УДК. 681.3

Р.Г.Бияшев, Р.К.Нысанбаев, Р.В. Егай



ПРИМЕНЕНИЕ МОДУЛЯРНОГО ШИФРОВАНИЯ В КОМПЛЕКСЕ ТЕСТИРОВАНИЯ АБИТУРИЕНТОВ

(Институт проблем информатики и управления МОН РК, г. Алма-Ата)

Представляются программные модули, осуществляющие шифрование файлов с правильными ответами для их записи на магнитные диски и расшифровки при их считывании с обнаружением ошибок. При этом доступ к процедуре дешифрования разрешается лишь в случае совпадения трех паролей, вводимых на этапе шифрования файлов.

Программные модули созданы для практической реализации криптографического метода и алгоритмов шифрования и дешифрования, основанных на использовании непозиционной мультипликативной композиции векторов и их представления в виде интерполяционных полиномов Лагранжа, а также метода помехоустойчивого кодирования на основе (n,k)-кода Лагранжа,

примененного для обнаружения и исправления одиночных ошибок в криптограммах.

Отдельно был создан модуль, реализующий формирование электронной цифровой подписи (Уранаев Н.Т., Бияшев Р.Г., Малько К.А.) для обнаружения несанкционированных изменений в листах ответов абитуриентов после их сканирования и записи в БД.

Модули были встроены в программный комплекс тестирования абитуриентов Республики Казахстан и уже в течение ряда лет используются в нем без каких-либо изменений.

Криптографический метод шифрования и дешифрования информации

Вводится следующая интерполяция векторов линейного пространства E_n. Пусть $w_1, w_2, ..., w_n$ - n различных элементов поля F_q $(n \le q)$, упорядоченных некоторым образом.

Под вектором $\vec{a} = (a_1, a_2, ..., a_n)$ понимается многочлен $a(x)$, который в точках $w_1, w_2, ..., w_n$ принимает значения $a_1, a_2, ..., a_n$ соответственно. Тогда под композицией $\vec{a} * \vec{b}$ векторов $\vec{a} = (a_1, a_2, ..., a_n)$ и $\vec{b} = (b_1, b_2, ..., b_n)$ понимается многочлен, принимающий в точках $w_1, w_2, ..., w_n$ значения $a_1 * b_1, a_2 * b_2, ..., a_n * b_n$.

Суть криптографического метода заключается в следующем. Пусть $p = 2$, $p(x)$ - неприводимый многочлен степени m, порождающий поле Галуа $GF(2^m)$.

Предположим, что шифруется некоторое исходное сообщение M длины L двоичных разрядов. Сообщение M интерпретируется как многочлен

$$M(x) = (a_1(x), a_2(x), ..., a_n(x)), \qquad (1)$$

где $a_i(x), i=\overline{1,n}$ - значения, которые он принимает в точках $w_i \in GF(2^m), i=\overline{1,n}$. В (1) $a_1(x), a_2(x), ..., a_n(x)$ выбираются так, что первым l_1 битам слова M ставятся в соответствие двоичные коэффициенты многочлена $a_1(x)$, следующим l_2 битам – двоичные коэффициенты $a_2(x)$, и так далее, наконец, последним l_n двоичным разрядам ставятся в соответствие двоичные коэффициенты полинома $a_n(x)$.

В процессе шифрования исходного сообщения длина секретного ключа K задается равной длине сообщения. Тогда ключ, также как и исходное сообщение M интерпретируется как многочлен

$$K(x) = (b_1(x), b_2(x), ..., b_n(x)), \qquad (2)$$

где $b_i(x), i=\overline{1,n}$ - значения, которые принимает многочлен $K(x)$ в точках $w_i \in GF(2^m), i=\overline{1,n}$.

Умножая (1) и (2) в выбранном поле, получаем некоторый многочлен

$$F(x) = M(x)K(x), \qquad (3)$$

значения которого в узлах w_i вычисляются посимвольно

$$g_i(x) \equiv a_i(x)b_i(x) \text{ (modd 2, } p(x)), \qquad (4)$$

тогда можно записать

$$F(x) = (g_1(x), g_2(x), ..., g_n(x)). \qquad (5)$$

Здесь $a_i(x), b_i(x), g_i(x)$ - элементы поля $GF(2^m)$. Из этого следует, что количество двоичных разрядов, соответствующих двоичным коэффициентам представления $a_i(x), b_i(x), g_i(x)$ для всех $i=\overline{1,n}$ имеет одинаковую длину, равную m, т.е. $l_i = m,\ i=\overline{1,n}$.

Двоичная запись длины L полученного многочлена $F(x)$ - зашифрованное сообщение (криптограмма).

В данном случае подлежат проверке $2^L \cdot n$ ключей, где L - длина ключа, а n - количество неприводимых многочленов с двоичными коэффициентами степени m.

Для дешифрования необходимо найти такой многочлен $K^{-1}(x)$, удовлетворяющий следующему сравнению

$$K^{-1}(x)F(x) \equiv M(x) \text{ (modd 2, } p(x)\text{)}, \qquad (6)$$

где

$$K^{-1}(x) = (b_1^{-1}(x), b_2^{-1}(x), \ldots, b_n^{-1}(x)). \qquad (7)$$

Для многочленов $b_i^{-1}(x), i = \overline{1,n}$ справедливы сравнения

$$K^{-1}(x) \equiv b_i^{-1}(x) \text{ (modd 2, } p(x)\text{)},$$

или

$$b_i^{-1}(x) b_i(x) \equiv 1 \text{ (modd 2, } p(x)\text{)}. \qquad (8)$$

Таким образом, определение элементов $b_i^{-1}(x), i = \overline{1,n}$ производится по выражению (8). Затем по формуле (6) восстанавливается значение многочлена $M(x)$, который и является исходным сообщением.

Алгоритм обнаружения и исправления одиночных ошибок в криптограммах на основе (n,k) - кода Лагранжа

Рассматривается код Лагранжа для исправления одиночных ошибок в криптограммах с использованием двух контрольных символов. Под ошибкой понимается любое искажение информационного или контрольного символа.

Пусть $g_1(x), g_2(x), \ldots, g_n(x)$ - информационные символы, тогда контрольные символы вычисляются в соответствии с выражениями

$$g_{n+1}(x) = \sum_{i=1}^{n} g_i(x) L^{(i)}(x_{n+1}),$$

(9)

$$g_{n+2}(x) = \sum_{i=1}^{n} g_i(x) L^{(i)}(x_{n+2}),$$

и зашифрованное сообщение

$$g_1(x), g_2(x), \ldots, g_n(x), g_{n+1}(x), g_{n+2}(x) \qquad (10)$$

хранится, пересылается или передается на обработку. Здесь $\forall g_i, L^{(i)}(x_{n+1}),\ L^{(i)}(x_{n+2}), 1 \le i \le n,$ являются элементами заданного конечного поля $GF(2^m)$. Если в кодовом слове произошла ошибка в i - том символе, $\overline{g_i} = g_i + \Delta g_i$, то после повторного вычисления контрольных символов g'_{n+1}, g'_{n+2} можно найти

$$\nabla_1 = g'_{n+1}(x) \oplus g_{n+1}(x) = \Delta g_i(x) L^{(i)}(x_{n+1}),$$

(11)

$$\nabla_2 = g'_{n+2}(x) \oplus g_{n+2}(x) = \Delta g_i(x) L^{(i)}(x_{n+2}).$$

Невязки ∇_1, ∇_2 однозначно определяют величину и местоположение ошибки. Очевидно, что если $\nabla_1 = \nabla_2 = 0$, ошибки нет; если $\nabla_1 \ne 0, \nabla_2 = 0$ или $\nabla_1 = 0, \nabla_2 \ne 0,$ ошибка соответственно в первом или втором контрольном символе; если $\nabla_1 \ne 0, \nabla_2 \ne 0$, то ошибка произошла в одном из информационных символов.

Если выбрать $x_{n+1} = 0,\ \ x_{n+2} = 1$, то

$$L^{(i)}(x_{n+1}) = 1, \qquad L^{(i)}(x_{n+2}) = i, \quad i = \overline{1,n}.$$

Выражение (9) для контрольных символов принимает вид

$$g_{n+1}(x) = \sum_{i=1}^{n} g_i(x),$$

$$g_{n+2}(x) = \sum_{i=1}^{n} i g_i(x), \tag{12}$$

а выражения для невязок (11) соответственно:

$$\nabla_1 = g'_{n+1}(x) \oplus g_{n+1}(x) = \Delta g_i(x),$$

$$\nabla_2 = g'_{n+2}(x) \oplus g_{n+2}(x) = i\Delta g_i(x). \tag{13}$$

В этом случае первое равенство (13) определяет величину ошибки, а номер ошибочного символа определяется умножением второго равенства (13) на мультипликативную инверсию величины ошибки, т.е.

$$\nabla_2 \nabla_1^{-1} = i\Delta g_i(x)(\Delta g_i(x))^{-1} = i.$$

Практическая реализация криптографического метода

Для описанного криптографического метода и алгоритма обнаружения ошибок в криптограммах на основе (n,k) - кода Лагранжа с целью шифрования и дешифрования информации, хранящейся на магнитных дисках, создано программное обеспечение на языке Delphi 4.0, работающее под операционной системой Windows 95/98. Программы шифруют и расшифровывают данные по блокам длиной $n = 254$ байта. Соответственно длина ключа также равна $n = 254$ байта. В качестве модуля был выбран неприводимый многочлен восьмой степени

$$p(x) = x^8 + x^4 + x^3 + x + 1.$$

Программное обеспечение удовлетворяет следующим требованиям: 1) доступ к процедуре дешифрования осуществляется лишь в случае совпадения трех паролей; 2) если в зашифрованном файле с кодами правильных ответов появились любого рода ошибки, то при считывании файла с магнитного диска на экране монитора выдается соответствующее сообщение, и в этом случае дешифрование искаженного шифрованного файла не выполняется.

Описание алгоритма шифрования

Суть алгоритма шифрования заключается в следующем.

1) Начало выполнения программы.

2) Ввод имени исходного файла, содержащего исходное сообщение M.

3) Задание имени шифрованного (выходного) файла, в который будет записываться зашифрованное сообщение.

4) Ввод трех паролей. В случае неправильного их задания, ввод паролей повторяется.

5) Ввод исходного сообщения M из исходного файла по блокам длиной n байт каждый.

6) Задание модуля $p(x)$ любым неприводимым многочленом восьмой степени. Шифрование производится поблочно. Каждому байту исходного сообщения ставятся в соответствие полиномы $a_i(x)$, $i = \overline{1,n}$, которые являются остатками по модулю $p(x)$. Коэффициентами остатков $a_i(x)$ будут биты соответственных байтов исходного сообщения.

7) Ввод ключа K, в качестве которого выбирается произвольная последовательность длины n байт. Тогда по аналогии с пунктом 6 каждому байту ключа K ставятся в соответствия многочлены $b_i(x)$, $i = \overline{1,n}$.

8) Умножение каждого из многочленов $a_i(x)$ на

соответственный многочлен $b_i(x)$, $i=\overline{1,n}$. В результате получаются некоторые многочлен $A_i(x)=a_i(x)b_i(x)$, $i=\overline{1,n}$, двоичные коэффициенты которых представляют собой пятнадцатиразрядное двоичное слово.

9) Вычисление по формуле (4) значений многочленов $g_i(x)$, $i=\overline{1,n}$. Двоичные коэффициенты полученных остатков $g_i(x)$ представляют собой зашифрованные байты.

10) Обнаружение ошибок в криптограмме: для этого используются два избыточных узла $x_{n+1}=0$ и $x_{n+2}=1$. Тогда контрольные символы $g_{n+1}(x)$ и $g_{n+2}(x)$ рассчитываются по формулам (12).

11) Запись всех вычисленных $g_i(x),\ i=\overline{1,n+2}$ в выходной файл.

12) Проверка завершения считывания исходного сообщения по блокам: когда ввод блоков закончен, то выполнение программы завершается. Иначе повторение выполнения программы, начиная с пункта 8.

13) Конец выполнения программы.

На рис. 1 изображена блок-схема изложенного алгоритма шифрования.

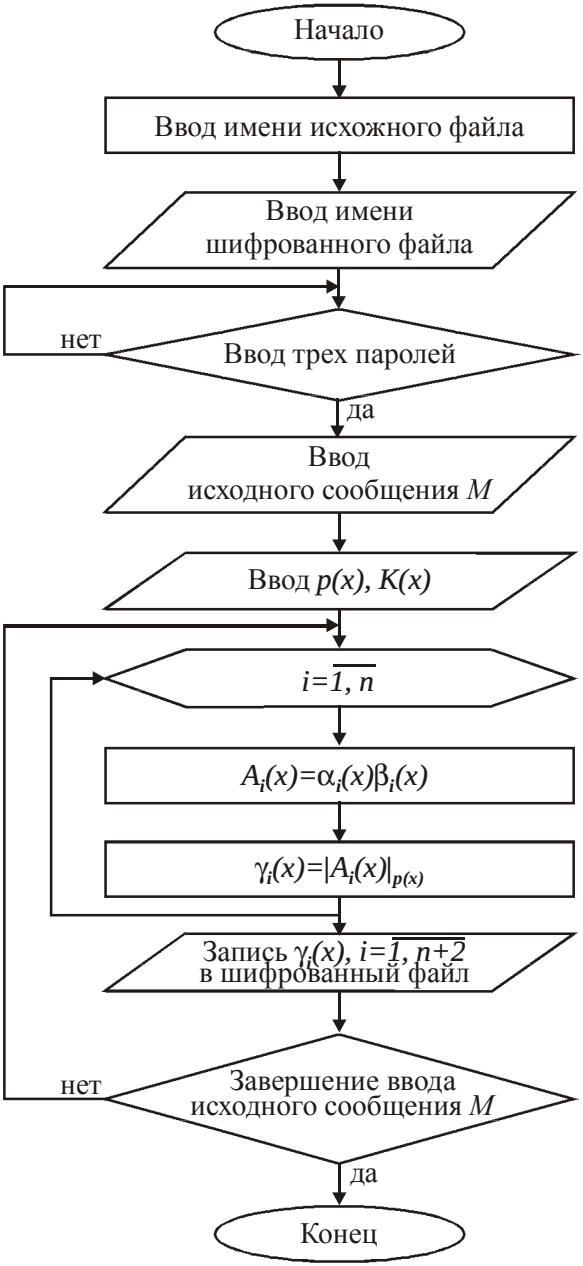

Рис. 1. Блок-схема алгоритма шифрования

Описание алгоритма дешифрования

Алгоритм дешифрования.

1) Начало выполнения программы.

2) Задание имени шифрованного файла, содержащего зашифрованное сообщение F.

3) Ввод имени дешифрованного файла, куда будет записываться расшифрованное сообщение.

4) Ввод трех паролей, которые указывались при шифровании исходного сообщения. Если один или несколько паролей заданы неверно, то необходимо повторить их ввод. Процедура ввода паролей может повторяться только три раза.

5) Если все три пароля введены верно, то задается шифрованное сообщение F по блокам длиной n байт каждый. Дешифрование производится поблочно.

6) Ввод модуля $p(x)$ и ключа $K(x)$, которые использовались в процессе шифрования. Каждому байту криптограммы ставятся в соответствие полиномы $g_i(x)$, $i=\overline{1,n}$, которые являются остатками по модулю $p(x)$. Коэффициентами остатков $g_i(x)$ будут биты соответственных байтов зашифрованного сообщения.

7) Обнаружение ошибок считывания зашифрованного сообщения: для этого вычисляются контрольные символы $g'_{n+1}(x)$ и $g'_{n+2}(x)$ по формулам (12).

8) Если $g_{n+1}(x)=g'_{n+1}(x)$ и $g_{n+2}(x)=g'_{n+2}(x)$, то ошибки в криптограмме нет, а если $g_{n+1}(x)\neq g'_{n+1}(x)$ и(или) $g_{n+2}(x)\neq g'_{n+2}(x)$, то в зашифрованном сообщении произошла ошибка.

9) Если в криптограмме есть ошибки, то выполнение программы завершается. Если же ошибок нет, то для каждого $b_i(x)$ ключа K, $i=\overline{1,n}$ вычисляются инверсные

многочлены $b_i^{-1}(x)$ по формуле (8).

10) Каждый из многочленов $b_i^{-1}(x)$ умножается на соответственный многочлен $g_i(x)$. В результате получаются некоторые многочлены $A_i^{-1}(x),\ i=\overline{1,n}$.

11) По формуле (6) производится восстановление многочленов $a_i(x),\ i=\overline{1,n}$.

12) Двоичные коэффициенты остатков $a_i(x)$ являются байтами расшифрованного, т.е. исходного сообщения.

13) Запись в дешифрованный файл всех вычисленных $a_i(x),\ i=\overline{1,n}$.

14) Проверка завершения считывания шифрованного файла. Если ввод завершен, то работа программы завершается, иначе управление передается на пункт 5.

15) Конец выполнения программы.

Блок-схема алгоритма дешифрования приведена на рис. 2

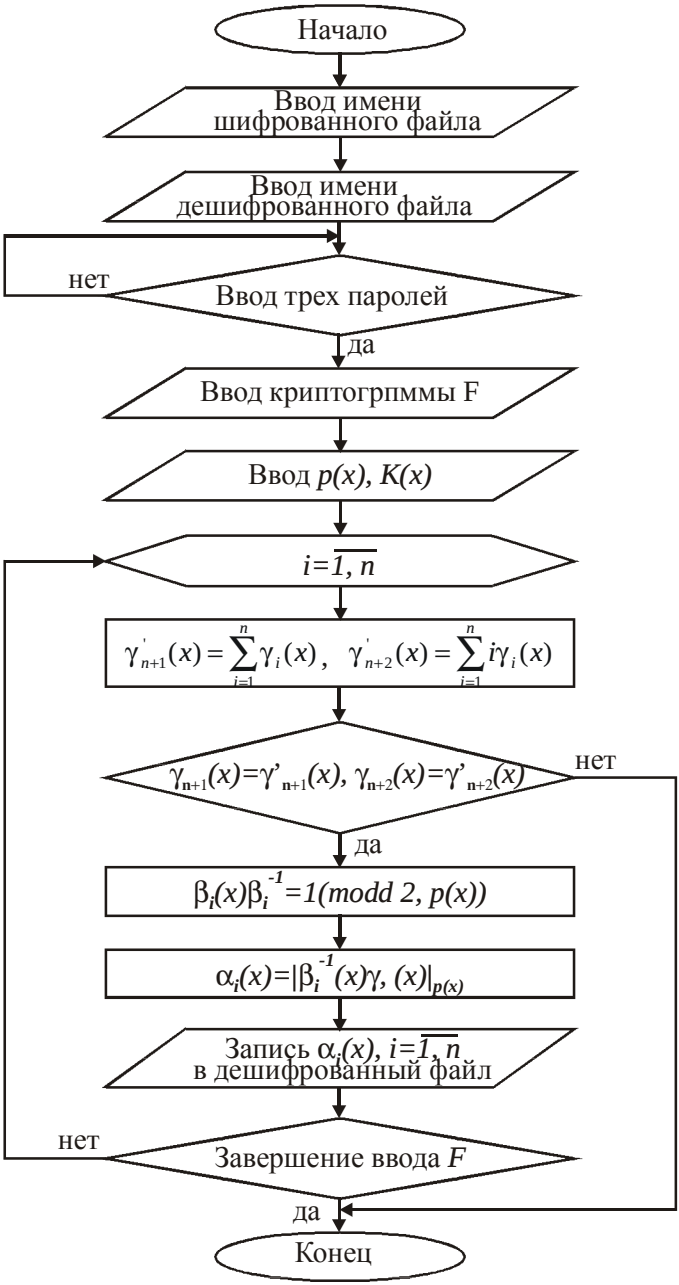

Рис. 2. Блок-схема алгоритма дешифрования