Лабораторная работа 4
Программная реализация алгоритма блочного шифра

Цель: получение навыков программной реализации блочного шифра.

Задание на лабораторную работу

1) Программно реализовать блочный шифр по ГОСТ Р 34.12-2015 с размером блока 64 бита в режиме простой замены по ГОСТ Р 34.13-2015 в соответствии со следующими требованиями:

- в программе должна быть реализована процедура ввода ключа (ключ шифрования должен считываться из отдельно сформированного бинарного файла с расширением .key);
- ключ должен заранее формироваться с использованием программного генератора случайных чисел, реализованного ранее;
- размер ключа шифрования 56 бит, в программе должна отсутствовать возможность ввода ключа другого размера;
- развертывание ключа размером 56 бит до размера в 256 бит осуществляется повторением (4*56 + старшие 4 байта ключа);
- в программе должен быть предусмотрен контроль срока действия ключа шифрования – при зашифровании более 10 Кбайт открытого текста должно выводиться предупреждение о необходимости смены ключа зашифрования; программа не должна допускать возможность зашифрования на одном ключе более 20 Кбайт открытого текста;
- дополнение блока открытого текста выполняется в соответствии с процедурой 2 по ГОСТ Р 34.13-2015 (в младшие разряды дописывается бит «1», затем биты «0»);
- результат зашифрования записывается в бинарный файл с расширением .enc;
- результат расшифрования записывается в бинарный файл с расширением .txt.

2) Выполнить зашифрование текстового файла (размер более 10 Кб, но менее 20 Кб) и выполнить следующие действия:

- расшифровать файл с шифртекстом, проверить правильность расшифрования;
- сформировать файлы (путем модификации файла с шифртекстом) со следующими искажениями:
  - удаление 1 байта данных из шифртекста;
  - удаление блока данных шифртекста, размер которого не кратен 64 битам;

– удаление блока данных шифртекста, размер которого кратен 64 битам;

– добавление блока данных в шифртекст, размер которого кратен 64 битам;

– перестановка двух блоков данных шифртекста, размер которых кратен 64 битам;

– расшифровать сформированные файлы, оценить помехоустойчивость и имитостойкость алгоритма шифрования;

3) Выполнить зашифрование произвольного файла размером более 20 Кб, продемонстрировать блокировку работы программы.

Теоретические сведения

Основные параметры шифра «Магма»:
– длина блока данных – 64 бита;
– длина ключа – 256 бит;
– длина раундового ключа – 32 бита;
– количество раундов – 32;
– структура – сеть Фейстеля.

Как и в классической сети Фейстеля, входной блок данных разбивается на два подблока по 32 бита, над которыми выполняются преобразования сложения с ключом, подстановки и перестановки.

Преобразование подстановки t выполняется следующим образом:

$$t : V_{32} \to V_{32},$$
$$t(a) = t(a_7 \,||\, ... \,||\, a_0) = \pi_7(a_7) \,||\, ... \,||\, \pi_0(a_0),$$

где $a = a_7||\ldots||a_0 \in V_{32}$, $a_i \in V_4$, $i = 0,1,...,7$. Преобразование t ставит в соответствие входному 32х битному вектору выходной вектор такого же размера. Преобразование заключается в разбиении блока на восемь (от нулевого до седьмого) четырехбитных векторов a_i, причем a_0 содержит младшие разряды, a_7 – старшие. Над каждым вектором выполняется нелинейное биективное преобразование $\pi_i(a_i)$, заданное следующим образом:

$$\pi_i = Vec_4 \pi_i' Int_4 : V_4 \to V_4,$$

где $\pi_i' : Z_{2^4} \to Z_{2^4}$, $i = 0,1,...,7$. Входной четырехбитный вектор преобразуется в элемент кольца Z_{2^4} (напомним, что операция Int_n ставит в соответствие вектору число), над элементом кольца выполняется табличная подстановка π_i', результат которой преобразуется в четырехбитный вектор (выполняется операция Vec_n, ставящая в соответствие числу вектор). Значения подстановок заданы в стандарте в виде массивов:

$$\pi_i' = (\pi_i'(0), \pi_i'(1), ..., \pi_i'(15)),\ i = 0,1,...,7.$$

Таблицы подстановок (S-блоки) следующие:

π_0'=(12, 4, 6, 2, 10, 5, 11, 9, 14, 8, 13, 7, 0, 3, 15, 1);
π_1'=(6, 8, 2, 3, 9, 10, 5, 12, 1, 14, 4, 7, 11, 13, 0, 15);
π_2'=(11, 3, 5, 8, 2, 15, 10, 13, 14, 1, 7, 4, 12, 9, 6, 0);
π_3'=(12, 8, 2, 1, 13, 4, 15, 6, 7, 0, 10, 5, 3, 14, 9, 11);
π_4'=(7, 15, 5, 10, 8, 1, 6, 13, 0, 9, 3, 14, 11, 4, 2, 12);
π_5'=(5, 13, 15, 6, 9, 2, 12, 10, 11, 7, 8, 1, 4, 3, 14, 0);
π_6'=(8, 14, 2, 5, 6, 9, 1, 12, 15, 4, 11, 0, 13, 10, 3, 7);
π_7'=(1, 7, 14, 13, 0, 5, 8, 3, 4, 15, 10, 6, 9, 12, 11, 2).

Принцип выполнения операции прост – значение заменяемого *i*-го четырехбитного вектора является индексом (позицией) заменяющего значения в *i*-й таблице. Выполним преобразование *t* над вектором *a* = (1111 1101 1011 1001 0111 0101 0011 0001):

	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0
	1111	1101	1011	1001	0111	0101	0011	0001
$Int_4(a_i)$	15	13	11	9	7	5	3	1
$\pi_i'(Int_4(a_i))$	2	10	1	9	6	15	3	4
$t(a)$	0010	1010	0001	1001	0110	1111	0011	0100

Следующее преобразование, обозначенное в стандарте как $g[k]$ описывает функцию шифрования сети Фейстеля:

$$g[k]: V_{32} \to V_{32},$$
$$g[k](a) = (t(Vec_{32}(Int_{32}(a) \oplus_{32} Int_{32}(k)))) <<<_{11},,$$
$$a, k \in V_{32}.$$

Функция шифрования g параметризована раундовым ключом k и ставит в соответствие 32х битному входному вектору a 32х битный выходной вектор (преобразование правого подблока в сети Фейстеля). Преобразование включает следующие шаги:

$$Int_{32}(a) \oplus_{32} Int_{32}(k),$$

означающее сложение значения обрабатываемого подблока с раундовым ключом в кольце $Z_{2^{32}}$. Входной подблок и ключ представляются числами в этом кольце, затем выполняется сложение по модулю 2^{32} (операция обозначена как $\oplus_{32}$). Результат опять представляется 32х разрядным вектором, над которым выполняется преобразование *t*:

$$t(Vec_{32}(Int_{32}(a) \oplus_{32} Int_{32}(k))).$$

Результатом подстановки также является 32х битный вектор, над которым выполняется операция циклического сдвига на 11 разрядов влево:

$$(t(Vec_{32}(Int_{32}(a) \oplus_{32} Int_{32}(k)))) <<<_{11}.$$

Раунд сети Фейстеля описывается в виде преобразования $G[k]$:

$$G[k]: V_{32} \times V_{32} \to V_{32} \times V_{32},$$
$$G[k](a_1, a_0) = (a_0, g[k](a_0) \oplus a_1),$$
$$k, a_0, a_1 \in V_{32}.$$

Преобразование, зависящее от раундового ключа, преобразует два 32х битных вектора a_1 и a_0 в два выходных вектора такого же размера. Над вектором a_0 выполняется преобразование g, результат преобразования покомпонентно складывается по модулю 2 с вектором a_1 и записывается в правый выходной блок сети Фейстеля. В левый выходной блок переписывается значение вектора a_0.

Результатом выполнения последнего раунда сети Фейстеля является 64х битный блок шифртекста:

$$G^*[k]: V_{32} \times V_{32} \to V_{64},$$
$$G^*[k](a_1, a_0) = (g[k](a_0) \oplus a_1) \| a_0,$$
$$k, a_0, a_1 \in V_{32}.$$

Последний раунд отличается тем, что результат преобразования g записывается в старшие разряды блока шифртекста, а на место младших разрядов записывается вектор a_0.

Алгоритм развертывания ключа в шифре «Магма» заключается в непосредственном использовании частей секретного ключа *k*. При шифровании используется 32 раундовых ключа k_i по 32 бита каждый:

$$k = \kappa_{255} \| ... \| \kappa_0, \kappa_j \in V_1, j = 0,1,...,255,$$
$$k_1 = \kappa_{255} \| ... \| \kappa_{224}, \quad k_5 = \kappa_{127} \| ... \| \kappa_{96},$$
$$k_2 = \kappa_{223} \| ... \| \kappa_{192}, \quad k_6 = \kappa_{95} \| ... \| \kappa_{64},$$
$$k_3 = \kappa_{191} \| ... \| \kappa_{160}, \quad k_7 = \kappa_{63} \| ... \| \kappa_{32},$$
$$k_4 = \kappa_{159} \| ... \| \kappa_{128}, \quad k_8 = \kappa_{31} \| ... \| \kappa_0.$$

$$k_{i+8} = k_{i+16} = k_i, \quad i = 1,2,...,8,$$
$$k_{i+24} = k_{9-i}, \quad i = 1,2,...,8.$$

Секретный ключ k разбивается на 8 частей $k_1,\ldots,k_8$. С первого по двадцать третий раунды части ключа используются в прямом порядке, в последних восьми раундах – в обратном порядке.

Алгоритм зашифрования E_k реализуется 32 раундами сети Фейстеля:

$$E_k(a) = G^*[k_{32}]G[k_{31}]..G[k_2]G[k_1](a_1, a_0),$$
$$a = a_1 \| a_0 \in V_{64}, \quad a_0, a_1 \in V_{32},$$
$$k_i \in V_{32}, \quad i = 1,2,...,32.$$

Алгоритм расшифрования D_k реализуется той же схемой с обратным порядком следования раундовых ключей:

$$D_k(a) = G^*[k_1]G[k_1]..G[k_{31}]G[k_{32}](a_1, a_0),$$
$$a = a_1 \| a_0 \in V_{64}, \quad a_0, a_1 \in V_{32},$$
$$k_i \in V_{32}, \quad i = 1,2,...,32.$$

На рисунке 1 представлена структурная схема алгоритма шифрования, приведенная в стандарте ГОСТ 28147-89 с обозначением по ГОСТ Р 34.12.

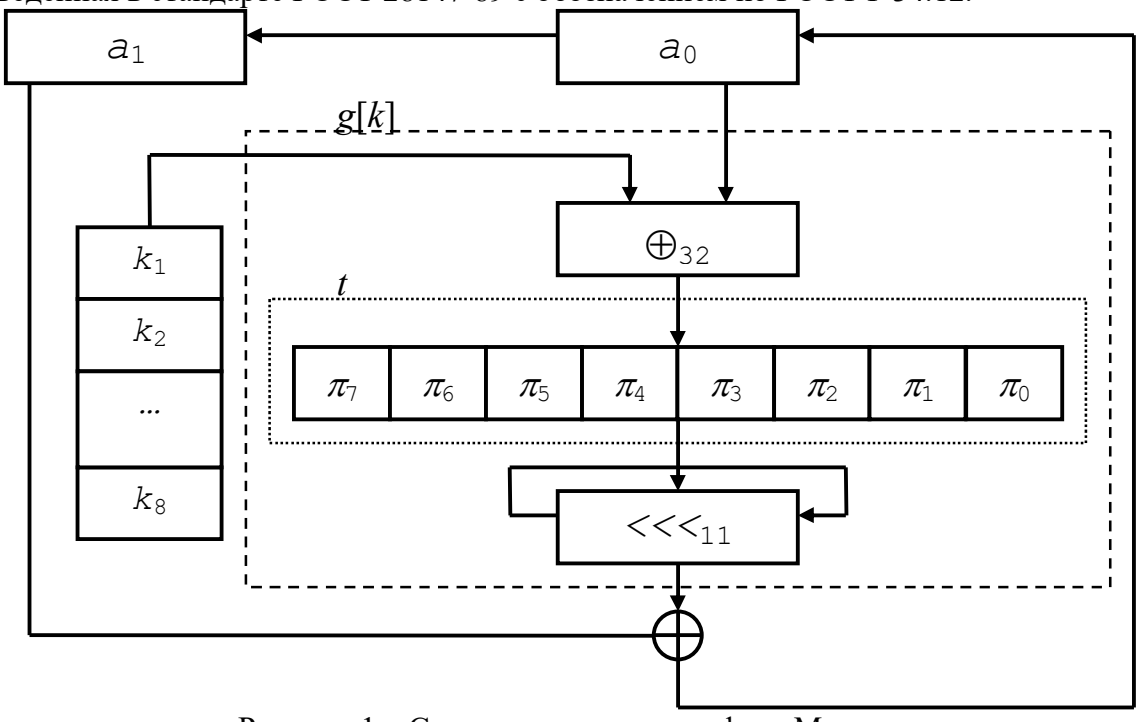

Рисунок 1 – Структурная схема шифра «Магма»