ЛАБОРАТОРНАЯ РАБОТА 4

РЕАЛИЗАЦИЯ ЭЛЕМЕНТОВ СХЕМЫ ШИФРОВАНИЯ ГОСТ 28147–89

Цель работы: формирование умений шифрования с использованием алгоритма шифрования ГОСТ 28147–89.

Теоретические сведения

Термины и обозначения

Описание стандарта шифрования Российской Федерации содержится в документе, озаглавленном «Алгоритм криптографического преобразования ГОСТ 28147–89». То, что в его названии вместо термина «шифрование» фигурирует более общее понятие «криптографическое преобразование» вовсе не случайно. Помимо нескольких тесно связанных между собой процедур шифрования, в документе описан один построенный на общих принципах с ними алгоритм выработки имитовставки. Последняя является не чем иным, как криптографической контрольной комбинацией, то есть кодом, вырабатываемым из исходных данных с использованием секретного ключа с целью имитозащиты, или защиты данных от внесения в них несанкционированных изменений.

На различных шагах алгоритмов ГОСТа данные, которыми они оперируют, интерпретируются и используются различным образом. В некоторых случаях элементы данных обрабатываются как массивы независимых битов, в других случаях – как целое число без знака, в третьих – как имеющий структуру сложный элемент, состоящий из нескольких более простых элементов. Поэтому во избежание путаницы следует договориться об используемых обозначениях.

Элементы данных в настоящем пособии обозначаются заглавными латинскими буквами с наклонным начертанием (например, X). Через $|X|$ обозначается размер элемента данных X в битах. Таким образом, если интерпретировать элемент данных X как целое неотрицательное число, можно записать следующее неравенство: $0 \le X < 2^{|X|}$.

Если элемент данных состоит из нескольких элементов меньшего размера, то этот факт обозначается следующим образом: $X = (X_0, X_1, \ldots, X_{n-1}) = X_0 \,||\, X_1 \,||\, \ldots \,||\, X_{n-1}$. Процедура объединения нескольких элементов данных в один называется *конкатенацией данных* и обозначается символом «||». Естественно, для размеров элементов данных должно выполняться следующее соотношение: $|X| = |X_0| + |X_1| + \ldots + |X_{n-1}|$. При задании сложных элементов данных и операции конкатенации составляющие элементы данных перечисляются в порядке возрастания старшинства. Иными словами, если интерпретировать составной элемент и все входящие в него элементы данных как целые числа без знака, то можно записать следующее равенство:

$$(X_0, X_1, \ldots, X_{n-1}) = X_0 \,||\, X_1 \,||\, \ldots \,||\, X_{n-1} = X_0 + 2^{|X0|} \times (X_1 + 2^{|X1|}(\ldots(X_{n-2} + 2^{|X\,n-2|}\, X_{n-1})\ldots)).$$

В алгоритме элемент данных может интерпретироваться как массив отдельных битов, в этом случае биты обозначаем той же самой буквой, что и массив, но в строчном варианте, как показано на следующем примере:

$$X = (x_0, x_1, \ldots, x_{n-1}) = x_0 + 2^1 \times x_1 + \ldots + 2^{n-1} \times x_{n-1}.$$

Таким образом, для ГОСТа принята так называемая «little-en- dian» нумерация разрядов, то есть внутри многоразрядных слов данных отдельные двоичные разряды и их группы с меньшими номерами являются менее значимыми. Об этом прямо говорится в пункте 1.3 стандарта: «При сложении и циклическом сдвиге двоичных векторов старшими разрядами считаются разряды накопителей с большими номерами». Далее, пункты стандарта 1.4, 2.1.1 и другие предписывают начинать заполнение данными регистров-накопителей виртуального шифрующего устройства с младших, то есть менее значимых разрядов. Точно такой же порядок нумерации принят в микропроцессорной архитектуре Intel x86, именно поэтому при программной реализации шифра на данной архитектуре никаких дополнительных перестановок разрядов внутри слов данных не требуется.

Если над элементами данных выполняется некоторая операция, имеющая логический смысл, то предполагается, что данная операция выполняется над соответствующими битами элементов. Иными словами $A \bullet B = (a_0 \bullet b_0, a_1 \bullet b_1, \ldots, a_{n-1} \bullet b_{n-1})$, *где* $n = |A| = |B|$, а символом «•» обозначается произвольная бинарная логическая операция; как правило, имеется в виду операция исключающего ИЛИ, она же – операция суммирования по модулю 2: $a \oplus b = (a + b) \bmod 2$.

Логика построения шифра и структура ключевой информации ГОСТа

В ГОСТе 28147–89 содержится описание алгоритмов нескольких уровней. На самом верхнем находятся практические алгоритмы, предназначенные для шифрования массивов данных и выработки для них имитовставки. Все они опираются на три алгоритма низшего уровня, называемые в тексте ГОСТа циклами. Эти фундаментальные алгоритмы упоминаются как базовые циклы, чтобы отличать их от всех прочих циклов. Они имеют следующие названия и обозначения, последние приведены в скобках, и смысл их будет объяснен позже:

цикл зашифрования (***32-З***);

цикл расшифрования (***32-Р***);

цикл выработки имитовставки (***16-З***).

В свою очередь, каждый из базовых циклов представляет собой многократное повторение одной единственной процедуры, называемой *основным шагом криптопреобразования*.

В ГОСТе ключевая информация состоит из двух структур данных. Помимо собственно ключа, необходимого для всех шифров, она содержит еще и таблицу замен. Ниже приведены основные характеристики ключевых структур ГОСТа.

Ключ является массивом из восьми 32-битовых элементов кода, далее в настоящей работе он обозначается символом K: $K = \{Kj\}, 0 \le j \le 7$. В ГОСТе элементы ключа используются как 32-разрядные целые числа без знака: $0 \le Kj < 2^{32}$. Таким образом, размер ключа составляет 32 • 8 = 256 бит или 32 байта.

Таблица замен является вектором, содержащим восемь узлов замены. Каждый узел замены, в свою очередь, является вектором, содержащим шестнадцать 4-битовых элементов замены, которые можно представить в виде целых чисел от 0 до 15, все элементы одного узла замены обязаны быть различными. Таким образом, таблица замен может быть представлена в виде матрицы размера 8 × 16 или 16 × 8, содержащей 4-битовые заменяющие значения. Для языков программирования, в которых двумерные массивы расположены в оперативной памяти по строкам, естественным является первый вариант (8 × 16), его-то мы и возьмем за основу. Тогда узлы замены будут строками таблицы замен. В настоящей статье таблица замен обозначается символом H: $H = \{H_{i,j}\}_{0 \le j \le 15}^{0 \le i \le 7}$, $0 \le H_{i,j} \le 15$.

Таким образом, общий объем таблицы замен равен: 8 узлов • 16 элементов/узел • 4 бита/элемент = 512 бит = 64 байта.

Основной шаг криптопреобразования

Основной шаг криптопреобразования является оператором, определяющим преобразование 64-битового блока данных. Дополнительным параметром этого оператора является 32-битовый блок, в качестве которого используется какой-либо элемент ключа. Схема алгоритма основного шага приведена на рис. 1.

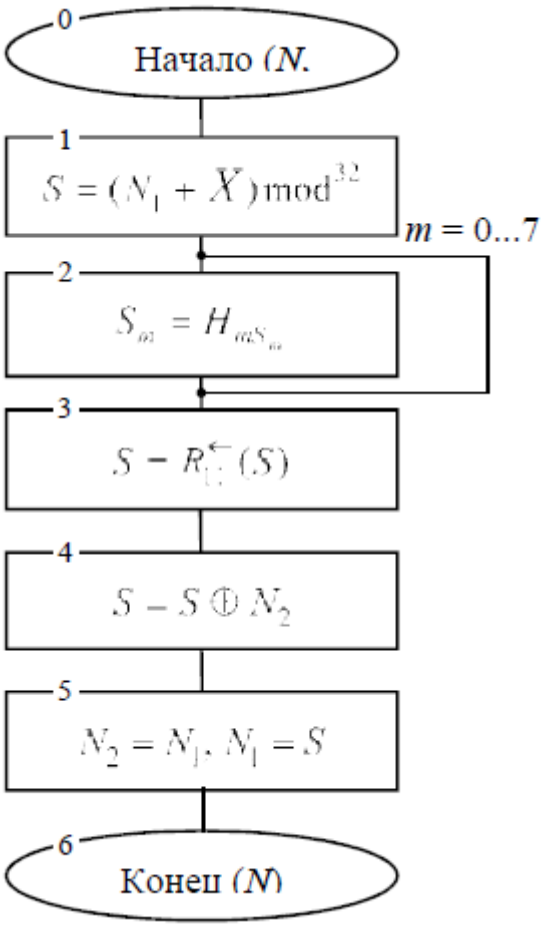

Рис. 1

Ниже даны пояснения к алгоритму основного шага:

Шаг 0

Определяет исходные данные для основного шага криптопреобразования:

N – преобразуемый 64-битовый блок данных, в ходе выполнения шага его младшая (N_1) и старшая (N_2) части обрабатываются как отдельные 32-битовые целые числа без знака. Таким образом, можно записать $N = (N_1, N_2)$.

X – 32-битовый элемент ключа.

Шаг 1

Сложение с ключом. Младшая половина преобразуемого блока складывается по модулю 2^{32} с используемым на шаге элементом ключа, результат передается на следующий шаг.

Шаг 2

Поблочная замена. 32-битовое значение, полученное на предыдущем шаге, интерпретируется как массив из восьми 4-битовых блоков кода: $S = (S_0, S_1, S_2, S_3, S_4, S_5, S_6, S_7)$, причем S_0 содержит четыре самых младших, а S_7 – четыре самых старших бита S.

Далее значение каждого из восьми блоков заменяется новым, которое выбирается по таблице замен следующим образом: значение блока Si меняется на Si-й по порядку элемент (нумерация с нуля) i-го узла замены (то есть i-й строки таблицы замен, нумерация также с нуля). Другими словами, в качестве замены для значения блока выбирается элемент из таблицы замен с номером строки, равным номеру заменяемого блока, и номером столбца, равным значению заменяемого блока как 4-битового целого неотрицательного числа. Отсюда становится понятным размер таблицы замен: число строк в ней равно числу 4-битовых элементов в 32-битовом блоке данных, то есть восьми, а число столбцов равно числу различных значений 4-битового блока данных, равному, как известно, в 24-битовом блоке – шестнадцати.

Шаг 3

Циклический сдвиг на 11 бит влево. Результат предыдущего шага сдвигается циклически на 11 бит в сторону старших разрядов и передается на следующий шаг. На схеме алгоритма символом $R_{11}^{\leftarrow}$ обозначена функция циклического сдвига своего аргумента на 11 бит влево, то есть в сторону старших разрядов.

Шаг 4

Побитовое сложение: значение, полученное на шаге 3, побитно складывается по модулю 2 со старшей половиной преобразуемого блока.

Шаг 5

Сдвиг по цепочке: младшая часть преобразуемого блока сдвигается на место старшей, а на ее место помещается результат выполнения предыдущего шага.

Шаг 6

Полученное значение преобразуемого блока возвращается как результат выполнения алгоритма основного шага криптопреобразования.

Базовые циклы криптографических преобразований

ГОСТ относится к классу блочных шифров, то есть единицей обработки информации в нем является блок данных. Следовательно, вполне логично ожидать, что в нем будут определены алгоритмы для криптографических преобразований, то есть для зашифрования/расшифрования и «учета» в контрольной комбинации одного блока данных. Именно эти алгоритмы и называются базовыми циклами ГОСТа, что подчеркивает их фундаментальное значение для построения этого шифра.

Базовые циклы построены из основных шагов криптографического преобразования, рассмотренного в предыдущем разделе. В процессе выполнения основного шага используется только один 32-битовый элемент ключа, в то время как ключ ГОСТа содержит восемь таких элементов. Следовательно, чтобы ключ был использован полностью, каждый из базовых циклов должен многократно выполнять основной шаг с различными его элементами. Вместе с тем кажется вполне естественным, что в каждом базовом цикле все элементы ключа должны быть использованы одинаковое число раз, по соображениям стойкости шифра это число должно быть больше одного.

Цикл зашифрования *32-3*:

$K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0$ (рис. 2).

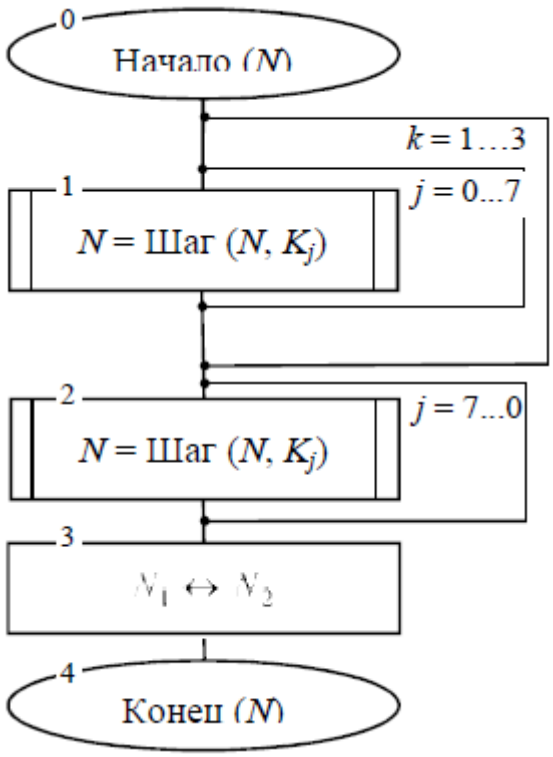

Рис.2

Цикл расшифрования *32-Р*:

K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0, K_7, K6, K_5, K_4, K_3, K_2, K_1, K_0 (рис. 3).

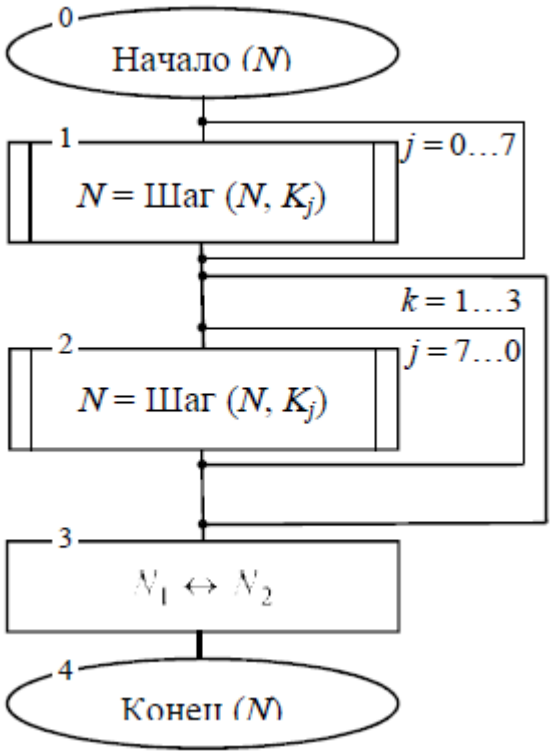

Рис 3.

Цикл выработки имитовставки *16-З*:

K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7 (рис. 4).

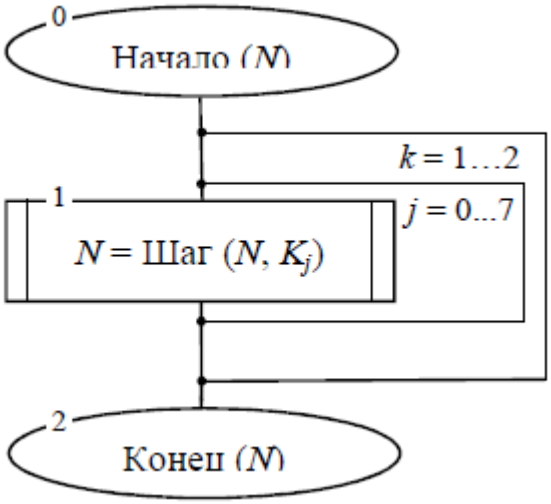

Рис. 4

Каждый из циклов имеет собственное буквенно-цифровое обозначение, соответствующее шаблону «*n-X*», где первый элемент обозначения (*n*), задает число повторений основного шага в цикле, а второй элемент обозначения (*X*), (буква), задает

порядок зашифрования («*З*») или расшифрования («*Р*») в использовании ключевых элементов.

Цикл расшифрования должен быть обратным циклу зашифрования, то есть последовательное применение этих двух циклов к произвольному блоку должно дать в итоге исходный блок, что отражается следующим соотношением: Ц 32-Р(Ц 32-З(Т)) = Т, где Т – произвольный 64-битовый блок данных, Ц *X*(Т) – результат выполнения цикла *X* над блоком данных Т. Для выполнения этого условия для алгоритмов, подобных ГОСТу, необходимо и достаточно, чтобы порядок использования ключевых элементов соответствующими циклами был взаимно обратным. В справедливости записанного условия для рассматриваемого случая легко убедиться, сравнив приведенные выше последовательности для циклов 32-З и 32-Р. Из сказанного вытекает одно интересное следствие: свойство цикла быть обратным другому циклу является взаимным, то есть цикл 32-З является обратным по отношению к циклу 32-Р. Другими словами, зашифрование блока данных теоретически может быть выполнено с помощью цикла рас - шифрования, в этом случае расшифрование блока данных должно быть выполнено циклом зашифрования. Из двух взаимно обратных циклов любой может быть использован для зашифрования, тогда второй должен быть использован для расшифрования данных, однако стандарт ГОСТ 28147–89 закрепляет роли за циклами и не предоставляет пользователю права выбора в этом вопросе.

Цикл выработки имитовставки вдвое короче циклов шифрования, порядок использования ключевых элементов в нем такой же, как в первых 16-ти шагах цикла зашифрования, в чем нетрудно убедиться, рассмотрев приведенные выше последовательности, поэтому этот порядок в обозначении цикла кодируется той же самой буквой «З».

Схемы базовых циклов приведены на рис.2–4. Каждый из них принимает в качестве аргумента и возвращает в качестве результата 64-битовый блок данных, обозначенный на схемах N. Символ «Шаг(*N, X*)» обозначает выполнение основного шага криптопреобразования для блока данных *N* с использованием ключевого элемента *X*. Между циклами шифрования и вычислния имитовставки есть еще одно отличие, не упомянутое выше: в конце базовых циклов шифрования старшая и младшая часть блока результата меняются местами, это необходимо для их взаимной обратимости.

Основные режимы шифрования

ГОСТ 28147–89 предусматривает три следующих режима шифрования данных:

простая замена;

гаммирование;

гаммирование с обратной связью;

дополнительный режим выработки имитовставки.

Содержание заданий

Разработайте программу, имитирующую реализацию элементов метода криптографической защиты информации в соответствии с ГОСТ 28147–89. Программа должна выполнять генерацию ключей, шифрование и расшифрование сообщения. В качестве сообщения используйте свою **фамилию**.

Контрольные вопросы

1. Что такое блочное шифрование?
2. Какое количество режимов работы имеет криптосистема ГОСТ 28147–89?
3. Перечислите основные преимущества и недостатки алгоритма ГОСТ 28147–89?

Отчетность по лабораторной работе

Распечатать код программы с подробными его комментариями и результатами выполнения программы.