Окончание табл. 3.1

Вариант	Сообщение	Ключевое слово
22	При обмене данными по сетям возникает проблема установления подлинности авторов	ЦВЕТОК
23	Получатель проверяет цифровую подпись, используя при этом открытый ключ	ВЕТЕР
24	В системах прозрачного шифрования преобразования осуществляются незаметно для пользователя	ПАПКА
25	Счастливые обстоятельства создают друзей, печальные – их испытывают	ЗАНЯТИЕ

Контрольные вопросы

1. Как формируется шифрующая таблица для реализации алгоритма Плейфейра?
2. Какие ограничения накладываются на шифруемый текст?
3. Что такое биграмма?
4. В чем заключается процедура шифрования с помощью алгоритма Плейфейра?

Отчетность по лабораторной работе

Выполните в рабочей тетради задания согласно своему варианту с подробным описанием хода решения.

ЛАБОРАТОРНАЯ РАБОТА 4
Шифрование с использованием системы Вижинера и шифра «двойной квадрат» Уитстона

Цель работы: формирование умений шифрования с использованием системы Вижинера и шифра «двойной квадрат» Уитстона.

Теоретические сведения

Шифры *сложной замены* называют многоалфавитными, так как для шифрования каждого символа исходного сообщения

применяют свой шифр простой замены. К таким шифрам относятся система Вижинера и «двойной квадрат» Уитстона.

Система Вижинера

Система Вижинера подобна такой системе шифрования Цезаря, у которой ключ подстановки меняется от буквы к букве. Этот шифр многоалфавитной замены описывается таблицей шифрования, называемой таблицей Вижинера (Приложение А).

Таблица Вижинера имеет два входа:

- верхнюю строку подчеркнутых символов, используемую для считывания очередной буквы исходного открытого текста;
- крайний левый столбец ключа.

Последовательность ключей получают из порядковых номеров в алфавите букв ключевого слова (начиная с 0).

При шифровании исходного сообщения его выписывают в строку, а под ним записывают ключевое слово или фразу. Если ключ оказался короче сообщения, то его циклически повторяют. В процессе шифрования находят в верхней строке таблицы очередную букву исходного текста и в левом столбце очередное значение ключа. Очередная буква шифртекста находится на пересечении столбца, определяемого шифруемой буквой, и строки, определяемой числовым значением ключа.

Рассмотрим пример шифрования сообщения «ПРИЛЕТАЮ ДЕСЯТОГО». Ключевое слово – «РАБОТА».

Ход шифрования и его результат отображены в табл. 4.1.

Таблица 4.1

Сообщение	п	р	и	л	е	т	а	ю		д	е	с	я	т	о	г	о
Ключ. слово	р	а	б	о	т	а	р	а		б	о	т	а	р	а	б	о
Ключи	16	0	1	14	18	0	16	0		1	14	18	0	16	0	1	14
Шифртекст	я	р	й	щ	ч	т	р	ю		е	у	г	я	в	о	д	ь

Шифр «двойной квадрат» Уитстона

Шифр «двойной квадрат» использует две таблицы со случайно расположенными в них буквами русского алфавита, размещенными по одной горизонтали; шифрование идет биграммами, как в шифре Плейфейра. Перед шифрованием исходное сообщение разбивают на биграммы. Каждая биграмма шифруется от-

дельно. Первую букву биграммы находят в левой таблице, а вторую букву в правой. Затем мысленно строят прямоугольник так, чтобы буквы биграммы лежали в его противоположных вершинах. Другие две вершины этого прямоугольника дают буквы биграммы шифртекста.

Пример шифрующих таблиц для данного метода приведен на рис. 4.1.

Ж	Щ	Н	Ю	Р
И	Т	Ь	Ц	Б
Я	М	Е	.	С
В	Ы	П	Ч	
:	Д	У	О	К
З	Э	Ф	Г	Ш
Ч	А	,	Л	Ъ

И	Ч	Г	Я	Т
,	Ж	Ь	М	О
З	Ю	Р	В	Щ
Ц	:	П	Е	Л
Ъ	А	Н	.	Х
Э	К	С	Ш	Д
Б	Ф	У	Ы	

Рис. 4.1

Предположим, что шифруется биграмма исходного текста ИЛ. Буква И находится в столбце 1 и строке 2 левой таблицы. Буква Л находится в столбце 5 и строке 4 правой таблицы. Это означает, что прямоугольник образован строками 2 и 4, а также столбцами 1 левой таблицы и 5 правой таблицы. Следовательно, в биграмму шифртекста входят буква О, расположенная в столбце 5 и строке 2 правой таблицы, и буква В, расположенная в столбце 1 и строке 4 левой таблицы, то есть получаем биграмму шифртекста ОВ.

Если обе буквы биграммы сообщения лежат в одной строке, то и буквы шифртекста берут из этой же строки. Первую букву биграммы шифртекста берут из правой таблицы в столбце, соответствующем первой букве биграммы сообщения. Вторая буква биграммы шифртекста берется из левой таблицы в столбце, соответствующем второй букве биграммы сообщения. Поэтому биграмма сообщения ТО превращается в биграмму шифртекста ЖБ.

Таким образом, в результате шифрования сообщения «ПРИЛЕТАЮ ШЕСТОГО» будет получен шифртекст «ПЕОВЩНФМЕШРФ ЖБДЦ».

Содержание заданий

Задание 1

Используя систему Вижинера, зашифруйте сообщения. Текст сообщения и ключевое слово должны соответствовать варианту задания лабораторной работы 3.

Задание 2

Используя шифр «двойной квадрат» Уитстона и шифрующие таблицы, представленные на рис. 4.1, выполните шифрование сообщения из задания лабораторной работы 3.

Контрольные вопросы

1. Чем шифры сложной замены отличаются от шифров простой замены?
2. Что используется в качестве ключа в системе Вижинера?
3. Как осуществляется шифрование текста с использованием системы Вижинера?
4. Какие требования предъявляются к шифруемому тексту при использовании шифра «двойной квадрат» Уитстона?
5. Как осуществляется шифрование текста с использованием шифра «двойной квадрат» Уитстона?

Отчетность по лабораторной работе

Выполните в рабочей тетради задания согласно своему варианту с подробным описанием хода решения.

ЛАБОРАТОРНАЯ РАБОТА 5
РЕАЛИЗАЦИЯ ЭЛЕМЕНТОВ КРИПТОСИСТЕМЫ RSA

Цель работы: формирование умений шифрования с использованием метода асимметрического шифрования RSA.

Теоретические сведения

RSA относится к так называемым *асимметричным алгоритмам*, у которых ключ шифрования не совпадает с ключом рас-

шифровки. Один из ключей доступен всем и называется *открытым ключом*, другой хранится только у хозяина и никому неизвестен. С помощью одного ключа можно производить операции только в одну сторону. Если сообщение зашифровано с помощью одного ключа, то расшифровать его можно только с помощью другого. Имея один из ключей практически невозможно найти другой ключ, если разрядность ключа высока.

Описание RSA

Алгоритм RSA состоит из следующих пунктов:

1. Выбрать простые числа p и q.
2. Вычислить $n = pq$.
3. Вычислить $m = (p - 1) \cdot (q - 1)$.
4. Выбрать число d взаимно простое с m.
5. Выбрать число e так, чтобы $ed = 1 \pmod m$.

Числа e и d являются ключами RSA. Шифруемые данные необходимо разбить на блоки – числа от 0 до $n - 1$. Шифрование и расшифровка данных производятся следующим образом:

- шифрование: $b = a^e \pmod n$;
- расшифрование: $a = b^d \pmod n$.

Следует также отметить, что ключи e и d равноправны, то есть сообщение можно шифровать как ключом e, так и ключом d, при этом расшифровка должна быть произведена с помощью другого ключа.

Нахождение простых чисел

В первом пункте алгоритма RSA сказано, что необходимо выбрать два простых числа p и q. Простой способ – деление предполагаемого простого числа на все числа меньшие его (не работоспособен уже с 32-битными числами, поскольку требуется большое количество времени на выполнение).

В данном случае, для выработки простых чисел используют вероятностные методы, но они не дают полной гарантии, что найденное число простое. Однако при достаточно небольшом количестве операций можно получить высокую вероятность, что найденное число является простым.

Алгоритм поиска простых чисел

1. N – нечетное число. Найти s и t, удовлетворяющие уравнению: $N - 1 = 2^s \cdot t$.

2. Случайным образом выбрать число a, $1 < a < N$.
3. Если N делится на a, перейти к пункту 6.
4. Если условие $at = 1 \pmod N$ выполняется, перейти к пункту 2.
5. Если найдется такое k, $0 \le k < s$, что $a^{2kt} = -1 \pmod N$, перейти к пункту 2.
6. Число N – составное: выбрать другое нечетное число N, перейти к пункту 1.

Если для какого-либо числа N проверено m чисел a, то математически доказанная вероятность того, что число является составным будет равняться $4 - m$ (но вероятность намного меньше этого значения). Исходя из этого, для числа N, состоящего из p бит, проверить p различных значений a. Если во время этого не обнаружится, что N – число составное, то вероятно, что оно является простым.

Нахождение взаимно простых чисел

На шаге 4 алгоритма RSA необходимо найти число d взаимно простое с m, то есть не имеющее общих делителей с ним, кроме единицы. Число d должно быть меньше m, таким образом, разрядность числа d равна сумме бит в числах p и q. Для нахождения взаимно простых чисел используется алгоритм Евклида, который находит наибольший общий делитель двух чисел. Если найденный делитель больше единицы, то необходимо выбрать другое число d и повторить проверку.

Алгоритм Евклида

1. Исходные числа a и b.
2. Вычислить r – остаток от деления a на b: $a = bq + r$.
3. Если $r = 0$, то b – искомое число (наибольший общий делитель), конец.
4. Если пункт 3 не выполняется, заменить пару чисел $<a, b>$ парой $<b, r>$, перейти к пункту 2.

При вычислении наибольшего общего делителя с помощью алгоритма Евклида будет выполнено не более $5p$ операций деления с остатком, где p есть количество цифр в десятичной записи меньшего из чисел a и b.

Решение уравнения $ax + by = 1$

В 5-м пункте алгоритма RSA предполагается нахождение такого числа e, чтобы $ed = 1 \pmod m$. Для этого нужно использо-

вать модифицированный алгоритм Евклида, который работает только если числа d и m взаимно просты. Вычисление числа e сводится к решению уравнения $mx + de = 1$ в натуральных числах. Число x не существенно.

Алгоритм решения уравнения $ax + by = 1$

1. Необходимо определить матрицу $E = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

2. Вычислить r – остаток от деления a на b: $a = bq + r$.

3. Если $r = 0$, то второй столбец матрицы дает решение: $\begin{bmatrix} x \\ y \end{bmatrix}$, конец.

4. Если пункт 3 не выполняется, то вычислить $E = E \begin{bmatrix} 0 & 1 \\ 1 & -q \end{bmatrix}$.

5. Заменить пару чисел $<a, b>$ парой $<b, r>$, перейти к пункту 2.

В данном алгоритме все вычисления можно производить по модулю большего из чисел a и b. Отрицательное число $–q$ заменяется положительным, полученным путем вычитания числа q из числа, взятого в качестве модуля. Например, если из чисел a и b большим является число b, то все вычисления можно производить по модулю числа b, при этом $–q$ будет представлено как $b – q$. Скорость работы алгоритма и количество производимых им операций примерно равняется соответствующим параметрам алгоритма Евклида, описанного выше.

Большие числа и работа с ними

На данный момент времени рекомендуется в качестве чисел e и d брать числа, длиной не менее 768 бит. Ключ в 1024 бит является достаточно надежным для обычных целей шифрования. Для повышенной безопасности рекомендуется брать ключи размером 2048 бит, то есть числа p и q должны иметь разрядность вдвое ниже чисел e, d, m и n (p и q рекомендуется брать примерно одного порядка, но не слишком близко друг к другу).

Содержание заданий

Разработайте программу, имитирующую реализацию элементов метода криптографической защиты информации RSA. Программа должна выполнять генерацию ключей, шифрование и расшифрование сообщения. В качестве сообщения используйте свою фамилию. Использовать n длиной в 4 разряда и более.

Контрольные вопросы

1. Что такое ассиметричное шифрование?
2. Отличие ассиметричного шифрования от блочного?
3. На сколько блоков (максимум) может разбиваться шифруемый текст?
4. Перечислите преимущества и недостатки алгоритма RSA?

Отчетность по лабораторной работе

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

ЛАБОРАТОРНАЯ РАБОТА 6
Реализация элементов схемы шифрования Эль-Гамаля

Цель работы: формирование умений шифрования с использованием метода асимметрического шифрования Эль-Гамаля.

Теоретические сведения

Генерация ключей

1. Генерируется случайное простое число p длины q битов.
2. Выбирается случайный примитивный элемент q поля Z_p.
3. Выбирается случайное целое число x такое, что $1 < x < p – 1$.
4. Вычисляется $y = g^x \bmod p$.
5. Открытым ключом является тройка чисел (p, g, y), закрытым ключом – x.

Шифрование

Сообщение M шифруется следующим образом:

1. Выбирается сессионный ключ – случайное целое число k, такое, что $1 < k < p – 1$.
2. Вычисляются числа $a = g^k \bmod p$ и $b = y^k M \bmod p$.
3. Пара чисел (a, b) является шифртекстом.

Длина шифротекста в схеме Эль-Гамаля вдвое длиннее исходного сообщения M.

Расшифрование

Зная закрытый ключ x, исходное сообщение можно вычислить из шифртекста (a, b) по формуле: $M = b(a^x)^{-1} \bmod p$. При этом нетрудно проверить, что $(a^x)^{-1} \equiv g^{-kx} \pmod p$, и поэтому $b(a^x)^{-1} \equiv (y^x N)g^{-xk} \equiv (g^{xk}M)g^{-xk} \equiv M \pmod p$.

Для практических вычислений больше подходит следующая формула: $M = b(a^x)^{-1} \bmod p = ba^{(p-1-x)} \bmod p$ (рис. 6.1).

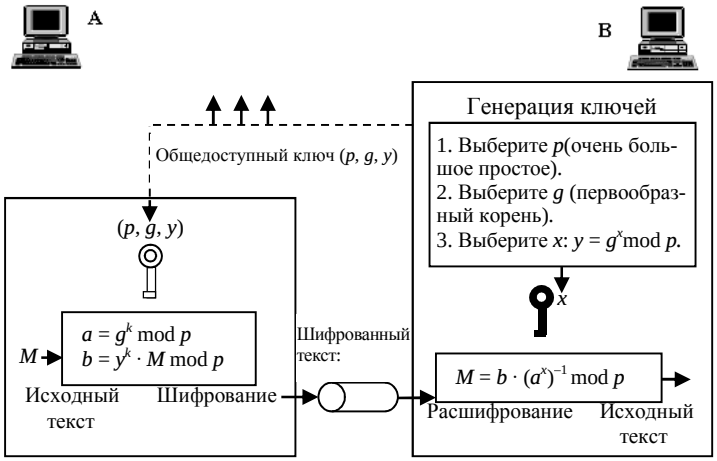

Рис. 6.1

Содержание заданий

Разработайте программу, имитирующую реализацию элементов метода криптографической защиты информации Эль-Гамаля. Программа должна выполнять генерацию ключей, шифрование и расшифрование сообщения. В качестве сообщения используйте свою фамилию.

Примечание. P – двузначное число, G, X – однозначные.

Контрольные вопросы

1. Что такое криптосистемы с открытым ключом?
2. Отличие схемы Эль-Гамаля от RSA?
3. Перечислите преимущества и недостатки алгоритма Эль-Гамаля?

Отчетность по лабораторной работе

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

ЛАБОРАТОРНАЯ РАБОТА 7
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: $X = (X_0, X_1, \ldots, X_{n-1}) = X_0 \parallel X_1 \parallel \ldots \parallel X_{n-1}$. 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:

$$\left(X_0, X_1, ..., X_{n-1}\right) = X_0 \parallel X_1 \parallel ... \parallel X_{n-1} = X_0 + 2^{|X_0|} \times$$
$$\times \left(X_1 + 2^{|X_1|}\left(...\left(X_{n-2} + 2^{|X_{n-2}|} X_{n-1}\right)...\right)\right).$$

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:

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

Таким образом, для ГОСТа принята так называемая «little-endian» 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 в данной статье 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: $K = \{K_j\}$, $0 \le j \le 7$. В ГОСТе элементы ключа используются как 32-разрядные целые числа без знака: $0 \le K_j < 2^{32}$. Таким образом, размер ключа составляет $32 \cdot 8 = 256$ 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. 7.1.

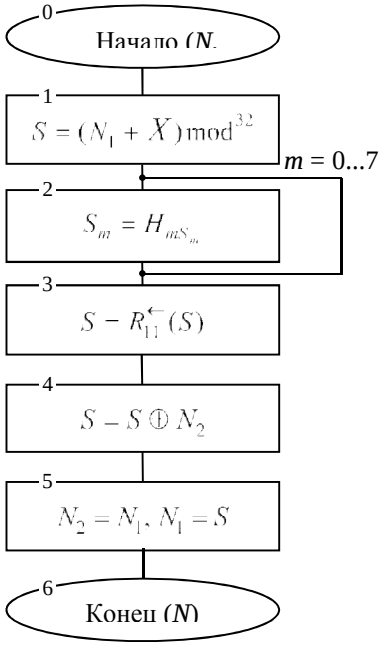

Рис. 7.1

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 S_i меняется на S_i-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-З:

$K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0$ (рис. 7.2).

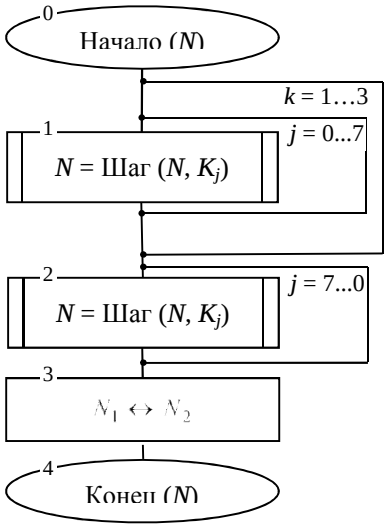

Рис. 7.2

Цикл расшифрования 32-Р:

$K_0, K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0, K_7, K_6, K_5, K_4, K_3, K_2, K_1, K_0$ (рис. 7.3).

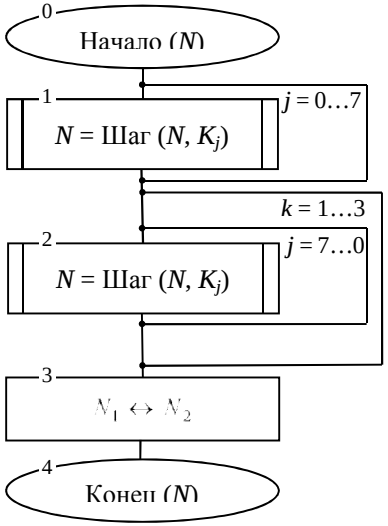

Рис. 7.3

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. 7.4).

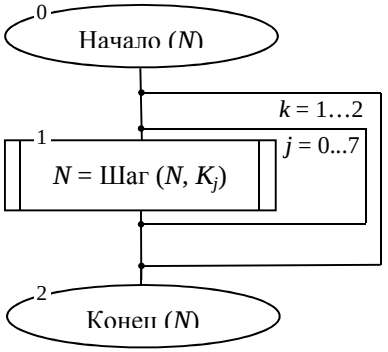

Рис. 7.4

Каждый из циклов имеет собственное буквенно-цифровое обозначение, соответствующее шаблону «*n-X*», где первый эле-

мент 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 расшифрования, 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. 7.2–7.4. 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 вычисления 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 и один 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.

ЛАБОРАТОРНАЯ РАБОТА 8
Протокол идентификации с нулевой передачей данных

Цель работы: формирование умений проверки подлинности удаленных пользователей с помощью протокола идентификации с нулевой передачей данных.

Теоретические сведения

Широкое распространение интеллектуальных карт (смарт-карт) для разнообразных коммерческих, гражданских и военных применений (кредитные карты, карты социального страхования, карты доступа в охраняемое помещение, компьютерные пароли и ключи, и т. п.) потребовало обеспечения безопасной идентификации таких карт и их владельцев. Во многих приложениях главная проблема заключается в том, чтобы при предъявлении интеллектуальной карты оперативно обнаружить обман и отказать обманщику в допуске, ответе или обслуживании.

Для безопасного использования интеллектуальных карт разработаны протоколы идентификации с нулевой передачей знаний. Секретный ключ владельца карты становится неотъемлемым признаком его личности. Доказательство знания этого секретного ключа с нулевой передачей этого знания служит доказательством подлинности личности владельца карты.

Схему идентификации с нулевой передачей знаний предложили в 1986 г. У. Фейге, А. Фиат и А. Шамир. Она является наиболее известным доказательством идентичности с нулевой передачей конфиденциальной информации.

Прежде всего в схеме идентификации с нулевой передачей знаний выбирают случайное значение модуля n, который является произведением двух больших простых чисел. Модуль n должен иметь длину 512...1024 бит. Это значение n может быть представлено группе пользователей, которым придется доказывать свою подлинность. В процессе идентификации участвуют две стороны:

- сторона A, доказывающая свою подлинность;
- сторона B, проверяющая представляемое стороной A доказательство.

Для того чтобы сгенерировать открытый и секретный ключи для стороны A, доверенный арбитр (Центр) выбирает некоторое число V, которое является квадратичным, вычетом по модулю n. Иначе говоря, выбирается такое число V, что сравнение $x^2 = V(\mathrm{mod}\ n)$ имеет решение, и существует целое число $V^{-1} \mathrm{mod}\ n$.

Выбранное значение V является открытым ключом для A. Затем вычисляют наименьшее значение S, для которого

$$S = \sqrt{(V^{-1})}(\mathrm{mod}\ n).$$

Это значение S является секретным ключом для A.

Теперь можно приступить к выполнению протокола идентификации.

1. Сторона A выбирает некоторое случайное число r, $r < n$. Затем она вычисляет $x = r^2 \mathrm{mod}\ n$ и отправляет x стороне B.

2. Сторона B посылает A случайный бит b.

3. Если $b = 0$, тогда A отправляет r стороне B. Если $b = 1$, то A отправляет стороне B: $y = rS \mathrm{mod}\ n$

4. Если $b = 0$, сторона B проверяет, что $x = r^2 \mathrm{mod}\ n$ чтобы убедиться, что A знает $\sqrt{(x)}$. Если $b = 1$, сторона B проверяет, что $x = y^2 \cdot V \mathrm{mod}\ n$, чтобы быть уверенной, что A знает $\sqrt{(V-1)}$.

Эти шаги образуют один цикл протокола, называемый *аккредитацией*. Стороны A и B повторяют этот цикл t раз при разных случайных значениях r и b до тех пор, пока B не убедится, что A знает значение S.

Если сторона A не знает значения S, она может выбрать такое значение r, которое позволит ей обмануть сторону B, если B отправит ей $b = 0$, либо A может выбрать такое r, которое позволит обмануть B, если B отправит ей $b = 1$. Но этого невозможно сделать в обоих случаях. Вероятность того, что A обманет B в одном цикле, составляет 1/2. Вероятность обмануть B в t циклах равна $(1/2)^t$.

Для того чтобы этот протокол работал, сторона A никогда не должна повторно использовать значение r. Если A поступила бы таким образом, а сторона B отправила бы стороне A на шаге 2 другой случайный бит b, то B имела бы оба ответа A. После этого B может вычислить значение S, и для A все закончено.

Содержание заданий

Разработайте программу, имитирующую функционирование упрощенной схемы идентификации с нулевой передачей данных.

Значения необходимых параметров должны выбираться случайным образом.

Выходная информация должна быть следующей:

Значение n;

Открытый ключ;

Секретный ключ;

Процесс идентификации
Сторона A;
Сторона B;
Сторона A;
Сторона B;
…
Примечания
1. Модуль n определяется как произведение двузначных чисел p и q.
2. Сравнение $x^2 = V(\mathrm{mod}\ n)$ равнозначно выражению $x^2 \bmod n = V$.
3. Сравнение $z = V^{-1}(\mathrm{mod}\ n)$ равнозначно выражению $z \cdot V \bmod n = 1$.

Контрольные вопросы

1. Что такое идентификация пользователей и для чего она нужна?
2. Какая рекомендуемая длина модуля?
3. Сколько итераций алгоритма нужно провести для уверенности в подлинности пользователя?

Отчетность по лабораторной работе

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

ЛАБОРАТОРНАЯ РАБОТА 9
Параллельная схема протокола интенсификации с нулевой передачей данных

Цель работы: формирование умений проверки подлинности удаленных пользователей с помощью параллельной схемы протокола идентификации с нулевой передачей данных.

Теоретические сведения

Параллельная схема идентификации позволяет увеличить число аккредитаций, выполняемых за один цикл, и тем самым уменьшить длительность процесса идентификации.

Сначала генерируется число n как произведение двух больших чисел. Для того, чтобы сгенерировать открытый и секретный ключи для стороны A, сначала выбирают K различных чисел $V_1, V_2, ..., V_K$, где каждое V_i является квадратичным вычетом по модулю n. Иначе говоря, выбирают значение V_i таким, что сравнение $x^2 = V_i \bmod n$ имеет решение, и существует число $V_i^{-1} \bmod n$. Полученная строка $V_1, V_2, ..., V_K$ является открытым ключом.

Затем вычисляют такие наименьшие значения S_i, что

$$S_i = \sqrt{(V_i^{-1})}(\mathrm{mod}\ n).$$

Эта строка $S_1, S_2, ..., S_K$ является секретным ключом стороны A.

Протокол процесса идентификации имеет следующий вид:

1. Сторона A выбирает некоторое случайное число r, $r < n$. Затем она вычисляет $x = r^2 \bmod n$ и посылает x стороне B.

2. Сторона B отправляет стороне А некоторую случайную двоичную строку из K бит: $b_1, b_2, ..., b_K$.

3. Сторона A вычисляет

$$y = r(S_1^{b_1} \cdot S_2^{b_2} \cdot ... \cdot S_K^{b_K}) \bmod n.$$

Перемножаются только те значения S_i, для которых $b_1 = 1$. Например, если $b_1 = 1$, то сомножитель S входит в произведение, если же $b_1 = 0$, то S_1 не входит в произведение, и т. д. Вычисленное значение y отправляется стороне B.

4. Сторона B проверяет, что

$$x = y(V_1^{b_1} \cdot V_2^{b_2} \cdot ... \cdot V_K^{b_K}) \bmod n.$$

Фактически сторона B перемножает только те значения V_i, для которых $b_i = 1$. Стороны A и B повторяют этот протокол t раз, пока B не убедится, что A знает $S_1, S_2, ..., S_K$.

Вероятность того, что A может обмануть B, равна $(1/2)^{Kt}$. Рекомендуется в качестве контрольного значения брать вероятность обмана B равной $(1/2)^{20}$ при $K = 5$ и $t = 4$.

Содержание заданий

Разработайте программу, имитирующую функционирование параллельной схемы идентификации с нулевой передачей данных.

Значения необходимых параметров должны выбираться случайным образом.

Выходная информация должна быть следующей:
Значение n;
Открытые ключи;
Секретные ключи;

Процесс идентификации
Сторона А;
Сторона В;
Сторона А;
Сторона В.

Контрольные вопросы

1. Что такое идентификация пользователей и для чего она нужна?
2. В чем отличие протокола идентификации с нулевой передачей знания и параллельной схемы протокола идентификации с нулевой передачей знания?
3. Какая вероятность того, что одному из пользователей удастся обмануть другого?

Отчетность по лабораторной работе

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

ЛАБОРАТОРНАЯ РАБОТА 10
РЕАЛИЗАЦИЯ ЭЛЕМЕНТОВ ЭЦП RSA

Цель работы: формирование умений подписи электронных документов электронной цифровой подписью с помощью алгоритма RSA.

Теоретические сведения

Электронная цифровая подпись (ЭЦП) – реквизит электронного документа, предназначенный для удостоверения источника данных и защиты данного электронного документа от подделки.

Общая схема
Схема электронной подписи обычно включает в себя:
- алгоритм генерации ключевых пар пользователя;
- функцию вычисления подписи;
- функцию проверки подписи.

Функция вычисления подписи на основе документа и секретного ключа пользователя вычисляет собственно подпись. В зависимости от алгоритма функция вычисления подписи может быть детерминированной или вероятностной. Детерминированные функции всегда вычисляют одинаковую подпись по одинаковым входным данным. Вероятностные функции вносят в подпись элемент случайности, что усиливает криптостойкость алгоритмов ЭЦП. Однако для вероятностных схем необходим надежный источник случайности (либо аппаратный генератор шума, либо криптографически надежный генератор псевдослучайных бит), что усложняет реализацию.

В настоящее время детерминированые схемы практически не используются.

Функция проверки подписи проверяет, соответствует ли данная подпись данному документу и открытому ключу пользователя. Открытый ключ пользователя доступен всем, так что любой может проверить подпись под данным документом.

Поскольку подписываемые документы переменной (и достаточно большой) длины, в схемах ЭЦП зачастую подпись ставится не на сам документ, а на его хэш. Для вычисления хэша используются криптографические хэш-функции, что гарантирует выявление изменений документа при проверке подписи. Хэш-функции не являются частью алгоритма ЭЦП, поэтому в схеме может быть использована любая надежная хэш-функция.

Защищенность
Цифровая подпись обеспечивает:
- удостоверение источника документа. В зависимости от деталей определения документа могут быть подписаны такие поля, как «автор», «внесенные изменения», «метка времени» и т. д.;
- защиту документа от изменений. При любом случайном или преднамеренном изменении документа (или подписи) изменится хэш, следовательно, подпись станет недействительной;
- невозможность отказа от авторства, так как создать корректную подпись можно лишь, зная закрытый ключ, а он известен только владельцу, то владелец не может отказаться от своей подписи под документом.

Возможны следующие угрозы цифровой подписи:
- злоумышленник может попытаться подделать подпись для выбранного им документа;
- злоумышленник может попытаться подобрать документ к данной подписи, чтобы подпись к нему подходила.

При использовании надежной хэш-функции, вычислительно сложно создать поддельный документ с таким же хэшем, как у подлинного. Однако эти угрозы могут реализоваться из-за слабостей конкретных алгоритмов хэширования, подписи или ошибок в их реализациях.

Тем не менее, возможны еще *следующие угрозы системам цифровой подписи*:

- злоумышленник, укравший закрытый ключ, может подписать любой документ от имени владельца ключа;
- злоумышленник может обманом заставить владельца подписать какой-либо документ, например, используя протокол слепой подписи;
- злоумышленник может подменить открытый ключ владельца на свой собственный, выдавая себя за него.

Реализацию алгоритма смотрите в лабораторной работе № 5, разница лишь в том, что открытый и секретный ключи меняются местами.

Содержание заданий

Реализуйте программу, которая будет подписывать документ MS Word, посмотрите, как будет изменяться подпись при незначительных и больших изменениях в исходном тексте.

Контрольные вопросы

1. Что такое хэш-функция?
2. Для чего применяется хэш-функция?
3. Особенности применения ЭЦП?

Отчетность по лабораторной работе

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

ЛАБОРАТОРНАЯ РАБОТА 11
РЕАЛИЗАЦИЯ ЭЛЕМЕНТОВ ЭЦП ГОСТ Р 34.10–94

Цель работы: формирование умений подписи электронных документов электронной цифровой подписью с помощью алгоритма ГОСТ Р 34.10–94.

Теоретические сведения

Первый российский стандарт цифровой подписи обозначается как ГОСТ Р 34.10–94. Алгоритм цифровой подписи, определяемый этим стандартом, концептуально близок к алгоритму DSA. В нем используются следующие параметры:

- p – большое простое число длиной от 509 до 512 бит либо от 1020 до1024 бит;
- q – простой сомножитель числа $(p - 1)$, имеющий длину 254–256 бит;
- a – любое число, меньшее $(p - 1)$, причем такое, что $a^q \bmod p = 1$;
- x – некоторое число, меньшее q;
- $y = a^x \bmod p$.

Кроме того, этот алгоритм использует однонаправленную хэш-функцию $H(x)$.

Стандарт ГОСТ Р 34.10–94 определяет хэш-функцию, основанную на использовании стандартного симметричного алгоритма ГОСТ 28147–89.

Первые три параметра – p, q и a – являются открытыми и могут быть общими для всех пользователей сети. Число x является секретным ключом. Число y является открытым ключом.

Чтобы подписать некоторое сообщение m, а затем проверить подпись, выполняются следующие шаги:

1. Пользователь A генерирует случайное число k, причем $k < q$.
2. Пользователь A вычисляет значения:

$r = (a^k \bmod p) \bmod q$,

$s = (xr + k(H(m))) \bmod q$.

Если $H(m) \bmod q = 0$, то значение $H(m) \bmod q$ принимают равным единице.

Если $r = 0$, то выбирают другое значение k и начинают снова.

Цифровая подпись представляет собой два числа: r и s.

Пользователь A отправляет эти числа пользователю B.

3. Пользователь B проверяет полученную подпись, вычисляя:

$v = H(m)^{q-2} \bmod q$;

$1z = (sv) \bmod q$;

$2z = ((q - r)v) \bmod q$;

$u = ((a^{z1} y^{z2}) \bmod p) \bmod q$.

Если $u = r$, то подпись считается верной.

Различие между этим алгоритмом и алгоритмом DSA заключается в том, что в DSA $s = (k(xr + (H(m))))\text{mod } q$, что приводит к другому уравнению верификации.

Следует также отметить, что в российском стандарте ЭЦП параметр q имеет длину 256 бит. Современных криптографов вполне устраивает q длиной примерно 160 бит. Различие в значениях параметра q является отражением стремления разработчиков российского стандарта к получению более безопасной подписи. Этот стандарт вступил в действие с начала 1995 г.

Содержание заданий

Реализуйте программу, которая будет подписывать документ MS Word, посмотрите, как будет изменяться подпись при незначительных и больших изменениях в исходном тексте.

Контрольные вопросы

1. Что такое хэш-функция?
2. Для чего применяется хэш-функция?
3. Особенности применения ЭЦП?

Отчетность по лабораторной работе

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

ЛАБОРАТОРНАЯ РАБОТА 12
СОЗДАНИЕ ВИРТУАЛЬНЫХ ЗАШИФРОВАННЫХ ДИСКОВ (ПРОГРАММНОЕ СРЕДСТВО TRUECRYPT)

Цель работы: формирование умений шифрования жесткого диска или его частей и работы с виртуальными образами зашифрованной области.

Теоретические сведения

TrueCrypt – это программная система для создания и использования шифруемого «на лету» тома (устройства хранения

данных). *Шифрование «на лету»* означает, что данные автоматически шифруются или расшифровываются прямо во время их считывания или записи без участия пользователя. Данные, сохраненные на зашифрованном томе, невозможно прочесть (расшифровать) без использования правильных пароля/ключевого файла или без правильных ключей шифрования. Шифруется вся файловая система, включая имена папок и файлов, содержимое файлов, пустое пространство, метаданные и т. п.).

Файлы могут копироваться с/на подключенный том TrueCrypt также, как они копируются с/на любой диск (к примеру, с помощью технологии перетаскивания – drag-n-drop). Файлы автоматически расшифровываются «на лету» (в память) во время чтения или копирования с зашифрованного тома TrueCrypt. Верно и обратное – файлы, записываемые или копируемые на том TrueCrypt, шифруются «на лету» в памяти прямо перед записью на диск. Однако это не значит, что весь файл, предназначенный для шифрования/расшифрования, должен целиком попасть в память перед шифрованием/расшифрованием. Для TrueCrypt не требуется дополнительная память.

Например, есть файл .avi, хранящийся на томе TrueCrypt и поэтому целиком зашифрованный. Пользователь, используя правильный пароль (и/или ключевой файл), подключает (открывает) том TrueCrypt. Когда пользователь дважды кликает по иконке видеофайла, операционная система запускает программу, связанную с этим типом файла, как правило, – медиаплеер. Медиаплеер начинает загружать в память небольшую начальную часть видеофайла с зашифрованного TrueCrypt-тома, чтобы проиграть его. В процессе загрузки этой небольшой части TrueCrypt автоматически расшифровывает ее в памяти. Расшифрованная часть видео, хранящаяся теперь в памяти, проигрывается медиаплеером. После проигрывания этой части медиаплеер начнет загружать следующую небольшую часть видеофайла с зашифрованного тома TrueCrypt в память, и процесс повторится. Этот процесс и называется шифрованием/расшифрованием «на лету», и он работает со всеми типами файлов, не только с видео.

TrueCrypt никогда не сохраняет никакие данные в незашифрованном виде на диск, он их временно хранит в памяти. Даже когда том подключен, данные на нем хранятся в зашифрованном виде. Когда вы перезапускаете Windows или выключаете ком-

пьютер, том отключается, и файлы, хранимые на нем, становятся недоступными, оставаясь зашифрованными. То же самое происходит в случае непредвиденного отключения электроэнергии (без правильного завершения работы системы). Чтобы получить к ним доступ снова, вы должны подключить том, используя правильные пароль и/или ключевой файл.

Шифрование и использование несистемного раздела диска

Шаг 1

Запустите TrueCrypt, дважды щелкнув по значку TrueCrypt.exe, или вызовите его из меню **Пуск**.

Шаг 2

На экране появится главное окно TrueCrypt (рис. 12.1). Нажмите **Создать том**.

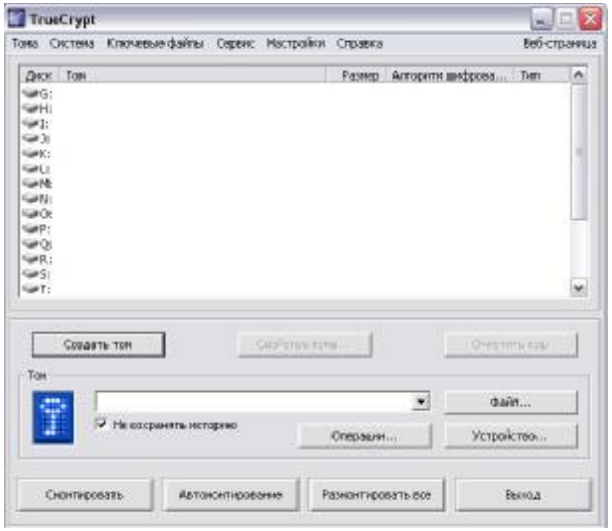

Рис. 12.1

Шаг 3

На экране должно появиться окно Мастера создания раздела TrueCrypt (рис. 12.2).

Выберите место, где нужно создать раздел TrueCrypt. Раздел TrueCrypt может быть размещен в файле, который также называют контейнером, в разделе диска или на дисковом устройстве. Зашифруйте запоминающее USB-устройство, выбрав вторую опцию **Зашифровать несистемный раздел/диск** и кликнув кнопку **Далее**.

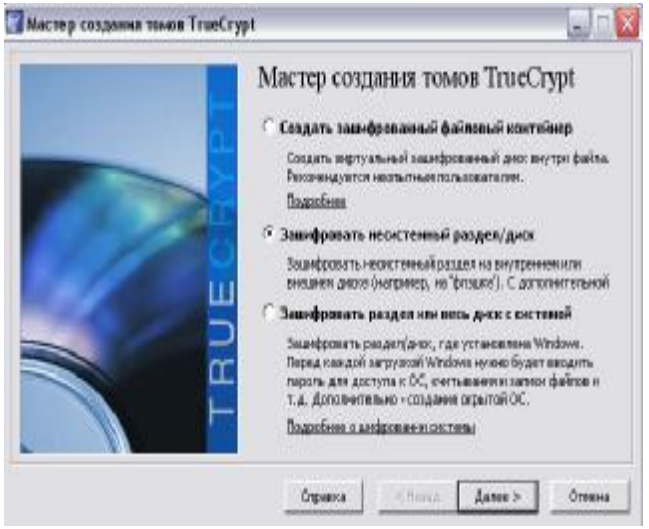

Рис. 12.2

Шаг 4

На данном этапе предстоит выбрать, каким будет наш тип тома (рис. 12.3).

Рис. 12.3

Возможны два варианта:

- обычный том TrueCrypt;
- скрытый том TrueCrypt.

Выберите опцию **Обычный том TrueCrypt**.

П р и м е ч а н и е. Опцию **Скрытый том TrueCrypt** рассмотрим в шаге 12.

Шаг 5

Выберите устройство, которое будет зашифровано (рис. 12.4).

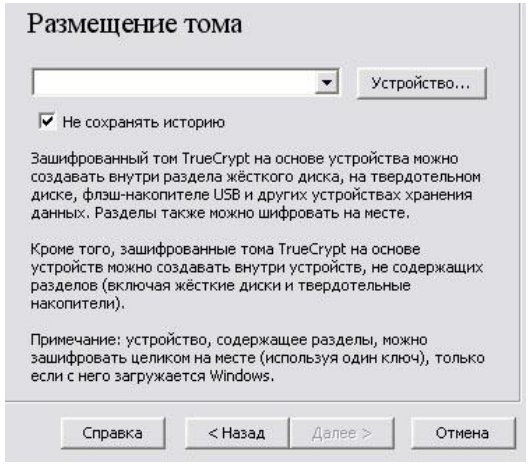

Рис. 12.4

В данном случае выбран диск **F:** (рис. 12.5). Затем перейдите к следующему шагу, нажав кнопку **Далее**.

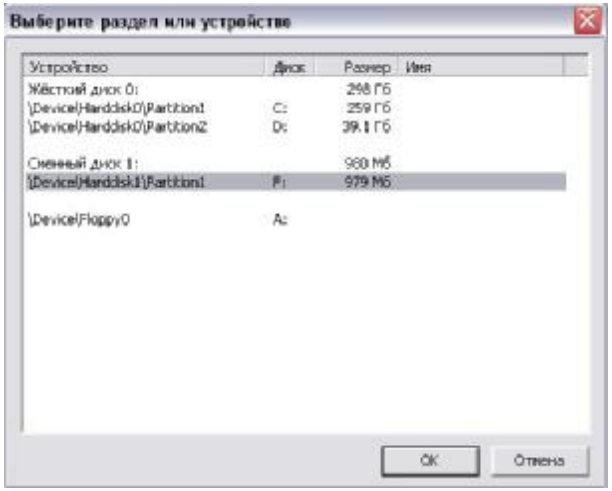

Рис. 12.5

Шаг 6

На данном этапе выберите одну из двух опций (рис. 12.6):

- создать зашифрованный том и отформатировать его;
- зашифровать раздел на его месте.

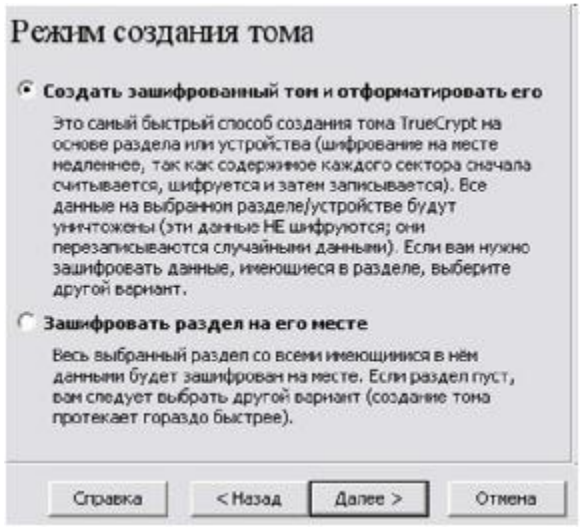

Рис. 12.6

Если на устройстве, выбранном в шаге 5, имеется важная информация, которую вы хотите сохранить и зашифровать, то следует выбрать пункт **Зашифровать раздел на его месте**. Если на выбранном устройстве нет информации либо она отсутствует, выберите пункт **Создать зашифрованный том и отформатировать его**. В этом случае все данные, хранимые на выбранном устройстве, будут утеряны. После того, как вы выбрали опцию, кликните кнопку **Далее**.

Шаг 7

Здесь вы можете выбрать алгоритм шифрования и хэш-алгоритм для раздела. Если вы не уверены в том, что выбрать, то можете использовать настройки по умолчанию и нажать **Далее**.

Шаг 8

На восьмом шаге убедитесь в том, что вы выбрали именно то устройство, которое хотели, и нажмите кнопку **Далее**.

Шаг 9

Это один из наиболее важных шагов. Здесь вы должны выбрать хороший пароль для тома. Внимательно прочитайте ин-

формацию, отображаемую в окне Мастера (рис. 12.7). После выбора, введите его в первое поле для ввода. Затем повторите его во втором поле и нажмите **Далее**.

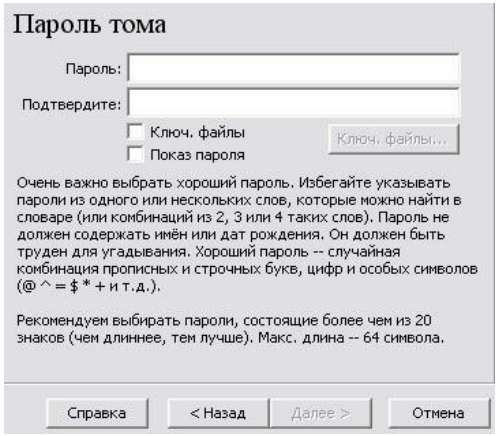

Рис. 12.7

П р и м е ч а н и е. Кнопка **Далее** будет неактивна пока пароли в обоих полях не будут совпадать.

Шаг 10

В окне Volume Creation Wizard как минимум 30 секунд случайно двигайте мышкой в разных направлениях, чем дольше вы будете ею двигать, тем значительно увеличите криптографическую силу ключей шифрования (которые, в свою очередь, усилят безопасность). Нажмите **Разметить** (рис. 12.8).

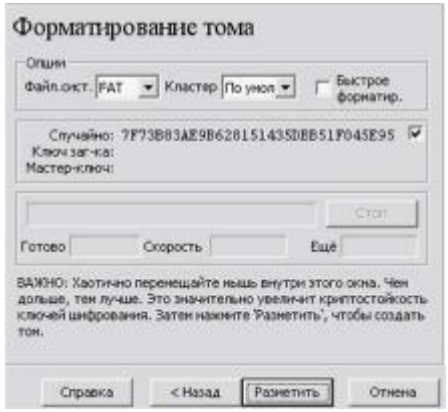

Рис. 12.8

Согласитесь с форматированием выбранного раздела и ждите окончания процесса.

Внимательно прочитайте и нажмите **OK**.

Итак, том создан (рис. 12.9).

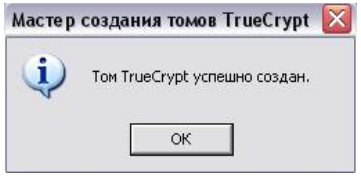

Рис. 12.9

После того, как вы кликните **OK**, TrueCrypt предложит вам создать еще один том (кнопка **Далее**). Если вы хотите прекратить работу Мастера создания томов, нажмите кнопку **Выход**.

Шаг 11

Теперь нужно научиться им пользоваться. При обычной попытке доступа к зашифрованному устройству (**Мой компьютер** – **Зашифрованное устройство**), Windows предложит его отформатировать (после процесса форматирования устройство окажется пустым и незашифрованным). Для доступа к созданному тому откройте TrueCrypt и на главной форме выберите устройство, которое шифровали (на нашем примере это диск **F:** (рис. 12.10)).

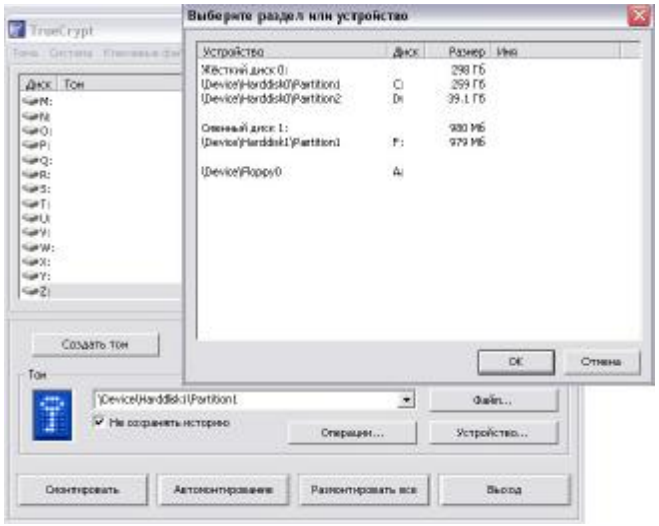

Рис. 12.10

Далее выберите незанятую букву диска (например, **Z:**) и нажмите кнопку **Смонтировать**.

Введите пароль, который был придуман в шаге 9 и нажмите **ОК** (рис. 12.11).

Рис. 12.11

Если пароль неверный (например, вы его неправильно ввели), то TrueCrypt сообщит об этом; введите пароль заново и нажмите **ОК**. Если пароль правильный, то раздел будет смонтирован.

Зашифрованное устройство успешно смонтировано как виртуальный диск **Z:** (рис. 12.12).

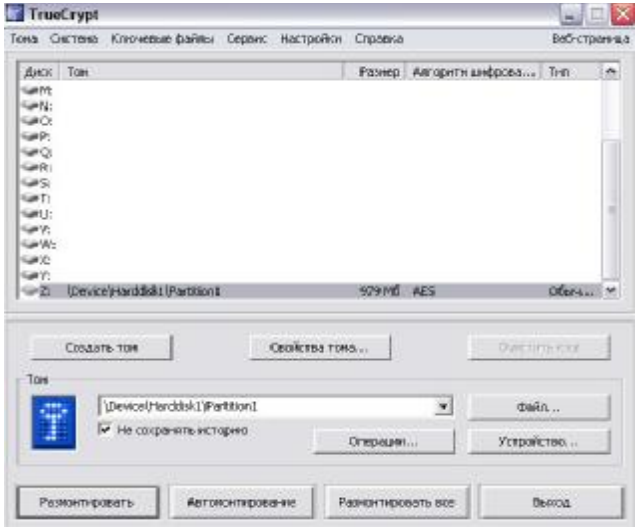

Рис. 12.12

Виртуальный диск полностью зашифрован (включая имена файлов, таблицы размещения, свободное место и т. д.). Вы можете сохранять (копировать, перемещать и т. д.) файлы на этот виртуальный диск, и они будут зашифрованы «на лету» во время записи.

Если откроете файл, хранящийся на разделе TrueCrypt, например, в медиаплеере, то он будет автоматически расшифрован «на лету» в ОЗУ (память) во время чтения.

П р и м е ч а н и е. Когда вы открываете файл, хранящийся на разделе TrueCrypt (или когда записываете/копируете файл с/на TrueCrypt раздел), запроса ввести пароль не будет. Ввести пароль потребуется только тогда, когда монтируете раздел.

Вы можете открыть смонтированный раздел, например, дважды кликнув на помеченной прямоугольником надписи, как ниже показано на снимке экрана (рис. 12.13).

Вы также можете найти и открыть смонтированный раздел, открыв **Мой компьютер** и дважды щелкнув кнопкой мыши по значку диска (в нашем случае это диск **Z:** (рис. 12.13)).

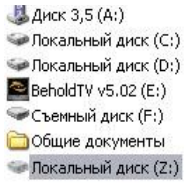

Рис. 12.13

Выберите раздел из списка смонтированных разделов в главном окне TrueCrypt и затем нажмите **Размонтировать** (рис. 12.14).

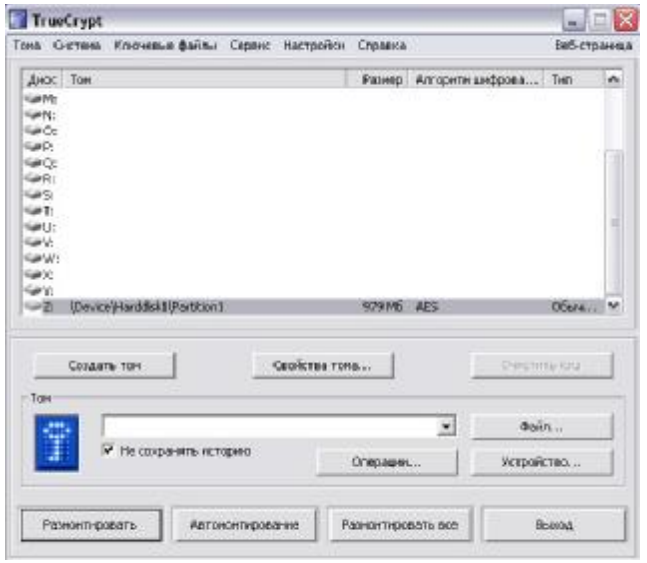

Рис. 12.14

Расшифровка устройства

Если потребуется обратно расшифровать устройство, выполните следующие действия:

- проверьте, что зашифрованное устройство размонтировано;
- зайдите в **Мой компьютер**;
- найдите в списке дисков зашифрованный (тот, который хотите расшифровать);
- кликните правой кнопкой по выбранному диску;
- выберите из списка контекстного меню пункт **Форматировать**.

По окончании форматирования диск будет расшифрован и доступен так же, как и раньше.

Шаг 12

Для создания скрытого тома повторите шаги один–три и на четвертом выберите пункт **Скрытый том TrueCrypt** (рис. 12.15).

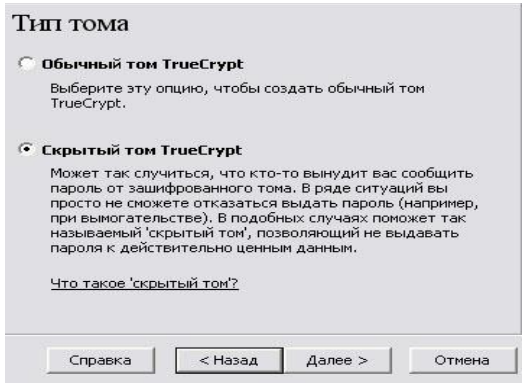

Рис. 12.15

Что такое «скрытый том» можно узнать, кликнув по соответствующей ссылке. Представим общие сведения:

- скрытый том создается внутри уже существующего обычного тома;
- доступ к скрытому тому осуществляется так же, как и к обычному, только при монтировке обычного тома нужно ввести пароль скрытого;
- о существовании скрытого тома невозможно узнать, что дает дополнительную защиту информации, хранимой на этом томе.

Шаг 13

Мастер создания томов предлагает выбрать режим создания скрытого тома (рис. 12.16). Возможны два варианта:

- обычный режим (создание обычного тома, после чего появится возможность создания внутри него скрытого тома);
- прямой режим (создание скрытого тома на уже существующем обычном томе).
- Выберите **Прямой режим**, так как обычный том уже был создан.

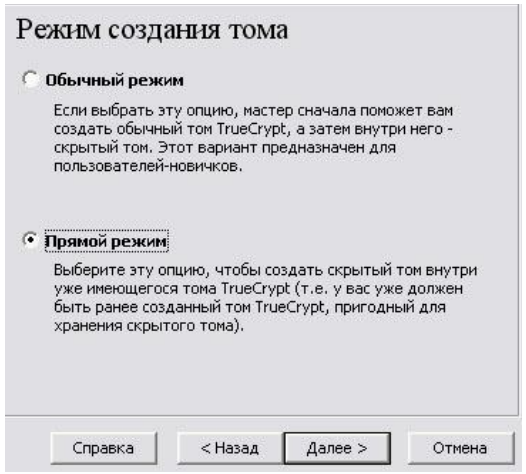

Рис. 12.16

Шаг 14

Выберите устройство (диск) или каталог, в котором нужно создать скрытый том (рис. 12.17).

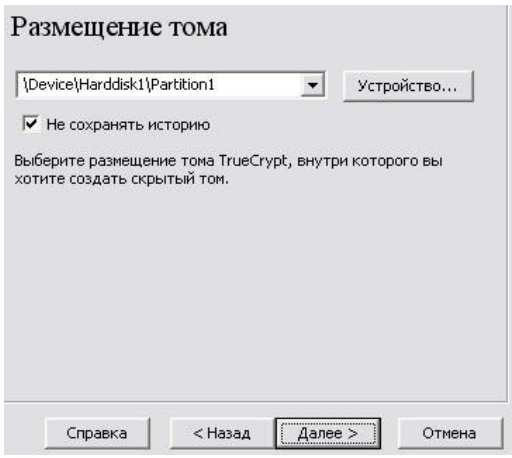

Рис. 12.17

Так же можно выбрать пункт сохранения истории происходящих действий, но не рекомендуется это делать, чтобы злоумышленник не добрался до скрытой информации данного файла.

Шаг 15

На этом шаге введите пароль, который применяется в обычном томе (этот пароль был использован в шаге 9 (рис. 12.18)). После ввода пароля кликните **Далее**.

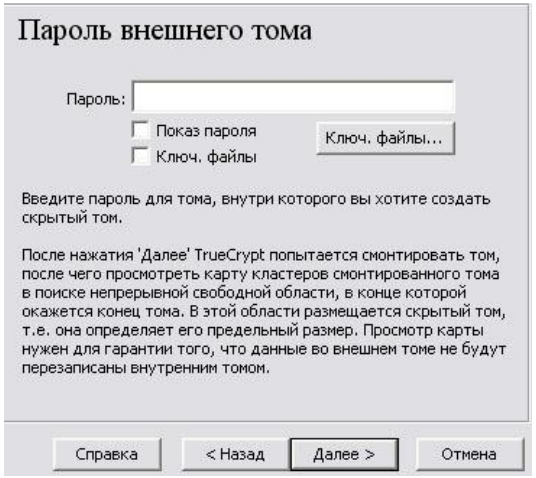

Рис. 12.18

Шаг 16

Внимательно прочитайте уведомление (рис. 12.19).

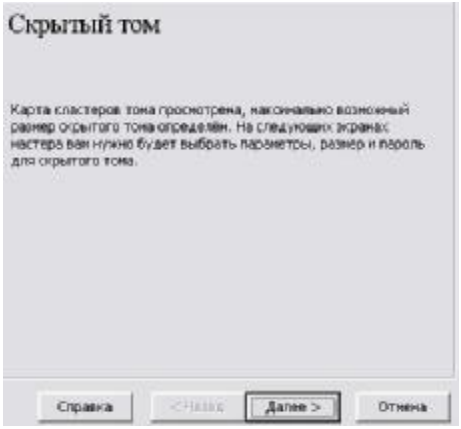

Рис. 12.19

Шаг 17

Данный шаг аналогичен шагу 7.

Шаг 18

На этом этапе выберите размер скрытого тома (рис. 12.20).

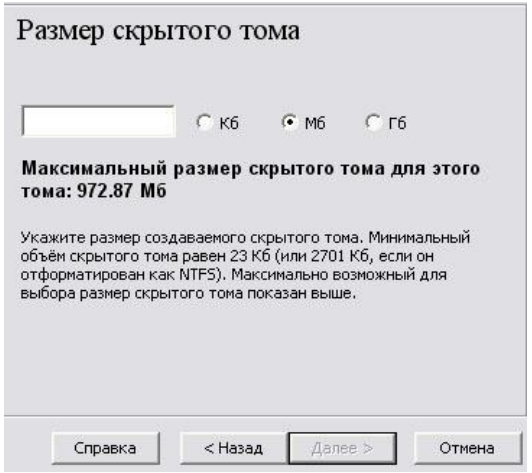

Рис. 12.20

П р и м е ч а н и е. Обратите внимание на максимально возможный размер.

Шаг 19

Этот шаг аналогичен шагу 9, но *пароль для скрытого тома должен отличаться от пароля внешнего* (рис. 12.21).

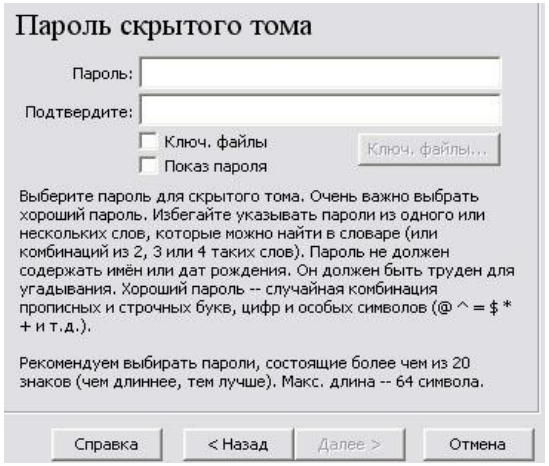

Рис. 12.21

Шаг 20

Данный шаг аналогичен шагу 10.

Прочитайте уведомление.

Скрытый том готов (рис. 12.22). Для создания еще одного тома нажмите кнопку **Далее**. Для прекращения работы Мастера создания томов кликните **Выход**.

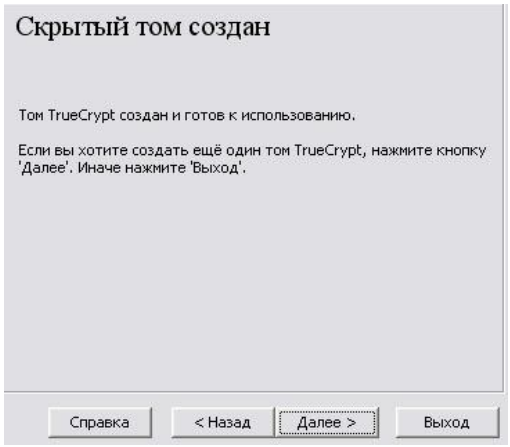

Рис. 12.22

Шаг 21

Работа со скрытым томом аналогична работе с внешним. Отличие лишь в том, что для доступа к скрытому тому необходимо вводить пароль из шага 19, а не из шага 9.

Содержание заданий

Зашифруйте:

- несистемный диск;
- часть диска;
- отдельные файлы разного формата;
- флешку.

Создайте виртуальные образы зашифрованных объектов и проверьте их работу.

Контрольные вопросы

1. Что значит термин шифрование «на лету»?
2. Какие есть режимы работы программы TrueCrypt?
3. Как удалить зашифрованный раздел?
4. Перечислите преимущества данного программного средства.

Отчетность по лабораторной работе

В тетради подробно опишите процесс создания шифрованной области винчестера и работы с виртуальным образом.

ЛАБОРАТОРНАЯ РАБОТА 13
Скрытие данных на винчестере

Цель работы: формирование умений работы с программными средствами, позволяющими скрывать отдельные файлы разных типов на жестком диске.

Теоретические сведения

В век высоких технологий информация представляется наибольшей ценностью. Поэтому неудивительно, что в последнее время создается множество средств для ее защиты. Среди соответствующих направлений наиболее развита криптография – алгоритмы постоянно совершенствуются, доказывается их стойкость, и у этого направления есть, по меньшей мере, два плюса. Во-первых, в отличие от теоретических принципов, в конкретные программные реализации могут закрадываться ошибки, приводящие к расшифровке за меньшее, чем расчетное, время. Во-вторых, очевидно, что в связи с развитием технологий через некоторое время перебор, занимающий на современном оборудовании не один год или даже десятилетие, будет выполняться за разумное время.

Основные принципы

Стеганография использует принципиально другой подход. Она скрывает не только информацию, но и сам факт ее наличия. В качестве примера из обычной жизни можно привести такой. Конечно, секретное письмо можно хранить в большом кованом сундуке с навесным замком, но можно и спрятать в потайном кармане. И если в первом случае информацию, возможно, кто-то попытается заполучить, то во втором случае у злоумышленника

не будет практически никаких зацепок, чтобы догадаться, где она может находиться.

Такой принцип сохранения и передачи ценой информации известен уже давно. Еще Геродот описывал послания, написанные на деревянных дощечках. В отличие от обычного способа записи, когда сначала наносился слой воска, а потом писался текст, здесь секретная запись выцарапывалась прямо на дощечке, которую потом покрывали воском, где уже и писали, чаще всего, ложное сообщение. Также известны случаи передачи сообщений на голове раба. Сначала его брили, затем писали сообщение, а когда волосы снова отрастали, отправляли в путь.

Компьютерная стеганография

Основной целью компьютерной стеганографии является скрытие файла-сообщения внутри файла-контейнера. Кроме того, такая операция должна остаться незамеченной. Файл-контейнер обязан не терять функций, а наличие скрытого сообщения должно быть максимально сложно обнаружено.

Рассмотрим основные направления программных реализаций.

1. Алгоритмы, основывающиеся на свойствах текста. Это направление наиболее близко к некомпьютерной стеганографии. В качестве такого универсального примера можно указать, например, акростих. Но есть и чисто компьютерные методы, основывающиеся, например, на сходстве написания кириллических и латинских символов (можно считать одни единицами, а другие – нулями). Также можно выделять отдельные буквы или слова из текста по определенному алгоритму. Это одно из немногих направлений в информационной безопасности, где собственные алгоритмы могут довольно успешно конкурировать с известными, уже использующимися, ведь чем менее изучен алгоритм, тем труднее будет определить наличие скрытого сообщения.

2. Методы, использующие особенности компьютерных форматов. Этот метод прост в реализации и зачастую не требует специального ПО. Конкретные примеры – поле комментариев в формате .jpeg и поле «Company» в свойствах, исполняемых .exe. Простота реализации оборачивается и простотой обнаружения. Хотя и данные алгоритмы могут использоваться тогда, когда у злоумышленников нет даже подозрения на наличие тайной информации.

3. Алгоритмы, использующие избыточность аудиовизуальной информации. Второе название этого метода – *метод младших битов*. Основными контейнерами в данном способе скрытия являются форматы так называемого *прямого кодирования*, например, .bmp для графики или .wav для звука. В них каждый минимальный элемент, каковым, например, является пиксель в .bmp, описывается отдельной записью и никак не связан с остальными. Так, в обычном .bmp на каждый пиксель отводится 24 бита – по 8 битов на канал. При изменении младшего бита изображение практически не изменится. Во всяком случае, не каждый человек и не всегда сможет заметить разницу между пустым и заполненным контейнером.

Это направление – самое популярное среди разработчиков. Современные программы научились обращаться с форматами, поддерживающими сжатие, а для самых популярных разработок появились дешифровщики.

Программа **Masker 7.0** (рис. 13.1) позволяет скрывать сообщения среди исполняемых видео- и аудиофайлов, а также в изображениях, причем поддерживается огромное число форматов, среди которых есть как форматы прямого кодирования, так и сжимающие (.jpeg, .mp3, .mpeg).

Программа Masker 7.0

Рис. 13.1

Чтобы начать работать, нужно на панели либо в меню выбрать пункт **Open Carrier File** и в появившемся окне указать файл-контейнер (рис. 13.2).

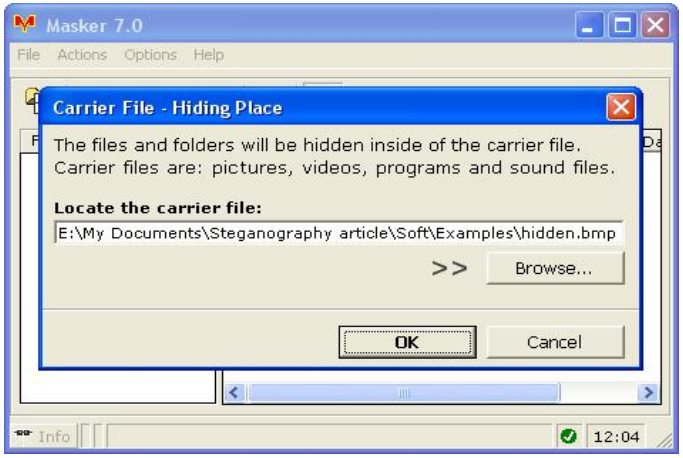

Рис. 13.2

После этого, в зависимости от ваших целей, нужно в следующем окне перейти на вкладку **Open Hideout**, где можно извлечь уже спрятанный файл, указав пароль, либо на **Create New Hideout**, где можно указать пароль и алгоритм шифрования для новых скрываемых данных. Шифрование – один из преимуществ программы: поддерживаются семь алгоритмов, среди которых есть BLOWFISH и TripleDES (рис. 13.3).

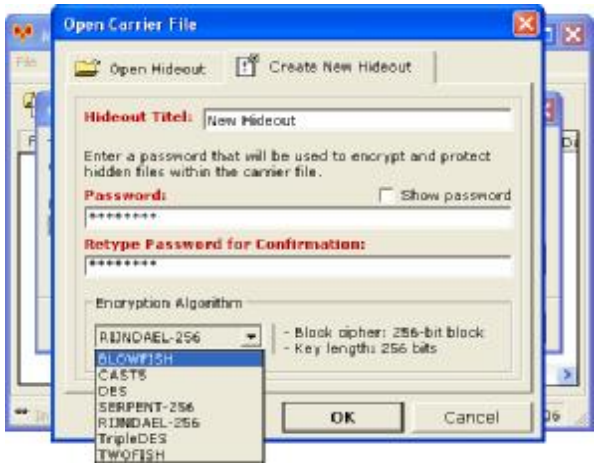

Рис. 13.3

После указания всех параметров в основной части окна будут отображаться скрытые файлы. Чтобы добавить туда файлы, нужно щелкнуть правой кнопкой мышки и выбрать **Hide/Add Files**. Появится окно, в котором нужно будет выбрать эти файлы, а затем и указать параметры их сохранения (рис. 13.4). Например, можно добавить целую папку, сохранив ее структуру, или дать файлу-контейнеру статус «read-only», чтобы сохранить скрытые файлы более надежно.

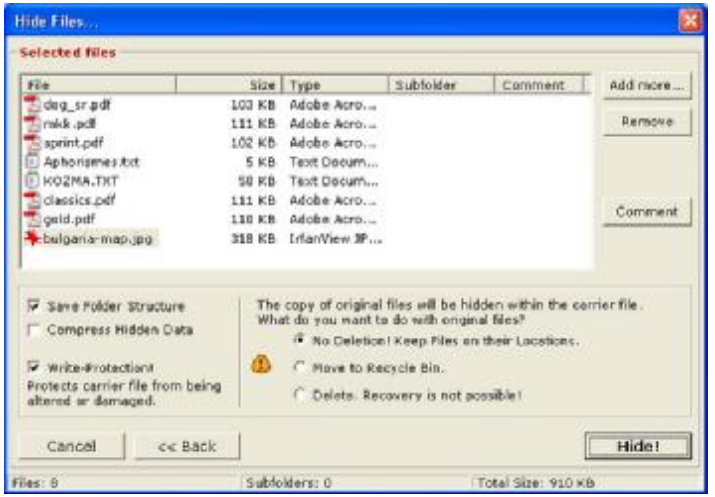

Рис. 13.4

Извлечение файлов не вызовет затруднений: при открытии файла-контейнера нужно зайти на нужную вкладку, указать пароль, и перед вами появится список спрятанных файлов.

Программа Safe Calculator

Программа не требует инсталляции, работает в портативном режиме. Это значит, что программа не оставляет следов на компьютере.

В сущности, программа представляет собой сейф, который содержит важные данные и открывается только после ввода пароля (здесь он называется PIN-код).

Вот так выглядит файл программы в файловом менеджере (рис. 13.5):

Рис. 13.5

А вот так при запуске (рис. 13.6):

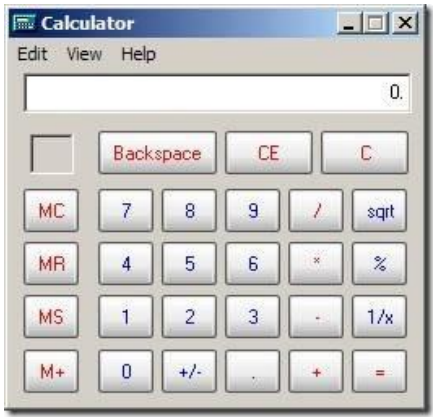

Рис. 13.6

В базовом режиме программа работает как стандартный Windows-калькулятор.

Для входа в секретный режим необходимо ввести PIN-код, заданный по умолчанию: 123. После этого нажимаем кнопку **MS**. Видим результат (рис. 13.7):

Рис. 13.7

Программа перешла в безопасный режим работы, в котором можно прятать данные.

Принцип работы прост: выберите файл, и программа сохранит его внутри своего тела, зашифрованным с помощью алгоритма RC4, который используется при WEP или WPA шифровании Wi-Fi сетей. Это не очень устойчивый алгоритм, но здесь идет расчет не на шифрование, а именно на стеганографию.

Для выбора файла необходимо нажать кнопку «+» в безопасном режиме работы программы. После этого необходимо нажать кнопку «=» для подтверждения того, что вам необходимо выбрать файл для сокрытия. Видим окно (рис. 13.8):

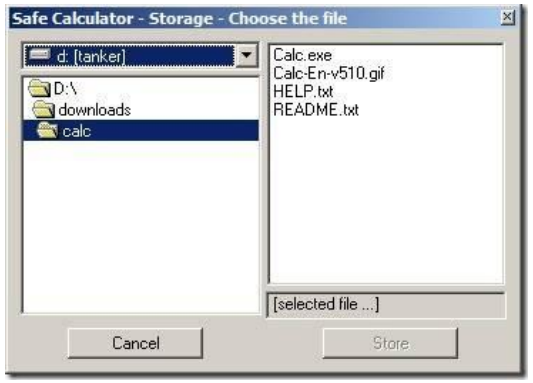

Рис. 13.8

Выбираем нужный файл (рис. 13.9):

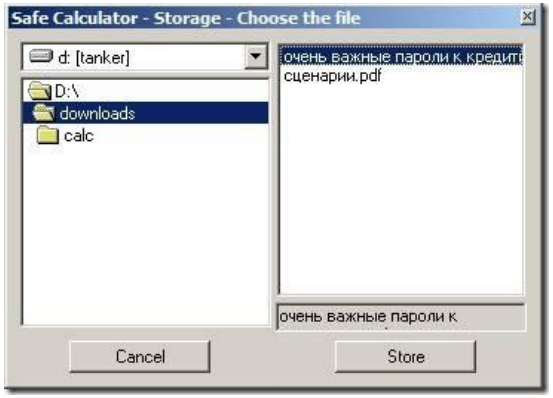

Рис. 13.9

В данном случае это текстовый файл с именем **очень важные пароли к кредиткам и банковским счетам.txt**. Жмем кнопку **Store**. Во время процесса сокрытия файла в тело программы, программа начнет мигать и исчезать на пару секунд – это нормальный процесс, когда программа изменяет свое тело. После успешного завершения операции увидим окно с отчетом (рис. 13.10):

Рис. 13.10

Для выхода из секретного режима необходимо нажать кнопку **MC**. Программа автоматически перейдет в нормальный режим калькулятора (рис. 13.11):

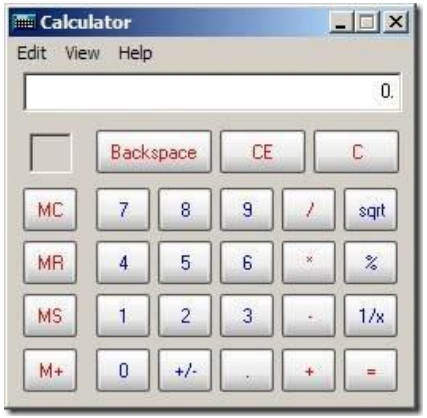

Рис. 13.11

Если посмотреть на файл программы в файловом менеджере (рис. 13.12), то станет видно, что размер файла увеличился (было 225 280 Мбайт).

Рис. 13.12

Скрытие больших файлов тоже работает, но и размер программы тоже станет огромным, и это натолкнет на подозрения – ведь таких простых калькуляторов размером в десятки и сотни Мбайт явно не существует.

Для получения доступа к скрытым файлам вновь введите PIN-код и нажмите **MS**, программа входит в секретный режим (рис. 13.13):

Рис. 13.13

Для работы с программой в секретном режиме предусмотрены несколько команд:

- **«+»** – добавление файлов для сокрытия. При этом файлы из первоначального размещения удаляются;
- **«-»** – извлечение файлов из хранилища. При этом файлы извлекаются в папку с программой. Из самой программы файлы удаляются;

- «*****» – копия защищенного файла помещается в папку программы. В самой программе также остается защищенный файл;
- «/» – просмотр или запуск файлов, которые помещены в тело программы. Если вы просмотрели текстовый файл, то он после просмотра удаляется из тела программы;
- **sqrt** – просмотр/запуск сохраненного файла, с последующим выходом из программы;
- **1/x** – замена сокрытого файла на новую версию файла с таким же именем и расширением;
- **MC** – возврат программы в режим классического калькулятора;
- **MS** – вывод на экран PIN-кода;
- **New PIN** – задание нового PIN-кода для доступа к программе. После нажатия необходимо ввести новый PIN, и нажать «=». PIN-код может быть от 1 до 32 цифр, в том числе отрицательным или дробным (в виде десятичной дроби). Для полного подтверждения нажмите клавишу **MS**;
- **Status** – вывод информации о сокрытых файлах;
- **Help** – вывод помощи. Работает в обоих режимах.

Если нажать кнопкой мышки на дисплей калькулятора, то можно включать/выключать режим скроллирования.

Кроме того, в режиме калькулятора есть одна важная команда: если нажать мышкой на дисплей калькулятора, то можно включать/выключать показ вводимого PIN-кода.

Содержание заданий

Скрыть текстовые документы, электронные таблицы, фотографии, аудиофайлы, видеоролики, фильмы и архивы различных объемов с помощью программ **Masker 7.0** и **Safe Calculator**. Определить, какой программой эффективнее скрывать какие файлы.

Контрольные вопросы

1. Что такое компьютерная стеганография?
2. Основные особенности реализации компьютерной стенографии?

3. Особенности программы **Masker 7.0**.
4. Особенности программы **Safe Calculator**.

Отчетность по лабораторной работе

В тетради подробно описать процесс скрытия различных файлов, и сделать вывод об эффективности скрытия каждого из видов файлов.

Литература

1. Варфоломеев, А. А. Управление ключами в системах криптографической защиты банковской информации / А. А. Варфоломеев, О. С. Доминина, М. Б. Пеленицын. – М. : МИФИ, 1996.
2. Основы информационной безопасности / Е. Б. Белов [и др.]. – М. : Горячая линия – Телеком, 2006.
3. Основы криптографии / А. П. Алферов [и др.]. – М. : Гелиос АРВ, 2002.
4. Романец, Ю. В. Защита информации в компьютерных системах и сетях / Ю. В. Романец, П. А. Тимофеев, В. Ф. Шаньгин. – М. : Радио и связь, 2001.
5. Хамидуллин, Р. Р. Методы и средства защиты компьютерной информации / Р. Р. Хамидуллин, И. А. Бригаднов, А. В. Морозов. – СПб., 2005.
6. Харин, Ю. С. Математические основы криптологии : учеб. пособие / Ю. С. Харин, В. И. Беоник, Г. В. Матвеев. – Минск : БГУ, 1999.

Приложение 1
(справочное)

Таблица Виженера для русского алфавита

	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
0	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
1	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а
2	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б
3	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в
4	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г
5	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д
6	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е
7	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж
8	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з
9	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и
10	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й
11	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к
12	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л
13	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м
14	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н
15	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о
16	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
17	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р
18	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с
19	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т
20	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у
21	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф
22	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х
23	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц
24	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч
25	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш
26	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ
27	ы	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ
28	ь	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы
29	э	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь
30	ю	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э
31	я	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю

Оглавление

Предисловие 3

Лабораторная работа 1
Шифрование с использованием метода
шифрующих таблиц и метода магического квадрата 4

Лабораторная работа 2
Шифрование с использованием систем Цезаря
и системы Трисемуса 15

Лабораторная работа 3
Реализация алгоритма шифрования Плейфейра 21

Лабораторная работа 4
Шифрование с использованием системы Вижинера
и шифра «двойной квадрат» Уитстона 24

Лабораторная работа 5
Реализация элементов криптосистемы RSA 27

Лабораторная работа 6
Реализация элементов схемы шифрования Эль-Гамаля 31

Лабораторная работа 7
Реализация элементов схемы шифрования
ГОСТ 28147–89 33

Лабораторная работа 8
Протокол идентификации с нулевой передачей данных 41

Лабораторная работа 9
Параллельная схема протокола интенсификации
с нулевой передачей данных 44

Лабораторная работа 10
Реализация элементов ЭЦП RSA 46

Лабораторная работа 11
Реализация элементов ЭЦП ГОСТ Р 34.10–94 48

Лабораторная работа 12
Создание виртуальных зашифрованных дисков
(программное средство TrueCrypt) 50

Лабораторная работа 13
Скрытие данных на винчестере 65

Литература 77

Приложение 1 **Таблица Виженера**
для русского алфавита 78