САМАРСКИЙ
УНИВЕРСИТЕТ
ИМЕНИ АКАДЕМИКА
С.П.КОРОЛЕВА

К.Е. КЛИМЕНТЬЕВ

ВВЕДЕНИЕ В ЗАЩИТУ КОМПЬЮТЕРНОЙ ИНФОРМАЦИИ

САМАРА 2020

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)

К.Е. КЛИМЕНТЬЕВ

ВВЕДЕНИЕ
В ЗАЩИТУ КОМПЬЮТЕРНОЙ ИНФОРМАЦИИ

Рекомендовано редакционно-издательским советом федерального государственного автономного образовательного учреждения высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева» в качестве учебного пособия для студентов, обучающихся по основной образовательной программе высшего образования по направлениям подготовки 02.03.02 и 02.04.02 – Фундаментальная информатика и информационные технологии, 09.03.01 и 09.04.01 – Информатика и вычислительная техника, 10.03.01 и 10.04.01 – Информационная безопасность, 10.05.03 – Информационная безопасность автоматизированных систем, 10.05.01 – Компьютерная безопасность, а также иным направлениям смежной тематики.

ISBN 978-5-7883-1526-3

Самара
Издательство Самарского университета
2020

УДК 004.056 (075)
ББК 018.27
К492

Рецензенты: д-р техн. наук, профессор В.В. Сергеев
канд. физ.-мат. наук, доцент В.А. Федосеев

Климентьев, Константин Евгеньевич

Введение в защиту компьютерной информации: учеб. пособие / К.Е. Климентьев. – Электрон. текст. дан. (Мб). – Самара: Издательство Самарского университета, 2020. – 1 опт. компакт-диск (CD-ROM). – Систем. требования: PC, процессор Pentium, 160 МГц; оперативная память 32 Мб; на винчестере 16 Мб; Microsoft Windows XP/Vista/7; разрешение экрана 1024x768 с глубиной цвета 16 бит; DVD-ROM 2-х и выше, мышь; Adobe Acrobat Reader. – Загл. с титул. экрана.

ISBN 978-5-7883-1526-3

В пособии рассматривается основные концепции, методы и средства защиты компьютерной информации в системах обработки информации и управления.

Пособие предназначено для студентов, обучающихся по направлениям подготовки 02.03.02 и 02.04.02 – Фундаментальная информатика и информационные технологии, 09.03.01 и 09.04.01 – Информатика и вычислительная техника, 10.03.01 и 10.04.01 – Информационная безопасность, 10.05.03 – Информационная безопасность автоматизированных систем, 10.05.01 – Компьютерная безопасность. Оно может быть полезно студентам смежных направлений и специальностей.

Пособие подготовлено на кафедре информационных систем и технологий.

УДК 004.056 (075)
ББК 018.27

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность проблемы защиты информации и необходимость изучения связанных с этой проблемой вопросов в настоящее время неоспоримы.

Целью настоящего пособия является введение в проблематику защиты информации, ориентированное на студентов средних и старших курсов специальностей, изучающих информационные технологии. В первую очередь оно предназначено студентам – будущим программистам. Поскольку Автор пособия попытался в своей работе ответить не только на вопросы «что?» и «зачем?», но и «как?», то для успешного восприятия материала от студентов потребуются элементарные знания, полученные в рамках дисциплин «Программирование», «Организация ЭВМ», и «Дискретная математика».

Пособие построено на материалах лекционных курсов «Методы и средства защиты информации», «Защита информации» и «Безопасность операционных систем», в течение нескольких лет читаемых автором для студентов очной, очно-заочной и заочной форм обучения специальностей 09.01.05 «Комплексное обеспечение информационной безопасности автоматизированных систем», 10.05.03 «Информационная безопасность», 23.01.02 «Автоматизированные системы обработки и управления» и 09.03.01 «Информатика и вычислительная техника».

Материал пособия разделен на семь тем.

Тема 1 посвящена изучению основных понятий и определений в области защиты информации. При необходимости материалы этой темы могут быть дополнены сведениями из Темы 7, посвященной вопросам обеспечения информационной безопасности на предприятиях или в организациях.

Тема 2 рассматривает основные концепции и принципы, которые находят свое применение в конкретных методах и средствах защиты информации.

Тема 3 затрагивает вопросы криптографической защиты информации.

Тема 4 посвящена методам и средствам аутентификации (проверки подлинности) участников информационного взаимодействия.

Тема 5 рассматривает вопросы, связанные с защитой программ и данных на компьютерах и мобильных устройствах.

Тема 6 посвящена вопросам защиты информации в вычислительных сетях.

Приложение содержит вопросы и задания, которые можно использовать для закрепления изученного материала.

ТЕМА 1. Основные понятия и определения

Проблема защиты информации сформировала вокруг себя комплексную междисциплинарную систему концепций, принципов, способов их применения и средств их практической реализации. В основе этой системы лежит ряд базовых понятий.

1.1. Различные определения понятия «информация»

Единственного и всеобъемлющего определения понятия «*информация*» (от лат. Informatio – сведение, разъяснение, ознакомление) не существует.

Философы считают «информацию» результатом отражения свойств материального мира в человеческом сознании.

Ученые-прикладники сужают это определение, понимая под «информацией» общенаучное понятие, включающее «обмен сведениями между людьми, человеком и автоматом, автоматом и автоматом; обмен сигналами в животном и растительном мире; передачу признаков от клетки к клетке, от организма к организму».

Еще более узкое понятие дается в «Федеральном законе №149-ФЗ» от 27 июля 2006 г., который служит законодательной основой для любой деятельности, связанной с информационными технологиями: «**информация – сведения (сообщения, данные) независимо от формы их представления**».

В этом определении не различаются понятия «информация» (как знания о мире) и «данные» (как знания, оформленные в виде электронных импульсов, знаков на бумаге или экране, звуков человеческой речи и т.п.) – это сделано преднамеренно, с целью упрощения использования определения на практике. В частности, сложившиеся к настоящему времени социальные и экономические отношения позволяют рассматривать «информацию» как товар, создаваемый посредством определенных трудозатрат и имеющий определенную стоимость. В таком контексте удобней понимать под «информацией» некую материальную сущность, которая может быть изготовлена, передана от одного лица другому, использована, уничтожена и тому подобное.

Но, тем не менее, в широком смысле понятия «информация» и «данные» следует различать.

Единицей измерения информации является «*бит*», то есть количество информации, которое содержится в опыте, имеющем два равновероятных исхода – «истина» или «ложь»[1].

1.2. Информационные технологии

Бурное развитие средств вычислительной техники позволило автоматизировать как процессы управления сложными техническими системами (бытовыми приборами, транспортными средствами, системами связи, технологическими установками, промышленными предприятиями и пр.), так и процессы принятия решений в гуманитарной области – в социологии, экономике, медицине и политике. Технологии, используемые при этом, принято называть «информационными технологиями». Более конкретно, *информационные технологии* – это процессы, методы поиска, сбора, хранения, обработки, предоставления, распространения информации и способы осуществления таких процессов и методов.

[1] Возможны и другие единицы, например, «*триты*» – имеющие состояния «истина», «ложь» и «неопределенность». С точки зрения экономичности представления данных «триты» эффективней «битов», но менее удобны при реализации

Таким образом, информация является основным ресурсом, которым оперируют информационные технологии. Над ней возможны следующие основные операции:

- выработка (поиск, сбор, получение);
- хранение;
- передача (распространение);
- предоставление (обеспечение доступа);
- обработка (преобразование);
- использование;
- уничтожение.

1.3 Качество информации и ее защита

Возможность успешного применения информационных технологий в тех или иных сферах человеческой деятельности зависит, в первую очередь, от «качества» используемой при этом информации.

Качество информации – совокупность свойств, определяющих пригодность информации для удовлетворения определенных потребностей в соответствии с ее назначением.

Основными свойствами, определяющими качество информации, являются:

1) *целостность* – сохранность информации в том виде, в каком она была получена;
2) *конфиденциальность* – недоступность выполнения операций над информацией для тех, кто не имеет на это право;
3) *доступность* – возможность использовать информацию.

Вместо «целостности» иногда рассматривают более широкое понятие «достоверности», которая, кроме собственно «целостности», так же включает «адекватность» (то есть, полноту и точность) информации.

Очевидно, частичная или полная утеря перечисленных свойств резко снижает качество информации, приводит к невозможности ее практического использования и, как следствие, к негативным явлениям во всех сферах человеческой деятельности. Последствия снижения качества информации могут нести катастрофический характер в рамках всей планеты – достаточно вспомнить, что современные системы управления ядерным оружием представляют собой высокоавтоматизированные комплексы, собирающие, обрабатывающие и принимающие решения на основе многочисленных источников различной информации.

Защита информации – процесс поддержания свойств целостности, конфиденциальности и доступности на уровне, обеспечивающем требуемое качество информации. Все методы защиты информации условно разделяют на три группы:

- организационные (административные);
- правовые;
- технические (или инженерно-технические).

Угрозы безопасности информации – факторы, негативно влияющие на основные свойства и, следовательно на качество информации. Конкретных угроз очень много, их можно классифицировать по степени опасности, природе возникновения, характеру проявления и т.п. Но в широком смысле, опираясь на перечень основных свойств, характеризующих качество информации, можно выделить следующие основные классы угроз:

- искажение и уничтожение информации;
- несанкционированный доступ к информации;
- блокирование информации.

Источниками этих угроз являются:

- злоумышленники;
- ошибки в реализации информационных систем и в применении информационных технологий;
- внешние (природные) воздействия.

Самым опасным источником угроз являются, очевидно, злоумышленники – лица и организации, преднамеренно и целенаправленно осуществляющие деятельность по снижению качества информации. В современных условиях основные мероприятия в области защиты информации направлены именно на противодействие злоумышленникам.

1.4. Законодательные основы защиты информации

Основными документами, регламентирующими вопросы защиты информации в РФ являются «Закон об информации» и «Доктрина информационной безопасности РФ». На их основе созданы и подлежат исполнению конкретные кодексы, постановления и приказы, исходящие от Президента РФ, а так же от различных министерств, ведомств, организаций и предприятий.

Примерами документов общегосударственного уровня являются федеральный закон №152-ФЗ «О персональных данных» или указ Президента РФ «Об утверждении перечня сведений конфиденциального характера».

Примером документа, актуального на уровне предприятия, является «Политика информационной безопасности в государственном бюджетном учреждении культуры «Самарская государственная филармония»».

1.5. Органы, осуществляющие защиту информации

Государственными органами РФ, контролирующими и осуществляющими деятельность в области защиты информации, являются:

- Комитет Государственной думы по безопасности;
- Совет безопасности России;
- Федеральная служба по техническому и экспортному контролю (ФСТЭК России);
- Федеральная служба безопасности Российской Федерации (ФСБ России);
- Федеральная служба охраны Российской Федерации (ФСО России);
- Служба внешней разведки Российской Федерации (СВР России);

- Министерство обороны Российской Федерации (Минобороны России);
- Министерство внутренних дел Российской Федерации (МВД России);
- Федеральная служба по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзор);
- Центральный банк Российской Федерации (Банк России).

При этом право лицензирующей деятельности (то есть разрешительно/ запретительной в области разработки, распространения, использования и сопровождения средств защиты информации) имеют только два из них:

- ФСБ – в ситуациях, когда вопрос затрагивает интересы всего государства в целом, а так же в сферах деятельности, связанных с использованием криптографии;
- ФСТЭК – в ситуациях, когда вопрос затрагивает интересы отдельных регионов, ведомств, организаций и предприятий.

Службы, которые могут организовывать и осуществлять защиту информации на уровне конкретного предприятия:

- служба экономической безопасности;
- служба безопасности персонала («режимный отдел», он же «1-ый отдел»);
- кадровая служба («отдел кадров»);
- специальная служба информационной безопасности.

ТЕМА 2. Основные концепции защиты информации

Практическая защита информации реализуется на всем спектре средств и методов, применяемых в информационных системах. Этот спектр включает в себя:

- программно-аппаратные средства;
- алгоритмы получения, обработки, хранения, передачи и отображения информации;
- протоколы взаимодействия участников информационного обмена;
- специальные организационные меры;
- правовые и юридические аспекты.

Но в основе процесса защиты информации лежит довольно ограниченный набор базовых принципов и концепций.

2.1. Методы управления доступом к информации

Политика безопасности – это система правил, позволяющая обеспечить корректное и безопасное взаимодействие участников обмена какими-либо ресурсами.

Это понятие встречается не только в сфере информационных технологий, но и в жизни. Например, на основании той или иной политики безопасности может быть организован:

- документооборот на предприятии;
- доступ посетителей в учреждение;
- выдача ключей сотрудникам организации.

Формы представления политики безопасности могут быть разнообразными. Она может быть доведена до участников в устной форме, занесена в какой-нибудь документ в виде списка правил и требований, оформлена в виде таблицы и т.п.

Разграничение доступа – универсальная концепция информационной защиты, направленная на решение вопросов управления конфиденциальностью и доступностью к информации. Именно на этой концепции базируются политики безопасности в сфере информационных технологий.

С формальной точки зрения любая политика разграничения доступа описывается отношениями четырех сущностей:

1) множеством объектов доступа;

2) множеством субъектов доступа;

3) множеством методов доступа субъектов к объектам;

4) множеством правил, разрешающих или запрещающих доступ субъектов к объектам.

Объект доступа – пассивный носитель информации, доступ к которому может быть ограничен (примеры: файл, внешнее устройство, область памяти, узел сети и т.п.). У каждого объекта может быть *владелец* (пользователь, системный процесс и т.п.), имеющий право изменять свойства объекта.

Субъект доступа – активный инициатор доступа к информации, содержащейся в каком-либо объекте (примеры: программа, запущенная от имени того или иного пользователя, порт ввода-вывода и т.п.).

Метод обмена информацией – способ доступа субъекта к объекту (примеры: чтение; запись; запуск программы на выполнение, проход через вахту в учебное заведение или на промышленное предприятие).

Возможно самостоятельное взаимодействие сущностей, проверяющих возможность доступа друг к другу. Но, как правило, процессом доступа управляет *монитор обращений* – специализированный компонент системы обеспечения безопасности, через который проходят все запросы на доступ. Монитор либо удовлетворяет, либо отвергает эти запросы в соответствии с системой правил, по которой он работает, и сведениями о конкретных объектах и субъектах, хранящимися в служебной базе данных (см. рис. 1.1).

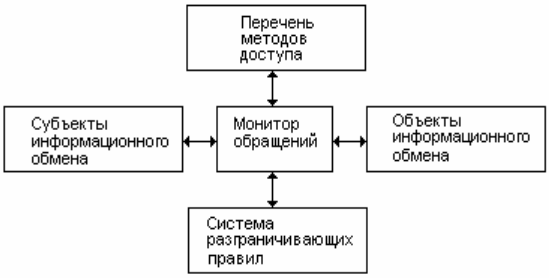


Рис. 2.1. Функциональная схема системы разграничения доступа

Примеры: антивирус является монитором обращений программ к вычислительным ресурсам, при этом он расширяет возможности стандартной политики безопасности операционной системы; сотрудник службы безопасности является монитором доступа, разрешающим или запрещающим проход студентов, преподавателей и посторонних лиц в корпуса университета.

2.1.1. Модели избирательного (дискреционного) доступа

Самая простая группа моделей дискреционного разграничения доступа (DAC – Discretionary Access Control) основана на матрицах типа табл. 2.1.

Таблица 2.1. – общий вид «матрицы доступа»

	S_1	S_2	…	S_N
O_1				
O_2				
…			$\{G_{ij}\}$ и $\{D_{ij}\}$	
O_K				

По одной координате расположены объекты доступа, по другой – субъекты, а в клетках размещаются характеристики доступа, определяющие наборы разрешенных G_{ij} и запрещенных D_{ij} методов взаимодействия субъекта S_i с объектом O_j.

Множество методов доступа всевозможных субъектов к конкретному объекту, образует *список контроля доступа* (ACL – Access Control List). Например, если субъектом является ключ от кабинета директора предприятия, то такой список мог бы иметь вид, как на табл. 2.2.

Таблица 2.2 – пример ACL

	Директор	**Главбух**	**Секретарша**	**…**
Ключ	Да	Нет	Да	…

Множество методов доступа, которыми обладает конкретный субъект, образует *мандат возможностей* (CL – capability list) – см. табл. 2.3.

Таблица 2.3. – Пример мандата возможностей

	Секретарша
Ключ от кабинета директора	Да
Ключ от склада	Нет
Ключ от цеха	Нет
…	…

Достоинство подобных моделей – простота. Недостаток – громоздкость. Конкретные реализации политик безопасности, основанные на использовании избирательного доступа, актуальны только для небольших систем с редко изменяемыми наборами субъектов, объектов и методов.

Частным случаем дискреционных политик являются политики *ролевого доступа*, которые лишены части недостатков. В этих моделях субъекты и объекты объединяются в группы (например, «суперпользователь», «рядовой пользователь», «читатель информации» и пр.), которым автоматически назначаются общие свойства. Это позволяет снизить размерность таблицы и упростить работу с ней. При необходимости наделить какой-либо субъект или объект уникальным набором свойств, он выделяется в отдельную группу.

2.1.2. Модели полномочного (мандатного) доступа

В моделях мандатного доступа (MAC – Mandatory Access Control) каждому объекту и субъекту присваивается некоторая «метка безопасности». Часто это просто число, которое является *уровнем допуска* (*секретности*, *конфиденциальности*, *защищенности*).

Пример «мандатной» политики: *модель Белла и ЛаПадулы*. В соответствии с ней субъект имеет право читать информацию только из объекта, чей «уровень допуска» ниже, и писать информацию только в тот объект, чей «уровень допуска» выше. Таким образом, блокируется перенос информации с более высоких уровней безопасности на более низкие.

Другой пример: *модель Биба*. Она является зеркальным отражением политики Белла и Ла-Падулы: участник информационного взаимодействия имеет право подвергаться модификации только со стороны другого участника, чья «защищенность»[1] выше, и – со своей стороны – модифицировать участника с менее высокой «защищенностью». Таким образом, блокируется модификация более привилегированных сущностей менее привилегированными.

На практике модель Белла и Ла-Падулы часто комбинируют с моделью Биба, вводя частные правила в тех случаях, если требования разных моделей противоречат друг другу.

2.1.3. Изоляционизм

Предельный случай концепции разграничения доступа – политика *изоляционизма*, т.е. полного запрета информационного взаимодействия между субъектами и объектами. Если такое взаимодействие необходимо, то оно разрешается в исключительных случаях и контролируется при помощи дискреционных либо мандатных политик.

2.1.4. Примеры практического применения политик безопасности

Рассмотрим некоторые конкретные примеры применения политик безопасности в сфере информационных технологий.

[1] В оригинале вместо термина «защищенность» используется consistency – «целостность»

2.1.4.1. Разграничение доступа к оперативной памяти

Современные процессоры фирмы Intel в *режиме с сегментной адресацией* реализуют мандатную политику Биба. В этом режиме оперативная память разбивается на непрерывные участки – *сегменты*, каждому из которых присваивается один из четырех уровней привилегий. Самым высоким считается 0-й уровень, самым низким – 3-й. Доступ программ, занимающих непривилегированные сегменты, к данным и программам, расположенным в более привилегированных сегментах, блокируется процессором – при такой попытке возбуждается ситуация исключения.

Благодаря аналогии со средневековой крепостью, эта система разграничения доступа получила наименование «колец защиты» (см. рис. 2.2). В современных операционных системах (Windows, Linux и т.п.) используются только два уровня привилегий: в сегментах 0-го кольца защиты размещается операционная система, в сегментах 3-го кольца – прикладные программы. Таким образом, прикладные программы (субъекты) не имеют доступа к сегментам (объектам), занимаемым операционной системой.

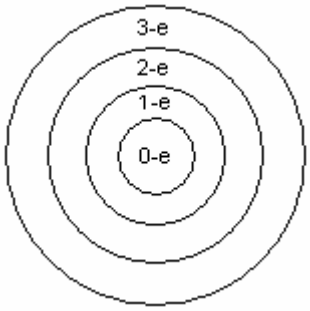


Рис 2.2. Кольца защиты

2.1.4.2. Изоляция виртуальных адресных пространств

Современные процессоры фирмы Intel в *режиме со страничной адресацией* реализуют политику изоляционизма. В этом режиме физическая память разбивается на *страницы* размером 4 Кб или 4 Мб и ячейкам в этих страницах назначаются виртуальные адреса. Это позволяет разделить память на несколько виртуальных адресных пространств, не имеющих пересечений (см. рис 2.3). В каждое виртуальное пространство загружается отдельная программа, и их доступ – ни для чтения, ни для записи – в иные пространства невозможен. Если он необходим, то осуществляется под контролем операционной системы. Например, для обмена данными между различными процессами в современных операционных системах существуют «разделяемая память» (shared memory), «каналы» (pipe), «почтовые ячейки» (mailslot) и т.п.

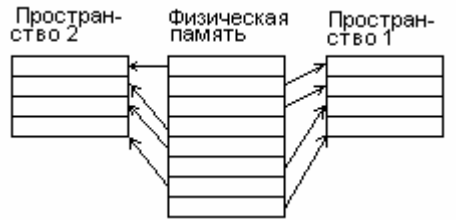


Рис. 2.3. Построение виртуальных пространств

Так же, в MS Windows возможно открыть чужое адресное пространство при помощи OpenProcess(), после чего читать или записывать в него при помощи ReadProcessMemory() и WriteProcess Memory(). Обычно режимы со страничной и сегментной адресацией используются совместно, что позволяет разделять на сегменты с разными уровнями привилегий не физическую память, а уже виртуальные адресные пространства.

2.1.4.3. Разграничение доступа в MS Windows

В состав Windows включена подсистема, реализующая «ролевую» политику безопасности[1].

Объектами являются:

- файлы;
- каталоги (директории, папки);
- устройства (диски, порты, клавиатура, мышь и т.п);
- каналы и почтовые ящики – средства передачи данных между процессами;
- ключи реестра;
- процессы и потоки;
- сервисы и диспетчер сервисов;
- рабочие столы и окна;
- фрагменты разделяемой памяти;
- символические связи;
- маркеры доступа;
- объекты синхронизации (семафоры, мьютексы, таймеры и т.п.).

Субъектами являются:

- пользователи;
- группы пользователей.

Фактически, роль субъектов выполняют процессы и потоки, запущенные от имени определенного пользователя или группы.

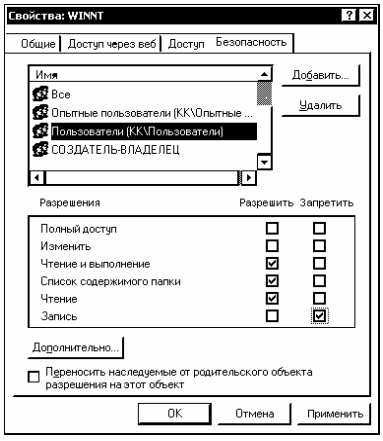


Рис. 2.4. Настройка правил доступа для файла

Каждому субъекту присваивается набор разрешений и запрещений, соответствующий его «роли» («Администратор», «Опытный пользователь», «Гость», «Оператор резервного сохранения» и т.п.). Кроме того, политика безопасности может быть настроена вручную пользователем, имеющим права на изменение, например, какому-нибудь другому пользователю можно запретить доступ к определенному каталогу или диску (см. рис. 2.4).

Правила согласования противоречий и неоднозначностей:

- если в списке несколько правил, относящихся к одному пользователю, то учитывается первое из них;
- запрет приоритетней разрешения;
- правила группы, в которую входит пользователь, приоритетней правил самого пользователя.

Политика безопасности работает только для носителей информации (дисков, флешек и т.п.) с файловой системой NTFS. Основным различительным признаком системных сущностей является уникальный *идентификатор*

[1] Начиная с Windows 7, наряду с ролевой присутствуют элементы мандатной политики.

безопасности (SID – Security Identifier), присваиваемый каждому субъекту и объекту. Идентификаторы сущностей уникальны, это длинные числа, которые для отображения могут быть представлены в текстовой форме, например:

- S-1-1-0 – «предопределенный» SID для группы, состоящей из всех пользователей системы;
- S-1-5-21-1078081533-1364589140-839522115-1003 – SID для «Администратора» одной из машин.

Дискреционная таблица хранится в виде системных списков ACL (от англ. Access Control List – Список контроля доступа), поставленных в соответствие каждому субъекту:

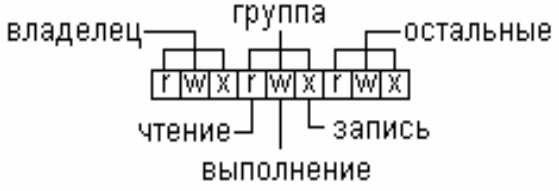
- DACL – список разрешений и запрещений;
- SACL – список флагов, описывающий необходимость или необязательность регистрации в «системном журнале» факта применения субъектом к объекту некоторого метода (см. далее раздел «Аудит в MS Windows»).

Максимальными правами обладают субъекты группы «Администратор».

2.1.4.4. Разграничение доступа в UNIX

В UNIX-подобных операционных системах каждому субъекту (пользователю и группе пользователей) присваивается некоторое уникальное число UID (от англ. User Identifier – Идентификатор пользователя) или GID (от англ. Group Identifier), а при создании любого объекта (файла, каталога или устройства) в системных областях носителя сохраняется информация об UID субъекта-владельца и его группы. Кроме того, объектам ставятся в соответствие 9 однобитовых «флагов» (см. рис. 2.5,а):

- «флаг» 1 имеет значения «r» или «-» и отвечает за возможность чтения содержимого объекта владельцем;
- «флаг» 2, принимающий значение «w» или «-», отвечает за видоизменение (запись) объекта со стороны владельца;
- «флаг» 3, со значениями «x» или «-», отвечает за возможность запуска объекта на выполнение со стороны владельца (для каталогов – за возможность перехода внутрь);
- «флаги» 4-6 несут нагрузку, эквивалентную «флагам» 1-3, но относятся к группе пользователей, к которой принадлежит владелец;
- «флаги» 7-9 также интерпретируются подобно «флагам» 1-3, но касаются всех остальных пользователей.



а) Назначение флагов

```
# ls -l ./
total 1496
drwxr-xr-x  2 root root      20 Aug  4  2009 bin
drwxrwxrwx  6 root root    4096 Jan 18  2011 boot
drwxr-xr-x 12 root root   14120 Dec 20 17:48 dev
drwxrwxrwx 46 root root     280 Dec 20 17:48 etc
drwxr-xr-x 66 root root      60 Jan 23  2011 usr
drwxr-xr-x 33 root root     160 Jan  5  2009 var
-rwx------  1 root root 1527232 Jan 23  2011 xds.lzm
```

б) Пример содержимого каталога

Рис. 2.5. Флаги доступа в UNIX

Комбинируя флаги, можно получать различные схемы доступа. Например: «rwx--xr--x» означает полный доступ к некоторому программному файлу (включая просмотр и изменение) только для «хозяина», остальные же могут лишь запускать его.

Кроме того, в некоторых версиях UNIX-подобных операционных систем могут присутствовать дополнительные флаги, расширяющие и уточняющие стандартную схему. Например, флаг «s» (он отображается в позиции флага «x») для общедоступного каталога «/tmp» запрещает пользователям удалять из этого каталога «чужие» файлы.

Просмотреть содержимое каталогов вместе с установленными флагами в UNIX можно при помощи команды «ls -l» (см. рис. 2.5,б)[1].

Изменить при помощи команды «chmod» флаги файла или каталога, а так же установить при помощи команды «chown» нового владельца может только его владелец.

Единственный «суперпользователь», который может менять любые флаги и признаки, вне зависимости от их принадлежности, это «root».

2.1.5 «Оранжевая книга»

Министерство обороны США (DOD – Departament of Defence) и Национальный комитет компьютерной безопасности (NCSC – National Committee of Computer Security) разработали и опубликовали систему стандартов в области компьютерной безопасности «Критерии оценки безопасности компьютерных систем» («Оранжевая книга»).

Эти стандарты основаны на понятиях формальных политик безопасности. Они представляют собой «шкалы», при помощи которых можно производить сравнение защищенности систем обработки информации.

В «Оранжевой книге» определены 7 классов защищенности систем.

Класс D. Минимальная защита. Присваивается системам, которые не прошли испытаний на более высокий уровень или тем системам, которые используют лишь отдельные функции (подсистемы) безопасности.

Класс C1. Защита, требующая, чтобы для каждого объекта и субъекта в системе задавался перечень допустимых типов доступа (пароль на вход в систему, пароль на запуск программы, пароль на печать и пр.).

Класс C2. Защита, основанная на управляемом доступе, под которым понимается политика, основанная на дискреционной модели. К требованиям класса C1 добавляются требования уникальной идентификации субъекта доступа (любой пользователь должен иметь уникальное имя), защиты по умолчанию («запрещено все, что не разрешено») и регистрации системных событий в системном журнале (эта операция называется *аудит*).

Класс B1. Меточная защита. Всем субъектам и объектам системы присваиваются «метки безопасности» (уровень доступа, секретности и т.п.).

[1] В этом списке вместе с флагами отображается тип объекта: d» – каталог, «b» – файл на блочном устройстве, «c» – файл на символьном устройстве, «s» – сокет, «p» – канал, «l» – ссылка.

Доступ субъекта к объекту разрешается на основании сравнения значения меток, причем сравнение может выполняться непосредственно субъектами и объектами (без участия монитора доступа).

Класс B2. Структурированная защита. Дополнительно предъявляется требование использования формальной «политики безопасности».

Класс B3. Доменная защита. Предъявляется требование наличия автономного монитора доступа, который должен:

- контролировать все взаимодействия субъектов с объектами;
- быть гарантированно защищен от несанкционированных изменений, порчи и подделки;
- быть простым для анализа и тестирования, а полнота системы тестов должна быть доказана.

Обязательным также является наличие процедур, обеспечивающих восстановление работоспособности системы.

Класс A1. Верифицированный проект, в котором для проверки правильности функционирования системы защиты применяются формальные математические методы.

Пример: операционная система Windows NT v4.0 (1996 г.) была официально сертифицирована по классу C2 с учетом выполнения дополнительных условий:

- загрузка системы и вход пользователя должны быть защищены паролем длиной не менее 6 символов;
- жесткие диски размечены под файловую систему NTFS (но не FAT!);
- запрещены анонимный и гостевой доступ;
- запрещено наличие в системе и использование любых средств низкоуровневого доступа к ресурсам компьютера (отладчиков, перехватчиков системных вызовов и пр.);
- запрещен выход из системы без входа в нее;
- тумблер выключения питания и кнопка Reset недоступны пользователю;
- осуществляется строгий контроль за использованием внешних носителей информации (дискет, компакт-дисков, Flash-накопителей);
- при переполнении журнала безопасности, в который подсистема аудита в соответствии со списком SACL заносит информацию обо всех событиях, работа с системой запрещена;
- доступ к объектам системных областей операционной системы (системному и корневому каталогам) разрешен только для администратора.

2.2. Методы управления целостностью данных

При управлении целостностью данных приходится решать задачи двух типов:

- обнаружение искажения данных;
- если данные искажены, то восстановление их в исходном виде.

Общий подход к решению заключается в том, что в данные вносится некоторая избыточность – дополнительные, так называемые «контрольные» порции данных. Величина $k_{изб} = n/(m+n)$, где *m* – объем «основных» данных, а n – объем «контрольных», носит наименование «*коэффициента избыточности*».

2.2.1. Многократное дублирование данных

Простейшим способом управления целостностью данных является их многократное дублирование.

Пусть, например, бит «1» кодируется сочетанием «11», а бит «0» – сочетанием «00». При этом сочетания «01» и «10» являются «запрещенными», и их появление однозначно свидетельствует об однократной ошибке. Разумеется, возможна и двукратная ошибка, которая преобразует «0» в «11» или «1» в «00», и эта ошибка двукратным дублированием обнаружена быть не может. Но вероятность многократных ошибок гораздо меньше вероятности однократной. Например, если вероятность однократной ошибки P, то вероятность двукратной равна P×P, трехкратной P×P×P и т.д.

Трехкратное дублирование бита позволяет не только обнаруживать однократные ошибки, но и исправлять их методом «голосования». Например, очевидно, что «запрещенное» сочетание «001» скорей всего получилось из «000», а не из «111». Правда, если это результат не однократной, а двукратной ошибки (менее вероятной!), то такое «исправление» будет некорректным.

Хорошим примером применения идеи дублирования являются RAID-технологии хранения данных (от англ. Redundant Array of Independent Disks – избыточный набор независимых дисков). Они предусматривают «параллельное» использование нескольких дисковых устройств. Работоспособность системы RAID-дисков в режиме «*зеркалирования*» (mirroring) сохраняется, пока работоспособен хотя бы один диск.

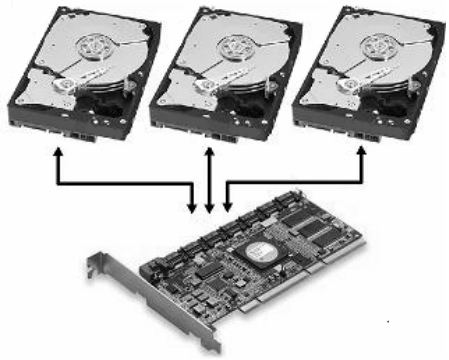


Рис. 2.6. Концепция RAID

Возможны как программные способы организации RAID-зеркал (например, встроенные средства современных версий Windows), так и аппаратные – специальные контроллеры (см. рис. 2.6).

2.2.2. Контрольные суммы и хеш-функции

Для контроля целостности данных часто используется вычисление по ним значения некоторой *хеш-функции*. При этом данные D могут быть произвольными, но результат вычисления хеш-функции H(D) всегда должен иметь фиксированную длину.

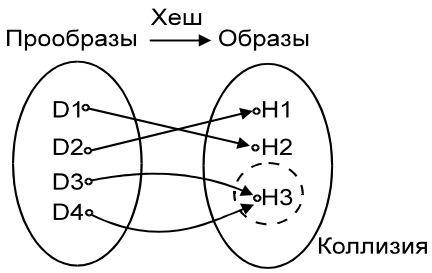


Рис. 2.7. Сущность хеш-функций

Значение хеш-функции вычисляется дважды: до выполнения потенциально «опасной» операции (например, передачи по каналу связи) и после нее. После эти значения сравниваются. Несовпадение значений однозначно свидетельствует об искажении данных. Но обратное неверно – совпадение не гарантирует отсутствия искажений, поскольку применение хеш-функций может приводить к *коллизиям* – то есть к ситуациям, когда несколько прообразов имеют один и тот же образ (см. рис. 2.7).

Итак, хэш-функции позволяют обнаруживать искажения лишь с определенной вероятностью.

Хеш-функция, используемая для обнаружения искажения данных, должна обладать хорошими «рассеивающими» свойствами. То есть «похожие» прообразы D должны отображаться на сильно различающиеся образы H(D)[1].

Рассмотрим несколько типичных примеров применения хеш-функций.

2.2.2.1. Бит четности

В технике связи (например, при передаче данных по интерфейсу RS-232) и в цифровой электронике (например, при записи/чтении данных в микросхемы памяти) используется контроль «*бита четности*». Исходные данные рассматриваются как блок (цепочка битов) определенной длины, а хеш представляет собой одиночный бит, который рассчитывается как результат сложения битов блока по модулю 2. Очевидно, если в исходной цепочке нечетное количество единичных битов, то значение хеш-функции равно 1; иначе 0.

Например, для данных 11100100 результат: 1⊕1⊕1⊕0⊕0⊕1⊕0⊕0=0.

Обычно бит, полученный в результате такого суммирования, передается вслед за исходной цепочкой битов. При этом он играет роль дополнения суммы битов до 0. На приемном конце все полученные биты (включая добавленный контрольный бит!) вновь суммируются по модулю 2. Очевидно, если результат отличен от 0, то делается вывод, что произошло искажение в одном или нескольких битах передаваемых данных.

Достоинство метода: простота аппаратной реализации и высокое быстродействие. Недостатки: 1) возможность обнаружения только нечетного количества искажений битов; 2) невозможность исправления обнаруженных ошибок.

2.2.2.2. Простые контрольные суммы

В информатике получили определенное распространение «простые» *контрольные суммы* (англ. checksum), основанные на арифметическом

[1] Это так называемы «лавинный» (или «каскадный») эффект – изменение одного бита исходных данных должно приводить к изменению не менее 50% битов хеш-функции

суммировании элементов числовых данных (байтов, слов, двойных слов и т.п.). Их главное достоинство – простота программирования и высокая скорость работы.

Например, содержимое ROM BIOS некоторых моделей персональных компьютеров разделено на блоки размером по N=4096 байтов. Каждые первые 4095 байтов каждого блока арифметически просуммированы[1], а на место 4096-го байта записано дополнение этой суммы до нуля. При включении компьютера суммируются все 4096 байтов каждого блока.

```
unsigned char csum=0;
for (i=0;i<N;i++) csum = csum + s[i];
```

Если результат отличен от 0, то делается вывод об искажении содержимого BIOS. Аналогичный метод используется так же для контроля целостности TCP-пакетов, только вместо 8-битовых байтов суммируются 16-битовые слова.

Вместо арифметического суммирования возможно так же применение операции «XOR».

```
unsigned char csum=0;
for (i=0;i<N;i++) csum = csum ^ s[i];
```

Основной недостаток метода – в невозможности обнаружить искажения типа «перестановка байтов местами».

Для избавления от этого недостатка можно воспользоваться «алгоритмом Бернштейна», который предусматривает на каждой итерации цикла умножение промежуточной суммы на какое-нибудь простое число (например, на 31 или 37):

```
unsigned char csum=0;
for (i=0;i<N;i++) csum = csum*31 + s[i];
```

Разумеется, все эти и подобные им «простые» контрольные суммы используются для обнаружения только случайных искажений – потому что искусственно сформировать набор данных под любую контрольную сумму и, таким образом, «подделать» ее, очень легко.

2.2.2.3. CRC – циклический избыточный код

Циклический избыточный код (англ. CRC – *cyclic redundancy code*) – важный, очень широко применяемый метод обнаружения искажений.

Он основан на представлении блока данных в виде непрерывного полинома с битовыми коэффициентами. Например, цепочка битов 10011 может быть представлена как полином $1\times x^4 + 0\times x^3 + 0\times x^2 + 1\times x^1 + 1\times x^0 = x^4 + x + 1$.

В качестве контрольного кода используется остаток от деления исходного полинома на более короткий «*порождающий*» *полином*, обычно степени N=4, 8, 12, 16 или 32. На эту роль годятся только «неприводимые» (т.е. не раскладываемые на сомножители) двоичные полиномы. Техника деления: 1) в конец блока данных добавляются N-1 нулевых битов; 2) вместо арифметического деления используется операция «сложение по модулю 2»;

[1] Переполнения игнорируются. То есть, фактически, вычисляется арифметическая сумма по модулю 256.

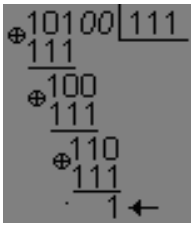


Рис. 2.8 Деление полиномов с остатком

3) фрагмент исходного полинома длиной N битов, с которым производится сложение «порождающего» полинома, каждый раз сдвигается влево до тех пор, пока его (фрагмента) старший бит не окажется равным 1.

Пример: исходные данные 101, «порождающий» полином 111, дополнительные биты 00, результат: 1 (см. рис. 2.8).

Циклическое избыточное кодирование используется, например, для контроля целостности данных в секторах гибких и жестких магнитных дисков, в архивах типа «ZIP» и «RAR» (см. рис. 2.9), в канальных протоколах технологии Ethernet, в файлах динамически загружаемых библиотек, некоторыми антивирусами при поиске вредоносных программ и в очень многих прочих приложениях и технологиях.

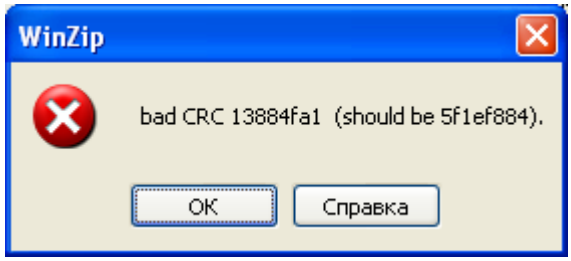


Рис. 2.9. Сообщение архиватора WinZip

Некоторые порождающие полиномы, как и алгоритмы их расчета, зафиксированы в международных стандартах – например, полином 16-ой степени «CRC-16» со значением 1A001h и полином 32-ой степени «CRC-32» со значением 1EDB88320h.

Стандартные алгоритмы расчета CRC нередко содержат «лишние» операции, например, побитовое инвертирование результата. Хеширующих свойств это не увеличивает, но облегчает аппаратную реализацию алгоритмов.

Следует так же отметить, что описанная выше техника расчета CRC удобна для аппаратной реализации с использованием сдвиговых регистров, но нерациональна при программировании на ЭВМ. Существуют «быстрые» модификации алгоритма расчета CRC, которые основаны на использовании таблиц с заранее рассчитанными промежуточными результатами вычислений.

Так же в стандартной библиотеке NTDLL.DLL от MS Windows имеется функция RtlComputeCrc32(), которая вычисляет «CRC-32» для указанного массива данных указанной длины. Ее прототип:

```
DWORD RtlComputeCrc32 {
  DWORD dwInitial,      // Начальное значение суммы, обычно 0
  BYTE * pData,         // Массив данных
  INT iLen              // Длина массива в байтах
}
```

Хорошими «конкурентами» для CRC являются хеш-функции: 16-битная контрольная сумма Флетчера и 32-битная сумма Марка Адлера. Они вычисляются несколько быстрей, чем CRC, но на коротких порциях данных производят несколько больше коллизий.

2.2.2.4. Криптографические хеш-функции

Хеш-функция, используемая для защиты от преднамеренных искажений данных, должна обладать следующими дополнительными свойствами:

- для любого «образа» хеш-функции H(D) невозможно сформировать или подобрать «прообраз» D;
- для любого «прообраза» D_1 невозможно сформировать или подобрать «прообраз» D_2 такой, что $D_1 \neq D_2$, но $H(D_1)=H(D_2)$;
- более того, невозможно сформировать или подобрать любые $D_1 \neq D_2$ такие, что $H(D_1)=H(D_2)$.

Разумеется, «обращение» хеш-функции, то есть подбор прообразов под образы, теоретически всегда возможно «*методом грубой силы*» (англ. brute force), а именно – путем перебора огромного количества вариантов. Поэтому под словом «невозможно» здесь понимается недоступная при современном уровне науки и техники вычислительная сложность.

Таблица 2.4 – Некоторые «криптографические» хеш-функции

Наименование	**Длина хеша, бит**	**Наименование**	**Длина хеша, бит**
MD2	128	MD4	128
MD5	128	MD6	До 512
SHA–1	160	SHA–2	224, 256, 384, 512
ГОСТ 34.11– 94	256	ГОСТ 34.11–2012	256, 512
SHA–3/Keccak	224, 256, 384, 512	RIPEMD	128, 160, 256, 320

Одним из очевидных условий практической «необратимости» хеш-функций является большая длина «образа» – не 16 или 32, а несколько сотен бит. В технологиях защиты данных от злонамеренных искажений с начала 1990-х годов широко используются сложные, трудно обратимые хеш-функции (см. табл. 2.4).

Сложность искусственного «обращения» этих функций настолько высока, а вероятность коллизий настолько мала, что любые два набора данных, имеющих один и тот же хеш, на практике можно считать идентичными. Поэтому подобные хеш-функции часто называют «дайджестами» набора данных (англ. message didgest) или «отпечатками пальцев» (англ. fingerprint).

Наиболее популярны хеш-функции SHA-1 и MD5, которые долгое время считались практически необратимыми. Однако, в 2003-2005 гг. была проведена их «*компрометация*», то есть демонстрация «ненадежности», а именно: показана возможность обратимости для отдельных частных случаев. Уже ведутся работы по разработке новых хеш-функций для замены SHA-1 и MD5 в протоколах шифрования данных, тем не менее, они все еще очень широко применяются.

Стандартный программный интерфейс CryptoAPI в MS Windows предоставляет любым прикладным и системным программам доступ к процедурам расчета наиболее популярных криптографических хеш-функций:

MD2, MD4, MD5 и SHA-1, SHA-2 и пр. Так же стандартная[1] консольная утилита «CERTUTIL», запущенная с ключом «-hashfile», позволяет рассчитывать хеши для указанных в командной строке файлов.

Иногда требуется *хеширование с ключом*, результат которого не может быть воспроизведен без знания некоторого секретного ключа. Для этой цели обычно применяется следующая процедура:

- хешируемые данные шифруются с применением ключа (см. ниже раздел «Методы шифрования данных»);
- результат шифрования разбивается на блоки одинаковой длины;
- в качестве значения хеш-функции берется результат побитовой операции «XOR» над всеми блоками.

Подобный принцип хеширования с ключом предусмотрен, например, в отечественном ГОСТ 34.11-94.

Результат «хеширования с ключом» может быть использован для защиты открыто передаваемых данных от преднамеренных искажений – то есть для «*имитозащиты*». Такой хеш, интегрированный в передаваемые данные, называется «*имитовставкой*». Очевидно, злоумышленник, исказив сами данные, не сможет скорректировать значение хеша, потому что не знает секретного ключа. В итоге, несоответствие «имитовставки» данным послужит признаком состоявшегося искажения.

Так же «хеширование с ключом» может быть использовано в протоколах аутентификации (проверки подлинности).

2.2.2.5. Кодирование с исправлением ошибок

Описанные выше методы и алгоритмы (за исключением методов, использующих дублирование) способны только обнаруживать искажения данных, но не способны их исправлять. Рассмотрим некоторые методы контроля целостности, лишенные этого недостатка.

1. «*Продольно-поперечный контроль четности*» – наиболее простой и очевидный метод обнаружения и исправления одиночных ошибок. Информационные биты условно располагаются в виде матрицы M×N элементов, для каждой строки и каждого столбца которой рассчитывается свой «бит четности» – см. рис. 2.10. Совокупность битов четности используется в качестве «*синдрома*» – то есть контрольного числа, которое может быть использовано не только для обнаружения, но и для исправления однократных искажений информационных битов. Например, если в результате выполнения «опасной» операции оказываются измененными контрольные биты c_{2*} и c_{*3}, то можно сделать вывод об искажении информационного бита b_{23} и исправить значение этого бита на противоположное. В случае, если в контрольном числе оказывается измененным только один бит, это означает, что именно он подвергся искажению. Все остальные случаи соответствуют многократным искажениям.

[1] Начиная с Windows 7.

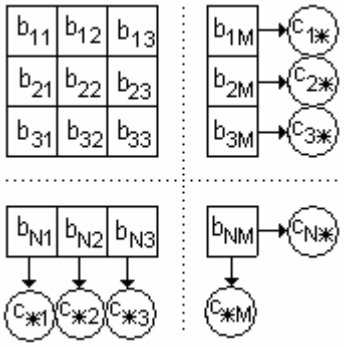


Рис. 2.10. Иллюстрация продольно-поперечного контроля четности

Продольно-поперечный контроль четности прост в реализации, но не оптимален с точки зрения требуемой длины синдрома.

2. *Коды Хэмминга* – это группа методов для обнаружения и исправления одиночных ошибок, которая обеспечивает минимально возможную длину синдрома.

Идея Хэмминга заключается в разбиении множества из K битов на $[log_2K+1]$ пересекающихся подмножеств так, чтобы каждый бит встречался, по крайней мере, в двух подмножествах[1].

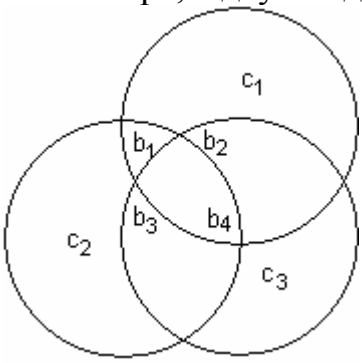


Рис. 2.11. Разбиение четырех битов на три группы

Например, для множества $\{b_1, b_2, b_3, b_4\}$ это могут быть три подмножества: $\{b_2, b_3, b_4\}$, $\{b_1, b_3, b_4\}$ и $\{b_1, b_2, b_4\}$ – см. рис. 2.11[2]. Биты четности, рассчитанные для каждого такого подмножества, образуют синдром. Для рассматриваемого примера это будет 3-битовое число $c_1c_2c_3$, в котором $c_3=b_2\oplus b_3\oplus b_4$, $c_2=b_1\oplus b_3\oplus b_4$ и $c_1=b_1\oplus b_2\oplus b_4$. Если произойдет искажение одного бита данных, то изменятся сразу два бита в контрольном числе, по которым легко вычислить, какой именно бит был искажен. Если же окажется, что изменился только один бит синдрома, значит, именно он и подвергся искажению – см. табл. 2.5.

Таблица 2.5 – Соответствие битов синдрома битам данных

Номер искаженного бита	**Изменятся биты синдрома**		
	c_3	**c_2**	**c_1**
b_1	Нет	Да	Да
b_2	Да	Нет	Да
b_3	Да	Да	Нет
b_4	Да	Да	Да
c_1	Нет	Нет	Да
c_2	Нет	Да	Нет
c_3	Да	Нет	Нет

[1] Здесь квадратные скобки […] означают округление до целого в меньшую сторону.
[2] Биты данных и синдрома должны нумероваться, начиная не с 0, а с 1. Это важно!

Если биты данных и биты синдрома расположены в «естественном» порядке $b_1b_2b_3b_4c_1c_2c_3$, то для исправления ошибок можно просто пользоваться таблицей 2.5.

Но целесообразно усовершенствовать код Хэмминга, переупорядочив строки таблицы таким образом, чтобы первая строка содержала {Нет, Нет, Да}, то есть число «1» в двоичной записи; вторая строка {Нет, Да, Нет} – число «2»; третья строка {Нет, Да, Да} – число «3»; и так далее до «7». В этом случае биты данных и синдрома окажутся расположены в порядке $c_1c_2b_1c_3b_2b_3b_4$. Теперь, если произошла ошибка, достаточно сложить (по модулю 2) два синдрома – исходный и рассчитанный по искаженным данным. Результат даст двоичное число – номер искаженного бита (проверьте!).

3. *Обнаружение и исправление* не только однократных, но и множественных ошибок тоже возможно.

Все рассмотренные ранее способы кодирования образуют два вида комбинаций битов – «разрешенные» и «запрещенные». В примере с двукратным дублированием битов «разрешенными» из 4-х возможных являлись только сочетания «00» и «11», остальные соответствуют ошибкам. А при кодировании 4-битовых символов методом Хэмминга можно образовать 128 всевозможных 7-битовых сочетаний «символ+синдром», из которых «разрешенными» являются только 16.

Расстоянием Хэмминга d между двумя сочетаниями битов называется количество несовпадений в них. Например, между «00» и «11» оно равно 2, а между «00» и «01» только 1. Расстояние Хэмминга между «0101» и «1010» равно 4, а между «0101» и «1111» только 2.

Минимальное расстояние Хэмминга d_{min} между двумя «разрешенными» сочетаниями какого-либо метода кодирования называется «*кодовым расстоянием*» для данного кода. Например, метод двукратного дублирования битов характеризуется расстоянием $d_{min}=2$, а метод трехкратного дублирования и коды Хэмминга – расстоянием $d_{min}=3$.

Существует общее правило. Чтобы метод кодирования позволял:

- обнаруживать n-кратные ошибки, необходимо $d_{min} \geq n+1$;
- обнаруживать и исправлять n-кратные ошибки, необходимо $d_{min} \geq n+2$.

Примерами методов кодирования с большим d_{min} являются БЧХ-коды[1] и их частный случай – *код Рида-Соломона*. Эти методы рассматривают в качестве элементов данных не отдельные биты, а группы битов – например, байты. Они находят применение в технике чтения-записи компакт-дисков, во время передачи больших массивов данных с космических аппаратов и т.п.

2.3. Методы сокрытия информации. Стеганография

Стеганография (от греч. τεγανός — скрытый, γράφω — писать) наука о сокрытии информации[2]. Методы и средства, основанные на выводах этой

[1] По первым буквам фамилий авторов: Боуз, Чоудхури и Хоквингем.

[2] В отличие от *криптографии*, стеганография скрывает сам факт наличия секретной информации.

науки, направлены, прежде всего, на предотвращение несанкционированного доступа к этой информации.

Общая схема использования стеганографических методов изображена на рис. 2.12.

Рис. 2.12. Модель стеганографической системы

Обычно предполагается, что скрываемая информация помещается внутрь некоторого несекретного носителя, который обеспечивает ее хранение, передачу и тому подобное. Этот носитель называется *стегоконтейнером* (или просто *контейнером*). Например, контейнером для информации, содержащейся в обычном письме, является бумажный конверт.

Устройство, метод или алгоритм, осуществляющий встраивание информации в контейнер, называется «*кодером*», обратная операция выполняется «*декодером*». Возможно (но необязательно!), перед помещением информации в контейнер она подвергается некоторому предварительному преобразованию. Устройство, метод или алгоритм, осуществляющий это преобразование, называется «*прекодером*»; обратное преобразование выполняется «*посткодером*».

Например, в случае письма роль «прекодера» заключается в складывании бумажного листа до нужного размера, чтобы он уместился в конверт. Если речь идет о цифровой информации, то «прекодер» и «посткодер» могут выполнять ее шифрование и расшифрование с некоторым ключом, сжатие и восстановление, разбиение на части и объединение в единое целое, и тому подобное.

Стеганография решает три больших группы задач:

- скрытую передачу или хранение секретной информации;
- доказательство (или, наоборот, опровержение) подлинности (или, наоборот, поддельности) контейнера, содержащего секретную метку;
- скрытое размещение в контейнере идентифицирующей или справочной информации (например, наименования или инвентарного номера).

Важным условием эффективности того или иного стеганографического метода является противодействие следующим атакаМ:

- детектирование (обнаружение) наличия скрытой информации внутри контейнера, без ее извлечения;
- несанкционированное извлечение информации из контейнера;
- уничтожение скрытой информации без уничтожения контейнера;
- целенаправленное искажение информации внутри контейнера.

Основным фактором, обеспечивающим противодействие атакам, является секретность метода внедрения или извлечения из контейнера стеганографической информации. В качестве вспомогательных факторов могут служить: 1) сложность и, следовательно, невоспроизводимость метода внедрения или извлечения; 2) секретность метода пред- или посткодирования. Однако сами по себе дополнительные факторы не являются предметом рассмотрения стеганографии.

Например, водяные знаки на денежных купюрах не являются секретом (секретом является только технология их создания), следовательно, они не имеют отношения к стеганографии. Однако секретные элементы рисунка на купюрах, видимые только в темноте под воздействием ультрафиолетового излучения, предназначены только для специалистов и, несомненно, являются примером применения стеганографических технологий.

С точки зрения устойчивости стеганографической системы к искажению контейнера, содержащего скрытую информацию, применима следующая классификация:

- «*робастные*» – устойчивые к любым искажениям;
- «*полухрупкие*» – устойчивые к некоторым видам искажений;
- «*хрупкие*» – не устойчивые к искажениям.

Различные методы сокрытия информации использовались человечеством на протяжении многих тысячелетий, используются они и сейчас. Условно их можно разделить на несколько групп.

2.3.1. «Классическая» стеганография

Наиболее старая, простая и распространенная группа методов, предполагает, что скрываемая информация представляет собой какой-нибудь текст, изображение или звук, а контейнером является физический объект. Немало примеров «*классической*» стеганографии можно найти в научно-популярной, художественной и исторической литературе.

Рассмотренный выше пример с бумажным письмом является иллюстрацией типичного метода «*физической*» стеганографии. Иные примеры:

> ...Ведь Гистией желал склонить Аристагора к восстанию, но не мог найти другого безопасного способа, так как все дороги охранялись. Тогда Гистией велел обрить голову своему верному слуге, наколол на голове татуировкой знаки, а затем, подождав, пока волосы отрастут, отослал его в Милет. Гистией дал слуге только одно поручение: прибыв в Милет, просить Аристагора обрить ему волосы и осмотреть голову...
>
> Геродот. История. Книга V. Терпсихора.

> ...Узнав о походе, Демарат захотел послать весть лакедемонянам. Так как иным способом он не мог известить лакедемонян, то придумал вот какую хитрость. Взяв двухстороннюю дощечку, он соскоблил с нее воск. Затем на дереве дощечки написал замысел царя и снова залил воском

написанное, чтобы чистая дощечка не могла возбудить подозрения у дорожных стражей. Когда же дощечку доставили в Лакедемон, то лакедемоняне не могли понять. Наконец, как мне рассказывали, дочь Клеомена, супруга Леонида, Горго разгадала смысл. Она сказала, что нужно соскоблить воск и тогда на дереве обнаружатся буквы. Лакедемоняне так и сделали, нашли надпись прочитав ее, отослали остальным эллинам...

Геродот. История. Книга VII. Полигимния.

Впечатляющие примеры дает «*химическая*» стеганография, которая очень широко использовалась начиная со Средних веков вплоть до середины XX века, да и в настоящее время не потеряла актуальности. Свидетельством тому являются, например, современные способы изобличения взяточников с помощью светящейся в темноте краски, нанесенной на денежные знаки и ценные подарки. Иные примеры:

...Вы, конечно, слыхали, что с давних времен существуют химические составы, при посредстве которых можно тайно писать и на бумаге, и на пергаменте. Запись становится видимой под влиянием тепла. Растворите цафру в «царской водке» и разведите потом в четырехкратном объеме воды, чернила будут зелеными. Растворите кобальтовый королек в нашатырном спирте – они будут красными. Ваша запись вскоре исчезнет, но появится вновь, если вы прогреете бумагу или пергамент вторично...

Э.По. Золотой жук.

...Писать в тюрьме было не таким уж простым делом. Родные имели право присылать арестованному книги. И вот тогда Ленин стал писать на этих книгах. А надо было так писать, чтобы никто в тюрьме не догадался, что тут в книге что-нибудь написано. Потому что в тюрьме проверяли все книги, перед тем как отдать родственникам. И если видели, что в книге хоть одно революционное слово написано, эту книгу сжигали. А революционеры знали, что можно писать молоком. Если на бумаге написать молоком, то решительно ничего не видно. А для того, чтобы прочесть написанное, надо погреть эту бумагу на лампе или на свечке, и тогда молоко начинает темнеть, на бумаге выступают коричневые буквы, и все можно прочесть, что написано. Вот Ленин так и писал: на полях книги и между строчками. А родные его об этом знали. И когда получали обратно книгу, грели каждый листик на лампе, читали и перечитывали. Так работал Ленин в тюрьме...

М. Зощенко. Рассказы о Ленине.

Так же имеет смысл упомянуть методы «*лингвистической*» стеганографии, предполагающие, что и скрываемыми данными, и контейнером является набор знаков – текст. Например, красивым, но очень сложным методом сокрытия одного текста внутри другого являются *акростихи*, где секретная информация заключена в первых буквах каждой строки:

Певца, гонимого судьбою,
Отвергнуть не захочешь ты
За то, что он любил с тобою
Делить с печалью и тоскою
Разбитой юности мечты.
А если б, как в былые годы,
В аккордах окрылять я мог
Любовь и радости свободы,—
Я снова, позабыв невзгоды,
Ютился у твоих бы ног.
К. Хетагуров

Возможны и более простые вариации похожей идеи:

...Он сел напротив меня, подвинул лампу к краю стола и протянул мне короткую записку - как видите, написанную второпях на клочке серой бумаги. «С дичью **дело**, мы полагаем, **закончено**. Глава предприятия **Хадсон**, по сведениям, **рассказал** о мухобойках **все**. Фазаньих курочек **берегитесь**»... Я обнаружил, что если взять каждое третье слово, то вместе они составят то самое письмо, которое довело старика Тревора до такого отчаяния. Письмо оказалось коротким, выразительным, и теперь, когда я прочел его моему другу, в нем явственно прозвучала угроза: **«Дело закончено. Хадсон рассказал все. Берегитесь»...**
А. Конан-Дойл. Глория Скотт

Анализ приведенных примеров позволяет сделать однозначный вывод: методы «классической» стеганографии сложны в реализации и неудобны в применении.

2.3.2. «Цифровая» стеганография

В современных условиях информация – будь то текст, изображение, звук или произвольные данные – хранится, обрабатывается и передается в цифровой форме, то есть в виде набора чисел, закодированных двоичным кодом. Соответственно, методы сокрытия информации, использующие это обстоятельство и реализуемые с помощью средств вычислительной техники, можно назвать средствами «*цифровой*» *стеганографии*.

Наиболее простые методы используют примерно такие же идеи, что и в «классической» стеганографии. Например, можно скрывать произвольные данные в обычном тексте, воспользовавшись следующим правилоМ: нулевые биты скрываемых данных кодируются двумя пробелами между словами, а единичные – одним пробелом. На глаз наличие «лишних» пробелов практически незаметно, поэтому извлечение скрытых данных лучше поручить несложной программе. На рис. 2.13 для облегчения визуального восприятия пробелы помечены «корытцами». Прочитайте скрытый текст самостоятельно!

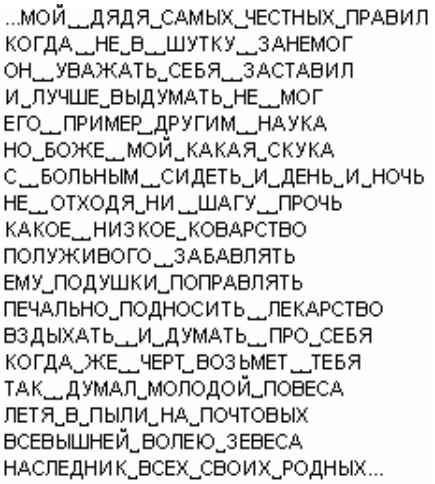


Рис 2.13. Текст в тексте

Разумеется, столь примитивный метод нельзя рекомендовать для практического использования. Но зато он хорошо иллюстрирует очень важный принцип: объем данных, внедряемых в цифровой контейнер, всегда меньше объема самого контейнера.

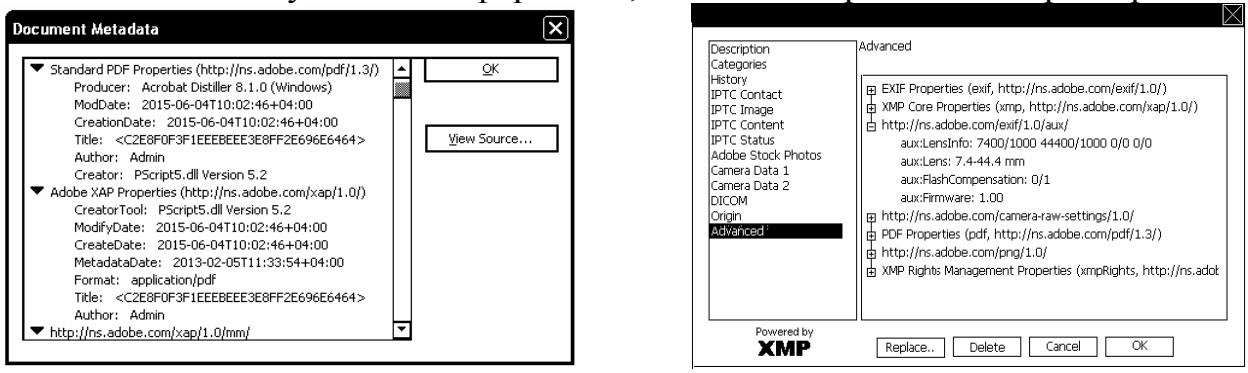
Для оценки качества конкретной стеганографической системы можно ввести характеризующий ее коэффициент $K_{ст} = N_д / N_к$, где $N_д$ - максимально возможный размер скрываемых данных; N_k – объем контейнера.

Этот коэффициент представляет собой «*пропускную способность стеганографического канала связи*».

2.3.2.1. Внедрение данных в «пустые» области

Очень большая, плохо изученная, но часто используемая группа методов цифровой стеганографии предусматривает сокрытие данных в неиспользуемых или редко используемых областях цифровых контейнеров.

Например, многие файловые форматы для графической, звуковой, текстовой и числовой информации предусматривают хранение вместе с данными так же «метаданных». *Метаданные* (или «данные о данных») – необязательная служебная информация, имеющая справочный характер.



а) Метаданные в «PDF» б) Метаданные в «TIFF»

Рис. 2.14. Примеры метаданных

В частности, в файлах формата «PDF», предназначенных для хранения и распространения электронных документов, возможно сохранение так же сведений о программах, с помощью которых изготовлен макет документа и произведено конвертирование в «PDF»; дате и времени совершения различных преобразовании документа; пользователе, выполнявшем эти преобразования; ключевых словах, характеризующих документ и пр. А в графических файлах форматов «JPEG» и «TIFF» в соответствии со стандартом «EXIF» можно сохранять уменьшенное изображение, а вместе с ним сведения о месте и времени съемки; используемой цифровой камере; версии прошивки; условиях съемки; фотографической экспозиции и пр. Поскольку все эти сведения используются исключительно редко, замена их некими секретными данными

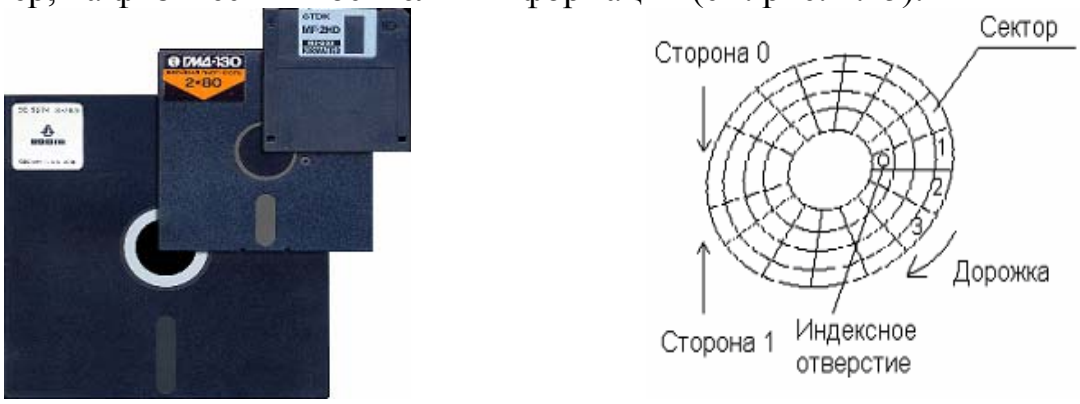
может остаться незамеченной. Области под метаданные присутствуют так же в аудиофайлах «MP3», видеофайлах «MP4» и многих других.

Другим важным фактором, облегчающим сокрытие информации, является избыточность форматов хранения данных – то есть наличие в них неиспользуемых областей.

Например, пиксели изображения в формате «BMP» описываются тройками 8-битовых целых чисел, из которых первое соответствует яркости красной, второе – зеленой и третье – синей составляющей цвета[1]. Но количество байтов, описывающих каждую строку изображения, должно быть кратно 4-м. Соответственно, в конец каждой строки могут появиться «пустые» байты, которые могут быть использованы под секретные данные.

Похоже обстоит дело и с файлами формата «EXE», которые предназначены для хранения кода и данных программ. Файлы разбиты на сегменты[2], размер которых должен быть кратен 1024 или 4096 байтам. Соответственно, если сегмент не заполнен, в конце его могут располагаться обширные пустые области. Этим обстоятельством пользовался, например, вирус Win9X.CIH (он же «Чернобыльский»), который записывался внутрь файла программы, не изменяя его длину.

«Пустые» области присутствуют не только внутри файлов, но и, например, на физических носителях информации (см. рис. 2.15).



а) Разные типы дискет б) Формат разметки

Рис. 2.15. Дискеты

На стандартно отформатированных дискетах 3.52”, содержащих 1.44Мб информации, присутствуют 2 стороны (и столько же головок чтения-записи), 80 дорожек (или «цилиндров») и 18 секторов. Стороны нумеруются: 0 и 1; дорожки: 0,1,2… 79; сектора: 1, 2…18. Каждый сектор адресуется тремя числами: {C,H,S} = {Дорожка, Сторона, Сектор}. Общее количество секторов на дискете $N_c=2\times18\times80=2880$. Начало каждой дорожки (сектор с номером 1) расположено напротив индексного отверстия. Объем данных в одном секторе стандартного размера – 512 байтов.

Жесткие диски имеют похожую организацию, только имеют не две, а несколько сторон и головок чтения-записи; несколько миллионов дорожек и до 256 секторов на дорожке. Объем данных может достигать нескольких Тб.

[1] Существуют и другие разновидности формата «BMP».

[2] Например, «.code» – для команд, «.data» – для данных, «.rsrc» – для ресурсов и т.д.

Разметка на дискетах и жестких дисках формируется стандартной программой FORMAT или аналогичными. Напрямую программируя контроллер, можно создавать форматы дискеты, например, с увеличенным до 82 количеством дорожек[1]. В эти области может быть записана секретная информация и стандартными средствами операционной системы получить к ней доступ невозможно. Так же существует возможность записывать группы секретных битов в промежутки между секторами. Эти приемы в 1990-х годах широко использовались в технологиях аутентификации при помощи «ключевых дискет» (например, в «Нота» или «Cop's CopyLock»).

Есть «пустые» области и на жестких дисках. Например, по умолчанию «пуста» вся первая дорожка диска, кроме самого первого сектора на ней[2]. В эту область часто записываются загрузочные вирусы и «буткиты».

Анализируя стеганографические методы, использующие «пробелы» между порциями данных, можно прийти к следующему выводу: все эти методы просты в применении, но легко обнаружимы.

2.3.2.2. Методы внедрения в наименее значимые биты

Существует большая группа методов, рассчитанная на сокрытие произвольной информации в массиве данных, полученном в результате оцифровывания физических характеристик реальных объектов или процессов. Результат подобного оцифровывания представляет собой набор целых чисел.

Например, звук, записанный в формате «RAW», есть набор числовых (по умолчанию 16-битовых) значений амплитуды колебаний.

Другой пример: как ранее отмечалось, пиксели изображения в формате «BMP» описываются тройками 8-битовых целых чисел, из которых первое соответствует яркости красной, второе – зеленой и третье – синей компоненты. Идея встраивания основана на том, что некоторые биты оцифрованного изображения или звука в контейнере «менее важны» для человеческого зрения и слуха, и поэтому могут быть заменены битами скрываемых данных. Группа методов, использующих эту идею, получила наименование: «LSB-методы» (от англ. Least Significant Bits – Наименее значимые биты).

Например, в изображении формата «BMP» могут быть заменены младшие биты каждой цветовой компоненты (см. рис. 2.16, а). С целью увеличения $K_{ст}$ можно заменять не один младший бит, а два, три и так далее, но при этом будут увеличиваться искажения изображения-контейнера (см. рис. 2.17). Для уменьшения искажений можно заменять младшие биты не в каждой компоненте цветности, а только в «синей», так как глаз наименее чувствителен именно к этому цвету. Так же возможна замена не «младших», а «случайных» битов (см. рис. 2.17, б). В этом случае «координаты» измененных битов могут

[1] Дополнительные дорожки называются «инженерными цилиндрами» дискеты.

[2] В этом 512-байтовом секторе располагается «главная загрузочная запись» (англ. MBR – Master Boot Record), содержащая таблицу описания «геометрии» диска и крохотную программу-загрузчик.

быть получены при помощи генератора псевдослучайных чисел[1]. Можно обойтись и без «координат», если разбить битовый массив контейнера на фрагменты (им могут соответствовать, например, прямоугольные участки изображения) и в каждом фрагменте изменить всего один произвольный бит, но таким образом, чтобы сумма по модулю 2 всех битов фрагмента приняла требуемое значение «0» или «1».

а) Встраивание в «младшие» биты б) Встраивание в «случайные» биты

Рис. 2.16. Разновидности «метода LSB»

а) Пустой контейнер б) Изменен 1 бит в) Изменены 4 бита г) Изменены 6 битов

Рис. 2.17. Искажения изображения в результате внедрения в младшие биты

Общими недостатками всех методов этой группы являются:

- избыточно большой объем контейнера;
- чувствительность к любым преобразованиям контейнера, например, к сжатию с потерями изображения (форматы «JPEG» или «JPEG2000») или звука (формат «MP3»).

2.3.2.3. Робастные и «полухрупкие» методы

Основное преимущество «полухрупких» и, тем более, робастных стенографических методов заключается в их устойчивости к искажениям контейнера.

Математические основы этих методов довольно сложны. Пример: метод Куттера встраивания секретного бита в пиксель изображения.

1. Для пикселя с координатами (x,y), в который планируется встраивание, вычисляется суммарная яркость по формуле $C_{x,y} = 0.3R(x,y) + 0.59G(x,y) + 0.11B(x,y)$, где R(.), G(.) и B(.) – яркости красной, зеленой и синей составляющей.

2. Яркость синей составляющей видоизменяется по правилу:

$B'(x,y) = B(x,y) + qC_{x,y}$, если встраиваемый бит равен 1;

[1] Параметры генератора в этом случае будут играть роль «ключа», необходимого для извлечения сообщения из контейнера. Подробней о генераторах псевдослучайных чисел см. в разделе «Потоковые шифры».

$B'(x,y) = B(x,y) - qC_{x,y}$, иначе.

Здесь *q* – небольшая константа, от значения которой зависит устойчивость системы к преобразованиям контейнера. Чем больше эта константа, тем выше устойчивость, но тем заметнее искажения изображения.

Извлечение значения бита выполняется на основе попытки предсказания «правильной» яркости синей составляющей пикселя.

1. Выполняется предсказание яркости синей составляющей пикселя на основании яркостей соседних пикселей, образующих «крест» (см. рис. 2.18):

$$B'(x,y) = \frac{1}{4n}\sum_{i=1}^{n}(B_{x,y+i} + B_{x,y-i} + B_{x+i,y} + B_{x-i,y}),$$

где n=1..3.

2. Если предсказанная яркость выше реальной, значит, был встроен бит «0», если значительно меньше – то бит «1».

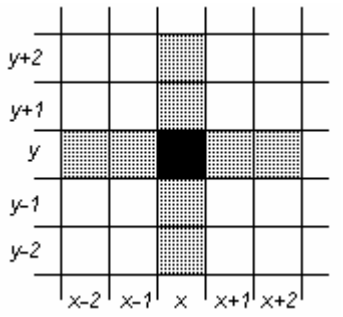


Рис. 2.18. «Соседи» пикселя

Изображение-контейнер, в которое по методу Куттера встроены секретные данные, может быть в дальнейшем подвергнуто сжатию с потерями информации, например, по алгоритму «JPEG» или «JPEG2000», однако это не приведет к уничтожению встроенных данных.

2.3.2.4. Практическое использование стеганографии

Стеганография как научно-техническая дисциплина бурно развивается. Ежегодно появляются сотни публикаций, описывающих новые алгоритмы и методы, на рынке программного обеспечения присутствуют многочисленные средства стенографической защиты. Однако в силу принципа секретности, лежащего в основе стеганографии, достоверные сведения о реальном использовании этих методов и средств в открытом доступе отсутствуют. Как только такие сведения появляются, описанная в них методика немедленно теряет актуальность.

Тем не менее, можно выделить основные направления развития и применения стеганографии.

1. *Цифровые водяные знаки* (ЦВЗ, digital watermark) – секретные метки внутри мультимедийных файлов (т.е. файлов, содержащих изображения, звуки, видео и т.п.), предназначенные для доказательства их подлинности. Все копии мультимедийного файла содержат одну и ту же метку. Как правило, ЦВЗ изготавливаются по «хрупкой» технологии. При любом искажении файла (например, при обработке изображения в программах типа FotoShop) ЦВЗ разрушаются.

2. *Цифровые отпечатки* (ЦО, digital fingerprint) – секретные метки внутри мультимедийных файлов, предназначенные для отслеживания их нелегального распространения. Все копии мультимедийного файла содержат разные метки, что позволяет при появлении «пиратской» копии определить ее

источник. Как правило, ЦО изготавливаются по «полухрупкой» или робастной технологии.

ЦВЗ и ЦО часто путают с несекретными «полупрозрачными» надписями поверх изображения. «Настоящие» ЦВЗ и ЦО невидимы.

3. *Скрытые метки аутентификации* – секретные признаки, при помощи которых производители коммерческого и условно-бесплатного программного обеспечения пытаются предотвратить нелегальное использование своего продукта. Это могут быть «счетчики количества запусков», «признаки исчерпания тестового периода» (trial period, evaluation period) и тому подобное. Обычно такие метки внедряются в «пустые области» файлов и носителей информации.

4. *Скрытая передача информации*. Использование стенографических методов спецслужбами государств и отдельных корпораций несомненно, но достоверных сведений об этом нет. Зато можно привести многочисленные примеры применения стеганографии в сфере киберкриминала.

> ...Исследователи из SecureWorks (дочерняя компания Dell, приобретенная в 2011 г.) сообщили об активизации трояна Stegoloader, хранящего свои модули в PNG-изображениях. Загруженные трояном изображения формата PNG выглядят как вполне обычные, но в их пикселях записан код модулей Stegoloader. Считывая этот код и подключая модули, троян «собирает» себя прямо в оперативной памяти персонального компьютера... Троян Stegoloader был впервые обнаружен в 2012 г. и является не единственным в своем роде. В апреле 2014 г. был обнаружен троян под названием Lurk, который также подгружал модули, спрятанные в компьютерные изображения формата BMP. В начале 2015 г. компанией AVG был обнаружен банковский троян Vawtrak (другие названия — Neverquest и Snifula), также использующий эту технологию...

ТЕМА 3. Методы шифрования данных. Криптография

Криптография (от греч. κρῦπτοσ– тайный, γράφω — писать) – известная с древности наука о шифровании информации. В более широком, современном смысле это наука о методах и средствах преобразования информации с целью обеспечения ее (информации) конфиденциальности и аутентичности[1].

Шифрование информации – это ее обратимое преобразование в форму, неудобную для восприятия и непосредственного использования (см. рис. 3.1). Шифрование выполняется на основе *ключа* – секретного параметра, являющегося важной частью алгоритма шифрования. После выполненного криптографического преобразования противник не должен получить ни

[1] В отличие от *стеганографии*, факт наличия информации секретом не считается.

единого бита достоверной информации об использованном ключе и содержимом зашифрованных данных.

Расшифрование – операция, обратная к шифрованию, так же выполняется на основе знания *ключа*. В большинстве случаев для прямой и обратной операций используется один и тот же ключ, однако существуют методы криптографического преобразования, в которых ключи шифрования и расшифрования не совпадают.

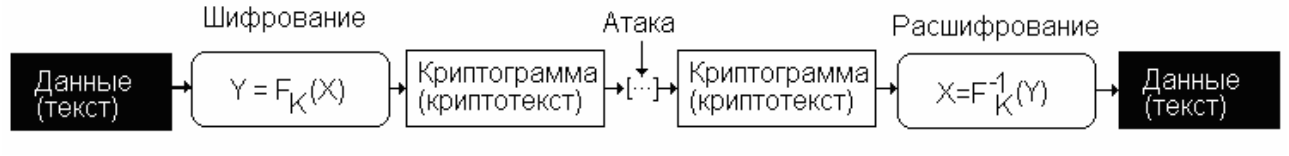


Рис. 3.1. Модель криптографического преобразования

На этом рисунке X – исходная информация, Y – зашифрованная информация, F – известный алгоритм шифрования, F^{-1} – обратный алгоритм, K – ключ.

Иногда отдельно выделяют «*дешифрование*» – операцию расшифрования, выполняемую «злоумышленником» без знания ключа. Правильно составленный и грамотно использованный шифр не должен поддаваться «дешифрованию». Раздел криптографии, изучающий методы «дешифрования» и противодействия им, иногда называют «*криптоанализом*».

Качество шифра определяется следующими характеристиками:

- стойкостью к «дешифрованию»;
- быстродействием операций шифрования и расшифрования;
- простотой реализации и удобством использования.

В настоящее время *стойкость* шифра к взлому измеряется в количестве вычислительных операций, достаточных для «дешифрования» зашифрованной информации.

Современная криптография базируется на *принципе Кирхгофа* (*Керкхоффа, Керккхоффена, Керхкоффса*), сформулированном еще в XIX в., который гласит: стойкость шифра ко взлому должна зависеть только от секретности ключа K, а алгоритм преобразования F может быть открытым.

3.1. История криптографии

Различные методы шифрования использовались еще в античные времена. Основной их целью было обеспечение невозможности прочтения «врагом» текстовой информации. На протяжении полутора тысячелетий, вплоть до середины XX века вопросы криптографии входили в сферу интересов правителей, военачальников, дипломатов, представителей духовенства и т.п. В основном применялись следующие классы криптографических преобразований:

- *перестановочные шифры*, которые основаны на перестановке местами символов текстового сообщения;
- *подстановочные шифры*, которые основаны на замене символов текстового сообщения (или их сочетаний) символами, принадлежащими другому алфавиту.

3.1.1. Перестановочные шифры

Рассмотрим несколько типичных методов криптографического преобразования, относящихся к классу «перестановочных» шифров.

1. «*Позиционные*» (или «*маршрутные*») *шифры*. Их принцип заключается в построении из букв шифруемого текста геометрических фигур и считывании этих букв в другом порядке.

Древнейшим примером криптографических шифров «позиционного» типа является «скитала» или «сцитала» (от греч σκυταλη – жезл). Вот описание способа шифрования:

...Отправляя к месту службы начальника флота или сухопутного войска, эфоры вручают отъезжающему круглую палку. Другую, совершенно одинаковой длины и толщины, оставляют себе. Эти палки и называют скиталами. Когда эфорам нужно сообщить какую-нибудь важную тайну, они вырезают длинную и узкую, вроде ремня, полосу папируса, плотно, без промежутков наматывают ее на свою скиталу и пишут на нем текст. Затем снимают полосу и без палки отправляют ее военачальнику. Так как буквы на ней стоят без всякой связи, разбросаны в беспорядке, прочитать написанное он может, только взяв свою скиталу и намотав на нее вырезанную полосу, чтобы, водя глазами вокруг палки и переходя от предыдущего к последующему, иметь перед собой связное сообщение...

Плутарх. Сравнительные жизнеописания.

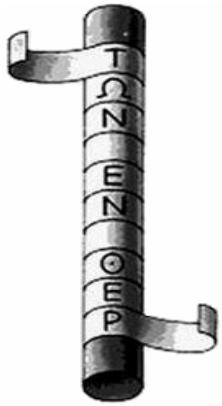


Рис. 3.2 – Скитала

Нетрудно видеть, что в «скитале» реализуется идея прореживания букв текста. Период прореживания зависит от диаметра жезла и, соответственно, от количества букв D, умещающихся на окружности. Таким образом, буквы первой части текста занимают позиции 1, D, 2×D и т.д., продолжение текста размещается в позициях 2, D+1, 2×D+1 и т.д. Например, текст «НАДСЕДОЙРАВНИНОЙМОРЯ» при D=4 примет вид «НДВЙАОНМДЙИО-СРНРЕАОЯ».

Другим примером «маршрутного» шифра является запись текста в виде прямоугольной матрицы, транспонирование ее и считывание букв в прежнем порядке. Например, для текста «КРИПТОГРАФИЯ» можно получить:

```
КРИ          КПГФ
ПТО      →   РТРИ
ГРА          ИОАЯ
ФИЯ
```

и, соответственно, зашифрованный вариант «КПГФРТРИИОАЯ».

Возможны так же иные модификации этого шифра, предусматривающие перестановку столбцов и строк матрицы по определенному правилу, обход матрицы «змейкой» или вдоль «кривой Пеано» и т.п.

2. *Шифры типа* «*Поворотная решетка*» (он же «*решетка Кардано*»). Один из вариантов такого шифра описан в книге Я. Перельмана «Занимательная математика», другой показан в титрах фильма «Шерлок Холмс и доктор Ватсон».

Рисунок 3.3. Решетка Кардано

Буквы сообщения поочередно вписываются в прорези специального трафарета (ключа), потом решетка поворачивается на 90° и буквы продолжают вписываться, затем следует новый поворот и т.д. – см. рис. 3.3. Оставшиеся клетки заполняются случайными буквами.

В случае не квадратного, а прямоугольного трафарета первый поворот осуществляется сразу на 180°, после чего трафарет переворачивается нижней стороной кверху.

Чтение шифра выполняется так же наложением и поворотами трафарета.

3. *Перестановочные шифры.* Их роль в современных технологиях защиты информации невелика. Они легко «вскрываются» методом полного перебора вариантов на ЭВМ. Однако некоторые идеи, использованные в этих шифрах, все же находят свое применение – например, при кодировании информации, передаваемой по каналам связи с редкими, но продолжительными по времени помехами. При передаче битов в исходном порядке это приводит к немногочисленным, но «длинным» ошибкам, в результате чего искажаются группы последовательно расположенных битов. Поэтому перед началом передачи данных «перемешивают» биты одним из способов, свойственных «позиционным» шифрам, а после приема – возвращают их в исходное положение. Это приводит к возникновению многочисленных, но одиночных ошибок, которые легко исправляются, например, при помощи кодов Хэмминга.

3.1.2. Подстановочные шифры

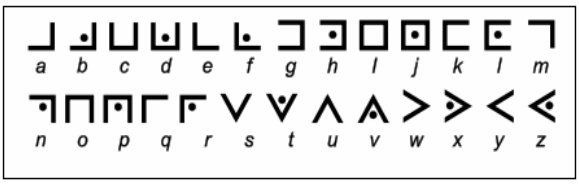
В основе подстановочных шифров лежат «*подстановки*», то есть взаимно-однозначные соответствия (биекции) между разными алфавитами:

$$\alpha_1 \leftrightarrow \beta_1$$
$$\alpha_2 \leftrightarrow \beta_2$$
$$\ldots\ldots\ldots\ldots$$
$$\alpha_N \leftrightarrow \beta_N.$$

Правила подстановок могут быть изображены в виде таблицы, представлены в виде формулы, описаны текстом и т.п. В докомпьютерную эпоху важнейшим достоинством шифра являлась легкость запоминания соответствующей подстановки.

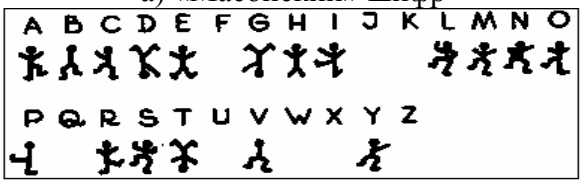
1. *Шифры простой одноалфавитной замены* – самые простые, эффектные, но весьма ненадежные способы шифрования. Первый алфавит – это обычный алфавит какого-нибудь языка (русского, английского, латинского,

греческого и т.п.), второй же может состоять из каких-нибудь экзотических знаков (см. рис.3.4), чисел, буквосочетаний и т.п.



а) «Масонский» шифр

в) «Цифирная азбука» Петра I



б) Шифр «Пляшущие человечки»

г) Шифр И.А.Толстого и В.В. Долгорукого

Рис. 3.4. Примеры шифров простой одноалфавитной замены

Так же возможно, что второй алфавит состоит из букв того же алфавита, переставленных местами по какому-либо правилу. Примером может служить старинный русский шифр «простая литорея» (он же «тарабарская грамота»). Идея построения подстановки заключается в том, что согласные буквы разбиваются на две группы – первая содержит 10 первых букв в прямом алфавитном порядке {Б, В, Г, Д, Ж, З, К, Л, М, Н}, а вторая – 10 последних в обратном порядке {Щ, Ш, Ч, Ц, Х, Ф, Т, С, Р, П}. При шифровании гласные остаются на своих местах, а согласные, принадлежащие разным группам, меняются местами. Например, слово «КРИПТОГРАФИЯ» превратится в «ТМИНКОЧМАЗИЯ».

Способ легкого построения и запоминания «перемешанных» алфавитов изобрел Полибий – греческий историк, полководец и государственный деятель III века до н.э. Идея заключается в сложении из букв алфавита прямоугольной матрицы, первую строку которой занимает какое-нибудь слово из разных букв, а остальные строки содержат оставшиеся буквы в алфавитном порядке (см. рис. 3.5). Получившийся «квадрат Полибия» можно использовать по-разному. Например, можно сопоставить каждой букве ее числовые «координаты», тогда из слова «КЛЮЧ» получится число «17284741».

	1	2	3	4	5	6	7	8
1	Ш	И	Ф	Р	О	В	К	А
2	Б	Г	Д	Е	Ж	З	Й	Л
3	М	Н	П	С	Е	У	Х	Ц
4	Ч	Щ	Ъ	Ы	Ь	Э	Ю	Я

Рис. 3.5. Пример «Полибианского квадрата»

Сам же Полибий предпочитал менять каждую букву исходного текста на ее «соседку», например, расположенную сверху-справа. В этом случае «КЛЮЧ» превратится в «ЯШЦН». Обратите внимание: матрица считается «закольцованной» по горизонтали и вертикали, так что справа от столбца 8 расположен столбец 1, а сверху от столбца 1 столбец 4.

Очень давно, еще в Средние века было обнаружено, что дешифрование любых подстановочных шифров легко выполняется с использованием таблиц частоты встречаемости букв и их сочетаний, характерных для того или иного языка.

Так, например, для русского языка частоты встречаемости букв и сочетаний таковы: пробел – 0.17, «ТО» – 0.12 , «СТ» – 0.11, «О» – 0.094, «Е» – 0.071, «А» – 0.069, «И» – 0.064, «Н» – 0.057, «Т» – 0.054, «С» – 0.046 и т.д.

Для английского языка: «E» – 0.12, «T» – 0.091, «A» – 0.082, «O» – 0.075, «I» – 0.069, «N» – 0.067, «S» – 0.063, «H» – 0.061, «R» – 0.060… и т.д.

Для любых достаточно длинных текстов (порядка 1000 знаков и более) эта статистика подтверждается настолько стабильно, что для «автоматического» дешифрования можно формально заменять наиболее часто встречающийся в русской криптограмме знак на «О», следующий по частоте на «Е» и т.д.; аналогично для криптограмм на других языках.

2. *Шифры многоалфавитной замены*. К началу эпохи Возрождения в Европе сложилось твердое мнение о недостаточной криптостойкости шифров простой одноалфавитной замены. Взамен широкое распространение получили шифры многоалфавитной замены.

Наиболее простые шифры этого типа предусматривали последовательное использование нескольких различных подстановок. Например, для шифра, подстановки которого изображены на рис. 3.6, правило кодирования будет следующим: первая буква текста заменяется по первой подстановке, вторая – по второй, третья – по третьей, четвертая – опять по первой и так далее. В этом случае слово «КРИПТОГРАФИЯ» будет зашифровано как «ХЦЩРШВЬЦЙЛМЮ».

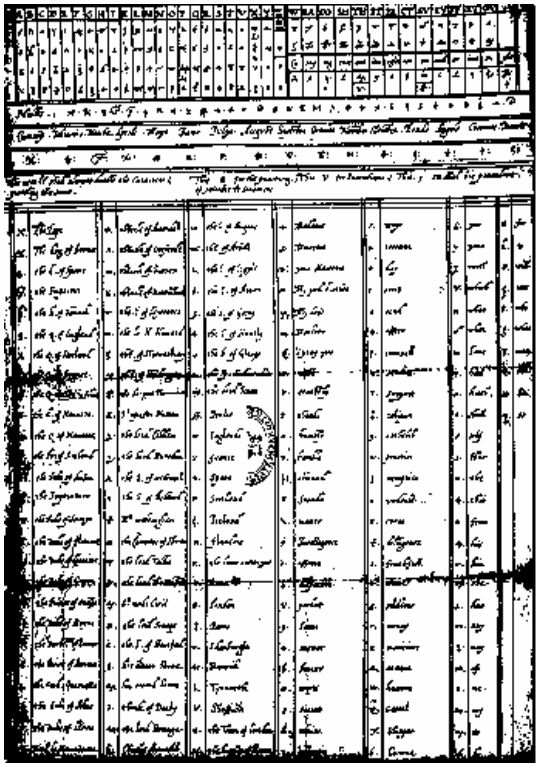
АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
ЯЮЭЬЫЪЩШЧЦХФУТСРПОНМЛКЙИЗЖЕДГВБА
БВГДЖЗКЛМНПРСТФХЦЧШЩЪЬАЕИЙОУЫЭЮЯ
ЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮ

Рис. 3.6. Пример ключа для шифра многоалфавитной замены

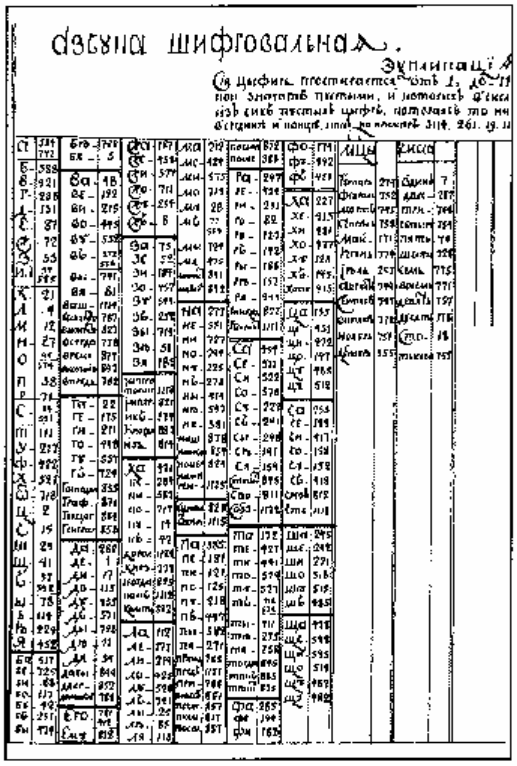
Шифры пропорциональной замены занимают промежуточное положение между «шифрами простой замены» и «шифрами многоалфавитной замены». В них редко встречающимся буквам соответствуют одиночные подстановки, а часто встречающимся (например, всем гласным) – множественные.

Более сложные системы шифрования, имевшие широкое распространение в XV-XIX в., были основаны на ключах типа «*номенклатор*» (или «*словарный код*»). Так назывались документы, описывавшие одну или несколько подстановок, а так же дополнительные правила их использования (так называемые «*суплементы*»), например:

- «пустышки» – сочетания знаков в криптограмме, которые при расшифровании должны просто пропускаться;
- «триггеры» – сочетания знаков, которые предписывали переключение с одной подстановки на другую;
- заранее предопределенные сочетания знаков для обозначения цифр, дней недели, месяцев года, географических наименований, имен государственных деятелей и т.п.



а) Шифр Марии Стюарт, XVI в.



б) Русский дипломатический шифр, XVIII в.

Рис. 3.7. Примеры простых «номенклаторов»

Простые «номенклаторы» умещались на одном листе бумаги (см. рис. 3.7), а сложные нередко представляли собой толстые «шифровальные книги». По такому принципу были построены дипломатические шифры посольств разных стран Европы, личные шифры правителей и государственных деятелей, российский военно-морской шифр XVIII-XIX вв. «мудрая литорея» и многие другие.

Несмотря на замысловатость, «номенклаторные» шифры нередко подвергались взлому. История знает имена выдающихся криптоаналитиков, в одиночку решавших сложнейшие проблемы такого рода: Чиро Симонетта – Милан, XV век; Джованни Соро (Венеция), Пирро Музефили (Флоренция), Филибер Бабу и Франсуа Виет (Франция), Ван Марникс (Фламандия), Томас Фелиппес (Англия) – все XVI век. Позже, в XVII-XIX веках взлом шифров был поставлен на поток в «*черных кабинетах*» – государственных службах европейских стран (Австрии, Англии, Франции, России и т.п.), занимавшихся перехватом и чтением чужой переписки.

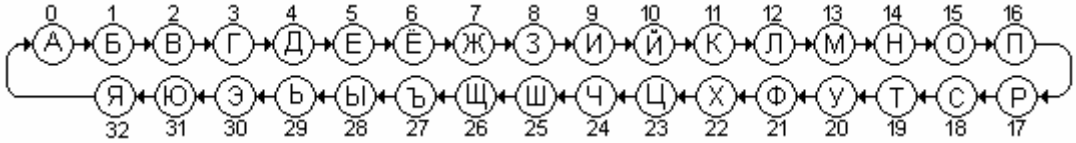
Среди шифров многоалфавитной замены можно так же выделить «книжные» шифры, которые приобрели популярность в XIX-XX веках. В шифрах этого типа в качестве таблицы подстановок использовался книжный текст. Каждая буква шифруемого теста заменялась на ее числовые «координаты» в книге. Поскольку одна и та же буква обычно встречается в книге неоднократно, этим обстоятельством легко обеспечивается множественность используемых алфавитов. Для шифрования обычно выбирались страницы в середине книги и фрагменты текста, начинающиеся не с первой строки.

```
...Слушайте  наши радиопередачи по-прежнему в то же время;
новый ключ к расшифровке вы найдете  в  романе Бичер-Стоу,
"Детгиз", 1965 год, страница 82, переданном вам во время
прошлого контакта...
Ю. Семенов. ТАСС уполномочен заявить
```

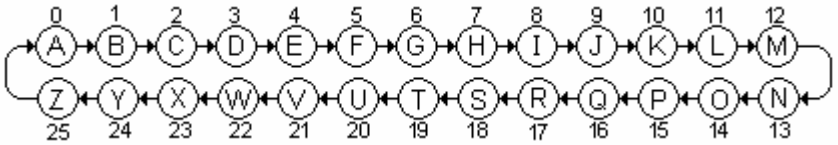
Например, применив для шифрования страницу с первыми строками «Евгения Онегина», изображенную на рис. 2.13, слово «СЕКРЕТ» можно зашифровать несколькими различными способами, вот два из них: «1/8 1/14 5/19 1/21 2/7 3/8» и «1/15, 2/7, 6/10, 5/5, 1/14, 8/14». Здесь в числителе каждой дроби указан номер строки, а в знаменателе – номер буквы.

На практике для увеличения криптостойкости «простых» шифров часто применялось «*перешифрование*» – совместное использование нескольких способов кодирования. Так, например, советская внешняя разведка 1930-40-х годов (Рихард Зорге, «Красная капелла» и пр.) с успехом применяла комбинацию «квадрата Полибия» и «книжного шифра».

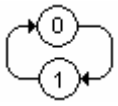
3. *Шифры группы Тритемиуса*. Шифры, описанные в книге аббата Иоганна Тритемиуса «Полиграфия» (1518 г.), по сути представляют собой несложные подстановочные шифры. Однако им уделено особое внимание в силу их практической ценности и теоретической важности.



а) Русский алфавит



б) Латинский алфавит



в) Двоичный алфавит

Рис. 3.7. «Зацикленные» алфавиты

Последовательно пронумеруем все буквы какого-либо алфавита и введем в этом алфавите арифметическую операцию «сложения» букв по правилу: сумма двух букв с номерами i и j есть буква, номер n которой в алфавите равен n=i+j. Для того, чтобы не возникало номеров больших, чем количество букв в алфавите, алфавит «зацикливается». То есть, за последней буквой в алфавите опять следует первая, потом вторая и так далее (см. рис. 3.7). Фактически, это означает:

$$n = (i+j) \bmod |A|,$$

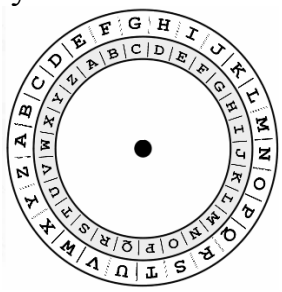
где |A| – «мощность» алфавита.

Например, для русского алфавита (см. рис. 3.7,а):

```
А⊕А=А
Б⊕В=Г
Б⊕Я=А.
```

Шифр Цезаря, относящийся к группе шифров Тритемиуса, но придуманный гораздо ранее римским императором Гаем Юлием Цезарем для секретной переписки со своими сенаторами, основан на простой подстановке:

каждая буква алфавита заменяется на другую букву того же алфавита, сдвинутую относительно нее на какое-то фиксированное расстояние. В терминах «сложения букв» это означает, что к каждой букве шифруемого сообщения прибавляется некоторая другая буква – одна и та же для всего сообщения. Например, если ко всем буквам слова «ЯСЕНЬ» прибавить букву «Н», то получится слово «МЯТЫЙ» (проверьте!).



а) Щит Цезаря

	АБВГДЕЁЖЗ	Я
А	АБВГДЕЁЖЗ	Я
Б	БВГДЕЁЖЗИ	А
В	ВГДЕЁЖЗИЙ	Б
...	...	...
Ю	ЮЯАБВГДЕЁ	Э
Я	ЯАБВГДУЕЖ	Ю

б) Фрагмент «таблицы сложения»

Рис. 3.8. Приспособления для облегчения шифрования методом Цезаря

Для того, чтобы облегчить сложение букв, Гай Юлий Цезарь пользовался круглым щитом, по ободу которого были выписаны буквы алфавита (см. рис. 3.8,а). Иоганн Тритемиус для этой же цели рекомендовал составить «таблицу сложения» и просто выписывать из нее буквы, стоящие на пересечении соответствующих строк и столбцов (см. рис. 3.8,б).

Шифр Цезаря взламывается очень легко – перебором всевозможных вариантов ключа.

Шифр Вижинера (названный в честь французского дипломата Блеза Вижинера) так же относится к «группе Тритемиуса» и является усложненным вариантом «шифра Цезаря»: теперь буквы исходного текста сдвигаются на разное количество позиций, а ключом для сложения букв является некоторое слово или фраза. Если ключ слишком длинен, то он усекается; если короток, то размножается необходимое количество раз.

Пусть слово «БУБЕН» шифруется с ключом «ПЯТНО». Складывая «Б» с «П», получим «Р»; складывая «У» с «Я», получим «Т» и т.д. Окончательно имеем: «БУБЕН»⊕«ПЯТНО» = «РТУТЬ» (проверьте!).

Шифр Вижинера долгое время считался исключительно надежным, но в 20-х годах XX века появилась математическая теория, которая сделала его взлом вполне возможным.

Шифр Вернама – разновидность шифра Вижинера, основанная на двоичном алфавите (см. рис. 3.7,в). Для двоичного алфавита, состоящего из 0 и 1, модульное сложение эквивалентно логической операции «XOR»:

0⊕0=0
0⊕1=1
1⊕0=1
1⊕1=0.

Последовательность нулей и единиц ключа, предназначенная для наложения на нули и единицы исходного сообщения, получила наименование «*гаммы*» шифра, а сама эта операция названа «*гаммированием*». Пример – гаммирование слова «МИР», записанного в двоичном коде, с ключом «1001» –

приведен на рис. 3.9. Патент на систему шифрования и соответствующее устройство был получен в 1919 Гилбертом Вернамом, инженером американской корпорации AT&T. Основным ее назначением считалось шифрование телеграфных сообщений, закодированных при помощи «азбуки Морзе». Ключ изготавливался в виде двух одинаковых, склеенных в кольцо бумажных лент с отверстиями и «пробелами». Первая копия использовалась для шифровании на источнике сообщений, другая – для расшифрования на приемнике (см. ниже рис. 3.14,а).

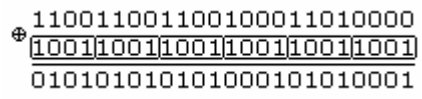


Рис. 3.9. Гаммирование

Роторные машины – средство механической автоматизации процесса шифрования. Они были изобретены в XIX веке, различные модели использовались вплоть до последних десятилетий XX века и потеряли актуальность лишь с появлением быстродействующих ЭВМ.

Шифрующий механизм машин представляет собой «упаковку» из нескольких роторов (дисков), насаженных на общую ось и касающихся друг друга электрическими контактами, которые размещены по окружности. Каждому контакту сопоставлена определенная буква алфавита. Таким образом, два соседних ротора реализуют некоторую «переменную» подстановку, способную изменяться в результате их поворота. Контакты, размещенные на разных сторонах роторов, связаны внутри ротора системой проводников и так же образуют некоторую «жесткую» подстановку.


а) Немецкая машина «Энигма» (1930-40-е годы) – 3 или 4 ротора


б) Советская машина «Фиалка М-125» (1950-80-е годы) – 10 роторов


в) Система роторов в сборке

Рис. 3.10. Примеры роторных машин

Итак, каждый входной контакт первого ротора связан с некоторым выходным контактом последнего, образуя непрерывную электрическую цепь. Замыкание ключа, сопоставленного одной из букв первого ротора, вызывает загорание лампочки, связанной с одной из букв последнего (см. рис. 3.11).

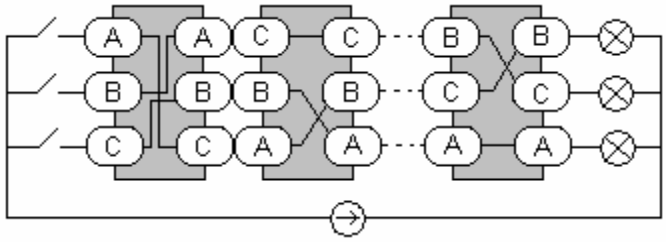


Рис. 3.11. Принцип действия роторной машины

Одновременно с замыканием ключа роторы через систему зубчатых шестерней поворачиваются на определенные углы, например: первый на одну позицию; второй на две и так далее, – благодаря чему каждый раз образовывается новая, совершенно иная система подстановок. Она будет использована для шифрования следующей буквы, после чего снова изменится, потом еще раз, и еще – пока весь текст не окажется зашифрован.

Очевидно, если размер алфавита N, а количество роторов M, то полное количество подстановок, используемых шифровальной машиной в псевдослучайном порядке, составляет N^M. В худшем случае для взлома текстов, зашифрованных 4-роторной машиной «Энигма», потребовалось бы около 9400 лет непрерывной работы процессора с тактовой частотой 3500 МГц. В 2007-2009 гг. в рамках проекта распределенных вычислений ENIGMA@HOME были успешно расшифрованы несколько немецких радиограмм времен Второй Мировой войны.

3.1.3. Абсолютно стойкие шифры

Абсолютно стойкие шифры, невозможность «взлома» которых доказана К.Шенноном в 1945 г., являются обобщением шифров, использующих модульное сложение в «зацикленных» алфавитах (см. рис. 3.7).

Недостатком традиционных шифров Вижинера и Вернама является простота их дешифрования при некоторых частных случаях ключей и шифруемых данных. Примеры использования недостатков:

- если исходные данные X=00…0, то «зашифрованные» данные представляют собой ключ шифра: X⊕K=K;
- если ключ K=00…0, то «зашифрованные» данные представляют собой исходные данные: X⊕0=X;
- если достоверно известен фрагмент зашифрованных данных (например, заголовок документа), то его сложение с соответствующим фрагментом зашифрованных данных представляет собой фрагмент ключа;
- если известны два разных сообщения $Y_1=X_1\oplus K$ и $Y_2=X_2\oplus K$, зашифрованных одним и тем же ключом K, то их сложение исключает ключ: $Y_1\oplus Y_2=(X_1\oplus K)\oplus(X_2\oplus K)=X_1\oplus X_2$.

К. Шеннон строго доказал утверждение о том, что любой шифр, основанный на «*гаммировании*», то есть на модульном сложении элементов данных с элементами ключа в «зацикленном» алфавите, является абсолютно стойким при соблюдении следующих условий:

- ключ шифрования имеет ту же длину, что и шифруемое сообщение (то есть содержит такое же количество элементов);
- ключ шифрования полностью случаен (то есть получен в результате какого-либо повторно невоспроизводимого физического процесса – например, подбрасывания монеты);
- ключ используется однократно, а после использования уничтожается.

Описанная схема шифрования получила наименование «*одноразового шифроблокнота*» или «*одноразовой кодовой книги*». Это наименование связано

с широким использованием подобных шифров в разведке и при обмене короткими дипломатическими сообщениями.

Однако во всех остальных случаях применение абсолютно стойких шифров экономически невыгодно. Так, например, при шифровании баз данных необходимо где-то хранить ключ такого же объема, что и сами эти данные. Так же, эти шифры совершенно не применимы для шифрования потоков данных, например, Интернет-трафика.

3.2. Общая характеристика современных шифров

Особенностью современных шифров является их ориентация на реализацию средствами быстродействующей вычислительной техники. Причем, под «вычислительной техникой» в первую очередь понимаются не обычные компьютеры с шифрующими и расшифровывающими программами, а специализированные вычислительные устройства на базе «сигнальных процессоров» криптографического назначения (см. рис. 3.12). Криптостойкость таких шифров на много порядков превышает возможности «классических» шифров, рассмотренных ранее[1].

Как правило, шифруемые данные в современных условиях являются не только текстами, но и изображениями, мелодиями, видеофильмами – фактически, в их роли выступает любая информация, представимая в виде набора битов.

а) Шифровальная «карточка» в формате PCMCI для подключения к ноутбуку (США)

б) Стационарный шифрователь сетевого трафика, Россия

в) Плата для прозрачного шифрования данных на жестком диске, Россия

Рис. 3.12. Криптографическое оборудование

Шифровальные ключи так же представляют собой наборы битов определенной длины.

Операция шифрования представляет собой очень сложную нелинейную (это важно!) функцию, включающую в себя подстановки и перестановки битов. Результат применения этой функции практически всегда выглядит как хаотичный «мусор», в котором никакие статистические тесты не обнаруживают никаких закономерностей.

Тем не менее, для того, чтобы «противник» после зашифрования не имел вообще никаких достоверных сведений ни об одном бите исходных данных, применяют дополнительную обработку исходных и зашифрованных данных.

[1] Разумеется, кроме «одноразовых шифроблокнотов».

Например, для того, чтобы одинаковые данные, будучи зашифрованы одним и тем же ключом, выглядели каждый раз по-разному, применяют «разбавление» их случайными битами, расположенными на заранее оговоренных позициях. Такая операция называется «подсаливанием» (salting), а случайные биты – «солью» (salt).

Однако, используемые на практике «компьютерные» шифры, отличаясь от «одноразового шифроблокнота», все же не являются абсолютно стойкими – все они подвержены «взлому» общим универсальным методом – полным перебором всевозможных вариантов ключа (его еще называют «brute force» (англ.) – метод «грубой силы»).

Криптостойкость современных шифров напрямую зависит от длины ключа и оценивается количеством вычислительных операций, достаточных для дешифрования. Так же используются иные, «производные» критерии криптостойкости: например, количество материальных ресурсов (процессоров, ячеек памяти и пр.), потребное для дешифрования за указанный интервал времени.

Произведем приблизительные расчеты стойкости современных шифров к методу «полного перебора». Пусть каждый из 8 миллиардов жителей планеты Земля (это примерно 2^{33} человек) имеет компьютер с быстродействующим процессором, способным перебирать миллиард (то есть, $10^9 \approx 2^{30}$) ключей в секунду. Тогда все вместе они переберут за секунду $2^{33+30} \approx 2^{63}$ ключей. В году $31536000 \approx 2^{25}$ секунд, следовательно, непрерывно производя вычисления в течение года, эти компьютеры переберут $2^{63+25}=2^{88}$ ключей[1]. Посчитайте дальше сами, за какой срок будут «взломаны» шифры с ключами длиной 128, 192 и 256 битов.

Шифры, для которых существуют способы дешифрования, менее трудоемкие, чем полный перебор ключей, считаются «ненадежными» или «уязвимыми». Тем не менее, некоторые из таких шифров продолжают использоваться, например, ввиду того, что количество материальных ресурсов, потребных для практической реализации атаки, превышает возможности потенциального «взломщика».

Так же, рассматривая понятие «криптостойкости» шифра, следует иметь в виду различные типы атак. Пусть, по-прежнему, операция шифрования заключается в реализации уравнения

$$Y=F_K(X),$$

где X – открытый текст (т.е. незашифрованные данные), Y – закрытый текст (т.е. криптограмма, результат шифрования), K – секретный ключ, F – общеизвестный алгоритм шифрования (см. рис. 3.1). Тогда возможны следующие типы атак.

Атака 1-го типа. «Взломщику» известно только некоторое количество Y_i, зашифрованных одним и тем же ключом K. Требуется определить ключ K и,

[1] Расчеты производились «с запасом». Реальные цифры быстродействия процессоров и их количества в мире существенно ниже.

соответственно, содержимое «открытых текстов» X_i. Это самый труднореализуемый тип атак, все современные шифры устойчивы к нему.

Атака 2-го типа. «Взломщику» известно некоторое количество пар (X_i,Y_i), зашифрованных одним и тем же ключом K. Требуется определить ключ K, чтобы в дальнейшем расшифровывать любые Y_i. Для некоторых современных шифров разработаны методы реализации атак подобного типа, правда, они требуют либо нереально большого количества пар (X_i,Y_i), либо нереального большого количества вычислительных ресурсов.

Атака 3-го типа. «Взломщику» доступен «шифрователь», он имеет возможность самостоятельно формировать «открытые тексты» X_i и получать для них зашифрованные Y_i. Требуется определить ключ K, чтобы в дальнейшем расшифровывать любые Y_i. Некоторые современные шифры могут при определенных условиях быть уязвимы к атакам подобного типа.

Рис. 3.13. Классификация современных шифров

В настоящее время существует и используется на практике большое количество разнообразных шифров. Их общая классификация изображена на рис. 3.13.

Классификация по способу использования:

- *блочные шифры* предусматривают предварительное разбиение шифруемых данных на фрагменты (блоки) и последующее их (блоков) шифрование;
- *потоковые шифры* способны последовательно шифровать данные по мере их поступления.

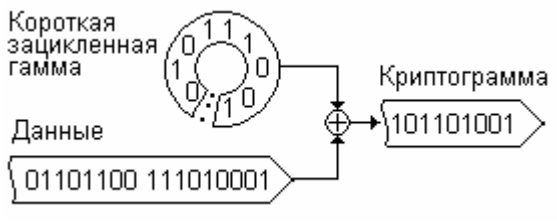
Классификация по виду ключа:

- в *симметричных* шифрах для шифрования и расшифрования используется один и тот же ключ K;
- в *асимметричных* (или *несимметричных*) шифрах для шифрования используется ключ K_E, а для расшифрования ключ K_D, причем $K_E \neq K_D$.

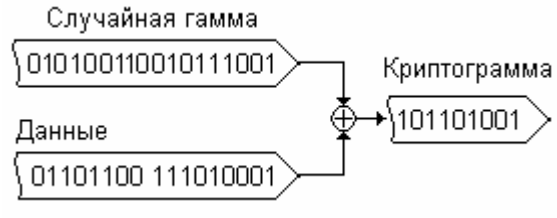
3.3. Потоковые шифры

Потоковые шифры преобразуют отдельные элементы данных – например, биты сообщения – по мере их поступления. Основное назначение таких шифров – работа в составе систем цифровых коммуникаций. Но это не отменяет возможного использования таких шифров для защиты, например, баз данных.

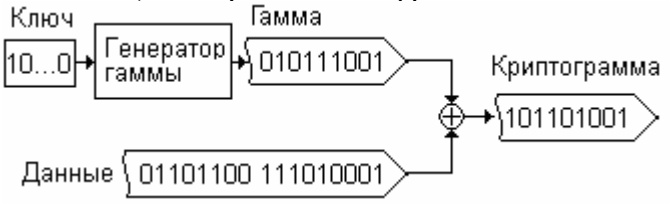
Большинство потоковых шифров восходят к шифру Вернама и основаны на методе наложения двоичной «гаммы» на исходное сообщение (см. рис. 3.9).



а) Шифр Вернама



б) Одноразовый шифроблокнот



в) Современные потоковые шифры

Рис. 3.14. Виды потоковых шифров

Рассмотренный выше «оригинальный» шифр Вернама использовал короткую «гамму» в виде закольцованной ленты с «нулями» и «единицами» (см. рис. 3.14,а). Для «одноразового шифроблокнота» гаммой служит абсолютно случайная последовательность «нулей» и «единиц», полученная, например, в результате подбрасывания монеты (см. рис. 3.14,б). Современные потоковые шифры занимают промежуточное положение между «шифром Вернама» и «одноразовым шифроблокнотом» – в них двоичная гамма есть результат работы некоторого «генератора нулей и единиц», инициализируемого при помощи относительно короткого ключа. Эта последовательность не является истинно случайной, так как жестко связана с ключом и в любой момент может быть воспроизведена.

Рассмотрим некоторые генераторы псевдослучайных чисел (ГПСЧ), которые могли бы использоваться в качестве источника такой «гаммы», и реально используемые потоковые шифры.

1. *Линейный конгруэнтный ГПСЧ* – наиболее простой и популярный способ для генерации последовательности чисел, мало отличающихся по статистическим свойствам от «абсолютно случайных». Идея их генерации основывается на рекуррентном правиле вида

$r_{i+1}=(A*r_i+B) \bmod M$,

где A, B и M – некоторые целые константы, а $\{r_i\}$ – итоговая последовательность псевдослучайных чисел.

Начальное значение r_0 (так называемая «*затравка*») выбирается любое, оно полностью определяет всю последовательность и поэтому может выступать в роли ключа K. Максимальная длина последовательности M, после чего вся последовательность повторяется. При неудачно выбранных A, B и M последовательность может получиться гораздо короче максимальной, поэтому правила оптимального выбора таковы:

- константы B и M не должны иметь общих сомножителей;
- если M – простое число, то A может быть любым;
- если M составное, то значение (A-1) должно быть кратно P для любого простого P, являющегося делителем M; при этом, если M делится на 4, то и A-1 должно делиться на 4.

Хорошими константами, являются, например:

- $M=2^{32}$, B=1, A=22695477 (используются в Borland C/C++);

- $M=2^{32}$, B=2531011, A=214013 (используются в MS Visual C/C++);
- $M=2^{48}$, B=11, A=25214903917 (используются в Java).

Генераторы такого типа включены во все компиляторы с языков программирования, они часто используются для создания эффекта «стохастизма» в компьютерных играх, при моделировании случайных процессов и явлений. Однако применение подобных генераторов в технологиях защиты информации ограничено, так как, имея четыре соседних значения r_{i-3}, r_{i-2}, r_{i-1}, r_i, всегда можно легко определить A, B и M. Тем не менее, двоичные гаммы, сгенерированные при мощи «линейных конгруэнтных генераторов», все же находят применение в тех случаях, когда от шифра не требуется криптостойкости, а цель шифрования – всего лишь закрыть доступ к данным «рядовому» пользователю. В качестве примера можно привести шифрование баз данных формата «.DBF» в СУБД Clipper (1990-е годы).

2. Сдвиговые регистры с линейной обратной связью (LFSR – Linear Feedback Shift Register) – аппаратная схема для генерации псевдослучайной последовательности «нулей» и «единиц».

Центральным ее элементом является регистр, содержащий некоторое N-битовое число. «*Вектор инициализации*», то есть первоначальное значение этого числа, выбирается предварительно и фиксируется. Поскольку вся последовательность псевдослучайных битов на выходе схемы жестко связана с «вектором инициализации», он может выступать в качестве ключа К.

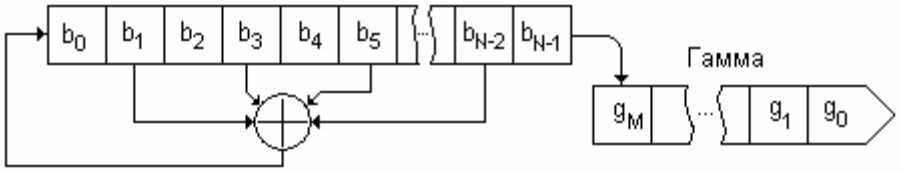


Рис. 3.15. Генерация псевдослучайной гаммы с помощью LFSR

Работа схемы состоит из последовательности одинаковых шагов. На каждом шаге некоторые, заранее выбранные биты содержимого регистра суммируются по модулю 2, образуя значение еще одного бита. Например, на рис. 3.15 этот бит «b» образуется как сумма $b=b_1 \oplus b_3 \oplus b_5 \oplus \ldots \oplus b_{N-2}$. После этого содержимое регистра сдвигается на одну позицию так, что один из крайних битов b_{N-1} «выталкивается», а позиция противоположного бита b_0 «освобождается». В конце шага «освобожденная» позиция заполняется вычисленным битом b, а значение «вытесненного» бита b_{N-1} не теряется и используется в качестве очередного бита «гаммы».

Последовательность битов «гаммы», сгенерированной при помощи N-разрядного LFSR с фиксированным «вектором инициализации», повторяется всего через 2^N-1 шагов. Поэтому на практике используются комбинации подобных схем. Например, алгоритм одного из вариантов шифра «A5», используемого для защиты трафика сотовой связи, использует комбинацию из трех регистров: 19-разрядного, 22-разрядного и 32-разрядного.

3. *Шифр RC4* (Р. Ривест, 1987 г.) – один из самых известных современных шифров. Отличается высокой скоростью работы и простотой программной

реализации. Последовательность псевдослучайных чисел, генерируемая этим шифром, выбирается из таблицы, постоянно «перемешиваемой» с помощью ключа, длины которого могут принимать значения 8, 16, 24…512 бит (типичное значение – 128).

Шифр RC4 является частью протоколов SSL/TLS и примерно до 2005 г. служил основным методом шифрования сетевого трафика в Интернете. Но в последние годы в нем обнаружен ряд уязвимостей к атакам 2-го и 3-го «типа» (см. раздел «Общая характеристика современных шифров»). Например, в рамках одной из демонстрационных атак бельгийские исследователи на основе примерно миллиарда пар {исходный текст+криптограмма} в течение 52 часов определили 94% битов 128-битового ключа.

Практическая же стойкость RC4 в условиях одноразового или часто изменяемого ключа по-прежнему очень высока. Чтобы не допускать повторяемости ключа, его комбинируют с «оказией» – псевдослучайным числом. В настоящее время RC4 шифр с успехом используется для сокрытия содержимого электронных документов в формате «PDF», документов и электронных таблиц младших версий MS Office[1], для шифрования трафика в беспроводных сетях (технология Wep) и пр. Применяют его и авторы вредоносных программ, шифруя файлы на жестком диске и вымогая за расшифрование деньги.

4. *Шифры семейства A5* предназначены для обеспечения конфиденциальности передаваемых данных в европейском сегменте мобильной цифровой связи стандарта GSM. Принцип действия основан на «каскадировании» нескольких LFSR (в ранних модификациях использовались 3 регистра, в более поздних – 4). Длина ключа, используемого в этом шифре, всего 64 бита, так что он (шифр) никогда не рассматривался в качестве средства для абсолютной защиты трафика. Основным назначением A5 считается предотвращение прослушивания переговоров в режиме реального времени. На протяжении всей истории этого шифра в нем обнаруживался ряд критических уязвимостей, которые последовательно устранялись введением новых модификаций: «A5/1», «A5/2» и т.д.

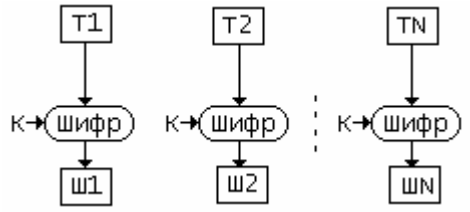
3.4. Блочные шифры

Блочные шифры преобразуют исходные данные в набор «блоков» одинакового размера (например, 32, 64 или 128 бит) и зашифровывают эти блоки независимо друг от друга. Теоретические исследования и результаты практического применения показывают, что шифры подобного типа при достаточной длине ключа обладают чрезвычайно высокой криптостойкостью.

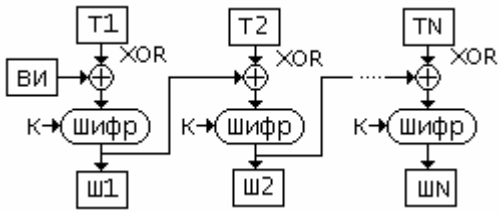
Для того, чтобы одинаковые блоки данных после шифрования выглядели по разному, применяют различные методы сцепления блоков друг с другом при помощи операции «XOR» (см. рис. 3.16). На этих рисунках $Т_i$ и $Ш_i$ – блоки текста до и после шифрования, К – ключ шифрования, $\oplus$ – операция

[1] В MS Office 2007 только AES, а в более поздних версиях – любой шифр из CryptoAPI.

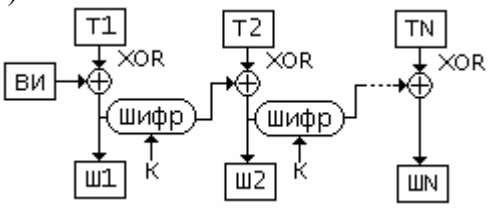
побитового «XOR», ВИ – вектор инициализации (заранее оговоренный набор битов).



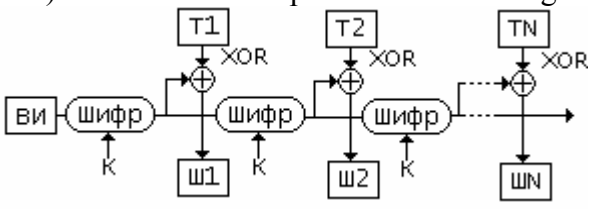
а) Режим ECB – Electronic Codebook



б) Режим CBC – Cipher Block Chaining



в) Режим CFB – Cipher Feedback

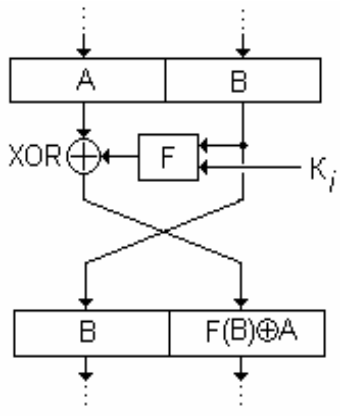


г) Режим OFB- Output Feedback

Рисунок 3.16. Режимы сцепления блоков и обратной связи

1. Шифры на основе сетей Фейстеля – наиболее обширный и хорошо изученный класс шифров, восходящих к разработкам исследовательских лабораторий фирмы IBM конца 1960-х годов. В их основе лежит идея использования циклической схемы преобразований – см. рис. 3.17,а. Для зашифрования необходимо последовательно выполнить N одинаковых «раундов», для расшифрования – их же, но в обратном порядке. Структура преобразований одного такого «раунда» изображена на рис. 3.17,б. Битовый блок разбивается на две половины, которые в ходе преобразований меняются местами. При этом одна из половин остается неизменной, а другая преобразуется при помощи некоторого алгоритма F с использованием одной из составных частей общего ключа K.

а) Общая схема преобразований



б) Отдельный раунд преобразований

Рис. 3.17. Структура шифра, использующего «сети Фейстеля»

Алгоритм F – это то, чем разные шифры отличаются друг от друга. Обычно в этих алгоритмах используются арифметические и логические операции, перестановки и подстановки битов.

Чаще всего подстановки оформляются в виде таблиц специального вида – так называемых «узлов замен» (англ. «substitution box» или «S-box»). Например,

такой S-бокс может преобразовывать 6-битовые числа в 4-битовые на основе таблицы (см. табл. 3.1).

Таблица 3.1 – Пример «S-бокса»

	0	**1**	**2**	**…**	**14**	**15**
0	$n_{0,0}$	$n_{0,1}$	$n_{0,2}$	…	$n_{0,14}$	$n_{0,15}$
1	$n_{1,0}$	$n_{1,1}$	$n_{1,2}$	…	$n_{1,14}$	$n_{1,15}$
2	$n_{2,0}$	$n_{2,1}$	$n_{2,2}$	…	$n_{2,14}$	$n_{2,15}$
3	$n_{3,0}$	$n_{3,1}$	$n_{3,2}$	…	$n_{3,14}$	$n_{3,15}$

Пусть необходимо найти подстановку для числа $38_{10} = 100010_2$. Два крайних бита используются для выбора строки, они образуют двоичное число 10_2, то есть 2_{10}. Четыре средних бита $0001_2=1_{10}$ выбирают столбец. Итак, результатом подстановки является число $n_{2,1}$.

Перестановки оформляются в виде «таблиц проволочной коммутации» (англ. «permutation box» или «P-box»). Например, такой P-бокс для 16-битовых чисел может выглядеть как на табл. 3.2.

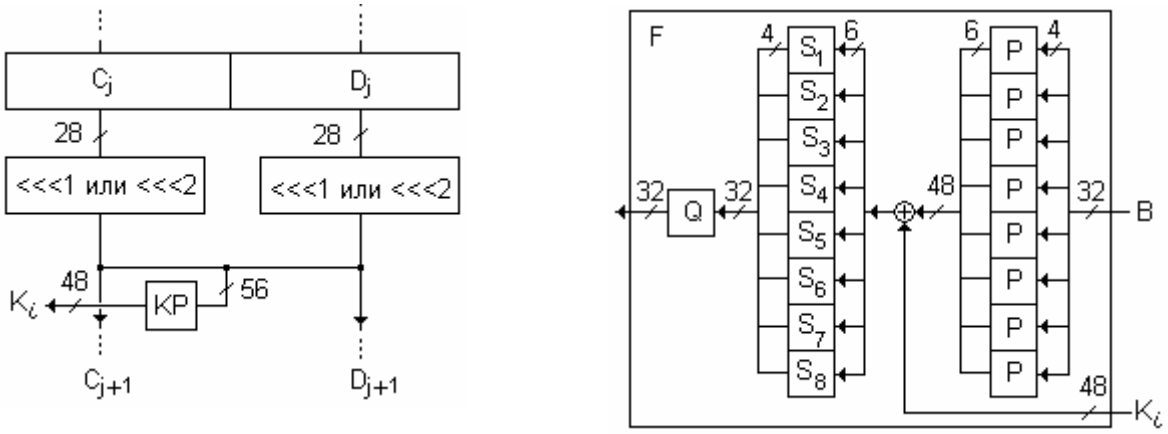
Таблица 3.2 – Пример «P-бокса»

0	**1**	**2**	**…**	**14**	**15**
n_0	n_1	n_2	…	n_{14}	n_{15}

В соответствии с этой таблицей новое положение для бита с номером 0 задается числом n_0, для бита с номером 1 – числом n_1 и так далее. Если размер таблицы больше количества битов входного числа, то это приведет к «размножению» некоторых битов, и такие перестановки называются «расширяющими». Встречаются и «сжимающие» перестановки, в которых для некоторых битов входного числа в итоге не находится места.

2. *Шифр DES* (он же *DEA*). В 1978 г. этот шифр был принят в качестве стандарта шифрования США, обязательного для использования в государственных структурах (в правительстве, в ФБР, в АНБ, в ЦРУ, в вооруженных силах и т.п.). Его характеристики:

- длина блока – 64 бита;
- длина ключа – 56 битов;
- количество раундов Фейстеля – 16.



а) Схема генерации подключей б) Структура алгоритма F

Рис. 3.18. Элементы алгоритма DES

Ключ шифра DES очень короток – всего 56 битов. На каждой из итераций сети Фейстеля из этих 56 битов выбираются 48 согласно алгоритму, изображенному на рис. 3.18,а. Здесь KP – «сжимающая» перестановка. Кроме того, на каждой итерации 28-битовые половины общего ключа циклически сдвигаются на 1 или 2 бита (количество зависит от номера итерации), так что в следующую итерацию попадает видоизмененный ключ.

Структура алгоритма F изображена на рис. 3.18,б. В этом алгоритме основную роль играет «расширяющая» перестановка P, восемь подстановок S_1, $S_2 \ldots S_8$ и «прямая» перестановка Q. Таблицы всех этих P-боксов и S-боксов опубликованы, так что шифр полностью определен.

Несмотря на длительную историю применения и исследования, до сих пор для этого шифра отсутствуют методы «взлома», отличные от полного перебора вариантов ключа. Однако уже к началу 90-х годов средства вычислительной техники позволяли осуществлять такой перебор в приемлемые сроки (до 1 года), а в 1998 была году продемонстрирована возможность «взлома» любых сообщений в течение 3 суток.

Поэтому с начала 1990-х годов и до настоящего времени используется модификация «тройной DES» (она же «3DES», она же «Triple DES», она же TDEA):

$3DES(X):=DES_{K3}(DES^{-1}{}_{K2}(DES_{K3}(X)))$

с 168-битовым ключом $K=K_1K_2K_3$, где X – шифруемое сообщение; $DES_N(.)$ – преобразование по стандарту DES с 56-битовым ключом N; $DES^{-1}{}_N(.)$ – преобразование, обратное преобразованию DES. Легко видеть, что при $K_1=K_2=K_3$ форматы 3DES и DES совпадают. Недостаток 3DES: невысокая, по сравнению с DES, скорость работы.

Другой модификацией является «DESX»:

$DESX(X):=K_1 \oplus DES_{K2}(X \oplus K_3)$,

где $\oplus$ – операция побитового «XOR». Эта модификация применяется, например, в файловой системе NTFS для шифрования данных на диске.

В 2001 г. в национальный стандарт шифрования США был включен шифр AES, который сейчас считается «основным». Шифры семейства DES из стандарта не исключены, но для использования в новых разработках криптографических средств не рекомендованы.

2. *Шифр ГОСТ 28147-89* «*Магма*». Этот шифр разработан в СССР в конце 1970-х годов, а в 1988 г. он открыто опубликован в качестве стандарта шифрования, обязательного для использования в государственных органах и организациях, силовых структурах, банках и т.п. Более того, применение в этих областях «иностранных» шифров не допускается. Так же ГОСТ 28147-89 является межгосударственным стандартом для стран СНГ. Характеристики шифра:

- длина блока – 64 бита;
- длина ключа – 256 битов.
- количество раундов Фейстеля – 32.

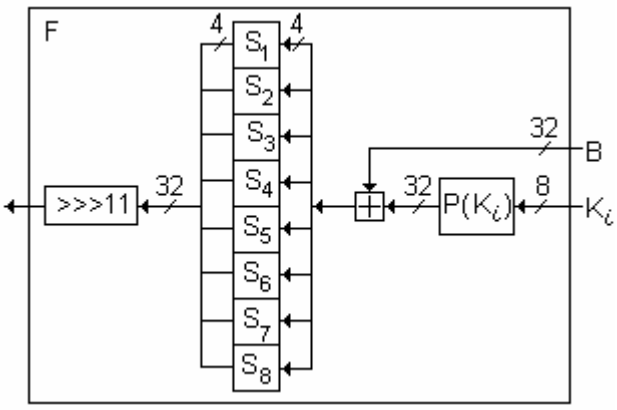


Рис. 3.19. Структура алгоритма F для шифра ГОСТ 28177-89

Структура алгоритма F для этого шифра изображена на рис. 3.19. На вход подается 32-битовая половина шифруемого блока B и 8-битовый подключ K_i. При помощи перестановки P подключ «расширяется» до 32 битов. Затем он складывается по модулю 2^{32} с подблоком B и разделяется на 8 «отрезков» по 4 бита каждый. Эти «отрезки» служат входами для восьми подстановок S_1, S_2… S_8, результаты работы которых объединяются вместе в 32-битовое число и, окончательно, циклически сдвигаются вправо на 11 позиций.

Шифр опубликован в тексте ГОСТ-а не полностью: его перестановки P приведены, а подстановки S_i пропущены, они зависят от назначения шифра. Одни группы подстановок имеют пометку «для служебного пользования» и описаны в документации, прилагаемой к системам шифрования банковской сферы и сетевых коммуникаций. Другие, которые применяются в силовых структурах и в системе правительственной связи, считаются «секретными»[1]. Таким образом, стандарт ГОСТ 28147-89 определяет не один шифр, а целое семейство однотипных шифров, сходных по алгоритму, но различных по числовым параметрам этого алгоритма.

Предусмотрены несколько режимов использования шифра:

- режим простой замены, при котором группы битов сообщения однозначно заменяются другими сочетаниями битов, выбираемыми из таблицы, зависящей от ключа;
- режим гаммирования при помощи последовательно преобразуемого ключа;
- режим гаммирования со сцеплением, при котором ключом для шифрования каждого блока является результат шифрования предыдущего блока.

В 2011 появилась статья, сообщавшая о теоретической «уязвимости» одного из режимов работы шифра ГОСТ 28147-89 с несекретными S-боксами. Согласно ей, имея 2^{64} пар {исходный текст+криптограмма}, возможно определить 8 из 256 битов ключа. Ввиду огромных требований к вычислительным ресурсам, данная атака на практике не реализуема.

В 2015 г. один из вариантов этого шифра с несекретными S-боксами опубликован под названием «Магма» в новом Российском стандарте ГОСТ Р 34.12-2015.

Совместно с ГОСТ 281407-89 используются и другие ГОСТ-ы, стандартизующие некоторые аспекты шифрования, например ГОСТ Р 34.10-94

[1] Различие между понятиями «государственная тайна» и «служебная информация» будут рассмотрены в главе «Деятельность по обеспечению информационной безопасности».

и ГОСТ Р 34.10-2001 (стандарты на алгоритм цифровой подписи), ГОСТ Р 34.11 – 94 (стандарт на функцию хеширования).

3. *Шифр AES* – новый национальный стандарт шифрования США, самый распространенный в мире шифр 2000-х годов. Разработан бельгийскими математиками и выиграл в 2001 г. международный конкурс, объявленный США для замены шифра DES. Алгоритм этого шифра, имеющий историческое наименование «Rijndaell», поддерживает разнообразные длины ключа, но Национальным Агентством Стандартов и Технологий США (NIST) для стандартизации выбраны только некоторые из них.

Параметры шифра:

- размер блока – 128 бит;
- размер ключа – 128 (по умолчанию), 196 или 256 бит.

В дальнейшем рассмотрим принципы работы шифра, характерные для случая 128-битового ключа.

Алгоритм шифра не основан на сети Фейстеля и описывается в виде системы матричных преобразований. Перед началом работы алгоритма 128 битов блока данных помещаются в матрицу размером 4x4 байтов, в такую же матрицу помещается ключ. В дальнейшем эти матрицы объединяются в одну общую матрицу, которая называется «матрицей состояний» и над которой выполняются все преобразования. Алгоритм Rijndael может быть составлен из 4-х операций:

- BS (ByteSub) – подстановка каждого байта матрицы в соответствии с соответствующими «S-боксами»;
- SR (ShiftRow) – циклический сдвиг строк матрицы так, что первая строка остается без изменений, а остальные сдвигаются влево на фиксированное число байт (например, для матрицы 4x4 строки 2, 3 и 4 сдвигаются, соответственно, на 1, 2 и 3 байта);
- MC (MixColumn) – умножение по правилам арифметики GF(256) каждого столбца матрицы состояний на фиксированную матрицу

  2 3 1 1
  1 2 3 1
  1 1 2 3
  3 1 1 2
- AK (AddRoundKey) – сложение по модулю 2 битов матрицы данных с битами матрицы ключа.

В процессе шифрования все эти операции выполняются в соответствии со следующей последовательностью:

AK, {BS, SR, MC, AK} (×R-1 раз), BS, SR, AK.

Здесь R =10 (по умолчанию), 12 или 14 – количество раундов вида {BS, SR, MC,AK}.

Кроме того, на каждом раунде матрица ключа постоянно преобразуется при помощи системы перестановок и циклических сдвигов.

Расшифрование данных производится выполнением тех же операций в обратном порядке.

Шифр AES в варианте с 128-битовым ключом является частью протокола SSL/TLS и по умолчанию используется для шифрования Интернет-трафика. Так же этот шифр (и только он) реализован аппаратно в некоторых процессорах фирмы Intel. На момент написания этих строк никаких уязвимостей в этом шифре не выявлено.

4. *Шифр «Кузнечик»* – новый отечественный метод криптографических преобразований, включенный в ГОСТ Р 34.12-2015 вместе с шифром «Магма».

Параметры шифра:

- размер блока – 128 бит;
- размер ключа – 256 бит.

Алгоритм шифра описывается в виде системы следующих элементарных преобразований:

1) побитовое сложение ключа с блоком данных;

2) S – подстановка, заданная на массиве из 16 байтов блока.

3) L – линейное преобразование байтов блока, напоминающее работу сдвигового регистра с обратной связью (см. рис. 3.20).

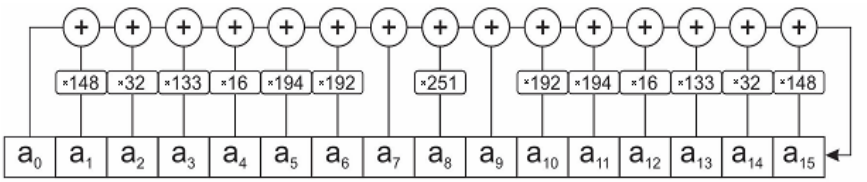


Рис. 3.20. Линейное преобразование шифра «Кузнечик»

Во время этого преобразования байты блока перед сложением умножаются на некоторые числовые коэффициенты, причем все эти операции выполняются по правилам арифметики GF(256) с неприводимым многочленом $x^8 + x^7 + x^6 + x + 1$. Шифрующие преобразования выполняются в процессе 9 одинаковых итераций цикла (см. рис. 3.21) и еще одного дополнительного сложения с итоговым ключом. На каждой из итераций ключ тоже преобразуется путем применения 8-раундовой сети Фейстеля с номером итерации в качестве «подключа» и F=L(S(X)).

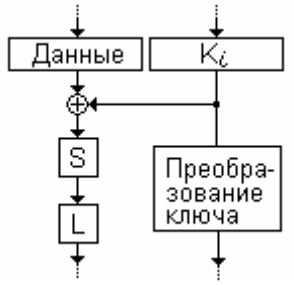


Рис. 3.21. Одна итерация цикла шифра «Кузнечик»

Алгоритм шифра «Кузнечик» легко реализуется как программно, так и аппаратно. Никакие уязвимости для этого шифра пока неизвестны.

5. Прочие шифры. В мире разработано множество блочных шифров. Они обладают высокой криптостойкостью и устойчивы, практически, к любым атакам, кроме «brute force». Характеристики некоторых шифров, нашедших практическое применение в различных системах защиты, приведены в табл. 3.3.

Таблица 3.3. – Некоторые симметричные блочные шифры

Имя	Блок	Ключ	Примечание
DES	64	56	Старый национальный стандарт США
3DES	64	3×56=168	Тройной DES: $Y=DES_{K1}(DES_{K2}(DES_{K3}(X)))$
DESX	64	3×56=168	DES + XOR: $Y=K1 \oplus DES_{K2}(X \oplus K3)$
AES	128, 192, 256	128, 192, 256	Новый национальный стандарт США
Skipjack	64	80	Встроен в чипы Fotrezza и Clipper, США
Магма	64	256	ГОСТ 28147-89 и ГОСТ Р 34.12-2015
Кузнечик	128	256	ГОСТ Р 34.12-2015
IDEA	128	128	С 2012 г. не запатентован
CAST	64, 128	128, 256	Свободен для использования
Blowfish	128	64÷448	Опубликован в книге Б. Шнайера [11]
TEA	64	128	Не запатентован, очень компактен
Калина	128,196,256,512	128,196,256,512	Стандарт ДСТУ 7624:2014, Украина
Belt	128	256	Стандарт СТБ 34.101.31-2011, Беларусь
Camellia	128	128,196,256	Разработан и используется в Японии
SM4	128	128	Стандарт КНР для беспроводных сетей
Serpent	128	128, 192, 256	Финалист конкурса «AES»

3.5. Принципы асимметричной криптографии

Все рассмотренные ранее шифры относились к классу симметричных криптопреобразований, которые использовали один и тот же общий ключ K как для зашифрования, так и для расшифрования. У такого подхода имеются два крупных недостатка.

Первый недостаток – сложность передачи ключа между участниками информационного обмена. Этот ключ нельзя передавать по тому же каналу связи, по которому происходит обмен информацией. Необходим какой-то другой, защищенный канал, предназначенный только для передачи ключа (см. рис. 3.22).

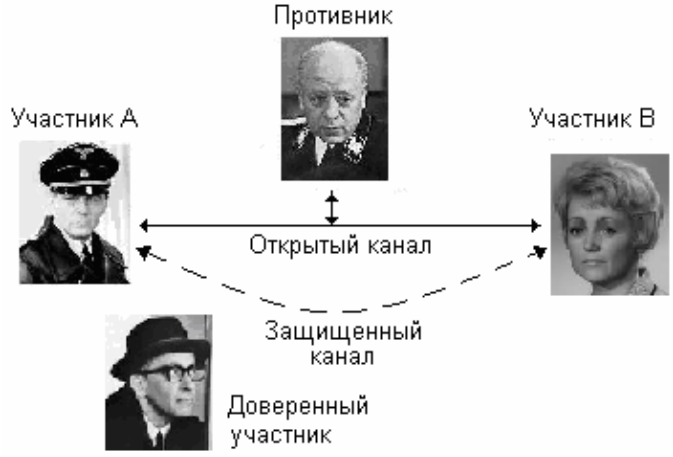


Рис. 3.22. Общая модель информационного обмена

Второй недостаток – невозможность ни доказательства, ни опровержения авторства сообщения, которое зашифровано общим ключом K.

Все эти проблемы могут быть решены при помощи методов асимметричной криптографии.

Основной принцип асимметричной криптографии заключается в том, что для шифрования используется ключ K_E, а для расшифрования ключ K_D, причем $K_E \neq K_D$. Фактически, они представляют собой «половинки» одного общего ключа (K_E,K_D), который первоначально генерируется как одно целое. Но зная

одну половину, например K_E, практически невозможно определить другую – K_D, и наоборот[1].

На основании принципа «асимметрии ключей» возможны две основных криптографических технологии.

1. *Шифрование с открытым ключом*. Пусть Участник А сгенерировал ключевую пару (K_E,K_D). Ключом K_E можно только шифровать сообщение и нельзя расшифровывать. Наоборот, ключом K_D можно только расшифровывать, но нельзя шифровать[2].

Теперь Участник А может отослать Участнику В ключ K_E по открытому каналу, не обращая внимания на возможность перехвата ключа Противником. Такой ключ называется «*открытым*» или «*публичным*». Имея этот ключ, Участник В может зашифровывать свои секретные сообщения и отсылать их Участнику А по тому же открытому каналу. Даже владея ранее перехваченным ключом K_E, Противник не сможет расшифровать эти сообщения, т.к. для этого необходим ключ K_D, которым владеет только Участник А. Такой ключ называется «*закрытым*», «*секретным*» или «*приватным*».

Разумеется, чтобы пересылать секретные сообщения в обратном направлении, Участнику В необходимо сгенерировать какую-то другую пару ключей и отослать участнику А другой зашифровывающий ключ.

Следует упомянуть, что шифрование и расшифрование с открытым ключом, по сравнению с методами симметричной криптографии, – очень медленные операции, поэтому на практике данные сначала шифруются «быстрым» симметричным алгоритмом (например, RC4, DES или AES) с одноразовым ключом К. Ключ К в данном случае это – «*сеансовый ключ*», обычно он генерируется как случайное число подходящей длины. Лишь потом этот, относительно короткий ключ К шифруется открытым «*мастер-ключом*» K_E и передается в таком виде вместе с зашифрованным сообщением (см. рис. 3.23).

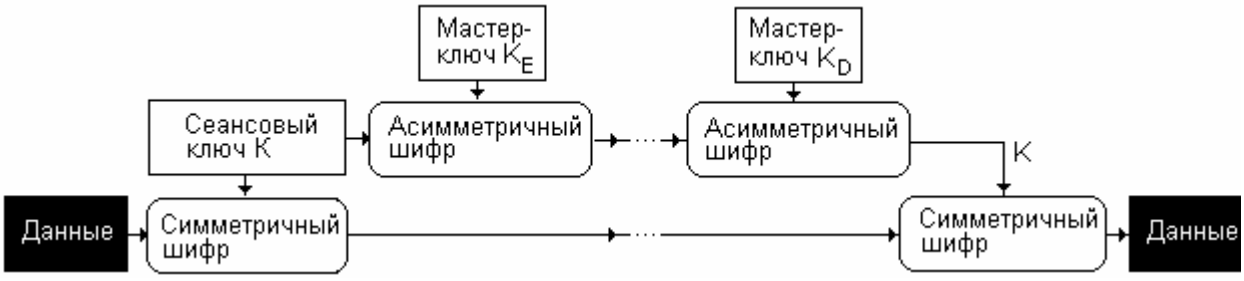


Рис. 3.23. Схема двухэтапного шифрования-расшифрования

Расшифрование данных так же происходит в два этапа: сначала «сеансовый ключ» К расшифровывается «мастер-ключом» K_D, а потом само сообщение расшифровывается «сеансовым ключом» К. Так обеспечивается приемлемая производительность методов шифрования с открытым ключом.

---

[1] Только полным перебором.

[2] Выбор «роли» произволен. Если в качестве K_E выбирается любая из «половинок», другая автоматически становится K_D.

Пример применения шифрования с открытым ключом. Почтовые клиенты семейства TheBat позволяют производить шифрование/расшифрование электронных писем по описанной схеме. Для шифрования данных «сеансовым» ключом используется блочный шифр IDEA, а для шифрования сеансового ключа «мастер-ключом» – шифр RSA (будет рассмотрен позже).

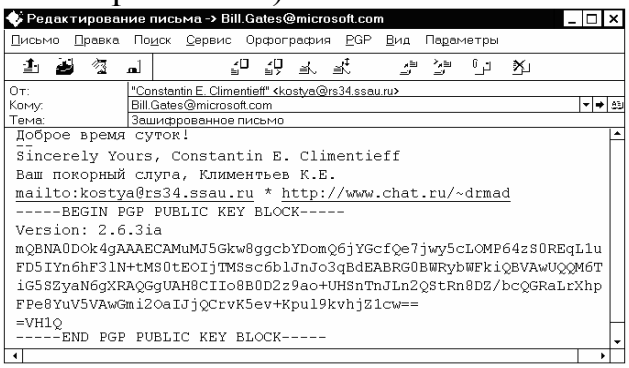


Рис. 3.24. Шифрование с открытым ключом в почтовом клиенте «TheBat»

TheBat поддерживает работу с «мастер-ключами» K_E и K_D, сгенерированными бесплатной программой PGP (будет рассмотрена позже). При этом закрытый ключ K_D сохраняется в служебной базе данных программы TheBat, а открытый ключ можно просто прикреплять к тексту своего электронного письма (см. рис. 3.24).

2. *Электронная цифровая подпись* (*ЭЦП*). Пусть Участник В сгенерировал ключевую пару (K_E,K_D). Ключом K_E можно только шифровать сообщение и нельзя расшифровывать. Наоборот, ключом K_D можно только расшифровывать, но нельзя шифровать.

Теперь Участник В может отослать Участнику А расшифровывающий ключ K_D по открытому каналу. Если Участник В хочет передать подлинную информацию Участнику А, то он шифрует ее ключом K_E. Противник же, имея только ключ K_D, не имея ключа K_E, может перехватить и расшифровать сообщение, но не может его подделать.

Участник А, получив сообщение, пытается расшифровать его при помощи имеющегося K_D и, если расшифровка происходит удачно, то делает вывод, что сообщение подлинное и послать сообщение мог только человек, обладающий ключом K_E, т.е. Участник В. Таким образом, данные, зашифрованные при помощи ключа K_E, представляют собой и само сообщение, и «*электронно-цифровую подпись*» (ЭЦП) автора этого сообщения. Электронный документ, который содержит открытый расшифровывающий ключ K_D, применяемый для проверки ЭЦП, называется *цифровым сертификатом*.

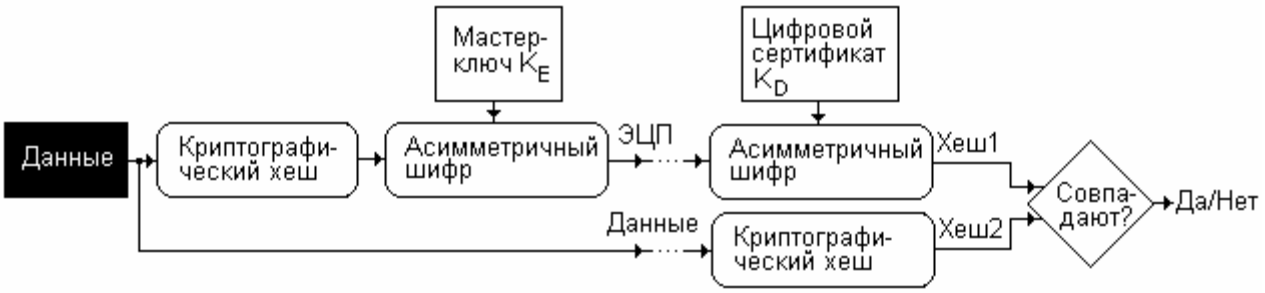


Рис. 3.25. Схема генерации ЭЦП и ее проверки

Следует упомянуть, что шифрование и расшифрование с асимметричным ключом – очень медленные операции, поэтому на практике подписывается не

само длинное сообщение, а хеш-функция от него, например, MD5. Считается, что это незначительно снижает достоверность проверки ЭЦП. Участник А, получив открытое сообщение вместе с зашифрованным хешем, выполняет следующие действия (см. рис. 3.25):

1) расшифровывает хеш имеющимся ключом K_D;

2) повторно рассчитывает хеш от полученного сообщения;

3) сравнивает два хеша – расшифрованный и рассчитанный – и, если они совпадают, делает вывод, что проверка ЭЦП успешно завершена и сообщение от Участника В подлинное.

3. *Технологии применения ЭЦП.* Технологии, связанные с электронными цифровыми подписями, бурно развиваются и приобретают все большее значение в нашей жизни.

Например, в большинстве стран мира, в том числе и в РФ, ЭЦП под электронным документом имеет такой же юридический статус, что и ручная подпись под бумажным документом. Пользуясь такой подписью, можно дистанционно подавать документы в ВУЗ, заказывать загранпаспорт, записываться к врачу, ставить на учет автомобиль, заключать сделки и пр.

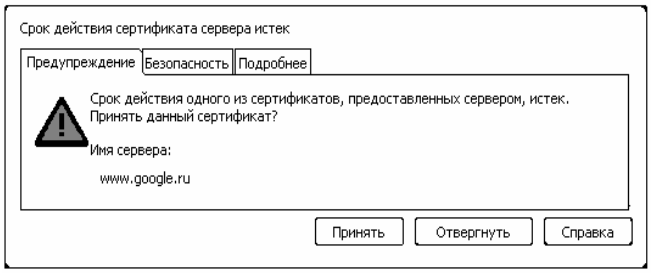


Рис. 3.26. Проверка WWW-клиентом ЭЦП Интернет-ресурса

Так же, ЭЦП очень широко используются в разнообразных областях информационных технологий – прежде всего, для проверки легальности устанавливаемого программного обеспечения и посещаемых Интернет-ресурсов (см. рис. 3.26).

Организационная система, поддерживающая генерацию, выдачу, распределение цифровых сертификатов и проверку ЭЦП, называется «*инфраструктурой открытых ключей*» (*PKI – Public Key Infrastructure*). В мире существует множество PKI международного, государственного и корпоративного уровня.

Существуют международные стандарты, описывающие правила построения и использования программных и аппаратных средств поддержки PKI, например, X.509.

В стандарте X.509 описан, в частности, универсальный формат цифрового сертификата, который должен содержать[1]:

- ключ K_D;
- серийный номер сертификата;
- имя владельца;
- период действия сертификата;
- идентификатор алгоритма, использованного для подписи;
- идентификатор организации, выдавшей сертификат («эмитента»);

[1] Цифровые сертификаты, создаваемые программой PGP, не удовлетворяют этому стандарту и не считаются «квалифицированными».

- ЭЦП, подтверждающая полномочия этой организации.

Так же определены классы цифровых сертификатов, различающихся по назначению:

- Class 1 — индивидуальные, для идентификации электронной почты;
- Class 2 — для организаций;
- Class 3 — для серверов и программного обеспечения;
- Class 4 — для онлайн-бизнеса и транзакций между компаниями;
- Class 5 — для частных компаний или правительственных служб.

В мире наиболее широко распространены алгоритмы цифровой подписи, определенные спецификациями PKCS (Public Key Criptography Standarts – Стандарты криптографии с открытым ключом). В РФ, в основном, используются алгоритмы, описанные в «старых» стандартах ГОСТ Р 34.10-94, ГОСТ Р 34.10-2001 и в «новом» стандарте ГОСТ Р 34.10-2012. Однако для подписания некоторых документов требуются ЭЦП как в формате ГОСТ, так и в формате PKCS.

ЭЦП, официально «привязанная» к физическому лицу или организации, называется «*квалифицированной*». Любое юридическое и физическое лицо может получить собственный цифровой сертификат для такой подписи в специальной организации, имеющей статус «*удостоверяющего центра*» (или «*центра сертификации*»). В роли «удостоверяющего центра» может выступать лишь организация, официально наделенная такими полномочиями, что должно быть подтверждено, в свою очередь, электронной подписью от «удостоверяющего центра» более высокого уровня. «Удостоверяющие центры» образуют древовидную структуру, на самом верху этой иерархии находятся «*корневые удостоверяющие центры*» (например, GeoTrust, Thawte, Verisign и прочие). Цифровые сертификаты как конечных пользователей, так и «удостоверяющих центров» всех уровней располагаются в общедоступных (например, через сеть) архивах сертификатов[1]. Таким образом, подтверждение ЭЦП сводится к проверке непрерывности и завершенности «цепочки» сертификатов, ведущей к тому или иному «корневому удостоверяющему центру»[2].

4. *Криптосистема RSA.* Асимметричный метод шифрования, связанный с разработками американских математиков R.Rivest, A. Shamir и L.Adleman, основан на сложности задачи разложения целых чисел на простые сомножители[3]. Конкретно, если известно составное число n, то найти такие простые p и q, что $n=p\times q$, можно только методами частичного перебора вариантов. Эти методы работают существенно быстрей, чем полный перебор вариантов, но они недостаточно эффективны для того, чтобы раскладывать

[1] Иногда корневой сертификат может быть встроен в само средство проверки ЭЦП, например, сертификаты от Microsoft встроены в операционную систему Windows.

[2] Ввиду возможности коллизий в хеш-функциях, слишком «длинные» цепочки сертификатов могут приводить к существенному снижению достоверности ЭЦП.

[3] Разложение числа на простые сомножители так же называется его «факторизацией».

числа любой длины за приемлемое время. Например, разложить 130-битовое число

```
114381625757888867669235779976146612010218296721242362562 5
618429357069352457338978305971235639587050589890751475992 9
0026879543541
```

на сомножители

```
349052951084765094914784961990 38
981334177646384933878439908205 77
```

и

```
327691329932667095499619881908 34
461413177642967992942539798288 533
```

не удавалось в течение 14 лет, с 1979 по 1993 гг. За прошедшие десятилетия быстродействие компьютеров выросло примерно в 100 раз, к услугам «противника» суперкомпьютеры с миллионами вычислительных ядер и «ботнеты» с миллионами зараженных компьютеров. Но значительного прогресса в области математики пока нет. С момента постановки задачи и до настоящего времени действует эмпирическое правило: факторизация числа из N бит требует приблизительно $2^{N/12}$ попыток. На момент написания этих строк предельные возможности потенциального «злоумышленника» составляют примерно 1000 бит, разложение на сомножители более длинных чисел пока невозможно.

Алгоритмы шифрования и расшифрования методом RSA заключаются в следующем.

Шаг 1. Участник А выбирает два больших простых числа p и q, рассчитывается значения $n=p\times q$ и $m=(p-1)\times(q-1)$.

Шаг 2. Участником А выбираются два числа E и D таких, что $(E\times D \bmod m)=1$. А конкретней, одно из этих чисел выбирается любым, а другое вычисляется при помощи алгоритма Евклида.

Шаг 3. Участник А передает по открытому каналу пару чисел $K_E=(E, n)$, которые представляют собой открытый «шифровальный» ключ. Отметим, что зная n, можно найти $m=(p-1)\times(q-1)$, только разложив n на сомножители, вычтя из них 1 и вновь перемножив. Таким образом, при достаточно больших n это число можно передавать по открытому каналу связи, причем практически невозможно определить при его помощи p, q, и m, а значит, и число D.

Шаг 4. Участник Б зашифровывает сообщение X при помощи открытого ключа (E,n) и соотношения

$$Y = X^E \bmod n$$

и передает Y по открытому каналу Участнику А.

Шаг 5. Участник А при помощи закрытого ключа (D,n) выполняет обратное расшифрование:

$$X = Y^D \bmod n .$$

Схема электронной цифровой подписи с использованием того же математического принципа строится аналогично.

Можно доказать, что взаимно-обратное соответствие соотношений $Y = X^E \bmod n$ и $X = Y^D \bmod n$ выполняются всегда. Пусть $Y=X^E \bmod p\times q$, составим и преобразуем выражение для расшифрования Y:

$Y^D \bmod p\times q = (X^E \bmod p\times q)^D \bmod p\times q =$

$= X^{ED} \bmod p\times q =$

$= X^{k\cdot(p-1)\cdot(q-1)+1} \bmod p\times q =$

$= X\cdot X^{k\cdot(p-1)(q-1)} \bmod p\times q =$ (по малой теореме Ферма)[1] =

$= X\cdot 1=X$ – что и требовалось доказать.

На протяжении последних десятилетий RSA – одна из наиболее известных и часто применяемых асимметричных криптосистем мира.

Однако распространению RSA мешают крупные недостатки:

- очень невысокая скорость шифрования и расшифрования;
- отсутствие достоверных сведений о существовании или отсутствии эффективных алгоритмов факторизации целых чисел;
- необходимость, вместе с прогрессом в области вычислительной техники, постоянного увеличения длины модуля n[2].

5. *Криптосистемы на основе дискретного логарифмирования*. Пусть G, n и m – целые числа. Вычислить $y=G^n \bmod m$ довольно просто последовательным умножением G самого на себя n раз. Но выполнить обратную операцию, то есть определить степень n, имея в своем распоряжении y и m, – очень трудная с вычислительной точки зрения задача. Справедливо правило: логарифмирование числа y, состоящего из N бит, требует приблизительно $2^{N/12}$ попыток перебора.

Это обстоятельство позволяет решить основную проблему, присущую симметричным шифрам, – трудность распространения шифровальных ключей и необходимость, поэтому, использования дополнительного, защищенного канала обмена информацией. Теперь эта проблема решается с использованием алгоритма, разработанного американскими математиками Диффи и Хеллманом.

Пусть Участник B и Участник A собираются обмениваться зашифрованной симметричным шифром информацией по доступному для Противника каналу связи, но не имеют общего ключа K. Необходимо, передавая по открытому каналу связи несекретную информацию, выработать общий секретный ключ K.

Решение сводится к следующему алгоритму.

Шаг 1. Участник B выбирает случайное число X_1, а Участник A выбирает случайное число X_2.

Шаг 2. Затем Участник B вычисляет $Y_1=G^{X1} \bmod m$, а Участник A $Y_2=G^{X2} \bmod m$, и оба они обмениваются значениями Y_1 и Y_2 по открытому каналу. Определить X_1 и X_2 по Y_1 и Y_2 Противник не может, так как это превышает его вычислительные возможности.

---

[1] Если A – целое число и M – простое число, то $A^M \equiv A \pmod M$ или, что тоже самое: $A^{M-1} \equiv 1 \pmod M$.

[2] На момент написания этих строк стойкими считаются ключи длиной не менее, чем 1024 бита.

Шаг 3. Получив Y_2, Участник В вычисляет $K_1=Y_2^{X1} \bmod m = (G^{X2} \text{ по mod } m)^{X1} \bmod m=(G^{X2\cdot X1} \bmod m)$. А Участник А, в свою очередь, вычисляет $K_2=Y_1^{X2} \bmod m = (G^{X1} \bmod m)^{X2} \bmod m=(G^{X2\cdot X1} \bmod m)$.

Шаг 4. $K_1=K_2=K$ – общий секретный ключ, выработанный совместно Участником А и Участником В в результате обмена по открытому каналу несекретной информацией.

Еще сложней задача определения аргумента экспоненты в том случае, если все операции – и возведение в степень, и логарифмирование – выполняются не на множестве целых чисел, а на множестве точек, принадлежащих так называемым «*эллиптическим кривым*», то есть кривым уравнений вида $y^2 = x^3 + ax + b$. В этом случае логарифмирование числа y, состоящего из N бит, требует приблизительно $2^{N/2}$ попыток перебора.

Таблица 3.4 – Применение различных методов в стандартах на ЭЦП

Метод	**Стандарт**
Факторизация целых чисел	PKCS#1
Дискретное логарифмирование	ГОСТ Р 34.10-94 (РФ), FIPS-186 DSA (США)
Логарифмирование на эллиптических кривых	ГОСТ Р 34.10-2001 (РФ), ГОСТ Р 34.10-2012, ANSI ECDSA.

В рамках задач логарифмирования как для множества целых чисел, так и для множества точек «эллиптических кривых» возможно построение шифров с открытым ключом и систем ЭЦП. Все они находят широкое применение на практике (см. табл. 3.4).

3.6. Практическое применение криптографии

Криптографические методы и средства повсеместно используются в современных информационных системах. Основное их назначение – обеспечение информационной безопасности как всего государства в целом, так и отдельных ведомств, предприятий, организаций, а так же сохранение конфиденциальности личной информации.

1. *Законодательные ограничения*. Однако закрытость личной информации при определенных условиях может наносить ущерб физической безопасности как всего государства, так и отдельных граждан. Поэтому государственные органы многих стран стремятся законодательно ограничивать использование криптографии негосударственными организациями и частными лицами.

Например, в соответствии с постановлением правительства №113 от 16.04.2012 в РФ любая деятельность, связанная с разработкой, производством, распространением, обслуживанием и т.п. технических и программных средств, обладающих криптографическими возможностями, требует наличия специальной лицензии. Правда, это требование не распространяется на средства, которые изготовлены другим производителем, а криптографические возможности встроены в них и не могут быть изменены, – например, на операционные системы. Кроме того, ограничения не касаются средств,

использующих заведомо «нестойкую» криптографию – например, симметричных шифров с ключами короче 56 бит и асимметричных с ключами короче 512 бит[1].

Другой пример: в США действуют ограничения на экспорт за пределы страны любых средств с криптостойкими шифрами, поэтому долгое время все «национальные» версии продуктов Microsoft (например, «русские» и «китайские» MS Windows, MS Office и т.п.) использовали шифрование с ключом длиной всего 40 бит.

2. *Свободно распространяемые шифровальные средства*. Попытками преодолеть эти ограничения являются бесплатные, свободно распространяемые программные продукты семейства «PGP» (Pretty Good Privacy (англ.) – достаточно хорошая приватность). Первые версии этой программы предназначались для выполнения в MS-DOS, современные работают под MS Windows, Linux и Mac OS X. Спецификации OpenPGP на форматы данных, протоколы аутентификации и т.п. не совпадают с международными стандартами ANSI/IEC/IEEE и пр., но образуют целостную систему и достаточно широко распространены. Например, ключевые пары для шифра RSA, генерируемые программой PGP, поддерживаются почтовым клиентом TheBAT (см. рис. 3.24). Так же сама программа PGP поддерживает шифрование файлов пользователя с использованием шифра RSA, а компонент PGP Disk обеспечивает создание «виртуального» диска, содержимое которого шифруется и расшифровывается «на лету» незаметно для пользователя.

Существуют бесплатные аналоги компонента PGP Disk, например, TrueCrypt для Windows, Linux и MacOS X (поддержка прекращена разработчиками в 2014 г.), VeraCrypt и т.п.

3. *Криптографические средства в операционных системах*. Для защиты данных пользователей используются так же криптографические средства, которые не являются бесплатными, но могут свободно применяться, поскольку «встроены» в операционные системы.

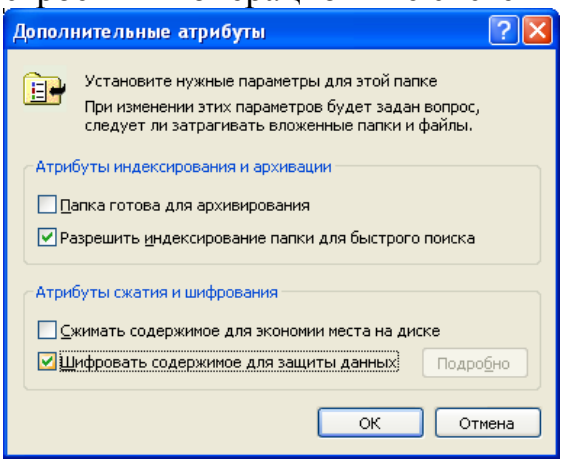


Рис. 3.27. Управление шифрованием файлов и папок

Например, в MS Windows, начиная с версии Windows 2000, присутствует подсистема EFS (Encrypting File System – Шифрующая файловая система), которая позволяет «прозрачно» шифровать файлы и папки пользователя на диске, размеченном по правилам NTFS. При этом используются шифры DESX, 3DES и AES, а ключи длиной до 128 бит генерируются из пароля, вводимого при входе в систему. Таким образом, данные на таком диске доступны только для конкретного пользователя (см. рис. 3.27).

[1] До 2012 г.: 40 бит и 128 бит.

Начиная с Windows Vista, в состав операционной системы включена подсистема BitLocker, позволяющая шифровать данные как на жестком диске, так и на съемных носителях, причем используемый для этой цели ключ может генерироваться пользователем самостоятельно и располагаться отдельно – например, в файле.

4. *Интерфейс CryptoAPI*. Пользоваться встроенными криптографическими возможностями Windows могут не только системные программы от Microsoft, но и программы пользователя. Этой цели служит CryptoAPI – программный интерфейс между прикладными программами и библиотеками, реализующими криптографические алгоритмы[1]. Такие библиотеки называются «*криптопровайдерами*». Например, криптопровайдеры, встроенные в Windows, реализуют основные криптографические алгоритмы – шифры RC2, RC4, DES, 3DES, RSA и пр., хеш-функции MD2, MD5, RIPEMD и т.п., а так же обеспечивают работу с цифровыми сертификатами. Интерфейс CryptoAPI позволяет простое добавление нестандартных криптопровайдеров. Например, это могут быть библиотеки, разработанные отечественной компанией «КриптоПро», которые либо самостоятельно реализуют шифр ГОСТ 281407-89 «Магма», либо служат драйверами к шифровальным платам семейства «Криптон» (см. рис. 2.12,в).

Для разработки своих приложений в среде UNIX-подобных операционных систем доступна библиотека OpenSSL, содержащая десятки популярных криптографических алгоритмов и распространяемая в виде исходных текстов в соответствие с принципом Open Source.

ТЕМА 4. Методы и средства аутентификации

Ранее были рассмотрены политики безопасности, лежащие в основе технологий разграничения доступа. Проверка прав доступа субъекта к объекту реализуется при помощи трех тесно связанных механизмов (см. рис. 4.1):

- идентификации;
- аутентификации;
- авторизации.

Идентификация объекта или субъекта – распознавание его как уникальной сущности. Она обычно выполняется при помощи *идентификатора*, то есть ключа для поиска учетной записи о конкретном субъекте или объекте в базе данных. Если субъектом является человек, то в качестве идентификатора может выступать фамилия-имя-отчество, табельный номер, ИНН, login, username, userID и т.п. Идентификатором программных субъектов и объектов может служить, например, числовой SID (см. раздел «Разграничение доступа в MS Windows»).

[1] Присутствующие в .NET Framework криптографические классы являются «обертками» над CryptoAPI.

Аутентификация объекта или субъекта – проверка соответствия субъекта или объекта его идентификатору. Различают три основных принципа аутентификации:

- знать какую-нибудь секретную информацию (например, пароль);
- владеть каким-нибудь уникальным предметом (например, ключом);
- быть кем-нибудь (отличаясь уникальным набором «биометрических» характеристик).

Конкретный аутентификатор, участвующий в процедуре, называется «*фактором аутентификации*». Если используется только один фактор, то аутентификация называется «*однофакторной*» и считается не слишком надежной. Общепринятой практикой является применение методов «*многофакторной*» аутентификации, например, когда банкомат проверяет не только владение банковской карточкой, но и знание пин-кода.

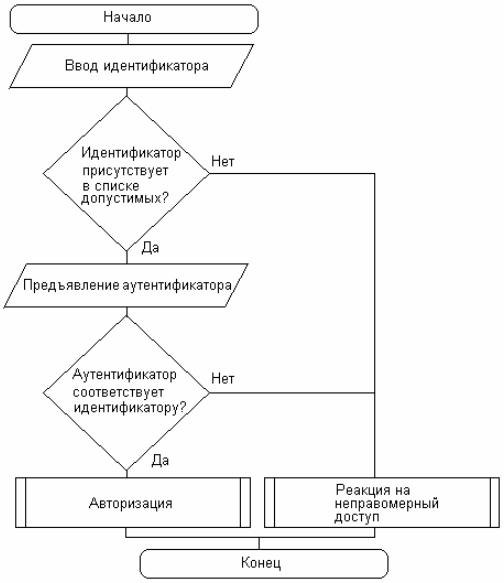


Рис. 4.1. Алгоритм проверки подлинности

Авторизация – действия по разрешению доступа субъекта к объекту, например, расшифрование данных, которые хранились в зашифрованном виде и поэтому не могли быть использованы. Авторизация выполняется только в том случае, если этапы идентификации и аутентификации успешно завершены.

Ключевым элементом алгоритма проверки прав доступа субъекта к объекту (см. рис. 4.1) является процедура аутентификации.

4.1. Парольная защита

Наиболее популярным методом аутентификации является парольная защита. Пароли предназначены для аутентификации субъекта-человека и поэтому почти всегда представляют собой алфавитно-цифровые строки, предназначенные для клавиатурного ввода. Примеры:

- пароль, записанный в CMOS-память и запрашиваемый у пользователя BIOS-программой POST при запуске компьютера;
- пароль, запрашиваемый у пользователя в начале работы с операционными системами семейств Windows и UNIX;
- пароль, который почтовая программа (например, OutLook или TheBAT) в соответствии с протоколами «POP3» и «SMTP» посылает серверу электронной почты для доступа к почтовому ящику;
- пароль, которым защищены данные в архивах формата «ZIP» или «RAR»;
- числовой пин-код (от англ. Personal Identification Number – персональный идентификационный номер), который пользователь вводит при работе с банковской картой.

Чаще всего пароль служит «пропуском» к простейшим процедурам авторизации, таким, например, как установка логического флага «доступ разрешен» в положение «истина». Однако возможны и более сложные процедуры: например, строковые пароли, вводимые при входе в операционную систему, могут служить так же для генерации числовых идентификаторов UID (англ. User IDentifier – идентификатор пользователя) и SID (англ. Security IDentiefir – идентификатор безопасности). Другой пример: из строкового пароля, вводимого пользователем при упаковке и распаковки данных в архивах типа «ZIP» или «RAR», генерируется длинный числовой ключ, который используется для зашифрования и расшифрования этих данных.

Парольная защита – очень простой, но довольно уязвимый механизм. Рассмотрим некоторые методы атак и способы противодействия им.

4.1.1. Похищение пароля

Наиболее простой и часто используемой злоумышленниками уязвимостью парольной защиты является похищение пароля. Эта уязвимость может проявляться на всех этапах аутентификации.

Наиболее просто совершить такую кражу, если пользователь хранит пароль в открытом виде – например, записав на бумажку и приклеив ее на системный блок компьютера.

Для похищения пароля, вводимого пользователем, часто используются «*клавиатурные шпионы*» (они же «*кейлоггеры*») – вредоносные программы, способные перехватывать все нажатия клавиш и отсылать их по сети злоумышленнику (см. раздел «Защита программ и данных»).

Так же злоумышленник может узнать пароль, получив в свое распоряжение базу данных, в которой хранятся пароли. Известно, например, что в операционных системах семейства Windows пароли хранятся в файле C:\Windows\System32\Config\SAM, а в UNIX-подобных операционных системах для этой цели служат файлы /etc/passwd и /etc/shadow. Чаще всего такое похищение выполняется при помощи вредоносных программ-шпионов.

Для похищения пароля, передаваемого по сети, злоумышленник может с успехом использовать «*сниффер*» – программу, способную перехватывать и анализировать сетевой трафик.

Наиболее общий принцип противодействия похищению пароля: хранение, передача и сравнение его с эталоном не в открытом виде, а в зашифрованном или хешированном виде.

При этом хеширование пароля более предпочтительно, чем шифрование, так как вместе с зашифрованным паролем может быть похищен и ключ, после чего восстановление исходного пароля становится тривиальным. Восстановить же пароль по похищенному значению хеш-функции – очень сложная с вычислительной точки зрения задача.

Основным недостатком хеширования пароля является возможность коллизий, что облегчает подбор пароля. Например, если для хеширования используется CRC-32, то пароль «pupkin» с хешем 0xBA5ADB31 имеет многочисленные «аналоги», например: «uyqvudT», «oBU9XdE», «malecemu»,

«3heispi» и др., имеющие тот же хеш. Злоумышленнику не обязательно подбирать оригинальный пароль, достаточно определить любой из его «аналогов». Поэтому на практике для хеширования паролей используются криптографические хеши MD5, SHA-1, RIPEMD и др., обеспечивающие очень маленькую вероятность коллизий.

Существуют стандарты на методы хеширования паролей. Например, в MS Windows разных версий применяются «LM-хеширование» и «NT-хеширование».

Еще одним широко распространенным стандартом является PBKDF2, который предназначен, в первую очередь, для «изготовления» числового шифровального ключа из текстовой строки. Однако он может быть использован и для генерации парольных хешей.

В PBKDF2 первоначально рекомендовалось 1000-кратное последовательное хеширование строки пароля P: H(H(H(…H(P))) и выделение из полученного результата группы битов требуемой длины.

Например, пусть результат первого применения функции CRC-32[1] от строки «pupkin» есть число 0xBA5ADB31, тогда повторное хеширование дает 0x83EF7545 и так далее – до тысячного, которое приводит к числу 0xB0EB7C31. Если требуется числовой ключ определенной длины, например, всего 10 битов, то берутся последние биты получившегося числа: 0000110001.

Большое количество последовательных применений функции хеширования служит как для затруднения «обратного вычисления» строки пароля по числовому ключу, так и для сильного замедления процедуры генерации числового ключа (необходимость этого будет обсуждена ниже). Однако в современных условиях рекомендуется увеличивать количество итераций минимум на порядок, то есть до 10000 раз.

4.1.2. Подбор пароля методом «грубой силы»

Атака методом «грубой силы» (brute force) – наиболее очевидный метод взлома парольной защиты. Эта атака может выполняться по разному.

1. Вручную, например, при попытке доступа к «запароленному» компьютеру или чужой банковской карте, защищенной пин-кодом. Скорость «ручного» перебора довольно мала, на опробование одного пароля требуется несколько секунд.

2. Удаленно по сети, например, при попытке получить учетные данные для доступа к чужому сетевому ресурсу – папке, почтовому ящику, страничке в социальных сетях, удаленному рабочему столу и т.п. Скорость перебора в этом случае ограничена пропускной способностью сети и производительностью сервера, в типичном случае она не превышает несколько миллионов паролей в секунду.

[1] В PBKDF2 рекомендованы «криптографические» хеши MD5, SHA-1, SHA-2 и пр.

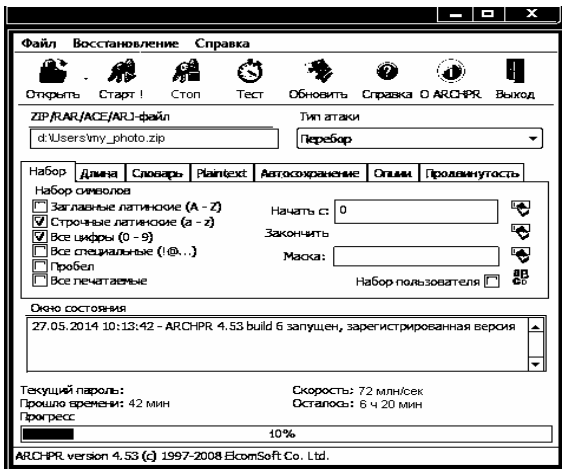


Рис. 4.2. Подбор пароля для ZIP-архива

3. «Локально», например, при попытке взломать «запароленный» архив форматов «ZIP» или «RAR» при помощи утилит типа ArchPR отечественной фирмы «ЭлкомСофт» (см. рис. 4.2). В этом случае скорость перебора напрямую зависит от производительности используемого компьютера и может составлять, например, 100 млн. паролей в секунду.

Кроме того, такие попытки взлома легко распараллеливаются: например, один компьютер пробует только пароли, начинающиеся с «A»; другой компьютер – только с «B» и т.д.

Подбор пароля заключается в генерации и опробовании всевозможных комбинаций из всевозможных знаков некоторого «алфавита». Возможны разные варианты «алфавитов».

1. Простейший «алфавит», используемый для «пин-кодов», состоит из 10 цифр.

2. Наиболее распространен «алфавит», включающий 26 букв латинского алфавита (без различения прописных и строчных) и 10 цифр, то есть, всего 36 знаков.

3. Возможен так же «алфавит», состоящий из всевозможных знаков, которые можно ввести при помощи компьютерной клавиатуры, – он включает 10 цифр, 52 прописные и строчные буквы, а так же 34 знаков пунктуации и специальных знаков, таких как «амперсанд» &, «собака» @, «диез» #, «знак денежной единицы» $ и т.п. – всего 96 знаков.

Пусть пароль имеет длину N знаков и составлен из M-значного алфавита. Тогда при N=1 для подбора в худшем случае потребуется M попыток, при N=2 – M×M попыток и т.д., то есть, в общем случае, потребуется M^N попыток. Приблизительные оценки «взломостойкости» паролей приведены в табл. 4.1.

Табл. 4.1. Взломостойкость паролей

	Алфавит «26 букв+10 цифр»				**Алфавит «96 знаков клавиатуры»**			
	6	**8**	**10**	**12**	**6**	**8**	**10**	**12**
По сети	36 мин	1 мес	120 лет	1.5×10^5 лет	9 дней	230 лет	2×10^6 лет	2×10^{10} лет
Компьютер	22 сек	8 час	14 мес	1500 лет	2 часов	2 года	2100 лет	2×10^8 лет
Кластер	<1 сек	<1сек	37 сек	13 часов	< 1 сек	7 сек	8 дней	190 лет

Выводы: «взлом» пароля методом полного перебора очень ресурсоемок. Однако путем распараллеливания этой задачи на множество компьютеров, процессоров или процессорных ядер возможны успешные атаки даже на пароли длиной 10-12 знаков.

Рекомендуются следующие методы противодействия подбору пароля:

- увеличение общей длины пароля (для алфавита «26 букв+10 цифр» – не менее 12 знаков, для алфавита «96 знаков клавиатуры» – не менее 10);
- увеличение размера алфавита хотя бы до 96 знаков;
- увеличение вычислительной сложности процедуры проверки паролей (например, при помощи многократного хеширования), чтобы каждая из них требовала значительных временных затрат;
- ограничение (например, до 3) количества неудачных попыток ввода пароля;
- ограничение срока действия пароля (например, сроком в 1 месяц).

4.1.3. «Словарный» подбор пароля

Большинство пользователей придумывают пароли, представляющие собой имена собственные (например, «Masha») и нарицательные (например, «medved»), а также даты (например, «1990»).

Распространенными методами «мутаций» для этих паролей являются:

- инвертирование (например, «Ahsam»),
- дополнение букв цифрами (например, «Masha1990»);
- замена букв цифрами и специальными знаками (например, «he110w0r1d» или «$vet04k@»).

Все эти способы генерации паролей недостаточно надежны, поскольку существующие словари собственных и нарицательных имен включают не более 100000 слов, а большинство популярных «мутаций» также хорошо известны. Таким образом, быстрый подбор пароля становится возможным даже на «слабом» компьютере.

В Интернете можно найти списки наиболее популярных паролей, составленные по результатам анонимных опросов многих миллионов пользователей. Среди них: «password», «parol», «123», «1234», «123456», «12345678», «111111», «qwerty», «asdfghjkl», «dragon», «pussy», «admin» и т.п. В 1988 г. сетевой червь Морриса, пользуясь словарем из 400 слов и применяя всего 4 «мутации», проник более чем на 6000 «защищенных» паролями узлов сети на территории США, Канады, Великобритании и Австралии.

Основной метод противодействия словарному подбору – использование длинных случайных паролей, например, «Zz0x^UQ#8». В состав операционных систем семейства UNIX входит стандартная утилита, позволяющая генерировать такие пароли. Для других операционных систем ее легко написать самостоятельно.

```
#include <iostream.h>
#include <stdlib.h>
#include <time.h>
#define NSYM 12  // Длина пароля
char s[]=
{"0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz!@#$%^&"};
main() {
 srand(time(0));
 while(1)  { // Бесконечный цикл, прервать по CTRL-C
  for (int i=0;i<NSYM;i++) cout << s[random(strlen(s))]; cout << endl;
 }
}
```

4.1.4. Использование «радужных таблиц»

В настоящее время наиболее эффективным методом взлома паролей являются «радужные таблицы». Так называются базы данных, содержащие предварительно рассчитанные значения однократных и многократных хешей для всевозможных алфавитно-цифровых цепочек ограниченной длины. То есть, подбор пароля при помощи «радужных таблиц» сводится к поиску подходящего хеша в базе данных. Использование современных многокомпьютерных кластеров или многоядерных процессоров для распараллеливания поиска позволяет подбирать пароли длиной 12-14 символов за несколько секунд. С учетом того, что несколько различных алфавитно-цифровых строк могут иметь один и тот же хеш, результатом такого поиска обычно является не сам правильный пароль, а его более короткий «аналог». Например, поскольку строки «qwerty123456» и «ZmU1pB» имеют общую контрольную сумму $13E210B3_{16}$, то в «радужной таблице» будет храниться более короткая строка, и «злоумышленник» во время попытки взлома найдет именно ее. Если система аутентификации проверяет подлинность, сравнивая хеши от паролей, то она окажется взломана, поскольку одинаково отреагирует и на пароль «qwerty123456», и на пароль «ZmU1pB».

Основной метод противодействия «радужным таблицам» – «*подсаливание*» (salting) паролей, т.е. их комбинирование перед расчетом хеша со случайными данными. В результате, радужная таблица позволит подобрать только короткий аналог «соленого» пароля, но не сам исходный пароль.

Конкретный пример. Пусть используется пароль «qwerty». Путем добавления к нему случайной строки «123456» формируется «соленый» пароль «qwerty123456», хеш которого вместе с «солью» будет храниться системой аутентификации в качестве эталона[1].

Пусть теперь злоумышленник взломал сервер и похитил как хеш от пароля, так и «соль». Путем применения «радужных таблиц» он может обнаружить строку «ZmU1pB». Однако удалить из этой строки «соль», равную «123456», и определить правильный пароль он уже не сможет.

4.2. Протоколы удаленной аутентификации

Наиболее сложно предотвратить похищение пароля, передаваемого по сети во время удаленной аутентификации. Как открытый, так и зашифрованный или хешированный пароль могут быть перехвачены и использованы Противником, выдающим себя за легального Пользователя. Информационные атаки такого типа называются «*человек-в-середине*» (англ. *MITM* – Man-In-The-Middle).

Для противодействия таким атакам используются специальные методы.

[1] Очевидно, в настоящее время в качестве «соли» следует выбирать такую строку, чтобы длина «соленого» пароля превышала 12-14 символов.

4.2.1. «Одноразовые» пароли

Первоначально, имея общий пароль P в качестве «затравки», оба участника информационного обмена независимо друг от друга вычисляют списки многократных хешей: H(P), H(H(P)), H(H(H(P))) и т.д.

При первой попытке аутентификации вводится и сравнивается последний элемент из списка. После удачной аутентификации он считается неверным и теряет актуальность. При следующей попытке правильным паролем будет служить уже предпоследний элемент списка, потом пред-предпоследний и т.д.

Пусть Противник перехватит во время ввода некий, i-й по счету элемент списка. Но он не сможет им воспользоваться повторно. Более того, поскольку хеш-функция необратима, злоумышленник не сможет вычислить ни один из предыдущих паролей, которые будут актуальны в следующие разы.

4.2.2. Аутентификация с использованием симметричного шифра

Протокол использует симметричные шифры (например, RC4 или AES):

- узел A и узел B знают общий секретный пароль P, на основании которого может быть вычислен ключ K (например, по стандарту PBKDF2);
- узел A хочет обратиться к узлу B;
- узел A посылает открытый «запрос» на доступ к B;
- узел B, получив «запрос», генерирует случайное число R («оказию»), запоминает его и отсылает узлу A;
- узел A шифрует полученное число R шифром F с ключом K и отсылает ему результат $F_K(R)$;
- узел B пытается расшифровать полученный результат своим ключом K и сравнить его с сохраненным числом R;
- если числа совпадают, то аутентификация считается успешной, и узел B начинает информационный обмен с узлом A.

В конкретных реализациях этого протокола вместе с «оказией» пересылается еще зашифрованная метка текущего времени, анализ которой позволяет отследить «дубликаты» сообщений аутентификации.

Итак, этот протокол обеспечивает аутентификацию, но не предусматривает пересылки пароля ни в открытом, ни в зашифрованном виде. Поскольку R – случайное число, то при каждом сеансе связи по сети передаются разные числовые значения, так что перехватить их Противник может, но повторно воспользоваться – нет.

В протоколе аутентификации CHAP вместо шифрования числа R используется его хеширование с ключом K (см. раздел «Криптографические хеш-функции»).

4.2.3. Аутентификация с использованием асимметричного шифра

Протокол использует асимметричные шифры (например, RSA), применяющие ключ E для шифрования, а ключ D для расшифрования.

Роль монитора безопасности играет «доверенное лицо» (сервер аутентификации), который хранит свой приватный ключ D_c и публичные ключи E_a, E_b… всех узлов. Рядовые узлы имеют только свои приватные ключи и публичный ключ E_c сервера безопасности.

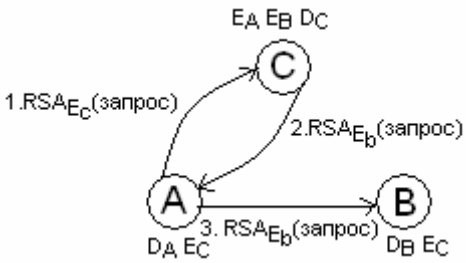


Рис. 4.3. Схема обмена запросами

Протокол (см. рис. 4.3):

- узел A хочет обратиться к узлу B;
- узел A, зная открытый ключ E_c сервера безопасности C, посылает ему «запрос» на доступ к B, зашифрованный этим ключом;
- сервер C расшифровывает «запрос» своим приватным ключом D_c и заново перешифровывает его открытым ключом E_b узла B, после чего возвращает его узлу A;
- узел A пересылает этот зашифрованный «запрос» на узел B;
- узел B успешно расшифровывает «запрос» своим приватным ключом D_b и начинает информационный обмен с узлом A; если бы расшифрование не удалось, «запрос» на информационный обмен был бы отвергнут.

Таким образом, аутентификация считается успешной после удачного расшифрования узлом B «запроса». Подлинность запроса при этом подтверждается сервером безопасности, то есть «доверенным лицом».

В конкретных реализациях этого протокола к «запросу» добавляется и пересылается вместе с ним еще зашифрованная метка текущего времени, анализ которой позволяет отследить «дубликаты» сообщений аутентификации.

4.2.4. Прочие методы аутентификации

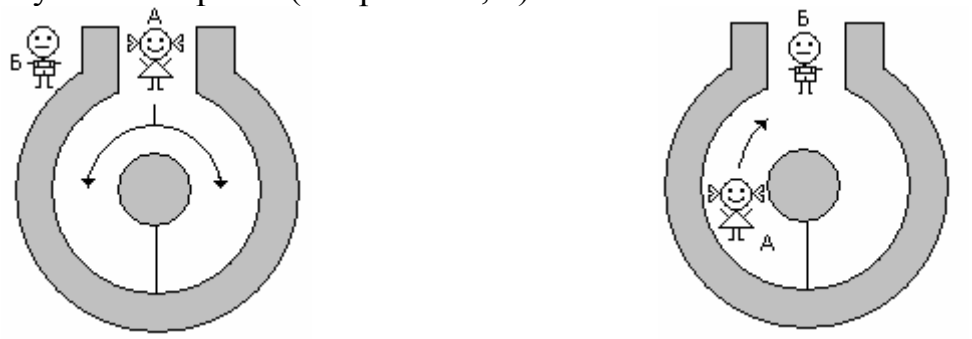
В рассмотренных ранее протоколах предполагалось, что участники информационного обмена владеют некоторым «секретом» (ключом, паролем и т.п.) и основная проблема состоит в демонстрации этого «секрета» таким образом, чтобы он не был перехвачен «злоумышленником». Если же это произошло, то перехваченная информация не должна позволить «злоумышленнику» в дальнейшем выдавать себя за одного из участников. Однако возможна ситуация, когда один из участников изначально является «злоумышленником», и в этом случае демонстрация ему «секрета» приведет к разрушению системы защиты. Для решения подобных проблем применяется специальный класс протоколов аутентификации, основанных на доказательстве знания «секрета», обладающем свойством «*нулевого разглашения*». Это значит, что Участник, проверяющий знание «секрета, в ходе выполнения подобного протокола не получает никакой информации о самом «секрете».

Пример – «пещера Али-Бабы». На далеком острове есть кольцеобразная пещера с одним входом и двумя коридорами – правым и левым, которые в

глубине пещеры разделены при помощи волшебной двери. Участник А знает заклинание, которое позволяет отпирать дверь, и он должен доказать Участнику Б это знание, не произнося (громко) самого заклинания.

На первом этапе протокола Участник Б занимает позицию снаружи пещеры так, чтобы ему не были видны перемещения Участника А. В свою очередь, участник А входит в пещеру и, случайным образом выбрав направление движения, встает перед дверью (см. рис. 4.4, а).

На втором этапе Участник Б перемещается внутрь пещеры, встает у развилки и требует, чтобы Участник А вышел через указанный Участником Б коридор. Очевидно, что если Участник А уже находится в требуемом коридоре, то он просто возвращается по нему. Если же он первоначально зашел в пещеру через другой коридор, то произнеся шепотом заклинание, он открывает дверь и выходит с нужной стороны (см. рис. 4.4, б).



а) Первый этап протокола б) Второй этап протокола

Рис. 4.4. Пещера Али-Бабы

Далее Участники занимают прежние позиции и выполняют оба этапа протокола несколько раз. Очевидно, во время первой итерации вероятность обмануть коллегу, то есть выйти через «правильный» коридор, не зная заклинания, равна P=0.5. После второй итерации она снижается до P=0.5×0.5=0.25, после третьей до P=0.5×0.5×0.5=0.125 и так далее, пока Участник Б не «поверит» в знание Участником А волшебного заклинания с требуемым уровнем достоверности.

Итак, с одной стороны, Участник Б может убедиться в знании Участником А секретного пароля. С другой стороны, он не получит никакой информации о самом секретном пароле.

4.3. Технические средства аутентификации

Аутентификация с использованием технических средств требует не «знания» некой информации, но «обладания» неким предметом, подтверждающим подлинность ее владельца. Основные требования к такому предмету:

- уникальность, то есть единственность;
- невозможность создания копии (любая копия считается поделкой).

Области применения технических средств аутентификации (ТСА):

- в СКУД – системах контроля и управления доступом людей, транспорта и товаров в/из помещения, здания, зоны и территории;

- в ТСЗАП – технических средствах защиты авторских прав (их еще называют DRM – digital rights management), которые намеренно ограничивают и затрудняют различные незаконные действия над программами и данными, представленными в электронной форме (запуск, просмотр, модификацию, создание копий и пр.);
- в сфере электронной коммерции.

Рис. 4.5. Устройство ТСА

Практически все ТСА содержат (см. рис. 4.5):

1) носитель информации, который содержит уникальный ключ или пароль;
2) интерфейс ко внешнему устройству, проверяющему подлинность;
3) (не обязательно) электронную логическую схему, которая обеспечивает выполнение некоторого протокола аутентификации (примеры см. в разделе «Протоколы удаленной аутентификации»).

Пример 1. «*Ключевой диск*» – носитель для ЭВМ (дискета, компакт диск и т.п.), содержащий секретный ключ, пароль или счетчик количества запусков программы. Обычно они используются в качестве простых и дешевых ТСА, предназначенных для предотвращения незаконного использования программных продуктов – видеоигр, прикладных программ и т.п. Обычно «ключевые диски» содержат специально созданные физические дефекты или логические нарушения формата хранения данных, что препятствует копированию их стандартными устройствами и программами ЭВМ. Такими свойствами обладает, например, система защиты от нелегального запуска видеоигр «StarForce». Недостатки «ключевых дисков» – недолговечность носителей и потенциальная возможность изучения и воспроизведения формата хранения данных со стороны злоумышленника.

Пример 2. «*Электронный ключ*» (так же используются термины «*токен аутентификации*», «*донгл*», security e-lock и пр.) представляет собой электронное устройство, содержащее:

- энергонезависимую память для хранения ключевой информации;
- специализированную микросхему (чип), обслуживающую протокол аутентификации;
- интерфейс для подключения к ЭВМ.

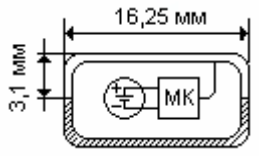
Обычно современные электронные ключи используют интерфейс USB (см. рис. 4.6,б), однако существуют решения на интерфейсах Centronix (см. рис. 4.6,а), COM, PCI, PCMCIA и т.п. Возможно так же применение технологий бесконтактной передачи информации, например, BlueTooth или RFID (см. далее).

а) С параллельным интерфейсом Centronix б) С последовательным интерфейсом USB

Рис. 4.6. Разновидности «электронных ключей»

Типичным является широкое использование «токенов аутентификации» для аутентификации доступа к программному обеспечению, для хранения ключей шифрования и цифровых сертификатов и пр.

Пример 3. Технология *Touch Memory* (соответствующие устройства носят наименование *iButton,* «даллас», «таблетка» и пр.), разработанная фирмой Dallas Semiconductor, используется для передачи идентифицирующей информации путем прикосновения электронного ключа к специальному устройству считывания. Подобные электронные ключи применяются, например, в дверных и автомобильных замках.



а) Внешний вид «таблетки» б) Устройство «таблетки» В) Считыватель с интерфейсной картой

Рис. 4.7. Электронный ключ типа Touch Memory

Типичный электронный ключ (см. рис. 4.7) содержит микросхему DS-1990A с запоминающим устройством объемом 64 бита, из которых 48 бит содержат секретный ключ, 8 бит отводится на тип устройства и 8 на контрольную сумму типа CRC. Информация передается по канальному протоколу 1-Wire со скоростью 16 кбит/сек. Питание подается со считывающего устройства или от встроенной литиевой батарейки напряжением 3В и сроком работы до 10 лет. Основной недостаток технологии: отсутствие в типичном случае какого-либо протокола аутентификации, что позволяет «записать», а вслед за этим «воспроизвести» обмен информацией между «ключом» и «замком» при помощи, например, звуковой карты компьютера.

Пример 4. Бесконтактная *RFID-технология* идентификации и аутентификации (Radio Frequency ID – радиочастотный идентификатор) основывается на миниатюрных микросхемах площадью несколько квадратных мм, содержащих ключевую информацию и приемо-передающее устройство (см. рис. 4.8).

а) Микросхема приемо-передатчика

б) Антенна на самоклеющейся пленке

Рис. 4.8. Устройство RFID-метки

Их работа не требует прикосновения к считывателю и возможна на расстоянии от нескольких сантиметров до нескольких метров. Маленькие размеры[1] микросхемы позволяют встраивать ее в смарт-карты, брелоки, наклеивать на продаваемые товары, вживлять под кожу и пр. Поскольку себестоимость RFID-микросхемы невысока, существует практика «сжигания» RFID-меток в сильном электромагнитном поле после покупки товара.

Пример 5. *Карты с магнитной полосой* – морально устаревший, но все еще используемый способ хранения идентифицирующей, аутентифицирующей и иной секретной информации, например, денежной суммы. Сама карта имеет стандартные размеры (например, 85.6×54 мм), изготавливается из пластика и содержит от одной до трех магнитных полос. Магнитная полоса это линия из магнитного материала, нанесенная на диэлектрическую основу. Ширина полосы 9.52 мм, она содержит три дорожки шириной по 2.79 мм, на которые и записывается секретная информация. Существуют стандарты на кодирование информации (например, ISO/IEC 8583), однако, распространена и практика умышленного отказа от стандартов. В настоящее время карты с магнитной полосой могут быть легко подделаны, поэтому подобная технология применяется лишь в случаях, когда неверная аутентификация не приведет к серьезным последствиям, – например, для изготовления корпоративных пропусков и ключей к офисным дверям. Так же магнитные полосы могут использоваться для хранения вспомогательной (несекретной) информации на устройствах других типов, например, на «смарт-картах».

Пример 6. *Смарт-карты* – современные устройства аутентификации, имеющие вид пластиковой карты (см. рис. 4.9) и снабженные специальной микросхемой, которая выполняет роли: 1) носителя секретной информации; 2) процессора, поддерживающего протокол аутентификации; 3) котроллера интерфейса. Такая микросхема содержит АЛУ, постоянную память, ОЗУ и работает под управлением специализированных операционных систем типа MULTOS. В формате «смарт-карт» в настоящее время изготавливаются банковские карты различных типов, SIM-карты мобильных телефонов, «электронные ключи» и иные ТСА.

Рис. 4.9. Смарт-карты

[1] Самая большая «деталь» RFID-устройства – антенна из проволоки или металлической фольги.

4.4. Методы биометрической аутентификации

Биометрическими характеристиками называются уникальные характеристики человека, которые отличают его от других людей, а именно: отпечатки пальцев, рисунок кровеносных сосудов, голос и прочее.

Средства распознавания человека по его биометрическим характеристикам чаще всего используются в СКУД (см. раздел «Технические средства аутентификации»), в этом случае они играют роль «биометрических идентификаторов». Так же они могут использоваться в качестве одного из факторов аутентификации в системах разграничения доступа к информационным ресурсам.

а) Распознавание по отпечаткам пальцев

б) Распознавание по особенностям глаза

Рис. 4.10. Замки с биометрической идентификаций

Наиболее простыми, дешевыми и популярными являются средства распознавания человека по его отпечаткам пальцев (см. рис. 4.10, а). Несколько реже используются средства распознавания по рисунку радужной оболочки или сетчатки глаза (см. рис. 4.10, б), по форме кровеносных сосудов руки. Используются также средства распознавания геометрии человеческого лица, но область их применения ограничена сферой криминалистики. Прочие методы и средства биометрического распознавания используются редко.

Термин «*биометрический паспорт*» применяется по отношению к персональным идентификаторам, которые сочетают ряд технологий.

Во-первых, они обладают свойствами «смарт-карт», поскольку содержат запрессованные в бумагу или пластик специализированные микросхемы. Память этих микросхем содержит основную информацию о человеке – изображение лица, фамилию, имя, отчество, дату рождения и прочее. Кроме того, эти микросхемы могут содержать[1] цифровой «эталон» для биометрических характеристик. Например, микросхема может хранить цифровые изображения отпечатков пальцев, которые сравниваются с реальными отпечатками «предъявителя».

Во-вторых, «биометрические паспорта» обладают свойствами RFID-устройств, поскольку поддерживают бесконтактное считывание информации.

«Биометрический паспорт» используется в качестве основного и единственного удостоверения личности на Украине, в Казахстане, Молдавии и в ряде стран ЕС. В Российской Федерации таким свойством обладают только загранпаспорта, выдаваемые с 2007 года.

[1] Международного стандарта на состав и формат хранения биометрической информации пока не существует.

ТЕМА 5. Защита программ и данных

Объекты информатизации – это:

- автоматизированные системы различного уровня и назначения, а так же системы связи, отображения и размножения информации;
- помещения, в которых установлены автоматизированные системы;
- помещения для конфиденциальных переговоров.

Наиболее распространенными объектами информатизации, которые чаще всего подвергаются кибератакам, являются автоматизированные системы.

Большинство реальных угроз качеству информации реализуются путем непосредственного или дистанционного взаимодействия с «*носителями информации*, используемыми на объектах информатизации. Примеры таких носителей:

- бумага или пластик, или любой другой объект с поверхностью, на которую можно тем или иным образом наносить изображения;
- магнитный диск, или лента, или любой другой объект с поверхностью, участки которой способны долговременно сохранять состояние намагниченности;
- токопроводящий кабель, по которому распространяются электрические сигналы;
- оптоволоконный кабель, по которому распространяются световые сигналы;
- микросхема памяти с набором полупроводниковых ячеек, способных сохранять электрические потенциалы.
- сама ЭВМ (персональный компьютер, планшет и т.п.), которая содержит внутри носители информации.

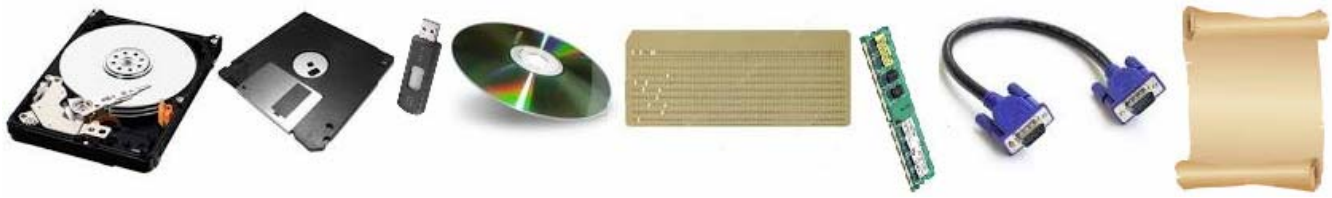


Рис. 5.1. Носители информации

Так же к «носителям информации» при некоторых оговорках могут быть отнесены «физическое поле» и «человек».

Информация на носителях хранится преимущественно в виде «файлов». *Файл* – именованный массив информации.

Основные виды информации в ЭВМ:

- *программа* – описание некоторого алгоритма на некотором языке, предназначенное для выполнения на некотором физическом или виртуальном «процессоре»;
- *данные* – информация, подлежащая обработке (обычно, с помощью программ) и использованию человеком, то есть тексты, изображения, звуки и пр.

В ЭВМ с «манчестероской» архитектурой понятия «программа» и «данные» часто оказываются трудноразделимы. Так, например, описание какого-либо алгоритма («программа») может не только выполняться, но и сохраняться на каком-либо носителе, а так же копироваться, модифицироваться и т.п. В этом случае оно (описание) должно рассматривается как «данные». Существуют подходы, позволяющие избавиться от этого недостатка – см. далее раздел «Архитектурные особенности защищенных операционных систем».

5.1. Проблема физического доступа

Наиболее просто угрозы качеству информации реализуются в результате физического доступа «злоумышленников» к элементам объектов информатизации, в том числе, к носителям информации. Например, похищение конфиденциальной информации можно организовать путем копирования файлов с ЭВМ на съемный носитель (флешку), а уничтожение – путем применения штатных средств операционной системы, вроде консольных команд «del» (для Windows) или «rm» (для UNIX).

Защита информации от таких «атак» выполняется путем применения как средств физической защиты, так и организационных мероприятий.

Средства физической защиты. Для защиты территорий, зданий и помещений, в которых расположены объекты информатизации, применяется их ограждение и физическая изоляция. Отдельные же элементы объектов информатизации помещаются в хранилища (шкафы, сейфы, кабель-каналы, специальные корпуса и т.п.) с использованием запирающих устройств (механических, электрических, электромеханических замков и задвижек).

Организационные мероприятия. Наиболее общим подходом является административный контроль доступа посторонних людей на охраняемые территории и помещения, а сотрудников к документам, носителям и т.п. Невозможность копирования информации с ЭВМ может быть организована путем опечатывания интерфейсных разъемов, используемых для подключения съемных носителей.

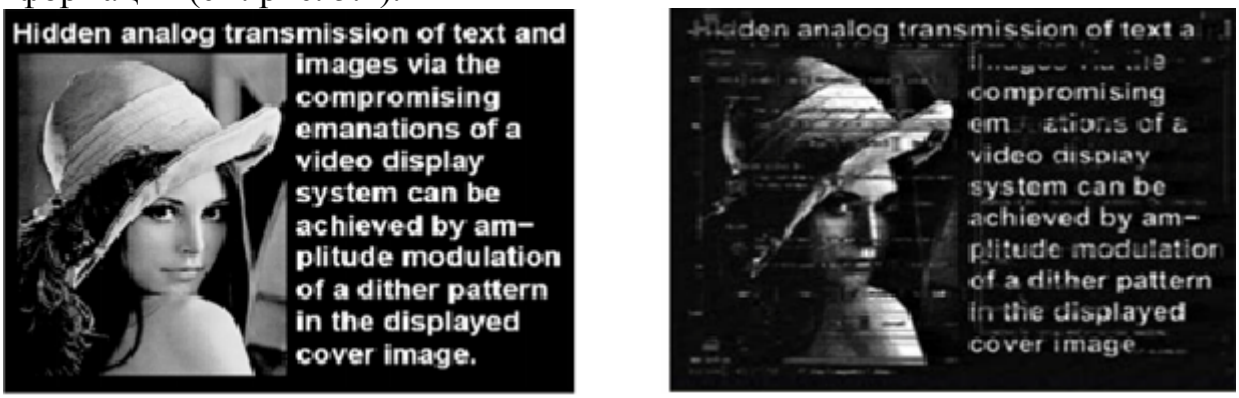
Состав конкретных физических и организационных средств определяется действующей в организации (на предприятии) политикой безопасности, которая, в свою очередь, зависит от уровня конфиденциальности обрабатываемой информации, от режима обработки данных и пр. Более подробно этот вопрос будет рассмотрен далее.

5.2. Проблема ПЭМИН

Проблема *ПЭМИН* – проблема побочных электромагнитных излучений и наводок. В англоязычной литературе используется понятие «*TEMPEST Problem*» – Проблема стандарта на компрометирующие излучения радиоэлектронной аппаратуры.

Суть проблемы заключается в том, что средства вычислительной техники и составные компоненты этих средств (включая коммуникационные шины и кабели, а так же отдельные микросхемы) в процессе работы непрерывно

излучают электромагнитные сигналы, несущие следы обрабатываемой информации[1]. Они (сигналы) могут быть дистанционно восприняты и декодированы «злоумышленником» с целью несанкционированного доступа к информации (см. рис. 5.2).



а) Исходное изображение б) Перехваченное изображение

Рис. 5.2. Перехват сигнала с видеокабеля (по материалам фирмы «Маском»)

Проблема усугубляется тем, что при этом действует эффект *переизлучения*, то есть электромагнитные сигналы от средств вычислительной техники «наводятся» на внешние токопроводящие объекты (так называемые *случайные антенны*) и «переизлучаются» во внешнюю среду уже от них. Примеры «случайных антенн»: кабели электропитания; металлические трубы отопления, водопровода и канализации. Эффект «переизлучения» может быть многократно усилен «злоумышленником», если он извне облучит вычислительную технику и случайные антенны мощным высокочастотным электромагнитным сигналом и воспримет «отражение», так же несущее в себе следы обрабатываемой информации.

Наиболее прост перехват информативных сигналов, излучаемых кабелями клавиатуры, принтера и видеокабелем (см. рис. 5.2). Расстояние, на котором возможны их (сигналов) перехват и декодирование, достигает 30-50 м[2].

Наиболее остро «проблема ПЭМИН» стоит в организациях и на предприятиях, где производится обработка секретной информации.

Борьба с «проблемой ПЭМИН» складывается из нескольких этапов (шагов) и носит итеративный характер.

Шаг 1. Визуально и при помощи специальных технических средств производится исследование средств вычислительной техники и «контролируемой зоны» на присутствие «*закладок*» – устройств, выполняющих несанкционированный «съем» информации (см. рис. 5.3). *Контролируемая зона* (или *зона* «*К*») – прилегающее к средствам обработки информации пространство, в котором исключено неконтролируемое пребывание

Рис. 5.3. «Закладка» на разъеме клавиатуры

[1] Возможно и обратное: сильная электромагнитная помеха способна исказить данные и нарушить процесс обработки.

[2] Вдали от посторонних помех (например, в пустыне) – до километра.

посторонних лиц и технических средств (например, это может быть отдельная комната или весь офис). В случае обнаружения «закладные устройства» удаляются.

Шаг 2. При помощи специальных технических средств выполняется исследование уровней излучения на разных расстояниях от средств вычислительной техники и случайных антенн. Производится разбиение пространства на «зоны» (см. рис. 5.4). Зона R_1 – внутри которой излучения от «случайных антенн» превышают норму. Зона R_2 – внутри которой излучения от компьютеров превышают норму.

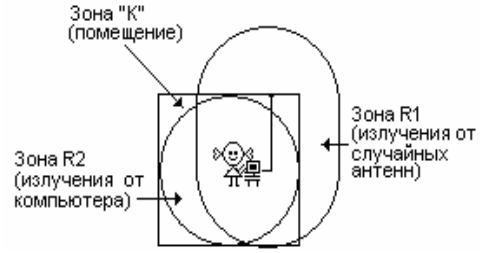


Рис. 5.4. Зонирование пространства

Шаг 3. Если зоны R_1 и R_2 выходят за пределы зоны «К», то применяются технические и административные меры, которые позволили бы увеличить зону «К» и уменьшить зоны R_1 и R_2, после чего производится возврат к Шагу 2.

Технические средства исследования уровня излучений, как правило, представляют собой программно-аппаратные комплексы, включающие в себя чувствительный приемник, комплект антенн, цифровой осциллограф и ЭВМ со специальным программным обеспечением (см. рис. 5.5). Они позволяют обнаруживать источники излучений на разных частотах, определять интенсивность излучений, отношение сигнал/шум, сопротивление заземлений и иные характеристики.


Рис. 5.5. Программно-аппаратный комплекс «Сигурд»

Технические меры уменьшения зон R_1 и R_2 могут быть *пассивными* (т.е. направленными на снижение уровня излучений) и *активными* (т.е. направленными на снижение отношения сигнал/шум в этих излучениях).

К «пассивным» средствам, например, можно отнести экранирование помещений, кабель-каналов и корпусов ЭВМ при помощи металлических листов и сеток (см. рис. 5.6, а), использование для покрытия стен, потолков и дверей специальных токопроводящих красок. Заметный позитивный эффект дает замена металлической сантехники на пластиковую. Специальные электрические фильтры помогают снизить уровень информативного сигнала в кабелях питания.

а) Защитно-экранирующий кожух для ЭВМ

б) Генератор электромагнитного шума

Рис. 5.6. Технические средства борьбы с ПЭМИН

«Активные» средства, в основном, представляют собой генераторы электромагнитного шума, которые не уменьшают уровня излучения, но «маскируют» его информативную составляющую (см. рис. 5.6, б).

5.3. Разрушающие программные воздействия

Разрушающее программное воздействие – реализация информационной угрозы, выполненная с использованием каких-либо программ.

Очевидно, разрушающее воздействие может быть реализовано и не только «специальными», но и «обычными» программами. Например, в операционных системах семейства Windows удалять файлы можно консольной командой «del», а в UNIX – командой «rm».

Вредоносные программы – программы, специально созданные для выполнения разрушающих программных воздействий (подробней см. далее).

В иностранных источниках обычно употребляется термин malware (от англ. malicious – нежелательный), однако под ним часто понимаются не только вредоносные программы, но и некоторые документы, изображения и, вообще, любые информационные объекты, которые пользователь по тем или иным причинам не хотел бы видеть на своем компьютере.

Реальные кибератаки обычно «комплексны», так как последовательно используют многочисленные различные разрушающие воздействия, реализуя тем самым угрозы сразу нескольким показателям качества информации. Например, вредоносные программы-вымогатели (ransomware):

- несанкционированно проникают на компьютер – это нарушение конфиденциальности;
- шифруют файлы данных – это нарушение целостности;
- скрывают ключ шифрования, лишая пользователя возможности восстановить истинное содержимое файлов – это нарушение доступности.

5.3.1. Исследование программного кода

Во время работы ЭВМ процессор поочередно считывает из внутренней памяти машинные команды и выполняет их. Команда, считываемая из памяти,

представляет собой набор чисел, в которых закодирована информация о выполняемой операции и операндах. Каждое семейство процессов имеет свою систему кодирования.

Есть своя система кодирования команд и у «виртуальных» процессоров, например, у Java-машин (для выполнения программ на языке Java), в Microsoft CLR (для выполнения программ на .NET-языках C#, Jscript и т.п.). Программные коды «виртуальных» процессоров часто называют «*байт-кодами*» или «*p-кодами*».

Процесс восстановления и исследования исходных текстов программ по машинному коду называется «*обратной разработкой*» (или «*реверсированием*» – от англ. Reverse Engineering). Методы обратной разработки делятся на две большие группы:

- статические – выполняются «вручную» по восстановленному исходному тексту;
- динамические – предусматривают наблюдение за контролируемым выполнением исследуемого алгоритма, которое выполняется под управлением программы-отладчика или «эмулятора».

Обычно любое исследование алгоритма работы «чужой» программы рассматривается ее правообладателем (автором) как нарушение конфиденциальности и прямо запрещается в лицензионном соглашении.

> Вы не можете... изменять, декомпилировать, дизассемблировать... лицензированную программу или её часть... Любое подобное неавторизованное использование приводит к немедленному и автоматическому прекращению действия этой лицензии и может повлечь за собой уголовное и/или гражданское преследование.
>
> Фрагмент Лицензионного соглашения,
> прилагаемого к программе WinRar.

Действительно, подобные действия подпадают под признаки уголовной статьи №272 «Неправомерный доступ к компьютерной информации». С другой стороны, законодательство РФ иногда допускает подобные действия.

> Лицо, правомерно владеющее экземпляром программы для ЭВМ, вправе без согласия правообладателя... воспроизвести и преобразовать объектный код в исходный текст (декомпилировать программу для ЭВМ) или поручить иным лицам осуществить эти действия, если они необходимы для достижения способности к взаимодействию... программы... с другими программами... при соблюдении следующих условий:
> 1) информация, необходимая для достижения способности к взаимодействию, ранее не была доступна этому лицу из других источников;
> 2) указанные действия осуществляются в отношении только тех частей декомпилируемой программы для ЭВМ, которые необходимы для достижения способности к взаимодействию;

3) информация, полученная в результате декомпилирования, может использоваться лишь для достижения способности к взаимодействию... программы для ЭВМ с другими программами...

Гражданский кодекс РФ (часть четвертая)
от 18.12.2006 №230-ФЗ. Статья 1280.

Например, работа сотрудника антивирусной компании, исследующего зараженную компьютерным вирусом программу, вполне законна.

«Злоумышленники» же обычно используют «обратную разработку» для следующих разрушающих программных воздействий:

- исследование алгоритма программы с целью «плагиата»;
- поиск «ошибок» и «уязвимостей» в программе с целью их дальнейшего злонамеренного использования;
- «взлом», то есть изменение алгоритма работы программы.

Основными техническими методами, применяемыми для борьбы с «обратной разработкой» программ, являются:

- использование «уязвимостей» в алгоритме работы программ-дизассемблеров и эмуляторов с целью нарушить их работу;
- шифрование машинного кода программ в файлах с расшифровкой во время запуска;
- *обфускация* (то есть, сознательное усложнение и запутывание) исходного текста программ.

5.3.2. «Взлом» программного обеспечения

Жаргонный термин «взлом» (англ. *cracking* – раскалывание) обычно связывают с нелегальным видоизменением машинных кодов программы. В результате «взлома» искажается алгоритм работы программы, например, программа теряет способность выполнять правильную аутентификацию.

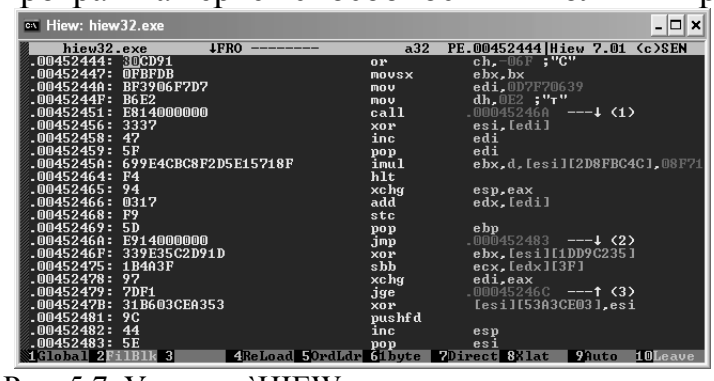


Рис. 5.7. Утилита `HIEW позволяет «редактировать» машинные команды

Чаще всего видоизменению подвергается машинный код в файле программы (см. рис. 5.7). «Взломанные» таким образом программы (компьютерные видеоигры, офисные пакеты, операционные системы и пр.) получали и получают очень широкое распространение среди пользователей.

Другой подход ко «взлому» заключается в том, чтобы видоизменять машинные команды не в файле, а уже после того, как программа будет загружена в память и начнет выполнение. Хакерские утилиты, которые выполняют подобные действия, называются «*трейнерами*». Ими часто пользуются подростки для того, чтобы во время работы игровой программы добавить своему игровому персонажу «здоровья», «патронов», «денег» и т.п.

Известны так же «трейнеры» для нелегального отключения режима ограниченной функциональности в условно-бесплатных программах.

Считается, что проблема «взлома» программного обеспечения радикального решения не имеет. Любая программа может быть «взломана». Именно поэтому шифрование и расшифрование информации высокого уровня секретности осуществляется не программными, а исключительно аппаратными средствами (см. раздел «Общая характеристика современных шифров»).

Приемлемым решением проблемы считается обеспечение такого положения дел, чтобы трудозатраты и стоимость «взлома» превышали бы полученную от него выгоду. Существуют общие подходы, позволяющие сильно затруднить «взлом» и, тем самым, увеличить его стоимость.

Например, поскольку одним из этапов «взлома» программы является ее «обратная разработка», то затруднить «злоумышленникам» работу можно с использованием методов, рассмотренных в разделе «Исследование программного кода».

Так же возможен контроль целостности программного кода путем расчета и проверки контрольных сумм и хеш-функций. Такой подход используется, например, для самопроверки наличия искажений в ROM BIOS (см. раздел «Простые контрольные суммы»). Кроме того, контрольные суммы типа CRC-32 рассчитываются и проверяются операционной системой Windows при загрузке DLL. Правда, применяя такой подход, следует иметь в виду, что «злоумышленник» может, исказив программный код, заново пересчитать контрольную сумму и подменить эталон.

5.3.3. Программные «закладки»

Программная закладка – намеренно оставленный автором или внедренный извне «злоумышленником» фрагмент программы, выполняющий недекларированные (то есть, не описанные в документации) действия. При помощи программных закладок может похищаться конфиденциальная информация, они могут выполнять команды удаленного «злоумышленника» и т.п.

Практикуются следующие способы внедрения программных закладок извне: «пересборка» и «заражение» программ.

«*Пересборка*» носит индивидуальный характер, выполняется «вручную» и складывается из следующих шагов:

- дизассемблировать или декомпилировать программу;
- изучить ее алгоритм;
- добавить в программу фрагмент, реализующий функционал «закладки»;
- выполнить обратную компиляцию программы.

Чаще всего такой модификации подвергаются программы, написанные на языке Java и предназначенные для выполнения в операционной системе Android. Но встречаются и «пересборки» Windows-программ. Программные закладки, внедренные методом «пересборки», очень трудно обнаружимы.

«*Заражение*» обычно выполняется компьютерными вирусами и направлено на массовое внедрение «закладок» внутрь многочисленных программ.

Форматы программных файлов в разных операционных системах похожи (см. рис. 5.8, а), их структурными частями являются: 1) заголовок программы; 2) область системных таблиц; 3) область программ и данных. Одно из полей заголовка содержит указатель на позицию первой исполняемой команды программы. Чаще всего «заражение» заключается в том, что закладка размещается в области программ и данных, а ссылка на начало программы модифицируется таким образом, чтобы указывала на закладку. Последней командой закладки является переход на правильное начало программы.

а) «Здоровая» программа б) Закладка внутри программы

Рис. 5.8. Внедрение закладки в программу по «вирусному» принципу

«Вирусные» закладки обнаруживать легче, чем внедренные методом «пересборки». Для их обнаружения необходимо исследовать стартовый фрагмент подозрительной команды.

Изредка встречаются и иные способы «заражения». Например, указатель на первую команду программы не изменяется, а куда-нибудь в середину программы вставляется машинная команда перехода на закладку. Такие закладки очень трудно обнаружимы.

5.3.4. Логические бомбы

Логическая бомба – программная закладка, которая действует не постоянно, а лишь при выполнении каких-либо условий (например, при наступлении какой-нибудь даты, нажатии какой-нибудь комбинации клавиш, появлении на компьютере какой-нибудь конкретной версии конкретной программы и т.п.).

Классическим примером «логической бомбы» является вредоносный фрагмент, намеренно оставленный осенью 1982 г. тольяттинским программистом Мурадом У. внутри программы, управляющей поставкой комплектующих на главный конвейер ВАЗа. По задумке «злоумышленника», эта «логическая бомба» должна была изредка вызывать сбои в поставке комплектующих. Но благодаря допущенной ошибке, сбои стали происходить слишком часто, решением проблемы занялся большой коллектив

программистов, программная «закладка» была обнаружена, а «злоумышленник» – разоблачен.

Часто «логические бомбы» находятся внутри компьютерных вирусов и червей. Например, 6 марта 1992 г. сработала «бомба» вируса Win9X.CIH (от же известен как «Чернобыльский») – в результате во всем мире были выведены из строя несколько сотен тысяч пользовательских компьютеров. А «бомба» червя Win32.Stuxnet, широко распространившегося по разным странам в 2010 г., на пользовательских компьютерах себя ничем не проявляла, зато вызывала сбои в работе секретных Иранских предприятий, которые занимались обогащением урана.

5.3.5. Программные «люки»

Программный люк (или «*бэкдор*» от англ. backdoor – задняя дверь) – постоянно действующая программная закладка, которая позволяет несанкционированно получать доступ к защищаемой информации.

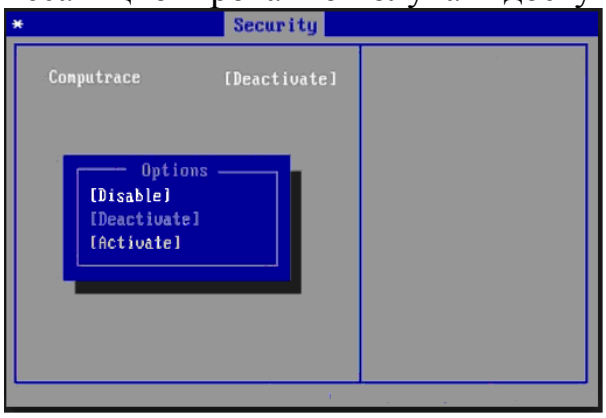


Рис. 5.9. LoJack включить можно, а выключить – нет

Иногда «программные люки» намеренно создаются автором программы и в таком виде передаются пользователю. Так, начиная с 2009 г. многие производители ноутбуков начали включать в «прошивку» (то есть, в области BIOS и UEFI) секретные программные фрагменты LoJack, разработанные компанией Absolute Computrace (см. рис. 5.9). Конечная цель – скрытно устанавливать в операционную систему Windows драйверы, которые способны выполнять удаленные команды, поступившие из Интернета. Утверждается, что эта технология предназначена для борьбы с хищениями ноутбуков, хотя, с таким же успехом она может использоваться и для слежения за легальным пользователем.

Так же «программный люк» может быть внедрен в программное обеспечение не самим автором программы, а ее «распространителем». Примерами таких «люков» в настоящее время являются «пиратские пересборки» программ. Так называются выложенные в Интернет «взломанные» версии операционных систем, полезных утилит и компьютерных видеоигр, в которых не только удалены фрагменты аутентификации (см. раздел «Проблема взлома программного обеспечения»), но и внедрены «программные люки». Такие «закладки» можно обнаружить только в результате очень сложного и трудоемкого исследования программ с использованием «обратной разработки». Проще не пользоваться «пиратскими» программами вообще.

Наконец, «программный люк» может появиться на ЭВМ, как часть компьютерного вируса или «троянской» программы (о них речь пойдет далее).

Забавно, что к классу «программных люков» можно отнести так же «*чит-коды*» (или просто «*читы*») – секретные пароли или клавиатурные комбинации, позволяющие подростку, играющему в видеоигру, получить те же нелегальные выгоды, что и при использовании «трейнеров» (см. раздел. «Взлом программного обеспечения»). В отличие от «трейнеров», «чит-коды» оставляются авторами игровых программ намеренно – обычно они официально «рассекречиваются» через некоторое время после начала продажи игры, чтобы поддержать к ней интерес.

5.3.6. Уязвимости программного обеспечения

Программная уязвимость – постоянно присутствующая ошибка или неточность в алгоритме программы, которая может облегчить «злоумышленнику» практическую реализацию «кибератаки».

1. *Алгоритмические ошибки*. Они достаточно часто встречаются как в прикладных программах, так и в операционных системах. Примером может служить процедура проверки правильности паролей при попытке удаленного доступа к «сетевым дискам» в операционных системах Windows 95 и Windows 98 (номер CVE-2000-0979 в каталоге «уязвимостей» Microsoft, пресс-релиз MS00-072). Соответствующий программный код выглядел примерно так.

```
int check_password( char *pass, char *etalon) {
  for (int i=0;i<strlen(pass);i++) {
   if  (pass[i]!=etalon[i]) return BAD_PASSWORD;
  }
  return PASSWORD_OK;
}
```

Найдите неточность самостоятельно и определите, к каким проблемам безопасности она может привести!

2. *SQL-инъекции*. Множество алгоритмических ошибок встречается в программных комплексах, в которых предусмотрено взаимодействие с базами данных через SQL-сервер (см. рис. 5.10). Чаще всего они связаны с некорректным перекодированием пользовательского запроса в запрос, оформленный по правилам универсального языка SQL.

Рис. 5.10. Схема взаимодействия компонентов СУБД

Например, пусть программа-клиент требует от пользователя ввести строку-идентификатор (например, «masha»), а потом генерирует на основании этой строки SQL-запрос вида

```
SELECT Value FROM Database WHERE Login='masha',
```

который, как предполагается, выдает соответствующее этому идентификатору значение поля «Value». Однако если пользователь введет в качестве идентификатора строку «`' OR '1'='1`», то сгенерируется SQL-запрос

```
SELECT Value FROM Database WHERE Password='' OR '1'='1',
```

в котором правая часть всегда имеет значение «истина» и, следовательно, пользователю будут возвращены значения поля «Value» всех пользователей, внесенных в базу данных. Таким образом, «обманув» программу-клиент, пользователь может получить доступ к конфиденциальной информации. Разнообразные методы, связанные с использованием ошибок генерации SQL-запросов, получили обще наименование: *методы SQL-инъекции*.

3. *Переполнение буфера* в программах возникает, когда данные (или их фрагменты) не умещаются в отведенную для них область памяти. Это случается при обработке массивов данных, имеющих нестандартный формат. Подобные ошибки могут быть использованы «злоумышленниками» для запуска вредоносных программ из «непрограммных» объектов, таких как изображения, документы, сетевые пакеты и т.п.

Эксплойт – вредоносная программа, предназначенная для использования ошибок и уязвимостей в других программах.

«*Шелл-код*» (от англ. shell – ракушка, скорлупа) – элемент эксплойта, который служит для провоцирования уязвимости. Нередко он представляет собой не программу, а «неправильные» данные (возможно, с «вкраплениями» программного кода).

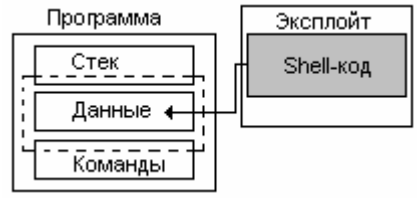


Рис. 5.11. Внедрение shell-кода в программу

За последние десятилетия обнаружены и продолжают обнаруживаться сотни уязвимостей типа «переполнение буфера» в самых разных программах – в браузере Internet Explorer, в редакторе документов Microsoft Word, в программе просмотра PDF-документов Adobe Acrobat, в компонентах операционных систем, и многих прочих. Они стали причиной многочисленных инцидентов, связанных с распространением сетевых червей, похищениями конфиденциальных данных и т.п. Например, в мае 2017 г. уязвимость CVE-2017-0144 в сетевом протоколе SMB позволила червю WannaCry проникнуть на сотни тысяч компьютеров и зашифровать на них информацию.

Меры, применяемые для борьбы с «переполнением буфера»:

- использование языков программирования, следящих за целостностью стека (например, Pascal, Modula-2, Java, C# и пр.);
- использование в «опасных» языках программирования (например, в C/C++) «надежных» функций (например, «scanf_s» вместо «scanf»);
- запрет использования в программах статических массивов;
- аппаратный запрет выполнения машинных команд, размещенных в стеке, путем установки бита «NX» в дескрипторах сегментов и страниц (это возможно только в современных моделях процессоров Intel).

4. *Борьба с уязвимостями.* Как правило, специальные подразделения крупных организаций – производителей программного обеспечения занимаются поиском «уязвимостей» в своих продуктах не только на этапе предварительного тестирования, но и после того, как продукт выпущен на рынок. Обнаружив «уязвимость», организация выпускает и рассылает легальным пользователям «обновления» и «*заплатки*» (или «*патчи*» от англ.

patch – заплата), исправляющие ошибку. Фирма Microsoft ведет общедоступный каталог обнаруженных в ее продуктах уязвимостей и регулярно выпускает пресс-релизы, информирующие компьютерную общественность.

Однако значительная часть «уязвимостей» обнаруживается сторонними исследователями, занимающимися «обратной разработкой» чужих программных продуктов. Среди таких «исследователей» выделяют три группы:

- «*белые шляпы*» сообщают об обнаруженной уязвимости только производителю программы, в надежде получить благодарность или небольшое вознаграждение;
- «*серые шляпы*» с целью саморекламы безвозмездно передают сведения об «уязвимости» всем желающим – и авторам уязвимой программы, и потенциальным «злоумышленникам», и просто любознательным «наблюдателям»;
- «*черные шляпы*» с целью финансовой выгоды[1] распространяют информацию об «уязвимости» только на «черном рынке» среди «злоумышленников».

И только немногие «уязвимости» обнаруживаются случайно[2], в результате практической эксплуатации программного обеспечения.

Техническая информация об «уязвимостях» обычно распространяется в виде «*эксплойтов*» – маленьких программ, демонстрирующих суть «уязвимости» и способы ее практического использования.

Существует обоснованное подозрение, что некоторые «уязвимости» на самом деле представляют собой «программные люки», намеренно оставленные разработчиками программ.

5.3.7. Похищение конфиденциальной информации

Одним из самых частых и потому опасных разрушающих программных воздействий является похищение конфиденциальной информации.

Похищение может быть выполнено в результате физического доступа «злоумышленника» к информации, например, путем копирования файлов на съемный носитель – дискету или флешку. Основной способ противодействия копированию – шифрование своих файлов и дисков. «Злоумышленник», получивший доступ к компьютеру или похитивший жесткий диск, не зная правильного пароля, увидит вместо файлов «мусор». Наиболее популярны следующие средства:

---

[1] Стоимость недавно обнаруженной, никому еще не известной «уязвимости» (англ. «zeroday vulnerability» или «0-day vulnerability» – уязвимость текущего момента) на черном рынке может достигать нескольких десятков тысяч долларов.

[2] «Искатели уязвимостей» сами используют случайность, подавая на вход программе случайные наборы данных случайной длины. Этот прием называется «фаззинг» (от англ. to fuzz – взъерошивать, придавать пушистый вид).

- файловая система NTFS позволяет «прозрачно» шифровать и расшифровывать на лету файлы, генерируя ключ из имени и пароля пользователя;
- встроенная в MS Windows Professional (но не Home Edition!) подсистема BitLocker позволяет «прозрачно» шифровать и расшифровывать на лету диски, используя ключ на Flash-носителе;
- прикладные программы PGP Disk и TrueCrypt позволяют шифровать и расшифровывать на лету файлы и диски, генерируя ключ на основе секретного пароля.

Другой способ похищения информации – внедрение в системное программное обеспечение программных закладок типа «*логгер данных*» (или «*автономный регистратор данных*»).

Клавиатурный шпион (или «*keylogger*» от англ. key – ключ) – программа-регистратор нажатий клавиш. Как правило, программы этого типа перехватывают и записывают в файл коды всех клавиш, нажатых в сеансе работы. Изучая содержимое этого файла, «злоумышленник» может получить сведения о секретных логинах и паролях пользователя, которые по умолчанию не отображаются на экране.

Существуют также «логгеры» информации, выводимой на печать, и «логгеры» движений мыши. В последние годы появились программные закладки, которые способны незаметно включать видеокамеру и микрофон, встроенные в ноутбук, и регистрировать визуальную и звуковую информацию. Для противодействия «логгерам» используются те же методы, что и для любых других программных закладок.

Обычно «логгеры» работают в паре с «дропперами» – вредоносными программами, передающими накопленную «логгерами» информацию по компьютерной сети. Использование сетей с целью похищения информации рассмотрено дальше.

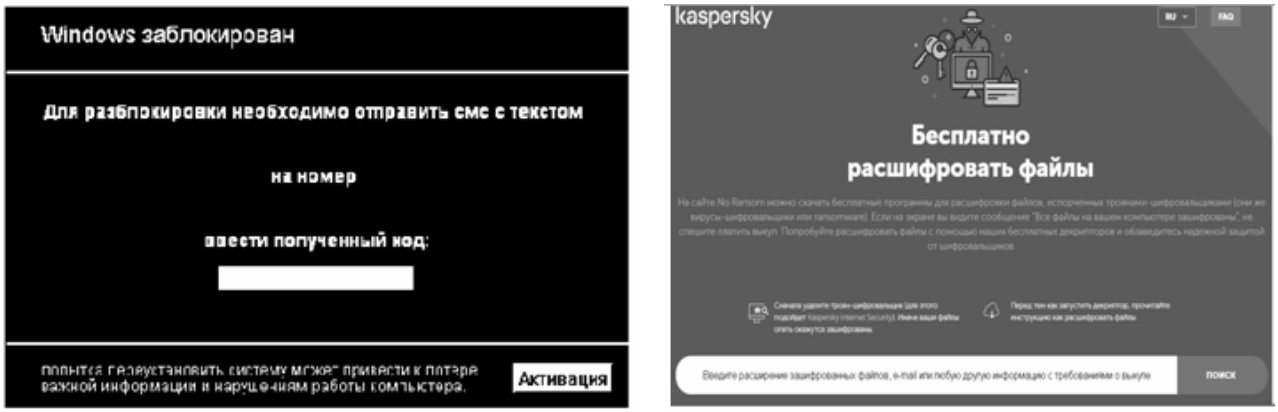
5.3.8. Блокирование доступа к информации

Очень опасным разрушающим программным воздействием является блокирование информации, обрабатываемой на компьютере.

В последнее десятилетие широкое распространение получили кибератаки, направленные на вымогательство у пользователя денежных средств путем блокирования доступа к файлам. Как правило, они реализуются при помощи вредоносных программ типа «*ransomware*» (от англ. ransom – вымогательство).

Очень эффектны так называемые «*винлокеры*» (от англ. to lock – запирать на замок). Получив управление, они работают как программная закладка и блокируют все устройства ввода информации, которыми мог бы воспользоваться пользователь – прежде всего клавиатуру и мышь, после чего помещают на экран сообщение с требованием «выкупа» (см. рис. 5.12,а). Файлы при этом никаким изменениям не подвергаются, и доступ к ним можно получить, загрузив операционную систему с «чистого» носителя.

Гораздо опаснее программы-«шифровальщики». Типичный пример – сетевой червь WannaCry (май 2017 г.). Попав на компьютер, он зашифровывает методом AES на диске практически все файлы данных, включая документы (расширения .DOC и .DOCX), электронные таблицы (расширения .XLS и .XLSX), изображения (расширения .JPG, .GIF и .PNG), видеоролики (расширения .AVI, .MP4, .WMV) – всего более сотни разных типов. Шифрование выполняется случайным ключом, а сам он зашифровывается методом RSA с открытым ключом. Приватная «половинка» ключа при этом остается у неизвестного «злоумышленника». После шифрования вымогатель извещает пользователя о требуемой за восстановление данных сумме и способах оплаты. Как правило, «восстановить» файлы путем выполнения этих требований не удается, «злоумышленники» просто собирают деньги.



а) Сообщение «винлокера» б) Антивирусный расшифровщик

Рис. 5.12. Вымогатель и средство против него

Существует небольшая вероятность расшифровать файлы (см. рис. 512,б). Это возможно только в том случае, если программа-вымогатель использовала симметричный шифр, и ключ находится внутри ее.

Блокирование удаленного доступа к информации путем искусственно созданных «отказов в обслуживании» так же будет рассмотрено далее.

5.4. Вредоносные программы

Итак, *вредоносные программы* – программы, специально созданные для выполнения разрушающих программных воздействий. Часто так же рассматривают более широкий класс так называемых «*malware*», этим термином описывают множество «нежелательных» объектов, не обязательно являющихся программами.

Рис. 5.13. Классификация вредоносных программ

Часто все вредоносные программы называют «вирусами», хотя среди специалистов существует более корректная классификация (см. рис. 5.13).

В УК РФ присутствует статья 273, преследующая за создание, распространение и использование вредоносных программ. Максимальное наказание – до 7 лет лишения свободы.

5.4.1. Компьютерные вирусы

Компьютерные вирусы-паразиты (или просто *вирусы*) – программы, способные к неконтролируемому созданию собственных копий, то есть к саморазмножению.

Основной способ размножения компьютерных вирусов – внедрение их внутрь программных объектов в качестве «программной закладки» (см. рис. 5.8 в соответствующем разделе) и распространение вместе с ними в том случае, если эти объекты передаются от пользователя к пользователю. Они стартуют в момент запуска пользователем зараженной программы.

Вирусы могут (хотя и необязательно!) нести в себе вредоносные фрагменты, способные уничтожать информацию, похищать конфиденциальные сведения, шифровать файлы и прочее.

Основные разновидности компьютерных вирусов приведены в табл. 5.1.

Признаками того, что программа заражена компьютерным вирусом, могут служить изменение даты создания файла, увеличение его длины и прочее, хотя многие вирусы способны скрывать подобные изменения.

Таблица 5.1. – Вирусы-паразиты

Тип вируса	**Носитель**	**Примеры**
Файловые вирусы	Файл загрузочного образа программы	«Буквопад» (1988), «Чернобыль» (1998-99), Sality (2003)
Загрузочные вирусы	Загрузочный сектор дискеты или иного носителя	«Микеланджело» (1991-92)
Макро-вирусы	Документ Word, электронная таблица Excel	Concept (1995), Proverb (2000)
Иные	Например, исходный текст программы	Induc (2009 г.)

Пик распространения загрузочных вирусов пришелся на конец 1980-х годов, макровирусов – на конец 1990-х. Файловые компьютерные вирусы для MS-DOS так же были чрезвычайно широко распространены в 1980-90-х гг., однако их разновидности для MS Windows в 2000-х годах большого распространения не имели. В общей массе вредоносных программ они до сих пор составляют лишь незначительно малую долю.

5.4.2. Компьютерные черви

Компьютерные вирусы-черви (или просто *черви*) – разновидность компьютерных вирусов, которые существуют автономно, не встраиваясь ни в какие программные файлы или документы.

Черви не могут стартовать самостоятельно. Чтобы стартовать в первый раз, они стараются обмануть пользователя, соблазняя или вынуждая его запустить вредоносную программу (см., например, Рис. 5.14).

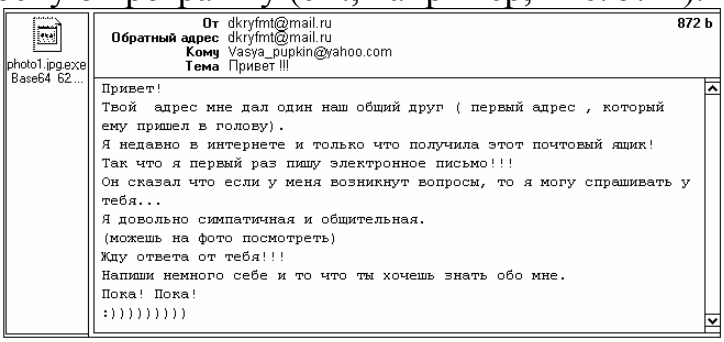


Рис. 5.14. Почтовый червь Win32.Stator «просит» его запустить

Существуют разновидности сетевых червей, которые стартуют автоматически, используя уязвимости прикладного программного обеспечения, – то есть, они запускаются при просмотре зараженных документов, при посещении «заминированной» WWW-страницы в Интернете со тэгом типа

```
<script type= ″text/javascript″ src=″http://... /virus.js″>
. . .
</script>,
```

а то и просто при некорректной обработке сетевых пакетов уязвимыми службами операционной системы. Так же некоторые разновидности червей могут стартовать с флешек и компакт-дисков, имеющих в корневом каталоге файл «AUTORUN.INF» с записью вида

```
[autorun]
open=VIRUS.EXE.
```

После старта черви копируются на носители компьютера и вносят специальные записи в конфигурационные файлы операционной системы, чтобы в дальнейшем стартовать каждый раз при включении или перезагрузке операционной системы. Например, черви часто «прописываются» в одном из перечисленных ниже ключей Реестра Windows:

```
HKLM\Software\Microsoft\Windows\CurrentVersion\Run
HKCR\exefile\shell\open\command
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\userini
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\shell.
```

Основные типы вирусов-червей см. в табл. 5.2.

Наиболее часто сетевые черви встречались в 1999-2005 гг, когда ежегодно возникали десятки обширных, быстро развивающихся эпидемий. При этом за считанные часы заражались сотни тысяч и миллионы узлов Интернета. Для настоящего времени более характерны «медленные» эпидемии файловых червей, распространяющихся при помощи флешек, и «почтовых» червей, распространяющихся в виде вложений в электронные письма, хотя сетевые эпидемии также не потеряли актуальности.

Таблица 5.2 – Вирусы-черви

Тип червя	**Принцип распространения**	**Способ 1-го запуска**	**Способ последующих запусков**	**Примеры**
Файловые черви	Вместе со съемными носителями (флешками)	В соответствии с командами файла AUTORUN.INF	Запись в Реестре	Kido (2008-2012), Stuxnet (2010)
Почтовые черви	В виде вложений в электронные письма	Вручную обманутым пользователем	Запись в Реестре	Melissa (1999), Mydoom (2004)
Сетевые черви	В виде пакетов сетевых протоколов	Путем использования уязвимостей в программах	Запись в Реестре	Blaster (2002), Wannacry (2017)
Мобильные черви	С телефона на телефон	Вручную обманутым пользователем	Запись в файле конфигурации телефона	Cabir (2007)

5.4.3. Троянские программы

Троянские программы (или «троянские кони», «троянцы», «трояны», «трои») свое название получили за попытку маскировки под полезные программы. Однако сейчас «троянскими» называются любые вредоносные программы, не способные к самостоятельному размножению (то есть, не вирусы и не черви). Они размножаются и распространяются не самостоятельно, а «вручную» злоумышленником или «автоматически» при помощи других программ. Первый запуск троянской программы происходит обычно в тех же обстоятельствах, что и запуск сетевого или почтового червя (см. выше). Так же очень часто троянские программы, паразитирующие в операционной системе Android, стартуют из «программной закладки», внедренной методом «пересборки» в какое-нибудь легальное приложение, которое в таком виде размещается злоумышленниками в общедоступных сетевых источниках. Однако в дальнейшем троянские программы не создают своих копий, то есть самопроизвольно не размножаются.

Вот несколько разновидностей (согласно классификациям Лаборатории Касперского и антивируса DrWeb) троянских программ:

- *Trojan-Spy* («шпионы»), *Trojan-Keylogger* («клавиатурные шпионы»), *Trojan-Banker* («банковские трояны»), *Trojan-Mailfinder* («искалка адресов»), *Trojan-IM* («похититель аккаунтов»), Trojan-PSW («похититель паролей»), *Trojan-GameThief* («похититель игровых паролей») – несколько групп вредоносных программ, предназначенных для похищения с компьютера конфиденциальных данных (например, логинов, паролей, pin-кодов, контактных адресов, цифровых сертификатов, документов и т.п.) и слежения за деятельностью пользователя;

- *Trojan-Ransom* («вымогатели») – вредоносные программы, блокирующие работу операционной системы или зашифровывающие файлы на диске, а потом вымогающие у пользователя деньги;
- *Trojan-Proxy* («прокси») – вредоносные программы, служащие ретрансляторами (посредниками) между удаленным злоумышленником и каким-либо Интернет-ресурсом, благодаря чему в качестве источника вредоносных воздействий на этот ресурс виден не компьютер злоумышленника, а зараженный компьютер;
- *Trojan-Backdoor* («программные люки») – вредоносные программы, предоставляющие злоумышленнику удаленный доступ к ресурсам зараженного компьютера, что позволяет ему запускать и останавливать на нем программы, копировать и удалять файлы, следить за работой пользователя, рассылать с этого компьютера спам и другие вредоносные программы и т.п.;
- *Trojan-Rootkit* («руткиты») – вредоносные программы, способные перехватывать и искажать результаты обращения к операционной системе, благодаря чему прикладные программы (например Проводник, файловые менеджеры типа Total Commander, диспетчер задач Windows, некоторые антивирусы) перестают «видеть» зараженные объекты на диске и в памяти;
- *Trojan-Bootkit* («буткиты») – вредоносные программы, заражающие загрузочный сектор носителя или BIOS и обеспечивающие загрузку в память не оригинальной, а искаженной версии операционной системы (например, с «программными люками»);
- *Trojan-DDoS* – вредоносные программы, работающие в режиме «логической бомбы» и предназначенные для скоординированных атак со стороны зараженных компьютеров на какой-нибудь Интернет-ресурс;
- *Trojan-Downloader* («загрузчик») и *Trojan-Dropper* («дроппер») – вредоносные программы, выполняющие служебную роль и предназначенные для закачивания на зараженный компьютер других вредоносных программ либо, наоборот, для передачи удаленному злоумышленнику файлов с зараженного компьютера;
- *Trojan-Clicker* («кликеры») – вредоносные программы, предназначенные для назойливого показа рекламы на зараженном компьютере;
- *Trojan-SMS* («звонилка») – вредоносные программы, несанкционированно посылающие SMS или выполняющие телефонные звонки с зараженного мобильного устройства;
- *Trojan-Notifier* («маяк», «осведомитель», «информатор») – формально безобидные программы, выполняющие служебную роль – оповещающие удаленного злоумышленника по сети о том, что компьютер уже заражен и «готов к использованию»;
- *Trojan-Hoax* («программные шутки») и Trojan-FakeAV («ложный антивирус») – формально безобидные программы, имитирующие программные и аппаратные сбои, заражение компьютера «страшными

вирусами», что, тем не менее, является нарушением нормальной работы компьютера.

Функционал троянских программ охватывает почти весь спектр разрушающих программных воздействий, рассмотренный выше. Обычно на зараженном компьютере с операционной системой Windows присутствует «комплект» троянских программ, запускающих друг друга по цепочке и совместно решающих общую вредоносную задачу. Для телефона или планшета с операционной системой Android более характерно наличие одиночной вредоносной программы, способной выполнять весь комплекс разрушающих воздействий.

В настоящий момент 99% всех вредоносных программ, «паразитирующих» на вычислительных и коммуникационных ресурсах мировой IT-индустрии, представляют собой именно троянские программы для Microsoft Windows и Google Android.

5.4.4. Прочие виды вредоносных объектов

Не все объекты, относящиеся к классу «нежелательных», являются «вредоносными», и не все являются «программами».

Эксплойты – комбинация программного кода и данных, предназначенная для провоцирования и использования ошибок в прикладном и системном программном обеспечении. Часто (но не обязательно!) «эксплойт» является частью вредоносной программы, например, «трояна». Подробней «эксплойты» рассмотрены ранее в разделе «Уязвимости программного обеспечения».

«*Не-вирус*» *EICAR* – крохотная безобидная программа в формате «COM» для операционной системы MS-DOS, специально разработанная сотрудниками EICAR (Европейского института антивирусных исследований) в качестве «тестового вируса». Она состоит из машинных команд, числовые значения которых суть ASCII-коды отображаемых алфавитно-цифровых символов.

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST- FILE!$H+H*
```

Благодаря этому программу можно как запускать на выполнение (она просто выведет на экран свое имя), так и просматривать в любом текстом редакторе, например, в «Блокноте» MS Windows. Однако любой правильно настроенный антивирус обязан распознать эту программу, заблокировать или уничтожить ее. Этим он продемонстрирует свою работоспособность.

«*Демонстраторы рекламы*» (adware) – формально безобидные программы, предназначенные для скачивания из Интернета и демонстрации на компьютере пользователя разнообразной рекламы. Нередко условно-бесплатные утилиты (архиваторы, проигрыватели аудио- и видеофайлов, конвертеры документов и т.п.) сознательно содержат в себе программные закладки, работающие по такому принципу.

«*Потенциально опасные программы*» (riskware) – формально безобидные, а иногда и просто полезные программы, которые, тем не менее, могут быть использованы злоумышленниками в своих целях. Например, очень часто

«потенциально опасными» становятся прокси-серверы, средства удаленного администрирования компьютеров, утилиты скачивания файлов из Интернета и иные небольшие программы, которые могут быть легально установлены пользователем на свой компьютер, а могут и появиться на нем «незаметно», в составе «комплекта» троянских программ.

«*Ложные антивирусы*» – программы, имитирующие работу антивируса и всегда «обнаруживающие» на компьютере множество вредоносных программ, даже если их там нет, но не способные обнаруживать реальные вредоносные программы. Изготовление и распространение таких «ложных антивирусов» представляют собой обман покупателя, мошенничество. Кроме того, такие «антивирусы» сильно вредят репутации антивирусной индустрии, вот почему они блокируются и уничтожаются антивирусами.

Непрограммные объекты. Нередко так же антивирусы блокируют и уничтожают генераторы ключей активации пиратских копий программ, сами такие ключи, используемые хакерами системные утилиты, исходные тексты вредоносных программ, документы с описанием хакерских технологий и прочие объекты.

5.4.5. Особенности устройства и поведения вредоносных программ

1. *Полиморфные* вредоносные программы – программы, любые два экземпляра которой обладают различными исполняемыми кодами, но работают по одному и тому же алгоритму. Например, вычисление дискриминанта программой, решающей квадратные уравнения, в одном экземпляре может выглядеть как

```
d = b*b – 4*a*c,
```

в другом экземпляре как

```
d = c*a*4*sin(4.7123889) + b*b,
```

а в третьем как

```
d = b*b – a*c – c*a – a*c – c*a.
```

Полиморфизм сильно затрудняет обнаружение вредоносных программ антивирусами, работающими по принципу сигнатурного анализа. Наиболее изощренные механизмы полиморфизма встроены в файловые вирусы, так как им при размножении приходится модифицировать самих себя. Многие троянские программы так же является полиморфными, но их различные варианты заранее генерируются внешними программами.

Вот почему в современных условиях разработчики антивирусов сообщают о ежегодном появлении миллионов новых вредоносных программ, хотя в реальности количество новых разновидностей не превышает нескольких тысяч.

2. «*Обфусцированные*» вредоносные программы – программы, чей код специально усложнен и «запутан» с целью затруднения изучения со стороны вирусологов и бдительных пользователей. Для получения эффекта «обфускации» используются примерно такие же приемы, что и в «полиморфных» вредоносных программах.

3. *Модели вирусных эпидемий*. Троянские программы саморазмножаться не умеют, поэтому их распространение находится под контролем злоумышленника-хозяина. Напротив, программы-вирусы и черви склонны к неконтролируемому размножению и распространению.

Коэффициент размножения β – среднее количество новых экземпляров вредоносной программы, возникающее в единицу времени:

- для файловых вирусов и червей – несколько штук в неделю;
- для почтовых и мобильных червей – несколько десятков в сутки;
- для сетевых червей – несколько штук в минуту.

Будем считать заражаемым «объектом» либо отдельную программу, в которую внедряется вирус, либо компьютер, на котором размещается червь.

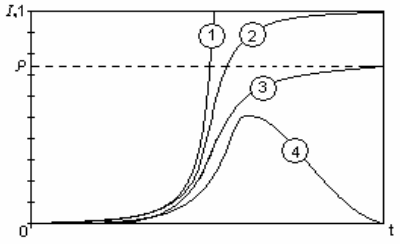


Рис. 5.15. Модели размножения вредоносных программ

Обозначим: S – доля объектов, доступных к заражению; I –доля зараженных объектов; R – доля «вылеченных» объектов, которые более недоступны для заражения. Пусть в момент времени t=0 имеет место S=1 (то есть, все 100% уязвимы).

Пусть каждый объект типа S в ходе эпидемии рано или поздно перейдет в состояние I, такие модели размножения называются «SI-моделями». Если каждый акт размножения вируса гарантированно завершается успехом, то рост количества новых экземпляров происходит по экспоненциальному закону и вирус довольно быстро заразит все доступные для заражения объекты (см. кривую (1) на рис. 5.15). Однако такой сценарий нереалистичен. На самом деле вредоносная программа ищет новые цели для заражения методом проб и ошибок, и чем большее количество объектов уже заражено, тем чаще вирус натыкается на уже «занятую» цель, что приводит к замедлению скорости эпидемии. Поэтому кривая развития реальных эпидемий напоминает латинскую букву «S» (см. линию (2) на Рис. 5.15).

Если процесс размножения испытывает сопротивление со стороны антивирусов, обнаруживающих и удаляющих вирусы из зараженных объектов со средней скоростью γ, то возможен как переход из состояния S в I, так и обратный из I в S. Это «SIS-модель», в рамках которой рано или поздно количество зараженных объектов занимает равновесное состояние на уровне $\rho=\beta/\gamma$ (см. кривую (3) на рис. 5.15).

Наконец, если антивирус не только «исцеляет» зараженные объекты, но и «вакцинирует» их, то есть делает неспособными к новому заражению, то возможна только смена состояний $S\rightarrow I\rightarrow R$, описывающая после быстрого роста плавное снижение количества зараженных объектов вплоть до нуля (см. кривую (4) на Рис. 5.15). На практике это означает, что для достижения в рамках «SIR-модели» полной победы над эпидемией необходимо после завершения работы антивируса проводить работу по предотвращению новых заражений – менять версию операционной системы и уязвимых прикладных программ, устанавливать «заплатки» (обновления), обновлять антивирусные

базы, переходить к использованию других антивирусов и т.п. К сожалению, это правило выполняется не всегда, вот почему эпидемии некоторых вирусов и червей иногда длятся годами.

4. *Ботнет* – сеть взаимодействующих компьютеров, зараженных вредоносной программой одного типа (обычно, относящейся к разновидности «Trojan-Backdoor»). *Бот* (или *агент*) – экземпляр такой вредоносной программы.

а) Ботнет с единым управляющим центром б) P2P-ботнет

Рис. 5.16. Разновидности ботнетов

«Агенты» могут получать и выполнять команды от общего управляющего центра (см. рис. 5.16,а) либо передавать такие команды другу-другу по цепочке (см. рис. 5.16,б). Обычно такие команды зашифрованы «приватным» ключом «хозяина» и могут быть расшифрованы только «публичным» ключом, хранящимся внутри «агента». Также распространена практика, когда «агенты» обращаются за инструкциями к содержимому какого-нибудь общедоступного Интернет-ресурса (например, форума), куда «хозяин» заранее помещает внешне безобидное сообщение с закодированными инструкциями.

В настоящее время в Интернете одновременно действуют многие десятки ботнетов размерами от нескольких тысяч до нескольких миллионов «зомбированных» компьютеров. Обычно «хозяева» ботнетов продают или сдают в аренду их отдельные фрагменты. Таким образом, часть «агентов» какого-либо ботнета может совместными запросами блокировать какой-нибудь Интернет-ресурс; другая часть – рассылать спам или вредоносные программы для подготовки нового поколения ботнета; третья часть – заниматься выгодными для «арендатора» вычислениями, например, подбирать чьи-нибудь пароли или «майнить» биткоины и т.п.

5.4.6. Происхождение вредоносных программ

В истории создания и применения вредоносных программ можно выделить три этапа.

До середины 1980-х гг. написание и использование вредоносных программ носило единичный характер. В основном этой деятельностью занимались ученые и инженеры в исследовательских или развлекательных целях.

С середины 1980-х гг., в связи с широким распространением персональных компьютеров и средств программирования для них, возможность создавать и использовать вредоносные программы получили широкие массы программистов и пользователей. Основную массу вредоносных программ (в тем времена – вирусов и червей) создавали продвинутые школьники, студенты и любознательные программисты. Побудительными мотивами написания таких

программ были озорство, любопытство, стремление «анонимно прославиться», игровой азарт и пр.

Примерно к 2005 г. информационные технологии окончательно интегрировались в производство, средства коммуникации, торговлю, медицину, финансы и прочие сферы мировой экономики, благодаря чему возникла возможность выполнять разрушающие программные воздействия на информацию с корыстными целями. Кроме того, средства антивирусной защиты стали настолько эффективными, что примитивные вредоносные программы, созданные «одиночками», потеряли возможность широкого распространения. Начиная с этого момента и до настоящего времени авторами многочисленных вредоносных программ являются квалифицированные специалисты и коллективы, участие в которых принимают не только программисты, но и администраторы компьютерных сетей, математики, инженеры-конструкторы, дизайнеры и т.п. Целью создания и использования вредоносных программ является получение финансовой, идеологической или политической выгоды. Создание и использование вредоносных программ построено по «индустриальному» принципу, когда одни группы занимаются «исследовательской» деятельностью[1], другие программированием, третьи распространением и т.д. Существует обширный «черный рынок» товаров и услуг в области создания и использования вредоносных программ, на котором можно, например, купить готовую «троянскую» программу или комплект «заготовок» для создания таких программ; похищенную у производителей аппаратных и программных средств документацию; огромные базы данных с паролями к Интернет-ресурсам, адресами электронной почты, персональными данными пользователей и т.п. Можно также арендовать ботнет или его часть, нанять временный или постоянный коллектив «хакеров». Основными участниками «черного рынка» являются как представители киберкриминала, так и спецслужбы различных государств. Однако присутствуют на этом «рынке» и малоквалифицированные «хакеры»-одиночки, которые приобретают готовый «продукт» и просто используют его в своих интересах.

5.4.7. Антивирусы и принцип их действия

Антивирус – специализированный программный комплекс, способный обнаруживать, блокировать и удалять вредоносные программы. Классификация антивирусов с точки зрения способов обнаружения вредоносных программ изображена на рис. 5.17. Антивирусы могут устанавливаться и на рабочие станции, и на выделенные серверы.

Рис 5.17. Классификация антивирусов

[1] См. понятие «черные шляпы» в разделе «Уязвимости программного обеспечения»

Сканеры – антивирусные программы, которые в поисках вредоносных программ последовательно проверяют все потенциально заражаемые объекты на компьютере, а именно: области оперативной памяти, фрагменты носителей информации, файлы, активные процессы в памяти, ветви и ключи Реестра и пр. Недостаток сканеров: значительное время, затрачиваемое на проверку, – оно в современных условиях может составлять несколько часов.

Резидентные мониторы – антивирусные программы, постоянно находящиеся в памяти компьютера и реагирующие только на потенциально опасные события – запуск программ, создание файлов, изменение ключей Реестра и т.п. Недостаток мониторов: высокие требования к вычислительным ресурсам – к оперативной и дисковой памяти, процессорному времени и прочему. Подчас на компьютере с активным резидентным монитором быстродействие обычных прикладных программ оказывается существенно сниженным.

Вакцинаторы (*иммунизаторы*) – антивирусные программы, видоизменяющие вычислительную среду таким образом, чтобы вредоносная программа отказывалась от работы. Недостаток: способность блокировать работу лишь небольшого количества вредоносных программ.

Диспетчеры изменений – программы, которые сохраняют сведения о программно-аппаратной среде, соответствующие определенному моменту времени, и в дальнейшем извещают обо всех подозрительных изменениях – о появлении новых программ, искажении контрольных сумм файлов и т.п. Недостаток диспетчеров: отсутствие возможности определить вид вредоносной программы и удалить ее.

Фаги (от греческого слова φαγω – поедать) – антивирусы, восстанавливающие исходное состояние зараженных или поврежденных объектов. Обычно этой способностью дополняются «сканеры» и «резидентные мониторы».

Брандмауэры или *файрволлы* (от Brandmauer (нем.) и firewall (англ.) – противопожарная стена) – программные или аппаратные средства, предназначенные для фильтрования сетевого трафика, что позволяет блокировать распространение вредоносных программ по сети.

Методы, при помощи которых антивирусы детектируют (распознают) вредоносные программы и отличают их от «нормальных», различны (см. рис. 5.18).

Рис. 5.18. Методы детектирования вредоносных программ

«*Черные списки*» – самый простой и, исторически, самый первый способ обнаружения вредоносных программ. В первой половине 1980-х годов, когда количество более или менее широко распространенных вредоносных программ не превышало нескольких десятков, достаточно было «вручную» сравнивать все вновь попадающие на компьютер программы с содержимым периодически обновляемого «черного списка», содержащего название и размер файла вредоносной программы. Пример такого списка: «Грязная дюжина» Тома Неффа и Эрика Ньюхауза, 1985 г.

«*Сигнатурный анализ*» – метод обнаружения вредоносных программ, основанный на сравнении участков файла программы с «*сигнатурами*» – фрагментами, характерными для конкретных вредоносных программ и только для них. Например, в качестве сигнатуры для «не-вируса EICAR» (см. выше), может быть использована строка ‘PZX54’, начинающаяся с 13-го по счету байта программного файла. В целях уменьшения баз данных современные антивирусы используют в качестве сигнатур не цепочки байтов, а контрольные суммы от этих цепочек или даже от всей неизменяемой части программного файла. Например, CRC-32 «не-вируса EICAR» есть 32-битовое число $6851CF3C_{16}$, и любой другой файл, имеющий такую же контрольную сумму, можно с высокой вероятностью отождествлять с файлом EICAR.COM. Основной недостаток сигнатурного анализа – невозможность обнаружения новых вредоносных программ, чьи сигнатуры и контрольные суммы еще не внесены в базу данных. Другой недостаток, проявившийся в последнее десятилетие, – чрезмерная громоздкость баз данных, вынужденных хранить записи о миллионах и десятках миллионов вредоносных программ.

«*Анализ поведения*» (или «*проактивный анализ*») – технология обнаружения, основанная на идее виртуализации вычислительной среды. Антивирус помещает каждую выполняемую программу в миниатюрную виртуальную «*песочницу*» (sandbox), и отслеживает последовательность действий, характерную для вредоносных программ. Например, для «файловых червей» (см. выше) характерны попытки доступа к сменным накопителям и создания на них файла «AUTORUN.INF». Антивирус разрешает «безобидные» действия и блокирует «подозрительные». Недостатками «поведенческого анализа» является замедление работы прикладных программ, уменьшение свободных вычислительных ресурсов, ложные срабатывания (например, в случае, если пользователь захочет самостоятельно создать или отредактировать файл «AUTORUN.INF» при помощи стандартной программы «Блокнот»).

«*Эвристический анализ*» – сложная технология обнаружения, основанная на воспроизведении мышления человека, пытающегося отличить «вредоносную» программу от «нормальной». При этом анализируются статистические закономерности, характерные для вредоносного кода; состав и взаимное расположение участков программного файла; типичные для вредоносных программ участки алгоритмов и прочее. Основной недостаток эвристического анализа – «нечеткость» вывода и, как следствие, определенный процент как пропусков, так и ложных срабатываний.

«*Белые списки*» – радикальный способ отличения «нормальных» программ от «вредоносных». Он основан на разрешении работы только ограниченного набора программ и блокировании всех остальных, априори считающихся «нежелательными».

Современные антивирусы, как правило, сочетают при работе разнообразные методы обнаружения вредоносных программ.

Несмотря на множество возможностей по предотвращению деятельности вредоносных программ, антивирусы инородны по отношению к операционной системе и снижают эффективность использования прикладных программ. Основными факторами борьбы с вредоносными программами должны служить встроенные особенности операционных систем:

- отсутствие уязвимостей;
- поддержка политик разграничения доступа;
- средства проверки целостности и подлинности программ;
- средства аутентификации;
- средства виртуализации.

Они должны сочетаться с жесткими политиками распространения прикладного программного обеспечения через доверенные репозитории. Хорошим примером борьбы с вредоносными программами без антивирусов является комплексная система безопасности, проводимая в жизнь компанией Apple для своих продуктов под управлением операционных систем Mac OS X и iOS.

5.5. Защитные средства операционных систем

Подсистема защиты – компонент операционной системы, ориентированный на сохранение конфиденциальности, целостности и доступности выполняемых программ и обрабатываемых данных. Существуют простые операционные системы, обходящиеся без таких подсистем, например MS-DOS. Но все современные операционные системы обязательно их содержат.

Считается необходимым, чтобы защитные возможности операционной системы удовлетворяли некоему общему, универсальному стандарту. Примером такого стандарта является ISO 15408 «Общие критерии защищенности информационных технологий», в основе которого лежит подробно рассмотренная ранее «Оранжевая книга».

5.5.1. Управление доступом

В основе управления доступом лежит модель, рассмотренная ранее в разделе «Методы управления доступом к информации». Эта модель предусматривает, что вся совокупность пользователей, устройств, программ и данных разделяется на два подмножества: 1) пассивных «объектов»; 2) активных «субъектов». Как правило, в операционных системах субъектами являются пользователи и группы пользователей (точнее, программы, выполняемые от их имени), а объектами – файлы программ и данных, устройства ЭВМ, фрагменты оперативной памяти и т.п.

В разделах «Разграничение доступа в MS Windows», «Разграничение доступа в UNIX» и «Смешанная политика безопасности в Android» ранее уже были рассмотрены базовые принципы, лежащие в основе политик разграничения доступа субъектов к объектам. Применяя эти политики, возможно настроить систему разграничения доступа программ и пользователей к файлам и устройствам.

Например, при инсталляции UNIX-подобных операционных систем для всех программных файлов по умолчанию устанавливаются битовые флаги «rwxr-x-r-x» (или в 8-ричной нотации 0755), что не позволяет пользователю, не являющемуся «хозяином», модифицировать и удалять их.

Так же в операционной системе Windows по умолчанию существуют ограничения на модификацию файлов, расположенных в некоторых «системных» каталогах (например, в C:\Windows), на просмотр содержимого некоторых каталогов и т.п. В MS Windows последних версий, в дополнение к традиционной дискреционной политике, включены элементы мандатных политик безопасности. Специальный режим UAC (User Account Control – Управление учетной записью пользователя) снабжает любого пользователя мандатной меткой безопасности с пониженными привилегиями. При этом попытка запустить «нестандартную» программу (например ту, которую пользователь проинсталлировал из дистрибутива или даже сам скомпилировал) вызывает диалог подтверждения полномочий (см. рис. 5.19). После того, как пользователь вручную подтверждает «безопасность» программы, ему временно назначается набор привилегий, достаточный для ее выполнения. Очевидно, этот режим ориентирован на блокирование запуска появившихся извне вредоносных программ, и отключать его не рекомендуется.

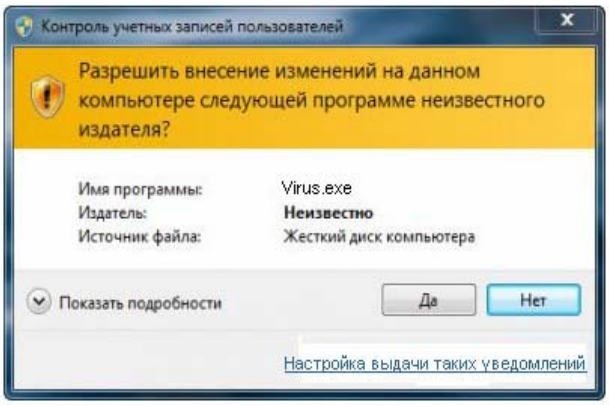


Рис. 5.19. Windows требует подтверждения полномочий на запуск

5.5.2. Аутентификация пользователей

Чтобы работал тот или иной механизм разграничения доступа, каждый субъект или объект должен иметь уникальный идентификатор. В Windows это SID (см. раздел «Разграничение доступа в MS Windows»), а в UNIX метки UID и GID (см. раздел «Разграничение доступа в UNIX»). Файловые объекты наделяются идентификаторами при создании, а субъекты – на завершающем этапе аутентификации, которая производится при входе пользователя в систему.

Строго говоря, операционная система различает «локальных» и «удаленных» субъектов, правила аутентификации для них разные. Здесь будут рассмотрены, преимущественно, вопросы «локальной» аутентификации.

Процедура аутентификации, выполняемая любой операционной системой, стандартна и состоит из трех шагов (см. раздел «Методы и средства аутентификации»):

1) идентификация – получение от пользователя некой информации;

2) аутентификация – проверка подлинности этой информации;

3) авторизация – наделение пользователя определенными правами.

5.5.2.1. Аутентификация в UNIX

Информация о паролях пользователей хранится в текстовом файле «/etc/passwd», каждая строка которого состоит из стандартных полей, разделенных двоеточиями (см. рис. 5.20).

Рис. 5.20. Строка файла «passwd»

В современных системах в поле «хеш пароля» обычно содержится признак «*» (хеш находится в общем файле «/etc/shadow») или «x» (хеш находится в отдельном файле с именем вида «/etc/tcb/имя_пользователя/shadow»). Строки файлов, в которых содержатся хеши паролей, так же имеют стандартный формат (см. рис. 5.21).

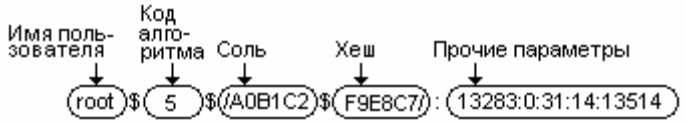


Рис. 5.21. Строка файла «shadow»

Поддерживаются самые разные алгоритмы хеширования: (пусто) – DES; «1» – MD5; «2a» и «2y» – Blowfish (хеширование с ключом); «5» и «6» – SHA длиной 256 или 512 битов и т.п. Иногда группа полей, содержащих код алгоритма, соль и хеш, может содержать признаки: (пусто) – у пользователя нет пароля; «!» – пароль заблокирован; «!!» – пароль не нужен; «LK» или «*» – учетная запись заблокирована. Под «прочими» понимаются параметры, описывающие темпоральные характеристики пароля: срок действия, в какой момент выдавать предупреждение о завершении этого срока и т.п.

Процедура аутентификации в UNIX выполняется гибко настраиваемой подсистемой PAM (от англ. Pluggable Authentication Module – Сменные модули аутентификации). Подсистема поддерживает совместную работу большого числа «модулей» – то есть программ, каждая из которых выполняет определенное элементарное действие: запрос пароля, чтение пароля с клавиатуры, получение аутентификатора от сенсорного экрана, выполнение протокола аутентификации с USB-токеном, вычисление хеш-функции, сравнение с «эталоном», проверка срока действия пароля и т.п. В специальном общем файле «/etc/pam.conf» или отдельных файлах «etc/pam.d/*.*» содержится описание последовательности запуска PAM-модулей, характерной для процедуры аутентификации того или иного пользователя. Изменяя содержимое

этого файла, можно конфигурировать различные сценарии входа в систему: например, разрешить пользователям вводить любой логин и пароль; запретить пользователям входить от имени «root»; разрешить только аутентификацию с применением USB-токена и т.п.

5.5.2.2. Аутентификация в Windows

Информация о паролях пользователей хранится в двоичном файле C:\Windows\System32\SAM, который непосредственно отображается на одну из записей Реестра. В этом файле содержится имена и RID[1] пользователей, а так же хеш пароля. В разных версиях Windows используются оригинальные хеш-функции «LM», «NTLM» и прочие, которые применяют разделение и слияние частей пароля, многократное промежуточное хеширование их криптографическими функциями, шифрование различными методами с ключом, сгенерированным из RID и иные операции.

Процедуры идентификации, аутентификации и авторизации инициируется системным процессом «Winlogon», выполняемым от имени псевдопользователя SYSTEM.

Непосредственно идентификацию и аутентификацию обслуживает процесс «lsass», который для этой цели пользуется услугами «пакетов аутентификации». Это структурные компоненты, оформленные в виде DLL-библиотек и выполняющие процедуру аутентификации в соответствии с тем или иным алгоритмом. Некоторые «пакеты» являются стандартными (например тот, который поддерживает локальную аутентификацию с использованием экрана и клавиатуры), иные могут поставляться вместе с конкретным оборудованием – USB-токенами, устройствами биометрической аутентификации и т.п. Для аутентификации удаленных пользователей существуют «пакеты»:

- «NTLM» – использует протокол, принцип работы которого описан в разделе «Аутентификация с использованием симметричного шифра»;
- «Kerberos» – использует сложный протокол, в основе которого лежит алгоритм, который будет описан далее в главе «Защита в сетях»;
- «Negotiate» – протокол выбора между «NTLM» и «Kerberos»;
- «Digest» – используется для аутентификации веб-приложений (см. далее в теме «Защита в сетях»);
- «Schannel» – поддерживает аутентификацию по правилам протоколов SSL и TLS (см. далее в теме «Защита в сетях»).

Алгоритмы действия «пакетов» не являются жестко фиксированными. В процессе проверки подлинности пароля «пакеты» обязаны учесть многочисленные предварительные условия, поставленные администратором системы, например (см. рис. 5.22): предельный срок действия пароля, максимально возможное число неудачных попыток входа в систему и т.п.

---

[1] Уникальный идентификатор, который, например, для Администратора всегда $1F4_{16}$, для Гостя $1F5_{16}$.

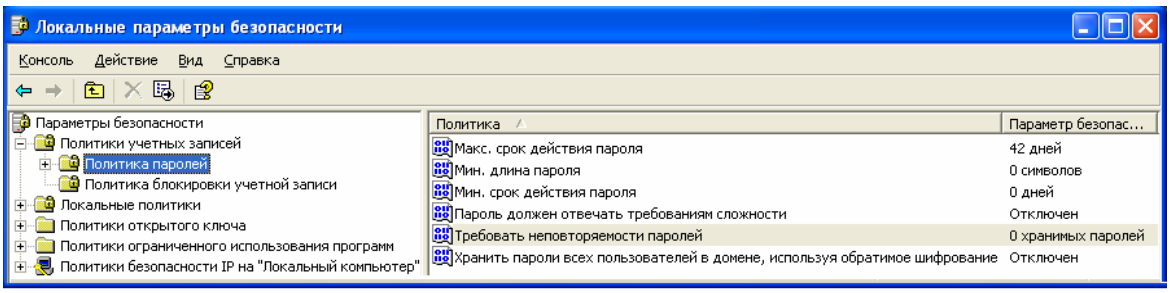


Рис. 5.22. Настройка дополнительных условий

После успешной аутентификации субъекта процесс «lsass» сообщает об этом процессу «Winlogon», который переходит к последнему шагу: инициирует процедуру авторизации – процесс «Userinit». На этом этапе реализуются групповые политики – настраивается рабочая среда Windows для конкретного пользователя, запускается рабочий стол и т.п.

Для младших версий Windows существовали «кибератки», позволявшие восстанавливать пароль из хеша. В настоящее время проблема решена применением более стойких алгоритмов хеширования.

5.5.3. Аудит системных событий

Аудит (или *журналирование*) *системных событий* – процесс регистрации в специальном журнале (текстовом файле, базе данных, syslog-е и т.п.) информации о потенциально «опасных» событиях, таких как удачная или неудачная аутентификация пользователя, создание или удаление файлов, вставка и удаление съемных носителей, запуск или завершение программ и т.п.

Исследование содержимого системного журнала позволяет обнаружить факты и изучить подробности своевременно не распознанных кибератак и иных нарушений.

Так же на аудите событий основан принцип действия «*систем обнаружения вторжений*» (или IDS – Intrusion Detection System). Так называют программные или аппаратные средства, которые не только регистрируют системные события, но и анализируют их в режиме реального времени, пытаясь обнаружить факт кибератаки или иного нарушения в момент возникновения.

Средства аудита событий, встроенные в современные операционные системы, от версии к версии постепенно усложняются, приобретая возможности систем обнаружения вторжений.

5.5.3.1. Аудит в MS Windows

В операционной системе Windows регистрацию системных событий выполняет служба «Журнал событий». События, которые могут быть зарегистрированы, разделены на несколько групп:

1) «приложение» – здесь сохраняется информация о запуске и завершении (в том числе, по ошибке) приложений;

2) «безопасность» – накапливает сведения об удачных и неудачных входах в систему;

3) «установка» – содержит события, связанные с установкой и удалением программ;

4) «система» – содержит сведения об ошибках в драйверах и других компонентах операционной системы;

5) перенаправленные события – накапливает сведения о событиях, произошедших на других узлах сети (если выполнена подписка на получение таких сведений).

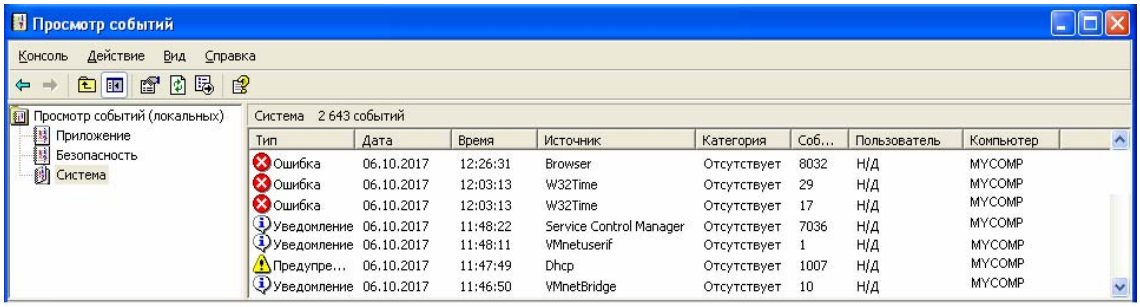


Рис. 5.23 – Просмотр списка событий

Принцип регистрации системных событий в Windows основан на некоторой дискреционной «политике аудита», которая во многом аналогична «политике разграничения доступа» (см. раздел «Разграничение доступа в MS Windows»).

Доступ к просмотру и очистке журнала событий имеют только пользователи, входящие в группу «Администраторы» (см. рис. 5.23). В последних версиях Windows, благодаря опции «Привязать задачу к событию», появилась возможность сопоставить определенным событиям некоторые действия, такие как запуск каких либо программ, вывод предупреждающих сообщений, посылка электронных писем на указанные адреса и т.п.

5.5.3.2. Аудит в UNIX

Компонент регистрации событий был добавлен в ядро Linux, начиная с версии 2.6. Он позволяет отслеживать критичные с точки зрения безопасности события, такие как запуск и завершение работы системы; чтение, запись и изменение прав доступа к файлам; инициация сетевых соединений; попытки неудачной авторизации в системе; изменение сетевых настроек; изменение информации о пользователях и группах; запуск и остановка приложений; выполнение системных вызовов.

Доступ к возможностям аудита обеспечивается отдельно инсталлируемым пакетом «auditd», который включает демон «auditd» и несколько вспомогательных утилит:

- «auditctl» — утилита для управления демоном, которая позволяет получать информацию о текущем состоянии подсистемы аудита, а также добавлять и удалять правила;
- autrace — утилита для аудита событий, порождаемых процессами;
- ausearch — утилита для поиска событий в журнальных файлах;

- aureport — утилита для генерации отчётов о работе системы аудита.

В текстовом файле «etc/audit/auditd.conf» содержатся основные настройки подсистемы и, в первую очередь, имя файла для сохранения логов. В текстовом файле «etc/audit/audit.rules» содержатся описание правил – какие события регистрировать и как на них реагировать.

Информация обо всех событиях сохраняется в текстовом виде и может быть просмотрена как визуально, так и при помощи сервисных утилит, которые облегчают поиск, сортировку, фильтрацию и иные операции над записями.

Для аудита системных событий в FreeBSD и Mac OS X используется демон «syslogd».

5.5.4. Виртуализация вычислительных ресурсов

В соответствии с ГОСТ Р 56938-2016, *виртуализация* – преобразование формата или параметров программных или сетевых запросов к компьютерным ресурсам с целью обеспечения независимости процессов обработки информации от программной или аппаратной платформы информационной системы. Фактически, виртуализация предусматривает предоставление некой исполняемой программе виртуальных (то есть, «не настоящих», смоделированных) вычислительных ресурсов:

- процессора;
- памяти;
- файловой системы;
- внешних устройств и иных ресурсов.

Виртуальная машина – программный комплекс, который создает для исполняемой программы виртуальную среду выполнения. В зависимости от сложности виртуальной машины, «исполняемой программой» для нее может служить даже операционная система.

Таблица 5.3 – классификация виртуальных машин (ВМ)

Тип	**Виртуализуемые ресурсы**				**Примеры**
	Процес-сор	**Память**	**Файловая система**	**Внешние устр-ва**	
Операционная система	Нет	Да	Нет	Нет	Windows, Linux.
Виртуальная среда исполнения программ	Да	Да	Нет	Нет	MS CLR, Mono, JVM
Виртуальная ЭВМ	Да/Нет	Да	Да	Нет	VmWare, Virtual PC, VirtualBox, DosBox, Qemu
«Песочница»	Нет	Да/Нет	Да	Нет	Sandboxie, Mbox
Программа-отладчик	Да	Нет	Нет	Нет	Turbo Debugger, Olly DBG, WinDeb

Гипервизор – «ядро» виртуальной машины. Согласно ГОСТ 56-938-2016, различают два типа гипервизоров:

- гипервизор I типа, представляет собой программу, непосредственно выполняющуюся на оборудовании ЭВМ;
- гипервизор II типа сам выполняется под управлением операционной системы или другой виртуальной машины.

С точки зрения виртуализуемых ресурсов можно выделить следующие типы виртуальных машин (см. табл. 5.3).

В настоящее время концепция виртуализации находит широчайшее применение в сфере защиты информации.

5.5.4.1. Технологии виртуализации

Любая современная операционная система (например, MS Windows или Linux) предоставляет для прикладных программ не настоящие, а виртуальные память и внешние устройства. Благодаря этому программы получают в свое распоряжение «параллельные» адресные пространства и не могут помешать друг другу (см. раздел «Изоляция виртуальных адресных пространств»). Так же адресное пространство программы разделятся на ряд регионов с разными привилегиями доступа, что позволяет защитить от воздействия прикладной программы код и данные операционной системы (см. раздел «Разграничение доступа к оперативной памяти»). Наконец, виртуализация внешних устройств позволяет не только избежать конфликтов при одновременном обращении к ним нескольких прикладных программ, но и заблокировать возможное негативное влияние одних прикладных программ на другие, выполняемое через «общее» оборудование[1].

5.5.4.2. Программы-отладчики

Современные программы-отладчики, встроенные в системы программирования (например, Turbo Debugger) или поставляемые отдельно (например, Olly DBG), позволяют исследовать алгоритмы программ. При этом отладчик считывает из памяти каждую очередную команду программы, анализирует ее и либо эмулирует (моделирует) ее выполнение, либо передает для выполнения процессору. Это позволяет управлять выполнением «отлаживаемой» программы, отслеживать содержимое ячеек памяти и регистров в любой момент времени и т.п.

5.5.4.3. Виртуальные среды выполнения программ

Они реализуют виртуальные процессоры и предоставляют программам ограниченную виртуальную память, благодаря чему обеспечивается «контролируемая» работа этих программ. Любой доступ к ресурсам вычислительной системы со стороны программы осуществляется только путем запроса к среде выполнения, которая вольна либо удовлетворить запрос, либо отвергнуть. Обычно программы, предназначенные для выполнения в

[1] Иногда этого не достаточно. Например, «винлокеры» блокируют мышь и клавиатуру, не обращаясь к ним, как к устройствам, а перехватывая сервисы операционной системы.

виртуальных средах, компилируются не в машинный код, а в некоторый промежуточный байт-код, так что процессор не способен напрямую выполнять команды (см. раздел «Виртуализация вычислительных ресурсов»). Во время работы программы виртуальный процессор считывает из памяти закодированные команды, «на лету» (англ. JIT – Just In Time) анализирует их, преобразует в машинные инструкции и передает для исполнения «физическому» процессору. Таким образом, программа не может ни обратиться к произвольным регионам памяти, ни выполнить «опасные» с точки зрения вычислительной системы команды. Примерами подобных виртуальных сред являются Java-машины (JVM), поддерживающие выполнение программ, написанных на языке Java, и «универсальные среды выполнения» CLR/Mono, поддерживающие выполнение программ, написанных на .NET-языках C#, Visual Basic и пр. Такой же принцип защиты используется в операционной системе Google Android.

5.5.4.4. «Песочницы»

Так называются виртуальные среды, поддерживающие контролируемое выполнение прикладных программ, скомпилированных в машинный код. Основной особенностью «песочниц» является виртуализация не только памяти, но и файловой системы, что позволяет ограничивать программам доступ к файлам и каталогам. В отличие от «блокирования» несанкционированно доступа (что характерно для операционных систем), «песочница» может просто не дать программе «увидеть» запрещенные объекты. Другой подход заключается в том, чтобы любой доступ разрешить, но вместо физических файлов и каталогов предоставить программе их виртуальные модели. После завершения операции результаты «разрешенных» действий переносятся в физическую файловую систему, а «не разрешенные» действия – игнорируются.

Типичным для «песочниц» является применение политик ограничения доступа, основанных на принципе «типичного поведения». Например, поскольку для графического редактора актуальна только работа с файлами изображений (форматы .JPG, .PNG, .GIF и др.), то его доступ ко всем остальным файлам данных может быть запрещен «автоматически»[1].

Защита данных на основе «песочниц» используется в операционной системе Apple iOS. Для иных операционных систем существуют «песочницы» от сторонних производителей, например SandboxIE (для MS Windows) или Mbox (для Linux). Простые «песочницы» включены в состав современных антивирусов – они позволяют «промоделировать» работу подозрительных программ и принять решение об их «вредоносности» или «безобидности».

[1] Для сравнения: текстовый редактор «Блокнот» из MS Windows позволяет загрузить и «испортить» любой, не обязательно текстовый, файл.

5.5.4.5. Виртуальные ЭВМ

Так называются полноценные виртуальные «модели» компьютера, виртуализующие его файловую систему, память, внешние устройства и, иногда, даже процессор. Программные и программно-аппаратные комплексы этого класса позволяют устанавливать и выполнять в виртуальных образах компьютера «гостевые» операционные системы. Очевидно, вредоносная прикладная программа, выполняющаяся на виртуальной машине под управлением такой операционной системы, не сможет распространить разрушающее воздействие ни на другие виртуальные образы, ни на «родительскую» операционную систему и ее приложения (см. рис. 5.24).

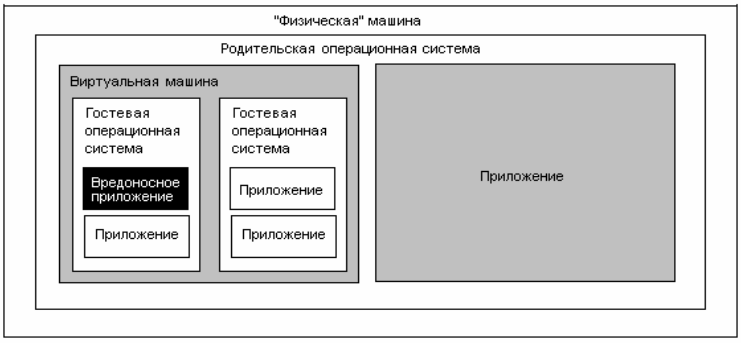


Рис. 5.24. Виртуальные машины

Примерами виртуальных ЭВМ являются VmWare Workstation/Player, Oracle VirtualBox и MS Virtual PC (выполняют приложения на физическом процессоре), а так же DosBox, QEMU (эмулируют машинные команды).

5.5.5. Архитектурные особенности защищенных операционных систем

На защищенность современных операционных систем оказывают влияние некоторые архитектурные особенности построения.

Рис. 5.25. Принцип ASLR

Механизм ASLR (англ. Address Space Layout Randomization – Рандомизация компоновки адресного пространства) предусматривает принудительное придание структуре виртуального адресного пространства случайного характера (см. рис. 5.25). В каждом сеансе работы операционной системы или при каждом новом запуске прикладной программы структура виртуального адресного пространства для этой программы «перемешивается» случайным образом. Это затрудняет работу вредоносным программам, ориентированным на фиксированные адреса памяти и постоянный порядок размещения системных библиотек и фрагментов программы (см., например, понятие «переполнение буфера» в разделе «Уязвимости программного обеспечения»).

Механизм ASLR задействован в MS Windows, Apple iOS, Apple Mac OS X и в ядре Linux.

Механизм DEP (англ. Data Execution Prevention – Предотвращение выполнения данных) или *NX* (англ. No eXecute – не исполнять) – программный механизм, избавляющий операционную систему от некоторых недостатков, вызванных использованием «манчестерской архитектуры» ЭВМ. Современные 64-разрядные процессоры позволяют задействовать в дескрипторах страниц «бит NX», запрещающий выполнение машинного кода. Таким образом, становится возможным блокирование работы вредоносных программ, размещающих свой машинный код в областях стека или данных (см., разделы «Уязвимости программного обеспечения» и «Программные закладки»). Механизм DEP задействован в современных версиях MS Windows, Google Android, Apple Mac OS X и в ядре Linux.

5.5.6. Защита в мобильных операционных системах

Под «мобильными» операционными системами понимаются операционные системы для «мобильных устройств», таких как смартфоны, планшетные компьютеры и т.п. Очевидно, проблемы безопасности программ и данных для таких систем имеют существенные особенности.

Во-первых, для них очень актуальна проблема защиты конфиденциальной информации от физического доступа со стороны «нарушителя». Для мобильного устройства невозможно обеспечить «контролируемую зону» и, более того, чрезвычайно велика вероятность утери такого устройства.

Во-вторых, на мобильные устройства часто возлагаются задачи обработки критически важной конфиденциальной информации, такой например, как атрибуты доступа к банковскому счету пользователя.

Наконец, мобильные устройства используются, преимущественно, неквалифицированными пользователями, не знакомыми даже с азами компьютерной грамотности.

Проблема обеспечения эффективной защиты программ и данных на мобильных устройствах не всегда решается корректно. Например, архитектурные особенности операционной системы Symbian для мобильных устройств марки Nokia в 2004-2007 гг. обусловили возможность развития на них эпидемий «телефонных» червей типа Cabir.

В настоящее время наибольшее распространение на мобильных устройствах имеют операционные системы Google Android и Apple iOS.

5.5.6.1. Аутентификация пользователя

По умолчанию аутентификация пользователей мобильных устройств или отключена, или чрезвычайно упрощена. Однако пользователь имеет возможность активировать процедуру полноценной аутентификации.

На устройстве с iOS необходимо открыть «Настройки» –> «Touch ID и пароль». Если Touch ID не поддерживается, надо перейти в раздел «Настройки»

–> «Пароль». Дальше потребуется нажать «Вкл. код-пароль» и ввести ПИН-код, состоящий из шести цифр. Можно также нажать «Параметры код-пароля», чтобы выбрать либо четырехзначный ПИН-код, либо произвольный ПИН-код либо произвольный буквенно-цифровой пароль. Чтобы подтвердить и активировать пароль, его потребуется ввести дважды. Операционная система iOS запросит пароль при выполнении следующих действий:

- включение или перезапуск устройства;
- разблокировка устройства нажатием кнопки «Домой» (можно изменить);
- обновление программного обеспечения;
- удаление данных;
- просмотр и изменение старого пароля;
- установка профилей конфигурации iOS

Чтобы активировать пароль на устройстве с Android, надо зайти в настройки системы и выбрать в группе «Личные данные» раздел «Безопасность», а далее «Блокировка экрана». После этого можно выбрать способ аутентификации:

- прокрутка пальцем изображения «замочка»;
- «фейсконтроль» – использует фотографию лица;
- графический ключ – требует соединить пальцем минимум 4 точки на экране;
- PIN длиной 4 цифры.
- алфавитно-цифровой пароль.

Пароль потребуется при включении или перезагрузке устройства.

5.5.6.2. Контроль за распространением приложений

Важнейшим принципом обеспечения защиты программ и данных на мобильных устройствах является контроль за распространением прикладных программ.

Рис. 5.26. Apple iOS блокирует установку «чужих» приложений

В операционной системе iOS разрешена установка лишь тех программ, которые подписаны приватным ключом фирмы Apple и получены с ее официальных репозиториев (например, App Store) – см. рис. 5.26. Все программное обеспечение, размещенное в этих репозиториях, протестировано (и «вручную», и в полуавтоматическом режиме) специалистами фирмы Apple. Это программное обеспечение создано либо самой фирмой Apple, либо независимыми производителями, имеющим лицензию iOS Developer и, соответственно, цифровой сертификат, удостоверенный в Apple.

Политика фирмы Google внешне напоминает политику фирмы Apple, а именно: настоятельно рекомендуется устанавливать лишь приложения,

полученные из официальных репозиториев, таких как Google Play или Android Market. Однако при установке программного обеспечения проверяется цифровая подпись производителя, не обязательно связанного с Google. Такая проверка гарантирует лишь целостность установленной программы, но никак не защищает от потенциальных программных закладок, внедренных в нее производителем или «злоумышленником» в результате «пересборки». Непосредственный контроль со стороны Google за содержимым Google Play так же отсутствует, программы тестируются неформальным сообществом «любителей Android» и автоматическим средством «Bouncer».

Таким образом, обеспечивается очень жесткий и очень эффективный контроль за безопасностью всех устанавливаемых в операционной системе iOS приложений. Контроль же за распространением приложений для операционной среды Android довольно формален и не всегда гарантирует отсутствие в них программных закладок. Однако отсутствие жесткого контроля стимулирует возникновение новых коллективов разработчиков и создание нового, самого разнообразного программного обеспечения для Android, привлекающего все новых и новых пользователей. По приблизительным оценкам количество пользователей Android в 3-4 раза больше количества пользователей iOS.

5.5.6.3. Шифрование данных

Операционная система iOS зашифровывает методом AES с 256-битовыми ключами: 1) все файлы на системном носителе (каждый файл своим случайным ключом!); 2) все случайные ключи, использованные для шифрования файлов; 3) кроме этого, все содержимое носителя, включая как уже зашифрованные файлы и сохраненные зашифрованные ключи, так и неиспользуемые области. Ключи генерируется закрытым алгоритмом на основе пользовательского пароля и уникального идентификационного номера устройства. В процессе работы информация «на лету» расшифровывается и зашифровывается.

О надежности подобной защиты красноречиво говорит история, получившая широкую огласку в 2016 г. ФБР столкнулась с необходимостью получить доступ к конфиденциальной информации, хранящейся на смартфоне террориста, но фирма Apple отказалась раскрыть подробности алгоритма генерации ключей. Не справившись с проблемой силами своих специалистов, ФБР сначала подала на Apple в суд, но в последний момент забрала иск и для получения ключей воспользовалась услугами неназванной «третьей стороны». В последующих версиях операционной системы iOS фирма Apple изменила алгоритм генерации ключей.

Операционная система Android версий до 5.0 могла[1] шифровать только конфиденциальные данные пользователя (атрибуты аккаунта Google, электронные письма, SMS-сообщения, списки контактов, изображения и т.п.), а остальные файлы оставляла в неизменном виде. Начиная с версии 5.0, на устройствах с 64-битовым ARM-процессором возможно шифрование всего

[1] На многих конкретных устройствах даже эта возможность была недоступна.

системного носителя, причем 128-битовые ключи генерируются на основе пользовательского пароля и 128-битовой «соли». Ключ хранится в «аппаратном хранилище» Keymaster, привязанном к конкретному устройству. Как и в случае с iOS, в процессе работы информация «на лету» расшифровывается и зашифровывается.

Резюмируя, и iOS и Android обладают эффективными криптографическими средствами, блокирующими «злоумышленнику» несанкционированный доступ.

5.5.6.4. Виртуализация ресурсов

Защита данных от приложений и одних приложений от других в мобильных операционных системах реализуется на основе технологий виртуализации ресурсов.

В основе операционной системы Android лежит ядро Linux Kernel. Но файловая система и дискреционная политика разграничения доступа, реализованные ядром Linux, при выполнении прикладных программ практически не актуальны. Все прикладные программы запускаются в независимых, изолированных друг от друга копиях JVM – виртуальных сред выполнения для программ, написанных на языке Java. В старых версиях Android использовалась среда Dalvik, в современных – CLR. Каждое приложение декларирует набор необходимых ему системных операций (типа «отправка SMS», «прием телефонных звонков» и т.п.) в специальном «модуле манифеста». Любое действие, не упомянутое в «манифесте», блокируется виртуальной средой.

Кроме того, ряд системных операций считаются «опасными», и при попытке их выполнить у пользователя спрашивается разрешение. «Опасными» считаются: 1) просмотр и изменение событий в календаре; 2) использование видеокамеры; 3) чтение и изменение контактов; 4) доступ к примерному (через точку подключения к Wi-Fi) и точному (через GPS) местоположению; 5) запись с микрофона; 6) операции с телефоном – звонки, чтение списка вызовов и т.п.; 7) доступ к датчикам и сенсорам (например, к пульсомеру); 8) операции с SMS и MMS – отправка, получение, просмотр списка полученных и отправленных; 9) чтение и запись на внешнюю память.

Помимо системных операций каждое приложение декларирует «специальные права», например: «стать приложением для отправки SMS», «отображать свое окно поверх других», «стать администратором устройства» и пр. Пользователь может (и должен!) при установке нового приложения контролировать затребованные им «опасные операции» и «специальные права» и, при несоответствии «типичному поведению», отказаться от установки (см. рис. 5.27). К сожалению, это простое правило сплошь и рядом не соблюдается, что приводит к массовому распространению на устройствах с операционной средой Android вредоносных программ.

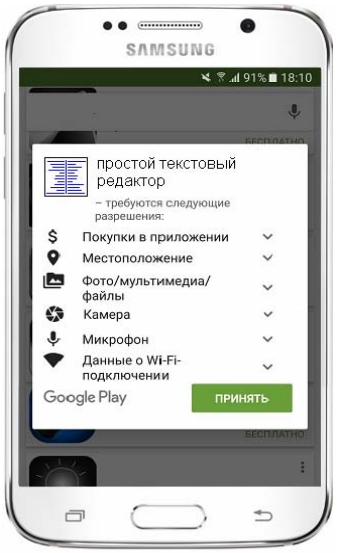


Рис. 5.27. Установка «подозрительного» приложения на Android

В основе операционной системы iOS лежит UNIX-подобное ядро Darwin, но характерная для нее дискреционная политика разграничения доступа практически не используется. Все приложения – и встроенные, и установленные пользователем – запускаются в «песочнице», ограничивающей их доступ с «чужим» ресурсам. Например, у каждого приложения есть «домашний» каталог, в котором хранятся его файлы, а доступ к каталогам других приложений невозможен. Так же «песочницы» строят для каждого приложения собственное адресное пространство, а доступ к памяти других приложений невозможен. Невозможно и непосредственное обращение к аппаратным ресурсам, оно осуществляется только через Cocoa API. Поскольку наборы ограничений для разных приложений отличаются, они описаны в файлах системного каталога «/System/Library/Sandbox/Profiles». В отличие от Android, в системе iOS не предусмотрено участие пользователя в распределении тех или иных прав доступа к тем или иным ресурсам, что существенно ограничивает возможность работы вредоносных программ.

К сожалению, и в iOS, и в Android постоянно обнаруживаются уязвимости. Кроме того, для обеих операционных систем существует практика нелегального отключения большинства защитных возможностей[1], что разрешает установку любых (а не только подписанных) приложений, открывает приложениям полный доступ к файловой системе и т.п. С одной стороны, это существенно расширяет вычислительные возможности устройства, с другой – открывает «злоумышленникам» практически неограниченный доступ к обрабатываемой (в том числе и конфиденциальной!) информации.

ТЕМА 6. Защита в компьютерных сетях

Компьютерная сеть (или *вычислительная сеть*) – распределенная в пространстве программно-аппаратная структура, предназначенная для обеспечения информационного взаимодействия своих *узлов*, в качестве которых могут выступать компьютеры, мобильные «гаджеты» (планшеты и телефоны), устройства офисной и бытовой техники (принтеры, видеокамеры и т.п.). Так же узлами сети могут служить «коммуникационные» устройства, предназначенные не для обработки данных, но исключительно для поддержания и обслуживания

[1] Для Android это «рутование», для iOS – «jailbreaking» («побег из тюрьмы»).

этой инфраструктуры, а именно: коммутаторы, маршрутизаторы, «точки доступа» и пр.

По степени удаленности друг от друга узлов различают *локальные сети* и *глобальные сети*. Глобальная *сеть Интернет* является объединением множества всевозможных локальных и более мелких глобальных сетей.

Сети типа «*интранет*» – глобальные сети предприятий, организаций и ведомств, изолированные от Интернета. Примеры: NMCI/NGEN – военная сеть США; ЗСПД (Закрытый сегмент передачи данных) — военная сеть РФ.

Конечные узлы сетей часто называются *хостами*, а узлы, служащие соединителями между подсетями – *маршрутизаторами* (или *роутерами*). Маршрутизаторы, соединяющие разнородные сети называют *шлюзами*.

Для того, чтобы обращаться к конкретному узлу сети, каждый узел должен иметь «*адрес*» (уникальный идентификатор).

Физический адрес – идентификатор устройства, в направлении которого передается информационный сигнал (например, поток битов по проводу). Обычно имеет постоянный характер. Пример: 48-битовый MAC-адрес устройства, имеющий вид 12:23:31:32:13:21.

Логический адрес – идентификатор, под которым узел известен другим узлам, программам и людям. Этот тип адресов к устройству жестко не привязан. Примеры: IP-адрес вида «111.222.100.200» или NETBIOS-адрес вида «COMP123».

Локальный логический адрес – идентификатор, под которым узел известен в «своей» локальной сети.

Глобальный логический адрес – идентификатор, под которым узел известен вне «своей» сети. Локальный и глобальный адреса не обязательно совпадают.

Так же в составе сетей могут встретиться коммуникационные устройства, обеспечивающие их работу, но не являющиеся узлами, так как не имеют собственного адреса, например (см. рис. 6.1):

- *повторители* (или *репитеры*) – устройства, включенные в линию связи, которые создают и пересылают дальше «копию» входного информационного сигнала, что позволяет бороться с его (сигнала) затуханием;
- цифровые *концентраторы* (или *хабы*) – устройства, пересылающие входной информационный сигнал сразу на все узлы, подключенные к выходным линиям;
- цифровые *коммутаторы* (или *цифровые ключи*, или «*свитчи*») – устройства, пересылающие информационный сигнал с единственного своего входа на определенный узел, подключенный к одной из выходных линий;
- точки доступа – устройства, согласовывающие разные формы представления сигнала, например, электрического (по проводам) с электромагнитным (по радиоэфиру).

а) Репитер б) Хаб в) Свитч г) Шлюз д) Шлюз + свитч е) Точка доступа

Рис. 6.1. Условные обозначения на схемах

Современные коммуникационные устройства могут сочетать различные возможности, например, служить одновременно и «шлюзом», и «свитчем». Роль коммуникационного устройства может выполнять обычный компьютер с соответствующим программным обеспечением.

Наиболее часто встречаются следующие варианты сетевых топологий.

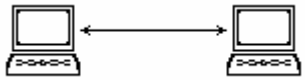
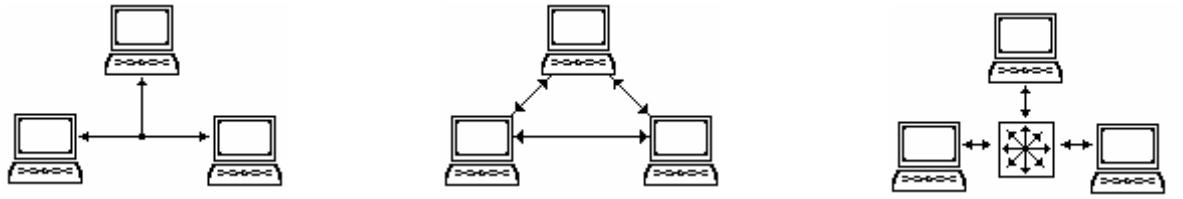


Рис. 6.2. Соединение «точка-точка»

1. *Точка-точка* (см. рис. 6.2.). Два хоста передают информационные сообщения непосредственно друг другу.

Для организации такой сети требуется простейшее оборудование (например, коммуникационные порты в стандарте RS-232 на каждом компьютере и кабель из трех проводов) и простейшее программное обеспечение (например, MS Windows поддерживает такую сеть в рамках режима «Прямое кабельное соединение»). Индивидуальные адреса хостам не требуются.



а) Каждый с каждым б) Кольцеобразная шина в) Звезда

Рис. 6.3. Варианты топологии локальных сетей

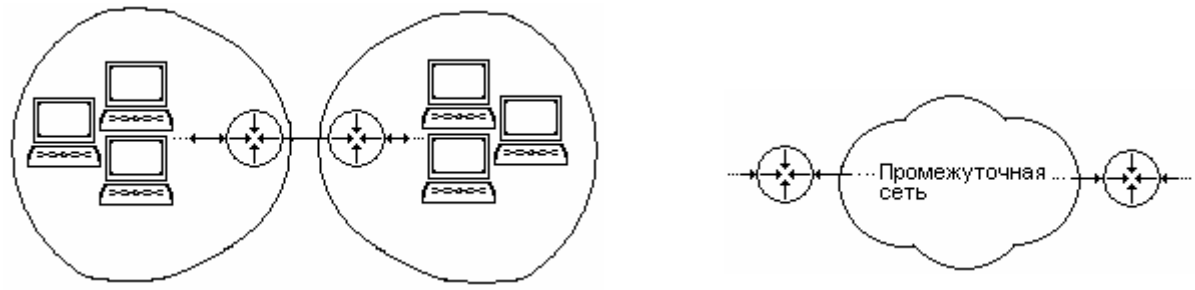
2. Локальная сеть типа «каждый с каждым» (или «пассивная» звезда) – см. рис. 6.3, а. В таких сетях все хосты равноправны – это, так называемые, «*одноранговые*» *сети*. Каждый из хостов передает сообщения всем остальным сразу. «Правильный» адресат принимает сообщение, остальные игнорируют. Для организации такого обмена каждый хост должен иметь свой адрес.

3. Локальная сеть типа «шина» (см. рис. 6.3, б). Сообщения в такой сети передаются последовательно – от хоста к хосту, пока не достигнут адресата. В такой сети каждый хост тоже должен обладать уникальным адресом.

4. Локальная сеть типа «звезда» или «активная звезда» (см. рис. 6.3, в). В таких сетях все хосты равноправны. Первоначально сообщения передаются в направлении выделенного устройства (например, коммутирующего маршрутизатора), которое дальше пересылает их адресату. И выделенные устройства, и хосты также должны обладать уникальными адресами, а в заголовке сообщения должны указываться два адреса: 1) адрес выделенного устройства; 2) адрес компьютера, которому предназначено сообщение.

5. Реальные сети представляют собой комбинации этих топологий. Чаще используются комбинации «активных звезд», образующие «дерево».

Возможны различные способы доступа узлов одной сети к узлам другой.



а) Непосредственное подключение б) Удаленный доступ

Рис. 6.4. Связь локальных сетей

1. Непосредственное подключение сетей (см. рис. 6.4, а). В конфигурациях, когда хосты одной сети посылают сообщения хостам другой, доступ осуществляется через маршрутизаторы. Каждое сообщение должно содержать в заголовке: 1) локальный адрес своего маршрутизатора; 2) глобальный адрес компьютера другой сети, кому предназначено сообщение.

2. *Удаленный доступ* к сети (см. рис. 6.4, б). Особенностью является организация доступа посредством некой промежуточной сети. В этом случае используют «туннелирование», то есть временное помещение передаваемого из сети в сеть сообщения внутрь некоторого другого «сообщения-конверта», имеющего формат, характерный для промежуточной сети. Фактически это означает организацию «*виртуальной частной сети*» (англ. VPN – Virtual Private Network). Подробней технологии VPN будут рассмотрены далее.

а) Принцип доступа б) Модем в) Wi-Fi роутер

Рис. 6.5. Организация коммутируемого доступа

3. *Коммутируемый доступ* к сети. Это способ временного включения внешнего хоста в состав локальной сети, чтобы воспользоваться ее ресурсами (принтером, базами данных, выходом в Интернет и т.п.). При этом хосту назначается «одноразовый» локальный адрес. Как правило, для установки подключения требуется аутентификация, то есть предъявление своих прав на такое подключение. На физическом уровне подключение часто производится через некоторое специализированное устройство, а именно: по проводам через «модем» (см. рис. 6.5,б) или по радиоэфиру через «точку доступа к Wi-Fi» (см. рис. 6.5,в).

«*Модем*» (сокр. от «модулятор-демодулятор») – устройство, позволяющее преобразовывать «цифровые» сигналы в «аналоговые» (и наоборот), чтобы передавать их по телефонным проводам или радиоканалу (см. рис. 6.5,б).

«*Точка доступа к Wi-Fi*» – приемопередатчик радиосигналов беспроводной сети, снабженный интерфейсом к проводной сети. Часто он является частью специализированного «Wi-Fi роутера» (см. рис. 6.5, в) или компьютерного «Wi-Fi адаптера».

Для передачи данных между узлами сетей повсеместно используются специальные протоколы.

Сетевой протокол – совокупность правил и алгоритмов, согласно которым элементы сетевой архитектуры взаимодействуют друг с другом. Различные протоколы регламентируют большое количество различных аспектов: от порядка посылки отдельных битов по линиям связи до правил оформления сообщений электронной почты. Справочные сведения по протоколам и отдельным их аспектам можно найти в системе документов *RFC* (англ. Request for Comments – Тема для обсуждения), которая свободно доступна в сети Интернет.

Обычно задачи, которые решаются при передаче данных в сетях, образуют иерархию из 7 уровней[1] (см. рис. 6.6):

- 1 – физический уровень (характеризует кабели, разъемы, уровни электрических сигналов и т.п.);
- 2 – канальный уровень (характеризует аппаратные адреса устройств, правила передачи отдельных битов и групп битов, методы обнаружения и исправления ошибок);
- 3 – сетевой уровень (характеризует логическую адресацию устройств в сети и способы определения маршрутов между ними);
- 4 – транспортный уровень (характеризует способы передачи данных по маршрутам);
- 5 – сеансовый уровень (характеризует права отдельных узлов сети на обмен данными);
- 6 – уровень представления данных (характеризует правила кодирования и сжатия передаваемых данных);
- 7 – уровень приложений (характеризует правила предоставления услуг пользователю и описывает системы обмена файлами, сообщениями электронной почты и т.п.).

Порции информации в сетях передаются в виде блоков, которые на каждом уровне называются по-разному (см. рис. 6.6), но имеют похожий формат, а именно: они состоят из 1) служебного заголовка; 2) собственно набора данных (это так называемый «*blob*» – массив двоичных данных).

[1] Эта иерархия закреплена в международном стандарте OSI/ISO «Базовая эталонная модель взаимодействия открытых систем». Здесь ISO – аббревиатура Международной организации по стандартизации.

Рис. 6.6. Семиуровневая модель OSI/ISO

Во время передачи текущий протокол просто добавляет к блоку, полученному от предыдущего протокола, свой заголовок и рассматривает все вместе как свои данные, – этот процесс называется «*инкапсуляцией*» блока данных (см. рис. 6.7). Во время приема разбор происходит в обратном порядке, поэтому совокупность совместимых протоколов различных уровней называется стеком протоколов.

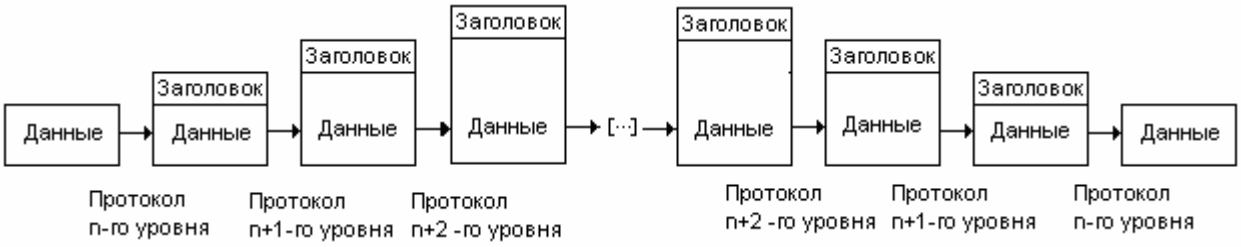


Рис. 6.7. Инкапсуляция блоков данных

Задача защиты информации актуальна на всех уровнях иерархии сетевых протоколов, но ставится и решается в них по-разному.

6.1. Защита информации на физическом уровне

Технологии физического и канального уровня совместно определяют так называемый «*сетевой интерфейс*» – комплекс технических и программных средств, а так же соответствующих им протоколов, предназначенный для обеспечения передачи информации между устройствами. Основными элементами определения интерфейса являются:

- технология и среда передачи информационного сигнала;
- протокол передачи битов.

На физическом уровне OSI/ISO рассматриваются вопросы передачи информационного сигнала от устройства-источника к устройству-приемнику через определенную физическую среду. Наиболее распространены следующие среды (см. табл. 5.1).

Табл. 5.1. Среды передачи сетевых сигналов

Среда	Технология	Типичное расстояние	Защищенность от ПЭМИН
Металлический проводник	Телефонный кабель	Сотни метров	Низкая
	Экранированный коаксиал	Десятки метров	Низкая/средняя
	Витая пара	Десятки метров	Низкая/средняя
Стекло, пластик	Оптоволокно	Десятки километров	Очень высокая
Радиоэфир	Сотовая связь	Километры	Отсутствует
	Wi-Fi, BlueTooth, IrDa	Десятки метров	Отсутствует
	Спутниковая связь	Тысячи километров	Отсутствует

Так же предметом рассмотрения на физическом уровне являются технические аспекты передачи сигнала. Например, для системы передачи по металлическому проводнику должны быть согласованы уровни электрических сигналов, длина и количество линий в кабеле, наличие или отсутствие заземления, геометрия разъемов и т.п.

Основной источник угроз качеству информации на физическом уровне – побочные электромагнитные излучения и наводки (ПЭМИН). Они могут приводить к несанкционированному «прослушиванию» информационного сигнала, к внесению в него искажений и к полной его блокировке.

Наиболее остро проблема ПЭМИН стоит в случае применения радиоэфира в качестве среды передачи информационного сигнала. Одним из подходов к решению проблемы можно считать «частотное *скремблирование*» – изменение частоты, на которой передается информационный сигнал, по псевдослучайному закону, известному и передатчику, и приемнику. Это делается как для обеспечения совместной работы большого числа передатчиков и приемников в ограниченном диапазоне частот, так и для сокрытия передаваемых данных от потенциального прослушивающего устройства.

В случае применения для передачи сигнала проволочных кабелей основным средством борьбы с ПЭМИН является экранирование проводника. Например, для витой пары существуют стандарты, которые предусматривают как экранирование всего кабеля, так и каждой отдельной пары проводов, а кроме того – предусматривающие сочетание обоих способов.

Максимальная защита от ПЭМИН обеспечивается в случае применения оптоволоконных кабелей.

6.2. Защита информации на канальном уровне

Разработаны и используются многочисленные протоколы канального уровня. Их общее назначение – корректная передача отдельных битов или групп битов от «источника» к «приемнику» через некоторую физическую среду передачи данных. При этом каждое устройство, участвующее в информационном обмене, должно иметь уникальный «аппаратный» адрес, жестко привязанный к устройству. Наиболее распространенным типом таких адресов являются 48-битовые «MAC-адреса» (от англ. Media Access Control – Управление доступом к среде). Пример MAC-адреса: «00:24:23:56:D4:11».

Задачами протоколов канального уровня являются: 1) обмен данными с сетевым уровнем OSI/ISO; 2) разбиение потока битов данных на «кадры»; 3) обнаружение и исправление ошибок передачи путем расчета и проверки

контрольных сумм и «синдромов»; 4) отслеживание физической топологии сети; 5) обеспечение правильного порядка передачи и приема кадров; 6) обнаружение и исправление коллизий и прочее. Иногда протоколы разбиваются на два уровня: 1) LLC (решают задачи 1, 2 и 3 для сетей типа «точка–точка») и 2) MAC (решают задачи 4, 5 и 6, возникающие в сложных сетях).

Группы битов, передаваемых канальными протоколами, оформляются в виде «кадров», типичный формат которых изображен на рис. 6.8. Реально используемые протоколы могут использовать более или менее полный вариант этого формата.

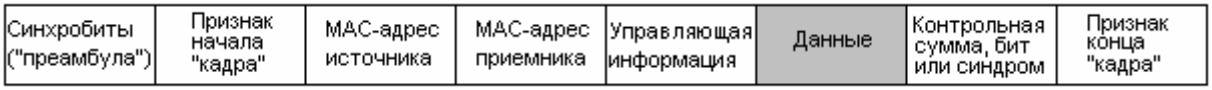


Рис. 6.8. Типичный формат «кадра» передачи данных

Примеры протоколов канального уровня и связанных с ними проблем.

1. *Стандарт RS-232* охватывает функции физического и канального уровня. Он описывает технологию низкоскоростной (до 115Кбит/с) передачи битов между двумя устройствами в режиме «точка-точка» по «телефонному» кабелю. Эти биты передаются группами по 5-8 штук, контроль целостности осуществляется при помощи дополнительного бита четности. Для обмена данными в противоположных направлениях могут быть использованы как отдельные линии синхронизации, так и условные сигналы, передаваемые по линиям данных: XON (11_{16}) – готов к передаче; XOFF (13_{16}) – не готов. По умолчанию технология RS-232 позволяет строить сети типа «точка-точка» (см. рис. 6.2), однако, если на компьютерах установлены несколько коммуникационных портов, то возможно построение сетей типа «шина» (см. рис. 6.3,б). Операционные системы MS Windows поддерживают удаленный доступ компьютеров к сети через RS-232-адаптеры[1], в режиме «Прямого кабельного соединения». Обмен данными между компьютерами, на которых установлены разные операционные системы, может быть организован при помощи универсальной программы KERMIT. Также стандарт RS-232 и родственные ему стандарты RS-422/485 (с канальными протоколами Profibus, Modbus, CAN и пр.) часто используется для организации «промышленных» сетей.

2. *Протокол SLIP* работает в режиме «точка-точка», он разработан для поддержки коммутируемого доступа к сети (см. рис. 6.5,а), имеющей IP-адресацию узлов. Протокол позволяет компьютеру-клиенту инкапсулировать пакет данных, полученный с сетевого уровня OSI/ISO, в свой кадр и передать этот кадр компьютеру-серверу, который извлечет пакет и отправит его компьютеру своей сети, имеющему указанный IP-адрес. Протокол SLIP не проверяет целостность передаваемых данных.

[1] Такие устройства называются «COM-портами» или «AUX-портами». В отечественных стандартах «Стык-2».

3. *Протокол PPP* работает в режиме «точка-точка», он разработан для поддержки коммутируемого доступа к сети (см. рис. 6.5,а), имеющей адресацию в форматах IP, IPX или NetBIOS. Протокол позволяет компьютеру-клиенту инкапсулировать пакет данных, полученный с сетевого уровня OSI/ISO, в свой кадр и передать этот кадр компьютеру-серверу, который извлечет пакет и отправит его компьютеру своей сети, имеющему указанный адрес. Протокол PPP проверяет целостность передаваемых данных путем расчета CRC-16.

Общими недостатками протоколов RS-232, SLIP и PPP являются:

- невозможность аутентификации при обращении клиента к серверу удаленного доступа;
- невозможность обеспечения конфиденциальности информации, передаваемой между клиентом и сервером удаленного доступа.

4. *Протоколы PPTP и L2TP* работают совместно с протоколом PPP, исправляя его недостатки. Принцип их действия основан на организации защищенных «туннелей» сквозь любую «промежуточную» сеть (см. рис. 6.4,б). Эти протоколы инкапсулируют кадры формата PPP, сформированные на клиенте, в свои кадры и осуществляют их передачу в направлении сервера удаленной сети только в том случае, если удачно завершилась аутентификация по методу CHAP (см. раздел «Аутентификация с использованием симметричного шифра»).

Протокол PPTP разработан Microsoft и предполагает, что удаленная сеть, к которой обеспечивается доступ, имеет IP-адресацию узлов и содержит компьютеры, на которых установлена MS Windows. Этот протокол шифрует содержимое своих кадров. Ключ шифрования генерируется из пароля, известного и источнику, и приемнику, а метод (RC4, DES, AES и т.п.) выбирается на этапе установки соединения.

Протокол L2TP разработан фирмой Cisco и предполагает, что удаленная сеть, к которой обеспечивается доступ, имеет адресацию в форматах IP, IPX или NetBIOS. Однако в этом протоколе не предусмотрено шифрование трафика. Предполагается, что оно должно выполняться протоколами более высокого уровня.

Итак, протоколы PPTP и L2TP обеспечивают «туннелирование», то есть прохождение «приватного» трафика через «промежуточную сеть» таким образом, что она не имеет доступа к содержимому этого трафика (см. рис. 6.4,б).

5. *Технология Ethernet*, охватывающая физический и канальный уровни OSI/ISO, в настоящее время считается основным способом организации локальных сетей с топологиями, более сложными, чем «точка-точка». Предполагается, что каждый узел сети имеет уникальный 48-битовый MAC-адрес. Иногда утверждается, что каждое сетевое устройство в мире имеет жестко «прошитый» уникальный MAC-адрес, но в реальности современные операционные системы позволяют протоколу Ethernet использовать не «настоящий» физический адрес, но адрес, программно заданный пользователем (см. рис. 6.10).

Рис. 6.9. Ethernet-плата с разъемами под витую пару и коаксиал

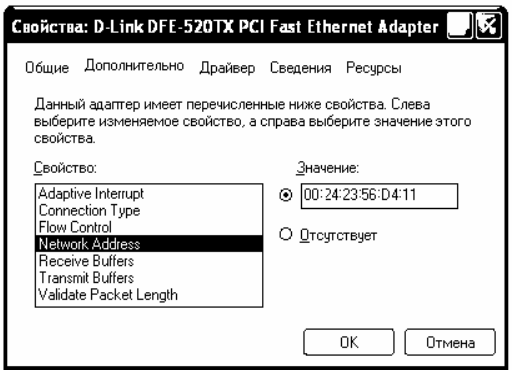


Рис. 6.10. Пример смены MAC-адреса для Ethernet-платы

Это обстоятельство позволяет выполнять подмену устройств и, таким образом, подключение к сети ложных узлов[1], например, для прослушивания трафика. Целостность передаваемых данных в протоколе Ethernet обеспечивается путем расчета и проверки CRC-32. Дополнительные функции, такие как аутентификация узлов или шифрование данных, не предусмотрены.

6. *Технология Wi-Fi* (англ. Wireless Fidelity – Беспроводное качество) широко используется для организации «коммутируемого» беспроводного доступа рабочих станций к локальным сетям (см. рис. 6.5,а). На физическом уровне ISO/OSI, согласно комплексу стандартов «802.11», передача информации может происходить по радиоканалу (где выделены два частотных диапазона) или посредством отражения инфракрасного излучения, направляемого в потолок. Функциями физического уровня является обнаружение несущей частоты, выбор устройства-«собеседника» (если их несколько), оценка «чистоты» радиоканала и передача модулированного сигнала.

На канальном уровне ISO/OSI во время кодирования данных предпринимаются серьезные меры для борьбы с потенциальными искажениями, например: контроль целостности при помощи CRC-32; «скремблирование» (то есть, псевдослучайное перемешивание) передаваемых битов для недопущения «длинных» ошибок; обмен синхронизирующими служебными сигналами; фиксирование моментов начала и окончания передачи кадров с целью определения состояния радиоканала (занят, свободен, произошла ошибка) и т.п.

Дополнительные функции, такие как аутентификация узлов или шифрование данных, технологией W-Fi не предусмотрены.

6.3. Квантовая криптография

Квантовая криптография[2] – группа перспективных методов передачи данных по оптоволоконному каналу связи, обеспечивающих целостность и

[1] Это так называемый *MAC-спуфинг* (от англ. to spoof – мистифицировать).

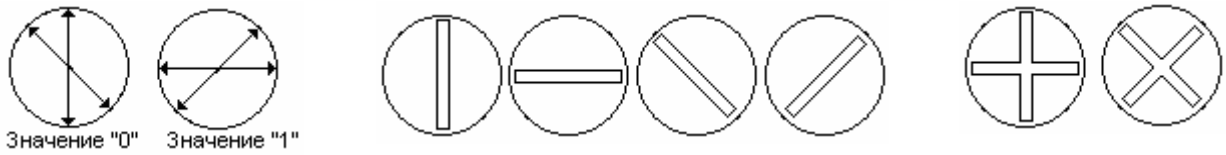
[2] Не имеет ничего общего ни с просто «криптографией», ни с «квантовыми вычислениями» на «квантовых компьютерах»!

конфиденциальность на основании «квантовых» свойств материи. Эти методы охватывают физический и канальный уровни сетевой модели OSI/ISO.

Рис. 6.11. Схема «квантовой» связи

Схема связи, соответствующая этим методам, изображена на рис. 6.11. Здесь «злоумышленник» – условная сущность. «Злоумышленник» может либо отсутствовать, либо присутствовать, и методы квантовой криптографии как раз и позволяют определить, имел ли место факт попытки подслушивания «квантового» канала связи.

Принцип работы «квантовой» связи заключается в следующем. Луч света, испускаемого светодиодом, состоит из волн, имеющих всевозможную поляризацию (то есть, ориентацию совершаемых колебаний в пространстве). Однако пропуская его через некоторые кристаллические среды (например, через пластинки турмалина), можно отфильтровать «лишние» волны, оставив только те, которые имеют желаемую поляризацию. Очень приближенно такой фильтр можно представить себе в виде узкой прорези, пропускающей только световые волны, ориентированные в пространстве так же, как и прорезь (см. рис. 6.12, б).



а) Кодирование битов б) Фильтры-поляризаторы в) Фотодетекторы

Рис. 6.12. Кодировании и измерение битов

Фотодетектор, то есть устройство для измерения угла поляризации волн, так же устроен на основе соответствующим образом ориентированного кристалла турмалина. Таким образом, если поляризующий фильтр и фотодетектор, расположенные после источника света, ориентированы одинаково, то на выходе у фотодетектора будет виден свет, иначе – нет. В соответствии с технологиями квантовой криптографии, фотодетекторы спроектированы так, чтобы не давали однозначного ответа (см. рис. 6.12,в). Так же неоднозначно спроектирована и система кодирования битов. Например, пусть углам поляризации 0º и 135º соответствует значение «0», а углам 90º и 45º значение «1». Волны с иной ориентацией отфильтровываются и в передаче не участвуют. Легко составить таблицу соответствия измеренных значений при разных вариантах поляризующих фильтров и фотодетекторов (см. табл. 5.2).

Чтобы принимающая сторона, в отличие от подслушивающего «злоумышленника», достоверно определила значение переданного бита, используется следующий протокол.

Табл. 5.2. – Соответствие принятых битов посланным

Бит	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**0**
Послано	|	–	/	\	|	–	/	\
Ф/детектор	+	+	+	+	×	×	×	×
Получено	1	1	0	0	0	0	1	1

Шаг 1. Передатчик, управляя поляризующим фильтром, кодирует бит в соответствии с рис. 6.12,а и посылает в направлении Приемника. При этом Передатчик случайным образом выбирает вариант кодирования. Например, для передачи «0» могут быть посланы либо «|», либо «\».

Шаг 2. Приемник, не зная, какой бит послан и как он закодирован, так же выбирает вариант фотодетектора случайным образом в соответствии с рис. 6.12,в.

Шаг 3. Получив некоторое значение бита и не зная, верно оно или нет, Приемник по открытому каналу сообщает Передатчику, какой именно вариант фотодетектора он использовал. Значение же полученного бита сохраняется в секрете.

Шаг 4. Передатчик, сверившись с таблицей 5.2, сообщает Приемнику по открытому каналу, «правилен» ли выбор Приемника. Например, для передаваемых «/» или «\» Приемник должен был использовать фотодетектор «×» – только в этом случае он получил бы достоверный результат.

Шаг 5. Если выбор был «правильным», то Приемник сохраняет значение бита, иначе – отбрасывает. Далее, процедура многократно повторяется, начиная с Шага 1, пока Передатчик не передаст, а Приемник не получит все запланированные биты.

Подобная схема передачи битов позволяет так же определить, имел ли место факт «подслушивания». Дело в том, что любой фотодетектор, если сигнал передается далее, неизбежно искажает направление поляризации случайным образом. Это явление основано на фундаментальном принципе Гейзенберга, утверждающем, что невозможно определить состояние квантового объекта, не исказив это состояние. Следовательно, после перехвата группы битов некоторые их них непременно окажутся искаженными. Периодически рассчитывая контрольные суммы или хеши для группы битов (см. раздел «Управление целостностью данных») и обмениваясь ими по открытому каналу, Передатчик и Приемник всегда могут определить, сохранена ли конфиденциальность передаваемой информации.

Поскольку алгоритм определения верных битов сложен и скорость передачи данных по каналам «квантовой» связи невелика, предполагается, что таким образом будут передаваться только шифровальные ключи, а сами данные – по открытому каналу в зашифрованном виде.

На момент написания этих строк «квантовая криптография» пока не получила широкого распространения.

6.4. Стандартные стеки протоколов

В то время, как протоколы «физического» и «канального» уровней считаются универсальными и не ориентированными на конкретный тип

адресации узлов, способ выбора маршрутов, методы аутентификации и прочее, протоколы более высоких уровней обычно объединены в большие группы, реализующие ту или иную технологию передачи данных. Среди них можно выделить:

- стек TCP/IP – доминирующая в настоящий момент технология организации локальных и глобальных сетей;
- стек NETBIOS – технология построения небольших, структурно простых, но очень быстрых локальных сетей;
- стек IPX – технология, нередко используемая для построения корпоративных Интранет–сетей среднего размера.

6.4.1. Стек TCP/IP

Технология TCP/IP в настоящее время является базовой для организации самых разных сетей. Многие альтернативные технологии (например, NetBIOS) в настоящее время лишь часть сетевых задач решают самостоятельно, а для остальных пользуются услугами TCP/IP.

1. *Сетевой уровень стека TCP/IP*. В настоящее время доминирующей в локальных и глобальных сетях является система адресации, поддерживаемая протоколом IP (от англ. Intertnet Protocol – Межсетевой протокол).

Протокол IP версии 4 (сокращенно «IPv4») предполагает, что каждый узел сети имеет 32-битовый адрес. Распространена форма записи IP-адреса в виде четырех десятичных октетов, например: 1234567890_{10}= $499602D2_{16}$= $49_{16}.96_{16}.02_{16}.D2_{16}$= 73.150.02.210. Некоторые IP-адреса имеют специальное назначение:

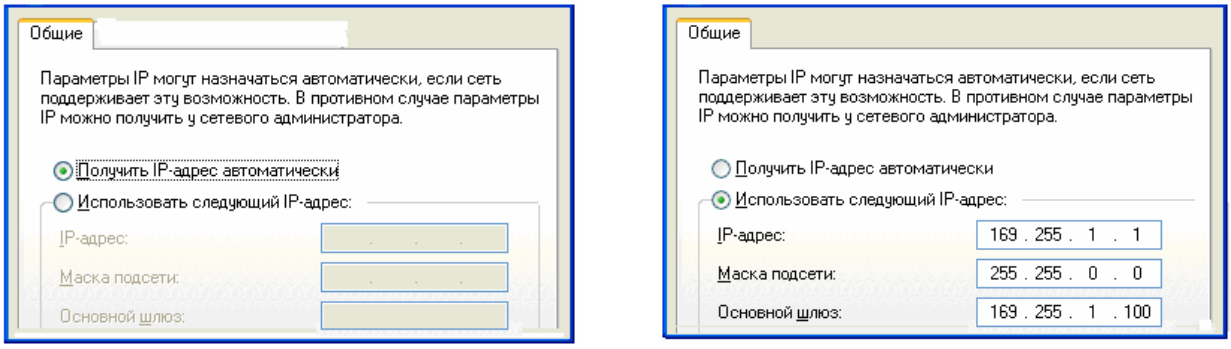
- 0.0.0.* – адрес машрутизатора «по умолчанию»;
- 10.0.0.* – используются в частных локальных сетях для «выделенных» узлов (серверов, маршрутизаторов, принтеров и т.п.);
- 100.64.*.* – используются в локальных сетях Интетрнет-провайдеров для «выделенных» узлов (серверов, маршрутизаторов, принтеров и т.п.);
- *.*.*.255 – широковещательный адрес;
- 127.0.0.* – собственный идентификатор любого сетевого узла;
- 169.254.*.*, 192.168.*.* и 172.16.*.* – рекомендуются для локальных сетей;
- 192.*.2.*, 198.51.100.*, 203.*.113.* – рекомендуются для использования в документации и примерах;
- 198.18.*.* – используются для тестирования производительности.

В настоящий момент общепризнано, что пространство 32-битовых IP-адресов, образующих Интернет, исчерпано[1]. Однако в рамках локальной сети, не имеющей выхода в Интернет, узлы сети могут иметь произвольные IP-адреса.

[1] До 25-30% 32-битовых IP-адресов Интернета хотя и имеют «хозяина», но реально не используются.

Более современный протокол IP версии 6 (сокращенно «IPv6») предполагает, что каждому узлу сети назначен 128-битовый адрес. Распространена форма записи в виде восьми 16-ричных числовых групп, например 1111:2222:3333:0abc:0edf:1234:4321:10ab. Для совместимости с «IPv4» предусмотрен диапазон адресов, в котором старшие группы равны 0, затем следует группа «ffff», а младшие 32 бита соответствуют адресу «IPv4», например, «::ffff:111.222.121.212».

Переход с «IPv4» на «IPv6» продолжается, но пока далек от завершения.



а) Динамическое назначение б) Статическое назначение

Рис. 6.13. Назначение IP-адресов

Динамический IP-адрес – назначается узлу автоматически в момент подключения его к локальной сети (например, в процессе установления коммутируемого доступа). Для этого существуют специальные протоколы, например, DHCP (см. рис. 6.13,а).

Статический IP-адрес – назначается узлу администратором локальной сети или провайдером Интернет-услуг (см. рис. 6.13, б).

Нередко используются жаргонные термины.

«*Черный адрес*» – IP-адрес локальной сети, который может иметь произвольное значение, поскольку «не виден» в Интернете.

«*Белый адрес*» – видимый в Интернете, официально зарегистрированный и привязанный к определенному узлу глобальной сети IP-адрес.

«*Серый адрес*» – видимый в Интернете, официально зарегистрированный, но не привязанный к определенному узлу глобальной сети IP-адрес. Как правило, привязка адреса к конкретным узлам выполнятся на временной основе путем применения технологии NAT.

NAT (англ. Network Address Translation) – технология, в соответствии с которой маршрутизатор не просто передает дейтаграммы из одной сети в другую, но и, пользуясь внутренней таблицей соответствия, подменяет реальный IP-адрес на какой-то другой. Это позволяет Интернет-провайдеру назначать абонентам, имеющим «черные» адреса, временные «серые» адреса.

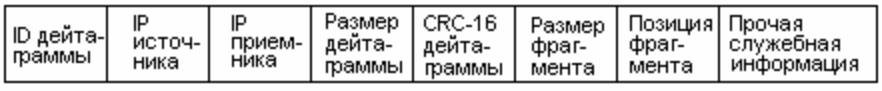


Рис. 6.14. Упрощенный формат заголовка IP-дейтаграммы

Единицей информации на сетевом уровне, где работает протокол IP, является «*дейтаграмма*», которая, как и пакеты других уровней, содержит служебный заголовок (см. рис. 6.14) и «блоб» (то есть, данные). Основными задачами протоколов семейства IP являются:

- прием дейтаграмм и передача в направлении указанного IP-адреса;
- разбиение дейтаграмм на фрагменты (во время передачи) и сборка их из отдельных фрагментов (во время приема).

В выборе направления передачи активно участвуют «таблицы маршрутизации», которые заполняются системным администратором (это характерно для узлов-маршрутизаторов) или составляются автоматически (это характерно для хостов, см. рис. 6.17). Протокол IP проверяет целостность пришедшей дейтаграммы при помощи CRC-16. Но, отсылая дейтаграмму, протокол не контролирует успешность доставки. При возникновении сбоев (например, если нарушена целостность пришедшей дейтаграммы) протокол возвращает узлу-отправителю сообщение об ошибке, используя протокол ICMP.

Вспомогательные протоколы сетевого уровня.

Протокол ARP – составляет таблицы «коммутации», то есть таблицы соответствия логических IP-адресов физическим ARP-адресам (см. рис. 6.15).

Протокол ICMP – поддерживает передачу диагностических сообщений (см. рис. 6.16).

Протокол RIP – строит таблицу маршрутизации, используя в качестве критерия оптимальности минимальное число «хопов» (то есть, «прыжков» от узла к узлу) – см. рис. 6.17.

```
C:\>arp -a
Интерфейс: 169.254.54.237 --- 0x10005
  Адрес IP             Физический адрес   Тип
  169.254.76.244       00-11-5b-c5-42-c4  динамический
  169.254.187.112      00-03-ff-91-e4-19  динамический
```

Рис. 6.15. Утилита ARP показывает таблицу коммутации

```
C:\>ping 169.254.187.112

Обмен пакетами с 169.254.187.112 по 32 байт:

Ответ от 169.254.187.112: число байт=32 время<1мс TTL=128
Ответ от 169.254.187.112: число байт=32 время<1мс TTL=128
Ответ от 169.254.187.112: число байт=32 время<1мс TTL=128
Ответ от 169.254.187.112: число байт=32 время<1мс TTL=128
Статистика Ping для 169.254.187.112:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0 (0% потерь),
Приблизительное время приема-передачи в мс:
    Минимальное = 0мсек, Максимальное = 0 мсек, Среднее = 0 мсек
```

Рис. 6.16. Утилита PING использует протокол ICMP
для проверки работоспособности узлов

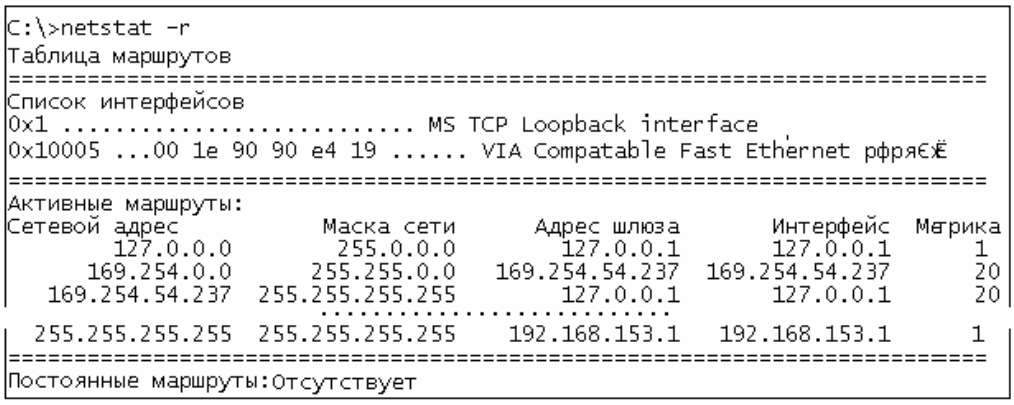


Рис. 6.17. Утилита NETSTAT показывает таблицу маршрутов, созданную при помощи протокола RIP

Недостатки IP: специальным образом сконфигурированный узел может: 1) перехватывать дейтаграммы, получая несанкционированный доступ к данным; 2) вмешиваться в работу вспомогательных протоколов с целью, например, перенаправления маршрутов на ложные узлы. Программы, предназначенные для перехвата трафика и его анализа, называются «*снифферами*». Примеры: WireShark (для Windows), TCPDump (для UNIX).

2. *Протокол IPSec* (англ. IP Secured – Защищенный IP). Этот протокол является развитием протокола IP, избавляющим его от основных недостатков. По сравнению с IP, он способен:

- выполнять взаимную аутентификацию узлов сети;
- шифровать и расшифровывать дейтаграммы.

Протокол IPSec не является жестко фиксированным, возможны различные варианты его конфигурирования и настройки. Он способен пользоваться возможностями трех внутренних протоколов, функции которых могут частично пересекаться:

- AH (англ. Authentification Header – Заголовок Аутентификации) – обеспечивает контроль целостности передаваемых данных на основе хеширования их с ключом (см. раздел «Криптографические хеш-функции»);
- ESP (англ. Encapsulating Security Protocol – Инкапсулирующий протокол безопасности) – обеспечивает аутентификацию узлов и контроль целостности передаваемых данных путем шифрования данных внутри дейтаграммы (транспортный режим) или всей дейтаграммы полностью (туннельный режим);
- IKE (англ. Internet Key Exchange – Обмен Интернет-ключами) – протокол, обеспечивающий обмен шифровальными ключами с использованием алгоритма Диффи–Хеллмана.

В рамках «IPv4» протокол IPSec опционален, для «IPv6» он обязателен.

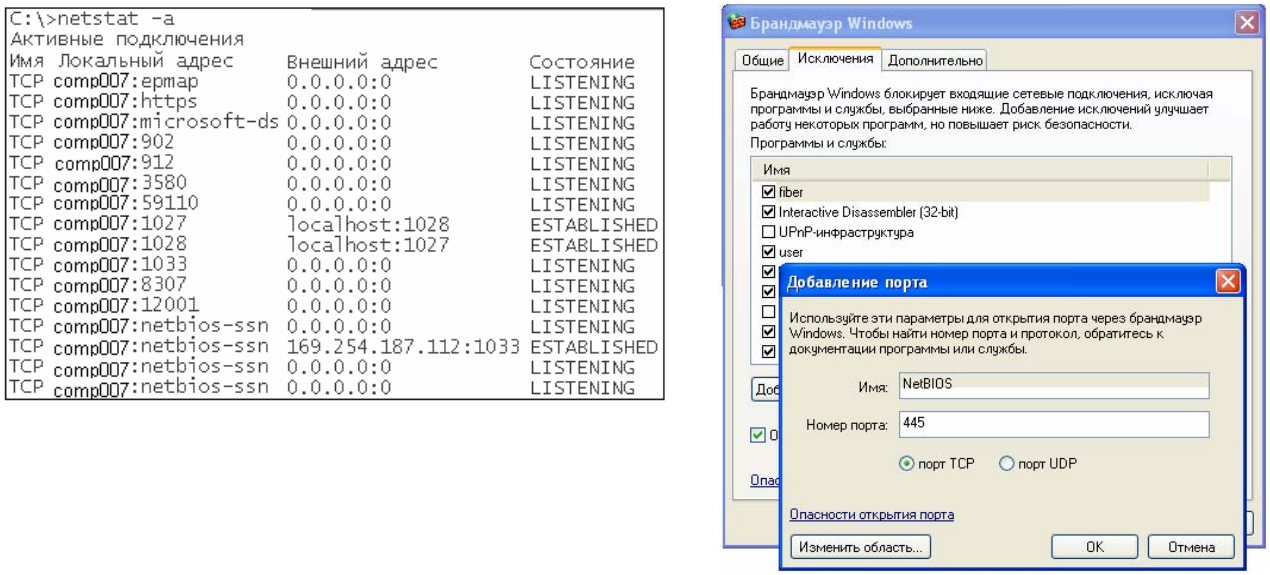
3. *Транспортный уровень стека TCP/IP*. Задача протоколов этого уровня: обеспечение обмена дейтаграммами между узлами, расположенными в разных сетях (см. рис. 6.4,б). При этом доставка дейтаграмм должна быть обеспечена не только с узла на узел с конкретным IP-адресом, но и в конкретный «порт».

Сетевой порт – целое число, идентифицирующее некоторую сетевую службу. Фактически, каждый порт соответствует некоторой задаче (процессу или потоку) на уровне приложений OSI/ISO (см. рис. 6.6). Например, порт 110 соответствует службе передачи e-mail по протоколу POP3 на почтовый сервер. Стандартные номера портов могут быть найдены в файлах:

- C:\windows\system32\drivers\etc\services – для Windows;
- /etc/services – для Unix.

Брандмауэр (он же *файрволл*, он же *межсетевой экран*) – программный или программно-аппаратный фильтр, блокирующий прохождение дейтаграмм на указанные при настройке сетевые порты (см. рис. 6.18,б). Брандмауэры существенно расширяют и дополняют возможности протоколов транспортного уровня по блокированию нежелательного трафика.

Сетевой сокет – абстрактный объект транспортного уровня, выполняющий передачу данных другим таким же сокетам. Каждый сокет однозначно характеризуется парой «IP-адрес:порт», например «169.254.187.112:110». При этом не конкретизируются формат передаваемых данных, способ их доставки и т.п. Программная реализация сокетов обеспечивает для уровня приложений OSI/ISO интерфейс к низкоуровневым возможностям транспортного уровня.



а) Список «открытых» портов б) Настройка брандмауэра

Рис. 6.18. Работа с сетевыми портами

Для решения задач транспортного уровня могут быть использованы два альтернативных протокола.

Протокол UDP. Этот простой протокол ничего не добавляет к транспортным возможностям протокола IP. Он просто обеспечивает передачу дейтаграмм в требуемый порт сетевого узла. Обычно протокол UDP используется для скоростного обмена короткими порциями данных. Например, при помощи UDP происходит взаимодействие HTTP-клиентов с DNS-серверами.

Протокол TCP. Этот сложный протокол обеспечивает надежную передачу данных между «удаленными» узлами (см. рис. 6.4,б). TCP способен разбивать дейтаграмму на отдельные «сегменты» и посылать их адресуемому узлу по разным маршрутам. При этом возможны дублирование «сегментов», передача их в неверном порядке и т.п. В задачи протокола также входит исправление ошибок и сборка на финишном узле из отдельных «сегментов» целостной дейтаграммы. Для этого протокол устанавливает соединение между узлами и поддерживает обмен не только данными, но и служебными сообщениями.

Недостатки протоколов сетевого уровня – специальным образом сконфигурированный узел, через который выполняется передача дейтаграмм и сегментов, может: 1) получать несанкционированный доступ к содержимому дейтаграмм; 2) вмешиваться в алгоритм работы протоколов TCP и UDP с целью, например, подмены легальных узлов промежуточной сети ложными. Это может привести к реализациям кибератак типа MITM (англ. Man In The Middle – Человек в середине).

4. *Верхние уровни стека TCP/IP*. В стеке TCP/IP отсутствует однозначное разделение функций между отдельными «уровнями», как этого требует модель ISO/OSI. Как правило, протоколы этой группы «универсальны» и решают задачи нескольких уровней.

6.4.2. Стек NetBIOS

NetBIOS (Базовая система ввода-вывода в сетях) – группа протоколов, предназначенная для быстрого обмена сообщениями в структурно простых локальных сетях. Кроме того, NetBIOS – это API (система интерфейсных библиотек), обеспечивающая для прикладных программ доступ к возможностям протокола. Обычно NetBIOS используется для организации доступа к «общим» каталогам (директориям, папкам), файлам и принтерам различных узлов сети.

1. *Сетевой уровень стека NetBIOS*. Он представлен собственно протоколом NetBIOS или его модификацией NetBEUI (NetBIOS Extended User Interface – Расширенный пользовательский интерфейс). Идентификатор сетевого узла состоит из текстовой строки длиной 15 символов[1] (например, «COMP123») и дополнительного байта с кодом типа ресурса (компьютер или принтер). Протокол не поддерживает маршрутизацию, передача пакетов выполняется в соответствии с принципом «пассивная звезда» (см. рис. 6.3,а) или «шина» (см. рис. 6.3,б). Возможна как неупорядоченная передача пакетов, так и передача, сопровождаемая служебными сигналами.

2. *Транспортный уровень стека NetBIOS*. Он базируется на протоколе SMB (англ. Server Message Block – Блок серверных сообщений), различные модификации которого так же известны под именами CIFS (Common Internet File System – Общая межсетевая файловая система) и Samba. Этот протокол,

[1] Если имя короче 15 символов, до остаток строки заполняется пробелами.

кроме транспортных, реализует также функции прикладного уровня – удаленный доступ к файлам, каталогам и принтерам.

3. *NetBIOS через TCP/IP*. В настоящее время технология, использующая оригинальные версии протоколов NetBIOS/NetBEUI и SMB, считается устаревшей. Взамен используются обобщенный протокол NBT (англ. NetBIOS over TCP/IP), предполагающий, что сетевые и транспортные возможности реализуются средствами стека TCP/IP, а NetBIOS/NetBEUI, работающий поверх, отвечает только за реализацию сетевого доступа к файловой системе и принтерам. Соответственно, в технологии «NetBIOS через TCP/IP» поддерживаются маршрутизация сетевых сообщений, аутентификация узлов сети и шифрование трафика.

При этом доступ к протоколам технологии осуществляется (в зависимости от конфигурации стека) либо через сетевой порт 445, либо через группу портов 137-139.

6.4.3. Стек IPX/SPX

Эта группа протоколов первоначально использовалась для сетей, работающих под управлением операционной системы Novell NetWare, а после ее ухода с рынка была реализована в MS-DOS, MS Windows и UNIX-подобных операционных системах. Обычно IPX/SPX используются для организации доступа к каталогам (директориям, папкам), файлам и принтерам «главных» узлов сети, так называемых «серверов Novell». В рамках этой технологии особенно удобно организовывать централизованные базы данных.

1. *Сетевой уровень IPX/SPX*. Этот уровень базируется на протоколе IPX (Internetwork Packet eXchange – Межсетевой обмен пакетами), который выполняет маршрутизируемую пересылку дейтаграмм с использованием алгоритма RIP, выбирающего маршрут в соответствии с критерием минимального числа «хопов» (не более 15). Адрес сетевого узла в технологии IPX представляет собой длинное число, составленное из:

- 4 байтов адреса сети (присваивается администратором);
- 6 байтов MAC-адреса (присваивается автоматически);
- 2 байтов с номером сокета (см. раздел «Стек TCP/IP»).

Протокол не предусматривает фрагментации дейтаграмм, но способен динамически изменять их размер в зависимости от пропускной способности сети. Заголовок дейтаграммы содержит поле для CRC-16, но в большинстве реализаций контроль целостности дейтаграмм не производится (значение всегда равно $FFFF_{16}$).

2. *Канальный уровень IPX/SPX*. Основан на протоколе SPX (англ. Sequenced Packet Exchange – Последовательный обмен пакетами). Он поддерживает не только функции канального, но и сеансового уровня. SPX осуществляет управление процессами установки логической связи, обмена и окончания связи между любыми двумя узлами сети, гарантирует очередность приема пакетов согласно очередности отправления. Протокол SPX может

работать поверх TCP/IP, самостоятельно расширяя и дополняя 32-битовые IP-адреса до своего формата.

3. *Протоколы верхних уровней IPX/SPX*. В сетях Nowell NetWare протоколом уровня приложений служит NCP (NetWare Core Protocol – Протокол ядра сети), который обеспечивает работу основных служб сетевой ОС и объединяет функции всех уровней от транспортного до прикладного модели OSI. С помощью этого протокола можно выполнить подключение к серверу, отображение каталогов сервера на виртуальные устройства рабочей станции (например, на диск «Z:\» в Windows или псевдофайл «/dev/sd*» в UNIX), работу с файлами (включая шифрование «на лету») и т.п. Еще один протокол SAP играет вспомогательную роль – его сообщения, регулярно рассылающиеся по узлам, содержат информацию о текущем состоянии активных ресурсов сети.

В современных реализациях NCP часто работает сразу поверх TCP/IP, не используя ни IPX, ни SPX.

6.5. Уровни организации сеансов и представления данных

На этих уровнях модели ISO/OSI сконцентрированы средства, непосредственно ориентированные на защиту передаваемой информации. Нередко эти средства представляют собой большие комплексы протоколов, соответствующих разным уровням ISO/OSI. Большинство этих протоколов основаны на идеях, рассмотренных ранее в разделе «Методы удаленной аутентификации».

6.5.1. Технология аутентификации Kerberos

Существует достаточно много алгоритмов аутентификации, которые используются в локальных сетях, построенных средствами различных операционных систем. Например, в сетях, содержащих узлы с разными версиями MS Windows, можно встретить применение алгоритмов LM, NTLM (он же NT-Hash) и Net-NTLM (см. раздел «Аутентификация в Windows»). В сетях на основе UNIX-подобных операционных систем средства аутентификации представляются клиент-серверными службами NIS (англ. Network Information Service – Служба сетевой информации). Также средства аутентификации, основанные на асимметричных шифрах (см. ранее раздел «Методы удаленной аутентификации»), поддерживаются протоколом SSH и основанными на нем службами операционных систем Windows и UNIX. Однако в современных условиях актуальны универсальные методы, основанные на межплатформенных стандартах.

Kerberos – это комплекс универсальных протоколов, работающих на разных уровнях ISO/OSI и обеспечивающих надежную аутентификацию узлов локальной сети. Наиболее широко распространены реализации Kerberos, работающие поверх TCP/IP.

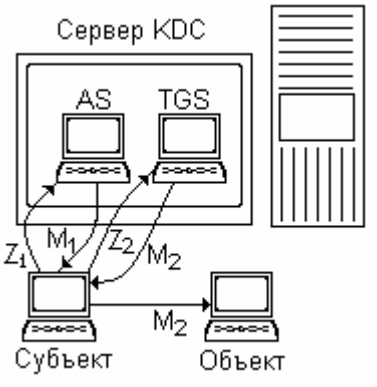


Рис. 6.19. Многошаговая аутентификация в KERBEROS

В технологии Kerberos предусмотрена аутентификация через «доверенного посредника» (см. рис. 6.19 и ср. с рис. 4.3 в разделе «Аутентификация с использованием асимметричного шифра»), однако при этом используются только симметричные шифры[1]. «Доверенный посредник» называется «центром распределения ключей» (англ. KDC – Key Distribution Center) и разделен на: 1) «сервер аутентификации» (англ. AS – Authentication Server); 2) «сервер выдачи разрешений» (англ. TGS – Ticket Granting Server). В процессе инсталляции Kerberos все хосты получают симметричный ключ K_{KDC} для доступа к KDC, а сам KDC симметричные ключи K_c и K_o для доступа к хостам (для примера – к «субъекту» и «объекту»). Кроме того, у KDC есть секретный ключ Ks. Очень упрощенно алгоритм аутентификации выглядит следующим образом.

Шаг 1. Узел-субъект посылает на AS запрос Z_1, содержащий свой идентификатор ID_c. Во избежание искажений запрос посылается в зашифрованном с ключом K_{KDC} виде. Поскольку запрос зашифрован «правильным» ключом, AS расшифровывает его и убеждается в подлинности узла-субъекта. В ответ он выдает узлу-субъекту персональный «мандат» («разрешение», «билет», «тикет») M_1, который дважды зашифрован: 1) сначала своим секретным ключом K_s (чтобы содержимое мандата никто не подменил); 2) потом ключом K_c, чтобы его мог расшифровать только узел-субъект. Шаг 1 выполняется однократно при подключении узла-субъекта к сети. Итогом этого шага является получение узлом-субъектом персонального разрешения на получение других мандатов.

Шаг 2. В целом, он соответствует схеме Шага 1, только на этот раз запрос Z_2, зашифрованный ключом K_{KDC}, посылается в направлении TGS, внутри Z_2 присутствуют собственный идентификатор ID_c, идентификатор узла-объекта ID_o и мандат M_1. Мандат является свидетельством ранее доказанной подлинности узла-субъекта. В ответ TGS возвращает дважды зашифрованный (ключами K_o и K_c) мандат M_2, содержащий ID_c и ID_o.

Шаг 3. Узел-субъект однократно расшифровывает мандат ключом K_c и результат (который не может расшифровать) пересылает узлу-объекту.

Шаг 4. Узел-объект, владея ключом K_o, расшифровывает M_2 и обнаруживает в нем запрос на обмен информацией со стороны узла-субъекта. Запрос удовлетворяется.

Шаги 2, 3 и 4 могут выполняться многократно при каждой попытке узла-субъекта обратиться к узлу-субъекту. Однако в Kerberos предусмотрено ограничение времени действия разрешений и мандатов, и по истечении срока (он вместе с идентификаторами включается в мандат) нужно повторно

[1] Ранние версии протокола поддерживали DES и RC4, современные AES с ключом 128 или 256 бит.

обращаться за получением. Также в запросы и мандаты добавляются и пересылаются вместе с ними случайные «оказии» и метки текущего времени, анализ которых позволяет отследить дубликаты сообщений аутентификации.

Kerberos повсеместно применяется в современных версиях MS Windows для аутентификации в локальных сетях с доменами Active Directory; в кроссплатформенных локальных сетях.

6.5.2. Технология аутентификации и шифрования SSL/TLS

SSL/TLS – технология, первоначально разработанная только для защиты Интернет-трафика, порождаемого протоколом HTTP, но в настоящих условиях защищает передачу файлов (протокол FTP), электронную почту (протоколы POP3, SMTP) и пр.

Задачи, решаемые технологией SSL/TLS:

- проверка подлинности (аутентификация) серверов;
- шифрование информации, передаваемой между клиентом и сервером;
- проверка целостности передаваемой информации.

SSL/TLS представляет собой семейство родственных протоколов, каждая новая версия которых расширяет возможности предыдущих:

- SSL (Secure Sockets Layer — Слой защищенных сокетов) версий 1.0 (экспериментальная) и 2.0 (потеряла актуальность);
- TLS (Transport Level Security – Защита транспортного уровня) версий 1.0 (она же SSL 3.0, содержит уязвимости), 1.1, 1.2 (самая распространенная), 1.3 (находится на стадии внедрения).

Протоколы работают поверх TCP, но существует вариант DTLS для работы поверх UDP. Работа SSL/TLS состоит из двух этапов.

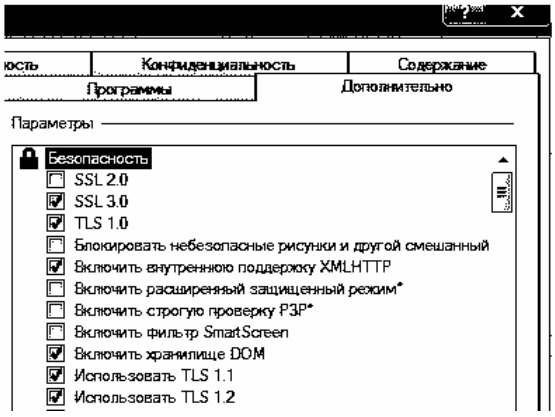


Рис. 6.20. Версии протокола, доступные в браузере

На первом этапе, который инициируется со стороны клиента (например, Интернет-браузера), устанавливается TCP-соединение, выполняются проверка подлинности сервера и согласование криптографических параметров. При этом согласовывается версия протокола, выбирается симметричный шифр из списка (null, RC2, RC4, DES, 3DES, IDEA, CHACHA20, AES 128 и 256 битов), хеш из списка (MD5, SHA-1, SHA-256) и метод обмена ключом симметричного шифрования (либо RSA с открытым ключом, либо алгоритм Диффи-Хеллмана), также выполняется обмен цифровыми сертификатами, организованными в формате X.509. Квалифицированный цифровой сертификат сервера (полученный от удостоверяющего центра) служит для доказательства его (сервера) подлинности, так же он (вместе с «самодельным» сертификатом клиента) служит для взаимного обмена сеансовым (одноразовым) ключом,

который в дальнейшем будет использоваться для шифрования трафика. Первый этап может завершиться неудачей, например, в том случае, если в списках алгоритмов, предлагаемых друг другу клиентом и сервером, нет пересечений – это возможно, если используется слишком «старый» клиент или слишком «новый» сервер, либо если в «клиенте» включены не все доступные версии протокола со своими алгоритмами (см. рис. 6.20). Другой причиной неудачи могут служить проблемы аутентификации, например, если сертификат сервера некорректен или просрочен.

На втором этапе работы протокола на узле-источнике выполняются разбиение потока данных на фрагменты, сжатие фрагментов[1], вычисление контрольных сумм и хешей, симметричное шифрование при помощи выбранного одноразового ключа и передача зашифрованных данных на транспортный уровень, а на узле-приемнике, соответственно, обратные операции.

Существуют библиотеки программных компонентов (например, OpenSSL для C или JSSE для Java), которые содержат реализации алгоритмов SSL/TLS.

Основная проблема SSL/TLS – использование большого количества различных Интертнет-браузеров (например, Internet Explorer, Mozilla Firefox, Google Chrome, Opera и т.п.) и WEB-серверов (MS IIS, Apache, Nginx, Tornado и т.п.), в том числе устаревших версий, что вынуждает всю систему в целом работать на уровне безопасности, соответствующем этим самым устаревшим версиям. Например, еще очень велика доля TLS-соединений, защищенных шифром RC4, и использующих контроль целостности на основе хеша MD5.

6.5.3. Технологии аутентификации и шифрования в мобильных сетях

В мобильных сетях, построенных по технологии Wi-Fi, используются специальные протоколы для защиты трафика и аутентификации узлов, временно подключаемых к сети (см. рис. 6.5, а) через «точку доступа». Основными особенностями схемы подключения являются:

- относительно невысокие скорости передачи данных;
- относительно невысокие вычислительные возможности «точки доступа»;
- наличие большого количества ошибок при передаче данных;
- легкость перехвата и анализа как передаваемых данных, так и служебных сообщений;
- отсутствие специального сервера-«посредника», обслуживающего процедуру аутентификации.

В защите нуждается, в первую очередь, «служебный» трафик, передаваемый в процессе инициализации подключения, в дальнейшем же защита «основного» трафика может быть выполнена средствами SSL/TLS. Однако «длинные» и «ресурсоемкие» протоколы аутентификации, используемые в проводных сетях (например, Kerberos или NTLM), в мобильных сетях неприемлемы.

[1] Используется редко.

Все технологии защиты Wi-Fi-трафика предусматривают шифрование с использованием сеансовых (одноразовых) ключей, но на начальном этапе инициализации соединения используются постоянные или закономерно видоизменяемые мастер-ключи, которые генерируются на основе строковых паролей.

1. *Технология Wep* (англ. Wired Equivalent Privacy – Приватность, эквивалентная проводной) использует шифрование трафика методом RC4 с ключами длиной 40 или 104 бита. Мастер-ключи либо назначаются участникам однократно (в этом случае они могут быть похищены), либо симметрично генерируются в каждом сеансе из заранее предопределенного, относительно небольшого набора «затравок» по довольно предсказуемому алгоритму (в этом случае они могут быть определены подбором). В настоящее время технология Wep считается уязвимой и устаревшей.

2. Технология WPA (англ. Wi-Fi protected access – Защищенный доступ к Wi-Fi) также использует шифрование трафика методом RC4 с ключом длиной 128 бит, в генерации которого участвует специальный алгоритм TKIP (англ. Temporary Key Integrity Protocol – Протокол целостности одноразовых ключей), благодаря чему подбор ключей существенно затруднен. Для аутентификации могут использоваться различные алгоритмы, например EAP-MD5 (на основе хеширования с ключом), EAP-TLS (на основе проверки ЭЦП), EAP-PSK (с предопределенным мастер-ключом), EAP-SIM (с использованием SIM-карт) и прочие, предусмотренные в спецификации EAP (англ. Extensible Autentification Protocol – Расширяемый протокол аутентификации). Технология WPA обеспечивает приемлемую стойкость ко «взлому», но самые современные устройства и операционные системы ее уже не поддерживают.

3. Технология WPA2 является расширением WPA, в основном, за счет шифрования трафика методом AES с ключом длиной до 256 битов. Технология WPA2 сейчас считается основной для использования в Wi-Fi.

На момент написания этих строк существует перспективная спецификация WPA3, но массовое внедрение ее еще не началось.

6.6. Защита информации на уровне приложений

Протоколы прикладного уровня работают по *технологии клиент-сервер*, а именно: клиент посылает запросы к серверу, сервер их выполняет (обычно, обрабатывая какую-то информацию) и возвращает результаты работы клиенту.

Табл. 5.3. Порты наиболее распространенных сетевых служб

Порт	**Служба**	**Порт**	**Служба**
20, 21	FTP – передача файлов	**119**	NNTP – группы новостей
990, 989	FTPS – передача файлов	**123**	NTP – служба сетевого времени
22	SFTP – передача файлов	**135**	RPC – службы Windows
69	TFTP – передача файлов	**137-139**	NetBIOS/NBT через TCP/IP
23	Telnet – виртуальный терминал	**445**	NetBIOS/NBT через TCP/IP
25	SMTP – передача E-Mail	**161**	SNMP – служебные запросы
110, 995	POP3 – прием E-Mail	**80, 443**	HTTP – доступ к WWW-страницам
143, 220	IMAP – прием/передача E-Mail	**53**	DNS – служба доменных имен

Большинство служб уровня приложений работают поверх TCP/IP, соответственно они используют TCP-соединение через определенный *сетевой порт*. На уровне приложений работу с сетевым портом обслуживает отдельная «служба», то есть системный процесс или поток, откликающийся на некоторое идентифицирующее число (*номер порта*). Номера сетевых портов для распространенных служб стандартизованы (см. табл.5.3).

Последовательный перебор различных номеров портов сетевого узла, выполняемый с целью определения активных служб, называется *сканированием портов*, это действие обычно предшествует попытке атаки на сервер.

6.6.1. Разделяемый доступ к сетевым ресурсам

Средства разделяемого удаленного доступа к сетевым ресурсам можно условно разделить на две группы:

- средства доступа к файловой системе;
- средства доступа к пользовательскому интерфейсу.

Доступ к серверу разделяемых удаленных ресурсов требует обязательной аутентификации со стороны удаленных клиентов. Аутентификация может основываться на разных протоколах: LM, NTLM, Kerberos и прочих.

1. *Удаленный доступ к файловым системам*. Для доступа к «сетевым дискам», «сетевым принтерам» и прочим удаленным ресурсам можно использовать отдельно инсталлируемые службы (например, Lantastic или Samba), а можно пользоваться встроенными средствами операционных систем (например, «Службой доступа к файлам и принтерам сетей Microsoft»). Эти службы и средства опираются на различные протоколы, например на CIFS/Samba, NFS (англ. Network File System – Сетевая файловая система), AFS (Andrew File System) и другие. Обычно они обеспечивают удаленный доступ к структуре каталогов и файлов, позволяют перемещаться по ней, просматривать, записывать, копировать, удалять файлы; отправлять данные на печать и прочее. При этом учитываются правила систем разграничения доступа, характерные для конкретных операционных систем (списки контроля доступа в Windows и биты доступа в UNIX). Таким образом, обращение возможно только к тем ресурсам, доступ к которым явно разрешен администратором сервера.

К сожалению, конкретные реализации средств удаленного доступа к файловым системам на протяжении многих десятилетий отличаются наличием большого количества ошибок и уязвимостей. Среди наиболее одиозных:

- CVE-1999-0166 (игнорирование ограничений доступа к удаленным каталогам, 1990-е годы);
- CVE-2000-0979 2001-0154 (ошибка в алгоритме парольной аутентификации – см. пункт «Алгоритмические ошибки» в разделе «Уязвимости программного обеспечения»);
- CVE-2017-0144 (возможность переполнения внутреннего буфера, что привело к распространению сетевого червя WannaCry, 2017 г.).

2. *Удаленные пользовательские интерфейсы*. Для доступа к «удаленным терминалам» и «удаленным рабочим столам» необходимо использовать отдельно инсталлируемое клиент-серверное программное обеспечение (например, MS RDS – Remote Desktop Service, VNC – Virtual Network Computing, RADMIN и т.п.). Они опираются на различные протоколы, например на RDP, Telnet, Rlogin, и обеспечивают доступ удаленного клиента к пользовательскому интерфейсу компьютера, позволяя ему (клиенту) выполнять файловые операции, запуск и завершение программ, манипуляции со внешними устройствами сервера и т.п., – с учетом действующих настроек разграничения доступа.

Рис. 6.21 – Блокирование антивирусом программы RADMIN

Обычно подобные программы позиционируются как «средства удаленного администрирования» и широко используются администраторами территориально распределенных сетей. Основная опасность заключается в том, что серверная программа может быть установлена на компьютере скрытно (например, троянской программой-дроппером или злоумышленником-инсайдером) и служить программной закладкой типа «программный люк» (backdoor). Вот почему компактные программы удаленного администрирования, будучи абсолютно безопасными и безусловно полезными, нередко распознаются антивирусами как «потенциально вредоносные» (riskware) и подвергаются блокированию и удалению.

3. *Telnet* – протокол поддержки информационного обмена с удаленным «терминалом», под которым традиционно понимается оконечное устройство ввода-вывода. Серверы Telnet встроены почти во все операционные системы, а клиентские утилиты играют роль «универсальных клиентов», поскольку воспринимает команды для удаленного сервера с клавиатуры и выводит ответы от удаленного сервера на экран. При помощи клиентской утилиты [1]можно, например, подключиться к удаленному серверу (например, к WEB-серверу через порт 80 или серверу E-Mail через порт 110) и обмениваться с ними текстовыми сообщениями. Подключение по Telnet не требует дополнительной аутентификации сверх той, которая штатно предусмотрена сервером для подключения любых иных клиентов.

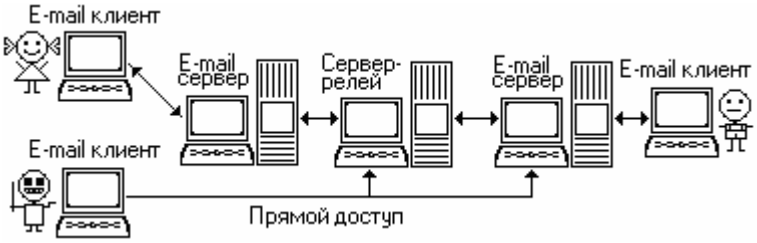
6.6.2. Электронная почта

Электронная почта (англ. E-Mail) – технология передачи по сети текстовых сообщений. При необходимости передачи вместе с текстовым сообщением двоичных данных (документов, изображений и т.п.) они

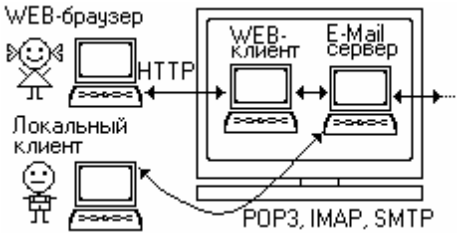
[1] В современных версиях Windows Telnet-клиент по умолчанию деактивирован.

подвергаются дополнительному кодированию и оформляются в виде «вложений» (attach). Виды кодирования:

- по основанию 64 (Base-64) с алфавитом «ABCDEFGHIJKLMNOPQRS…»;
- quoted printable с алфавитом «=00=01=02=03=04=05=06=07=08=09=10…»;
- Unix-to-Unix Encoding (UUE) с алфавитом «`!"#$%&'()*+,./01234567…».



а) Общая схема взаимодействия клиентов и серверов



б) Виды E-Mail клиентов

Рис. 6.22. Технология E-mail

Электронная почта (см. рис. 6.22) базируется на работе E-Mail серверов, которые обеспечивают:

- прием сообщений от клиентов или других серверов при помощи протокола SMTP;
- пересылку сообщений клиентам протоколов POP3 или IMAP;
- временное хранение сообщений.

Промежуточные E-Mail сервера называются «*релеями*» (англ. relay) – см. рис. 6.22,а.

Примеры почтовых серверов: SendMail (для UNIX и Windows), CMS – Courier Mail Server (для Windows).

Работа с сообщениями со стороны пользователей осуществляется при помощи E-Mail клиентов, которые обеспечивают:

- создание и редактирование сообщений;
- отображение содержимого сообщений и их вложений;
- постоянное хранение сообщений;
- (при необходимости) шифрование и расшифрование сообщений;
- (при необходимости) подписание сообщений при помощи ЭЦП и проверку их целостности и подлинности.

Возможны два типа E-Mail клиентов (см. рис. 6.22,б):

1) «локальные», которые расположены на компьютере пользователя и самостоятельно взаимодействуют с сервером при помощи протоколов POP3, IMAP и SMTP;

2) «удаленные», которые совмещены с E-Mail сервером, а пользовательский доступ к ним осуществляется как к WEB-странице по протоколу HTTP.

Примеры «локальных» клиентов: MS Outlook Express, TheBat (для Windows); mail, elm (для UNIX). Их возможности широки, они также позволяют шифровать сообщения, выполнять генерацию и проверку ЭЦП, осуществлять «прямой доступ» по сетевым протоколам непосредственно к «релею» или «финишному» E-mail серверу (см. рис. 6.22,а).

«Удаленные» клиенты, как правило, обладают минимальными возможностями по приему и передаче почтовых сообщений. Они характерны для ресурсов «бесплатной» электронной почты, таких как: mail.com, hotmal.com, gmail.com, mail.ru, rambler.ru, yandex.ru и др.

Протокол POP3 (доступ через сетевой порт 110) требует обязательной аутентификации клиента, принимающего почтовое сообщение от сервера, при помощи логина и пароля. По умолчанию POP3 передает логин (командой USER) и пароль (командой PASS) в открытом виде, что вполне допустимо в случае взаимодействия с E-Mail сервером через WEB-клиент, но неприемлемо в случае «локального» клиента (см. рис. 6.22,б). Способы решения проблемы основаны на использовании защищенной версии POP3S (доступ через сетевой порт 995), которая может поддерживать:

- передачу пароля (командой APOP) в виде хеша MD5 от комбинации пароля и временн*о*го штампа, присланного сервером;
- использование длинного алгоритма аутентификации типа «запрос-ответ», построенного в соответствии со спецификацией SASL (англ. Simple Autentification and Security Layer – Слой простой аутентификации и защиты);
- шифрование всего POP3-трафика средствами SSL/TLS.

Альтернативой POP3 является протокол IMAP, который в некоторых смыслах более удобен, но распространен меньше.

Протокол SMTP (доступ через сетевой порт 25 или 2525) по умолчанию не требует аутентификации, что может приводить к массовой рассылке нежелательных почтовых сообщений со стороны анонимных источников. В современных условиях для доступа E-Mail клиента к серверу используется защищенная версия протокола SMTPS (доступ через порт 465), в которой проверка логина и пароля обязательны. Однако взаимная аутентификация E-Mail серверов и «релеев» невозможна, поэтому ограничение «прямого доступа» к серверам (см. рис. 6.22,а) обеспечивается следующими методами:

- допускается доступ только со стороны IP-адресов, для которых в записях DNS установлен признак «MX» (англ. Message eXchanger – Ретранслятор сообщений), – то есть со стороны официально зарегистрированных E-Mail серверов и «релеев»;
- допускается доступ только со стороны E-Mail серверов и «релеев», отсутствующих в «черном списке» DNSBL (англ. DNS BlockList – Список блокировки DNS).

Часто обновляемые DNSBL содержат, среди прочего, списки «*открытых релеев*» (open relays), то есть E-Mail серверов, которые сами не пользуются информацией из DNSBL и, таким образом, не контролируют источники проходящих через них почтовых сообщений.

Почтовое сообщение имеет формат, изображенный на рис. 6.23. Кроме основной информации об источнике письма и его теме, заголовок письма содержит список «релеев» вместе с их IP-адресами.

```
Received: from [10.8.2.22] (HELO mx22.rambler.ru)
          by mail80.rambler.ru (CommuniGate Pro SMTP 4.2.10)
          with ESMTP id 23780972 for masha-vesnushkina@rambler.ru;
          Sat, 07 Feb 2013 17:16:04 +0300
Received: from n7.bullet.mail.ac4.yahoo.com
          (n7.bullet.mail.ac4.yahoo.com [76.13.13.235])
          by mx22.rambler.ru (Postfix) with SMTP id 5A6AA89A457
          for <masha-vesnushkina@rambler.ru>; Sat,  7 Feb 2013 17:16:04 +0300 (MSK)
Received: from [76.13.13.25]
          by n7.bullet.mail.ac4.yahoo.com with NNFMP; 07 Feb 2013 14:16:03 -0000
Received: from [76.13.10.177]
          by t4.bullet.mail.ac4.yahoo.com with NNFMP; 07 Feb 2013 14:16:03 -0000
Received: from [127.0.0.1]
          by omp118.mail.ac4.yahoo.com with NNFMP; 07 Feb 2013 14:16:03 -0000
Received: (qmail 15799 invoked by uid 60001); 7 Feb 2013 14:16:02 -0000
Received: from [85.113.33.18] by web111212.mail.gq1.yahoo.com via HTTP;
          Sat, 07 Feb 2013 06:16:02 PST
Date:     Sat, 7 Feb 2013 06:16:02 -0800 (PST)
From:     Vasya Pupkin <vasya-pupkin@yahoo.com>
Reply-To: vasya-pupkin@yahoo.com
Subject:  Привет!
To:       masha-vesnushkina@rambler.ru

Привет! Как ты живешь? Я живу хорошо! Пиши почаще! Чмоки-чмоки,
--
Вася
```

Рис. 6.23. Формат почтового сообщения

Основные проблемы, связанные с электронной почтой.

1. Спам, то есть массовые рассылки несанкционированной рекламы. Эта проблема актуальна на протяжении всей истории существования E-Mail. В разные моменты этой истории объем спама составлял до 90% от всего почтового трафика в Интернете. Источниками рассылки спама в настоящее время являются:

- хосты, которые принадлежат официально зарегистрированным организациям, занимающимся рекламой в Интернете;
- «взломанные» хосты и сервера;
- хосты, входящие в состав «ботнетов».

Основным техническим фактором, способствующим рассылке спама, является присутствие в Интернете «открытых релеев».

Методы борьбы со спамом:

- блокирование работы и ликвидация «открытых релеев»;
- фильтрация спама.

Фильтрация спама обычно выполняется на E-Mail серверах при помощи программ типа «Spamassassin». Критериями работы фильтров являются анализ контента (например, блокируются письма, содержащие ключевые слова «приз» и «наследство») и анализ сетевой статистики (например, блокируются массовые рассылки из одного источника).

2. Похищение (например, путем вредоносных программ) или подбор реквизитов аутентификации. Корректные методы генерации и хранения паролей, а также методы борьбы с вредоносными программами были подробно рассмотрены ранее в соответствующих разделах.

3. Перехват почтового трафика. Он может быть выполнен при помощи программ типа WireShark. Проблема может быть решена, например, использованием «S/Mime» – протокола шифрования электронной почты. Но он обеспечивает криптографическую защиту всего канала связи – от одного локального E-Mail клиента до другого, что несовместимо с технологией, использующей Web-клиенты. Взамен применяется работа почтовых протоколов

поверх SSL/TLS. Также возможно шифрование писем на «локальных» клиентах (см. рис. 3.24 в разделе «Принципы асимметричной криптографии»).

4. Уязвимости в серверном и клиентском программном обеспечении. Примером может служить ошибка CVE-2001-0145 программы MS Outlook Express, приводившая к автоматическому запуску вложений (например, программных файлов, переименованных злоумышленниками из «virus.exe» в «myfoto.jpg»).

5. Неоправданное доверие содержимому почтовых сообщений. Программные уязвимости рано или поздно исправляются, а методы «*социальной инженерии*» (то есть, методы обмана и мошенничества), сохраняют актуальность. В рассматриваемом контексте наиболее типичны почтовые сообщения, в которых получателя убеждают самостоятельно запустить присланную во вложении программу (см. рис. 6.24). В настоящее время почтовые вложения остаются основным источником вредоносных программ, используемых в кибератаках на рядовых и корпоративных пользователей.

6.6.3. Протоколы семейства FTP

FTP – протокол обмена между клиентом и сервером большими объемами информации (например, файлами). Как правило, полученные данные первоначально хранятся на носителе информации узла-источника и, после передачи по сети, предназначены для сохранения на аналогичном носителе узла-получателя. Обычно FTP используется для доступа к большим файловым архивам, расположенным на удаленных узлах сети. Также FTP – основной способ загрузки данных и программ на сервера во время их удаленного администрирования.

Протокол FTP работает через TCP и использует два соединения: 1) для обмена управляющей информацией (через сетевой порт 21h); 2) для обмена данными (на сервере – через сетевой порт 20h). На клиенте для обмена данными может быть использован любой порт. Существуют два режима работы клиента:

- активный – во время которого клиент инициирует взаимодействие с сервером, сообщает ему желаемый порт для обмена данными и приступает к обмену;
- пассивный – во время которого клиент инициирует взаимодействие с сервером, но затем переходит в режим пассивного ожидания и предоставляет серверу право самому выбрать порт для обмена данными.

Очевидно, FTP может вступить в конфликт с межсетевым экраном, использующим фиксированные списки «закрытых» портов.

FTP поддерживает аутентификацию клиентов посредством логина и пароля, которые передаются в открытом виде. FTP-сервера, на которых аутентификация отключена, называются «*анонимными*». Для решения проблемы перехвата атрибутов аутентификации используются версии SFTP (работает поверх SSH через порт 22) и FTPS (работает поверх SSL/TLS через

порты 990 и 989), что позволяет шифровать служебные сообщения и данные внутри FTP-трафика.

Примеры FTP-клиентов: утилита FTP.EXE, утилита CUTEFTP.EXE, файловые менеджеры FAR, Total Commander и прочие. Также FTP-клиенты встроены почти во все Интернет-браузеры[1].

Существует упрощенный протокол TFTP, работающий поверх UDP (доступ через сетевой порт 69), передающий данные блоками по 512 байтов. Аутентификация клиентов в этом протоколе не предусмотрена, он преимущественно используется для удаленного администрирования серверов внутри локальной сети. Пример клиента: утилита TFTP.EXE.

6.6.4. Протоколы HTTP и HTTPS

HTTP – простой и быстрый протокол обмена между клиентом и сервером сравнительно небольшими порциями данных (типичный размер – в пределах мегабайта), работающий в режиме «запрос-ответ» (через 80 порт TCP). Как правило, от клиента к серверу направляются запросы на получение данных, а в обратном направлении – сами данные (или сообщения о невозможности удовлетворения запроса). При этом данные могут представлять собой текстовые «страницы», изображения и т.п. Они предназначены, в первую очередь, для отображения браузером на экране компьютера.

Примеры HTTP-серверов: Microsoft IIS, Apache, Ningx и пр. Примеры HTTP-клиентов: Интернет-браузеры Microsoft Internet Explorer, Netscape Navigator, Opera, FireFox, Google Chrome и т.п. Возможны также клиенты, не встроенные в Интернет-браузеры, а представляющие собой «переходники», которые обеспечивают доступ к HHTP со стороны других протоколов.

HTTP не поддерживает ни методов аутентификации участников информационного обмена, ни защиты трафика. Поэтому основной проблемой, связанной с использованием протокола HTTP, является несанкционированный доступ к передаваемой информации. Для решения проблемы разработана и широко применяется HTTPS (доступ через порт 443) – защищенная версия HTTP, использующая для аутентификации серверов и шифрования трафика возможности технологии SSL/TLS. Фактически, HTTPS и SSL/TLS разрабатывались одновременно и совместно, как целостное средство защиты HTTP-трафика. Сервера, поддерживающие только HTTP, но не HTTPS, в настоящее время объявляются «не защищенными» и «небезопасными», проводится активная политика по их дискредитации и ограничения доступа к ним[2].

Простота HTTP/HTTPS обусловили легкость инкапсуляции в них любых других протоколов прикладного уровня. Например, в рамках такого подхода служебные сообщения, идентифицирующая информация и сообщения

---

[1] FTP-клиент принудительно исключен из некоторых современных браузеров как «редко используемый».

[2] Однако переключение сервера с HTTP на HTTPS сопряжено с существенными финансовыми затратами

электронной почты могут передаваться между почтовым клиентом и сервером в виде зашифрованных порций данных по протоколу HTTPS.

6.7. Межсетевые экраны

Брандмауэр (он же *файрволл*, он же *межсетевой экран*), упомянутый уже ранее в разделе «Стек TCP/IP», – это программный или программно-аппаратный фильтр, работающий на транспортном уровне модели ISO/OSI и блокирующий прохождение дейтаграмм и пакетов на указанные при настройке сетевые порты.

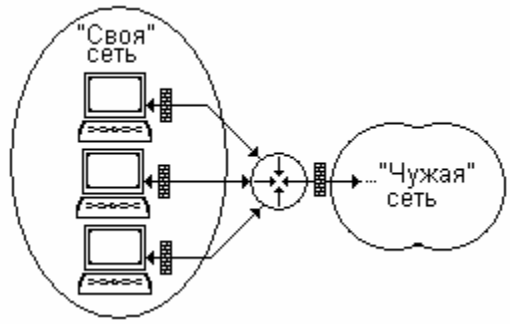


Рис. 6.24. Размещение брандмауэров в сети

Правила размещения брандмауэров в сети диктуются ее топологией и ролью в ней отдельных узлов. Безусловно, брандмауэрами должен огораживаться «*периметр*» сети, то есть подмножество узлов (включающее маршрутизатор), имеющих непосредственный доступ «наружу». В современных условиях использование «*бастионов*», то есть узлов «периметра», которые (якобы!) не подвержены «взлому» и по этой причине не нуждаются в огораживании брандмауэрами, недопустимо. Зато «*демилитаризованная зона*» (ДМЗ), то есть часть «периметра», которая предоставляет свои ресурсы и «своей» сети, и «чужой» (например, является сервером какой-либо сетевой службы вроде E-Mail), должна быть огорожена с обеих сторон. Также имеет смысл отгораживание брандмауэрами друг от друга внутренних узлов сети (см. рис. 6.24). Во-первых, это поможет заблокировать распространение с зараженного узла по внутренней сети вируса-червя. Во-вторых, это ограничит доступ к ресурсам сети (например, к базам данных) злоумышленника-инсайдера.

Фактически, настройки брандмауэра (см. рис. 6.18, б) реализуют систему правил некоторой политики разграничения доступа, субъектами которой являются внешние, а объектами – внутренние сетевые порты защищаемой сети (или отдельного узла). Наиболее проста и понятна настройка политик «дискреционного» (точнее, «ролевого») доступа к портам сети, которую первоначально можно представить в виде таблицы, и руководствуясь которой затем выполнять настройки.

Таблица 5.4. Пример разграничения доступа к портам

	169.254.*.* + 137,138,139,445	**192.169.*.* + 137,138,139,445**
169.254.*.* + 137,138,139,445	разрешить	запретить
192.169.*.* + 137,138,139,445	запретить	разрешить

Например, в таблице 5.4 описана политика разграничения доступа NetBIOS-трафика (через порты 137, 138, 139 и 445) для узлов двух сегментов локальной сети (первый с адресами 169.254.*.*, второй с адресами 192.169.*.*), так что разрешен доступ к «своему» сегменту и нет доступа к «чужому».

Целесообразно выделить два наиболее важных типа брандмауэров.

1. «Локальные брандмауэры» – устанавливаются на рабочих станциях с целью блокирования нежелательного трафика, поступающего извне или, наоборот, исходящего наружу. Например, простой брандмауэр встроен во все версии операционной системы Windows (см. рис. 6.18, б).

Существует также возможность агрегирования «локальных» брандмауэров со службами уровня приложений в единый программный комплекс, блокирующий нежелательный трафик на основании политик разграничения доступа более сложных, чем дискреционная. Подобный комплекс способен не только фильтровать пакеты и дейтаграммы, приходящие или посылаемые на определенные IP-адреса, но и, в соответствии с контекстом (какие протоколы и в каком порядке задействованы, каков результат их работы), анализировать внутреннее содержимое дейтаграмм (ведь на уровне приложений они присутствуют в расшифрованном виде!), блокируя опасные сетевые операции (например, сканирование портов), вредоносные программы, спам и иной нежелательный «контент». Многие современные антивирусные решения представляют собой именно такие программные комплексы.

2. «Пакетные фильтры» – работают на «пограничных» узлах сетей, таких, например, как «умные машрутизаторы». Поскольку они, в общем случае, не имеют доступа к зашифрованному содержимому пакетов и дейтаграмм, то блокирование нежелательного трафика выполняется на основании: IP-адреса источника и получателя; типа транспортного протокола; полей служебных заголовков протоколов сетевого и транспортного уровней; сетевых портов источника и получателя; статистики соединений со внешними IP-адресами.

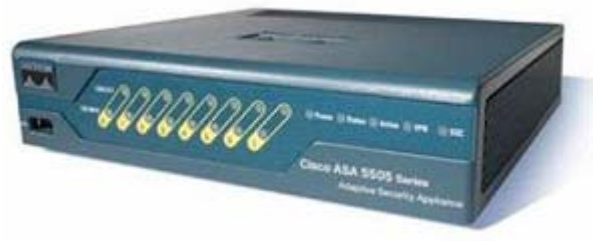


Рис. 6.25. Cisco ASA Firewall

Подобные брандмауэры способны, например, обнаруживать «аномалии» трафика, характерные для вредоносных воздействий, таких как DDOS-атаки, сканирование портов и т.п. Нередко часть функций «пакетных фильтров» выполняется аппаратно (см. рис. 6.25), что обеспечивает их высокую надежность, устойчивость ко «взлому», высокое быстродействие и большую пропускную способность.

6.8. Виртуальные частные сети (VPN)

Виртуальные частные сети (англ. VPN – Virtual Private Network) это способ организации одной сети внутри другой, причем вторая сеть не должна иметь доступа к потокам информации внутри первой.

Рис. 6.26. Принцип туннелирования

Обычно VPN используются для создания «туннелей», то есть защищенных каналов связи сквозь «чужую» сеть (см. рис. 6.26). Типичные применения VPN:

- защита удаленного доступа клиентов к сети Интернет-провайдера;
- защита трафика территориально распределенной организации при прохождении его сквозь Интернет.

Рис. 6.27. Пример стека протоколов

Обычно VPN-протоколы представляют собой протоколы канального уровня, которые инкапсулируют в себя весь трафик, организованный средствами протоколов более высокого уровня (см. рис. 6.27, также рис. 6.7). VPN-протоколы защищают инкапсулированный трафик путем аутентификации соединений и шифрования. Очевидно, VPN скрывает не только содержимое трафика (как SSL/TLS), но и адресную информацию (IP-адреса источника и приемника).

Примеры VPN-протоколов канального уровня: OpenVPN, PPPoE (он же PPP over Ethernet), Microsoft SSTP, L2TP+IPSec, PPTP и др.

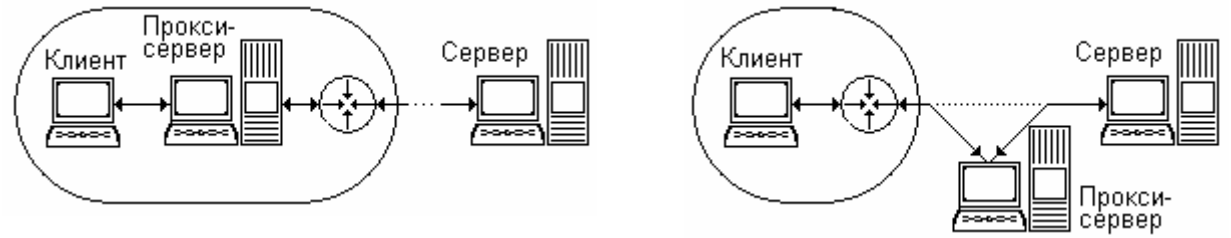
VPN может быть организована средствами MS Windows, в которую встроены VPN-клиент и VPN-сервер, поддерживающие несколько VPN-протоколов. Кроме того, существуют как проприетарные, так и свободно распространяемые решения от других производителей программного обеспечения для любых операционных систем.

Также в Интернете действует некоторое количество общедоступных (в том числе, условно-бесплатных) «международных» Интернет-провайдеров, удаленный доступ к которым защищен по технологии VPN. Для пользования их услугами необходимо применение специализированных VPN-клиентов (например, HotSpot Shield или Avira Phantom VPN). В РФ использование частными лицами подобных VPN-ресурсов ограничено на основании федерального закона №276-ФЗ от 29.07.1917.

6.9. Технологии PROXY

Прокси (от англ. proxy – доверенное лицо, полномочный представитель, посредник) – семейство технологий, расширяющих возможности протоколов уровня приложений, таких как HTTP, POP3/SMTP, FTP и др. Эти технологии предусматривают использование в канале связи между клиентом и сервером дополнительных серверов-посредников (см. рис. 6.29), решающих вспомогательные задачи, такие как:

- кэширование сообщений;
- учет и ограничение трафика;
- трансляция адресов.



а) Прокси-сервер в локальной сети б) Внешний прокси-сервер

Рис. 6.28. Использование прокси-сервера

Типичный Proxy-сервер поддерживает либо какой-то отдельный сетевой протокол уровня приложений (HTTP, FTP, POP3/SMTP и т.п.), либо сразу несколько, причем обрабатывает их независимо друг от друга. Существуют также специальные протоколы семейства SOCKS, которые обеспечивают инкапсуляцию различных протоколов в единый «пакет», обрабатываемый Proxy-сервером единообразно, вне зависимости от внутреннего содержимого.

1. Кэширование сообщений имеет смысл для HTTP-протокола. Промежуточный прокси-сервер, перенаправляя запросы от HHTP-клиента к удаленному HTTP-серверу, кэширует (накапливает) содержимое страниц, к которым выполнялся доступ. Если он обнаруживает повторное обращение к одной и той же странице со стороны клиента (или разных клиентов), то это обращение не выполняется, а клиенту выдается сохраненное содержимое страницы. Такой подход существенно ускоряет доступ HTTP-клиентов к контенту «далеких» и «медленных» HTTP-серверов. Пример: широко известны и пользуются популярностью «быстрые» кэширующие прокси-серверы, подключаемые к браузеру Opera в режиме «turbo».

2. Учет и ограничение трафика актуальны для любых протоколов (HTTP, FTP, POP3/SMTP и прочих). Proxy-сервер, располагаясь на периметре локальной сети (см. рис. 6.28,а), анализирует весь проходящий через него трафик; выполняет расчет его объемов; ограничение трафика, объем которого выходит за определенные квоты; блокирование трафика, направленного на определенные адреса (или, наоборот, приходящего с определенных адресов) и т.п. В этой роли он является частью «*биллинговой системы*», которая предназначена для учета и контроля телекоммуникационных услуг, предоставляемых Интернет-провайдерами.

3. Трансляция (подмена) адресов, которая может выполняться Proxy-сервером, приводит почти к такому же результату, что и применение технологии NAT (см. раздел. «Стандартные стеки протоколов»), а именно: запрос, направленный от клиента в направлении сервера, после прохождения через Proxy-сервер рассматривается сервером как поступивший не от клиента, а от этого Proxy-сервера (см. рис. 6.28). Существуют два типа Proxy-серверов:

- «анонимные», которые скрывают от сервера адрес источника;

- «неанонимные», которые в специальном поле заголовка HTTP-запроса указывают адрес подлинного источника.

«Неанонимные» локальные Proxy-сервера (например, AnalogX Proxy) широко используются для выхода в Интернет с разных узлов локальной сети через единственный глобальный адрес (см. рис. 6.28,а). Кэширующие и «биллинговые» Proxy-сервера так же, преимущественно, не «анонимны».

«Анонимные» Proxy-сервера, выстроенные в «цепочку», являются довольно надежным способом скрыть подлинный источник Интернет-сообщений, в том числе, от правоохранительных органов. Часто «цепочки» Proxy-серверов, встроенных во вредоносные программы, вытягиваются между узлами «ботнетов» (см. раздел. «Особенности устройства и поведения вредоносных программ»). Существуют также постоянно действующие сети Proxy-серверов, поддерживаемые энтузиастами Интернет-анонимности, например TOR. В РФ использование подобных средств ограничено на основании федерального закона №276-ФЗ от 29.07.1917.

6.10. Служба доменных имен (DNS)

Важную роль в Интернете играет DNS (англ. Domain Name System – Система доменных имен) – глобальная служба сопоставления числовым IP-адресам типа 100.200.111.222 символических имен типа «domain.net». DNS использует возможности «быстрого» протокола UDP.

Текстовая таблица соответствия символических имен числовым адресам располагается в специальных файлах операционной системы, например, в MS Windows и в UNIX это файл «hosts». При обращении к глобальному сетевому ресурсу по символическому имени в первую очередь проверяется содержимое этого файла и, если нужная запись найдена, то производится попытка соединения именно с указанным в ней адресом.

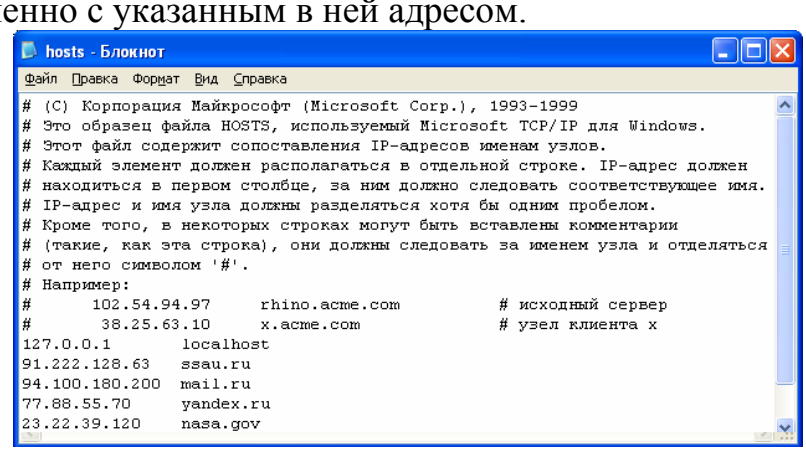


Рис. 6.29. Содержимое файла hosts в MS Windows

Вредоносные программы могут подменить строки этого файла. Поставьте на него защиту от записи (см. раздел «Разграничение доступа в MS Windows») и внимательно следите за его содержимым.

Если файл hosts пуст, то сетевые клиенты по «быстрому» протоколу UDP посылают короткий запрос к DNS-серверу провайдера, который хранит такую же таблицу, но с миллионами записей. В тех случаях, если искомое

символическое имя не найдено в его таблице, DNS-сервер провайдера обращается за справкой к одному из 13 «корневых» серверов, расположенных в США.

Непосредственная атака на содержимое базы данных DNS-серверов, по-видимому, невозможна. Однако в 2012 г. злоумышленниками на сайте компании Comodo Security была подменена информация[1] об адресах общедоступных DNS-серверов, в результате чего множество пользователей, воспользовавшись их услугами, регулярно попадали на фальшивые сайты[2].

Рис. 6.30. DNS-шторм

«*DNS-шторм*» – атака на алгоритм обмена между сетевым клиентом и DNS-сервером. «Фальшивый» сервер с высокой интенсивностью посылает на атакуемый узел сети сообщения, имитирующие ответы DNS-сервера. Поскольку протокол UDP не поддерживает аутентификацию сторон, то клиент не имеет возможности отличить правильный отклик, пришедший от DNS-сервера, от фальшивого. В результате, во время «DNS-шторма» высока вероятность перехода по неверному адресу (см. рис. 6.30).

Для защиты от DNS-шторма в протоколе обмена сообщениями между DNS-клиентом и DNS-сервером участвует уникальное число (TXID), которое сопровождает сообщения в рамках одной сессии связи. Также отклик отсылается на случайно выбранный (в заранее известном диапазоне) UDP-порт. Однако, случайным образом перебирая различные TXID и порты, «фальшивый» сервер сохраняет определенную вероятность обеспечить успех своей кибератаки. Более надежным способом противодействия подобным атакам является протокол DNSSEC, который требует подписания каждого сообщения при помощи ЭЦП. К сожалению, DNSSEC усложняет и замедляет процедуру обмена сообщениями между DNS-клиентом и DNS-сервером.

6.11. Защита от DDoS-атак

Отказ в обслуживании (англ. DoS – Denial of Service) – состояние неработоспособности узла компьютерной сети, которое вызвано чрезмерно большим количеством или сложностью внешних запросов, подлежащих обслуживанию. Это состояние может возникнуть естественным образом, но обычно оно является следствием злонамеренной кибернетической атаки.

DDoS-атака (от англ. distributed – распределенный) – DoS-атака, выполняемая из множества источников. Эффективность DDoS-атак очень велика.

---

[1] «Дейфейсинг», то есть искажение содержимого Интернет-страниц, будет подробней рассмотрен далее.

[2] «Фишинг», то есть использование фальшивых Интернет-ресурсов, будет подробней рассмотрен далее.

Рис. 6.31. Типичный отклик WWW-сервера, подвергнутого DOS-атаке.

В настоящее время DDoS-атаки чрезвычайно широко распространены. Мотивы могут быть самыми различными.

> ...Образовательный портал республики Татарстан больше года подвергался постоянным DDoS-атакам разных типов. Обнаруженная группа в «ВКонтакте» оказалась населена учащимися различных учебных заведений Татарстана. Описание группы гласит: «Каждый день в 14:00 ДДосим сайт еду татар»...
>
> (с) Пресс-Релиз "Лаборатории Касперского", сентябрь 2015 г.
>
> ...За несколько тысяч рублей атаку на чужой сайт может организовать практически любой желающий. Больше всех страдают сервисы, связанные с финансами... Больше всего атак за рекордный 2018 год пришлось на игровую индустрию (64%), электронную коммерцию (16%), финансовый сектор (7%). Рекордный рост показали атаки на телекоммуникационные компании (с 5% до 10%)... Считается, что DDoS позволяет на время заблокировать ресурсы конкурента или использоваться в качестве орудия шантажа... Часто автосалоны покупают атаки друг на друга в период смены летних покрышек на зимние, а цветочные магазины — перед 14 февраля и 8 марта... На днях... было объявлено об атаке на сайт выдачи ветеринарных сертификатов и... разрешений на ввоз, вывоз и транзит животных.
>
> Новостное сообщение с сайта "Фонтанка.Ру", март 2019 г.
>
> ...Что касается мотивов преступников, то специалисты отмечают значительную долю политизированных кампаний. Такие эпизоды, как атаки на Эквадор в поддержку Джулиана Ассанжа или сбои Telegram в дни беспорядков в Гонконге... Из коммерчески мотивированных..., когда злоумышленники вмешались в запуск дополнения к игре Rainbow Six Siege.
>
> (с) Пресс-Релиз "Лаборатории Касперского", август 2019 г.

Реализация кибератак может быть основана на различных принципах, вот некоторые из них.

1. *HTTP-флуд*. Этой простой атаке чаще всего подвергаются WWW-серверы, обслуживающие запросы в формате протоколов HTTP и HTPS. Цель – заблокировать деятельность какого-либо Интернет–сайта.

В ответ на очень короткий запрос типа

```
GET /HTTP/1.1
Host: server.net
```

WWW-сервер обязан прочитать с носителя (или программно сформировать) большую HTML-страницу и отослать ее запросившей программе-клиенту. Таким образом, «слабый» стимул должен вызвать «сильную» (и довольно ресурсоемкую!) реакцию. Эта диспропорция и является основным принципом, на котором основана DDoS-атака. Простейшая защита от «HTTP-флуда» заключается в двухступенчатой организации HTML-страниц атакуемого сервера, которые должны содержать примерно вот такое требование переадресации:

```
<script type="text/javascript">
  window.location = "server.ru/new_index.php"
</script>
```

«Легальный» браузер, в отличие от «нелегального» вредоносного клиента, выполнит скрипт и перейдет по рекомендованному адресу.

2. *SYN-флуд*. Этой атаке подвержены любые сетевые ресурсы, доступ к которым производится по протоколу TCP. Идея кибератаки базируется на искажении алгоритма установки соединения, который, в соответствии со стандартом, должен состоять из трех шагов.

Шаг 1. Клиент посылает серверу сообщение ACK.

Шаг 2. Сервер возвращает клиенту сообщение SYN+ACK.

Шаг 3. Клиент завершает алгоритм, посылая серверу сообщение SYN.

Атакующий клиент никогда не выполняет последний шаг алгоритма, а вместо этого повторно перезапускает процедуру установки соединения, многократно посылая ACK. Таким образом, существует реальная возможность переполнить очередь, в которой сервер сохраняет контексты незавершенных соединений, и вывести его (сервер) из строя.

3. Атаки с «амплификацией» (от. англ. amplification — усиление). Идея этих атак основана на подмене обратного адреса источника запросов. Например, единственный атакующий клиент может послать широковещательный запрос на все узлы какой-нибудь большой сети (то есть, на «общий» IP-адрес с тетрадами 255), а отклик будет возвращен на адрес «жертвы». Для атак этого типа часто используются протоколы ICMP и UDP.

4. Атаки через «ботнет». В современных условиях DDoS-атаки не требуют дополнительной «ампилификации», так как выполняются со стороны «ботнета», то есть управляемой злоумышленниками сети «зомбированных» компьютеров.

На момент написания этих строк типичная мощность DDoS-атаки составляет от 10 до 100 МБит/с, а наиболее мощные атаки могут достигать 1 ÷ 1.5 ТБит/с, что соответствует нескольким сотням тысяч одновременно

атакующих компьютеров. В 2016 г. была зафиксирована DDoS-атака со стороны 140 тыс. взломанных контроллеров в составе уличных, офисных и домашних видеокамер.

К сожалению, радикальных способов предотвращения DDoS-атак в настоящее время не существует. Вот основные принципы противодействия.

1. «Шунтирование» трафика, то есть временное переключение его на каналы связи с повышенной пропускной способностью и на сервера с увеличенным количеством вычислительных ресурсов. Такие услуги предлагают многие Интернет-провайдеры.

2. Фильтрация вредоносного трафика. Как правило, она производится при помощи специализированных программно-аппаратных комплексов, устанавливаемых на входных шлюзах сети, и позволяющих методами искусственного интеллекта отличать «нелегальные» внешние запросы от «легальных». Пример: Kaspersky DDoS Guard.

6.12. Защита Web

В настоящее время на физических ресурсах всевозможных локальных и глобальных сетей, образующих Интернет, развернута логическая сеть WWW (англ. World Wide Web – *Всемирная Паутина*), представляющая собой систему документов (изображений, текстов, аудио- и видео-объектов и т.п.), связанных между собой посредством *гиперссылок*. В основе механизма гиперссылок лежит *спецификация HTML* – языка гипертекстовой разметки, который ориентирован на описание текстовых документов (страниц), содержащих ссылки на иные документы (страницы, изображения, медиа-объекты и т.п.). Источником HTML-документов являются WEB-серверы, использующие в качестве интерфейса доставки HTTP-серверы. Для получения, «выполнения» и отображения HTML-документов служат WEB-клиенты (WEB-браузеры), которые почти всегда также являются HTTP-клиентами. Общая схема взаимодействия изображена на рис. 6.32.

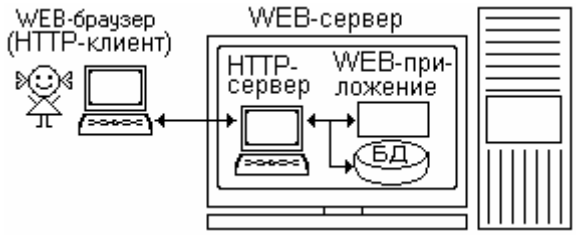


Рис. 6.32. Клиент-серверное взаимодействие в рамках WEB-технологии

Текст WWW-страницы может храниться на WWW-сервере в готовом виде и просто пересылаться клиенту. Также текст WWW-страницы может формироваться каждый раз заново средствами WEB-приложения – специальной прикладной программы, запущенной на WEB-сервере и обслуживающей работу какого-либо набора страниц (сайта). WEB-приложение может быть написано на любом языке программирования, главное, чтобы оно взаимодействовало с WEB-сервером по правилам CGI (англ. Common Gateway Interface – Общий шлюзовой интерфейс). Существуют специальные языки программирования,

специально оптимизированные на написание WEB-приложений с CGI-интерфейсом, например Perl или PHP.

Кроме информационного содержимого, WWW-страницы могут нести в себе программный код или ссылки на него.

Во-первых, это WEB-скрипты. Они пишутся на языках VBScript или JavaScript и размещаются в тексте HTML-страницы в виде исходного кода между тэгами <script> и </script>. Программный код WEB-скриптов транслируется и выполняется WEB-браузером на WEB-клиенте. Этот код не имеет непосредственного доступа к вычислительным ресурсам компьютера и предназначен, например, для ведения диалога с пользователем, отправки маленьких порций данных на WEB-сервер, размещения и считывания «куков» и прочего. «*Куки*» (англ. cookies – «вкусняшки») – небольшие порции данных, которые WEB-сервера (точнее, сайты) оставляют на компьютере. Обычно они содержат дату посещения, перечень посещенных страниц, использованные пароли и иные сведения, которые могут быть использованы сайтом при повторных посещениях. Часть этой информации имеет конфиденциальный характер, поэтому рекомендуется ее удалять из «куков» (см. рис. 6.33).

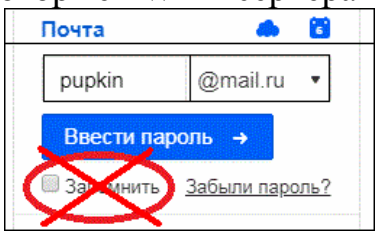


Рис. 6.33. Не сохранять «куки»

Во-вторых, это так называемые «апплеты». Так называется предварительно откомпилированный код, хранящийся где-то в Интернете, а HTML-страница содержит лишь ссылку на этот код. WWW-клиент загружает код и передает его на выполнение соответствующему «виртуальному процессору» (см. раздел «Виртуальные среды выполнения программ»), например:

- Java-апплеты выполняются JVM;
- Flash-апплеты выполняются при помощи Adobe Flash Player и т.п.

Известны многочисленные кибератаки, заключающиеся в подмене апплетов на эксплойты, использующие уязвимости в Adobe Flash Player (см. раздел «Уязвимости программного обеспечения»). С целью защиты от потенциально опасных апплетов практикуется доверие только коду, загруженному с того же WEB-сервера, что и сама страница, а так же подписанные правильной ЭЦП.

1. Атаки на WEB-сервер несут в себе очень большую опасность. Чаще всего злоумышленники несанкционированно получают административные права на сервере и вносят вредоносные изменения в содержимое WEB-страниц. Основные способы:

- использование внутренних уязвимостей программного обеспечения WEB-сервера;
- подбор пароля, принадлежащего администратору сервера;
- при помощи SQL-инъекции получение доступа к базам данных, в которых хранится пароль администратора, и определение этого пароля, например, при помощи «радужных таблиц».

Ранее, в соответствующих разделах уже были рассмотрены подробности выполнения всех трех типов атак и способы противодействия им.

2. Полная подмена содержимого WEB-страниц часто имеет целью так называемый «дейфейс» (от англ. deface – смена лица). Обычно это делается из хулиганских побуждений, однако возможны и иные намерения, такие как распространение рекламы, высказывание своих политических взглядов и т.п. Как правило, продолжительность существования подмененной страницы невелика.

3. Внесение скрытых изменений в содержимое WEB-страниц. Как правило, это действие выполняется с целью скрытного перенаправления пользователя, просматривающего контент, на совсем другую страницу. Например, тэг <iframe>, внедренный в HTML-текст текущей страницы, открывает в окне некоторую другую страницу, но не показывает ее.

```
<iframe src="http://..." height=0 width=0>
```

Такой же эффект достигается JavaScript-инструкцией:

```
document.write('<iframe src="http://..." height=0 width=0>');
```

При этом запускаются скрипты и апплеты «невидимой» страницы, и если они используют уязвимость, присутствующую в программном обеспечении WEB-клиента, то могут выполнить на компьютере свой вредоносный функционал. Кибератаки, использующие этот принцип, называются *XSS-атаками* (англ. X-Site Scripting – Передача скриптов между сайтами).

Вредоносный тэг <iframe> или инструкция «document.write» могли быть внедрены в HTML-страницу, хранящуюся на сервере, а могли быть сгенерированы «взломанным» WEB-приложением на сервере.

4. *Фишинг* (от. англ. fishing – рыбалка) – кибератака на основе подлога.

Сначала где-либо в Интернете (на странице с «похожим» именем, например, «mai1.ru» вместо «mail.ru») выполняется полная имитация внешнего вида и поведения какой-либо Интернет-страницы (например, WEB-клиента электронной почты или заглавной страницы банковского сайта).

Далее, тем или иным способом пользователь завлекается на эту страницу. Например, это может быть выполнено при помощи тэга <iframe> на ранее взломанной странице какого-либо сайта. Так же «приглашение» может прийти в обычном письме электронной почты.

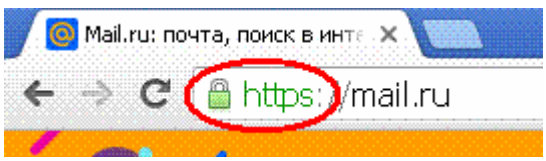


Рис. 6.34. Признак HTTPS в адресе подлинного сайта

Посетив «фальшивую» страницу, пользователь вводит в соответствующие поля свои секретные реквизиты (например, логин и пароль от своего почтового ящика или банковского счета) и, таким образом, сообщает их злоумышленнику. Надежных способов противодействия «фишингу» не существует. Однако пользователь сам может легко обнаружить подмену,

памятуя, что практически все «серьезные» сайты используют для доступа протокол HTTPS, и информация об этом присутствует в адресной строке браузера (см. рис. 6.34).

ТЕМА 7. Деятельность по обеспечению информационной безопасности

В первой главе было определены понятия «информация» и «качество информации».

Информационная безопасность (ИБ) – состояние систем обработки информации, при котором поддерживается требуемый уровень ее качества.

Обычно деятельность по поддержанию ИБ в организациях и на предприятиях реализуется при помощи специальной системы обеспечения информационной безопасности (СОИБ), то есть совокупности специальных органов, служб, средств, методов и мероприятий. Состав и структура СОИБ зависят от рода деятельности предприятия, ее масштабов и т.п. Например, на большом предприятии обеспечение ИБ могут осуществлять специальные службы, а в маленькой организации этим может заниматься единственный сотрудник. В любом случае, в деятельности по обеспечению ИБ и на больших, и на малых предприятиях обязательно должны принимать участие сами пользователи информационных систем.

Основными направлениями деятельности по обеспечению ИБ на предприятии (в организации) являются:

- организационное;
- техническое.

1. *Организационное направление*. Главными должностными лицами, ответственными за организацию и обеспечение ИБ, являются: 1) руководитель предприятия (организации); 2) заместитель руководителя по информационной безопасности. К организационным мерам, в первую очередь, относится разработка положений и регламентов, направленных на обеспечение ИБ. Основным руководящим документом, регламентирующим эту деятельность на любом предприятии (организации), является «Политика информационной безопасности», которая определяет:

- информационные ресурсы предприятия;
- лиц, ответственных за информационную безопасность на предприятии (как минимум, это заместитель руководителя по информационной безопасности);
- полномочия и ответственность отделов и служб в отношении безопасности;
- правила разграничения доступа сотрудников к информационным ресурсам;
- описание пропускного режима для сотрудников и посетителей;

- перечень и правила использования программно-технических средств защиты;
- иные требования общего характера.

Разумеется, к организационным мерам относится также деятельность по контролю за соблюдением сотрудниками требований «Политики безопасности» и иных документов.

2. *Техническое направление*. Это направление обеспечения ИБ подразумевает создание и грамотную эксплуатацию комплекса программно-аппаратных средств, входящих в СОИБ. Программно-аппаратное обеспечение типичной СОИБ содержит:

- подсистему управления доступом;
- подсистему регистрации и учета событий информационной безопасности;
- подсистему обеспечения целостности данных;
- подсистему антивирусной защиты;
- подсистему обеспечения безопасности межсетевого взаимодействия;
- подсистему обнаружения вторжений и утечек (англ. IDS – Intrusion Detection System);
- подсистему предотвращения вторжений и утечек (англ. DLP – Data Leak Prevention).

Состав конкретных средств и принципы их функционирования подробней рассмотрены ранее на страницах настоящего пособия. При этом ряд организаций, к которым относятся:

- государственные организации;
- негосударственные организации, работающие со служебной информацией государственных органов;
- организации, работающие с персональными данными –

обязаны использовать в качестве средств защиты в составе СОИБ только средства, сертифицированные ФСЭК и ФСБ (средства криптографической защиты). Понятия «сертификация» и «персональные данные» рассмотрены далее.

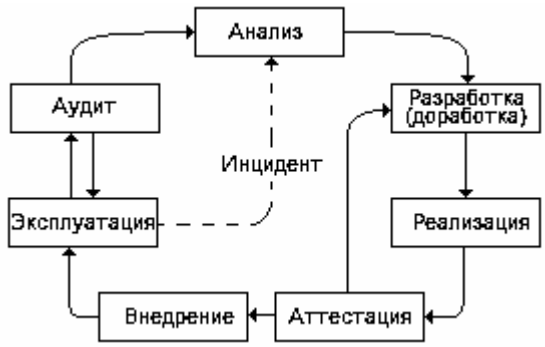


Рис. 7.1. Жизненный цикл СОИБ

3. *Жизненный цикл СОИБ*. Обеспечение информационной безопасности средствами СОИБ – циклический процесс, который включает следующие этапы (см. рис. 7.1):

- анализ текущей ситуации ИБ;

- разработку или доработку СОИБ;
- реализацию СОИБ;
- аттестацию СОИБ;
- внедрение СОИБ;
- эксплуатацию СОИБ;
- плановый (или экстренный) аудит СОИБ.

Основным является этап эксплуатации СОИБ. На Рис. 7.1. пунктиром показана «экстренная» ситуация, требующая внеплановой доработки действующей СОИБ в результате возникновения так называемых «инцидентов безопасности».

Аттестация СОИБ – подтверждение соответствия реализованной системы требованиям государственных стандартов или иных нормативных документов, регламентирующих вопросы защиты информации. Может проводиться самой организацией, выполняющей реализацию СОИБ. Результатом процесса аттестации служит специальный документ – «аттестат соответствия».

Сертификация средств защиты информации – подтверждение (обычно, по итогам испытаний) соответствия средств, являющихся составными частями СОИБ, требованиям государственных стандартов или иных нормативных документов, регламентирующих вопросы защиты информации. Проводится организацией или предприятием, не являющимися производителями сертифицируемых средств, и имеющим соответствующую лицензию и средства проведения испытаний. Результатом процесса сертификации служит специальный документ – «сертификат соответствия».

Широко известен и используется «Государственный Реестр сертифицированных средств защиты информации» (операционных систем, антивирусов, программно-аппаратных комплексов и т.п.), формируемый и поддерживаемый ФСТЭК РФ. Например, на момент написания этих строк в реестре содержатся:

- антивирус Касперского 8.0;
- антивирус Dr.Web Enterprise Security Suite;
- антивирус «ESET NOD32 Secure Enterprise Pack»;
- операционная система «Astra Linux Special Edition»;
- операционная система«SUSE Linux EnterpriseServer»;
- операционная система «Альт Линукс СПТ 6.0»
- операционная система Windows NT 4.0 Server;
- операционная система MS Windows NT 4.0 Workstation (Russian) с пакетом обновления Service Pack 5 (Russian);
- операционная система MS Windows Server 2008 Standard Edition;
- операционная система Microsoft Windows 7;
- операционная система Microsoft Windows Server Standard 2012;
- операционная система Windows 8.1
- операционная система Microsoft Windows Server 2016.

Прочие популярные антивирусы и операционные системы в перечне ФСТЭК на текущий момент отсутствуют.

Также деятельность по сертификации средств защиты информации (прежде всего, обладающих криптографическими возможностями) выполняет ФСБ РФ.

Аудит СОИБ – анализ текущего состояния СОИБ на предмет обеспечения ей требуемого уровня информационной безопасности. Может выполняться как организацией, эксплуатирующей СОИБ (внутренний аудит), так и иной организацией (внешний аудит).

7.1. Классификация информации ограниченного доступа

С точки зрения доступности информацию можно разделить на «общедоступную информацию» и «информацию ограниченного доступа» (см. рис. 7.2).

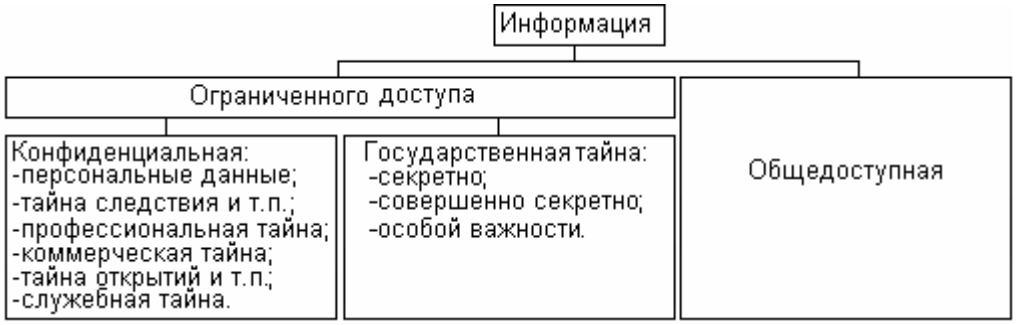


Рис. 7.2. Виды информации

Государственная тайна – сведения в области военной, внешнеполитической, экономической, научно-исследовательской, разведывательной, контрразведывательной и оперативно-розыскной деятельности, разглашение которых может нанести ущерб государству.

Персональные данные – любая информация, прямо или косвенно относящаяся к физическому лицу, в том числе его имя, фамилия, отчество[1], дата и место рождения, адрес, семейное, социальное, имущественное положение, образование, профессия, доходы, состояние здоровья и т.п.

Также к информации ограниченного характера относятся[2]:

- *тайна следствия*, *тайна судопроизводства*, *тайна исполнения наказаний и принудительного лечения*;
- *профессиональная тайна* (например, врачебная, адвокатская и т.п.);
- *коммерческая тайна* – производственные, технические и иные сведения, которые имеют коммерческую ценность в силу своей закрытости;

[1] Иногда фамилию, имя и отчество считают «личной» информацией, не подлежащей ограничениям.

[2] На момент написания этих строк дискутируется вопрос о введении понятия «Военная тайна».

- сведения об изобретениях или открытиях до момента их официальной публикации;
- *служебная тайна* – сведения, которые становятся известными различным должностным лицам по роду их служебной деятельности, однако в силу своего особого характера не могут свободно распространяться.

7.2. Защита государственной тайны

Ранее было дано определение понятию «государственная тайна» (ГТ).

Требования по защите этой информации и контроль за их соблюдением регламентируются Законом РФ «О государственной тайне», в котором установлен перечень сведений, составляющих государственную тайну. Однако предусмотрен и запрет на необоснованное засекречивание ряда сведений, таких, например, как информация о катастрофах и неблагоприятной экологической обстановке; информация о состоянии здоровья руководителей государства; информация о фактах нарушения законности органами государственной власти и их должностными лицами и т.п.

Государственные органы, реализующие и контролирующие защиту ГТ в РФ[1]:

- межведомственная комиссия по защите государственной тайны;
- федеральный орган, уполномоченный в области обеспечения безопасности;
- федеральный орган, уполномоченный в области обороны;
- федеральный орган, уполномоченный в области внешней разведки;
- федеральный орган (и его территориальные подразделения), уполномоченный в области противодействия техническим разведкам и технической защиты информации;
- органы государственной власти, предприятия, учреждения и организации и их структурные подразделения по защите государственной тайны.

Определены следующие уровни конфиденциальности сведений, относящихся к ГТ, каждому из которых соответствует определенный «гриф секретности»:

- «*секретно*» (для доступа к ней требуется наличие «3 формы допуска»);
- «*совершенно секретно*» (требуется «2 форма допуска»);
- «*особой важности*» (требуется «1 форма допуска»).

На носитель секретной информации и сопроводительную документацию должны быть нанесены следующие реквизиты:

- степень («гриф») секретности со ссылкой на перечень сведений, подлежащих засекречиванию;
- сведения об органе государственной власти, предприятии, учреждении или организации, осуществивших засекречивание;
- регистрационный номер;
- дата или условия рассекречивания сведений.

[1] Конкретные наименования с течением времени подвергаются изменениям.

Методы и средства защиты ГТ регламентируются комплексом законов, постановлений, стандартов, методик и иных нормативных документов, значительная часть которых сама относится к информации ограниченного доступа. Общие требования нормативных документов заключаются в том, что организации и учреждения, работающие с ГТ, обязаны:

- иметь и использовать только аттестованные[1] средства для обработки ГТ;
- иметь и использовать только аттестованные помещения и оборудование;
- иметь и привлекать для работы с ГТ только специально подготовленных сотрудников.

Работа со сведениями ГТ должна осуществляться в рамках специального «режима секретности», который включает:

- особый порядок доступа конкретных сотрудников к конкретной информации и в конкретные помещения;
- особые правила делопроизводства с использованием ГТ;
- пропускной режим на объект (и в отдельные зоны внутри объекта), где производятся работы с ГТ.

В УК РФ предусмотрена ответственность за преступления в сфере оборота сведений ГТ:

- ст. 275 – государственная измена;
- ст. 276 – шпионаж;
- ст. 283 – разглашение государственной тайны;
- ст. 284 – утрата документов, содержащих государственную тайну.

7.3. Защита персональных данных

Ранее было дано определение понятию «персональные данные» (ПДн).

Задача обработки ПДн встречаются практически на любом предприятии или в любой организации, вне зависимости от формы их собственности и масштаба.

Оператор персональных данных – государственный или муниципальный орган, юридическое или физическое лицо, организующие и (или) осуществляющие обработку ПДн.

Примеры:

- образовательное учреждение хранит и использует данные о сотрудниках и учениках (воспитанниках, студентах);
- лечебное учреждение хранит и использует данные о пациентах, также передает их страховой компании;
- банк или страховая компания хранит и обрабатывает сведения о своих клиентах и корреспондентах;
- магазин (центр сервисного обслуживания, ателье, парикмахерская и т.п.) хранит и использует данные о покупателях (клиентах), собираемые и аккумулируемые при помощи Интернет-сайта.

[1] Определение термина «аттестация» см. выше.

Требования по защите ПДн и контроль за их соблюдением регламентируются Законом РФ № 152 – ФЗ «О персональных данных». Методы и средства защиты ПДн регламентируются комплексом законов, постановлений, стандартов, методик и иных нормативных документов.

Регуляторы (то есть уполномоченные государственные органы) в области защиты ПДн:

- Роскомнадзор (Федеральная служба по надзору в сфере связи и массовых коммуникаций);
- ФСТЭК РФ;
- ФСБ РФ.

Общие требования к обработке ПДн, установленные №152 – ФЗ:

- оператор персональных данных определяет цели их обработки в соответствии со своими полномочиями;
- объем и характер обрабатываемых персональных данных должен соответствовать целям их обработки;
- недопустимо объединять созданные для разных целей персональные данные (например, в одну базу данных);
- персональные данные подлежат уничтожению по достижении целей (утраты необходимости в) их обработки;
- обработка персональных данных может осуществляться оператором только с письменного согласия субъектов ПДн.

ПДн могут храниться и обрабатываться не только в «автоматизированном», но и в «ручном» режиме (например, при помощи ведения записей в специальных журналах). Общие требования к защите ПДн распространяются на оба способа.

Ответственность за нарушения требований по защите ПДн:

- административная – штраф;
- дисциплинарная – увольнение провинившегося работника;
- уголовная (в соответствии со ст. ст. 137, 140, 272 УК РФ) – исправительные работы, лишение свободы.

Для защиты ПДн, обрабатываемых в *информационных системах персональных данных* (ИСПДн), применяются специализированные СОИБ, состав и принципы работы которых регламентированы законодательством РФ.

7.4. Защита коммерческой тайны

Ранее было дано определение понятию «коммерческая тайна» (КТ).

Основными нормативными документами, регулирующими отношения в области защиты КТ, являются:

- закон № 98-ФЗ «О коммерческой тайне»;
- Трудовой кодекс РФ, который регулирует вопросы обработки информации, составляющей коммерческую тайну, в рамках трудовых отношений;
- Гражданский кодекс РФ, регулирующий отношения по использованию прав на секрет производства.

Регулирование режимов защиты КТ со стороны государства отсутствует. Организация (или лицо), нуждающееся в организации защиты своей КТ, самостоятельно определяет:

- перечень информации, составляющей КТ;
- порядок обращения с КТ и контроль над соблюдением такого порядка;
- правила учета лиц и организаций, которые имеют доступ к информации, составляющей КТ;
- регулирование отношений по использованию КТ работниками на основании трудовых договоров;
- правила установки ограничительных пометок на материальные носители (документы), содержащие конфиденциальную информацию.

Техническая защита сведений КТ может выполняться с использованием любых методов и средств, обеспечивающих необходимые требования по безопасности информации. Однако на такие СОИБ распространяются ограничения ФСБ РФ, связанные с использованием СКЗИ (средств криптографической защиты информации). Также следует иметь в виду, что в организации и на предприятии, хранящем и обрабатывающем КТ, обязательно выполняется обработка других видов конфиденциальной информации (например, ПДн), так что СОИБ, обеспечивающие защиту разных классов конфиденциальной информации, не должны вступать в конфликт.

7.5. Защита банковской тайны

Банковская тайна (БТ) – конфиденциальные сведения:

- об операциях, счетах и вкладах клиентов и корреспондентов;
- о финансовом положении клиентов и корреспондентов;
- о структуре организаций – клиентов, включая информацию о руководителях, видах деятельности и т.п.;
- об отчетности по результатам деятельности банка (за исключением подлежащих опубликованию);
- о технических реквизитах защиты информации (например, о пин-кодах банковских карт);
- об устройстве системы охраны банка, клиентов и банковской тайны.

За незаконное разглашение БТ предусмотрены гражданская, административная и уголовная ответственность (ст. 183 УК РФ). Однако возможно раскрытие конфиденциальных сведений по запросу бюро кредитных историй, судов, следственных, налоговых и таможенных органов, Счетной палаты, АСВ, ПФР, Фонда социального страхования, ФССП, ФСФН, ФТС. В этом случае информация приобретает характер «служебной тайны».

Основными нормативными документами по защите БТ, являются:

- Положение Банка России №382-П от 09.06.2012 «О требованиях к обеспечению защиты информации при осуществлении переводов денежных средств и о порядке осуществления Банком России контроля за соблюдением требований к обеспечению защиты информации при осуществлении переводов денежных средств»;

- Положение Банка России №552-П от 24.08.2016 «О требованиях к защите информации в платежной системе Банка России»;
- Стандарт Банка России СТО БР ИББС-1.0-2014 «Обеспечение информационной безопасности организаций банковской системы Российской Федерации. Общие положения»;
- ГОСТ Р 57580.1-2017 «Безопасность финансовых (банковских) операций. Защита информации финансовых организаций. Базовый набор организационных и технических мер».

На момент написания этих строк первые два Положения носят обязательный характер, а стандарты СТО БР ИББС-1.0-2014 и ГОСТ Р 57580.1-2017 – рекомендательный. Планируется, что в ближайшие годы появится законодательное требование об обязательном соблюдении банками требований ГОСТ Р 57580.1-2017. Этот ГОСТ содержит требования к организации всех основных процессов информационной защиты, включая противодействие вредоносному коду, утечкам информации и нарушению целостности информационной инфраструктуры. В нем содержатся требования к защите информации при осуществлении удаленного доступа с использованием мобильных устройств. Также предлагается комплексный подход к планированию, реализации, контролю и совершенствованию процесса защиты информации в финансовых организациях. Кроме того, в ГОСТ-е приведены требования к защите информации на всех этапах жизненного цикла автоматизированных систем и приложений, используемых финансовыми компаниями и банками.

7.6. Уголовная ответственность за компьютерные преступления

К сожалению, нынешнее состояние нашего общества таково, что важную роль в поддержании ИБ играют не только меры административного или программно-технического, но и уголовно-правового характера. Статьи УК Российской Федерации, определяющие наказание за компьютерные преступления, введены в 1997 г. и несколько раз подвергались исправлениям и дополнениям (как правило, в сторону расширения квалификационных признаков и усиления наказания). В УК РФ (редакция от 16.10.2019) присутствуют следующие статьи.

Статья 272 «Неправомерный доступ к компьютерной информации».

Статья 273 «Создание, использование и распространение вредоносных программ».

Статья 274 «Нарушение правил эксплуатации средств хранения, обработки или передачи компьютерной информации и информационно-телекоммуникационных сетей».

Статья 146 «Нарушение авторских и смежных прав» предусматривает наказание за незаконное использование объектов авторского права, а также за

приобретение, хранение, перевозку контрафактных экземпляров произведений или фонограмм в целях сбыта.

С 2012 в УК присутствуют также статьи 159.3 «Мошенничество с использованием электронных средств платежа» и 159.6 «Мошенничество в сфере компьютерной информации».

Отдельное место занимает Статья 274.1 УК РФ «Неправомерное воздействие на критическую информационную инфраструктуру Российской Федерации», которая определяет наказание за три вида компьютерных преступлений:

- неправомерный доступ к компьютерной информации;
- создание, использование и распространение вредоносных программ;
- нарушение правил эксплуатации информационных систем, -

если были затронуты критически важные информационные ресурсы РФ (например, средства правительственной связи; средства связи и управления Вооруженных сил и прочее).

Все эти статьи (кроме 274.1) предусматривают в качестве минимального наказания штрафы размером в несколько сотен тысяч рублей и (или) исправительные работы, но может последовать и лишение свободы сроком на 2÷4 года. Сроки лишения свободы увеличиваются до 5 лет в случае совершения преступления в совокупности с отягчающими обстоятельствами, которыми считаются совершение преступления группой лиц по предварительному сговору либо группой/лицом с использованием своего служебного положения. Также отягчающим обстоятельством считается наличие корыстных побуждений. Наконец, при наступлении тяжелых последствий либо при причинении особо крупного ущерба применяются максимальные меры наказания, включающие лишение свободы на срок до 7 лет. Статья 274.1 предусматривает более жесткие наказания. Так, например, при наступлении тяжких последствий сроки лишения свободы могут достигать 10 лет.

ЛИТЕРАТУРА

Ко всем темам:

1. Шаньгин, В. Ф. Информационная безопасность компьютерных систем и сетей: учебное пособие. – М.: Форум: Инфра-М, 2017. – 415 с.

К Темам 1 и 7:

2. Грибунин В.Г., Чудовский В.В. Комплексная система защиты информации на предприятии. – М.: Издательский центр «Академия», 2009. – 416 с.

3. Жук А.П., Жук Е.П., Лепешкин О.М., Тимошкин А.И. Защита информации: учебное пособие. – М.: Инфра-М, 2015. – 392 с.

4. Зайцев А.П., Голубятников И.В., Мещеряков Р.В., Шелупанов А.А. Программно-аппаратные средства обеспечения информационной безопасности: учебное пособие. – М.: Машиностроение, 2006. − 260 с.

5. Моисеев А.И., Жмуров Д.Б. Информационная безопасность распределенных информационных систем. – Самара: Изд-во СГАУ, 2013.

К Теме 2:

6. Девянин П.Н., Михальский О.О., Правиков Д.И., Щербаков А.Ю. Теоретические основы компьютерной безопасности: учебное пособие для вузов. – М.: Радио и связь, 2000. – 192 с.

7. Понятов А.А. Теория информации и кодирования. – М.: МИИТ, 2010. – 188 с.

К Теме 3:

8. Бернет С., Пейн С. Криптография. Официальное руководство RSA Security. – М.: Бином, 2007. – 381 с.

9. Баричев С.Г., Гончаров В.В., Серов Р.Е. Основы современной криптографии: учебный курс. – М.: Горячая Линия – Телеком, 2002. – 175 с.

10. Введение в криптографию / Под ред. В.В. Ященко. – СПб.: Питер, 2001. – 288 с.

11. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. – М.: Триумф, 2002. – 816 с.

К Теме 4:

12. Афанасьев А.А., Веденьев Л.Т., Воронцов А.А. Аутентификация. Теория и практика обеспечения безопасного доступа к информационным ресурсам. Учебное пособие для вузов. – М.: Горячая линия–Телеком, 2009. – 552 с.

К Теме 5:

13. Проскурин В.Г. Защита в операционных системах. – М.: Горячая Линия – Телеком, 2014. – 193 с.

14. Климентьев К.Е. Компьютерные вирусы и антивирусы. – М.: ДМК-Пресс, 2013. – 656 с.

К Теме 6:

15. Столингс В. Криптография и защита сетей: принципы и практика. – М.: Вильямс, 2001. – 669 с.

16. Столлингс В. Основы защиты сетей: приложения и стандарты. – М.: Вильямс, 2002. – 429 с.

17. Мамаев М., Петренко С. Технологии защиты информации в Интернете. Специальный справочник. – СПб: Питер, 2002. – 848 с.

18. Голдовский И. Безопасность платежей в Интернете. – СПб.: Питер, 2001. – 240 с.

ПРИЛОЖЕНИЕ. Вопросы и задания

Вопросы и задания к Теме 1 и Теме 7

1.1. Охарактеризуйте основные определения «информации», чем они отличаются?

1.2. Перечислите основные операции, осуществляемые над информацией. Приведите собственные примеры.

1.3. Перечислите основные свойства, от которых зависит качество информации. Охарактеризуйте суть эти свойств.

1.4. Перечислите основные классы угроз качеству информации. Приведите собственные примеры.

1.5. Встречались ли вы с конкретной информационной угрозой, к какому классу она относилась?

1.6. Перечислите службы, которые могут организовывать и осуществлять защиту информации на конкретном предприятии? Какие службы решают подобные задачи на предприятии (организации, учебном заведении), в котором вы (ваши знакомые, родственники) работаете (учитесь)?

1.7. Какой «гриф» ставится в РФ на сведения самой высокой степени секретности?

1.8. Приведите пример предприятия (организации) и персональных данных, которые на этом предприятии (организации) обрабатываются.

1.9. Поясните, чем отличаются друг от друга понятия «аттестация», «сертификация» и «аудит» средств защиты информации.

Вопросы и задания к Теме 2

2.1. К какой формальной модели относится политика предприятия, описывающая правила прохода внутрь на основании пропуска или удостоверения?

2.2. К какой формальной модели относится политика предприятия, описывающая правила выдачи ключей тем или иным сотрудникам на основании должности? На основании ФИО?

2.3. Охарактеризуйте возможности работы в UNIX с программой, файл которой имеет атрибуты доступа «--x--x--x».

2.4. Приведите пример практической реализации политики изоляционизма.

2.5. Какие технологии защиты информации использовал главный герой повести Куприна «Поединок» для написания секретного письма девушке?

2.6. Чему равен бит четности для набора данных 10101?

2.7. Каково главное отличие «контрольных сумм» от «криптографических» хешей?

2.8. Как вы думаете, почему в качестве порождающих при расчете CRC используют неприводимые полиномы?

2.9. Пусть компьютер способен рассчитывать миллиард криптографических хешей в секунду. Сколько времени (в наихудшем случае) потребуется, чтобы подобрать прообраз для MD5? Сколько таких компьютеров потребуется, чтобы подбор занял год?

2.10. Воспользуйтесь стандартными возможностями операционной системы Windows, чтобы определить, чему равно значение хеш-функции MD5 от строки «1234567890»? Хеш-функции SHA-1 от той же строки?

Вопросы и задания к Теме 3

3.1. Какова длина в битах ключа для шифра «поворотная решетка Кардано», основанного на квадрате 8x8 клеток?

3.2. Зашифруйте методом Цезаря слово «БАГАЖ» с ключом «М» .

3.3. Зашифруйте методом Цезаря слово «WHEEL» с ключом «H».

3.4. Дешифруйте слово «СУХИЕ», зашифрованное методом Цезаря с неизвестным ключом.

3.5. Дешифруйте слово «FROG», зашифрованное методом Цезаря с неизвестным ключом.

3.6. Зашифруйте методом Вижинера слово «РЫЦАРЬ» с ключом «БУЛАВА».

3.7. Зашифруйте методом Вижинера слово «WOMAN» с ключом «LABEL».

3.8. Дешифруйте фразу «KQZLPYK26YKG EUOIPYK 7ZLPF 7 BKYA2, RAFG 3FA 6KQAZ YLPQP2ANZ22AZ AO4Z6F7A 7 QPLZ», зашифрованную методом простой замены.

3.9. Пусть компьютер способен перебирать миллиард шифровальных ключей в секунду. Сколько времени (в наихудшем случае) ему потребуется, чтобы взломать полным перебором шифр DES? Шифр 3DES? Сколько таких компьютеров потребуется, чтобы взломать 3DES в течение года?

3.10. В 8-битовом сдвиговом регистре с линейной обратной связью (LFSR) хранится число 5Ah. Обратной связью объединены 0-й и 7-ой биты. Какое значение будет храниться в регистре на следующем шаге?

3.11. Каков мультипликативно-обратный элемент для 3 в GF(5)?

3.12. Открытый параметр «n» для шифра RSA равен 51. Чему равен секретный модуль «m»?

3.13. При шифровании методом RSA p=3, q=11. Чему равны «n» и «m»?

3.14. При шифровании методом RSA p=3, q=7. Чем эти числа плохи?

3.15. В условиях задания 3.11 выбран ключ K_E=7, чему равен ключ K_D?

3.16. В условиях задания 3.11 и 3.12 некоторое сообщение зашифровано методом RSA, результат равен 29. Расшифруйте это сообщение.

3.17. Какова длина ключа асимметричного шифра RSA, эквивалентного по криптостойкости ключу симметричного шифра длиной 96 битов?

Вопросы и задания к Теме 4

4.1. В пароле разрешено использовать только 26 латинских букв и 10 цифр. За сколько попыток (в худшем случае) злоумышленник подберет пароль длиной 3 буквы?

4.2. Придумайте «хороший», но легко запоминаемый пароль.

4.3. Пусть на подбор одного варианта требуется секунда. Сколько времени (в худшем случае) злоумышленнику потребуется для подбора 4-значиного PIN-кода?

4.4. Сколько раз нужно повторить вход-выход из «пещеры Али-Бабы», чтобы доказать владение секретом с достоверностью не менее 0.99?

4.5. Сколько факторов аутентификации используется при проходе в метро? А сколько их использовано в повести Рыбакова «Кортик» для ограничения доступа к секретной карте?

Вопросы и задания к Теме 5

5.1. Проверьте, как реагируют антивирусы на текстовый файл со строкой «X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*».

5.2. Вирус-червь делает одну с вою копию в течение секунды. Сколько экземпляров червя появится через одну секунду? Через две? Сколько времени потребуется, чтобы заразить весь Интернет?

5.3. Перечислите факторы, которые благоприятствуют появлению программной уязвимости типа «переполнение буфера»?

5.4. Пользуясь стандартными средствами операционной системы Windows, определите хотя бы несколько программ, которые запускаются автоматически при включении вашего компьютера.

Вопросы и задания к Теме 6

6.1. Пользуясь стандартными средствами операционной системы и ресурсами Интернета, определите, каков текущий IP-адрес вашего компьютера.

6.2. Пользуясь стандартными средствами операционной системы и ресурсами Интернета, определите, каков IP-адрес домена «ssau.ru»? Домена «mail.ru»? На кого они зарегистрированы?

6.3. Пользуясь стандартными средствами операционной системы и ресурсами Интернета, определите, какое количество маршрутизаторов встречается на пути от Вашего узла Интернета к узлу «nasa.gov».

6.4. На некотором узле сети открыты сетевые порты 25 и 110. Что можно сказать о роли этого узла?

Редактор Ю.Н. Литвинова
Компьютерная вёрстка Ю.Н. Литвиновой

Подписано для тиражирования _____.
Объем издания _____ Кб.
Количество носителей _____ экз.

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С. П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)
443086, Самара, Московское шоссе, 34.

Изд-во Самарского университета.
443086 Самара, Московское шоссе, 34