрования приведена на рисунке 2.

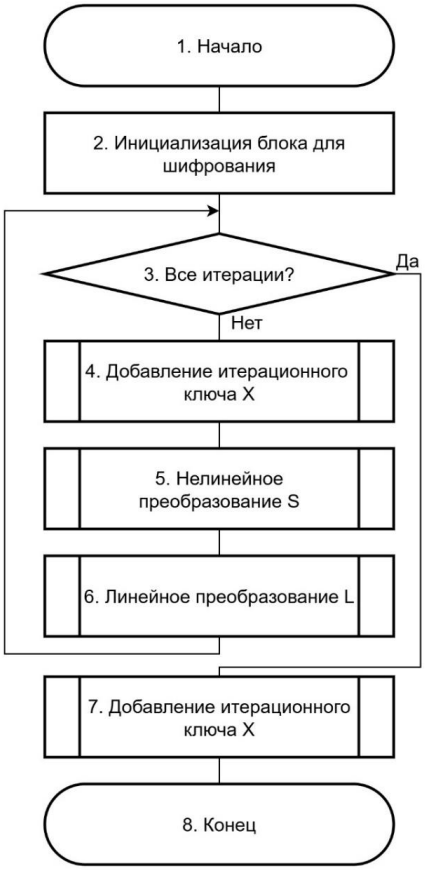

Рисунок 2 – Блок-схема алгоритма зашифрования

Корректная работа шифра требует преобразования мастер-ключа в набор итерационных ключей. Алгоритм развёртывания (*key schedule*) генерирует 10 подключей по 128 бит. Процесс включает операции сдвига, нелинейной замены и наложения констант раунда. Поскольку в режиме *CTR* ключ не меняется в процессе генерации гаммы, процедура развёртывания выполняется единожды при инициализации генератора, как указано в работе [4].

Каждый раунд шифрования (за исключением финального) состоит из четырёх этапов:

- наложение ключа. Сложение по модулю 2 с итерационным ключом раунда;

- линейное преобразование *L*. Обеспечивает диффузию битов внутри блока. Базируется на вспомогательном преобразовании *R*, применяемом 16 раз;

- преобразование *R*. Комбинация линейной функции над полем Галуа $GF(2^8)$ и циклического сдвига байтов;

- нелинейное преобразование *S*. Байтовая замена по таблице подстановок. Вносит нелинейность, защищая от линейного криптоанализа.

Финальный раунд отличается отсутствием линейного преобразования *L*. Такая структура обеспечивает баланс между производительностью и стойкостью к известным атакам.

Разработанные алгоритмические схемы обладают рядом характеристик, важных для интеграции в распределённые системы:

- гибкость длины выходной последовательности. Благодаря использованию режима гаммирования (счётчика, *CTR*), генератор не привязан к кратности размера блока в 128 бит. Для этого используется обрезка последнего блока гаммы. С помощью этой обрезки последнего блока можно формировать последовательности произвольной битовой длины для взаимодействия с программными инструментами, требующими специфических размеров последовательностей от генератора;

- модульность архитектуры генератора. Проектирование алгоритмической базы выполнено по модульному принципу, то есть операции развёртывания ключа, раундовых преобразований, зашифрования и другие операции реализованы как отдельные единицы или же функции (методы) в контексте использования в языках программирования;

- соответствие нормативным требованиям. Алгоритмическая база генератора разработана в соответствии со стандартами ГОСТ Р 34.12–2018 и ГОСТ Р 34.13–2018 и полностью регламентирована ими;

- верифицируемость свойств. Структура данного генератора предполагает и даже рекомендует последующее статистическое тестирование качества выходных последовательностей. Как показано в работе [3], для генераторов на базе шифра «Кузнечик», как и в целом любых ГПСП, важно подтверждение свойств случайности через наборы статистических тестов. Например, можно воспользоваться набором тестов из *NIST SP* 800-22. Разработка блок-схем генератора обеспечивает готовность для последующей разработки алгоритмов его верификации;

- требования к уникальности параметров. При будущей реализации на языках программирования необходимо обеспечить строгий контроль уникальности векторов инициализации (*IV*) для каждого ключа. Нарушение этого требования в режиме гаммирования приводит к повторению гаммы, что недопустимо для криптографических целей. Алгоритмическая база предусматривает механизмы инкремента счётчика, минимизирующие риск коллизий при корректной организации хранения состояния.

Подводя итог, можно сказать, что описанная алгоритмическая реализация даёт возможность построения генератора псевдослучайных последовательностей, полностью соответствующего требованиям отечественных стандартов ГОСТ Р 34.12–2018 и ГОСТ Р 34.13–2018. Модульная архитектура ГПСП, основанная на режиме гаммирования, позволяет формировать крипто-

графическую гамму произвольной длины с сохранением детерминированности, отсутствием повторений гаммы и возможности параллельных вычислений, а также демонстрирует готовность к последующей верификации статистических свойств выходных последовательностей по указанному ранее набору статистических тестов.

Полученные результаты могут быть использованы при создании защищённых модулей для распределённых информационно-управляющих систем, где требуется соответствие регуляторным требованиям РФ и гарантия криптостойкости ключевой информации.

Список литературы:

1. ГОСТ Р 34.12–2018. Информационная технология. Криптографическая защита информации. Блочные шифры [Текст]. – Взамен ГОСТ 28147–89 ; введ. 01–06–2019. – М.: Стандартинформ, 2018. – 17 с.

2. ГОСТ Р 34.13–2018. Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров [Текст]. – Взамен ГОСТ 28147–89 ; введ. 01–06–2019. – М.: Стандартинформ, 2018. – 28 с.

3. Беляев, С. А. Разработка функции генерации псевдослучайных последовательностей на основе криптографического алгоритма «Кузнечик» [Текст] / С. А. Беляев [и др.] // Вопросы кибербезопасности. – 2021. – № 4. – С. 25–34.

4. Курганов, Е. А. О глубине аппаратной реализации блочного шифра Кузнечик [Текст] / Е. А. Курганов // Интеллектуальные системы. Теория и приложения. – 2016. – Т. 20, № 1. – С. 61–78.