УДК 681.3

Бардаченко В.Ф., Клименко А.А.
Центр таймерных вычислительных систем ИК
им.В.М.Глушкова НАН Украины

Концептуальные основы применения таймерного троичного кодирования в системах защиты информации

Рассмотрены методы кодирования. Авторами разработан программный продукт, который осуществляет блочное гаммирование с использованием G -последовательностей, символы которых принадлежат алфавитам с различными основаниями (в частности использовалась двоичная и троичная G -последовательности). Авторы утверждают, что шифрование различной информации с помощью блочного гаммирования является достаточно удобным средством сокрытия истинной информации от несанкционированного доступа.

In this paper we consider the problem of existing a new methods representation information. The methods permitting to create new G-sequence.

Введение

Одним из широко распространенных методов кодирования является наложение по известному правилу на открытый текст (ОТ) последовательности с определенным набором свойств. Вследствие этого наложения ОТ изменяется и превращается в шифрованный текст(ШТ), который затем передается по каналу связи получателю. Такая накладываемая последовательность называется гаммирующая или G -последовательность, а сам метод шифрования – шифрование гаммированием. Гаммирование – один из методов потокового шифрования, в котором каждый символ ОТ последовательно подвергается обработке.

Известно, что одним из критериев, по которому возможна классификация кодов, является алфавит. Коды бывают двоичные и многоосновные (алфавит которых содержит более 2 символов). Нас будут интересовать многоосновные коды, в частности троичный, потому что возникла идея использовать в качестве G -последовательности именно троичную последовательность [1].

Создание троичной G -последовательности

Первой проблемой, с которой пришлось столкнуться, реализуя шифрование гаммированием, явилось создание качественной троичной G-последовательности. Есть два принципиально различных метода решения этой проблемы:

а) создание троичной G -последовательности на основе качественной двоичной;

б) разработка алгоритма создания троичной G -последовательности, не прибегая к помощи двоичных последовательностей.

Пункт а) оказался менее трудоемким, к тому же проблема создания качественной двоичной последовательности уже давно и успешно решена. Известно достаточно много методов создания таких последовательностей. Исходя из всего вышеизложенного, было предложено два алгоритма создания троичной G -последовательности. Опишем и подробно исследуем каждый из этих методов.

1. Исходная двоичная последовательность рассматривается как десятичное число, представленное в двоичной форме. Последовательность разбивается на подпоследовательности фиксированной длины L. Далее этот двоичный код преобразуется в троичный. Вообще говоря, разбиения можно устраивать различными способами, при этом вовсе не обязательно разбивать на подпоследовательности фиксированной длины.

Пример. Пусть $L =$. Каждую такую подпоследовательность преобразуем в троичный код. Если номер подпоследовательности нечетный, то к троичному коду добавим 1, в противном случае 0. Это делается для того, чтобы на выходе могла появиться троичная подпоследовательность ‘22’, которая не может быть получена прямым преобразованием нашей двоичной последовательности. Это обыкновенное математическое ухищрение, целью которого является улучшение качества выходной последовательности. Прежде всего, речь идет о приближении вероятностей появления символов троичного алфавита к идеальному случаю – когда появление всех символов равновероятно.

Максимальное десятичное число, представимое 3-мя битами информации равно 7. Для представления всех десятичных чисел до 7 включительно в троичном коде достаточно 2 –х тритов информации. Следовательно, из одной исходной двоичной подпоследовательности длины $L = 3$ на выходе получим троичную подпоследовательность длины $D = 2$. Пусть имеем N таких двоичных подпоследовательностей. Тогда длина исходной двоичной последовательности $L2 = 3 * N$, а длина выходной троичной $D = 2 * N$. Таким образом выходная последовательность на треть уступает исходной по мощности.

Теперь рассмотрим некоторые характеристики полученной троичной последовательности. Посчитаем вероятности появления 0,1,2 в нашей последовательности. Для наглядности представим полученные результаты в табл. 1.

Таблица 1

Входная подпоследовательность	**Выходная подпоследовательность**	**Вероятность**
000	00	0.0625
001	01	0.125
010	02	0.125
011	10	0.125
100	11	0.125
101	12	0.125
110	20	0.125
111	21	0.125
111	22	0.0625

Первый столбец в таблице – это все возможные двоичные подпоследовательности. Второй – соответствующая троичная подпоследовательность. Третий – вероятность появления такой троичной подпоследовательности на выходе. Эта вероятность рассчитывалась следующим образом.

Так как исходная двоичная последовательность качественная, то вероятности появления 0 и 1 равны и статистически независимы. Следовательно, вероятность появления среди разбиений исходной двоичной последовательности конкретной подпоследовательности будет равна $0.125\ 0.5^{*}0.5\ 0.5\ )$. Но ведь мы добавляем ровно в половине случаев к троичному коду 1. Поэтому получить на выходе 00 можем лишь в случае одновременного выполнения двух условий:

– исходная двоичная подпоследовательность равна 000;
– текущая подпоследовательность имеет четный номер;

$P\ 000) = 0.125$, а четный номер подпоследовательности лишь в половине случаев. Отсюда имеем вероятность на выходе $P\ 00) = P\ 000)/2 = 0.0625$. Аналогично рассчитаем вероятность получить на выходе 22. Для этого необходимо выполнение следующих двух требований :

– исходная двоичная подпоследовательность равна 111;
– текущая подпоследовательность имеет нечетный номер;

Очевидно, что, следуя тем же рассуждениям, получим P(22)=0.0625.

Что касается вероятностей появления остальных комбинаций, то они равновероятны. Действительно, каждая из них может быть получена в двух случаях : путем прямого преобразования двоичной подпоследовательности без добавления 1(вероятность этого события равна 0.0625) и добавлением к уже преобразованному из двоичного троичному коду 1, если номер подпоследовательности нечетный(вероятность этого события также равна 0.0625). Сложив вероятности этих событий, получим, что появление на выходе троичных подпоследовательностей отличных от 00 и 22 равновероятно и вероятность этого равна 0.125

Посчитаем вероятности появления в 3-ной последовательности 0,1,2. Для этого воспользуемся формулой полной вероятности. Она имеет следующий вид:

В нашем случае $P(H\ )$ – это вероятности появления на выходе соответствующей троичной подпоследовательности, а $P(A/H\ )$ – вероятность появления среди этой пары символа алфавита. Так аналогично $P(1) = 0.375$, а $P(2) = 0.3125$.

Как видим, вероятности появления различных символов алфавита несколько отличаются от идеального соотношения. Однако, это вовсе не означает, что предложенный метод создания 3-ной последовательности не может быть использован. Вот лишь некоторые из возможных направлений, где предложенный алгоритм может быть использован довольно эффективно:

— для кодировки информации, не представляющей государственную или коммерческую тайну;
— для передачи информации небольшого объема, которая, даже будучи атакована криптоаналитиком противника, не позволит применить статистический анализ из-за невозможности набрать достоверную статистику;

— для передачи информации, ценность которой ниже затрат, необходимых на преобразование перехваченного ШТ в ОТ;
— для сокрытия информации, актуальность которой после получения ее адресатом резко уменьшается с течением времени.

Этот список можно еще долго продолжать. Однако и вышеперечисленные направления применения предлагаемого алгоритма достаточно убедительно показывают его перспективность.

В дополнение ко всему, он очень прост в реализации и не требует привлечения супермощной техники. В современных условиях быстрота так же является далеко не последним параметром, с помощью которого оценивается удачность предлагаемого метода решения проблемы.

На этом анализ первого из предложенных алгоритмов создания 3-ной последовательности завершим. Перейдем ко второму.

2. Из исходной качественной двоичной последовательности по известному правилу выбирается пара символов, которые затем анализируются. На выходе – один символ троичного алфавита. Вот один из возможных вариантов:
— выбираются текущий элемент последовательности и следующий за ним через p элементов Y);
— если среди выбранных элементов нет «2» и пара не (1,1), то на выходе получаем «2»;
— если среди выбранных элементов нет «1» и пара не (0,0), то на выходе получаем «1»;
— если среди выбранных элементов нет «0» и пара не (2,2) то на выходе получаем «0»;
— теперь X становится следующий элемент, а Y —следющий за ним через p элементов, т.е. идет сдвиг вправо на 1. И так до последнего элемента последовательности.

Посмотрим, характеризуется ли данный метод равномерным законом распределения чисел? Учитывая, что исходная двоичная последовательность качественная, возникновение всех 9-ти пар равновероятно, и появление конкретной пары происходит с вероятностью равной 1/9. Так как количество пар, дающих на выходе 0,1,2, равно, то с учетом сказанного ранее и используя классическое определение вероятности, получим :

$$P\ 0) = P(1\ = P(2) = 3/9 = 1/3.$$

Как видим, предложенный алгоритм создания качественной гаммирующей последовательности удовлетворяет требованию о равновероятности появления всех символов алфавита.

Для увеличения мощности выходной последовательности можно организовать несколько альтернативных разбиений исходной последовательности на подпоследовательности. Так же можно различными способами выбирать анализируемые элементы. Возможно одновременное применение обоих методов и создание новых алгоритмов на их основе.

Зачем же создавать качественную троичную последовательность? Может быть можно было обойтись двоичной G - последовательностью? Безусловно,

вполне можно использовать в качестве G - последовательности двоичную, однако использование троичной делает криптосистему более устойчивой в случае нападения на последнюю криптоаналитика противника. Докажем это.

Утверждение

Пусть при передаче сообщения по каналу связи произошла утечка информации и противнику удалось перехватить сообщение длины L. Противник осведомлен о том, что для сокрытия истинного текста использовалось шифрование гаммированием. Предположим, что работу над расшифровкой перехваченного ШТ ведут два криптоаналитика. Один проверяет гипотезу о наложении 2-ной последовательности, другой – 3-ной. Тогда для того, чтобы перебрать все возможные последовательности второму понадобится проанализировать в 1.5 раз больше вариантов.

Доказательство

Количество комбинаций, которые необходимо проанализировать, чтобы с вероятностью, равной 1, успешно расшифровать перехваченный ШТ (при этом все же необходимо знать особенности наложения G -последовательности на ОТ) при использовании G -последовательности, порожденной алфавитом А, определяется величиной равной произведению основания алфавита А на себя столько раз, какова длина последовательности. В нашем случае первому и второму криптоаналитикам будет необходимо проанализировать 2 и 3 вариантов соответственно. Получается, что 2-му придется проанализировать в 1.5 раз вариантов больше. Утверждение доказано.

Приведем графическую интерпретацию (рис. 1) резкого повышения криптостойкости системы в случае, когда в распоряжении противника лишь ШТ и предпринимается атака криптоаналитиком противника путем перебора всех возможных вариантов G -последовательности и обратного наложения каждой такой последовательности на перехваченный ШТ.

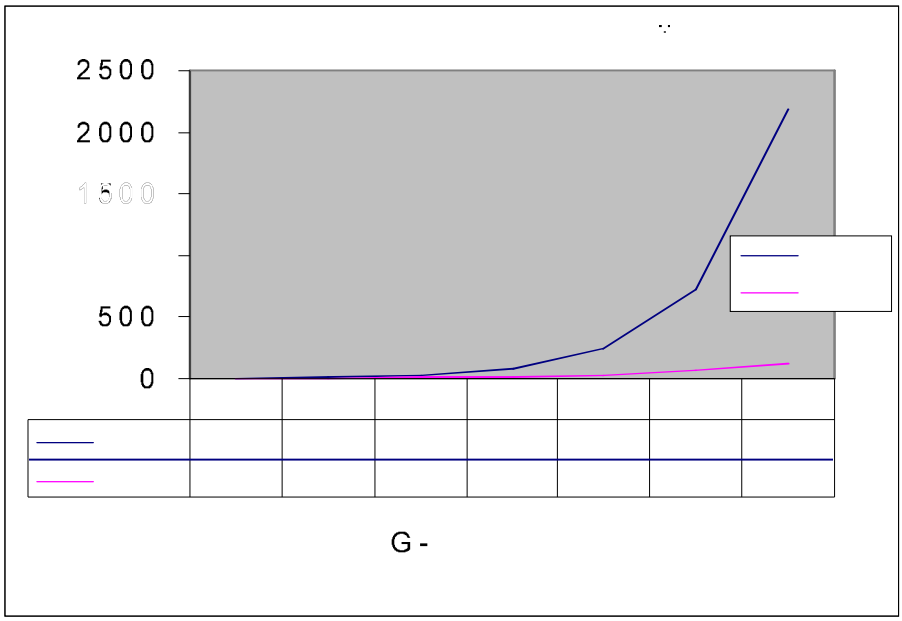

Рис.1

Ряд1 показывает, какое количество комбинаций соответствующей длины необходимо перебрать, чтобы проанализировать все возможные варианты при использовании троичной G - последовательности. Соответственно Ряд2 – при использовании двоичной G - последовательности. Для иллюстрации взяты сравнительно малые длины сообщений, так как при увеличении длины сообщения скорости роста этих зависимостей очень разнятся, что крайне трудно увидеть на одной координатной сетке, не прибегая к какому-либо масштабированию. Так, например, при длине в 100 символов количество вариантов при использовании троичной последовательности в $4*10$ раз больше, чем при двоичной.

Рассмотрим следующий вопрос. двоичная G - последовательность довольно давно и успешно применяется для кодирования информации. Каким образом и что именно поменяется при использовании вместо двоичной G - последовательности троичной, и поменяется ли вообще что-либо? Но прежде докажем вспомогательную лемму.

Лемма

Шифрование ОТ проводится накладыванием G - последовательности без учета переноса разряда(поэлементное сложение). Тогда, зная G - последовательность и, имея ШТ, можно однозначно определить ОТ.

Доказательство

Пусть ОТ представлен с помощью алфавита А, содержащего $n1$ различный символ, а G -последовательность представлена с помощью алфавита AG, основание которого равно $n2$. Тогда понятно, что сложение происходит по модулю k, где $k = \max(n1, n2)$. Рассматривать процесс восстановления ОТ будем поэлементно.

Докажем утверждение леммы от противного. Пусть:

X, Y — предполагаемые элементы ОТ;

– — элемент G - последовательности;

H – элемент ШТ;

Теперь пусть существует хотя бы два элемента X и Y, причем $X < Y$, такие, что при сложении с известным символом G - последовательности дают известный символ ШТ, т.е.

$$(X + G) \bmod k = H$$

$$Y + G) \bmod k = H$$

Так как $X < k, G < k$ то $X + G < 2\ k$. Аналогично $Y + G < 2\ k$. Учитывая это и вышеизложенное должно выполняться следующее: $X + G = H$. Отнимем от второго уравнения первое: $Y - X = k$. Мы пришли к противоречию, так как $|Y| < k, X > 0$. Следовательно, наше предположение о существовании хотя бы двух различных символов, наложение на которые G - последовательности даст один и тот же символ ШТ – неверно.

Утверждение

Использование качественной 3-ной G -последовательности, точно так же как и использование 2-ной G -последовательности, приводит к равновероят-

ному появлению всех символов используемого алфавита в ШТ, при условии равновероятного закона распределения чисел в ОТ.

Доказательство

Проверим равновероятность появления 0,1,2 (в тех случаях когда она вообще может появиться). Наложение G-последовательности будем проводить следующим образом : последовательности будут поэлементно складываться без учета переноса разряда(попытка сохранять признак переноса удлинит последовательность, к тому же возникнет проблема вставки в уже существующую последовательность признаков переноса разряда, не нарушая при этом однородности сообщения). А, зная G-последовательность и имея ШТ, можно однозначно определить T и без сохранения переноса разряда(см. лемму). Рассмотрим все возможные варианты. Их всего четыре. Когда OT и G-последовательность представлены двоичным кодом, троичным и еще два варианта, когда одна из последовательностей двоичная, а другая троичная. Полученные результаты поместим в табл. 2.

Таблица 2

OT	G	ШТ					
		0	1	2	P(0)	P(1)	P(2)
2	2	(0,0) (1,1)	(0,1) (1,0)	---	0.5	0.5	0
2	3	(0,0) (1,2)	(0,1) (1,0)	(0,2) (1,1)	0.(3)	0.(3)	0.(3)
3	2	(0,0) (2,1)	(0,1) (1,0)	(2,0) (1,1)	0.(3)	0.(3)	0.(3)
3	3	(0,0) (1,2) (2,1)	(1,0) (0,1) (2,2)	(0,2) (2,0) (1,1)	0.(3)	0.(3)	0.(3)

В табл. 2 два первых столбца содержат основания алфавита, с помощью которого представлены ОТ и G-последовательность. Все то, что находится под шапкой ШТ относится к шифрованному тексту. Столбец ШТ включает в себя шесть подстолбцов. Три левых столбца иллюстрируют возможности появления в ШТ соответствующего символа. На пересечении строки с указанием представления ОТ и G-последовательности и столбца, соответствующего одному из символов G-последовательности записаны пары. Первый элемент такой пары – это символ ОТ, второй – G-последовательности. Одновременное появление этих символов при накладывании G- последовательности на ОТ и дает в ШТ символ, открывающий этот столбец. Всего таких столбца три. Последние три столбца – это вероятности появления в ШТ 0,1,2 в зависимости от оснований используемых алфавитов. Эти вероятности рассчитаны по классическому определению вероятности. Проведенные рассчеты показали, что появление в ШТ 0,1,2 (когда возможно) происходит с одинаковой вероятностью.

Как видно из таблицы, применение 3-ной G-последовательности сохраняет равномерный закон распределения в ШТ. Утверждение доказано.

Исследованы необходимые ресурсы для попытки криптоатаки на предложенную криптосистему на основе перехваченного ШТ. Проведен эксперимент на ПК с процессором P166MMX. Постановка задачи: перехвачено сообщение, зашифрованное с помощью наложенной G-последовательности. Необходимо реализовать перебор всех возможных G-последовательностей, осуществить обратное наложение каждой из этих последовательностей на ШТ и проанализировать временные затраты на эту работу, а также оценить общее количество печатной информации, которую после обратного наложения G-последовательности на ШТ придется анализировать криптоаналитику противника. Исходными данными этого эксперимента являлись следующие параметры:

— длина перехваченного сообщения;

— основание используемого алфавита;

G - последовательности – без учета переноса разрядов;

Эксперимент дал следующие результаты.

Таблица 3

Длина сообщения	**2-ное представление**		
	Количесто вариантов, шт	**Печатный объем, том**	**Необходимое время**
10	1024	0.0102	0:0:0:0
13	8192	0.106	0:0:0:5
15	32768	0.5	0:0:0:54
20	1048576	20	0:0:16:31
25	33554432	8389	0:12:19:7
30	10737741824	32216	до 13ч
50	1.13*10{15}	5.6*10{10}	10957дн

Таблица 4

Длина сообщения	**3-ное представление**		
	Количесто вариантов, шт	**Печатный объем, том**	**Необходимое время**
10	59049	1.48	0:0:0:54
13	1594323	39.85	0:0:20:59
15	14348907	229	0:4:21:17
20	3486784401	55780	до 21ч
25	847288609443	1.4*10{7}	393дн
30	2.2*10{14}	3.3*10{9}	178804дн
50	7.2*10{23}	1.1*10{19}	7.6*10{15}дн

В табл. 3, 4 «Количество вариантов» означает количество различных G-последовательностей, с помощью которых могло быть закодировано сообщение соответствующей длины. В этих таблицах для удобства представления и наглядности объем представлен в томах, где один том содержит 500 страниц, каждая из которых содержит 2000 печатных символов. Необходимое время – время на генерацию всех вариантов и обратное наложение выбранной G-последовательности на ШТ. Здесь не учтено время, необходимое на представление

полученного обратного наложения в печатном виде(с помощью принтера или вывод полученного ОТ непосредственно на экран монитора), а так же время на анализ полученного результата криптоаналитиком противника. Очевидно, что учет этих составляющих сделает ресурс времени абсолютно недостижимым в реальных условиях актуальности информации. Время в этих таблицах кроме дней представлено в следующем формате: Часы: Минуты: Секунды: Сотые. один день равен 24 ч.

Все вышеизложенное относится к так называемому каноническому гаммированию. Каноническое гаммирование предполагает преобразование открытого текста (ОТ) в числовую последовательность, на которую по известному уже закону накладывается G -последовательность. Эта последовательность должна обладать целым набором свойств для успешного применения в качестве G -последовательности. Основными требованиями, которым должна удовлетворять гаммирующая последовательность, являются равномерный закон распределения чисел и достаточно большой период. Применение гаммирования как метода сокрытия истинного текста довольно эффективно, так как в случае перехвата шифрованного сообщения противником, последнему придется перебрать множество вариантов и привлечь для их анализа необходимое количество опытных криптоаналитиков. Все изложенные моменты убеждают в целесообразности использования гаммирования в качестве метода потокового шифрования.

Рассмотрим основные проблемы, которые необходимо решать, реализуя шифрование с помощью G -последовательности. Сначала необходимо построить генератор псевдослучайных чисел, который будет генерировать накладываемую последовательность с заданными параметрами. Затем необходимо каждой букве алфавита, с помощью которого представлен ОТ, поставить в соответствие числовую последовательность. Таким образом, после подстановки в ОТ соответствующих числовых последовательностей ОТ, будет представлять собой просто набор цифр. Кроме этого, есть еще знаки препинания, которыми также нельзя пренебречь (классический пример «Казнить нельзя, помиловать») и цифры. Буквы бывают еще и заглавными. С учетом всего сказанного, приходится (понятное дело, для каждого алфавита и предметной области это число различно) определять не менее чем 40 (сорок) числовых последовательностей, соответствующих какому-либо символу в ОТ.

Понятно, что длины этих последовательностей зависят от выбранного алфавита и его основания. Посчитаем эти длины при использовании двоичного и троичного (используется троичная G -последовательность и ОТ представлен троичным кодом) алфавитов. Данные занесем в таблицу.

Таблица 5

Длина последовательности	Двоичное гаммирование	Троичное гаммирование
1	2	3
2	4	9
3	8	27
4	16	81
5	32	243
6	64	729

Даже беглый взгляд на предложенную таблицу убедительно показывает, что использование троичного гаммирования более предпочтительно. Так, выделив 4 разряда для представления символов алфавита, использование двоичного гаммирования представляется очень затруднительным, если не сказать больше. Троичное же гаммирование предоставляет достаточное количество комбинаций даже для специальных символов. При использовании же двоичного понадобится не менее 6 разрядов, да и они не дадут той эффективности как 4 разряда при троичном. Но даже при троичном гаммировании и выделении всего 4-х разрядов на представление каждого символа из ОТ длина нашего сообщения увеличивается ровно в 4 раза.

Проанализировав полученные результаты, понимая что информация бывает различной важности и секретности, предлагается использовать так называемое блочное гаммирование. Его суть заключается в следующем: на ОТ накладывается G - последовательность. В зависимости от того, какой символ накладывается на элемент ОТ, на выходе шифрующего устройства получается символ соответствующей строки матрицы замены. Матрица замены представляет собой матрицу, каждая строка которой содержит все элементы, которые могут встретиться в ОТ. При этом количество строк в матрице равно основанию алфавита, используемого при создании G -последовательности, плюс единица (строка содержащая информацию о символах ОТ). Будем считать, что первая строка хранит информацию о всех символах ОТ. В качестве G -последовательности можно использовать и 2-ную и 3-ную последовательности. При этом в случае появления в G -последовательности символа '0' ОТ можно оставлять без изменения. Тогда в матрице замены можно хранить на одну строку меньше. С другой стороны подача на выход шифрующего устройства символа ОТ без искажений в случае накладывания на него символа '0' из гаммирующей последовательности значительно облегчает работу по расшифровке сообщения. Понятно, что в столбцах символы могут повторяться. Это никоим образом не отразится на однозначности расшифрованного текста. Более того, возможен вариант, при котором матрица замены будет такой, что какой-то определенный символ или группа символов могут вообще не возникнуть в ШТ. Достоверного знания ШТ и G-последовательности достаточно для абсолютно правильного восстановления ОТ. Точно таким образом можно попытаться сделать закон распределения символов в ШТ близким к равномерному, что существенно затруднит попытку криптоатаки аналитиком противника на перехваченное сообщение с помощью алгоритмов статистического анализа. Правда для этого необходимо знать статистические характеристики ОТ и уже на основании этих данных формировать матрицу замены. Практически предлагается иметь несколько вариантов матрицы замены, каждый из которых используется при кодировании информации, отображающей состояние конкретной предметной области. Так, например, в случае преобладания в ОТ цифровой информации используется одна матрица замены, если же в сообщении превалируют буквы русского алфавита – другая, и т.д.

Напомним, что криптографическая система - это совокупность алгоритмов криптографического преобразования и методов управления ключами. Ключ, в свою очередь, представляет собой ни что иное, как конкретное секретное состояние параметров алгоритма, определяющее один из множества возможных вариантов криптографического преобразования. Из построения алгоритма блочного

гаммирования видно, что показатели криптографической защищенности системы хуже, чем у классической гаммирующей криптосистемы, однако это вовсе не означает, что стоит полностью отказаться от блочного гаммирования. Известно, что целью криптографической защиты является повышение стоимости рабочего фактора. Рабочий фактор – усилия и средства, необходимые для раскрытия ключа. Раскрытие ключа равнозначно разрушению системы. Таким образом, если удастся сделать стоимость рабочего фактора выше защищаемой информации, то можно утверждать, что цель достигнута. Ведь в такой ситуации абсолютно нецелесообразно заниматься расшифровкой перехваченного сообщения, затрачивать материальные, временные, умственные ресурсы в попытке получить информацию, которая не стоит усилий, затраченных на ее получение.

Учитывая все сказанное выше, авторы утверждают, что шифрование различной информации с помощью блочного гаммирования является достаточно удобным средством сокрытия истинной информации от несанкционированного доступа. Авторами разработан прикладной программный продукт, который осуществляет блочное гаммирование с использованием G –последовательностей, символы которых принадлежат алфавитам с различными основаниями (в частности использовались двоичная и троичная G -последовательности). Кроме этого, приложение позволяет выбирать метод наложения гаммирующей последовательности на ОТ. Возможно использование одного из двух предлагаемых вариантов:

1. Вне зависимости от символа G -последовательности происходит замена символа ОТ на соответствующий символ из матрицы замены;

2. В случае появления в G -последовательности символа $0'$ символ ОТ, на который накладывается этот $'0'$, искажаясь попадает на выход шифрующего устройства.

Помимо всего этого, описываемый программный продукт способен выполнять функции декодирующего устройства, т. е. в случае верно заданных значений параметров криптологической системы, при которых осуществлялся процесс кодирования исходной информации, программа успешно и абсолютно корректно решает проблему расшифровки полученного ШТ. ОТ, G -последовательности, ШТ – вся эта информация сохраняется в файлах соответствующего формата. Все файлы далее могут использоваться по усмотрению лиц, в компетенцию которых входят хранение, изменение, удаление и другие операции с этими носителями информации. Программный продукт снабжен удобными и понятными даже не специалисту подсказками. Дружественный интерфейс позволяет людям слабо знакомым с современной вычислительной техникой быстро выработать необходимые навыки и успешно их применять в качестве пользователей предлагаемого продукта.

Литература

1. Патент RU 2082220 C1, Способ цифровой магнитной записи, авторы: Бардаченко В.Ф. и др., Бюл. №17 от 20.06.97 г.