МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

В.И. Глухих

ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ И ЗАЩИТА ДАННЫХ

Рекомендовано в качестве учебного пособия
Редакционно-издательским советом
Иркутского государственного технического университета

Издательство
Иркутского государственного технического университета
2011

УДК 004.43(031)
ББК 32.973.26-018.2
С72

Глухих В.И.

С72 Информационная безопасность и защита данных: учебное пособие / В.И. Глухих; Иркутский государственный технический университет. − Иркутск: Изд-во Иркутского государственного технического университета, 2011. – 250 с.

В пособии формулируются основные понятия информационной безопасности, анализируются источники угрозы целостности и конфиденциальности данных. Приводятся основные международные и российские стандарты информационной безопасности. Рассматриваются современные методы защиты информации от непреднамеренных и умышленных искажений, неправомерного использования данных. Описываются основные криптографические методы и алгоритмы защиты информации в компьютерных системах и информационно-вычислительных сетях. Подробно рассматриваются технологии идентификации и аутентификации пользователей конфиденциальной информации.

Пособие предназначено для обучения магистров по специальности 230100 «Сети ЭВМ и телекоммуникации».

УДК 004.43(031)
ББК 32.973.26-018.2

Рецензент

Доктор физико-математических наук,
заведующий кафедрой радиофизики
Иркутского государственного университета
В.И. Сажин

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Главная тенденция развития современного общества тесно связана с ростом информационной составляющей (информационные ресурсы, информационные технологии и т.п.) и, как следствие, информационной безопасности. Вопросы информационной безопасности на современном этапе рассматриваются как приоритетные в государственных структурах, в научных учреждениях и в коммерческих фирмах. Информационные системы специального назначения (банковские системы, силовые ведомства и т.п.), являясь приоритетными в структуре государства, не могут оставаться в вопросах обеспечения информационной безопасности только на уровне традиционных средств: криптографическая защита, совершенствование систем разделения доступа, реализация специальных требований для абонентского трафика, проведение организационных мероприятий по усилению режима.

Нет смысла перечислять все преимущества, которые получает организация, подключившись к глобальной сети Интернет. Однако при этом необходимо учитывать и отрицательные стороны такой акции. В общедоступной сети – это возможные атаки на подключенные к ней локальные сети и компьютеры. Общепризнанно, что ежегодные убытки из-за недостаточно защищенных корпоративных информационных систем исчисляются десятками миллионов долларов. Видя свои преимущества от использования сетевых технологий, первоначально пользователи часто не придают значения выбору средств защиты информации, реально ставя под угрозу свое финансовое положение, репутацию и конкурентоспособность.

Информационная безопасность, как и защита информации, задача комплексная, направленная на обеспечение безопасности, реализуемая внедрением системы безопасности. Проблема защиты информации является многоплановой и комплексной и охватывает ряд важных задач. Проблемы информационной безопасности постоянно усугубляются процессами проникновения во все сферы общества технических средств обработки и передачи данных и, прежде всего, вычислительных систем.

Защита информации – это лишь одна составляющая задачи обеспечения информационной безопасности. Другая ее часть – это обеспечение бесперебойной работы оборудования. Выход из строя того или иного узла в результате хакерской атаки приводит как к затратам на его восстановление, когда требуется обновить или заменить программное обеспечение, так и к потере части клиентуры. Поэтому при включении компьютера в сеть, при интеграции корпоративной информационной системы в сеть необходимо в первую очередь продумать вопросы обеспечения защиты этой системы.

На сегодняшний день сформулировано три базовых принципа, которые должна обеспечивать информационная безопасность:

- целостность данных – защита от сбоев, ведущих к потере информации, а также зашита от неавторизованного создания или уничтожения данных;
- конфиденциальность информации;
- доступность информации для всех авторизованных пользователей.

При разработке компьютерных систем, выход из строя или ошибки в работе которых могут привести к тяжелым последствиям, вопросы компьютерной

безопасности становятся первоочередными. Известно много мер, направленных на обеспечение компьютерной безопасности, основными среди них являются технические, организационные и правовые.

Обеспечение безопасности информации – дорогое дело, и не только из-за затрат на закупку или установку средств защиты, но также из-за того, что бывает трудно квалифицированно определить границы разумной безопасности и обеспечить соответствующее поддержание системы в работоспособном состоянии.

Существующие на сегодняшний день методы и средства защиты информации в автоматизированных системах достаточно разнообразны, что отражает многообразие способов и средств возможных несанкционированных действий. Главным недостатком существующих методов и средств защиты информации, включая современные средства поиска уязвимостей автоматизированных систем и обнаружения несанкционированных действий, является то, что они, в подавляющем большинстве случаев, позволяют организовать защиту информации лишь от постфактум выявленных угроз, что отражает определенную степень пассивности обороны.

Адекватный уровень информационной безопасности в состоянии обеспечить только комплексный подход, предполагающий целенаправленное использование традиционных организационных и программно-технических правил обеспечения безопасности на единой концептуальной основе с одновременным поиском и постоянным изучением новых приемов и средств, как защиты информации, так и хакерских атак на нее.

В данном пособии основное внимание уделено двум источникам угроз информации – умышленным и неумышленным воздействиям.

Неумышленные или непреднамеренные воздействия связаны с объективными природными процессами, явлениями, особенностями конструктивного строения оборудования и уровня технологического развития. Статистические закономерности этих воздействий хорошо изучены. Основным средством защиты от непреднамеренных искажений информации при ее хранении или передаче на расстоянии являются различные способы кодирования.

Умышленные искажения информации, как и неправомерный доступ к ней, носят обычно корыстный характер и обусловлены особенностями образа жизни и характера хозяйственной и др. деятельности людей. Отсюда неправомерная атака на информацию, так и защита от нее носят антагонистический соревновательный характер. Все что ни придумает один человек, рано или поздно всегда разгадает другой человек. Отсутствует абсолютный метод защиты находящейся в обращении информации. Вопрос времени – информация считается защищенной, если время на ее вскрытие (взлом защиты) превосходит время актуальности информации. Отсюда основной способ защиты информации от умышленных воздействий – криптирование или шифрование. Криптирование, это обратимое искажение информации с помощью ключа, знание которого позволяет законному владельцу практически мгновенно восстановить информацию, а оппонент, не знающий ключа, должен потратить на это десятки, а может и тысячи лет …

Настоящее пособие не является инструкцией по применению тех или иных приемов защиты, его главная цель повышение общей культуры пользователя в вопросах информационной безопасности, т.е. предполагается определенное знание им основ современных сетевых и телекоммуникационных технологий.

ЧАСТЬ I. ПОМЕХОУСТОЙЧИВОЕ КОДИРОВАНИЕ

Одной из причин искажения информации при ее хранении или передаче по каналам связи являются непреднамеренные искажения, связанные с шумами, помехами, сбоями в работе оборудования и т.п. Возникновение искажений обусловлено, с одной стороны, объективно протекающими физическими процессами в средах хранения и передачи информации и, с другой стороны, несовершенством оборудования, обеспечивающего хранение, передачу и обработку информации.

В принципе возможно создание сколь угодно совершенного оборудования для хранения, передачи и обработки информации, но это крайне дорого и все равно не достаточно для обеспечения абсолютной надежности (достоверности) информации.

Для текущего уровня развития техники, задача обеспечения целостности информации, т.е. обеспечения ее защиты от неумышленных искажений, решается путем специального конструирования данных – носителей информации, таким образом, чтобы данные содержали средства для обнаружения и исправления искажения информации. Такими средствами являются различные методы кодирования данных.

1.1. ОСНОВНЫЕ ПОНЯТИЯ

Информация (от лат. *informatio* — разъяснение, изложение, осведомленность) — понятие, связанное с объективным свойством материальных объектов и явлений (процессов) порождать многообразие состояний, которые посредством взаимодействий передаются другим объектам и запечатлеваются в их структуре. В настоящее время не существует единого определения термина *информация*. С точки зрения различных областей знания, данное понятие описывается своим специфическим набором признаков.

Будем считать, что информация это сведения о состоянии интересующего нас объекта. Если объект имеет счетное (конечное) число состояний, то информация о нем называется цифровой или дискретной. Если же объект имеет несчетное (бесконечное) число состояний, то информация о нем называется непрерывной. Дискретная информация может быть представлена в виде перечня состояний объекта, а непрерывная – в виде графика или функции.

Из приведенного представления вытекает выбор единицы измерения количества дискретной информации. Пусть объект может находиться всего в двух состояниях. Присвоим одному из них код «1», а другому – «0». Это минимальное количество информации, которое может содержать объект. Оно и является единицей измерения информации и называется **бит** (от английского bit – binary digit –двоичная цифра).

Кодирование дискретной информации представляет собой описание состояний объекта с помощью алфавита, число символов которого меньше числа состояний объекта. Число символов алфавита носит название **основание кода** (*d*).

Математическая теория информации показывает, что для ряда приложений, оптимальным основанием кода является: $d_{опт} = e \approx 2.718281828$ – основание натурального логарифма.

Действительно, первые ЭВМ имели в своем составе машины, представлявшие внутренние данные с основанием кода близким к оптимальному основанию – $d=2$ и

d=3. Однако, наибольшее распространение получили «двоичные» ЭВМ, так как в их основе лежит элементная база (логические интегральные схемы), которые наиболее просто реализуют двоичные вычисления и двоичное представление данных.

Равномерное кодирование представляет собой простейший и наиболее распространенный способ кодирования, когда состояние объекта описывается словом-состоянием, имеющим число символов: $N = \log_d(S)$, где d – основание кода алфавита, S – порядковый номер состояния объекта.

В табл.1.1 приведены примеры равномерных кодов по основанию *d=2* и *d=3* для букв русского и английского алфавитов.

Таблица 1.1. Равномерный код и частоты встречаемости букв алфавитов

	Русский алфавит				Английский алфавит			
№	Буква	Двоичный код	Троичный код	Частота	Буква	Двоичный код	Троичный код	Частота
0	**О**	00000	0000	*0,0764*	**E**	00000	000	*0,1286*
1	**Е**	00001	0001	*0,0732*	**T**	00001	001	*0,0972*
2	**А**	00010	0002	*0,0629*	**A**	00010	002	*0,0796*
3	**И**	00011	0010	*0,0577*	**I**	00011	010	*0,0777*
4	**Т**	00100	0011	*0,0549*	**N**	00100	011	*0,0751*
5	**Н**	00101	0012	*0,049*	**R**	00101	012	*0,0683*
6	**Р**	00110	0020	*0,0459*	**O**	00110	020	*0,0662*
7	**С**	00111	0021	*0,0404*	**S**	00111	021	*0,0662*
8	**В**	01000	0022	*0,0355*	**H**	01000	022	*0,0539*
9	**П**	01001	0100	*0,033*	**D**	01001	100	*0,0401*
10	**К**	01010	0101	*0,0302*	**L**	01010	101	*0,0351*
11	**Л**	01011	0102	*0,0299*	**C**	01011	102	*0,0284*
12	**М**	01100	0110	*0,0275*	**F**	01100	110	*0,0262*
13	**Д**	01101	0111	*0,065*	**U**	01101	111	*0,0248*
14	**У**	01110	0112	*0,0222*	**M**	01110	112	*0,0243*
15	**Я**	01111	0120	*0,0153*	**G**	01111	120	*0,0199*
16	**Ы**	10000	0121	*0,0143*	**P**	10000	121	*0,0181*
17	**Ь**	10001	0122	*0,0138*	**W**	10001	122	*0,018*
18	**З**	10010	0200	*0,0133*	**B**	10010	200	*0,016*
19	**Й**	10011	0201	*0,0125*	**Y**	10011	201	*0,0152*
20	**Б**	10100	0202	*0,0114*	**V**	10100	202	*0,0115*
21	**Ч**	10101	0210	*0,0094*	**K**	10101	210	*0,0041*
22	**Г**	10110	0211	*0,0083*	**Q**	10110	211	*0,0017*
23	**Ю**	10111	0212	*0,0081*	**X**	10111	212	*0,0017*
24	**Ж**	11000	0220	*0,0079*	**J**	11000	220	*0,0016*
25	**Х**	11001	0221	*0,0048*	**Z**	11001	221	*0,0005*
26	**Щ**	11010	0222	*0,0042*				
27	**Ф**	11011	1000	*0,0036*				
28	**Ш**	11100	1001	*0,0026*				
29	**Э**	11101	1002	*0,0023*				
30	**Ц**	11110	1010	*0,0021*				
31	**Ъ**	11111	1011	*0,0003*				

Кодирование непрерывной информации осуществляется путем замены бесконечно близко отстоящих друг от друга значений непрерывного сигнала на дискретные значения, отстоящие друг от друга на равные промежутки времени *T*. Эти дискретные значения называются «выборками», а процедура преобразования – дискретизацией или оцифровыванием непрерывного сигнала. Дискретизация выполняется с помощью аналого-цифровых преобразователей (АЦП) с частотой дискретизации $F = 1/T$ и глубиной преобразования определяемой разрядностью АЦП. Полученные таким образом выборки, кодируются и обрабатываются как обычная дискретная информация, рис. 1.1.

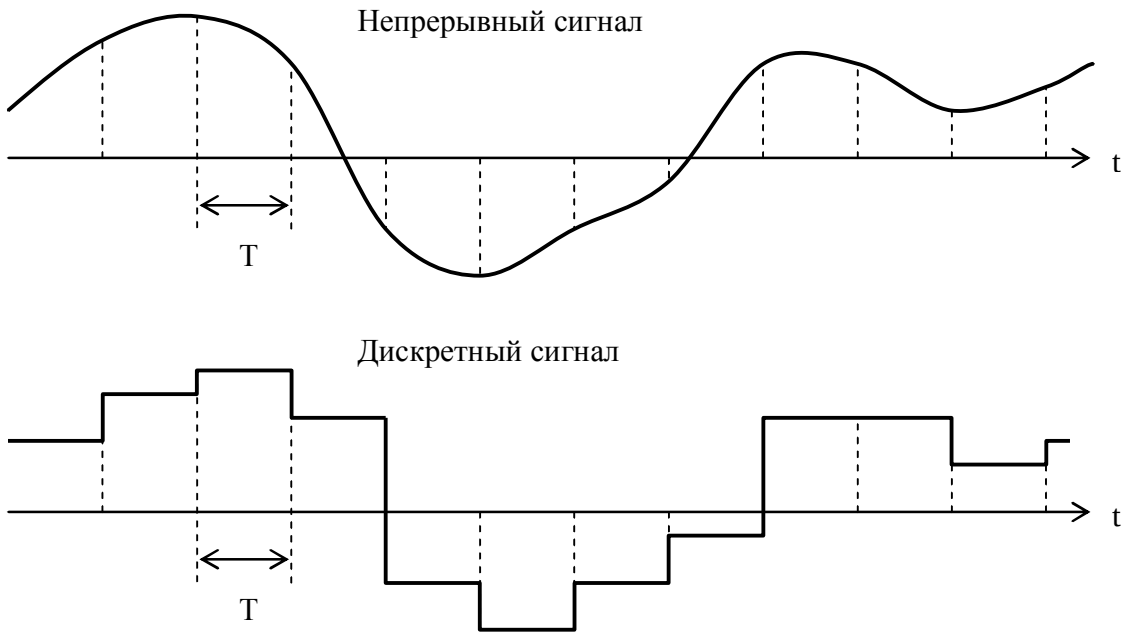


Рис. 1.1. Дискретизация непрерывного сигнала

Чем выше частота дискретизации, тем точнее происходит преобразование непрерывной информации в дискретную информацию. Но с ростом частоты дискретизации растет и объем дискретных данных, получаемых при таком преобразовании. Это, в свою очередь, увеличивает сложность обработки, передачи и хранения данных. Для повышения точности дискретизации необязательно безграничное увеличение ее частоты. Эту частоту разумно увеличивать только до предела, определяемого теоремой о выборках, называемой теоремой Котельникова-Найквиста (*Nyquist*).

Согласно преобразованию Фурье, любая непрерывная величина описывается множеством наложенных друг на друга волновых процессов, называемых гармониками. Гармоники это функции вида $A\sin(wt + j)$, где A – амплитуда, w – круговая частота, t – время и j – фаза. Преобразование Фурье переводит временное представление сигналов в их частотное представление – спектр. Любые технически реализуемые сигналы имеют ограниченный сверху спектр частот, характеризуемый граничной частотой $F_{ГР}$, в пределах которой сосредоточено около 90% энергии сигнала.

Теорема о выборках утверждает, что для однозначного воспроизведения непрерывного сигнала по его выборкам, шаг дискретизации T должен определяться соотношением:

$$T \le \frac{1}{2F_{ГР}}. \qquad (1.1)$$

Если шаг выборки будет больше, то произойдет необратимая потеря части информации, если меньше, то хранение или передача информации будет осуществляться неэкономично.

Примером использования этой теоремы являются лазерные компакт-диски, звуковая информация на которых хранится в цифровой форме. Чем выше будет частота дискретизации, тем точнее будут воспроизводиться звуки и тем меньше их можно будет записать на один диск, но ухо обычного человека способно различать звуки с частотой до 20 кГц, поэтому точно записывать звуки с большей частотой бессмысленно. Согласно теореме о выборках частоту дискретизации нужно выбрать не меньшей 40 кГц (в промышленном стандарте используется частота 44.1 кГц).

В цифровых сетях телекоммуникаций носителем информации являются прямоугольные импульсы тока или напряжения. Частотный спектр прямоугольного импульса похож на косинусоиду с непрерывно убывающей амплитудой и для него 90% энергии сигнала сосредоточены до частоты $F_{ГР} = 1/t$, где τ - длительность импульса. Для обеспечения скорости передачи данных 1 Гбит/с необходимо разрешение между двумя соседними импульсами менее 1нс (10^{-9} с). Согласно теореме о выборках, аппаратура сетей телекоммуникаций должна обеспечивать полосу пропускания аналоговых сигналов от 0 до 0.5 ГГц.

1.2. НЕРАВНОМЕРНЫЕ КОДЫ

Неравномерные коды позволяют сжимать данные, т.е. уменьшить количество бит для хранения или передачи заданной информации. Такое сжатие дает возможность передавать сообщения более быстро и хранить более экономно и оперативно. Сжатие данных позволяет записать больше информации на дискету, «увеличить» размер жесткого диска, ускорить работу с модемом и т.д. При работе с компьютерами широко используются программы-архиваторы данных формата ZIP, GZ, ARJ и других. Методы сжатия информации были разработаны как математическая теория, которая долгое время (до первой половины 80-х годов прошлого века), мало использовалась на практике.

Из табл. 1.1 видно, что одни буквы в текстовом сообщении встречаются чаще, другие реже. Если часто встречающиеся буквы кодировать меньшим числом бит, а редко встречающиеся большим числом бит, то для передачи (хранения) такой же информации, как и при равномерном кодировании, потребуется меньшее количество бит.

Впервые эта идея была высказана в 1948 году независимо друг от друга двумя американскими учеными: Шенноном (*Shannon*) и Фано (*Fano*).

Код Шеннона-Фано строится с помощью дерева (графа Фано). Построение этого дерева начинается от корня. Всё множество кодируемых элементов соответствует корню дерева (вершине первого уровня). Оно разбивается на два подмножества с примерно одинаковыми суммарными вероятностями. Эти подмножества соответствуют двум вершинам второго уровня, которые соединяются с корнем. Далее каждое из этих подмножеств разбивается на два подмножества с

примерно одинаковыми суммарными вероятностями. Им соответствуют вершины третьего уровня. Если подмножество содержит единственный элемент, то ему соответствует концевая вершина кодового дерева и такое подмножество дальнейшему разбиению не подлежит. Подобным образом поступают до тех пор, пока не получатся все концевые вершины. Ветви кодового дерева размечают символами «1» и «0».

Эффективность кодирования характеризуется средней длиной кодового слова $L_{ср}$:

$$L_{ср} = \sum_{i=1}^{N} p_i \times n_i, \qquad (1.2)$$

где N – число состояний объекта, p_i – вероятность i-го состояния, n_i – число символов в описании i-го состояния.

На рис. 1.2 приведен пример построения дерева кода Шеннона-Фано по данным табл. 1.2, который существенно экономичнее равномерного кода ($L_{ср}$=2.30 против $L_{ср}$=3.00 для равномерного кода).

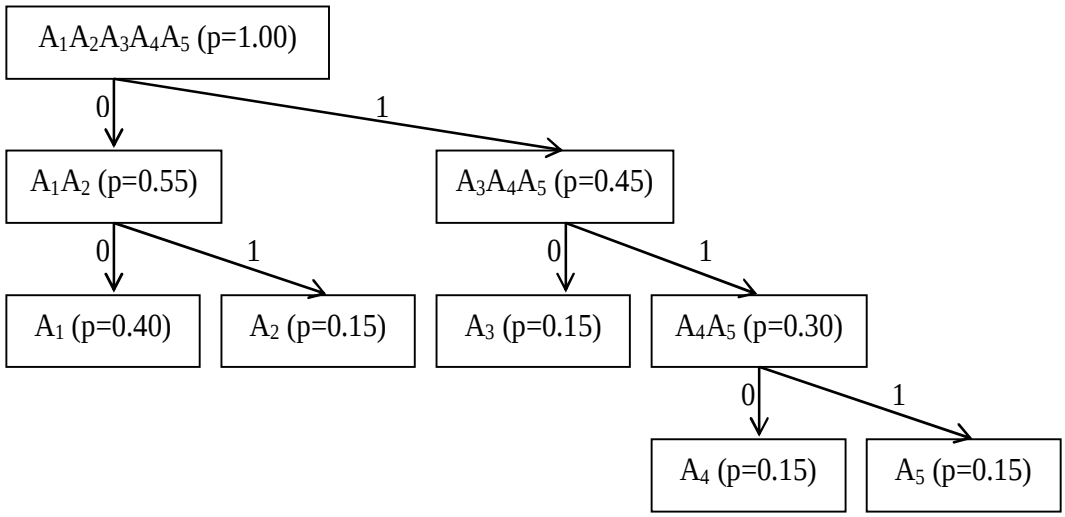


Рис. 1.2. Построение кода Шеннона-Фано с помощью графа Фано

В табл. 1.2 приведены примеры кодов для описания дискретного объекта с пятью состояниями. Наибольшую среднюю длину имеет равномерный код $L_{ср}$=3.00, который и является эталоном «неэффективности» кодирования данных.

Таблица 1.2. Неравномерные коды объекта с 5 состояниями

Состояния	p_i	Равномерный код	Код Шеннона-Фано	Код Хаффмена	Беспрефексный код
A_1	0.4	000	00	1	0
A_2	0.15	001	01	010	1
A_3	0.15	010	10	011	00
A_4	0.15	011	110	000	01
A_5	0.15	100	111	001	10
$L_{ср}$		***3.00***	***2.30***	***2.20***	***1.45***

Метод Хаффмена (*Huffman*) разработан в 1952 г. Он более практичен и по степени сжатия не уступает методу Шеннона-Фано. Более того, он сжимает максимально плотно и поэтому является **оптимальным кодом** или наиболее эффективным кодом. Код строится при помощи двоичного дерева. Вероятности значений дискретных величин приписываются его листьям; все дерево строится, опираясь на листья. Величина, приписанная к узлу дерева, называется весом узла. Два листа с наименьшими весами создают родительский узел с весом, равным сумме их весов; в дальнейшем этот узел учитывается наравне с оставшимися листьями, а образовавшие его узлы от такого рассмотрения устраняются. После постройки корня нужно приписать каждой из ветвей, исходящих из родительских узлов, значения 0 или 1. Код каждого значения дискретной величины – это число, получаемое при обходе ветвей от корня к листу, соответствующему данному значению.

Для методов Хаффмена и Шеннона-Фано каждый раз вместе с собственно сообщением нужно передавать и таблицу кодов. Например, для случая из примера, табл.1.2, нужно сообщить, что коду 00 соответствует состояние A_1, коду 01 – A_2 и т.д.

Коды Шеннона-Фано, Хаффмена и равномерный код относятся к **префиксным кодам**, у которых никакое кодовое слово не является началом другого кодового слова, т.е. префиксом. Любое сообщение из префиксных кодов может быть однозначно последовательно декодировано, начиная с первого слова. Так, для кода Шеннона-Фано: 00000111100000101010 0 → 00 00 01 111 00 00 01 01 01 00 → $A_1A_1A_2A_5A_1A_1A_2A_2A_2A_1$.

Беспрефиксные коды, у которых одни кодовые слова могут быть началами других кодовых слов, обладают еще большей экономичностью. Однако для обеспечения однозначной декодируемости требуют введения специальных разделительных символов между словами, например, запятой «,».

ВЫВОДЫ:

1) Исключение избыточности данных, позволяет сокращать размеры памяти при хранении или уменьшать время передачи одного и того же объема информации.
2) Замечено, что все естественные языки, обладая существенной избыточностью, обладают свойством невосприимчивости к искажениям. Такие дефекты речи как акцент, заикание, невыговаривание отдельных звуков и звукосочетаний, почти не сказываются на правильном восприятии информации.
3) Оптимальные коды, полностью лишенные избыточности, оказываются совершенно беззащитными перед помехами и искажениями. Утрата или изменение даже одного бита в начале передачи оптимального кода может сделать невозможным правильное декодирование всего сообщения.

Приведенные соображения легли в основу защиты информации от естественных источников искажений сигналов несущих информацию: шумы, помехи, сбои в работе аппаратуры, воздействия температуры, радиации, ионизирующего излучения и др. **Помехоустойчивость сообщения обеспечивается путем придания ему избыточности**.

1.3. ИЗБЫТОЧНЫЕ КОДЫ

1.3.1. ПРИНЦИПЫ ПОМЕХОУСТОЙЧИВОГО КОДИРОВАНИЯ

В реальных условиях прием двоичных символов может происходить с ошибками, когда вместо символа «1» принимается символ «0» и наоборот. Ошибки могут возникать из-за помех, действующих в канале связи (особенно помех импульсного характера), изменения за время передачи характеристик канала (например, замирания), снижения уровня сигнала передачи, нестабильности амплитудно- и фазо-частотных характеристик канала и т.п. Критерием оценки качества передачи в дискретных каналах является, нормированная на знак или символ, допустимая вероятность ошибки для данного вида сообщений. Так, допустимая вероятность ошибки при телеграфной связи может составлять 10^{-3} (на знак), а при передаче данных – не более 10^{-6} (на символ). Для обеспечения таких значений вероятностей, одного улучшения качественных показателей канала связи может оказаться недостаточным. Поэтому основной мерой защиты данных от искажений является применение специальных методов повышающих качество приема передаваемой информации. Эти методы можно разбить на две группы.

К первой группе относятся технические методы увеличения помехоустойчивости приема единичных элементов (символов) дискретной информации, связанные с выбором уровня сигнала, отношения сигнал-шум, ширины полосы пропускания канала, метода модуляции, способа приема и т. п.

Ко второй группе относятся методы обнаружения и исправления ошибок, основанные на искусственном введении избыточности в передаваемое сообщение.

Практические возможности увеличения помехоустойчивости за счет мощности и ширины спектра сигнала в системах передачи дискретной информации по стандартным каналам сетей коммуникаций резко ограничены. Поэтому для повышения качества приема используют следующие основные способы:

1) многократную передачу кодовых комбинаций (метод повторения);
2) одновременную передачу кодовой комбинации по нескольким параллельно работающим каналам;
3) помехоустойчивое (корректирующее) кодирование, т. е. использование кодов, исправляющих ошибки.

Иногда применяют комбинации этих способов.

Многократное повторение кодовой комбинации является самым простым способом повышения достоверности приема и легко реализуется, особенно в низкоскоростных системах передачи для каналов с быстро меняющимися параметрами.

Способу многократного повторения аналогичен способ передачи одной и той же информации по нескольким параллельным каналам связи. В этом случае необходимо иметь не менее трех каналов связи (например, с частотным разнесением), несущие частоты которых нужно выбирать таким образом, чтобы ошибки в каналах были независимы. Достоинством таких систем являются надежность и малое время задержки в получении информации. Основным недостатком многоканальных систем так же, как и систем с повторением, является нерациональное использование избыточности.

Наиболее рационально избыточность используется при применении помехоустойчивых (корректирующих) кодов.

В обычном равномерном непомехоустойчивом коде число разрядов n в кодовых комбинациях определяется числом сообщений и основанием кода.

Коды, у которых все кодовые комбинации разрешены, называются простыми или равнодоступными и являются полностью безызбыточными. Безызбыточные коды обладают большой «чувствительностью» к помехам. Внесение избыточности при использовании помехоустойчивых кодов связано с увеличением n – числа разрядов кодовой комбинации. Таким образом, все множество $N = 2^n$ комбинаций можно разбить на два подмножества: подмножество разрешенных комбинаций, обладающих определенными признаками, и подмножество запрещенных комбинаций, этими признаками не обладающих.

Помехоустойчивый код отличается от обычного кода тем, что в канал передаются не все кодовые комбинации N, которые можно сформировать из имеющегося числа разрядов n, а только их часть N_k, которая составляет подмножество разрешенных комбинаций. Если при приеме выясняется, что кодовая комбинация принадлежит к запрещенным, то это свидетельствует о наличии ошибок в комбинации, т.е. таким образом решается задача обнаружения ошибок. При этом принятая комбинация не декодируется (не принимается решение о переданном сообщении). В связи с этим помехоустойчивые коды называют корректирующими кодами. Корректирующие свойства избыточных кодов зависят от правила их построения, определяющего структуру кода, и параметров кода (длительности символов, числа разрядов, избыточности и т. п.).

Первые работы по корректирующим кодам принадлежат Хеммингу, который ввел понятие минимального кодового расстояния $d_{\min}$ и предложил код, позволяющий однозначно указать ту позицию в кодовой комбинации, где произошла ошибка. К информационным элементам k в коде Хемминга добавляется m проверочных элементов для автоматического определения местоположения ошибочного символа. Таким образом, общая длина кодовой комбинации составляет: $n = k + m$.

1.3.2. КЛАССИФИКАЦИЯ ПОМЕХОУСТОЙЧИВЫХ КОДОВ

Помехоустойчивые (корректирующие) коды делятся на блочные и непрерывные или поточные.

Блочными называются коды, в которых информационный поток символов разбивается на отрезки и каждый из них преобразуется в определенную последовательность (блок) кодовых символов. В блочных кодах кодирование при передаче (формирование проверочных элементов) и декодирование при приеме (обнаружение и исправление ошибок) выполняются в пределах каждой кодовой комбинации (блока) в отдельности по соответствующим алгоритмам.

Непрерывные или рекуррентные коды образуют последовательность символов, не разделяемую на отдельные кодовые комбинации. Кодирование и декодирование непрерывно совершаются над последовательностью элементов без деления их на блоки. Формирование проверочных символов ведется по рекуррентным (возвратным) правилам, поэтому непрерывные коды часто называют рекуррентными или цепными.

В простейшем цепном коде каждый проверочный элемент формируется путем сложения по модулю 2 соседних или отстоящих друг от друга на определенное число позиций информационных элементов. В канал телекоммуникаций передается

последовательность импульсов, в которой за каждым информационным следует проверочный. Подобную чередующуюся последовательность разрядов имеет, например корреляционный манчестерский код.

К непрерывным кодам относятся и сверточные коды, в которых каждый информационный символ, поступающий на вход кодирующего устройства, вызывает появление на его выходе ряда проверочных элементов, образованных суммированием *«по модулю 2»* данного символа и *k–1* предыдущих информационных символов. Рекуррентные коды позволяют исправлять групповые ошибки («пачки») в телекоммуникационных каналах. Блочные коды делятся на равномерные и неравномерные. В равномерных кодах, в отличие от неравномерных, все кодовые комбинации содержат одинаковое число n символов (разрядов) с постоянной длительностью t_0 импульсов символов кода. Равномерные коды в основном и применяются в системах связи, так как это упрощает технику передачи и приема.

Классическими примерами неравномерного кода являются код Морзе, широко применяемый в телеграфии, и код Хаффмена, применяемый для компрессии информации (факсимильная связь, ЭВМ).

Никаких специальных мер по исправлению и обнаружению ошибок в коде Морзе не предусматривается в связи с большой избыточностью самого передаваемого текста. В этом смысле код Морзе не относится к классу корректирующих кодов.

Почти все блочные корректирующие коды принадлежат к разделяемым кодам, в которых кодовые комбинации состоят из двух частей: информационной и проверочной. Их символы всегда занимают одни и те же позиции, т. е. располагаются на определенных местах. Как правило, в таких кодах, все кодовые комбинации которых содержат n символов, первые k символов являются информационными, а за ними располагаются $n-k=m$ – проверочных символов. В соответствии с этим разделяемые коды получили условное обозначение – (n,k)-коды. В неразделяемых кодах деление на информационные и проверочные символы отсутствует. К таким кодам относятся, в частности, коды с постоянным весом, так называемые равновесные коды.

Систематические коды образуют наиболее обширную группу (n,k)-разделимых кодов. Особенностью этих кодов является то, что проверочные (корректирующие) символы образуются с помощью линейных операций над информационными символами. Кроме того, любая разрешенная кодовая комбинация может быть получена в результате линейной операции над набором k линейно независимых кодовых комбинаций.

В частности, суммирование «по модулю 2» двух и более разрешенных комбинаций также дает разрешенную кодовую комбинацию. Поскольку теоретической основой получения таких комбинаций является математический аппарат линейной алгебры, то коды и называют линейными, а учитывая, что проверочные символы формируются по определенной системе (правилам), блочные равномерные разделимые линейные коды получили название систематических. Использование аппарата линейной алгебры породило и другое название – групповые коды.

Эти коды получили наибольшее применение в системах передачи дискретной информации. Несистематические (нелинейные) коды указанными выше свойствами

не обладают и применяются значительно реже в специальных случаях. Примером нелинейного кода является уже упоминавшийся неразделимый, равновесный код. Эти коды обычно используются в несимметричных каналах связи, в которых вероятность перехода «1» → «0» значительно больше вероятности перехода «0» → «1», или наоборот. В таких каналах очень маловероятно, чтобы в одном блоке были переходы обоих видов, и поэтому почти все ошибки приводят к изменению веса блока, и, следовательно, обнаруживаются. Другим примером несистематического кода является код с контрольным суммированием – итеративный код. В этом коде проверочные разряды формируются в результате суммирования значений разрядов как в данной кодовой комбинации, так и одноименных разрядов в ряде соседних с ней комбинаций, образующих совместный блок. Итеративные коды позволяют получить так называемые мощные коды, т. е. коды с длинными блоками и большим кодовым расстоянием при сравнительно простой процедуре декодирования. Итеративные коды могут строиться как комбинационные посредством произведения двух или более систематических кодов.

К комбинационным кодам можно отнести также антифединговые коды, предназначенные для обнаружения и исправления ошибок в каналах с замираниями (федингом) сигналов. Для таких каналов с группированием ошибок применяют метод перемежения символов или декорреляции ошибок. Он заключается в том, что символы, входящие в одну кодовую комбинацию, передаются не непосредственно друг за другом, а перемежаются символами других кодовых комбинаций исходного систематического или любого другого кода. Если интервал между символами, входящими в одну кодовую комбинацию, сделать длиннее «памяти» (интервала корреляции) канала с замираниями, то в пределах длительности одной исходной кодовой комбинации группирования ошибок не будет. На приеме после обратной «расфасовки» в кодовых комбинациях можно производить декодирование с обнаружением и исправлением ошибок.

В систематических кодах различают два метода формирования проверочной группы символов: поэлементный и в целом. Наиболее известны среди систематических кодов коды Хемминга, которые исторически были найдены раньше многих других кодов и сыграли большую роль в развитии теории корректирующих кодов. В этих кодах используется принцип проверки на четность определенного ряда информационных символов. Проверочная группа из m символов формируется поэлементно по соответствующему алгоритму. Коды Хемминга, имеющие $d_{\min} = 3$, позволяют исправить одиночные ошибки.

Расширенные коды Хемминга строятся в результате дополнения кодов с $d_{\min} = 3$ общей проверкой каждой из кодовых комбинаций на четность, т. е. еще одним проверочным символом. Это позволяет увеличить минимальное кодовое расстояние до $d_{\min} = 4$.

Циклические коды также относятся к классу линейных систематических кодов и обладают всеми их свойствами. Коды названы циклическими потому, что циклический сдвиг любой разрешенной кодовой комбинации также является разрешенной комбинацией. Теория построения циклических кодов базируется на разделах высшей алгебры, изучающей свойства двоичных многочленов. Особую роль в этой теории играют так называемые неприводимые многочлены, т. е. полиномы, которые не могут быть представлены в виде произведения многочленов

низших степеней. В связи с этим циклические коды относят к разновидности полиномиальных кодов.

Среди циклических кодов особое место занимает класс кодов, предложенных Боузом и Рой-Чоудхури и независимо от них Хоквингемом. Коды Боуза-Чоудхури-Хоквингема получили сокращенное наименование БЧХ-коды и отличаются специальным выбором порождающего (образующего) циклический код полинома, что приводит к простой процедуре декодирования.

В циклических кодах m проверочных символов, добавляемых к исходным k информационным символам, могут быть получены сразу в результате умножения исходной подлежащей передаче кодовой комбинации $Q(x)$ простого кода на одночлен x^m и добавлением к этому произведению остатка $R(x)$, полученного в результате деления произведения на порождающий полином $P(x)$. Отметим, что коды Хемминга также можно получить по алгоритмам формирования циклических кодов.

Проблема помехоустойчивого кодирования представляет собой обширную область теоретических и прикладных исследований. Основными задачами при этом являются:

- отыскание кодов, эффективно исправляющих ошибки требуемого вида;
- нахождение методов кодирования и декодирования;
- нахождение простых способов программной и аппаратной реализации кодов.

Наиболее разработаны эти задачи применительно к систематическим кодам. Такие коды успешно применяются в вычислительной технике, различных автоматизированных цифровых устройствах и цифровых системах передачи информации.

1.3.3. ОСНОВНЫЕ ХАРАКТЕРИСТИКИ КОРРЕКТИРУЮЩИХ КОДОВ

В настоящее время наибольшее внимание с точки зрения технических приложений уделяется двоичным блочным корректирующим кодам. При использовании блочных кодов цифровая информация передается в виде отдельных кодовых комбинаций (блоков) равной длины. Кодирование и декодирование каждого блока осуществляется независимо друг от друга.

Почти все блочные коды относятся к разделимым кодам, кодовые комбинации которых состоят из двух частей: информационной и проверочной. При общем числе n символов в блоке число информационных символов равно k, а число проверочных символов:

$$m = n - k. \tag{1.3}$$

К основным характеристикам корректирующих кодов относятся:

- число разрешенных и запрещенных кодовых комбинаций;
- избыточность кода;
- минимальное кодовое расстояние;
- число обнаруживаемых или исправляемых ошибок;
- корректирующие возможности кодов.

Для блочных двоичных кодов, с числом символов в блоках, равным n, общее число возможных кодовых комбинаций определяется значением

$$N_n = 2^n. \tag{1.4}$$

Число разрешенных кодовых комбинаций при наличии k информационных разрядов в первичном коде:

$$N_k = 2^k. \tag{1.5}$$

Очевидно, что **число запрещенных комбинаций**:

$$N_m = N_n - N_k = 2^n - 2^k, \tag{1.6}$$

а с учетом (1.3) отношение будет

$$N_n / N_k = 2^n / 2^k = 2^{n-k} = 2^m, \tag{1.7}$$

где m – число избыточных (проверочных) разрядов в блочном коде.

Избыточностью корректирующего кода называют величину

$$c = \frac{m}{n} = \frac{n-k}{n} = 1 - \frac{k}{n}, \tag{1.8}$$

откуда следует:

$$B_k = \frac{k}{n} = 1 - c. \tag{1.9}$$

Эта величина показывает, какую часть общего числа символов кодовой комбинации составляют информационные символы. В теории кодирования величину B_k называют относительной скоростью кода. Если производительность источника информации равна H символов в секунду, то скорость передачи после кодирования этой информации будет

$$B = H\frac{k}{n}, \tag{1.10}$$

поскольку в закодированной последовательности из каждых n символов только k символов являются информационными.

Если число ошибок, которые нужно обнаружить или исправить, значительно, то необходимо иметь код с большим числом проверочных символов. Чтобы при этом скорость передачи оставалась достаточно высокой, необходимо в каждом кодовом блоке одновременно увеличивать как общее число символов, так и число информационных символов.

При этом длительность передачи кодовых блоков будет существенно возрастать, что приведет к задержке информации при ее передаче и приеме. Чем сложнее кодирование, тем длительнее временная задержка информации. Одним из путей ускорения кодирования может быть использование параллельных вычислений или использование конвейерных операций.

Минимальное кодовое расстояние – d_{min}. Для того чтобы можно было обнаружить и исправлять ошибки, разрешенная комбинация должна как можно больше отличаться от запрещенной. Если ошибки в канале связи действуют независимо, то вероятность преобразования одной кодовой комбинации в другую будет тем меньше, чем большим числом символов они различаются.

Если интерпретировать кодовые комбинации как точки в пространстве, то отличие выражается в близости этих точек, т. е. в расстоянии между ними.

Количество разрядов (символов), которыми отличаются две кодовые комбинации, можно принять за кодовое расстояние между ними. Для определения этого расстояния нужно сложить две кодовые комбинации *«по модулю 2»* и подсчитать число единиц в полученной сумме. Например, две кодовые комбинации x_i = 01011 и x_j = 10010 имеют расстояние $d(x_i, x_j)$, равное 3, так как:

$$\begin{array}{l} x_i = 01011 \rightarrow W = 3 \\ \oplus \\ x_j = 10010 \rightarrow W = 2 \\ \hline x_i \oplus x_j = 11001 \rightarrow d(x_i, x_j) = 3 \end{array} \qquad (1.11)$$

Здесь под операцией ⊕ понимается сложение *«по модулю 2»*.

Заметим, что кодовое расстояние $d(x_i,x_0)$ между комбинацией x_i и нулевой x_0 = 00...0 называют весом W комбинации x_i, т.е. вес x_i равен числу «1» в ней.

Расстояние между различными комбинациями некоторого конкретного кода могут существенно отличаться. Так, в частности, в безызбыточном первичном натуральном коде $n = k$ это расстояние для различных комбинаций может изменяться от единицы до величины n, равной разрядности кода. Особую важность для характеристики корректирующих свойств кода имеет минимальное кодовое расстояние d_{min}, определяемое при попарном сравнении всех кодовых комбинаций, которое называют расстоянием Хемминга. Это расстояние принято считать кодовым расстоянием всего пространства разрешенных кодовых слов.

В безызбыточном коде все комбинации являются разрешенными и его минимальное кодовое расстояние равно единице – d_{min}=1. Поэтому достаточно исказиться одному символу, чтобы вместо переданной комбинации была принята другая разрешенная комбинация. Чтобы код обладал корректирующими свойствами, необходимо ввести в него некоторую избыточность, которая обеспечивала бы минимальное расстояние между любыми двумя разрешенными комбинациями не менее двух – $d_{\min} \geq 2.$.

Минимальное кодовое расстояние является важнейшей характеристикой помехоустойчивых кодов, указывающей на гарантируемое число обнаруживаемых или исправляемых заданным кодом ошибок.

Число обнаруживаемых или исправляемых ошибок

При применении двоичных кодов учитывают только дискретные искажения, при которых единица переходит в нуль («1» → «0») или нуль переходит в единицу («0» → «1»). Переход «1» → «0» или «0» → «1» только в одном элементе кодовой комбинации называют единичной ошибкой (единичным искажением). В общем случае под кратностью ошибки подразумевают число позиций кодовой комбинации, на которых под действием помехи одни символы оказались замененными на другие. Возможны двукратные (g = 2) и многократные (g > 2) искажения элементов в кодовой комбинации в пределах $0 \leq g \leq n$.

Минимальное кодовое расстояние является основным параметром, характеризующим корректирующие способности данного кода. Если код используется только для обнаружения ошибок кратностью g_o, то необходимо и достаточно, чтобы минимальное кодовое расстояние было равно $d_{\min} \geq g_o + 1$.

В этом случае никакая комбинация из g_o ошибок не может перевести одну разрешенную кодовую комбинацию в другую разрешенную. Таким образом, условие обнаружения всех ошибок кратностью g_0 можно записать

$$g_o \leq d_{\min} - 1. \qquad (1.12)$$

Чтобы можно было исправить все ошибки кратностью g_u и менее, необходимо иметь минимальное расстояние, удовлетворяющее условию $d_{min} \geq 2g_u$

$$d_{\min} \geq 2g_u + 1. \qquad (1.13)$$

В этом случае любая кодовая комбинация с числом ошибок g_u отличается от каждой разрешенной комбинации не менее чем в g_u+1 позициях. Если условие (1.13) не выполнено, возможен случай, когда ошибки кратности g исказят переданную комбинацию так, что она станет ближе к одной из разрешенных комбинаций, чем к переданной или даже перейдет в другую разрешенную комбинацию. В соответствии с этим, условие исправления всех ошибок кратностью не более g_u можно записать:

$$g_u \le (d_{\min} - 1)/2. \qquad (1.14)$$

Из (1.12) и (1.13) следует, что если код исправляет все ошибки кратностью g_u, то число ошибок, которые он может обнаружить, равно $g_o = 2g_u$. Следует отметить, что соотношения (1.12) и (1.13) устанавливают лишь гарантированное минимальное число обнаруживаемых или исправляемых ошибок при заданном d_{min} и не ограничивают возможность обнаружения ошибок большей кратности. Например, простейший код с проверкой на четность с $d_{min} = 2$ позволяет обнаруживать не только одиночные ошибки, но и любое нечетное число ошибок в пределах $g_o < n$.

Корректирующие возможности кодов

Вопрос о минимально необходимой избыточности, при которой код обладает нужными корректирующими свойствами, является одним из важнейших в теории кодирования. Этот вопрос до сих пор не получил полного решения. В настоящее время получен лишь ряд верхних и нижних оценок (границ), которые устанавливают связь между максимально возможным минимальным расстоянием корректирующего кода и его избыточностью.

1.4. КОРРЕКТИРУЮЩИЕ КОДЫ ХЕММИНГА

Построение кодов Хемминга базируется на принципе проверки на четность веса W (числа единичных символов «1») в информационной группе кодового блока.

Поясним идею проверки на четность на примере простейшего корректирующего кода, который так и называется кодом с проверкой на четность или кодом с проверкой по паритету (равенству).

В таком коде к кодовым комбинациям безызбыточного первичного двоичного k-разрядного кода добавляется один дополнительный разряд (символ проверки на четность, называемый проверочным, или контрольным). Если число символов «1» исходной кодовой комбинации четное, то в дополнительном разряде формируют контрольный символ «0», а если число символов «1» нечетное, то в дополнительном разряде формируют символ «1». В результате общее число символов «1» в любой передаваемой кодовой комбинации всегда будет четным.

Таким образом, правило формирования проверочного символа сводится к следующему:

$$m_i = i_1 \oplus i_2 \oplus \ldots \oplus i_k,$$

где i – соответствующий информационный символ («0» или «1»); k – общее их число а, под операцией $\oplus$ здесь и далее понимается сложение *«по модулю 2»*. Очевидно, что добавление дополнительного разряда увеличивает общее число возможных комбинаций вдвое по сравнению с числом комбинаций исходного первичного кода, а условие четности разделяет все комбинации на разрешенные и неразрешенные. Код с проверкой на четность позволяет обнаруживать одиночную ошибку при приеме кодовой комбинации, так как такая ошибка нарушает условие четности, переводя разрешенную комбинацию в запрещенную.

Критерием правильности принятой комбинации является равенство нулю результата S суммирования *«по модулю 2»* всех n символов кода, включая проверочный символ m_1. При наличии одиночной ошибки S принимает значение 1:

$$S = m_1 \oplus i_1 \oplus i_2 \oplus ... \oplus i_k = 0 \qquad \text{– ошибок нет,}$$

$$S = m_1 \oplus i_1 \oplus i_2 \oplus ... \oplus i_k = 1 \quad \text{– однократная ошибка.}$$

Этот код является *(k+1,k)*-кодом, или (*n,n–1*)-кодом. Минимальное расстояние кода равно двум (d_{min} = 2), и, следовательно, никакие ошибки не могут быть исправлены. Простой код с проверкой на четность может использоваться только для обнаружения (но не исправления) однократных ошибок.

Увеличивая число дополнительных проверочных разрядов, и формируя по определенным правилам проверочные символы m, равные «0» или «1», можно усилить корректирующие свойства кода так, чтобы он позволял не только обнаруживать, но и исправлять ошибки. На этом и основано построение кодов Хемминга.

Коды Хемминга позволяют исправлять одиночную ошибку, с помощью непосредственного описания. Для каждого числа проверочных символов m =3, 4, 5… существует классический код Хемминга с маркировкой

$$(n, k) = (2^m - 1, 2^m - 1 - m), \qquad (1.15)$$

т.е. (7,4), (15,11) (31,26) …

При других значениях числа информационных символов k получаются так называемые усеченные (укороченные) коды Хемминга. Так для кода имеющего 5 информационных символов, потребуется использование корректирующего кода (9,5), являющегося усеченным от классического кода Хемминга (15,11), так как число символов в этом коде уменьшается (укорачивается) на 6.

Для примера рассмотрим классический код Хемминга (7,4), который можно сформировать и описать с помощью кодера, представленного на рис. 1.4. В простейшем варианте при заданных четырех информационных символах: i_1, i_2, i_3, i_4 (k = 4), будем полагать, что они сгруппированы в начале кодового слова, хотя это и не обязательно. Дополним эти информационные символы тремя проверочными символами (m = 3), задавая их следующими равенствами проверки на четность, которые определяются соответствующими алгоритмами, где знак $\oplus$ означает сложение *«по модулю 2»*: $r_1 = i_1 \oplus i_2 \oplus i_3$, $r_2 = i_2 \oplus i_3 \oplus i_4$, $r_3 = i_1 \oplus i_2 \oplus i_4$.

В соответствии с этим алгоритмом определения значений проверочных символов m_i, в табл. 1.3 выписаны все возможные 16 кодовых слов (7,4)-кода Хемминга.

Таблица 1.3 Кодовые слова (7,4)-кода Хемминга

k = 4				m = 3		
i1	i2	i3	i4	r1	r2	r3
0	0	0	0	0	0	0
0	0	0	1	0	1	1
0	0	1	0	1	1	0
0	0	1	1	1	0	1
0	1	0	0	1	1	1
0	1	0	1	1	0	0
0	1	1	0	0	0	1

0	1	1	1	0	1	0
1	0	0	0	1	0	1
1	0	0	1	1	0	0
1	0	1	0	0	1	1
1	0	1	1	0	0	0
1	1	0	0	0	1	0
1	1	0	1	0	0	1
1	1	1	0	1	0	0
1	1	1	1	1	1	1

На рис.1.3 приведена блок-схема кодера – устройства автоматически кодирующего информационные разряды в кодовые комбинации в соответствии с табл.1.3.

На рис. 1.4 приведена схема декодера для (7,4) – кода Хемминга, на вход которого поступает кодовое слово $V=(i_1', i_2', i_3', i_4', r_1', r_2', r_3')$. Апостроф означает, что любой символ слова может быть искажен помехой в телекоммуникационном канале.

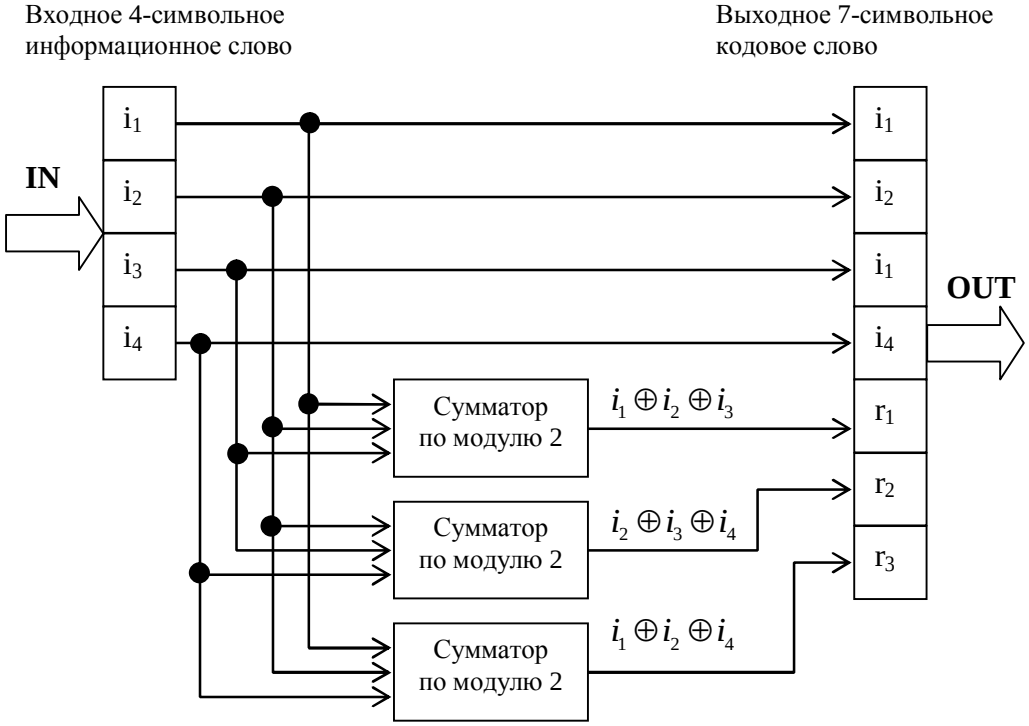


Рис. 1.3. Кодер для (7,4)-кода Хемминга

В декодере в режиме исправления ошибок строится последовательность:

$s_1 = r_1' \oplus i_1' \oplus i_2' \oplus i_3'$, $s_2 = r_2' \oplus i_2' \oplus i_3' \oplus i_4'$, $s_3 = r_3' \oplus i_1' \oplus i_2' \oplus i_4'$.

Трехсимвольная последовательность *(s1, s2, s3)* называется синдромом. Термин «синдром» используется и в медицине, где он обозначает сочетание признаков, характерных для определенного заболевания. В данном случае синдром *S = (s1, s2, s3)* представляет собой сочетание результатов проверки на четность

соответствующих символов кодовой группы и характеризует определенную конфигурацию ошибок (шумовой вектор).

Число возможных синдромов определяется выражением:

$$S = 2^m. \qquad (1.16)$$

При числе проверочных символов m =3 имеется восемь возможных синдромов (2^3 = 8). Нулевой синдром (000) указывает на то, что ошибки при приеме отсутствуют или не обнаружены. Всякому ненулевому синдрому соответствует определенная конфигурация ошибок, которая и исправляется. Классические коды Хемминга (1.15) имеют число синдромов, точно равное их необходимому числу (что позволяет исправить все однократные ошибки в любом информативном и проверочном символах) и включают один нулевой синдром. Такие коды называются плотноупакованными.

Усеченные коды являются неплотноупакованными, так как число синдромов у них превышает необходимое. Так, в коде (9,5) при четырех проверочных символах число синдромов будет равно 2^4 =16, в то время как необходимо всего 10. Лишние 6 синдромов свидетельствуют о неполной упаковке кода (9,5).

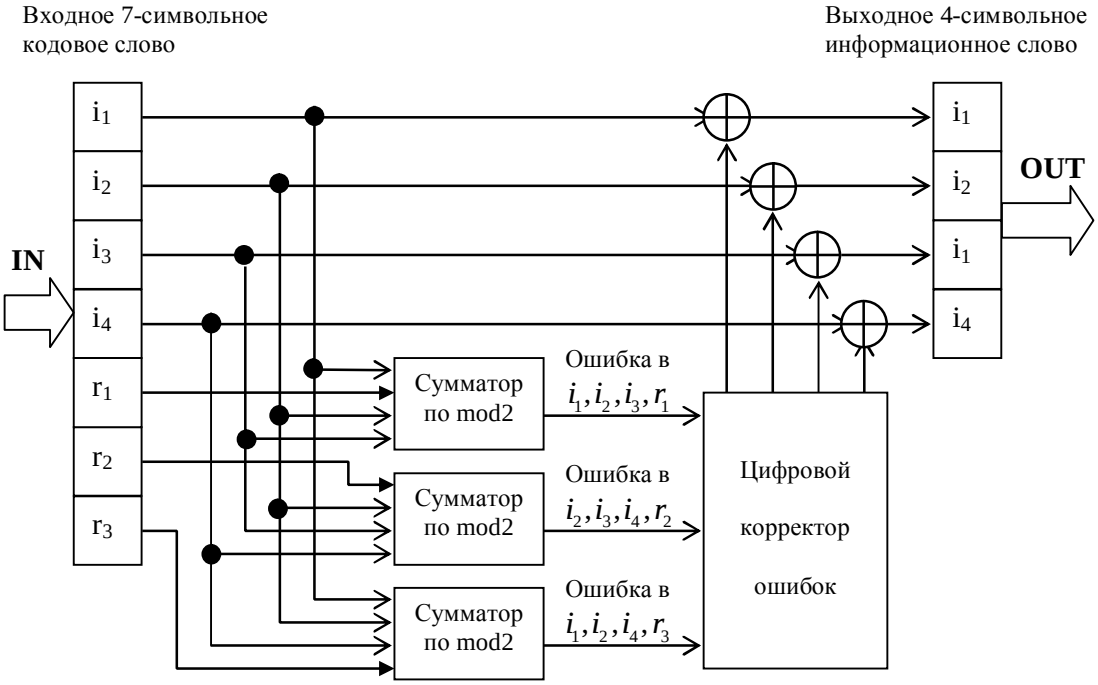


Рис. 1.4. Декодер для (7, 4)-кода Хемминга

Для рассматриваемого кода (7,4) в табл. 1.4 представлены ненулевые синдромы и соответствующие конфигурации ошибок.

Таблица 1.4. Синдромы (7, 4)-кода Хемминга

Синдром	001	010	011	100	101	110	111
Конфигурация ошибок	0000001	0000010	0001000	0000100	1000000	0010000	0100000
Ошибка в символе	m_1	m_2	i_4	m_1	i_1	i_3	i_2

Таким образом, (7,4)-код позволяет исправить все одиночные ошибки. Простая проверка показывает, что каждая из ошибок имеет свой единственный синдром. При этом возможно создание такого цифрового корректора ошибок (дешифратора синдрома), который по соответствующему синдрому исправляет соответствующий символ в принятой кодовой группе. После внесения исправления проверочные символы r_i можно на выход декодера (рис. 1.4) не выводить. Две или более ошибок превышают возможности корректирующего кода Хемминга, и декодер будет ошибаться. Это означает, что он будет вносить неправильные исправления и выдавать искаженные информационные символы.

Идея построения подобного корректирующего кода, естественно, не меняется при перестановке позиций символов в кодовых словах. Все такие варианты также называются (7,4)-кодами Хемминга.

1.5. ЦИКЛИЧЕСКИЕ КОДЫ

Циклические коды составляют большую группу наиболее широко используемых на практике линейных, систематических кодов. Их основное свойство, давшее им название, состоит в том, что каждый вектор, получаемый из исходного кодового вектора путем циклической перестановки его символов, также является разрешенным кодовым вектором. Принято описывать циклические коды (ЦК) при помощи порождающих полиномов $G(X)$ степени $m = n - k$, где m – число проверочных символов в кодовом слове. В связи с этим ЦК относятся к разновидности полиномиальных кодов.

Операции кодирования и декодирования ЦК сводятся к известным процедурам умножения и деления полиномов. Для двоичных кодов эти операции легко реализуются технически с помощью линейных переключательных схем (ЛПС), при этом получаются относительно простые схемы кодеков, в чем состоит одно из практических достоинств ЦК.

Среди циклических кодов особое место занимает класс кодов, предложенных Боузом и Чоудхури и независимо от них Хоквингемом. Коды Боуза-Чоудхури-Хоквингема получили сокращенное наименование БЧХ-коды. БЧХ-коды являются обобщением кодов Хемминга на случай исправления нескольких независимых ошибок ($g_u > 1$). Частными случаями БЧХ-кодов являются коды Файра, предназначенные для обнаружения и исправления серийных ошибок («пачек» ошибок), код Голея – код, исправляющий одиночные, двойные и тройные ошибки ($d_{min}= 7$), коды Рида–Соломона (РС-коды), у которых символами кода являются многоразрядные двоичные числа.

1.5.1. ПОЛИНОМИНАЛЬНОЕ ОПРЕДЕЛЕНИЕ ЦИКЛИЧЕСКИХ КОДОВ И ОПЕРАЦИИ С НИМИ

Циклические коды являются частным случаем систематических, линейных (n, k)-кодов. Название ЦК получили из-за своего основного свойства: циклическая перестановка символов разрешенной кодовой комбинации дает также разрешенную кодовую комбинацию. При циклической перестановке символы кодового слова перемещаются слева направо на одну позицию, причем крайний справа символ переносится на место крайнего левого.

Если, например A_1 – 101100, то разрешенной кодовой комбинацией будет и A_2 – 010110, полученная циклической перестановкой. Отметим, что перестановка производится вместе с проверочными символами, и по правилам линейных кодов

сумма «по модулю 2» разрешенных кодовых комбинаций дает также очередную разрешенную кодовую комбинацию.

Описание ЦК связано с представлением кодовых комбинаций в виде полиномов (многочленов) фиктивной переменной X. Для примера переведем кодовое слово $A_1 = 101100$ в полиномиальный вид

i	6	5	4	3	2	1
Код	1	0	1	1	0	0

При этом $A_1(X)=1X^5+0X^4+1X^3+1X^2+0X^1+0X^0 = X^5+X^3+X^2$ (A_1=101100).

Сдвиг влево на один разряд дает: $1X^6+0X^5+1X^4+1X^3+0X^2+0X^1+\_$ (101100_). Для получения циклического сдвига, нужно левое слагаемое перенести в крайнюю правую позицию: $0X^5+1X^4+1X^3+0X^2+0X^1+1X^6$ (011001) и принять $X^6 \equiv X^0$. Тогда $A2(X)= 0X^5+1X^4+1X^3+0X^2+0X^1+1X^0$ (A_1=011001). Откуда следует:

$$X^6 = X^n = X^0 = 1.$$

Действия с кодовыми векторами, представленными в виде полиномов, производятся по правилам арифметики «по модулю 2», в которой вычитание равносильно сложению. Действительно, прибавив к левой и правой частям по единице, имеем $X^n + 1 = 1 \oplus 1 = 0$ или $X^n = -1$.

Таким образом, вместо двучлена $X^n - 1$ можно ввести бином $X^n + 1$ или $1 + X^n$, из чего следует, что $X^k \oplus X^k = X^k (1 \oplus 1) = 0$ и при последующих операциях с полиномами необходимо вычеркивать пары фиктивных переменных X с одинаковыми степенями.

Приведем далее порядок суммирования (вычитания), умножения и деления полиномов с учетом того, что операция суммирования осуществляется *«по модулю 2»*. В примерах используем вышеприведенные кодовые комбинации $A_1(X)=X^5+X^3+X^2$ и $A_2(X)=X^4+X^2+X$.

Суммирование (вычитание):

$A_1+A_2 = A_1-A_2 = X^5+X^4+X^3+X^2+X^2+X = X^5+X^4+X^3+X$

или

$$\begin{array}{lll} A_1 & 101100 & \\ & \oplus & \\ A_2 & 010110 & \\ \hline & 111010 & = X^5 + X^4 + X^3 + X. \end{array}$$

Умножение:

$A_1 \times A_2 = (X^5+X^3+X^2) \times (X^4+X^2+X) =$

$X^9+X^7+X^6+X^7+X^5+X^4+X^6+X^4+X^3 = X^9+X^5+X^3 = 1000101000.$

Деление: $\frac{A_1}{A_2}$

$$\begin{array}{l|l} X^5 + X^3 + X^2 & X^4 + X^2 + X \\ \hline X^5 + X^3 + X^2 & X \\ \hline 0 \quad 0 \quad 0 & \end{array}$$

- остаток при делении $R(X) = 0$

Из последнего примера следует, что циклический сдвиг полинома вправо на один разряд эквивалентен делению его на *X*, а циклический сдвиг влево на один разряд – эквивалентен умножению полинома на *X*.

1.5.2. ПОРОЖДАЮЩИЕ ПОЛИНОМЫ ЦИКЛИЧЕСКИХ КОДОВ

Формирование разрешенных кодовых комбинаций ЦК $B_i(X)$ основано на предварительном выборе так называемого порождающего (образующего) полинома $G(X)$, который обладает важным отличительным признаком: все комбинации $B_i(X)$ делятся на порождающий полином $G(X)$ без остатка, т. е.

$$\frac{B_i(X)}{G(X)} = A_i(X) \qquad \text{(при остатке } R(X) = 0\text{)}, \tag{1.17}$$

где $A_i(X)$ — информативный полином (кодовая комбинация первичного кода, преобразуемого в корректирующий ЦК).

Поскольку, как отмечалось выше, ЦК относятся к классу блочных разделимых кодов, у которых при общем числе символов n число информационных символов в $A_i(X)$ равно k, то степень порождающего полинома определяет число проверочных символов $m = n - k$.

Из этого свойства следует сравнительно простой способ формирования разрешенных кодовых комбинаций ЦК – умножение комбинаций информационного кода $A_i(X)$ на порождающий полином $G(X)$:

$$B_i(X) = A_i(X)G(X). \tag{1.18}$$

В теории циклических кодов доказывается, что порождающими могут быть только такие полиномы, которые являются делителями двучлена (бинома) $X^n + 1$:

$$\frac{X^n + 1}{G(X)} = H(X) \qquad \text{(при остатке } R(X) = 0\text{)}. \tag{1.19}$$

Возможные порождающие полиномы, найденные с помощью ЭВМ, сведены в таблицы. Некоторые из порождающих полиномов приведены в табл. 1.5. Так, $G(X)$ приведены с записью полиномов в восьмеричной системе счисления (*mod 8*). В этом случае весовые коэффициенты k_i представляют три двоичных знака в соответствии со следующим кодом: 0-000 1-001 2-010 3-011 4-100 5-101 6-110 7-111.

Двоичные символы являются весовыми коэффициентами порождающих полиномов, коэффициенты восьмеричной системы счисления расположены слева от них с учетом того, что $0 \le k_i \le 7$ (при *mod 8*).

Например, 3425 обозначает многочлен 10-й степени, В двоичной записи числу 3425 (*mod 8*) эквивалентно число 011 100 010 101 и соответствующий многочлен равен $X^{10} + X^9 + X^8 + X^4 + X^2 + 1$. Как видно из этого примера, восьмеричная система счисления для записи многочленов выбрана, в частности, из соображений экономии длины записи в три раза при больших объемах табулированных значений, что подчеркивает известный недостаток двоичной системы счисления.

Таблица 1.5. Порождающие полиномы циклических кодов

m-степень полинома $G(X)$	Порождающий полином $G(X)$	Запись полинома по *mod2*	Запись полинома по *mod 8*	n	k	Примечание
1	X+1	11	3	3	2	Код с проверкой на четность (КПЧ)
2	X^2+X+1	111	7	3	1	Код с повто-

						рением
3	X^3+X^2+1 X^3+X+1	1101	13 15	7 7	4 4	Классический код Хемминга
4	X^4+X^3+1 X^4+X+1 X^4+X^2+X+1 $X^4+X^3+X^2+1$	11001 10011 10111 11101	31 23 27 35	15 15 7 7	11 11 3 3	Классический код Хемминга Коды Файра-Абрамсона
5	X^5+X^2+1 X^5+X^3+1 ……	100101 101001	45 51	31 31	26 26	Классический код Хемминга
6	$X^6+X^5+X^4+X^3+X^2+X+1$ ……..	1111111	177	7	1	Код с повторением

Следует отметить, что с увеличением максимальной степени порождающих полиномов m резко увеличивается их количество. Так, при m = 3 имеется всего два полинома, а при m = 10 их уже несколько десятков.

Первый порождающий полином минимальной степени m = 1, удовлетворяющий условию (1.19), формирует код с проверкой на четность (КПЧ) при двух информативных символах и одном проверочном, обеспечивающем обнаружение однократной ошибки, поскольку минимальное кодовое расстояние $d_{min}= 2$. В общем случае коэффициент избыточности КПЧ минимален:

$$k = \frac{m}{n} = \frac{1}{n}, \tag{1.20}$$

а относительная скорость кода – максимальна и будет

$$B_k = \frac{k}{n} = \frac{n-1}{n}, \tag{1.21}$$

в связи с этим КПЧ иногда называют быстрым кодом.

Второй порождающий полином степени m = 2, являющийся «партнером» первого $G(X) = X + 1$ при разложении бинома с n = 3, определяет код с повторением единственного информативного символа k = 1 («0» или «1»).

Отметим, что ЦК принадлежат к классу линейных кодов, у которых кодовые комбинации «000 ... 00» и «111 ... 11» являются разрешенными.

У кода с повторением возможности обнаружения и исправления ошибок безграничны, поскольку число повторений l определяет минимальное кодовое расстояние:

$$d_{\min} = l. \tag{1.22}$$

В общем случае коэффициент избыточности кодов с повторением кодовых комбинаций является максимально возможным: $k = \frac{nl-n}{nl} = \frac{l-1}{l} = 1-\frac{1}{l}$, и при увеличении l приближается к 1, а скорость (1.21) – минимальна

$$B_k = 1-k = \frac{1}{l}. \tag{1.23}$$

Таким образом, коды с проверкой на четность и коды с повторением – до некоторой степени антиподы. Первый код очень быстр (всего один дополнительный символ), но зачастую «легкомыслен». Возможности второго кода с повторением по исправлению ошибок теоретически безграничны, но он крайне «медлителен».

Следующие порождающие полиномы в табл. 1.5 со степенью m = 3 позволяют сформировать набор классических корректирующих (7,4)-кодов Хемминга. Коды Хемминга также принадлежат к классу ЦК, однако при этом группа проверочных символов кода получается сразу «в целом» при делении информативной кодовой группы на порождающий полином, а не «поэлементно», когда последовательное суммирование по модулю 2 соответствующих информативных символов давало очередной символ проверочной группы. Отметим, что два варианта порождающих полиномов кода Хемминга (7,4), с записью по модулю 2 в виде 1101 и 1011, представляют собой, так называемые двойственные многочлены (полиномы).

Двойственные многочлены определяются следующим образом: если задан полином в виде $h(X) = h_0 + h_1X + h_2X^2 + \ldots + h_mX^m$, то двойственным к нему полиномом, т. е. весовые коэффициенты исходного полинома, зачитываемые слева направо, становятся весовыми коэффициентами двойственного полинома при считывании их справа налево. Говоря образно, набор весовых коэффициентов «вывертывается наизнанку». Следует обратить внимание на то, что в полных таблицах порождающих ЦК полиномов двойственные полиномы, как правило, не приводятся.

Наряду с тем, что порождающие полиномы кода Хемминга (7,4) являются двойственными друг другу, они также являются неприводимыми. Неприводимые полиномы не делятся ни на какой другой полином степени меньше r, поэтому их называют еще неразложимыми, простыми и примитивными.

Далее в табл. 1.5 при значениях m = 4 и 5 попадают следующие классические коды Хемминга (15, 11) и (31, 26). Порождающие их полиномы также являются двойственными друг к другу и неприводимыми. Напомним, что к классическим кодам Хемминга относятся коды, у которых n = 2m – 1, а k = 2m – 1 – m, с минимальным кодовым расстоянием d_{min}= 3, позволяющим исправлять однократные ошибки и обнаруживать двойные.

При значениях m = 4 в табл. 1.5 попадают порождающие многочлены кода Абрамсона (7,3), являющиеся частным случаем кодов Файра, порождающие полиномы для которых имеют вид

$$G(X) = p(X)(X^c + 1), \qquad (1.24)$$

где p (X) – неприводимый полином.

Коды Абрамсона совпадают с кодами Файра, если положить c = 1. Число проверочных символов m= 4 определяет общее число символов в коде (разрядность кода), поскольку для этих кодов $n = 2^{m-1} - 1$. Эти коды исправляют все одиночные и смежные двойные ошибки (т. е. серии длиной 2). Помещенные в табл. 1.5 коды Абрамсона (7,3) являются первыми циклическими кодами, исправляющими серийные ошибки (пакеты ошибок). В этом применении ЦК оказываются особенно эффективными. Обратим внимание на то, что при c = 1 порождающими полиномами p (X) являются двойственные полиномы $X^3 + X^2 + 1$ и $X^3 + X + 1$, образующие код Хемминга (7,4) при m= 3.

Серийные ошибки возникают в результате воздействия в канале телекоммуникаций помех импульсного характера, длительность которых больше

длительности одного символа. При этих условиях ошибки уже не независимы, а возникают пачками, общая длительность которых соответствует длительности помехи.

В заключение на основании данных табл. 1.5 приведем все возможные порождающие полиномы для кодовых комбинаций с числом символов n = 7. В соответствии со свойством (1.17) порождающих полиномов G (X) бином X^7 +1 раскладывается на три неприводимых полинома

$$X^7 + 1 = (X + 1)(X^3 + X^2 + 1)(X^3 + X + 1) = G_1(X) \times G_2(X) \times G_3(X), \quad (1.25)$$

каждый из которых является порождающим для следующих кодов:

$G_1(X) = X + 1$ – код с проверкой на четность, КПЧ (7, 6);

$G_2(X) = X^3 + X^2 + 1$ – первый вариант кода Хемминга (7,4);

$G_3(X) = X^3 + X + 1$ – двойственный $\tilde{G_2}(X)$, , второй вариант кода Хемминга.

1.5.3. ПРИНЦИПЫ ФОРМИРОВАНИЯ И ОБРАБОТКИ РАЗРЕШЕННЫХ КОДОВЫХ КОМБИНАЦИЙ ЦИКЛИЧЕСКИХ КОДОВ

На основании материалов предыдущего раздела можно дать следующее определение циклических кодов (ЦК). Циклические коды составляют множество многочленов $B_i(X)$ степени $n-1$ и менее (до $m = n - k$, где m – число проверочных символов), кратных порождающему (образующему) полиному $G(X)$ степени m, который, в свою очередь, должен быть делителем бинома $X^n + 1$, т. е. остаток после деления бинома на $G(X)$ должен равняться нулю. Учитывая, что ЦК принадлежат к классу линейных, групповых кодов, сформулируем ряд основных свойств, им присущих.

1) Сумма разрешенных кодовых комбинаций ЦК образует разрешенную кодовую комбинацию
$$B_i(X) \oplus B_j(X) = B_k(X). \quad (1.26)$$
2) Поскольку к числу разрешенных кодовых комбинаций ЦК относится нулевая комбинация 000 ... 00, то минимальное кодовое расстояние d_{min} для ЦК определяется минимальным весом разрешенной кодовой комбинации:
$$d_{\min} = W_{\min}. \quad (1.27)$$
3) Циклический код не обнаруживает только такие искаженные помехами кодовые комбинации, которые приводят к появлению на стороне приема других разрешенных комбинаций этого кода из набора N_n.
4) Значения проверочных элементов $m = n - k$ для ЦК могут определяться путем суммирования «по модулю 2» ряда определенных информационных символов кодовой комбинации $A_i(X)$. Например, для кода Хемминга (7,4) с порождающим полиномом $G(X) = X^3 + X + 1$ алгоритм получения проверочных символов будет следующим:
$$r_1 = i_1 \oplus i_2 \oplus i_3, \qquad r_2 = i_2 \oplus i_3 \oplus i_4, \qquad r_3 = i_1 \oplus i_2 \oplus i_4. \quad (1.28)$$

Эта процедура свидетельствует о возможности «поэлементного» получения проверочной группы для каждой кодовой комбинации $A_i(X)$. В соответствии с (1.28) могут строиться кодирующие устройства для ЦК.

5) Умножение полинома на X приводит к сдвигу членов полинома на один разряд влево, а при умножении на X^m, соответственно, на r разрядов влево, с

заменой m младших разрядов полинома «нулями». Умножение полинома на X свидетельствует о том, что при этой процедуре X является «оператором сдвига». Деление полинома на X приводит к соответствующему сдвигу членов полинома вправо с уменьшением показателей членов на 1. Процедура сдвига позволяет к исходной кодовой комбинации $A_i(X)$ после умножения ее на X^m, дописать справа m проверочных символов.

6) Поскольку разрешенные кодовые слова ЦК $B_i(X)$ без остатка делятся на порождающий полином $G(X)$ с получением итога в виде информационной комбинации $A_i(X)$ (1.17), то имеется возможность формировать $B_i(X)$ на стороне передачи (кодирующее устройство) простым методом умножения (1.18).

Два последних свойства ЦК позволяют осуществить построение кодеров ЦК двумя методами: методом умножения и методом деления полиномов. Рассмотрим достоинства и недостатки этих методов с учетом вариантов построения декодеров ЦК, соответствующих этим методам.

Метод умножения позволяет при формировании разрешенных кодовых комбинаций по алгоритму (1.18) использовать любой порождающий полином, лишь бы его максимальная степень была равна числу необходимых проверочных символов m.

Однако этот метод обладает двумя существенными недостатками.

Во-первых, при формировании ЦК методом умножения в полученной комбинации $B_i(X)$ в явном виде не содержатся информационные символы. Код получается неразделимым с «перетасованными» информативными и проверочными символами, что затрудняет его декодирование, так как это приводит к необходимости применять метод максимального правдоподобия в декодирующем устройстве (ДУ).

Метод максимального правдоподобия (ММП) предполагает при исправлении ошибок принимаемую кодовую комбинацию отождествлять с той разрешенной, к которой принятая находится ближе всего. При таком непосредственном способе декодирования в памяти запоминающего устройства (ЗУ) декодера необходимо хранить все разрешенные кодовые комбинации N_n, что требует на стороне приема больших объемов ЗУ и большого времени обработки при декодировании. Эти обстоятельства являются вторым недостатком метода умножения при кодировании ЦК.

Исследования показывают, что хороший циклический корректирующий код с кратностью исправляемых ошибок $g_и \geq 5$ при относительной скорости кода $B_к \geq 0.5$, т. е. коэффициенте избыточности $x \leq 0.5$, должен иметь число информационных символов $k \geq 40$. Это значение и приводит к техническим трудностям при процедуре декодирования по ММП, сводящейся к сравнению принятой кодовой комбинации со всеми N_n разрешенными.

Для примера определим время декодирования $T_{ДК}$ принятой кодовой комбинации, если число информационных символов в ней k = 40 и для сравнения используется ЭВМ со скоростью 10^7 операций в секунду.

Будем полагать, что для сравнения принятой кодовой комбинации с одной из разрешенных достаточно одной операции на ЭВМ. Тогда для проведения $N^n = 2^k = 2^{40}$ сравнений потребуется время декодирования

$$T_{ДК} = \frac{N_n}{V_{ЭВМ}} = \frac{2^{40}}{10^7} = \frac{1.1 \cdot 10_{12}}{10^7} = 1.1 \cdot 10^5 c = \frac{1.1 \cdot 10^5}{3600c} = 30час.$$

Как видно из примера, задача декодирования простым перебором и сравнением непосильна даже для современных ЭВМ.

В соответствии с этим, основным направлением в теории кодирования является поиск таких кодов и алгоритмов их формирования и обработки, для которых не требуется хранение в ЗУ разрешенных кодовых комбинаций. Эти задачи решаются, в частности, при построении кодеров на основе деления полиномов, а при декодировании – на основе синдромного метода декодирования (СМД).

Метод деления полиномов позволяет представить разрешенные к передаче кодовые комбинации в виде разделенных информационных $A_i(X)$ и проверочных $M_i(X)$ символов, т. е. получить блочный код.

Поскольку число проверочных символов равно m, то для компактной их записи в последние младшие разряды кодового слова надо предварительно к $A_i(X)$ справа приписать m «нулей», что эквивалентно умножению $A_i(X)$ на оператор сдвига X^m (см. свойство 5 ЦК).

На практике предпочитают использование метода деления полиномов при построении кодеков, поскольку при этом имеется возможность представить кодовую комбинацию в виде разделенных информационных и проверочных символов:

$$B_i(X) = A_i(X)X^m + R_i(X), \qquad (1.29)$$

где $R_i(X)$ – остаток от деления $A_i(X)X^m / G(X)$.

В алгоритме (1.29) можно выделить три этапа формирования разрешенных кодовых комбинаций в кодирующем устройстве:

1) к комбинации первичного кода $A_i(X)$ дописывается справа m нулей, что эквивалентно умножению $A_i(X)$ на X^m;
2) произведение $A_i(X)\ X^m$ делится на соответствующий порождающий полином $G(X)$ и определяется остаток $M_i(X)$, степень которого не превышает $m - 1$, этот остаток и дает группу проверочных символов;
3) вычисленный остаток присоединяется справа к $A_i(X)\ X^m$.

Синдромный метод декодирования (СМД) предполагает в ДУ принятую кодовую комбинацию поделить на порождающий полином. Если принятая комбинация является разрешенной, т. е. не искажена помехами в канале связи, то остаток от деления будет нулевым. Ненулевой остаток свидетельствует о наличии в принятой кодовой комбинации ошибок, остаток от деления и называется синдромом.

Термин «синдром» заимствован из медицинской практики (от греч. вместе бегущий) и означает сочетание (комплекс) симптомов болезни, характерное для определенного заболевания. В теории кодирования синдром, который также называют опознавателем ошибки, обозначает совокупность признаков, характерных для определенной ошибки. Для исправления ошибки на стороне приема необходимо знать не только факт ее существования, но и ее местонахождение, которое определяется по установленному виду вектора ошибки $z(X)$.

После передачи по каналу с помехами принимается кодовое слово

$$B_i^{'}(X) = B_i(X) + z(X), \qquad (1.30)$$

где $B_i(X)$ – передаваемая кодовая комбинация; $z(X)$ – полином (вектор) ошибки, имеющий степень от 1 до $n-1$.

При декодировании принятое кодовое слово делится на $G(X)$

$$\frac{B_i^{'}(X)}{G(X)} = U_i(X) + S_i(X), \qquad (1.31)$$

где остаток от деления $S_i(X)$ и является синдромом.

Если при делении получается нулевой остаток $S_i(X) = 0$, то выносится решение об отсутствии ошибки $z(X) = 0$. Если остаток (синдром) ненулевой $S_i(X) \neq 0$, то выносится решение о наличии ошибки и определяется шумовой вектор (полином) z(X), а затем – передаваемое кодовое слово, поскольку из (1.30) следует

$$B_i(X) = B_i^{'}(X) + z(X).$$

Всякому ненулевому синдрому соответствует определенное расположение (конфигурация) ошибок. Взаимосвязь между видом синдрома и местоположением ошибочного символа находится довольно просто. Достаточно в любую разрешенную кодовую комбинацию ввести ошибку и выполнить деление на $G(X)$. Полученный остаток (1.31) – синдром и будет указывать на ошибку в этом символе.

1.5.4. ПОСТРОЕНИЕ ПОРОЖДАЮЩИХ И ПРОВЕРОЧНЫХ МАТРИЦ ЦИКЛИЧЕСКИХ КОДОВ

Наряду с полиномиальным способом задания кода, структуру построения кода можно определить с помощью матричного представления. При этом в ряде случаев проще реализуется построение кодирующих и декодирующих устройств ЦК.

Рассмотрим варианты формирования и обработки ЦК, заданных в виде порождающих и проверочных матриц, на конкретном примере ЦК Хемминга (7,4), воспользовавшись выражением (1.25), в котором определены двойственные (дуальные) порождающие полиномы кода:

$$X^7 + 1 = (X+1)(X^3 + X^2 + 1)(X^3 + X + 1) = G_1(X) \times G_2(X) \times G_3(X),$$

что соответствует кодам (7, 6), (7, 4) и (7, 4).

Пример

Задан ЦК(7,4) дуальными порождающими полиномами $G(7,4) = X^3 + X + 1$ и $G^{\sim}(7,4) = X^3 + X^2 + 1$. Составить порождающие матрицы для формирования разрешенных кодовых комбинаций и проверочные матрицы для получения синдромов.

Первой строкой в матрице записывается порождающий полином (в двоичном представлении) с домножением его на оператор сдвига X^m для резервирования места под запись $m = 3$ проверочных символов. Следующие $k-1$ строк матриц получаются путем последовательного циклического сдвига базового кодового слова матрицы G и $G^{\sim}$ на одну позицию вправо, поскольку при этом по определению ЦК также получаются разрешенные к передаче кодовые комбинации:

$$G(7,4) = \left|\begin{array}{ccccccc|c} 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 3 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 4 \end{array}\right| \qquad G^{\sim}(7,4) = \left|\begin{array}{ccccccc|c} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 3 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 4 \end{array}\right| \qquad (1.32)$$

Однако в таком виде эти порождающие матрицы размерностью $k \times n$ – (n столбцов, k строк) могут образовать только неразделимый ЦК, т. е. код, у которого не определены жестко места информационных и проверочных элементов. Для построения порождающей матрицы, формирующей разделимый блочный код, необходимо матрицу преобразовать к каноническому виду путем простых линейных операций над строками, промаркированными № 1–4.

С учетом свойства ЦК (1.26), каноническую форму матрицы можно получить путем сложения ряда разрешенных кодовых комбинаций. Каноническая матрица должна в левой части порождающей ЦК матрицы содержать единичную диагональную квадратную подматрицу Е порядка k для получения в итоге блочного ЦК. С этой целью для получения первой строки канонической матрицы $G_k(7,4)$ необходимо сложить «по модулю 2» строки с номерами 1, 3 и 4 матрицы $G(7,4)$, а для матрицы $G_k^{\sim}(7,4)$ – строки с номерами 1, 2 и 3 матрицы $G^{\sim}(7,4)$. В этом случае в матрицах (1.32) в первых строках остаются «1» только на первых позициях, а остальные «*k–1*» символов заменяются «0». Это и соответствует первым строкам единичных подматриц порядка «k». Нормирование последующих трех строк канонических матриц производится путем соответствующего суммирования строк матриц (1.32).

В итоге имеем следующий вид дуальных канонических матриц:

$$G_k(7,4) = \left|\begin{array}{cccc|ccc|l} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 = 1 \oplus 3 \oplus 4 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 2 = 2 \oplus 4 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 3 = 3 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 4 = 4 \end{array}\right|$$

$$G^{\sim}(7,4) = \left|\begin{array}{cccc|ccc|l} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 = 1 \oplus 2 \oplus 3 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 2 = 2 \oplus 3 \oplus 4 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 3 = 3 \oplus 4 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 4 = 4 \end{array}\right| \qquad (1.33)$$

Процесс кодирования первичных кодов на стороне источника сообщений сводится к умножению информационных посылок, представленных в виде векторов $\vec{A}_i(X)$, на соответствующую порождающую каноническую матрицу:

$$\vec{B}_i(X) = \vec{A}_i(X) G_k. \qquad (1.34)$$

Эта процедура позволяет получить блочные коды Хемминга «в целом», т. е. получить проверочную группу символов m_1, m_2, m_3 сразу после выполнения операции (1.34).

Наряду с этим имеется возможность формировать символы проверочной группы поэлементно:

$$\begin{array}{ll} r_1 = i_1 \oplus i_2 \oplus i_3, & r_1 = i_1 \oplus i_3 \oplus i_4, \\ r_2 = i_2 \oplus i_3 \oplus i_4, & r_2 = i_1 \oplus i_2 \oplus i_3, \\ r_3 = i_1 \oplus i_2 \oplus i_4, & r_3 = i_2 \oplus i_3 \oplus i_4. \end{array} \qquad (1.35)$$

Обратим внимание на то, что алгоритм (1.35) просто получается из рассмотрения порождающих коды Хемминга матриц (1.33), в которых проверочные подматрицы, содержащие 3 столбца (r_1, r_2, r_3), имеют символы «1» в тех строках,

номера которых совпадают с маркировкой информационных символов i в равенствах (1.35).

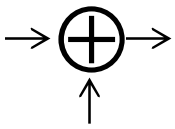
При матричном варианте обработки принятых кодов на стороне получателя сообщений для получения синдрома $\vec{S}$ необходимо принятую, возможно искаженную в канале, кодовую комбинацию $B_i^{'}(X)$ умножить на проверочную матрицу H(X):

$$\vec{S} = B_i^{'}(X)H(X). \qquad (1.36)$$

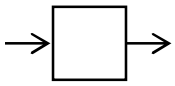
1.5.5. СТРУКТУРНЫЙ СОСТАВ ЛИНЕЙНЫХ ПЕРЕКЛЮЧАТЕЛЬНЫХ СХЕМ

Цикличность перестановок при формировании разрешенных кодовых комбинаций ЦК лежит в основе техники построения кодирующих устройств (КУ) и декодирующих устройств (ДУ) циклических кодов. Эта техника применяет сдвигающие регистры (СР) в виде триггерных цепочек с теми или иными обратными связями. Такие СР называют также многотактными Линейными Переключательными Схемами (ЛПС) и линейными кодовыми фильтрами Хафмена, который первым начал изучение ЛПС с точки зрения линейных фильтров. Кстати, Д. Хафмен является и автором принципа, состоящего в том, что «две точки зрения лучше, чем одна», получившего широкое применение в настоящее ком-промиссное время.

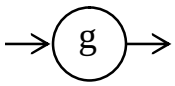
При построении ЛПС используется три вида элементарных устройств:



Сумматор имеет, как правило, два входа и один выход, причем для двоичных кодов суммирование осуществляется «по модулю 2»



Запоминающее Устройство имеет один вход и один выход и представляет собой одну триггерную ячейку (один разряд) СР



Устройство умножения на постоянную величину, имеет один вход и один выход.

Линейными переключательными схемами с конечным числом состояний называются любые схемы, содержащие конечное число сумматоров, устройств памяти и устройств умножения на константу, соединенных любым допустимым способом.

В бинарном случае сумматор (равно как и вычитатель) представляет собой логический элемент «исключающее ИЛИ», а устройство памяти является устройством задержки (D-триггером). Устройства задержки, включенные последовательно, составляют сдвигающий регистр (СР), в ячейках которого выходной символ совпадает с входным символом в предшествующий момент времени. К СР подводится шина сдвига, с помощью которой тактовыми импульсами (ТИ) осуществляется продвижение по разрядам СР записанной кодовой

информации. Как правило, шина сдвига не показывается на схемах с изображениями ЛПС.

При формировании и обработке двоичных ЦК введение в схему ЛПС умножителя на константу, равную 1, эквивалентно введению дополнительного соединения, а умножитель на константу, равную 0, соответствует отсутствию такого соединения.

Предполагается, что на вход СР, входящего в состав ЛПС, кодовая комбинация подается последовательно, с периодичностью, равной периоду следования ТИ в шине сдвига. Аналогично, последовательно во времени, появляются кодовые символы на выходе СР. Когда входом или выходом является многочлен, представляющий при двоичной обработке набор «1» и «0», то на входном или выходном конце СР появляются только коэффициенты («1» или «0»), начиная с коэффициентов высших порядков. Это обусловливается тем, что при делении у делителя сначала должны быть обработаны коэффициенты высших порядков.

1.5.6. УМНОЖЕНИЕ ПОЛИНОМОВ НА БАЗЕ ЛПС

Схема, изображенная на рис. 1.5, используется для умножения любого полинома на входе $A(X) = a_0 + a_1 X + a_2 X^2 + \ldots + a_k X^k$ на фиксированный полином, в частности, порождающий: $G(X) = g_0 + g_1 X + g_2 X2 + \ldots + g_m X^m$.

Предполагается, что первоначально все разряды СР содержат нули, а на вход коэффициенты полинома A(X) поступают, начиная с коэффициентов высших порядков (со старших разрядов), после чего следует m нулей.

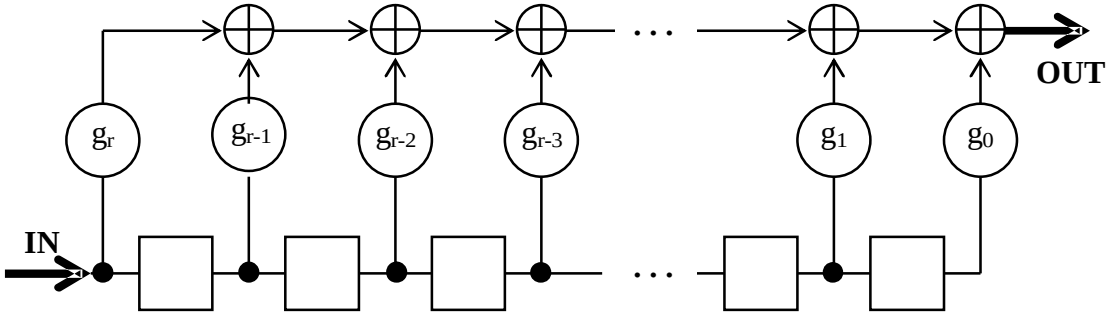


Рис. 1.5. Первый вариант схемы умножителя полиномов

Произведение полиномов

$$A(X)G(X) = a_0 g_0 + (a_o g_1 + a_1 g_0)X + ... + a_k g_m X^{k+m}. \quad (1.37)$$

Когда на входе ЛПС появляется первый (старший) коэффициент полинома A(X), то он умножится в первом устройстве умножения на g_r и появится на выходе уже как результат перемножения $a_k g_r$, проследовав «транзитом» через все схемы суммирования «по модулю 2». Кроме того, a_k запишется в первом разряде СР, а все остальные разряды СР будут содержать нули. Спустя единицу времени, с появлением в шине сдвига 2-го ТИ, на входе появится a_{k-1}, который перемножается с g_m и сложится в первой схеме суммирования «по модулю 2» с $a_k g_{r-1}$, сформировав на выходе сумму $a_{k-1} g_r + a_k g_{r-1}$, т. е. второй коэффициент произведения $A(X)G(X)$. Дальнейшие операции производятся аналогичным образом. После $m + k$ сдвигов СР полностью обнуляется и на выходе появляется значение $a_0 g_0$, равное первому коэффициенту произведения (1.43), так что произведение на выходе ЛПС

последовательно получается в полном составе. Второй вариант ЛПС для умножения полиномов показан на рис. 1.6.

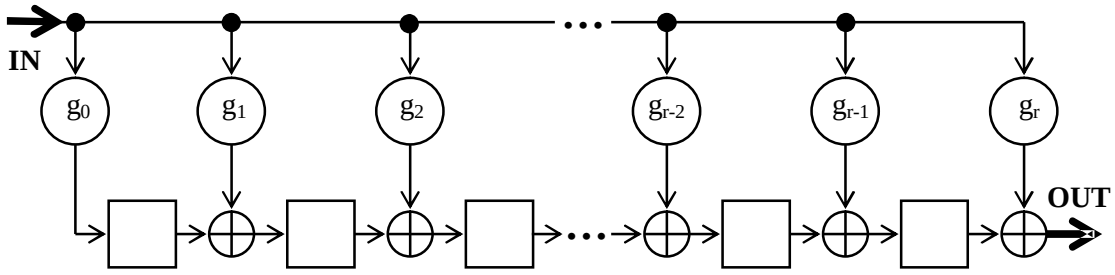


Рис. 1.6. Второй вариант схемы умножителя полиномов

Коэффициенты произведения формируются непосредственно в СР. После того, как первый символ подается на вход, на выходе появляется последний коэффициент (1.37) $a_k g_m$, а разряды СР содержат только нули. После одного сдвига ячейки СР содержат элементы $a_k\ g_0$, $a_k\ g_1$, ..., $a_k\ g_{m-1}$, а вход равен a_{k-1}. При этом выход СР равен $a_k\ g_{m-1} + a_{k-1}\ g_m$, т. е. равен второму коэффициенту (1.37). После появления очередного ТИ в шине сдвига (не показана на рис. 1.5 и 1.6) на выходе появляется третий коэффициент (1.37). Дальнейшие операции производятся аналогичным образом.

Схемы умножения могут иметь более чем один вход, если добавить к ЛПС, изображенной на рис. 1.7, вторую шину с цепочкой устройств умножения, связанных с соответствующими схемами суммирования «по модулю 2». Тогда схема будет реализовывать процедуру суммирования произведений двух пар полиномов

$$C(X) = A_1(X)\ G_1(X) + A_2(X)G_2(X)\ , \qquad (1.38)$$

причем ЗУ в виде СР – только одно.

1.5.7. ДЕЛЕНИЕ ПОЛИНОМОВ НА БАЗЕ ЛПС

Схема для деления полинома $A(X) = a_0 + a_1 X + a_2 X^2 + \ldots + a_k X^k$ на полином $G(X) = g_0 + g_1 X + g_2 X^2 + \ldots + g_m X^m$ показана на рис. 1.8. Динамическое ЗУ в виде СР вначале должно содержать все нули. Для деления полиномов СР охвачен обратной связью, т. е. выход СР соединяется с входом. Для подчеркивания противоположного направления шины обратной связи коэффициент умножителя обозначается как g_{m-1}. Однако для двоичных кодов результат умножения и деления на единицу одинаков, поэтому указанное обозначение в дальнейшем использоваться не будет. Первый вариант ЛПС для деления полиномов (рис. 1.7).

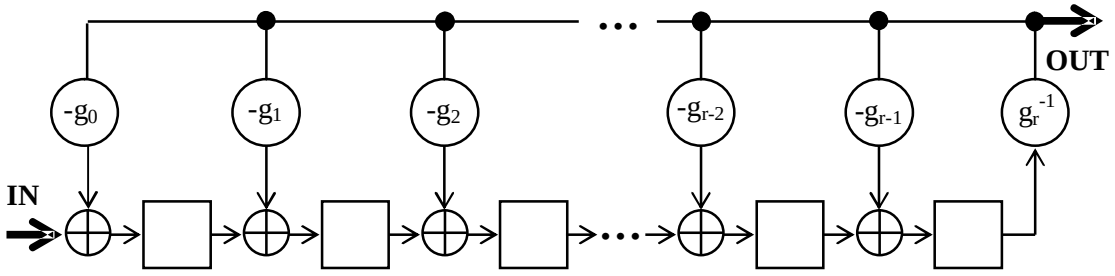


Рис. 1.7. Первый вариант схемы делителя на генераторный полином

Для первых m-сдвигов, т. е. до тех пор, пока первый входной символ не достигнет конца РС, выход принимает значения, равные «0». После этого на выходе появляется первый ненулевой выход, который равен $a_k g_{m-1}$ – первому коэффициенту частного. Для каждого коэффициента частного g_j необходимо вычесть из делимого полином *G(X)*. Это вычитание производится с помощью обратной связи. После k сдвигов на выходе появится частное от деления, а остаток от деления будет находиться в СР.

Работу схемы легче всего понять с помощью примеров построения КУ и ДКУ на базе ЛПС, рассматриваемых далее в разд. 1.5.8. Второй вариант ЛПС с делением на генераторный полином (рис. 1.8).

При построении КУ ЦК, а также генераторов различных кодовых последовательностей, в частности, последовательностей максимальной длины (М-последовательностей), применяется в ряде случаев так называемый генераторный полином H(X). Этот полином называют также проверочным, если он получается при

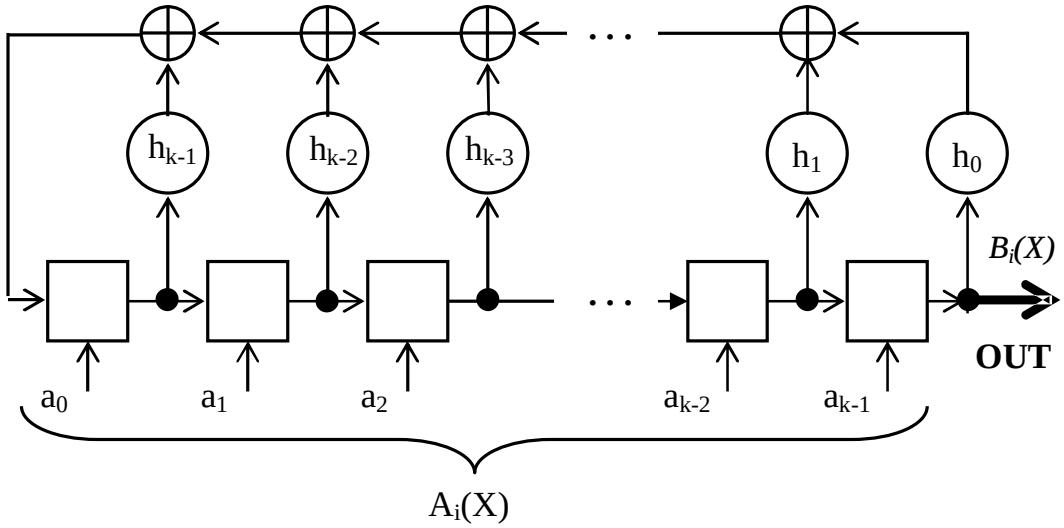


Рис. 1.8. Второй вариант схемы делителя на генераторный полином

делении бинома $1+X^n$ на порождающий полином *G(X):*

$$H(X)=\frac{1+X^n}{G(X)}. \qquad (1.39)$$

При использовании этой схемы в качестве КУ ЦК, исходную кодовую комбинацию A(X) параллельно и одновременно записывают в k разрядов СР. С первым тактом на выход будет выдан коэффициент $b_{n-1} = a_{k-1}$, произойдет сдвиг вправо в СР, и в освободившуюся ячейку памяти будет записано вычисленное значение проверочного бита $m_{n-k-1} = h_0 a_{k-1} + h_1 a_{k-2} + ... + h_{k-1} a_0$. Со вторым тактом на выход будет считан коэффициент $b_{n-2} = a_{k-2}$, произойдет сдвиг, и в освободившуюся первую ячейку СР запишется второй проверочный бит $m_{n-k-2} = h_0 a_{k-2} + h_1 a_{k-3} + ... + h_{k-1} m_{n-k-1}$. Через $n - k$ тактов будут вычислены все $n - k$ проверочных символов $r_0, r_1, \ldots, m_{n-k-1}$ и записаны в СР. После k тактов, т. е. после вывода на выход всех информационных символов, станут выводиться проверочные символы в том же порядке, в каком они вычислялись. На выходе получается блочный код. После k тактов процесс кодирования одной комбинации $A_i(X)$

заканчивается, и СР принимает исходное состояние. Для кодирования следующей комбинации необходимо стереть $A_i(X)$, ввести в СР новую $A_j(X)$ и повторить цикл из n тактов.

Рассмотрим более конкретно работу этой схемы на примере использования ее в качестве КУ с привязкой начальных условий к данным предыдущих примеров.

Пример

Построить схему КУ, обеспечивающего кодирование ЦК Хемминга (7,4) с порождающим полиномом $G(X) = 1 + X + X^2$ путем вычисления блока проверочных символов «в целом», используя проверочный полином H(X). Проследить по тактам процесс кодирования и состояние элементов схемы при кодировании исходной комбинации $1001 \sim 1 + X^3 = A(X)$. Построение схемы КУ определяется проверочным полиномом (1.39)

$$H(X) = \frac{1+X^7}{1+X+X^3} = 1+X+X^2+X^4.$$

Так как k = 4, то число разрядов СР равно четырем. По виду проверочного полинома определяем, что $h_0 = h_1 = h_2 = h_4 = 1$, $h_3 = 0$.

Схема КУ для условий примера показана на рис. 1.9. Состояние ячеек сдвигового регистра СР и выхода схемы по тактам – в табл. 1.6. В исходном положении в триггерные ячейки сдвигового регитсра записываются информационные символы $A_i(X) = 1 + X^3 \sim 1001$. Учитывая наличие обратной связи в СР с выхода на вход, суммирование «по модулю 2» выходов ячеек X^1, X^2 и X^3 даст символ записи в ячейку X^0. После первого сдвига в X^0 будет записан символ проверочной группы r_1, который при последующих сдвигах продефилирует на выход СР. Из табл.1.6 видно, что после n = 7 тактов на выходе образуется комбинация 0111001 (старшим разрядом вперед).

Таблица 1.6. Состояния КУ

Номер такта	Состояния ячеек				Выход
	X^0	X^1	X^2	X^3	
A(X)	1	0	0	1	–
1	1	1	0	0	1
2	1	1	1	0	0
3	0	1	1	1	0
4	1	0	1	1	1
5	0	1	0	1	1
6	0	0	1	0	1
7	1	0	0	1	0

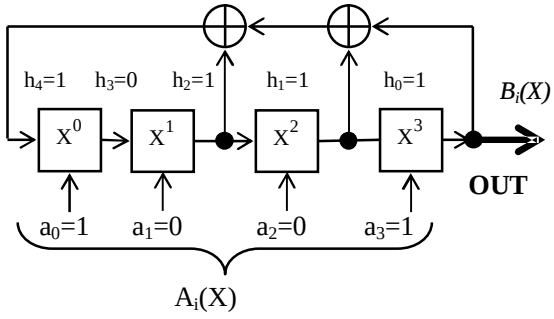


Рис.1.9 Схема кодера Хемминга (7,4)

При этом триггерные ячейки СР принимают исходное значение 1001, и при необходимости возможно повторение процедуры кодирования этой же кодовой комбинации $A_i(X)$ путем подачи очередных следующих $n = 7$ тактов. Таким образом, этот способ кодирования так же, как и первый вариант схемы для деления полиномов, обеспечивает получение кодовых комбинаций разделимого, блочного циклического кода ЦК.

Рассмотрение вариантов построения ЛПС, выполняющих операции умножения и деления полиномов, с целью использования в кодеках ЦК, позволяет сделать следующие выводы:

1) В КУ ЦК процедура умножения полиномов приводит к получению неразделимых кодов, что усложняет их последующее декодирование. Поэтому операция умножения редко используется в устройствах формирования и обработки ЦК.
2) При делении на порождающий полином G(X) код на выходе КУ получается разделимым и СР содержит r разрядов. Так как в большинстве случаев используются ЦК, у которых число проверочных символов r существенно меньше числа информационных $(m < k)$, то СР в этом случае будет иметь меньшее число разрядов, чем при делении на генераторный полином.
3) При делении в КУ исходной кодовой комбинации на генераторный многочлен ЦК также получается разделимым, но в СР требуется использовать не m, а k разрядов, которых, как правило, больше.

1.5.8. КОДИРУЮЩЕЕ И ДЕКОДИРУЮЩЕЕ УСТРОЙСТВО ДЛЯ КОДА ХЕММИНГА (7, 4)

Покажем, как реализуются схемы с учетом того, что коды Хемминга относятся и к классу ЦК.

Кодер для кода Хемминга (7,4). Для построения КУ по классической схеме деления (см. рис. 1.7), так как кодирование путем вычисления остатка «в целом» требует предварительного выполнения операции умножения на оператор сдвига X^m и сложения полинома – остатка с полиномом – произведением $A_i(X)X^m$ (1.29), требуется предварительно видоизменить структуру схемы. Для выполнения операции умножения следует разместить сумматор, на который подключен вход, в конце СР, перед обратной связью g_{m-1}. Такое подключение входа эквивалентно умножению на X^m, так как исключается задержка на m ТИ.

Для выполнения операции сложения остатка $R_i(X)$ с полиномом $A_i(X)X^m$ (1.29) необходимо выход КУ подключить к одному из входов схемы логического сложения (ИЛИ), ко второму входу которой подключается вход схемы для выдачи на выход без задержки информационной кодовой комбинации $A_i(X)$ (старшим разрядом вперед). Подробнее рассмотрим работу схемы на конкретном примере.

Пример

Построить схему КУ, обеспечивающего кодирование ЦК (7,4) с порождающим полиномом $G(X) = 1 + X + X^3$ путем определения проверочной группы методом деления полиномов и определения остатка R(X). Проследить по тактам процесс кодирования и состояние элементов схемы при кодировании исходного полинома

$A_i(X) = 1 + X^3 \leftrightarrow 1001.$

Схема кодера для условий примера изображена на рис. 1.10, состояние ячеек *СР* и выхода схемы по тактам — в табл. 1.7. Наряду с особенностями построения схемы, КУ дополнено двумя ключевыми схемами, роль которых выполняют схемы логического умножения DD_1 и DD_2 соответственно. В течение первых $k = 4$ тактов на второй вход схемы DD_1 поступают ТИ, обеспечивая прохождение символов от выходного сумматора в шину обратной связи *СР*. Начиная с 5-го по 7-й такт, ТИ на второй вход схемы DD_1 не поступают, и обратная связь разрывается. В это время

поступают ТИ на второй вход схемы DD_2, благодаря чему выход *CP* подключается к выходу всего КУ, обеспечивая выдачу остатка от деления кодовой комбинации A_i *(X)* на порождающий полином *G(X)* на выход, для подстыковки проверочных символов к $A_i(X)$.

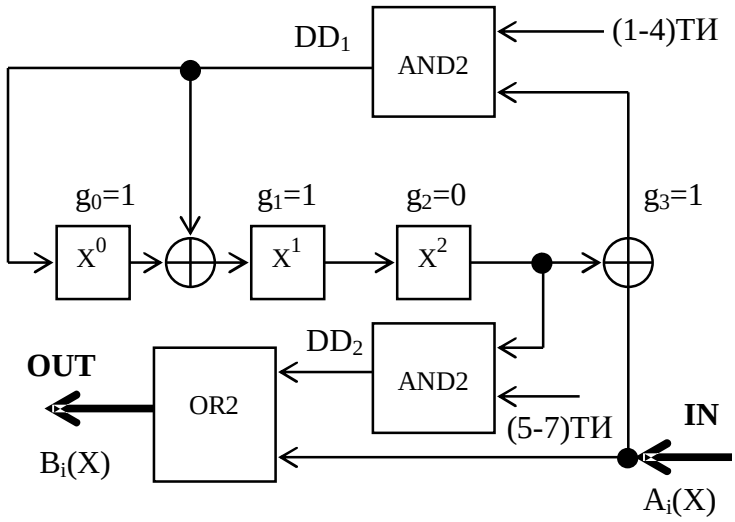


Рис. 1.10. Схема кодера для (7,4)-кода Хемминга.

Из табл.1.7 видно, что после 4-го такта в *CP* образуется остаток 011, т. е. $R(X) = X + X^2$, а в течение n тактов на выход поступает кодовая комбинация $0111001 \sim X + X^2 + X^3 + X^6$ (старшим разрядом вперед).

Декодер для кода Хемминга (7,4). При аппаратурной реализации декодеров ЦК для определения синдрома используют схему, осуществляющую процедуру деления полинома на полином (см. рис. 1.7). При построении ДУ следует дополнительно включать ЗУ на *k* элементов и схему опроса остатка при делении.

Таблица 1.7. Состояния КУ обеспечивающего кодирование ЦК (7,4)

Номер такта	Выход	Состояние					
		ячеек			ключей		
		X^0	X^1	X^2	Выход	DD_1	DD_2
0	–	0	0	0	–	Замкнут	Разомкнут
1	1	1	1	0	1		
2	0	0	1	1	0		
3	0	1	1	1	0		
4	1	0	1	1	1		
5	–	0	0	1	1	Разомкнут	Замкнут
6	–	0	0	0	1		
7	–	0	0	0	0		

Эта схема состоит из схемы логического сложения (*OR*) на m входов и схемы логического умножения (*AND*) на два входа; *CP* и обратные связи должны соответствовать структуре порождающего полинома *G(X)*, т. е. число ячеек *CP*

должно быть равным m, а замкнутая обратная связь должна соответствовать ненулевым коэффициентам полинома *G(X).*

Пример

Построить схему ДУ для ЦК Хемминга (7,4) с порождающим полиномом G(X) = 1 + X + X3 и по тактам сдвигающих импульсов проследить за его работой. Схема ДУ должна решать задачу обнаружения ошибок. На рис. 1.11 приведена схема ДУ, в табл. 1.8 представлены состояния ячеек *СР* при декодировании входной кодовой комбинации $B_i(X) = X + X^2 + X^3 + X^6 \sim 0111001$, принимаемой без ошибок. Декодирующее устройство работает следующим образом.

Таблица 1.8. Состояния ДУ для ЦК Хемминга (7, 4)

Вход B(X)	Номер такта	Состояние ячеек			Выход СР
		X^0	X^1	X^2	
–	Исходное состояние	0	0	0	–
1	1	1	0	0	0
0	2	0	1	0	0
0	3	0	0	1	0
1	4	0	1	0	1
1	5	1	0	1	0
1	6	0	0	0	1
0	7	0	0	0	0

Кодовая комбинация $B_i(X)$ старшим разрядом вперед поступает на СР для определения остатка при делении и в ЗУ на *k* элементов через открытую схему DD_1, которая через k тактов закрывается, так как прекращается подача из синхронизатора ТИ на один из входов схемы DD_1.

При этом в ЗУ запоминаются *k* информационных символов принимаемой кодовой комбинации $B_i(X)$. В *СР* поступают все n элементов $B_i(X)$, и после n тактов происходит опрос состояния ячеек СР путем подачи циклового импульса с

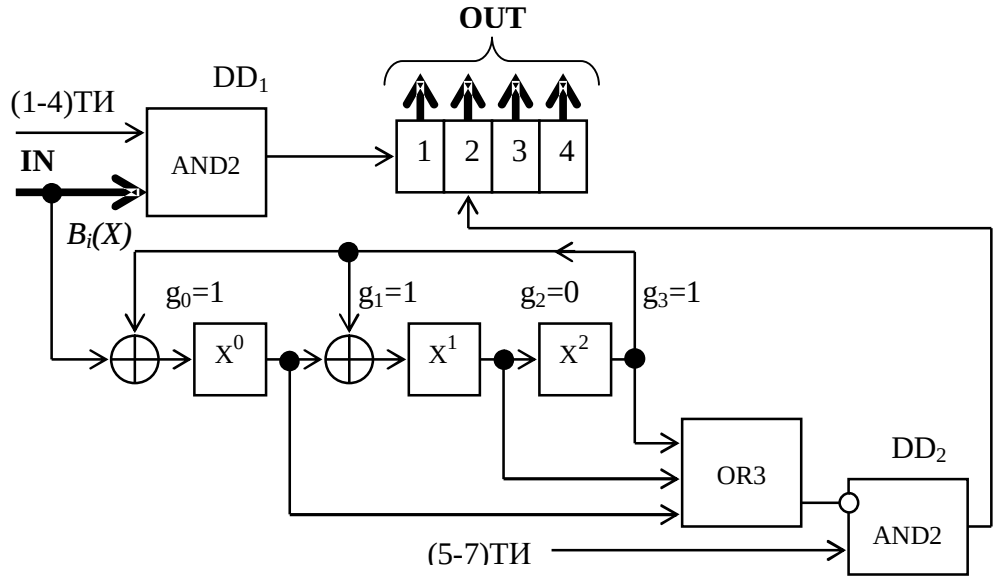


Рис. 1.11. Схема ДУ для ЦК Хемминга (7, 4)

синхронизатора на схему *AND2*. . Если $R(X) \neq 0$, то на выходе схемы *AND2* импульс не появится и считывания с ЗУ принятых информационных символов не произойдет. Если $R(X) = 0$, то появившийся на выходе *AND2* импульс считывает $A_i(X)$ на выход и выдает четыре информационные бита получателю сообщений.

1.5.9. ПРИНЦИПЫ ПОСТРОЕНИЯ ДЕКОДИРУЮЩИХ УСТРОЙСТВ ДЛЯ ЦИКЛИЧЕСКИХ КОДОВ С ИСПРАВЛЕНИЕМ ОШИБОК

Декодирование принятых комбинаций ЦК можно производить различными методами. Наряду с синдромным методом декодирования, основанным на вычислении остатка от деления принятой комбинации на порождающий код полином, существует целый ряд других методов, упрощающих процедуру декодирования и не требующих хранения в памяти ДУ большого числа синдромов при обработке длинных кодов. Для длинных ЦК разработаны специальные итеративные процедуры декодирования с исправлением нескольких ошибок, например, метод Берлекэмпа или более совершенный итеративный алгоритм Тренча-Берлекэмпа-Месси (ТБМ-метод), оперирующий с полиномами над полями Галуа. Различные методы декодирования так же, как и коды, получают авторские наименования. Известны алгоритмы декодирования Хемминга, Питерсона, Ченя, Мэггита, Витерби и других.

Декодирующие устройства для кодов, предназначенных только для обнаружения ошибок, по существу, не отличаются от схем Кодирующего Устройства (см. 1.5.8). В них добавляется лишь буферный регистр для хранения принятого сообщения на время проведения операции деления. Если остаток–синдром при делении оказывается нулевым, что свидетельствует об отсутствии ошибки, то информация с буферного регистра считывается в дешифратор сообщения. Если остаток обнаружен, что свидетельствует о наличии ошибки, то информация в буферном регистре уничтожается и на передающую сторону к источнику сообщения посылается сигнал запроса повторной передачи по обратному каналу связи.

В случае исправления ошибок схема Декодирующего Устройства, естественно, усложняется. Информацию о разрядах, в которых произошла ошибка, т. е. о виде шумового вектора $Z(X)$ (1.30), содержит, как и ранее, синдром, получаемый в результате деления полиномов. Структурная схема ДУ, решающего задачу исправления ошибок, показана на рис. 1.12.

Символы подлежащей декодированию кодовой комбинации, возможно, содержащей ошибку, последовательно, начиная со старшего разряда, вводятся в n-разрядный буферный регистр сдвига и одновременно в схему определителя синдрома, где за n тактов деления определяется остаток, который в случае синхронной, непрерывной передачи кодовых комбинаций сразу же переписывается в аналогичный *CP* схемы анализатора синдрома.

В состав схемы анализатора синдрома может входить ПЗУ, в котором записаны все возможные конфигурации синдромов с соответствующими им шумовыми векторами. Кодовые комбинации шумовых векторов (1.30) содержат «единичные» символы на тех позициях, которые в процессе передачи сообщения по каналу связи оказались искаженными помехами.

Локатор ошибок (определитель места ошибок) представляет собой комбинаторно-логическую схему, выдающую на выход единичные символы в те моменты времени, когда каждый из ошибочных символов принятой кодовой

комбинации занимает в буферном регистре крайнюю правую ячейку. При последующем тактовом сдвиге локатор ошибки (детектор ошибки) формирует символ «1», который поступает на сумматор коррекции, представляющий собой схему суммирования «по модулю 2», где исправляется искаженный символ.

Одновременно по цепи обратной связи с выхода локатора ошибки подается

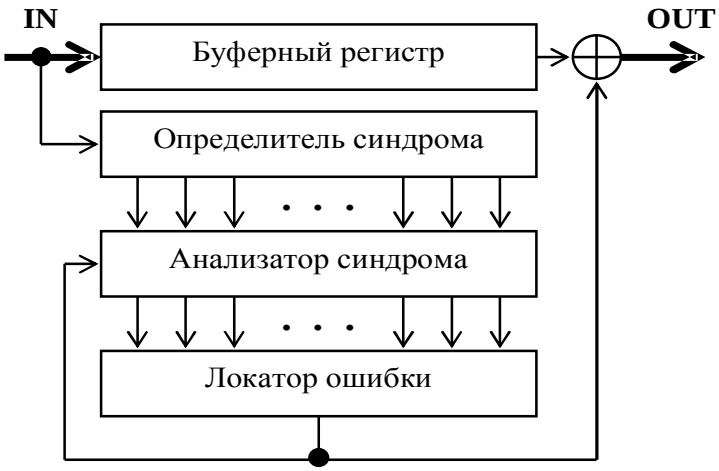


Рис.1.12. Структурная схема декодирующего устройства

единичный символ на анализатор синдрома, что в ряде конкретных схемных решений построения анализатора упрощает его построение на базе ЛПС без использования ПЗУ. Сложность анализатора синдрома и локатора ошибки зависит от гарантированного числа исправляемых и обнаруживаемых ошибок. Естественно, простейшие схемные решения получаются при обработке кодов, рассчитанных на исправление единичных ошибок.

Как видно из рассмотрения логики работы структурной схемы декодера (см. рис. 1.12), наиболее сложной частью его является необходимость запоминания заранее вычисленных синдромных полиномов и соответствующих им векторов ошибок. Достоинством ЦК как раз и является то, что анализатор синдрома можно значительно упростить, воспользовавшись алгебраической структурой кода для отыскания связей между синдромами при числе исправляемых ошибок $g_и > 1$. Опираясь на эти связи, можно запомнить в ПЗУ только полиномы ошибок, соответствующие некоторым типичным синдромным полиномам, а вычисление остальных осуществить затем с помощью простых вычислительных алгоритмов. Именно на таких принципах работают различные варианты декодеров Мэггита.

ЗАКЛЮЧЕНИЕ

История кодирования, контролирующего ошибки, началась в 1948 г. публикацией знаменитой статьи Клода Шеннона. Из шенноновской теории информации следует тот важный вывод, что построение слишком хороших каналов является расточительством; экономически выгоднее использовать кодирование. Фактически в работе Шеннона утверждается, что мощность сигнала, шум в канале и полоса частот ограничивают лишь скорость передачи, а не ее точность. Шеннон, однако, не указал, как найти подходящие коды, а лишь доказал их существование. В пятидесятые годы много усилий было потрачено на попытки построения в явном виде классов кодов, позволяющих получить обещанную сколь угодно малую вероятность ошибки, но результаты были скудными. В следующем десятилетии

решению этой увлекательной задачи уделялось меньше внимания; вместо этого исследователи кодов предприняли длительную атаку по двум основным направлениям.

Первое направление носило чисто алгебраический характер и преимущественно рассматривало блоковые коды. Первые блоковые коды были введены в 1950 г., когда Хэмминг описал класс блоковых кодов, исправляющих одиночные ошибки. Коды Хэмминга были разочаровывающе слабы по сравнению с обещанными Шенноном гораздо более сильными кодами. Несмотря на усиленные исследования, до конца пятидесятых годов не было построено лучшего класса кодов. В течение этого периода без какой-либо общей теории были найдены многие коды с малой длиной блока. Основной сдвиг произошел, когда Боуз и Рой-Чоудхури [1960] и Хоквингем [1959] нашли большой класс кодов, исправляющих кратные ошибки (коды БЧХ), а Рид и Соломон [1960] нашли связанный с кодами БЧХ класс кодов для недвоичных каналов. Хотя эти коды остаются среди наиболее важных классов кодов, общая теория блоковых кодов, контролирующих ошибки, с тех пор успешно развивалась.

Открытие кодов БЧХ привело к поиску практических методов построения жестких или мягких реализации кодеров и декодеров. Первый хороший алгоритм был предложен Питерсоном. Впоследствии мощный алгоритм выполнения описанных Питерсоном вычислений был предложен Берлекэмпом и Месси, и их реализация вошла в практику как только стала доступной новая цифровая техника. Второе направление исследований по кодированию носило скорее вероятностный характер. Ранние исследования были свя-заны с оценками вероятностей ошибки для лучших семейств блоковых кодов, несмотря на то, что эти лучшие коды не были известны. С этими исследованиями были связаны попытки понять кодирование и декодирование с вероятностной точки зрения, и эти попытки привели к появлению последовательного декодирования. В последовательном декодировании вводится класс неблоковых кодов бесконечной длины, которые можно описать деревом и декодировать с помощью алгоритмов поиска по дереву. Наиболее полезными древовидными кодами являются коды с тонкой структурой, известные под названием сверточных кодов. Эти коды можно генерировать с помощью цепей линейных регистров сдвига, выполняющих операцию свертки информационной последовательности. В конце 50-х годов для сверточных кодов были успешно разработаны алгоритмы последовательного декодирования. Интересно, что наиболее простой алгоритм декодирования - алгоритм Витерби - не был разработан для этих кодов до 1967 г. Применительно к сверточным кодам умеренной сложности алгоритм Витерби пользуется широкой популярностью, но для более мощных сверточных кодов он не практичен.

В 70-х годах эти два направления исследований опять стали переплетаться. Теорией сверточных кодов занялись алгебраисты, представившие ее в новом свете. В теории блоковых кодов за это время удалось приблизиться к кодам, обещанным Шенноном: были предложены две различные схемы кодирования (одна Юстесеном, а другая Гоппой), позволяющие строить семейства кодов, которые одновременно могут иметь очень большую длину блока и очень хорошие характеристики. Обе схемы, однако, имеют практические ограничения. Между тем к началу 80-х годов кодеры и декодеры начали появляться в конструкциях цифровых систем связи и цифровых систем памяти.

ЧАСТЬ 2. КВИТИРОВАНИЕ И ХЕШИРОВАНИЕ ДАННЫХ

Современное помехоустойчивое кодирование способно сколь угодно надежно защитить информацию, хранящуюся на носителях или циркулирующую в открытых каналах связи от случайных (непреднамеренных) искажений, вызванных шумами, помехами, сбоями в работе аппаратуры и т.п. Однако, это обходится крайне дорого с точки зрения объемов хранения и передачи информации. Так, использование кода Хемминга, увеличивая объем передаваемых данных на 50%, позволяет исправлять только одиночные ошибки.

Статистические данные по обнаружению и исправлению ошибок обнаруживают закономерности, которые показывают, что случайные ошибки возникают либо относительно редко, либо, если возникают то целыми пакетами, которые исправить весьма проблематично.

Если данные уникальны и их утрата невосполнима, а экономическая сторона не существенна, то помехоустойчивое кодирование с использованием самовосстанавливающихся кодов дает прекрасный инструмент для надежного сохранения или передачи этих данных.

Если же передаваемые данные могут быть возобновлены, их обычно передают наиболее экономичным способом, но сопровождают специальной структурой – квитанцией, по которой можно судить о наличие или отсутствии ошибок в принятых данных. Если такие ошибки обнаруживаются, то принимающая сторона делает запрос передающей стороне на повторную передачу данных.

2.1. ОСНОВНЫЕ ПОНЯТИЯ

Квитанция – это сжатый образ (дайджест) блока передаваемых или хранимых данных, представляет собой структуру, имеющую относительно небольшой постоянный объем.

К квитанции предъявляется ряд требований:

- квитанция однозначно отражает содержимое файла.
- квитанция имеет постоянную длину.
- по квитанции не может быть восстановлен (реконструирован) квитируемый файл.
- квитанция должна быть легко реализуема с минимальными затратами вычислительных ресурсов как программными, так и аппаратными средствами.

В общем случае однозначного соответствия между исходными данными и квитанцией нет в силу того, что количество значений квитанций меньше, чем вариантов исходного массива данных. Существует множество массивов данных, дающих одинаковые квитанции – так называемые **коллизии**.

Основная задача квитанции – защита передаваемых или хранимых данных от непреднамеренных искажений.

Простейшими примерами квитанций являются различные контрольные суммы:

- Арифметическая сумма, легко реализуется аппаратно, однако ее формирование требует относительно большие вычислительные ресурсы. Для обеспечения требования фиксированной длины, в качестве квитанции

используют один или два младших байта суммы, а старшие байты отбрасывают. Используется редко.

- Логическая сумма «по модулю 2» формирует бит «четности», который используется для защиты очень коротких сообщений, обычно не превышающих 1 – 2 байта. Вычисление требует минимальных затрат вычислительных ресурсов.
- Логическая сумма «по модулю 2» 32- или 16-битных слов, на которые предварительно разбивается сообщение. Используется, например, в TCP/IP.
- Семейство контрольных сумм CRC (cyclic redundancy check «циклический избыточный код») основано на использовании циклических кодов применяемых в помехоустойчивом кодировании. Требует умеренных затрат вычислительных ресурсов. Получило в настоящее время наибольшее распространение, например, CRC32 применяется в аппаратуре *Ethernet*, CRC8 и CRC16 – квитируют аудио- и видео-файлы, содержимое пластиковых карт и др.

Разновидностью квитанций являются хеш-функции, к которым предъявляются требования криптостойкости. Кроме собственно квитирования, хеш-функция предназначена для обнаружения умышленных искажений хранимых или передаваемых данных. Иными словами, хеш-функция – это «усиленная» квитанция с дополнительными свойствами, и как следствие, требующая значительно больших затрат вычислительных ресурсов для своего формирования.

Хеширование (иногда хэширование, англ. *hashing*) – преобразование входного массива данных произвольной длины в выходную битовую строку фиксированной длины. Такие преобразования также называются **хеш-функциями** или **функциями свёртки**, а их результаты называют **хешем**, **хеш-кодом** или **дайджестом** сообщения (англ. *message digest*).

Хеширование применяется для сравнения данных: если у двух массивов хеш-коды разные, массивы гарантированно различаются; если одинаковые – массивы, скорее всего, одинаковы. В общем случае, количество значений хеш-функций меньше, чем вариантов входного массива, поэтому существует множество массивов данных (файлов), дающих одинаковые хеш-коды – так называемые **коллизии**. Вероятность возникновения коллизий играет важную роль в оценке качества хеш-функций. Идеальная хеш-функция не должна иметь коллизий.

2.2. КОНТРОЛЬНАЯ СУММА CRC

Контрольная сумма CRC (Cyclic Redundancy Check - циклический избыточный код), представляет собой метод выявления ошибок при хранении или передаче данных, но не вносит исправлений при обнаружении ошибок. CRC используется в основном в передаче данных. В методе CRC, к передаваемому сообщению добавляется определенное количество контрольные биты, которые часто называют «контрольной суммой». Принимающая сторона, получив сообщение, может с определенной степенью вероятности определить, действительно ли контрольные биты согласуются с принятыми данными. Если произошла ошибка, принимающая сторона посылает обратно отправителю «отрицательный признак» (negative acknowledgement NAK) с просьбой о повторной передаче сообщения.

Метод CRC также применяется и к сохраняемым данным, например, на жестком диске. В этом случае, каждый блок на диске имеет контрольные биты CRC,

и, если обнаруживается ошибка, автоматически аппаратно инициируется повторное чтение блока или делается сообщение об этом программному обеспечению.

Есть несколько методов для создания проверочных битов, которые могут быть добавлены в сообщение. Самый простой метод состоит в добавлении одного бита, называемого «бит четности» или «бит паритета» (parity bit). Бит четности делает общее количество единиц «1» в кодовом векторе сообщения четным (при проверке на четность «even parity») или нечетным (при проверке на нечетность «odd parity»). Если при передаче произойдет изменение одного бита, это изменит бит паритета с четного на нечетный (или наоборот). Отправитель генерирует бит простым суммированием битов сообщения «по модулю 2», то есть выполняет логическую операцию «исключающее ИЛИ». Затем он добавляет бит четности (или его дополнение) к сообщению. Получатель может проверить сообщение, суммируя все биты сообщения «по модулю 2» (логическая функция XOR или ⊕) и убедиться, что сумма согласуется с битом четности, если результирующая сумма всех бит (сообщения и четности) будет 0 (проверка на четность).

Эта простая техника четности может обнаруживать одиночные ошибки. На самом деле обнаруживается любое нечетное число ошибочных бит (включая и бит четности), но это слабое утешение знать, что вы обнаружили 3-битовые ошибки и пропустили 2-битовые или 4-битовые ошибки.

При отправке и получении однобитовых последовательностей (последовательная передача данных), аппаратные средства для генерации и проверки бита четности очень просты. Они состоят из одного ключа XOR и некоторой схемы управления. При параллельной передаче нескольких бит, т.е. слова информации, может быть использовано дерево из отдельных ключей XOR, рис. 2.1..

Другие методы для вычисления контрольной суммы либо формируют

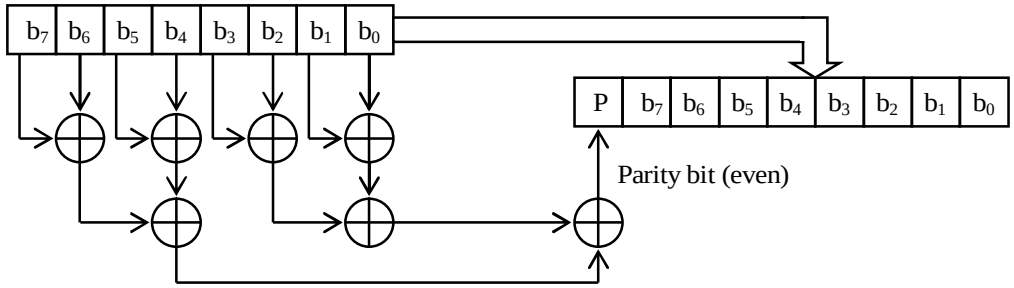


Рис. 2.1. Схема аппаратного получения бита четности для 8-ми битного слова

«исключительной ИЛИ» для всех байтов в сообщении, либо вычисляют сумму с кольцевым переносом всех байтов. В последнем методе, перенос из каждой 8-битной суммы добавляется к младшему разряду аккумулятора. Последнее позволяет наиболее просто обнаружить ошибки, чем простое «исключающее ИЛИ» или сумма байт с отбрасыванием переноса.

Считается, что метод циклического контроля избыточности CRC, наиболее хорош как с точки зрения обнаружения ошибок, так и легкости реализации на аппаратном уровне. Это еще один метод вычисления контрольной суммы, которая, как правило, имеет длину бит 8, 16 или 32 бит, и добавляется к сообщению. Мы кратко рассмотрим теорию, а затем дадим некоторые алгоритмы вычисления 32-

разрядной контрольной суммы CRC, которая часто используется в программном обеспечении.

Теория

CRC основан на полиномиальной арифметике, в частности, на вычислении остатка от деления одного многочлена на другой в GF(2) (поле Галуа из двух элементов). Многочлен GF(2) является многочленом от одной переменной X, коэффициентами которого являются 0 или 1.

Сложение и вычитание выполняются «по модулю 2», то есть, они выполняют операцию «исключающее ИЛИ», которая представляет собой логическая функцию XOR.

Приведем далее порядок суммирования (вычитания), умножения и деления полиномов с учетом того, что операция суммирования осуществляется *«по модулю 2»*. В примерах используем полиномы (многочлены): $A_1(X)=X^5+X^3+X^2$ и $A_2(X)=X^4+X^2+X$.

Суммирование (вычитание):

$A_1+A_2 = A_1-A_2 = X^5+X^4+X^3+X^2+X^2+X = X^5+X^4+X^3+X$

или

A_1 101100

$\oplus$

A_2 010110

111010 $= X^5 + X^4 + X^3 + X.$

Умножение:

$A_1 \times A_2 = (X^5+X^3+X^2) \times (X^4+X^2+X) =$

$X^9+X^7+X^6+X^7+X^5+X^4+X^6+X^4+X^3 = X^9+X^5+X^3 = 1000101000.$

Произведение многочленов при вычислении CRC не используется.

Деление: $\frac{A_1}{A_2}$

$X^5 + X^3 + X^2$	$X^4 + X^2 + X$
$X^5 + X^3 + X^2$	X
0 0 0	- остаток при делении $R(X) = 0$

Из последнего примера следует, что сдвиг полинома вправо на один разряд эквивалентен делению его на *X*, а сдвиг влево на один разряд – эквивалентен умножению полинома на *X*. Однако, при вычислении контрольной суммы CRC используется только остаток от деления, который, например, при делении $X^7+X^6+X^5+X^2+X$ на X^4+X^3+1 составляет X^2+1, а целая часть – X^3+X+1. Последнее можно проверить обратной операцией – умножением: $(X^3+X+1)(X^4+X^3+1)+X^2+1 = X^3X^4+X^3X^3+X^3 +XX^4+XX^3+X +X^4+X^3+1 +X^2+1 = X^7+X^6+X^3+ X^5+X^4+X +X^4+X^3+1+X^2+1$. После приведения «по модулю 2», и учитывая, что $1\oplus1=0$ (четное число одинаковых степеней – приводится), получим: $X^7+X^6+X^5+X^2+X$.

Метод CRC рассматривает всякое сообщение как многочлен GF(2). Например, сообщение 11001001, где младший бит справа, рассматривается как представление полинома $X^7+X^6+X^3+1$. Отправитель и получатель должны договориться о некотором фиксированном многочлене, называемом «порождающим многочленом». Например, для 16-битных CRC комитетом CCITT был выбран образующий полином $X^{16}+X^{12}+X^5+1$, который в настоящее время широко используется для формирования 16-битной контрольной суммы CRC-16. Для вычисления *r*-битной контрольной

суммой *CRC-r*, порождающий полином должны быть степени *r*. Отправитель дополняет *r* 0-битами *m*-битное сообщение и делит полученный многочлен степени *m+r-1* на порождающий полином. Это дает полином-остаток степени *r-1*, или меньше. Полученный полином-остаток имеет *r* коэффициентов и является контрольной суммой CRC. Целочисленный полином отбрасывается. Передаваемых данных, кодовый вектор, является оригинальным *m*-битным сообщением, дополненным *r*-битной контрольной суммой.

Есть два способа для оценки принимающей стороной правильности передачи. Первый, это когда принимающая сторона вычисляет контрольную сумму для первых *m*-бит сообщения и удостоверяется, что она совпадает с последними *r* разрядами полученного сообщения. Во втором способе, который обычно и используется, все полученные биты нужно разделить на порождающий полином и результирующий остаток должен быть равен 0. Чтобы убедиться, что остаток должен быть равен 0, примем, что *M* это полиномиальное представление сообщения, а *R* – полиноминальное представление остатка, который был вычислен при отправлении сообщения. Переданным данным соответствует многочлен MX^r-R , или, что эквивалентно, MX^r+R (умножение на X^r эквивалентно сдвигу всего числа влево на *r*-бит или, что тоже самое, добавлению к выражению справа *r*-нулей). Известно, что MX^r=QG+R, где *G* является порождающим полиномом и *Q* является целой частью (которая была отброшена). Поэтому переданные данные представляют собой выражение *QG*, кратное *G*. Хотя процесс оценки правильности передачи сообщения изменился незначительно, второй способ не чувствителен к произвольному числу начальных и конечных 0-бит передаваемых данных. Однако если произошел сбой, который вызвал в полученных данных все-0, включая контрольную сумму, сообщение будет принято как правильное!

Выбор «хорошего» порождающего полинома *G* сродни к искусству, и выходит за рамки нашего рассмотрения. Два простых наблюдений: Для *r*-битной контрольной суммы, *G* должна быть степени *r*, поскольку в противном случае первый бит контрольной суммы всегда будет 0. Аналогично, последний коэффициент должен быть равен 1 (то есть, *G* не должен делиться на *X*), поскольку в противном случае последний бит контрольной суммы всегда будет 0 (поскольку MX^r = QG + R, если *G* делится на *X*, то и *R* также должен делиться).

- Строгая теория дает следующие свойства порождающего полинома:
- если *G* содержит больше двух членов, то обнаруживаются все одиночные ошибки;
- если *G* не делится на *X* (то есть, если последний член равен 1), а *e* наименьшее положительное целое число такое, что *G* нацело делит X^e+1, то обнаруживаются все двойные ошибки, которые находятся в пределах *e*. Наилучшими является многочлен $X^{15}+X^{14}+1$, для которого *e* = 32767;
- если X+1 является делителем для *G*, то обнаруживаются все ошибки для нечетного числа бит;
- *r*-битная контрольная сумма CRC обнаруживает все пакеты ошибок длины ≤ *r* (пакетом ошибок длиной *r* является строка битов *r*, в котором первый и последний – ошибки, а промежуточные *r*-2 биты могут быть или не быть ошибочными.).

Порождающий полином $x+1$ создает контрольную сумму длины 1, которая является битом четности в сообщении. (Чтобы удостовериться в этом, нужно найти остаток от деления x^k на $x+1$ для сообщения с $k \geq 0$).

Интересно отметить, что если контрольная сумма любого типа обнаруживает все одиночные и двойные ошибки, то, в принципе, можно исправить все одиночные ошибки. Чтобы убедится в этом, предположим, что получены данные содержащие однобитную ошибку. Будем последовательно дополнять (инвертировать) каждый бит сообщения один раз за проход. Тогда всякий раз будет возникать двухбитная ошибка, кроме случая сбойного байта, на котором произойдет исправление ошибки, а контрольная сумма покажет отсутствие ошибок.

В табл. 2.1 приведены порождающие полиномы, используемые некоторыми общими стандартами CRC. «Hex» колонка показывает шестнадцатеричное представление порождающего полинома (запись по *mod8*), у которого старший бит пропущен, так как он всегда равен 1.

Таблица 2.1. Коллекция порождающих полиномов CRC

Степень *G(X)*	Обозначение контрольной суммы	Образующий полином *G(X)*	Hex
1	CRC-1	$x + 1$ (*parity bit* – бит четности)	0x1
4	CRC-4-ITU	$x^4 + x + 1$ (ITU-T G.704)	0x3
5	CRC-5-EPC	$x^5 + x^3 + 1$ (Gen 2 RFID)	0x09
5	CRC-5-ITU	$x^5 + x^4 + x^2 + 1$ (ITU-T G.704)	0x15
5	CRC-5-USB	$x^5 + x^2 + 1$ (USB token packets)	0x05
6	CRC-6-ITU	$x^6 + x + 1$ (ITU-T G.704)	0x03
7	CRC-7	$x^7 + x^3 + 1$ (telecom systems, ITU-T G.707, ITU-T G.832, MMC, SD)	0x09
8	CRC-8-CCITT	$x^8 + x^2 + x + 1$ (ATM HEC), ISDN Header Error Control and Cell Delineation ITU-T I.432.1 (02/99)	0x07
8	CRC-8-Dallas/Maxim	$x^8 + x^5 + x^4 + 1$ (1-Wire bus)	0x31
8	CRC-8	$x^8 + x^7 + x^6 + x^4 + x^2 + 1$	0xD5
8	CRC-8-SAE J1850	$x^8 + x^4 + x^3 + x^2 + 1$	0x1D
	CRC-8-WCDMA	$x^8 + x^7 + x^4 + x^3 + x + 1$	0x9B
10	CRC-10	$x^{10} + x^9 + x^5 + x^4 + x + 1$ (ATM; ITU-T I.610)	0x233
11	CRC-11	$x^{11} + x^9 + x^8 + x^7 + x^2 + 1$ (FlexRay)	0x385
12	CRC-12	$x^{12} + x^{11} + x^3 + x^2 + x + 1$ (telecom systems)	0x80F
15	CRC-15-CAN	$x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1$	0x4599
16	CRC-16-IBM	$x^{16} + x^{15} + x^2 + 1$ (Bisync, Modbus, USB, ANSI X3.28, many others; also known as *CRC-16* and *CRC-16-ANSI*)	0x8005
16	CRC-16-CCITT	$x^{16} + x^{12} + x^5 + 1$ (X.25, V.41, HDLC, XMODEM, Bluetooth, SD, many others; known as *CRC-CCITT*)	0x1021

16	CRC-16-T10-DIF	$x^{16} + x^{15} + x^{11} + x^9 + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ (SCSI DIF)	0x8BB7
16	CRC-16-DNP	$x^{16} + x^{13} + x^{12} + x^{11} + x^{10} + x^8 + x^6 + x^5 + x^2 + 1$ (DNP, IEC 870, M-Bus)	0x3D65
16	CRC-16-DECT	$x^{16} + x^{10} + x^8 + x^7 + x^3 + 1$ (cordless telephones)	0x0589
16	CRC-16-Fletcher	Not a CRC; see Fletcher's checksum	
24	CRC-24	$x^{24} + x^{22} + x^{20} + x^{19} + x^{18} + x^{16} + x^{14} + x^{13} + x^{11} + x^{10} + x^8 + x^7 + x^6 + x^3 + x + 1$ (FlexRay)	0x5D6DCB
24	CRC-24-Radix-64	$x^{24} + x^{23} + x^{18} + x^{17} + x^{14} + x^{11} + x^{10} + x^7 + x^6 + x^5 + x^4 + x^3 + x + 1$ (OpenPGP)	0x864CFB
30	CRC-30	$x^{30} + x^{29} + x^{21} + x^{20} + x^{15} + x^{13} + x^{12} + x^{11} + x^8 + x^7 + x^6 + x^2 + x + 1$ (CDMA)	0x2030B9C7
32	CRC-32-Adler	Not a CRC; see Adler-32	
32	CRC-32-IEEE 802.3	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ (V.42, Ethernet, SATA, MPEG-2, PNG, POSIX cksum)	0x04C11DB7
32	CRC-32C (Castagnoli)	$x^{32} + x^{28} + x^{27} + x^{26} + x^{25} + x^{23} + x^{22} + x^{20} + x^{19} + x^{18} + x^{14} + x^{13} + x^{11} + x^{10} + x^9 + x^8 + x^6 + 1$ (iSCSI & SCTP, G.hn payload, SSE4.2)	0x1EDC6F41
32	CRC-32K (Koopman)	$x^{32} + x^{30} + x^{29} + x^{28} + x^{26} + x^{20} + x^{19} + x^{17} + x^{16} + x^{15} + x^{11} + x^{10} + x^7 + x^6 + x^4 + x^2 + x + 1$	0x741B8CD7
32	CRC-32Q	$x^{32} + x^{31} + x^{24} + x^{22} + x^{16} + x^{14} + x^8 + x^7 + x^5 + x^3 + x + 1$ (aviation; AIXM)	0x814141AB
40	CRC-40-GSM	$x^{40} + x^{26} + x^{23} + x^{17} + x^3 + 1$ (GSM control channel)	0x0004820009
64	CRC-64-ISO	$x^{64} + x^4 + x^3 + x + 1$ (HDLC — ISO 3309, Swiss-Prot/TrEMBL; considered weak for hashing)	0x000000000000001B
64	CRC-64-ECMA-182	$x^{64} + x^{62} + x^{57} + x^{55} + x^{54} + x^{53} + x^{52} + x^{47} + x^{46} + x^{45} + x^{40} + x^{39} + x^{38} + x^{37} + x^{35} + x^{33} + x^{32} + x^{31} + x^{29} + x^{27} + x^{24} + x^{23} + x^{22} + x^{21} + x^{19} + x^{17} + x^{13} + x^{12} + x^{10} + x^9 + x^7 + x^4 + x + 1$ (as described in ECMA-182)	0x42F0E1EBA9EA3693
128	CRC-128 *		
256	CRC-256 *		

* CRC-128 и CRC-256 в настоящее время вытеснены хеш-функциями.

В то время, как циклические избыточные коды являются частью стандартов, сами они не стандартизированы. Например, существуют три описания полинома для CRC-12, десять противоречивых определений CRC-16 и четыре – CRC-32. При этом многие широко используемые полиномы не являются наиболее эффективными из всех возможных. В 1993–2004 годах Koopman, Castagnoli и другие, исследовали пространство полиномов разрядности до 16, 24 и 32 битов, найдя полиномы, дающие лучшую производительность, чем полиномы из существующих протоколов, и опубликовали лучшие из них с целью улучшения качества обнаружения ошибок в будущих стандартах. Один из результатов этого исследования уже нашёл своё применение в протоколе iSCSI.

Самый популярный, рекомендуемый IEEE полином для CRC-32, используемый в Ethernet, FDDI и др., является генератором кода Хемминга, и был выбран, основываясь на его производительности и способности обнаружения ошибок передачи данных. Использование другого полинома CRC-32C, предложенного Castagnoli и применяемого, в частности, в iSCSI, позволяет достичь такой же производительности (при длине исходного сообщения от 58 бит до 131 кбит). А в некоторых диапазонах длины входного сообщения, включая два наиболее распространенных размера IP-пакетов, скорость вычисления CRC-32C может быть даже выше. Стандарт ITU-T G.hn также использует CRC-32C с целью обнаружения ошибок в полезной нагрузке, а для заголовков PHY – CRC16МККТТ.

Аппаратный способ формирования CRC

Процесс использует сдвиговый регистр (СР) с обратными связями, которые выполняют сложение «по модулю 2» или логическую операцию XOR. Калькулятор CRC делит полиноминальное представление входного файла на соответствующий порождающий полином и получает остаток, который и является контрольной суммой CRC.

Формирование калькулятора CRC рассмотрим на примере. Пусть сообщение $M=X^7+X^6+X^5+X^2+X$, а порождающий полином – $G= X^3+X+1$. Калькулятор представляет собой последовательно соединенные триггеры, число которых равно степени полинома *G,* на входы которых поступают биты с выходов элементов XOR (сумматор «по модулю 2»). На один вход XOR поступает бит с предыдущего триггера, а на второй вход – с выхода последнего (старшего) триггера. Если коэффициент при X равен 0, то на второй вход поступает постоянно 0-бит и XOR становится повторителем и может быть опущен. На незадействованный вход XOR первого (младшего) триггера последовательно старшим битом вперед (MSB), поступает сообщение *M*, рис. 2.2-а.

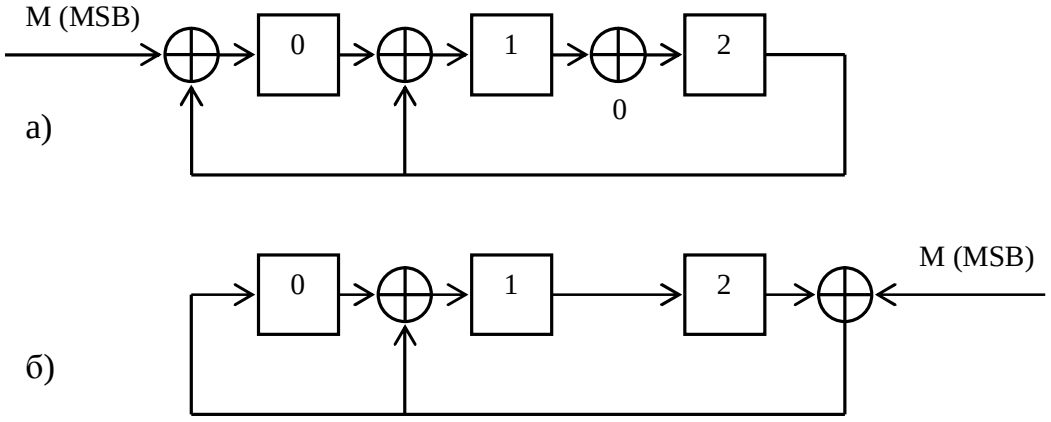


Рис. 2.2. Схема получения остатка от деления на $G=X^3+X+1$ (а) и схема формирования контрольной суммы CRC для $G=X^3+X+1$ (б).

Сначала найдем остаток от деления самого сообщения $M=X^7+X^6+X^5+X^2+X$ (11100110 – младший бит справа) на порождающий полином $G= X^3+X+1$ (1011 – младший бит справа), схема рис.2.2-а:

Такт	Входное сообщение, MSB	X^0	X^1	X^2	Остаток
0	0 1 1 0 0 1 1 1	0	0	0	
1	0 1 1 0 0 1 1	1	0	0	100
2	0 1 1 0 0 1	1	1	0	110
3	0 1 1 0 0	1	1	1	111
4	0 1 1 0	1	0	1	101
5	0 1 1	1	0	0	100
6	0 1	1	1	0	110
7	0	1	1	1	111
		1	0	1	101 – младший бит слева

Теперь дополним в конце сообщение M тремя нулями, так как степень полинома G равна трем и продолжим деление:

Такт	Входное сообщение, MSB	X^0	X^1	X^2	Остаток
8	0 0 0	1	0	1	101
9	0 0	1	0	0	100
10	0	0	1	0	010
11		0	0	1	001 – младший бит слева

Остаток в сдвиговом регистре является искомой контрольной суммой CRC=’100’ – младший бит справа. Если вычислить контрольную суммы CRC для принятого от отправителя двоичного сообщения M*'100’, и она окажется равной 0, то это свидетельствует об отсутствии ошибок при передаче данных и M=M*.

Использование для обработки сообщения схемы на рис.2.2-а, требует после получения остатка дополнить сообщения числом нулей равным степени полинома G и продолжить вычисления до получения кода CRC. Схема на рис.2.2-б позволяет избежать дополнения нулями и сразу получить код CRC:

Такт	Входное сообщение, MSB	X^0	X^1	X^2	Остаток
	0 1 1 0 0 1 1 1	0	0	0	
1	0 1 1 0 0 1 1	1	1	0	110
2	0 1 1 0 0 1	1	0	1	101
3	0 1 1 0 0	0	1	0	010
4	0 1 1 0	0	0	1	001
5	0 1 1	1	1	0	110
6	0 1	1	0	1	101
7	0	0	1	0	010
8		0	0	1	001 – младший бит слева

Получили такую же контрольную сумму CRC='100' – младший бит справа.

На рис. 2.3.приведена схема аппаратного вычисления контрольной суммы CRC-32 образующего полинома $G= X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$ протоколов V.42, Ethernet, SATA и др.

Контрольная сумма CRC вычисляется за число тактов, равное объему квитируемого файла в битах. Программные методы вычисления CRC позволяют делать свертку файлов байтами.

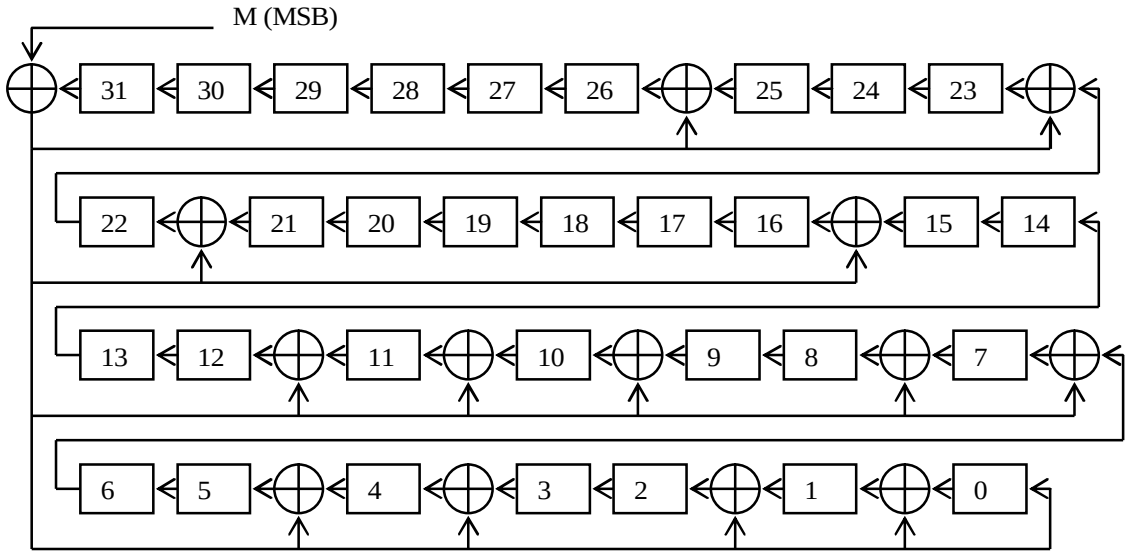


Рис. 2.3. Схема аппаратного вычисления контрольной суммы CRC-32 образующего полинома $G= X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$ (hex-cod=04C11DB7).

Программный способ формирования CRC

Существует множество алгоритмов программного вычисления контрольной суммы CRC. Для всех их характерно получение результата не для отдельных бит, а для целых байт. Поэтому степень образующего полинома должна быть кратна 8: CRC-8, CRC-16, CRC-32, CRC-64 и т.д.

Наибольшая производительность вычисления CRC достигается использованием табличного метода. Обычно стремятся избежать необходимости предварительного дополнения входными нулями, число которых равно степени образующего полинома *G*. Тем не менее, входной фай должен быть выровнен на

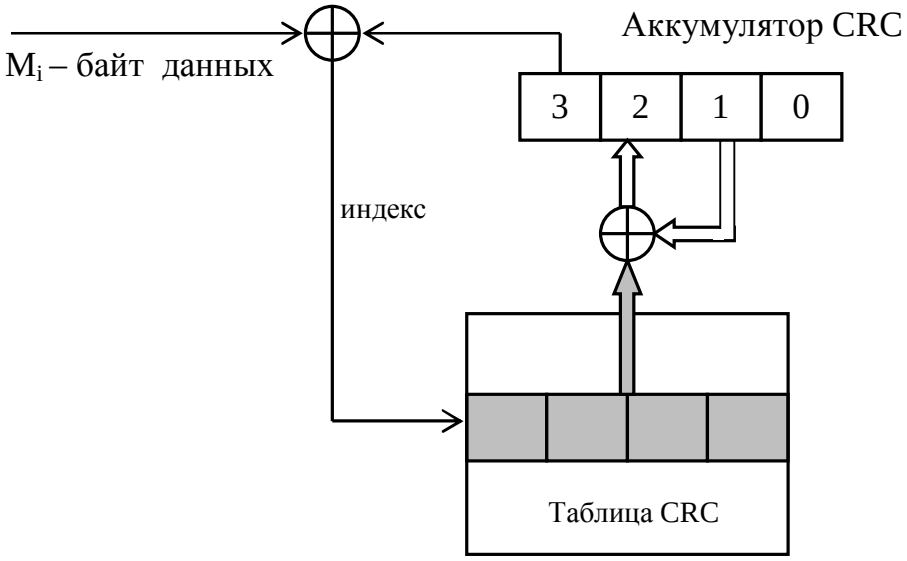


Рис. 2.4. Схема табличного вычисления CRC.

целое число байт. На рис.2.4 приведена схема прямого табличного вычисления контрольной суммы CRC.

Порядок вычисления CRC:

Шаг 1: Предварительная инициализация аккумулятора CRC начальным кодом

Шаг 2: Побитное сложение «по модулю 2» входного байта данных («старшим байтом вперед») со старшим байтом содержимого аккумулятора CRC. Получается указатель (индекс) для адресации значений (слов) в таблице CRC.

Шаг 3: Выбор индексированного слова в таблице CRC.

Шаг 4: Сложение «по модулю 2» выбранного слова с содержимым аккумулятора CRC и помещение результата снова в аккумулятор CRC.

Шаг 5: Шаги 2 – 4 повторяются, пока не будет исчерпан весь файл.

Шаг 6: После исчерпания последнего байта файла, выполняется заключительное сложение «по модулю 2» содержимого аккумулятора с конечным кодом. В аккумуляторе находится контрольная сумма CRC.

В табл. 2.2 – 2.4 приведены таблицы CRC для наиболее распространенных алгоритмов: CRC-8 (hex-cod: 0x31, начальный код: 0x00, конечный код: 0x00), CRC-16 (hex-cod: 0x8005, начальный код: 0x0000, конечный код: 0x0000), CRC-32 (hex-cod: 0x04C11DB7, начальный код: 0xFFFFFFFF, конечный код: 0xFFFFFFFF).

Таблица 2.2. CRC-8 ($x^8 + x^5 + x^4 + 1$)

00	00	5E	BC	E2	61	3F	DD	83	C2	9C	7E	20	A3	FD	1F	41
10	9D	C3	21	7F	FC	A2	40	1E	5F	01	E3	BD	3E	60	82	CD
20	23	7D	9F	C1	42	1C	FE	A0	E1	BF	5D	03	80	DE	3C	62
30	BE	E0	02	5C	DF	81	63	3D	7C	22	C0	9E	1D	43	A1	FF
40	46	18	FA	A4	27	79	9B	C5	84	DA	38	66	E5	BB	59	07
50	DB	85	67	39	BA	E4	06	58	19	47	A5	FB	78	26	C4	9A
60	65	3B	D9	87	04	5A	B8	E6	A7	F9	1B	45	C6	98	7A	24
70	F8	A6	44	1A	99	C7	25	7B	3A	64	86	D8	5B	05	E7	B9
80	8C	D2	30	6E	ED	B3	51	0F	4E	10	F2	AC	2F	71	93	CD
90	11	4F	AD	F3	70	2E	CC	92	D3	8D	6F	31	B2	EC	0E	50
A0	AF	F1	13	4D	CE	90	72	2C	6D	33	D1	8F	0C	52	B0	EE
B0	32	6C	8E	D0	53	0D	EF	B1	F0	AE	4C	12	91	CF	2D	73
C0	CA	94	76	28	AB	F5	17	49	08	56	B4	EA	69	37	D5	8B
D0	39	09	EB	B5	36	68	8A	D4	95	CB	29	77	F4	AA	48	16
E0	E9	B7	55	0B	88	D6	34	6A	2B	75	97	C9	4A	14	F6	A8
F0	74	2A	C8	96	15	4B	A9	F7	B6	E8	0A	54	D7	89	6B	35

Таблица 2.3. CRC-16 ($x^{16} + x^{15} + x^2 + 1$)

00	0000	C0C1	C181	0140	C301	03C0	0280	C241	C601	06C0	0780	C741	0500	C5C1	C481	0440
10	CC01	0CC0	0D80	CD41	0F00	CFC1	CE81	0E40	0A00	CAC1	CB81	0B40	C901	09C0	0880	C841
20	D801	18C0	1980	D941	1B00	DBC1	DA81	1A40	1E00	DEC1	DF81	1F40	DD01	1DC0	1C80	DC41
30	1400	D4C1	D581	1540	D701	17C0	1680	D641	D201	12C0	1380	D341	1100	D1C1	D081	1040
40	F001	30C0	3180	F141	3300	F3C1	F281	3240	3600	F6C1	F781	3740	F501	35C0	3480	F441
50	3C00	FCC1	FD81	3D40	FF01	3FC0	3E80	FE41	FA01	3AC0	3B80	FB41	3900	F9C1	F881	3840
60	2800	E8C1	E981	2940	EB01	2BC0	2A80	EA41	EE01	2EC0	2F80	EF41	2D00	EDC1	EC81	2C40
70	E401	24C0	2580	E541	2700	E7C1	E681	2640	2200	E2C1	E381	2340	E101	21C0	2080	E041
80	A001	60C0	6180	A141	6300	A3C1	A281	6240	6600	A6C1	A781	6740	A501	65C0	6480	A441
90	6C00	ACC1	AD81	6D40	AF01	6FC0	6E80	AE41	AA01	6AC0	6B80	AB41	6900	A9C1	A881	6840
A0	7800	B8C1	B981	7940	BB01	7BC0	7A80	BA41	BE01	7EC0	7F80	BF41	7D00	BDC1	BC81	7C40

B0	B401	74C0	7580	B541	7700	B7C1	B681	7640	7200	B2C1	B381	7340	B101	71C0	7080	B041
C0	5000	90C1	9181	5140	9301	53C0	5280	9241	9601	56C0	5780	9741	5500	95C1	9481	5440
D0	9C01	5CC0	5D80	9D41	5F00	9FC1	9E81	5E40	5A00	9AC1	9B81	5B40	9901	59C0	5880	9841
E0	8801	48C0	4980	8941	4B00	8BC1	8A81	4A40	4E00	8EC1	8F81	4F40	8D01	4DC0	4C80	8C41
F0	4400	84C1	8581	4540	8701	47C0	4680	8641	8201	42C0	4380	8341	4100	81C1	8081	4040

Таблица 2.4. CRC-32 ($x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$)

00	00000000	77073096	EE0E612C	990951BA	076DC419	706AF48F	E963A535	9E6495A3
08	0EDB8832	79DCB8A4	E0D5E91E	97D2D988	09B64C2B	7EB17CBD	E7B82D07	90BF1D91
10	1DB71064	6AB020F2	F3B97148	84BE41DE	1ADAD47D	6DDDE4EB	F4D4B551	83D385C7
18	136C9856	646BA8C0	FD62F97A	8A65C9EC	14015C4F	63066CD9	FA0F3D63	8D080DF5
20	3B6E20C8	4C69105E	D56041E4	A2677172	3C03E4D1	4B04D447	D20D85FD	A50AB56B
28	35B5A8FA	42B2986C	DBBBC9D6	ACBCF940	32D86CE3	45DF5C75	DCD60DCF	ABD13D59
30	26D930AC	51DE003A	C8D75180	BFD06116	21B4F4B5	56B3C423	CFBA9599	B8BDA50F
38	2802B89E	5F058808	C60CD9B2	B10BE924	2F6F7C87	58684C11	C1611DAB	B6662D3D
40	76DC4190	01DB7106	98D220BC	EFD5102A	71B18589	06B6B51F	9FBFE4A5	E8B8D433
48	7807C9A2	0F00F934	9609A88E	E10E9818	7F6A0DBB	086D3D2D	91646C97	E6635C01
50	6B6B51F4	1C6C6162	856530D8	F262004E	6C0695ED	1B01A57B	8208F4C1	F50FC457
58	65B0D9C6	12B7E950	8BBEB8EA	FCB9887C	62DD1DDF	15DA2D49	8CD37CF3	FBD44C65
60	4DB26158	3AB551CE	A3BC0074	D4BB30E2	4ADFA541	3DD895D7	A4D1C46D	D3D6F4FB
68	4369E96A	346ED9FC	AD678846	DA60B8D0	44042D73	33031DE5	AA0A4C5F	DD0D7CC9
70	5005713C	270241AA	BE0B1010	C90C2086	5768B525	206F85B3	B966D409	CE61E49F
78	5EDEF90E	29D9C998	B0D09822	C7D7A8B4	59B33D17	2EB40D81	B7BD5C3B	C0BA6CAD
80	EDB88320	9ABFB3B6	03B6E20C	74B1D29A	EAD54739	9DD277AF	04DB2615	73DC1683
88	E3630B12	94643B84	0D6D6A3E	7A6A5AA8	E40ECF0B	9309FF9D	0A00AE27	7D079EB1
90	F00F9344	8708A3D2	1E01F268	6906C2FE	F762575D	806567CB	196C3671	6E6B06E7
98	FED41B76	89D32BE0	10DA7A5A	67DD4ACC	F9B9DF6F	8EBEEFF9	17B7BE43	60B08ED5
A0	D6D6A3E8	A1D1937E	38D8C2C4	4FDFF252	D1BB67F1	A6BC5767	3FB506DD	48B2364B
A8	D80D2BDA	AF0A1B4C	36034AF6	41047A60	DF60EFC3	A867DF55	316E8EEF	4669BE79
B0	CB61B38C	BC66831A	256FD2A0	5268E236	CC0C7795	BB0B4703	220216B9	5505262F
B8	C5BA3BBE	B2BD0B28	2BB45A92	5CB36A04	C2D7FFA7	B5D0CF31	2CD99E8B	5BDEAE1D
C0	9B64C2B0	EC63F226	756AA39C	026D930A	9C0906A9	EB0E363F	72076785	05005713
C8	95BF4A82	E2B87A14	7BB12BAE	0CB61B38	92D28E9B	E5D5BE0D	7CDCEFB7	0BDBDF21
D0	86D3D2D4	F1D4E242	68DDB3F8	1FDA836E	81BE16CD	F6B9265B	6FB077E1	18B74777
D8	88085AE6	FF0F6A70	66063BCA	11010B5C	8F659EFF	F862AE69	616BFFD3	166CCF45
E0	A00AE278	D70DD2EE	4E048354	3903B3C2	A7672661	D06016F7	4969474D	3E6E77DB
E8	AED16A4A	D9D65ADC	40DF0B66	37D83BF0	A9BCAE53	DEBB9EC5	47B2CF7F	30B5FFE9
F0	BDBDF21C	CABAC28A	53B39330	24B4A3A6	BAD03605	CDD70693	54DE5729	23D967BF
F8	B3667A2E	C4614AB8	5D681B02	2A6F2B94	B40BBE37	C30C8EA1	5A05DF1B	2D02EF8D

2.3. ХЕШИРОВАНИЕ ДАННЫХ

Функция хэширования (хэш-функция) представляет собой преобразование, на вход которого подается сообщение переменной длины M, а выходом является строка фиксированной длины h(M). Иначе говоря, хэш-функция h() принимает в качестве аргумента сообщение (документ) M произвольной длины и возвращает хэш-значение (хэш) H = h(M) фиксированной длины.

Хэш-значение h(M) – это дайджест сообщения M, то есть сжатое двоичное представление основного сообщения M произвольной длины. Хеш-значение h(M) формируется функцией хеширования.

Функция хеширования позволяет сжать подписываемый документ M до 128 бит и более (в частности, 128 или 256 бит), тогда как M может быть размером в

мегабайты или более. Следует отметить, что значение хеш-функции h(M) зависит сложным образом от документа M и не позволяет восстановить сам документ M.

Функция хеширования должна обладать следующими свойствами:

1) Хеш-функция может быть применена к аргументу любого размера.
2) Выходное значение хеш-функции имеет фиксированный размер.
3) Хеш-функцию $h(x)$ достаточно просто вычислить для любого x. Скорость вычисления хэш-функции должна быть такой, чтобы скорость выработки и проверки электронной цифровой подписи (ЭЦП) при использовании хеш-функции была значительно больше, чем при использовании самого сообщения.
4) Хеш-функция должна быть чувствительна к всевозможным изменениям в тексте M, таким как вставки, выбросы, перестановки и т.п.
5) Хеш-функция должна быть однонаправленной, то есть обладать свойством необратимости. Иными словами, задача подбора документа M', который обладал бы требуемым значением хеш-функции, должна быть вычислительно неразрешима.
6) Вероятность того, что значения хеш-функций двух различных документов (вне зависимости от их длин) совпадут, должна быть ничтожно мала: то есть для любого фиксированного x с вычислительной точки зрения невозможно найти $x' \neq x$, такое что $h(x') = h(x)$.

Теоретически возможно, что два различных сообщения могут быть сжаты в одну и ту же свертку (так называемая коллизия, или столкновение). Поэтому для обеспечения стойкости функции хеширования необходимо предусмотреть способ избегать столкновений. Полностью столкновений избежать нельзя, поскольку в общем случае количество возможных сообщений превышает количество возможных выходных значений функции хеширования. Однако вероятность столкновения должна быть низкой.

Свойство 5 эквивалентно тому, что h() является односторонней функцией. Свойство 6 гарантирует, что не может быть найдено другое сообщение, дающее ту же свертку. Это предотвращает фальсификацию сообщения.

Таким образом, функция хеширования может использоваться для обнаружения изменений сообщения, то есть она может служить для формирования криптографической контрольной суммы (также называемой кодом обнаружения изменений или кодом аутентификации сообщения). В этом качестве хеш-функция используется для контроля целостности сообщения, при формировании и проверке электронной цифровой подписи. Хеш-функции широко используются также в целях аутентификации пользователей.

Некоторые функции хеширования:

- **MD (Message Digest)** – ряд алгоритмов хеширования, наиболее распространенных в мире. Каждый из них вырабатывает 128-битовый хеш-код. Алгоритм MD2 – самый медленный из них, MD4 – самый быстрый. Алгоритм MD5 является модификацией MD4, при которой пожертвовали скоростью ради увеличения безопасности. Алгоритм MD5 применяется в последних версиях Microsoft Windows для преобразования пароля пользователя в 16-байтовое число;
- **SHA (Secure Hash Algorithm)** – это алгоритм вычисления дайджеста сообщений, вырабатывающий 160-битовый хеш-код входных данных. Широко распространен в мире, используется во многих сетевых протоколах защиты информации;

- **Российский стандарт ГОСТ Р34.11-94**. Вычисляет хеш-функцию размером 32 байт.
- **RIPEMD-320** – хэш-функция разработанная Хансом Доббертином, Антоном Боселаерсом и Бартом Принилом в 1996 году. Размер хэша — 320 бит. Размер блока входных данных – 512 бит. Уязвимостей на текущий момент не обнаружено.

Хеш-функции широко используются также для аутентификации пользователей. Существует множество криптографических протоколов, основанных на применении хеш-функций.

Хэш-функция MD5

Рассмотрим алгоритм получения *дайджеста сообщения MD5* (RFC 1321), разработанный Роном Ривестом из MIT.

Алгоритм получает на входе сообщение произвольной длины и создает в качестве выхода дайджест сообщения длиной 128 бит, рис. 2.5.

Алгоритм использует при своей работе две константы:

1) Входной вектор (IV) начальной инициализации MD-буфера, состоящий из четырех 32-битных величин a – 01234567, b – 89ABCDEF, c – FEDCBA98, d – 76543210;
2) Массива T[0..63], i-ый элемент которого, обозначаемый T[i], имеет 32-битное значение, равное целой части от $2^{32} * abs(\sin(i))$, i – задается в радианах.

Четыре функции:

- f_F = F(b,c,d) = (B and C) or ((not B) and D);
- f_G = G(b,c,d) = (B and D) or (C and (not D));
- f_H = H(b,c,d) = B ⊕ C ⊕ D;
- f_I = I(b,c,d) = C ⊕ (B and (not D));

где A, B, C, D – 32-битные регистры промежуточного 128-битного MD-буфера.

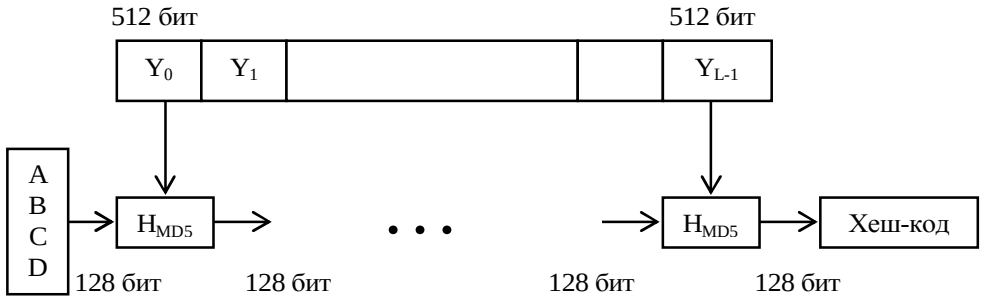


Рис. 2.5. Схема получения дайджеста по алгоритму MD5.

Алгоритм состоит из следующих шагов:

Шаг 1: Добавление недостающих битов.

Формируется расширенное сообщение путем добавления к исходному сообщению битов таким образом, чтобы его длина стала равна 448 по модулю 512 (mod 512). Это означает, что длина добавленного сообщения на 64 бита меньше, чем число, кратное 512. Добавление производится всегда, даже если сообщение имеет нужную длину. Например, если длина сообщения 448 битов,

оно дополняется 512 битами до 960 битов. Таким образом, число добавляемых битов находится в диапазоне от 1 до 512. Добавление состоит из единицы, за которой следует необходимое количество нулей.

Шаг 2: Добавление длины

Окончательно расширенное сообщение получается добавлением 64-битного представления длины исходного сообщения в битах присоединением к результату первого шага. Если первоначальная длина больше, чем 2^{64}, то используются только последние 64 бита. Таким образом, поле содержит длину исходного сообщения по модулю 2^{64} (mod 2^{64}). В результате первых двух шагов создается расширенное сообщение, длина которого кратна 512 битам. Это расширенное сообщение представляется как последовательность 512-битных блоков $Y_0, Y_1, \ldots, Y_{L-1}$, при этом общая длина расширенного сообщения равна L*512 битам. Таким образом, длина полученного расширенного сообщения кратна шестнадцати 32-битным словам. Каждый блок во время вычисления, представляется как массив X[0..15] из 16 32-битных значений.

Шаг 3: Инициализация MD-буфера

Используется 128-битный буфер для хранения промежуточных и окончательных результатов *хэш-функции*. Буфер может быть представлен как четыре 32-битных регистра (A, B, C, D). Эти регистры инициализируются 32-битными числами: *a, b, c* и *d*, соответственно.

Шаг 4: Обработка последовательности 512-битных блоков

Основой алгоритма является модуль, состоящий из четырех циклических обработок, называемых раундами. Четыре раунда имеют похожую структуру, рис. 2.6, но каждый использует свою элементарную логическую функцию, обозначаемые f_F, f_G, f_H, f_I.

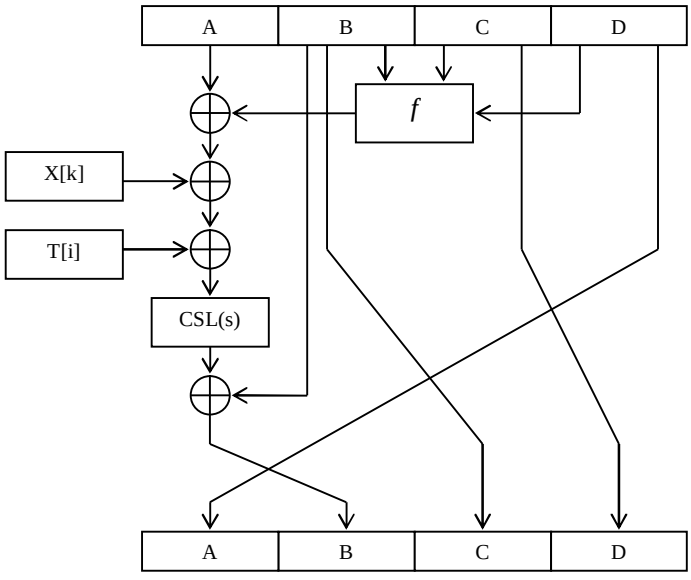


Рис. 2.6. Схема раундов MD5

Каждый раунд принимает в качестве входа текущий 512-битный блок Y_q, обрабатывающийся в данный момент, и 128-битное значение буфера ABCD,

которое является промежуточным значением *дайджеста*, и изменяет содержимое этого буфера. На время обработки очередного блока Y_q, значения MD-буфера предыдущего блока Y_{q-1} сохраняются в промежуточных переменных: AA = A, BB = B, CC = C, DD = D (для нулевого блока сохраняются значения вектора инициализации IV).

Каждый раунд состоит из 16 операторов. Все операторы однотипны и имеют вид: [abcd k s i], определяемый как $a = b \oplus \mathrm{CSL}((a \oplus Fun(b,c,d) \oplus X[k] \oplus T[i]), s)$, где X – блок данных. X[k] = M [n * 16 + k], где k – номер 32-битного слова из n-го 512-битного блока сообщения, и s – циклический сдвиг влево на s бит полученого 32-битного аргумента. После каждого шага цикла происходит циклический сдвиг влево (от младшего к старшему) четырех слов A, B, C и D.

Раунд 1 {[abcd k s i] a = b ⊕ CSL((a ⊕ F(b,c,d) ⊕ X[k] ⊕ T[i]), s)}

[ABCD 0 7 1] [DABC 1 12 2] [CDAB 2 17 3] [BCDA 3 22 4]
[ABCD 4 7 5] [DABC 5 12 6] [CDAB 6 17 7] [BCDA 7 22 8]
[ABCD 8 7 9] [DABC 9 12 10] [CDAB 10 17 11] [BCDA 11 22 12]
[ABCD 12 7 13] [DABC 13 12 14] [CDAB 14 17 15] [BCDA 15 22 16]

Раунд 2 {[abcd k s i] a = b ⊕ CSL((a ⊕ G(b,c,d) ⊕ X[k] ⊕ T[i]),s)}

[ABCD 1 5 17] [DABC 6 9 18] [CDAB 11 14 19] [BCDA 0 20 20]
[ABCD 5 5 21] [DABC 10 9 22] [CDAB 15 14 23] [BCDA 4 20 24]
[ABCD 9 5 25] [DABC 14 9 26] [CDAB 3 14 27] [BCDA 8 20 28]
[ABCD 13 5 29] [DABC 2 9 30] [CDAB 7 14 31] [BCDA 12 20 32]

Раунд 3 {[abcd k s i] a = b ⊕ CSL((a ⊕ H(b,c,d) ⊕ X[k] + T[i]), s)}

[ABCD 5 4 33] [DABC 8 11 34] [CDAB 11 16 35] [BCDA 14 23 36]
[ABCD 1 4 37] [DABC 4 11 38] [CDAB 7 16 39] [BCDA 10 23 40]
[ABCD 13 4 41] [DABC 0 11 42] [CDAB 3 16 43] [BCDA 6 23 44]
[ABCD 9 4 45] [DABC 12 11 46] [CDAB 15 16 47] [BCDA 2 23 48]

Раунд 4 {[abcd k s i] a = b ⊕ CSL((a ⊕ I(b,c,d) ⊕ X[k] ⊕ T[i]), s)}

[ABCD 0 6 49] [DABC 7 10 50] [CDAB 14 15 51] [BCDA 5 21 52]
[ABCD 12 6 53] [DABC 3 10 54] [CDAB 10 15 55] [BCDA 1 21 56]
[ABCD 8 6 57] [DABC 15 10 58] [CDAB 6 15 59] [BCDA 13 21 60]
[ABCD 4 6 61] [DABC 11 10 62] [CDAB 2 15 63] [BCDA 9 21 64]

После завершения четвертого раунда выполняется суммирование «по модулю 2» с сохраненными значениями для предыдущего q-1 блока (MD_{q-1}):

A = AA ⊕ A; B = BB ⊕ B; C = CC ⊕ C; D = DD ⊕ D.

Результатом является очередное промежуточное хеш-значение MD_q.

Шаг 5. Результат вычислений находится в буфере ABCD, это и есть конечное хеш-значение сообщения MD (A – младший байт).

Хеш содержит 128 бит (16 байт) и обычно представляется как последовательность из 32 шестнадцатеричных цифр.

Алгоритм *MD5* можно суммировать следующим образом:

$\mathbf{MD_0 = IV,}$

$\mathbf{MD_{q+1} = MD_q + f_I[Y_q, f_H[Y_q, f_G[Y_q, f_F[Y_q, MD_q]]]],}$

$\mathbf{MD = MD_{L-1},}$

где `IV` – начальное значение буфера `ABCD (input vector)`, определенное на шаге 3,

Y_q – q-ый 512-битный блок сообщения,

`L` – число блоков в сообщении (включая поля дополнения и длины),
`MD` – окончательное значение дайджеста сообщения.

Примеры хешей MD5:

1) MD5("md5") = 1bc29b36f623ba82aaf6724fd3b16718
2) MD5("md4") = c93d3bf7a7c4afe94b64e30c2ce39f4f

Даже небольшое изменение входного сообщения (в нашем случае на один бит: ASCII символ «5» с кодом $0x35_{16}$ = 000110101_2 заменяется на символ «4» с кодом $0x34_{16}$ = 000110100_2) приводит к полному изменению хеша. Такое свойство алгоритма называется лавинным эффектом.

Пустая строка имеет вполне определенный хеш.

3) MD5("") = d41d8cd98f00b204e9800998ecf8427e

Хэш-функция SHA-1

Безопасный хэш-алгоритм (Secure Hash Algorithm) был разработан национальным институтом стандартов и технологии (NIST) и опубликован в качестве федерального информационного стандарта (FIPS PUB 180) в 1993 году. SHA-1, как и MD5, основан на алгоритме MD4.

Алгоритм получает на входе сообщение максимальной длины 2^{64} бит и создает в качестве выхода дайджест сообщения длиной 160 бит, рис. 2.7.

Алгоритм использует при своей работе:

- пять 32-битных значений входного вектора (IV) начальной инициализации SHA-буфера: a – 67452301, b – EFCDAB89, c – 98BADCFE, d – 10325476, e – C3D2E1F0;
- четыре функции и соответствующих им четыре константы для 80 раундов вычислений:

$0 \le t \le 19$	$F_t(b,c,d) =$ (b and c) or ((not b) and d)	$K_t =$ 5A827999
$20 \le t \le 39$	$F_t(b,c,d) =$ b ⊕ c ⊕ d	$K_t =$ 6ED9EBA1
$40 \le t \le 59$	$F_t(b,c,d) =$ (b and c) or (b and d) or (c and d)	$K_t =$ 8F1BBCDC
$60 \le t \le 79$	$F_t(b,c,d) =$ b ⊕ c ⊕ d	$K_t =$ CA62C1D6

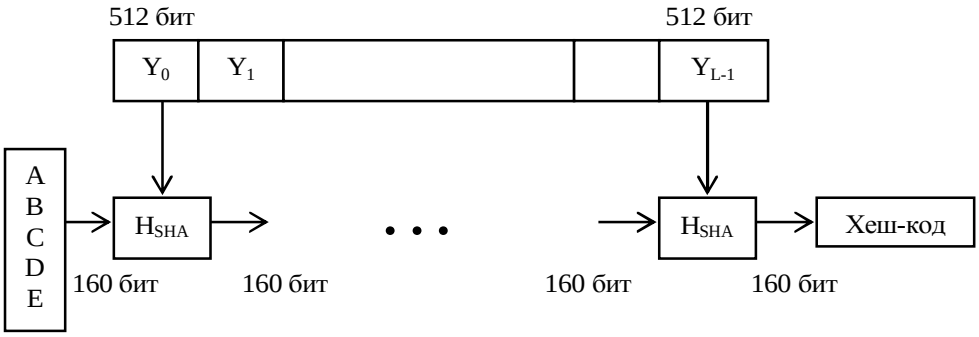


Рис. 2.7. Схема получения дайджеста по алгоритму SHA-1.

SHA-1 реализует хеш-функцию, построенную на идее функции сжатия. Входами функции сжатия являются блок сообщения длиной 512 бит и выход предыдущего блока сообщения. Выход представляет собой значение всех хеш-

блоков до этого момента. Иными словами хеш блока Y_q равен $h_q = f(Y_q, h_{q-1})$. Хеш-значением всего сообщения является выход последнего блока.

Алгоритм состоит из следующих шагов:

Шаг 1: Добавление недостающих битов

Формируется расширенное сообщение путем добавления к исходному сообщению битов таким образом, чтобы его длина стала равна 448 по модулю 512 (mod 512). Это означает, что длина добавленного сообщения на 64 бита меньше, чем число, кратное 512. Добавление производится всегда, даже если сообщение имеет нужную длину. Например, если длина сообщения 448 битов, оно дополняется 512 битами до 960 битов. Таким образом, число добавляемых битов находится в диапазоне от 1 до 512.

Добавление состоит из единицы, за которой следует необходимое количество нулей.

Шаг 2: Добавление длины

Окончательно расширенное сообщение получается добавлением 64-битного представления длины исходного сообщения в битах присоединением к результату первого шага. Если первоначальная длина больше, чем 2^{64}, то используются только последние 64 бита. Таким образом, поле содержит длину исходного сообщения по модулю 2^{64} (mod 2^{64}).

В результате первых двух шагов создается расширенное сообщение, длина которого кратна 512 битам. Это расширенное сообщение представляется как последовательность 512-битных блоков $Y_0, Y_1, \ldots, Y_{L-1}$, при этом общая длина расширенного сообщения равна L*512 битам. Таким образом, длина полученного расширенного сообщения кратна шестнадцати 32-битным словам.

Шаг 3: Инициализация SHA-буфера

Используется 160-битный буфер для хранения промежуточных и окончательных результатов *хэш-функции*. Буфер может быть представлен как пять 32-битных регистра (A, B, C, D, E). Эти регистры инициализируются 32-битными числами: *a, b, c, d* и *e*, соответственно.

Шаг 4: Обработка последовательности 512-битных блоков

Основой алгоритма является модуль, итеративно обрабатывающий каждый 512-битный блок, рис. 2.7. Итерация состоит из четырех этапов по двадцать операций в каждом. Блок сообщения преобразуется из 16 32-битовых слов M_i в 80 32-битовых слов W_j по следующему правилу:

$W_t = M_t$ при $0 \le t \le 15$

$W_t = \mathrm{CSL}((W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}), 1)$ при $16 \le t \le 79$,

для t от 0 до 79

$\mathrm{temp} = \mathrm{CSL}(a,5) \oplus F_t(b,c,d) \oplus e \oplus W_t \oplus K_t$

e = d

d = c

c = CSL(b,30)

b = a

a = temp

После этого a, b, c, d, e прибавляются к A, B, C , D , E соответственно. Начинается следующая итерация.

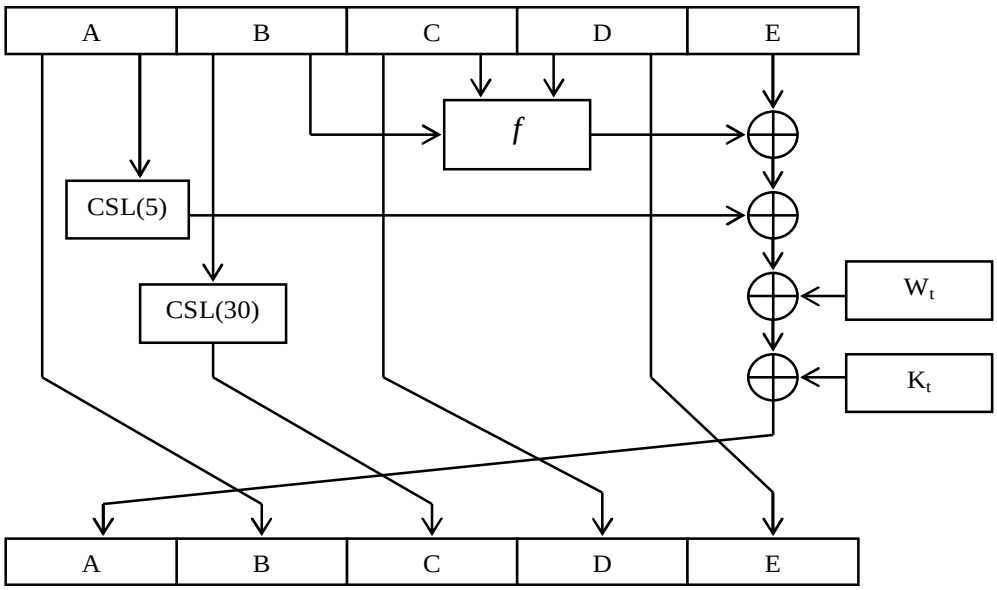


Рис. 2.7. Одна итерация алгоритма SHA-1

Шаг 5. Результат вычислений находится в буфере ABCDE, это и есть конечное хеш-значение сообщения SHA (A – младший байт).

Хеш содержит 160 бит (20 байт) и обычно представляется как последовательность из 40 шестнадцатеричных цифр.

Алгоритм *SHA-1* можно суммировать следующим образом:

$\mathbf{SHA_0 = IV,}$

$\mathbf{SHA_{q+1} = S32(SHA_q, ABCDE_q),}$

$\mathbf{SHA = SHA_{L-1},}$

где IV – начальное значение буфера ABCDE,

$ABCDE_q$ – результат обработки q-того блока сообщения,

L – число блоков в сообщении, включая поля добавления и длины,

Σ32 – сумма по модулю 2^{32}, выполняемая отдельно для каждого слова буфера,

SHA – значение дайджеста сообщения.

Примеры хешей SHA-2:

1) SHA-1("В чащах юга жил бы цитрус? Да, но фальшивый экземпляр!")
   = 9e32295f 8225803b b6d5fdfc c0674616 a4413c1b
2) SHA-1("sha")
   = d8f45903 20e1343a 915b6394 170650a8 f35d6926

Небольшое изменение исходного текста (одна буква в верхнем регистре) приводит к сильному изменению самого хеша. Это происходит вследствие лавинного эффекта.

3) SHA-1("Sha")
   = ba79baeb 9f10896a 46ae7471 5271b7f5 86e74640
4) SHA-1("")
   = da39a3ee 5e6b4b0d 3255bfef 95601890 afd80709

ЧАСТЬ 3. ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

Современные методы обработки, передачи и накопления информации способствовали появлению угроз, связанных с возможностью потери, искажения и раскрытия данных, адресованных или принадлежащих конечным пользователям. Поэтому обеспечение информационной безопасности компьютерных систем и сетей является одним из ведущих направлений развития информационных технологий.

3.3. ОСНОВНЫЕ ПОНЯТИЯ

Защита информации – это деятельность по предотвращению утечки защищаемой информации, несанкционированных, преднамеренных и непреднамеренных воздействий на защищаемую информацию.

Объект защиты – информация, или носитель информации, или информационный процесс, в отношении которого необходимо обеспечивать защиту в соответствии с поставленной целью защиты информации.

Цель защиты информации – это желаемый результат защиты информации. Целью защиты информации может быть предотвращение ущерба собственнику, владельцу, пользователю информации в результате возможной утечки информации и/или несанкционированного, преднамеренного и непреднамеренного воздействия на информацию.

Эффективность защиты информации – степень соответствия результатов защиты информации поставленной цели.

Защита информации от утечки – деятельность по предотвращению неконтролируемого распространения защищаемой информации от ее разглашения, несанкционированного доступа к защищаемой информации и от получения защищаемой информации злоумышленниками.

Защита информации от несанкционированного воздействия – деятельность по предотвращению воздействия на защищаемую информацию с нарушением установленных прав и/или правил на изменение информации, приводящего к искажению, уничтожению, копированию, блокированию доступа к информации, а также к утрате, уничтожению или сбою функционирования носителя информации.

Защита информации от непреднамеренного воздействия – деятельность по предотвращению воздействия на защищаемую информацию ошибок пользователя информацией, сбоя технических и программных средств информационных систем, а также природных явлений или иных нецеленаправленных на изменение информации воздействий, связанных с функционированием технических средств, систем или с деятельностью людей и приводящих к искажению, уничтожению, копированию, блокированию доступа к информации, а также к утрате, уничтожению или сбою функционирования носителя информации.

Защита информации от разглашения – деятельность по предотвращению несанкционированного доведения защищаемой информации до неконтролируемого количества получателей информации.

Защита информации от несанкционированного доступа (НСД) – деятельность по предотвращению получения защищаемой информации заинтересованным субъектом с нарушением установленных правовыми документами или собственником, владельцем информации прав или правил доступа

к защищаемой информации. Заинтересованным субъектом, осуществляющим несанкционированный доступ к защищаемой информации, может выступать государство, юридическое лицо, группа физических лиц, в том числе общественная организация, отдельное физическое лицо.

Система защиты информации – совокупность органов и/или исполнителей, используемая ими техника защиты информации, а также объекты защиты, организованные и функционирующие по правилам, установленным соответствующими правовыми, организационно-распорядительными и нормативными документами по защите информации.

Под **информационной безопасностью** понимают защищенность информации от незаконного ознакомления, преобразования и уничтожения, а также защищенность информационных ресурсов от воздействий, направленных на нарушение их работоспособности. Природа этих воздействий может быть самой разнообразной. Это и попытки проникновения злоумышленников, и ошибки персонала, и выход из строя аппаратных и программных средств, стихийные бедствия (землетрясение, ураган, пожар и т.п.).

Современная автоматизированная система обработки информации (АС) представляет собой сложную систему, состоящую из большого числа компонентов различной степени автономности, которые связаны между собой и обмениваются данными. Практически каждый компонент может подвергнуться внешнему воздействию или выйти из строя. Компоненты АС можно разбить на следующие группы:

1) аппаратные средства – компьютеры и их составные части (процессоры, мониторы, терминалы, периферийные устройства – дисководы, принтеры, контроллеры, кабели, линии связи и т.д.);
2) программное обеспечение – приобретенные программы, исходные, объектные, загрузочные модули; операционные системы и системные программы (компиляторы, компоновщики и др.), утилиты, диагностические программы и т.д.;
3) данные – хранимые временно и постоянно, на магнитных носителях, печатные архивы, системные журналы и т.д.;
4) персонал – обслуживающий персонал и пользователи.

Одной из особенностей обеспечения информационной безопасности в АС является то, что таким абстрактным понятиям, как информация, объекты и субъекты системы, ставятся в соответствие физические представления в компьютерной среде:

- для представления информации – машинные носители информации в виде внешних устройств компьютерных систем (терминалов, печатающих устройств, различных накопителей, линий и каналов связи), оперативной памяти, файлов, записей и т.д.;
- под объектами системы понимают пассивные компоненты системы, хранящие, принимающие или передающие информацию. Доступ к объекту означает доступ к содержащейся в нем информации;
- под субъектами системы понимают активные компоненты системы, которые могут стать причиной потока информации от объекта к субъекту или изменения состояния системы. В качестве субъектов могут выступать пользователи, активные программы и процессы.

Информационная безопасность компьютерных систем достигается обеспечением конфиденциальности, целостности и достоверности обрабатываемых данных, а также доступности и целостности информационных компонентов и ресурсов системы. Перечисленные выше базовые свойства информации нуждаются в более полном толковании.

Конфиденциальность данных – это статус, предоставленный данным и определяющий требуемую степень их защиты. К конфиденциальным данным можно отнести, например, следующие:

- личная информация пользователей;
- учетные записи (имена и пароли);
- данные о кредитных картах;
- данные о разработках и различные внутренние документы;
- бухгалтерские сведения.

Конфиденциальная информация должна быть известна только допущенным и прошедшим проверку (авторизованным) субъектам системы (пользователям, процессам, программам). Для остальных субъектов системы эта информация должна быть неизвестной.

Установление градаций важности защиты защищаемой информации (объекта защиты) называют категорированием защищаемой информации.

Под **целостностью информации** понимается свойство информации сохранять свою структуру и/или содержание в процессе передачи и хранения. Целостность информации обеспечивается в том случае, если данные в системе не отличаются в семантическом отношении от данных в исходных документах, то есть если не произошло их случайного или преднамеренного искажения или разрушения. Обеспечение целостности данных является одной из сложных задач защиты информации.

Достоверность информации – свойство информации, выражающееся в строгой принадлежности субъекту, который является ее источником, либо тому субъекту, от которого эта информация принята.

Юридическая значимость информации означает, что документ, являющийся носителем информации, обладает юридической силой.

Доступность данных. Работа пользователя с данными возможна только в том случае, если он имеет к ним доступ.

Доступ к информации – получение субъектом возможности ознакомления с информацией, в том числе при помощи технических средств. Субъект доступа к информации – участник правоотношений в информационных процессах.

Оперативность доступа к информации – это способность информации или некоторого информационного ресурса быть доступными для конечного пользователя в соответствии с его оперативными потребностями.

Собственник информации – субъект, в полном объеме реализующий полномочия владения, пользования, распоряжения информацией в соответствии с законодательными актами.

Владелец информации – субъект, осуществляющий владение и пользование информацией и реализующий полномочия распоряжения в пределах прав, установленных законом и/или собственником информации.

Пользователь информации – субъект, пользующийся информацией, полученной от ее собственника, владельца или посредника в соответствии с

установленными правами и правилами доступа к информации либо с их нарушением.

Право доступа к информации – совокупность правил доступа к информации, установленных правовыми документами или собственником, владельцем информации.

Правило доступа к информации – совокупность правил, регламентирующих порядок и условия доступа субъекта к информации и ее носителям.

Различают санкционированный и несанкционированный доступ к информации.

Санкционированный доступ к информации – это доступ к информации, не нарушающий установленные правила разграничения доступа. Правила разграничения доступа служат для регламентации права доступа к компонентам системы.

Несанкционированный доступ (НСД) к информации характеризуется нарушением установленных правил разграничения доступа. Лицо или процесс, осуществляющие несанкционированный доступ к информации, являются нарушителями правил разграничения доступа. Несанкционированный доступ является наиболее распространенным видом компьютерных нарушений.

Ответственным за защиту компьютерной системы от несанкционированного доступа к информации является администратор защиты.

Доступность информации подразумевает также доступность компонента или ресурса компьютерной системы, то есть свойство компонента или ресурса быть доступным для законных субъектов системы. Вот примерный перечень ресурсов, которые должны быть доступны: принтеры; серверы; рабочие станции; данные пользователей; любые критические данные, необходимые для работы.

Целостность ресурса или компонента системы – это свойство ресурса или компонента быть неизменными в семантическом смысле при функционировании системы в условиях случайных или преднамеренных искажений либо разрушающих воздействий.

С допуском к информации и ресурсам системы связана группа таких важных понятий, как идентификация, аутентификация, авторизация.

С каждым субъектом системы (сети) связывают некоторую информацию (число, строку символов), идентифицирующую субъект. Эта информация является идентификатором субъекта системы (сети). Субъект, имеющий зарегистрированный идентификатор, является законным (легальным) субъектом.

Идентификация субъекта – это процедура распознавания субъекта по его идентификатору. Идентификация выполняется при попытке субъекта войти в систему (сеть).

Следующим шагом взаимодействия системы с субъектом является аутентификация субъекта.

Аутентификация субъекта – это проверка подлинности субъекта с данным идентификатором. Процедура аутентификации устанавливает, является ли субъект именно тем, кем он себя объявил.

После идентификации и аутентификации субъекта выполняют процедуру авторизации.

Авторизация субъекта – это процедура предоставления законному субъекту, успешно прошедшему идентификацию и аутентификацию, соответствующих полномочий и доступных ресурсов системы (сети).

Под угрозой безопасности АС понимаются возможные действия, способные прямо или косвенно нанести ущерб ее безопасности. Ущерб безопасности подразумевает нарушение состояния защищенности информации, содержащейся и обрабатываемой в системе (сети).

С понятием угрозы безопасности тесно связано понятие уязвимости компьютерной системы (сети). Уязвимость компьютерной системы – это присущее системе неудачное свойство, которое может привести к реализации угрозы.

Атака на компьютерную систему – это поиск и/или использование злоумышленником той или иной уязвимости системы. Иными словами, атака – это реализация угрозы безопасности.

Противодействие угрозам безопасности является целью средств защиты компьютерных систем и сетей.

Защищенная система - это система со средствами защиты, которые успешно и эффективно противостоят угрозам безопасности.

Способ защиты информации - порядок и правила применения определенных принципов и средств защиты информации.

Средство защиты информации - техническое, программное средство, вещество и/или материал, предназначенные или используемые для защиты информации.

Комплекс средств защиты (КСЗ) представляет собой совокупность программных и технических средств, создаваемых и поддерживаемых для обеспечения информационной безопасности системы (сети). КСЗ создается и поддерживается в соответствии с принятой в данной организации политикой безопасности.

Техника защиты информации - средства защиты информации, средства контроля эффективности защиты информации, средства и системы управления, предназначенные для обеспечения защиты информации.

Корпоративные сети относятся к распределенным автоматизированным системам (АС), осуществляющим обработку информации. Обеспечение безопасности АС предполагает организацию противодействия любому несанкционированному вторжению в процесс функционирования АС, а также попыткам модификации, хищения, выведения из строя или разрушения ее компонентов, то есть защиту всех компонентов АС - аппаратных средств, программного обеспечения, данных и персонала. Конкретный подход к проблеме обеспечения безопасности основан на разработанной для АС политике безопасности.

Политика безопасности - это совокупность норм, правил и практических рекомендаций, регламентирующих работу средств защиты компьютерной системы от заданного множества угроз.

3.2. АНАЛИЗ УГРОЗ БЕЗОПАСНОСТИ

Под угрозой (в общем смысле) обычно понимают потенциально возможное событие (воздействие, процесс или явление), которое может привести к нанесению ущерба чьим-либо интересам. В дальнейшем рассмотрении под угрозой безопасности автоматизированной системы обработки информации будем понимать возможность воздействия на АС, которое прямо или косвенно может нанести ущерб ее безопасности.

В настоящее время известен достаточно обширный перечень угроз информационной безопасности АС, содержащий сотни позиций. Рассмотрение возможных угроз информационной безопасности проводится с целью определения полного набора требований к разрабатываемой системе защиты.

Перечень угроз, оценки вероятностей их реализации, а также модель нарушителя служат основой для анализа риска реализации угроз и формулирования требований к системе защиты АС. Кроме выявления возможных угроз, целесообразно проведение анализа этих угроз на основе их классификации по ряду признаков. Каждый из признаков классификации отражает одно из обобщенных требований к системе защиты. Угрозы, соответствующие каждому признаку классификации, позволяют детализировать отражаемое этим признаком требование.

Необходимость классификации угроз информационной безопасности АС обусловлена тем, что хранимая и обрабатываемая информация в современных АС подвержена воздействию чрезвычайно большого числа факторов, в силу чего становится невозможным формализовать задачу описания полного множества угроз. Поэтому для защищаемой системы обычно определяют не полный перечень угроз, а перечень классов угроз.

Классификация возможных угроз информационной безопасности АС может быть проведена по ряду базовых признаков:

1) По природе возникновения различают:
- естественные угрозы, вызванные воздействиями на АС объективных физических процессов или стихийных природных явлений;
- искусственные угрозы безопасности АС, вызванные деятельностью человека.

2) По степени преднамеренности проявления различают:
- угрозы, вызванные ошибками или халатностью персонала, например некомпетентное использование средств защиты; ввод ошибочных данных и т.п.;
- угрозы преднамеренного действия, например действия злоумышленников.

3) По непосредственному источнику угроз. Источниками угроз могут быть:
- природная среда, например стихийные бедствия, магнитные бури и пр.;
- человек, например вербовка путем подкупа персонала, разглашение конфиденциальных данных и т.п.;
- санкционированные программно-аппаратные средства, например удаление данных, отказ в работе операционной системы;
- несанкционированные программно-аппаратные средства, например заражение компьютера вирусами с деструктивными функциями.

4) По положению источника угроз. Источник угроз может быть расположен:
- вне контролируемой зоны АС, например перехват данных, передаваемых по каналам связи, перехват побочных электромагнитных, акустических и других излучений устройств;
- в пределах контролируемой зоны АС, например применение подслушивающих устройств, хищение распечаток, записей, носителей информации и т.п.;
- непосредственно в АС, например некорректное использование ресурсов АС.

5) По степени зависимости от активности АС. Угрозы проявляются:
- независимо от активности АС, например вскрытие шифров криптозащиты информации;

- только в процессе обработки данных, например угрозы выполнения и распространения программных вирусов.

6) По степени воздействия на АС различают:
- пассивные угрозы, которые при реализации ничего не меняют в структуре и содержании АС, например угроза копирования секретных данных;
- активные угрозы, которые при воздействии вносят изменения в структуру и содержание АС, например внедрение «троянских коней» и вирусов.

7) По этапам доступа пользователей или программ к ресурсам АС различают:
- угрозы, проявляющиеся на этапе доступа к ресурсам АС, например угрозы несанкционированного доступа в АС;
- угрозы, проявляющиеся после разрешения доступа к ресурсам АС, например угрозы несанкционированного или некорректного использования ресурсов АС.

8) По способу доступа к ресурсам АС различают:
- угрозы с использованием стандартного пути доступа к ресурсам АС, например незаконное получение паролей и других реквизитов разграничения доступа с последующей маскировкой под зарегистрированного пользователя;
- угрозы с использованием скрытого нестандартного пути доступа к ресурсам АС, например несанкционированный доступ к ресурсам АС путем использования недокументированных возможностей ОС.

9) По текущему месту расположения информации, хранимой и обрабатываемой в АС, различают:
- угрозы доступа к информации на внешних запоминающих устройствах, например несанкционированное копирование секретной информации с жесткого диска;
- угрозы доступа к информации в оперативной памяти, например чтение остаточной информации из оперативной памяти; доступ к системной области оперативной памяти со стороны прикладных программ;
- угрозы доступа к информации, циркулирующей в линиях связи, например незаконное подключение к линиям связи с последующим вводом ложных сообщений или модификацией передаваемых сообщений; незаконное подключение к линиям связи с целью прямой подмены законного пользователя с последующим вводом дезинформации и навязыванием ложных сообщений;
- угрозы доступа к информации, отображаемой на терминале или печатаемой на принтере, например запись отображаемой информации на скрытую видеокамеру.

Как уже отмечалось, опасные воздействия на АС подразделяют на случайные и преднамеренные. Анализ опыта проектирования, изготовления и эксплуатации АС показывает, что информация подвергается различным случайным воздействиям на всех этапах цикла жизни и функционирования АС.

Причинами случайных воздействий при эксплуатации АС могут быть:
- аварийные ситуации из-за стихийных бедствий и отключений электропитания;
- отказы и сбои аппаратуры;
- ошибки в программном обеспечении;
- ошибки в работе обслуживающего персонала и пользователей;

– помехи в линиях связи из-за воздействий внешней среды.

Ошибки в программном обеспечении (ПО) являются распространенным видом компьютерных нарушений. Программное обеспечение серверов, рабочих станций, маршрутизаторов и т.д. написано людьми, поэтому оно практически всегда содержит ошибки. Чем выше сложность подобного программного обеспечения, тем больше вероятность обнаружения в нем ошибок и уязвимостей. Большинство из них не представляет никакой опасности, некоторые же могут привести к серьезным последствиям, таким как получение злоумышленником контроля над сервером, неработоспособность сервера, несанкционированное использование ресурсов (использование компьютера в качестве плацдарма для атаки и т.п.). Обычно подобные ошибки устраняются с помощью пакетов обновлений, регулярно выпускаемых производителем ПО. Своевременная установка таких пакетов является необходимым условием безопасности информации.

Преднамеренные угрозы связаны с целенаправленными действиями нарушителя.

В качестве нарушителя могут выступать служащий, посетитель, конкурент, наемник и т.д. Действия нарушителя могут быть обусловлены разными мотивами: недовольством служащего своей карьерой, сугубо материальным интересом (взятка), любопытством, конкурентной борьбой, стремлением самоутвердиться любой ценой и т.п.

Исходя из возможности возникновения наиболее опасной ситуации, обусловленной действиями нарушителя, можно составить гипотетическую модель потенциального нарушителя:

– квалификация нарушителя может быть на уровне разработчика данной системы;
– нарушителем может быть как постороннее лицо, так и законный пользователь системы;
– нарушителю известна информация о принципах работы системы;
– нарушитель выберет наиболее слабое звено в защите.

В частности, для банковских АС можно выделить следующие преднамеренные угрозы:

– несанкционированный доступ посторонних лиц, не принадлежащих к числу банковских служащих, и ознакомление с хранимой конфиденциальной ин формацией;
– ознакомление банковских служащих с информацией, к которой они не должны иметь доступ;
– несанкционированное копирование программ и данных;
– кража магнитных носителей, содержащих конфиденциальную информацию;
– кража распечатанных банковских документов;
– умышленное уничтожение информации;
– несанкционированная модификация банковскими служащими финансовых документов, отчетности и баз данных;
– фальсификация сообщений, передаваемых по каналам связи;
– отказ от авторства сообщения, переданного по каналам связи;
– отказ от факта получения информации;
– навязывание ранее переданного сообщения;
– разрушение информации, вызванное вирусными воздействиями;

- разрушение архивной банковской информации, хранящейся на магнитных носителях;
- кража оборудования.

Наиболее распространенным и многообразным видом компьютерных нарушений является несанкционированный доступ (НСД). Суть НСД состоит в получении пользователем (нарушителем) доступа к объекту в нарушение правил разграничения доступа, установленных в соответствии с принятой в организации политикой безопасности. НСД использует любую ошибку в системе защиты и возможен при нерациональном выборе средств защиты, их некорректной установке и настройке. НСД может быть осуществлен как штатными средствами АС, так и специально созданными аппаратными и программными средствами.

Перечислим основные каналы несанкционированного доступа, через которые нарушитель может получить доступ к компонентам АС и осуществить хищение, модификацию и/или разрушение информации:

- штатные каналы доступа к информации (терминалы пользователей, оператора, администратора системы; средства отображения и документирования ин формации; каналы связи) при их использовании нарушителями, а также за конными пользователями вне пределов их полномочий;
- технологические пульты управления;
- линии связи между аппаратными средствами АС;
- побочные электромагнитные излучения от аппаратуры, линий связи, сетей электропитания и заземления и др.

Из всего разнообразия способов и приемов несанкционированного доступа остановимся на следующих распространенных и связанных между собой нарушениях:

- перехват паролей;
- «маскарад»;
- незаконное использование привилегий.

Перехват паролей осуществляется специально разработанными программами. При попытке законного пользователя войти в систему программа-перехватчик имитирует на экране дисплея ввод имени и пароля пользователя, которые сразу пересылаются владельцу программы-перехватчика, после чего на экран выводится сообщение об ошибке и управление возвращается операционной системе. Пользователь предполагает, что допустил ошибку при вводе пароля. Он повторяет ввод и получает доступ в систему. Владелец программы-перехватчика, получивший имя и пароль законного пользователя, может теперь использовать их в своих целях. Существуют и другие способы перехвата паролей.

«Маскарад» – это выполнение каких-либо действий одним пользователем от имени другого пользователя, обладающего соответствующими полномочиями. Целью «маскарада» является приписывание каких-либо действий другому пользователю либо присвоение полномочий и привилегий другого пользователя. Примерами реализации «маскарада» являются:

- вход в систему под именем и паролем другого пользователя (этому «маска раду» предшествует перехват пароля);
- передача сообщений в сети от имени другого пользователя.

«Маскарад» особенно опасен в банковских системах электронных платежей, где неправильная идентификация клиента из-за «маскарада» злоумышленника может привести к большим убыткам законного клиента банка.

Незаконное использование привилегий. Большинство систем защиты устанавливает определенные наборы привилегий для выполнения заданных функций. Каждый пользователь получает свой набор привилегий: обычные пользователи – минимальный, администраторы – максимальный. Несанкционированный захват привилегий, например посредством «маскарада», приводит к возможности выполнения нарушителем определенных действий в обход системы защиты. Следует отметить, что незаконный захват привилегий возможен либо при наличии ошибок в системе защиты, либо из-за халатности администратора при управлении системой и назначении привилегий.

Для именования некоторых распространенных угроз безопасности АС употребляются специфические названия «троянский конь», «вирус», «сетевой червь». Дадим краткую характеристику этих распространенных угроз безопасности АС.

«Троянский конь» представляет собой программу, которая наряду с действиями, описанными в ее документации, выполняет некоторые другие действия, ведущие к нарушению безопасности системы и деструктивным результатам. Аналогия такой программы с древнегреческим «троянским конем» вполне оправдана, так как в обоих случаях не вызывающая подозрений оболочка таит серьезную угрозу. Радикальный способ защиты от этой угрозы заключается в создании замкнутой среды исполнения программ, которые должны храниться и защищаться от несанкционированного доступа.

Компьютерный вирус представляет собой своеобразное явление, возникшее в процессе развития компьютерной и информационной техники. Суть этого явления состоит в том, что программы-вирусы обладают рядом свойств, присущих живым организмам, – они рождаются, размножаются и умирают. Термин «вирус» в применении к компьютерам был предложен Фредом Коэном из Университета Южной Калифорнии. Исторически первое определение, данное Ф. Коэном, было следующим: «Компьютерный вирус – это программа, которая может заражать другие программы, модифицируя их посредством включения в них своей, возможно, измененной копии, причем последняя сохраняет способность к дальнейшему размножению». Компьютерные вирусы наносят ущерб системе за счет быстрого размножения и разрушения среды обитания. Сетевой червь является разновидностью программы-вируса, которая распространяется по глобальной сети. Следует отметить, что «троянские кони» и компьютерные вирусы относятся к весьма опасным угрозам АС.

Принято считать, что вне зависимости от конкретных видов угроз или их проблемно-ориентированной классификации АС удовлетворяет потребности эксплуатирующих ее лиц, если обеспечиваются следующие важные свойства информации и систем ее обработки: конфиденциальность, целостность и доступность информации.

Иными словами, в соответствии с существующими подходами считают, что информационная безопасность АС обеспечена в случае, если для информационных ресурсов в системе поддерживаются определенные уровни:

- конфиденциальности (невозможности несанкционированного получения какой-либо информации);

- целостности (невозможности несанкционированной или случайной модификации информации);
- доступности (возможности за разумное время получить требуемую информацию).

Соответственно, для автоматизированных систем рассматривают три основных вида угроз:

- угрозы нарушения конфиденциальности, направленные на разглашение конфиденциальной или секретной информации. При реализации этих угроз информация становится известной лицам, которые не должны иметь к ней доступ. В терминах компьютерной безопасности, угроза нарушения конфиденциальности имеет место всякий раз, когда получен несанкционированный доступ к некоторой закрытой информации, хранящейся в компьютерной системе или передаваемой от одной системы к другой;
- угрозы нарушения целостности информации, хранящейся в компьютерной системе или передаваемой по каналу связи, которые направлены на ее изменение либо искажение, приводящее к нарушению ее качества или полному уничтожению. Целостность информации может быть нарушена преднамеренно злоумышленником, а также в результате объективных воздействий со стороны среды, окружающей систему. Эта угроза особенно актуальна для систем передачи информации – компьютерных сетей и систем телекоммуникаций. Умышленные нарушения целостности информации не следует путать с ее санкционированным изменением, которое выполняется полномочными лицами с обоснованной целью (например, таким изменением является периодическая коррекция баз данных);
- угрозы нарушения работоспособности (отказ в обслуживании), направленные на создание таких ситуаций, когда определенные преднамеренные действия либо снижают работоспособность АС, либо блокируют доступ к некоторым ее ресурсам. Например, если один пользователь системы запрашивает доступ к некоторой службе, а другой предпринимает действия по блокированию этого доступа, то первый пользователь получает отказ в обслуживании. Блокирование доступа к ресурсу может быть постоянным или временным.

Данные виды угроз можно считать первичными, или непосредственными, поскольку реализация этих угроз ведет к непосредственному воздействию на защищаемую информацию.

Для современных информационных технологий подсистемы защиты являются неотъемлемой частью АС обработки информации. Атакующая сторона должна преодолеть эту подсистему защиты, чтобы нарушить, например, конфиденциальность АС. Однако нужно сознавать, что не существует абсолютно стойкой системы защиты, вопрос лишь во времени и средствах, требуемых на ее преодоление. Исходя из данных условий, рассмотрим следующую модель: защита информационной системы считается преодоленной, если в ходе исследования этой системы определены все ее уязвимости.

Преодоление защиты также представляет собой угрозу, поэтому для защищенных систем можно рассматривать четвертый вид угрозы – угрозу раскрытия параметров АС, включающей в себя подсистему защиты. На практике

любое проводимое мероприятие предваряется этапом разведки, в ходе которого определяются основные параметры системы, ее характеристики и т.п. Результатом этого этапа является уточнение поставленной задачи, а также выбор наиболее оптимального технического средства.

Угрозу раскрытия параметров АС можно считать опосредованной угрозой. Последствия ее реализации не причиняют какой-либо ущерб обрабатываемой информации, но дают возможность реализовать первичные, или непосредственные, угрозы, перечисленные выше.

При рассмотрении вопросов защиты АС целесообразно использовать четырех-уровневую градацию доступа к хранимой, обрабатываемой и защищаемой АС информации. Такая градация доступа поможет систематизировать как возможные угрозы, так и меры по их нейтрализации и парированию, то есть поможет систематизировать весь спектр методов обеспечения защиты, относящихся к информационной безопасности.

Это следующие уровни доступа:

- уровень носителей информации;
- уровень средств взаимодействия с носителем;
- уровень представления информации;
- уровень содержания информации.

Введение данных уровней обусловлено следующими соображениями.

Во-первых, информация для удобства манипулирования чаще всего фиксируется на некотором материальном носителе, которым может быть дискета или что-нибудь подобное.

Во-вторых, если способ представления информации таков, что она не может быть непосредственно воспринята человеком, возникает необходимость в преобразователях информации в доступный для человека способ представления. Например, для чтения информации с дискеты необходим компьютер, оборудованный дисководом соответствующего типа.

В-третьих, как уже было отмечено, информация может быть охарактеризована способом своего представления, или тем, что еще называется языком в обиходном смысле. Язык символов, язык жестов и т.п. – все это способы представления информации.

В-четвертых, человеку должен быть доступен смысл представленной информации, ее семантика.

К основным направлениям реализации злоумышленником информационных угроз относятся:

- непосредственное обращение к объектам доступа;
- создание программных и технических средств, выполняющих обращение к объектам доступа в обход средств защиты;
- модификация средств защиты, позволяющая реализовать угрозы информационной безопасности;
- внедрение в технические средства АС программных или технических механизмов, нарушающих предполагаемую структуру и функции АС.

В табл. 3.1 перечислены основные методы реализации угроз информационной безопасности.

Таблица 3.1. Основные методы реализации угроз информационной безопасности

Уровень доступа к информации в АС	Угроза раскрытия параметров системы	Угроза нарушения конфиденциальности	Угроза нарушения целостности	Угроза отказа служб (отказа доступа к информации)
Уровень носителей информации	Определение типа и параметров носителей информации	Хищение (копирование) носителей информации.	Уничтожение машинных носителей информации	Выведение из строя машинных носителей информации
Уровень средств взаимодействия с носителем	Получение информации о программно-аппаратной среде. Получение детальной информации о функциях, выполняемых АС. Получение данных о применяемых системах защиты	Несанкционированный доступ к ресурсам АС. Совершение пользователем несанкционированных действий. Несанкционированное копирование программного обеспечения. Перехват данных, передаваемых по каналам связи	Внесение пользователем несанкционированных изменений в программы и данные. Установка и использование нештатного программного обеспечения. Заражение программными вирусами	Проявление ошибок проектирования и разработки программно-аппаратных компонент АС. Обход механизмов защиты АС
Уровень представления информации	Определение способа представления информации	Визуальное наблюдение. Раскрытие представления информации (дешифрование)	Внесение искажения в представление данных; уничтожение данных языка	Искажение соответствия синтаксических и семантических конструкций
Уровень содержания информации	Определение содержания данных на качественном уровне	Раскрытие содержания информации	Внедрение дезинформации	Запрет на использование информации

Для достижения требуемого уровня информационной безопасности АС необходимо обеспечить противодействие различным техническим угрозам и минимизировать возможное влияние человеческого фактора.

3.3. СТАНДАРТЫ БЕЗОПАСНОСТИ

Проблемой информационной компьютерной безопасности начали заниматься с того момента, когда компьютер стал обрабатывать данные, ценность которых высока для пользователя. В последние годы в связи с развитием компьютерных сетей и ростом спроса на электронные услуги ситуация в сфере информационной безопасности серьезно обострилась, а вопрос стандартизации подходов к ее решению стал особенно актуальным как для разработчиков, так и для пользователей средств ИТ.

3.3.1. РОЛЬ СТАНДАРТОВ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Главная задача стандартов информационной безопасности – создать основу для взаимодействия между производителями, потребителями и экспертами по квалификации продуктов информационных технологий. Каждая из этих групп имеет свои интересы и свои взгляды на проблему информационной безопасности.

Потребители заинтересованы в методике, позволяющей обоснованно выбирать продукт, отвечающий их нуждам и решающий их проблемы, для чего им необходима шкала оценки безопасности. Потребители так же нуждаются в инструменте, с помощью которого они могли бы формулировать свои требования производителям. При этом потребителей интересует исключительно характеристики и свойства конечного продукта, а не методы и средства их достижения. К сожалению, многие потребители не понимают, что требования безопасности обязательно противоречат функциональным требованиям (удобству работы, быстродействию и т.д.), накладывая ограничения на совместимость и, как правило, вынуждают отказаться от широко распространенных и поэтому незащищенных прикладных программных средств.

Производители нуждаются в стандартах как средстве сравнения возможностей своих продуктов и в применении процедуры сертификации как механизма объективной оценки их свойств, а также в стандартизации определенного набора требований безопасности, который мог бы ограничить фантазию заказчика конкретного продукта и заставить его выбирать требования из этого набора. С точки зрения производителя, требования должны быть максимально конкретными и регламентировать необходимость применения тех или иных средств, механизмов, алгоритмов и т.д. Кроме того, требования не должны противоречить существующим парадигмам обработки информации, архитектуре вычислительных систем и технологиям создания информационных продуктов. Этот подход также нельзя признать в качестве доминирующего, так как он не учитывает нужд пользователей и пытается подогнать требования защиты под существующие системы и технологии.

Эксперты по квалификации и специалисты по сертификации рассматривают стандарты как инструмент, позволяющий им оценить уровень безопасности, обеспечиваемый продуктами информационных технологий, и предоставить потребителям возможность сделать обоснованный выбор. Эксперты по квалификации находятся в двойственном положении: с одной стороны они, как и производители, заинтересованы в четких и простых критериях, над которыми не надо ломать голову, как их применить к конкретному продукту, а с другой стороны, они должны дать обоснованный ответ пользователям – удовлетворяет продукт их нужды или нет. Таким образом, перед стандартами информационной безопасности стоит непростая задача – примирить три разные точки зрения и создать эффективный механизм взаимодействия всех сторон. Причем ущемление

потребностей хотя бы одной из них приведет к невозможности взаимопонимания и взаимодействия и, следовательно, не позволит решить общую задачу - создание защищенной системы обработки информации.

В последнее время в разных странах появилось новое поколение стандартов в области защиты информации, посвященных практическим вопросам управления информационной безопасностью компании. Это, прежде всего, международные стандарты управления информационной безопасностью ISO 15408, ISO 17799 и некоторые другие. Представляется целесообразным проанализировать наиболее важные из этих документов, сопоставить содержащиеся в них требования и критерии, а также оценить эффективность их практического применения.

В соответствии с международными и национальными стандартами обеспечение информационной безопасности в любой компании предполагает следующее:

- определение целей обеспечения информационной безопасности компьютерных систем;
- создание эффективной системы управления информационной безопасностью;
- расчет совокупности детализированных качественных и количественных показателей для оценки соответствия информационной безопасности постав ленным целям;
- применение инструментария обеспечения информационной безопасности и оценки ее текущего состояния;
- использование методик управления безопасностью, позволяющих объектив но оценить защищенность информационных активов и управлять информационной безопасностью компании.

Рассмотрим наиболее известные международные стандарты в области защиты информации.

3.3.2. СТАНДАРТЫ ISO/IEC 17799:2002 (BS 7799:2000)

В настоящее время международный стандарт ISO/IEC 17799:2000 (BS 7799-1:2000) «Информационные технологии – Управление информационной безопасностью» (Information technology – Information security management) является одним из наиболее известных стандартов в области защиты информации. Данный стандарт был разработан на основе первой части британского стандарта BS 7799-1:1995 «Практические рекомендации по управлению информационной безопасностью» (Information security management – Part 1: Code of practice for information security management) и относится к новому поколению стандартов информационной безопасности компьютерных информационных систем.

Текущая версия стандарта ISO/IEC 17799:2000 (BS 7799-1:2000) рассматривает следующие актуальные вопросы обеспечения информационной безопасности организаций и предприятий:

- необходимость обеспечения информационной безопасности;
- основные понятия и определения информационной безопасности;
- политика информационной безопасности компании;
- организация информационной безопасности на предприятии;
- классификация и управление корпоративными информационными ресурсами;

- кадровый менеджмент и информационная безопасность;
- физическая безопасность;
- администрирование безопасности корпоративных информационных систем;
- управление доступом;
- требования по безопасности к корпоративным информационным системам в ходе их разработки, эксплуатации и сопровождения;
- управление бизнес-процессами компании с точки зрения информационной безопасности;
- внутренний аудит информационной безопасности компании.

Вторая часть стандарта BS 7799-2:2000 «Спецификации систем управления информационной безопасностью» (Information security management - Part 2: Specification for information security management systems) определяет возможные функциональные спецификации корпоративных систем управления информационной безопасностью с точки зрения их проверки на соответствие требованиям первой части данного стандарта. В соответствии с положениями этого стандарта также регламентируется процедура аудита информационных корпоративных систем.

Дополнительные рекомендации для управления информационной безопасностью содержат руководства Британского института стандартов (British Standards Institution - BSI), изданные в период 1995-2003 годов в виде следующей серии:

- «Введение в проблему управления информационной безопасности» (Information security managment: an introduction);
- «Возможности сертификации на требования стандарта BS 7799» (Preparing for BS 7799 sertification);
- «Руководство BS 7799 по оценке и управлению рисками» (Guide to BS 7799 risk assessment and risk management);
- «Руководство BS 7799 для проведения аудита на требования стандарта» (Guide to BS 7799 auditing);
- «Практические рекомендации по управлению безопасностью информационных технологий» (Code of practice for IT management).

В 2002 году международный стандарт ISO 17799 (BS 7799) был пересмотрен и существенно дополнен. В новом варианте этого стандарта большое внимание уделено вопросам повышения культуры защиты информации в различных международных компаниях, в том числе вопросам обучения и изначальной интеграции процедур и механизмов оценки и управления информационной безопасности в информационные технологии корпоративных систем. По мнению специалистов, обновление международного стандарта ISO 17799 (BS 7799) позволит не только повысить культуру защиты информационных активов компании, но и скоординировать действия различных ведущих государственных и коммерческих структур в области защиты информации.

3.3.3. ГЕРМАНСКИЙ СТАНДАРТ BSI

В отличие от ISO 17799, германское «Руководство по защите информационных технологий для базового уровня защищенности» посвящено детальному рассмотрению частных вопросов управления информационной безопасности компании.

В германском стандарте BSI представлены:

- общая методика управления информационной безопасностью (организация менеджмента в области ИБ, методология использования руководства);
- описание компонентов современных информационных технологий;
- описание основных компонентов организации режима информационной безопасности (организационный и технический уровни защиты данных, планирование действий в чрезвычайных ситуациях, поддержка непрерывности бизнеса);
- характеристики объектов информатизации (здания, помещения, кабельные сети, контролируемые зоны);
- характеристики основных информационных активов компании (в том числе аппаратное и программное обеспечение, например рабочие станции и серверы под управлением операционных систем семейства DOS, Windows и UNIX);
- характеристики компьютерных сетей на основе различных сетевых технологий, например сети Novell NetWare, UNIX и Windows;
- характеристики активного и пассивного телекоммуникационного оборудования ведущих поставщиков, например Cisco Systems;
- подробные каталоги угроз безопасности и мер контроля (более 600 наименований в каждом каталоге).

Вопросы защиты приведенных информационных активов компании рассматриваются по определенному сценарию: общее описание информационного актива компании – возможные угрозы и уязвимости безопасности – возможные меры и средства контроля и защиты.

3.3.4. МЕЖДУНАРОДНЫЙ СТАНДАРТ ISO 15408 «ОБЩИЕ КРИТЕРИИ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ»

Одним из главных результатов стандартизации в сфере систематизации требований и характеристик защищенных информационных комплексов стала система международных и национальных стандартов безопасности информации, которая насчитывает более сотни различных документов. Важное место в этой системе стандартов занимает стандарт ISO 15408, известный как «Общие критерии» (Common Criteria).

В 1990 году Международная организация по стандартизации (ISO) приступила к разработке международного стандарта по критериям оценки безопасности информационных технологий для общего использования «Common Criteria», или «Общие критерии оценки безопасности информационных технологий» (ОК).

В разработке «Общих критериев» участвовали: Национальный институт стандартов и технологии и Агентство национальной безопасности (США), Учреждение безопасности коммуникаций (Канада), Агентство информационной безопасности (Германия), Агентство национальной безопасности коммуникаций (Голландия), органы исполнения Программы безопасности и сертификации ИТ (Англия), Центр обеспечения безопасности систем (Франция), которые опирались на свой солидный задел.

«Общие критерии» обобщили содержание и опыт использования Оранжевой книги, развили европейские и канадские критерии и воплотили в реальные структуры концепцию типовых профилей защиты федеральных критериев США. За

десятилетие разработки «Общие критерии» неоднократно редактировались лучшими специалистами мира. В результате был подготовлен международный стандарт ISO/IEC 15408.

Первые две версии «Общих критериев» были опубликованы соответственно в январе и мае 1998 года. Версия 2.1 этого стандарта утверждена 8 июня 1999 года Международной организацией по стандартизации (ISO) в качестве международного стандарта информационной безопасности ISO/IEC 15408 под названием «Общие критерии оценки безопасности информационных технологий».

В ОК проведена классификация широкого набора требований безопасности ИТ, определены структуры их группирования и принципы использования. Главные достоинства ОК – полнота требований безопасности и их систематизация, гибкость в применении и открытость для последующего развития.

«Общие критерии» адаптированы к потребностям взаимного признания результатов оценки безопасности ИТ в мировом масштабе и предназначены для использования в качестве основы для такой оценки. Они позволяют сравнить результаты независимых оценок информационной безопасности и допустимых рисков на основе множества общих требований к функциям безопасности средств и систем ИТ, а также гарантий, применяемых к ним в процессе тестирования.

Основываясь на общем перечне (наборе) требований, в процессе выработки оценки уровня защиты устанавливается уровень доверия. Результаты оценок защиты позволяют определить для компании достаточность защиты корпоративной информационной системы.

Ведущие мировые производители оборудования ИТ основательно подготовились к этому моменту и сразу стали поставлять заказчикам средства, полностью отвечающие требованиям ОК.

Принятый базовый стандарт информационной безопасности ISO 15408, безусловно, очень важен для российских разработчиков.

«Общие критерии» разрабатывались в расчете на то, чтобы удовлетворить запросы трех групп специалистов, в равной степени являющихся пользователями этого документа: производителей и потребителей продуктов информационных технологий, а также экспертов по оценке уровня их безопасности. «Общие критерии» обеспечивают нормативную поддержку процесса выбора ИТ-продукта, к которому предъявляются требования функционирования в условиях действия определенных угроз; служат руководящим материалом для разработчиков таких систем;а также регламентируют технологию их создания и процедуру оценки обеспечиваемого уровня безопасности.

«Общие критерии» рассматривают информационную безопасность, во-первых, как совокупность конфиденциальности и целостности информации, обрабатываемой ИТ-продуктом, а также доступности ресурсов ВС, и во-вторых, ставят перед средствами защиты задачу противодействия угрозам, актуальным для среды эксплуатации этого продукта и реализации политики безопасности, принятой в данной среде эксплуатации. Поэтому в концепцию «Общих критериев» входят все аспекты процесса проектирования, производства и эксплуатации ИТ-продуктов, предназначенных для работы в условиях действия определенных угроз безопасности.

Потребители ИТ-продуктов озабочены наличием угроз безопасности, приводящих к определенным рискам для обрабатываемой информации. Для противодействия этим угрозам ИТ-продукты должны включать в свой состав

средства защиты, противодействующие этим угрозам и направленные на устранение уязвимостей, однако ошибки в средствах защиты, в свою очередь, могут приводить к появлению новых уязвимостей. Сертификация средств защиты позволяет подтвердить их адекватность угрозам и рискам.

«Общие критерии» регламентируют все стадии разработки, квалификационного анализа и эксплуатации ИТ-продуктов. «Общие критерии» предлагают концепцию процесса разработки и квалификационного анализа ИТ-продуктов, требующую от потребителей и производителей большой работы по составлению и оформлению довольно объемных и подробных нормативных документов.

Требования «Общих критериев» являются практически всеобъемлющей энциклопедией информационной безопасности, поэтому их можно использовать в качестве справочника по безопасности информационных технологий.

Стандарт ISO 15408 поднял стандартизацию информационных технологий на межгосударственный уровень. Возникла реальная перспектива создания единого безопасного информационного пространства, в котором сертификация безопасности систем обработки информации будет осуществляться на глобальном уровне, что предоставит возможности для интеграции национальных информационных систем, а это, в свою очередь, откроет новые сферы применения информационных технологий.

3.3.5. РОССИЙСКИЕ СТАНДАРТЫ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Исторически сложилось, что в России проблемы безопасности ИТ решались в основном в сфере охраны государственной тайны. Аналогичные задачи коммерческого сектора экономики долгое время не находили соответствующих решений.

Информация, содержащаяся в системах или продуктах ИТ, является критическим ресурсом, позволяющим организациям успешно решать свои задачи. Кроме того, частные лица вправе ожидать, что их персональная информация, будучи размещенной в продуктах или системах ИТ, останется приватной, доступной им по мере необходимости и сможет быть подвергнута несанкционированной модификации.

При выполнении продуктами или системами ИТ их функций следует осуществлять надлежащий контроль информации, что обеспечивало бы ее защиту от опасностей типа нежелательного или неоправданного распространения, изменения или потери. Понятие «безопасность ИТ» охватывает предотвращение и уменьшение этих и аналогичных опасностей.

Проблема защиты информации в коммерческой автоматизированной системе имеет свои особенности, которые необходимо учитывать, поскольку они оказывают серьезное влияние на информационную безопасность (ИБ). Перечислим основные особенности:

1) Приоритет экономического фактора. Для коммерческой автоматизированной системы важно снизить либо исключить финансовые потери и обеспечить получение прибыли владельцем и пользователями данного инструментария в условиях реальных рисков. Важным условием при этом, в частности, является минимизация типично банковских рисков (например, потерь за счет

ошибочных направлений платежей, фальсификации платежных документов и т.п.).

2) Открытость проектирования, предусматривающая создание подсистемы защиты информации из средств, широко доступных на рынке и работающих в открытых системах.
3) Юридическая значимость коммерческой информации, которую можно определить как свойство безопасной информации, позволяющее обеспечить юридическую силу электронным документам или информационным процессам в соответствии с законодательством Российской Федерации.

Среди различных стандартов по безопасности информационных технологий, существующих в настоящее время в России, следует выделить нормативные документы по критериям оценки защищенности средств вычислительной техники и автоматизированных систем и документы, регулирующие информационную безопасность (табл. 3.1, пункты 1-10). К ним можно добавить нормативные документы по криптографической защите систем обработки информации и информационных технологий (табл. 3.1, пункты 11-13).

Стандарты в структуре информационной безопасности выступают как связующее звено между технической и концептуальной стороной вопроса.

Введение в 1999 году международного стандарта ISO 15408 в области обеспечения информационной безопасности имело большое значение как для разработчиков компьютерных информационных систем, так и для пользователей. Стандарт ISO 15408 стал своего рода гарантией качества и надежности сертифицированных по нему программных продуктов. Этот стандарт позволил потребителям лучше ориентироваться при выборе программного обеспечения и приобретать продукты, соответствующие их требованиям безопасности и повысил конкурентоспособность ИТ-компаний, сертифицирующих свою продукцию в соответствии с ISO 15408.

Стандарт «Критерии оценки безопасности информационных технологий» ГОСТ Р ИСО/МЭК 15408

С января 2004 года в России действует стандарт «Критерии оценки безопасности информационных технологий» ГОСТ Р ИСО/МЭК 15408, который является аналогом стандарта ISO 15408. Стандарт ГОСТ Р ИСО/МЭК 15408, называется еще «Общими критериями» (ОК), является на сегодня самым полным стандартом, определяющим инструменты оценки безопасности информационных систем и порядок их использования.

Таблица 3.1. Российские стандарты, регулирующие информационную безопасность

№ п/п	Стандарт	Наименование
1	ГОСТ Р ИСО/МЭК 15408-1-2002	Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 1. Введение и общая модель. Госстандарт России
2	ГОСТ Р ИСО/МЭК 15408-2-2002	Методы и средства обеспечения безопасности. Критерии оценки

		безопасности информационных технологий. Часть 2. Функциональные требования безопасности. Госстандарт России
3	ГОСТ Р ИСО/МЭК 15408-3-2002	Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 3. Требования доверия к безопасности. Госстандарт России.
4	ГОСТ Р 50739-95	Средства вычислительной техники. Защита от несанкционированного доступа к информации. Общие требования. Госстандарт России.
5	ГОСТ Р 50922-96	Защита информации. Основные термины и определения. Госстандарт России.
6	ГОСТ Р 51188-98	Защита информации. Испытания программных средств на наличие компьютерных вирусов. Типовое руководство. Госстандарт России.
7	ГОСТ Р 51275-99	Защита информации. Объект информации. Факторы, воздействующие на информацию. Общие положения. Госстандарт России.
8	ГОСТ Р ИСО 7498-1-99	Информационная технология. Взаимосвязь открытых систем. Базовая эталонная модель. Часть 1. Базовая модель. Госстандарт России.
9	ГОСТ Р ИСО 7498-2-99	Информационная технология. Взаимосвязь открытых систем. Базовая эталонная модель. Часть 2. Архитектура защиты информации. Госстандарт России.
10	ГОСТ Р 50739-95	Средства вычислительной техники. Защита от несанкционированного доступа к информации. Общие технические требования.
11	ГОСТ 28147-89	Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования.
12	ГОСТ Р 34.10-2001	Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи.
13	ГОСТ Р 34.11-94	Информационная технология. Криптографическая защита информации. Функция хэширования.

«Общие критерии» направлены на защиту информации от несанкционированного раскрытия, модификации, полной или частичной потери и применимы к защитным мерам, реализуемым аппаратными, программно-аппаратными и программными средствами.

«Общие критерии» предназначены служить основой при оценке характеристик безопасности продуктов и систем ИТ. Заложенные в стандарте наборы требований позволяют сравнивать результаты независимых оценок безопасности. На основании этих результатов потребитель может принимать решение о том, достаточно ли безопасны ИТ-продукты или системы для их применения с заданным уровнем риска.

Стандарт ГОСТ Р ИСО/МЭК 15408 состоит из трех частей.

В первой части (ГОСТ Р ИСО/МЭК 15408-1 «Введение и общая модель») устанавливается общий подход к формированию требований безопасности и оценке безопасности, на их основе разрабатываются основные конструкции (профиль защиты и задание по безопасности) представления требований безопасности в интересах потребителей, разработчиков и оценщиков продуктов и систем ИТ. Требования безопасности объекта оценки (ОО) по методологии «Общих критериев» определяются исходя из целей безопасности, которые основываются на анализе назначения ОО и условий среды его использования (угроз, предположений, политики безопасности).

Часть вторая (ГОСТ Р ИСО/МЭК 15408-2 «Функциональные требования безопасности») содержит универсальный каталог функциональных требований безопасности и предусматривает возможность их детализации и расширения по определенным правилам.

Третья часть (ГОСТ Р ИСО/МЭК 15408-3 «Требования доверия к безопасности») включает в себя систематизированный каталог требований доверия, определяющих меры, которые должны быть приняты на всех этапах жизненного цикла продукта или системы ИТ для обеспечения уверенности в том, что они удовлетворяют предъявленным к ним требованиям. Здесь же содержатся оценочные уровни доверия (ОУД), определяющие шкалу требований, которые позволяют с возрастающей степенью полноты и строгости оценить проектную, тестовую и эксплуатационную документацию, правильность реализации функций безопасности ОО, уязвимости продукта или системы ИТ, стойкость механизмов защиты и сделать заключение об уровне доверия к безопасности объекта оценки.

Обобщая изложенное, можно отметить, что каркас безопасности, заложенный частью 1 стандарта ГОСТ Р ИСО/МЭК 15408, заполняется содержимым из классов, семейств и компонентов в части 2, а третья часть определяет, как оценить прочность всего «строения».

Стандарт «Общие критерии безопасности информационных технологий» отражает достижения последних лет в области информационной безопасности. Впервые документ такого рода содержит разделы, адресованные потребителям, производителям и экспертам по оценке безопасности ИТ-продуктов.

Главные достоинства стандарта ГОСТ Р ИСО/МЭК 15408:

- полнота требований к информационной безопасности;
- гибкость в применении;

– открытость для последующего развития с учетом новейших достижений науки и техники.

3.4. СТАНДАРТЫ ДЛЯ БЕСПРОВОДНЫХ СЕТЕЙ

Стандарт IEEE 802.11. В 1990 году Комитет IEEE 802 сформировал рабочую группу 802.11 для разработки стандарта для беспроводных локальных сетей. Работы по созданию стандарта были завершены через 7 лет. В 1997 году была ратифицирована первая спецификация беспроводного стандарта IEEE 802.11, обеспечивающего передачу данных с гарантированной скоростью 1 Мбит/с (в некоторых случаях до 2 Мбит/с) в полосе частот 2,4 ГГц. Эта полоса частот доступна для нелицензионного использования в большинстве стран мира.

Стандарт IEEE 802.11 является базовым стандартом и определяет протоколы, необходимые для организации беспроводных локальных сетей WLAN (Wireless Local Area Network). Основные из них - протокол управления доступом к среде MAC (Medium Accsess Control - нижний подуровень канального уровня) и протокол PHY передачи сигналов в физической среде. В качестве физической среды допускается использование радиоволн и инфракрасного излучения.

В основу стандарта IEEE 802.11 положена сотовая архитектура, причем сеть может состоять как из одной, так и нескольких ячеек. Каждая сота управляется базовой станцией, называемой точкой доступа AP (Access Point), которая вместе с находящимися в пределах радиуса ее действия рабочими станциями пользователей образует базовую зону обслуживания BSS (Basic Service Set). Точки доступа многосотовой сети взаимодействуют между собой через распределительную систему DS (Distribution System), представляющую собой эквивалент магистрального сегмента кабельных ЛС. Вся инфраструктура, включающая точки доступа и распределительную систему, образует расширенную зону обслуживания ESS (Extended Service Set). Стандартом предусмотрен также односотовый вариант беспроводной сети, который может быть реализован и без точки доступа, при этом часть ее функций выполняется непосредственно рабочими станциями. Для обеспечения перехода мобильных рабочих станций из зоны действия одной точки доступа к другой в многосотовых системах предусмотрены специальные процедуры сканирования (активного и пассивного прослушивания эфира) и присоединения (Association), однако строгих спецификаций по реализации роуминга стандарт 802.11 не предусматривает.

Для защиты WLAN стандартом IEEE 802.11 предусмотрен алгоритм WEP (Wired Equivalent Privacy). Он включает средства противодействия несанкционированному доступу к сети, а также шифрование для предотвращения перехвата информации. Однако заложенная в первую спецификацию стандарта IEEE 802.11 скорость передачи данных в беспроводной сети уже не удовлетворяла потребностям пользователей. Алгоритм WEP страдал рядом существенных недостатков - для него характерны отсутствие управления ключом, использование общего статического ключа, малые разрядности ключа и вектора инициализации, сложности использования алгоритма RC4.

Чтобы сделать технологию Wireless LAN недорогой, популярной и удовлетворяющей жестким требованиям бизнес-приложений, разработчики были вынуждены создать семейство новых спецификаций стандарта IEEE 802.11 a, b, ..., i. Стандарты этого семейства, по сути, являются беспроводными расширениями

протокола Ethernet, что обеспечивает хорошее взаимодействие с проводными сетями Ethernet.

Стандарт IEEE 802.11b применяется наиболее широко из всех стандартов 802.11. Высокоскоростной стандарт 802.11b был ратифицирован IEEE в сентябре 1999 года как развитие базового стандарта 802.11; в стандарте 802.11b используется полоса частот 2,4 ГГц, скорость передачи достигает 11 Мбит/с (подобно Ethernet). Благодаря ориентации на освоенный диапазон 2,4 ГГц стандарт 802.11b завоевал большую популярность у производителей оборудования. В качестве базовой радиотехнологии в нем используется метод распределенного спектра с прямой последовательностью DSSS (Direct Sequence Spread Spectrum), который отличается высокой устойчивостью к искажению данных помехами, в том числе преднамеренными. Этот стандарт получил широкое распространение, и беспроводные LAN стали привлекательным решением с технической и финансовой точки зрения.

Для простоты запоминания в качестве общего имени для стандартов 802.11b и 802.11a, а также всех последующих, относящихся к беспроводным локальным сетям (WLAN), был введен термин Wi-Fi (Wireless Fidelity). Этот термин введен Ассоциацией беспроводной совместимости с Ethernet WECA (Wireless Ethernet Compatibility Aliance). Если устройство помечено этим знаком, оно протестировано на совместимость с другими устройствами 802.11.

Стандарт IEEE 802.11a предназначен для работы в частотном диапазоне 5 ГГц. Скорость передачи данных до 54 Мбит/с, то есть примерно в пять раз быстрее сетей 802.11b. Ассоциация WECA называет этот стандарт WiFi5. Он наиболее широкополосный из семейства стандартов 802.11. Определены три обязательные скорости – 6, 12 и 24 Мбит/с и пять необязательных – 9, 18, 36, 48 и 54 Мбит/с. В качестве метода модуляции сигнала принято ортогональное частотное мультиплексирование OFDM (Orthogonal Frequency Division Multiplexing). Его отличие от метода DSSS заключается в том, что OFDM предполагает параллельную передачу полезного сигнала одновременно по нескольким частотам диапазона, в то время как технологии расширения спектра DSSS передают сигналы последовательно. В результате повышается пропускная способность канала и качество сигнала. К недостаткам стандарта 802.11a относятся большая потребляемая мощность радиопередатчиков для частот 5 ГГц, а также меньший радиус действия (около 100 м).

Стандарт IEEE 802.11g представляет собой развитие 802.11b и обратно совместим с 802.11b. Предназначен для обеспечения скоростей передачи данных до 54 Мбит/с. В числе достоинств 802.11g – низкая потребляемая мощность, большие расстояния (до 300 м) и высокая проникающая способность сигнала.

Стандарт IEEE 802.11i. В 2004 году IEEE ратифицировал стандарт обеспечения безопасности в беспроводных сетях IEEE 802.11i. Этот стандарт решил существовавшие проблемы в области аутентификации и протокола шифрования, обеспечив значительно более высокий уровень безопасности. Стандарт 802.11i может применяться в сетях Wi-Fi независимо от используемого стандарта – 802.11a, b или g.

В настоящее время существует два очень похожих стандарта – WPA и 802.11i. Они оба используют механизм 802.1x для обеспечения надежной аутентификации, оба применяют сильные алгоритмы шифрования, оба предназначены для замены протокола WEP. W PA был разработан в Wi-Fi Alliance как решение, которое можно

применить немедленно, не дожидаясь завершения длительной процедуры ратификации 802.11i в IEEE.

Основное отличие двух стандартов заключается в использовании различных механизмов шифрования. В WPA применяется протокол TKIP (Temporal Key Integrity Protocol), который, так же как и WEP, использует шифр RC4, но значительно более безопасным способом. Обеспечение конфиденциальности данных в стандарте IEEE 802.11i основано на использовании алгоритма шифрования AES (Advanced Encryption Standard). Использующий его защитный протокол получил название CCMP (Counter-Mode CBC MAC Protocol). Алгоритм AES обладает высокой криптостойкостью. Длина ключа AES равна 128, 192 или 256 бит, что обеспечивает наиболее надежное шифрование из доступных сейчас.

Стандарт 802.11i предполагает наличие трех участников процесса аутентификации. Это сервер аутентификации AS (Authentication Server), точка доступа AP (Access Point) и рабочая станция STA (Station). В процессе шифрования данных участвуют только AP и STA (AS не используется). Стандарт предусматривает двустороннюю аутентификацию (в отличие от WEP, где аутентифицируется только рабочая станция, но не точка доступа). При этом местами принятия решения о разрешении доступа являются сервер аутентификации AS и рабочая станция STA, а местами исполнения этого решения - точка доступа AP и STA.

Для работы по стандарту 802.11i создается иерархия ключей, включающая мастер-ключ MK (Master Key), парный мастер-ключ PMK (Pairwise Master Key), парный временный ключ PTK (Pairwise Transient Key), а также групповые временные ключи GTK (Group Transient Key), служащие для защиты широковещательного сетевого трафика.

MK - это симметричный ключ, реализующий решение STA и AS о взаимной аутентификации. Для каждой сессии создается новый MK.

PMK - обновляемый симметричный ключ, владение которым означает разрешение (авторизацию) на доступ к среде передачи данных в течение данной сессии. PMK создается на основе MK. Для каждой пары STA и AP в каждой сессии создается новый PMK.

PTK - это коллекция операционных ключей, которые используются для привязки PMK к данным STA и AP, для распространения GTK и шифрования данных.

Процесс аутентификации и доставки ключей определяется стандартом 802.1x. Он предоставляет возможность использовать в беспроводных сетях такие традиционные серверы аутентификации, как RADIUS (Remote Authentication Dial-In User Server). Стандарт 802.11i не определяет тип сервера аутентификации, но использование RADIUS для этой цели является стандартным решением.

Транспортом для сообщений 802.1x служит протокол EAP (Extensible uthentication Protocol). EAP позволяет легко добавлять новые методы аутентификации. Точке доступа не требуется знать об используемом методе аутентификации, поэтому изменение метода никак не затрагивает точку доступа.

Наиболее популярные методы EAP - это LEAP, PEAP, TTLS и FAST Каждый из методов имеет свои сильные и слабые стороны, условия применения, по-разному поддерживается производителями оборудования и программного обеспечения. Можно выделить пять фаз работы 802.11i.

Первая фаза - обнаружение. В этой фазе рабочая станция STA находит точку доступа AP, с которой может установить связь, и получает от нее используемые в

данной сети параметры безопасности. Таким образом STA узнает идентификатор сети SSID и методы аутентификации, доступные в данной сети. Затем STA выбирает метод аутентификации, и между STA и AP устанавливается соединение. После этого STA и AP готовы к началу второй фазы. 802.1x.

Вторая фаза – аутентификация. В этой фазе выполняется взаимная аутентификация STA и сервера AS, создаются MK и PMK. В данной фазе STA и AP блокируют весь трафик, кроме трафика 802.1x.

В третьей фазе AS перемещает ключ PMK на AP. Теперь STA и AP владеют действительными ключами PMK. Четвертая фаза – управление ключами 802.1x. В этой фазе происходит генерация, привязка и верификация ключа PTK.

Пятая фаза – шифрование и передача данных. Для шифрования используется соответствующая часть PTK.

Стандартом 802.11i предусмотрен режим PSK (Pre-Shared Key), который позволяет обойтись без сервера аутентификации AS. При использовании этого режима на STA и на AP вручную вводится Pre-Shared Key, который используется в качестве PMK. Дальше генерация PTK происходит описанным выше порядком. Режим PSK может использоваться в небольших сетях, где нецелесообразно устанавливать сервер AS.

3.5. СТАНДАРТЫ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ В ИНТЕРНЕТЕ

В последнее время в мире бурно развивается электронная коммерция посредством сети Интернет. Развитие электронной коммерции в основном определяется прогрессом в области безопасности информации. При этом базовыми задачами являются обеспечение доступности, конфиденциальности, целостности и юридической значимости информации.

По оценке Комитета ООН по предупреждению преступности и борьбе с ней, компьютерная преступность вышла на уровень одной из международных проблем. Поэтому чрезвычайно важно добиваться эффективного решения проблем обеспечения безопасности коммерческой информации в глобальной сети Интернет и смежных intranet-сетях, которые по своей технической сущности не имеют принципиальных отличий и различаются в основном масштабами и открытостью.

Рассмотрим особенности стандартизации процесса обеспечения безопасности коммерческой информации в сетях с протоколом передачи данных IP/TCP и с акцентом на защиту телекоммуникаций.

Обеспечение безопасности информационных технологий (ИТ) особенно актуально для открытых систем коммерческого применения, обрабатывающих информацию ограниченного доступа, не содержащую государственную тайну.

Под открытой системой понимают совокупность всевозможного вычислительного и телекоммуникационного оборудования разного производства, совместное функционирование которого обеспечивается соответствием требованиям международных стандартов. Термин «открытые системы» подразумевает также, что если вычислительная система соответствует стандартам, то она будет открыта для взаимосвязи с любой другой системой, которая соответствует тем же стандартам. Это, в частности, относится и к механизмам криптографической защиты информации или к защите от несанкционированного доступа (НСД) к информации.

Важная заслуга Интернета состоит в том, что он заставил по-новому взглянуть на такие технологии. Во-первых, Интернет поощряет применение открытых стандартов, доступных для внедрения всем, кто проявит к ним интерес.

Во-вторых, он представляет собой крупнейшую в мире и, вероятно, единственную сеть, к которой подключается такое множество разных компьютеров. И наконец, Интернет становится общепринятым средством представления быстро меняющихся новой продукции и технологий на мировом рынке.

В Интернете уже давно существует целый ряд комитетов, в основном из организаций-добровольцев, которые осторожно проводят предлагаемые технологии через процесс стандартизации. Эти комитеты, составляющие основную часть Рабочей группы инженеров Интернета IETF (Internet Engineering Task Force), провели стандартизацию нескольких важных протоколов, ускоряя их внедрение в Интернете. Непосредственными результатами усилий IETF являются такие протоколы, как семейство TCP/IP для передачи данных, SMTP (Simple Mail Transport Protocol) и POP (Post Office Protocol) для электронной почты, а так же SNMP (Simple Network Management Protocol) для управления сетью. В Интернете популярны протоколы безопасной передачи данных, а именно SSL, SET, IPSec. Перечисленные протоколы появились в Интернете сравнительно недавно как ответ на необходимость защиты ценной информации и сразу стали стандартами де-факто.

Протокол SSL

Протокол SSL (Secure Socket Layer) является сейчас популярным сетевым протоколом с шифрованием данных для безопасной передачи по сети. Он позволяет устанавливать защищенное соединение, производить контроль целостности данных и решать различные сопутствующие задачи. Протокол SSL обеспечивает защиту данных между сервисными протоколами (такими, как HTTP, FTP и др.) и транспортными протоколами (TCP/IP) с помощью современной криптографии.

Протокол SET

Протокол SET (Security Electronics Transaction) - перспективный стандарт безопасных электронных транзакций в сети Интернет, предназначенный для организации электронной торговли через сеть Интернет. Протокол SET основан на использовании цифровых сертификатов по стандарту X.509.

Протокол выполнения защищенных транзакций SET является стандартом, разработанным компаниями MasterCard и Visa при значительном участии IBM, GlobeSet и других партнеров. Он позволяет покупателям приобретать товары через Интернет, используя защищенный механизм выполнения платежей.

SET является открытым стандартным многосторонним протоколом для проведения безопасных платежей с использованием пластиковых карточек в Интернете. SET обеспечивает кросс-аутентификацию счета держателя карты, продавца и банка продавца для проверки готовности оплаты, а также целостность и секретность сообщения, шифрование ценных и уязвимых данных. Поэтому SET более правильно было бы назвать стандартной технологией или системой протоколов выполнения безопасных платежей с использованием пластиковых карт через Интернет. SET позволяет потребителям и продавцам подтвердить подлинность всех участников сделки, происходящей в Интернете, с помощью криптографии, в том числе применяя цифровые сертификаты.

Объем потенциальных продаж в области электронной коммерции ограничивается достижением необходимого уровня безопасности информации, который обеспечивают вместе покупатели, продавцы и финансовые институты,

обеспокоенные вопросами безопасности в Интернете. Как упоминалось ранее, базовыми задачами защиты информации являются обеспечение ее доступности, конфиденциальности, целостности и юридической значимости. SET, в отличие от других протоколов, позволяет решать указанные задачи защиты информации в целом.

SET, в частности, обеспечивает следующие специальные требования защиты операций электронной коммерции:

- секретность данных оплаты и конфиденциальность информации заказа, переданной наряду с данными об оплате;
- сохранение целостности данных платежей. Целостность информации платежей обеспечивается с помощью цифровой подписи;
- специальную криптографию с открытым ключом для проведения аутентификации;
- аутентификацию держателя по кредитной карточке. Она обеспечивается применением цифровой подписи и сертификатов держателя карт;
- аутентификацию продавца и его возможности принимать платежи по пластиковым карточкам с применением цифровой подписи и сертификатов продавца;
- аутентификацию того, что банк продавца является действующей организацией, которая может принимать платежи по пластиковым карточкам через связь с процессинговой карточной системой. Аутентификация банка продав ца обеспечивается использованием цифровой подписи и сертификатов банка продавца;
- готовность оплаты транзакций в результате аутентификации сертификата с открытым ключом для всех сторон;
- безопасность передачи данных посредством преимущественного использования криптографии.

Основное преимущество SET, по сравнению с многими существующими системами обеспечения информационной безопасности, заключается в использовании цифровых сертификатов (стандарт X509, версия 3), которые ассоциируют держателя карты, продавца и банк продавца с рядом банковских учреждений платежных систем Visa и Mastercard. Кроме того, SET позволяет сохранить существующие отношения между банком, держателями карт и продавцами и интегрируется с существующими системами.

Протокол IPSec

Спецификация IPSec входит в стандарт IPv6 и является дополнительной по отношению к текущей версии протоколов TCP/IP. Она разработана Рабочей группой IP Security IETF. В настоящее время IPSec включает три алгоритмо-независимых базовых спецификации, представляющих соответствующие RFC-стандарты. Протокол IPSec обеспечивает стандартный способ шифрования трафика на сетевом (третьем) уровне IP и защищает информацию на основе сквозного шифрования: независимо от работающего приложения, при этом шифруется каждый пакет данных, проходящий по каналу. Это позволяет организациям создавать в Интернете виртуальные частные сети.

ЧАСТЬ 4. ТЕХНОЛОГИИ ЗАЩИТЫ ДАННЫХ

Безопасность данных означает их конфиденциальность, целостность и подлинность.

Конфиденциальность данных предполагает их доступность только для тех лиц, которые имеют на это соответствующие полномочия. Под обеспечением конфиденциальности информации понимается создание таких условий, при которых понять содержание передаваемых данных может только законный получатель, кому и предназначена данная информация.

Целостность информации предполагает ее неизменность в процессе передачи от отправителя к получателю. Под обеспечением целостности информации понимается достижение идентичности отправляемых и принимаемых данных.

Подлинность информации предполагает соответствие этой информации ее описанию и содержанию, в частности соответствие действительным характеристикам: отправителя, времени отправления и содержания. Обеспечение подлинности информации, реализуемое на основе аутентификации, состоит в достоверном установлении отправителя, а также защите информации от изменения при ее передаче от отправителя к получателю.

Большинство средств защиты информации базируется на использовании криптографических шифров и процедур шифрования/дешифрования.

4.1. ОСНОВНЫЕ ПОНЯТИЯ КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ ИНФОРМАЦИИ

Криптография является методологической основой современных систем обеспечения безопасности информации в компьютерных системах и сетях. Исторически криптография (в переводе с греческого этот термин означает «тайнопись») зародилась как способ скрытой передачи сообщений. Криптография представляет собой совокупность методов преобразования данных, направленных на то, чтобы защитить эти данные, сделав их бесполезными для незаконных пользователей. Такие преобразования обеспечивают решение трех главных проблем защиты данных: гарантию конфиденциальности, целостности и подлинности передаваемых или сохраняемых данных. Для обеспечения безопасности данных необходимо поддерживать три основные функции:

- защиту конфиденциальности передаваемых или хранимых в памяти данных;
- подтверждение целостности и подлинности данных;
- аутентификацию абонентов при входе в систему и при установлении соединения.

Для реализации указанных функций используются криптографические технологии шифрования, цифровой подписи и аутентификации. Конфиденциальность обеспечивается с помощью алгоритмов и методов симметричного и асимметричного шифрования, а также путем взаимной аутентификации абонентов на основе многоразовых и одноразовых паролей, цифровых сертификатов, смарт-карт и т.п.

Целостность и подлинность передаваемых данных обычно достигается с помощью различных вариантов технологии электронной подписи, основанных на односторонних функциях и асимметричных методах шифрования. Аутентификация разрешает устанавливать соединения только между легальными пользователями и

предотвращает доступ к средствам сети нежелательных лиц. Абонентам, доказавшим свою легальность (аутентичность), предоставляются разрешенные виды сетевого обслуживания. Основой большинства криптографических средств защиты информации является шифрование данных.

Под шифром понимают совокупность процедур и правил криптографических преобразований, используемых для зашифровывания и дешифрования информации по ключу шифрования. Под зашифровыванием информации понимается процесс преобразования открытой информации (исходного текста) в зашифрованный текст (шифротекст). Процесс восстановления исходного текста по криптограмме с использованием ключа дешифрования называют дешифрованием (расшифровыванием).

Алгоритмы шифрования/дешифрования обычно не являются секретными и публикуются открыто или почти открыто в специальной литературе. Основную нагрузку по защите информации методами шифрования несут ключи. Администрирование ключей призвано придать им необходимые свойства и обеспечить нормальное функционирование на всех стадиях жизни (использования) ключей. Стадиями жизни ключей являются:

- генерация или формирование;
- распределение;
- верификация и аутентификация;
- хранение;
- использование;
- модификация;
- ликвидация или утилизация.

Для обеспечения необходимого уровня защиты информации в информационных системах, к алгоритмам шифрования и системе ключей предъявляются определенные требования:

1) криптограмма (шифротекст) должна дешифроваться только при наличии ключа;
2) число операций, необходимое для вскрытия ключа по открытому тексту и соответствующей ему криптограмме, должно быть не меньше числа всех возможных ключей;
3) знание алгоритма шифрования не должно упрощать процедуры дешифрования, выполняемого с целью вскрытия ключей и дешифрования криптограмм;
4) незначительное изменение ключа, с использованием которого шифруется открытый текст, должно приводить к существенному изменению соответствующей криптограммы;
5) незначительное изменение открытого текста при неизменном ключе должно приводить к существенному изменению соответствующей криптограммы;
6) структура алгоритма шифрования должна быть постоянной;
7) в процессе шифрования должен быть предусмотрен контроль за шифруемым открытым текстом и ключом;
8) длина криптограммы должна быть равна длине открытого текста;
9) сложность вскрытия ключа, используемого для шифрования очередного открытого текста, по последовательности ключей, использованных для шифрования предшествующих текстов, должна быть сопоставимой со

сложностью вскрытия ключа по открытому тексту и соответствующей ему криптограмме;

10) множество всех возможных ключей должно быть однородным и не должно содержать «слабых» ключей, применительно к которым процедуры криптоанализа относительно более просты и эффективны;
11) криптограмма должна быть однородной, то есть не должна делиться на фрагменты (символы, биты), одни из которых априори (умозрительно или интуитивно) известны как относящиеся к открытому тексту, а другие – как включенные в шифротекст в процессе шифрования;
12) алгоритм шифрования должен допускать как программную, так и аппаратную реализацию;
13) изменение длины ключа не должны ухудшать характеристики алгоритма шифрования.

Алгоритм шифрования, удовлетворяющий перечисленным требованиям, считается криптостойким и пригодным для использования для защиты информации в информационных системах.

Обобщенная схема криптосистемы шифрования приведена на рис. 4.1. Исходный текст передаваемого сообщения (или хранимой информации) M зашифровывается с помощью криптографического преобразования E_{k1} с получением в результате шифротекста C:

$$C = E_{k1}(M), \tag{4.1}$$

где k1 — параметр функции E, называемый ключом шифрования.

Шифротекст C, называемый еще криптограммой, содержит исходную информацию M в полном объеме, однако последовательность знаков в нем внешне представляется случайной и не позволяет восстановить исходную информацию без знания ключа шифрования k1.

Ключ шифрования является тем элементом, с помощью которого можно варьировать результат криптографического преобразования. Ключ может принадлежать конкретному пользователю или группе пользователей и являться для них уникальным. Обычно ключ шифрования представляет собой файл или массив данных и хранится на персональном ключевом носителе, например дискете или смарт-карте. Зашифрованная с использованием конкретного ключа информация может быть расшифрована только его владельцем (или владельцами).

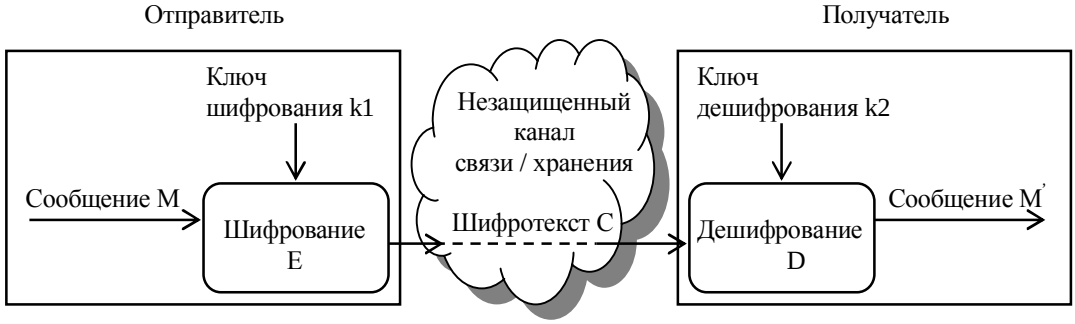


Рис. 4.1. Обобщенная схема криптосистемы шифрования

Обратное преобразование имеет следующий вид:

$$M' = D_{k2}(C). \tag{4.2}$$

Функция D является обратной к функции E и производит дешифрование шифротекста. Она также имеет дополнительный параметр в виде ключа k2. Ключ дешифрования k2 должен однозначно соответствовать ключу k1, в этом случае полученное в результате дешифрования сообщение M' будет эквивалентно M. При отсутствии верного ключа k2 получить исходное сообщение M' = M с помощью функции D невозможно.

Преобразование шифрования может быть симметричным или асимметричным относительно преобразования дешифрования. Соответственно, различают два основных класса криптосистем:

1) симметричные криптосистемы;
2) асимметричные криптосистемы.

Известно несколько классификаций криптографических алгоритмов (КА). Одна из них подразделяет КА в зависимости от числа ключей, применяемых в конкретном алгоритме:

1) бесключевые КА - не используют в вычислениях никаких ключей;
2) одноключевые КА - работают с одним ключевым параметром (секретным ключом);
3) двухключевые КА – на различных стадиях работы в них применяются два ключевых параметра: секретный и открытый ключи.

Более детальная классификация приведена на рис. 4.2.

Хэширование - это метод криптозащиты, представляющий собой контрольное преобразование информации: из данных неограниченного размера путем выполнения криптографических преобразований вычисляется хэш-значение (хэш-функция) фиксированной длины, однозначно соответствующее исходным данным. Хэширование может выполняться как с использованием некоторого секретного ключа, так и без него. Такое криптографическое контрольное суммирование широко используется в различных методах защиты информации, в частности, для подтверждения целостности данных, если использование электронной подписи невозможно (например, из-за большой ресурсоемкости) или избыточно. Кроме того, данный метод применяется в схемах электронной подписи («подписывается» обычно хэш-значение данных, а не все данные целиком), а также в схемах аутентификации пользователей (при проверке, действительно ли пользователь является тем, за кого себя выдает).

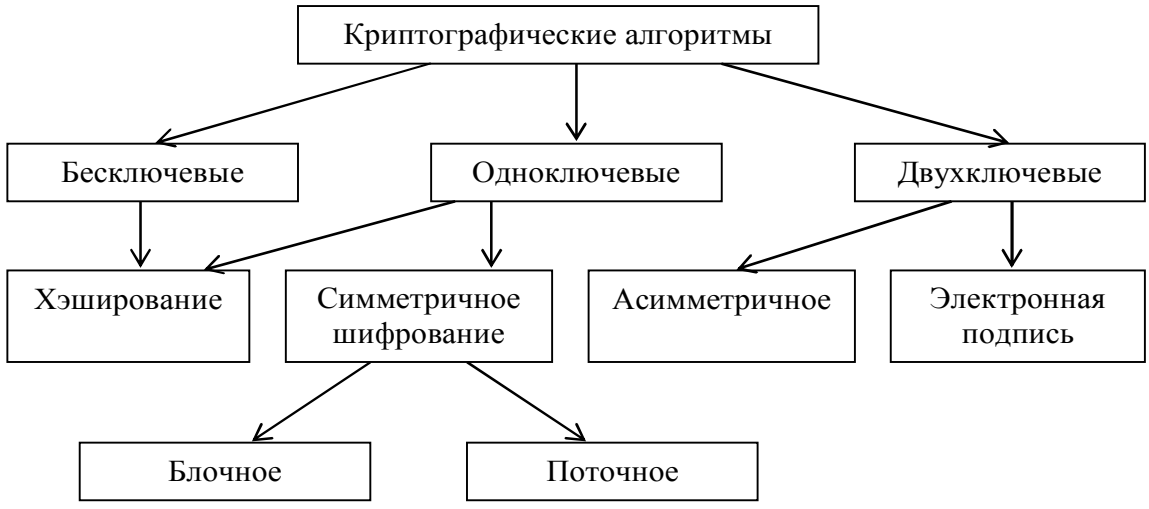


Рис. 4.2. Классификация криптографических алгоритмов защиты информации

Симметричное шифрование использует один и тот же ключ как для зашифровывания, так и для дешифрования информации. Фактически оба ключа (шифрования и дешифрования) могут и различаться, но если в каком-либо КА их легко вычислить один из другого, такой алгоритм однозначно относится к симметричному шифрованию.

Симметричное шифрование подразделяется на два вида: блочное и поточное, хотя стоит сразу отметить, что в некоторых классификациях они не разделяются и считается, что поточное шифрование - это шифрование блоков единичной длины.

Блочное шифрование характеризуется тем, что информация предварительно разбивается на блоки фиксированной длины (например, 64 или 128 бит). При этом в различных КА или даже в разных режимах работы одного и того же алгоритма блоки могут шифроваться как независимо друг от друга, так и «со сцеплением» - когда результат шифрования текущего блока данных зависит от значения предыдущего блока или от результата шифрования предыдущего блока.

Поточное шифрование применяется прежде всего тогда, когда информацию невозможно разбить на блоки, - скажем, есть некий поток данных, каждый символ которых требуется зашифровать и отправить, не дожидаясь остальных данных, достаточных для формирования блока. Алгоритмы поточного шифрования шифруют данные побитно или посимвольно.

Асимметричное шифрование характеризуется применением двух типов ключей: открытого - для шифрования информации - и секретного - для ее дешифрования. Секретный и открытый ключи связаны между собой достаточно сложным соотношением или могут быть вообще независимыми – ортогональными. Главное в этом соотношении – легкость вычисления открытого ключа из секретного и невозможность, за ограниченное время при реальных ресурсах, вычисления секретного ключа из открытого.

Электронная цифровая подпись (ЭЦП) используется для подтверждения целостности и авторства данных. Как и в случае асимметричного шифрования, в данном методе применяются двухключевые алгоритмы с таким же простым вычислением открытого ключа из секретного и практической невозможностью обратного вычисления. Однако назначение ключей ЭЦП совершенно иное. Секретный ключ применяется для вычисления ЭЦП, открытый ключ необходим для ее проверки. При соблюдении правил безопасного хранения секретного ключа никто, кроме его владельца, не в состоянии вычислить верную ЭЦП какого-либо электронного документа.

4.2. СИММЕТРИЧНЫЕ КРИПТОСИСТЕМЫ ШИФРОВАНИЯ

Исторически первыми появились симметричные криптографические системы. В симметричной криптосистеме шифрования используется один и тот же ключ для шифрования и дешифрования информации. Это означает, что любой, кто имеет доступ к ключу шифрования, может расшифровать сообщение. Соответственно, с целью предотвращения несанкционированного раскрытия зашифрованной информации, все ключи шифрования в симметричных криптосистемах должны держаться в секрете. Именно поэтому симметричные криптосистемы называют криптосистемами с секретным ключом – ключ шифрования должен быть доступен только тем, кому предназначено сообщение. Симметричные криптосистемы называют еще одноключевыми криптографическими системами или криптосистемами с закрытым ключом.

Схема симметричной криптосистемы шифрования показана на рис. 4.3. Данные криптосистемы характеризуются наиболее высокой скоростью шифрования, и с их помощью обеспечивается как конфиденциальность и подлинность, так и целостность передаваемой информации.

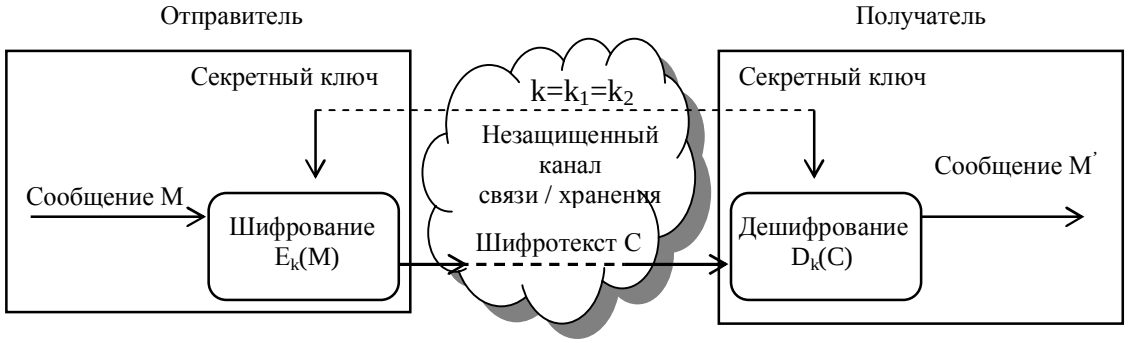


Рис. 4.3. Схема симметричной криптосистемы шифрования

Конфиденциальность передачи информации с помощью симметричной криптосистемы зависит от надежности шифра и обеспечения конфиденциальности ключа шифрования. Обычно ключ шифрования представляет собой файл или массив данных и хранится на персональном ключевом носителе, например дискете или смарт-карте; обязательно принятие мер, обеспечивающих недоступность персонального ключевого носителя кому-либо, кроме его владельца.

Подлинность обеспечивается за счет того, что без предварительного расшифровывания практически невозможно осуществить смысловую модификацию и подлог криптографически закрытого сообщения. Фальшивое сообщение не может быть правильно зашифровано без знания секретного ключа.

Целостность данных обеспечивается присоединением к передаваемым данным специального кода (имитовставки), вырабатываемого по секретному ключу. **Имитовставка** является разновидностью контрольной суммы, то есть некоторой эталонной характеристикой сообщения, по которой осуществляется проверка целостности последнего. Алгоритм формирования имитовставки должен обеспечивать ее зависимость по некоторому сложному криптографическому закону от каждого бита сообщения. Проверка целостности сообщения выполняется получателем сообщения путем выработки по секретному ключу имитовставки, соответствующей полученному сообщению, и ее сравнения с полученным значением имитовставки. При совпадении делается вывод о том, что информация не была модифицирована на пути от отправителя к получателю.

Симметричное шифрование идеально подходит в случае шифрования информации «для себя», например, с целью предотвратить несанкционированный доступ к ней в отсутствие владельца. Это может быть как архивное шифрование выбранных файлов, так и прозрачное (автоматическое) шифрование целых логических или физических дисков.

Обладая высокой скоростью шифрования, одноключевые криптосистемы позволяют решать многие важные задачи защиты информации. Однако автономное использование симметричных криптосистем в компьютерных сетях порождает проблему распределения ключей шифрования между пользователями.

Перед началом обмена зашифрованными данными необходимо обменяться секретными ключами со всеми адресатами. Передача секретного ключа

симметричной криптосистемы не может быть осуществлена по общедоступным каналам связи, секретный ключ надо передавать отправителю и получателю по защищенному каналу.

Существуют реализации алгоритмов симметричного шифрования для абонентского шифрования данных – то есть для отправки шифрованной информации абоненту, например, через Интернет. Использование одного ключа для всех абонентов подобной криптографической сети недопустимо по соображениям безопасности. Действительно, в случае компрометации (утери, хищения) ключа, под угрозой будет находиться документооборот всех абонентов. В этом случае может быть использована матрица ключей:

	1	2	3	…	n	
1	K_{11}	K_{12}	K_{13}	…	K_{1n}	Набор ключей для абонента 1
2	K_{21}	K_{22}	K_{23}	…	K_{2n}	Набор ключей для абонента 2
3	K_{31}	K_{32}	K_{33}	…	K_{3n}	Набор ключей для абонента 3
4	K_{41}	K_{42}	K_{43}	…	K_{4n}	Набор ключей для абонента 4
…	..	…	…	…	…	…
n	K_{n1}	K_{n2}	K_{n3}	…	K_{nn}	Набор ключей для абонента n

Матрица ключей представляет собой таблицу, содержащую ключи парной связи абонентов. Каждый элемент таблицы K_{ij} предназначен для связи абонентов i и j и доступен только двум данным абонентам. Соответственно, для всех элементов матрицы ключей соблюдается равенство $K_{ij} = K_{ji}$

Каждая i-я строка матрицы представляет собой набор ключей конкретного абонента i для связи с остальными $(N - 1)$ абонентами. Наборы ключей (сетевые наборы) распределяются между всеми абонентами криптографической сети. Аналогично сказанному выше, сетевые наборы должны распределяться по защищенным каналам связи или «из рук в руки».

Характерной особенностью симметричных криптоалгоритмов является то, что в ходе своей работы они производят преобразование блока входной информации фиксированной длины и получают результирующий блок того же объема, но недоступный для прочтения сторонним лицам, не владеющим ключом. Схему работы симметричного блочного шифра можно описать функциями

$$C = E_K(M) \quad \text{и} \quad M = D_K(C),$$

где M – исходный (открытый) блок данных, C – зашифрованный блок данных.

Ключ K является параметром симметричного блочного криптоалгоритма и представляет собой блок двоичной информации фиксированного размера. Исходный M и зашифрованный C блоки данных также имеют фиксированную разрядность, равную между собой, но необязательно равную длине ключа K.

Блочные шифры являются той основой, на которой реализованы практически все симметричные криптосистемы. Симметричные криптосистемы позволяют кодировать и декодировать файлы произвольной длины. Практически все алгоритмы используют для преобразований определенный набор обратимых математических преобразований.

Методика создания цепочек из зашифрованных блочными алгоритмами байтов позволяет шифровать ими пакеты информации неограниченной длины. Отсутствие

статистической корреляции между битами выходного потока блочного шифра используется для вычисления контрольных сумм пакетов данных и в хэшировании паролей. На сегодняшний день разработано достаточно много стойких блочных шифров.

Криптоалгоритм считается идеально стойким, если для прочтения зашифрованного блока данных необходим перебор всех возможных ключей до тех пор, пока расшифрованное сообщение не окажется осмысленным. В общем случае стойкость блочного шифра зависит только от длины ключа и возрастает экспоненциально с ее ростом.

К. Шеннон предложил для получения стойких блочных шифров использовать два общих принципа: рассеивание и перемешивание.

Рассеивание представляет собой распространение влияния одного знака открытого текста на много знаков шифротекста, что позволяет скрыть статистические свойства открытого текста.

Перемешивание предполагает использование таких шифрующих преобразований, которые усложняют восстановление взаимосвязи статистических свойств открытого и шифрованного текстов. Однако шифр должен не только затруднять раскрытие, но и обеспечивать легкость шифрования и дешифрования при известном пользователю секретном ключе.

Распространенным способом достижения эффектов рассеивания и перемешивания является использование составного шифра, то есть такого шифра, который может быть реализован в виде некоторой последовательности простых шифров, каждый из которых вносит свой вклад в значительное суммарное рассеивание и перемешивание.

В составных шифрах в качестве простых шифров чаще всего используются простые перестановки и подстановки. При перестановке просто перемешивают символы открытого текста, причем конкретный вид перемешивания определяется секретным ключом. При подстановке каждый символ открытого текста заменяют другим символом из того же алфавита, а конкретный вид подстановки также определяется секретным ключом. Следует заметить, что в современном блочном шифре блоки открытого текста и шифротекста представляют собой двоичные последовательности обычно длиной 64 или 128 бит. При длине 64 бит каждый блок может принимать 2^{64} значений. Поэтому подстановки выполняются в очень большом алфавите, содержащем до $2^{64} \approx 1019$ «символов».

При многократном чередовании простых перестановок и подстановок, управляемых достаточно длинным секретным ключом, можно получить стойкий шифр с хорошим рассеиванием и перемешиванием.

Все действия производимые блочным криптоалгоритмом над данными, основаны на том факте, что преобразуемый блок может быть представлен в виде целого неотрицательного числа из диапазона, соответствующего его разрядности. Например, 32-битовый блок данных можно интерпретировать как число из диапазона 0–4294967295. Кроме того, блок, разрядность которого представляет собой степень числа 2, можно трактовать как сцепление нескольких независимых неотрицательных чисел из меньшего диапазона (указанный выше 32-битовый блок можно также представить в виде сцепления двух независимых 16-битовых чисел из диапазона 0–65535 или в виде сцепления четырех независимых 8-битовых чисел из диапазона 0–255).

Над этими числами блочный криптоалгоритм производит действия по определенной схеме, представленные в табл. 4.1.

Таблица 4.1. Действия, выполняемые криптоалгоритмом над числами

Действие	Функция
Математические функции	
Сложение	X = X + V
Исключающее ИЛИ	X = X XOR V
Умножение по модулю (2N + 1)	X' = (X * V) mod (2N + 1)
Умножение по модулю 2N	X' = (X * V) mod (2N)
Битовые сдвиги	
Арифметический сдвиг влево	X = X SHL V
Арифметический сдвиг вправо	X = X SHR V
Циклический сдвиг влево	X = X ROL V
Циклический сдвиг вправо	X = X ROR V
Табличные подстановки	
S-box (substitute)	X' = Table [X, V]

В качестве параметра V для любого из этих преобразований может использоваться:

- фиксированное число (например, X = X + 125);
- число, получаемое из ключа (например, X = X + F(K));
- число, получаемое из независимой части блока (например, X2' =X2 + F(X1)).

Последний вариант используется в схеме, называемой сетью Фейстеля (по имени ее создателя). Последовательность выполняемых над блоком операций, комбинации перечисленных выше вариантов *V* и сами функции *F* и составляют отличительные особенности конкретного симметричного блочного криптоалгоритма.

Характерным признаком блочных криптоалгоритмов является многократное и косвенное использование материала ключа. Это определяется требованием невозможности обратного декодирования в отношении ключа при известных исходном и зашифрованном текстами. Для решения этой задачи в приведенных выше преобразованиях чаще всего используется не само значение ключа или его части, а некоторая, иногда необратимая, функция от материала ключа. Более того, в подобных преобразованиях один и тот же блок или элемент ключа используется многократно. Это позволяет при выполнении условия обратимости функции относительно величины X сделать функцию необратимой относительно ключа K.

4.2.1. АЛГОРИТМ ШИФРОВАНИЯ ДАННЫХ DES

Алгоритм шифрования данных DES (Data Encryption Standard) был опубликован в 1977 году. Блочный симметричный алгоритм DES остается пока наиболее распространенным алгоритмом, используемым в системах защиты коммерческой информации. Алгоритм DES построен в соответствии с методологией сети Фейстеля и состоит из чередующейся последовательности перестановок и подстановок. Алгоритм DES осуществляет шифрование 64-битовых блоков данных с помощью 64-битового ключа, в котором значащими являются 56 бит (остальные 8 бит – проверочные биты для контроля на четность). Обобщенная схема процесса шифрования в блочном алгоритме DES показана на рис. 4.4.

$C = DES(M, K)$.

Алгоритм **шифрования** DES функционирует следующим образом.

Шаг 1 – выполняется начальная перестановка *IP* входного блока *M*. Порядок перестановки входного блока приведен в табл. 4.2. Так, первый бит входного блока помещается в позицию 58, второй бит – в позицию 50 и т.д.

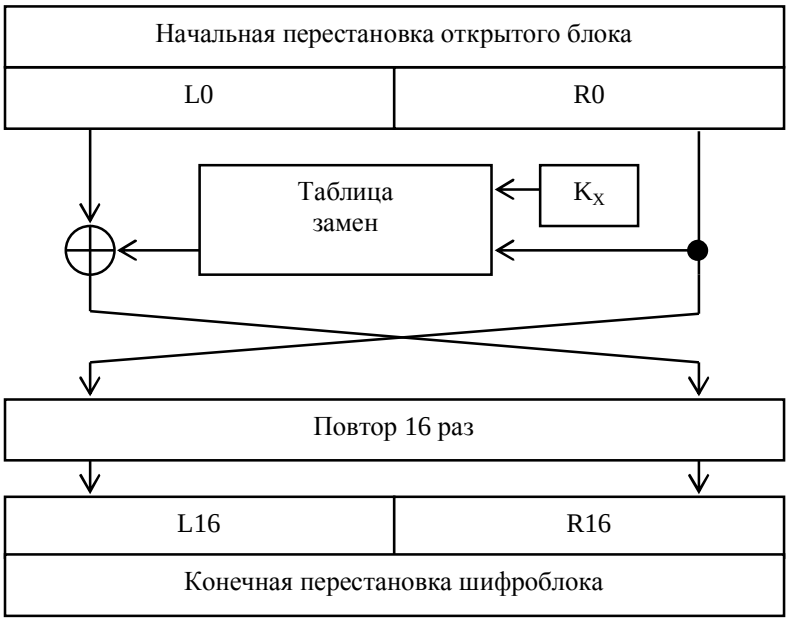


Рис. 4.4. Схема шифрования по алгоритму DES

Результатом шифрования 64-битного блока открытого теста *M* при помощи ключа *K* является 64-битный шифротекст *C*, что обозначается равенством

Таблица 4.2. Начальная перестановка IP

1→58	9→60	17→62	25→64	33→57	41→59	49→61	57→63
2→50	10→52	18→54	26→56	34→49	42→51	50→53	58→55
3→42	11→44	19→46	27→48	35→41	43→43	51→45	59→47
4→34	12→36	20→38	28→40	36→33	44→35	52→37	60→39
5→26	13→28	21→30	29→32	37→25	45→27	53→29	61→31
6→18	14→20	22→22	30→24	38→17	46→19	54→21	62→23
7→10	15→12	23→14	31→16	39→9	47→11	55→13	63→15
8→2	16→4	24→6	32→8	40→1	48→3	56→5	64→7

Шаг 2 – после начальной перестановки 32 младших бита (32..1) образуют правый полублок – R0, а 32 старших бита (64..33) образуют левый полублок – L0.

Шаг 3 – правый полублок R0 без изменения переносится в левый полублок – L1. Правый полублок R1 образуется сложением «по модулю 2» левого полублока L0 и результата табличной замены правого полублока R0 функций $F(R0, K1)$, аргументами которой являются 32 бита правого полублока R0 и 48 бит ключевой последовательности K1. Шаг 3 выполняется 16 раз, пока не сформируются полублоки R16 и L16.

$L_i = R_{i-1}$,

$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$.

Шаг 4 – полублоки R16 и L16 объединяются в 64-битный выходной блок.

Шаг 5 – выполняется конечная перестановка IP^{-1} выходного блока в шифроблок *C*. Порядок перестановки приведен в табл. 4.3. Так, первый бит выходного блока помещается в позицию 40 шифроблока, второй бит – в позицию 8, третий бит – в позицию 48 и т.д.

Таблица 4.3. Конечная перестановка IP^{-1}

1→40	9→39	17→38	25→37	33→36	41→35	49→34	57→33
2→8	10→7	18→6	26→5	34→4	42→3	50→2	58→1
3→48	11→47	19→46	27→45	35→44	43→43	51→42	59→41
4→16	12→15	20→14	28→13	36→12	44→11	52→10	60→9
5→56	13→55	21→54	29→53	37→52	45→51	53→50	61→49
6→24	14→23	22→22	30→21	38→20	46→19	54→18	62→17
7→64	15→63	23→62	31→61	39→60	47→59	55→58	63→57
8→32	16→31	24→30	32→29	40→28	48→27	56→26	64→25

Структурная схема алгоритма работы функции $F(R_{i-1},K_i)$ приведена на рис. 4.5. Вычисление начинается с расширения *E* очередного 32-битного полублока R_{i-1} в 48-битный блок. Логика расширения приведена в табл. 4.4. Результат расширения суммируется «по модулю 2» с 48-битной ключевой последовательностью K_i.

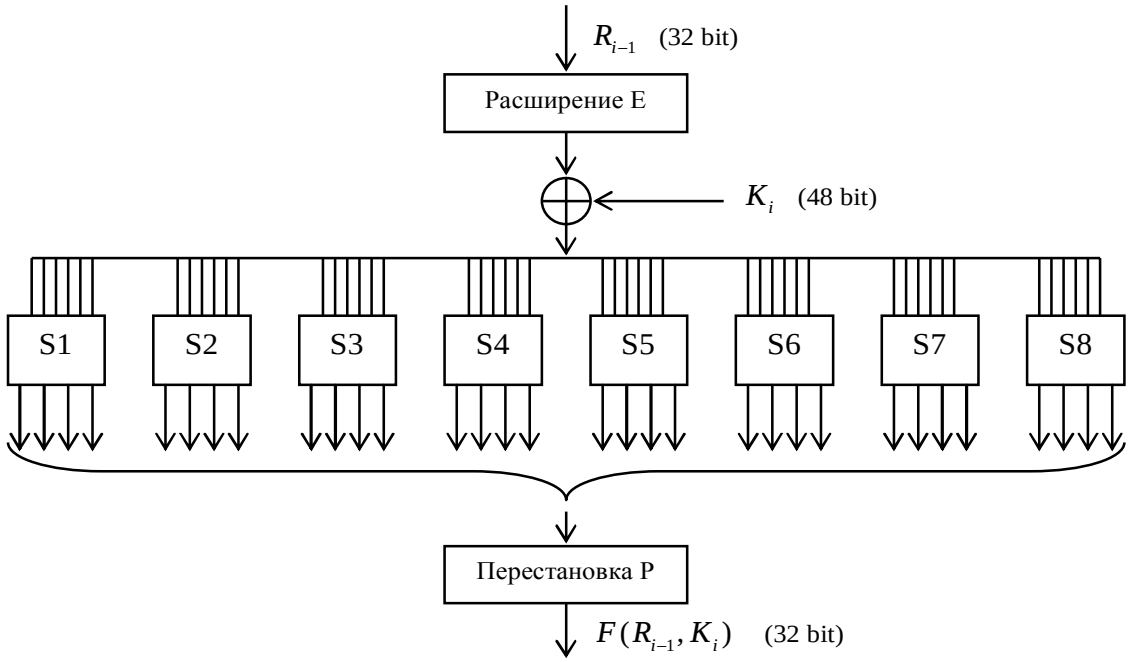


Рис.4.5. Структурная схема алгоритма работы функции $F(R_{i-1},K_i)$

Полученный 48-битный блок разбивается на восемь 6-битных векторов: $S_1, S_2,\ldots,S_8$. Входные 6 бит каждого вектора – $b_1b_2b_3b_4b_5b_6$, заменяются на 4-битное число – $d_1d_2d_3d_4$ из табл. 4.5 для соответствующего вектора. 4-Битное число выбирается на пересечении строки с номером $r_j=b_1b_6$ и столбца с номером $c_k=b_2b_3b_4b_5$. 4-Битные

выходы восьми векторов объединяются в выходной блок, биты которого подвергаются перестановке *P*, табл. 4.6.

Пример

Пусть на вход S_3 поступает вектор V_{in}=100110. Тогда выходом S_3 будет вектор V_{out}=1001 – число $9_{10}=1001_2$, взятое из табл. 4.5-S_3 на пересечении строки $r_2=b_1b_6=10$ и столбца $c_3=b_2b_3b_4b_5=0011$.

Таблица 4.4. Расширение E

1→48, 2	9→12, 14	17→24, 26	25→36, 38
2→3	10→15	18→27	26→39
3→4	11→16	19→28	27→40
4→5, 7	12→17, 19	20→29, 31	28→41, 43
5→6, 8	13→18, 20	21→30, 32	29→42, 44
6→9	14→21	22→33	30→45
7→10	15→22	23→34	31→46
8→11, 13	16→23, 25	24→35, 37	32→47, 1

Ключевые последовательности

Ключевые последовательности K_i, i=1,…,16, используются в функции $F(R_{i-1},K_i)$ в ходе выполнения 16 циклов блочного шифрования *DES*.

Алгоритм формирования ключевых последовательностей представлен структурной схемой на рис. 4.6.

Исходный 64-битный ключ K разбивается на восемь 8-битных блоков. Восьмой бит каждого блока служит для контроля четности, который позволяет выявлять ошибки типа искажения нечетного числа битов (подобные ошибки могут возникать при хранении или непосредственно в ходе формирования ключей). При

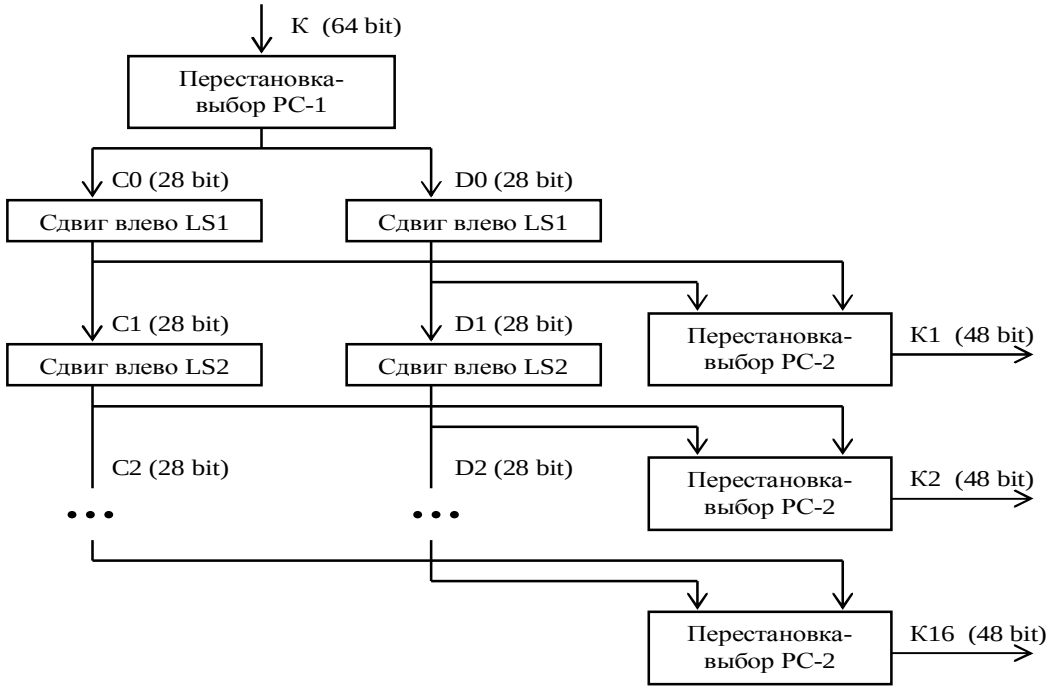


Рис. 4.6. Структурная схема алгоритма формирования ключевых последовательностей DES

этом значение указанного восьмого бита представляет собой результат сложения «по модулю 2» предшествующих семи битов. Контрольные биты в процессе формирования ключей не используются.

Шаг 1 – выполняется операция перестановки-выбора *PC-1* битов исходного ключа *K*, согласно табл. 4.7. Из 56 битов ключа формируется два 28-битных полублока *C0* и *D0*.

Шаг 2 – полублоки C_i и D_i циклически сдвигаются влево на LS_i позиций согласно табл. 4.9, и образуют следующие полублоки C_{i+1} и D_{i+1}.

Шаг 3 – полублоки C_i и D_i объединяются, подвергаются перестановке-выбору *PC-2* по табл. 4.8 и образуют соответствующую 48-битную ключевую последовательность K_i.

Таблица 4.5. Замена векторов

	c	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S1	r0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	r1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	r2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	r3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S2	r0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	r1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	r2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	r3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S3	r0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	r1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	r2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	r3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S4	r0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	r1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	r2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	r3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S5	r0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	r1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	r2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	r3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S6	r0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	r1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	r2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	r3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S7	r0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	r1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	r2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	r3	6	11	3	8	1	4	10	7	9	5	0	15	14	2	3	12
S8	r0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	r1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	r2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	r3	2	1	14	7	4	10	8	13	5	12	9	0	3	5	5	11

Шаги 2 и 3 повторяются, пока не будут сформированы все 16 ключевых последовательностей K_i, i=1,…,16. Последовательности формируются один раз и используются для шифрования и дешифрования для заданного ключа K.

Дешифрование в DES является операцией, обратной шифрованию, и выполняется путем повторения операций шифрования в обратной последовательности.

$R_{i-1} = L_i$

$L_{i-1} = R_i \oplus F(R_{i-1}, K_i)$.

Таблица 4.6. Перестановка P

1→9	9→24	17→8	25→32
2→17	10→16	18→14	26→12
3→23	11→30	19→25	27→22
4→31	12→6	20→3	28→7
5→13	13→26	21→4	29→5
6→28	14→20	22→29	30→27
7→2	15→10	23→11	31→15
8→18	16→1	24→19	32→21

Таблица 4.7. Перестановка PC-1

1→8	9→7	17→6	25→5	33→4	41→3	49→2	57→1
2→16	10→15	18→14	26→13	34→12	42→11	50→10	58→9
3→24	11→23	19→22	27→21	35→20	43→19	51→18	59→17
4→56	12→55	20→54	28→53	36→28	44→27	52→26	60→25
5→52	13→51	21→50	29→49	37→48	45→47	53→46	61→45
6→44	14→43	22→42	30→41	38→40	46→39	54→38	62→37
7→36	15→35	23→34	31→33	39→32	47→31	55→30	63→29

Таблица 4.8. Перестановка PC-2

1→5	8→18	15→9	22	29→47	36→46	43	50→45
2→24	9	16→19	23→13	30→31	37→28	44→37	51→33
3→7	10→12	17→2	24→4	31→27	38	45→34	52→26
4→16	11→3	18	25	32→48	39→39	46→43	53→42
5→6	12→15	19→14	26→17	33→35	40→32	47→29	54
6→10	13→23	20→22	27→21	34→41	41→25	48→36	55→30
7→20	14→1	21→11	28→8	35→	42→44	49→38	56→40

Таблица 4.9. Порядок сдвига LS

LS1	LS2	LS3	LS4	LS5	LS6	LS7	LS8	LS9	LS10	LS11	LS12	LS13	LS14	LS15	LS16
1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Основные достоинства алгоритма DES:

- используется только один ключ длиной 56 бит;
- относительная простота алгоритма обеспечивает высокую скорость обработки;

- зашифровав сообщение с помощью одного пакета программ, для дешифровки можно использовать любой другой пакет программ, соответствующий алгоритму DES;
- криптостойкость алгоритма вполне достаточна для обеспечения информационной безопасности большинства коммерческих приложений.

Современная микропроцессорная техника позволяет уже сегодня за достаточно приемлемое время взламывать симметричные блочные шифры с длиной ключа 40 бит. Для такого взламывания используется метод полного перебора - тотального опробования всех возможных значений ключа (метод «грубой силы»).

До недавнего времени блочный алгоритм DES, имеющий ключ с эффективной длиной 56 бит, считался относительно безопасным алгоритмом шифрования. Он, в течение 20 лет, многократно подвергался тщательному криптоанализу, и самым практичным способом его взламывания является метод перебора всех возможных значений ключа. Ключ шифра DES имеет 2^{56} возможных значений.

В настоящее время на рынок поступили FPGA-чипы, обладающие возможностью перебирать до 30 миллионов значений ключа в секунду. Еще большие возможности имеют ASIC-чипы - они реализуют скорость перебора до 200 миллионов ключей в секунду. Стоимость этих чипов составляет всего лишь десятки долларов. Поэтому вполне актуальны оценки криптостойкости шифра DES, включающие ориентировочные расчеты времени и материальных средств, которые необходимо затратить на взламывание этого шифра методом полного перебора всех возможных значений ключа с использованием как стандартных компьютеров, так и специализированных криптоаналитических аппаратных средств. В табл. 4.10 приведены результаты анализа трудоемкости взламывания криптоалгоритма DES [xxx].

Таблица 4.10. Сравнительный анализ трудоемкости взлома криптоалгоритма DES

№ п/п	Тип атакующего	Бюджет атакующего	Средства атаки	Затраты времени на успешную атаку
1	Хакер	До 500 долларов	ПК	Несколько десятков лет
2	Небольшие фирмы	До 10 тыс. долларов	FPGA	18 месяцев
3	Корпоративные департаменты	До 300 тыс. долларов	FPGA ASIC	19 дней 3 дня
4	Большие корпорации	До 10 млн. долларов	FPGA, ASIC суперЭВМ	13 часов 6 минут
5	Специальные агентства	?	?	?

Возникает естественный вопрос: нельзя ли использовать DES в качестве строительного блока для создания другого алгоритма с более длинным ключом?

Комбинирование блочных алгоритмов

В принципе, существует много способов комбинирования блочных алгоритмов для получения новых алгоритмов. Одним из таких способов комбинирования является многократное шифрование, то есть использование

блочного алгоритма несколько раз с разными ключами для шифрования одного и того же блока открытого текста.

У. Тачмен предложил шифровать блок открытого текста P три раза с помощью двух ключей K1 и K2 (рис. 4.7) . Процедура шифрования:

$$C = E_K(D_K(E_K(P))),$$

то есть блок открытого текста P сначала шифруется ключом K_1, затем расшифровывается ключом K_2 и окончательно зашифровывается ключом K_1. Этот режим иногда называют режимом EDE (encrypt-decrypt-encrypt). Введение в данную схему операции дешифрования DK позволяет обеспечить совместимость этой схемы со схемой однократного использования блочного алгоритма DES. Если в схеме трехкратного использования DES выбрать все ключи одинаковыми, то эта схема превращается в схему однократного использования DES. Процедура дешифрования выполняется в обратном порядке:

$$P = D_K(E_K(D_K(C))),$$

то есть блок шифртекста C сначала расшифровывается ключом K_1, затем зашифровывается ключом K_2 и окончательно расшифровывается ключом K_1.

Рис. 4.7. Схемы трехкратного применения блочного алгоритма симметричного шифрования с двумя различными ключами

Если исходный блочный алгоритм имеет n-битовый ключ, то схема трехкратного шифрования имеет $2n$-битовый ключ. Данная схема приводится в стандартах X9.17 и ISO 8732 в качестве средства улучшения характеристик алгоритма DES.

При трехкратном шифровании можно применить три различных ключа. При этом возрастает общая длина результирующего ключа. Процедуры шифрования и дешифрования описываются выражениями

$$C = E_K(D_K(E_K(P))),$$

$$P = D_K(E_K(D_K(C))).$$

Трехключевой вариант имеет еще большую стойкость. Алгоритм 3-DES (Triple DES – тройной DES) используется в ситуациях, когда надежность алгоритма DES считается недостаточной. Чаще всего используется вариант шифрования на трех ключах: открытый текст шифруется на первом ключе, полученный шифртекст – на втором ключе и наконец данные, полученные после второго шага, шифруются на третьем ключе. Все три ключа выбираются независимо друг от друга. Этот криптоалгоритм достаточно стоек ко всем атакам. Применяется также каскадный вариант 3-DES. Это стандартный тройной DES, к которому добавлен такой механизм обратной связи, как CBC, OFB или CFB.

Сегодня все шире используются два современных криптостойких алгоритма шифрования: отечественный стандарт шифрования ГОСТ 28147-89 и новый криптостандарт США – AES (Advanced Encryption Standard).

4.2.2. СТАНДАРТ ШИФРОВАНИЯ ГОСТ 28147-89

Этот алгоритм криптографического преобразования данных предназначен для аппаратной и программной реализации, удовлетворяет криптографическим требованиям и не накладывает ограничений на степень секретности защищаемой информации. Алгоритм шифрования данных, определяемый ГОСТ 28147-89, представляет собой 64-битовый блочный алгоритм с 256-битовым ключом.

Данные, подлежащие шифрованию, разбивают на 64-разрядные блоки. Эти блоки разбиваются на два полублока L0 и R0 по 32 бит (рис. 4.8). Полублок L0 обрабатывается определенным образом, после чего его значение складывается со значением полублока R0 (сложение выполняется «по модулю 2», то есть применяется логическая операция XOR – исключающее ИЛИ), а затем полублоки меняются местами. Данное преобразование выполняется определенное число раз (раундов): 16 или 32 в зависимости от режима работы алгоритма. В каждом раунде выполняются следующие операции.

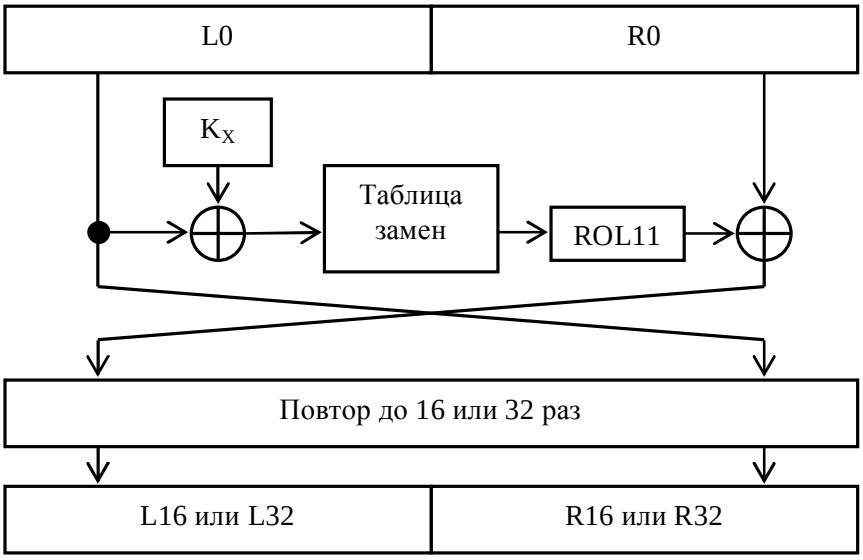


Рис. 4.8. Схема алгоритма ГОСТ 28147-89

Первая операция – наложение ключа. Содержимое полублока L0 складывается по модулю 2^{32} с 32-битовой частью ключа K_X. Полный ключ шифрования представляется в виде конкатенации 32-битовых подключей: K0, K1, K2, K3, K4, K5, K_6, K7. В процессе шифрования используется один из этих подключей – в зависимости от номера раунда и режима работы алгоритма.

Вторая операция – табличная замена. После наложения ключа полублок L0 разбивается на 8 частей по 4 бита, значение каждой из которых заменяется в соответствии с таблицей замены для данной части полублока. Затем выполняется побитовый циклический сдвиг полублока влево на 11 бит (ROL11). Табличные замены. Блок подстановки S-box (Substitution box) часто используются в современных алгоритмах шифрования, поэтому стоит пояснить, как организуется подобная операция.

Блок подстановки S-box состоит из восьми узлов замены (S-блоков замены) S0, S1 , ..., S7 с памятью 64 бит каждый. Поступающий на блок подстановки S 32-битовый вектор разбивают на восемь последовательно идущих 4-битовых векторов, каждый из которых преобразуется в 4-битовый вектор соответствующим узлом замены.

Каждый узел замены можно представить в виде таблицы-перестановки шестнадцати 4-битовых двоичных чисел в диапазоне 0000–1111. Входной вектор указывает адрес строки в таблице, а число в этой строке является выходным вектором. Затем 4-битовые выходные векторы последовательно объединяют в 32-битовый вектор. Узлы замены (таблицы-перестановки) представляют собой ключевые элементы, которые являются общими для сети ЭВМ и редко изменяются. Эти узлы замены должны сохраняться в секрете.

Алгоритм, определяемый ГОСТ 28147-89, предусматривает четыре режима работы: простой замены, гаммирования, гаммирования с обратной связью и генерации имитовставок. В них используется одно и то же описанное выше шифрующее преобразование, но, поскольку назначение режимов различно, осуществляется это преобразование в каждом из них по-разному.

В режиме простой замены для шифрования каждого 64-битового блока информации выполняются 32 описанных выше раунда. При этом 32-битовые подключи используются в следующей последовательности:

- K0, K1, K2, K3, K4, K5, K6, K7, K0, K1 и т.д. – в раундах с 1-го по 24-й;
- K7, K6, K5, K4, K3, K2, K1, K0 – в раундах с 25-го по 32-й.

Дешифрование в данном режиме проводится точно так же, но с несколько другой последовательностью применения подключей:

- K0, K1, K2, K3, K4, K5, K6, K7 – в раундах с 1-го по 8-й;
- K7, K6, K5, K4, K3, K2, K1, K0, K7, K6 и т.д. – в раундах с 9-го по 32-й.

Все блоки шифруются независимо друг от друга, то есть результат шифрования каждого блока зависит только от его содержимого (соответствующего блока исходного текста). При наличии нескольких одинаковых блоков исходного (открытого) текста соответствующие им блоки шифртекста тоже будут одинаковы, что дает дополнительную полезную информацию для пытающегося вскрыть шифр криптоаналитика. Поэтому данный режим применяется в основном для шифрования самих ключей шифрования (очень часто реализуются многоключевые схемы, в которых по ряду соображений ключи шифруются друг на друге). Для шифрования собственно информации предназначены два других режима работы – гаммирования и гаммирования с обратной связью.

В режиме гаммирования каждый блок открытого текста побитно складывается «по модулю 2» с блоком гаммы шифра размером 64 бит. Гамма шифра - это специальная последовательность, которая получается в результате определенных операций с регистрами L0 и R0:

1) В регистры L0 и R0 записывается их начальное заполнение - 64-битовая величина, называемая синхропосылкой.
2) Выполняется шифрование содержимого регистров L0 и R0 (в данном случае синхропосылки) в режиме простой замены.
3) Содержимое регистра L0 складывается по модулю $(2^{32} - 1)$ с константой $C1 = 2^{24} + 2^{16} + 2^8 + 2^4$, а результат сложения записывается в регистр R1.

4) Содержимое регистра R0 складывается по модулю 2^{32} с константой C2 = 2^{24} +2^{16} + 2^{8} + 1, а результат сложения записывается в регистр L1.
5) Содержимое регистров L1 и R1 подается на выход в качестве 64-битового блока гаммы шифра (в данном случае L1 и R1 образуют первый блок гаммы).
6) Если необходим следующий блок гаммы (то есть необходимо продолжить шифрование или дешифрование), выполняется возврат к операции 2.
7) Для дешифрования гамма вырабатывается аналогичным образом, а затем к битам зашифрованного текста и гаммы снова применяется операция XOR. Поскольку эта операция обратима, в случае правильно выработанной гаммы получается исходный текст (табл. 4.11).

Таблица 4.11. Шифрование и дешифрование в режиме гаммирования

	Операция	Результат
Исходный текст		100100
	XOR	111000
	=	011100
	XOR	111000
Исходный текст	=	100100

Для выработки нужной для расшифровки гаммы шифра у пользователя, расшифровывающего криптограмму, должен быть тот же ключ и то же значение синхропосылки, которые применялись при шифровании информации. В противном случае получить исходный текст из зашифрованного текста не удастся.

В большинстве реализаций алгоритма ГОСТ 28147-89 синхропосылка не секретна, однако есть системы, где синхропосылка - такой же секретный элемент, как и ключ шифрования. Для таких систем эффективная длина ключа алгоритма (256 бит) увеличивается еще на 64 бит секретной синхропосылки, которую также можно рассматривать как ключевой элемент.

В режиме гаммирования с обратной связью для заполнения регистров L0 и R0 начиная со 2-го блока используется не предыдущий блок гаммы, а результат шифрования предыдущего блока открытого текста (рис. 4.9). Первый же блок в данном режиме генерируется полностью аналогично предыдущему.

Рис. 4.9. Выработка гаммы шифра в режиме гаммирования с обратной связью

Рассматривая режим генерации имитовставок, следует определить понятие предмета генерации. Имитовставка – это криптографическая контрольная сумма, вычисляемая с использованием ключа шифрования и предназначенная для проверки целостности сообщений. При генерации имитовставки выполняются следующие операции: первый 64-битовый блок массива информации, для которого вычисляется

имитоприставка, записывается в регистры L0 и R0 и зашифровывается в сокращенном режиме простой замены (выполняются первые 16 раундов из 32).

Полученный результат суммируется «по модулю 2» со следующим блоком информации с сохранением результата в L0 и R0.

Цикл повторяется до последнего блока информации. Получившееся в результате этих преобразований 64-битовое содержимое регистров L16 и R16 или его часть и называется имитоприставкой.

Размер имитоприставки выбирается исходя из требуемой достоверности сообщений: при длине имитоприставки r бит вероятность, что изменение сообщения останется незамеченным, равна 2^{-r}.

Чаще всего используется 32-битовая имитоприставка, то есть половина содержимого регистров. Этого достаточно, поскольку, как любая контрольная сумма, имитоприставка предназначена прежде всего для защиты от случайных искажений информации. Для защиты же от преднамеренной модификации данных применяются другие криптографические методы – в первую очередь электронная цифровая подпись.

При обмене информацией имитоприставка служит своего рода дополнительным средством контроля. Она вычисляется для открытого текста при шифровании какой-либо информации и посылается вместе с шифртекстом. После дешифрования вычисляется новое значение имитоприставки, которое сравнивается с присланной.

Если значения не совпадают – значит, шифртекст был искажен при передаче или при дешифровании использовались неверные ключи. Особенно полезна имитоприставка для проверки правильности дешифрования ключевой информации при использовании многоключевых схем.

Алгоритм ГОСТ 28147-89 считается очень стойким – в настоящее время для его раскрытия не предложено более эффективных методов, чем упомянутый выше метод «грубой силы». Его высокая стойкость достигается в первую очередь за счет большой длины ключа – 256 бит. При использовании секретной синхропосылки эффективная длина ключа увеличивается до 320 бит, а засекречивание таблицы замен прибавляет дополнительные биты. Кроме того, криптостойкость зависит от количества раундов преобразований, которых по ГОСТ 28147-89 должно быть 32 (полный эффект рассеивания входных данных достигается уже после 8 раундов).

4.2.3. СТАНДАРТ ШИФРОВАНИЯ AES

В 1997 году Американский институт стандартизации NIST (National Institute of Standards & Technology) объявил конкурс на новый стандарт симметричного крип-тоалгоритма, названного AES (Advanced Encryption Standard). К его разработке были подключены самые крупные центры криптологии со всего мира.

К криптоалгоритмам - кандидатам на новый стандарт AES были предъявлены следующие требования:

- алгоритм должен быть симметричным;
- алгоритм должен быть блочным шифром;
- алгоритм должен иметь длину блока 128 бит и поддерживать три длины ключа: 128, 192 и 256 бит.
- Дополнительно разработчикам криптоалгоритмов рекомендовалось:

- использовать операции, легко реализуемые как аппаратно (в микрочипах), так и программно (на персональных компьютерах и серверах);
- ориентироваться на 32-разрядные процессоры;
- не усложнять без необходимости структуру шифра, для того чтобы все заинтересованные стороны были в состоянии самостоятельно провести независимый криптоанализ алгоритма и убедиться, что в нем не заложено каких-либо недокументированных возможностей.

На этот конкурс было представлено 15 алгоритмов-претендентов, разработанных как известными в области криптографии организациями (RSA Security, Counterpane и т.д.), так и частными лицами. Итоги конкурса были подведены в октябре 2000 года: победителем был объявлен алгоритм Rijndael, разработанный двумя криптографами из Бельгии, Винсентом Риджменом (Vincent Rijmen) и Джоан Даймен (Joan Daemen).

Алгоритм Rijndael стал новым стандартом шифрования данных AES.

Алгоритм AES не похож на большинство известных алгоритмов симметричного шифрования, структура которых носит название сети Фейстеля. В отличие от них, алгоритм AES представляет каждый блок обрабатываемых данных в виде двухмерного байтового массива размером 4×4, 4×6 или 4×8 в зависимости от установленной длины блока (допускается использование нескольких фиксированных размеров шифруемого блока информации). Далее на соответствующих этапах производятся преобразования либо над независимыми столбцами, либо над независимыми строками, либо вообще над отдельными байтами.

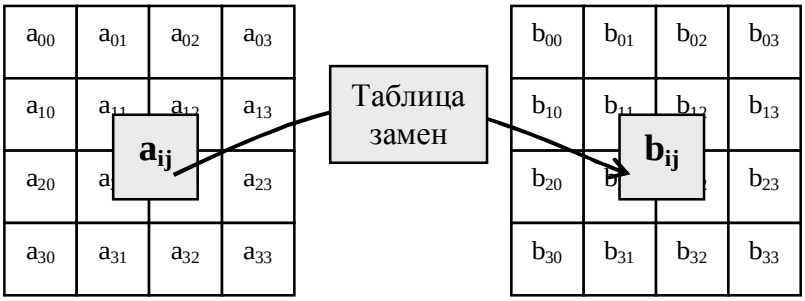


Рис. 4.10. Преобразование BS (ByteSub) использует таблицу замен (подстановок) для обработки каждого байта массива

Алгоритм AES состоит из определенного количества раундов (от 10 до 14 – это зависит от размера блока и длины ключа) и выполняет четыре преобразования:

- BS (ByteSub) - табличная замена каждого байта массива (рис. 4.10);
- SR (ShiftRow) - сдвиг строк массива (рис. 4.11). При этой операции первая строка остается без изменений, а остальные циклически побайтно сдвигаются влево на фиксированное число байтов, зависящее от размера массива. Например, для массива размером 4×4 строки 2, 3 и 4 сдвигаются соответственно на 1, 2 и 3 байта;
- MC (MixColumn) – операция над независимыми столбцами массива (рис. 4.12), когда каждый столбец по определенному правилу умножается на фиксированную матрицу c(x) ;

– AK (AddRoundKey) – добавление ключа. Каждый бит массива складывается «по модулю 2» с соответствующим битом ключа раунда, который, в свою очередь, определенным образом вычисляется из ключа шифрования (рис. 4.13).

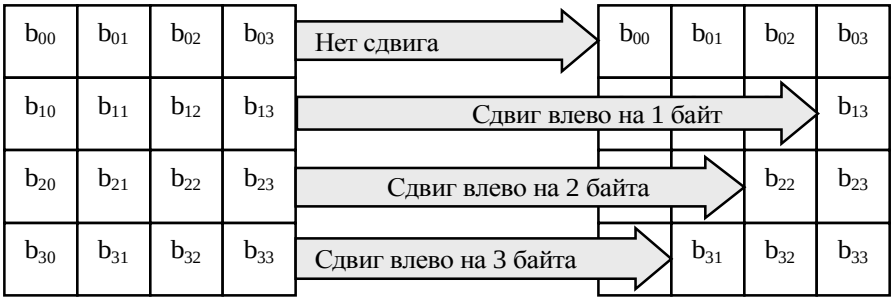


Рис. 4.11. Преобразование SR (ShiftRow) циклически сдвигает три последние строки в массиве

Эти преобразования воздействуют на массив State, который адресуется с помощью указателя 'state'. Преобразование AddRoundKey использует дополнительный указатель для адресации ключа раунда Round Key.

Преобразование BS (ByteSub) является нелинейной байтовой подстановкой, которая воздействует независимо на каждый байт массива State, используя таблицу замен (подстановок) S-box.

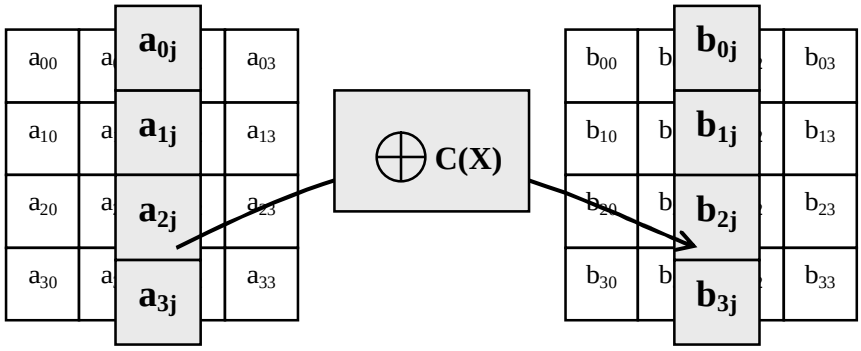


Рис. 4.12. Преобразование MC (MixColumn) поочередно обрабатывает столбцы массива

В каждом раунде (с некоторыми исключениями) над шифруемыми данными поочередно выполняются перечисленные преобразования (рис. 4.14). Исключения касаются первого и последнего раундов: перед первым раундом дополнительно выполняется операция AK, а в последнем раунде отсутствует MC.

В результате последовательность операций при шифровании выглядит так:

$AK, \{BS, SR, MC, AK\}$ (повторяется $R-1$ раз), BS, SR, AK.

Количество раундов шифрования R в алгоритме AES переменное (10, 12 или 14 раундов) и зависит от размеров блока и ключа шифрования (для ключа также предусмотрено несколько фиксированных размеров).

Дешифрование выполняется с помощью следующих обратных операций:

1) Табличная замена BS обращается применением другой таблицы, являющейся инверсной относительно таблицы, применяемой при шифровании.
2) Обратная операция к SR – это циклический сдвиг строк вправо, а не влево.
3) Обратная операция для MC – умножение по тем же правилам на другую матрицу d(x), удовлетворяющую условию $c(x)\times d(x)=1$.
4) Добавление ключа AK является обратным самому себе, поскольку в нем используется только операция XOR.

Эти обратные операции применяются при дешифровании в последовательности, обратной той, что использовалась при шифровании.

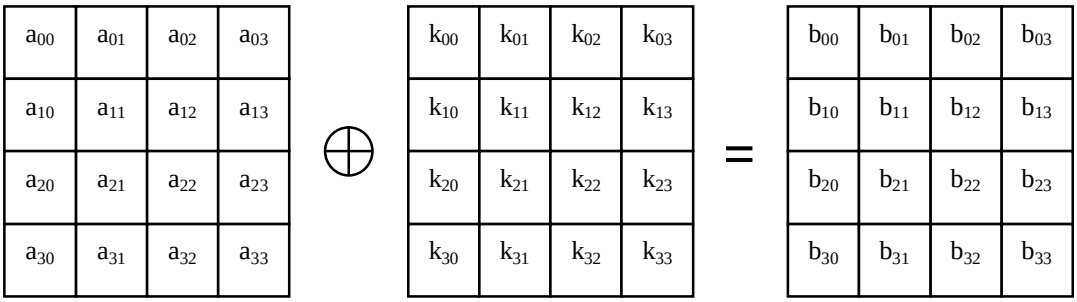


Рис. 4.13. Преобразование AK (AddRoundKey) производит сложение

Все преобразования в шифре AES имеют строгое математическое обоснование.

Сама структура и последовательность операций позволяют выполнять данный алгоритм эффективно как на 8-битовых так и на 32-битовых процессорах. В структуре алгоритма заложена возможность параллельного исполнения некоторых операций, что может поднять скорость шифрования на многопроцессорных рабочих станциях в несколько раз.

Алгоритм Rijndael стал новым стандартом шифрования данных AES благодаря целому ряду преимуществ перед другими алгоритмами. Прежде всего, он обеспечивает высокую скорость шифрования на всех платформах: как при программной, так и при аппаратной реализации. Кроме того, требования к ресурсам для его работы минимальны, что важно при его использовании в устройствах, обладающих ограниченными вычислительными возможностями.

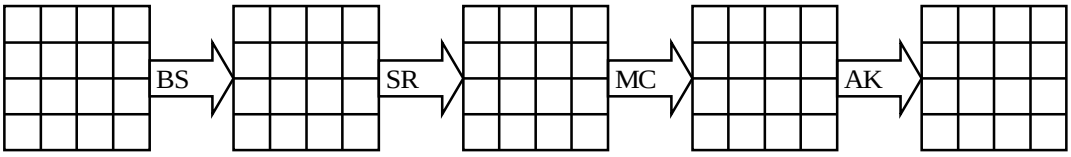


Рис. 4.14. Раунд алгоритма AES

Недостатком же алгоритма AES можно считать лишь свойственную ему нетрадиционную схему. Дело в том, что свойства алгоритмов, основанных на сети Фейстеля, хорошо исследованы, а AES, в отличие от них, может содержать скрытые уязвимости, которые могут обнаружиться только по прошествии какого-то времени с момента начала его широкого распространения.

Другие симметричные криптоалгоритмы

Для шифрования данных применяются и другие блочные симметричные криптоалгоритмы.

Алгоритм IDEA (International Data Encryption Algorithm) – еще один 64-битовый блочный шифр с длиной ключа 128 бит. Этот европейский стандарт криптоалгоритма предложен в 1990 году. Алгоритм IDEA по скорости не уступает алгоритму DES, а по стойкости к криптоанализу превосходит DES.

Алгоритм RC2 представляет собой 64-битовый блочный шифр с ключом переменной длины. Этот алгоритм приблизительно в 2 раза быстрее, чем DES. Может
использоваться в тех же режимах, что и DES, включая тройное шифрование. Владельцем алгоритма является компания RSA Data Security. Алгоритм RC5 представляет собой быстрый блочный шифр, который имеет размер блока 32, 64 или 128 бит, ключ длиной от 0 до 2048 бит. Алгоритм выполняет от 0 до 255 проходов. Алгоритмом владеет компания RSA Data Security.

Алгоритм Blowfish – это 64-битовый блочный шифр, имеет ключ переменного размера до 448 бит, выполняет 16 проходов, на каждом проходе осуществляются
перестановки, зависящие от ключа, и подстановки, зависящие от ключа и данных. Этот алгоритм быстрее алгоритма DES.

4.2.4. ОСНОВНЫЕ РЕЖИМЫ РАБОТЫ БЛОЧНОГО СИММЕТРИЧНОГО АЛГОРИТМА

Рассмотрим основные режимы работы блочного симметричного алгоритма. Большинство блочных симметричных криптоалгоритмов непосредственно преобразуют 64-битовый входной открытый текст в 64-битовый выходной шифрованный текст, однако данные редко ограничиваются 64 разрядами.

Чтобы воспользоваться блочным симметричным алгоритмом для решения разнообразных криптографических задач, разработано четыре рабочих режима:

- электронная кодовая книга ECB (Electronic Code Book);
- сцепление блоков шифра CBC (Cipher Block Chaining);
- обратная связь по шифртексту CFB (Cipher Feed Back);
- обратная связь по выходу OFB (Output Feed Back).

Эти рабочие режимы первоначально были разработаны для блочного алгоритма DES, но в любом из них могут работать и другие блочные криптоалгоритмы. В качестве примера будем использовать блочный алгоритм DES.

Режим «Электронная кодовая книга» Длинный файл разбивают на 64-битовые отрезки (блоки) по 8 байт. Каждый из этих блоков шифруют независимо с использованием одного и того же ключа шифрования (рис. 4.15).

Основное достоинство – простота реализации. Недостатка два. Первый недостаток – относительно слабая устойчивость против криптоаналитических атак. Из-за фиксированного характера шифрования при ограниченной длине блока 64 бит возможно проведение криптоанализа «со словарем». Блок такого размера может повториться в сообщении вследствие большой избыточности в тексте на естественном языке. Это приводит к тому, что идентичные блоки открытого текста в сообщении будут представлены идентичными блоками шифротекста, что дает криптоаналитику некоторую информацию о содержании сообщения. Второй недостаток связан с общим требованием шифрования – равенства длины открытого

текста и соответствующего ему шифротекста, что трудно выдержать в случае если длина открытого текста не кратна длине блока шифрования.

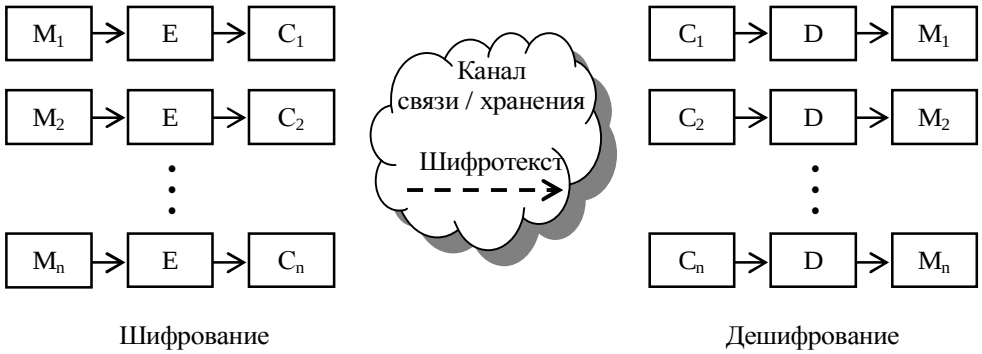


Рис. 4.15. Схема работы блочного алгоритма в режиме «Электронная кодовая книга» (ECB)

Режим «Сцепление блоков шифра» В этом режиме исходный файл M разбивается на 64-битовые блоки: M_0, M_1, … , M_n. Первый блок M складывается «по модулю 2» с 64-битовым начальным вектором IV (Input Vector), который меняется ежедневно или при каждом сеансе связи и держится в секрете (рис. 4.16). Полученная сумма затем шифруется с использованием ключа шифра, известного и отправителю, и получателю информации. Полученный 64-битовый блок шифротекста C складывается «по модулю 2» со вторым блоком текста, результат шифруется и получается второй 64-битовый блок шифртекста C и т.д. Процедура повторяется до тех пор, пока не будут обработаны все блоки текста.

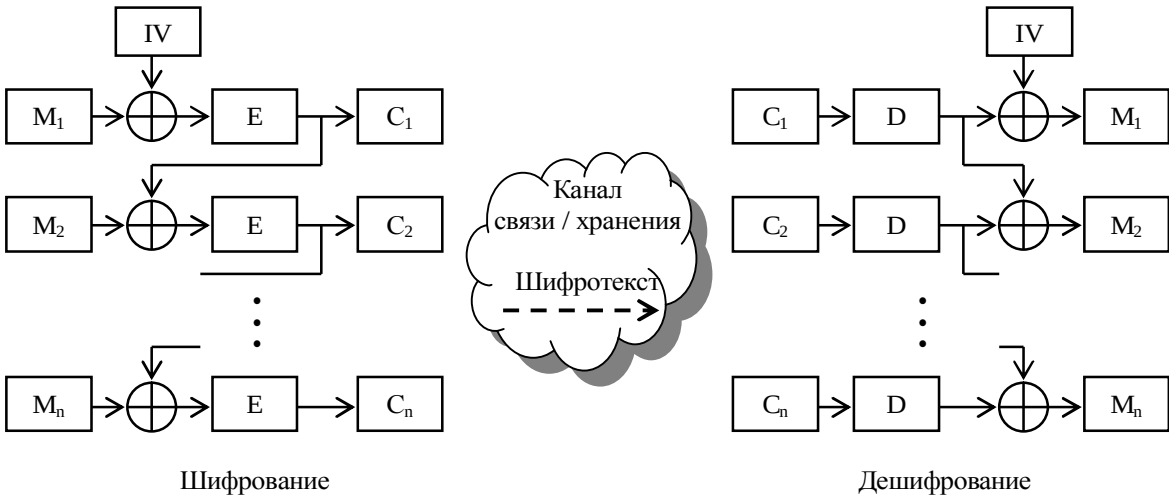


Рис. 4.16. Схема работы блочного алгоритма в режиме «Сцепление блоков шифра» (CBC)

Таким образом, для всех i = 1…n (n – число блоков) результат шифрования C_i определяется следующим образом:

$$C_i = E(M_i \oplus C_{i-1}),$$

где C_0=IV – начальное значение шифра, равное начальному вектору (вектору инициализации).

Очевидно, что последний 64-битовый блок шифртекста является функцией секретного ключа, начального вектора и каждого бита открытого текста независимо от его длины. Этот блок шифртекста называют кодом аутентификации сообщения MAC (Message Authentication Code).

Код MAC может быть легко проверен получателем, владеющим секретным ключом и начальным вектором, путем повторения процедуры, выполненной отправителем. Посторонний, не владеющий секретным ключом и начальным вектором, не может осуществить генерацию MAC, который воспринялся бы получателем как подлинный, чтобы добавить его к ложному сообщению, либо отделить MAC от истинного сообщения для использования его с измененным или ложным сообщением. Достоинство данного режима в том, что он не позволяет накапливаться ошибкам при передаче. Блок M_i является функцией только C_{i-1} и C_i. Поэтому ошибка при передаче приведет к потере только двух блоков исходного текста.

Режим «Обратная связь по шифртексту» В этом режиме размер блока может отличаться от 64 бит (рис. 4.17). Файл, подлежащий шифрованию (дешифрованию), считывается последовательными блоками длиной k битов (k = 1…64).

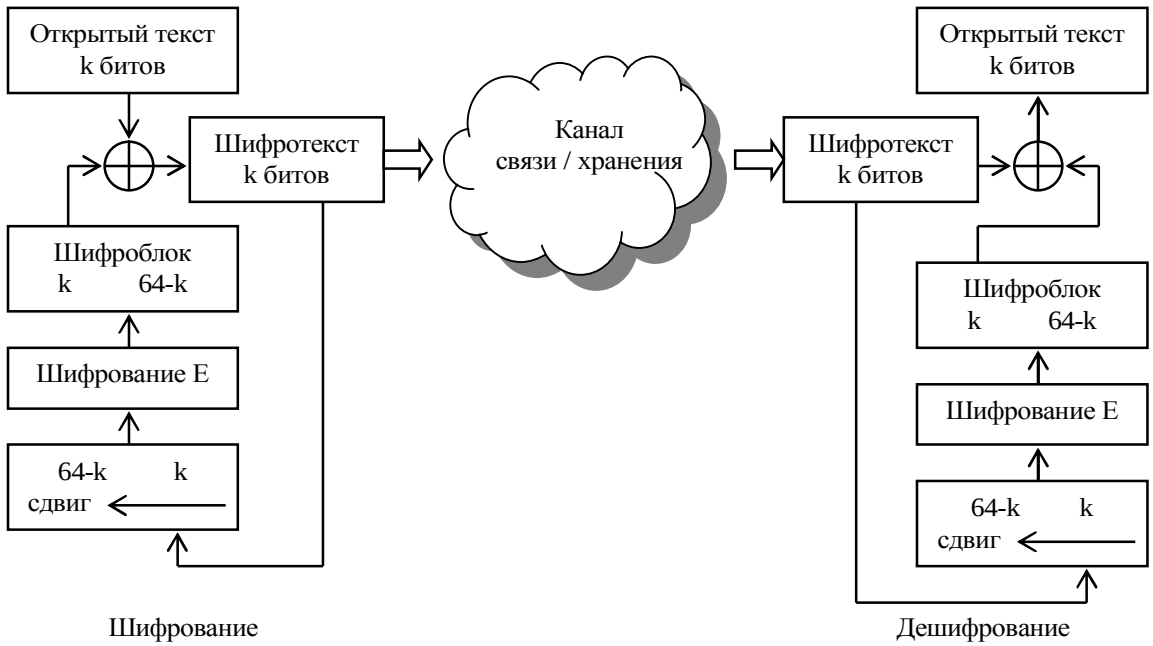


Рис. 4.17. Схема работы блочного алглритма в режиме «Обратная связи по шифротексту» (CFB)

Входной блок (64-битовый регистр сдвига) вначале содержит вектор инициализации IV, выровненный по правому краю.

Предположим, что в результате разбиения на блоки мы получили n блоков длиной k битов каждый. Тогда для любого i = 1…n блок шифртекста:

$$C_i = M_i \oplus P_{i-1},$$

где P_{i-1} означает k старших битов предыдущего зашифрованного блока.

Обновление сдвигового регистра осуществляется путем удаления его старших k битов и записи C_i в регистр. Восстановление зашифрованных данных выполняют относительно просто: P_{i-1} и C_i вычисляются аналогичным образом и

$M_i = C_i \oplus P_{i-1}$.

Режим «Обратная связь по выходу» Этот режим тоже использует переменный размер блока и сдвиговый регистр, инициализируемый так же, как в режиме CFB, а именно – входной блок вначале содержит вектор инициализации IV, выровненный по правому краю (рис. 4.18).

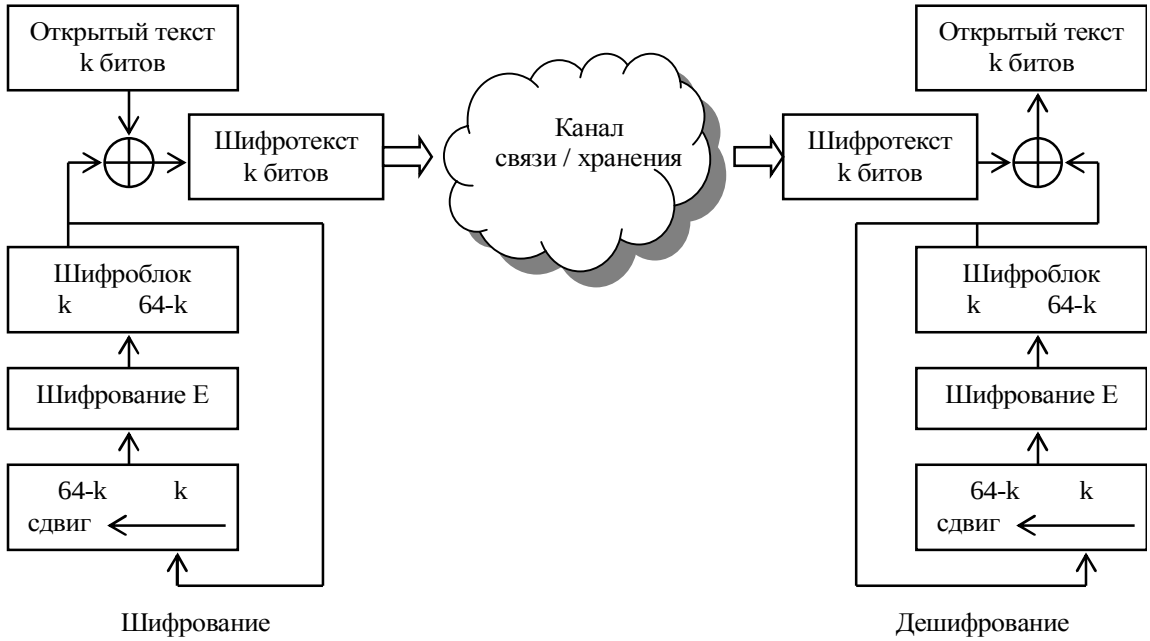


Рис. 4.18. Схема работы блочного алгоритма в режиме обратной связи по выходу (OFB)

Положим, $M = M_1 M_2 \ldots M_n$.

Для всех i = 1…n

$C_i = M_i \oplus P_i$,

где P_i – старшие k битов операции $E(C_{i-1})$.

Каждому из рассмотренных режимов (ECB, CBC, CFB, OFB) свойственны свои достоинства и недостатки, что обусловливает области их применения.

Режим ECB хорошо подходит для шифрования ключей; режим CFB, как правило, предназначается для шифрования отдельных символов, а режим OFB нередко применяется для шифрования в спутниковых системах связи.

Режимы CBC и CFB пригодны для аутентификации данных. Эти режимы позволяют также использовать блочные симметричные криптоалгоритмы для:

- интерактивного шифрования при обмене данными между терминалом и главной ЭВМ;
- шифрования криптографического ключа в практике автоматизированного распространения ключей;
- шифрования файлов, почтовых отправлений, данных спутников и других практических задач.

4.2.5. ОСОБЕННОСТИ ПРИМЕНЕНИЯ АЛГОРИТМОВ СИММЕТРИЧНОГО ШИФРОВАНИЯ

Алгоритмы симметричного шифрования используют ключи относительно небольшой длины и могут быстро шифровать большие объемы данных. При симметричной методологии шифрования отправитель и получатель применяют для осуществления процессов шифрования и дешифрования сообщения один и тот же секретный ключ.

Алгоритмы симметричного шифрования строятся исходя из предположения, что зашифрованные данные не сможет прочитать никто из тех, кто не обладает ключом для их дешифрования. Если ключ не был скомпрометирован, то при дешифровании автоматически выполняется аутентификация отправителя, так как только отправитель имеет ключ, с помощью которого можно зашифровать информацию, и только получатель имеет ключ, позволяющий расшифровать информацию.

Алгоритмы симметричного шифрования применяются для абонентского шифрования данных – то есть для шифрования информации, предназначенной для отправки кому-либо, например, через Интернет. Использование только одного секретного ключа для всех абонентов сети, конечно, недопустимо по соображениям безопасности: в случае компрометации (утери, хищения) ключа под угрозой будет находиться документооборот всех абонентов сети.

Порядок использования систем с симметричными ключами:

1) Симметричный секретный ключ должен создаваться, распространяться и сохраняться безопасным образом.
2) Для получения зашифрованного текста отправитель применяет к исходному сообщению симметричный алгоритм шифрования вместе с секретным симметричным ключом. Таким образом неявно подготавливается аутентификация отправителя и получателя, так как только отправитель знает симметричный секретный ключ и может зашифровать этот текст. Только получатель знает симметричный секретный ключ и может расшифровать этот текст.
3) Отправитель передает зашифрованный текст. Симметричный секретный ключ никогда не передается в открытой форме по незащищенным каналам связи.
4) Получатель применяет к зашифрованному тексту тот же самый симметричный алгоритм шифрования / дешифрования вместе с тем же самым симметричным ключом (который уже есть у получателя) для восстановления исходного текста. Его успешное восстановление аутентифицирует того, кто знает секретный ключ.

Для симметричных криптосистем актуальна проблема безопасного распределения симметричных секретных ключей. Всем системам симметричного шифрования присущи следующие недостатки:

- принципиальным является требование защищенности и надежности канала передачи секретного ключа для каждой пары участников информационного обмена;
- предъявляются повышенные требования к службе генерации и распределения ключей, обусловленные тем, что для n абонентов при схеме взаимодействия «каждый с каждым» требуется $n\times(n-1)/2$ ключей, то есть

зависимость числа ключей от числа абонентов является квадратичной; например, для n = 1000 абонентов требуемое количество ключей будет равно $n\times(n-1)/2$ = 499 500 ключей.

Поэтому без эффективной организации защищенного распределения ключей широкое использование обычной системы симметричного шифрования в больших сетях и, в частности, в глобальных сетях практически невозможно.

4.3. АСИММЕТРИЧНЫЕ КРИПТОСИСТЕМЫ ШИФРОВАНИЯ

Асимметричные криптографические системы были разработаны в 1970-х годах. Принципиальное отличие асимметричной криптосистемы от криптосистемы симметричного шифрования состоит в том, что для шифрования информации и ее последующего дешифрования используются различные ключи:

- открытый ключ K: используется для шифрования информации, вычисляется из секретного ключа k;
- секретный ключ k: используется для дешифрования информации, зашифрованной с помощью парного ему открытого ключа K.

Эти ключи различаются таким образом, что с помощью вычислений нельзя вывести секретный ключ k из открытого ключа K. Поэтому открытый ключ K может свободно передаваться по каналам связи.

Асимметричные системы называют еще двухключевыми криптографическими системами или криптосистемами с открытым ключом.

Обобщенная схема асимметричной криптосистемы шифрования с открытым ключом показана на рис. 4.19.

Для криптографического закрытия и последующего дешифрования передаваемой информации используются открытый и секретный ключи получателя сообщения *B*. В качестве ключа шифрования должен использоваться открытый ключ получателя K_B, а в качестве ключа дешифрования – его секретный ключ k_B.

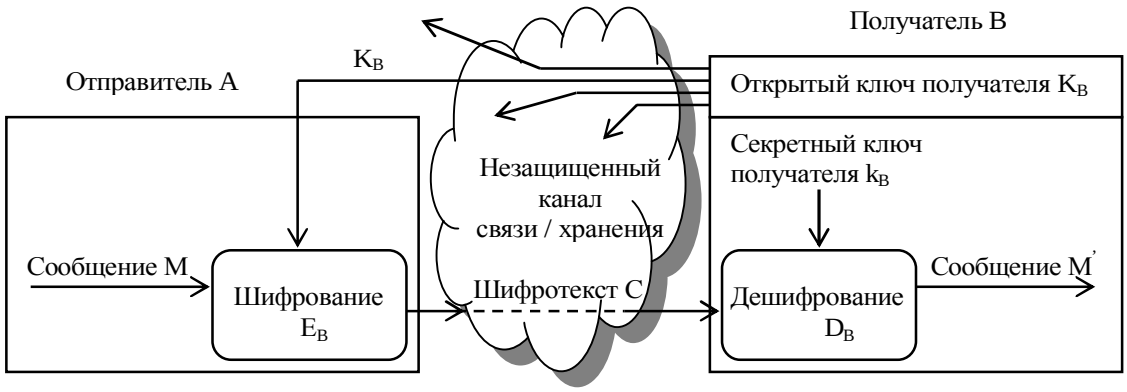


Рис. 4.19. Обобщенная схема асимметричной криптосистемы шифрования

Секретный и открытый ключи генерируются попарно. Секретный ключ должен оставаться у его владельца; он должен быть надежно защищен от несанкционированного доступа (аналогично ключу шифрования в симметричных алгоритмах). Копия открытого ключа должна иметься у каждого абонента криптографической сети, с которым обменивается информацией владелец секретного ключа.

Процесс передачи зашифрованной информации в асимметричной криптосистеме осуществляется следующим образом:

1) Подготовительный этап. Абонент В генерирует пару ключей: секретный ключ k_B и открытый ключ $К_в$. Открытый ключ $К_B$ посылается абоненту А и остальным абонентам (или делается доступным, например, на разделяемом ресурсе).
2) Использование - обмен информацией между абонентами А и В. Абонент А зашифровывает сообщение с помощью открытого ключа $К_B$ абонента В и отправляет шифротекст абоненту В. Абонент В расшифровывает сообщение с помощью своего секретного ключа k_B. Никто другой (в том числе абонент А) не может расшифровать данное сообщение, так как не имеет секретного ключа абонента В. Защита информации в асимметричной криптосистеме основана на секретности ключа k_B получателя сообщения.

Отметим характерные особенности асимметричных криптосистем:

1) Открытый ключ $К_B$ и криптограмма С могут быть отправлены по незащищенным каналам, то есть противнику известны $К_B$ и С.
2) Алгоритмы шифрования и дешифрования

$$E_B : M \to C,$$

$$D_B : C \to M$$

являются открытыми.

У. Диффи и М. Хеллман сформулировали требования, выполнение которых обеспечивает безопасность асимметричной криптосистемы [xxx]:

1) Вычисление пары ключей ($К_B$, k_B) получателем В на основе начального условия должно быть простым.
2) Отправитель А, зная открытый ключ $К_B$ и сообщение М, может легко вычислить криптограмму

$$C = E_{K_B}(M).$$

3) Получатель В, используя секретный ключ k_B и криптограмму С, может легко восстановить исходное сообщение

$$M = D_{k_B}(C).$$

4) Противник, зная открытый ключ $К_B$, при попытке вычислить секретный ключ k_B наталкивается на непреодолимую вычислительную проблему.
5) Противник, зная пару ($К_B$,С), при попытке вычислить исходное сообщение М наталкивается на непреодолимую вычислительную проблему.

Концепция асимметричных криптографических систем с открытым ключом основана на применении однонаправленных функций. Неформально однонаправленную функцию можно определить следующим образом [xxx]. Пусть X и Y - некоторые произвольные множества. Функция $f: X \to Y$ является однонаправленной, если для всех $x \in X$ можно легко вычислить функцию

$y = f(x)$, где $y \in Y$.

И в то же время для большинства $y \in Y$ достаточно сложно получить значение $x \in X$, такое, что $f(x) = y$ (при этом полагают, что существует по крайней мере одно такое значение x).

Основным критерием отнесения функции f к классу однонаправленных функций является отсутствие эффективных алгоритмов обратного преобразования $Y \to X$.

В качестве примера однонаправленной функции можно указать целочисленное умножение. Прямая задача – вычисление произведения двух очень больших целых чисел P и Q, то есть нахождение значения

$$N = P \times Q$$

является относительно несложной задачей для компьютера.

Обратная задача - факторизация, или разложение на множители большого целого числа, то есть нахождение делителей P и Q большого целого числа $N = P \times Q$ является практически неразрешимой задачей при достаточно больших значениях N. По современным оценкам теории чисел, при целом $N = 2^{664}$ и $P \approx Q$ для разложения числа N потребуется около 10^{23} операций, то есть задача практически неразрешима для современных компьютеров.

Другой характерный пример однонаправленной функции является модульная экспонента с фиксированными основанием и модулем. Пусть A и N – целые числа, такие, что $1 \le A < N$. Определим множество Z_N:

$$Z_N = \{0, 1, 2, \ldots N-1\}.$$

Тогда модульная экспонента с основанием A по модулю N представляет собой функцию

$$f_{A,N} : Z_N \to Z_N,$$

$$f_{A,N}(x) = A^x (\mathrm{mod}\, N),$$

где X – целое число, $1 \le x \le N-1$.

Существуют эффективные алгоритмы, позволяющие достаточно быстро вычислить значения функции $f_{A,N}(x)$.

Если $y = A^x$, то естественно записать $x = \log_A(y)$.

Поэтому задачу обращения функции $f_{A,N}(x)$ называют задачей нахождения дискретного логарифма, или задачей дискретного логарифмирования. Задача дискретного логарифмирования формулируется следующим образом. Для известных целых A, N, y найти целое число x, такое, что

$$A^x \bmod N = y.$$

Пример

Найти значение x, при котором $3^x \bmod 17 = 13$ (остаток от деления на 17).

Найдем значение $y = 3^x \bmod 17$ методом перебора: 3^1=3, 3^2=9, 3^3=10, **3^4=13**, 3^5=5, 3^6=15, 3^7=11, 3^8=16, 3^9=14, 3^{10}=8, 3^{11}=7, 3^{12}=4, 3^{13}=12, 3^{14}=2, 3^{15}=6, 3^{16}=1, …
Таким образом, $x = 4$.

Алгоритм вычисления дискретного логарифма за приемлемое время пока не найден. Поэтому модульная экспонента считается однонаправленной функцией. По современным оценкам теории чисел, при целых числах A =2^{664} и N=2^{664}, решение задачи дискретного логарифмирования (нахождение показателя степени x для известного y) потребует около 10^{26} операций, то есть эта задача имеет в 10^3 раз большую вычислительную сложность, чем задача разложения на множители. При увеличении длины чисел разница в оценках сложности задач возрастает.

Следует отметить, что пока не удалось доказать невозможность существования эффективного алгоритма вычисления дискретного логарифма за приемлемое время. Исходя из этого, модульная экспонента отнесена к однонаправленным функциям условно, что, однако, не мешает с успехом применять ее на практике. Вторым важным классом функций, используемых при построении криптосистем с открытым ключом, являются так называемые однонаправленные функции с секретом. Дадим неформальное определение такой функции. Функция

$$f : X \to Y$$

относится к классу однонаправленных функций с секретом в том случае, если она является однонаправленной и, кроме того, возможно эффективное вычисление обратной функции, если известен секрет (секретное число, строка или другая информация, ассоциируемая с данной функцией).

В качестве примера однонаправленной функции с секретом можно указать используемую в криптосистеме RSA модульную экспоненту с фиксированными модулем и показателем степени. Переменное основание модульной экспоненты используется для представления числового значения сообщения M либо криптограммы C.

Как и в случае симметричных криптографических систем, с помощью асимметричных криптосистем обеспечивается не только конфиденциальность, но также подлинность и целостность передаваемой информации. Подлинность и целостность любого сообщения обеспечивается формированием цифровой подписи этого сообщения и отправкой в зашифрованном виде сообщения вместе с цифровой подписью.

Проверка соответствия подписи полученному сообщению после его предварительного расшифровывания представляет собой проверку целостности и подлинности принятого сообщения.

Асимметричные криптографические системы обладают следующими важными преимуществами перед симметричными криптосистемами:

- в асимметричных криптосистемах решена сложная проблема распределения ключей между пользователями, так как каждый пользователь может сгенерировать свою пару ключей сам, а открытые ключи пользователей могут свободно публиковаться и распространяться по сетевым коммуникациям;
- исчезает квадратичная зависимость числа ключей от числа пользователей; в асимметричной криптосистеме количество используемых ключей связано с количеством абонентов линейной зависимостью (в системе из N пользователей используется $2 \times N$ ключей), а не квадратичной, как в симметричных системах;
- асимметричные криптосистемы позволяют реализовать протоколы взаимодействия сторон, которые не доверяют друг другу, поскольку при использовании асимметричных криптосистем закрытый ключ должен быть известен только его владельцу.
- Однако у асимметричных криптосистем существуют и недостатки:
- на настоящий момент нет математического доказательства необратимости используемых в асимметричных алгоритмах функций;
- по сравнению с симметричным шифрованием, асимметричное существенно медленнее, поскольку при шифровании и расшифровке используются весьма

ресурсоемкие операции. По этой же причине реализовать аппаратный шифратор с асимметричным алгоритмом существенно сложнее, чем аппаратно симметричный алгоритм;

– необходимо защищать открытые ключи от подмены.

Последнее рассмотрим более подробно. Предположим, на компьютере абонента А хранится открытый ключ K_B абонента В. Злоумышленник n имеет доступ к открытым ключам, хранящимся у абонента А. Он генерирует свою пару ключей K_n и k_n и подменяет у абонента А открытый ключ K_B абонента В на свой открытый ключ K_n.

Для того чтобы отправить некую информацию абоненту В, абонент А зашифровывает ее на ключе K_n, думая, что это ключ K_B. Соответственно, это сообщение не сможет прочитать абонент В, но зато легко расшифрует и прочитает злоумышленник n. От подмены открытых ключей может спасти процедура сертификации открытых ключей.

4.3.1. АЛГОРИТМ ШИФРОВАНИЯ RSA

Криптоалгоритм RSA предложили в 1978 году три автора: Р. Райвест (Rivest), А. Шамир (Shamir) и А. Адлеман (Adleman). Алгоритм получил свое название по первым буквам фамилий его авторов. Алгоритм RSA стал первым алгоритмом с открытым ключом, который может работать в режиме как шифрования данных, так и электронной цифровой подписи.

Надежность алгоритма RSA основывается на трудности факторизации больших чисел и сложности вычисления дискретных логарифмов в конечном поле.

В алгоритме RSA открытый ключ K_B, секретный ключ k_B, сообщение M и криптограмма C принадлежат множеству целых чисел

$Z_N = \{0, 1, 2, ..., N - 1\}$,

где N – модуль:

$N = P \times Q$.

Здесь P и Q – случайные большие простые числа. Для обеспечения максимальной безопасности выбирают P и Q равной длины и хранят в секрете.

Множество Z_N с операциями сложения и умножения по модулю N образует арифметику по модулю N.

Открытый ключ K_B выбирают случайным образом так, чтобы выполнялись следующие условия:

$1 < K_B \le j(N)$, НОД $(K_B, j(N)) = 1$,

$j(N) = (P-1)(Q-1)$,

где $j(N)$ - функция Эйлера.

Функция Эйлера $j(N)$ указывает количество положительных целых чисел в интервале от 1 до N, которые взаимно просты с N.

Второе из указанных выше условий означает, что открытый ключ K_B и функция Эйлера $j(N)$ должны быть взаимно простыми.

Далее, используя расширенный алгоритм Евклида, вычисляют секретный ключ k_B, такой, что

$k_B \times K_B = 1 (\mathrm{mod}\ j(N))$

или

$k_B = K_B^{-1} (\mathrm{mod}\ (P - 1)(Q - 1))$.

Это можно осуществить, так как получатель B знает пару простых чисел (P, Q) и может легко найти $j(N)$. Заметим, что k_B и N должны быть взаимно простыми.

Открытый ключ K_B используют для шифрования данных, а секретный ключ k_B – для дешифрования.

Процедура шифрования определяет криптограмму C через пару (открытый ключ K_B, сообщение M) в соответствии со следующей формулой:

$$C = E_{K_B}(M) = M^{K_B} (\mathrm{mod}\, N).$$

В качестве алгоритма быстрого вычисления значения C используют ряд последовательных возведений в квадрат целого M и умножений на M с приведением по модулю N.

Дешифрование криптограммы C выполняют, используя пару (секретный ключ k_B, криптограмма C) по следующей формуле:

$$M = D_{k_B}(C) = C^{k_B} (\mathrm{mod}\, N).$$

Процедуры шифрования и дешифрования в алгоритме RSA

Предположим, что пользователь A хочет передать пользователю B сообщение в зашифрованном виде, используя алгоритм RSA. В таком случае пользователь A выступает в роли отправителя сообщения, а пользователь B – в роли получателя. Как отмечалось выше, криптосистему RSA должен сформировать получатель сообщения, то есть пользователь B. Рассмотрим последовательность действий пользователей B и A:

1) Пользователь B выбирает два произвольных больших простых числа P и Q.
2) Пользователь B вычисляет значение модуля $N = P \times Q$.
3) Пользователь B вычисляет функцию Эйлера
   $j(N) = (P-1)(Q-1)$
   и выбирает случайным образом значение открытого ключа K_B с учетом выполнения условий
   $1 < K_B \le j(N), НОД(K_B, j(N)) = 1.$
4) Пользователь B вычисляет значение секретного ключа k_B, используя расширенный алгоритм Евклида при решении сравнения
   $k_B \equiv K_B^{-1} (\mathrm{mod}\, j(N)).$
5) Пользователь B пересылает пользователю A пару чисел (N, K_B) по незащищенному каналу.

Если пользователь A хочет передать пользователю B сообщение M, он выполняет следующие шаги:

6) Пользователь A разбивает исходный открытый текст M на блоки, каждый из которых может быть представлен в виде числа
   $M_i = 0, 1, 2, ..., N-1.$
7) Пользователь A шифрует текст, представленный в виде последовательности чисел M_i, по формуле
   $C_i = M_i^{K_B} (\mathrm{mod}\, N)$
   и отправляет криптограмму
   $C_1, C_2, C_3, ..., C_i, ...$
   пользователю B.
8) Пользователь B расшифровывает принятую криптограмму
   $C_1, C_2, C_3, ..., C_i, ...,$

используя секретный ключ k_B, по формуле

$$M = D_{k_B}(C) = C^{k_B} (\mathrm{mod}\, N).$$

В результате будет получена последовательность чисел M_i, которые представляют собой исходное сообщение *M*. При практической реализации алгоритма RSA необходимо иметь возможность без существенных затрат генерировать большие простые числа, уметь оперативно вычислять значения ключей K_B и k_B.

Пример:

Шифрование сообщения «CAB» и его дешифрование.

Для простоты вычислений будут использоваться небольшие числа. На практике применяются очень большие числа (длиной 250-300 десятичных разрядов).

Действия пользователя B:

1) Выбирает P = 3 и Q = 11.
2) Вычисляет модуль $N = P \times Q = 3 \times 11 = 33$.
3) Вычисляет значение функции Эйлера для N = 33:

   $j(N) = j(33) = (P-1)(Q-1) = 2 \times 10 = 20.$

   Выбирает в качестве открытого ключа K_B произвольное число с учетом выполнения условий

   $1 < K_B < 20, НОД(K_B, 20) = 1.$

   Пусть $K_B = 7$.
4) Вычисляет значение секретного ключа k_B, используя расширенный алгоритм Евклида при решении сравнения

   $k_B = 7^{-1} (\mathrm{mod}\, 20).$

   Решение дает $k_B = 3$.
5) Пересылает пользователю A пару чисел (N = 33, $K_B = 7$).

Действия пользователя A:

6) Представляет шифруемое сообщение как последовательность целых чисел в диапазоне 0-32. Пусть буква A представляется как число 1, буква B - как число 2, буква C - как число 3. Тогда сообщение CAB можно представить как последовательность чисел 312, то есть $M_1 = 3$, $M_2 = 1$, $M_3 = 2$.
7) Шифрует текст, представленный в виде последовательности чисел M1, M2 и M3, используя ключ $K_B = 7$ и N = 33, по формуле

   $C_i = M_i^{K_B} (\mathrm{mod}\, N) = M_i^7 (\mathrm{mod}\, 33).$

   Получает

   $C_1 = 3^7 (\mathrm{mod}\ 33) = 2187 (\mathrm{mod}\ 33) = 9,$
   $C_2 = 1^7 (\mathrm{mod}\ 33) = 1 (\mathrm{mod}\ 33) = 1,$
   $C_3 = 2^7 (\mathrm{mod}\ 33) = 128 (\mathrm{mod}\ 33) = 29.$

   Отправляет пользователю B криптограмму

   $C_1, C_2, C_3 = 9, 1, 29.$

Действия пользователя B:

8) Расшифровывает принятую криптограмму C_1, C_2, C_3, используя секретный ключ $k_B = 3$, по формуле

   $M_i = C_i^{k_B} (\mathrm{mod}\, N) = C_i^3 (\mathrm{mod}\, 33).$

   Получает

   $M_1 = 9^3 (\mathrm{mod}\ 33) = 729 (\mathrm{mod}\ 33) = 3,$

$M_2 = 1^3 (\text{mod } 33) = 1 (\text{mod } 33) = 1,$

$M_3 = 29^3 (\text{mod } 33) = 24389 (\text{mod } 33) = 2.$

Таким образом, восстановлено исходное сообщение CAB: 312.

Криптоалгоритм RSA всесторонне исследован и признан стойким при достаточной длине ключей. В настоящее время длина ключа 1024 бит считается приемлемым вариантом. Существует мнение, что с ростом мощности процессоров криптоалгоритм RSA потеряет стойкость к атаке полного перебора. Однако увеличение мощности процессоров позволит применять более длинные ключи, что повышает стойкость RSA. Следует отметить, что алгоритм RSA можно применять как для шифрования сообщений, так и для электронной цифровой подписи.

Нетрудно видеть, что в асимметричной криптосистеме RSA количество используемых ключей связано с количеством абонентов линейной зависимостью (в системе из N пользователей используется 2 × N ключей), а не квадратичной, как в симметричных системах.

Сравнивая наиболее популярных представителей асимметричного и симметричного шифрования, следует отметить, что быстродействие RSA существенно ниже быстродействия DES, а программная и аппаратная реализация криптоалгоритма RSA гораздо сложнее, чем DES. Поэтому криптосистема RSA, как правило, используется при передаче сообщений небольшого объема.

4.3.2. АСИММЕТРИЧНЫЕ КРИПТОСИСТЕМЫ НА БАЗЕ ЭЛЕПТИЧЕСКИХ КРИВЫХ

К криптосистемам третьего тысячелетия, несомненно, следует отнести асимметричные криптосистемы на базе эллиптических кривых. Криптосистемы на базе эллиптических кривых позволяют реализовать криптоалгоритм асимметричного шифрования, протокол выработки разделяемого секретного ключа для симметричного шифрования и криптоалгоритмы электронной цифровой подписи [xxx].

Криптосистемы на базе эллиптических кривых имеют более высокую производительность и позволяют использовать существенно меньшие размеры ключей при сохранении требуемого уровня безопасности.

Для различных реализаций используются эллиптические кривые двух видов:

- эллиптическая кривая в конечном поле F_p, где p – простое число, $p > 3$;
- эллиптическая кривая в конечном поле F_2^m.

Эллиптическая кривая в конечном поле F_p

Пусть задано простое число $p > 3$. Тогда эллиптической кривой E, определенной над конечным простым полем F_p, называется множество пар чисел (x, y), $x \hat{I} F_p$, $y \hat{I} F_p$, удовлетворяющих тождеству

$$y^2 \equiv x^2 + ax + b (\text{mod } p), \tag{4.1}$$

где $a, b \in F_p$ и $4a^3 + 27b^2$ не сравнимо с нулем по модулю p.

Инвариантом эллиптической кривой называется величина $J(E)$, удовлетворяющая тождеству

$$J(E) = 1728 \frac{4a^3}{4a^3 + 27b^2} (\text{mod } p).$$

Коэффициенты a, b эллиптической кривой E по известному инварианту $J(E)$ определяются следующим образом:

$$\begin{cases} a \equiv 3k (\mathrm{mod}\ p) \\ b \equiv 2k (\mathrm{mod}\ p), \end{cases} \text{где } k = \frac{J(E)}{1728 - J(E)} (\mathrm{mod}\ p), J(E) \neq 0 \text{ или } 1728.$$

Пары (x, y), удовлетворяющие тождеству (4.1), называются точками эллиптической кривой E; x и y – соответственно x- и y- координатами точки.

Точки эллиптической кривой будем обозначать Q(x, y) или просто Q. Две точки эллиптической кривой равны, если равны их соответствующие x- и y- координаты.

На множестве всех точек эллиптической кривой E введем операцию сложения, которую будем обозначать знаком +. Для двух произвольных точек $Q_1(x_1, y_1)$ и $Q_2(x_2, y_2)$ эллиптической кривой E рассмотрим несколько вариантов.

Пусть координаты точек Q_1 и Q_2 удовлетворяют условию $x_1 \neq x_2$. В этом случае их суммой будем называть точку $Q_3(x_3, y_3)$, координаты которой определяются сравнениями

$$\begin{cases} x_3 \equiv l^2 - x_1 - x_2 (\mathrm{mod}\ p) \\ y_3 \equiv l(x_1 - x_3) - y (\mathrm{mod}\ p) \end{cases}, \text{ где } l \equiv \frac{y_2 - y_1}{x_2 - x_1} (\mathrm{mod}\ p).$$

Если выполнены равенства $x_1 = x_2$ и $y_1 = y_2 \neq 0$, то определим координаты точки Q_3 следующим образом:

$$\begin{cases} x_3 \equiv l^2 - 2x_1 (\mathrm{mod}\ p) \\ y_3 \equiv l(x_1 - x_3) - y_1 (\mathrm{mod}\ p) \end{cases}, \text{ где } l \equiv \frac{3x2_1^2 + a}{2y_1} (\mathrm{mod}\ p).$$

В случае когда выполнено условие $x_1 = x_2$ и $y_1 = -y_2 (\mathrm{mod}\ p)$, сумму точек Q_1 и Q_2 будем называть нулевой точкой O, не определяя ее x- и y- координаты. В этом случае точка Q_2 называется отрицанием точки Q_1. Для нулевой точки O выполнены равенства

Q + O = O + Q = Q,

где Q - произвольная точка эллиптической кривой E.

Относительно введенной операции сложения множество всех точек эллиптической кривой E, вместе с нулевой точкой, образуют конечную абелеву (коммутативную) группу порядка m, для которого выполнено неравенство

$$p + 1 - 2\sqrt{p} \leq m \leq p + 1 + 2\sqrt{p}.$$

Точка Q называется точкой кратности k, или просто кратной точкой эллиптической кривой E, если для некоторой точки P выполнено равенство

$$Q = \underbrace{P + \ldots + P}_{k} = kP.$$

Эллиптическая кривая в конечном поле F_{2^m} определяется соотношением

$$y^2 + xy \equiv x^3 + ax^2 + b$$

при ненулевом b.

Эллиптической кривой $E(F_{2^m})$, является группа решений $(x, y), x \in F_{2^m}, y \in F_{2^m}$ приведенного выше соотношения при определенных значениях a и b, а также нулевая точка O.

Аналогично группе эллиптической кривой $E(F_p)$, множество всех точек эллиптической кривой $E(F_{2^m})$ вместе с нулевой точкой образуют конечную абелеву группу.

С помощью описанных выше правил сложения можно вычислить точку kP для любого целого числа k и любой точки P эллиптической кривой.

Однако решение обратной задачи – нахождение числа k по известным точкам P и kP – является слишком трудным. Данную задачу называют проблемой дискретного логарифма эллиптической кривой ECDLP (Elliptic Curve Discrete Logariphm Problem).

Решение проблемы ECDLP является значительно более сложным, чем проблемы дискретного логарифмирования (нахождение числа x по заданному числу $y = g^x \bmod p$ при известных основании g и модуле p), на которой базируются RSA-подобные асимметричные криптосистемы.

Сложность решения проблемы ECDLP обусловлена ресурсоемкостью операций сложения и дублирования точек, с помощью которых вычисляется kP, как видно из приведенных выше формул. Отсюда следует возможность применения более коротких ключей. Например, ключу размером 1024 бит алгоритма DSA соответствует по криптостойкости ключ размером 160 бит алгоритма ECDSA (DSA на эллиптических кривых).

Существует несколько реализаций известных криптоалгоритмов на базе эллиптических кривых (стандартизованы в IEEE P1363).

4.3.3. АЛОРИТМ АСИММЕТРИЧНОГО ШИФРОВАНИЯ ECES

В алгоритме ECES (Elliptic Curve Encryption Scheme) сначала должны быть определены следующие параметры, являющиеся открытой информацией, общей для всех пользователей системы:

- конечное поле F_q;
- эллиптическая кривая $E(F_q)$;
- большой простой делитель количества точек кривой n;
- точка P, координаты которой должны иметь тот же порядок, что и число n.
- Каждый пользователь системы генерирует пару ключей следующим образом:
- выбирается случайное целое число $d, 1 < d < n-1$.
- вычисляется точка $Q = dP$.

Секретным ключом пользователя является число d, открытым ключом – точка Q.

Шифрование сообщения (пользователь A шифрует сообщение M для пользователя B):

- сообщение разбивается на блоки M_i , которые определенным образом дополняются слева (длина каждого блока равна $2L$ – 16 бит, где L равно ближайшему большему целому от $\log_2 q$);
- полученный блок разбивается на две части равной длины: m_{i1} и m_{i2};
- выбирается случайное целое число $k, 1 < k < n-1$;
- вычисляется точка $(x_1, y_1) = kP$;
- вычисляется точка $(x_2, y_2) = kQ_B$;
- с помощью определенного преобразования из m_{i1}, m_{i2} и x_2 получают c_1 и c_2;

– зашифрованные данные: (x_1, y_1, c_1, c_2).

Дешифрование сообщения (пользователь B расшифровывает полученное от пользователя A зашифрованное сообщение):

– вычисляется точка $(x_2, y_2) = d(x_1, y_1)$;
– восстанавливается исходное сообщение m_{i1}, m_{i2} из c_1, c_2 и x_2.

4.4. ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ

Электронная цифровая подпись используется для аутентификации текстов, передаваемых по телекоммуникационным каналам. При таком обмене электронными документами существенно снижаются затраты на обработку и хранение документов, ускоряется их поиск. Но возникает проблема аутентификации автора электронного документа и самого документа, то есть установления подлинности автора и отсутствия изменений в полученном электронном документе.

Целью аутентификации электронных документов является их защита от возможных видов злоумышленных действий, к которым относятся:

– **активный перехват** – нарушитель, подключившийся к сети, перехватывает документы (файлы) и изменяет их;
– **маскарад** – абонент С посылает документ абоненту В от имени абонента А;
– **ренегатство** – абонент А заявляет, что не посылал сообщения абоненту В, хотя на самом деле послал;
– **подмена** – абонент В изменяет или формирует новый документ и заявляет, что получил его от абонента А;
– **повтор** – абонент С повторяет ранее переданный документ, который абонент А посылал абоненту В.

Эти виды злоумышленных действий могут нанести существенный ущерб банковским и коммерческим структурам, государственным предприятиям и организациям, частным лицам, применяющим в своей деятельности компьютерные информационные технологии.

Проблему проверки целостности сообщения и подлинности автора сообщения позволяет эффективно решить методология электронной цифровой подписи.

4.4.1. ОСНОВНЫЕ ПРОЦЕДУРЫ ЦИФРОВОЙ ПДПИСИ

Функционально цифровая подпись аналогична обычной рукописной подписи и обладает ее основными достоинствами:

– удостоверяет, что подписанный текст исходит от лица, поставившего подпись;
– не дает самому этому лицу возможности отказаться от обязательств, связанных с подписанным текстом;
– гарантирует целостность подписанного текста.

Электронная цифровая подпись (ЭЦП) представляет собой относительно небольшое количество дополнительной цифровой информации, передаваемой вместе с подписываемым текстом. ЭЦП основана на обратимости асимметричных шифров, а также на взаимосвязанности содержимого сообщения, самой подписи и пары ключей. Изменение хотя бы одного из этих элементов сделает невозможным подтверждение подлинности цифровой подписи. ЭЦП реализуется при помощи асимметричных алгоритмов шифрования и хэш-функций.

Технология применения системы ЭЦП предполагает наличие сети абонентов, посылающих друг другу подписанные электронные документы. Для каждого абонента генерируется пара ключей: секретный и открытый. Секретный ключ хранится абонентом в тайне и используется им для формирования ЭЦП. Открытый ключ известен всем другим пользователям и предназначен для проверки ЭЦП получателем подписанного электронного документа.

Система ЭЦП включает две основные процедуры:

- процедуру формирования цифровой подписи;
- процедуру проверки цифровой подписи.

В процедуре формирования подписи используется секретный ключ отправителя сообщения, в процедуре проверки подписи – открытый ключ отправителя.

Процедура формирования цифровой подписи

На подготовительном этапе этой процедуры абонент A – отправитель сообщения генерирует пару ключей: секретный ключ k_A и открытый ключ K_A. Открытый ключ K_A вычисляется из парного ему секретного ключа k_A. Открытый ключ K_A рассылается остальным абонентам сети (или делается доступным, например, на разделяемом ресурсе) для использования при проверке подписи.

Для формирования цифровой подписи отправитель A прежде всего вычисляет значение хэш-функции $h(M)$ подписываемого текста M (рис. 4.20). Хэш-функция служит для сжатия исходного подписываемого текста M в дайджест m – относительно короткое число, состоящее из фиксированного небольшого числа

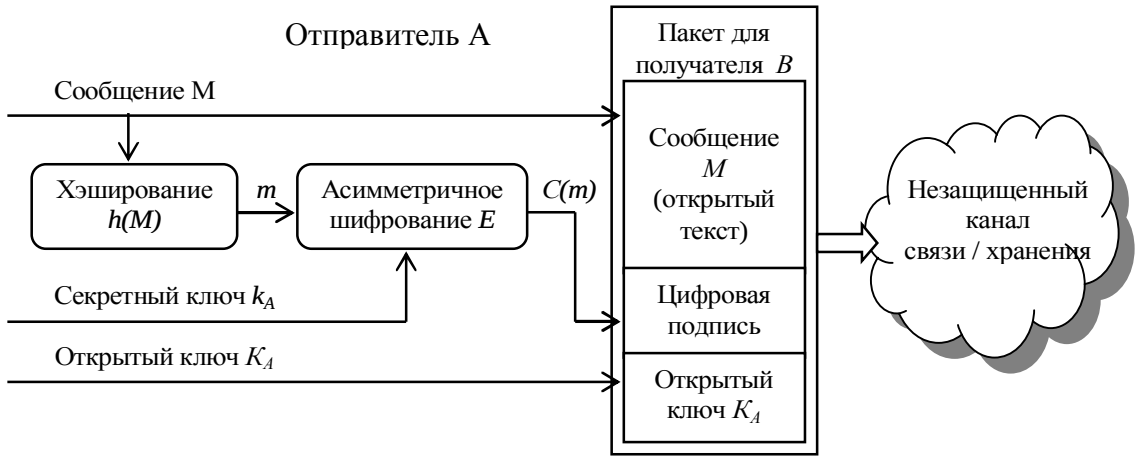


Рис. 4.20. Схема формирования электронной цифровой подписи

битов и характеризующее весь текст M в целом . Далее отправитель A шифрует дайджест своим секретным ключом k_A. Получаемая при этом пара чисел представляет собой цифровую подпись для данного текста M. Сообщение M вместе с цифровой подписью отправляется в адрес получателя.

Процедура проверки цифровой подписи

Абоненты сети могут проверить цифровую подпись полученного сообщения M с помощью открытого ключа отправителя K_A этого сообщения (рис.4.21). При проверке ЭЦП абонент B – получатель сообщения M – расшифровывает принятый дайджест т открытым ключом K_A отправителя A. Кроме того, получатель сам

вычисляет с помощью хэш-функции $h(M)$ дайджест m' принятого сообщения M и сравнивает его с расшифрованным. Если эти два дайджеста m и m' – совпадают, то цифровая подпись является подлинной. В противном случае либо подпись подделана, либо изменено содержание сообщения.

Принципиальным моментом в системе ЭЦП является невозможность подделки ЭЦП пользователя без знания его секретного ключа подписывания. Поэтому необходимо защитить секретный ключ подписывания от несанкционированного доступа. Секретный ключ ЭЦП, аналогично ключу симметричного шифрования, рекомендуется хранить на персональном ключевом носителе в защищенном виде. Электронная цифровая подпись представляет собой уникальное число, зависящее от подписываемого документа и секретного ключа абонента. В качестве подписываемого документа может быть использован любой файл. Подписанный файл создается из неподписанного путем добавления в него одной или более электронных подписей.

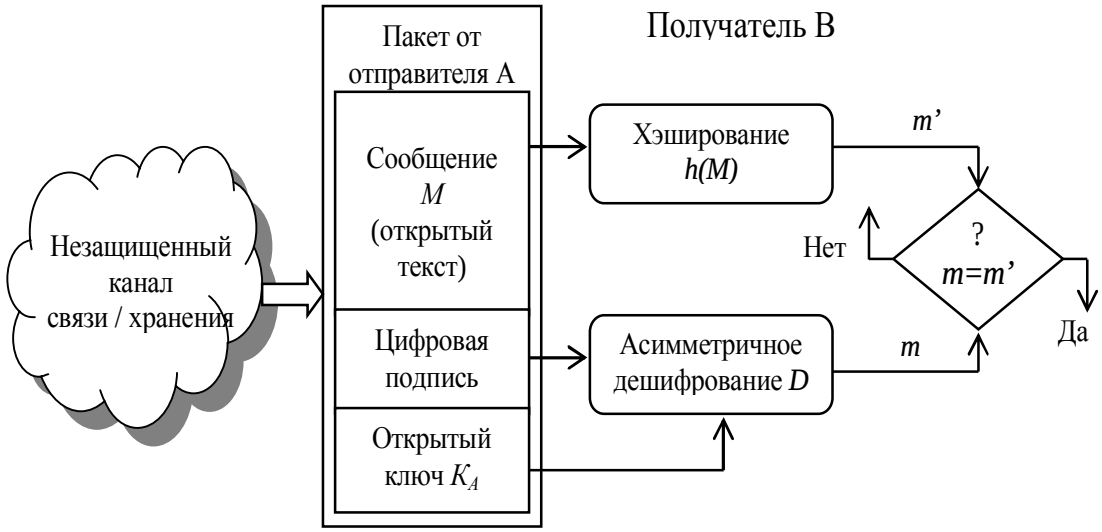


Рис. 4.21. Схема проверки электронной цифровой подписи

Помещаемая в подписываемый файл (или в отдельный файл электронной подписи) структура ЭЦП обычно содержит дополнительную информацию, однозначно идентифицирующую автора подписанного документа. Эта информация добавляется к документу до вычисления ЭЦП, что обеспечивает и ее целостность. Каждая подпись содержит следующую информацию:

- дату подписи;
- срок окончания действия ключа данной подписи;
- информацию о лице, подписавшем файл (Ф.И.О., должность, краткое наименование фирмы);
- идентификатор подписавшего (имя открытого ключа);
- собственно цифровую подпись.

Важно отметить, что, с точки зрения конечного пользователя, процесс формирования и проверки цифровой подписи отличается от процесса криптографического закрытия передаваемых данных следующими особенностями.

При формировании цифровой подписи используется закрытый ключ отправителя, тогда как при шифровании применяется открытый ключ получателя.

При проверке цифровой подписи используется открытый ключ отправителя, а при дешифровании – закрытый ключ получателя.

Проверить сформированную подпись может любое лицо, так как ключ проверки подписи является открытым. При положительном результате проверки подписи делается заключение о подлинности и целостности полученного сообщения, то есть о том, что это сообщение действительно отправлено тем или иным отправителем и не было модифицировано при передаче по сети. Однако, если пользователя интересует, не является ли полученное сообщение повторением ранее отправленного или не было ли оно задержано на пути следования, то он должен проверить дату и время его отправки, а при наличии – порядковый номер.

Аналогично асимметричному шифрованию, необходимо обеспечить невозможность подмены открытого ключа, используемого для проверки ЭЦП. Если предположить, что злоумышленник n имеет доступ к открытым ключам, которые хранит на своем компьютере абонент B, в том числе к открытому ключу K_A абонента A, то он может выполнить следующие действия:

- прочитать из файла, в котором содержится открытый ключ K_A, идентификационную информацию об абоненте A;
- сгенерировать собственную пару ключей k_n и K_n, записав в них идентификационную информацию абонента A;
- подменить хранящийся у абонента B открытый ключ K_A своим открытым ключом K_n, но содержащим идентификационную информацию абонента A.

После этого злоумышленник n может посылать документы абоненту B, подписанные своим секретным ключом k_n. При проверке подписи этих документов абонент B будет считать, что документы подписаны абонентом A и их ЭЦП верна, то есть они не были модифицированы кем-либо. До выяснения отношений непосредственно с абонентом A у абонента B может не появиться сомнений в полученных документах.

Открытые ключи ЭЦП можно защитить от подмены с помощью соответствующих цифровых сертификатов.

Сегодня существует большое количество алгоритмов ЭЦП.

4.4.2. АЛГОРИТМ ЦИФРОВОЙ ПОДПИСИ DSA

Алгоритм цифровой подписи DSA (Digital Signature Algorithm) был предложен в 1991 году Национальным институтом стандартов и технологии США (National Institute of Standards and Technology – NIST) и стал стандартом США в 1993 году. Алгоритм DSA является развитием алгоритмов цифровой подписи Эль Гамаля и К. Шнорра. Ниже приводятся процедуры генерации ключей, генерации подписи и проверки подписи в алгоритме DSA.

Генерация ключей DSA

Отправитель и получатель электронного документа используют при вычислениях большие целые числа: g и p – простые числа, длиной L битов каждое ($512 \le L \le 1024$);

q – простое число длиной 160 бит (делитель числа ($p - 1$). Числа g, p, q являются открытыми и могут быть общими для всех пользователей сети.

Отправитель выбирает случайное целое число x, $1 < x < q$. Число x является секретным ключом отправителя для формирования электронной цифровой подписи.

Затем отправитель вычисляет значение

$y = g^x \bmod p$.

Число y является открытым ключом для проверки подписи отправителя. Число y передается всем получателям документов.

Генерация подписи DSA

Этот алгоритм предусматривает использование односторонней функции хэширования $h(\cdot)$. В стандарте определен алгоритм безопасного хэширования SHA-1. Для того чтобы подписать сообщение M, участник A выполняет следующие шаги:

Шаг 1 – выбирает случайное целое k в интервале [1, q – 1].

Шаг 2 – вычисляет $r = (g^k \bmod p) \bmod q$.

Шаг 3 – вычисляет $k^{-1} \bmod q$.

Шаг 4 – вычисляет $s = k^{-1}\{h(M) + xr\} \bmod q$, где h есть алгоритм хэширования SHA-1.

Шаг 5 – если s = 0 тогда перейти к шагу 1. (Если s = 0, тогда $s^{-1} \bmod q$ не существует; s требуется на шаге 2 процедуры проверки подписи.)

Шаг 6 – подпись для сообщения M есть пара целых чисел (r, s).

Проверка подписи DSA

Для того чтобы проверить подпись (r, s) сообщения M от участника A, участник B делает следующие шаги:

Шаг 1 – Получает подлинную копию открытого ключа y участника A.

Шаг 2 – Вычисляет $w = s^{-1} \bmod q$ и хэш-значение $h(M)$.

Шаг 3 – Вычисляет значения $u_1 = h(M)w \bmod q$ и $u_2 = (rw) \bmod q$.

Шаг 4 – Используя открытый ключ y, вычисляет значение $v = (g^{u1} y^{u2} \bmod p) \bmod q$.

Шаг 5 – Признает подпись (r, s) под документом M подлинной, если $v = r$.

Поскольку r и s являются целыми числами, причем каждое меньше q, подписи DSA имеют длину 320 бит. Безопасность алгоритма цифровой подписи DSA базируется на трудностях задачи дискретного логарифмирования.

4.4.3. СТАНДАРТ ЦИФРОВОЙ ПОДПИСИ ГОСТ Р 34.10-94

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 до 1024 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 34.11-94 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:

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

   $s = (x \times r + kH(m))) \bmod q$.

   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:

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

   $z_1 = (s \times v) \bmod q$,

   $z_2 = ((q - r) \times v) \bmod q$,

   $u = ((a^{z_1} \times y^{z_2}) \bmod p) \bmod q$.

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

   Различие между этим алгоритмом и алгоритмом DSA заключается в том, что в DSA

   $s = (k(x \times r + (H(m)))) \bmod q$,

   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 года.

4.4.4. АЛГОРИТМ ЦИФРОВОЙ ПОДПИСИ ECDSA

В алгоритме ЭЦП ECDSA (Elliptic Curve Digital Signature Algorithm) определение параметров системы и генерация ключей аналогичны алгоритму асимметричного шифрования ECES.

Генерация ЭЦП (пользователь A подписывает сообщение M):

- вычисляется хэш-собщения *H(M);*
- выбирается случайное целое число k, взаимно простое с n (то есть не имеющее других общих с n делителей, кроме 1; поскольку n является простым числом по определению, данное условие выполняется автоматически),
- $1 < k < n - 1$;
- вычисляется точка $(x, y) = kP$ и $r = x \bmod n$. В случае если $r = 0$, повторяется выбор k;
- вычисляется $s = k^{-1}(H(M) + rd) \bmod n$;
- цифровой подписью сообщения M является пара чисел (r, s).

Проверка ЭЦП (пользователь *B* проверяет ЭЦП пользователя *A* под сообщением *M*):

- если $r = 0$, то полученная ЭЦП неверна;
- вычисляется хэш-собщения H(M);
- вычисляются $u = s^{-1}H(M) \bmod n$ и $v = s^{-1}r \bmod n$;
- вычисляется точка $(x_1, y_1) = uP + vQ$;
- вычисляется $r' = x_1 \bmod n$;
- ЭЦП считается верной, если $r' = r$.

4.4.5. СТАНДАРТ ЦИФРОВОЙ ПОДПИСИ ГОСТ Р 34.10-2001

Российский стандарт цифровой подписи ГОСТ Р 34.10-2001 был принят в 2001 году. Этот стандарт разработан взамен первого стандарта цифровой подписи ГОСТ Р 34.10-94. Необходимость разработки стандарта ГОСТ Р 34.10-2001 вызвана потребностью в повышении стойкости электронной цифровой подписи к несанкционированным изменениям. Стойкость ЭЦП основывается на сложности вычисления дискретного логарифма в группе точек эллиптической кривой, а также на стойкости используемой хэш-функции по ГОСТ Р 34.11.

Принципиальное отличие нового стандарта от предыдущего ГОСТ Р 34.10-94 состоит в том, что все вычисления при генерации и проверке ЭЦП в новом алгоритме производятся в группе точек эллиптической кривой, определенной над конечным полем F_p.

Принадлежность точки (пары чисел *x* и *y*) к данной группе определяется следующим соотношением:

$$y^2 = x^3 + ax + b(\bmod p),$$

где модуль системы *p* является простым числом, большим 3, а *a* и *b* - константы, удовлетворяющие следующим соотношениям: $a, b \in F_p$ и $4a^3 + 27b^2$ не сравнимо с нулем по модулю *p*.

При этом следует отметить, что принципы вычислений по данному алгоритму схожи с предшествующим российским стандартом ЭЦП. Математические подробности реализации этого алгоритма приводятся ниже.

Обозначения

В данном стандарте использованы следующие обозначения:

V_{256} – множество всех двоичных векторов длиной 256 бит;

$V_¥$ – множество всех двоичных векторов произвольной конечной длины;

Z – множество всех целых чисел;

p – простое число, $p > 3$;

F_p – конечное простое поле, представляемое как множество из *p* целых чисел $\{0, 1, ..., p-1\}$;

$b(\bmod p)$ – минимальное неотрицательное число, сравнимое с *b* по модулю *p*;

M – сообщение пользователя, $M \in V_¥$;

$(\overline{h_1} \square \overline{h_2})$ – конкатенация (объединение) двух двоичных векторов;

a, *b* – коэффициенты эллиптической кривой;

m – порядок группы точек эллиптической кривой;

q – порядок подгруппы группы точек эллиптической кривой;

O – нулевая точка эллиптической кривой;

P – точка эллиптической кривой порядка q;

d – целое число – ключ подписи;

Q – точка эллиптической кривой – ключ проверки;

w – цифровая подпись под сообщением M.

Общие положения

Механизм цифровой подписи реализуется посредством двух основных процессов:

- формирования цифровой подписи;
- проверки цифровой подписи.

В процессе формирования цифровой подписи в качестве исходных данных используются сообщение M, ключ подписи d и параметры схемы ЭЦП, а в результате формируется цифровая подпись w.

Ключ подписи d является элементом секретных данных, специфичным для субъекта и используемым только данным субъектом в процессе формирования цифровой подписи.

Параметры схемы ЭЦП – элементы данных, общие для всех субъектов схемы цифровой подписи, известные или доступные всем этим субъектам. Электронная цифровая подпись w представляет собой строку битов, полученную в результате процесса формирования подписи. Данная строка имеет внутреннюю структуру, зависящую от конкретного механизма формирования подписи.

В процессе проверки цифровой подписи в качестве исходных данных используются подписанное сообщение, ключ проверки Q и параметры схемы ЭЦП, а результатом проверки является заключение о правильности или ошибочности цифровой подписи.

Ключ проверки Q является элементом данных, математически связанным с ключом подписи d и используемым проверяющей стороной в процессе проверки цифровой подписи.

Схематическое представление подписанного сообщения показано на рис. 4.22. Поле «Текст», дополняющее поле «Цифровая подпись», может содержать

Рис. 4.22. Схема подписанного сообщения

идентификаторы субъекта, подписавшего сообщение, и/или метку времени.

Установленная в данном стандарте схема цифровой подписи должна быть реализована с использованием операций группы точек эллиптической кривой, определенной над конечным простым полем, а также хэш-функции.

Криптографическая стойкость данной схемы цифровой подписи основывается на сложности решения задачи дискретного логарифмирования в группе точек эллиптической кривой, а также на стойкости используемой хэш-функции. Алгоритм вычисления хэш-функции установлен в ГОСТ Р 34.11.

Цифровая подпись представляет собой двоичный вектор длиной 512 бит, вычисляется и проверяется с помощью определенного набора правил изложенных ниже.

Параметры схемы цифровой подписи для ее формирования и проверки:

- простое число p – модуль эллиптической кривой. Это число должно удовлетворять неравенству $p > 2^{255}$. Верхняя граница данного числа должна определяться при конкретной реализации схемы цифровой подписи;
- эллиптическая кривая E, задаваемая своим инвариантом $J(E)$ или коэффициентами $a, b \in F_p$;
- целое число m - порядок группы точек эллиптической кривой E;
- простое число q – порядок циклической подгруппы группы точек эллиптической кривой E, для которого выполнены следующие условия:
- $\begin{cases} m = nq, n \in Z, n \geq 1 \\ 2^{254} < q < 2^{256} \end{cases}$;
- точка $P \neq 0$ эллиптической кривой E с координатами (x_p, y_p), удовлетворяющая равенству $qP = 0$;
- хэш-функция h(·): $V_¥$ ® V_{256}, отображающая сообщения, представленные в виде двоичных векторов произвольной конечной длины, в двоичные век торы длины 256 бит. Хэш-функция определена в ГОСТ Р 34.11.

Каждый пользователь схемы цифровой подписи должен обладать личными ключами:

- ключом подписи – целым числом d, удовлетворяющим неравенству $0 < d < q$;
- ключом проверки – точкой эллиптической кривой Q с координатами (x_q, y_q), удовлетворяющей равенству $dP = Q$.

На приведенные выше параметры схемы цифровой подписи накладываются следующие требования:

- должно быть выполнено условие $p^t \neq 1 (\mathrm{mod}\ p)$ для всех целых $t = 1, 2, ..., B$, где B удовлетворяет неравенству $B \geq 31$;
- должно быть выполнено неравенство $m \neq p$;
- инвариант кривой должен удовлетворять условию $j(E) \neq 0$ или 1728.

Двоичные векторы

Для определения процессов формирования и проверки цифровой подписи необходимо установить соответствие между целыми числами и двоичными векторами длиной 256 бит.

Рассмотрим следующий двоичный вектор длиной 256 бит, в котором младшие биты расположены справа, а старшие - слева:

$$\bar{h} = (a_{255}, ..., a_0), \bar{h} \in V_{255},$$

где a_i, i = 0..256, равно либо 1, либо 0. Будем считать, что число $a \in Z$ соответствует двоичному вектору h, если выполнено равенство

$$a = \sum_{i=0}^{255} a_i 2^i.$$

Для двух двоичных векторов $\bar{h}_1$ и $\bar{h}_2$, соответствующих целым числам a и b, определим операцию конкатенации (объединения) следующим образом. Пусть

$\bar{h}_1 = (a_{255},...,a_0)$,

$\bar{h}_2 = (b_{255},...,b_0)$,

тогда их объединение имеет вид

$$\bar{h}_1 \| \bar{h}_2 = (a_{255},...,a_0, b_{255},...,b_0)$$

и представляет собой двоичный вектор длиной 512 бит, составленный из коэффициентов векторов $\bar{h}_1$ и $\bar{h}_2$.

С другой стороны, приведенные формулы определяют способ разбиения двоичного вектора $\bar{h}$ длиной 512 бит на два двоичных вектора длиной 256 бит, конкатенацией которых он является.

Основные процессы

В данном разделе определены процессы формирования и проверки электронной цифровой подписи под сообщением пользователя.

Для реализации данных процессов необходимо, чтобы всем пользователям были известны параметры схемы цифровой подписи, удовлетворяющие приведенным выше требованиям.

Кроме того, каждому пользователю необходимо иметь ключ подписи d и ключ проверки подписи $Q(x_q,y_q)$, которые также должны удовлетворять приведенным выше требованиям.

Формирование цифровой подписи. Для получения цифровой подписи под сообщением $M \in Vm$ необходимо выполнить следующие действия (шаги).

Шаг 1 – вычислить хэш-код сообщения M:

$\bar{h} = h(M)$.

Шаг 2 – вычислить целое число a, двоичным представлением которого является вектор $\bar{h}$, и определить значение

$e \equiv a \pmod{q}$.

Если $e = 0$, то определить $e = 1$.

Шаг 3 – сгенерировать случайное (псевдослучайное) целое число k, удовлетворяющее неравенству

$0 < k < q$.

Шаг 4 – вычислить точку эллиптической кривой $C = kP$ и определить

$r \equiv x_c \pmod{q}$,

где x_i – x-координата точки C. Если $r = 0$, то вернуться к шагу 3.

Шаг 5 – вычислить значение

$s = (rd + ke) \pmod{q}$.

Если $s = 0$, то вернуться к шагу 3.

Шаг 6 – вычислить двоичные векторы $\bar{r}$ и $\bar{s}$, соответствующие r и s, и определить цифровую подпись $w = (\bar{r} \| \bar{s})$ как конкатенацию двух двоичных векторов.

Исходными данными этого процесса являются ключ подписи d и подписываемое сообщение M, а выходным результатом – цифровая подпись w.

Проверка цифровой подписи. Для проверки цифровой подписи w, под полученным сообщением M необходимо выполнить следующие действия (шаги).

Шаг 1 – по полученной подписи w вычислить целые числа r и s. Если выполнены неравенства $0 < r < q$, $0 < s < q$, то перейти к следующему шагу. В противном случае подпись неверна.

Шаг 2 – вычислить хэш-код полученного сообщения M:

$$\overline{h} = h(M).$$

Шаг 3 – вычислить целое число a, двоичным представлением которого является вектор $\overline{h}$, и определить

$$e \equiv a \pmod{q}.$$

Если $e = 0$, то определить $e = 1$.

Шаг 4 – вычислить значение

$$v \equiv e^{-1} \pmod{q}.$$

Шаг 5 – вычислить значения

$$z_1 \equiv sv \pmod{q}, \quad z_2 \equiv -rv \pmod{q}.$$

Шаг 6 – вычислить точку эллиптической кривой $C = z_1 P + z_2 Q$ и определить

$$R \equiv x_c \pmod{q},$$

где x_c – x-координата точки C.

Шаг 7 – если выполнено равенство $R = r$, то подпись принимается, в противном случае подпись неверна.

Исходными данными этого процесса являются подписанное сообщение M, цифровая подпись w и ключ проверки Q, а выходным результатом – свидетельство о достоверности или ошибочности данной подписи.

Внедрение цифровой подписи на базе стандарта ГОСТ Р 34.10-2001 повышает, по сравнению с предшествующей схемой цифровой подписи, уровень защищенности передаваемых сообщений от подделок и искажений. Этот стандарт рекомендуется использовать в новых системах обработки информации различного назначения, а также при модернизации действующих систем.

4.5. УПРАВЛЕНИЕ КРИПТОКЛЮЧАМИ

Любая криптографическая система основана на использовании криптографических ключей. Под ключевой информацией понимают совокупность всех действующих в информационной сети или системе ключей. Если не обеспечено достаточно надежное управление ключевой информацией, то, завладев ею, злоумышленник получает неограниченный доступ ко всей информации в сети или системе. Управление ключами включает реализацию таких функций, как генерация, хранение и распределение ключей. Распределение ключей – самый ответственный процесс в управлении ключами.

При использовании симметричной криптосистемы две вступающие в информационный обмен стороны должны сначала согласовать секретный сессионный ключ, то есть ключ для шифрования всех сообщений, передаваемых в процессе обмена. Этот ключ должен быть неизвестен всем остальным и должен периодически обновляться одновременно у отправителя и получателя. Процесс согласования сессионного ключа называют также обменом или распределением ключей.

Асимметричная криптосистема предполагает использование двух ключей – открытого и закрытого (секретного). Открытый ключ можно разглашать, а закрытый

надо хранить в тайне. При обмене сообщениями необходимо пересылать только открытый ключ, обеспечив его подлинность.

К распределению ключей предъявляются следующие требования:

- оперативность и точность распределения;
- конфиденциальность и целостность распределяемых ключей.

Для распределения ключей между пользователями компьютерной сети используются следующие основные способы:

1) Использование одного или нескольких центров распределения ключей.
2) Прямой обмен ключами между пользователями сети.

Проблемой первого подхода является то, что центру распределения ключей известно, кому и какие ключи распределены, и это позволяет читать все сообщения, передаваемые по сети. Возможные злоупотребления могут существенно нарушить безопасность сети. При втором подходе проблема состоит в том, чтобы надежно удостовериться в подлинности субъектов сети.

Задача распределения ключей сводится к построению такого протокола распределения ключей, который обеспечивает:

- взаимное подтверждение подлинности участников сеанса;
- подтверждение достоверности сеанса;
- использование минимального числа сообщений при обмене ключами.

Остановимся подробнее на втором подходе – прямом обмене ключами между пользователями сети.

При использовании для защищенного информационного обмена криптосистемы с симметричным секретным ключом, два пользователя, желающие обменяться криптографически защищенной информацией, должны обладать общим секретным ключом. Эти пользователи должны обменяться общим ключом по каналу связи безопасным образом. Если пользователи меняют ключ достаточно часто, то доставка ключа превращается в серьезную проблему.

Для решения этой проблемы можно применить два основных способа:

1) Использование асимметричной криптосистемы с открытым ключом для защиты секретного ключа симметричной криптосистемы.
2) Использование системы открытого распределения ключей Диффи-Хеллмана.

Реализация первого способа, осуществляется в рамках комбинированной криптосистемы с симметричными и асимметричными ключами. При таком подходе симметричная криптосистема применяется для шифрования и передачи исходного открытого текста, а асимметричная криптосистема с открытым ключом – для шифрования, передачи и последующего дешифрования только секретного ключа симметричной криптосистемы.

Второй способ безопасного распространения секретных ключей основан на применении алгоритма открытого распределения ключей Диффи-Хеллмана. Этот алгоритм позволяет пользователям обмениваться ключами по незащищенным каналам связи.

4.5.1. ИСПОЛЬЗОВАНИЕ КОМБИНИРОВАННОЙ КРИПТОСИСТЕМЫ

Анализ рассмотренных выше особенностей симметричных и асимметричных криптографических систем показывает, что при совместном использовании эти криптосистемы могут эффективно дополнять друг друга, компенсируя недостатки каждого из них.

Действительно, главным достоинством асимметричных криптосистем с открытым ключом является их потенциально высокая безопасность: нет необходимости ни передавать, ни сообщать кому-либо значения секретных ключей, ни убеждаться в их подлинности. Однако быстродействие асимметричных криптосистем с открытым ключом обычно в сотни и более раз меньше быстродействия симметричных криптосистем с секретным ключом.

В свою очередь, быстродействующие симметричные криптосистемы страдают существенным недостатком: обновляемый секретный ключ симметричной криптосистемы должен регулярно передаваться партнерам по информационному обмену и во время этих передач возникает опасность раскрытия секретного ключа.

Совместное использование этих криптосистем позволяет эффективно реализовать такую базовую функцию защиты, как криптографическое закрытие передаваемой информации с целью обеспечения ее конфиденциальности.

Комбинированное применение симметричного и асимметричного шифрования позволяет устранить основные недостатки, присущие обоим методам. Комбинированный (гибридный) метод шифрования позволяет сочетать преимущества высокой секретности, предоставляемые асимметричными криптосистемами с открытым ключом, с преимуществами высокой скорости работы, присущими симметричным криптосистемам с секретным ключом.

Метод комбинированного использования симметричного и асимметричного шифрования заключается в следующем.

Симметричную криптосистему применяют для шифрования исходного открытого текста, а асимметричную криптосистему с открытым ключом – только для шифрования секретного ключа симметричной криптосистемы. В результате асимметричная криптосистема с открытым ключом не заменяет, а лишь дополняет симметричную криптосистему с секретным ключом, позволяя повысить в целом защищенность передаваемой информации. Такой подход иногда называют схемой электронного «цифрового конверта».

Пусть пользователь A хочет применить комбинированный метод шифрования для защищенной передачи сообщения M пользователю B. Тогда последовательность действий пользователей A и B будет следующей.

Действия пользователя A:

1) Создает (например, генерирует случайным образом) сеансовый секретный ключ K_S, который будет использован в алгоритме симметричного шифрования для шифрования конкретного сообщения или цепочки сообщений.
2) Зашифровывает симметричным алгоритмом сообщение M на сеансовом секретном ключе K_S.
3) Зашифровывает асимметричным алгоритмом секретный сеансовый ключ K_S на открытом ключе K_B пользователя B (получателя сообщения).
4) Передает по открытому каналу связи в адрес пользователя B зашифрованное сообщение M вместе с зашифрованным сеансовым ключом K_S.

Действия пользователя A иллюстрируются схемой шифрования сообщения комбинированным методом (рис. 4.23).

Действия пользователя B (при получении электронного «цифрового конверта» – зашифрованного сообщения M и зашифрованного сеансового ключа k_S), рис 4.22:

5) Расшифровывает асимметричным алгоритмом сеансовый ключ k_S с помощью своего секретного ключа k_B.

6) Расшифровывает симметричным алгоритмом принятое сообщение *M* с помощью полученного сеансового ключа k_S.

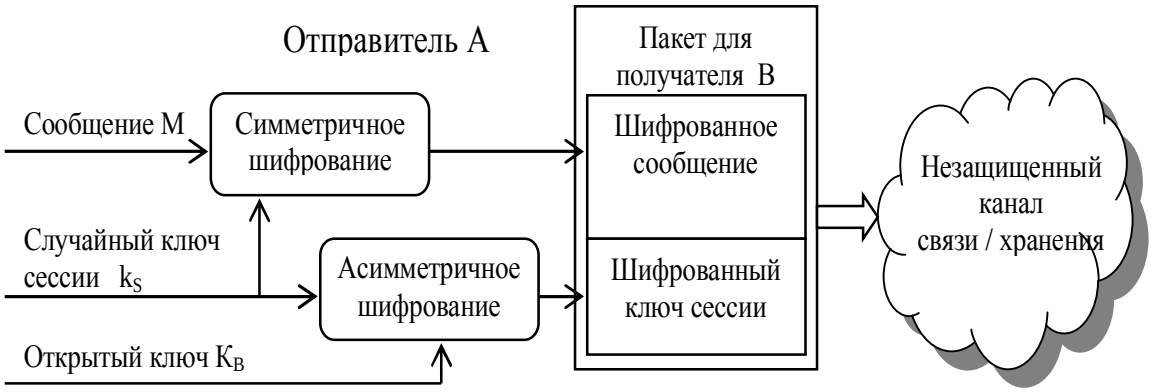


Рис. 1.23. Схема шифрования сообщения комбинированным методом

Действия пользователя *B* иллюстрируются схемой дешифрования сообщения комбинированным методом (рис.4.24).

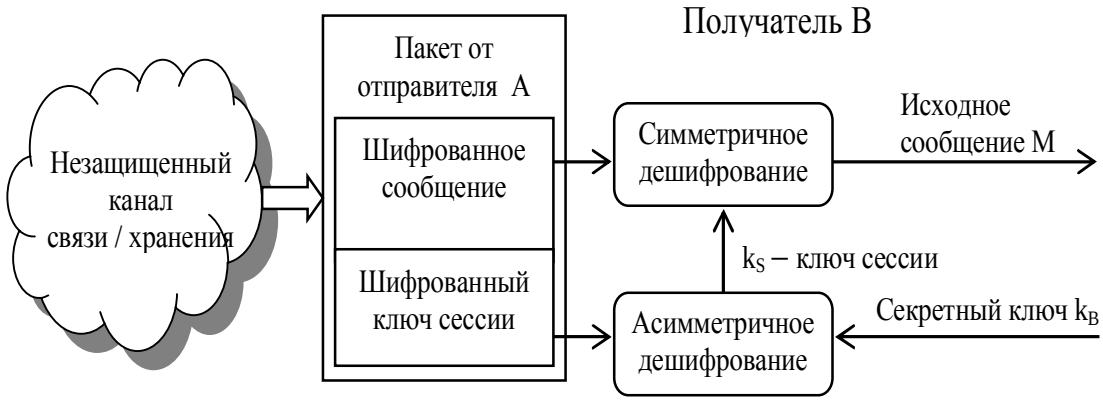


Рис. 4.24. Схема дешифрования сообщения комбинированным методом

Полученный электронный «цифровой конверт» может раскрыть только законный получатель – пользователь *B*. Только пользователь *B*, владеющий личным секретным ключом k_B, сможет правильно расшифровать секретный ключ сессии k_S и затем с помощью этого ключа расшифровать и прочитать полученное сообщение *M*.

При методе «цифрового конверта» недостатки симметричного и асимметричного криптоалгоритмов компенсируются следующим образом:

- проблема распространения ключей симметричного криптоалгоритма устраняется тем, что сеансовый ключ k_S, на котором шифруется собственно сообщения, передается по открытым каналам связи в зашифрованном виде; для шифрования ключа k_S используется асимметричный криптоалгоритм;
- проблемы медленной скорости асимметричного шифрования в данном случае практически не возникает, поскольку асимметричным

криптоалгоритмом шифруется только короткий ключ k_S, а все данные шифруются быстрым симметричным криптоалгоритмом.

В результате получают быстрое шифрование в сочетании с удобным распределением ключей.

С целью защиты от разглашения секретных ключей симметричного шифрования любой из сторон обмена, когда требуется реализовать протоколы взаимодействия не доверяющих друг другу сторон, используется следующий способ взаимодействия. Для каждого сообщения на основе случайных параметров генерируется отдельный секретный ключ симметричного шифрования, который зашифровывается асимметричной системой для передачи вместе с сообщением, зашифрованным этим ключом. В этом случае разглашение ключа симметричного шифрования не будет иметь смысла, так как для шифрования следующего сообщения будет использован другой случайный секретный ключ.

При комбинированном методе шифрования применяются криптографические ключи как симметричных, так и асимметричных криптосистем. Очевидно, выбор длин ключей для криптосистемы каждого типа следует осуществлять таким образом, чтобы злоумышленнику было одинаково трудно атаковать любой механизм защиты комбинированной криптосистемы.

В табл. 4.12 приведены распространенные длины ключей симметричных и асимметричных криптосистем, для которых трудность атаки полного перебора примерно равна трудности факторизации соответствующих модулей асимметричных криптосистем.

Таблица 4.12. Длины ключей для симметричных и асимметричных криптосистем при одинаковой их криптостойкости

Длина ключа симметричной криптосистемы, бит	Длина ключа асимметричной криптосистемы, бит
56	384
64	512
80	768
112	1792
128	2304

Если используется короткий сеансовый ключ (например, 56-битовый алгоритма DES), то не имеет значения, насколько велики асимметричные ключи. Злоумышленник будет атаковать не их, а сеансовый ключ.

4.5.2. МЕТОД РАСПРЕДЕЛЕНИЯ КЛЮЧЕЙ ДИФФИ-ХЕЛЛМАНА

У.Диффи и М.Хеллман изобрели метод открытого распределения ключей в 1976 году. Этот метод позволяет пользователям обмениваться ключами по незащищенным каналам связи. Его безопасность обусловлена трудоемкостью вычисления дискретных логарифмов в конечном поле, в отличие от легкости решения прямой задачи дискретного возведения в степень в том же конечном поле.

Суть метода Диффи-Хеллмана заключается в следующем (рис. 4.25).

Пользователи A и B, участвующие в обмене информацией, генерируют независимо друг от друга свои случайные секретные ключи k_A и k_B (ключи k_A и k_B – случайные большие целые числа, которые хранятся пользователями A и B в секрете).

Затем пользователь A вычисляет на основании своего секретного ключа k_A открытый ключ

$$K_A = g^{k_A} (\mathrm{mod}\, N),$$

одновременно пользователь B вычисляет на основании своего секретного ключа k_B открытый ключ

$$K_B = g^{k_B} (\mathrm{mod}\, N),$$

где N и g – большие целые простые числа. Арифметические действия выполняются с приведением по модулю N. Числа N и g могут не храниться в секрете. Как правило, эти значения являются общими для всех пользователей сети или системы.

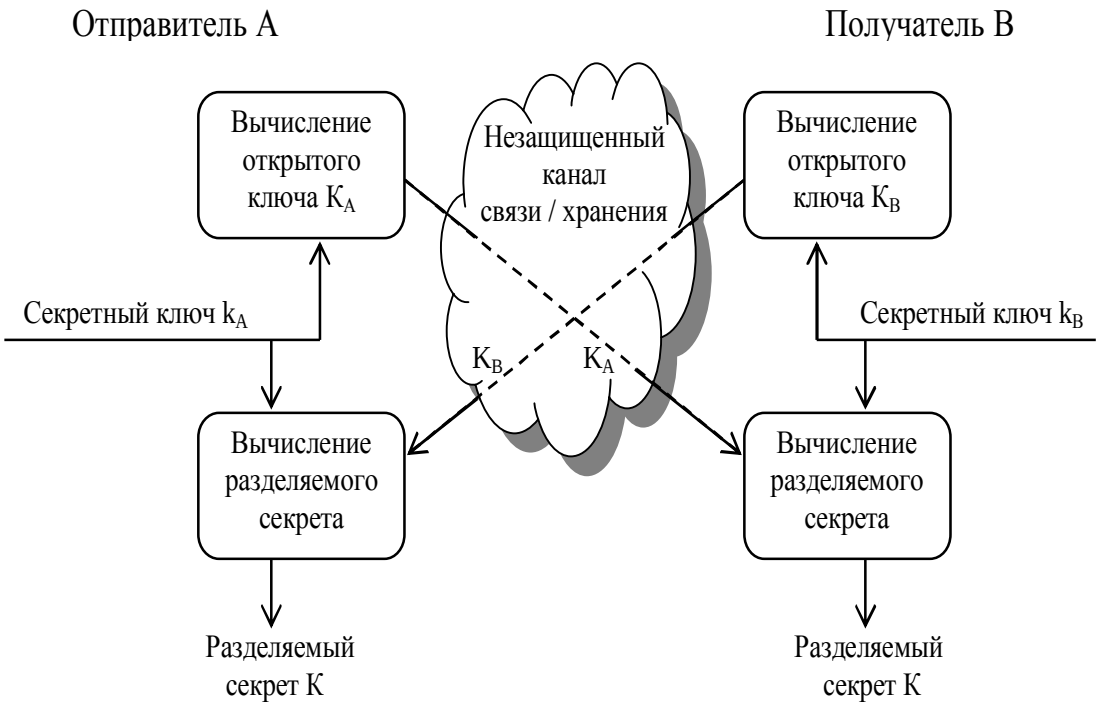


Рис. 4.25. Схема открытого распределения ключей Диффи-Хеллмана

Затем пользователи A и B обмениваются своими открытыми ключами K_A и K_B по незащищенному каналу и используют их для вычисления общего сессионного ключа K (разделяемого секрета):

- пользователь A: $K = (K_B)^{k_A} (\mathrm{mod}\, N) = (g^{k_B})^{k_A} (\mathrm{mod}\, N)$;
- пользователь B: $K' = (K_A)^{k_B} (\mathrm{mod}\, N) = (g^{k_A})^{k_B} (\mathrm{mod}\, N)$;
- при этом K = K', так как $(g^{k_B})^{k_A} (\mathrm{mod}\, N) = (g^{k_A})^{k_B} (\mathrm{mod}\, N)$.

Таким образом, результатом этих действий оказывается общий сессионный ключ, который является функцией обоих секретных ключей – k_A и k_B.

Злоумышленник, перехвативший значения открытых ключей K_A и K_B, не может вычислить сессионный ключ K, потому что он не имеет секретных ключей k_A и k_B. Благодаря использованию однонаправленной функции операция вычисления открытого ключа необратима, то есть невозможно по значению открытого ключа абонента вычислить его секретный ключ.

Уникальность метода Диффи-Хеллмана заключается в том, что пара абонентов имеет возможность получать известное только им секретное число, передавая по

открытой сети открытые ключи. После этого абоненты могут приступить к защите передаваемой информации с использованием полученного разделяемого секрета.

Схема Диффи-Хеллмана дает возможность шифровать данные при каждом сеансе связи на новых ключах. Не следует забывать, что любое хранение секретов повышает вероятность попадания их в руки конкурентов или противника.

Схема Диффи-Хеллмана позволяет реализовать метод комплексной защиты конфиденциальности и аутентичности передаваемых данных. Эта схема предоставляет пользователям возможность сформировать и использовать одни и те же ключи для выполнения цифровой подписи и симметричного шифрования передаваемых данных.

Метод комплексной защиты конфиденциальности и аутентичности передаваемых данных

Для одновременной защиты целостности и конфиденциальности данных целесообразно применять шифрование и электронную цифровую подпись в комплексе. Промежуточные результаты работы схемы Диффи-Хеллмана могут быть использованы в качестве исходных данных для реализации метода комплексной защиты целостности и конфиденциальности передаваемых данных.

Действительно, согласно данному алгоритму пользователи A и B сначала генерируют секретные ключи k_A и k_B и вычисляют свои открытые ключи K_A и K_B. Затем абоненты A и B используют эти промежуточные результаты для одновременного вычисления общего разделяемого секретного ключа K, который может использоваться для симметричного шифрования данных.

Метод комплексной защиты конфиденциальности и аутентичности передаваемых данных работает по следующей схеме:

- абонент A подписывает сообщение M с помощью своего секретного ключа k_A, используя стандартный алгоритм цифровой подписи;
- абонент A вычисляет совместно разделяемый секретный ключ K по алгоритму Диффи-Хеллмана из своего секретного ключа k_A и открытого ключа K_B абонента B;
- абонент A зашифровывает сообщение M на полученном совместно разделяемом ключе K, используя согласованный с партнером по обмену алгоритм симметричного шифрования;
- абонент B при получении зашифрованного сообщения M вычисляет по алгоритму Диффи-Хеллмана совместно разделяемый секретный ключ K из своего секретного ключа k_B и открытого ключа K_A абонента A;
- абонент B расшифровывает полученное сообщение M на ключе K;
- абонент B проверяет подпись расшифрованного сообщения M с помощью открытого ключа абонента K_A.

На основании схемы Диффи-Хеллмана, функционируют протоколы управления криптоключами SKIP (Simple Key management for Internet Protocols) и IKE (Internet Key Exchange), применяемые при построении защищенных виртуальных сетей VPN на сетевом уровне.

4.5.3. ПРОТОКОЛ ВЫЧИСЛЕНИЯ КЛЮЧА ПАРНОЙ СВЯЗИ ECKER

В протоколе вычисления ключа эллиптической кривой ECKER (Elliptic Curve Key Establishment Protocol) определение параметров системы и генерация ключей аналогичны алгоритму ассиметричного шифрования ECES.

Предположим , что общий ключ вычисляется пользователями *A* и *B*.

Пользователь *A* имеет секретный ключ *a* и открытый ключ

$Q_A = aP = (x_A, y_A)$.

Аналогично, пользователь *B* имеет секретный ключ *b* и открытый ключ

$Q_B = bP = (x_B, y_B)$.

Вычисление ключа парной связи производится в четыре этапа.

Этап 1. Действия пользователя А:

- выбираем случайное число k_A, $1 \le k_A \le n - 1$;
- вычисляется точка $R_A = k_A P$;
- вычисляется точка $(x_1, y_1) = k_A Q_B$;
- вычисляется $s_A = k_A + a x_A x_1 \bmod n$;
- R_A отправляется пользователю *B*.

Этап 2. Действия пользователя В:

- выбираем случайное число k_B, $1 \le k_B \le n - 1$;
- вычисляется точка $R_B = k_B P$;
- вычисляется точка $(x_2, y_2) = k_B Q_A$;
- вычисляется $s_B = k_B + b x_B x_2 \bmod n$;
- R_B отправляется пользователю *A*.

Этап 3. Действия пользователя А:

- вычисляется $(x_2, y_2) = aR_B$;
- вычисляется ключ парной связи $K = s_A (R_B + x_B x_2 Q_B)$.

Этап 4. Действия пользователя В:

- вычисляется $(x_1, y_1) = bR_A$;
- вычисляется ключ парной связи $K = s_B(R_A + x_A x_1 Q_A)$, что эквивалентно значению $s_A (R_B + x_B x_2 Q_B)$.

Важным достоинством схемы распределения ключей Диффи-Хеллмана и протокола вычисления ключа парной связи ECKER является то, что они позволяют обойтись без защищенного канала для передачи ключей. Однако необходимо иметь гарантию того, что пользователь *A* получил открытый ключ именно от пользователя *B*, и наоборот. Эта проблема решается с помощью сертификатов открытых ключей, создаваемых и распространяемых центрами сертификации *CA* (Certification Authoriti) в рамках инфраструктуры управления открытыми ключами PKI (Public Key Infrastructure).

ЧАСТЬ 5. ТЕХНОЛОГИИ ИДЕНТИФИКАЦИИ И АУТЕНТИФИКАЦИИ

Применение открытых каналов передачи данных создает потенциальные возможности для действий злоумышленников (нарушителей). Поэтому одной из важных задач обеспечения информационной безопасности при взаимодействии пользователей является использование методов и средств, позволяющих одной (проверяющей) стороне убедиться в подлинности другой (проверяемой) стороны. Обычно для решения данной проблемы применяются специальные приемы, дающие возможность проверить подлинность проверяемой стороны.

5.1. ОСНОВНЫЕ ПОНЯТИЯ

С каждым зарегистрированным в компьютерной системе субъектом (пользователем или процессом, действующим от имени пользователя) связана некоторая информация, однозначно идентифицирующая его. Это может быть число или строка символов, именующая данный субъект. Эту информацию называют идентификатором субъекта. Если пользователь имеет идентификатор, зарегистрированный в сети, он считается легальным (законным) пользователем; остальные пользователи относятся к нелегальным. Прежде чем получить доступ к ресурсам компьютерной системы, пользователь должен пройти процесс первичного взаимодействия с компьютерной системой, который включает идентификацию и аутентификацию.

Идентификация (Identification) – это процедура распознавания пользователя по его идентификатору (имени). Эта функция выполняется в первую очередь, когда пользователь делает попытку войти в сеть. Пользователь сообщает системе по ее запросу свой идентификатор, и система проверяет в своей базе данных его наличие.

Аутентификация (Authentication) – процедура проверки подлинности заявленного пользователя, процесса или устройства. Эта проверка позволяет достоверно убедиться, что пользователь (процесс или устройство) является именно тем, кем себя объявляет. При проведении аутентификации, проверяющая сторона убеждается в подлинности проверяемой стороны, при этом проверяемая сторона тоже активно участвует в процессе обмена информацией. Обычно пользователь подтверждает свою идентификацию, вводя в систему уникальную, неизвестную другим пользователям информацию о себе (например, пароль или сертификат).

Идентификация и аутентификация являются взаимосвязанными процессами распознавания и проверки подлинности субъектов (пользователей). Именно от них зависит последующее решение системы, можно ли разрешить доступ к ресурсам системы конкретному пользователю или процессу. После идентификации и аутентификации субъекта выполняется его авторизация.

Авторизация (Authorization) – процедура предоставления субъекту определенных полномочий и ресурсов в данной системе. Иными словами, авторизация устанавливает сферу действия субъекта и доступные ему ресурсы.

Если система не может надежно отличить авторизованное лицо от неавторизованного, конфиденциальность и целостность информации в ней могут быть нарушены. Организации необходимо четко определить свои требования к безопасности, чтобы принимать решения о соответствующих границах авторизации.

С процедурами аутентификации и авторизации тесно связана процедура администрирования действий пользователя.

Администрирование (Accounting) – это регистрация действий пользователя в сети, включая его попытки доступа к ресурсам. Хотя эта учетная информация может быть использована для выписывания счета, с позиций безопасности она особенно важна для обнаружения, анализа инцидентов безопасности в сети и соответствующего реагирования на них. Записи в системном журнале, аудиторские проверки и администрирование ПО – все это может быть использовано для обеспечения подотчетности пользователей, если что-либо случится при входе в сеть с их идентификатором.

Необходимый уровень аутентификации определяется требованиями безопасности, которые установлены в организации. Общедоступные Web-серверы могут разрешить анонимный или гостевой доступ к информации. Финансовые транзакции могут потребовать строгой аутентификации. Примером слабой формы аутентификации может служить использование IP-адреса для определения пользователя. Подмена (spoofing) IP-адреса может легко разрушить этот механизм аутентификации.

Надежная аутентификация является тем ключевым фактором, который гарантирует, что только авторизованные пользователи получат доступ к контролируемой информации.

При защите каналов передачи данных должна выполняться взаимная аутентификация субъектов, то есть взаимное подтверждение подлинности субъектов, связывающихся между собой по линиям связи. Процедура подтверждения подлинности выполняется обычно в начале сеанса в процессе установления соединения абонентов.

Термин «соединение» указывает на логическую связь (потенциально двустороннюю) между двумя субъектами сети. Цель данной процедуры - обеспечить уверенность, что соединение установлено с законным субъектом и вся информация дойдет до места назначения.

Для подтверждения своей подлинности субъект может предъявлять системе разные сущности. В зависимости от предъявляемых субъектом сущностей процессы аутентификации могут быть разделены на следующие категории:

- на основе знания чего-либо. Примерами могут служить пароль, персональный идентификационный код PIN (Personal Identification Number), а также секретные и открытые ключи, знание которых демонстрируется в протоколах типа запрос-ответ;
- на основе обладания чем-либо. Обычно это магнитные карты, смарт-карты, сертификаты и устройства touch memory;
- на основе каких-либо неотъемлемых характеристик. Эта категория включает методы, базирующиеся на проверке биометрических характеристик пользователя (голос, радужная оболочка и сетчатка глаза, отпечатки пальцев, геометрия ладони и др.). В данной категории не используются криптографические методы и средства. Аутентификация на основе биометрических характеристик применяется для контроля доступа в помещения или к какой-либо технике.

Пароль – это то, что знает пользователь и что также знает другой участник взаимодействия. Для взаимной аутентификации участников взаимодействия может быть организован обмен паролями между ними.

Персональный идентификационный номер (PIN) – является испытанным способом аутентификации держателя пластиковой карты и смарт-карты. Секретное значение PIN-кода должно быть известно только держателю карты.

Динамический (одноразовый) пароль – это пароль, который после однократного применения никогда больше не используется. На практике обычно используется регулярно меняющееся значение, которое базируется на постоянном пароле или ключевой фразе.

Система запрос-ответ – одна из сторон инициирует аутентификацию с помощью посылки другой стороне уникального и непредсказуемого значения «запрос», а другая сторона посылает ответ, вычисленный с помощью запроса и секрета. Так как обе стороны владеют одним секретом, то первая сторона может проверить правильность ответа второй стороны.

Сертификаты и цифровые подписи – если для аутентификации используются сертификаты, то требуется применение цифровых подписей на этих сертификатах.

Сертификаты выдаются ответственным лицом в организации пользователя, сервером сертификатов или внешней доверенной организацией. В рамках Интернета появился ряд коммерческих инфраструктур управления открытыми ключами PKI (Public Key Infrastructure) для распространения сертификатов открытых ключей. Пользователи могут получить сертификаты различных уровней.

Процессы аутентификации можно также классифицировать по уровню обеспечиваемой безопасности. В соответствии с данным подходом процессы аутентификации разделяются на следующие типы:

- аутентификация, использующая пароли и PIN-коды;
- строгая аутентификация на основе использования криптографических методов и средств;
- процессы (протоколы) аутентификации, обладающие свойством доказательства с нулевым знанием;
- биометрическая аутентификация пользователей.

С точки зрения безопасности, каждый из перечисленных типов способствует решению своих специфических задач, поэтому процессы и протоколы аутентификации активно используются на практике. В то же время следует отметить, что интерес к протоколам аутентификации, обладающим свойством доказательства с нулевым знанием, носит скорее теоретический, нежели практический характер, но, возможно, в будущем их начнут активно использовать для защиты информационного обмена.

Основными атаками на протоколы аутентификации являются:

- маскарад (impersonation). Пользователь пытается выдать себя за другого с целью получения полномочий и возможности действий от лица другого пользователя;
- подмена стороны аутентификационного обмена (interleaving attack). Злоумышленник в ходе данной атаки участвует в процессе аутентификационного обмена между двумя сторонами с целью модификации проходящего через него трафика;
- повторная передача (replay attack). Заключается в повторной передаче аутентификационных данных каким-либо пользователем;

– принудительная задержка (forced delay). Злоумышленник перехватывает некоторую информацию и передает ее спустя некоторое время;
– атака с выборкой текста (chosen-text attack). Злоумышленник перехватывает аутентификационный трафик и пытается получить информацию о долговременных криптографических ключах.

Для предотвращения таких атак при построении протоколов аутентификации применяются следующие приемы:

– использование механизмов типа запрос-ответ, меток времени, случайных чисел, идентификаторов, цифровых подписей;
– привязка результата аутентификации к последующим действиям пользователей в рамках системы. Примером подобного подхода может служить осуществление в процессе аутентификации обмена секретными сеансовыми ключами, которые применяются при дальнейшем взаимодействии пользователей;
– периодическое выполнение процедур аутентификации в рамках уже установленного сеанса связи и т.п.

Механизм запроса-ответа состоит в следующем. Если пользователь *A* хочет быть уверенным, что сообщения, получаемые им от пользователя *B*, не являются ложными, он включает в посылаемое для *B* сообщение непредсказуемый элемент-запрос X (например, некоторое случайное число). При ответе пользователь *B* должен выполнить над этим элементом некоторую операцию (например, вычислить некоторую функцию *f(X)*). Это невозможно осуществить заранее, так как пользователю *B* неизвестно, какое случайное число *X* придет в запросе. Получив ответ с результатом действий *B*, пользователь *A* может быть уверен, что *B* – подлинный.

Недостаток этого метода – возможность установления закономерности между запросом и ответом.

Механизм отметки времени подразумевает регистрацию времени для каждого сообщения. В этом случае каждый пользователь сети может определить, насколько «устарело» пришедшее сообщение, и решить не принимать его, поскольку оно может быть ложным.

В обоих случаях для защиты механизма контроля следует применять шифрование, чтобы быть уверенным, что ответ послан не злоумышленником.

При использовании отметок времени возникает проблема допустимого временного интервала задержки для подтверждения подлинности сеанса. Ведь сообщение с «временным штемпелем», в принципе, не может быть передано мгновенно. Кроме того, компьютерные часы получателя и отправителя не могут быть абсолютно синхронизированы.

При сравнении и выборе протоколов аутентификации необходимо учитывать следующие характеристики:

– наличие взаимной аутентификации. Это свойство отражает необходимость обоюдной аутентификации между сторонами аутентификационного обмена;
– вычислительная эффективность. Количество операций, необходимых для выполнения протокола;
– коммуникационная эффективность. Данное свойство отражает количество сообщений и их длину, необходимую для осуществления аутентификации;

- наличие третьей стороны. Примером третьей стороны может служить доверенный сервер распределения симметричных ключей или сервер, реализующий дерево сертификатов для распределения открытых ключей;
- гарантии безопасности. Примером может служить применение шифрования и цифровой подписи.

5.2. МЕТОДЫ АУТЕНТИФИКАЦИИ, ИСПОЛЬЗУЮЩИЕ ПАРОЛИ И PIN-КОДЫ

Одной из распространенных схем аутентификации является простая аутентификация, которая основана на применении традиционных многоразовых паролей с одновременным согласованием средств его использования и обработки. Аутентификация на основе многоразовых паролей является простым и наглядным примером использования разделяемой информации. Пока в большинстве защищенных виртуальных сетей VPN (Virtual Private Network) доступ клиента к серверу разрешается по паролю. Однако все чаще применяются более эффективные средства аутентификации, например программные и аппаратные системы аутентификации на основе одноразовых паролей, смарт-карт, PIN-кодов и цифровых сертификатов.

5.2.1. АУТЕНТИФИКАЦИЯ НА ОСНОВЕ МНОГОРАЗОВЫХ ПАРОЛЕЙ

Базовый принцип «единого входа» предполагает достаточность одноразового прохождения пользователем процедуры аутентификации для доступа ко всем сетевым ресурсам. Поэтому в современных операционных системах предусматривается централизованная служба аутентификации, которая выполняется одним из серверов сети и использует для своей работы базу данных. В этой базе данных хранятся учетные данные о пользователях сети. В эти учетные данные наряду с другой информацией включены идентификаторы и пароли пользователей.

Процедуру простой аутентификации пользователя в сети можно представить следующим образом. При попытке логического входа в сеть пользователь набирает на клавиатуре компьютера свой идентификатор и пароль. Эти данные поступают для обработки на сервер аутентификации. В базе данных, хранящейся на сервере аутентификации, по идентификатору пользователя находится соответствующая запись, из нее извлекается пароль и сравнивается с тем паролем, который ввел пользователь. Если они совпали, то аутентификация прошла успешно, пользователь получает легальный статус, а также права и ресурсы сети, которые определены для его статуса системой авторизации.

В схеме простой аутентификации передача пароля и идентификатора пользователя может производиться следующими способами:

- в незашифрованном виде; например, согласно протоколу парольной аутентификации PAP (Password Authentication Protocol) пароли передаются по линии связи в открытой незащищенной форме;
- в защищенном виде; все передаваемые данные (идентификатор и пароль пользователя, случайное число и метки времени) защищены посредством шифрования или однонаправленной функции.

Схема простой аутентификации с использованием пароля показана на рис. 5.1.

Очевидно, что вариант аутентификации с передачей пароля пользователя в незашифрованном виде не гарантирует даже минимального уровня безопасности,

так как подвержен многочисленным атакам и легко компрометируется. Чтобы защитить пароль, его нужно зашифровать перед пересылкой по незащищенному каналу. Для этого в схему включены средства шифрования *EK* и дешифрования *DK*, управляемые разделяемым секретным ключом *K*. Проверка подлинности пользователя основана на сравнении присланного пользователем пароля *PA* и исходного значения *PA'*, хранящегося на сервере аутентификации. Если значения *PA* и *PA'* совпадают, то пароль *PA* считается подлинным, а пользователь *A* – законным.

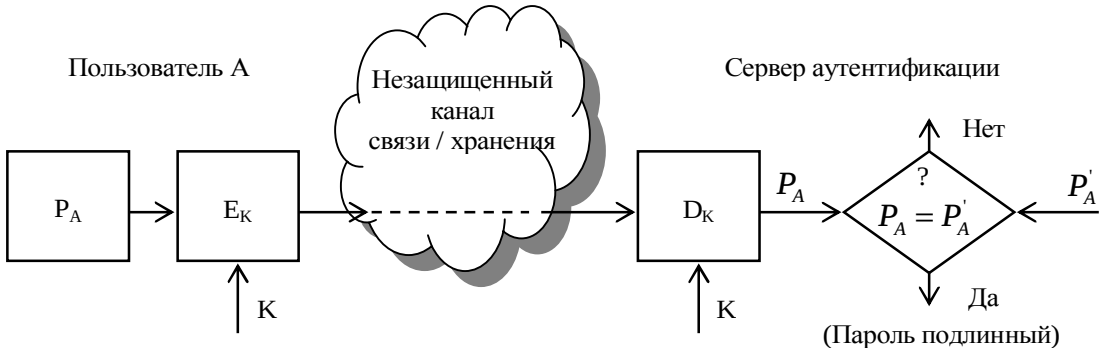


Рис. 5.1. Простая аутентификация с использованием пароля

Схемы организации простой аутентификации отличаются не только методами передачи паролей, но и видами их хранения и проверки. Наиболее распространенным способом является хранение паролей пользователей в открытом виде в системных файлах, причем на эти файлы устанавливаются атрибуты защиты от чтения и записи (например, при помощи описания соответствующих привилегий в списках контроля доступа операционной системы). Система сопоставляет введенный пользователем пароль с хранящейся в файле паролей записью. При этом способе не используются криптографические механизмы, такие как шифрование или однонаправленные функции. Очевидным недостатком данного способа является возможность получения злоумышленником в системе привилегий администратора, включая права доступа к системным файлам и, в частности, к файлу паролей.

Для обеспечения надежной защиты операционной системы пароль каждого пользователя должен быть известен только этому пользователю и никому другому, в том числе и администраторам системы. На первый взгляд то, что администратор знает пароль некоторого пользователя, не отражается негативно на безопасности системы, поскольку администратор, войдя в систему от имени обычного пользователя, получает права, меньшие, чем те, которые он получит, зайдя в систему от своего имени. Однако, входя в систему от имени другого пользователя, администратор получает возможность обходить систему аудита, а также совершать действия, компрометирующие этого пользователя, что недопустимо в защищенной системе. Таким образом, пароли пользователей не должны храниться в операционной системе в открытом виде.

С точки зрения безопасности предпочтительным является метод передачи и хранения паролей с использованием односторонних функций. Обычно для шифрования паролей в списке пользователей используют одну из известных криптографически стойких хэш-функций. В списке пользователей хранится не сам пароль, а образ пароля, являющийся результатом применения к паролю хэш-функции.

Однонаправленность хэш-функции не позволяет восстановить пароль по образу пароля, но дает возможность, вычислив хэш-функцию, получить образ введенного пользователем пароля и таким образом проверить правильность введенного пароля.

В простейшем случае в качестве хэш-функции используется результат шифрования некоторой константы на пароле.

Например, односторонняя функция h(·) может быть определена следующим образом:

$$h(P) = E_P(ID),$$

где P - пароль пользователя;

ID - идентификатор пользователя;

E_P - процедура шифрования, выполняемая с использованием пароля P в качестве ключа.

Такие функции удобны, если длина пароля и ключа одинакова. В этом случае проверка подлинности пользователя A с помощью пароля P_A состоит из пересылки серверу аутентификации отображения $h(P_A)$ и сравнения его с предварительно вычисленным и хранимым в базе данных сервера аутентификации эквивалентом $h'(P_A)$ – рис. 5.2. Если отображения $h(P_A)$ и $h'(P_A)$ равны, то считается, что пользователь успешно прошел аутентификацию.

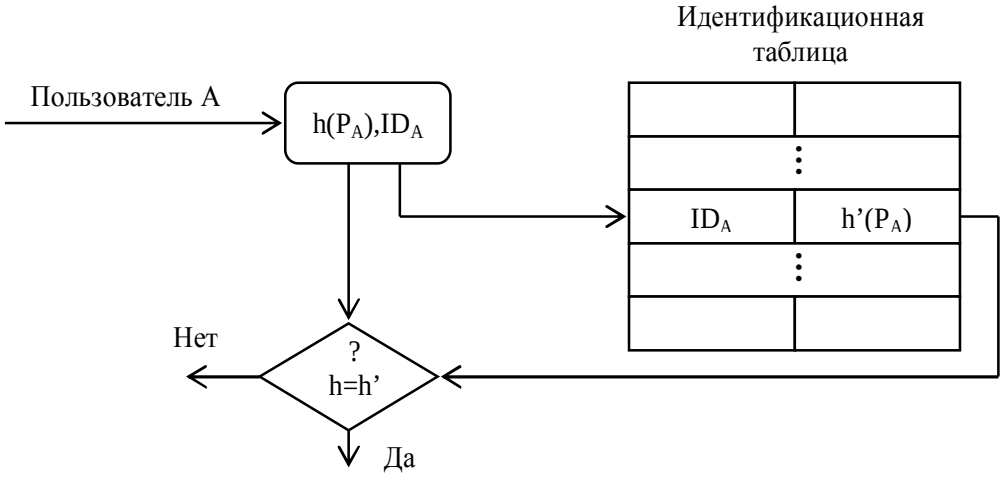


Рис. 5.2. Использование односторонней функции для проверки пароля

На практике пароли состоят лишь из нескольких символов, чтобы дать возможность пользователям запомнить их. Короткие пароли уязвимы к атаке полного перебора всех вариантов. Для того чтобы предотвратить такую атаку, функцию h(P) можно определить иначе, например в следующем виде:

$$h(P) = E_{P \oplus K}(ID),$$

где K и ID – соответственно ключ и идентификатор отправителя.

Различают две формы представления объектов, аутентифицирующих пользователя:

- внешний аутентифицирующий объект, не принадлежащий системе;
- внутренний объект, принадлежащий системе, в который переносится информация из внешнего объекта.

Внешние объекты могут быть представлены на различных носителях информации – пластиковых картах, смарт-картах, гибких магнитных дисках и т.п. Естественно, что внешняя и внутренняя формы представления аутентифицирующего объекта должны быть семантически тождественны.

Допустим, что в компьютерной системе зарегистрировано n пользователей. Пусть i-й аутентифицирующий объект i-го пользователя содержит два информационных поля:

- ID_i – неизменный идентификатор i-го пользователя, который является аналогом имени и используется для идентификации пользователя;
- K_i – аутентифицирующая информация пользователя, которая может изменяться и используется для аутентификации (например, пароль $P_i = K_i$).

Описанная структура соответствует практически любому ключевому носителю информации, используемому для опознания пользователя. Например, для носителей типа пластиковых карт выделяется неизменяемая информация ID_i первичной персонализации пользователя и объект в файловой структуре карты, содержащий K_i.

Совокупную информацию в ключевом носителе можно назвать первичной аутентифицирующей информацией i-го пользователя. Очевидно, что внутренний аутентифицирующий объект не должен существовать в системе длительное время (больше времени работы конкретного пользователя). Для длительного хранения следует использовать данные в защищенной форме.

Рассмотрим две типовые схемы идентификации и аутентификации.

Схема 1. В компьютерной системе выделяется объект-эталон для идентификации и аутентификации пользователей. Структура объекта-эталона для схемы 1 показана в табл. 5.1.

Таблица 5.1. Структура объекта-эталона

Номер пользователя	Информация для идентификации	Информация для аутентификации
1	ID_1	E_1
2	ID_2	E_2
…	…	…
N	ID_n	E_n

Здесь $E_i = F(ID_i, K_i)$,

где F – функция, которая обладает свойством «невосстановимости» значения K_i по E_i и ID_i.

«Невосстановимость» K_i оценивается некоторой пороговой трудоемкостью T_R решения задачи восстановления аутентифицирующей информации K_i по E_i и ID_i. Кроме того, для пары K_i и K_j возможно совпадение соответствующих значений E. В связи с этим вероятность ложной аутентификации пользователя не должна быть больше некоторого порогового значения P_R. На практике задают значения $T_R = 10^{20}$–10^{30}, $P_R = 10^{-7}$–10^{-9}.

Протокол идентификации и аутентификации (для схемы 1):

1) Пользователь предъявляет системе свой идентификатор ID.
2) Система сверяет предъявленный ID с зарегистрированными ID_i, i = 1..n. Если ID не совпадает ни с одним ID_i, зарегистрированным в системе, то

идентификация отвергается – пользователь не допускается к работе. Если *ID* = ID_i, то считается, что пользователь, назвавшийся пользователем *i*, прошел идентификацию.

3) Система запрашивает у пользователя его аутентификатор.
4) Пользователь предъявляет системе аутентификатор *K*.
5) Система вычисляет значение $Y = F(ID_i, K)$ и сравнивает значения *Y* и E_i.

При совпадении этих значений устанавливается, что данный пользователь успешно прошел аутентификацию в системе. Информация об этом пользователе передается в программные модули, использующие ключи пользователей (то есть в систему шифрования, разграничения доступа и др.). В противном случае аутентификация отвергается – пользователь не допускается к работе.

Данная схема идентификации и аутентификации пользователя может быть модифицирована.

Схема 2. В компьютерной системе выделяется модифицированный объект-эталон, структура которого показана в табл. 5.2.

Таблица 5.2. Структура модифицированного объекта-эталона

Номер пользователя	Информация для идентификации	Информация для аутентификации
1	ID_1, S_1	E_1
2	ID_2, S_2	E_2
…	…	…
N	ID_n, S_n	E_n

В отличие от схемы 1, в схеме 2 значение

$$E_i = F(S_i, K_i),$$

где S_i – случайный вектор, задаваемый при создании идентификатора пользователя, то есть при создании строки, необходимой для идентификации и аутентификации пользователя;

F – функция, которая обладает свойством «невосстановимости» значения K_i по E_i и S_i.

Протокол идентификации и аутентификации (для схемы 2):

1) Пользователь предъявляет системе свой идентификатор *ID*.
2) Если *ID* не совпадает ни с одним ID_i, зарегистрированным в компьютерной системе, то идентификация отвергается – пользователь не допускается к работе. Если существует $ID_i = ID$, тогда устанавливается, что пользователь, назвавшийся пользователем i, прошел идентификацию.
3) По идентификатору ID_i выделяется вектор S_i.
4) Система запрашивает у пользователя аутентификатор.
5) Пользователь предъявляет системе аутентификатор *K*.
6) Система вычисляет значение $Y = F(S_i, K)$ и сравнивает значения *Y* и E_i.

При совпадении этих значений устанавливается, что данный пользователь успешно прошел аутентификацию в системе. В противном случае аутентификация отвергается и пользователь не допускается к работе.

Вторая схема аутентификации применяется в *ОС UNIX*. В качестве идентификатора *ID* используется имя пользователя (запрошенное по *Login*), в качестве аутентификатора K_i – пароль пользователя (запрошенный по *Password*),

функция *F* представляет собой алгоритм шифрования *DES*. Эталоны для идентификации и аутентификации содержатся в файле Etc/passwd.

Системы простой аутентификации на основе многоразовых паролей имеют пониженную стойкость, поскольку в них выбор аутентифицирующей информации происходит из относительно небольшого множества слов. Срок действия многоразового пароля должен быть определен в политике безопасности организации, и такие пароли необходимо регулярно изменять. Выбирать пароли нужно так, чтобы они были трудны для угадывания и не присутствовали в словаре.

5.2.2. АУТЕНТИФИКАЦИЯ НА ОСНОВЕ ОДНОРАЗОВЫХ ПАРОЛЕЙ

Схемы аутентификации, основанные на традиционных многоразовых паролях, не обладают достаточной безопасностью. Такие пароли можно перехватить, разгадать, подсмотреть или просто украсть. Более надежными являются процедуры аутентификации на основе одноразовых паролей.

Суть схемы одноразовых паролей – использование различных паролей при каждом новом запросе на предоставление доступа. Одноразовый динамический пароль действителен только для одного входа в систему, и затем его действие истекает. Даже если кто-то перехватил его, пароль окажется бесполезен. Динамический механизм задания пароля является одним из лучших способов защитить процесс аутентификации от угроз извне. Обычно системы аутентификации с одноразовыми паролями используются для проверки удаленных пользователей.

Известны следующие методы применения одноразовых паролей для аутентификации пользователей:

1) Использование механизма временных меток на основе системы единого времени.
2) Использование списка случайных паролей, общего для легального пользователя и проверяющего, и надежного механизма их синхронизации.
3) Использование генератора псевдослучайных чисел, общего для пользователя и проверяющего, с одним и тем же начальным значением.

Генерация одноразовых паролей может осуществляться аппаратным или программным способом. Некоторые аппаратные средства доступа на основе одноразовых паролей реализуются в виде миниатюрных устройств со встроенным микропроцессором, внешне похожих на платежные пластиковые карточки. Такие карты, обычно называемые ключами, могут иметь клавиатуру и небольшой дисплей.

В качестве примера реализации первого метода рассмотрим технологию аутентификации *SecurID* на основе одноразовых паролей с использованием аппаратных ключей и механизма временной синхронизации. Эта технология аутентификации разработана компанией Security Dynamics и реализована в коммуникационных серверах ряда компаний, в частности в серверах компании Cisco Systems и др.

Схема аутентификации с использованием временной синхронизации базируется на алгоритме генерации случайных чисел через определенный интервал времени. Этот интервал устанавливается и может быть изменен администратором сети. Схема аутентификации использует два параметра:

– секретный ключ, представляющий собой уникальное 64-битовое число, назначаемое каждому пользователю и хранящееся в базе данных аутентификационного сервера и в аппаратном ключе пользователя;

– значение текущего времени.

Когда удаленный пользователь делает попытку логического входа в сеть, ему предлагается ввести его персональный идентификационный номер *PIN*, состоящий из четырех десятичных цифр, а также шесть цифр случайного числа, отображаемого в этот момент на дисплее аппаратного ключа. Используя введенный пользователем PIN-код, сервер извлекает из базы данных секретный ключ пользователя и выполняет алгоритм генерации случайного числа, используя в качестве параметров извлеченный секретный ключ и значение текущего времени. Затем сервер проверяет, совпадают ли сгенерированное число и число, введенное пользователем. Если эти числа совпадают, то сервер разрешает пользователю осуществить логический вход в систему.

При использовании этой схемы аутентификации, естественно, требуется жесткая временная синхронизация аппаратного ключа и сервера. Поскольку аппаратный ключ может работать несколько лет, вполне возможно постепенное рассогласование внутренних часов сервера и аппаратного ключа. Для решения этой проблемы компания Security Dynamics применяет два способа:

– при производстве аппаратного ключа точно измеряется отклонение частоты его таймера от номинала. Величина этого отклонения учитывается как параметр алгоритма сервера;
– сервер отслеживает коды, генерируемые конкретным аппаратным ключом, и при необходимости динамически подстраивается под этот ключ.

Со схемой аутентификации, основанной на временной синхронизации, связана еще одна проблема. Генерируемое аппаратным ключом случайное число является достоверным паролем в течение небольшого конечного промежутка времени. Поэтому, в принципе, возможна кратковременная ситуация, когда хакер может перехватить PIN-код и случайное число, чтобы использовать их для доступа в сеть. Это самое уязвимое место схемы аутентификации, основанной на временной синхронизации.

Существуют и другие варианты аппаратной реализации процедуры аутентификации с использованием одноразовых паролей, например аутентификация по схеме запрос–ответ. При попытке пользователя осуществить логический вход в сеть, аутентификационный сервер передает ему запрос в виде случайного числа. Аппаратный ключ пользователя зашифровывает это случайное число, используя, например, алгоритм DES и секретный ключ пользователя, хранящийся в памяти аппаратного ключа и в базе данных сервера. Случайное число-запрос возвращается в зашифрованном виде на сервер. Сервер, в свою очередь, также зашифровывает сгенерированное им самим случайное число с помощью того же алгоритма DES и того же секретного ключа пользователя, извлеченного из базы данных сервера. Затем сервер сравнивает результат своего шифрования с числом, пришедшим от аппаратного ключа. При совпадении этих чисел пользователь получает разрешение на вход в сеть. Следует отметить, что схема аутентификации запрос-ответ сложнее в использовании по сравнению со схемой аутентификации с временной синхронизацией.

Второй метод применения одноразовых паролей для аутентификации пользователей основан на использовании списка случайных паролей, общего для пользователя и проверяющего, и надежного механизма их синхронизации. Разделяемый список одноразовых паролей представляется в виде

последовательности или набора секретных паролей, где каждый пароль употребляется только один раз. Данный список должен быть заранее распределен между сторонами аутентификационного обмена. Вариантом данного метода является использование таблицы запросов-ответов, в которой содержатся запросы и ответы, используемые сторонами для проведения аутентификации, причем каждая пара должна применяться только один раз.

Третий метод применения одноразовых паролей для аутентификации пользователей основан на использовании генератора псевдослучайных чисел, общего для пользователя и проверяющего, с одним и тем же начальным значением. Известны следующие варианты реализации этого метода:

- последовательность преобразуемых одноразовых паролей. В ходе очередной сессии аутентификации пользователь создает и передает пароль именно для данной сессии, зашифрованный на секретном ключе, полученном из пароля предыдущей сессии;
- последовательности паролей, основанные на односторонней функции. Суть данного метода составляет последовательное использование односторонней функции (известная схема Лампорта). Этот метод является более предпочтительным с точки зрения безопасности по сравнению с методом последовательно преобразуемых паролей.

Одним из наиболее распространенных протоколов аутентификации на основе одноразовых паролей является стандартизованный в Интернете протокол S/Key (RFC 1760). Данный протокол реализован во многих системах, требующих проверки подлинности удаленных пользователей, в частности в системе TACACS+ компании Cisco.

5.2.3. АУТЕНТИФИКАЦИЯ НА ОСНОВЕ PIN-КОДА

Наиболее распространенным методом аутентификации держателя пластиковой карты и смарт-карты является ввод секретного числа, которое обычно называют PIN-кодом (Personal Identification Number - персональный идентификационный код) или иногда CHV (CardHolder Verification). Защита PIN-кода карты является критичной для безопасности всей системы. Карты могут быть потеряны, украдены или подделаны. В таких случаях единственной контрмерой против несанкционированного доступа остается секретное значение PIN-кода. Вот почему открытая форма PIN должна быть известна только законному держателю карты. Очевидно, значение PIN нужно держать в секрете в течение всего срока действия карты.

Длина PIN-кода должна быть достаточно большой, чтобы минимизировать вероятность определения правильного PIN-кода методом проб и ошибок. С другой стороны, длина PIN-кода должна быть достаточно короткой, чтобы дать возможность держателям карт запомнить его значение. Согласно рекомендации стандарта ISO 9564-1, PIN-код должен содержать от четырех до двенадцати буквенно-цифровых символов. Однако в большинстве случаев ввод нецифровых символов технически невозможен, поскольку доступна только цифровая клавиатура. Поэтому обычно PIN-код представляет собой четырехразрядное число, каждая цифра которого может принимать значение от 0 до 9.

PIN-код вводится с помощью клавиатуры терминала или компьютера и затем отправляется на смарт-карту. Смарт-карта сравнивает полученное значение PIN-

кода с эталонным значением, хранимым в карте, и отправляет результат сравнения на терминал. Ввод PIN-кода относится к мерам безопасности, особенно для финансовых транзакций, и следовательно, требования к клавиатуре часто определяются в этой прикладной области. PIN-клавиатуры имеют все признаки модуля безопасности, и они шифруют PIN-код сразу при его вводе. Это обеспечивает надежную защиту против проникновения в клавиатуру для того, чтобы перехватить PIN-код в то время, когда он вводится.

Различают статические и изменяемые PIN-коды. Статический PIN-код не может быть изменен пользователем, поэтому пользователь должен надежно его хранить. Если он станет известен постороннему, пользователь должен уничтожить карту и получить новую карту с другим фиксированным PIN-кодом.

Изменяемый PIN-код может быть изменен согласно пожеланиям пользователя или заменен на число, которое пользователю легче запомнить. Однако при таком подходе возникает опасность быстрого раскрытия подобного PIN-кода, поскольку числа, которые большинство людей считают удобными для запоминания, являются тривиальными числами вида 1234, 4321, 5115 и т.п. Смарт-карта обычно не проверяет употребление таких тривиальных чисел, поскольку для хранения необходимой таблицы не хватает доступной памяти. Однако терминал может воспрепятствовать замене PIN на такое число.

Некоторые приложения используют также транспортные PIN-коды. Смарт-карта персонализируется со случайным значением PIN-кода, и держатель карты получает значение PIN-кода в заказном письме. Однако, начиная применять карту, пользователь должен заменить PIN-код, использованный при персонализации карты, на выбранный им самим. Такая процедура исключает возможность того, что PIN-код, выслеженный во время персонализации, позже может быть незаконно использован.

Вероятность угадывания PIN-кода. Простейшей атакой на PIN-код, помимо подглядывания через плечо за вводом его с клавиатуры, является угадывание его значения. Вероятность угадывания зависит от длины угадываемого PIN-кода, от составляющих его символов и от количества разрешенных попыток ввода.

Для оценки риска, связанного с использованием конкретного PIN-кода, могут быть использованы формулы вычисления вероятности угадывания.

Введем обозначения:

x - число возможных комбинаций PIN-кода;

m - число возможных символов на позиции;

n - число позиций в PIN-коде;

P - вероятность угадывания PIN-кода;

i - число попыток угадывания.

Тогда число возможных комбинаций PIN-кода определяется формулой

$$x = m^n.$$

Вероятность угадывания PIN-кода за i попыток определяется формулой

$$P = i / m^n.$$

Если PIN-код состоит из четырех десятичных цифр, то есть $n = 4$ и $m = 10$, тогда число возможных комбинаций PIN-кода равно $x = m^n = 10^4 = 10\ 000$, то есть злоумышленник, пытающийся угадать значение PIN-кода, оказывается перед проблемой выбора одной из десяти тысяч комбинаций.

Если число разрешенных попыток ввода i = 3, тогда вероятность угадывания правильного значения PIN-кода из четырех десятичных цифр за три попытки ввода составляет $P = i / m^n = 3/10^4 = 0.00003$ или 0,03%.

Спецификации PC/SC рекомендуют, чтобы в смарт-картах были установлены ограничения на число неверных попыток ввода PIN-кода. Когда число обнаруженных неверных попыток достигает заданного предела, процесс ввода должен быть заблокирован, препятствуя дальнейшим попыткам аутентификации. Рекомендуется устанавливать допускаемое число неверных попыток в диапазоне от 1 до 255. Метод, используемый для разблокирования процесса ввода, должен быть защищен независимым механизмом аутентификации.

Генерация PIN-кода. Для генерации PIN-кода смарт-карты используются генератор случайных чисел и алгоритм, который преобразует случайное число в PIN-код необходимой длины. Затем можно использовать таблицу известных тривиальных комбинаций, чтобы распознать и отбросить значение PIN-кода, совпадающее с одной из таких комбинаций. Наконец этот PIN-код записывается в смарт-карту в виде соответствующей криптограммы. Вычисленное значение PIN-кода передается также держателю смарт-карты через защищенный канал.

В настоящее время все дебетовые карты (такие, как карты Eurocheque) из соображений совместимости имеют магнитные полосы, даже если они оснащены микроконтроллерами. Для гибридных карт с чипом и магнитной полосой генерация PIN-кода несколько усложняется. Это обусловлено тем, что банкомат или кассир-автомат, работающий в автономном режиме, должен иметь возможность проверить введенный PIN-код, основанный на данных, расположенных на магнитной полосе. Поэтому алгоритм генерации PIN-кода для гибридных карт с чипом и магнитной полосой должен быть детерминированным, то есть он должен всегда выдавать один и тот же результат для заданного набора входных величин. Генератор случайных чисел не позволяет это сделать. Соответственно, нужна процедура, которая может генерировать PIN-код, основанный на данных магнитной полосы. Чтобы избежать зависимости безопасности системы от самой процедуры генерации PIN-кода, в процесс вычисления должен быть также включен секретный ключ.

Схема алгоритма генерации PIN-кода с использованием симметричного шифра с секретным ключом показана на рис. 5.3.

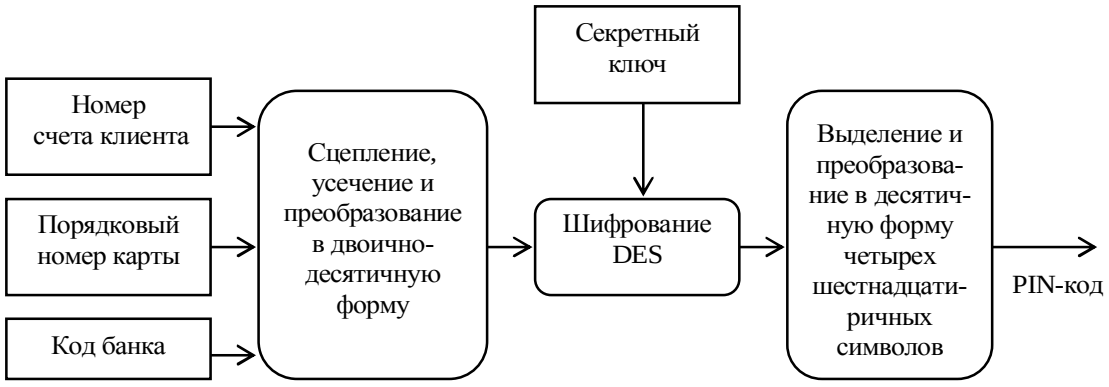


Рис. 5.3. Схема алгоритма генерации PIN-кода

Этот алгоритм генерирует четырехразрядный PIN-код. Входами алгоритма генерации являются три связанных с картой элемента данных (номер счета, порядковый номер карты, код банка). Над этими данными выполняются операции сцепления, усечения и преобразования в двоично-десятичную форму. Затем выполняется шифрование с использованием симметричного шифра 3-DES с секретным ключом. После выполнения процесса шифрования, выделяются четыре шестнадцатеричных символа и выполняется преобразование шестнадцатеричного числа в десятичное.

Главное требование безопасности использования PIN-кода состоит в том, что значение PIN-кода должно запоминаться держателем карты и его нельзя хранить в любой читаемой форме. Но память людей несовершенна, и часто они забывают значения своих PIN-кодов. Поэтому эмитенты карт должны иметь специальные процедуры для таких случаев. Эмитент может реализовать один из следующих подходов. Первый основан на восстановлении забытого клиентом значения PIN-кода и отправке его обратно владельцу карты. При втором подходе просто генерируется новое значение PIN-кода.

При идентификации клиента по значению PIN-кода и предъявленной карте используется два основных способа проверки PIN-кода: неалгоритмический и алгоритмический .

Неалгоритмический способ проверки PIN-кода не требует применения специальных алгоритмов. Проверка PIN-кода осуществляется путем непосредственного сравнения введенного клиентом PIN-кода со значениями, хранимыми в базе данных. Обычно база данных со значениями PIN-кодов клиентов шифруется методом прозрачного шифрования, чтобы повысить ее защищенность, не усложняя процесса сравнения.

Алгоритмический способ проверки PIN-кода заключается в том, что введенный клиентом PIN-код преобразуют по определенному алгоритму с использованием секретного ключа и затем сравнивают со значением PIN-кода, хранящимся в определенной форме на карте. Достоинства этого метода проверки:

- отсутствие копии PIN-кода на главном компьютере исключает его раскрытие обслуживающим персоналом;
- отсутствие передачи PIN-кода между банкоматом или кассиром-автоматом и главным компьютером банка исключает его перехват злоумышленником или навязывание результатов сравнения;
- упрощение работы по созданию программного обеспечения системы, так как уже нет необходимости действий в реальном масштабе времени.

5.3. СТРОГАЯ АУТЕНТИФИКАЦИЯ

Идея строгой аутентификации, реализуемая в криптографических протоколах, заключается в следующем. Проверяемая (доказывающая) сторона доказывает свою подлинность проверяющей стороне, демонстрируя знание некоторого секрета. Например, этот секрет может быть предварительно распределен безопасным способом между сторонами аутентификационного обмена. Доказательство знания секрета осуществляется с помощью последовательности запросов и ответов с использованием криптографических методов и средств.

Существенным является тот факт, что доказывающая сторона демонстрирует только знание секрета, но сам секрет в ходе аутентификационного обмена не

раскрывается. Это обеспечивается посредством ответов доказывающей стороны на различные запросы проверяющей стороны. При этом результирующий запрос зависит только от пользовательского секрета и начального запроса, который обычно представляет произвольно выбранное в начале протокола большое число.

В большинстве случаев строгая аутентификация заключается в том, что каждый пользователь аутентифицируется по признаку владения своим секретным ключом. Иначе говоря, пользователь имеет возможность определить, владеет ли его партнер по связи надлежащим секретным ключом и может ли он использовать этот ключ для подтверждения того, что он действительно является подлинным партнером по информационному обмену.

В соответствии с рекомендациями стандарта X.509 различают процедуры строгой аутентификации следующих типов:

- односторонняя аутентификация;
- двусторонняя аутентификация;
- трехсторонняя аутентификация.

Односторонняя аутентификация предусматривает обмен информацией только в одном направлении. Данный тип аутентификации позволяет:

- подтвердить подлинность только одной стороны информационного обмена;
- обнаружить нарушение целостности передаваемой информации;
- обнаружить проведение атаки типа «повтор передачи»;
- гарантировать, что передаваемыми аутентификационными данными может воспользоваться только проверяющая сторона;

Двусторонняя аутентификация по сравнению с односторонней содержит дополнительный ответ проверяющей стороны доказывающей стороне, который должен убедить ее, что связь устанавливается именно с той стороной, которой были предназначены аутентификационные данные.

Трехсторонняя аутентификация содержит дополнительную передачу данных от доказывающей стороны проверяющей. Этот подход позволяет отказаться от использования меток времени при проведении аутентификации.

Следует отметить, что данная классификация достаточно условна. Отмеченные особенности носят в большей степени теоретический характер. На практике набор используемых приемов и средств зависит непосредственно от конкретных условий реализации процесса аутентификации. Необходимо также учитывать, что проведение строгой аутентификации требует обязательного согласования сторонами используемых криптографических алгоритмов и ряда дополнительных параметров.

Прежде чем перейти к рассмотрению конкретных вариантов протоколов строгой аутентификации, следует остановиться на назначении и возможностях так называемых одноразовых параметров, используемых в протоколах аутентификации. Эти одноразовые параметры иногда называют *nonces*. По определению, *nonce* – это величина, используемая для одной и той же цели не более одного раза.

Среди используемых на сегодняшний день одноразовых параметров следует выделить случайные числа, метки времени и номера последовательностей.

Одноразовые параметры позволяют избежать повтора передачи, подмены стороны аутентификационного обмена и атаки с выбором открытого текста. При помощи одноразовых параметров можно обеспечить уникальность, однозначность и

временные гарантии передаваемых сообщений. Различные типы одноразовых параметров могут употребляться как отдельно, так и дополнять друг друга.

Можно привести следующие примеры применения одноразовых параметров:

- проверка своевременности в протоколах, построенных по принципу запрос-ответ. При такой проверке могут использоваться случайные числа, метки времени с синхронизацией часов или номера последовательностей для конкретной пары (проверяющий, доказывающий);
- обеспечение своевременности или гарантий уникальности. Осуществляется путем непосредственного контроля одноразовых параметров протокола (посредством выбора случайного числа) либо косвенно (путем анализа информации, содержащейся в разделяемом секрете);
- однозначная идентификация сообщения или последовательности сообщений. Осуществляется посредством выработки одноразового значения из монотонно возрастающей последовательности (например, последовательности серийных номеров или меток времени) или случайных чисел соответствующей длины.

Следует отметить, что одноразовые параметры широко используются и в других вариантах криптографических протоколов (например, в протоколах распределения ключевой информации).

В зависимости от используемых криптографических алгоритмов протоколы строгой аутентификации можно разделить на следующие группы:

- протоколы строгой аутентификации на основе симметричных алгоритмов шифрования;
- протоколы строгой аутентификации на основе однонаправленных ключевых хэш-функций;
- протоколы строгой аутентификации на основе асимметричных алгоритмов шифрования;
- протоколы строгой аутентификации на основе алгоритмов электронной цифровой подписи.

5.3.1. СТРОГАЯ АУТЕНТИФИКАЦИЯ, ОСНОВАННАЯ НА СИММЕТРИЧНЫХ АЛГОРИТМАХ

Для работы протоколов аутентификации, построенных на основе симметричных алгоритмов, необходимо, чтобы проверяющий и доказывающий с самого начала имели один и тот же секретный ключ. Для закрытых систем с небольшим количеством пользователей каждая пара пользователей может заранее разделить его между собой. В больших распределенных системах, применяющих технологию симметричного шифрования, часто используются протоколы аутентификации с участием доверенного сервера, с которым каждая сторона разделяет знание ключа. Такой сервер распределяет сеансовые ключи для каждой пары пользователей всякий раз, когда один из них запрашивает аутентификацию другого. Кажущаяся простота данного подхода является обманчивой, на самом деле разработка протоколов аутентификации этого типа является сложной и с точки зрения безопасности неочевидной.

Протоколы аутентификации с симметричными алгоритмами шифрования

Ниже приводится три примера отдельных протоколов аутентификации, специфицированных в ISO/IEC 9798-2. Эти протоколы предполагают предварительное распределение разделяемых секретных ключей. Рассмотрим следующие варианты аутентификации:

- односторонняя аутентификация с использованием меток времени;
- односторонняя аутентификация с использованием случайных чисел;
- двусторонняя аутентификация.

В каждом из этих случаев пользователь доказывает свою подлинность, демонстрируя знание секретного ключа, так как производит дешифрование запросов с помощью этого секретного ключа.

При использовании в процессе аутентификации симметричного шифрования необходимо также реализовать механизмы обеспечения целостности передаваемых данных на основе общепринятых способов.

Введем следующие обозначения:

r_A – случайное число, сгенерированное участником A;

r_B – случайное число, сгенерированное участником B;

t_A – метка времени, сгенерированная участником A;

E_K – симметричное шифрование на ключе K (ключ K должен быть предварительно распределен между A и B).

1) Односторонняя аутентификация, основанная на метках времени:
   $A \rightarrow B : E_K(t_A, B)$

   После получения и дешифрования данного сообщения участник B убеждается в том, что метка времени t_A действительна и идентификатор B, указанный в сообщении, совпадает с его собственным. Предотвращение повторной передачи данного сообщения основывается на том, что без знания ключа невозможно изменить метку времени t_A и идентификатор B.
2) Односторонняя аутентификация, основанная на использовании случайных чисел:

   $A \leftarrow B : r_B$,

   $A \rightarrow B : E_K(r_B, B)$.

   Участник B отправляет участнику A случайное число r_B. Участник A шифрует сообщение, состоящее из полученного числа r_B и идентификатора B, и отправляет зашифрованное сообщение участнику B. Участник B расшифровывает полученное сообщение и сравнивает случайное число, содержащееся в сообщении, с тем, которое он послал участнику A. Дополнительно он проверяет имя, указанное в сообщении.
3) Двусторонняя аутентификация, использующая случайные значения:

   $A \leftarrow B : r_B$,

   $A \rightarrow B : E_K(r_A, r_B, B)$,

   $A \leftarrow B : E_K(r_A, r_B)$.

   При получении второго сообщения участник B выполняет те же проверки, что и в предыдущем протоколе, и дополнительно расшифровывает случайное число r_A для включения его в третье сообщение для участника A. Третье сообщение, полученное участником A, позволяет ему убедиться на основе проверки значений r_A и r_B, что он имеет дело именно с участником B.

Широко известными представителями протоколов, обеспечивающих аутентификацию пользователей с привлечением в процессе аутентификации третьей стороны, являются протокол распределения секретных ключей Нидхэма и Шредера и протокол *Kerberos*.

Протоколы, основанные на использовании однонаправленных ключевых хэш-функций

Протоколы, представленные выше, могут быть модифицированы путем замены симметричного шифрования на шифрование с помощью односторонней ключевой хэш-функции. Это бывает необходимо, если алгоритмы блочного шифрования недоступны или не отвечают предъявляемым требованиям (например, в случае экспортных ограничений).

Своеобразие шифрования с помощью односторонней хэш-функции заключается в том, что оно, по существу, является односторонним, то есть не сопровождается обратным преобразованием – дешифрованием на принимающей стороне. Обе стороны (отправитель и получатель) используют одну и ту же процедуру одностороннего шифрования.

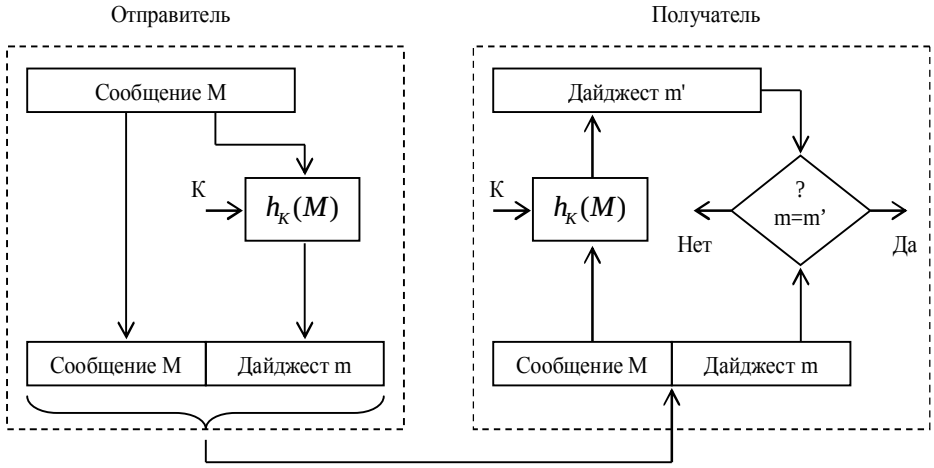


Рис. 5.4. Применение для аутентификации односторонней хэш-функции с параметром-ключом

Односторонняя хэш-функция $h_K(\cdot)$ с параметром-ключом K, примененная к шифруемым данным M, дает в результате хэш-значение m (дайджест), состоящее из фиксированного небольшого числа байтов (рис. 5.4).

Дайджест $m = h_K(M)$ передается получателю вместе с исходным сообщением M. Получатель сообщения, зная, какая односторонняя хэш-функция была применена для получения дайджеста, заново вычисляет ее, используя расшифрованное сообщение M. Если значения полученного дайджеста m и вычисленного дайджеста m' совпадают, значит, содержимое сообщения M не было подвергнуто никаким изменениям.

Знание дайджеста не дает возможности восстановить исходное сообщение, но позволяет проверить целостность данных. Дайджест можно рассматривать как своего рода контрольную сумму для исходного сообщения. Однако между дайджестом и обычной контрольной суммой имеется и существенное различие. Контрольную сумму используют как средство проверки целостности передаваемых сообщений по ненадежным линиям связи. Это средство проверки не рассчитано на

борьбу со злоумышленниками, которым в такой ситуации ничто не мешает подменить сообщение, добавив к нему новое значение контрольной суммы, Получатель в таком случае не заметит никакой подмены.

В отличие от обычной контрольной суммы, при вычислении дайджеста применяются секретные ключи. В случае если для получения дайджеста используется односторонняя хэш-функция с параметром-ключом *K*, который известен только отправителю и получателю, любая модификация исходного сообщения будет немедленно обнаружена.

На рис. 5.5 показан другой вариант использования односторонней хэш-функции для проверки целостности данных. В этом случае односторонняя хэш-

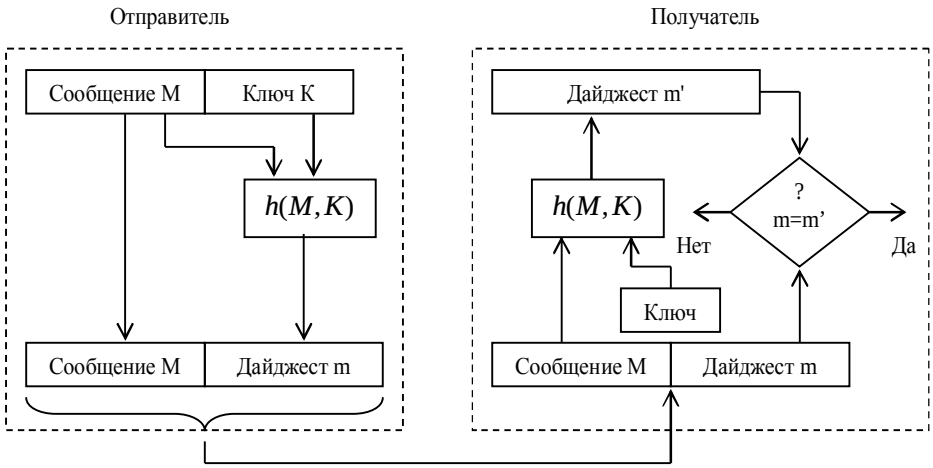


Рис. 5.5. Применение односторонней хэш-функции к сообщению, дополненному секретным ключом K

функция *h()* не имеет параметра-ключа, но зато применяется не просто к сообщению *M*, а к сообщению, дополненному секретным ключом *K*, то есть отправитель вычисляет дайджест $m = h(M, K)$. Получатель, извлекая исходное сообщение *M*, также дополняет его тем же известным ему секретным ключом *K*, после чего применяет к полученным данным одностороннюю хэш-функцию *h(.)*. Результат вычислений – дайджест *m'*, который сравнивается с полученным по сети дайджестом *m*.

При использовании для аутентификации односторонних функций шифрования в рассмотренные выше протоколы необходимо внести следующие изменения:

- функция симметричного шифрования E_k заменяется функцией h_k;
- проверяющий, вместо установления факта совпадения полей в расшифрованных сообщениях с предполагаемыми значениями, вычисляет значение однонаправленной функции и сравнивает его с полученным от другого участника обмена информацией;
- для обеспечения возможности независимого вычисления значения однонаправленной функции получателем сообщения в протоколе 1, метка времени t_A должна передаваться дополнительно в открытом виде, а в сообщении 2 протокола 3 случайное число r_A должно передаваться дополнительно в открытом виде.

Модифицированный вариант протокола 3 с учетом сформулированных изменений имеет структуру, изображенную на рис. 5.5.

$A \leftarrow B : r_B,$

$A \rightarrow B : r_A, h_K(r_A, r_B, B),$

$A \leftarrow B : h_K(r_A, r_B, A).$

Заметим, что в третье сообщение протокола включено поле A. Результирующий протокол обеспечивает взаимную аутентификацию и известен как протокол SKID 3.

5.3.2. СТРОГАЯ АУТЕНТИФИКАЦИЯ, ОСНОВАННАЯ НА АСИММЕТРИЧНЫХ АЛГОРИТМАХ

В протоколах строгой аутентификации могут быть использованы асимметричные алгоритмы с открытыми ключами. В этом случае доказывающий может продемонстрировать знание секретного ключа одним из следующих способов:

- расшифровать запрос, зашифрованный на открытом ключе;
- поставить свою цифровую подпись на запросе.

Пара ключей, необходимая для аутентификации, не должна использоваться для других целей (например, для шифрования) по соображениям безопасности. Следует также предостеречь потенциальных пользователей о том, что выбранная система с открытым ключом должна быть устойчивой к атакам с выборкой шифрованного текста даже в том случае, если нарушитель пытается получить критичную информацию, выдавая себя за проверяющего и действуя от его имени.

Аутентификация с использованием асимметричных алгоритмов шифрования

В качестве примера протокола, построенного на использовании асимметричного алгоритма шифрования, можно привести следующий протокол аутентификации:

$A \leftarrow B : h(r), B, P_A(r, B)$

$A \rightarrow B : r$

Участник B выбирает случайным образом r и вычисляет значение $x = h(r)$ (значение x демонстрирует знание r без раскрытия самого значения r), далее он вычисляет значение $e = P_A(r, B)$. Под P_A подразумевается алгоритм асимметричного шифрования (например, RSA), а под h() - хэш-функция. Участник B отправляет сообщение участнику А. Участник А расшифровывает $e = P_A(r, B)$ и получает значения r^1 и B^1, а также вычисляет $x^1 = h(r^1)$. После этого производится ряд сравнений, доказывающих, что $x = x^1$ и что полученный идентификатор B^1 действительно указывает на участника B. В случае успешного проведения сравнения участнике A посылает r. Получив его, участник B проверяет, то ли это значение, которое он отправил в первом сообщении.

В качестве следующего примера приведем модифицированный протокол Нидхэма и Шредера, основанный на асимметричном шифровании. Рассматривая вариант протокола Нидхэма и Шредера, используемый только для аутентификации, будем подразумевать под P_B алгоритм шифрования открытым ключом участника B. Протокол имеет следующую структуру:

$A \rightarrow B : P_B(r_1, A)$

$A \leftarrow B : P_A(r_2, r_1)$

$A \leftarrow B : r_2$

Аутентификация, основанная на использовании цифровой подписи

В рекомендациях стандарта X.509 специфицирована схема аутентификации, основанная на использовании цифровой подписи, меток времени и случайных чисел.

Для описания данной схемы аутентификации введем следующие обозначения:

t_A, r_A и r_B - временная метка и случайные числа соответственно;

S_A – подпись, сгенерированная участником A;

S_B – подпись, сгенерированная участником B;

$cert_A$ – сертификат открытого ключа участника A;

$cert_B$ - сертификат открытого ключа участника B.

Если участники имеют аутентичные открытые ключи, полученные друг от друга, тогда можно не пользоваться сертификатами, в противном случае они служат для подтверждения подлинности открытых ключей.

В качестве примеров приведем следующие протоколы аутентификации:

1) Односторонняя аутентификация с применением меток времени:

   $A \rightarrow B : cert_A, t_A, B, S_A(t_A, B)$

   После принятия данного сообщения участник B проверяет правильность метки времени t_A, полученный идентификатор B и, используя открытый ключ из сертификата $cert_A$, корректность цифровой подписи $S_A(t_A, B)$.

2) Односторонняя аутентификация с использованием случайных чисел:

   $A \leftarrow B : r_B$

   $A \rightarrow B : cert_A, r_A, B, S_A(r_A, r_B, B)$

Участник B, получив сообщение от участника A, убеждается, что именно он является адресатом сообщения; используя открытый ключ участника A, взятый из сертификата $cert_A$, проверяет корректность подписи $S_A(r_A,r_B,B)$ под числом r_A, полученным в открытом виде, числом r_B, которое было отослано в первом сообщении, и его идентификатором B. Подписанное случайное число r_A используется для предотвращения атак с выборкой открытого текста.

3) Двусторонняя аутентификация с использованием случайных чисел:

   $A \leftarrow B : r_B$

   $A \rightarrow B : cert_A, r_A, B, S_A(r_A, r_B, B)$

   $A \leftarrow B : cert_B, A, S_B(r_A, r_B, A)$

   В данном протоколе обработка сообщений 1 и 2 выполняется так же, как и в предыдущем протоколе, а сообщение 3 обрабатывается аналогично сообщению 2.

5.4. БИОМЕТРИЧЕСКАЯ АУТЕНТИФИКАЦИЯ ПОЛЬЗОВАТЕЛЯ

Процедуры идентификации и аутентификации пользователя могут базироваться не только на секретной информации, которой обладает пользователь (пароль, персональный идентификатор, секретный ключ и т.п.). Привычные системы аутентификации не всегда удовлетворяют современным требованиям в области информационной безопасности, особенно если речь идет об ответственных

приложениях (онлайновые финансовые приложения, доступ к удаленным базам данных и т.п.).

В последнее время все большее распространение получает биометрическая аутентификация пользователя, позволяющая уверенно аутентифицировать потенциального пользователя путем измерения физиологических параметров и характеристик человека, особенностей его поведения. Использование решений, основанных на биометрической технологии, позволяет в ряде случаев улучшить положение дел в области аутентификации.

Для методов аутентификации, основанных на использовании многоразовых паролей, характерен следующий недостаток: многоразовый пароль может быть скомпрометирован множеством способов. Недостатком методов, связанных с использованием токенов (токен – компактное устройство в виде *USB*-брелока, которое служит для авторизации пользователя), является возможность потери, кражи, дублирования токенов – носителей критической информации. Биометрические методы, использующие для идентификации уникальные характеристики пользователя, свободны от перечисленных недостатков.

Отметим основные достоинства биометрических методов аутентификации пользователя по сравнению с традиционными:

- высокая степень достоверности аутентификации по биометрическим признакам из-за их уникальности;
- неотделимость биометрических признаков от дееспособной личности;
- трудность фальсификации биометрических признаков.

В качестве биометрических признаков, которые активно используются при аутентификации потенциального пользователя, можно выделить следующие:

- отпечатки пальцев;
- геометрическая форма кисти руки;
- форма и размеры лица;
- особенности голоса;
- узор радужной оболочки и сетчатки глаз.

Рассмотрим типичную схему функционирования биометрической подсистемы аутентификации. При регистрации в системе пользователь должен продемонстрировать один или несколько раз свои характерные биометрические признаки. Эти признаки (известные как подлинные) регистрируются системой как «контрольный образ» (биометрическая подпись) законного пользователя. Этот образ пользователя хранится системой в электронной форме и используется для проверки идентичности каждого, кто выдает себя за соответствующего законного пользователя. В зависимости от совпадения или несовпадения совокупности предъявленных признаков с зарегистрированными в контрольном образе их предъявивший признается законным пользователем (при совпадении) или нет (при несовпадении).

С точки зрения потребителя, эффективность биометрической аутентификационной системы характеризуется двумя параметрами:

- коэффициентом ошибочных отказов FRR (false-reject rate);
- коэффициентом ошибочных подтверждений FAR (false-alarm rate).

Ошибочный отказ возникает тогда, когда система не подтверждает личность законного пользователя (типичные значения *FRR* составляют порядка одной ошибки на 100). Ошибочное подтверждение происходит в случае подтверждения

личности незаконного пользователя (типичные значения *FAR* составляют порядка одной ошибки на 10000). Коэффициент ошибочных отказов и коэффициент ошибочных подтверждений связаны друг с другом; каждому коэффициенту ошибочных отказов соответствует определенный коэффициент ошибочных подтверждений.

В совершенной биометрической системе оба параметра ошибки должны быть равны нулю. К сожалению, биометрические системы не идеальны, поэтому приходится чем-то пожертвовать. Обычно системные параметры настраивают так, чтобы добиться требуемого коэффициента ошибочных подтверждений, что определяет соответствующий коэффициент ошибочных отказов.

К настоящему времени разработаны и продолжают совершенствоваться технологии аутентификации по отпечаткам пальцев, радужной оболочке глаза, по форме кисти руки и ладони, по форме и размеру лица, по голосу и «клавиатурному почерку».

Наибольшее число биометрических систем в качестве параметра идентификации использует отпечатки пальцев (дактилоскопические системы аутентификации). Такие системы просты и удобны, обладают высокой надежностью аутентификации.

Дактилоскопические системы аутентификации. Одной из основных причин широкого распространения таких систем является наличие больших банков данных по отпечаткам пальцев. Основными пользователями подобных систем во всем мире являются полиция, различные государственные и некоторые банковские организации.

В общем случае биометрическая технология распознавания отпечатков пальцев заменяет защиту доступа с использованием пароля. Большинство систем использует отпечаток одного пальца, который пользователь предоставляет системе.

Основными элементами дактилоскопической системы аутентификации являются:

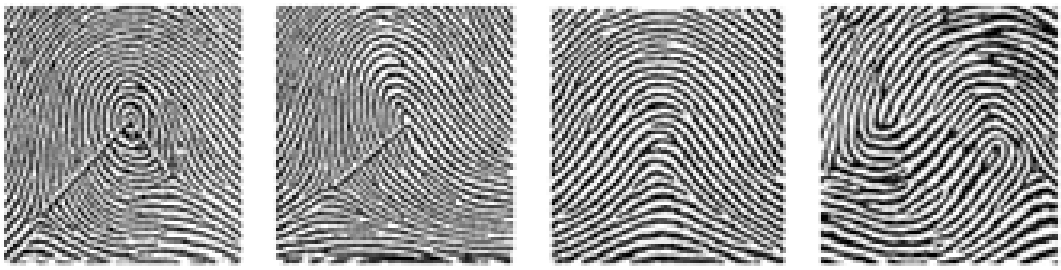
- сканер;
- ПО идентификации формирующее идентификатор пользователя;
- ПО аутентификации, производящее сравнение отсканированного отпечатка пальца с имеющимися в базе данных «паспортами» пользователей.

Дактилоскопическая система аутентификации работает следующим образом. Сначала производится регистрация пользователя. Как правило, производится несколько вариантов сканирования в разных положениях пальца на сканере. Понятно, что образцы будут немного отличаться и требуется сформировать некоторый обобщенный образец, «паспорт». Результаты сохраняются в базе данных аутентификации. При аутентификации производится сравнение отсканированного отпечатка пальца с «паспортами», хранящимися в базе данных.

Задача формирования «паспорта», так же как и распознавания предъявляемого образца, является задачей распознавания образов. Для этого используются различные алгоритмы, являющиеся ноу-хау фирм-производителей подобных устройств. Обычно «паспортом» выступает не само изображение отпечатка пальца, а результат разложения его на такие составляющие элементы как «завиток», «дуга», «петля» и др., рис. 5.6.

Сканеры отпечатков пальцев. Многие производители все чаще переходят от дактилоскопического оборудования на базе оптики к продуктам, основанным на интегральных схемах.

Продукты на базе интегральных схем имеют значительно меньшие размеры, чем оптические считыватели, и поэтому их проще реализовать в широком спектре периферийных устройств.



завиток петля дуга сложный

Рис. 5.6. Основные элементы отпечатка пальцев

Ряд производителей комбинируют биометрические системы со смарт-картами и картами-ключами. Например, в биометрической идентификационной смарт-карте *Authentic* реализован следующий подход. Образец отпечатка пальца пользователя сохраняется в памяти карты в процессе внесения в списки идентификаторов пользователей, устанавливая соответствие между образцом и личным ключом шифрования. Затем, когда пользователь вводит смарт-карту в считыватель и прикладывает палец к сенсору, ключ удостоверяет его личность. Комбинация биометрических устройств и смарт-карт является удачным решением, повышающим надежность процессов аутентификации и авторизации.

Небольшой размер и невысокая цена датчиков отпечатков пальцев на базе интегральных схем превращает их в идеальный пользовательский интерфейс для систем защиты. Их можно будет встраивать в брелок для ключей, и пользователи получат универсальный ключ, который обеспечит защищенный доступ ко всему, начиная от компьютеров и заканчивая входными дверьми, дверцами автомобилей и банкоматами.

Системы аутентификации по форме ладони используют сканеры формы ладони, обычно устанавливаемые на стенах. Следует отметить, что подавляющее большинство пользователей предпочитают системы этого типа.

Устройства считывания формы ладони создают объемное изображение ладони, измеряя длину пальцев, толщину и площадь поверхности ладони. Например, продукты компании Recognition Systems выполняют более 90 измерений, которые преобразуются в девятиразрядный образец для дальнейших сравнений. Этот образец может быть сохранен локально, на индивидуальном сканере ладони, либо в централизованной базе данных.

По уровню доходов, устройства сканирования формы ладони занимают второе место среди биометрических устройств, однако редко применяются в сетевой среде из-за высокой стоимости и размера. Однако сканеры формы ладони хорошо подходят для вычислительных сред со строгим режимом безопасности и напряженным трафиком, включая серверные комнаты. Они достаточно точны и

обладают довольно низким коэффициентом ошибочного отказа FRR, то есть процентом отклоненных законных пользователей.

Системы аутентификации по лицу и голосу являются наиболее доступными из-за их дешевизны, поскольку большинство современных компьютеров имеет видео- и аудиосредства. Системы данного класса применяются при удаленной идентификации субъекта доступа в телекоммуникационных сетях.

Технология сканирования черт лица подходит для тех приложений, где прочие биометрические технологии непригодны. В этом случае для идентификации и верификации личности используются особенности глаз, носа и губ. Производители устройств распознавания черт лица используют собственные математические алгоритмы для идентификации пользователей

Исследования, проводимые компанией International Biometric Group, говорят о том, что сотрудники многих организаций не доверяют устройствам распознавания по чертам лица отчасти из-за того, что камера их фотографирует, а затем выводит снимки на экран монитора; при этом многие опасаются, что используемая камера низкого качества. Кроме того, по данным этой компании, сканирование черт лица – единственный метод биометрической аутентификации, который не требует согласия на выполнение проверки (и может осуществляться скрытой камерой), а потому имеет негативный для пользователей подтекст.

Следует отметить, что технологии распознавания черт лица требуют дальнейшего совершенствования. Большая часть алгоритмов распознавания черт лица чувствительна к колебаниям в освещении, вызванным изменением интенсивности солнечного света в течение дня. Изменение положения лица также может повлиять на узнаваемость. Различие в положении в 15% между запрашиваемым изображением и изображением, которое находится в базе данных, напрямую сказывается на эффективности. При различии в 45° распознавание становится неэффективным.

Системы аутентификации по голосу экономически выгодны по тем же причинам, что и системы распознавания по чертам лица. В частности, их можно устанавливать с оборудованием (например, микрофонами), поставляемым в стандартной комплектации со многими ПК.

Системы аутентификации по голосу при записи образца и в процессе последующей идентификации опираются на такие уникальные для каждого человека особенности голоса, как высота, модуляция и частота звука. Эти показатели определяются физическими характеристиками голосового тракта и уникальны для каждого человека. Распознавание голоса уже применяется вместо набора номера в определенных системах *Sprint*. Такой вид распознавания голоса отличается от распознавания речи. В то время как технология распознавания речи интерпретирует то, что говорит абонент, технология распознавания голоса абонента подтверждает личность говорящего.

Поскольку голос можно просто записать на пленку или другие носители, некоторые производители встраивают в свои продукты операцию запроса отклика. Эта функция предлагает пользователю при входе ответить на предварительно подготовленный и регулярно меняющийся запрос, например такой: «Повторите числа 0, 1, 3».

Оборудование аутентификации по голосу более пригодно для интеграции в приложения телефонии, чем для входа в сеть. Обычно оно позволяет абонентам получить доступ в финансовые или прочие системы посредством телефонной связи.

Технологии распознавания говорящего имеют некоторые ограничения. Различные люди могут говорить похожими голосами, а голос любого человека может меняться со временем в зависимости от самочувствия, эмоционального состояния и возраста. Более того, разница в модификации телефонных аппаратов и качество телефонных соединений могут серьезно усложнить распознавание.

Поскольку голос сам по себе не обеспечивает достаточной точности, распознавание по голосу следует сочетать с другими биометриками, такими как распознавание черт лица или отпечатков пальцев.

Системы аутентификации по узору радужной оболочки и сетчатки глаз могут быть разделены на два класса:

- использующие рисунок радужной оболочки глаза;
- использующие рисунок кровеносных сосудов сетчатки глаза.

Сетчатка человеческого глаза представляет собой уникальный объект для аутентификации. Рисунок кровеносных сосудов глазного дна отличается даже у близнецов. Поскольку вероятность повторения параметров радужной оболочки и сетчатки глаза имеет порядок 10^{-78}, такие системы являются наиболее надежными среди всех биометрических систем. Такие средства идентификации применяются там, где требуется высокий уровень безопасности (например, в режимных зонах военных и оборонных объектов).

По информации консалтинговой компании International Biometric Group из Нью-Йорка по уровню спроса наиболее популярной технологией стало сканирование отпечатков пальцев. По уровню продаж биометрических устройств 44% приходится на дактилоскопические сканеры. Системы распознавания черт лица занимают второе место по уровню спроса, который составляет 14%; далее следуют устройства распознавания по форме ладони (13%), по голосу (10%) и радужной оболочке глаза (8%). Устройства верификации подписи в этом списке составляют 2%.

Биометрический подход позволяет упростить процесс выяснения, «кто есть кто». При использовании дактилоскопических сканеров и устройств распознавания голоса для входа в сети сотрудники избавляются от необходимости запоминать сложные пароли. Ряд компаний интегрируют биометрические возможности в системы однократной аутентификации SSO (Single Sign-On) масштаба предприятия. Подобная консолидация позволяет сетевым администраторам заменить службы однократной аутентификации паролей биометрическими технологиями.

Одной из первых областей широкого применения биометрической аутентификации личности станут мобильные системы. Проблема не сводится только к потерям компьютеров из-за краж, нарушение защиты информации может привести к значительно большим потерям. Кроме того, ноутбуки часто предоставляют доступ к корпоративной сети через программные соединения (выполняемые с помощью паролей, хранящихся на мобильных компьютерах).

Твердотельные датчики отпечатков пальцев – небольшие, недорогие и низко энергоемкие – позволяют решить эти проблемы. С помощью соответствующего программного обеспечения эти устройства дают возможность выполнять аутентификацию для четырех уровней доступа к информации, хранящейся на мобильном компьютере: регистрация, выход из режима сохранения экрана, загрузка и дешифровка файлов.

Биометрическая аутентификация пользователя может играть серьезную роль в шифровании, обеспечивая блокировку доступа к секретному ключу, который позволяет воспользоваться этой информацией только истинному владельцу частного ключа. Владелец может затем применять свой секретный ключ для шифрования информации, передаваемой по частным сетям или через Интернет.

Ахиллесовой пятой многих систем шифрования является проблема безопасного хранения самого криптографического секретного ключа. Зачастую доступ к ключу длиной 128 или даже больше разрядов защищен лишь паролем из 6 символов, то есть 48 разрядов. Отпечатки пальцев обеспечивают намного более высокий уровень защиты, и, в отличие от пароля, их невозможно забыть.

5.5. АППАРАТНО-ПРОГРАММНЫЕ СИСТЕМЫ ИДЕНТИФИКАЦИИ И АУТЕНТИФИКАЦИИ

Основным способом защиты информации от злоумышленников считается внедрение так называемых средств *ААА*, или *ЗА* (authentication, authorization, administration – аутентификация, авторизация, администрирование). Среди средств *ААА* значимое место занимают аппаратно-программные Системы Идентификации и Аутентификации (*СИА*) и Устройства Ввода Идентификационных Признаков (*УВИП* – термин соответствует ГОСТ Р 51241-98), предназначенные для обеспечения защиты от *НСД* к компьютерам.

При использовании *СИА* сотрудник получает доступ к компьютеру или в корпоративную сеть только после успешного прохождения процедуры идентификации и аутентификации.

Идентификация заключается в распознавании пользователя по присущему или присвоенному ему идентификационному признаку.

Проверка принадлежности пользователю предъявленного им идентификационного признака осуществляется в процессе аутентификации.

В состав аппаратно-программных *СИА* входят идентификаторы, устройства ввода-вывода (считыватели, контактные устройства, адаптеры, платы доверенной загрузки, разъемы системной платы и др.) и соответствующее ПО.

Идентификаторы предназначены для хранения уникальных идентификационных признаков. Кроме того, они могут хранить и обрабатывать разнообразные конфиденциальные данные. Устройства ввода-вывода и ПО, пересылают данные между идентификатором и защищаемым компьютером.

На мировом рынке информационной безопасности сегмент *ААА* стабильно растет.

5.5.1. КЛАССИФИКАЦИЯ СИСТЕМ ИДЕНТИФИКАЦИИ И АУТЕНТИ ФИКАЦИИ

Современные *СИА* по виду используемых идентификационных признаков, разделяются на электронные, биометрические и комбинированные, рис. 5.7. В электронных системах идентификационные признаки представляются в виде цифрового кода, хранящегося в памяти идентификатора. Такие *СИА* разрабатываются на базе следующих идентификаторов:

- идентификаторы *iButton* (information button – информационная «таблетка»);
- контактные смарт-карты (smart card – интеллектуальная карта);
- бесконтактные радиочастотные идентификаторы (RFID-системы);

– бесконтактные смарт-карты;
– USB-ключи или USB-токены (token – опознавательный признак, маркер).

В биометрических системах идентификационными признаками являются индивидуальные особенности человека, называемые биометрическими характеристиками. В основе идентификации и аутентификации этого типа лежит процедура считывания предъявляемого биометрического признака пользователя и его сравнение с предварительно полученным шаблоном.

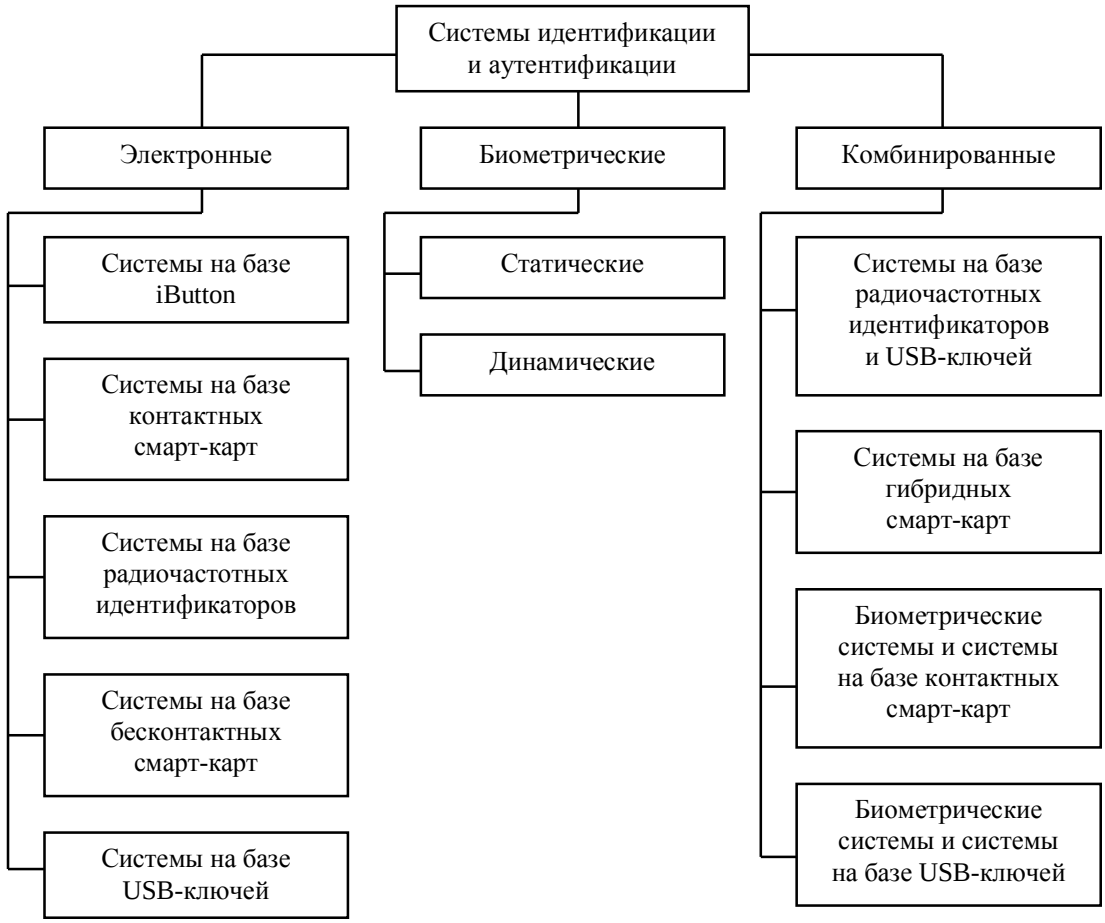


Рис. 5.7. Классификация Систем Идентификации и Аутентификации по виду идентификационных признаков

В зависимости от вида используемых характеристик биометрические системы делятся на статические и динамические.

Статическая биометрия (также называемая физиологической) основывается на данных, получаемых из измерений анатомических особенностей человека (отпечатков пальцев, формы кисти руки, узора радужной оболочки глаза, схемы кровеносных сосудов лица, рисунка сетчатки глаза, черт лица, фрагментов генетического кода и др.).

Динамическая биометрия (также называемая поведенческой) основывается на анализе совершаемых человеком действий (параметров голоса, динамики и формы подписи).

Несмотря на многочисленность биометрических характеристик, разработчики СИА основное внимание уделяют технологиям распознавания по отпечаткам пальцев, чертам лица, геометрии руки и радужной оболочке глаза. Например, согласно отчету International Biometric Group (www.biometricgroup.com), на мировом рынке средств биометрической защиты в 2004 году доля систем распознавания составила: по отпечаткам пальцев – 48%, по чертам лица – 12%, по геометрии руки – 11%, по радужной оболочке глаза – 9%, по параметрам голоса – 6%, по подписи – 2%. Оставшаяся доля (12%) – относится к промежуточным средствам.

В комбинированных системах для идентификации используется одновременно несколько идентификационных признаков. Такая интеграция позволяет воздвигнуть перед злоумышленником дополнительные преграды, которые он не сможет преодолеть, а если и сможет, то со значительными трудностями. Разработка комбинированных систем осуществляется по двум направлениям:

- интеграция идентификаторов в рамках системы одного класса;
- интеграция систем разного класса.

В первом случае для защиты компьютеров от НСД используются системы, базирующиеся на бесконтактных смарт-картах и USB-ключах, а также на гибридных (контактных и бесконтактных) смарт-картах. Во втором случае разработчики умело «скрещивают» биометрические и электронные СИА (далее такой конгломерат называется биоэлектронными СИА).

По способу обмена данными между идентификатором и устройством ввода-вывода электронные СИА подразделяются на контактные и бесконтактные.

Контактное считывание идентификационных признаков подразумевает непосредственное соприкосновение идентификатора с устройством ввода-вывода.

Бесконтактный (дистанционный) способ обмена данными не требует четкого позиционирования идентификатора и устройства ввода-вывода. Чтение или запись данных происходит при поднесении идентификатора на определенное расстояние к устройству ввода-вывода.

Основным элементом электронных контактных и бесконтактных смарт-карт и USB-ключей являются одна или более встроенных интегральных микросхем (чипов), которые могут представлять собой микросхемы памяти, микросхемы с жесткой логикой и микропроцессоры (процессоры). В настоящее время наибольшей функциональностью и степенью защищенности обладают идентификаторы с процессором.

Основу чипа микропроцессорной контактной смарт-карты составляют:

- центральный процессор, специализированный криптографический процессор (опционально);
- оперативная память (ОЗУ, оперативное запоминающее устройство – RAM);
- постоянная память (ПЗУ, постоянное запоминающее устройство – ROM);
- энергонезависимая программируемая постоянная память (ППЗУ, программируемое постоянное запоминающее устройство – PROM);
- датчик случайных чисел, таймеры, последовательный коммуникационный порт.

Оперативная память используется для временного хранения данных, например результатов вычислений, произведенных процессором. Ее емкость составляет несколько килобайтов.

В постоянной памяти хранятся команды, исполняемые процессором, и другие неизменяемые данные. Информация в ПЗУ записывается при производстве карты. Емкость памяти может составлять десятки килобайтов.

Выбор СИА целесообразно проводить путем сравнения наиболее важных характеристик изделий. К таким характеристикам можно отнести следующие:

- структура идентификатора;
- структура и состав устройства ввода-вывода;
- надежность изделия;
- интеграция с системами защиты информации (СЗИ);
- стоимость изделия.

С точки зрения стоимости более предпочтительны СИА на базе USB-ключей и *iButton*, в составе которых отсутствуют дорогостоящие считыватели.

5.5.2. ЭЛЕКТРОННЫЕ ИДЕНТИФИКАТОРЫ

Первый шаг в создании карт-идентификаторов был сделан в Германии в 1968 году, когда Юргену Деслофу и Гельмуту Гротруппу удалось поместить интегральную схему в кусочек пластика. В 1974 году француз Ролан Морено запатентовал идею интеграции микросхемы в пластиковую карту. Но только в конце 80-х годов достижения в области микроэлектроники сделали возможным воплощение этой идеи в жизнь.

История развития СИА на базе изделий *iButton* началась в 1991 году с создания корпорацией Dallas Semiconductor первых идентификаторов *Touch Memory* (таково их начальное название). В настоящее время Dallas Semiconductor представляет собой дочернее предприятие компании Maxim Integrated Products. Результатом их совместной деятельности является выпуск более 20 моделей идентификаторов *iButton*.

Системы идентификации и аутентификации на базе USB-ключей появились в конце 90-х годов. Являясь преемником технологий смарт-карт и электронных ключей, используемых для защиты программного обеспечения, USB-ключи довольно быстро завоевали популярность.

СИА на базе смарт-карт и радиочастотных идентификаторов можно отнести по времени их создания к старшему поколению, *iButton* – к среднему, а USB-ключей – к младшему.

Идентификаторы iButton

Идентификатор *iButton* относится к классу электронных контактных идентификаторов. Модельный ряд идентификаторов *iButton* довольно широк и разнообразен (более 20 моделей). В общем виде идентификатор iButton представляет собой микросхему (чип), вмонтированную в герметичный стальной корпус. Питание микросхемы (чипа) обеспечивает миниатюрная литиевая батарейка. Корпус имеет вид стандартного аккумулятора (рис. 5.8) и имеет диаметр 17,35 мм при высоте 5,89 мм (корпус F5) или 3,1 мм (корпус F3).

Рис. 5.8. Идентификаторы iButton

Корпус обеспечивает высокую

степень защищенности идентификатора от воздействия агрессивных сред, пыли, влаги, внешних электромагнитных полей, механических ударов и т.п. Идентификатор легко крепится на носителе (карточке, брелоке).

Основу чипа составляют мультиплексор и память. Память идентификаторов *iButton* состоит из следующих компонентов:

- ПЗУ;
- энергонезависимая NV (nonvolatile) оперативная память NV RAM;
- сверхоперативная SM (scratchpad memory), или блокнотная, память.

В ПЗУ хранится 64-разрядный код, состоящий из 48-разрядного уникального серийного номера (идентификационного признака), восьмиразрядного кода типа идентификатора и восьмиразрядной контрольной суммы.

Память NV RAM может быть использована для хранения как общедоступной, так и конфиденциальной информации (криптографических ключей, паролей доступа и других данных).

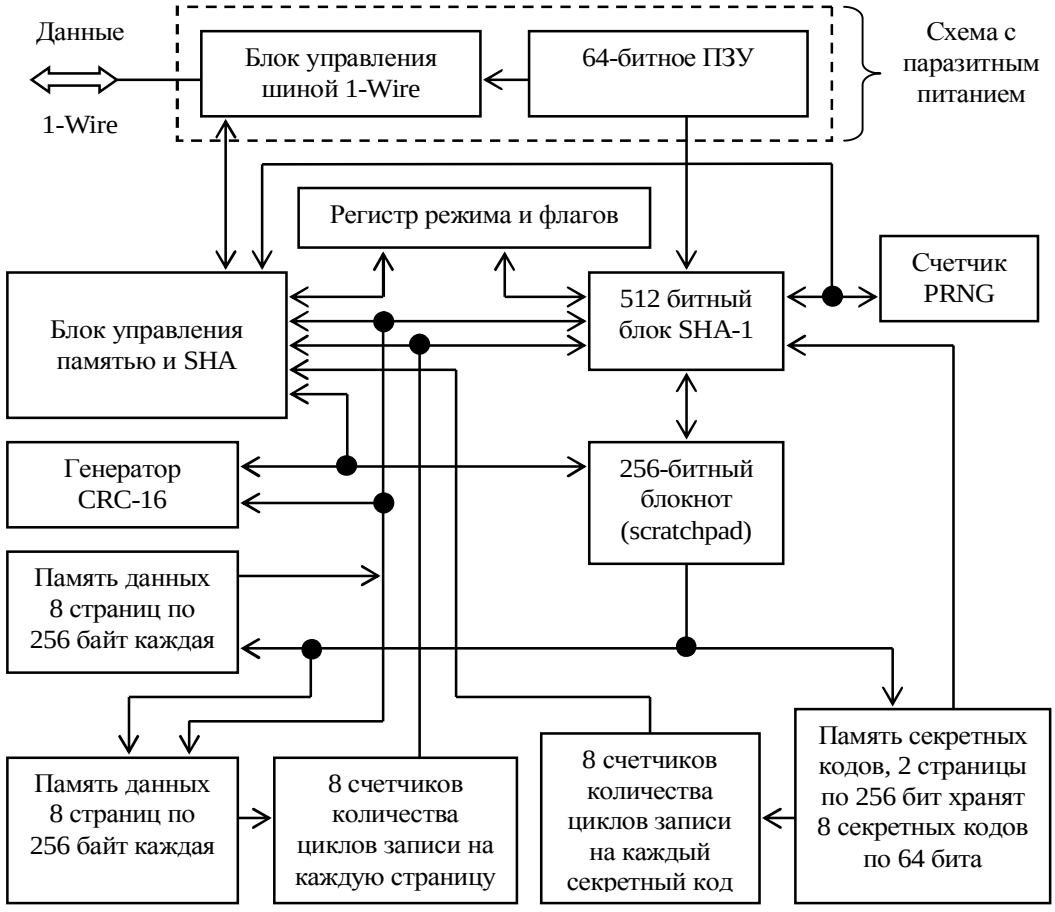


Рис. 5.9. Структурная схема iButton DS1963S

Память SM является буферной и выполняет функции блокнотной памяти.

Существует несколько модификаций идентификаторов *iButton* семейства DS199X, которые различаются емкостью памяти и функциональными возможностями. В табл. 5.3 представлены основные характеристики памяти идентификаторов *iButton*, используемых для защиты компьютеров от НСД.

Помимо этого некоторые типы идентификаторов содержат дополнительные компоненты. Например, в идентификаторе DS1963S, рис. 5.9, имеется микроконтроллер, предназначенный для вычисления в соответствии со стандартом хэш-функции SHA-1 160-разрядного кода аутентификации сообщений и генерации ключей доступа для страниц памяти, а в корпус идентификатора DS1994L встроены часы реального времени.

Обмен информацией между идентификатором и компьютером происходит в соответствии с протоколом 1-Wire с помощью разнообразных считывающих устройств (адаптеров последовательного, параллельного и USB-портов). Для записи и считывания данных из идентификатора нужно, чтобы корпус *iButton* соприкоснулся со считывающим устройством. Время контакта - не более 5 мс, гарантированное количество контактов составляет несколько миллионов. Интерфейс 1-Wire обеспечивает обмен информацией на скоростях 16 или 142 Кбит/с (ускоренный режим).

К достоинствам идентификаторов на базе электронных ключей *iButton* относятся:

- надежность, долговечность (время хранения информации в памяти идентификатора составляет не менее 10 лет);
- высокая степень механической и электромагнитной защищенности;
- малые размеры;
- относительно невысокая стоимость.

Недостатком этого устройства является зависимость его срабатывания от точности соприкосновения идентификатора и считывателя, осуществляемого вручную.

Идентификаторы на базе контактных смарт-карт

Контактные смарт-карты относятся к классу электронных контактных идентификаторов. Контактные смарт-карты принято делить на процессорные карты и карты с памятью. Обычно они выпускаются в виде пластиковых карточек. На рынке безопасности сначала появились карты с открытой памятью, затем – с защищенной памятью и наконец – процессорные смарт-карты. Физический, электрический, механический и программный интерфейсы смарт-карт определяются базовым стандартом ISO 7816 (части 1–10).

Таблица 5.3. Основные характеристики памяти идентификаторов iButton

Тип изделия	Емкость NV RAM	Емкость SM, битов	Емкость ПЗУ,байтов	Примечание
DS1963S	4 Кбит (16 страниц по 256 бит)	256	64	8 страниц NV RAM защищаются паролями. Реализация SHA-1
DS1991	1152 бит (4 страницы по 256 бит)	256	64	3 блока NV RAM защищаются паролями
DS1992L	1 кбит (4 страницы по 256 бит)	256	64	Незащищенная NV RAM
DS1993L	4 кбит (16 страниц	256	64	Незащищенная

	по 256 бит)			NV RAM
DS1994L	4 кбит (16 страниц по 256 бит)	256	64	Незащищенная NV RAM. Часы реального времени
DS1995L	16 кбит (64 страницы по 256 бит)	256	64	Незащищенная NV RAM
DS1996L	64 кбит (256 страниц по 256 бит)	256	64	Незащищенная NV RAM

Основу внутренней структуры современной процессорной смарт-карты составляет чип, в состав которого входят центральный процессор, ОЗУ, ПЗУ и ЭСППЗУ (Электрически стираемое программируемое постоянное запоминающее устройство – EEPROM). Как правило, в чипе также присутствует специализированный сопроцессор, рис. 5.10.

Оперативная память используется для временного хранения данных, например результатов вычислений, произведенных процессором. Емкость памяти составляет несколько килобайтов.

В ПЗУ (обычно масочная память) хранятся команды, исполняемые процессором, и другие неизменяемые данные. Информация в ПЗУ записывается в процессе производства карты. Емкость памяти может составлять десятки килобайтов.

В смарт-картах используется два типа ППЗУ: СППЗУ (Стираемое программируемое постоянное запоминающее устройство – EPROM) и более распространенное ЭСППЗУ. В последнем хранятся пользовательские данные, которые могут считываться, записываться и модифицироваться, а также конфиденциальные данные (например, криптографические ключи), недоступные

Рис. 5.10. Идентификатор на базе смарт-карт

для прикладных программ. Емкость памяти составляет десятки и сотни килобайтов.

Центральный процессор смарт-карты (обычно это RISC-процессор) обеспечивает реализацию разнообразных процедур обработки данных, контроль доступа к памяти и управление ходом выполнения вычислительного процесса.

На специализированный процессор возлагается реализация различных процедур, необходимых для повышения защищенности СИА, в том числе:

– генерация криптографических ключей;

- реализация криптографических алгоритмов (ГОСТ 28147-89, DES, 3DES, RSA, SHA-1);
- выполнение операций с электронной цифровой подписью (генерация и проверка);
- выполнение операций с PIN-кодом и др.

В ПЗУ хранится исполняемый код процессора, оперативная память используется в качестве рабочей, ЭСППЗУ необходимо для хранения изменяемых данных владельца карты.

Примером идентификатора на смарт-карте может служить карта CryptoFlex имеющая криптопроцессоры RSA и Тройной DES, а использование 1024 битных ключей RSA гарантирует «непробиваемую» защиту. В состав идентификатора входит:

- микропроцессор и криптопроцессор;
- 4 килобайта EEPROM;
- один источник питания в 5 вольт;
- надежность хранения данных – 10 лет;
- может иметь место для подписи и/или магнитной полосы;
- персонализация или номерная сериализация термическим или лазерным принтером;
- операционная система;
- стандарт ISO 7816 - 1,2,3,4;
- управление совместимыми с X.509 сертификатами;
- динамическое и логическое управление файлами с данными;
- защита данных (PIN, секретные ключи, взаимная аутентификация и т.п.);
- использует алгоритмы DES, Triple-DES, RSA 512/768/1024;
- вычисление подписи RSA;
- проверка подписи RSA;
- внутренняя генерация ключей RSA;
- защищенный генератор случайных чисел;
- управление многочисленными ключами RSA;
- совместим с продуктами серии MultiFlex, CyberFlex и др.

Бесконтактные радиочастотные идентификаторы

Бесконтактные радиочастотные идентификаторы, или RFID-системы (radio-frequency identification - радиочастотная идентификация), относятся к классу электронных бесконтактных радиочастотных устройств. Радиочастотные идентификаторы типа Proximity (от англ. proximity - близость, соседство) выпускаются в виде карточек, брелоков, браслетов, ключей и т.п. Каждый из них имеет собственный уникальный серийный номер.

Основными их компонентами являются интегральная микросхема, осуществляющая связь со считывателем, и встроенная антенна. В состав чипа входит микросхема памяти (или микросхема с жесткой логикой) со вспомогательными блоками: модулем программирования, модулятором, блоком управления и другими модулями. Емкость памяти составляет от 8 до 256 байт. В радиочастотном идентификаторе в основном используется СППЗУ, но встречается и ЭСППЗУ. В памяти содержатся уникальный номер идентификатора, код устройства

и служебная информация (биты четности, биты начала и конца передачи кода и т.д.).

В табл. 5.4 представлены основные характеристики бесконтактных идентификаторов.

Внутри радиочастотного идентификатора может находиться источник питания – литиевая батарея. Такие идентификаторы называются активными. Они обеспечивают взаимодействие со считывателем на значительном расстоянии (в несколько метров).

Обычно радиочастотные идентификаторы являются пассивными и не содержат источника питания. В этом случае питание микросхемы происходит посредством электромагнитного поля, излучаемого считывателем. Чтение данных осуществляется считывателем со скоростью 4 Кбит/с на расстоянии до 1 м.

Таблица 5.4. Бесконтактные идентификаторы

Характеристика	Идентификаторы Proximity	Смарт-карта	
		Стандарт ISO/IEC 14443	Стандарт ISO/IEC 15693
Частота радиоканала	125кГц, 13.56МГц	13.56МГц	13.56МГц
Дистанция чтения	До 1м	До 10м	До 1м
Встроенные типы чипов	Микросхема памяти, микросхема с жесткой логикой	Микросхема памяти, микросхема с жесткой логикой, процессор	Микросхема памяти, микросхема с жесткой логикой
Функция памяти	Только чтение	Чтение / запись	Чтение / запись
Емкость памяти	8 - 256 байт	64 байт – 64 Кб	256 байт – 2 Кб
Алгоритмы шифрования и аутентификации	Нет	Технология MIFARE, DES, 3DES, AES, RSA, ECC	DES, 3DES
Механизм антиколлизии	Опционально	Есть	Есть

Считывающее устройство постоянно излучает радиосигнал. Когда идентификатор оказывается на определенном расстоянии от считывателя, антенна поглощает сигнал и передает его на микросхему. Получив энергию, идентификатор излучает дентификационные данные, принимаемые считывателем. Дистанция считывания в значительной степени зависит от характеристик антенного и приемо-передающего трактов считывателя. Весь процесс занимает несколько десятков микросекунд.

Считывающее устройство может размещаться внутри корпуса компьютера. Взаимная ориентация идентификатора и считывателя не имеет значения, а ключи и другие предметы, находящиеся в контакте с картой, не мешают передаче информации. В соответствии с используемой несущей частотой RFID-системы классифицируются по частоте:

- низкочастотные (100–500 кГц) характеризуются незначительным расстоянием считывания (десятки сантиметров). Идентификационный код считывается через одежду, сумки, портмоне и т.п.;

– устройства промежуточной частоты (10–15 МГц) способны передавать значительные объемы данных;
– высокочастотные (850–950 МГц или 2,4–5 ГГц) характеризуются большой дистанцией считывания (в несколько метров).
– Системы идентификации и аутентификации на базе радиочастотных идентификаторов обычно криптографически не защищены (за исключением заказных систем).

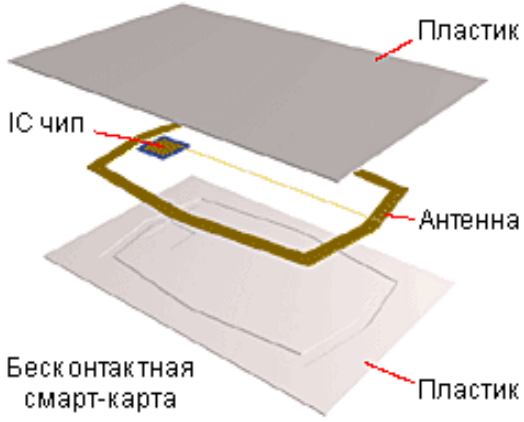
Основными достоинствами радиочастотных идентификаторов (RFID-систем) являются:

– бесконтактная технология считывания;
– долговечность пассивных идентификаторов (некоторые фирмы-производители дают на карты пожизненную гарантию);
– точность, надежность и удобство считывания идентификационных признаков.

К недостаткам RFID-систем относят слабую электромагнитную защищенность и относительно высокую стоимость, учитывая затраты на считыватели.

Идентификаторы на базе бесконтактных смарт-карт

В отличие от контактных смарт-карт, в состав бесконтактных смарт-карт на базе стандарта MIFARE дополнительно входит радиочастотный модуль со встроенной антенной, необходимой для связи со считывателем и питания микросхемы. Бесконтактные смарт-карты функционируют на частоте 13,56 МГц и разделяются на два класса, которые базируются на международных стандартах ISO/IEC 14443 и ISO/IEC 15693.



Стандарт ISO/IEC 14443 включает в себя версии A и B, различающиеся способами модуляции передаваемого радиосигнала. Стандарт поддерживает обмен (чтение/запись) данными со скоростью 106 Кбит/с (возможно увеличение скорости до 212, 424 или 848 Кбит/с), дистанция чтения – до 10 см. Для реализации функций шифрования и аутентификации в идентификаторах стандарта ISO/IEC 14443 могут применяться чипы трех видов: микросхема с жесткой логикой MIFARE, процессор или криптографический процессор. Технология MIFARE является разработкой компании Philips Electronics и представляет собой расширение ISO/IEC 14443 (версии A).

Стандарт ISO/IEC 15693 увеличивает дистанцию применения бесконтактного идентификатора до 1 м. На этом расстоянии обмен данными осуществляется со скоростью 26,6 Кбит/с.

Каждая смарт-карта обладает собственным уникальным серийным номером. Он задается на заводе-изготовителе, его нельзя изменить на протяжении всего срока эксплуатации карты. Идентификация по серийному номеру, шифрование данных и аутентификация областей памяти с помощью секретных ключей обеспечивают надежную защиту смарт-карт от взлома.

По отношению к компьютеру устройства чтения смарт-карт могут быть внешними и внутренними (например, встроенными в клавиатуру, гнездо дисковода 3,5'', корпус компьютера). Считыватель работает под управлением специальной программы – драйвера устройства чтения.

На базе ISO 7816 разработан единый стандартный интерфейс для работы со смарт-картами. Включенные в него спецификации PC/SC облегчают интеграцию технологий смарт-карт в программно-аппаратные комплексы на базе платформы персонального компьютера и создание средств разработки приложений для смарт-карт.

Несомненными достоинствами идентификаторов на базе смарт-карт считается удобство хранения идентификатора (например, его можно держать в бумажнике вместе с другими карточками) и считывания идентификационных признаков.

К недостаткам можно отнести ограниченный срок эксплуатации из-за неустойчивости смарт-карты к механическим повреждениям и относительно высокую стоимость считывателей смарт-карт.

Идентификаторы на базе USB-ключей

Идентификаторы на базе USB-ключей относятся к классу электронных контактных устройств. Идентификаторы данного типа не требуют дорогостоящих аппаратных считывателей. Идентификатор, называемый USB-ключом, подключается к USB-порту непосредственно или с помощью соединительного кабеля.

Рис. 5.11. Идентификатор ruToken

Конструктивно USB-ключи выпускаются в виде брелоков (рис. 5.11), которые легко размещаются на связке с обычными ключами. Брелоки выпускаются в цветных корпусах и снабжаются световыми индикаторами состояния. Каждый идентификатор имеет прошиваемый при изготовлении собственный уникальный 32/64-разрядный серийный номер.

Как было отмечено выше, USB-ключи являются преемниками контактных смарт-карт. Поэтому структуры USB-ключей и смарт-карт практически идентичны. Объемы аналогичных запоминающих устройств также соответствуют друг другу.

В состав USB-ключей могут входить:

- процессор – управление и обработка данных;
- криптографический процессор – реализация алгоритмов ГОСТ 28147-89, DES, 3-DES, RSA, DSA, MD5, SHA-1 и других криптографических преобразований;
- USB-контроллер – обеспечение интерфейса с USB-портом компьютера;
- ОЗУ – хранение изменяемых данных;
- ЭСППЗУ – хранение ключей шифрования, паролей, сертификатов и других важных данных;
- ПЗУ – хранение команд и констант.

Поддержка спецификаций PC/SC позволяет без труда переходить от смарт-карт к USB-ключам и встраивать их как в существующие приложения, так и в новые.

В табл. 5.5 представлены некоторые характеристики USB-ключей.

Достоинства идентификаторов на базе USB-ключей заключаются в отсутствии аппаратного считывателя, малых размерах и удобстве хранения идентификаторов, а также в простоте подсоединения идентификатора к USB-порту. К недостаткам можно отнести относительно высокую стоимость и слабую механическую защищенность брелока.

Таблица 5.5. Характеристики USB-ключей

Изделие	Емкость памяти, Кб	Разрядность серийного номера	Алгоритмы шифрования
iKey 20xx	8/32	64	DES (режим ECB и CDC), 3-DES, RC2, RC4, RC5, MDS, RSA-1024/2048
eToken R2	16/32/64	32	DESX (ключ 120 бит), MDS
eToken PRO	16/32	32	RSA/1024, DES, 3-DES, SHA-1
ePass1000	8/32	64	MD5, MD5-HMAC
ePass2000	16/32	64	RSA, DES, 3-DES, DSA, MD5, SHA-1
ruToken	8/16/32/64/128	32	ГОСТ 28147-89, RSA, DES, 3-DES, RC2, RC4, MD4, MD5, SHA-1

Обмен данными между идентификатором и защищаемым компьютером

Выбор СИА может зависеть от структуры и состава устройства ввода-вывода, обеспечивающего обмен данными между идентификатором и защищаемым компьютером.

Наиболее просто обмен данными осуществляется в СИА на базе USB-ключей. В этих системах аппаратное устройство ввода-вывода отсутствует: идентификатор подсоединяется к USB-порту рабочей станции, портативного компьютера, клавиатуры или монитора напрямую либо с помощью кабеля-удлинителя.

В СИА на базе *iButton* обмен информацией с компьютером идет в соответствии с протоколом однопроводного интерфейса 1-Wire через последовательный, параллельный и USB-порты, а также дополнительную плату расширения. Данные записываются в идентификатор и считываются из него путем прикосновения корпуса *iButton* к контактному устройству, встроенному в адаптер соответствующего порта, либо к контактному устройству с удлинительным кабелем, присоединенным к адаптеру. Гарантированное количество контактов *iButton* составляет несколько миллионов соединений.

Интерфейс 1-Wire обеспечивает обмен данными в полудуплексном режиме со скоростями 16 и 142 Кбит/с (вариант ускоренного обмена). Взаимодействие устройств по однопроводному интерфейсу организовано по принципу ведущий–ведомый. При этом контактное устройство всегда ведущее, а один или несколько идентификаторов *iButton* – ведомые.

В состав СИА на базе смарт-карт (контактных и бесконтактных) и RFID-идентификаторов входят считывающие устройства (считыватели, ридеры), которые подключаются к параллельному, последовательному, USB-портам, дополнительной плате расширения компьютера, к плате PC Card портативного компьютера. По отношению к корпусу компьютера считыватели могут быть внешними и внутренними. Питаются считыватели от различных источников – блока питания компьютера, внешнего источника питания или стандартных батареек.

Надежность

При обсуждении надежности СИА обычно рассматривают самое важное и в то же время самое слабое звено системы – идентификатор. В свою очередь, надежность идентификаторов связывают со степенью их защищенности от механических воздействий, влияния температуры, внешних электромагнитных полей, агрессивных сред, пыли, влаги, а также от атак, направленных на вскрытие чипов, хранящих секретные данные.

Разработчики идентификаторов *iButton* обеспечивают сохранность характеристик своих изделий при механическом ударе 500 g, падении с высоты 1,5 м на бетонный пол, рабочем диапазоне температур от -40 до 70 °С, воздействии электромагнитных полей и атмосферы. Этому способствует герметичный стальной корпус идентификатора, сохраняющий прочность при миллионе контактов с устройством ввода-вывода. Память некоторых идентификаторов (DS1991, DS1963S) защищена от доступа. Срок эксплуатации идентификатора *iButton* составляет 10 лет и определяется сроком службы литиевой батарейки.

К недостаткам СИА на базе *iButton* следует отнести отсутствие встроенных в идентификаторы криптографических средств, реализующих шифрование данных при их хранении и передаче в компьютер. Поэтому *iButton* обычно используется совместно с другими системами, на которые возлагаются функции шифрования.

По степени механической надежности радиочастотные идентификаторы, смарт-карты и USB-ключи уступают *iButton*. Выход из строя карты вследствие механических повреждений является не таким уж редким событием. Проводившиеся в ходе реализации французского проекта GIE Carte Bancaire десятилетние исследования над 22 миллионами карт показали, что вероятность их отказа по ряду причин (куда также входят механические повреждения) составляет 0,022.

Слабым местом USB-ключей является и ресурс их USB-разъемов. Разработчики данных идентификаторов даже включают этот показатель в технические спецификации изделий. Например, для идентификаторов семейства *eToken* гарантированное число подключений составляет не менее 5000 раз.

Достоинство радиочастотных идентификаторов, смарт-карт и USB-ключей состоит в том, что в их состав входят защищенная энергонезависимая память и криптографический процессор, позволяющие повысить уровень защиты устройств. Однако и атакующая сторона придумывает разнообразные способы вскрытия секретной информации.

Опубликованы работы, в которых описываются разнообразные атаки на чипы идентификаторов. Эти исследования носят как теоретический, так и практический характер. К теоретическим методам вскрытия относят, в частности, атаки Bellcore, дифференциальный анализ искажений DFA (Differential Fault Analysis) и питания DPA (Differential Power Analysis). К практическим методам можно отнести

физические атаки, направленные на распаковку чипа и извлечение необходимой информации.

Разработчики криптографических процессоров стремятся по мере возможности адекватно реагировать на атаки с помощью разнообразных механизмов внешней и внутренней защиты. К механизмам внешней защиты относят установку датчиков (емкостный либо оптический сенсор), покрытие чипа металлическим слоем, специальными клеями и т.д.; к внутренним – шифрование шины, случайное тактирование, проведение повторных вычислений, генерирование шума.

5.5.3. БИОМЕТРИЧЕСКИЕ ИДЕНТИФИКАТОРЫ

В основе биометрической идентификации и аутентификации лежит считывание и сравнение предъявляемого биометрического признака пользователя с имеющимся эталоном. Такого рода признаки включают в себя отпечатки пальцев, форму и термограмму лица, рисунок сетчатки и радужной оболочки глаза, геометрию руки, узор, образуемый кровеносными сосудами ладони человека, речь и т.д. Высокий уровень защиты определяется тем, что биометрия позволяет идентифицировать человека, а не устройство. Биометрические идентификаторы могут быть контактными и бесконтактными (дистанционными).

До недавнего времени широкое использование биометрических идентификаторов в средствах контроля доступа к компьютерам тормозилось их высокой ценой. Успехи в технологиях, теории распознавания и микроэлектронике позволили разработчикам снизить стоимость устройств и тем самым активизировать рынок. Особенно эта тенденция характерна для устройств дактилоскопического доступа. Отпечаток пальца считается одним из наиболее устойчивых идентификационных признаков (не изменяется со временем, при повреждении кожного покрова идентичный папиллярный узор полностью восстанавливается, при сканировании не вызывает дискомфорта у пользователя).

Считыватели (или сканеры) отпечатков пальцев представляют собой устройства, либо подключаемые к одному из портов компьютера, либо встраиваемые в компьютерные мыши (рис. 5.12), клавиатуры, корпусы мониторов. Широкое распространение получили дактилоскопические мыши.

Рис. 5.12. Устройства сканирования отпечатка пальцев компании BioLink

Ряд биометрических идентификаторов имеет отличные характеристики точности распознавания (вероятность ложного отказа в доступе составляет 10^{-2}, а вероятность ложного доступа – 10^{-9}).

Комбинированные системы идентификации и аутентификации

В последнее время предметом пристального внимания и интереса стали средства двухфакторной аутентификации, которые эффективно решают задачу по обеспечению безопасного доступа к информационным ресурсам своих компаний. Такие средства аутентификации можно получить с помощью комбинированных СИА. Внедрение комбинированных СИА в корпоративную систему информационной безопасности позволяет увеличить количество используемых идентификационных признаков (факторов). Эффективность защиты компьютеров от НСД повышается при комбинировании идентификаторов различных типов (табл. 5.6).

Таблица 5.6. Основные функции комбинированных СИА

Функция	СИА на базе бесконтактных идентификаторов и USB-ключей	СИА на базе гибридных смарт-карт	Биоэлектронные системы
Идентификация и аутентификация	Есть	Есть	Есть
Блокировка работы компьютера и разблокирование при предъявлении персонального идентификатора	Есть	Есть	Есть
Идентификация и аутентификация сотрудников при их доступе в здание, помещение (и при выходе из него)	Есть	Есть	Нет
Хранение конфиденциальной информации (ключей шифрования, паролей, сертификатов)	Есть	Есть	Есть
Визуальная идентификация	Нет	Есть	Есть

В настоящее время применяются комбинированные системы идентификации и аутентификации следующих типов:

- системы на базе радиочастотных идентификаторов и USB-ключей;
- системы на базе гибридных смарт-карт;
- биоэлектронные системы.

Кроме того, некоторые типы таких систем способны управлять физическим доступом в здания и помещения и контролировать его.

Радиочастотные идентификаторы и USB-ключи

Аппаратная интеграция USB-ключей и радиочастотных идентификаторов предполагает, что в корпус брелока встраиваются антенна и микросхема, поддерживающая бесконтактный интерфейс. Это позволяет с помощью одного идентификатора организовать управление доступом и к компьютеру, и в помещения офиса. Для входа в служебное помещение сотрудник использует свой идентификатор в качестве бесконтактной карты, а при допуске к защищенным компьютерным данным - в качестве USB-ключа. Кроме того, при выходе из помещения он извлекает идентификатор из USB-разъема (чтобы потом войти обратно) и тем самым автоматически блокирует работу компьютера.

В 2004 году появилось два комбинированных идентификатора такого типа:

- RFiKey - разработка компании Rainbow Technologies;
- eToken PRO RM - разработка компании Aladdin Software Security

Идентификатор RFiKey представляет собой USB-ключ iKey со встроенной микросхемой Proximity. Изделие RFiKey поддерживает интерфейс USB 1.1/2.0 и функционирует со считывателями HID Corporation.

К основным характеристикам RFiKey можно отнести следующие показатели:

- частота функционирования радиочастотного идентификатора Proximity - 125 кГц;
- тактовая частота процессора - 12 МГц;
- реализуемые криптографические алгоритмы - MD5, RSA, DES, 3-DES, RC2, RC4, RC5;
- наличие аппаратного датчика случайных чисел;
- поддерживаемые стандарты - PKCS#11, MS Crypto API, PC/SC;
- файловая система с тремя уровнями доступа к данным;
- поддерживаемые операционные системы - Windows 95/98/ME/NT4/ 2000/XP.

Идентификатор eToken RM представляет собой USB-ключ eToken Pro со встроенным чипом, поддерживающим бесконтактный интерфейс. Например, радиочастотный пассивный идентификатор БИМ-002 российской компании «Ангстрем» изготовлен в виде круглой метки. Он построен на базе микросхемы 5004xk1, основой которой являются память СППЗУ емкостью 64 бит и блок программирования, используемый для записи уникального идентификационного кода.

Разница между стоимостью комбинированных и обычных USB-ключей приблизительно соответствует цене радиочастотного идентификатора Proximity. Отсюда следует, что интеграция бесконтактных радиочастотных идентификаторов и USB-ключей почти не ведет к росту затрат на аппаратную часть при переходе на комбинированную систему идентификации и аутентификации. Выигрыш же очевиден: один идентификатор вместо двух.

Гибридные смарт-карты

Гибридные смарт-карты содержат не связанные между собой разнородные чипы. Один чип поддерживает контактный интерфейс, другие (Proximity, ISO 14443/15693) – бесконтактный. Как и в случае интеграции USB-ключей и радиочастотных идентификаторов, СИА на базе гибридных смарт-карт решают двоякую задачу: защиту от несанкционированного доступа к компьютерам и в помещения компании, где они содержатся. Кроме этого на смарт-карте помещается фотография сотрудника, что позволяет идентифицировать его визуально.

Анализ затрат при переходе на применение гибридных смарт-карт, как и в случае комбинирования радиочастотных идентификаторов и USB-ключей, снова подтверждает торжество принципа «два в одном». Если же на идентификатор поместить фотографию сотрудника, то этот принцип трансформируется в «три в одном».

Биоэлектронные системы

Для защиты компьютеров от НСД биометрические системы обычно объединяются с двумя классами электронных СИА – на базе контактных смарт-карт и на базе USB-ключей.

Интеграция с электронными системами на базе бесконтактных смарт-карт главным образом используется в системах управления физическим доступом в помещения.

Как уже было замечено, технологии идентификации по отпечаткам пальцев сегодня лидируют на рынке биометрических средств защиты. Этот успех дактилоскопии обусловлен следующими обстоятельствами:

- это самый старый и наиболее изученный метод распознавания;
- его биометрический признак устойчив: поверхность кожного покрова на пальце не меняется со временем;
- высокие значения показателей точности распознавания;
- простота и удобство процедуры сканирования;
- эргономичность и малый размер сканирующего устройства;
- самая низкая цена среди биометрических систем идентификации.

В связи с этим сканеры отпечатков пальцев стали широко используемой составной частью комбинированных СИА, применяемых для защиты компьютеров от НСД. Примером такого рода интеграции служат изделия Precise 100 MC компании Precise Biometrics AB. Чтобы получить доступ к информационным ресурсам компьютера с помощью подобных средств, пользователю необходимо вставить в считыватель смарт-карту и приложить палец к сканеру. Шаблоны отпечатков пальцев хранятся в зашифрованном виде в защищенной памяти смарт-карты. При совпадении изображения отпечатка с шаблоном разрешается доступ к компьютеру. Пользователь очень доволен: не надо запоминать пароль или PIN-код, процедура входа в систему значительно упрощается.

Изделие Precise 100 MC - это USB-устройство, работающее в среде Windows. Считыватель смарт-карт поддерживает все типы микропроцессорных карточек, удовлетворяющих стандарту ISO 7816-3 (протоколы T=0, T=1). Дактилоскопический считыватель представляет собой сканер емкостного типа со скоростью сканирования 4 отпечатка пальцев в секунду.

Объединение USB-ключа с дактилоскопической системой идентификации называют USB-биоключом. В ближайшем будущем USB-биоключи могут получить широкое распространение благодаря своим достоинствам:

- высокий уровень защищенности (наличие дактилоскопического сканера, хранение секретных данных, шифрование обмена данными с компьютером);
- аппаратная реализация криптографических преобразований;
- отсутствие аппаратного считывателя;
- уникальность признака, малые размеры и удобство хранения идентификаторов.

Особенности применения внешних носителей ключевой информации для идентификации и аутентификации

В этом случае информация, идентифицирующая и аутентифицирующая пользователя, хранится на внешнем носителе информации, который может представлять собой электронный ключ, пластиковую карту и т.д. При входе в систему пользователь подключает к компьютеру носитель ключевой информации, и операционная система считывает с него идентификатор пользователя и соответствующий ему ключ, при этом идентификатор пользователя используется в качестве имени, а ключ – в качестве пароля.

Поскольку ключ, хранящийся на внешнем носителе, может быть гораздо более длинным, чем пароль, подобрать такой ключ практически невозможно. Однако актуальна угроза утери или кражи ключевой информации. Если процедура аутентификации не предусматривает дополнительных мер защиты, любой обладатель носителя ключевой информации, в том числе и злоумышленник, укравший этот носитель у легального пользователя системы, может войти в систему с правами пользователя, которому принадлежит носитель.

Поэтому данный механизм аутентификации, как правило, используется в совокупности с паролем. При этом пользователь должен не только «предъявить» компьютеру носитель ключевой информации, но и ввести соответствующий этому носителю пароль. Ключевая информация на носителе информации хранится зашифрованной на этом пароле, что не позволяет случайному обладателю ключа воспользоваться им. Основной угрозой при использовании описываемого механизма аутентификации является угроза кражи носителя ключевой информации с последующим его копированием и подбором пароля на доступ к ключу. Для затруднения подбора пароля на доступ к ключу используют следующие меры защиты:

- защиту ключевого носителя от копирования;
- блокировку или уничтожение ключевой информации после определенного количества неудачных попыток ввода пароля на доступ к ключу.

Однако эти меры защиты неприменимы, если в качестве носителя ключевой информации применяются электронные ключи Touch Memory или пластиковые карты Memory Card.

В отличие от перечисленных носителей информации, интеллектуальные пластиковые карты Smart Card помимо энергонезависимой памяти содержат микропроцессор, способный выполнять криптографические преобразования информации. Поэтому интеллектуальные карты способны самостоятельно проверять правильность пароля на доступ к ключевой информации, и при аутентификации пользователя с использованием интеллектуальной карты проверку пароля на доступ к карте производит не операционная система, а сама карта. Интеллектуальная карта может быть запрограммирована на стирание хранимой информации после превышения максимально допустимого количества неправильных попыток ввода пароля, что не позволяет подбирать пароль без частого копирования карты, а это весьма дорого. Использование для аутентификации пользователей не только паролей, но еще и внешних носителей информации позволяет заметно повысить защищенность операционной системы. В наибольшей мере защищенность системы повышается при использовании интеллектуальных карт.

ЧАСТЬ 6. ТЕХНОЛОГИИ ЗАЩИТЫ ОТ ВИРУСОВ

Компьютерный вирус - это своеобразное явление, возникшее в процессе развития компьютерной техники и информационных технологий. Суть этого явления состоит в том, что программы-вирусы обладают рядом свойств, присущих живым организмам, - они рождаются, размножаются и умирают. Термин «компьютерный вирус» впервые употребил сотрудник Университета Южной Калифорнии Фред Коэн в 1984 году на 7-й конференции по безопасности информации, проходившей в США. Этим термином был назван вредоносный фрагмент программного кода. Конечно, это была всего лишь метафора. Фрагмент программного кода похож на настоящий вирус не больше, чем человек на робота. И тем не менее это один из тех редких случаев, когда значение метафоры становилось со временем все менее метафорическим и все более буквальным.

Компьютерные вирусы теперь способны делать практически все тоже, что и настоящие вирусы: переходить с одного объекта на другой, изменять способы атаки и мутировать, чтобы проникнуть мимо выставленных против них защитных кордонов. Проникнув в информационную систему, компьютерный вирус может ограничиться безобидными визуальными или звуковыми эффектами, но может и вызвать потерю или искажение данных, утечку личной и конфиденциальной информации. В худшем случае информационная система, пораженная вирусом, окажется под полным контролем злоумышленника. Сегодня компьютерам доверяют решение многих критических задач. Поэтому выход из строя информационных систем может иметь весьма тяжелые последствия, вплоть до человеческих жертв.

Как и в повторяющейся каждый год истории, когда эпидемиологическим центрам приходится гадать от какой разновидности вируса гриппа надо готовить вакцины к середине зимы, появление новых компьютерных вирусов и их «лечение» поставщиками антивирусных средств разделяется определенным интервалом времени. Поэтому организациям и пользователям необходимо знать, что происходит, когда новый, не идентифицированный вирус попадает в сеть организации и персональный компьютер, как быстро антивирусное решение способно оказать помощь и как не допустить распространения этого компьютерного вируса.

6.1. КОМПЬЮТЕРНЫЕ ВИРУСЫ И ПРОБЛЕМЫ АНТИВИРУСНОЙ ЗАЩИТЫ

Существует много определений компьютерного вируса. Исторически первое определение было дано в 1984 году Фредом Коэном: «Компьютерный вирус - это программа, которая может заражать другие программы, модифицируя их посредством включения в них своей, возможно, измененной копии, причем последняя сохраняет способность к дальнейшему размножению». Ключевыми понятиями в этом определении компьютерного вируса являются способность вируса к саморазмножению и способность к модификации вычислительного процесса. Указанные свойства компьютерного вируса аналогичны паразитированию биологического вируса в живой природе. С тех пор острота проблемы вирусов многократно возросла – к концу XX века в мире насчитывалось более 14300 модификаций вирусов. Разнообразие вирусов столь велико, что просто невозможно указать достаточное условие (перечислить набор признаков, при выполнении

которых программу можно однозначно отнести к вирусам) – всегда найдутся программы с данными признаками, не являющиеся вирусами.

В настоящее время под компьютерным вирусом принято понимать программный код, обладающий следующими свойствами:

1) Способностью к созданию собственных копий, необязательно совпадающих с оригиналом, но обладающих свойствами оригинала (самовоспроизведение).
2) Наличием механизма, обеспечивающего внедрение создаваемых копий в исполняемые объекты вычислительной системы.

Следует отметить, что эти свойства являются необходимыми, но не достаточными. Указанные свойства необходимо дополнить свойствами деструктивности и скрытности действий данной вредоносной программы в вычислительной среде.

6.1.1. КЛАССИФИКАЦИЯ КОМПЬЮТЕРНЫХ ВИРУСОВ

На сегодняшний день известны десятки тысяч различных компьютерных вирусов. Несмотря на такое изобилие число типов вирусов, отличающихся друг от друга механизмом распространения и принципом действия, достаточно ограничено. Существуют и комбинированные вирусы, которые можно отнести одновременно к нескольким типам. Вирусы можно разделить на классы по следующим основным признакам:

- среда обитания;
- операционная система (ОС);
- особенности алгоритма работы;
- деструктивные возможности.

Основной и наиболее распространенной классификацией компьютерных вирусов является классификация по среде обитания, или, иначе говоря, по типам объектов компьютерной системы, в которые внедряются вирусы (рис. 6.1). По среде обитания компьютерные вирусы можно разделить на:

- файловые;
- загрузочные;
- макровирусы;
- сетевые.

Файловые вирусы либо внедряются в выполняемые файлы (наиболее распространенный тип вирусов) различными способами, либо создают файлы-двойники (компаньон-вирусы), либо используют особенности организации файловой системы (link-вирусы).

Загрузочные вирусы записывают себя либо в загрузочный сектор диска (boot-сектор), либо в сектор, содержащий системный загрузчик винчестера (Master Boot Record). Загрузочные вирусы замещают код программы, получающей управление при загрузке системы. В результате при перезагрузке управление передается вирусу. При этом оригинальный boot-сектор обычно переносится в какой-либо другой сектор диска. Иногда загрузочные вирусы называют бутовыми вирусами.

Макровирусы заражают макропрограммы и файлы документов современных систем обработки информации, в частности файлы-документы и электронные таблицы популярных редакторов Microsoft Word, Microsoft Excel и др. Для размножения макровирусы используют возможности макроязыков и при их помощи переносят себя из одного зараженного файла в другие. Вирусы этого типа получают

управление при открытии зараженного файла и инфицируют файлы, к которым впоследствии идет обращение из соответствующего офисного приложения.

Сетевые вирусы используют для своего распространения протоколы или

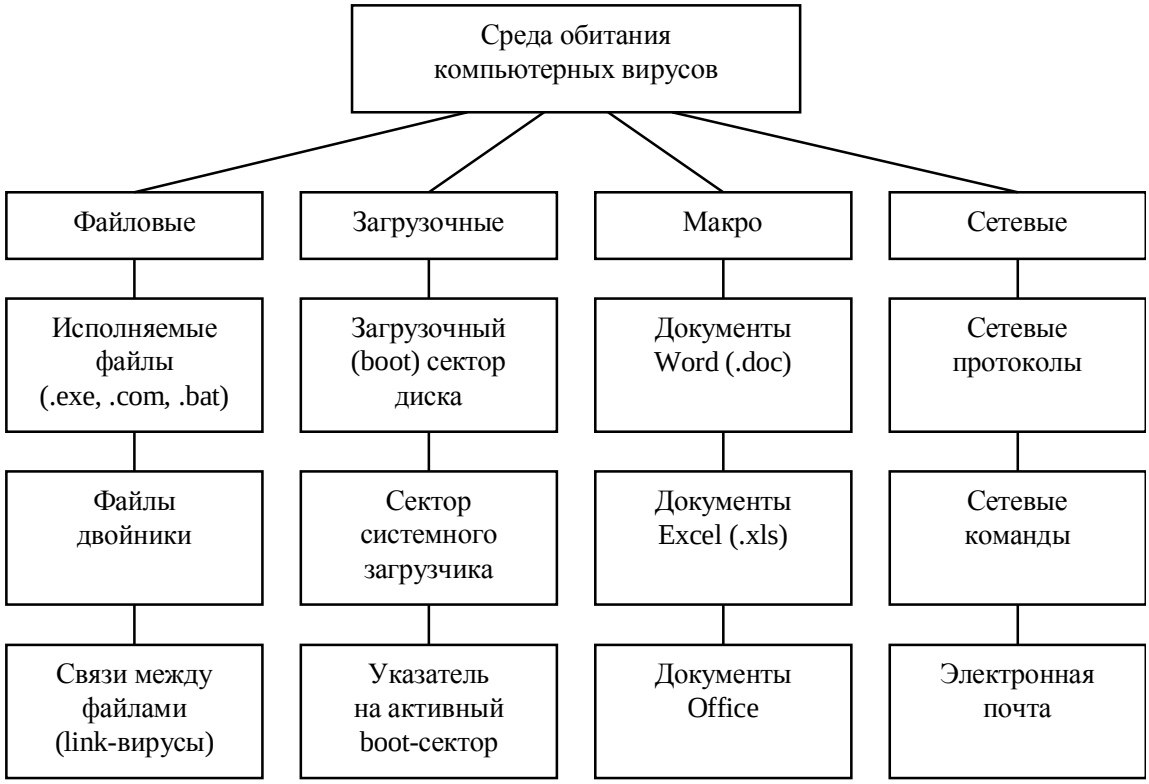


Рис. 6.1. Классификация компьютерных вирусов по среде обитания

команды компьютерных сетей и электронной почты. Иногда сетевые вирусы называют программами типа «червь». Сетевые черви подразделяются на интернет-черви (распространяются по Интернету), LAN-черви (распространяются по локальной сети), IRC-черви (Internet Relay Chat – распространяются через чаты). Существуют также смешанные типы, которые совмещают в себе сразу несколько технологий.

Существует много комбинированных типов компьютерных вирусов, например известен сетевой макровирус, который заражает редактируемые документы, а также рассылает свои копии по электронной почте. В качестве другого примера вирусов комбинированного типа можно указать файлово-загрузочные вирусы, заражающие как файлы, так и загрузочные сектора дисков. Такие вирусы имеют усложненный алгоритм работы и применяют своеобразные методы проникновения в систему.

Другим признаком деления компьютерных вирусов на классы является операционная система, объекты которой подвергаются заражению. Каждый файловый или сетевой вирус заражает файлы какой-либо одной или нескольких операционных систем – DOS, Windows 95/98, Windows NT/2000 и т.д. Макровирусы заражают файлы форматов Word, Excel и других приложений пакета Microsoft Office. На определенные форматы расположения системных данных в загрузочных секторах дисков также ориентированы загрузочные вирусы.

Естественно, эти схемы классификации не являются единственно возможными, существует много различных схем типизации вирусов. Однако ограничимся пока классификацией компьютерных вирусов по среде обитания, поскольку она является базовой, и перейдем к рассмотрению общих принципов функционирования вирусов. Анализ основных этапов «жизненного цикла» этих вредоносных программ позволяет выделить их различные признаки и особенности, которые могут быть положены в основу дополнительных классификаций.

6.1.2. ЖИЗНЕННЫЙ ЦИКЛ ВИРУСОВ

Как и у любой программы, у компьютерных вирусов можно выделить две основные стадии жизненного цикла – хранение и исполнение.

Стадия хранения соответствует периоду, когда вирус просто хранится на диске совместно с объектом, в который он внедрен. На этой стадии вирус является наиболее уязвимым со стороны антивирусного программного обеспечения, так как он не активен и не может контролировать работу операционной системы с целью самозащиты.

Некоторые вирусы на этой стадии используют механизмы защиты своего кода от обнаружения. Наиболее распространенным способом защиты является шифрование большей части тела вируса. Его использование совместно с механизмами мутации кода делает невозможным выделение сигнатур – устойчивых характеристических фрагментов кода вирусов.

Стадия исполнения компьютерных вирусов, как правило, включает пять этапов:

1) Загрузка вируса в память.
2) Поиск жертвы.
3) Заражение найденной жертвы.
4) Выполнение деструктивных функций.
5) Передача управления программе-носителю вируса.

Рассмотрим эти этапы подробнее.

Загрузка вируса в память. Загрузка вируса в память осуществляется операционной системой одновременно с загрузкой исполняемого объекта, в который вирус внедрен. Например, если пользователь запустил на исполнение программный файл, содержащий вирус, то, очевидно, вирусный код будет загружен в память как часть этого файла. В простейшем случае процесс загрузки вируса представляет собой не что иное как копирование с диска в оперативную память, сопровождаемое иногда настройкой адресов, после чего происходит передача управления коду тела вируса. Эти действия выполняются операционной системой, а сам вирус находится в пассивном состоянии. В более сложных ситуациях вирус может после получения управления выполнять дополнительные действия, которые необходимы для его функционирования. В связи с этим рассматриваются два аспекта.

Первый из них связан с максимальным усложнением процедуры обнаружения вирусов. Для обеспечения защиты на стадии хранения некоторые вирусы используют достаточно сложные алгоритмы. К таким усложнениям можно отнести шифрование основного тела вируса. Однако использование только шифрования является полумерой, так как в открытом виде должна храниться та часть вируса, которая обеспечивает дешифрование вируса на стадии загрузки. Для избежания подобной ситуации разработчики вирусов используют механизмы «мутаций» кода

расшифровщика. Суть этого метода состоит в том, что при внедрении в объект копии вируса часть ее кода, относящаяся к расшифровщику, модифицируется так, чтобы возникли текстуальные различия с оригиналом, но результаты работы остались неизменными. Обычно применяют следующие приемы модификации кода:

- изменение порядка независимых инструкций;
- замену некоторых инструкций на эквивалентные по результату работы;
- замену используемых в инструкциях регистров на другие;
- введение случайным образом зашумляющих инструкций.

Вирусы, использующие подобные механизмы мутации кода, получили название полиморфных вирусов. При совместном использовании механизмов шифрования и мутации внедряемая копия вируса окажется отличной от оригинала, так как одна ее часть будет изменена, а другая окажется зашифрованной на ключе, сгенерированном специально для этой копии вируса. А это существенно осложняет выявление вируса в вычислительной системе.

Полиморфные вирусы (polymorphic) – это трудно обнаруживаемые вирусы, не имеющие сигнатур, то есть не содержащие ни одного постоянного участка кода. В большинстве случаев два образца одного и того же полиморфного вируса не будут иметь ни одного совпадения. Полиморфизм встречается в вирусах всех типов – файловых, загрузочных и макровирусах.

Дополнительные действия, которые выполняют полиморфные вирусы на этапе загрузки, состоят в дешифровании основного тела вируса.

При использовании стелс-алгоритмов вирусы могут полностью или частично скрыть себя в системе. Наиболее распространенный стелс-алгоритм осуществляет перехват системных запросов с целью контроля действий операционной системы. Вирусы, использующие стелс-алгоритмы, носят название «стелс-вирусы».

Стелс-вирусы (Stealth) способны скрывать свое присутствие в системе и избегать обнаружения антивирусными программами. Эти вирусы могут перехватывать запросы операционной системы на чтение/запись зараженных файлов, при этом они либо временно лечат эти файлы, либо «подставляют» вместо себя незараженные участки информации, эмулируя чистоту зараженных файлов.

В случае макровирусов наиболее популярным способом является запрет вызовов меню просмотра макросов. Одним из первых файловых стелс-вирусов был вирус *Frodo*, первым загрузочным стелс-вирусом был вирус *Brain*.

Нередко в вирусах используются различные нестандартные приемы с целью глубже спрятаться в ядре ОС, либо защитить от обнаружения свою резидентную копию, либо затруднить лечение от вируса и т.п.

Второй аспект связан с так называемыми резидентными вирусами. Поскольку вирус и объект, в который он внедрен, являются для операционной системы единым целым, то после загрузки они располагаются, естественно, в едином адресном пространстве. После завершения работы объекта он выгружается из оперативной памяти, при этом одновременно выгружается и вирус, переходя в пассивную стадию хранения. Однако некоторые типы вирусов способны сохраняться в памяти и оставаться активными после окончания работы вирусоносителя. Эти вирусы получили название резидентных.

Резидентные вирусы при инфицировании компьютера оставляют в оперативной памяти свою резидентную часть, которая затем перехватывает обращения операционной системы к объектам заражения и внедряется в них.

Резидентные вирусы находятся в памяти и являются активными вплоть до выключения компьютера или перезагрузки операционной системы.

Резидентными можно считать макровирусы, так как для большинства из них выполняются основные требования - постоянное присутствие в памяти компьютера на все время работы зараженного редактора и перехват функций, используемых при работе с документами. При этом роль операционной системы берет на себя редактор, а понятие «перезагрузка операционной системы» трактуется как выход из редактора.

Нерезидентные вирусы не заражают память компьютера и сохраняют активность ограниченное время. Некоторые вирусы оставляют в оперативной памяти небольшие резидентные программы, которые не распространяют вирус. Такие вирусы считаются нерезидентными.

Следует отметить, что деление вирусов на резидентные и нерезидентные справедливо в основном для файловых вирусов. Загрузочные вирусы, как и макровирусы, относятся к резидентным вирусам.

Поиск жертвы. По способу поиска жертвы вирусы можно разделить на два класса. К первому относятся вирусы, осуществляющие активный поиск с использованием функций операционной системы. Примером являются файловые вирусы, использующие механизм поиска исполняемых файлов в текущем каталоге.

Второй класс составляют вирусы, реализующие пассивный механизм поиска, то есть вирусы, расставляющие «ловушки» для программных файлов. Как правило, файловые вирусы устраивают такие ловушки путем перехвата функции *exec* операционной системы, а макровирусы – с помощью перехвата команд типа *Save as* (Сохранить как) из меню *File* (Файл).

Заражение найденной жертвы. В простейшем случае заражение представляет собой самокопирование кода вируса в выбранный в качестве жертвы объект. Классификация вирусов на этом этапе связана с анализом особенностей этого копирования, а также способов модификации заражаемых объектов.

Рассмотрим сначала особенности заражения файловыми вирусами.

По способу инфицирования жертвы вирусы можно разделить на два класса.

К первому относятся вирусы, которые не внедряют свой код непосредственно в программный файл, а изменяют имя файла и создают новый, содержащий тело вируса.

Второй класс составляют вирусы, внедряющиеся непосредственно в файлы-жертвы. Они характеризуются местом внедрения. Возможны следующие варианты.

1) Внедрение в начало файла. Этот способ является наиболее удобным для COM-файлов MS-DOS, так как данный формат не предусматривает наличия служебных заголовков. При внедрении данным способом вирусы могут либо производить конкатенацию собственного кода и кода программы-жертвы, либо переписывать начальный фрагмент файла в конец, освобождая место для себя.
2) Внедрение в конец файла. Это наиболее распространенный тип внедрения. Передача управления коду вирусов обеспечивается модификацией первых команд программы (COM) или заголовка файла (EXE).
3) Внедрение в середину файла. Как правило, этот способ используется вирусами применительно к файлам с заранее известной структурой (например, к файлу COMMAND.COM) или же к файлам, содержащим последовательность байтов с одинаковыми значениями, длина которой

достаточна для размещения вируса. Во втором случае вирусы архивируют найденную последовательность и замещают ее собственным кодом. Помимо этого вирусы могут внедряться в середину файла, освобождая себе место путем переноса фрагментов кода программы в конец файла или же «раздвигая» файл.

Для загрузочных вирусов особенности этапа заражения определяются особенностями объектов, в которые они внедряются, – загрузочными секторами гибких и жестких дисков и главной загрузочной записью (MBR) жестких дисков. Основной проблемой является ограниченный размер этих объектов. В связи с этим вирусам необходимо сохранить на диске ту свою часть, которая не уместилась на месте жертвы, а также перенести оригинальный код инфицированного загрузчика. Существуют различные способы решения этой задачи. Ниже приводится классификация, предложенная Е. Касперским:

1) Используются псевдосбойные сектора. Вирус переносит необходимый код в свободные сектора диска и помечает их как сбойные, защищая тем самым себя и загрузчик от перезаписи.
2) Используются редко применяемые сектора в конце раздела. Вирус переносит необходимый код в эти свободные сектора в конце диска. С точки зрения операционной системы эти сектора выглядят как свободные.
3) Используются зарезервированные области разделов. Вирус переносит необходимый код в области диска, зарезервированные под нужды операционной системы, а потому - неиспользуемые.
4) Короткие вирусы могут уместиться в один сектор загрузчика и полностью взять на себя функции MBR или загрузочного сектора.

Для макровирусов процесс заражения сводится к сохранению вирусного макрокода в выбранном документе-жертве. Для некоторых систем обработки информации это сделать не совсем просто, так как формат файлов документов может не предусматривать возможность сохранения макропрограмм. В качестве примера приведем Microsoft Word 6.0. Сохранение макрокода для этой системы возможно только в файлах шаблонов (имеющих по умолчанию расширение .DOT). Поэтому для своего сохранения вирус должен контролировать обработку команды *Save as* из меню *File*, которая вызывается всякий раз, когда происходит первое сохранение документа на диск. Этот контроль необходим, чтобы в момент сохранения изменить тип файла-документа (имеющего по умолчанию расширение .DOC) на тип файла-шаблона. В этом случае на диске окажутся и макрокод вируса, и содержимое документа.

Помимо простого копирования кода вируса в заражаемый объект на этом этапе могут использоваться более сложные алгоритмы, обеспечивающие защиту вируса на стадии хранения. К числу таких вирусов относятся описанные выше полиморфные вирусы.

Выполнение деструктивных функций. Вирусы могут выполнять помимо самокопирования деструктивные функции.

По деструктивным возможностям вирусы можно разделить на безвредные, неопасные, опасные и очень опасные.

Безвредные вирусы – это вирусы, в которых реализован только механизм самораспространения. Они не наносят вред системе, за исключением расхода свободной памяти на диске в результате своего распространения.

Неопасные вирусы – это вирусы, присутствие которых в системе связано с различными эффектами (звуковыми, видео) и уменьшением свободной памяти на диске, но которые не наносят вред программам и данным.

Опасные вирусы – это вирусы, которые могут привести к серьезным сбоям в работе компьютера. Последствием сбоя может стать разрушение программ и данных.

Очень опасные вирусы – это вирусы, в алгоритм работы которых заведомо заложены процедуры, непосредственно приводящие к разрушению программ и данных, а также к стиранию информации, записанной в системных областях памяти и необходимой для работы компьютера.

На степень опасности вирусов оказывает существенное влияния та среда, под управлением которой вирусы работают.

Так, вирусы, созданные для работы в MS-DOS, обладают практически неограниченными потенциальными возможностями.

Распространение вирусов под управлением Windows NT/2000 ограничивается развитой системой разграничения доступа.

Возможности макровирусов напрямую определяются возможностями макроязыков, на которых они написаны. В частности, язык Word Basic позволяет создать мощные макровирусы, способные доставить пользователям серьезные неприятности.

Дополняя эту классификацию, можно отметить также деление вирусов на наносящие вред системе вообще и предназначенные для целенаправленных атак на определенные объекты.

Передача управления программе-носителю вируса. Здесь следует указать на деление вирусов на разрушающие и неразрушающие.

Разрушающие вирусы не заботятся о том, чтобы при инфицировании программ сохранять их работоспособность, поэтому для них этот этап функционирования отсутствует.

Для неразрушающих вирусов этот этап связан с восстановлением в памяти программы в том виде, в котором она должна корректно исполняться, и передачей управления программе-носителю вируса.

Вредоносные программы других типов

Кроме вирусов принято выделять еще несколько видов вредоносных программ. Это «троянские» программы; логические бомбы; хакерские утилиты скрытого администрирования удаленных компьютеров; программы, ворующие пароли доступа к ресурсам Интернета и прочую конфиденциальную информацию. Четкого разделения между ними не существует: «троянские» программы могут содержать вирусы, в вирусы могут быть встроены логические бомбы и т.д.

«Троянские» программы не размножаются и не рассылаются сами. Внешне «троянские» программы выглядят совершенно безобидно и даже предлагают полезные функции. Но когда пользователь загрузит такую программу в свой компьютер и запустит ее, она может незаметно выполнять вредоносные функции. Чаще всего «троянские» программы используются для первоначального распространения вирусов, для получения удаленного доступа к компьютеру через Интернет, кражи данных или их уничтожения.

Логической бомбой называется программа или ее отдельные модули, которые при определенных условиях выполняют вредоносные действия. Логическая бомба может, например, сработать по достижении определенной даты или тогда, когда в

базе данных появится или исчезнет запись, и т.п. Такая бомба может быть встроена в вирусы, «троянские» программы и даже в обычные программы.

6.1.3. ОСНОВНЫЕ КАНАЛЫ РАСПРОСТРАНЕНИЯ ВИРУСОВ И ВРЕДОНОСНЫХ ПРОГРАММ

Для того чтобы создать эффективную систему антивирусной защиты компьютеров и корпоративных сетей, необходимо четко представлять себе, откуда грозит опасность. Вирусы находят самые разные каналы распространения, причем к старым способам постоянно добавляются новые.

Классические способы распространения

Файловые вирусы распространяются вместе с файлами программ в результате обмена дискетами и программами, загрузки программ из сетевых каталогов, с Web- или FTP-серверов. Загрузочные вирусы попадают на компьютер, когда пользователь забывает зараженную дискету в дисководе, а затем перезагружает ОС. Загрузочный вирус также может быть занесен на компьютер вирусами других типов. Макрокомандные вирусы распространяются в результате обмена зараженными файлами офисных документов, такими как файлы Microsoft Word, Excel, Access.

Если зараженный компьютер подключен к локальной сети, вирус легко может оказаться на дисках файл-сервера, а оттуда через каталоги, доступные для записи, попасть на все остальные компьютеры сети. Так начинается вирусная эпидемия. Системному администратору следует помнить, что вирус имеет в сети такие же права, что и пользователь, на компьютер которого этот вирус пробрался. Поэтому он может попасть во все сетевые каталоги, доступные пользователю. Если же вирус завелся на рабочей станции администратора сети, последствия могут быть очень тяжелыми.

Электронная почта

В настоящее время глобальная сеть Интернет является основным источником вирусов. Большое число заражений вирусами происходит при обмене письмами по электронной почте в форматах Microsoft Word. Электронная почта служит каналом распространения макрокомандных вирусов, так как вместе с сообщениями часто отправляются офисные документы.

Заражения вирусами могут осуществляться как непреднамеренно, так и по злому умыслу. Например, пользователь зараженного макровирусом редактора, сам того не подозревая, может рассылать зараженные письма адресатам, которые, в свою очередь, отправляют новые зараженные письма и т.д. С другой стороны, злоумышленник может преднамеренно послать по электронной почте вместе с вложенным файлом исполняемый модуль вирусной или «троянской» программы, вредоносный программный сценарий Visual Basic, зараженную или «троянскую» программу заставки экрана монитора, словом – любой опасный программный код.

Распространители вирусов часто пользуются для маскировки тем фактом, что диалоговая оболочка Microsoft Windows по умолчанию не отображает расширения зарегистрированных файлов. Например, файл с именем FreeCreditCard.txt.exe, будет показан пользователю как FreeCreditCard.txt. Если пользователь попытается открыть такой файл, будет запущена вредоносная программа.

Сообщения электронной почты часто приходят в виде документов HTML, которые могут включать ссылки на элементы управления ActiveX, апплеты Java и другие активные компоненты. Из-за ошибок в почтовых клиентах злоумышленники могут воспользоваться такими активными компонентами для внедрения вирусов и

«троянских» программ на компьютеры пользователей. При получении сообщения в формате HTML почтовый клиент показывает его содержимое в своем окне. Если сообщение содержит вредоносные активные компоненты, они сразу же запускаются и выполняют заложенные в них функции. Чаще всего таким способом распространяются «троянские» программы и черви.

«Троянские» Web-сайты

Пользователи могут получить вирус или «троянскую» программу во время простой навигации по сайтам Интернета, посетив «троянский» Web-сайт. Ошибки в браузерах пользователей зачастую приводят к тому, что активные компоненты «троянских» Web-сайтов (элементы управления ActiveX или апплеты Java) внедряют на компьютеры пользователей вредоносные программы. Здесь используется тот же самый механизм, что и при получении сообщений электронной почты в формате HTML. Но заражение происходит незаметно: активные компоненты Web-страниц могут внешне никак себя не проявлять. Приглашение посетить «троянский» сайт пользователь может получить в обычном электронном письме.

Локальные сети

Локальные сети также представляют собой путь быстрого заражения. Если не принимать необходимых мер защиты, то зараженная рабочая станция при входе в локальную сеть заражает один или несколько служебных файлов на сервере. В качестве таких файлов могут выступать служебный файл LOGIN.COM, Excel-таблицы и стандартные документы-шаблоны, применяемые в фирме. Пользователи при входе в эту сеть запускают зараженные файлы с сервера, и в результате вирус получает доступ на компьютеры пользователей.

Другие каналы распространения вредоносных программ

Одним из серьезных каналов распространения вирусов являются пиратские копии программного обеспечения. Часто нелегальные копии на дискетах и компакт-дисках содержат файлы, зараженные разнообразными типами вирусов. К источникам распространения вирусов следует также отнести электронные конференции и файл-серверы FTP и BBS. Часто авторы вирусов закладывают зараженные файлы сразу на несколько файл-серверов FTP/BBS или рассылают одновременно по нескольким электронным конференциям, причем зараженные файлы обычно маскируют под новые версии программных продуктов и даже антивирусов. Компьютеры, установленные в учебных заведениях и интернет-центрах и работающие в режиме общего пользования, также могут легко оказаться источниками распространения вирусов. Если один из таких компьютеров оказался зараженным вирусом с дискеты очередного пользователя, тогда дискеты и всех остальных пользователей, работающих на этом компьютере, окажутся зараженными.

По мере развития компьютерных технологий совершенствуются и компьютерные вирусы, приспосабливаясь к новым для себя сферам обитания. В любой момент может появиться компьютерный вирус, «троянская» программа или червь нового, неизвестного ранее типа, либо известного типа, но нацеленного на новое компьютерное оборудование. Новые вирусы могут использовать неизвестные или не существовавшие ранее каналы распространения, а также новые технологии внедрения в компьютерные системы. Чтобы исключить угрозу вирусного заражения, системный администратор корпоративной сети должен не только

внедрять методики антивирусной защиты, но и постоянно отслеживать новости в мире компьютерных вирусов.

6.2. АНТИВИРУСНЫЕ ПРОГРАММЫ И КОМПЛЕКСЫ

Для защиты от компьютерных вирусов могут использоваться следующие методы и средства:

- общие методы и средства защиты информации;
- специализированные программы для защиты от вирусов;
- профилактические меры, позволяющие уменьшить вероятность заражения вирусами.

Общие средства защиты информации полезны не только для защиты от вирусов. Они используются также как страховка от физической порчи дисков, неправильно работающих программ или ошибочных действий пользователя.

Существует две основных разновидности этих средств:

- средства копирования информации; применяются для создания копий файлов и системных областей дисков;
- средства разграничения доступа; предотвращают несанкционированное использование информации, в частности обеспечивают защиту от изменений программ и данных вирусами, неправильно работающими программами и ошибочными действиями пользователей.

При заражении компьютера вирусом важно его обнаружить. К внешним признакам проявления деятельности вирусов можно отнести следующие:

- вывод на экран непредусмотренных сообщений или изображений;
- подача непредусмотренных звуковых сигналов;
- изменение даты и времени модификации файлов;
- исчезновение файлов и каталогов или искажение их содержимого;
- частые зависания и сбои в работе компьютера;
- медленная работа компьютера;
- невозможность загрузки операционной системы;
- существенное уменьшение размера свободной оперативной памяти;
- прекращение работы или неправильная работа ранее успешно функционировавших программ;
- изменение размеров файлов;
- неожиданное значительное увеличение количества файлов на диске.

Однако следует заметить, что перечисленные выше явления необязательно вызываются действиями вируса, они могут быть следствием и других причин. Поэтому правильная диагностика состояния компьютера всегда затруднена и обычно требует привлечения специализированных программ.

6.2.1. АНТИВИРУСНЫЕ ПРОГРАММЫ

Для обнаружения и защиты от компьютерных вирусов разработано несколько видов специальных программ, которые позволяют обнаруживать и уничтожать компьютерные вирусы. Такие программы называются антивирусными. Практически все антивирусные программы обеспечивают автоматическое восстановление зараженных программ и загрузочных секторов. Антивирусные программы используют различные методы обнаружения вирусов.

К основным методам обнаружения компьютерных вирусов можно отнести следующие:

- метод сравнения с эталоном;
- эвристический анализ;
- антивирусный мониторинг;
- метод обнаружения изменений;
- встраивание антивирусов в BIOS компьютера и др..

Метод сравнения с эталоном. Самый простой метод обнаружения заключается в том, что для поиска известных вирусов используются так называемые маски. Маской вируса является некоторая постоянная последовательность кода, специфичная для этого конкретного вируса. Антивирусная программа последовательно просматривает (сканирует) проверяемые файлы в поиске масок известных вирусов. Антивирусные сканеры способны найти только уже известные вирусы, для которых определена маска. Если вирус не содержит постоянной маски или длина этой маски недостаточно велика, то используются другие методы. Применение простых сканеров не защищает компьютер от проникновения новых вирусов. Для шифрующихся и полиморфных вирусов, способных полностью изменять свой код при заражении новой программы или загрузочного сектора, невозможно выделить маску, поэтому антивирусные сканеры их не обнаруживают.

Эвристический анализ. Для того чтобы размножаться, компьютерный вирус должен совершать какие-то конкретные действия: копирование в память, запись в сектора и т.д. Эвристический анализатор (который является частью антивирусного ядра) содержит список таких действий и проверяет программы и загрузочные секторы дисков и дискет, пытаясь обнаружить в них код, характерный для вирусов. Эвристический анализатор может обнаружить, например, что проверяемая программа устанавливает резидентный модуль в памяти или записывает данные в исполняемый файл программы. Обнаружив зараженный файл, анализатор обычно выводит сообщение на экране монитора и делает запись в собственном или системном журнале. В зависимости от настроек антивирус может также направлять сообщение об обнаруженном вирусе администратору сети. Эвристический анализ позволяет обнаруживать неизвестные ранее вирусы. Первый эвристический анализатор появился в начале 90-х годов прошлого века. Практически все современные антивирусные программы реализуют собственные методы эвристического анализа. В качестве примера такой программы можно указать сканер McAffee VirusScan.

Антивирусный мониторинг. Суть данного метода состоит в том, что в памяти компьютера постоянно находится антивирусная программа, осуществляющая мониторинг всех подозрительных действий, выполняемых другими программами. Антивирусный мониторинг позволяет проверять все запускаемые программы, создаваемые, открываемые и сохраняемые документы, файлы программ и документов, полученные через Интернет или скопированные на жесткий диск с дискеты либо компакт-диска. Антивирусный монитор сообщит пользователю, если какая-либо программа попытается выполнить потенциально опасное действие. Пример такой программы – сторож Spider Guard, который входит в комплект сканера Dr. Web и выполняет функции антивирусного монитора.

Метод обнаружения изменений. При реализации метода обнаружения изменений антивирусные программы, называемые ревизорами диска, запоминают

предварительно характеристики всех областей диска, которые могут подвергнуться нападению, а затем периодически проверяют их. Заражая компьютер, вирус изменяет содержимое жесткого диска: например, дописывает свой код в файл программы или документа, добавляет вызов программы-вируса в файл AUTOEXEC.BAT, изменяет загрузочный сектор, создает файл-спутник. При сопоставлении значений характеристик областей диска антивирусная программа может обнаружить изменения, сделанные как известным, так и неизвестным вирусом.

Встраивание антивирусов в BIOS компьютера. В системные платы компьютеров тоже встраивают простейшие средства защиты от вирусов. Эти средства позволяют контролировать все обращения к главной загрузочной записи жестких дисков, а также к загрузочным секторам дисков и дискет. Если какая-либо программа пытается изменить содержимое загрузочных секторов, срабатывает защита и пользователь получает соответствующее предупреждение. Однако эта защита не очень надежна. Известны вирусы, которые пытаются отключить антивирусный контроль BIOS, изменяя некоторые ячейки в энергонезависимой памяти (CMOS-памяти) компьютера.

Виды антивирусных программ

Различают следующие виды антивирусных программ:

- программы-фаги (сканеры);
- программы-ревизоры (CRC-сканеры);
- программы-блокировщики;
- программы-иммунизаторы.

Самыми популярными и эффективными антивирусными программами являются программы-фаги (антивирусные сканеры). На втором месте по эффективности и популярности находятся программы-ревизоры (CRC-сканеры). Обычно оба указанных вида программ объединяют в одну универсальную антивирусную программу, что значительно повышает ее мощность. Применяются также различного типа блокировщики и иммунизаторы.

Программы-фаги (сканеры) используют для обнаружения вирусов метод сравнения с эталоном, метод эвристического анализа и некоторые другие методы. Программы-фаги осуществляют поиск характерной для конкретного вируса маски путем сканирования содержимого оперативной памяти и файлов и при обнаружении выдают соответствующее сообщение. Программы-фаги не только находят зараженные вирусами файлы, но и лечат их, то есть удаляют из файла тело программы-вируса, возвращая файлы в исходное состояние. В начале своей работы программы-фаги сканируют оперативную память, обнаруживают вирусы и уничтожают их и только затем переходят к лечению файлов. Среди фагов выделяют полифаги, то есть программы-фаги, предназначенные для поиска и уничтожения большого количества вирусов.

Программы-фаги можно разделить на две категории: универсальные и специализированные сканеры. Универсальные сканеры рассчитаны на поиск и обезвреживание всех типов вирусов вне зависимости от операционной системы, на работу в которой рассчитан сканер. Специализированные сканеры предназначены для обезвреживания ограниченного числа вирусов или только одного их класса, например макровирусов. Специализированные сканеры, рассчитанные только на

макровирусы, оказываются более удобным и надежным решением для защиты систем документооборота в средах MS Word и MS Excel.

Программы-фаги делятся также на резидентные мониторы, производящие сканирование «на лету», и нерезидентные сканеры, обеспечивающие проверку системы только по запросу. Резидентные мониторы обеспечивают более надежную защиту системы, поскольку они немедленно реагируют на появление вируса, в то время как нерезидентный сканер способен опознать вирус только во время своего очередного запуска.

К достоинствам программ-фагов всех типов относится их универсальность. К недостаткам программ-фагов следует отнести относительно небольшую скорость поиска вирусов и относительно большие размеры антивирусных баз.

Наиболее известные из программ-фагов: Aidstest, Scan, Norton AntiVirus, Dr. Web. Учитывая, что постоянно появляются новые вирусы, программы-фаги быстро устаревают и требуется регулярное обновление версий.

Программы-ревизоры (CRC-сканеры) используют для поиска вирусов метод обнаружения изменений. Принцип работы CRC-сканеров основан на подсчете CRC-сумм (кодов циклического контроля) для присутствующих на диске файлов/системных секторов. Затем в базе данных антивируса сохраняются эти CRC-суммы, а также некоторая другая информация: длины файлов, даты их последней модификации и другие параметры. При последующем запуске CRC-сканеры сверяют данные, содержащиеся в базе данных, с реально подсчитанными значениями. Если информация о файле, записанная в базе данных, не совпадает с реальными значениями, то CRC-сканеры сигнализируют о том, что файл был изменен или заражен вирусом. Как правило, сравнение состояний производят сразу после загрузки операционной системы.

CRC-сканеры, использующие антистелс-алгоритмы, являются довольно мощным средством против вирусов: практически 100% вирусов оказываются обнаруженными почти сразу после их появления на компьютере. Однако у CRC-сканеров имеется недостаток, заметно снижающий их эффективность. CRC-сканеры не могут определить вирус в новых файлах (в электронной почте, на дискетах, в файлах, восстанавливаемых из резервной копии или распаковываемых из архива), поскольку в их базах данных отсутствует информация об этих файлах.

К числу CRC-сканеров относится широко распространенная в России программа Adinf (Advanced Diskinfoscope) и ревизор AVP Inspector. Вместе с ADinf применяется лечащий модуль ADinf Cure Module (ADinfExt), который использует собранную ранее информацию о файлах для их восстановления после поражения неизвестными вирусами. В состав ревизора AVP Inspector также входит лечащий модуль, способный удалять вирусы.

Программы-блокировщики реализуют метод антивирусного мониторинга. Антивирусные блокировщики – это резидентные программы, перехватывающие вирусоопасные ситуации и сообщающие об этом пользователю. К вирусоопасным ситуациям относятся вызовы на открытие для записи в выполняемые файлы, запись в загрузочные сектора дисков или MBR жесткого диска, попытки программ остаться в памяти резидентно и т.п., то есть вызовы, которые характерны для вирусов в моменты их размножения.

При попытке какой-либо программы произвести указанные действия блокировщик посылает пользователю сообщение и предлагает запретить соответствующее действие. К достоинствам блокировщиков относится их

способность обнаруживать и останавливать вирус на самой ранней стадии его размножения, что бывает особенно полезно в случаях, когда регулярно появляется давно известный вирус. Однако они не лечат файлы и диски. Для уничтожения вирусов требуется применить другие программы, например фаги. К недостаткам блокировщиков можно отнести существование путей обхода их защиты и их «назойливость» (например, они постоянно выдают предупреждение о любой попытке копирования исполняемого файла).

Следует отметить, что созданы антивирусные блокировщики, выполненные в виде аппаратных компонентов компьютера. Наиболее распространенной является встроенная в BIOS защита от записи в MBR жесткого диска.

Программы-иммунизаторы – это программы, предотвращающие заражение файлов. Иммунизаторы делятся на два типа: сообщающие о заражении и блокирующие заражение каким-либо типом вируса. Иммунизаторы первого типа обычно записываются в конец файлов и при запуске файла каждый раз проверяют его на изменение. У таких иммунизаторов имеется один серьезный недостаток: они не могут обнаружить заражение стелс-вирусом. Поэтому этот тип иммунизаторов практически не используется в настоящее время.

Иммунизатор второго типа защищает систему от поражения вирусом определенного вида. Этот иммунизатор модифицирует программу или диск таким образом, чтобы это не отражалось на их работе, а вирус будет воспринимать их зараженными и поэтому не внедрится. Такой тип иммунизации не может быть универсальным, поскольку нельзя иммунизировать файлы от всех известных вирусов. Однако подобные иммунизаторы могут в качестве полумеры вполне надежно защитить компьютер от нового неизвестного вируса вплоть до того момента, когда он будет определяться антивирусными сканерами.

Критерии качества антивирусной программы

Качество антивирусной программы можно оценить по нескольким критериям. Эти критерии, перечисленные в порядке убывания их важности, следующие:

- надежность и удобство работы – отсутствие «зависаний» антивируса и прочих технических проблем, требующих от пользователя специальной подготовки;
- качество обнаружения вирусов всех распространенных типов, сканирование внутри файлов-документов/таблиц (MS Word, Excel и других приложений из пакета Office), архивированных файлов. Возможность лечения заражен ных объектов;
- существование версий антивируса под все популярные платформы (DOS, Windows, Novell NetWare, OS/2, Alpha, Linux и т.д.); наличие режимов сканирования по запросу и сканирования «на лету», существование серверных версий с возможностью администрирования сети;
- скорость работы и другие полезные особенности.

Надежность работы антивируса является наиболее важным критерием, поскольку даже идеальный антивирус может оказаться бесполезным, если он будет не в состоянии довести процесс сканирования до конца, то есть «повиснет» и не проверит часть дисков и файлов и в результате вирус останется незамеченным в системе.

Качество обнаружения вирусов стоит на следующем месте по вполне естественной причине. Главная обязанность антивирусных программ –

обнаруживать 100% вирусов и лечить их. При этом антивирусная программа не должна иметь высокий уровень ложных срабатываний.

Следующим по важности критерием является многоплатформенность антивируса, поскольку только программа, рассчитанная на конкретную операционную систему, может полностью использовать функции этой системы. Достаточно важным свойством антивируса является также возможность проверки файлов «на лету». Моментальная и принудительная проверка приходящих на компьютер файлов и вставляемых дискет является практически 100-процентной гарантией от заражения вирусом. Если в серверном варианте антивируса присутствуют возможность антивирусного администрирования сети, то его ценность еще более возрастает.

Скорость работы также является важным критерием качества антивирусной программы. В разных антивирусах используются различные алгоритмы поиска вирусов, один алгоритм может оказаться более быстрым и качественным, другой – медленным и менее качественным.

Профилактические меры защиты

Своевременное обнаружение зараженных вирусами файлов и дисков, полное уничтожение обнаруженных вирусов на каждом компьютере позволяют избежать распространения вирусной эпидемии на другие компьютеры. Абсолютно надежных программ, гарантирующих обнаружение и уничтожение любого вируса, не существует. Важным методом борьбы с компьютерными вирусами является своевременная профилактика. Для того чтобы существенно уменьшить вероятность заражения вирусом и обеспечить надежное хранение информации на дисках, необходимо выполнять следующие меры профилактики:

- применять только лицензионное программное обеспечение;
- оснастить свой компьютер современными антивирусными программами, на пример Aidstest, AVP, Dr. Web, и постоянно обновлять их версии;
- перед считыванием с дискет информации с других компьютеров всегда проверять эти дискеты на наличие вирусов, запуская антивирусные программы своего компьютера;
- при переносе на свой компьютер файлов в архивированном виде проверять их сразу же после разархивации на жестком диске, ограничивая область проверки только вновь записанными файлами;
- периодически проверять на наличие вирусов жесткие диски компьютера, запуская антивирусные программы для тестирования файлов, памяти и системных областей дисков с защищенной от записи дискеты, предварительно загрузив операционную систему с защищенной от записи системной дискеты;
- всегда защищать свои дискеты от записи при работе на других компьютерах, если на них не будет производиться запись информации;
- обязательно делать архивные копии на дискетах ценной для пользователя информации;
- не оставлять в кармане дисковода А дискеты при включении или перезагрузке операционной системы, чтобы исключить заражение компьютера загрузочными вирусами;
- использовать антивирусные программы для входного контроля всех исполняемых файлов, получаемых из компьютерных сетей;

– для обеспечения большей безопасности сочетать применение Aidstest и Dr. Web с повседневным использованием ревизора диска Adinf.

У каждого типа антивирусных программ есть свои достоинства и недостатки. Только комплексное использование нескольких типов антивирусных программ может привести к приемлемому результату. Программные средства защиты информации представляют собой комплекс алгоритмов и программ специального и общего обеспечения функционирования компьютеров и вычислительных сетей, нацеленных на контроль, разграничение доступа и исключение проникновения несанкционированной информации. Это наиболее распространенные методы защиты информации. Они обладают универсальностью, простотой реализации, гибкостью, адаптивностью и др.

6.2.2. АНТИВИРУСНЫЕ ПРОГРАММНЫЕ КОМПЛЕКСЫ

Существует целый спектр программных комплексов, предназначенных для профилактики заражения вирусом, обнаружения и уничтожения вирусов.

Антивирус Касперского (AVP) Personal

Этот российский антивирусный пакет – один из лидеров антивирусной индустрии.

В состав пакета входят: поведенческий блокиратор Office Guard – обеспечивает 100-процентную защиту от макровирусов; ревизор Inspector – отслеживает все изменения, происходящие на компьютере, и при обнаружении вирусной активности позволяет восстановить оригинальное содержимое диска и удалить вредоносные коды; фоновый перехватчик вирусов Monitor – постоянно присутствует в памяти компьютера и проводит антивирусную проверку всех файлов в момент их запуска, создания или копирования. Это позволяет программе полностью контролировать все файловые операции и предотвращать заражение даже самыми технологически совершенными вирусами; антивирусный модуль Scanner дает возможность проводить полномасштабную проверку всего содержимого локальных и сетевых дисков. Можно запустить сканер вручную или автоматически в заданное время.

В пакете реализована уникальная технология поиска неизвестных вирусов благодаря эвристическому анализатору второго поколения. С его помощью программа способна защитить компьютер от неизвестных вирусов. Кроме того, осуществляется постоянная антивирусная фильтрация электронной почты и комплексная проверка почтовой корреспонденции, имеется перехватчик вирусов для MS Office и система перехвата вирусов-сценариев, поддержка архивированных и компрессированных файлов. Обновление антивирусной базы осуществляется через Интернет. Антивирус Касперского (AVP) обеспечивает антивирусный контроль на платформах DOS, Windows 95/98/NT/2000/XP, NetWare, Linux, FreeBSD, BSDi. Также поддерживаются Microsoft Exchange, Microsoft Office, CheckPoint FireWall-1, почтовые сервисы UNIX – sendmail и qmail. Компания предоставляет возможность ежедневного обновления пакета через Интернет. Продукты Лаборатории Касперско-го являются хорошим решением для небольших офисов, а также компаний, широко использующих в своей работе продукты Linux и FreeBSD.

Антивирус Dr. Web

Популярная российская антивирусная программа для платформ Windows 9x/NT/2000/XP предназначена для поиска и обезвреживания файловых, загрузочных

и файлового загрузочных вирусов. Программа включает в себя резидентный сторож SpIDer Guard, автоматическую систему получения обновлений вирусных баз через Интернет и планировщик расписания автоматических проверок. Реализована проверка почтовых файлов. Кроме того, программа обнаруживает вирусы внутри архивов, вакцинированных файлов, файлов документов MS Word и Excel. В настоящий момент вирусная база содержит более 28 тыс. записей, но это не значит, что она менее полная, чем у других программ, – просто в программе Dr. Web одной записью в базе может определяться до нескольких сотен вирусов.

Существенной особенностью программы Dr. Web, выделяющей ее среди других, является использование оригинального эвристического анализатора наряду с традиционным методом обнаружения вирусов по их сигнатурам. Использование эвристического анализатора позволяет выявлять вирусы, сигнатуры которых еще неизвестны. Алгоритмы, используемые в Dr. Web, позволяют выявлять все известные в настоящее время типы вирусов. Другая существенная особенность программы Dr. Web – использование эмулятора процессора, что позволяет обнаруживать сложные шифрованные и полиморфные вирусы, для которых в принципе не работает обычный сигнатурный поиск.

Антивирус Norton AntiVirus от Symantec

Norton AntiVirus - набор антивирусных продуктов компании Symantec, предлагаемый корпоративным пользователям. Объединяет все антивирусные продукты Symantec – для серверов Windows NT и Novell, рабочих станций, коммуникационных пакетов Lortus Notes и MS Exchange, SMTP почтовых серверов и брандмауэров, а также включает управляющую консоль Symantec System Center.

Применение продуктов Symantec целесообразно при общем количестве рабочих мест не менее 100 и наличии хотя бы одного сервера Windows NT/2000 или NetWare.

Отличительными особенностями данного пакета являются:

- иерархическая модель управления;
- наличие механизма реакции на возникновение новых вирусов.

Программа Norton AntiVirus предназначена для обнаружения и обезвреживания вирусов, злонамеренных программ ActiveX, апплетов Java. Как и все современные антивирусные пакеты, содержит сканер и монитор. Реализована новая технология эвристического анализа Bloodhound. Производится автоматическое сканирование электронной почты (MS Outlook, MS Outlook Express, Eudora Pro, Eudora Lite, Netscape Messenger, Netscape Mail). Поддерживается функция **Script Blocking** (Блокировка сценариев), которая постоянно проверяет сценарии и оповещает пользователя о наличии вредоносной программы, останавливая и обезвреживая вирус до того, как он распространится и заразит файлы. Программа осуществляет обнаружение и лечение вирусов в сжатых файлах (MIME/UU, LHA/LZH, ARJ, CAB, PKLite, LZEXE, ZIP). LiveUpdate проверяет наличие обновлений к базам данных по вирусам при загрузке системы (с центрального сервера), автоматически скачивает и устанавливает последние версии на вашу систему.

Антивирус McAfee

Антивирус McAfee Active Virus Defense охватывает все операционные системы и групповые приложения, используемые в современных корпоративных сетях: клиентские ОС Windows 3x/95/98/ME/NT Workstation/2000 Professional/XP,

OS/2, DOS, Macintosh; серверные ОС Windows NT Server, Windows 2000 Server/Advanced Server/Datacenter, Novell Netware, FreeBSD, Linux, HP-UX, AIX, SCO, Solaris; групповые приложения MS Exchange и Lotus Notes/Domino; интернетшлюзы MS Proxy Server; ОС микрокомпьютеров (PDA) Windows CE, Palm OS, Pocket PC, EPOC (Psion). Является хорошим решением на уровне почтовых шлюзов а также для платформы HP-UX. Целесообразно применять при количестве рабочих мест более 500.

Антивирус AntiVir Personal Edition

Эта антивирусная программа обладает почти такими же возможностями, как Dr. Web, AV P и другие антивирусные программы. В комплект поставки входят: сканер дисков, резидентный сторож, программа управления, планировщик. Программа сканирует файлы, загружаемые из Интернета. Продукт бесплатен для частного некоммерческого использования и выпускается в двух вариантах – для Windows 9x и Windows NT/2000/XP.

Большая антивирусная база (более 40 тыс. записей) обновляется раз в неделю и доступна на сайте производителя. Есть также функция автоматической проверки и загрузки обновлений через Интернет. Поддерживается технология drag-and-drop (перетаскивание мышью): любой файл, архив, каталог, перенесенные в основное окно программы, тут же будут проверены. Программа проверяет память, загрузочные сектора. Имеется обширный справочник по вирусам. Для коммерческого использования доступна версия AntiVir Professional с расширенным набором функций, работающая с различными операционными системами.

6.3. ПОСТРОЕНИЕ СИСТЕМЫ АНТИВИРУСНОЙ ЗАЩИТЫ СЕТИ

В настоящее время проблема антивирусной защиты является одной из приоритетных проблем безопасности корпоративных информационных ресурсов организации. Актуальность данной проблемы объясняется следующими причинами:

- лавинообразный рост числа компьютерных вирусов. Данные независимых отчетов свидетельствуют о том, что средний уровень заражения вирусами корпоративных компьютерных сетей увеличился с 55% в 1995 году, до 99,9% в 2001 году;
- неудовлетворительное состояние антивирусной защиты в существующих корпоративных компьютерных сетях. Сегодня сети компаний находятся в постоянном развитии. Однако вместе с этим развитием постоянно растет и число точек проникновения вирусов в корпоративные сети Интернет/intranet. Как правило, такими точками проникновения вирусов являются: шлюзы и серверы Интернета, серверы файл-приложений, серверы групповой работы и электронной почты, рабочие станции.

Для небольших предприятий, использующих до десятка узлов, целесообразны решения по антивирусной защите, имеющие удобный графический интерфейс и допускающие локальное конфигурирование без применения централизованного управления, например локальные решения AVP Лаборатории Касперского. Для крупных предприятий предпочтительнее системы антивирусной защиты с несколькими консолями и менеджерами управления, подчиненными некоторому единому общему центру, например Trend Enterprise Solution Suite (TESS) компании Trend Micro. Такие решения позволяют обеспечить оперативное централизованное управление локальными антивирусными клиентами и дают возможность при

необходимости интегрироваться с другими решениями в области безопасности корпоративных сетей.

6.3.1. АКТУАЛЬНОСТЬ ЦЕНТРАЛИЗОВАННОГО УПРАВЛЕНИЯ АНТИВИРУСНОЙ ЗАЩИТОЙ КОРПОРАТИВНОЙ СЕТИ ПРЕДПРИЯТИЯ

В настоящее время корпоративная компьютерная сеть средней компании включает в себя десятки и сотни рабочих станций, десятки серверов, различное активное и пассивное телекоммуникационное оборудование и имеет, как правило, достаточно сложную структуру, рис. 6.2.

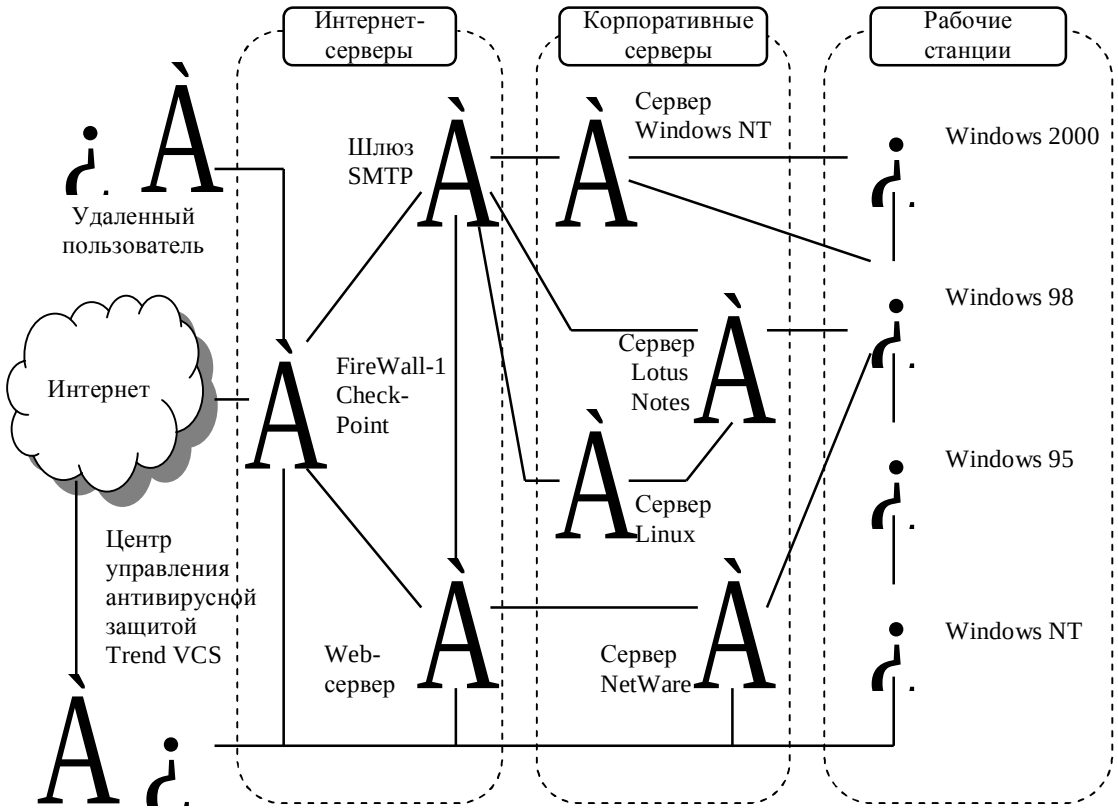


Рис. 6.2. Типовая архитектура корпоративной сети

Согласно отчетам компании Trend Micro корпоративные пользователи постоянно сталкиваются с фактами проникновения вирусов в свои сети. Опыт практической работы показывает, что вирусные атаки на корпоративные системы Интернет/intranet происходят регулярно, а заражение рабочей станции пользователя, осуществленное с помощью принесенного инфицированного носителя информации, является обычным делом.

Приведем описания некоторых вирусов, которые нанесли существенный ущерб корпоративным пользователям в последнее время:

- PE_FUNLOVE.4099 – это уже не новый резидентный вирус под Windows, который был недавно обнаружен несколькими пользователями Интернета. PE_FUNLOVE инфицирует файлы как на локальных дисках, так и на дисках, доступных по сети;

- TROJ_NAVIDAD.E – это вариант TROJ_NAVIDAD.A, который был впервые обнаружен в ноябре 2000 года. Этот вирус инсталлируется в системе, после чего рассылает себя по адресам из адресной книги инфицированного пользователя в виде присоединенного файла EMANUEL.EXE;
- PE_KRIZ.4050 (деструктивный вирус) – это 32-битовый вирус под Windows. Содержит деструктивную функцию, сходную с функцией вируса PE_CIH, которая позволяет ему изменять данные в CMOS и обнулять BIOS;
- VBS_FUNNY – это новое семейство червей, написанных на языке Visual Basic. При запуске эти черви ищут определенный ключ в реестре, и если его нет, то они рассылают по почте сообщения по всем адресам из адресной книги Microsoft Outlook с присоединенным к ним вирусом. Если указанный ключ найден, то черви записывают на диск исполняемый файл (startx.exe), который является известным «троянцем», похищающим пароли;
- VBS_COLOMBIA – это новая модификация вируса VBS_LOVELETTER.A, имеющего деструктивную функцию, нацеленную на файлы с расширениями: .VBS, .VBE, .JS, .JSE, .CSS, .WSH, .SCT, .HTA, .JPG, .JPEG, .MP3 и .MP2.

Когда корпоративная сеть становится объектом атаки вирусов и других вредоносных программ, часто вся антивирусная защита сети сводится к следующему: через некоторое время после атаки осуществляют сканирование и лечение ряда рабочих станций с помощью локального антивирусного ПО – и считают, что защита обеспечена. На самом деле такая локализация проблемы только отодвигает, но не решает проблему эффективной антивирусной защиты, является минимальной мерой и не гарантирует устойчивого функционирования корпоративной системы в дальнейшем.

Широко известные факты распространения вирусов показывают, что использование локальных антивирусных решений в корпоративной сети является необходимым, но не достаточным средством для эффективной реализации антивирусной защиты предприятия. Сложившаяся ситуация требует создания эффективной системы антивирусной защиты корпоративной сети предприятия.

Эффективная корпоративная система антивирусной защиты – это гибкая динамичная система с обратными связями, реализованная по технологии клиент-сервер, чутко улавливающая любое подозрительное действие в сети. Такая система не допускает распространения вирусов и других враждебных программ в рамках внутренней структуры корпоративной сети. Эффективная корпоративная система антивирусной защиты обнаруживает и нейтрализует различные вирусные атаки – как известные, так неизвестные – на самой ранней стадии их проявления.

Стоимость обслуживания корпоративной сети быстро увеличивается вместе с ростом числа подключаемых рабочих станций. Расходы на антивирусную защиту корпоративной сети являются не последним пунктом в списке общих расходов предприятия. Оптимизация и снижение этих расходов возможны путем реализации централизованного управления антивирусной защитой корпоративной сети в реальном масштабе времени. Такие решения дают возможность администраторам сети предприятия отслеживать все точки проникновения вирусов с единой консоли управления и эффективно управлять всеми присутствующими в корпоративной сети антивирусными средствами различных производителей.

Цель централизованного управления антивирусной защитой довольно проста – блокировать все возможные точки проникновения вирусов, а именно:

- проникновение вирусов на рабочие станции при использовании на рабочей станции инфицированных файлов с переносимых источников (дискеты, компакт-диска, Zip, Jazz, Floptical, Flash и т.д.);
- заражение вирусами с помощью бесплатного инфицированного программного обеспечения, полученного из Интернета через Web или FTP и сохраненного на локальной рабочей станции;
- проникновение вирусов при подключении к корпоративной сети инфицированных рабочих станций удаленных или мобильных пользователей;
- заражение вирусами с удаленного сервера, подсоединенного к корпоративной сети и обменивающегося инфицированными данными с корпоративными серверами файлприложений и баз данных;
- распространение сообщений электронной почты, содержащих в приложениях файлы Excel и Word, инфицированные макровирусами.

Современные корпоративные системы антивирусной защиты выпускают ряд компаний: Trend Micro, Symantec, McAfee, Computer Associates и др.

6.3.2. ЭТАПЫ ПОСТРОЕНИЯ СИСТЕМЫ АНТИВИРУСНОЙ ЗАЩИТЫ КОРПОРАТИВНОЙ СЕТИ

Решения антивирусной защиты корпоративной сети должны эффективно защищать от вирусов и других вредоносных программ все основные компоненты корпоративной сети (шлюзы Интернет/intranet, брандмауэры, серверы, рабочие станции).

Методически процесс построения корпоративной системы защиты от вирусов и других вредоносных программ состоит из следующих этапов.

1 этап. Проведение анализа объекта защиты и определение основных принципов обеспечения антивирусной безопасности

На первом этапе необходимо выявить специфику защищаемой сети, выбрать и обосновать несколько вариантов антивирусной защиты. Этап разбивается на следующие работы:

- проведение аудита состояния компьютерной системы и средств обеспечения антивирусной безопасности (АВБ);
- обследование и картирование информационной системы;
- анализ возможных сценариев реализации потенциальных угроз, связанных с проникновением вирусов. Результатом первого этапа является оценка общего состояния антивирусной защиты.

2 этап. Разработка политики антивирусной безопасности

Этап содержит следующие шаги:

- классификация информационных ресурсов – перечень и степень защиты раз личных информационных ресурсов организации;
- создание сил обеспечения АВБ, разделение полномочий – структура и обязанности подразделения, ответственного за организацию антивирусной безопасности;
- организационно-правовая поддержка обеспечения АВБ – перечень документов, определяющих обязанности и ответственность различных групп пользователей за соблюдение норм и правил АВБ;

- определение требований к инструментам АВБ – к антивирусным системам, которые будут установлены в организации;
- расчет затрат на обеспечение антивирусной безопасности.

Результатом данного этапа является политика антивирусной безопасности предприятия.

3 этап. Разработка плана обеспечения антивирусной безопасности

На этом этапе осуществляется выбор программных средств, средств автоматизированной инвентаризации и мониторинга информационных ресурсов. Разработка требований и выбор средств антивирусной защиты для:

- серверов в локальной сети;
- рабочих станций в локальной сети;
- удаленных серверов/удаленных пользователей;
- групповых приложений и электронной почты типа Microsoft Exchange, Lotus Notes, HP OpenMail;
- шлюзов Интернета (брандмауэров, ргоху-серверов, серверов электронной почты Интернета).

Разработка перечня организационных мероприятий по обеспечению АВБ, разработка (корректировка) должностных и рабочих инструкций персонала с учетом политики АВБ и результатов анализа рисков:

- периодический анализ и оценка ситуации по обеспечению АВБ;
- мониторинг средств АВБ;
- план и порядок обновления средств АВБ;
- контроль соблюдения персоналом своих обязанностей по обеспечению АВБ;
- план обучения определенных категорий пользователей;
- порядок действий в критических ситуациях.

Здесь основным результатом является план обеспечения антивирусной защиты предприятия.

4 этап. Реализация плана антивирусной безопасности

В ходе выполнения последнего этапа реализуется выбранный и утвержденный план антивирусной безопасности. Этап содержит следующие шаги:

- приобретение антивирусных средств;
- внедрение антивирусных средств;
- поддержка антивирусных средств.

В результате выполнения данных работ становится возможным построение эффективной системы корпоративной антивирусной защиты.

ЧАСТЬ 7. СИСТЕМЫ ЭЛЕКТРОННЫХ ПЛАТЕЖЕЙ

Одним из наиболее распространенных приложений, требующих использования методов защиты информации и данных является система денежных обращений.

7.1. БАНКОВСКАЯ СИСТЕМА ПЛАСТИКОВЫХ КАРТ

Использование пластиковых карт для безналичных расчетов имеет большие преимущества перед наличными деньгами. Для владельцев карт это оперативность расчетов; отсутствие риска потери, ограбления и ошибок в расчетах, связанных с использованием наличных денег; возможность получения процентов на остаток средств, хранящихся на картах; обеспечение конфиденциальности информации, хранящейся на карте.

Для предприятий торговли: простота и оперативность обслуживания клиентов; снижение риска ограбления и сложностей, связанных с инкассацией наличных денег; оперативность перевода денежных средств на счета магазинов после инкассации.

Для банка-эмитента: появление новых источников доходов за счет средств, привлеченных на карты; получение комиссионных, взимаемых с операций по картам; увеличение числа клиентов за счет предоставления услуг нового типа; уменьшение расходов на обслуживание наличного оборота.

7.1.1. ПЛАСТИКОВАПЯ КАРТОЧКА КАК ПЛАТЕЖНЫЙ ИНСТРУМЕНТ

Пластиковая карточка – это персонифицированный платежный инструмент, предоставляющий пользующемуся карточкой лицу возможность безналичной оплаты товаров и/или услуг, а также получения наличных средств в отделениях (филиалах) банков и банковских автоматах (*банкоматах*). Принимающие карточку предприятия торговли/сервиса и отделения банков образуют **сеть точек обслуживания** карточки (или приемную сеть).

Особенностью продаж и выдач наличных по карточкам является то, что эти операции осуществляются магазинами и, соответственно, банками «в долг» – товары и наличные предоставляются клиентам сразу, а средства в их возмещение поступают на счета обслуживающих предприятий чаще всего через некоторое время (не более нескольких дней). Гарантом выполнения платежных обязательств, возникающих в процессе обслуживания пластиковых карточек, является выпустивший их **банк-эмитент**. Поэтому карточки на протяжении всего срока действия остаются собственностью банка, а клиенты (*держатели карточек*) получают их лишь в пользование. Характер гарантий банка-эмитента зависит от платежных полномочий, предоставляемых клиенту и фиксируемых классом карточки.

При выдаче карточки клиенту осуществляется ее **персонализация** – на нее заносятся данные, позволяющие идентифицировать карточку и ее держателя, а также осуществляют проверку платежеспособности карточки при приеме ее к оплате или выдаче наличных денег. Процесс утверждения продажи или выдачи наличных по карточке называется **авторизацией**. Для ее проведения точка обслуживания делает запрос **платежной системе** о подтверждении полномочий предъявителя карточки и его финансовых возможностей. Технология авторизации зависит от схемы платежной системы, типа карточки и технической оснащенности

точки обслуживания. Традиционно авторизация проводится «вручную», когда продавец или кассир передает запрос по телефону оператору (голосовая авторизация), или автоматически, карточка помещается в **POS-терминал** или **торговый терминал** (POS - Point Of Sale), данные считываются с карточки, кассиром вводится сумма платежа, а держателем карточки со специальной клавиатуры – секретный **PIN-код** (Personal Identification Number - персональный идентификационный код). После этого терминал осуществляет авторизацию либо устанавливая связь с базой данных платежной системы (**on-line** режим), либо осуществляя дополнительный обмен данными с самой карточкой (**off-line** авторизация). В случае выдачи наличных денег процедура носит аналогичный характер с той лишь особенностью, что деньги в автоматическом режиме выдаются специальным устройством – банкоматом, который и проводит авторизацию.

При осуществлении расчетов держатель карточки ограничен рядом лимитов. Характер лимитов и условия их использования могут быть весьма разнообразными. Однако в общих чертах все сводится к двум основным сценариям.

Держатель **дебетовой карточки** должен заранее внести на свой счет в банке-эмитенте некоторую сумму. Ее размер и определяет лимит доступных средств. При осуществлении расчетов с использованием карточки синхронно уменьшается и лимит. Контроль лимита осуществляется при проведении авторизации, которая при использовании дебетовой карточки является обязательной всегда. Для возобновления (или увеличения) лимита держателю карточки необходимо вновь внести средства на свой счет.

Для обеспечения платежей держатель карточки может не вносить предварительно средства, а получить в банке-эмитенте кредит. Подобная схема реализуется при оплате посредством **кредитной карточки**. В этом случае лимит связан с величиной предоставленного кредита, в рамках которого держатель карточки может расходовать средства. Кредит может быть как однократным, так и возобновляемым. Возобновление кредита в зависимости от договора с держателем карточки происходит после погашения либо всей суммы задолженности, либо некоторой ее части.

Как кредитная, так и дебетовая карточки могут быть также **семейными** и **корпоративными**. Корпоративные карточки предоставляются компанией своим сотрудникам для оплаты командировочных или других служебных расходов. Корпоративные карточки компании связаны с каким-либо одним ее счетом. Карточки могут иметь разделенный и неразделенный лимиты. В первом случае каждому из держателей корпоративных карт устанавливается индивидуальный лимит. Второй вариант больше подходит небольшим компаниям и не предполагает разграничение лимита. Корпоративные карточки позволяют компании детально отслеживать служебные расходы сотрудников.

Семейные карточки в определенном смысле аналогичны корпоративным – право произведения платежей в рамках установленного лимита предоставляется членам семьи держателя карточки. При этом дополнительным пользователям предоставляются отдельные персонализированные карточки.

7.1.2. ЭМИТЕНТЫ И ЭКВАЙЕРЫ

Банк-эмитент, выпуская карточки и гарантируя выполнение финансовых обязательств, связанных с использованием выпущенной им пластиковой карточки как платежного средства, сам не занимается деятельностью, обеспечивающей ее прием предприятиями торговли и сферы услуг. Эти задачи решает **банк-эквайер**, осуществляющий весь спектр операций по взаимодействию с точками обслуживания карточек: обработку запросов на авторизацию, перечисление на расчетные счета точек средств за товары и услуги, предоставленные по карточкам, прием, сортировку и пересылку документов (бумажных и электронных), фиксирующих совершение сделок с использованием карточек, распространение *стоп-листов* (перечней карточек, операции по которым по тем или иным причинам на сегодняшний день приостановлены) и др. Кроме того, банк-эквайер может осуществлять выдачу наличных по карточкам как в своих отделениях, так и через принадлежащие ему банкоматы. Банк может и совмещать выполнение функций эквайера и эмитента. Следует отметить, что основными, неотъемлемыми функциями банка-эквайера являются финансовые, связанные с выполнением расчетов и платежей точкам обслуживания. Что же касается перечисленных выше технических атрибутов его деятельности, то они могут быть делегированы эквайером специализированным сервисным организациям - *процессинговым центрам*.

Выполнение эквайерами своих функций влечет за собой расчеты с эмитентами. Каждый банк-эквайер осуществляет перечисление средств точкам обслуживания по платежам держателей карточек банков-эмитентов, входящих в данную платежную систему. Поэтому соответствующие средства (а также, возможно, средства, возмещающие выданную наличность) должны быть затем перечислены эквайеру этими эмитентами. Оперативное проведение взаиморасчетов между эквайерами и эмитентами обеспечивается наличием в платежной системе *расчетного банка* (одного или нескольких), в котором банки - члены системы открывают корреспондентские счета.

7.1.3. ПЛАТЕЖНАЯ СИСТЕМА

Платежной системой называют совокупность методов и реализующих их субъектов, обеспечивающих в рамках системы условия для использования банковских пластиковых карточек оговоренного стандарта в качестве платежного средства. Одна из основных задач, решаемых при создании платежной системы, состоит в выработке и соблюдении общих правил обслуживания карточек входящих в систему эмитентов, проведения взаиморасчетов и платежей. Эти правила охватывают как чисто технические аспекты операций с карточками – стандарты данных, процедуры авторизации, спецификации на используемое оборудование и пр., так и финансовые стороны обслуживания карточек – процедуры расчетов с предприятиями торговли и сервиса, входящими в состав приемной сети, правила взаиморасчетов между банками, тарифы и т.д.

Таким образом, с организационной точки зрения ядром платежной системы является основанная на договорных обязательствах ассоциация банков. В состав платежной системы также входят предприятия торговли и сервиса, образующие сеть точек обслуживания. Для успешного функционирования платежной системы необходимы и специализированные нефинансовые организации, осуществляющие техническую поддержку обслуживания карточек: процессинговые и коммуникационные центры, центры технического обслуживания и т.п.

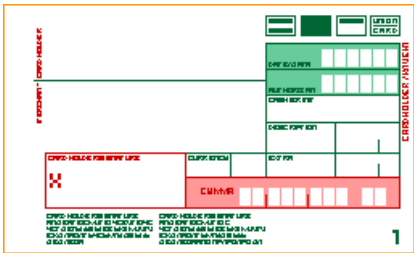
Процессинговый центр – специализированная сервисная организация – обеспечивает обработку поступающих от эквайеров (или непосредственно из точек обслуживания) запросов на авторизацию и/или протоколов транзакций – фиксируемых данных о произведенных посредством карточек платежах и выдачах наличных. Для этого центр ведет базу данных, которая, в частности, содержит данные о банках – членах платежной системы и держателях карточек. Центр хранит сведения о лимитах держателей карточек и выполняет запросы на авторизацию в том случае, если банк-эмитент не ведет собственной базы (**off-line банк**). В противном случае (**on-line банк**) процессинговый центр пересылает полученный запрос в банк-эмитент авторизуемой карточки. Очевидно, что центр обеспечивает и пересылку ответа банку-эквайеру. Кроме того, на основании накопленных за день протоколов транзакций процессинговый центр готовит и рассылает итоговые данные для проведения взаиморасчетов между банками-участниками платежной системы, а также формирует и рассылает банкам-эквайерам (а, возможно, и непосредственно в точки обслуживания) стоп-листы. Процессинговый центр может также обеспечивать потребности банков-эмитентов в новых карточках, осуществляя их заказ на заводах и последующую персонализацию. Следует отметить, что разветвленная платежная система может иметь несколько процессинговых центров, роль которых на региональном уровне могут выполнять и банки-эквайеры.

Коммуникационные центры обеспечивают субъектам платежной системы доступ к сетям передачи данных. Использование специальных высокопроизводительных линий коммуникации обусловлено необходимостью передачи больших объемов данных между географически распределенными участниками платежной системы при авторизации карточек в торговых терминалах, при обслуживании карточек в банкоматах, при проведении взаиморасчетов между участниками системы и в других случаях.

7.1.4. ТЕХНИЧЕСКИЕ СРЕДСТВА

7.1.4.1. ВИДЫ ПЛАСТИКОВЫХ КАРТОЧЕК

Пластиковая карточка представляет собой пластину стандартных размеров (85.6 мм 53.9 мм 0.76 мм), изготовленную из специальной, устойчивой к механическим и термическим воздействиям, пластмассы. Из проведенного в предыдущих разделах рассмотрения следует, что одна из основных функций пластиковой карточки – обеспечение идентификации использующего ее лица как субъекта платежной системы. Для этого на пластиковую карточку наносятся логотипы банка-эмитента и платежной системы, обслуживающей карточку, имя держателя карточки, номер его счета, срок действия карточки и пр. Кроме этого, на карточке может присутствовать фотография держателя и его подпись. Алфавитно-цифровые данные – имя, номер счета и др. – могут быть **эмбоссированы**, т.е. нанесены рельефным шрифтом. Это дает возможность при ручной обработке принимаемых к оплате карточек быстро



перенести данные на чек с помощью специального устройства, **импринтера**, осуществляющего «прокатывание» карточки (в точности так же, как получается второй экземпляр при использовании копировальной бумаги).

Графические данные обеспечивают возможность **визуальной идентификации** карточки. Карточки, обслуживание которых основано на таком принципе, могут с успехом использоваться в малых локальных системах – как клубные, магазинные карточки и т.п. Однако для использования в банковской платежной системе визуальной «обработки» оказывается явно недостаточно. Представляется целесообразным хранить данные на карточке в виде, обеспечивающем проведение процедуры автоматической авторизации. Эта задача может быть решена с использованием различных физических механизмов.

Карточки со **штрих-кодом**. В карточках со **штрих-кодом** в качестве идентифицирующего элемента используется штриховой код, аналогичный коду, применяемому для маркировки товаров**.** Штрих-код (штриховой код) – это последовательность чёрных и белых полос, представляющая некоторую информацию в удобном для считывания техническими средствами виде. Линейными (обычными) называются штрихкоды, читаемые в одном направлении (по горизонтали). Наиболее распространённые линейные символики: EAN (EAN-8 состоит из 8 цифр, EAN-13 - используются 13 цифр), UPC (UPC-A, UPC-E), Code39, Code128 (UCC/EAN-128), Codabar, «Interleaved 2 of 5». Линейные символики позволяют кодировать небольшой объём информации (до 20-30 символов, обычно цифр). Обычно кодовая полоска покрыта непрозрачным составом и считывание кода происходит в инфракрасных лучах. Карточки со штрих-кодом весьма дешевы и, по сравнению с другими типами карт, относительно просты в изготовлении. Последняя особенность обуславливает их слабую защищенность от подделки и поэтому делает малопригодными для использования в платежных системах.

Карточки с **магнитной полосой**. Карточки с **магнитной полосой** являются на сегодняшний день наиболее распространенными – в обращении находится свыше двух миллиардов карт подобного типа. Магнитная полоса располагается на обратной стороне карты и, согласно стандарту ISO 7811, состоит из трех дорожек. Из них первые две предназначены для хранения идентификационных данных, а на третью можно записывать информацию (например, текущее значение лимита дебетовой карточки). Однако из-за невысокой надежности многократно повторяемого процесса записи/считывания, запись на магнитную полосу, как правило, не практикуется, и такие карты используются только в режиме считывания информации. Защищенность карт с магнитной полосой существенно

выше, чем у карт со штрих-кодом. Однако и такой тип карт относительно уязвим для мошенничества. Тем не менее, развитая инфраструктура существующих платежных систем и, в первую очередь, мировых лидеров «карточного» бизнеса – компаний VISA и MasterCard/Europay является причиной интенсивного использования карточек с магнитной полосой и сегодня. Отметим, что для повышения защищенности карточек системы VISA и MasterCard/Europay используются дополнительные графические средства защиты: голограммы и нестандартные шрифты для эмбоссирования.

На лицевой стороне карточки с магнитной полосой обычно указывается: логотип банка-эмитента, логотип платежной системы, номер карточки (первые 6 цифр - код банка, следующие 9 - банковский номер карточки, последняя цифра - контрольная, последние четыре цифры нанесены на голограмму), срок действия карточки, имя держателя карточки; на оборотной стороне - магнитная полоса, место для подписи.

Смарт-карты В **смарт-картах** носителем информации является микросхема. У простейших из существующих смарт-карт – *карт памяти* – объем памяти может иметь величину от 32 байт до 16 килобайт. Эта память может быть реализована или в виде ППЗУ (EPROM), которое допускает однократную запись и многократное считывание, или в виде ЭСППЗУ (EEPROM), допускающее и многократное считывание, и многократную запись. Карты памяти подразделяются на два типа: с незащищенной (полнодоступной) и защищенной памятью. В картах первого типа нет никаких ограничений на чтение и запись данных. Доступность всей памяти делает их удобными для моделирования произвольных структур данных, что представляется важным в некоторых приложениях. Карты с защищенной памятью имеют область идентификационных данных и одну или несколько прикладных областей. Идентификационная область карт допускает лишь однократную запись при персонализации, и в дальнейшем доступна только на считывание. Доступ к прикладным областям регламентируется и осуществляется по предъявлению соответствующего ключа (пароля). Уровень защиты карт памяти выше, чем у магнитных карт, и они могут быть использованы в прикладных системах, в которых финансовые риски, связанные с мошенничеством, относительно невелики. Что же касается стоимости карт памяти, то они дороже, чем магнитные карты. Однако в последнее время цены на них значительно

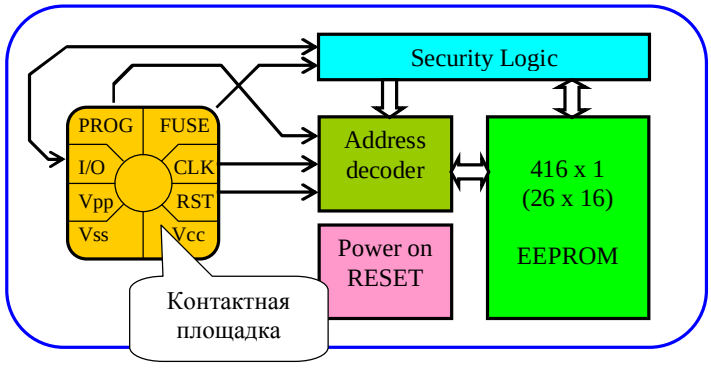


Рис. 7.1. Внутренняя архитектура карты-памяти GPM416

снизились в связи с усовершенствованием технологии и ростом объемов производства. Стоимость карты памяти непосредственно зависит от стоимости микросхемы, определяемой, в свою очередь, емкостью памяти.

Примером ***карты-памяти*** может служить карта GPM416 компании Gemplus оснащенная электронно-перепрограммируемой интегральной схемой ST1301 компании SGS-Thomson. Ее память объемом 416 бит (26 слов по 16 бит) защищена плавкой перемычкой (Fuse) и секретными кодами (как пользователя, так и владельца), а также содержит программируемую логическую матрицу. На рис. 7.1 приведена внутренняя структура карты-памяти. Доступ к карте осуществляется по синхронному последовательному протоколу SPI, обычно на скорости 9600 бит/сек. Для работы карте требуется напряжение питания Vcc (5В) и для персонализации – Vpp (21В). Vss – GND (0В). Сигнал RST низким уровнем сбрасывает адрес памяти в начальное состояние и сигнализирует начало кадра обмена данными. Данные из карты или в карту поступают последовательно бит за битом через вывод I/O в сопровождении тактовых импульсов CLK. После завершения персонализации (записи постоянных характеристик – сведений о банке-эмитенте, плательщике, параметрах защиты и т.п.) плавкая перемычка FUSE пережигается и дальнейшие изменение идентификационной области памяти (96 бит) становятся невозможным.

Карта памяти GPM416EEPROM

<table>
<tr><td colspan="2">Зона изготовителя</td><td>(16 бит)</td></tr>
<tr><td colspan="2">Зона владельца</td><td>(48 бит)</td></tr>
<tr><td colspan="2">Ключ пользователя</td><td>(16 бит)</td></tr>
<tr><td>Счетчик ошибок (4 бит)</td><td>Зона #1</td><td>(12 бит)</td></tr>
<tr><td colspan="2">Зона #2</td><td>(16 бит)</td></tr>
<tr><td>PREN</td><td>RDEN</td><td></td></tr>
<tr><td colspan="2">Рабочая зона</td><td>(208 бит)</td></tr>
<tr><td colspan="2">Ключ владельца</td><td>(32 бит)</td></tr>
<tr><td colspan="2">Счетчик стираний</td><td>(64 бит)</td></tr>
</table>

Всего: 416 бит

Зоны карты памяти:

- Зона пользователя размером 16 бит. Программируется раз и навсегда непосредственно на заводе и служит для идентификации заказчика (иначе говоря, издателя карты или владельца);
- Зона владельца размером 48 бит, раз и навсегда программируемая при персонализации карты;
- Ключ-носитель 16 бит. Программируется заводом-изготовителем одновременнор с кодом, называемым «транспортным», который заказчик должен предоставлять при персонализации карты. Впоследствии это будет конфиденциальный код пользователя, и его можно заменить на новый, если сначала представить старый код;
- Счетчик ошибок длиной 4 бита, который фиксирует количество попыток ввести неверный код. Он обнуляется при вводе правильного кода, но если было последовательно произведены четыре неверных попытки, то микросхема полностью блокируется;

– Зон #1 длиной 12 бит, доступна как считывания, так и для записи данных. Полностью стирается при вводе правильного кода;
– Зона #2 длиной 16 бит, доступная для записи или стирания после введения правильного «транспортного» кода;
– Рабочая зона объемом 208 бит, в которой режимы записи или чтения определяются ее первыми двумя битами (PREN, или Program Enable – разрешение записи, и RDEN, или Read Enable – разрешение чтения) и тем, какой введен код пользователя – правильный или нет. Для стирание этой области необходимо ввести код пользователя, а затем клюя «владельца»;
– Ключ владельца длиной 32 бита. Программируется при персонализации, после которой должен держаться в секрете;
– Счетчик стираний длиной 64 бита, служит для подсчета числа стираний рабочей зоны и может служить ограничителем числа таких операции, если предварительно записать в него соответствующую величину.

Частным случаем карт памяти являются **карты-счетчики**, в которых значение, хранимое в памяти, может изменяться лишь на фиксированную величину. Подобные карты используются в специализированных приложениях с предоплатой (плата за использование телефона-автомата, оплата автостоянки и т.д.).

Карты с микропроцессором представляют собой по сути микрокомпьютеры и содержат все соответствующие основные аппаратные компоненты: центральный процессор, ОЗУ, ПЗУ, ППЗУ, ЭСППЗУ. Параметры наиболее мощных современных микропроцессорных карт сопоставимы с характеристиками персональных компьютеров начала восьмидесятых. Операционная система, хранящаяся в ПЗУ микропроцессорной карты, принципиально ничем не отличается от операционной системы персонального компьютера и предоставляет большой набор сервисных операций и средств безопасности. Операционная система поддерживает файловую систему, базирующуюся в ЭСППЗУ (емкость которого обычно находится в диапазоне 1-8 Кбайта, но может достигать и 64 Кбайт) и обеспечивающую регламентацию доступа к данным. При этом часть данных может быть доступна только внутренним программам карточки, что вместе со встроенными криптографическими средствами делает микропроцессорную карту высокозащищенным инструментом, который может быть использован в финансовых приложениях, предъявляющих повышенные требования к защите информации. Именно поэтому микропроцессорные карты (и смарт-карты вообще) рассматриваются в настоящее время как наиболее перспективный вид пластиковых карт. Кроме того, смарт-карты являются наиболее перспективным типом пластиковых карт также и с точки зрения

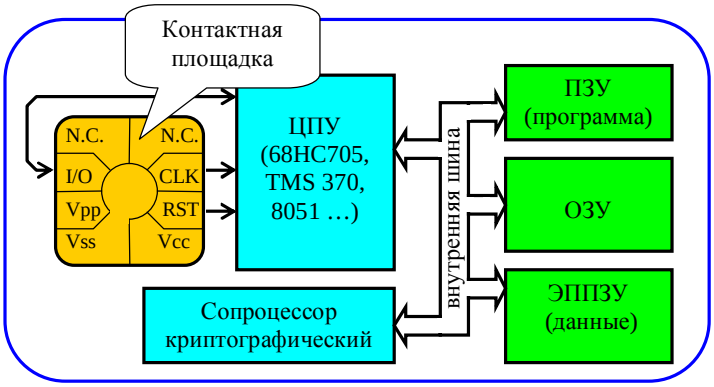


Рис. 7.2. Структурная схема микропроцессорной карты

функциональных возможностей. Вычислительные возможности смарт-карт позволяют использовать, например, одну и ту же карту и в операциях с on-line авторизацией и как многовалютный электронный кошелек.

На рис. 7.2 показана внутренняя структура, характерная практически для всех микропроцессорных карт. Они построены на базе специализированного микропроцессора, в большинстве случаев относящегося к определенному семейству (68HC705 компаний Motorola или SGS-Thomson, TMS 370 компании Texas Instruments, 8051 компании Philips и т.д.). Карты обычно используют асинхронный последовательный интерфейс и поэтому часто называются асинхронными картами. Микропроцессорная или асинхронная карта, как и все чип-карты, должна в первую очередь обеспечивать безопасность данных, хранящихся в ее памяти. Это требование выполняется благодаря тому, что прямой доступ извне к содержимому памяти невозможен. Любая операция чтения, записи или установления подлинности осуществляется через внутренний микропроцессор, который является единственным блоком, имеющим физический доступ к памяти. Именно с помощью микропроцессора (согласно определенным правилам безопасности, запрограммированным заранее) решается вопрос о допустимости выполнения команды, поданной карте. В процессе обработки карта выдает «отчет», уточняющий операции, которые будут проведены после проверки принятой команды, и если надо передать блок данных, то он, как правило, шифруется. На сегодняшнем этапе развития техники все обмены «вопросами и ответами» осуществляются в полудуплексном режиме (half-duplex) по одной линии последовательного ввода/вывода (контакт I/O). Стандарт ISO 7816 предусматривает различные протоколы связи, но диалог часто происходит со скоростью 9600 бит/сек, пакетами по 8 бит данных с битом контроля четности и, по крайней мере, двумя стоп-битами.

Кроме описанных выше типов пластиковых карточек, используемых в финансовых приложениях, существует еще ряд карточек, основанных на иных механизмах хранения данных. Такие карточки (оптические, индукционные и пр.) используются в медицинских системах, системах безопасности и др.

7.1.4.2. POS-ТЕРМИНАЛЫ

POS-терминалы, или торговые терминалы, предназначены для обработки транзакций при финансовых расчетах с использованием пластиковых карточек с магнитной полосой и смарт-карт. Использование POS-терминалов позволяет автоматизировать операции по обслуживанию карточки и существенно уменьшить время обслуживания. Возможности и комплектация POS-терминалов варьируются в широких пределах, однако типичный современный терминал снабжен устройствами чтения как смарт-карт, так и карт с магнитной полосой, энергонезависимой памятью, портами для подключения ПИН-клавиатуры (клавиатуры для набора ПИН-кода), принтера, соединения с ПК или с электронным кассовым аппаратом.

Кроме того, обычно POS-терминал бывает оснащен модемом с возможностью автодозвона. POS-терминал обладает «интеллектуальными» возможностями – его

можно программировать. В качестве языков программирования используются ассемблер, а также диалекты C и Basic'a. Все это позволяет проводить не только on-line авторизацию карт с магнитной полосой и смарт-карт, но и использовать при работе со смарт-картами режим off-line с накоплением протоколов транзакций. Последние во время сеансов связи передаются в процессинговый центр. Во время сеанса связи POS-терминал может также принимать и запоминать информацию, передаваемую ЭВМ процессингового центра. В основном это бывают стоп-листы, но подобным же образом может осуществляться и перепрограммирование POS-терминалов.

Стоимость POS-терминалов в зависимости от комплектации, возможностей, фирмы-производителя может меняться от нескольких сотен до нескольких тысяч долларов, однако обычно не превышает полутора – двух тысяч. Размеры и вес POS-терминала сопоставимы с аналогичными параметрами телефонного аппарата, а зачастую бывают и меньше.

7.1.4.3. БАНКОМАТЫ

Банкоматы – банковские автоматы для выдачи и инкассирования наличных денег при операциях с пластиковыми карточками. Кроме этого, банкомат позволяет держателю карточки получать информацию о текущем состоянии счета (в том числе и выписку на бумаге), а также, в принципе, проводить операции по перечислению средств с одного счета на другой. Очевидно, банкомат снабжен устройством для чтения карты, а для интерактивного взаимодействия с держателем карточки – также дисплеем и клавиатурой. Банкомат оснащен персональной ЭВМ, которая обеспечивает управление банкоматом и контроль его состояния. Последнее весьма важно, поскольку банкомат является хранилищем наличных денег. На сегодняшний день большинство моделей рассчитано на работу в on-line режиме с карточками с магнитной полосой, однако появились и устройства, способные работать со смарт-картами и в off-line режиме. Для обеспечения коммуникационных функций банкоматы оснащаются платами X.25, а, в некоторых случаях, – модемами.

Денежные купюры в банкомате размещаются в кассетах, которые, в свою очередь, находятся в специальном сейфе. Число кассет определяет количество номиналов купюр, выдаваемых банкоматом. Размеры кассет регулируются, что дает возможность заряжать банкомат практически любыми купюрами.

Банкоматы – стационарные устройства солидных габаритов и веса. Примерные размеры: высота – 1.5-1.8 м, ширина и глубина – около 1 м, вес - около тонны. Более того, с целью пресечения возможных хищений их монтируют капитально. Банкоматы могут размещаться как в помещениях, так и непосредственно на улице и работать круглосуточно.

7.1.4.4. ПРОЦЕССИНГОВЫЙ ЦЕНТР И КОММУНИКАЦИИ

Процессинговый центр – специализированный вычислительный центр, являющийся технологическим ядром платежной системы. Процессинговый центр функционирует в достаточно жестких условиях, гарантированно обрабатывая в реальном масштабе времени интенсивный поток транзакций. Действительно, использование дебетовой карточки приводит к необходимости on-line авторизации каждой сделки в любой точке обслуживания платежной системы. Для операций с кредитной карточкой авторизация необходима не во всех случаях, но, например, при получении денег в банкоматах она также проводится всегда. Не меньшие требования к вычислительным возможностям процессингового центра предъявляет и подготовка данных для проведения взаиморасчетов по итогам дня, поскольку обработке подлежат протоколы значительной (если не подавляющей) части транзакций, а требуемые сроки выполнения расчетов невелики – несколько часов.

Помимо вычислительных мощностей, процессинговый центр, если он осуществляет весь спектр сервисных функций, должен быть оснащен также оборудованием для персонализации пластиковых карточек (включая, возможно, и смарт-карты), а также иметь базу для технического сопровождения и ремонта POS-терминалов и банкоматов.

Таким образом, поддержание надежного, устойчивого функционирования платежной системы требует, во-первых, наличия существенных вычислительных мощностей в процессинговом центре (или центрах – в развитой системе) и, во-вторых, развитой коммуникационной инфраструктуры, поскольку процессинговый центр системы должен иметь возможность одновременно обслуживать достаточно большое число географически удаленных точек. Кроме того, неизбежна также маршрутизация запросов, что еще больше ужесточает требования к коммуникациям. В заключение укажем еще один источник сообщений – электронные документы, которыми обмениваются банки-участники с расчетным банком, а, возможно, и друг с другом при регулярном проведении взаиморасчетов. Очевидно, что для эффективного решения изложенных проблем необходимо использование высокопроизводительных сетей передачи данных с коммутацией пакетов. Со структурной точки зрения сеть передачи данных при этом становится внутренним неотъемлемым элементом платежной системы.

7.1.5. ТЕХНОЛОГИЯ БЕЗНАЛИЧНЫХ РАСЧЕТОВ НА ОСНОВЕ ПЛАСТКОВЫХ КАРТ

7.1.5.1. КРЕДИТНЫЕ КАРТОЧКИ

Вопрос о выдаче кредитной карточки решается банком-эмитеном на основе доступных ему сведений о кредитной истории клиента, то есть о том, каковы доходы клиента, где и когда клиент пользовался кредитом, насколько аккуратно возвращал его, насколько часто берутся кредиты и т.д. Кредитная история позволяют банку оценить степень риска при выдаче карточки и, соответственно, сформулировать требования о предоставлении клиентом тех или иных гарантий.

Лимиты операций по кредитным карточкам – величина кредитной линии, количество и максимальные суммы приобретений и/или получение наличных за тот или иной промежуток времени – устанавливаются индивидуально для каждого клиента. По завершению очередного «делового периода» (обычно один, два месяца), пользователь карточки получает сообщения банка, содержащие данные за период о всех платежах по карточке, информация о которых поступила в банк.

При наступлении контрольной даты (обычно по истечению нескольких дней после получения сообщения) пользователь должен вернуть кредит, после чего ему вновь открывается кредитная линия. Поэтому не обязательно возмещать всю сумму сразу. Достаточно внести некоторый ранее оговоренный минимум. Кредитная линия будет открыта в размере неиспользованной части кредитного лимита. Остаток по задолженности, на который уже будут начисляться проценты, можно гасить в течение достаточно длительного срока (например, года). Отметим, что именно проценты по не полностью возвращенным кредитам и формируют основную часть дохода банка при операциях с кредитными карточками.

7.1.5.2. ДЕБЕТОВЫЕ КАРТОЧКИ

Получение дебетовой карточки – существенно более простой процесс по сравнению с получением кредитной. Для того чтобы стать держателем этой карточки, необходимо открыть счет в этом банке, внести некоторый минимум средств и оплатить карточку. В стоимость карточки, как правило, включается и плата за годовое обслуживание.

Операции с дебетовой карточкой также регламентируются рядом лимитов. При этом процедура авторизации должна проводиться всегда вне зависимости от конкретной технологии обслуживания. Лимит каждый раз уменьшается на сумму сделки. В зависимости от конкретных условий допускается уменьшение лимита либо до нуля, либо до некоторого «неуменьшаемого» значения.

Как и в случае кредитных карточек, банк-эмитент является собственником дебетовых карточек и несет по ним полную финансовую ответственность перед прочими участниками платежной системы. Банк-эмитент управляет размерами лимита и вправе уменьшать или увеличивать их по своему усмотрению (в рамках существующих соглашений с платежной системой). Реально же соотношение этих величин с суммами на счете держателя карточки – внутреннее дело банка. Как правило, для восстановления лимита банк требует внесения средств на счет. Однако банк-эмитент может разрешить клиенту овердрафт, предоставить кредит и т.д. сообщая тем самым дебетовой карточки черты кредитной. В таком случае карточку обычно называют смешанной. Тем не менее, с точки зрения платежной системы эти карточки неразличимы и обслуживаются по единой технологии.

7.1.6. ЦИКЛ ОПЕРАЦИЙ ПРИ ОБСЛУЖИВАНИИ КАРТОЧКИ

Последовательности операций, выполняемых при расчетах за товары и услуги посредством кредитной и дебетовой карточек, весьма близки. Рассмотрим всю последовательность для карточек, рис.7.3.

1) Желая совершить покупку (или получить услугу), держатель карточки предъявляет ее продавцу.
2) Продавец устанавливает возможно ли совершение сделки. Для этого он осуществляет проверку подлинности карточки и, при необходимости, правомочность распоряжения ею покупателем. Затем продавец проводит либо голосовую авторизацию по телефону, либо электронную посредством POS-терминала. Авторизация не проводится в том случае, если сумма покупки (или услуги) ниже торгового лимита. При голосовой авторизации в случае положительного ответа продавцу сообщается код авторизации. Если по каким либо причинам банк-эмитент не дает разрешение на проведение

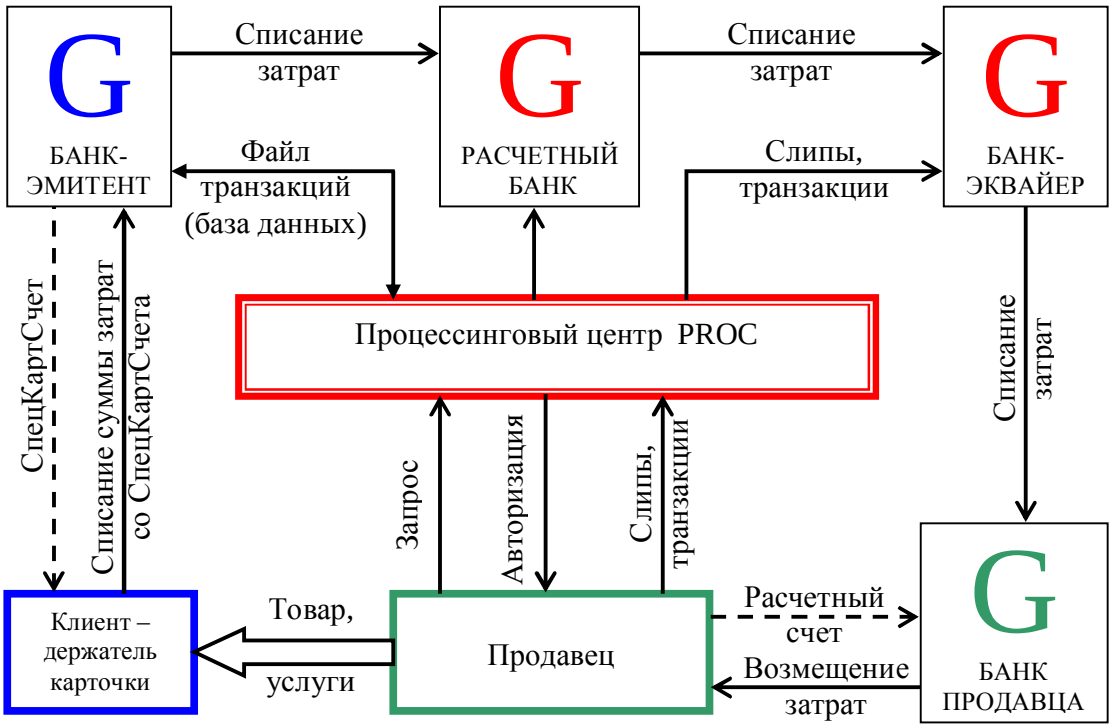


Рис.7.3. Схема обслуживания по плпстиковой карточке

операции, то генерируется код отказа, который также передается в торговую точку.

3) После утверждения сделки происходит оформление торгового векселя (*чека* или *слипа*), на котором фиксируются данные с карточки. При ручной обработке для этого используется импринтер. Кроме того, в этом случае на чек обязательно заносится код авторизации, т.к. при отсутствии кода чек не будет принят к оплате банком-эквайером. Держатель карточки за тем подписывает все экземпляры чека (обычно три). При автоматической обработке чек печатается POS-терминалом. В последнем случае подпись может быть необязательной, если авторизация проводилась с использованием ПИН-кода.
4) Экземпляр чека и товар передаются покупателю.
5) В конце каждого дня (или реже – несколько раз в неделю) точка обслуживания пересылает в банк-эквайер экземпляры чеков, которые служат документальным основанием для проведения расчетов с точкой обслуживания. При автоматической обработке таким основанием могут служить электронные протоколы транзакции, генерируемые при авторизации и непосредственно поступающие в банк-эквайер или процессинговый центр.
6) Банк-эквайер верифицирует транзакции и осуществляет расчеты с точкой обслуживания (производит перечисление средств на ее расчетный счет). В случае ручной обработки верификация сводится к проверке полученных чеков. Для верификация же электронных транзакций в конце каждого дня торговые терминалы связываются с эквайером и повторно пересылают транзакции, накопленные в их памяти за день. Кроме того, банк-эквайер

передает в процессинговый центр «чужие» транзакции, т.е. транзакции, произведенные держателями карточек других банков-эмитентов.

7) Процессинговый центр обрабатывает полученные за день транзакции и формирует итоговые данные для проведения взаиморасчетов между банками – участниками платежной системы. Общие итоги передаются в расчетный банк системы, а частные рассылаются банкам-участникам в качестве извещений на проведение взаиморасчетов.
8) Расчетный банк проводит взаиморасчеты между банками-участниками, открывшими у него корреспондентские счета. Прочие банки-участники осуществляют перечисления самостоятельно.
9) Банки-эмитенты в порядке возмещения осуществляют снятие средств со счетов держателей дебетовых карточек за приобретенные товары и услуги. Держатели кредитных карточек возвращают банкам-эмитентам средства, предоставленные им как кредит при приобретении товаров и услуг.

7.1.7. АВТОРИЗАЦИЯ

7.1.7.1. ON_LINE РЕЖИМ

Рассмотрим более подробно второй шаг приведенной выше схемы. Процедура принятия решения о допустимости сделки предполагают несколько уровней ответственности.

Если сума сделки не превышает торгового лимита, то решение о продаже (предоставления услуг) принимается самостоятельно точкой обслуживания. Таким образом, если торговый лимит точки обслуживания равен, например 50$, то решение о продаже товаров, стоимость которых не превышает этой суммы, принимается самостоятельно продавцом. При этом продавец обязан проверить, соответствует ли карточка установленным данной платежной системой спецификациям, отсутствует ли она в стоп-листе. При автоматической обработки такая проверка осуществляется с помощью POS-терминала, в память которого обычно загружается стоп-лист. При этом терминал функционирует в режиме Off-line. Кроме, этого после оформления чека продавец обязан проверить совпадение подписи на нем с образцом подписи на карточке (если образец подписи имеется). При возникновении сомнений продавец вправе потребовать предъявить документ, удостоверяющий личность покупателя. Чек подписывается покупателем и при автоматической обработке поскольку при таких сделках ПИН-код не вводится.

При выполнении перечисленных требований продавец может осуществить сделку. Если впоследствии банком-эквайером будет обнаружено какое-либо нарушение перечисленных правил, то банк вправе отказаться от возмещения продавцу сумму сделки.

Ненулевой торговый лимит характерен для кредитных карточек. Однако, некоторыми платежными системами он устанавливается и для дебетовых карт. Что же касается международной практики, то, например, в США на сегодняшний день торговый лимит в точках обслуживания систем VISA и MasterCard устанавливается нулевым. Это мера, позволяющая снизить потери от мошенничества, стала возможной благодаря повсеместному оснащению торговых точек POS-терминалами.

Если сумма сделки превышает торговый лимит, то продавец обязан провести авторизацию. Для этого он либо непосредственно связывается с банком-эквайером (или эквайер-центром) по телефону и передает данные устно, либо эта процедура

осуществляется в автоматическом режиме POS-терминалом. Эквайер осуществляет маршрутизацию транзакции, которая в конечном итоге попадает в центр, уполномоченный на авторизацию данной транзакции.

В международной практике таким центром обычно является сам банк-эмитент. Однако эмитент может на постоянной основе делегировать права на проведения авторизации какому-либо процессинговому центру или даже банку-эквайеру.

При принятии решения центр авторизации руководствуется данными, поступившими от точки обслуживания, а также имеющейся в базе данных информацией о держателе карточки, его лимитах, совершенных сделках и проч. Если сумма сделки и другие ее параметры не противоречат установленным лимитам и ограничениям, то решение об осуществлении сделки принимает центр авторизации. В противном случае пытается связаться с банком-эмитентом. Если это оказывается невозможным, то транзакция отклоняется.

При достижении авторизуемой транзакцией ЭВМ, содержащей данные о лимитах держателя соответствующей карточки, происходит следующее.

1) Проверяется, не занесена ли обслуживаемая карточка в стон-лист.
2) Если карточка не блокирована, то, исходя из суммы сделки, величины остатка лимита, времени сделки, типа и местоположения точки обслуживания и, возможно, каких-либо специальных условий (например, наличие для данного привилегированного держателя разрешения на превышение лимита) определяется возможность платежа по карточке.
3) Если платеж возможен, то транзакция утверждается, и остаток лимита уменьшается на сумму сделки.
4) Утверждение транзакции вызывает также следующие действия:

– при ручном режиме точке обслуживания сообщается код авторизации, который обязательно должен быть занесен на чек и является подтверждением полученного разрешения;
– в автоматическом режиме POS-терминалу отдается команда на фиксацию транзакции и распечатку чека.

Подчеркнем, что ответственность перед торговой точкой по возмещению стоимости сделки несет банк-эквайер, и, аналогично предыдущему случаю, при нарушениях правил авторизации платежная система вправе не возмещать сумму сделки.

7.1.7.2. OFF-LINE РЕЖИМ. ЭЛЕКТРОННЫЙ КОШЕЛЕК

Ключевым фактором функционирования платежной системы является проведение авторизации при каждой сделке. Однако использование on-line режима при авторизации может оказаться либо невозможным, либо нецелесообразным. Первое относится, в частности, к России, где телефонная сеть, за исключением самых крупных городов, развита слабо и практически везде функционирует ненадежно. Что же касается западных стран, то там использование on-line режима признано нецелесообразным при мелких платежах, которые осуществляются, например, в такси, табачных киосках и т.д. Этот до сих пор не был охвачен пластиковыми карточками, но недавно две крупнейшие системы –VISA и MasterCard – заявили о своих планах его освоения.

В обоих случаях выход состоит в изменении технологии авторизации таким образом, чтобы исключить необходимость on-line сеанса и осуществлять авторизацию в off-line предъявляет следующее требования:

- наличие на карточке данных о текущем значении лимита;
- возможность контролируемого уменьшения лимита в результате авторизации («дебетование» карточки);
- возможность восстановления лимита на карточке («кредитование карточки»).

Для обеспечения подобных возможностей карточка должна, как минимум, обладать перезаписываемой памятью. Кроме того, необходимо, чтобы POS-терминалы обладали некоторыми «интеллектуальными» возможностями для проведения подобных операций, также памятью (внутренней и внешней) достаточно большой емкостью, где можно было бы накапливать транзакцию для последующей передачи (обычно именуемой в таких случаях инкассацией) в банк-эквайер или процессинговый центр.

В принципе, карты с магнитной полосой допускают возможность подобного использования. Однако малая емкость памяти (сотни байт) и, главное, слабая защищенность от несанкционированного изменения данных, записанных на магнитную полосу, делает их не пригодными для обслуживания в off-line режиме. Напротив, смарт-карты обладают всеми необходимыми предпосылками для реализации схемы обслуживания с off-line авторизацией. Действительно, смарт-карты обладают достаточным объемом памяти (несколько килобайт) для хранения данных о текущем состоянии платежного лимита и некоторого количества последних транзакций. Кроме того, наличие микропроцессора на многих типах карт делает возможным выполнение при авторизации сравнительно сложных процедур. Наличие на карте специальных защищенных зон памяти и криптографических средств обеспечивают высокий уровень безопасности смарт-карт (особенно микропроцессорных карт).

В схеме off-line авторизацией каждая карточка становиться мобильным элементом распределенной базы данных платежной системы и выполняет функции процессингового микроцентра. Для проведения авторизации карточка помещается в считывающие устройство POS-терминала. При этом карточка и терминал, основываясь на хранящихся в них системных данных, обмениваются информацией и производят взаимное опознание. Если эта процедура завершается успешно, то держатель может вводит ПИН-код, а продавец – сумму. После этого карточка проверяет, не превышен ли лимит. Если сделка осуществима, то карточка уменьшает лимит на сумму сделки, а POS-терминал фиксирует данные о транзакции. Об этом процессе условно говорят как о дебетовании карточки и кредитовании терминала, а о самой карточке в таком режиме – как об **электроном кошельке**. POS-терминал печатает чек, карточка изымается из терминала и возвращается держателю.

Говоря о процедуре опознания, а также о накоплении транзакций в торговом терминале, мы игнорировали различные варианты технической реализации этих процессов. В действительности эти функции могут быть возложены как на собственно терминал, так и на персональный компьютер, к которому он подключен и который управляет работой терминала. В торговый терминал одновременно с картой покупателя может также помещаться и специальная смарт-карта торговой

точки, которая и берет на себя все «интеллектуальные» функции, а терминал при этом выполняет роль коммуникационного звена.

Накопленные терминалом транзакции передаются в процессинговый центр (или банк-эквайер) во время сеанса связи. Этот сеанс в зависимости от загрузки телефонной сети и доступности канала связи может либо происходить несколько раз за день, либо осуществляться в вечернее или ночное время. Более того, возможно перенесение данных и на внешние носители информации – флоппи-диск или технологическую смарт-карту, которые затем и доставляются в процессинговый центр. Возможно также инкассация и на портативный персональный компьютер, с которым представитель процессингово центра или банка-эквайера объезжает торговые точки.

Рассмотрим процедуру управления лимитами. Первоначально значения лимитов расхода заносятся на карту при ее персонализации в банке-эмитенте или специальном персонализационном центре. Восстановление лимитов в процессе использования дебетовой смарт-карты может происходить несколькими способами в зависимости от возможностей оборудования и принятой в платежной системе технологии.

В наиболее типичном случае держатель карточки должен посетить банк или одно из его отделений, где уполномоченный сотрудник, поместив карточку в специально предназначенный для этого терминал, восстановит лимит. При этом предполагается, что состояние счета держателя карточки дает основания для проведения такой операции.

Во втором варианте восстановление можно осуществить на любом торговом терминале, для чего должен быть проведен специальный сеанс в on-line режиме. При этом устанавливается связь с процессинговым центром банка-эквайера (или центром системы), который и пересылает на терминал сообщение, обеспечивающее выполнение действий по восстановлению лимитов.

«Интеллектуальная мощь» микропроцессорных смарт-карт дает возможность реализовывать и более изощренные варианты. Одним из них является режим самокредитования карты. Суть его состоит в том, что на карту при персонализации заносятся как общий, так и, например, месячный лимиты. Текущие расходы ограничиваются месячным лимитом, который в начале каждого очередного месяца при первом же взаимодействии с POS-терминалом самостоятельно восстанавливается картой. Общий лимит при этом определяет возможное число таких самокредитований. Как и обычно, средства, обеспечивающие расходы держателя карточки в течение всего срока действия процедуры сомокредитования, должны быть заранее размещены на счете.

7.1.8. ОБРАБОТКА ТРАНЗАКЦИЙ

7.1.8.1. МАРШРУТИЗАЦИЯ ТРАНЗАКЦИЙ В ON-LINE СИСТЕМАХ

Обработка поступающих транзакций является одной из важнейших функции платежной системы. В свою очередь, маршрутизация транзакций в системах с on-line авторизацией представляет существенный элемент этого процесса.

Вычислительный комплекс банка-эквайера (или процессингого центра) осуществляет **идентификацию транзакции**, определяя, инициирована ли она держателем карточки – клиентом данного банка-эквайера или клиентом другого банка платежной системы. В первом случае обработка транзакции происходит на

месте, во втором осуществляется ее пересылка в какой-либо процессинговый узел системы, где происходит либо ее обработка, либо дальнейшая маршрутизация.

На рис. 7.4 представлена схема развитой географически распределенной платежной системы с децентрализованной организацией. Масштаб системы привел к разделению функций. Коммуникационные центры (NET1 и NET2), объединенные в глобальную сеть, осуществляют непосредственную адресацию транзакций. Кроме этого, в нештатных ситуациях – отказах глобальной сети – локальные коммуникационные центры осуществляют авторизацию транзакций по карточкам удаленных банков. Так, при выходе глобальной сети из строя коммуникационный центр NET1 будет авторизовывать транзакции банков D, E и F, а центр NET2 – транзакции банков A, B и C.

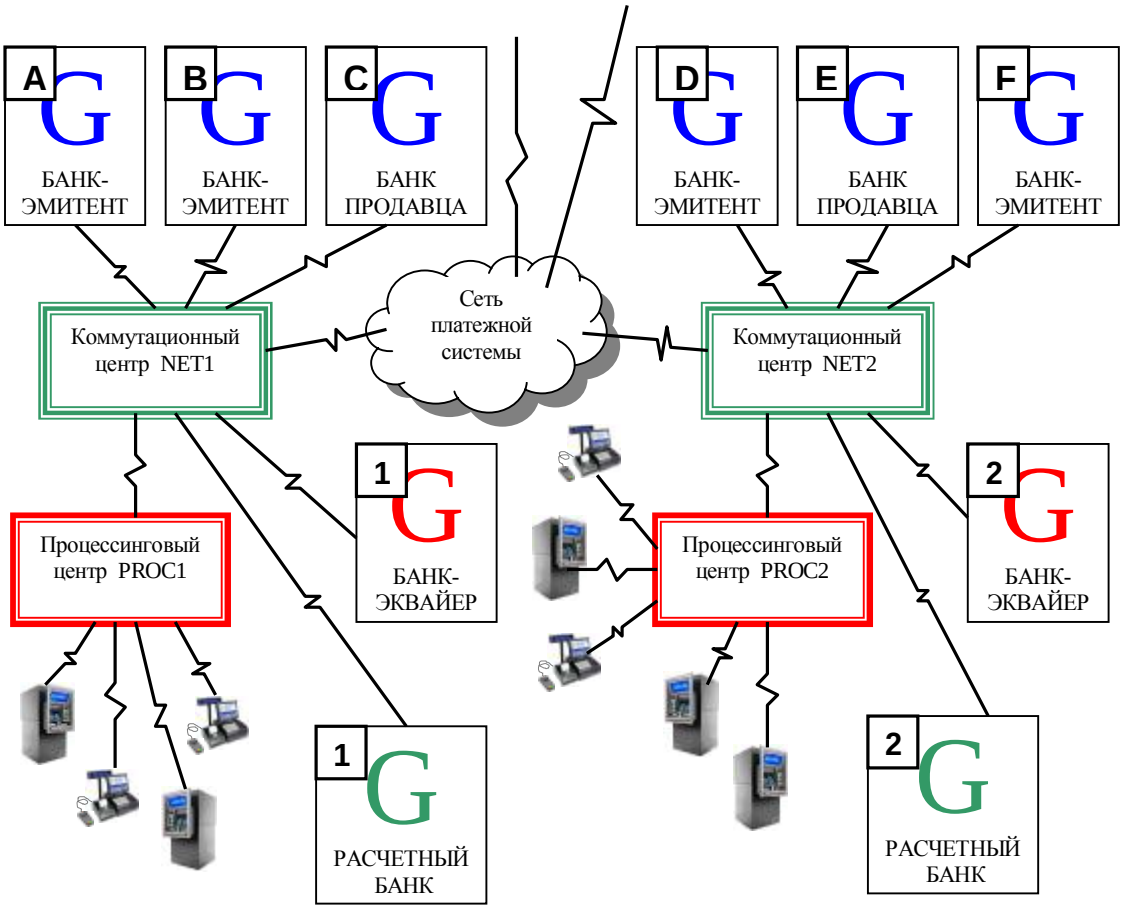


Рис. 7.4. Схема развернутой платежной системы с сетевой организацией процессинговых центров.

Техническое взаимодействие с точками обслуживания осуществляют процессинговые центры. Банки-эквайеры осуществляют лишь функции расчетов с обслуживающими карточками предприятиями. Основная часть входящих в систему банков – чистые эмитенты, функционирующие как on-line банки. Авторизацию для оставшихся off-line банков выполняют процессинговые центры. Они же авторизуют и транзакции on-line банков при технических сбоях – нарушении связи соответствующего локального коммуникационного центра с банком, отказе вычислительных средств банка и пр. перенаправление транзакций в этом случае осуществляет локальный коммуникационный центр. Так, в случае отказа ЭВМ

банка А авторизация его транзакций будет осуществляться процессинговым центром PROC1.

Процессинговые центры выполняют и широкий спектр сервисных операций – персонализацию карточек по заказу эмитентов, техническое обслуживание периферийных – банкоматов и торговых терминалов и др.

Отметим, что авторизация, которую при наступлении форс-мажорных обстоятельств, авторизация осуществляется процессинговыми и коммуникационными центрами по упрощенному сценарию с ограниченными лимитами, рассчитанному на несколько дней.

Рассмотренная схема не исчерпывают всех возможных вариантов организации платежных систем с on-line авторизацией.

7.1.8.2. OFF-LINE СИСТЕМЫ

Для систем с off-line авторизацией проблема маршрутизации транзакций имеет меньшее значение. Действительно, авторизация (и, следовательно, утверждение транзакции) в таких системах происходит «на месте» – непосредственно в точке обслуживания. Пересылка же транзакций для обеспечения проведения взаиморасчетов происходит не в режиме реального времени. Кроме того, формирование итоговых данных для проведения взаиморасчетов происходит в одном или нескольких процессинговых центрах, что также уменьшает требования к коммуникационным возможностям системы.

Структура системы с off-line авторизацией в принципе ни чем не отличается от приведенных в предыдущем разделе схем. Отличия касаются информационных потоков. Утвержденные транзакции накапливаются в течение дня в точках обслуживания. По завершению дня в ходе специальных сеансов транзакции либо непосредственно, либо через промежуточные центры пересылаются в процессинговый центр, где и происходит их итоговая обработка. Кроме того, они могут пересылаться и банкам-эмитентам. Однако такая рассылка может происходить в разных формах – частности, вместе с итоговыми показателями из процессингового центра системы.

Следует, однако, учитывать, что и в системах такого типа ряд операций – например, «кредитование» карты или авторизация транзакций на особо крупные суммы может потребовать использования on-line режима со всеми вытекающими отсюда последствиями.

7.1.9. ПРОВЕДЕНИЕ РАСЧЕТОВ

Возмещение средств происходит в следующем порядке (рис. 7.5):

1) После получения из точек обслуживания протоколов транзакций (в виде чеков или в электронном виде) банки-эквайеры перечисляют средства на счет точек.
2) Банки-эквайеры сортируют транзакции на «свои», относящиеся к держателям собственных карточек (если эквайер является также и эмитентом), и чужие.
3) «Чужие» транзакции пересылаются в процессинговый центр системы, и после итоговой обработки в центре расчетный банк кредитует корреспондентские счета эквайеров в соответствующем размере, и, возможно, дебетует их, если из других банков поступили транзакции по карточкам, принадлежащим данным эквайерам. Кроме того, расчетный банк дебетует корреспондентские счета «чистых» эмитентов.

4) Банки-участники системы осуществляют необходимое перечисление средств на свои корреспондентские счета (или снятие с корреспондентских счетов) в расчетном банке.
5) Банки-эмитенты осуществляют взыскание средств с держателей карточек.

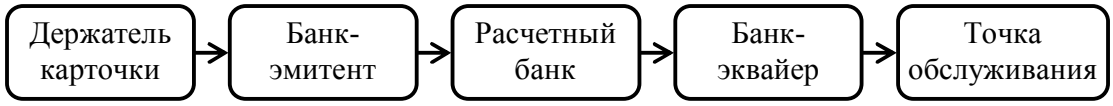


Рис.7.5. Движение средств в платежной системе

При осуществлении некоторых из перечисленных выше платежей могут взиматься комиссионные. Так, банк-эквайер может брать комиссионные от сумм сделок с точки обслуживания. Такая практика является общепринятой в мире и мотивируется тем, что обслуживание карточек способствует привлечению покупателей и увеличению оборота точки. При этом иногда говорят, что эквайер выкупает торговые векселя у точки обслуживания по неполной стоимости. Что же касается России, то во многих случаях такой вид комиссионных не взимается, поскольку из-за малой распространенности карточек на сегодняшний день преимущества их обслуживания не очевидны для торговых и сервисных предприятий. В свою очередь, банк-эквайер обычно уплачивает комиссионные – часть торговой скидки – банку-эмитенту при обслуживании его карточек («чужих» карточек). За выдачу наличных по «чужим» карточкам могут браться дополнительные комиссионные. Расчетный банк может также взимать комиссионные за расчетное обслуживание. Плата взимается также сетями передачи данных. Все это формирует возможные источники дохода для участников платежной системы.

Источником конфликтов при проведении расчетов является отказ от оплаты того или иного участника платежной цепочки в следствии сомнений в корректности проведенных операций. Например, платежная система может посчитать, что поступившие от банка-эквайера чеки оформлены с нарушением правил. В этом случае заинтересованная сторона (в нашем случае банк-эквайер, уже заплативший в соответствии с правилами точке обслуживания) проводит разбор ситуации. В результате банк-эквайер либо может добиться от платежной системы перечисления средств, либо в свою очередь, попытаться предъявить претензии к торговой точке. Для проведения подобной деятельности в банке-эквайере должно существовать специальное подразделение, способное провести анализ ситуации, предложить возможные варианты преодоления возникшей коллизии и выработать рекомендации по предотвращению повторения конфликта в будущем.

7.2. ЭЛЕКТРОННЫЕ ДЕНЬГИ

7.2.1. ПОНЯТИЕ ЭЛЕКТРОННЫХ ДЕНЕГ

Электронные деньги полностью моделируют реальные деньги. При этом, эмиссионная организация - эмитент - выпускает их электронные аналоги, называемые в разных системах по-разному (например, купоны). Далее, они покупаются пользователями, которые с их помощью оплачивают покупки, а затем продавец погашает их у эмитента. При эмиссии каждая денежная единица заверяется электронной печатью, которая проверяется выпускающей структурой перед погашением.

Одна из особенностей физических денег - их анонимность, то есть на них не указано, кто и когда их использовал. Некоторые системы, по аналогии, позволяют покупателю получать электронную наличность так, чтобы нельзя было определить связь между ним и деньгами. Это осуществляется с помощью схемы слепых подписей.

Стоит еще отметить, что при использовании электронных денег отпадает необходимость в аутентификации, поскольку система основана на выпуске денег в обращение перед их использованием.

На рис.7.6 приведена схема платежа с помощью цифровых денег.

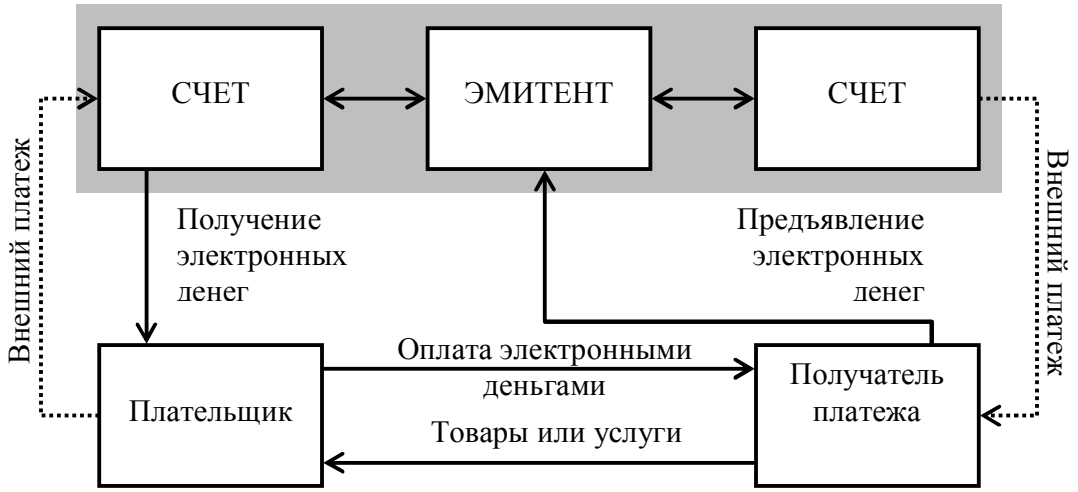


Рис.7.6. Схема платежа с помощью электронных денег

Покупатель заранее обменивает реальные деньги на электронные. Хранение наличности у клиента может осуществляться двумя способами, что определяется используемой системой:

- на жестком диске компьютера.
- на смарт-картах.

Разные системы предлагают разные схемы обмена. Некоторые открывают специальные счета, на которые перечисляются средства со счета покупателя в обмен на электронные купюры. Некоторые банки могут сами эмитировать электронную наличность. При этом она эмитируется только по запросу клиента с последующим ее перечислением на компьютер или карту этого клиента и снятием денежного эквивалента с его счета. При реализации же слепой подписи покупатель сам создает электронные купюры, пересылает их в банк, где при поступлении реальных денег на счет они заверяются печатью и отправляются обратно клиенту. Наряду с удобствами такого хранения, у него имеются и недостатки. Порча диска или смарт-карты оборачивается невозвратимой потерей электронных денег.

Покупатель перечисляет на сервер продавца электронные деньги за покупку.

Деньги предъявляются эмитенту, который проверяет их подлинность.

В случае подлинности электронных купюр счет продавца увеличивается на сумму покупки, а покупателю отгружается товар или оказывается услуга.

Одной из важных отличительных черт электронных денег является возможность осуществлять микроплатежи. Это связано с тем, что номинал купюр

может не соответствовать реальным монетам (например, 37 копеек). Эмитировать электронные наличные могут как банки, так и небанковские организации. Однако до сих пор не выработана единая система конвертирования разных видов электронных денег. Поэтому только сами эмитенты могут гасить выпущенную ими электронную наличность. Кроме того, использование подобных денег от нефинансовых структур не обеспечено гарантиями со стороны государства. Однако, малая стоимость транзакции делает электронную наличность привлекательным инструментом платежей в Интернет.

Кредитные системы.

Интернет-кредитные системы являются аналогами обычных систем, работающих с кредитными картами. Отличие состоит в проведении всех транзакций через Интернет, и как следствие, в необходимости дополнительных средств безопасности и аутентификации.

Общая схема платежей в такой системе приведена на рис.7.7.

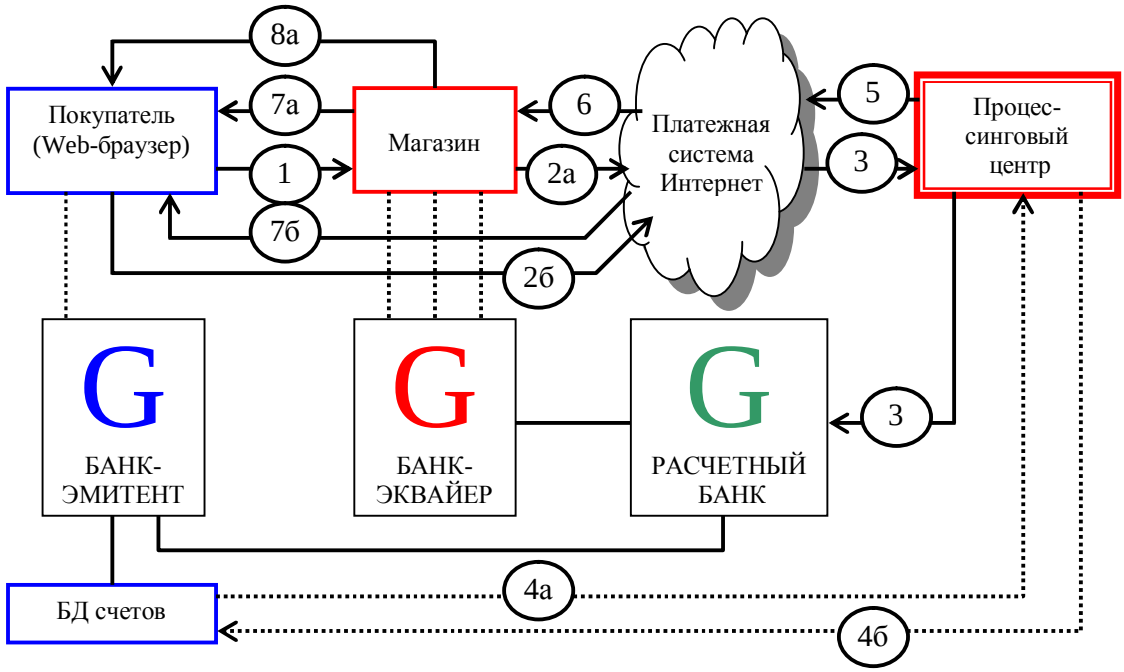


Рис.7.7. Схема платежной системы в сети Интернет

В проведении платежей через Интернет с помощью кредитных карточек участвуют:

Покупатель. Клиент, имеющий компьютер с Web-браузером и доступом в Интернет.

Банк-эмитент. Здесь находится расчетный счет покупателя. Банк-эмитент выпускает карточки и является гарантом выполнения финансовых обязательств клиента.

Продавцы. Под продавцами понимаются сервера Электронной Коммерции, на которых ведутся каталоги товаров и услуг и принимаются заказы клиентов на покупку.

Банки-эквайеры. Банки, обслуживающие продавцов. Каждый продавец имеет единственный банк, в котором он держит свой расчетный счет.

Платежная система Интернет. Электронные компоненты, являющиеся посредниками между остальными участниками.

Традиционная платежная система. Комплекс финансовых и технологических средств для обслуживания карт данного типа. Среди основных задач, решаемых платежной системой, - обеспечение использования карт как средства платежа за товары и услуги, пользование банковскими услугами, проведение взаимозачетов и т.д. Участниками платежной системы являются физические и юридические лица, объединенные отношениями по использованию кредитных карт.

Процессинговый центр платежной системы. Организация, обеспечивающая информационное и технологическое взаимодействие между участниками традиционной платежной системы .

Расчетный банк платежной системы. Кредитная организация, осуществляющая взаиморасчеты между участниками платежной системы по поручению процессингового центра.

Покупатель в электронном магазине формирует корзину товаров и выбирает способ оплаты "кредитная карта".

Далее, параметры кредитной карты (номер, имя владельца, дата окончания действия) должны быть переданы платежной системе Интернет для дальнейшей авторизации. Это может быть сделано двумя способами:

- через магазин, то есть параметры карты вводятся непосредственно на сайте магазина, после чего они передаются платежной системе Интернет (2а);
- на сервере платежной системы (2б).

Очевидны преимущества второго пути. В этом случае сведения о картах не остаются в магазине, и, соответственно, снижается риск получения их третьими лицами или обмана продавцом. И в том, и в другом случае при передаче реквизитов кредитной карты, все же существует возможность их перехвата злоумышленниками в сети. Для предотвращения этого данные при передаче шифруются. Шифрование, естественно, снижает возможности перехвата данных в сети, поэтому связи покупатель/продавец, продавец/платежная система Интернет, покупатель/платежная система Интернет желательно осуществлять с помощью защищенных протоколов. Наиболее распространенным из них на сегодняшний день является протокол SSL (Secure Sockets Layer). В его основе лежит схема асимметричного шифрования с открытым ключом, а в качестве шифровальной схемы используется алгоритм RSA. Ввиду технических и лицензионных особенностей этого алгоритма он считается менее надежным, поэтому сейчас постепенно вводится стандарт защищенных электронных транзакций SET (Secure Electronic Transaction), призванный со временем заменить SSL при обработке транзакций, связанных с расчетами за покупки по кредитным картам в Интернет. Среди плюсов нового стандарта можно отметить усиление безопасности, включая возможности аутентификации всех участников транзакций. Его минусами являются технологические сложности и высокая стоимость. Платежная система Интернет передает запрос на авторизацию традиционной платежной системе.

Последующий шаг зависит от того, ведет ли банк-эмитент онлайновую базу данных (БД) счетов. При наличии БД процессинговый центр передает банку-эмитенту запрос на авторизацию карты (4б) и затем, (4а) получает ее результат. Если же такой базы нет, то процессинговый центр сам хранит сведения о состоянии счетов держателей карт, стоп-листы и выполняет запросы на авторизацию. Эти сведения регулярно обновляются банками-эмитентами. Результат авторизации передается платежной системе Интернет. Магазин получает результат авторизации.

Покупатель получает результат авторизации через магазин (7а) или непосредственно от платежной системы Интернет (7б). При положительном результате авторизации магазин оказывает услугу, или отгружает товар (8а);

процессинговый центр передает в расчетный банк сведения о совершенной транзакции (8б). Деньги со счета покупателя в банке-эмитенте перечисляются через расчетный банк на счет магазина в банке-эквайере. Для проведения подобных платежей в большинстве случаев необходимо специальное программное обеспечение. Оно может поставляться покупателю, (называемое электронным кошельком), продавцу и его обслуживающему банку. Для примера рассмотрим систему электронных платежей WebMoney Transfer.

7.2.2. СИСТЕМА ЭЛЕКТРОННЫХ ПЛАТЕЖЕЙ В СЕТИ ИНТЕРНЕТ WEBMONEY TRANSFER

7.2.2.1. WebMoney Transfer

WebMoney Transfer – глобальная информационная система трансфера имущественных прав, открытая для свободного использования всеми желающими. Система Webmoney Transfer создавалась специально для сети Интернет, она имеет универсальную гибкую структуру, обеспечивающую работу с любыми товарами. Она предоставляет возможность любому пользователю сети Интернет осуществлять безопасные наличные расчеты в реальном времени. Клиентами системы являются Продавцы и Покупатели товаров и услуг. С одной стороны, это WEB-магазины, с другой - любой пользователь Интернета, не имеющий возможности или не желающий использовать традиционные методы расчетов (кредитные карточки и т.п.) из-за длительности транзакций, низкой безопасности, или по другим причинам. С помощью WebMoney Transfer можно совершать мгновенные транзакции, связанные с передачей имущественных прав на любые online-товары и услуги, создавать собственные web-сервисы и сетевые предприятия, проводить операции с другими участниками, выпускать и обслуживать собственные инструменты.

Транзакционным средством в системе служат титульные знаки WebMoney (WM) нескольких типов, хранящиеся на электронных кошельках их владельцев: WM-R – эквивалент RUR – на R-кошельках, WM-Z – эквивалент USD – на Z-кошельках, WM-C и WM-D – эквивалент USD для кредитных операций – на C- и D-кошельках. При переводе средств используются однотипные кошельки, а обмен WM-R на WM-Z производится в обменных пунктах.

Стать Клиентом Системы позволяет Клиентское программное обеспечение - WEBMONEY KEEPER. С помощью этой программы возможно фиксировать определенные суммы для расчетов, контролировать движения собственных или перечисленных владельцу кошелька средств.

Получить WebMoney на кошелек возможно:

- переводом из любого банка, а также почтовым переводом на расчетный счет одного из официальных агентов системы (сумма перевода будет автоматически конвертирована в WM и зачислена на указанный вами кошелек);
- с помощью WM-карты (для Z-кошельков);
- от других участников системы в обмен на товары, услуги или наличные деньги.

Хранящиеся на кошельке WebMoney находятся в полном распоряжении владельца кошелька и в любой момент – круглосуточно и ежедневно - могут быть использованы им для расчетов. При необходимости он сможет снять WebMoney с кошелька и перевести на указанный банковский счет с одновременной конвертацией в соответствующую валюту.

7.2.2.2. ТЕХНОЛОГИЯ

Средства программно-аппаратного комплекса WebMoney Transfer позволяют осуществлять мгновенные безопасные транзакции, для проведения которых достаточно иметь подключенный к сети Интернет компьютер. Действующий в системе порядок идентификации позволяет однозначно определять и фиксировать все проводимые операции, а специальный комплекс мер безопасности полностью исключает несанкционированный доступ к средствам и информации. С помощью встроенной службы конфиденциальных сообщений можно вести защищенную переписку с другими участниками, обсуждать детали сделок, комментировать проведение транзакций.

7.2.2.3. БЕЗОПАСНОСТЬ ФИНАНСОВЫХ ТРАНЗАКЦИЙ

При использовании WebMoney Transfer, WEBMONEY приходят к получателю по сети Интернет. Открытая архитектура Интернет требует строгих мер безопасности против попыток перехватить WEBMONEY или информацию о торговой сделке.

Рассмотрим подробно некоторые аспекты безопасности, примененные в WebMoney Transfer.

1) Для входа в программу WEBMONEY KEEPER необходимо знание уникального 12-значного идентификатора пользователя, его личного пароля, а также места расположения в памяти компьютера файлов с секретным ключом и кошельками.

   Идентификатор - генерируется автоматически, уникален для каждой регистрации участника в WebMoney Transfer. Это имя пользователя в системе, которое пользователь может сделать анонимным для других пользователей. Необходим для входа в программу WEBMONEY KEEPER и осуществления сделок в системе Webmoney Transfer.

   Пароль – назначается пользователем лично. Необходим для входа в программу WEBMONEY KEEPER и осуществления сделок в системе Webmoney Transfer.

   Перевод и получение денежных средств осуществляется только между однотипными кошельками клиентов Системы.
   Для осуществления сделок необходимо сообщить партнеру ***номер кошелька***, при этом он сможет только отправить деньги на кошелек пользователя (и пользователь может отказаться от их принятия); и никто не сможет снять деньги с кошелька пользователя с удаленного компьютера. Более того, пользователь может легко и быстро создавать отдельный кошелек для разовой сделки и по ее завершении удалять его.

   Невозможно снять деньги с использованного кошелька (этой возможности лишена, например, система оплаты с помощью кредитных карт).

Подобное сочетание личных настроек и случайным образом генерированных настроек программы гарантирует невозможность несанкционированного использования программы и получения доступа к средствам пользователя третьими лицами.

2) Все сообщения в системе передаются в закодированном виде, с использованием алгоритма защиты информации подобного RSA с длиной ключа более 1024 бит. Для каждого сеанса используются уникальные сеансовые ключи. Поэтому в течение сеанса (времени осуществления транзакции) никто, кроме пользователя, не имеет возможности определить назначение платежа и его сумму.
3) Никто не сможет совершить никаких денежных операций, основываясь на реквизитах прошлых сделок (этой возможности лишена, например, система оплаты с помощью кредитных карт). Для каждой сделки используются уникальные реквизиты, и попытка использовать их вторично немедленно отслеживается и гасится.
4) Устойчивость по отношению к обрывам связи. Если любая операция в системе не была успешно завершена по причине обрыва связи, то система не учитывает данную операцию.

Таким образом, Клиент WebMoney Transfer, соблюдающий элементарные правила предосторожности по сохранности своих единиц WEBMONEY, идентификатора пользователя, пароля и секретного ключа, может быть уверен в безопасности управления своими денежными средствами с помощью WEBMONEY KEEPER. Очевидно, что реальная безопасность WebMoney значительно выше, чем у любых иных средств расчетов "через Интернет".

Все процессы, совершаемые в системе, - хранение WebMoney на кошельках, выписка счетов, WM-расчеты между участниками, обмен сообщениями - выполняются с использованием алгоритма кодирования, эквивалентного RSA, с длиной ключей не менее 1024 бит, что определяет абсолютную устойчивость системы к взломам и обрывам связи. Для каждой транзакции назначаются уникальные сеансовые реквизиты, и попытка их повторного использования мгновенно отслеживается и пресекается. Если та или иная операция не была успешно завершена, она не учитывается системой.

Анонимность. При желании клиент может не указывать истинных сведений о себе (имя, фамилия, E-mail, почтовый адрес, номера банковских счетов и т.п.) при регистрации в программе WEBMONEY KEEPER на сервере платежной системы, при осуществлении операций в системе WebMoney Transfer.

ЛИТЕРАТУРА

1. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии: Учебное пособие. – М.: Гелиос АРБ, 2001. – 480 с.
2. Анин Б.Ю. Защита компьютерной информации. – СПб.: БХВ-Санкт-Петербург, 2000. –384 с.
3. Асанович В.Я., Маньшин Г.Г. Информационная безопасность: анализ и прогноз информационного взаимодействия. – Мн.: Амалфея, 2006. – 204 с.
4. Аршинов М. Н., Садовский Л. Е. Коды и математика (рассказы о кодировании). – М.: Наука, 1983. –144 с.
5. Баричев С.Г., Гончаров В.В., Серов Р.Е. Основы современной криптографии: Учебное пособие. – М.: Горячая линия – Телеком, 2001. – 120 с.
6. Бармен С. Разработка правил информационной безопасности. – М.: Вильямс, 2002. – 208 с.
7. Безопасность сети на основе Windows 2000. Учебный курс MCSE. – М.: ИТД Русская Редакция. 2001. – 912 с.
8. Белов Е.Б., Лось В.П., Мещеряков Р.В., Шелупанов А.А. Основы информационной безопасности. Учебное пособие для вузов, – М.: Горячая линия – Телеком, 2006. – 544 с.
9. Бернет С., Пэйн С. Криптография. Официальное руководство RSA Security. – М.: Бином-Пресс, 2002. – 384 с.
10. Блейхут Р. Теория и практика кодов, контролирующих ошибки: Пер. с англ. – М.: Мир, 1986. – 576 с.
11. Варакин Л. Е, Системы связи с шумоподобными сигналами. – М.: Радио и связь, 1985. – 384 с.
12. Варфоломеев А.А., Пеленицин М.Б. Методы криптографии и их применение в банковских технологиях. – М.: МИФИ, 1995. – 116 стр.
13. Варфоломеев А.А., Гаврилкевич М.В., Устюжанин Д.Д., Фомичев В.М. Методические указания к выполнению лабораторного практикума “Информационная безопасность. Криптографические методы защиты информации”, ч.1 ,ч.2. – М.: МИФИ, 1995. – 44 с., – 38с.
14. Варфоломеев А.А., Жуков А.Е. и др. Блочные криптосистемы. Основные свойства и методы анализа стойкости. – М.: МИФИ, 1998. – 198 с.
15. Венбо Мао. Современная криптография: теория и практика.: Пер. с англ. – М.: Вильямс, 2005. – 768 с.
16. Габидулин, Э. М., Пилипчук, Н. И. Лекции по теории информации. – М.: МФТИ, 2007. – 214 с.
17. Гайкович В.Ю., Першин А.Ю. Безопасность электронных банковских систем. – М., 1994. – 363 с.
18. Галатенко В.А. Стандарты информационной безопасности. Курс лекций. – М.: ИНТУИТ. РУ, 2004. – 328 с.
19. Галатенко В.А. Основы информационной безопасности. Курс лекций. – М.: ИНТУИТ. РУ, 2006г. – 205 с.
20. Галицкий А.В., Рябко С.Д., Шаньгин В.Ф. Защита информации в сети – анализ технологий и синтез решений. – М.: ДМК Пресс, 2004. – 616 с.
21. Герасименко В.А., Малюк А.А. Основы защиты информации. – М.: МИФИ, 1997. – 538 с.

22. Голдовский И. Безопасность платежей в Интернете. – СПб: Питер, 2001. – 240 с.

23. Государственная тайна и ее защита. Собрание законодательных и нормативных правовых актов. – М.: «Ось-89», 2004. – 160 с.

24. ГОСТ 28147-89. Система обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. – М.: Госстандарт СССР, 1989.

25. ГОСТ Р 34.10-2001. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи.

26. ГОСТ Р 34.10-94. Информационная технология. Криптографическая защита информации. Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма. – М.: Госстандарт России, 1994.

27. ГОСТ Р 34.11-94. Информационная технология. Криптографическая защита информации. Функция хэширования. – М.: Госстандарт России, 1994.

28. ГОСТ Р ИСО/МЭК 15408-1-2002. Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 1. Введение и общая модель. – М.: ИПК «Издательство стандартов», 2002.

29. ГОСТ Р ИСО/МЭК 15408-2-2002. Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 2. Функциональные требования безопасности. – М.: ИПК «Издательство стандартов», 2002.

30. ГОСТ Р ИСО/МЭК 15408-3-2002. Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 3. Требования доверия к безопасности. – М.: ИПК «Издательство стандартов», 2002.

31. Горохов П.К. Информацинная безопасность. Англо-русский словарь. – М.: Радио и связь, 1995. – 224 с.

32. Девянин П.Н., Михальский О.О., Правиков Д.И., Щербаков А.Ю. Теоретические основы компьютерной безопасности, – М: Радио и связь, 2000. – 192 с.

33. Девянин П. Н. Модели безопасности компьютерных систем: Уч. пособие для студентов ВУЗов.– М.: Академия, 2005. – 144 с.

34. Диффи У. Первые десять лет криптографии с открытым ключом // ТИИЭР. – 1988. – Т. 76. – № 5. – с. 54–74.

35. Домашев А.В., Попов В.О., Правиков Д.И., Прокофьев И.В., Щербаков А.Ю. Программирование алгоритмов защиты информации: Учебное пособие. – М.: Нолидж, 2000. – 288 с.

36. Дшхунян В. Л., Шаньгин В. Ф. Электронная идентификация. Бесконтактные электронные идентификаторы и смарт-карты. – М.: НТ Пресс, 2004. – 695 с.

37. Запечников С.В., Милославская Н.Г., Толстой А.И., Ушаков Д.В. Информационная безопасность открытых систем: Уч. Для вузов. В 2-х томах. Том 1 – Угрозы, уязвимости, атаки и подходы к защите. М.: Горячая линия – Телеком, 2006. – 536 с.

38. Защита информации. Специальные защитные знаки. Классификация и общие требования. – М.: Гостехкомиссия России, 1992.

39. Зегжда Д. П., Ивашко А. М. Основы безопасности информационных систем. – М.: Горячая линия – Телеком, 2000. – 452 с.

40. Земор Ж. Курс криптографии. – М.-Ижевск: НИЦ"Регулярная и хаотическая динамика"; Институт компьютерных исследований, 2006. – 256.

41. Зима В. М., Молдовян А. А., Молдовян Н. А. Безопасность глобальных сетевых технологий. – СПб.: БХВ-Петербург, 2000. – 320 с.

42. Зима В. М., Молдовян А. А., Молдовян Н. А. Компьютерные сети и защита передаваемой информации. – СПб: Издательство СПбГУ, 1998. – 328 с.

43. Зубов А.Ю. Математика кодов аутентификации. – М.: Гелиос АРБ, 2007. – 480 с.

44. Казанцев С.Я., Згадзай О.Э., Оболенский Р.М., Белов Е.Б., Полникова С.В. Правовое обеспечение информационной безопасности: Учеб. пособие. – М.: Издательский центр «Академия», 2005. – 240 с.

45. Касперский Е. Компьютерные вирусы: что это такое и как с ними бороться. – М.: СК Пресс, 1998. – 285 с.

46. Конеев И., Беляев А. Информационная безопасность предприятия. – СПб.: БХВ-Петербург, 2003. – 752 с.

47. Лукацкий А. Обнаружение атак – СПб.: БХВ-Петербург, 2003. – 608 с.

48. Малюк А.А., Пазизин С.В., Погожин Н.С. Введение в защиту информации в автоматизированных системах – М.: Горячая линия-телеком, 2001. – 148 с.
Мамаев М., Петренко С. Технологии защиты информации в Интернете. Специальный справочник. – СПб.: Питер, 2002. – 848 с.

49. Меньшаков Ю.К. Защита объектов и информации от технических средств разведки. Учеб. пособие. – М.: РГГУ, 2002. – 399 с.

50. Милославская Н.Г., Толстой А.И. Интрасети:доступ в Internet, защита: Учебное пособие. – М.: ЮНИТИ-ДАНА, 2000. – 527 с.

51. Михайлов С.Ф., Петров В.А., Тимофеев Ю.А. Информационная безопасность. Защита информации в автоматизированных системах. Основные концепции. Учеб. пособие. – М.: МИФИ, 1995. – 112 с.

52. Молдовян А.А., Молдовян Н.А., Советов Б.Я. Криптография. – СПб.: Издательство «Лань»,2000. – 224 с.

53. Никитин Г. И. Помехоустойчивые циклические коды: Учеб. пособие/: СПбГУАП. СПб., 2003. – 33 с.

54. Никитин Г.И., Поддубный С.С. Помехоустойчивые циклические коды: Учеб. пособие/: СПбГУАП. СПб., 1998. – 72 с.

55. Новиков А.А., Устинов Г.Н. Уязвимость и информационная безопасность телекоммуникационных технологий: Учеб. пособие. – М.: Радио и связь. 2003. – 296 с.

56. Осипенко А.Л. Борьба с преступностью в глобальных компьютерных сетях: Международный опыт: Монография. – М.: Норма, 2004. – 432 с.

57. Партыка Т.Л., Попов И.И. Информационная безопасность. Учебное пособие. – М.: ФОРУМ: ИНФРА-М. 2006. – 368 с.

58. Парфенов В.И. Защита информации. Словарь. Воронеж.: НП РЦИБ Факел. 2003, – 292 с.

59. Петренко С.А., Курбатов В.А. Политики информационной безопасности. – М.: Компания АйТи, 2006. – 400 с.

60. Петров А.А. Компьютерная безопасность. Криптографические методы защиты. – М.:ДМК, 2000. – 448 с.

61. Питерсон У., Уэлдон Э, Коды, исправляющие ошибки: Пер. с англ. – М.: Мир, 1976. – 600 с.

62. Проскурин В.Г., Крутов С.В., Мацкевич И.В. Защита в операционных системах. – М: Радио и связь, 2000. – 168 с.

63. Пярин В.А., Кузьмин А.С., Смирнов С.Н. Безопасность электронного бизнеса. – М.: Гелиос АРБ, 2002. – 432 с.

64. Расторгуев С.П. Программные методы защиты информации в компьютерах и сетях. – М.: Яхтсмен, 1993. – 188 с.

65. Романец Ю.Ф., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях./ Под ред. В.Ф. Шаньгина. – М: Радио и связь, 2001. – 376 с.

66. Семкин С.Н., Беляков Э.В., Гребенев С.В., Козачок В.И. Основы организационного обеспечения информационной безопасности объектов информатизации: Учеб. пособие. – М.: Гелиос АРВ, 2005. – 192 с.

67. Сердюк В.А. Новое в защите от взлома корпоративных систем. – М.: Техносфера, 2007. – 360 с.

68. Скляров Д.В., Искусство защиты и взлома информации. – СПб.: БХВ - Петербург, 2004. – 288 с.

69. Смарт Н. Криптография. – М.: Техносфера, 2005. – 528 с.

70. Смит Р. Аутентификация: от паролей до открытых ключей. – М.: Вильямс, 2002. – 432 с.

71. Снытников А.А. Лицензирование и сертификация в области защиты информации. – М.: Гелиос АРВ, 2003. – 192 с.

72. Спесивцев А.В., Вегнер В.А. и др. Защита информации в персональных ЭВМ. – М.: Радио и связь, 1992. – 191 с.

73. Соболев А.Н., Кириллов В.М. Физические основы технических средств обеспечения информационной безопасности: Учебное пособие. – М.: Гелиос АРВ, 2004. – 144 с.

74. Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах. – М.: ДМК Пресс, 2002. – 656 с.

75. Столлингс В. Криптография и защита сетей: принципы и практика, 2-е изд. – М.: Вильямс, 2001. – 672 с.

76. Стрельцов А.А. Правовое обеспечение информационной безопасности России: теоретические и методологические основы. – Минск.: БЕЛЛІТФОНД, 2005. – 304 с.

77. Стрельцов А.А. Обеспечение информационной безопасности России. – М.: МЦНМО, 2002. – 296 с.

78. Тайли Э. Безопасность персонального компьютера. – Мн.: ООО Попурри. 1997. – 480 с.

79. Тилборг Ван Х.К.А. Основы криптологии. Профессиональное руководство и интерактивный учебник. – М.; Мир, 2006, – 471 с.

80. Тихонов В.А., Райх В.В. Информационная безопасность: концептуальные, правовые, организационные и технические аспекты: учеб. пособие. – М.: Гелиос АРВ, 2006. – 528 с.

81. Торокин А.А. Основы инженерно-технической защиты информации. – М.: Ось-89, 1998. – 336 с.

82. Трубачев А.П., Долинин М.Ю., Кобзарь М.Т., Сидак А.А., Сороковиков В.И. Оценка безопасности информационных технологий. – М.: Издательство СИП РИА, 2001. – 356 с.

83. Устинов Г.Н. Основы информационной безопасности систем и сетей передачи данных. Учебное пособие. – М.: СИНТЕГ, 2000, – 248 с.

84. Фатьянов А.А. Правовое обеспечение безопасности информации в Российской Федерации. – М.: Юрист, 2001 г., – 415 с.

85. Филин С.А. Информационная безопасность: Уч. Пособие. – М.: Альфа-Пресс, 2006. – 412с.

86. Фисун А.П., Касилов А.Н., Глоба Ю.А., Савин В.И., Белевская ю.А. Право и информационная безопасность: Учебное пособие// – М.: Приор-издат, 2005. – 272 с.

87. Харин Ю.С., Агиевич С.В. Компьютерный практикум по математическим методам защиты информации. – Мн.:БГУ, 2001. – 190 с.

88. Чмора А.Л. Современная прикладная криптография. 2-е изд., стер. – М.: Гелиос АРБ, 2002. – 256 с.

89. Шаньгин В.Ф. Информационная безопасность компьютерных систем и сетей: учебн. Пособие.- М.: ИД «ФОРУМ»: ИНФРА-М, 2008. – 416 с.

90. Шаньгин В.Ф. Защита компьютерной информации. Эффективные методы и средства – М.: ДМК Пресс, 2010. – 544 с.

91. Шеннон К. Работы по теории информации и кибернетике. – М.: Изд-во иностранной литературы, 1963. – 830 с.

92. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. – М.: Триумф, 2002. – 816 с.

93. Шумский А.А., Шелупанов А.А. Системный анализ в защите информации: Учеб. пособие. – М.: Гелиос АРВ, 2005. – 224 с.

94. Эдвардс М.Дж. Безопасность в Интернете на основе Windows NT. – М.: ТОО Channel Trading Ltd. 1999. – 656 с.

95. Ярочкин В.И. Информационная безопасность. – М.: Международные отношения, 2000. – 400 с.

Интернет-ресурсы

96. Базовый стандарт организации беспроводных локальных сетей IEEE 802.11 // http://standards.ieee.org/reading/ieee/std/lanman/802.11-1999.pdf

97. Беляев А. В. Методы и средства защиты информации // http://www.citforum.ru/ internet/infsecure/its2000_01.shtml

98. Касперский Е. Компьютерные вирусы // http://www.kaspersky.ru/

99. Коротыгин С. Развитие технологии беспроводных сетей: стандарт IEEE 802.11 // http://www.ixbt.com/comm/wlan.shtml

100. Кузнецов С. Защита файлов в операционной системе UNIX // http:// www.citforum.ru/database/articles/art_8.shtml

101. Олифер Н. А., Олифер В. Г. Сетевые операционные системы // http://citforum.ru/ operating_systems/sos/contents.shtml

102. Семейство стандартов IEEE 802.11 // http://www.wireless.ru/wireless/ wrl_base80211

103. Скородумов Б. И. Стандарты для безопасности электронной коммерции в сети Интернет // http://www.stcarb.comcor.ru

104. Advanced Encryption Standard (AES) Development Effort. – Feb. 2001 // csrc.nist.gov/CryptoToolkit/aes/index2.html
105. Daemen J., Rijmen V. AES Proposal: Rijndael. Document version 2. – Sept. 1999 // www.esat.kuleuven.ac.be/~rijmen/rijndael
106. Dierks T., Allen C. RFC 2246: The TLS Protocol Version 1.0. – Jan. 1999 // www.ietf.org/rfc/rfc2246.txt
107. FIPS Publication 197. Announcing the Advanced Encryption Standard (AES). – Nov. 2001 // csrc.nist.gov/publications/fips/fipsl97/fips-197.pdf
108. Hodges J., Morgan R. RFC 3377: Lightweight Directory Access Protocol (v3): Technical Specification. – Sept. 2002 // www.ietf.org/rfc/rfc3377.txt
109. Housley R., Ford W., Polk W. etc. RFC 2459: Internet X.509 Public Key Infrastructure. – Jan. 1999 // www.ietf.org/rfc/rfc2459.txt
110. Kent S., Atkinson R. RFC 2401: Security Architecture for IP. – Nov. 1998 // www.ietf.org/rfc/rfc2401.txt
111. Orman H. RFC 2412: The OAKLEY Key Determination Protocol. – Nov. 1998 // www.ietf.org/rfc/rfc2412.txt
112. PKCS #1 v2.1: RSA Cryptography Standard. RSA Laboratories. – June 2002 // www.rsasecurity.com/rsalabs/pkcs/pkcs-l
113. RFC 1928: SOCKS Protocol Version 5 / M. Leech, M. Ganis, Y. Lee etc. – March 1996 // www.ietf.org/rfc/rfcl928.txt
114. RFC 2311: S/MIME Version 2 Message Specification / S. Dusse, P. Hoffman, B. Ramsdell etc. – March 1998 // www.ietf.org/rfc/rfc231l.txt
115. RFC 2408: Internet Security Association and Key Management Protocol (ISAKMP) / D. Maughan, M. Schertler etc. – Nov. 1998 // www.ietf.org/rfc/rfc2408.txt
116. Rivest R. The MD5 Message-Digest Algorithm. – April 1992 // www.ietf.org/ rfc/rfcl321.txt
117. The RC6 Block Cipher. Version 1.1 / R. Rivest, M. J. B. Robshaw, R. Sidney etc. – Aug. 1998 // www.rsasecurity.com/rsalabs/aes
118. Zeilenga K. RFC 3673: Lightweight Directory Access Protocol version 3 (LDAPv3): All Operational Attributes. – Dec. 2003 // www.ietf.org/rfc/rfc3673.txt